# S32K1xx系列参考手册

支持S32K116、S32K118、S32K142、S32K144、S32K146和S32K148

文件编号。S32K1XXRM
Rev. 4, 06/2017

# 内容

科目编号 标题 页码

## 第1章 关于本手册


1.1 听众。 47

1.2 组织。 47

1.3 模块描述。 47

1.3.1 例如：澄清同一章内容的特定芯片信息。 48

1.3.2 例如：芯片的特定信息，指的是不同的章节。 49

1.4 登记册描述。 50

1.5 公约。 51

1.5.1 注意事项、注意事项和警告。 51

1.5.2 编码系统。 51

1.5.3 排版记号。 52

1.5.4 特别条款。 52


## 第二章 简介


2.1 概述。 55

2.2 S32K1xx系列介绍。 55

2.2.1 S32K14x. 55

2.2.2 S32K11x 56

2.3 特点总结。 57

2.4 框图。 60

2.5 特征比较。 62

2.6 应用。 63

2.7 模块功能类别。 64

2.7.1 ARM Cortex-M4F核心模块。 66

2.7.2 ARM Cortex-M0+核心模块。 66

2.7.3 系统模块。 ..... 67
2.7.4 存储器和存储器接口。 ..... 68
2.7.5 电源管理。 ..... 68
2.7.6 计时。 ..... 69
2.7.7 模拟模块。 ..... 69
2.7.8 计时器模块。 ..... 70
2.7.9 通信接口。 ..... 70
2.7.10 调试模块。 ..... 71

## 第3章 内存图

3.1 介绍。 ..... 73
3.2 SRAM内存图。 ..... 73
3.2.1 S32K14x。SRAM内存图 ..... 73
3.2.2 S32K11x。SRAM内存图 ..... 73
3.3 闪存地图。 ..... 74
3.4 外围桥（AIPS-Lite）内存图。 ..... 74
3.4.1 读后写的顺序和所需的内存操作的序列化。 ..... 75
3.5 私人外设总线（PPB）内存图。 ..... 76
3.6 用于CM4核心的异化位带区域。 ..... 77

## 第四章
## 信号复用和引脚分配

4.1 介绍。 ..... 79
4.2 功能描述。 ..... 79
4.3 垫子描述。 ..... 80
4.4 默认的垫子状态。 ..... 81
4.5 信号复用表。 ..... 82
4.5.1 IO信号表 ..... 82
4.5.2 输入多路复用表。 ..... 83

4.6 引脚图。 ..... 84

## 第5章 安全概述

5.1 介绍。 ..... 85

5.2 设备安全。 ..... 85

5.2.1 闪存的安全性。 ..... 85

5.2.2 加密服务引擎（CSEc）的安全功能。 ..... 86

5.2.3 设备启动模式。 ..... 87

5.3 安全用例的例子。 ..... 87

5.3.1 安全启动：检查引导程序的完整性和真实性。 ..... 87

5.3.2 信任链：检查闪存的完整性和真实性。 ..... 88

5.3.3 安全通信。 ..... 89

5.3.4 组件保护。 ..... 90

5.3.5 消息-认证的例子。 ..... 91

5.4 故障分析前需要的步骤。 ..... 92

5.5 安全编程流程示例（安全启动）。 ..... 93

## 第6章 安全概述

6.1 介绍。 ..... 95

6.2 S32K1xx的安全概念。 ..... 96

6.2.1 Cortex-M4/M0+结构核心自我测试（SCST）。 ..... 97

6.2.2 内存和闪存上的ECC。 ..... 98

6.2.3 电源监控。 ..... 98

6.2.4 时钟监测。 ..... 99

6.2.5 时间上的保护。 ..... 99

6.2.6 操作干扰保护。 ..... 99

6.2.7 CRC. ..... 101

6.2.8 系统资源的多样性。 ..... 101

## 第7章 核心概述

7.1 ARM Cortex-M4F内核配置。 ..... 103
7.1.1 总线、互连和接口。 ..... 104
7.1.2 系统滴答计时器。 ..... 104
7.1.3 调试设施。 ..... 104
7.1.4 缓存。 ..... 105
7.1.5 核心权限级别。 ..... 105
7.2 嵌套的矢量中断控制器（NVIC）配置。 ..... 106
7.2.1 中断优先级水平。 ..... 106
7.2.2 无任务的中断。 ..... 106
7.2.3 确定配置一个特定中断的位域和寄存器位置。 ..... 107
7.3 异步唤醒中断控制器（AWIC）配置。 ..... 108
7.3.1 唤醒来源。 ..... 108
7.4 FPU配置。 ..... 109
7.5 JTAG控制器配置。 ..... 110

## 第8章 杂项控制模块（MCM）

8.1 芯片特定的MCM信息。 ..... 111
8.2 介绍。 ..... 111
8.2.1 特点是。 ..... 111
8.3 内存图/寄存器描述。 ..... 112
8.3.1 交叉条开关（AXBS）从属配置（MCM_PLASC）。 ..... 113
8.3.2 交叉条交换机（AXBS）主配置（MCM_PLAMC）。 ..... 113
8.3.3 核心平台控制寄存器（MCM_CPCR）。 ..... 115
8.3.4 中断状态和控制寄存器（MCM_ISCR）。 ..... 118
8.3.5 进程ID寄存器（MCM_PID）。 ..... 121
8.3.6 计算操作控制寄存器（MCM_CPO）。 ..... 122
8.3.7 本地内存描述符寄存器（MCM_LMDRn）。 ..... 123

8.3.8 本地内存描述符寄存器2（MCM_LMDR2）。 ..... 126
8.3.9 LMEM奇偶校验和ECC控制寄存器（MCM_LMPECR）。 ..... 130
8.3.10 LMEM奇偶校验和ECC中断寄存器（MCM_LMPEIR）。 ..... 130
8.3.11 LMEM故障地址寄存器（MCM_LMFAR）。 ..... 132
8.3.12 LMEM故障属性寄存器（MCM_LMFATR）。 ..... 133
8.3.13 LMEM故障数据高位寄存器（MCM_LMFDHR）。 ..... 134
8.3.14 LMEM故障数据低寄存器（MCM_LMFDLR）。 ..... 134
8.4 功能描述。 ..... 135
8.4.1 中断。 ..... 135

## 第九章
## 系统集成模块（SIM）。

9.1 芯片特定的SIM卡信息。 ..... 137
9.1.1 SIM寄存器位域的实现。 ..... 137
9.2 介绍。 ..... 137
9.2.1 特点是。 ..... 137
9.3 内存图和寄存器定义。 ..... 138
9.3.1 SIM寄存器的描述。 ..... 138

## 第十章
## 端口控制和中断（PORT）。

10.1 芯片特定的端口信息。 ..... 165
10.1.1 PCR的数量。 ..... 165
10.1.2 I/O配置顺序 ..... 166
10.1.3 数字输入滤波器配置顺序 ..... 166
10.2 介绍。 ..... 167
10.3 概述。 ..... 167
10.3.1 特点是。 ..... 168
10.3.2 操作模式。 ..... 168

10.4 外部信号描述。 ..... 169
10.5 详细的信号描述。 ..... 169
10.6 内存图和寄存器定义。 ..... 170
10.6.1 引脚控制寄存器n（PORT_PCRn）。 ..... 172
10.6.2 全局引脚控制低寄存器（PORT_GPCLR）。 ..... 175
10.6.3 全局引脚控制高位寄存器（PORT_GPCHR）。 ..... 175
10.6.4 全局中断控制低寄存器（PORT_GICLR）。 ..... 176
10.6.5 全局中断控制高寄存器（PORT_GICHR）。 ..... 176
10.6.6 中断状态标志寄存器（PORT_ISFR）。 ..... 177
10.6.7 数字滤波器启用寄存器（PORT_DFER）。 ..... 178
10.6.8 数字滤波器时钟寄存器（PORT_DFCR）。 ..... 178
10.6.9 数字滤波宽度寄存器（PORT_DFWR）。 ..... 179
10.7 功能描述。 ..... 179
10.7.1 引脚控制。 ..... 179
10.7.2 全局引脚控制。 ..... 180
10.7.3 全局中断控制。 ..... 181
10.7.4 外部中断。 ..... 181
10.7.5 数字滤波器。 ..... 182

## 第十一章
## 通用输入/输出(GPIO)

11.1 芯片特定的GPIO信息。 ..... 183
11.1.1 实例化信息。 ..... 183
11.1.2 GPIO端口内存图。 ..... 183
11.2 介绍。 ..... 184
11.2.1 特点是。 ..... 184
11.2.2 操作模式。 ..... 184
11.2.3 GPIO信号描述。 ..... 185

11.3 内存图和寄存器定义。 ..... 186
11.3.1 GPIO寄存器的描述。 ..... 186
11.4 功能描述。 ..... 193
11.4.1 通用输入。 ..... 193
11.4.2 通用输出。 ..... 193

## 第十二章
## 轻型横梁开关 (AXBS-Lite)

12.1 特定芯片的AXBS-Lite信息。 ..... 195
12.1.1 横梁开关的主控任务。 ..... 195
12.1.2 横梁开关的从属分配。 ..... 195
12.2 介绍。 ..... 196
12.2.1 特点是。 ..... 196
12.3 功能描述。 ..... 197
12.3.1 一般操作。 ..... 197
12.3.2 仲裁。 ..... 197
12.4 初始化/应用信息。 ..... 199

## 第十三章
## 内存保护单元（MPU）

13.1 芯片特定的MPU信息。 ..... 201
13.1.1 MPU从属端口的分配。 ..... 201
13.1.2 MPU逻辑总线主站的分配。 ..... 202
13.1.3 目前的PID。 ..... 202
13.1.4 区域描述符和从属端口配置。 ..... 202
13.2 介绍。 ..... 202
13.3 概述。 ..... 203
13.3.1 框图。 ..... 203
13.3.2 特点是。 ..... 204

13.4 MPU寄存器的描述。 ..... 204
13.4.1 MPU内存图。 ..... 205
13.4.2 控制/错误状态寄存器（CESR）。 ..... 207
13.4.3 错误地址寄存器，从属端口n（EAR0 - EAR4）。 ..... 210
13.4.4 错误详细信息寄存器，从属端口n（EDR0 - EDR4）。 ..... 211
13.4.5 区域描述符n，字0（RGD0_WORD0 - RGD15_WORD0）。 ..... 212
13.4.6 区域描述符0，字1（RGD0_WORD1）。 ..... 214
13.4.7 区域描述符0，字2（RGD0_WORD2）。 ..... 215
13.4.8 区域描述符0，字3（RGD0_WORD3）。 ..... 218
13.4.9 区域描述符n，字1（RGD1_WORD1 - RGD15_WORD1）。 ..... 219
13.4.10 区域描述符n，字2（RGD1_WORD2 - RGD15_WORD2）。 ..... 220
13.4.11 区域描述符n，字3（RGD1_WORD3 - RGD15_WORD3）。 ..... 223
13.4.12 区域描述符备用访问控制0（RGDAAC0）。 ..... 225
13.4.13 区域描述符备用访问控制n（RGDAAC1 - RGDAAC15）。 ..... 228
13.5 功能描述。 ..... 231
13.5.1 访问评估宏。 ..... 231
13.5.2 把这一切放在一起，错误的终止。 ..... 233
13.5.3 电源管理。 ..... 233
13.6 初始化信息。 ..... 234
13.7 申请信息。 ..... 234

## 第14章 周边桥（AIPS-Lite）

14.1 特定芯片的AIPS信息。 ..... 237
14.1.1 实例化信息。 ..... 237
14.1.2 记忆地图。 ..... 237
14.2 介绍。 ..... 238
14.2.1 特点是。 ..... 238

14.2.2 一般操作。 ..... 238
14.3 内存图/寄存器定义。 ..... 239
14.3.1 AIPS寄存器描述。 ..... 239
14.4 功能描述。 ..... 283
14.4.1 访问支持。 ..... 283

## 第十五章
## 直接内存访问复用器（DMAMUX）

15.1 芯片特定的DMAMUX信息。 ..... 285
15.1.1 通道数量 ..... 285
15.1.2 通过TRGMUX触发的DMA传输。 ..... 285
15.2 介绍。 ..... 286
15.2.1 概述。 ..... 286
15.2.2 特点是。 ..... 286
15.2.3 操作模式。 ..... 287
15.3 外部信号描述。 ..... 287
15.4 内存图/寄存器定义。 ..... 287
15.4.1 DMAMUX寄存器的描述。 ..... 287
15.5 功能描述。 ..... 289
15.5.1 具有周期性触发能力的DMA通道。 ..... 290
15.5.2 没有触发能力的DMA通道。 ..... 292
15.5.3 始终启用的DMA源。 ..... 292
15.6 初始化/应用信息。 ..... 293
15.6.1 重置。 ..... 293
15.6.2 启用和配置源。 ..... 293

## 第16章
## 增强型直接内存访问（eDMA）

16.1 芯片特定的eDMA信息 ..... 297

16.1.1 频道数量 ..... 297
16.2 介绍。 ..... 297
16.2.1 eDMA系统框图。 ..... 298
16.2.2 块状部件。 ..... 298
16.2.3 特点是。 ..... 299
16.3 操作模式。 ..... 301
16.4 内存图/寄存器定义。 ..... 301
16.4.1 TCD记忆。 ..... 301
16.4.2 TCD初始化。 ..... 301
16.4.3 TCD结构。 ..... 302
16.4.4 保留的内存和位域。 ..... 302
16.4.5 DMA寄存器的描述。 ..... 302
16.5 功能描述。 ..... 360
16.5.1 eDMA基本数据流。 ..... 360
16.5.2 故障报告和处理。 ..... 363
16.5.3 频道抢占。 ..... 366
16.6 初始化/应用信息。 ..... 366
16.6.1 eDMA初始化。 ..... 366
16.6.2 编程错误。 ..... 368
16.6.3 仲裁模式的考虑。 ..... 369
16.6.4 执行DMA传输。 ..... 369
16.6.5 监控传输描述符的状态。 ..... 373
16.6.6 通道连接。 ..... 375
16.6.7 动态编程。 ..... 376
16.6.8 暂停/恢复一个有活动硬件服务请求的DMA通道。 ..... 380

## 第十七章
## 触发器MUX控制（TRGMUX）

17.1 芯片特定的TRGMUX信息。 ..... 383
17.1.1 模块的互连性。 ..... 383
17.1.2 芯片特定的TRGMUX寄存器。 ..... 387
17.2 介绍。 ..... 387
17.3 特点是。 ..... 387
17.4 内存图和寄存器定义。 ..... 388
17.4.1 TRGMUX寄存器的描述。 ..... 388

## 第18章
## 外部看门狗监控器（EWM）

18.1 特定芯片的EWM信息 ..... 427
18.1.1 EWM_OUT信号配置。 ..... 427
18.1.2 EWM内存地图访问。 ..... 427
18.1.3 EWM低功率模式。 ..... 427
18.2 介绍。 ..... 427
18.2.1 特点是。 ..... 428
18.2.2 操作模式。 ..... 428
18.2.3 框图。 ..... 429
18.3 EWM信号描述。 ..... 430
18.4 内存图/寄存器的定义。 ..... 431
18.4.1 EWM寄存器的描述。 ..... 431
18.5 功能描述。 ..... 436
18.5.1 EWM_OUT_b信号。 ..... 436
18.5.2 EWM_OUT_b引脚在低功率模式下的状态。 ..... 437
18.5.3 EWM_in信号。 ..... 437
18.5.4 EWM计数器。 ..... 438
18.5.5 EWM比较寄存器。 ..... 438
18.5.6 EWM刷新机制。 ..... 438

18.5.7 EWM中断。 ..... 439
18.5.8 计数器时钟预分频器。 ..... 439

## 第十九章
## 错误注入模块（EIM）

19.1 芯片特定的EIM信息。 ..... 441
19.1.1 EIM通道的分配。 ..... 441
19.2 介绍。 ..... 441
19.2.1 概述。 ..... 441
19.2.2 特点是。 ..... 442
19.3 EIM寄存器的描述。 ..... 443
19.3.1 EIM内存地图。 ..... 444
19.3.2 错误注入模块配置寄存器（EIMCR）。 ..... 444
19.3.3 错误注入通道启用寄存器（EICHEN）。 ..... 445
19.3.4 错误注入通道描述符n，字0（EICHD0_WORD0 - EICHD1_WORD0）。 ..... 448
19.3.5 错误注入通道描述符n，字1（EICHD0_WORD1 - EICHD1_WORD1）。 ..... 450
19.4 功能描述。 ..... 451
19.4.1 错误注入情况。 ..... 451

## 第二十章
## 错误报告模块（ERM）。

20.1 特定芯片的ERM信息。 ..... 453
20.1.1 内存错误事件的来源。 ..... 453
20.2 介绍。 ..... 453
20.2.1 概述。 ..... 453
20.2.2 特点是。 ..... 454
20.3 ERM寄存器的描述。 ..... 454
20.3.1 ERM内存地图。 ..... 454
20.3.2 ERM配置寄存器0（CR0）。 ..... 455

20.3.3 ERM状态寄存器0（SR0）。 ..... 457
20.3.4 ERM内存n错误地址寄存器（EAR0 - EAR1）。 ..... 459
20.4 功能描述。 ..... 460
20.4.1 单位校正事件。 ..... 460
20.4.2 不可纠正的错误事件。 ..... 461
20.5 初始化。 ..... 462

## 第21章 看门狗定时器（WDOG）

21.1 芯片特有的WDOG信息。 ..... 463
21.1.1 WDOG的时钟。 ..... 463
21.1.2 WDOG低功率模式。 ..... 463
21.1.3 默认看门狗超时 ..... 463
21.2 介绍。 ..... 464
21.2.1 特点是。 ..... 464
21.2.2 框图。 ..... 465
21.3 内存图和寄存器定义。 ..... 465
21.3.1 WDOG寄存器的描述。 ..... 465
21.4 功能描述。 ..... 472
21.4.1 时钟源。 ..... 472
21.4.2 看门狗刷新机制。 ..... 473
21.4.3 配置看门狗。 ..... 475
21.4.4 使用中断来延迟复位。 ..... 476
21.4.5 备份重置。 ..... 476
21.4.6 调试和低功耗模式下的功能。 ..... 477
21.4.7 对看门狗进行快速测试。 ..... 477
21.5 申请信息。 ..... 479
21.5.1 禁用看门狗。 ..... 479

21.5.2 复位后禁用看门狗。 ..... 479
21.5.3 配置看门狗。 ..... 480
21.5.4 刷新看门狗。 ..... 480

## 第二十二章
## 循环冗余检查（CRC）

22.1 芯片特定的CRC信息。 ..... 481
22.2 介绍。 ..... 481
22.2.1 特点是。 ..... 481
22.2.2 框图。 ..... 482
22.2.3 操作模式。 ..... 482
22.3 内存图和寄存器描述。 ..... 482
22.3.1 CRC寄存器的描述。 ..... 482
22.4 功能描述。 ..... 487
22.4.1 CRC初始化/重新初始化。 ..... 487
22.4.2 CRC的计算方法。 ..... 487
22.4.3 移调功能。 ..... 488
22.4.4 CRC结果补充。 ..... 490

## 第23章 重置和
## 启动

23.1 介绍。 ..... 491
23.2 重置。 ..... 491
23.2.1 开机复位（POR）。 ..... 492
23.2.2 系统复位源。 ..... 492
23.2.3 单片机复位。 ..... 495
23.2.4 复位针 ..... 496
23.2.5 调试复位。 ..... 497
23.3 靴子。 ..... 498

23.3.1 启动源。 ..... 498
23.3.2 FOPT启动选项。 ..... 498
23.3.3 启动顺序。 ..... 499

## 第二十四章
## 复位控制模块（RCM）

24.1 芯片特有的RCM信息。 ..... 501
24.2 在STOP1/2模式下的复位引脚过滤操作 ..... 501
24.3 介绍。 ..... 501
24.4 复位内存图和寄存器描述。 ..... 502
24.4.1 版本ID寄存器（RCM_VERID）。 ..... 502
24.4.2 参数寄存器（RCM_PARAM）。 ..... 503
24.4.3 系统复位状态寄存器（RCM_SRS）。 ..... 505
24.4.4 复位引脚控制寄存器（RCM_RPC）。 ..... 508
24.4.5 粘性系统复位状态寄存器（RCM_SSRS）。 ..... 510
24.4.6 系统复位中断启用寄存器（RCM_SRIE）。 ..... 512

## 第25章 时钟分布

25.1 介绍。 ..... 515
25.2 高水平的时钟图。 ..... 515
25.3 时钟的定义。 ..... 516
25.4 内部时钟要求。 ..... 518
25.4.1 复位后的时钟分频器值。 ..... 521
25.4.2 HSRUN模式的时钟。 ..... 521
25.4.3 VLPR模式的时钟。 ..... 521
25.4.4 VLPR/VLPS模式条目。 ..... 522
25.5 时钟门控。 ..... 522
25.6 模块时钟。 ..... 522

## 第二十六章

# 系统时钟发生器（SCG）


26.1 芯片特定的SCG信息。 ..... 535

26.1.1 支持的频率范围。 ..... 535

26.1.2 振荡器的限制。 ..... 536

26.1.3 通过SCG_FIRCCSR[FIRCEN]启用/禁用FIRC的准则。 ..... 536

26.1.4 系统时钟切换 ..... 536

26.2 介绍。 ..... 537

26.2.1 特点是。 ..... 537

26.3 内存图/寄存器的定义。 ..... 538

26.3.1 版本ID寄存器（SCG_VERID）。 ..... 539

26.3.2 参数寄存器（SCG_PARAM）。 ..... 540

26.3.3 时钟状态寄存器（SCG_CSR）。 ..... 540

26.3.4 运行时钟控制寄存器（SCG_RCCR）。 ..... 543

26.3.5 VLPR时钟控制寄存器（SCG_VCCR）。 ..... 545

26.3.6 HSRUN时钟控制寄存器（SCG_HCCR）。 ..... 547

26.3.7 SCG CLKOUT配置寄存器（SCG_CLKOUTCNFG）。 ..... 549

26.3.8 系统OSC控制状态寄存器（SCG_SOSCCSR）。 ..... 551

26.3.9 系统OSC分频寄存器（SCG_SOSCDIV）。 ..... 553

26.3.10 系统振荡器配置寄存器（SCG_SOSCCFG）。 ..... 554

26.3.11 慢速IRC控制状态寄存器（SCG_SIRCCSR）。 ..... 556

26.3.12 慢速IRC除法寄存器（SCG_SIRCDIV）。 ..... 557

26.3.13 慢速IRC配置寄存器（SCG_SIRCCFG）。 ..... 558

26.3.14 快速IRC控制状态寄存器（SCG_FIRCCSR）。 ..... 559

26.3.15 快速IRC除法寄存器（SCG_FIRCDIV）。 ..... 561

26.3.16 快速IRC配置寄存器（SCG_FIRCCFG）。 ..... 562

26.3.17 系统PLL控制状态寄存器（SCG_SPLLCSR）。 ..... 563

26.3.18 系统PLL除法寄存器（SCG_SPLLDIV）。 ..... 565

26.3.19 系统PLL配置寄存器（SCG_SPLLCFG）。 ..... 566
26.4 功能描述。 ..... 567
26.4.1 SCG时钟模式转换。 ..... 567

## 第二十七章
## 外围时钟控制器（PCC）

27.1 芯片特定的PCC信息。 ..... 571
27.1.1 芯片特定的PCC寄存器信息。 ..... 571
27.2 介绍。 ..... 571
27.3 特点是。 ..... 571
27.4 功能描述。 ..... 572
27.5 内存图和寄存器定义。 ..... 573
27.6 PCC寄存器的描述。 ..... 573
27.6.1 PCC内存地图。 ..... 573
27.6.2 PCC FTFC寄存器（PCC_FTFC）。 ..... 574
27.6.3 PCC DMAMUX寄存器（PCC_DMAMUX）。 ..... 576
27.6.4 PCC FlexCAN0寄存器（PCC_FlexCAN0）。 ..... 577
27.6.5 PCC FlexCAN1寄存器（PCC_FlexCAN1）。 ..... 579
27.6.6 PCC FTM3寄存器（PCC_FTM3）。 ..... 580
27.6.7 PCC ADC1寄存器（PCC_ADC1）。 ..... 582
27.6.8 PCC FlexCAN2寄存器（PCC_FlexCAN2）。 ..... 583
27.6.9 PCC LPSPI0寄存器（PCC_LPSPI0）。 ..... 585
27.6.10 PCC LPSPI1寄存器（PCC_LPSPI1）。 ..... 586
27.6.11 PCC LPSPI2寄存器（PCC_LPSPI2）。 ..... 588
27.6.12 PCC PDB1寄存器（PCC_PDB1）。 ..... 589
27.6.13 PCC CRC寄存器（PCC_CRC）。 ..... 591
27.6.14 PCC PDB0寄存器（PCC_PDB0）。 ..... 592
27.6.15 PCC LPIT寄存器（PCC_LPIT）。 ..... 594
27.6.16 PCC FTM0寄存器（PCC_FTM0）。 ..... 595

27.6.17 PCC FTM1寄存器（PCC_FTM1）。 ..... 597
27.6.18 PCC FTM2寄存器（PCC_FTM2）。 ..... 598
27.6.19 PCC ADC0寄存器（PCC_ADC0）。 ..... 600
27.6.20 PCC RTC寄存器（PCC_RTC）。 ..... 602
27.6.21 PCC LPTMR0寄存器（PCC_LPTMR0）。 ..... 603
27.6.22 PCC PORTA寄存器（PCC_PORTA）。 ..... 605
27.6.23 PCC PORTB寄存器（PCC_PORTB）。 ..... 607
27.6.24 PCC PORTC寄存器（PCC_PORTC）。 ..... 608
27.6.25 PCC PORTD寄存器（PCC_PORTD）。 ..... 610
27.6.26 PCC PORTE寄存器（PCC_PORTE）。 ..... 611
27.6.27 PCC SAI0寄存器（PCC_SAI0）。 ..... 613
27.6.28 PCC SAI1寄存器（PCC_SAI1）。 ..... 614
27.6.29 PCC FlexIO寄存器（PCC_FlexIO）。 ..... 616
27.6.30 PCC EWM寄存器（PCC_EWM）。 ..... 617
27.6.31 PCC LPI2C0寄存器（PCC_LPI2C0）。 ..... 619
27.6.32 PCC LPI2C1寄存器（PCC_LPI2C1）。 ..... 620
27.6.33 PCC LPUART0寄存器（PCC_LPUART0）。 ..... 622
27.6.34 PCC LPUART1寄存器（PCC_LPUART1）。 ..... 623
27.6.35 PCC LPUART2寄存器（PCC_LPUART2）。 ..... 625
27.6.36 PCC FTM4寄存器（PCC_FTM4）。 ..... 627
27.6.37 PCC FTM5寄存器（PCC_FTM5）。 ..... 628
27.6.38 PCC FTM6寄存器（PCC_FTM6）。 ..... 630
27.6.39 PCC FTM7寄存器（PCC_FTM7）。 ..... 632
27.6.40 PCC CMP0寄存器（PCC_CMP0）。 ..... 633
27.6.41 PCC QSPI寄存器（PCC_QSPI）。 ..... 635
27.6.42 PCC ENET寄存器（PCC_ENET）。 ..... 636

## 第二十八章
## 时钟监测单元（CMU）

28.1 介绍。 ..... 639

28.2 寄存器描述。 ..... 640

28.2.1 CMU内存图。 ..... 640

28.2.2 CMU频率检查全局配置寄存器（CMU_FCGCR）。 ..... 641

28.2.3 CMU频率检查参考计数配置寄存器（CMU_FCRCCR）。 ..... 642

28.2.4 CMU频率检查高阈值配置寄存器（CMU_FCHTCR）。 ..... 643

28.2.5 CMU频率检查低阈值配置寄存器（CMU_FCLTCR）。 ..... 644

28.2.6 CMU频率检查状态寄存器（CMU_FCSR）。 ..... 645

28.2.7 CMU频率检查中断/事件启用寄存器（CMU_FCIER）。 ..... 646

28.3 功能描述。 ..... 647

28.4 编程指南。 ..... 648

## 第29章
## 存储器和存储器接口

29.1 介绍。 ..... 651

29.2 闪存控制器和闪存模块。 ..... 651

29.3 SRAM配置。 ..... 651

29.3.1 SRAM尺寸。 ..... 652

29.3.2 SRAM的可及性。 ..... 653

29.3.3 SRAM仲裁和优先权控制。 ..... 654

29.3.4 SRAM保留：电源模式和复位。 ..... 654

## 第30章
## 本地存储器控制器（LMEM）

30.1 特定芯片的LMEM信息 ..... 655

30.1.1 LMEM区域描述。 ..... 655

30.1.2 LMEM SRAM尺寸。 ..... 655

30.2 介绍。 ..... 655

30.2.1 框图。 ..... 655

30.2.2 缓存功能。 ..... 657
30.3 内存图/寄存器的定义。 ..... 659
30.3.1 LMEM寄存器的描述。 ..... 659
30.4 功能描述。 ..... 668
30.4.1 LMEM功能。 ..... 668
30.4.2 SRAM功能。 ..... 669
30.4.3 缓存功能。 ..... 671
30.4.4 缓存控制。 ..... 672

## 第31章
## 杂项系统控制模块（MSCM）。

31.1 芯片特定的MSCM信息。 ..... 677
31.1.1 芯片特定的TMLSZ/TMUSZ信息。 ..... 677
31.1.2 芯片特定的寄存器信息。 ..... 677
31.2 概述。 ..... 678
31.3 芯片配置和启动。 ..... 678
31.4 MSCM内存图/寄存器的定义。 ..... 678
31.4.1 CPU配置内存图和寄存器 ..... 678
31.4.2 MSCM寄存器的描述。 ..... 679

## 第32章
## 闪存控制器（FMC）

32.1 芯片特定的FMC信息。 ..... 711
32.1.1 FMC大师。 ..... 711
32.2 介绍。 ..... 712
32.2.1 概述。 ..... 712
32.2.2 特点是。 ..... 712
32.3 操作模式。 ..... 713
32.4 外部信号描述。 ..... 713

32.5 功能描述。 ..... 713
32.5.1 默认配置。 ..... 713
32.5.2 推测读数。 ..... 714
32.6 初始化和应用信息。 ..... 715

## 第三十三章
## 闪存模块（FTFC）

33.1 芯片特定的FTFC信息。 ..... 717
33.1.1 闪存类型。 ..... 717
33.1.2 闪存的尺寸。 ..... 718
33.1.3 闪存地图。 ..... 718
33.1.4 闪存的安全性。 ..... 719
33.1.5 闪存编程的电源模式限制。 ..... 719
33.1.6 闪存模式。 ..... 719
33.1.7 擦除闪存中的所有内容。 ..... 719
33.1.8 通过FTFC_FOPT寄存器定制MCU操作。 ..... 720
33.1.9 同时对PFLASH读取分区进行操作 ..... 720
33.2 介绍。 ..... 720
33.2.1 特点是。 ..... 722
33.2.2 框图。 ..... 724
33.2.3 词汇表。 ..... 729
33.3 外部信号描述。 ..... 732
33.4 内存图和寄存器。 ..... 732
33.4.1 闪存配置字段描述。 ..... 732
33.4.2 程序闪光0 IFR地图。 ..... 733
33.4.3 数据闪存0 IFR地图。 ..... 733
33.4.4 登记册描述。 ..... 734
33.5 功能描述。 ..... 752

33.5.1 闪光保护。 ..... 752
33.5.2 FlexNVM描述。 ..... 754
33.5.3 中断。 ..... 757
33.5.4 在低功耗模式下的闪光操作。 ..... 757
33.5.5 运作的功能模式。 ..... 758
33.5.6 闪存的读取和忽略的写入。 ..... 758
33.5.7 边写边读（RWW）。 ..... 758
33.5.8 Flash编程和擦除。 ..... 759
33.5.9 FTFC的指挥业务。 ..... 759
33.5.10 保证金读取命令。 ..... 766
33.5.11 Flash命令描述。 ..... 767
33.5.12 安全性。 ..... 793
33.5.13 加密服务引擎（CSEc）。 ..... 795
33.5.14 复位顺序。 ..... 833

## 第34章
## 四个串行外设接口（QuadSPI）。

34.1 芯片特定的QuadSPI信息。 ..... 835
34.1.1 概述。 ..... 835
34.1.2 内存大小要求 ..... 835
34.1.3 QuadSPI寄存器的复位值。 ..... 835
34.1.4 使用案例。 ..... 836
34.1.5 支持的读取模式。 ..... 836
34.1.6 外部存储器选项。 ..... 837
34.1.7 推荐的软件配置。 ..... 837
34.1.8 推荐的编程顺序。 ..... 838
34.1.9 QuadSPI时钟之间的时钟比率 ..... 838
34.1.10 QuadSPI_MCR[SCLKCFG]的实现 ..... 838

34.1.11 QuadSPI_SOCCR[SOCCFG]的实现 ..... 839
34.2 介绍。 ..... 841
34.2.1 特点是。 ..... 841
34.2.2 框图。 ..... 842
34.2.3 QuadSPI的操作模式。 ..... 843
34.2.4 首字母缩写词和缩略语。 ..... 844
34.2.5 QuadSPI模块的术语。 ..... 844
34.3 外部信号描述。 ..... 846
34.3.1 驱动外部信号。 ..... 847
34.4 内存地图和寄存器的定义。 ..... 849
34.4.1 寄存器写访问。 ..... 849
34.4.2 周边总线寄存器描述。 ..... 850
34.4.3 串行闪存地址的分配。 ..... 893
34.5 闪存映射的AMBA总线。 ..... 894
34.5.1 AHB总线访问的考虑因素。 ..... 895
34.5.2 内存映射的串行闪存数据--闪存A上的单独闪存模式。 ..... 895
34.5.3 内存映射的串行闪存数据--闪存B上的单独闪存模式。 ..... 896
34.5.4 AHB RX数据缓冲器（QSPI_ARDB0至QSPI_ARDB31）。 ..... 897
34.6 中断信号。 ..... 899
34.7 功能描述。 ..... 900
34.7.1 串行闪存访问方案。 ..... 900
34.7.2 正常模式。 ..... 900
34.7.3 支持HyperRAM。 ..... 919
34.8 初始化/应用信息。 ..... 920
34.8.1 上电和复位。 ..... 920
34.8.2 可用的状态/旗帜信息。 ..... 920
34.8.3 为AHB命令独占访问串行闪存。 ..... 923

34.8.4 指挥部仲裁 ..... 924
34.8.5 闪存设备选择。 ..... 925
34.8.6 DMA使用情况。 ..... 925
34.9 字节排序 - Endianness。 ..... 929
34.9.1 编程闪存数据。 ..... 930
34.9.2 将闪存数据读入RX缓冲区。 ..... 931
34.9.3 将闪存数据读入AHB缓冲区。 ..... 932
34.10 在单一和双重模式下驱动闪光控制信号。 ..... 932
34.11 串行闪存设备。 ..... 933
34.11.1 序列示例。 ..... 933
34.12 对串行闪存输入数据进行采样。 ..... 939
34.12.1 基本描述。 ..... 939
34.12.2 支持的读取模式。 ..... 940
34.12.3 数据选通（DQS）采样方法。 ..... 942
34.13 闪存的数据输入保持要求。 ..... 946

## 第35章 电源管理

35.1 介绍。 ..... 947
35.2 电源模式描述。 ..... 947
35.3 进入和退出电源模式。 ..... 949
35.4 时钟模式。 ..... 949
35.4.1 时钟门控。 ..... 949
35.4.2 停止模式选项。 ..... 949
35.4.3 DMA唤醒。 ..... 950
35.4.4 计算操作（CPO）。 ..... 951
35.4.5 周边打盹。 ..... 952
35.5 电源模式转换。 ..... 953
35.6 电源模式的关机排序。 ..... 954

35.7 闪存编程的电源模式限制。 ..... 954
35.8 模块在可用的低功率模式下运行。 ..... 954
35.9 QuadSPI、以太网和SAI操作 ..... 958

## 第三十六章
## 系统模式控制器（SMC）

36.1 介绍。 ..... 959
36.2 操作模式。 ..... 959
36.3 内存图和寄存器描述。 ..... 961
36.3.1 SMC版本ID寄存器（SMC_VERID）。 ..... 961
36.3.2 SMC参数寄存器（SMC_PARAM）。 ..... 962
36.3.3 电源模式保护寄存器（SMC_PMPROT）。 ..... 963
36.3.4 电源模式控制寄存器（SMC_PMCTRL）。 ..... 965
36.3.5 停止控制寄存器（SMC_STOPCTRL）。 ..... 967
36.3.6 电源模式状态寄存器（SMC_PMSTAT）。 ..... 968
36.4 功能描述。 ..... 969
36.4.1 电源模式转换。 ..... 969
36.4.2 电源模式进入/退出顺序。 ..... 970
36.4.3 运行模式。 ..... 972
36.4.4 停止模式。 ..... 974
36.4.5 在低功率模式下进行调试。 ..... 976

## 第37章
## 电源管理控制器（PMC）

37.1 芯片特定的PMC信息。 ..... 977
37.1.1 支持的模式。 ..... 977
37.2 介绍。 ..... 977
37.3 特点是。 ..... 977
37.4 操作模式。 ..... 977

37.4.1 全性能模式（FPM）。 ..... 977
37.4.2 低功率模式（LPM）。 ..... 978
37.5 低电压检测（LVD）系统。 ..... 978
37.5.1 低电压复位（LVR）操作。 ..... 978
37.5.2 LVD中断操作。 ..... 979
37.5.3 低电压警告（LVW）中断操作。 ..... 979
37.6 内存地图和寄存器的定义。 ..... 979
37.6.1 PMC寄存器的描述。 ..... 979

## 第38章 ADC配置

38.1 实例化信息。 ..... 987
38.1.1 ADC通道的数量。 ..... 987
38.1.2 ADC连接/通道分配。 ..... 988
38.2 登记实施。 ..... 989
38.3 ADC上的DMA支持。 ..... 989
38.4 ADC硬件交错通道。 ..... 990
38.5 ADC内部电源监控。 ..... 991
38.6 ADC参考选项。 ..... 991
38.7 ADC触发源。 ..... 991
38.7.1 PDB的触发方案。 ..... 993
38.7.2 TRGMUX触发方案。 ..... 994
38.8 触发器选择。 ..... 995
38.9 触发器锁定和仲裁。 ..... 996
38.10 ADC的低功耗模式。 ..... 998
38.11 ADC触发器概念 - 使用案例。 ..... 998
38.12 ADC校准方案。 ..... 1000

## 第39章

# 模拟数字转换器（ADC）


39.1 芯片特定的ADC信息。 ..... 1003

39.2 介绍。 ..... 1003

39.2.1 特点是。 ..... 1003

39.2.2 框图。 ..... 1004

39.3 ADC信号描述。 ..... 1005

39.3.1 模拟电源（VDDA）。 ..... 1005

39.3.2 模拟地（VSSA）。 ..... 1005

39.3.3 电压基准选择。 ..... 1005

39.3.4 模拟通道输入（ADx）。 ..... 1006

39.4 ADC寄存器的描述。 ..... 1006

39.4.1 ADC内存图。 ..... 1006

39.4.2 ADC状态和控制寄存器1（SC1A - aSC1P）。 ..... 1008

39.4.3 ADC配置寄存器1（CFG1）。 ..... 1012

39.4.4 ADC配置寄存器2（CFG2）。 ..... 1013

39.4.5 ADC数据结果寄存器（RA - aRP）。 ..... 1014

39.4.6 比较值寄存器（CV1-CV2）。 ..... 1016

39.4.7 状态和控制寄存器2（SC2）。 ..... 1017

39.4.8 状态和控制寄存器3（SC3）。 ..... 1020

39.4.9 BASE偏移寄存器（BASE_OFS）。 ..... 1021

39.4.10 ADC偏移校正寄存器（OFS）。 ..... 1022

39.4.11 USR_OFS）。 ..... 1023

39.4.12 ADC X偏移校正寄存器（XOFS）。 ..... 1024

39.4.13 ADC Y偏移校正寄存器（YOFS）。 ..... 1025

39.4.14 ADC增益寄存器（G）。 ..... 1026

39.4.15 ADC用户增益寄存器（UG）。 ..... 1027

39.4.16 ADC通用校准值寄存器S（CLPS）。 ..... 1028

39.4.17 ADC Plus-Side通用校准值寄存器3（CLP3）。 ..... 1030
39.4.18 ADC Plus-Side通用校准值寄存器2（CLP2）。 ..... 1030
39.4.19 ADC Plus-Side通用校准值寄存器1（CLP1）。 ..... 1031
39.4.20 ADC Plus-Side通用校准值寄存器0（CLP0）。 ..... 1032
39.4.21 ADC Plus-Side通用校准值寄存器X（CLPX）。 ..... 1033
39.4.22 ADC Plus-Side通用校准值寄存器9（CLP9）。 ..... 1034
39.4.23 ADC通用校准偏移值寄存器S（CLPS_OFS）。 ..... 1035
39.4.24 ADC Plus-Side通用校准偏移值寄存器3（CLP3_OFS）。 ..... 1036
39.4.25 ADC Plus-Side通用校准偏移值寄存器2（CLP2_OFS）。 ..... 1037
39.4.26 ADC Plus-Side通用校准偏移值寄存器1（CLP1_OFS）。 ..... 1038
39.4.27 ADC Plus-Side通用校准偏移值寄存器0（CLP0_OFS）。 ..... 1039
39.4.28 ADC Plus-Side通用校准偏移值寄存器X（CLPX_OFS）。 ..... 1040
39.4.29 ADC Plus-Side通用校准偏移值寄存器9（CLP9_OFS）。 ..... 1041
39.4.30 ADC状态和控制寄存器1（SC1Q - SC1AF）。 ..... 1042
39.4.31 ADC数据结果寄存器（RQ - RAF）。 ..... 1045
39.5 功能描述。 ..... 1046
39.5.1 时钟选择和分频控制。 ..... 1046
39.5.2 电压基准选择。 ..... 1047
39.5.3 硬件触发器和通道选择。 ..... 1047
39.5.4 转换控制。 ..... 1048
39.5.5 自动比较功能。 ..... 1052
39.5.6 校准功能。 ..... 1053
39.5.7 用户定义的偏移功能。 ..... 1054
39.5.8 单片机正常停止模式操作。 ..... 1055

## 第40章 比较器（CMP）

40.1 芯片特定的CMP信息。 ..... 1057

40.1.1 实例化信息。 ..... 1057
40.1.2 CMP输入连接。 ..... 1057
40.1.3 CMP的外部参考资料。 ..... 1058
40.1.4 外部窗口/样品输入。 ..... 1058
40.1.5 CMP触发模式。 ..... 1058
40.1.6 编程建议。 ..... 1059
40.2 介绍。 ..... 1059
40.3 特点是。 ..... 1060
40.3.1 CMP的特点。 ..... 1060
40.3.2 8位DAC的主要特点。 ..... 1061
40.3.3 ANMUX的主要特点。 ..... 1061
40.4 CMP、DAC和ANMUX图。 ..... 1061
40.5 CMP框图。 ..... 1062
40.6 CMP引脚描述。 ..... 1064
40.6.1 外部引脚。 ..... 1064
40.7 CMP的功能模式。 ..... 1065
40.7.1 禁用模式（#1）。 ..... 1066
40.7.2 连续模式（#2A和2B）。 ..... 1066
40.7.3 采样，非过滤模式（#s 3A & 3B）。 ..... 1067
40.7.4 采样，过滤模式（#s 4A & 4B）。 ..... 1069
40.7.5 窗口模式（#s 5A & 5B）。 ..... 1071
40.7.6 窗口/取样模式（#6）。 ..... 1073
40.7.7 窗口/过滤模式（#7）。 ..... 1074
40.8 内存图/寄存器的定义。 ..... 1075
40.8.1 CMP控制寄存器0（CMP_C0）。 ..... 1075
40.8.2 CMP控制寄存器1（CMP_C1）。 ..... 1079
40.8.3 CMP控制寄存器2（CMP_C2）。 ..... 1082

40.9 CMP功能描述。 ..... 1084
40.9.1 初始化。 ..... 1084
40.9.2 低通滤波器。 ..... 1085
40.10 中断。 ..... 1087
40.11 支持DMA。 ..... 1087
40.12 DAC功能描述。 ..... 1088
40.12.1 数字-模拟转换器框图。 ..... 1088
40.12.2 电压参考源选择。 ..... 1088
40.13 DAC复位。 ..... 1089
40.14 DAC时钟。 ..... 1089
40.15 DAC中断。 ..... 1089
40.16 触发模式。 ..... 1089

## 第41章 可编程延迟块（PDB）

41.1 芯片特定的PDB信息。 ..... 1093
41.1.1 实例化信息。 ..... 1093
41.1.2 PDB触发器与ADC和TRGMUX的互连。 ..... 1094
41.1.3 背对背的确认连接。 ..... 1094
41.1.4 脉冲输出使能寄存器的实现。 ..... 1101
41.2 介绍。 ..... 1101
41.2.1 特点是。 ..... 1101
41.2.2 实施。 ..... 1102
41.2.3 背对背的确认连接。 ..... 1102
41.2.4 框图。 ..... 1103
41.2.5 操作模式。 ..... 1104
41.3 内存图和寄存器定义。 ..... 1104
41.3.1 状态和控制寄存器（PDB_SC）。 ..... 1106

41.3.2 模数寄存器（PDB_MOD）。 ..... 1109
41.3.3 计数器寄存器（PDB_CNT）。 ..... 1110
41.3.4 中断延时寄存器（PDB_IDLY）。 ..... 1110
41.3.5 通道n控制寄存器1（PDB_CHnC1）。 ..... 1111
41.3.6 通道n状态寄存器（PDB_CHnS）。 ..... 1112
41.3.7 通道n延迟0寄存器（PDB_CHnDLY0）。 ..... 1112
41.3.8 通道n延迟1寄存器（PDB_CHnDLY1）。 ..... 1113
41.3.9 通道n延时2寄存器（PDB_CHnDLY2）。 ..... 1114
41.3.10 通道n延迟3寄存器（PDB_CHnDLY3）。 ..... 1114
41.3.11 通道n延迟4寄存器（PDB_CHnDLY4）。 ..... 1115
41.3.12 通道n延迟5寄存器（PDB_CHnDLY5）。 ..... 1116
41.3.13 通道n延迟6寄存器（PDB_CHnDLY6）。 ..... 1116
41.3.14 通道n延迟7寄存器（PDB_CHnDLY7）。 ..... 1117
41.3.15 脉冲输出n启用寄存器（PDB_POEN）。 ..... 1117
41.3.16 脉冲输出n延时寄存器（PDB_POnDLY）。 ..... 1118
41.4 功能描述。 ..... 1118
41.4.1 PDB预触发和触发输出。 ..... 1118
41.4.2 PDB触发输入源选择。 ..... 1120
41.4.3 脉冲输出的。 ..... 1121
41.4.4 更新延迟寄存器。 ..... 1122
41.4.5 中断。 ..... 1124
41.4.6 DMA. ..... 1124
41.5 申请信息。 ..... 1124
41.5.1 使用预分频器和乘法系数对定时分辨率的影响。 ..... 1124

## 第四十二章

## FlexTimer模块（FTM）

42.1 特定芯片的FTM信息。 ..... 1127

42.1.1 实例化信息。 ..... 1127
42.1.2 FTM中断。 ..... 1128
42.1.3 FTM故障检测输入。 ..... 1128
42.1.4 FTM硬件触发器和同步化。 ..... 1130
42.1.5 FTM输入捕捉选项。 ..... 1132
42.1.6 支持FTM霍尔传感器。 ..... 1133
42.1.7 FTM调制的实现。 ..... 1133
42.1.8 FTM全球时基。 ..... 1134
42.1.9 FTM BDM和调试停顿模式。 ..... 1135
42.2 介绍。 ..... 1135
42.2.1 FlexTimer理念。 ..... 1135
42.2.2 特点是。 ..... 1136
42.2.3 操作模式。 ..... 1138
42.2.4 框图。 ..... 1138
42.3 FTM信号描述。 ..... 1141
42.4 内存图和寄存器定义。 ..... 1141
42.4.1 内存地图。 ..... 1141
42.4.2 寄存器描述。 ..... 1142
42.4.3 FTM寄存器描述。 ..... 1142
42.5 功能描述。 ..... 1201
42.5.1 时钟源。 ..... 1201
42.5.2 预分频器。 ..... 1201
42.5.3 柜台。 ..... 1202
42.5.4 通道模式。 ..... 1208
42.5.5 输入捕捉模式。 ..... 1210
42.5.6 输出比较模式。 ..... 1215
42.5.7 边缘对齐的PWM（EPWM）模式。 ..... 1216

42.5.8 中心对齐的PWM（CPWM）模式。 ..... 1218
42.5.9 合并模式。 ..... 1220
42.5.10 修改后的组合式PWM模式。 ..... 1228
42.5.11 互补模式。 ..... 1231
42.5.12 从写缓冲区更新的寄存器。 ..... 1232
42.5.13 PWM同步。 ..... 1234
42.5.14 颠倒了。 ..... 1250
42.5.15 软件输出控制模式。 ..... 1251
42.5.16 死亡时间的插入。 ..... 1253
42.5.17 输出掩码。 ..... 1257
42.5.18 故障控制。 ..... 1258
42.5.19 极性控制。 ..... 1262
42.5.20 初始化。 ..... 1263
42.5.21 特点优先。 ..... 1263
42.5.22 外部触发器。 ..... 1264
42.5.23 初始化触发器。 ..... 1265
42.5.24 捕获测试模式。 ..... 1267
42.5.25 DMA. ..... 1268
42.5.26 双边缘捕获模式。 ..... 1269
42.5.27 正交解码器模式。 ..... 1277
42.5.28 调试模式。 ..... 1283
42.5.29 重新装填的点数。 ..... 1284
42.5.30 全球负载。 ..... 1287
42.5.31 全球时基（GTB）。 ..... 1288
42.5.32 通道触发输出。 ..... 1289
42.5.33 外部控制通道的输出。 ..... 1290
42.5.34 抖动。 ..... 1290

42.6 重置概述。 ..... 1301
42.7 FTM中断。 ..... 1303
42.7.1 定时器溢出中断。 ..... 1303
42.7.2 重新加载点中断。 ..... 1303
42.7.3 通道（n）中断。 ..... 1303
42.7.4 故障中断。 ..... 1303
42.8 初始化程序。 ..... 1304

## 第43章
## 低功耗中断定时器（LPIT）。

43.1 特定芯片的LPIT信息。 ..... 1307
43.1.1 实例化信息。 ..... 1307
43.1.2 LPIT/DMA周期性触发器的分配..... 1307
43.1.3 LPIT输入触发器..... 1308
43.1.4 LPIT/ADC触发器。 ..... 1308
43.2 介绍。 ..... 1309
43.2.1 概述。 ..... 1309
43.2.2 框图。 ..... 1310
43.3 操作模式。 ..... 1311
43.4 存储器地图和寄存器。 ..... 1312
43.4.1 LPIT寄存器的描述。 ..... 1312
43.5 功能描述。 ..... 1325
43.5.1 初始化。 ..... 1325
43.5.2 计时器模式。 ..... 1326
43.5.3 计时器的触发器控制。 ..... 1327
43.5.4 通道连锁。 ..... 1328
43.5.5 详细时间。 ..... 1328

## 第四十四章

## 低功耗定时器（LPTMR）


44.1 芯片特定的LPTMR信息。 ..... 1341

44.1.1 实例化信息。 ..... 1341

44.1.2 LPTMR脉冲计数器输入选项。 ..... 1341

44.2 介绍。 ..... 1342

44.2.1 特点是。 ..... 1342

44.2.2 操作模式。 ..... 1342

44.3 LPTMR信号描述。 ..... 1343

44.3.1 详细的信号描述。 ..... 1343

44.4 内存图和寄存器定义。 ..... 1343

44.4.1 LPTMR 寄存器的描述。 ..... 1343

44.5 功能描述。 ..... 1349

44.5.1 LPTMR电源和复位。 ..... 1349

44.5.2 LPTMR的时钟。 ..... 1349

44.5.3 LPTMR预分频器/突变滤波器。 ..... 1350

44.5.4 LPTMR比较。 ..... 1351

44.5.5 LPTMR计数器。 ..... 1351

44.5.6 LPTMR硬件触发。 ..... 1352

44.5.7 LPTMR中断。 ..... 1352


## 第45章
## 实时时钟（RTC）


45.1 芯片特定的RTC信息。 ..... 1355

45.1.1 RTC实例化。 ..... 1355

45.1.2 RTC中断 ..... 1355

45.1.3 软件推荐。 ..... 1355

45.2 介绍。 ..... 1355

45.2.1 特点是。 ..... 1356

45.2.2 操作模式。 ..... 1356
45.2.3 RTC信号描述。 ..... 1356
45.3 登记册定义。 ..... 1357
45.3.1 RTC寄存器的描述。 ..... 1357
45.4 功能描述。 ..... 1367
45.4.1 电源、时钟和复位。 ..... 1367
45.4.2 时间计数器。 ..... 1368
45.4.3 赔偿。 ..... 1369
45.4.4 时间报警。 ..... 1369
45.4.5 更新模式。 ..... 1370
45.4.6 注册锁。 ..... 1370
45.4.7 中断。 ..... 1370

## 第46章
## 低功耗串行外设接口（LPSPI）。

46.1 芯片特定的LPSPI信息。 ..... 1371
46.1.1 实例化信息。 ..... 1371
46.2 介绍。 ..... 1372
46.2.1 概述。 ..... 1372
46.2.2 特点是。 ..... 1372
46.2.3 框图。 ..... 1372
46.2.4 操作模式。 ..... 1373
46.2.5 信号描述。 ..... 1373
46.3 内存地图和寄存器。 ..... 1374
46.3.1 LPSPI寄存器的描述。 ..... 1374
46.4 功能描述。 ..... 1396
46.4.1 时钟和复位。 ..... 1396
46.4.2 主模式。 ..... 1397

46.4.3 从属模式。 ..... 1402
46.4.4 中断和DMA请求。 ..... 1404
46.4.5 周边触发器。 ..... 1405

## 第47章
## 低功耗互连电路（LPI2C）。

47.1 芯片特定的LPI2C信息。 ..... 1407
47.1.1 实例化信息。 ..... 1407
47.2 介绍。 ..... 1408
47.2.1 特点是。 ..... 1408
47.2.2 框图。 ..... 1410
47.2.3 操作模式。 ..... 1410
47.2.4 信号描述。 ..... 1411
47.2.5 接线选项。 ..... 1411
47.3 存储器地图和寄存器。 ..... 1413
47.3.1 LPI2C寄存器描述。 ..... 1413
47.4 功能描述。 ..... 1452
47.4.1 时钟和复位。 ..... 1452
47.4.2 主模式。 ..... 1452
47.4.3 从属模式。 ..... 1458
47.4.4 中断和DMA请求。 ..... 1461
47.4.5 周边触发器。 ..... 1463

## 第48章
## 低功耗通用异步接收器/发射器（LPUART）。

48.1 芯片特定的LPUART信息。 ..... 1465
48.1.1 实例化信息。 ..... 1465
48.2 介绍。 ..... 1466
48.2.1 特点是。 ..... 1466

48.2.2 操作模式。 ..... 1467
48.2.3 信号描述。 ..... 1467
48.2.4 框图。 ..... 1467
48.3 登记簿的定义。 ..... 1469
48.3.1 LPUART寄存器描述。 ..... 1469
48.4 功能描述。 ..... 1494
48.4.1 波特率的产生。 ..... 1494
48.4.2 发射器功能描述。 ..... 1495
48.4.3 接收器功能描述。 ..... 1498
48.4.4 额外的LPUART功能。 ..... 1505
48.4.5 红外线接口。 ..... 1507
48.4.6 中断和状态标志。 ..... 1508
48.4.7 周边触发器。 ..... 1509

## 第49章 灵活的I/O（FlexIO）

49.1 芯片特定的FlexIO信息。 ..... 1511
49.1.1 FlexIO配置。 ..... 1511
49.2 介绍。 ..... 1511
49.2.1 概述。 ..... 1511
49.2.2 特点是。 ..... 1512
49.2.3 框图。 ..... 1512
49.2.4 操作模式。 ..... 1513
49.2.5 FlexIO信号描述。 ..... 1513
49.3 存储器地图和寄存器。 ..... 1514
49.3.1 FLEXIO寄存器描述。 ..... 1514
49.4 功能描述。 ..... 1538
49.4.1 时钟和复位。 ..... 1538

49.4.2 变速器操作。 ..... 1538
49.4.3 计时器操作。 ..... 1541
49.4.4 引脚操作。 ..... 1544
49.4.5 中断和DMA请求。 ..... 1545
49.4.6 周边触发器。 ..... 1545
49.5 申请信息。 ..... 1545
49.5.1 UART传输。 ..... 1546
49.5.2 UART接收。 ..... 1547
49.5.3 SPI主控。 ..... 1548
49.5.4 SPI从机。 ..... 1550
49.5.5 I2C主站。 ..... 1552
49.5.6 I2S主控。 ..... 1554
49.5.7 I2S从机。 ..... 1555

## 第50章 FlexCAN

50.1 特定芯片的FlexCAN信息。 ..... 1559
50.1.1 实例化信息。 ..... 1559
50.1.2 MDIS位的复位值。 ..... 1560
50.1.3 FlexCAN外部时间刻度 ..... 1560
50.1.4 FlexCAN中断。 ..... 1560
50.1.5 低功率模式下的FlexCAN操作。 ..... 1561
50.1.6 FlexCAN 振荡器时钟。 ..... 1561
50.2 介绍。 ..... 1561
50.2.1 概述。 ..... 1562
50.2.2 FlexCAN模块的特点。 ..... 1563
50.2.3 操作模式。 ..... 1565
50.3 FlexCAN信号描述。 ..... 1567
50.3.1 CAN Rx ..... 1567

50.3.2 CAN Tx ..... 1567
50.4 内存图/寄存器定义。 ..... 1567
50.4.1 FlexCAN内存映射。 ..... 1567
50.4.2 CAN寄存器的描述。 ..... 1569
50.4.3 消息缓冲区结构。 ..... 1635
50.4.4 用于CAN FD的FlexCAN内存分区。 ..... 1641
50.4.5 FlexCAN消息缓冲区内存图。 ..... 1642
50.4.6 Rx FIFO结构。 ..... 1645
50.5 功能描述。 ..... 1647
50.5.1 传输过程。 ..... 1648
50.5.2 仲裁程序。 ..... 1649
50.5.3 接收过程。 ..... 1652
50.5.4 匹配过程。 ..... 1655
50.5.5 假装联网模式下的接收过程。 ..... 1659
50.5.6 移动过程。 ..... 1664
50.5.7 数据的一致性。 ..... 1666
50.5.8 Rx FIFO。 ..... 1669
50.5.9 CAN协议的相关功能。 ..... 1672
50.5.10 时钟域和限制。 ..... 1692
50.5.11 操作模式的细节。 ..... 1696
50.5.12 中断。 ..... 1700
50.5.13 总线接口。 ..... 1701
50.6 初始化/应用信息。 ..... 1702
50.6.1 FlexCAN初始化序列。 ..... 1702

## 第51章 同步音频接口 (SAI)

51.1 特定芯片的SAI信息 ..... 1705

51.1.1 SAI配置。 ..... 1705
51.1.2 芯片特定的寄存器信息。 ..... 1706
51.2 介绍。 ..... 1706
51.2.1 特点是。 ..... 1706
51.2.2 框图。 ..... 1707
51.2.3 操作模式。 ..... 1707
51.3 外部信号。 ..... 1708
51.4 内存图和寄存器定义。 ..... 1708
51.4.1 I2S寄存器描述。 ..... 1708
51.5 功能描述。 ..... 1738
51.5.1 SAI打卡。 ..... 1738
51.5.2 SAI复位。 ..... 1740
51.5.3 同步模式。 ..... 1740
51.5.4 帧同步配置。 ..... 1741
51.5.5 数据FIFO。 ..... 1742
51.5.6 字掩码寄存器。 ..... 1745
51.5.7 中断和DMA请求。 ..... 1746

## 第52章 以太网 MAC（ENET）

52.1 芯片特定的ENET信息。 ..... 1749
52.1.1 ENET运行期间的软件指南。 ..... 1749
52.2 介绍。 ..... 1749
52.3 概述。 ..... 1750
52.3.1 特点是。 ..... 1750
52.3.2 框图。 ..... 1753
52.4 外部信号描述。 ..... 1753
52.5 内存图/寄存器定义。 ..... 1755

52.5.1 中断事件寄存器（ENET_EIR）。 ..... 1760
52.5.2 中断屏蔽寄存器（ENET_EIMR）。 ..... 1763
52.5.3 接收描述符活动寄存器（ENET_RDAR）。 ..... 1766
52.5.4 发送描述符激活寄存器（ENET_TDAR）。 ..... 1766
52.5.5 以太网控制寄存器（ENET_ECR）。 ..... 1767
52.5.6 MII管理帧寄存器（ENET_MMFR）。 ..... 1769
52.5.7 MII速度控制寄存器（ENET_MSCR）。 ..... 1770
52.5.8 MIB控制寄存器（ENET_MIBC）。 ..... 1772
52.5.9 接收控制寄存器（ENET_RCR）。 ..... 1773
52.5.10 发送控制寄存器（ENET_TCR）。 ..... 1776
52.5.11 物理地址下限寄存器（ENET_PALR）。 ..... 1778
52.5.12 物理地址上位寄存器（ENET_PAUR）。 ..... 1778
52.5.13 操作码/暂停持续时间寄存器（ENET_OPD）。 ..... 1779
52.5.14 描述符个人上位地址寄存器（ENET_IAUR）。 ..... 1779
52.5.15 描述符个人下位地址寄存器（ENET_IALR）。 ..... 1780
52.5.16 描述符组上位地址寄存器（ENET_GAUR）。 ..... 1780
52.5.17 描述符组下位地址寄存器（ENET_GALR）。 ..... 1781
52.5.18 发送FIFO水印寄存器（ENET_TFWR）。 ..... 1781
52.5.19 接收描述符环开始寄存器（ENET_RDSR）。 ..... 1782
52.5.20 发送缓冲区描述符环形启动寄存器（ENET_TDSR）。 ..... 1783
52.5.21 最大接收缓冲区大小寄存器（ENET_MRBR）。 ..... 1784
52.5.22 接收FIFO段满阈值（ENET_RSFL）。 ..... 1785
52.5.23 接收FIFO部分空阈值（ENET_RSEM）。 ..... 1785
52.5.24 接收FIFO几乎清空的阈值（ENET_RAEM）。 ..... 1786
52.5.25 接收FIFO几乎满的阈值（ENET_RAFL）。 ..... 1786
52.5.26 发送FIFO部分空阈值（ENET_TSEM）。 ..... 1787
52.5.27 发送FIFO几乎为空的阈值（ENET_TAEM）。 ..... 1787

52.5.28 发送FIFO几乎满的阈值（ENET_TAFL）。 ..... 1788
52.5.29 发送包间间隙（ENET_TIPG）。 ..... 1788
52.5.30 帧截断长度（ENET_FTRL）。 ..... 1789
52.5.31 发送加速器功能配置（ENET_TACC）。 ..... 1789
52.5.32 接收加速器功能配置（ENET_RACC）。 ..... 1790
52.5.33 保留统计寄存器（ENET_RMON_T_DROP）。 ..... 1791
52.5.34 Tx数据包计数统计寄存器（ENET_RMON_T_PACKETS）。 ..... 1792
52.5.35 Tx广播包统计寄存器（ENET_RMON_T_BC_PKT）。 ..... 1792
52.5.36 Tx Multicast Packets Statistic Register（ENET_RMON_T_MC_PKT）。 ..... 1793
52.5.37 含有CRC/Align错误的Tx数据包统计寄存器（ENET_RMON_T_CRC_ALIGN）。 ..... 1793
52.5.38 Tx数据包小于字节和良好CRC统计寄存器（ENET_RMON_T_UNDERSIZE）。 ..... 1793
52.5.39 Tx数据包GT MAX_FL字节和良好CRC统计寄存器（ENET_RMON_T_OVERSIZE）。 ..... 1794
52.5.40 小于64字节的Tx数据包和坏CRC统计寄存器（ENET_RMON_T_FRAG）。 ..... 1794
52.5.41 大于MAX_FL字节的Tx数据包和坏CRC统计寄存器（ENET_RMON_T_JAB）。 ..... 1795
52.5.42 Tx碰撞计数统计寄存器（ENET_RMON_T_COL）。 ..... 1795
52.5.43 Tx 64字节数据包统计寄存器（ENET_RMON_T_P64）。 ..... 1795
52.5.44 Tx 65-127字节数据包统计寄存器（ENET_RMON_T_P65TO127）。 ..... 1796
52.5.45 Tx 128-255字节数据包统计寄存器（ENET_RMON_T_P128TO255）。 ..... 1796
52.5.46 Tx 256-至511字节的数据包统计寄存器（ENET_RMON_T_P256TO511）。 ..... 1797
52.5.47 Tx 512-至1023字节数据包统计寄存器（ENET_RMON_T_P512TO1023）。 ..... 1797
52.5.48 Tx 1024-至2047字节数据包统计寄存器（ENET_RMON_T_P1024TO2047）。 ..... 1798
52.5.49 Tx Packets Greater Than 2048 Bytes Statistic Register（ENET_RMON_T_P_GTE2048）。 ..... 1798
52.5.50 Tx Octets统计寄存器（ENET_RMON_T_OCTETS）。 ..... 1798
52.5.51 保留统计寄存器（ENET_IEEE_T_DROP）。 ..... 1799
52.5.52 帧传输OK统计寄存器（ENET_IEEE_T_FRAME_OK）。 ..... 1799
52.5.53 用单次碰撞统计寄存器（ENET_IEEE_T_1COL）传输的帧数。 ..... 1800
52.5.54 传送的帧有多次碰撞统计寄存器（ENET_IEEE_T_MCOL）。 ..... 1800

52.5.55 延迟延迟后传输的帧数统计寄存器（ENET_IEEE_T_DEF）。 ..... 1800
52.5.56 用晚期碰撞统计寄存器传输的帧（ENET_IEEE_T_LCOL）。 ..... 1801
52.5.57 碰撞过多时传输的帧数统计寄存器（ENET_IEEE_T_EXCOL）。 ..... 1801
52.5.58 用 Tx FIFO Underrun Statistic Register（ENET_IEEE_T_MACERR）传输的帧。 ..... 1802
52.5.59 用载波感应错误统计寄存器传输的帧（ENET_IEEE_T_CSERR）。 ..... 1802
52.5.60 保留统计寄存器（ENET_IEEE_T_SQE）。 ..... 1802
52.5.61 流量控制暂停帧传输统计寄存器（ENET_IEEE_T_FDXFC）。 ..... 1803
52.5.62 误差统计寄存器（ENET_IEEE_T_OCTETS_OK）中传输的帧的八位数。 ..... 1803
52.5.63 Rx数据包计数统计寄存器（ENET_RMON_R_PACKETS）。 ..... 1804
52.5.64 Rx广播包统计寄存器（ENET_RMON_R_BC_PKT）。 ..... 1804
52.5.65 Rx Multicast Packets Statistic Register (ENET_RMON_R_MC_PKT)。 ..... 1804
52.5.66 Rx数据包与CRC/Align错误统计寄存器（ENET_RMON_R_CRC_ALIGN）。 ..... 1805
52.5.67 小于64字节的Rx数据包和良好的CRC统计寄存器（ENET_RMON_R_UNDERSIZE）。 ..... 1805
52.5.68 Rx数据包大于MAX_FL和良好CRC统计寄存器（ENET_RMON_R_OVERSIZE）。 1806
52.5.69 小于64字节的Rx数据包和坏CRC统计寄存器（ENET_RMON_R_FRAG）。 ..... 1806
52.5.70 大于MAX_FL字节的Rx数据包和坏CRC统计寄存器（ENET_RMON_R_JAB）。 ..... 1806
52.5.71 保留统计寄存器（ENET_RMON_R_RESVD_0）。 ..... 1807
52.5.72 Rx 64字节数据包统计寄存器（ENET_RMON_R_P64）。 ..... 1807
52.5.73 Rx 65-127-字节数据包统计寄存器（ENET_RMON_R_P65TO127）。 ..... 1808
52.5.74 Rx 128-255字节数据包统计寄存器（ENET_RMON_R_P128TO255）。 ..... 1808
52.5.75 Rx 256- to 511-Byte Packets Statistic Register (ENET_RMON_R_P256TO511). ..... 1808
52.5.76 Rx 512- to 1023-Byte Packets Statistic Register (ENET_RMON_R_P512TO1023)。 ..... 1809
52.5.77 Rx 1024- to 2047-Byte 数据包统计寄存器 (ENET_RMON_R_P1024TO2047)。 ..... 1809
52.5.78 大于2048字节的Rx数据包统计寄存器（ENET_RMON_R_P_GTE2048）。 ..... 1810
52.5.79 Rx Octets统计寄存器（ENET_RMON_R_OCTETS）。 ..... 1810
52.5.80 帧未正确计数的统计寄存器（ENET_IEEE_R_DROP）。 ..... 1810
52.5.81 帧接收OK统计寄存器（ENET_IEEE_R_FRAME_OK）。 ..... 1811

52.5.82 CRC错误统计寄存器（ENET_IEEE_R_CRC）中收到的帧。 ..... 1811
52.5.83 收到的帧中有对齐错误统计寄存器（ENET_IEEE_R_ALIGN）。 ..... 1812
52.5.84 接收FIFO溢出计数统计寄存器（ENET_IEEE_R_MACERR）。 ..... 1812
52.5.85 流量控制暂停帧的接收统计寄存器（ENET_IEEE_R_FDXFC）。 ..... 1812
52.5.86 无错误接收的帧的八位数统计寄存器（ENET_IEEE_R_OCTETS_OK）。 ..... 1813
52.5.87 可调定时器控制寄存器（ENET_ATCR）。 ..... 1813
52.5.88 计时器值寄存器（ENET_ATVR）。 ..... 1815
52.5.89 定时器偏移寄存器（ENET_ATOFF）。 ..... 1816
52.5.90 计时器周期寄存器（ENET_ATPER）。 ..... 1816
52.5.91 计时器校正寄存器（ENET_ATCOR）。 ..... 1817
52.5.92 时间戳时钟周期寄存器（ENET_ATINC）。 ..... 1817
52.5.93 最后传输的帧的时间戳（ENET_ATSTMP）。 ..... 1818
52.5.94 计时器全局状态寄存器（ENET_TGSR）。 ..... 1818
52.5.95 计时器控制状态寄存器（ENET_TCSRn）。 ..... 1819
52.5.96 定时器比较捕获寄存器（ENET_TCCRn）。 ..... 1820
52.6 功能描述。 ..... 1821
52.6.1 以太网MAC帧格式。 ..... 1821
52.6.2 IP和更高层次的帧格式。 ..... 1824
52.6.3 IEEE 1588信息格式。 ..... 1828
52.6.4 MAC接收。 ..... 1832
52.6.5 MAC传输。 ..... 1838
52.6.6 全双工流量控制操作。 ..... 1842
52.6.7 神奇的数据包检测。 ..... 1844
52.6.8 IP加速器功能。 ..... 1845
52.6.9 复位和停止控制。 ..... 1849
52.6.10 IEEE 1588功能。 ..... 1852
52.6.11 FIFO阈值。 ..... 1856

52.6.12 回路选项。 ..... 1859
52.6.13 遗留的缓冲区描述符。 ..... 1860
52.6.14 增强的缓冲区描述符。 ..... 1861
52.6.15 客户端FIFO应用接口。 ..... 1867
52.6.16 FIFO保护。 ..... 1870
52.6.17 参考时钟。 ..... 1872
52.6.18 PHY管理接口。 ..... 1873
52.6.19 以太网接口。 ..... 1876

## 第53章 调试

53.1 介绍。 ..... 1881
53.2 CM4 ROM表。 ..... 1883
53.3 调试端口。 ..... 1884
53.3.1 JTAG-to-SWD的变化顺序。 ..... 1884
53.4 调试端口引脚描述。 ..... 1885
53.5 系统TAP连接。 ..... 1885
53.5.1 IR代码。 ..... 1885
53.6 JTAG状态和控制寄存器。 ..... 1886
53.6.1 MDM-AP控制寄存器。 ..... 1887
53.6.2 MDM-AP状态寄存器。 ..... 1888
53.7 调试复位。 ..... 1889
53.8 AHB-AP ..... 1890
53.9 ITM ..... 1890
53.10 核心跟踪连接。 ..... 1891
53.11 TPIU ..... 1891
53.12 DWT ..... 1892
53.13 在低功耗模式下进行调试。 ..... 1892
53.13.1 在低功耗模式下调试模块状态。 ..... 1893

53.14 调试和安全。 ..... 1893

## 第54章
## JTAG控制器（JTAGC）

54.1 芯片特定的JTAGC信息。 ..... 1895
54.2 介绍。 ..... 1895
54.2.1 框图。 ..... 1895
54.2.2 特点是。 ..... 1896
54.2.3 操作模式。 ..... 1896
54.3 外部信号描述。 ..... 1898
54.3.1 TCK-测试时钟输入。 ..... 1898
54.3.2 TDI-测试数据输入。 ..... 1898
54.3.3 TDO-测试数据输出。 ..... 1898
54.3.4 TMS-测试模式选择。 ..... 1899
54.4 登记描述。 ..... 1899
54.4.1 指示寄存器。 ..... 1899
54.4.2 绕行登记。 ..... 1899
54.4.3 设备识别寄存器。 ..... 1900
54.4.4 边界扫描寄存器。 ..... 1900
54.5 功能描述。 ..... 1901
54.5.1 JTAGC复位配置。 ..... 1901
54.5.2 IEEE 1149.1-2001（JTAG）测试接入端口。 ..... 1901
54.5.3 TAP控制器状态机。 ..... 1901
54.5.4 JTAGC块指令。 ..... 1904
54.5.5 边界扫描。 ..... 1907
54.6 初始化/应用信息。 ..... 1907

# 第一章
# 关于本手册

## 1.1 观众

本参考手册是为那些想用该设备开发产品的系统软件和硬件开发人员以及应用程序员准备的。它假定读者了解操作系统、微处理器系统设计，以及软件和硬件的基本原理。本手册介绍了S32K1xx系列的超集束器件的功能。对于可用的功能，特定的S32K1xx衍生产品（衍生设备）的注册实现，请参考各自的芯片-特定*模块*信息。

## 1.2 组织机构

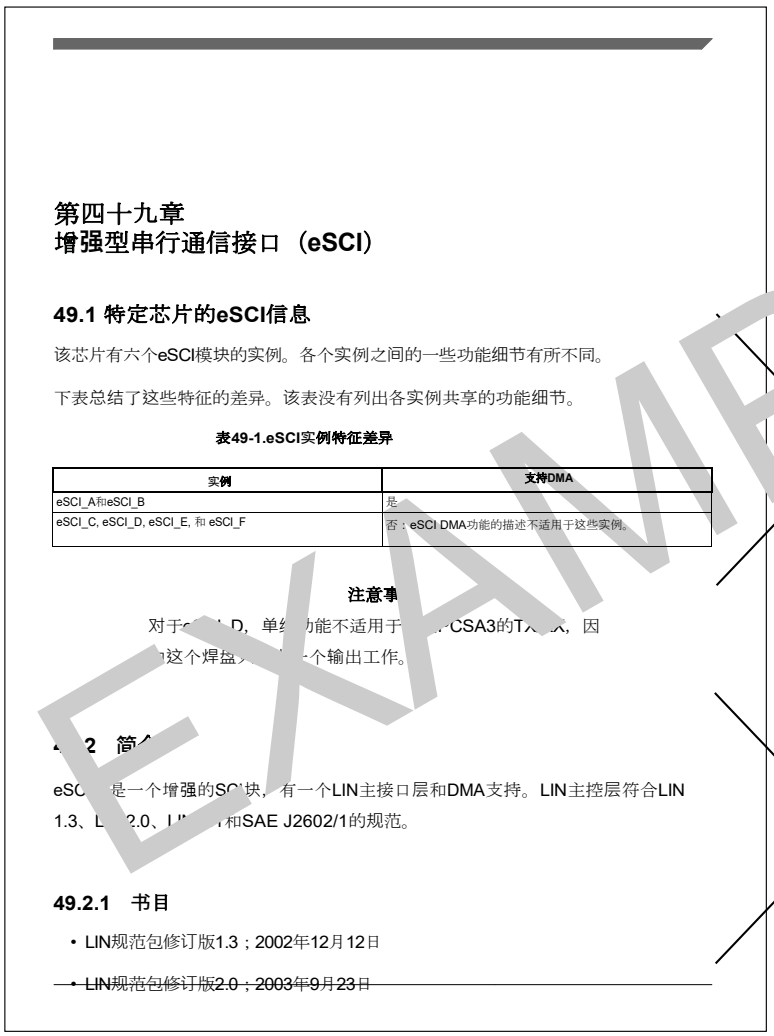
本手册有两套主要的章节。

1. 第一组中的章节包含适用于芯片上所有组件的信息。
2. 第二套书中的各章按功能分组，详细介绍了特定的功能领域。
    - 这些分组的例子是时钟、定时器和通信接口。
    - 每个分组都包括对各个模块进行技术描述的章节。

## 1.3 模块 说明

每个模块的章节有两个主要部分。

- 第一部分，*芯片特定的[模块名称]信息*，提供了一些细节，如芯片上的模块实例数量以及模块与其他模块的连接。*首先*阅读这一部分，因为它的内容对理解本章其他部分的信息至关重要。
- 接下来的章节提供了关于模块的一般信息，包括其信号、寄存器和功能描述。



第四十九章
增强型串行通信接口（eSCI）

49.1 特定芯片的eSCI信息

该芯片有六个eSCI模块的实例。各个实例之间的一些功能细节有所不同。

下表总结了这些特征的差异。该表没有列出各实例共享的功能细节。

表49-1.eSCI实例特征差异

| 实例 | 支持DMA |
|---|---|
| eSCI_A和eSCI_B | 是 |
| eSCI_C, eSCI_D, eSCI_E, 和 eSCI_F | 否；eSCI DMA功能的描述不适用于这些实例。 |

注意事

对于 D，单 功能不适用于 CSA3的T ，因
这个焊盘 个输出工作。

2 简

eSC 是一个增强的SC 块，有一个LIN主接口层和DMA支持。LIN主控层符合LIN 1.3、L 2.0、L 和SAE J2602/1的规范。

49.2.1 书目

- LIN规范包修订版1.3；2002年12月12日
- LIN规范包修订版2.0；2003年9月23日

EXAMPLE

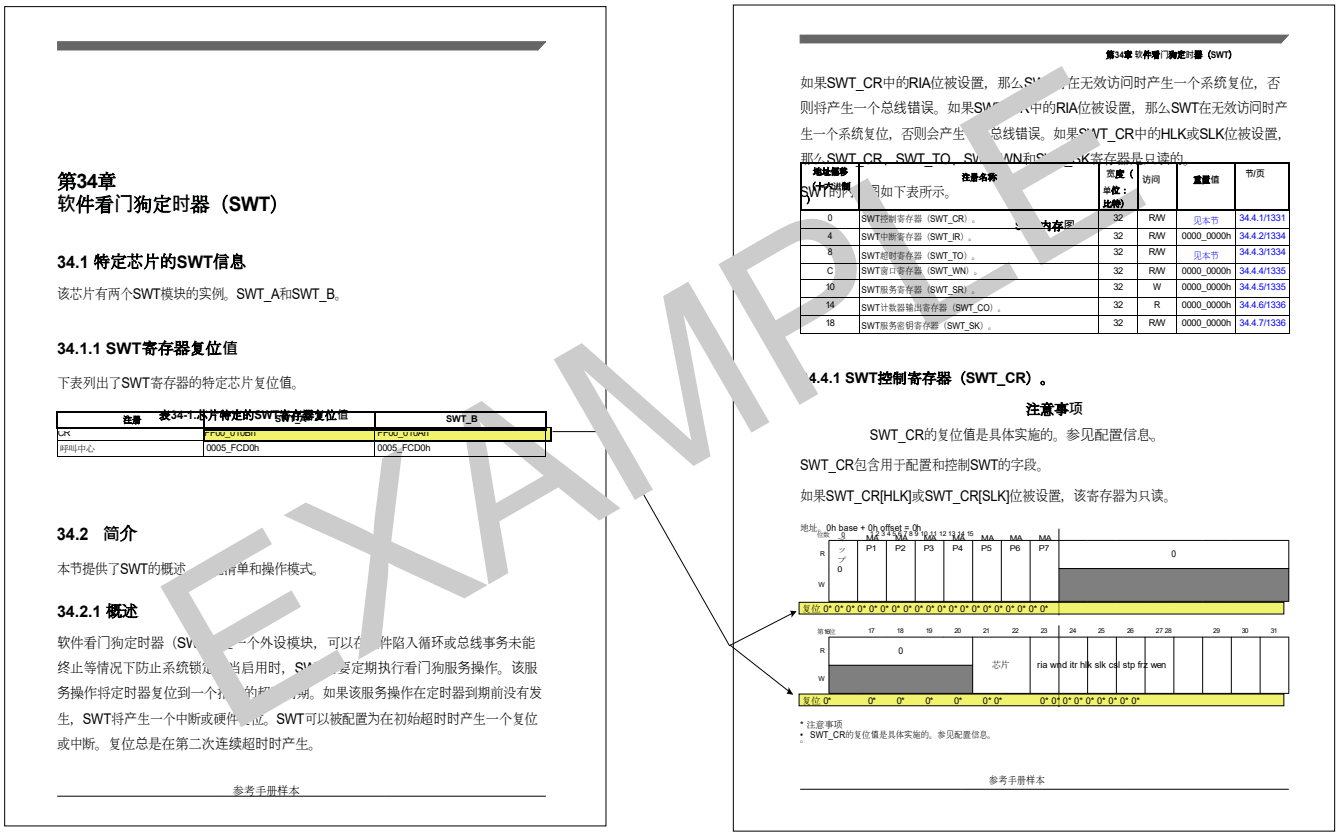
应首先阅读的芯片特定信息

一般模块信息的开始

**图1-1.例子：章节芯片特定信息和一般模块信息**

## 1.3.1 例如：澄清内容的芯片特定信息，在同一章中。

下面的例子显示了芯片的特定信息，澄清了本章后面介绍的一般模块信息。在这种情况下，芯片特定的寄存器复位值取代了出现在寄存器图中的复位值。



第34章
软件看门狗定时器（SWT）

34.1 特定芯片的SWT信息

该芯片有两个SWT模块的实例。SWT_A和SWT_B。

34.1.1 SWT寄存器复位值

下表列出了SWT寄存器的特定芯片复位值。

表34-1. 芯片特定的SWT寄存器复位值

| 注册 | | SWT_B |
|---|---|---|
| CR | FF00_0108h | FF00_0108h |
| 呼叫中心 | 0005_FCD0h | 0005_FCD0h |

34.2 简介

本节提供了SWT的概述，[illegible]清单和操作模式。

34.2.1 概述

软件看门狗定时器（SW[illegible]一个外设模块，可以在[illegible]件陷入循环或总线事务未能终止等情况下防止系统锁定[illegible]当启用时，SW[illegible]要定期执行看门狗服务操作。该服务操作将定时器复位到一个[illegible]的超[illegible]满。如果该服务操作在定时器到期前没有发生，SWT将产生一个中断或硬件复位。SWT可以被配置为在初始超时时产生一个复位或中断。复位总是在第二次连续超时时产生。

参考手册样本

第34章 软件看门狗定时器（SWT）

如果SWT_CR中的RIA位被设置，那么SW[illegible]在无效访问时产生一个系统复位，否则将产生一个总线错误。如果SW[illegible]R中的RIA位被设置，那么SWT在无效访问时产生一个系统复位，否则会产生[illegible]总线错误。如果SWT_CR中的HLK或SLK位被设置，那么SWT_CR，SWT_TO，SW[illegible]WN和SW[illegible]SK寄存器是只读的。

SWT的[illegible]列如下表所示。

| 地址偏移（从基地址） | 注册名称 | 宽度（单位：比特） | 访问 | 重置值 | 节/页 |
|---|---|---|---|---|---|
| 0 | SWT控制寄存器（SWT_CR）。 | 32 | R/W | 见本节 | 34.4.1/1331 |
| 4 | SWT中断寄存器（SWT_IR）。 | 32 | R/W | 0000_0000h | 34.4.2/1334 |
| 8 | SWT超时寄存器（SWT_TO）。 | 32 | R/W | 见本节 | 34.4.3/1334 |
| C | SWT窗口寄存器（SWT_WN）。 | 32 | R/W | 0000_0000h | 34.4.4/1335 |
| 10 | SWT服务寄存器（SWT_SR）。 | 32 | W | 0000_0000h | 34.4.5/1335 |
| 14 | SWT计数器输出寄存器（SWT_CO）。 | 32 | R | 0000_0000h | 34.4.6/1336 |
| 18 | SWT服务密钥寄存器（SWT_SK）。 | 32 | R/W | 0000_0000h | 34.4.7/1336 |

4.4.1 SWT控制寄存器（SWT_CR）。

注意事项

SWT_CR的复位值是具体实施的。参见配置信息。

SWT_CR包含用于配置和控制SWT的字段。

如果SWT_CR[HLK]或SWT_CR[SLK]位被设置，该寄存器为只读。

地址：0h base + 0h offset = 0h

| 比特 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | ソフト 0 | MA | MA | MA | MA | P1 | P2 | P3 | P4 | P5 | P6 | P7 | 0 | | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0* | 0* | 0* | 0* | 0* | 0* | 0* | 0* | 0* | 0* | 0* | 0* | 0* | 0* | 0* | 0* |

| 前领域 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | 芯片 | ria | wnd | itr | hlk | slk | csl | stp | frz | wen | |
| W | | | | | | | | | | | | | | | |
| 复位 | 0* | 0* | 0* | 0* | 0* | 0* | 0* | 0* | 0* | 0* | 0* | 0* | 0* | 0* | 0* |

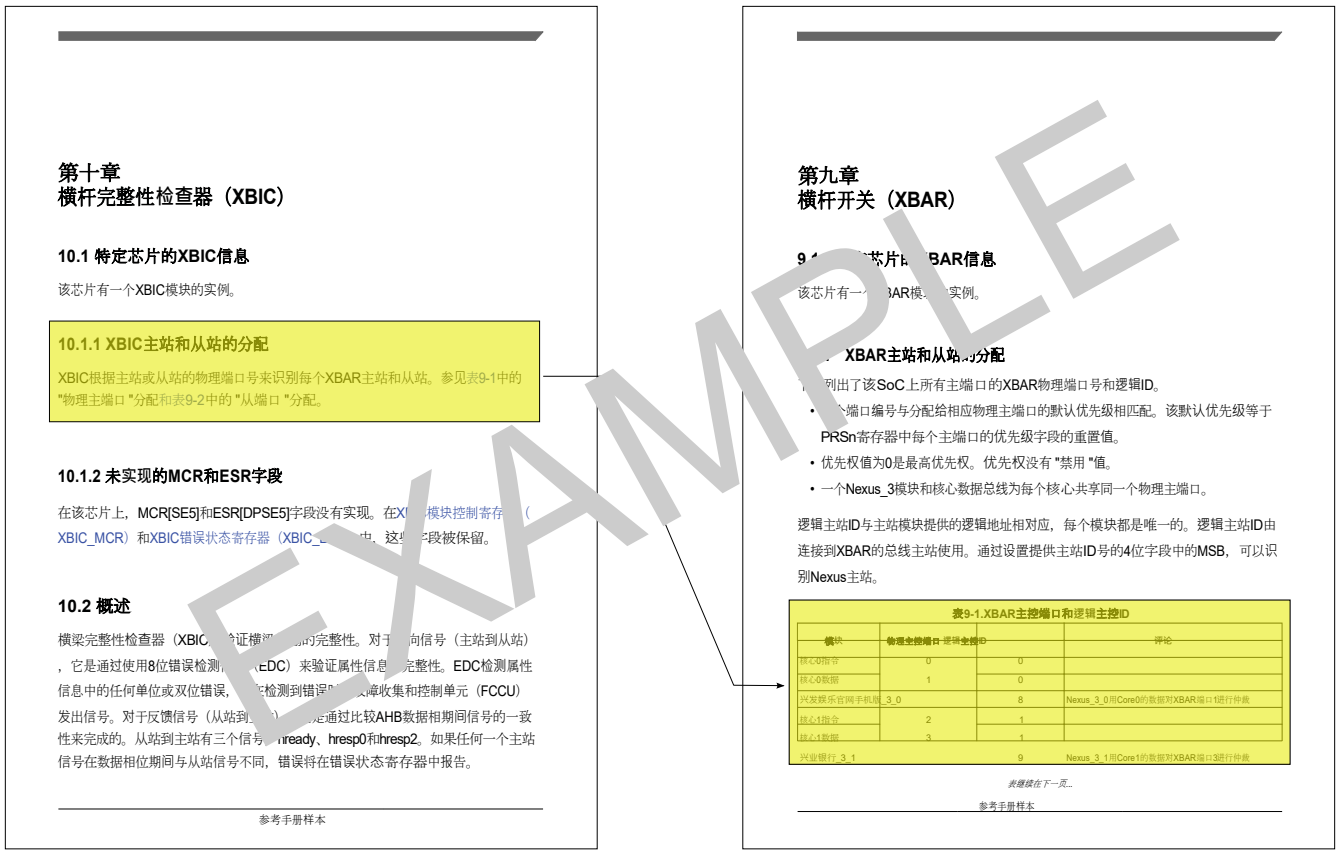
* 注意事项
• SWT_CR的复位值是具体实施的。参见配置信息。
。

参考手册样本

图1-2.例子：澄清同一章节内容的特定芯片信息

## 1.3.2 例如：芯片的特定信息，指的是不同的 章。

下面的芯片特定信息是指另一章的芯片特定信息。在这种情况下，在进一步阅读本章之前，请阅读这两套芯片专用信息。



第十章
横杆完整性检查器（XBIC）

10.1 特定芯片的XBIC信息

该芯片有一个XBIC模块的实例。

10.1.1 XBIC主站和从站的分配

XBIC根据主站或从站的物理端口号来识别每个XBAR主站和从站。参见表9-1中的"物理主端口"分配和表9-2中的"从端口"分配。

10.1.2 未实现的MCR和ESR字段

在该芯片上，MCR[SE5]和ESR[DPSE5]字段没有实现。在XBIC模块控制寄存器（XBIC_MCR）和XBIC错误状态寄存器（XBIC_ESR）中，这些字段被保留。

10.2 概述

横梁完整性检查器（XBIC）验证横梁互连的完整性。对于每个向信号（主站到从站），它是通过使用8位错误检测码（EDC）来验证属性信息的完整性。EDC检测属性信息中的任何单位或双位错误，并将检测到错误的情况报告给故障收集和控制单元（FCCU）发出信号。对于反馈信号（从站到主站），它是通过比较AHB数据相期间信号的一致性来完成的。从站到主站有三个信号：hready、hresp0和hresp2。如果任何一个主站信号在数据相位期间与从站信号不同，错误将在错误状态寄存器中报告。

参考手册样本

第九章
横杆开关（XBAR）

9.1 芯片的XBAR信息

该芯片有一个XBAR模块的实例。

9.1.1 XBAR主站和从站的分配

下表列出了该SoC上所有主端口的XBAR物理端口号和逻辑ID。

- 每个端口编号与分配给相应物理主端口的默认优先级相匹配。该默认优先级等于PRSn寄存器中每个主端口的优先级字段的重置值。
- 优先权值为0是最高优先权。优先权没有"禁用"值。
- 一个Nexus_3模块和核心数据总线为每个核心共享同一个物理主端口。

逻辑主站ID与主站模块提供的逻辑地址相对应，每个模块都是唯一的。逻辑主站ID由连接到XBAR的总线主站使用。通过设置提供主站ID号的4位字段中的MSB，可以识别Nexus主站。

表9-1.XBAR主控端口和逻辑主控ID

| 模块 | 物理主控端口 | 逻辑主控ID | 评论 |
|---|---|---|---|
| 核心0指令 | 0 | 0 | |
| 核心0数据 | 1 | 0 | |
| 兴发娱乐官网手机版_3_0 | | 8 | Nexus_3_0用Core0的数据对XBAR端口1进行仲裁 |
| 核心1指令 | 2 | 1 | |
| 核心1数据 | 3 | 1 | |
| 兴业银行_3_1 | | 9 | Nexus_3_1用Core1的数据对XBAR端口3进行仲裁 |

表继续在下一页...

参考手册样本

EXAMPLE

**图1-3.例子：指的是不同章节的芯片特定信息**

## 1.4 寄存器 描述

模块的各章介绍了寄存器的信息。

- 包含的记忆图谱。
  - 从模块的基址出发的一个偏移量
  - 每个登记册的名称和缩写/简称
  - 每个寄存器的宽度（单位：位）
  - 每个寄存器的复位值
  - 描述每个寄存器的页码
- 登记数字
- 字段描述表
- 相关文本

寄存器的数字采用下图中的惯例显示字段结构。

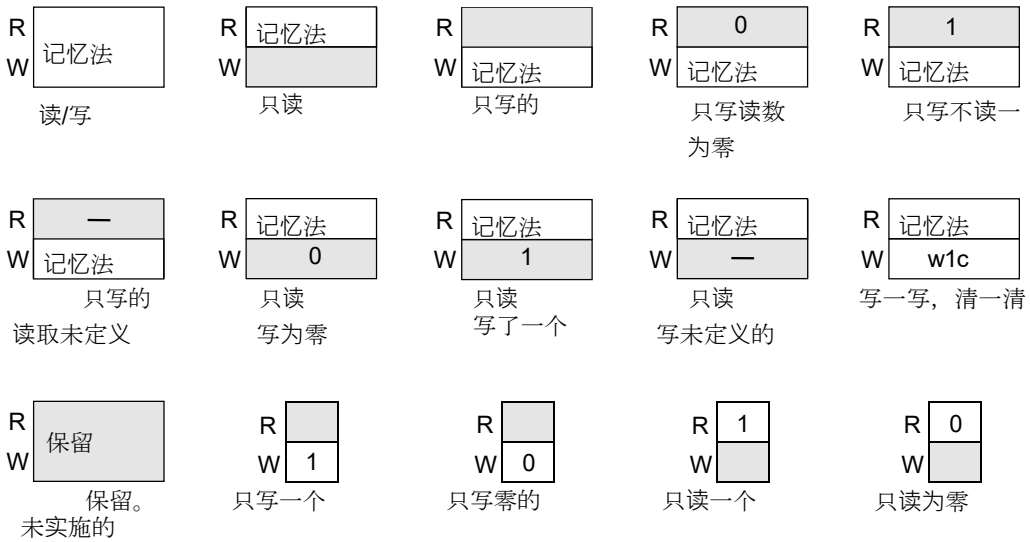



**图1-4.寄存器数字约定**

## 1.5 公约

### 1.5.1 注意事项、注意事项和 警告

本手册中到处都有 "注意"、"注意 "和 "警告
"提示。每种通知类型都提醒读者注意一种特定的信息。

**注意事项**

注释传达的信息可能与某一主题相关，或可能不适用于所有读者。

**注意事项**

注意事项通知指出了读者在采取进一步行动之前应该知道的信息。注意事项经常指出可能损坏芯片或电路板的故障点。

**警告**

警告性公告告知读者可能导致不必要后果的行为，特别是可能造成身体伤害的行为。

## 1.5.2 编号 系统

以下是识别不同编号系统的后缀。

| 这个后缀 | 确定了一个 |
|---|---|
| b | 二进制数。例如，数字5的二进制等价物被写成101b。在某些情况下，二进制数字以前缀*0b*表示。 |
| d | 十进制数字。只有当存在混淆的可能性时，小数才会在后面加上这个后缀。在一般情况下，小数的数字显示时没有后缀。 |
| h | 十六进制的数字。例如，数字60的十六进制相当于写成3Ch。 在某些情况下，十六进制数字用前缀*0x*表示。 |

## 1.5.3 排版 记号

本文件自始至终使用以下排版符号。

| 例子 | 描述 |
|---|---|
| *占位符*，x | 斜体字的项目是您提供的信息的占位符。斜体字也被用于出版物的标题和强调。普通的小写字母也被用来作为单个字母和数字的占位符。 |
| 编码 | 固定宽度类型表示必须完全按图索骥输入的文本。它用于指令助记符、指令、符号、子命令、参数和操作符。固定宽度类型也用于示例代码。文本和表格中的指令助记符和指令用大写字母表示，例如，BSR。 |
| SR[SCM] | 括号内的助记符代表寄存器中的一个命名字段。这个例子指的是状态寄存器（SR）中的缩放模式（SCM）字段。 |
| Revno[6:4], XAD[7:0] | 括号内的数字，用冒号隔开，代表其中之一。<br>• 一个寄存器的命名字段的子集<br>例如，REVNO[6:4]指的是占据REVNO寄存器6-0位的COREREV字段中的第6-4位。<br>• 一条总线的单个信号的连续范围<br>例如，XAD[7:0]指的是XAD总线的7-0信号。 |

## 1.5.4 特别是 条款

以下术语具有特殊含义。

| 术语 | 意义 |
|---|---|
| 宣称的 | 指的是一个信号的状态，如下所示。<br>• 当高电平（1）时，就会发出一个高电平信号。<br>• 当低电平（0）时为有效低电平信号。 |

| | |
|---|---|
| 去势 | 指的是一个信号的状态，如下所示。<br>• 一个高电平信号在低电平（0）时被取消。<br>• 高电平(1)时是一个有源低电平信号。<br><br>在某些情况下，去势的信号被描述为*否定的*。 |

*表继续在下一页...*

公约

| 术语 | 意义 |
| --- | --- |
| 保留 | 指的是一个内存空间、寄存器、字段或编程设置。当保留的位置被写入任何数值时，不保证设备操作。<br>• 不要修改保留的编程设置的默认值，如保留寄存器字段的复位值。<br>• 将内存中未定义的位置视为保留位置。 |
| w1c | 写1来清除：指的是必须写成1才能 "清除 "的寄存器位域。 |

# 第二章 简介

## 2.1 概述

S32K1xx产品系列进一步扩展了汽车行业中高度可扩展的ARM® Cortex®-M0+/M4F
MCU的组合。它建立在KEA系列的基础上，同时引入了更高的内存选项和更丰富的外设集，将能力扩展到各种汽车应用。S32K产品系列器件采用2.70-5.5V电源，注重汽车环境的稳健性，非常适用于电气恶劣环境下的各种应用，并为成本敏感型应用提供了低引脚数选择。S32K产品系列提供广泛的存储器、外设和封装选项。它共享共同的外设和引脚数，使开发人员能够在一个MCU系列内或在MCU系列之间轻松迁移，以利用更多的内存或功能集成。这种可扩展性使开发人员能够将S32K产品系列作为其终端产品平台的标准，最大限度地提高硬件和软件的重复使用，并缩短上市时间。

## 2.2 S32K1xx系列 介绍

### 2.2.1 S32K14x

S32K14x系列器件是基于ARM Cortex-M4F内核的32位通用汽车微控制器。它们提供卓越的性能、大内存和该类产品中最可扩展的外设。该产品系列提供高达112 MHz的CPU性能，支持DSP和FPU，拥有高达2 MB的闪存和高达256 KB的SRAM。器件特性概述。

- 32位ARM Cortex-M4F内核，带FPU，频率高达112MHz（HSRUN）和80MHz（正常RUN）。

- 高达2MB的代码闪存和64KB的FlexMem（支持高达4KB的仿真EEPROM与4KB FlexRAM）。
- 高达256 KB的SRAM，支持CPU专用访问和交叉条访问，提供指令和数据的并行访问
- 修改后的哈佛连接与本地内存控制器（LMEM），支持紧密耦合的RAM和4 KB代码缓存1
- 集成的时钟架构，带有片上快速IRC 48-60 MHz，慢速IRC 8 MHz / 2 MHz，128 KHz LPO和一个PLL单元
- 提供精密混合信号能力的模拟模块，包括12位至两个1 Msps的SAR ADC，高速比较器
- 电源管理控制器（PMC）具有内部稳压器，能够支持多种电源模式，包括。
  - 兴发娱乐官网手机版
  - 跑步
  - 停止
  - VLPR
  - VLPS
- 支持2.7 V至5.5 V电源的I/O
- 宽工作电压范围（2.7-5.5 V），具有全功能的闪存编程/擦除/读取操作
- 64 LQFP、100 LQFP、144 LQFP、176 LQFP和100 MAPBGA封装，最高可达156个GPIO引脚
- 环境工作温度范围从-40 °C到125 °C

S32K14x MCU组合得到了一套非常全面的开发工具和软件的支持。该启用包包括。恩智浦Arduino兼容评估板、S32K软件开发工具包（SDK），包括图形配置能力和S32 Design Studio软件，以及IAR Systems、Cosmic Software、Green Hills和其他合作伙伴的广泛支持。

### 2.2.2 S32K11x

S32K11x系列器件是基于ARM Cortex-M0+内核的32位通用汽车微控制器。它们提供了卓越的性能、大内存和该类产品中最可扩展的外设。这个产品系列提供48MHz的CPU性能，有高达256KB的Flash和高达25KB的SRAM。器件特性概述。

- 32位ARM Cortex-M0+内核，48MHz的CPU

1. 这是指可由ARM CM4代码总线寻址的区域，用于缓存该区域的代码以及数据。参见S32K1xx_memory_map.xlsx以了解不同区域的缓存能力的更多细节。

- 高达256KB的代码闪存和32KB的FlexMem（支持高达2KB的仿真EEPROM与2KB FlexRAM）。
- 高达25KB的SRAM
- 具有片上快速IRC 48MHz、慢速IRC 8MHz和128KHz LPO的集成时钟架构
- 提供精密混合信号能力的模拟模块，包括12位1 Msps SAR ADC、高速比较器
- 电源管理控制器（PMC）具有内部稳压器，能够支持多种电源模式，包括。
  - 跑步
  - 停止
  - VLPR
  - VLPS
- 支持2.7V至5.5V电源的I/O
- 宽工作电压范围（2.7-5.5 V）， 具有全功能的闪存编程/擦除/读取操作
- 32-pin QFN, 48-pinLQFP, 64-pinLQFP， 最多有58个GPIO引脚
- 环境工作温度范围从-40 °C到125 °C

S32K11x
MCU组合得到了一套非常全面的开发工具和软件的支持。该启用包包括。恩智浦Arduino兼容评估板、S32K软件开发工具包（SDK）， 包括图形配置能力和S32 Design Studio软件，以及IAR Systems、Cosmic Software、Green Hills和其他合作伙伴的广泛支持。

## 2.3 专题 摘要

下表列出了集成在S32K1xx产品系列上的功能。

**表2-1.器件功能概要**

| 特点 | S32K14x产品系列 | S32K11x产品系列 |
|---|---|---|
| 硬件特征 | | |
| 包装 | 64针LQFP，10*10毫米，0.5毫米间距<br>100针LQFP，14*14毫米，0.5毫米间距<br>144针LQFP，20*20毫米，0.5毫米间距<br>176针LQFP，24*24毫米，0.5毫米间距<br>100引脚MAPBGA，11*11毫米，1毫米球距 | 32-pin QFN<br>48-pin LQFP<br>64-pin LQFP |
| 电压范围 | 2.7 V至5.5 V | |
| 温度范围 ($T_A$) | -40℃至125℃1 | |

*表继续在下一页...*

| 特点 | S32K14x产品系列 | S32K11x产品系列 |
|---|---|---|
| 温度范围 (TJ ) | -40°C至135°C | |
| 系统 | | |
| 中央处理单元（CPU） | ARM Cortex-M4F | ARM Cortex-M0+ |
| 最大的CPU频率 | 112兆赫 | 48兆赫 |
| 数字信号处理器（DSP） | 是 | 没有 |
| 浮点单元（FPU） | 是 | 没有 |
| 系统内存保护单元（MPU）[2] | 是 | |
| 代码缓存大小 | 4 KB | 不详 |
| 嵌套的矢量中断控制器（NVIC） | 多达240个矢量中断<br>16个可编程的中断优先级别 | TBD |
| 唤醒中断控制器（WIC） | 是 | |
| 直接内存访问（DMA） | 16个通道 | 4个通道 |
| 直接内存访问多路复用器（DMAMUX）。 | 是 | |
| 无任务中断（NMI）。 | 是 | |
| 软件监督员 | 是 | |
| 硬件看门狗 | 是的，有外部显示器引脚。 | 不详 |
| 调试 | 2个针脚的串行线调试（SWD） | |
| | IEEE 1149.1联合测试接入组（JTAG）。 | 仅适用于BSR |
| 追踪 | 跟踪端口接口单元（TPIU） | 山地车 |
| 边界扫描 | 是 | |
| 独特的识别（ID）号码 | 128位宽 | |
| 记忆 | | |
| 对闪存进行编程 | 高达2MB | 最多 256 KB |
| 弹性内存（FlexMemory | 64 KB<br>数据闪存（D-flash）/仿真EEPROM备份（E-Flash）存储器。4KB额外的FlexRAM支持高端保险，非易失性的仿真EEPROM | 32 KB<br>数据闪存（D-flash）/仿真EEPROM备份（E-Flash）存储器：2KB额外的FlexRAM，支持高端耐用、非易失性的仿真EEPROM |
| 闪存控制器高速缓存 | 是的（仅限单一投机性预取缓冲区）。 | |
| 闪存访问控制（FAC） | 没有 | |
| 随机存取存储器（RAM） | 最高可达256 KB。4 KB是EEERAM解决方案的一部分 | 最高可达25KB。2 KB是EEERAM解决方案的一部分 |
| 低泄漏的待机存储器 | 在所有模式下保留RAM | |
| QuadSPI | 支持SDR和HyperRAM模式，分别多达4条和8条双向数据线 | 无法获得 |

表2-1.器件功能概要（续）

| | | |
|---|---|---|
| 循环冗余检查(CRC) | 16位或32位CRC，具有可编程的生成器多项式 | |
| 钟表 | | |
| 系统时钟发生器（SCG） | 奥斯陆、芬兰、西班牙、菲律宾 | 奥斯陆、芬兰、西班牙 |

*表继续在下一页...*

## 表2-1.器件功能概要（续）

| 特点 | S32K14x产品系列 | S32K11x产品系列 |
|---|---|---|
| 外部晶体振荡器或谐振器 | OSC：4 - 40 MHz，低功率或全摆幅 | |
| 外部方波输入时钟 | 直流至50兆赫 | |
| 内部时钟参考（IRC） | 48MHz内部IRC（FIRC），全温度下最大偏差为1%。<br>8MHz内部IRC（SIRC），全温度下最大偏差为3%。 | |
| 锁相环（PLL） | 高达320兆赫的VCO | 不详 |
| 人机界面（HMI） | | |
| 通用输入/输出（GPIO） | 多达156个GPIO | 多达58个GPIO |
| | 引脚中断/DMA请求能力 | |
| | 可配置的数字突变滤波器 | |
| | 所有输入引脚的滞后、上拉和下拉控制 | |
| | NMI_b和RESET_B输入引脚的无源输入滤波器 | |
| | 多达32个引脚的高驱动能力 | |
| 模拟 | | |
| 电源管理控制器（PMC） | 低电压警告<br>内部稳压器提供各种功率模式 128 kHz LPO时钟 | |
| 12位模数转换器 | 12位模式下为1 Msps | |
| | 1.2 Msps，10位模式 | |
| | 多达64个单端外部通道 | 多达32个单端外部通道 |
| | 多达64个控制和结果寄存器 | 多达32个控制和结果寄存器 |
| | 支持结果比较 | |
| 高速比较器（CMP） | 自带8位数模转换器的比较器 | |
| 计时器 | | |
| 可编程延迟块（PDB） | PDB0有多达4个ADC通道，每个通道有8个ADC0的预触发器<br>1个脉冲输出通道 | |
| | PDB1有多达2个ADC通道，每个通道有8个预触发器，用于ADC1<br>1个脉冲输出通道 | 不详 |
| 灵活的定时器（FTM） | 16位，64个通道 | 16位，16个通道 |
| | GTB和Global Load | |
| | 多达2个，带正交解码器 | |
| | 死亡时间》作者是一对 | |
| | 启用颤振功能时最多可有2个 | |
| 32位低功耗可编程中断定时器（LPIT）。 | 4个独立通道 | |
| 实时时钟（RTC） | 是 | |

表2-1.器件功能概要（续）

| | |
|---|---|
| 低功耗定时器(LPTMR) | LPTMR<br>1个通道，16位脉冲计数器或周期性中断 |
| 通信接口 | |

*表继续在下一页...*

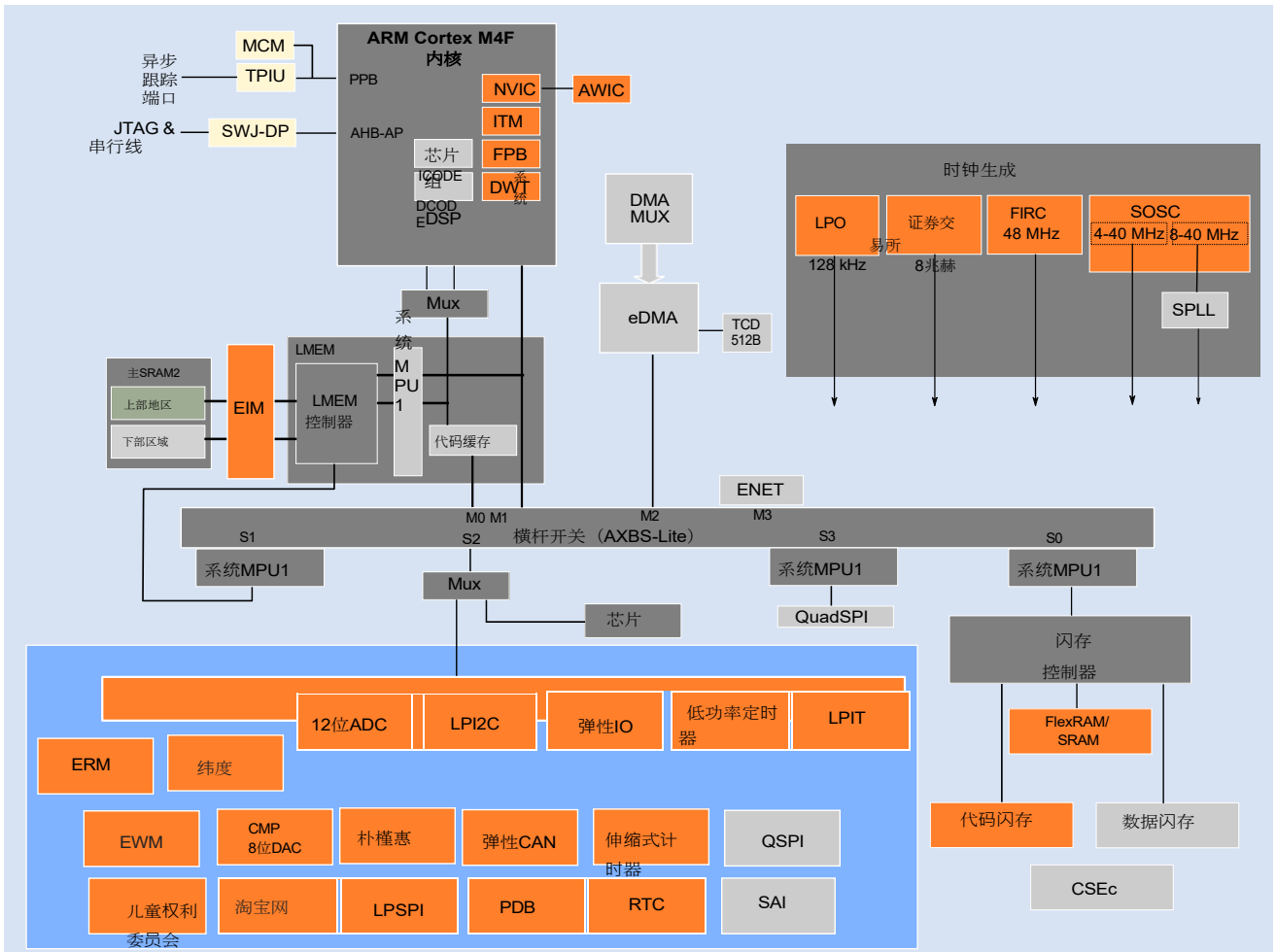
表2-1.器件功能概要（续）

| 特点 | S32K14x产品系列 | S32K11x产品系列 |
|---|---|---|
| 以太网控制器 | 双速10/100-Mbit/s以太网MAC，符合IEEE802.3-2002标准。<br>与半双工或全双工10/100-Mbit/s以太网LAN兼容 | 不详 |
| 控制区网络（CAN） | 具有可选的ISO CAN-FD支持 | 具有可选的ISO CAN-FD支持 |
| 弹性IO | 能够支持广泛的协议（UART、I2C、SPI、I2S）和PWM/波形生成 | |
| 低功耗串行外设接口（LPSPI）。 | 支持DMA，在所有LPSPIs上支持4字FIFO | |
| 低功耗互连电路（LPI2C）。 | 在LPI2C0上支持标准的SMBUS<br>兼容I2C 4字FIFO DMA支持<br>1 Mbps的能力（即使最大I2C总线负载为400 pF）。<br>只有高驱动引脚能够支持1 Mbps的速度 | |
| 低功耗 UART (LPUART) | 支持LIN协议1.3、2.0、2.1版本，以及SAE J2602标准功能<br>可配置4倍至32倍的超采样，支持LIN从机操作<br>32位数据宽度<br>所有LPUART都支持DMA和4字FIFO | |
| SAI | 支持具有帧同步的全双工串行接口，如I2S、AC97、TDM和编解码/DSP接口。 | 不详 |

1. HSRUN模式的最大环境温度限制为105°C $T_A$
2. 在该器件上，恩智浦的系统MPU实现了安全机制，以防止主控器访问受限的内存区域。该系统MPU在Crossbar开关的层面上提供内存保护。每个Crossbar主站（Core、DMA、Ethernet）可以被分配到每个受保护的内存区域的不同访问权限。该系列中的ARM
   M4内核版本没有集成ARM内核MPU，它将同时监控只有内核发起的内存访问。在本文件中，术语MPU指的是恩智浦的系统MPU。

## 2.4 块 图

下图是S32K14x产品系列的框图。




1: 在该器件上，恩智浦的系统MPU实现了安全机制，以防止主控器访问受限的内存区域。该系统MPU在Crossbar开关的层面上提供内存保护。每个Crossbar主站（Core、DMA、Ethernet）都可以被分配到
对每个受保护的内存区域有不同的访问权限。该系列中的ARM M4内核版本没有集成ARM Core MPU，后者将同时监测仅由内核发起的内存访问。在本文件中，术语MPU指的是恩智浦的系统MPU。

2: 关于器件的具体尺寸，请参见《S32K14x系列参考手册》中 "存储器和存储器接口 "一章的 "片上SRAM尺寸 "表。

钥匙。

**图2-1.S32K14x产品系列框图**

下图是S32K11x产品系列的框图。

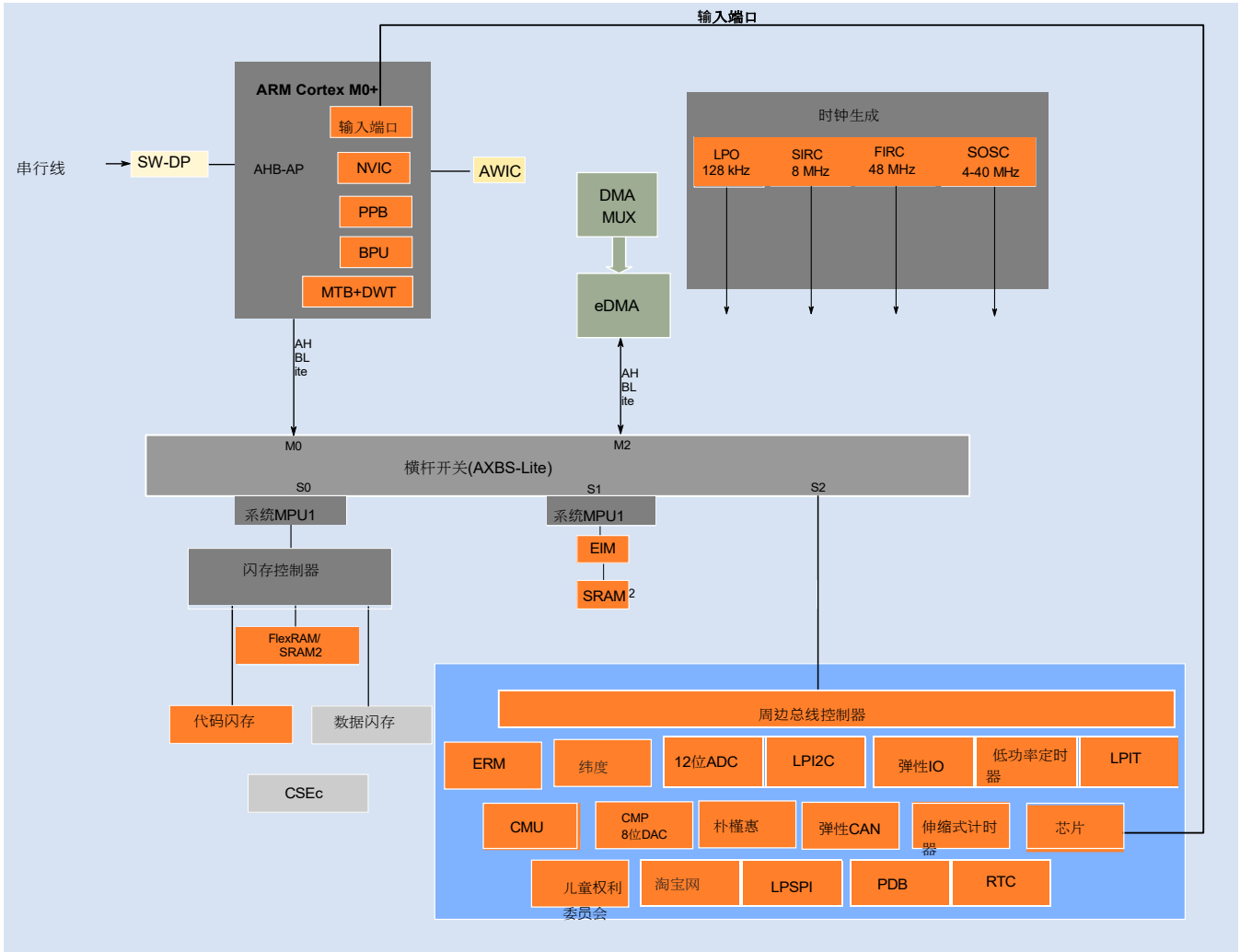



**图2-2.S32K11x产品系列框图**

## 2.5 特征 比较

下图总结了S32K产品系列的存储器和封装选项，并说明了该器件在整个系列中的位置。所有共用封装的器件都是针脚兼容的。

| | 参数 | S32K11x | | S32K14x | | | |
|---|---|---|---|---|---|---|---|
| | | K116 | K118 | K142 | K144 | K146 | K148 |
| | 核心 | ARM® Cortex™-M0+ | | ARM® Cortex™-M4F | | | |
| | 频率 | 48兆赫 | | 高达112MHz（HSRUN）。 | | | |
| 系统 | IEEE-754 FPU | o | | ● | | | |
| | HW安全模块（CSEc）[1] | ● | | ● | | | |
| | CRC模块 | 1x | | 1x | | | |
| | ISO 26262 | 能够达到ASIL-B | | 能够达到ASIL-B | | | |
| | 周边速度 | 高达48兆赫 | | 高达112MHz（HSRUN）。 | | | |
| | 横杆 | ● | | ● | | | |
| | DMA | o | | ● | | | |
| | EWM | ● | | ● | | | |
| | 存储器保护单元 | ● | | o | | | |
| | FIRC CMU | | | | | | |
| | 看门狗 | 1x | | 1x | | | |
| | 低功率模式 | ● | | ● | | | |
| | HSRUN模式 | o | | ● | | | |
| | I/O的数量 | 最高可达43 | 最高可达58 | 高达89 | | 最多可达128 | 最高可达156 |
| | 单一电源电压 | 2.7 - 5.5 V | | 2.7 - 5.5 V | | | |
| | 工作温度（Ta）环境温度 | -40 to +85ºC / +105ºC / +125ºC | | -40 to +85ºC / +105ºC / +125ºC | | | |
| 记忆 | 闪光灯 | 128 KB | 256 KB | 256 KB | 512 KB | 1 MB | 2 MB2 |
| | 纠错代码（ECC） | ● | | ● | | | |
| | 系统RAM（包括FlexRAM和MTB） | 17 KB | 25 KB | 32 KB | 64 KB | 128 KB | 256 KB |
| | FlexRAM（也可作为系统RAM使用） | 2 KB | | 4 KB | | | |
| | 缓存 | o | | 4 KB | | | |
| | 由FlexRAM1仿真的EEPROM | 2 KB (最多32 KB D-Flash) | | 4 KB（最多64 KB的D-Flash）。 | | | 4 KB（作为2MB闪存的一部分，最多512 KB D-Flash）[3] |
| | 外部存储器接口 | | | | | | QuadSPI包括HyperBus™。 |
| 计时器 | 低功耗中断定时器 | 1x | | 1x | | | |
| | FlexTimer（16位计数器）8通道 | 2x (16) | | 4x (32) | | 6x (48) | 8x (64) |
| | 低功率定时器（LPTMR） | 1x | | 1x | | | |
| | 实时计数器（RTC） | 1x | | 1x | | | |
| | 可编程延迟块（PDB） | 1x | | o | | 2x | |
| 模拟 | 触发器多路复用器（TRGMUX）。 | 1x (43) | 1x (45) | 1x (64) | | 1x (73) | 1x (81) |
| | 12位SAR ADC（每个1 MSPS）。 | 1x (14) | 1x (16) | 2x (16) | | 2x (24) | 2x (32) |
| | 带8位DAC的比较器 | 1x | | 1x | | | |
| 沟通 | 100 Mbit IEEE-1588以太网MAC | | | | | | 1x |
| | 串行音频接口（AC97、TDM、I2S）。 | | | | | | 2x |
| | 低功耗 UART/LIN (支持LIN协议1.3、2.0、2.1和SAE J2602版本。) | 2x | | 2x | 3x | | |
| | 低功耗SPI | 1x | 2x | 2x | 3x | | |
| | 低功耗I2C | 1x | | 1x | | | 2x |
| | 弹性CAN (can-fd iso/cd 11898-1) | 1x (1个有FD) | | 2x (1个有FD) | 3x (1个有FD) | 3x (2倍于FD) | 3x (3倍于FD) |
| | FlexIO（8个引脚可配置为UART、SPI、I2C、I2S）。 | 1x | | 1x | | | |
| IDEs | 调试和跟踪 | swd, mtb (1 kb), jtag4 | | SWD, JTAG (ITM, SWV, SWO) | | | SWD，JTAG（ITM，SWV，SWO），ETM |
| | 生态系统 (IDE、编译器、调试器) | 恩智浦S32设计工作室（GCC）+SDK, IAR，GHS，COSMIC，劳特巴赫，iSystems | | 恩智浦S32设计工作室（GCC）+SDK, IAR，GHS，COSMIC，劳特巴赫，iSystems | | | |
| 其他 | 节目包 | QFN-32 LQFP-48 | lqfp-48 lqfp-64 | lqfp-64 lqfp-100 | lqfp-64 lqfp-100 mapbga-100 | LQFP-64 mapbga-100 lqfp-100 lqfp-144 | mapbga-100 lqfp-144 lqfp-176 |

传说。

o 未实施 在设备上可用

● 1 当芯片处于VLPR或HSRUN模式时，没有FTFC命令，包括CSE命令（CSEc部分）可用。

2 当不使用EEEPROM、CSEc和数据闪存时可用。否则，只有高达1,984 KB的程序闪存可用。

3 多达64KB的闪存被用作EEPROM的备份，最后512KB块的剩余448KB可以用作数据闪存或程序闪存。详见FTFC一章。

4 仅适用于BSR

**图2-3.S32K1xx产品系列比较**

## 2.6 应用

S32K1xx产品系列是通用汽车应用的理想选择，这些应用包括但不限于：。

- 外部和内部照明
- 暖通空调
- 门/窗/雨刷/座椅控制器
- BLDC/PMSM电机控制
- 公园助理
- E-shifter
- TPMS
- 信息娱乐系统的实时控制
- 电池管理系统
- 人机界面，如触摸感应控制
- 安全的车辆数据传输
- 安全控制器
- 空中更新[2]

此外，S32K148上的100 Mbit IEEE-1588以太网MAC和串行音频接口（AC97、TDM和I2S）使其完美地适用于车载以太网连接的边缘节点，以及音频流应用。

S32K116和S32K118提供小型封装。5*5 mm 32 QFN、7*7 mm 48 LQFP和10*10 mm 64 LQFP，适用于对尺寸敏感的广泛的车辆应用，如各种传感器控制器。

除汽车应用外，S32K1xx产品系列还可用于工业、自动化、通信、运输、医疗和研发等需要高水平质量、可靠性和安全性的挑战性环境中。

**注意事项**

对于安全意识和关键应用，用户必须参考S32K14x安全手册和使用S32K1xx EEPROM功能的应用说明，可在http://www.nxp.com/，了解正确的零件操作要求。

## 2.7 模块功能 类别

该设备上的模块被分成了几个功能类别。以下各节更详细地描述了分配给每个类别的模块。

2. 参见S32K架构和能力，以实现空中更新。

## 表2-2.模块功能类别

| 模块类别 | 描述 |
| --- | --- |
| ARM® Cortex®-M4F内核 | • 来自ARM的Cortex-M类的32位MCU内核，增加了DSP指令和基于ARMv7架构的单精度浮点单元 |
| ARM® Cortex®-M0+内核 | • 来自ARM的Cortex-M类的32位MCU内核，实现了ARMv6-M架构配置文件。 |
| 系统 | • 杂项控制模块（MCM）<br>• 系统集成模块（SIM）。<br>• 端口控制和中断（PORT）。<br>• 通用输入/输出控制器（GPIO）<br>• 横杆开关(AXBS-Lite)<br>• 周边桥接（AIPS-Lite)<br>• 触发器多路复用控制（TRGMUX）。<br>• 看门狗定时器（WDOG）<br>• 循环冗余检查（CRC） |
| 计时 | • 可从内部和外部生成的时钟中选择多种时钟生成方式<br>• 系统振荡器，为MCU提供时钟源 |
| 记忆 | • 内部存储器包括。<br>  • 对闪存进行编程<br>  • 弹性内存（FlexMemory<br>  • 弹性虚拟机（FlexNVM<br>  • 弹性RAM<br>  • SRAM<br>• 直接内存访问（DMA）控制器带有多路复用器，以增加可用的DMA请求。DMA现在可以处理VLPS模式下的传输 |
| 电源管理 | 电源管理和模式控制器（PMC）<br>• 基于高速运行、运行、停止的多种动力模式可供选择 |
| 安全问题 | • 闪存和SRAM存储器上的纠错代码（ECC）<br>• 128位独特的识别（ID）号码<br>• 系统内存保护单元（MPU）模块 |
| 安全问题 | 时钟监测单元（CMU），用于监测FIRC时钟 |
| 电源管理 | 电源管理和模式控制器（PMC）<br>• 基于高速运行、运行、停止的多种动力模式可供选择 |
| 模拟 | • 高速模数转换器（ADC）<br>• 比较器（CMP），包含8位参考DAC<br>• 带隙电压参考（1V参考电压） |
| 计时器 | • 可编程延迟块（PDB）<br>• 灵活性计时员<br>• 低功耗定期中断定时器（LPIT）。<br>• 低功率定时器（LPTMR） |

| 模块功能类别 | |
|---|---|
| | • 独立的实时时钟（RTC） |
| 通讯 | • 符合IEEE802.3-2002标准的以太网MAC。<br>• 低功耗串行外设接口（LPSPI）。<br>• 低功耗集成电路（LPI2C）。<br>• 低功耗 UART (LPUART)<br>• 同步音频接口（SAI）/集成芯片间的声音（I2S）。<br>• 弹性IO<br>• 弹性CAN |
| 调试 | • JTAG控制器（JTAGC） |

## 2.7.1 ARM® Cortex®-M4F核心 模块

该设备上有以下核心模块。

**表2-3.核心模块**

| 模块 | 描述 |
| --- | --- |
| ARM Cortex-M4F | ARM® Cortex®-M4F 是 Cortex M 系列处理器的最新成员，其目标是专注于对成本非常敏感、确定性、中断驱动环境的微控制器内核。Cortex M4F处理器基于ARMv7架构和Thumb®-2 ISA，与Cortex M3、Cortex M1和Cortex M0架构向上兼容。Cortex M4F的改进包括一个ARMv7 Thumb-2 DSP（从ARMv7-A/R配置文件架构中移植），提供32位指令，具有SIMD（单指令多数据）DSP风格的乘法-累加和饱和运算。 |
| 浮点单元（FPU） | 一个符合*IEEE浮点运算标准*（IEEE 754）的单精度浮点运算单元（FPU）。 |
| NVIC | ARMv7-M异常模型和嵌套式中断控制器（NVIC）实现了一个可重定位的向量表，支持许多外部中断、单个不可屏蔽的中断（NMI）和优先级。<br>NVIC取代了影子寄存器，具有同等的系统和简化的可编程性。NVIC包含了一个特定处理程序要执行的函数的地址。该地址通过指令端口获取，允许并行寄存器堆叠和查找。前16个条目被分配给ARM内部源，其他条目被映射到MCU定义的中断。 |
| AWIC | 异步唤醒中断控制器（AWIC）的主要功能是检测停止模式下的异步唤醒事件，然后向时钟控制逻辑发出信号以恢复系统时钟。时钟重启后，NVIC观察待处理的中断并进行正常的中断或事件处理。 |
| 调试接口 | 该设备的大部分调试功能都是基于ARM CoreSight™架构。支持以下调试接口。<br>• 串行线IEEE 1149.1 JTAG调试端口（SWJ-DP），有2针串行线调试（SWD）用于外部调试器<br>• 调试观察点和跟踪（DWT），有四个可配置的比较器作为硬件观察点<br>• 串行线输出（SWO）--支持同步跟踪数据<br>• 具有软件和硬件追踪功能的仪器仪表追踪巨室（ITM），加上时间标记<br>• 闪存补丁和断点（FPB），能够将代码和数据从代码空间修补到系统空间<br>• 串行线浏览器（SWV）。追踪功能，提供读、写、异常、PC Samples和printf的显示。<br>• 支持4针追踪接口 |

## 2.7.2 ARM Cortex-M0+内核 模块

该设备上有以下核心模块。

表2-4.核心模块

| 模块 | 描述 |
| --- | --- |
| ARM Cortex-M0+ | ARM® Cortex™-M0+处理器是一个可配置的、多级的32位RISC处理器。它有一个AMBA AHB-Lite接口，包括一个NVIC组件。它还具有可选的硬件调试、单周期I/O接口和内存保护功能。该处理器可以执行Thumb代码，并与其他Cortex-M配置文件处理器兼容。 |
| NVIC | NVIC具有多达32个外部中断输入，每个中断都有四级优先级。它实现了向量表的可选重定位。<br>它具有专用的非屏蔽中断（NMI）输入，支持可选的唤醒中断控制器（WIC），提供超低功率睡眠模式支持。 |
| AWIC | 异步唤醒中断控制器（AWIC）的主要功能是检测停止模式下的异步唤醒事件，并向时钟控制逻辑发出信号以恢复系统时钟。时钟重启后，NVIC观察待处理的中断并进行正常的中断或事件处理。 |
| 单周期I/O端口 | 为了高速、单周期地访问外设，Cortex-M0+处理器实现了一个专用的单周期I/O端口。 |
| 调试接口 | 该设备的大部分调试是基于ARM CoreSight™架构的。支持一个调试接口。<br>• 串行线调试（SWD）<br>• 微量缓冲液（MTB） |

## 2.7.3 系统 模块

该设备上有以下系统模块。

表2-5.系统模块

| 模块 | 描述 |
| --- | --- |
| 杂项控制模块（MCM） | MCM包括核心和系统模块的杂项控制逻辑。 |
| 系统集成模块（SIM）。 | SIM包括各种设备配置和状态寄存器。 |
| 港口 | 端口控制和中断（PORT）模块提供对端口控制、数字滤波和外部中断功能的支持。 |
| 芯片 | 通用输入和输出（GPIO）模块通过零等待状态接口与处理器内核进行通信，以实现最大的引脚性能。GPIO寄存器支持8位、16位或32位访问。 |
| 横杆开关(AXBS-Lite) | AXBS-Lite连接了总线主站和总线从站，允许所有总线主站同时访问不同的总线从站，并在总线主站访问同一从站时提供仲裁。 |
| 系统内存保护单元（MPU） | 系统MPU提供内存保护和任务隔离。它同时监测从属连接的所有总线主事务。 |
| 周边桥接（AIPS-Lite) | AIPS-Lite将横梁开关接口转换为允许访问设备上大多数外设的接口。 |

| 模块功能类别 | |
|---|---|
| 直接内存访问复用器（DMAMUX）。 | DMAMUX从许多DMA请求中选择下来，为DMA控制器选择一个较小的数字。 |

*表继续在下一页...*

表2-5.系统模块（续）

| 模块 | 描述 |
| --- | --- |
| 增强型直接内存访问控制器（eDMA） | eDMA提供带有传输控制描述符的可编程通道，通过双地址传输进行数据移动，用于8位、16位、32位、16字节和32字节的数据值。 |
| 淘宝网 | 触发器多路复用器（TRGMUX）模块允许软件为各种外设配置触发源。 |
| 外部看门狗监视器（EWM） | EWM是软件看门狗模块的一个冗余机制，监测内部和外部系统运行的故障状况。 |
| 软件看门狗（WDOG）。 | WDOG监测内部系统的运行，并在发生故障时强制复位。它可以从一个独立的128kHz低功率振荡器上运行，有一个可编程的刷新窗口，以检测程序流或系统频率的偏差。 |
| 儿童权利委员会 | 使用16/32位移位寄存器的硬件CRC发生器电路。对所有单位、双位、奇数和大多数多位错误的错误检测，可编程的初始种子值，以及通过转置寄存器对输入数据和CRC结果进行转置的可选功能。 |

## 2.7.4 存储器和存储器 接口

该设备上有以下存储器和存储器接口。

表2-6.存储器和存储器接口

| 模块 | 描述 |
| --- | --- |
| 本地存储器控制器(LMEM) | 管理多个主外设和内核对系统RAM的同时访问。同时提供高速缓存控制，通过提供指令和数据管道的单周期访问来提高系统性能。 |
| 杂项系统控制模块（MSCM）。 | 杂项系统控制模块（MSCM）包含CPU配置寄存器和片上存储器控制器寄存器。 |
| 闪存 | • 程序闪存 - 可以执行程序代码的非易失性闪存<br>• FlexMemory - 包括以下内存类型。<br>  • FlexNVM - 非易失性闪存，可以执行程序代码，存储数据，或备份模拟EEPROM数据<br>  • FlexRAM - RAM存储器，可用作传统的SRAM或高端耐力仿真EEPROM存储，还能加速闪存编程 |
| QuadSPI | 四个串行外设接口（QuadSPI）块作为外部串行闪存设备的一个接口。它支持SDR和HyperRAM模式，分别达到4和8条双向数据线。 |
| 闪存控制器 | 管理设备和片上闪存之间的接口。 |
| SRAM | 内部系统RAM。 |

## 2.7.5 电源 管理

该设备上有以下模块可用于电源管理和系统电源模式控制。

**表2-7.电源管理模块**

| 模块 | 描述 |
| --- | --- |
| PMC | PMC包含内部电压调节器、上电复位（POR）和低电压检测（LVD）系统。 |
| SMC | 系统模式控制器（SMC）负责对系统进入和退出所有低功率的停止和运行模式进行排序。 |

## 2.7.6 计时

该设备上有以下时钟模块。

**表2-8.时钟模块**

| 模块 | 描述 |
| --- | --- |
| 系统时钟发生器（SCG） | SCG为MCU提供了几个时钟源，其中包括。<br>• 锁相环(PLL) - 电压控制的振荡器(VCO)<br>• 快速内部参考时钟（FIRC）--内部产生的48MHz时钟，可作为其他片上外设的时钟源。<br>• 慢速内部参考时钟（SIRC）--内部产生的8MHz时钟，可作为其他片上外设的时钟源。<br>• 系统振荡器（OSC）--系统振荡器与外部晶体或谐振器一起，为MCU生成参考时钟。 |
| 低功率振荡器（LPO） | 一个内部产生的低功耗时钟，典型频率为128kHz，可作为在低功耗模式下运行的模块的时钟源。 |
| 外围时钟控制（PCC） | 控制大多数模块的时钟选择。 |

## 2.7.7 模拟 模块

该设备上有以下模拟模块。

**表2-9.模拟模块**

| 模块 | 描述 |
| --- | --- |

| | |
|---|---|
| 12位模数转换器（ADC） | 12位逐次逼近的ADC |
| 模拟比较器（CMP） | 在电源电压的整个范围内比较两个模拟输入电压。 |
| CMP内的8位数模转换器（DAC）。 | 256抽头的电阻梯形网络，为需要电压基准的应用提供一个可选择的电压基准。 |

## 2.7.8 定时器 模块

该设备上有以下定时器模块。

**表2-10.定时器模块**

| 模块 | 描述 |
| --- | --- |
| 可编程延迟块（PDB） | • 16位分辨率<br>• 3位预分频器<br>• 触发事件信号的正向转换启动了计数器<br>• 支持多个被触发的延迟输出信号，每个信号都有一个独立控制的触发事件的延迟。<br>• 支持连续脉冲输出或单次拍摄模式，每个输出都是独立启用的，有可能触发事件<br>• 支持旁路模式<br>• 支持DMA |
| 灵活的定时器模块（FTM） | • 可选的FTM源时钟，可编程的预分频器<br>• 16位计数器支持自由运行或初始/最终值，计数是向上或向下的。<br>• 输入捕捉，输出比较，以及边缘对齐和中心对齐的PWM模式<br>• 将FTM通道作为具有相同输出的一对、具有互补输出的一对或具有独立输出的独立通道进行操作<br>• 每个互补对都有死机时间插入<br>• 硬件触发器的生成<br>• 对PWM输出的软件控制<br>• 多达4个故障输入，用于全局故障控制<br>• 可配置的通道极性<br>• 在输入捕捉、参考比较、计数器溢出或检测到的故障条件和重新加载机会时，可设置中断。<br>• 带输入滤波器的正交解码器、相对位置计数、位置计数中断或在外部事件中捕获位置计数<br>• 对FTM事件的DMA支持 |
| 低功耗定期中断定时器（LPIT）。 | • 四个通用的中断定时器<br>• 用于触发ADC转换的中断定时器<br>• 32位计数器分辨率<br>• 该计数器由一个异步时钟驱动，在低功耗模式下可以保持启用。<br>• 支持DMA<br>• 支持定时器通道的连锁反应 |
| 低功率定时器（LPTMR） | • 可选择预分频器/突变滤波器的时钟为128kHz（内部LPO），或内部参考时钟<br>• 可配置的突变滤波器或带有16位计数器的预分频器<br>• 带有比较功能的16位时间或脉冲计数器 |

| | |
|---|---|
| | • 定时器比较时产生的中断<br>• 在定时器比较上产生硬件触发 |
| 实时时钟（RTC） | • 带有32位报警的32位秒计数器<br>• 16位预分频器，具有补偿功能，可以纠正0.12ppm和3906ppm之间的误差 |

## 2.7.9 通讯 接口

该设备上有以下通信接口。

**表2-11.通信模块**

| 模块 | 描述 |
|---|---|
| 低功耗串行外设接口（LPSPI）。 | 用于与外部设备通信的同步串行总线。LPSPI可选择在低功率模式下保持功能。 |
| 低功耗集成电路（LPI2C）。 | 允许一些设备之间的通信。还支持系统管理总线（SMBus）规范，版本2。LPI2C可选择在低功耗模式下保持功能。 |
| 低功耗通用异步接收器/发射器（LPUART）。 | 异步串行总线通信接口，支持LIN主和从操作。LPUART可选择在低功率模式下保持功能。 |
| SAI/I2S | I2S（或I2S）模块提供一个同步音频接口（SAI），支持具有帧同步的全双工串行接口，如I2S、AC97、TDM和编解码/DSP接口。 |
| ENET | 该核心实现了一个符合IEEE802.3-2002标准的双速10/100-Mbit/s以太网MAC。MAC层提供与半双工或全双工10/100-Mbit/s以太网LAN的兼容性。 |
| 弹性CAN | FlexCAN模块是一个根据ISO 11898-1标准和CAN 2.0 B协议规范实施CAN协议的通信控制器。 |

## 2.7.10 调试 模块

该设备上有以下调试模块。

**表2-12.调试模块**

| 模块 | 描述 |
|---|---|
| JTAGC | JTAGC块提供了测试芯片功能和连接性的方法，同时在不处于测试模式时对系统逻辑保持透明。测试是通过边界扫描技术进行的，如IEEE 1149.1-2001标准所定义。所有输入和输出到JTAGC模块的数据都是以串行格式进行通信。 |

# 第三章 内存 地图

## 3.1 简介

该芯片包含各种存储器和存储器映射的外设，它们位于一个32位连续的存储器空间中。本章介绍了该存储器空间内的存储器和外设位置。

关于内存图的细节出现在本文件所附的电子表格中。S32K1xx_memory_map.xlsx。要访问这个电子表格，请查看该文件的附件列表。

## 3.2 SRAM内存 地图

### 3.2.1 S32K14x。SRAM内存 地图

片上RAM被分成两个区域。SRAM_L和SRAM_U。RAM的实现使得SRAM_L和SRAM_U的范围在内存地图中形成一个连续的块。详情见本文件所附的S32K1xx_memory_map.xlsx。

对SRAM_L和SRAM_U存储器范围的访问超出了芯片上的RAM数量，导致总线周期以错误终止，随后在请求的总线主站中出现适当的响应。

### 3.2.2 S32K11x。SRAM内存 地图

在S32K11x中，片上RAM可以用于以下应用。

- 安全关键应用。可以使用SRAM_U，它从2000_0000开始。
- 非安全关键应用。可以使用SRAM_U和1KB MTB

**注意事项**

CM0+架构实现了单一的RAM控制器，因此SRAM_U和MTB被称为单一的连续的内存区域。S32K11x中的MTB与S32K14x中的SRAM_L位于同一位置。

详见本文件所附的S32K1xx_memory_map.xlsx。对内存范围的访问超出了芯片上的RAM数量，会导致总线周期以一个错误终止，随后请求总线主站做出适当的响应。

## 3.3 闪存 地图

各种闪存和闪存寄存器位于不同的基址，如下图所示。各自的基址在本文件所附的S32K1xx_memory_map.xlsx文件中规定。

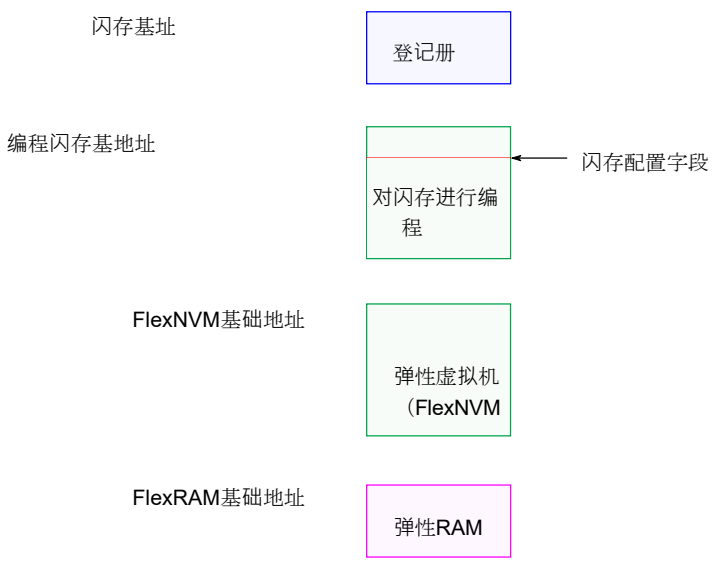



**图3-1.闪存地图**

## 3.4 外围桥（AIPS-Lite）内存 地图

外围存储器图可以通过一个横杆从属端口访问。

有三个与周边空间相关的区域，如下表所示。

表3-1.与周边空间有关的区域

| 地址空间 | 区域描述 |
| --- | --- |
| 0x4000_0000–0x4001_FFFF | 一个128KB的区域，被划分为32个空间，每个空间4KB大小，保留给平台上的外围设备。AIPS-Lite为所有32个空间生成唯一的模块授权。 |
| 0x4002_0000–0x4007_FFFF | 一个384KB的区域，被划分为96个空间，每个空间4KB大小，保留给非平台模块。AIPS-Lite为所有96个空间生成唯一的模块授权。 |
| 0x400F_F000 | 一个用于访问GPIO模块的4KB区域。这个区块通过端口分配器连接到AMBA总线上，并提供直接的主控访问，而不会产生与通过AIPS-Lite模块访问相关的等待状态。GPIO只在这个区域的上部空间实现（从0x400F_F000开始的4 KB）。 |

通过PCC/SIM寄存器中的时钟门控制位被禁用的模块会禁用相关的AIPS-Lite插槽。对未实施或禁用的外围桥接插槽内的任何地址的访问都会导致传输错误终止。

**注意事项**

当试图访问不可用功能的内存地图区域时（见SIM_SDID[FEATURES]），通过PCC
CGC位启用相应的模块时钟，不会出现任何传输错误终止
。

### 3.4.1 内存操作的先读后写顺序和所需序列化

在某些情况下，对一个外设的写入必须在后续动作发生之前完全完成。这种情况的例子包括。

- 退出一个中断服务例程（ISR）。
- 改变一个模式
- 配置一个功能

在这些情况下，应用软件必须执行先读后写的顺序，以保证所需的内存操作的序列化，如下表所示。

表3-2.保证内存操作所需的序列化的后读后写顺序

| 步骤 | 行动 |
| --- | --- |
| 1 | 写入外围寄存器。 |
| 2 | 读取写入的外围寄存器以验证写入。 |
| 3 | 继续进行后续操作。 |

**注意事项**

在S32K14x系列器件中，造成这些情况的一个因素是处理器的写缓冲。处理器架构有一个可编程的配置字段来禁用写缓冲。actlr[disdefwbuf]。(详见ARM® Cortex® M4处理器技术参考手册，修订版r0p1，在http://arm.com)。然而，禁用缓冲写入可能会降低系统性能，而不是简单地在真正需要的情况下执行所需的内存序列化。

## 3.5 私人外设总线（PPB）内存 地图

PPB是定义的ARM总线架构的一部分，它提供对选定的处理器本地模块的访问。这些资源只能从内核中访问；其他系统主站不能访问它们。

**表3-3.CM4的PPB内存图**

| 系统32位地址范围 | 资源 |
|---|---|
| 0xE000_0000–0xE000_0FFF | 仪器仪表跟踪巨室（ITM）。 |
| 0xE000_1000–0xE000_1FFF | 数据观察点和跟踪（DWT） |
| 0xE000_2000–0xE000_2FFF | 闪存补丁和断点（FPB） |
| 0xE000_3000–0xE000_DFFF | 保留 |
| 0xE000_E000–0xE000_EFFF | 系统控制空间（SCS）（用于NVIC和FPU |
| 0xE000_F000–0xE003_FFFF | 保留 |
| 0xE004_0000–0xE004_0FFF | 跟踪端口接口单元（TPIU） |
| 0xE004_1000–0xE004_1FFF | 保留 |
| 0xE004_2000–0xE004_2FFF | 保留 |
| 0xE004_3000–0xE004_3FFF | 保留 |
| 0xE004_4000–0xE007_FFFF | 保留 |
| 0xE008_0000–0xE008_0FFF | 杂项控制模块（MCM） |
| 0xE008_1000–0xE008_1FFF | 保留 |
| 0xE008_2000–0xE008_2FFF | 缓存控制器（LMEM） |
| 0xE008_3000–0xE00F_EFFF | 保留 |
| 0xE00F_F000–0xE00F_FFFF | ARM核心ROM表1--允许自动检测调试组件 |

1.ARM Core ROM 表是 ARM CoreSight 调试基础结构可选择的要求，以发现芯片上的组件。该ROM表与MCU启动ROM没有任何关系。

表3-4.用于CM0+的PPB内存图

| 系统32位地址范围 | 资源 |
|---|---|
| 0xE000_0000–0xE000_0FFF | 保留 |
| 0xE000_1000–0xE000_1FFF | 数据观察点和跟踪（DWT） |
| 0xE000_2000–0xE000_2FFF | ARM内核ROM表 |
| 0xE000_3000–0xE000_DFFF | 保留 |
| 0xE000_E000–0xE000_EFFF | 系统控制空间（SCS）（用于NVIC）。 |
| 0xE000_F000–0xE003_FFFF | 保留 |
| 0xE004_0000–0xE004_0FFF | 保留 |
| 0xE004_1000–0xE004_1FFF | 保留 |
| 0xE004_2000–0xE004_2FFF | 保留 |
| 0xE004_3000–0xE004_3FFF | 保留 |
| 0xE004_4000–0xE007_FFFF | 保留 |
| 0xE008_0000–0xE008_0FFF | 保留 |
| 0xE008_1000–0xE008_1FFF | 保留 |
| 0xE008_2000–0xE008_2FFF | 保留 |
| 0xE008_3000–0xE00F_EFFF | 保留 |
| 0xE00F_F000–0xE00F_FFFF | 保留 |
| 0xE010_0000-0xEFFF_FFFF | 保留 |

## 3.6 CM4的异化位带区域 核心的异化位带区域

SRAM_U、AIPS-Lite和通用输入/输出（GPIO）模块资源位于Cortex-M4F处理器位带区域。

该处理器还包括与两个1MB的位带空间相关的两个32MB的别名位带区域。32MB空间中的每个32位位置都映射到位带区域的一个单独的位。在别名区域的32位写操作与在位带区域的目标位上的读-修改-写操作具有相同的效果。

写入别名区域的数值的第0位决定了写入目标位的数值。

- 写入一个位为0的值时，会向目标位写入一个1。
- 在0位清零的情况下写一个值，会在目标位上写一个0

。在别名区域的32位读数会返回以下两种情况之一。

- 一个0x0000_0000的值，表示目标位已清零
- 一个0x0000_0001的值，表示目标位被设置。

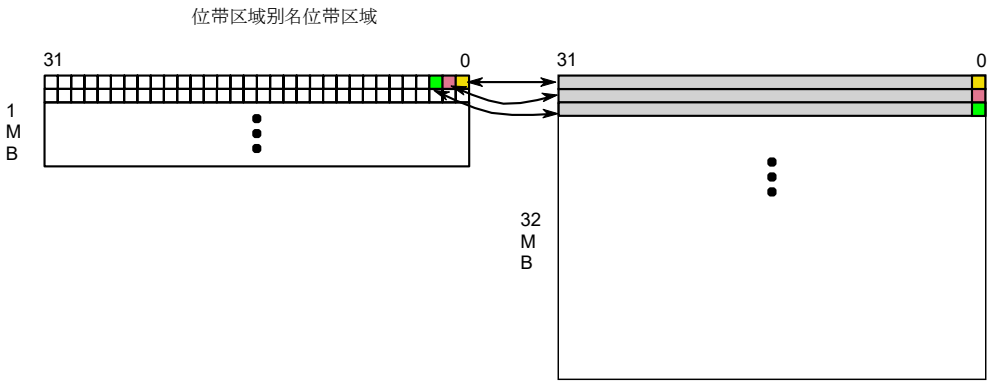



**图3-2.别名的位带映射**

### 注意事项

一个位带区域的每个位都有一个等效的位，可以通过别名位带区域中相应的长字中的位0进行操作。

### 注意事项

不要对w1c状态位使用位带。

### 注意事项

S32K产品系列和软件驱动程序支持位带，但ARM不再提倡使用它。因此，我们建议不要使用位带。

# 章 4
# 信号复用和引脚 分配

## 4.1 简介

信号多路复用使多个功能共用一个焊盘。

信号复用单元包括来自GPIO、PORT和焊盘接口逻辑的控制信号。信号复用单元由几个独立的子单元组成，每个单元处理一个焊盘的信号复用。

端口控制块控制模块特定的焊盘设置（上拉等）和外部引脚上存在的信号。控制信号的描述见PORT_PCR。关于每个端口的复位值，见《参考手册》所附的IO信号描述输入复用表。

## 4.2 功能 描述

信号复用的架构实现如下图所示。

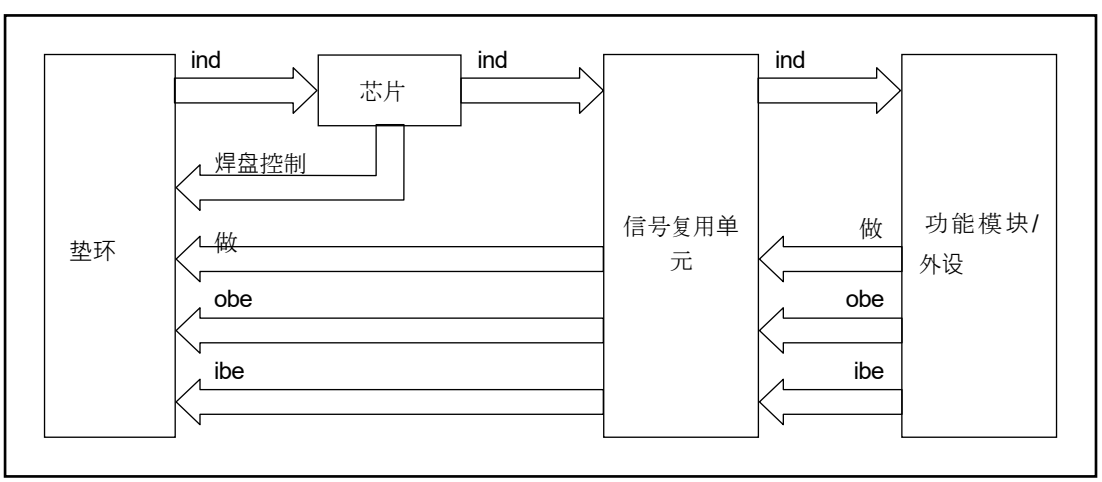



**图4-1.信号复用**

## 4.3 Pad 描述

下图显示了一个GPIO焊盘的基本表示。

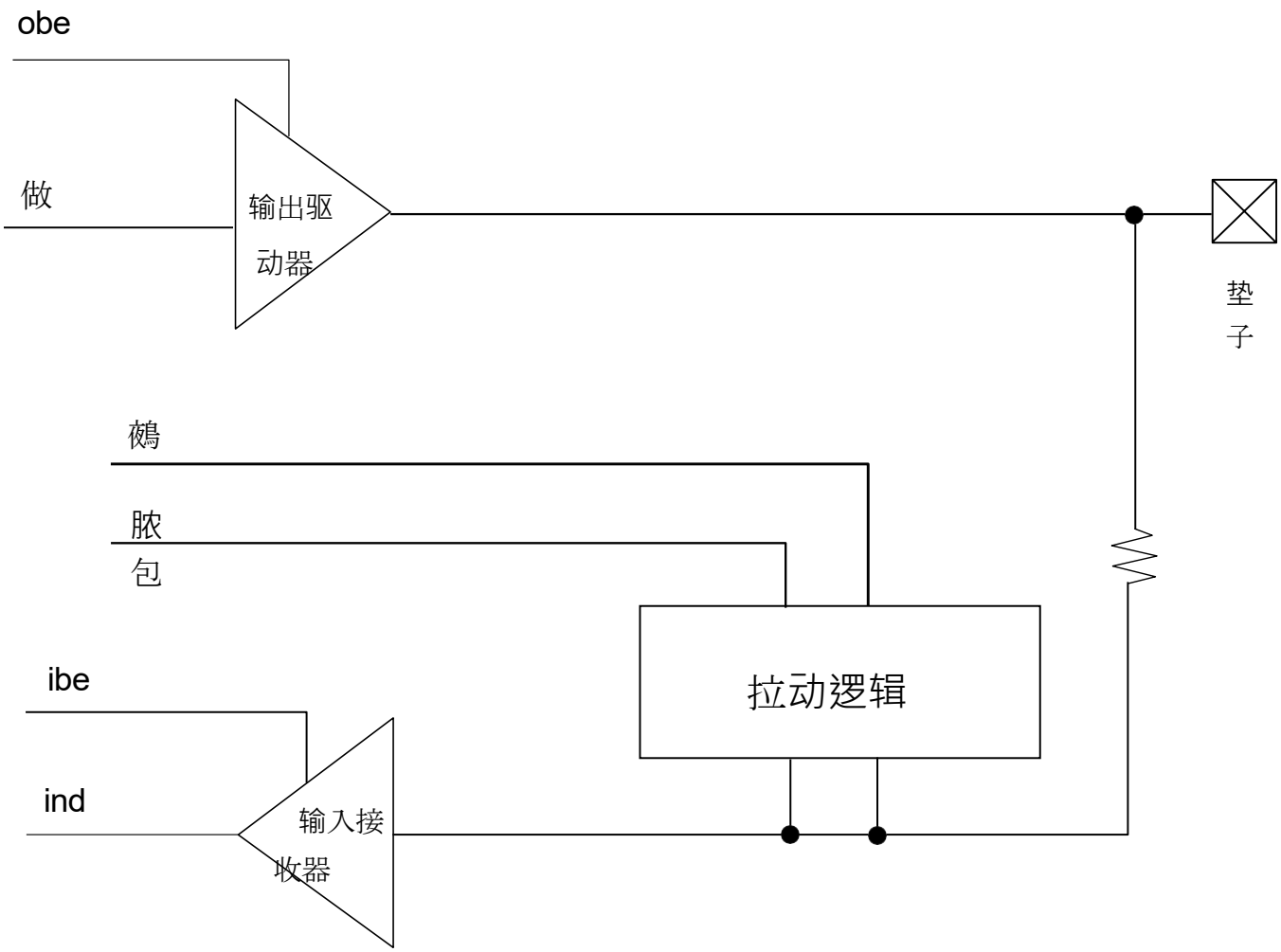



**图4-2.GPIO焊盘的表示方法 表4-1.焊盘信号**

**描述**

| 信号名称 | 方向 | 描述 |
|---|---|---|
| 垫子 | 输入/输出 | 输入/输出到外部世界 |
| 做 | I | 数据从磁芯进入磁垫 |
| obe | I | 启用输出驱动器 |
| 鹇 | I | 0：禁用内部上拉或下拉电阻 1：启用内部上拉或下拉电阻 |
| 腋包 | I | 0：如果pue被设置，则启用内部下拉电阻 1：如果pue被设置，则启用内部上拉电阻 |
| ibe | I | 启用输入接收器 |
| ind | O | 从垫子出来的数据进入核心 |

表4-2.真值表

| obe | 做 | 垫子 | 描述 |
|---|---|---|---|
| 0 | X | Z | 输出缓冲器禁用，垫高Z（如果不配置为输入）。 |
| 1 | 0/1 | 0/1 | 输出缓冲区启用，垫子=做 |

| 鹅 | 脓包 | 垫子 | 描述 |
|---|---|---|---|
| 0 | X | - | 禁用微弱的拉力。焊盘保留之前的状态 |
| 1 | 0 | 0 | 启用了微弱的下拉功能 |
| 1 | 1 | 1 | 启用了微弱的拉升功能 |

| ibe | 垫子 | ind | 描述 |
|---|---|---|---|
| 0 | X | 0 | 输入缓冲器被禁用，Ind得到低电平 |
| 1 | 0/1 | 0/1 | 启用输入缓冲区，ind = pad |

**注意事项**

该器件不支持所有引脚的开漏。只有被配置为需要开漏的协议的引脚（例如，LPI2C、LPUART单线）才能在开漏模式下工作。

## 4.4 默认的垫子 状态

复位后的默认焊盘配置如下。对于PTA4、PTA5、PTC4和PTC5，默认配置是按照协议规范/要求进行的。

表4-3.默认焊盘配置

| 钉子 | 默认功能 | 默认的垫子状态 | ibe | obe | 鹅 | 脓包 |
|---|---|---|---|---|---|---|
| PTA4 | JTAG_TMS | 启用微弱的上拉功能 | 1 | 0 | 1 | 1 |
| PTA5 [1] | RESET_b | 启用微弱的上拉功能 | 1 | 0 | 1 | 1 |
| PTC4 | JTAG_TCK | 启用弱下拉功能 | 1 | 0 | 1 | 0 |

| | | | | | | |
|---|---|---|---|---|---|---|
| PTC5 | JTAG_TDI | 启用微弱的<br>上拉功能 | 1 | 0 | 1 | 1 |
| 其他 | 失能 | 高阻抗 | 0 | 0 | 0 | 0 |

1.在复位时，除了复位引脚外，其他引脚的行为是相同的。芯片通过obe=1，pue=0驱动pad为低电平，直到复位序列完成，以指示复位到片外连接的设备。

## 4.5 信号复用 sheet

输入信号描述 《参考手册》所附的输入复用表包含了该设备的引脚/球的信息。

表中的 "IO信号表 "和 "输入多路复用 "标签与信号复用信息相对应。IO信号表 "包括所有引脚复用的细节，"输入复用
"指定了输入复用的优先级，其中一个输入路径是由一个以上的焊盘驱动。

### 4.5.1 IO信号 表

下面是一个IO信号表的例子片段。为了选择任何功能，需要对焊盘PCR寄存器（参考PORT_PCRn）进行相应配置。

| Port | CR | SSS | Function | Module | Description | Direction | Pad Type | S32K144_100lqfp | S32K144_64lqfp | S32K144_BGA | PCR | | | | | | | | Reset | Bits Configurable |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | | ISF | IRQC[3:0] | | LK | MUX[2:0] | | DSE | PFE | PE | PS | | |
| PTA0 | PCR_PTA0 | 0000_0000 | DISABLED | | Signal Path Disabled | - | GPIO | 79 | 50 | A9 | 0 | 0000 | | 0 | 000 | | - | - | 0 | 0 | '00000000000000000000000000000000' | '00000001000011111000011100000011' |
| | | 0000_0001 | PTA0 | PTA | Port A I/O | I/O | | | | | | | | | | | | | | | | |
| | | 0000_0010 | FTM2_CH1 | FTM2 | FTM Channel | I/O | | | | | | | | | | | | | | | | |
| | | 0000_0011 | LPI2C0_SCLS | LPI2C0 | LPI2C Secondary Clock I/O | I/O | | | | | | | | | | | | | | | | |
| | | 0000_0100 | FXIO_D2 | FXIO | FlexIO Bi-directional Shift/timer I/O | I/O | | | | | | | | | | | | | | | | |
| | | 0000_0101 | FTM2_QD_PHA | FTM2 | FTM quadrature Decode PhaseA | I | | | | | | | | | | | | | | | | |
| | | 0000_0110 | LPUART0_CTS | LPUART0 | Clear To Send (bar) | I | | | | | | | | | | | | | | | | |
| | | 0000_0111 | TRGMUX_OUT3 | TRGMUX | Trigger Mux Output | O | | | | | | | | | | | | | | | | |
| | - | - | ADC0_SE0 | ADC0 | ADC Single Ended Input | I | | | | | | | | | | | | | | | | |
| | - | - | CMP0_IN0 | CMP0 | Comparator Input Signal | I | | | | | | | | | | | | | | | | |

**图4-3.IO信号表片段**

上图中的各列描述如下。

- 端口：IO信号表中的这个字段指定了设备的PAD名称。
- CR（控制寄存器）。这个字段规定了与端口字段相对应的PCR的名称。在这个设备上有五个PORT实例，即PTA、PTB、PTC、PTD和PTE。每个焊盘都有一个相应的控制寄存器，在IO信号表中被称为PCR_PTXn，其中X指的是PORT实例，n指的是该PORT实例的相应引脚。关于PCR字段的描述，请参考PORT_PCR。
- SSS：该字段根据PCR[Mux_mode]规定了ALT操作模式。不是所有的引脚都支持所有的引脚复用槽。未实现的引脚复用槽被保留。相应的引脚被配置在下面的引脚复用槽中，具体如下。
  - 000：备选0（信号路径禁用）。
  - 001：备选1（GPIO）
  - 010:替代方案2（特定芯片）
  - 011:替代方案3（特定芯片）
  - 100：替代方案4（特定芯片）。
  - 101: 备选方案5（芯片专用）

- 110：备选方案6（特定芯片）
- 111：备选方案7（特定芯片）

模拟功能在这一栏中用"-"指定。这里ADC_SE0和CMP_IN0代表模拟功能。

默认情况下，ALT0 模式（由 PTXn_PCR[SSS]配置为 3'b000）对应于禁用功能，焊盘代表禁用（高阻抗）状态。如果垫子由模拟功能组成，一旦模拟模块被配置为启用相应的通道/输入，ALT0模式就对应于模拟功能。

例如，PTA0支持ADC0通道0和CMP通道0。默认情况下，垫子是禁用的。在ADC0被配置为使用ADC0_SC1n[ADCH]启用通道0后，垫子就作为ADC输入通道发挥作用。另外，如果CMP被配置为使用CMP_C1[PSEL]或CMP_C1[MSEL]启用通道0，则垫子作为CMP输入通道发挥作用。软件必须确保一次只启用一个功能。

- 功能。这个字段根据SSS字段所指定的相应ALT模式，指定垫子的功能。
- 模块。模块字段包含管理ALT模式的垫子的模块。
- 描述。这个字段提到了对垫子功能的简短描述。
- 方向。这个字段指定了有关功能的垫子的方向（输入、输出或进出）。
- 接下来的几列指定了该器件支持的封装中的引脚编号。
- PCR。这个字段指定了相应焊盘的默认PCR值。关于PCR字段的描述，请参考PORT_PCR。
- 接下来的两列规定了复位值和与垫子对应的PCR的可配置位域。
- 垫子类型。这个字段提到了相应的垫子的类型。
  - GPIO: 通用IO焊盘
  - GPIO-HD: 支持高驱动功能的通用IO焊盘

**注意事项**

如果振荡器被启用，那么为EXTAL/XTAL引脚启用GPIO或LPI2C功能会导致器件损坏。这种情况必须通过软件来避免。

### 4.5.2 输入多路复用 表

由于同一功能可以通过配置各自的PCR复用到多个焊盘上，因此有优先级输入复用的功能。在同一输入被多个焊盘驱动的情况下，具有最高优先级的焊盘（1为最高）将驱动该输入。以下是输入复用表的一个片段。

| Destination Instance | Destination Function | Priority | SSS | Source Instance | Source Signal |
|---|---|---|---|---|---|
| CAN0 | CAN0_RX | 1 | 0000_0011 | IO_PAD | PTC2 |
| | | 2 | 0000_0101 | IO_PAD | PTE4 |
| | | 3 | 0000_0101 | IO_PAD | PTB0 |
| | | 4 | | - | disable low |

**图4-4.输入多路复用表片断**

下面简要介绍一下图中的各栏。

- 目的地实例。这个字段包含输入路径的实例名称，信号将从padring传播到那里。
- 目的地功能。这个字段提到了输入路径的功能名称。
- 优先级。这个字段指定了路径的优先级。优先级1是最高的，然后往下递减。
- SSS：该字段指定了与源信号栏中指定的焊盘相对应的PCR[Mux_mode]值。当没有垫子驱动输入路径时，默认信号源为空白。
- 源实例。这个字段指定了源垫的类型。当没有垫子驱动输入路径时，默认的源会提到一个空白。
- 源信号。这个字段提到了垫子的名字。一个 "禁用低电平"/"禁用高电平 "指定了没有任何垫子驱动输入路径时的信号行为。

## 4.6 插脚 图示

请参阅《参考手册》所附的IO信号描述输入多路复用表，了解与可用封装相对应的引脚图。

# 第5章 安全 概述

## 5.1 简介

该S32K1xx产品系列的所有芯片都有一套全面的客户可配置的安全功能，旨在保护代码和数据免受未经授权的访问。

## 5.2 设备 安全

闪存安全对CSEc和非CSEc部分都是可用的。SIM_SDID[7]
指示你的设备上是否有CSEc。

CSEc和非CSEc用户都需要运行PGMPART命令来配置设备的密钥大小。更多信息，请参见Program
Partition命令。这个过程涉及到配置一些用户的密钥。对于非CSEc用户，密钥大小将是00。

- EEERAM的总大小因存储用户密钥所需的空间而减少。(关于可用的EEERAM，见介绍一章)。
- 用户密钥空间在EEERAM中有效地成为不可寻址的空间。
- 对于非CSEc部件和密钥大小=00的CSEc部件，CSEc_PRAM的访问不被保证。

### 5.2.1 闪存 安全

闪存模块根据FSEC[SEC]字段持有的状态向MCU提供安全信息。反过来，MCU确认安全请求并限制对闪存资源的访问。在复位过程中，闪存模块使用从闪存配置字段的安全字节中读取的数据初始化FSEC寄存器。

**注意**

**事项**

安全功能只适用于通过调试的外部访问。CPU对闪存的访问不受FSEC状态的影响。

在不安全的状态下，所有的闪存命令对编程接口（SWD（串行线调试）和JTAG（联合测试访问组））都是可用的，用户代码执行闪存控制器命令也是如此。当闪存是安全的（FSEC[SEC] = 00b, 01b, 或
11b），编程器接口只允许启动大规模擦除操作，而不能访问内存位置。

有关闪存安全选项和启用/禁用闪存安全的进一步信息，可在闪存模块中找到。

#### 5.2.1.1 闪存安全与调试的相互作用

当闪存安全被激活时，JTAG端口不能访问MCU的内存资源。边界扫描链操作正常，但调试功能被禁用，因此调试端口不能读取闪存内容。

当闪存安全被激活时，SWD端口不能访问MCU的内存资源。

尽管大多数调试功能被禁用，但只要MEEN（大规模擦除）被启用，调试器就可以写到MDM-
AP控制寄存器中的Flash大规模擦除进行中位，以触发大规模擦除（擦除所有块）命令。运行大规模擦除命令会使FSEC中断，之后JTAG端口可以访问存储器资源。即使某些内存位置被保护，也允许通过调试器进行大规模擦除。

当大规模擦除被禁用时，通过调试器的大规模擦除被阻止了。因此，你将不能在一个安全的设备中连接调试器（MEEN禁用）。在这种情况下，一个替代方案是通过任何通信接口运行验证后门密钥访问。

FTFC_FCSESTAT[EDB]位的清除状态如下。

- 如果调试器已经切换到SWD模式，FTFC_FCSESTAT[EDB]位在POR时被重置。
- 如果调试器保持在JTAG模式下，如果调试器正确断开连接，FTFC_FCSESTAT[EDB]位会在pin_reset上复位。

### 5.2.2 加密服务引擎（CSEc）的安全 特征

FTFC模块的加密服务引擎（CSEc）实现了SHE功能规范中描述的一套全面的加密功能，包括：1:

- >10个一般用途键
- AES-128, CBC, ECB, CMAC
- 顺序、平行和严格的启动模式
- AES-128 CMAC计算和认证
- 伪随机数生成（PRNG）和真随机数生成（TRNG）。

### 5.2.3 设备启动 模式

在并行安全启动模式下，安全启动和应用程序启动将并行进行。同时，即使安全启动失败，应用程序也会启动。但BOOT_PROT属性设置为1的密钥将被禁用。应用程序可以使用其余的密钥和加密/解密/SHE功能。但是，在试图执行任何SHE命令之前，应用程序代码应该轮询繁忙标志FCSESTAT[BSY]
。此外，在并行安全启动期间，任何由Core发起的模式转换将被中止，安全启动将继续进行。

在并行启动操作期间，FLASH_CLK必须是默认的FIRC_CLK，在启动操作完成之前不应改变。一旦启动操作完成，它可以被改变为任何数值，包括它在RUN模式下的最大值26.6MHz和在HSRUN模式下的28MHz。

在顺序安全启动中，应用程序只有在成功完成安全启动后才能启动。而且，应用程序可以执行任何命令，而不必理会FCSESTAT[BSY]标志。然而，在严格的启动模式下，如果安全启动失败，芯片将根本无法启动，并将保持在复位状态。

## 5.3 安全用例 例子

### 5.3.1 安全启动：检查引导程序的完整性和 真实性

下图说明了检测和防止修改引导程序的用例。

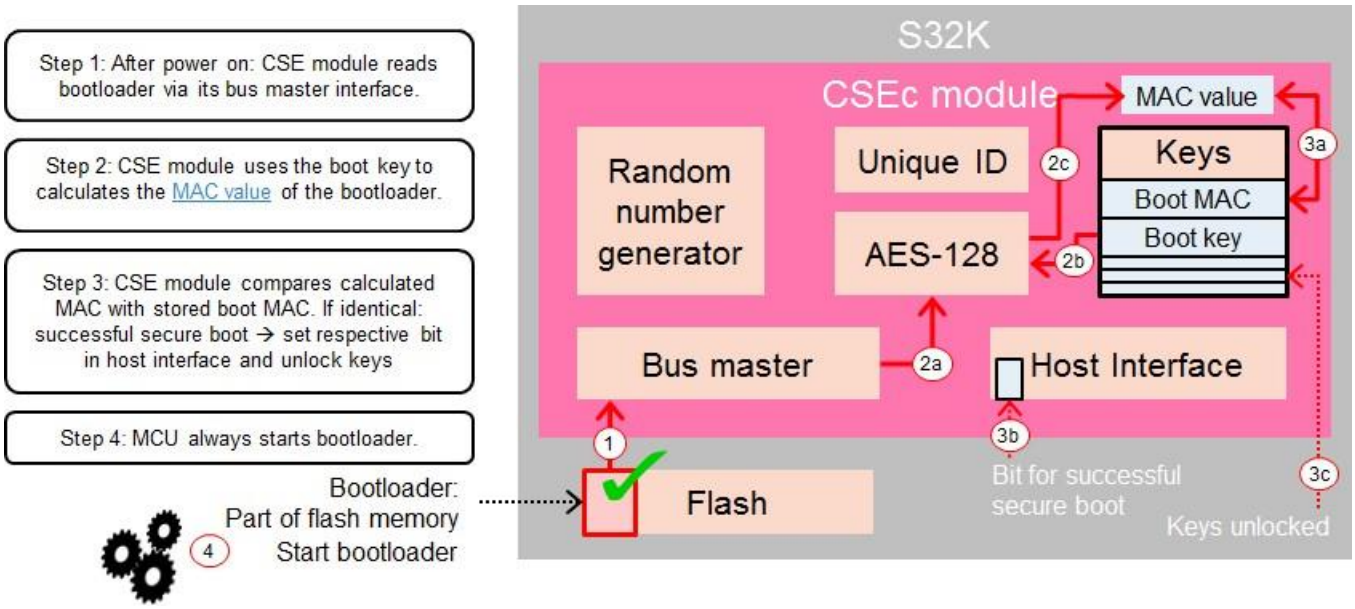



**图5-1.启动程序的完整性和自主性**

MAC可以防止对引导程序的修改，并依赖于（秘密）引导密钥。

只有当计算出的MAC值与存储的启动MAC值相匹配时：才会发生成功的安全启动，并且钥匙被解锁以便进一步使用。

## 5.3.2 信任链：检查闪存的完整性和 真实性

在这个用例中。

- 引导程序受到安全引导程序的保护。
- 存储在引导程序中的MAC提供了闪存中相关部分的完整性和真实性。
- 闪存的逐个部分检查在执行之前确保每个部分的完整性和真实性。闪存的关键部分（例如，MCU配置/IRQ表）被检查，然后尽快执行。

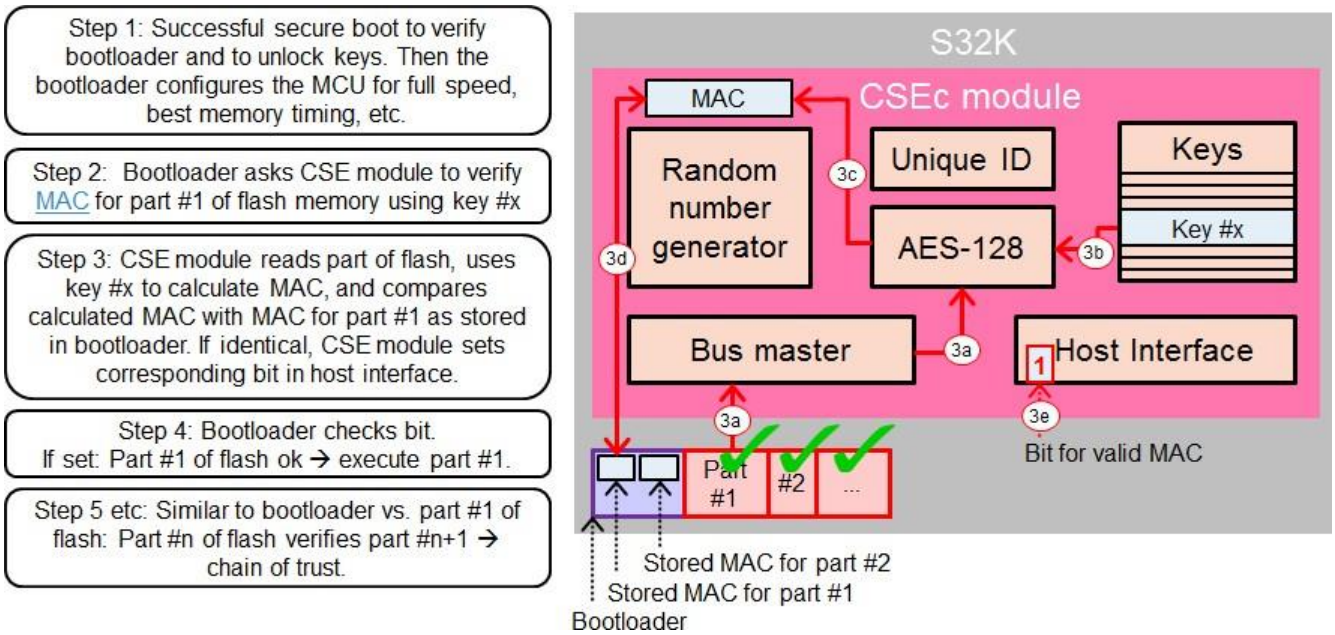



**图5-2.闪存的完整性和真实性**

## 5.3.3 安全的 通讯

这个用例展示了如何防止ECU发送的非法信息。

- 随机数生成和检查可防止重放攻击。
- 加密可以防止窃听。
- 随机数生成/检查和加密确保数据的完整性和真实性。

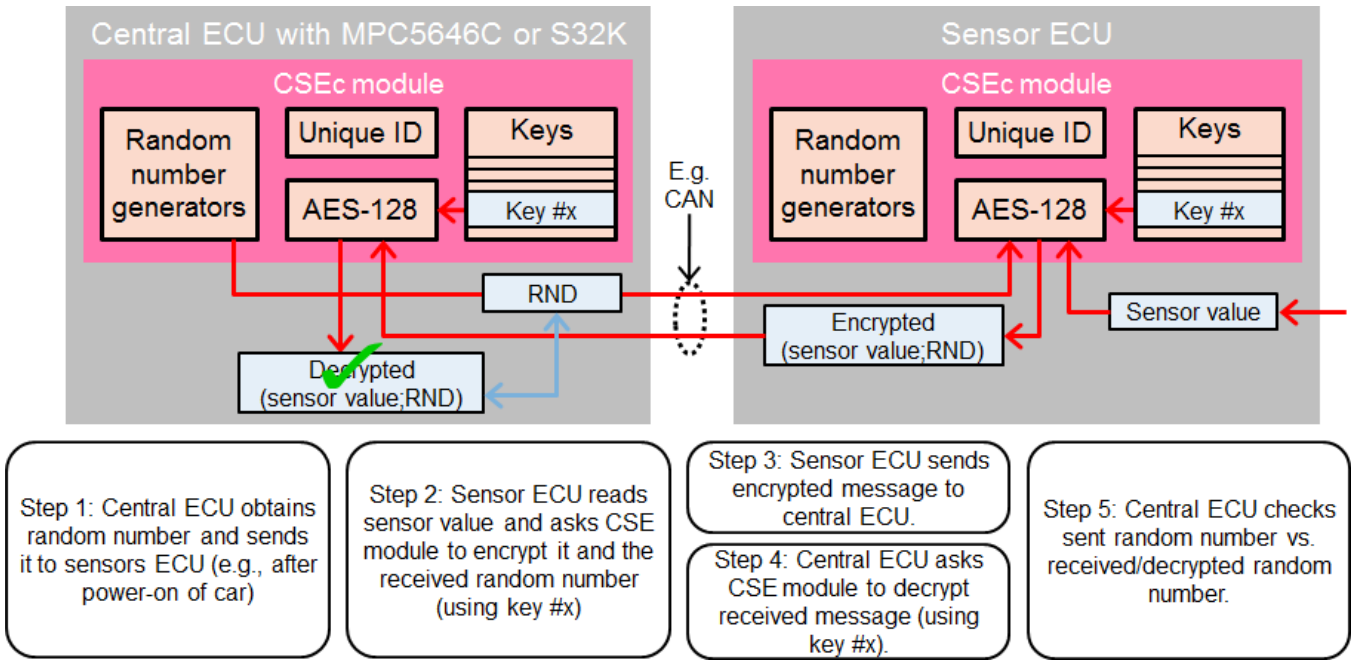



**图5-3.安全通信**

## 5.3.4 组件 保护

ECU<n>的更换或修改将改变其唯一的ID和/或密钥。这个用例显示了如何检测这两种变化。

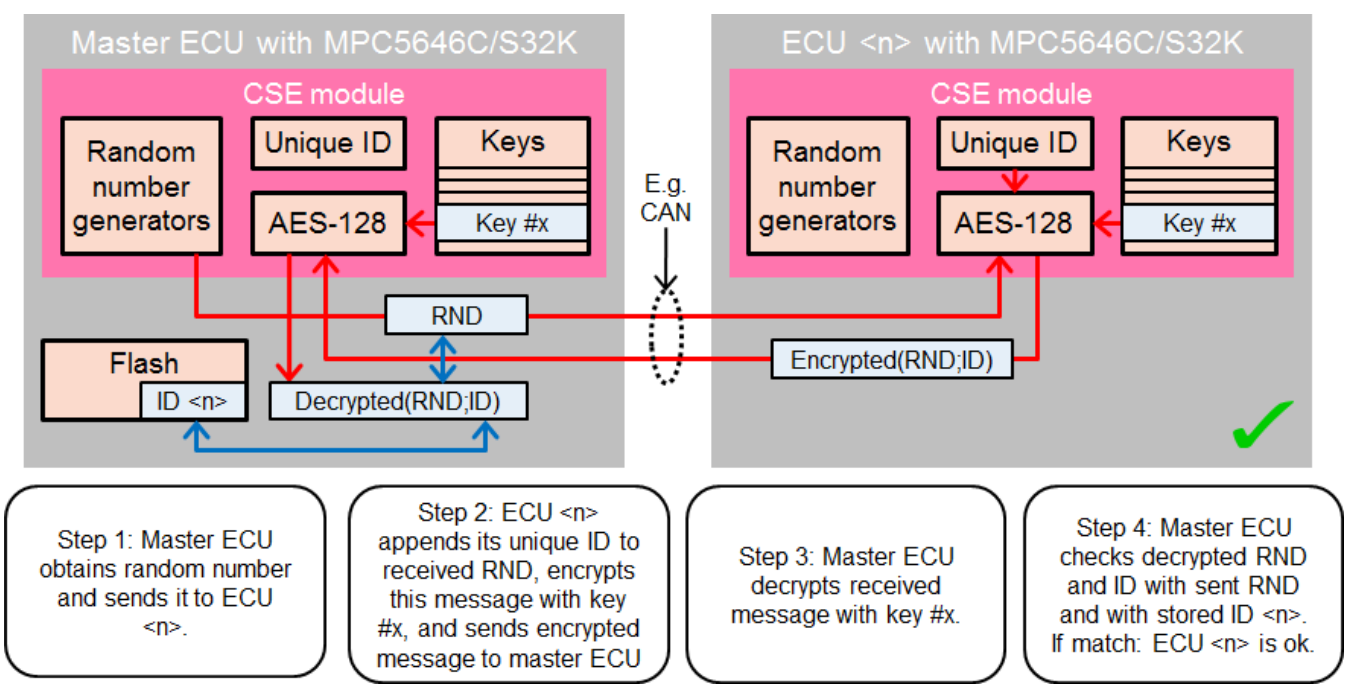


图5-4.组件保护

### 5.3.5 消息-认证 例子

该用例包括一个Rx-CAN消息认证方案。

1. 存储在本地缓冲区的CAN数据
2. FlexCAN触发中断到核心/DMA
3. 向CSEc存储器传输数据（最多12个8字节的CAN报文+16字节的CMAC）。
4. 触发CSEc CMAC计算/验证
5. CSEc触发了对核心的中断
6. 内核读取已处理的信息数据

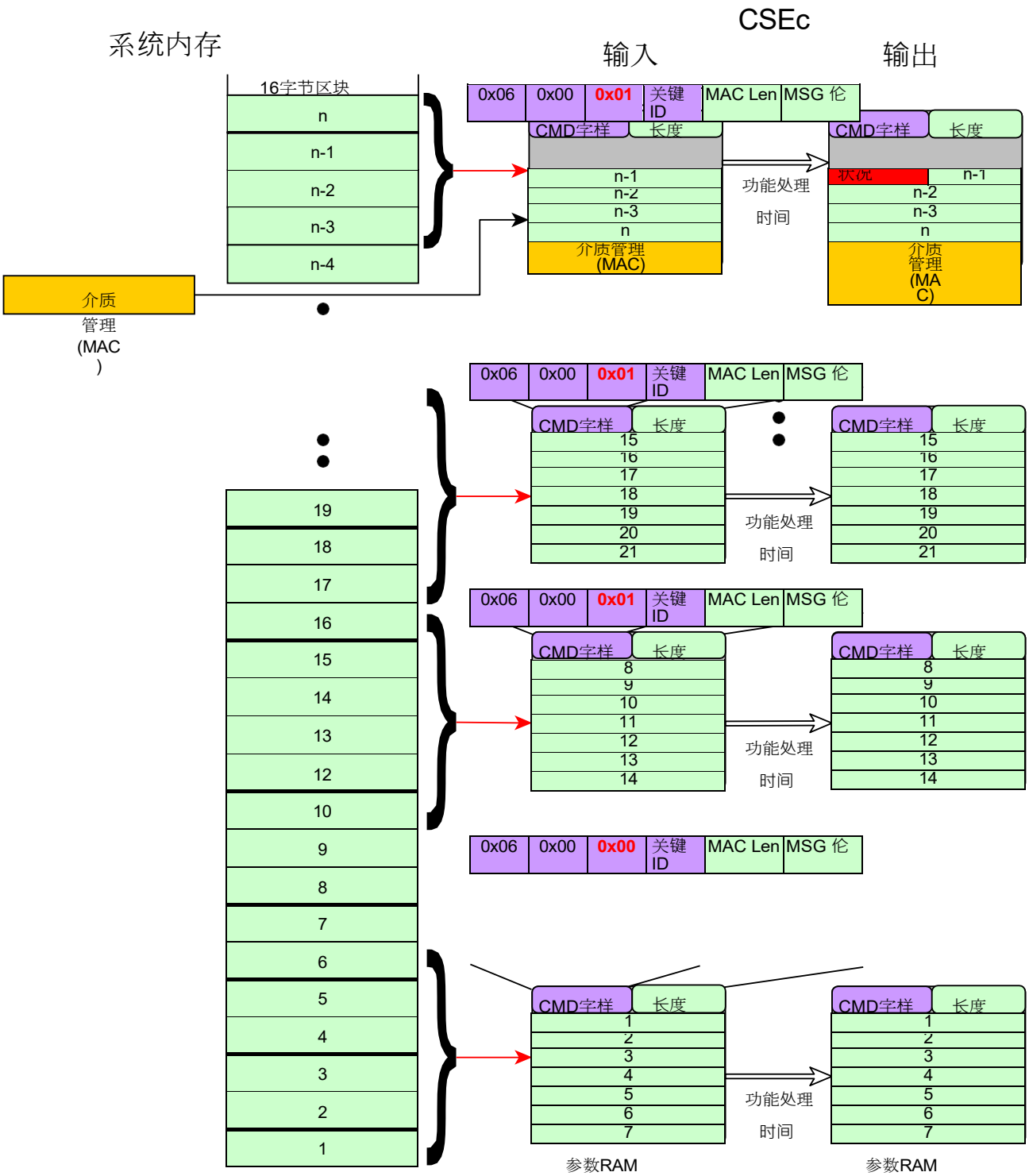



图5-5.信息验证

## 5.4 失败前需要采取的步骤 分析

在将设备返回恩智浦进行故障分析之前，用户必须运行CMD_DBG_CHAL和CMD_DBG_AUTH命令，并确保所有的用户密钥被删除。这是在恩智浦启用故障分析的一个强制性步骤。

**注意事项**

如果任何键设置了WRITE_PROTECTION标志，那么用户将无法用DEBUG_CHAL和DEBUG_AUTH命令删除该键，故障分析也就不可能了。

## 5.5 安全编程流程示例（安全 启动）

1. 运行PGMPART以配置所需的钥匙数量和其他参数
2. Pflash中的程序代码部分将在安全启动中被检查。
3. load_key(boot_mac_key)。
4. CMD_BOOT_DEFINE选择启动的方式和数据的大小，以便在Pflash中验证。

   然后可选择将部件重置为"自动计算"，并对BOOT_MAC进行编程，或由用户通过外部计算加载BOOT_MAC。

# 第六章 安全 概述

## 6.1 简介

S32K1xx系列是根据ISO26262开发的，它有一个集成的安全概念，目标是ISO26262 ASIL-B完整性等级。以下文件支持将S32K1xx芯片集成到安全相关系统中。

- **参考手册（S32K1xxRM）：描述S32K1xx芯片的**编程模型和功能
- 数据表（S32K1xx）：描述S32K1xx的工作条件以及时序和电气特性
- 安全手册（S32K14xSM）：描述S32K14x的安全概念和可能的安全机制（集成在S32K14x、系统级硬件或系统级软件中），以及减少依赖性故障的措施。
- 动态FMEDA：归纳分析，实现系统级安全机制的定制，包括由此产生的ISO 26262（SPFM、LFM和PMHF）和IEC 61508（SFF和Beta IC因子）的安全指标。
- FMEDA报告：描述FMEDA方法和FMEDA中支持的安全机制，包括故障率的来源、故障模式和分析过程中的假设。

S32K1xx系列是一种SafeAssure™解决方案。有关恩智浦功能安全的更多信息，请访问http://www.nxp.com/safeassure。

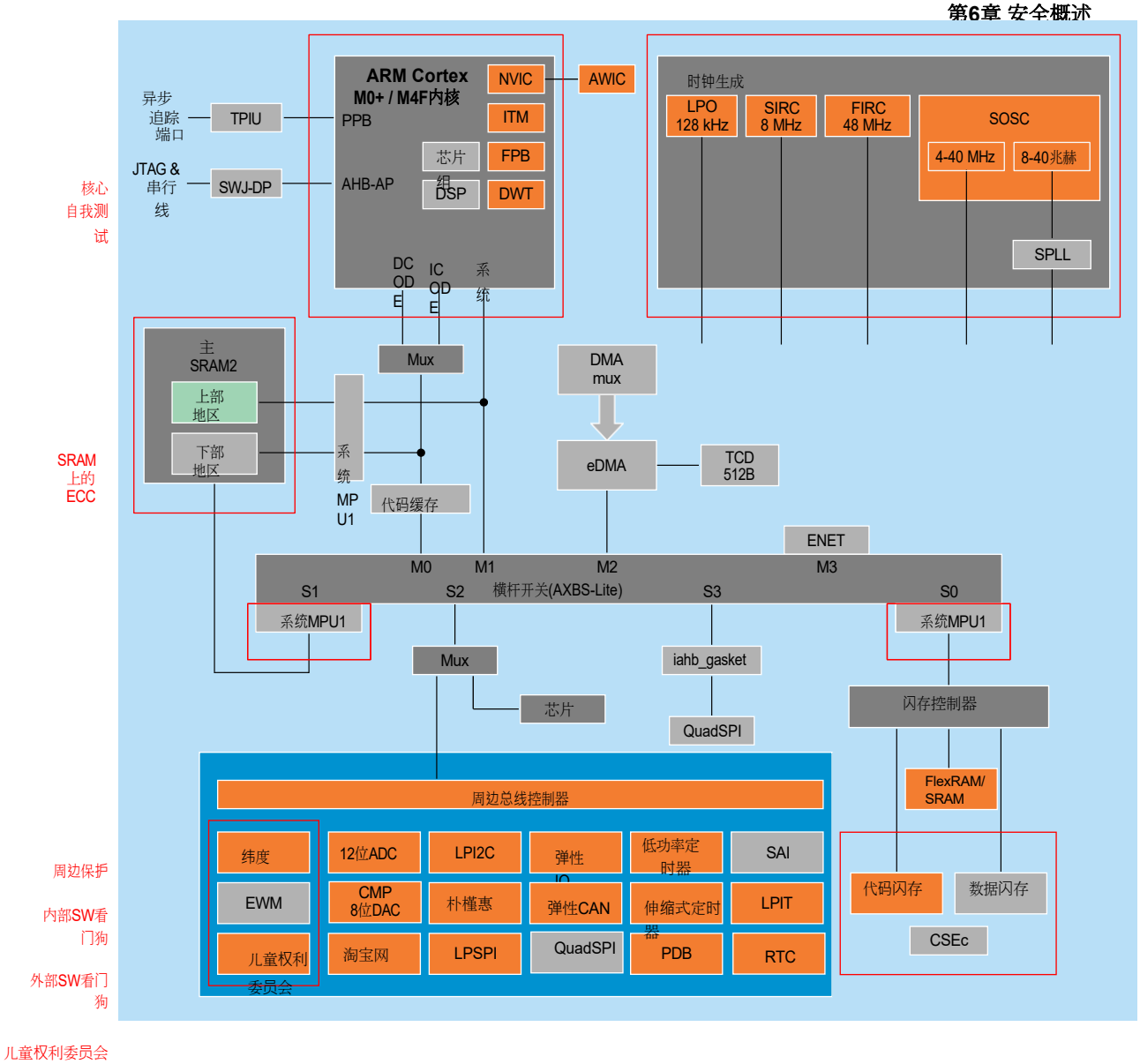



**图6-1.功能安全概述**

## 6.2 S32K1xx安全 概念

S32K1xx系列有一个集成的安全概念，针对需要ASIL-B安全完整性级别的安全相关系统。一般来说，安全完整性是通过使用和应用《安全手册》中描述的S32K1xx安全特性来实现的。

下图提供了一个集成的S32K1xx架构和安全功能的概述。




注册保护　　数字模拟信号路径的多样性　　沟通渠道的多样性

1：在该器件上，恩智浦的系统MPU实现了安全机制，以防止主控器访问受限的内存区域。该系统MPU在Crossbar开关的层面上提供内存保护。每个Crossbar主控器（Core、DMA、Ethernet）可以对每个受保护的内存区域分配不同的访问权。该系列中的ARM M4内核版本没有
集成了ARM Core MPU，它将同时监测仅由内核启动的内存访问。在本文件中，术语MPU指的是恩智浦的系统MPU。

钥匙。

所有S32K设备上的设备架构IP。

所有S32K设备上的外设。

存在于选定的S32K设备上的外设。请参阅 "功能比较 "一节。

2: 请参阅S32K14x系列参考手册中的存储器和存储器接口一章。器件特定尺寸的片上SRAM尺寸表

**图6-2.S32K1xx安全框图**

### 6.2.1 Cortex-M4/M0+结构核心自测试 （SCST）

Cortex-M4/M0+结构性内核自检（SCST）是恩智浦的一个软件产品。它是为检测内核中的硬件永久性故障而开发的，方法是用一组固定的操作数执行机器操作码并比较其执行结果。这个库被认为是一个脱离了背景的安全元件，是根据ASIL-B开发的。

SCST交付包括SCST库、一个质量包和一个安全包。

- 质量包包含代码覆盖率分析、MISRA报告、软件需求规范和测试规范。
- 安全包包括SCST故障覆盖率估算、安全分析和概念，以及SCST安全手册。

结构化内核自检库的安全手册包含了一份建议和假设清单，用户在正确使用Cortex-M4/M0+内核的SCST库时应该履行和验证。

### 6.2.2 内存和闪存上的ECC 内存

纠错码用于保护从CPU内核到闪存以及从CPU内核到系统内存的传输。纠错码（ECC）是用单次纠错、双次检测（SECDED）实现的。

参考文献。

- 功能描述：在本参考手册中，见MCM、EIM、ERM、LMEM和FTFC。
- 安全概念中的RAM和闪存的ECC：见安全手册

### 6.2.3 电源 监测

S32K1xx包括一个管理低电压条件的系统，以便在电源电压变化期间保护存储器内容和控制MCU系统状态。

- 通电复位（POR）
- 低电压检测（LVD）。

参考文献。

- 功能描述：在本参考手册中，见电源管理
- 安全概念中的电源监控：见安全手册章节

### 6.2.4 时钟 监测

S32K1xx中的时钟是由时钟监控单元监督的。

- 系统PLL时钟监控器：监控PLL时钟的损失
- 系统振荡器（SOSC）时钟监控：监控振荡器时钟的损失
- CMU: 显示器
  - 丢失时钟
  - 不在高、低档范围内

参考文献。
- 功能描述：在本参考手册中，见时钟分配
- 安全概念中的时钟监测：见安全手册章节

### 6.2.5 时时彩平台代理 保护

在安全概念中，看门狗定时器提供时间上的保护，并通过期待软件的定期通信来监控系统的运行。S32K1xx提供两个看门狗定时器。

- 看门狗定时器（WDOG）。一个独立的定时器，可供系统使用。它提供了一个安全功能，以确保软件按计划执行，CPU不会卡在无限循环中。
- 外部看门狗监视器（EWM）。一个冗余的看门狗系统，用于安全。EWM提供一个独立的输出信号。它不会重置MCU和外围设备。

参考文献。
- 功能描述：在本参考手册中，见WDOG和EWM
- 安全概念中的时间保护：见安全手册章节

### 6.2.6 操作干扰 保护

S32K1xx提供安全机制，以。

- 防止非安全大师干扰安全核心的运作
- 管理具有不同（较低）ASIL的软件的同时执行。

一个分层的内存保护方案，其中包括以下内容，可以防止干扰。

- 系统内存保护单元（MPU）
- 外围桥（AIPS-Lite）
- 注册保护

### 6.2.6.1 系统内存保护单元（MPU）

对于ASIL-
B应用，系统内存保护单元（MPU）被用于执行控制。它分配访问权限，确保只有授权的软件任务才能配置模块和访问其分配的资源。

- 系统MPU1在横梁开关处提供内存保护。它将物理内存分割成16个不同的区域。
- 每个XBAR主站（核心、DMA、以太网）可以被分配到每个区域的不同访问权限。
- 系统MPU可以用来防止非安全主控器（包括DMA或以太网控制器）访问受限的内存区域。

参考文献。
- 功能描述：在本参考手册中，见MPU
- 安全概念中的MPU：见安全手册章节

### 6.2.6.2 周边保护

外设保护是基于外设桥（AIPS-
Lite）模块的。它通过定义总线主站对芯片上各种外设的访问权限来提供内存保护功能。

参考文献。
- 功能描述：在本参考手册中，见AIPS-Lite
- 安全概念中的外围保护：见安全手册章节

1. 在该器件上，恩智浦的系统MPU实现了安全机制，以防止主控器访问受限的内存区域。该系统MPU在Crossbar开关

的层面上提供内存保护。每个Crossbar主控器（Core、DMA、Ethernet）都可以对每个受保护的内存区域分配不同的访问权。该系列中的ARM M4内核版本没有集成ARM内核MPU，它将同时监测只有内核发起的内存访问。在本文件中，术语MPU指的是恩智浦的系统MPU。

#### 6.2.6.3 注册保护

S32K1xx为安全关键寄存器提供寄存器保护。这种保护是基于CPU程序执行模式--用户与主管（特权）模式，以及另一种基于对某些寄存器实施的锁功能的访问限制。基于锁的访问限制抑制了寄存器的更新，直到下一次系统复位或需要一个特殊的钥匙来解锁访问。

参考文献。

- 功能描述：在本参考手册中，参见SIM、SCG、ERM、RCM、MSCM等寄存器描述部分中关于特定寄存器或寄存器组的寄存器保护功能的说明。
- 安全概念中的注册保护：见安全手册

### 6.2.7 儿童权利委员会

CRC单元通过计算其CRC签名并将其与先前计算的CRC相比较，支持检测内存或配置寄存器中数据的意外更改。CRC模块还可以检测传输或存储过程中数据的错误损坏。

参考文献。

- 功能描述：在本参考手册中，见CRC
- 安全概念中的CRC：见《安全手册》章节

### 6.2.8 系统的多样性 资源

与功能安全相关的功能通常在系统中都有冗余的支持。S32K1xx提供多样化的系统资源来提供这种支持。

- 数字输入可被复制，以冗余地获得安全关键输入
- 安全关键的数字输出总是可以被冗余写入或与回读操作相结合
- 模拟输入可以在超采样模式下运行，以检测影响ADC通道的瞬时故障，两个ADC单元可以获取连接到安全相关信号的模拟信号的冗余副本并将其数字化。
- 对于出于安全原因的通信通道冗余的情况，S32K14x提供通信外设的冗余实例。
  - 同步串行通信控制器（LPSPI）模块

- FlexIO模块：能够支持多种协议（UART、I2C、SPI、I2S）和PWM/波形生成
- UART模块：支持UART和LIN通信

参考文献。

- 功能描述：在本参考手册中，见SIM、LPSPI、LPI2C、FlexIO和LPUART
- 安全概念中的系统资源多样性：见《安全手册》。

# 第七章 核心 概述

## 7.1 ARM Cortex-M4F内核 配置

本节总结了该模块在芯片 中的配置方式。该模块的完整文档由ARM提供， 可在**arm.com**上找到。

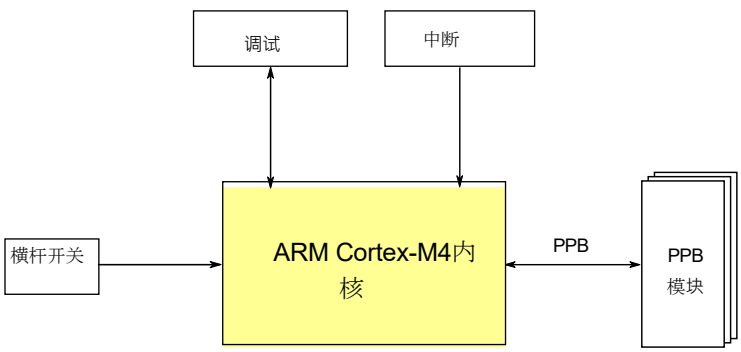



**图7-1.核心配置**

**表7-1.相关信息的参考链接**

| 主题 | 相关模块 | 参考资料 |
|---|---|---|
| 完整描述 | ARM Cortex-M4F内核 | ARM Cortex-M4F技术参考手册 |
| 系统内存图 | - | 见本文件所附的S32K1xx_memory_map.xlsx。 |
| 计时 | - | 时钟分布 |
| 电源管理 | - | 电源管理 |
| 系统/指令/数据总线模块 | 横杆开关 | 横杆开关 |
| 调试 | IEE 1149.1 JTAG<br>串行线调试（SWD）<br>ARM实时跟踪接口 | 调试 |
| 中断 | 嵌套式矢量中断控制器（NVIC） | NVIC |

表继续在下一页...

表7-1.相关信息的参考链接（续）

| 主题 | 相关模块 | 参考资料 |
| --- | --- | --- |
| 专用外围总线（PPB）模块 | 杂项控制模块（MCM） | MCM |
| 专用外围总线（PPB）模块 | 单精度浮点单元（FPU） | 芯片组 |

## 7.1.1 总线、互连和 接口

ARM Cortex-M4内核有四条总线，如下表所述。

表7-2.ARM核心总线

| 巴士名称 | 描述 |
| --- | --- |
| 指令代码（ICODE）总线 | ICODE和DCODE总线是混用的。这个混杂的总线被称为CODE总线，通过一个主端口连接到横梁开关。 |
| 数据代码（DCODE）总线 | |
| 系统总线 | 系统总线连接到横梁上一个单独的主端口。 |
| 私人外围（PPB）总线 | PPB提供对这些模块的访问。<br>• ARM模块，如NVIC、ITM、DWT、FBP和ROM表<br>• 恩智浦杂项控制模块（MCM） |

## 7.1.2 系统滴答 计时器

系统定时器的时钟源始终是核心时钟CORE_CLK。这就导致了以下情况。

- SysTick**控制和状**态寄存器中的CLKSOURCE位总是被设置来选择核心时钟。
- 由于定时参考（CORE_CLK）是一个可变的频率，SysTick校准值寄存器中的TENMS位总是为零。
- SysTick校准值寄存器中的NOREF位总是被设置，这意味着CORE_CLK是唯一可用的参考定时源。

## 7.1.3 调试 设施

该芯片具有广泛的调试功能，包括运行控制和跟踪功能。这是一个标准的ARM调试端口，支持JTAG和SWD接口。

### 7.1.4 缓存

该器件包括一个4KB的代码缓存，以尽量减少内存访问延迟的性能影响。代码高速缓存存在于I/D总线上，系统总线上没有高速缓存。

缓存的特点是。

- 2路集合关联性
- 4字行
- 线路可以单独冲洗
- 整个缓存可以一次性刷新

#### 7.1.4.1 控制

为了控制目的，缓存可以处于这些状态之一。

1. 回写/写入分配（WBWA）
2. 写透
3. 没有缓存

对于每个定义的区域，在控制寄存器上会分配2个比特（见PCCRMR，它决定了与此部分相关的内存区域的缓存状态。鉴于WBWA > WT > NC的固定关系，用户只能 "降低 "缓存属性--
所以，你可以把一个WBWA区域降为WT或NC，你可以把一个WT空间降为NC。为了向上改变状态，需要进行系统重置。

**注意事项**

缓存重置状态见LMEM。

### 7.1.5 核心权限 水平

ARM文档使用与本文档不同的术语来区分权限级别。

**表7-3.使用的术语**

| 如果你看到这个术语... | 它也意味着这个术语... |
|---|---|
| 享有特权的 | 监事长 |
| 无权者或用户 | 用户 |

## 7.2 嵌套的矢量中断控制器（NVIC） 配置

本节总结了该模块在芯片中的配置方式。该模块的完整文档由ARM提供，可在**arm.com**上找到。

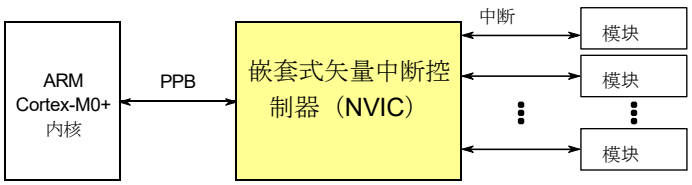


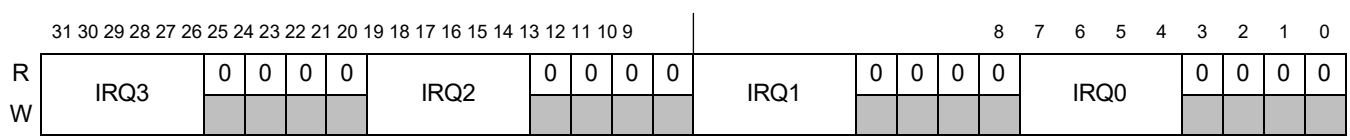

**图7-2.NVIC配置**

**表7-4.相关信息的参考链接**

| 主题 | 相关模块 | 参考资料 |
|---|---|---|
| 完整描述 | 嵌套式矢量中断控制器（NVIC） | ARM Cortex-M4F技术参考手册 - 嵌套的矢量中断控制器 |
| 系统内存图 | - | 请参考本文件所附的S32K1xx_memory_map.xlsx。 |
| 计时 | - | 时钟分布 |
| 电源管理 | - | 电源管理 |
| 私人外围总线（PPB） | ARM Cortex-M4内核 | ARM Cortex-M4F 技术参考手册 - 专用外设总线（PPB） |

### 7.2.1 中断优先级 级

该设备支持16个中断的优先级。因此，在NVIC中，IPR寄存器中的每个源包含4位。例如，IPR0的图示如下。




### 7.2.2 无任务的 中断

对NVIC的非屏蔽中断请求是由外部NMI信号控制的。NMI信号被复用的引脚必须

被配置为NMI功能，以便产生非屏蔽中断请求。

关于NMI垫的详情，请参见《参考手册》所附的IO信号描述输入复用表。

### 7.2.3 确定位域和寄存器的位置，以配置一个特定的中断

假设你需要配置低功耗定时器（LPTMR）的中断。下表是本参考手册所附文件'S32K1xx_DMA_INT_mapping.xlsm'中LPTMR行的摘录。

**表7-6.LPTMR中断向量分配**

| 地址 | 矢量 | IRQ1 | NVIC非IPR注册号2 | NVIC知识产权注册号3 | 来源模块 |
|---|---|---|---|---|---|
| 0x0000_0128 | 74 | 58 | 1 | 14 | 低功率定时器 |

1. 表示NVIC的中断源编号。
2. 表示该IRQ使用的NVIC的ISER、ICER、ISPR、ICPR和IABR寄存器编号。计算该值的公式为：。IRQ除以32
3. 表示该IRQ使用的NVIC的IPR寄存器编号。计算该值的公式为：。IRQ除以4

你用来配置中断的NVIC寄存器是。

- **燕京大学**
- 镍钴锰酸钾（NVICER）1
- 辽宁沈阳
- 辽宁沈阳
- 镍钴锰酸锂(NVICIABR1)
- NVICIPR14

要确定特定IRQ在这些特定寄存器中的位域位置。

- NVICISER1, NVICICER1, NVICISPR1, NVICICPR1, NVICIABR1位位置 = IRQ mod 32 = 26
- NVICIPR14位域的起始位置=8×（IRQ mod 4）+4=20

  由于NVICIPR位域是4位宽（16个优先级），NVICIPR14位域范围为20-23

因此，下列位域位置被用来配置LPTMR中断。

- NVICISER1[26]
- 薇信1[26]
- NVICISPR1[26]
- 镍钴锰酸钾1[26]。

- NVICIABR1[26]
- nvicipr14[23:20]

## 7.3 异步唤醒中断控制器（AWIC） 配置

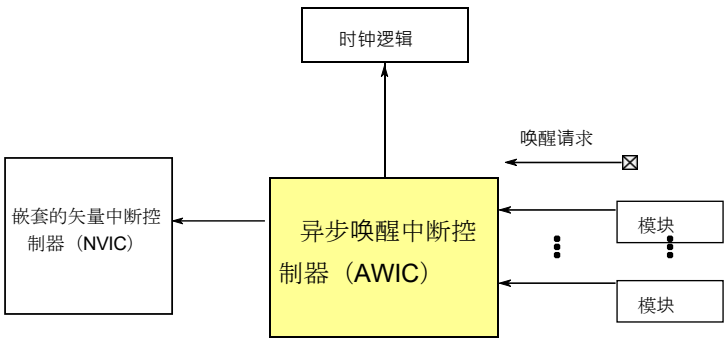



**图7-3.异步唤醒中断控制器配置 表7-7.相关信息的参考链接**

| 主题 | 相关模块 | 参考资料 |
|---|---|---|
| 系统内存图 | - | 详情请参考《参考手册》所附的S32K1xx_memory_map.xlsx。 |
| 计时 | - | 时钟分布 |
| 电源管理 | - | 电源管理 |
| 嵌套式矢量中断控制器（NVIC） | - | NVIC |
| 唤醒请求 | - | AWIC 唤醒源 |

### 7.3.1 唤醒 来源

**表7-8.AWIC停止和VLPS唤醒来源**

| 唤醒源 | 描述 |
|---|---|
| 可用的系统复位 | RESET引脚，WDOG，JTAG |
| 引脚中断 | 端口控制模块 - 任何启用的引脚中断都能够唤醒系统 |
| ADCx | (1) STOP1中的异步中断 (2) STOP2中的同步中断 |
| CMP | (1) STOP1中的异步中断 (2) STOP2中的同步中断 (3) 在VLPS模式下起作用，将导致异步中断的唤醒 |
| LPI2C0 | (1) STOP1中的异步中断 (2) STOP2中的同步中断 |
| 朴槿惠 | (1) STOP1中的异步中断 (2) STOP2中的同步中断 |

*表继续在下一页...*

表7-8.AWIC停止和VLPS唤醒源（续）

| 唤醒源 | 描述 |
|---|---|
| LPSPI | (1) STOP1中的异步中断 (2) STOP2中的同步中断 |
| 呼叫中心 | (1) STOP1中的异步中断 (2) STOP2中的同步中断 (3) 在VLPS模式下起作用，将导致异步中断的唤醒 |
| RTC | (1) STOP1中的异步中断 (2) STOP2中的同步中断 |
| CAN | PNET在STOP1/2模式下被支持，并会导致唤醒。只有CAN0支持PNET功能。 |
| NMI | 无任务的中断 |
| 纬度 | (1) STOP1中的异步中断 (2) STOP2中的同步中断 (3) 在VLPS模式下起作用，将导致异步中断的唤醒 |
| 弹性IO | (1) STOP1中的异步中断 (2) STOP2中的同步中断 |
| LPIT | (1) STOP1中的异步中断 (2) STOP2中的同步中断 |
| EWM | 在STOP1和VLPS中模块关闭。可以通过STOP2中的同步中断引起唤醒。 |
| 儿童权利委员会 | 在STOP1和VLPS中模块关闭。可以通过STOP2中的同步中断引起唤醒。 |
| 渣打集团 | (1) STOP1中的异步中断 (2) STOP2中的同步中断 |

## 7.4 FPU 配置

本节总结了该模块在芯片中的配置情况。

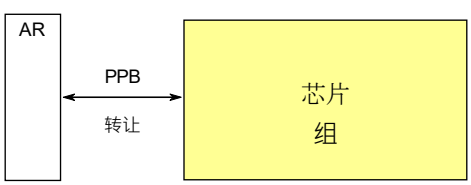



图7-4.FPU配置

表7-9.相关信息的参考链接

| 主题 | 相关模块 | 参考资料 |
|---|---|---|
| 完整描述 | 芯片组 | ARM Cortex-M4技术参考手册 - 浮点单元 |
| 系统内存图 | - | 请参考本文件所附的S32K1xx_memory_map.xlsx。 |
| 计时 | - | 时钟分布 |
| 电源管理 | - | 电源管理 |
| 转移<br>私人外围总线（PPB） | ARM Cortex M4内核 | ARM Cortex-M4技术参考手册 - 专用外设总线 |

# 7.5 JTAG控制器 配置

本节总结了该模块在芯片中的配置情况。

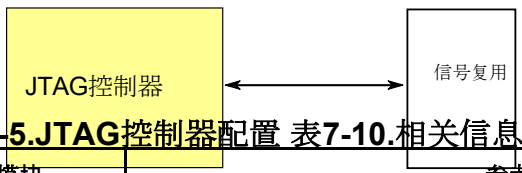



图7-5.JTAG控制器配置 表7-10.相关信息的参考链

| 主题 | 相关模块 | 参考资料 |
| --- | --- | --- |
| 完整描述 | 接 JTAGC | JTAGC |
| 信号复用 | 港口控制 | 详见本文件所附《参考手册》中的IO信号描述输入复用表。 |

# 第八章
# 杂项控制模块（MCM）

## 8.1 特定芯片的MCM 信息

下表总结了该模块在产品系列中各芯片的特定寄存器复位值。

**表8-1.MCM寄存器的复位值**

| 注册 | S32K116 | S32K118 | S32K142 | S32K144 | S32K146 | S32K148 |
|---|---|---|---|---|---|---|
| 辽宁沈阳 | 8104_0000 | 8104_0000 | 8504_0003 | 8604_0003 | 0x8704_0003 | 8804_0003 |
| 锂离子电池组（LMDR1 | 8504_2000 | 9604_2000 | 8504_2003 | 8604_2003 | 0x8704_2003 | 8804_2003 |
| 辽宁沈阳 | 这是对的。 | 这是对的。 | 8424_40A0 | 8424_40A0 | 8424_40A0 | 8424_40A0 |
| PLASC | 0007 | 0007 | 0007 | 0007 | 0007 | 000F |
| PLAMC | 0005 | 0005 | 0007 | 0007 | 0007 | 000F |

## 8.2 简介

杂项控制模块（MCM）提供杂项控制功能。S32K14x上不支持高速缓存写缓冲器。

### 8.2.1 特点

MCM包括以下功能。

- 关于平台配置和修订的程序可见的信息

## 8.3 内存图/寄存器 描述

下面的内存图和寄存器描述了杂项控制模块的寄存器。

**注意事项**

- 所有的寄存器都只能在监督员模式下访问。用户模式的访问将产生一个错误。
- 写入只读的MCM_PLASC和MCM_PLAMC寄存器，将产生一个总线错误。

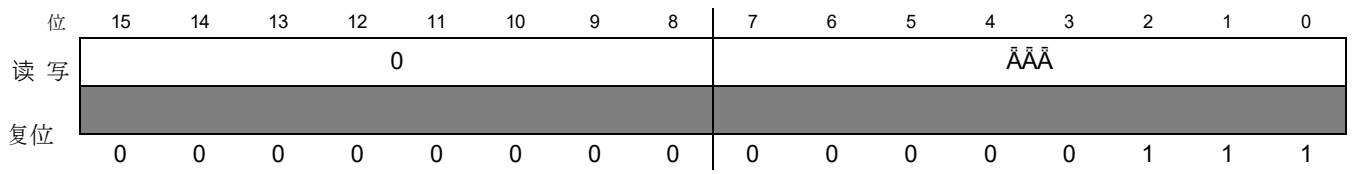
**MCM内存图**

| 地址偏移（十六进制） | 注册名称 | 宽度（单位：比特） | 访问 | 重置值 | 节/页 |
|---|---|---|---|---|---|
| 8 | 横梁开关（AXBS）从属配置（MCM_PLASC）。 | 16 | R | 0007h | 8.3.1/113 |
| A | 横梁开关（AXBS）主配置（MCM_PLAMC）。 | 16 | R | 0007h | 8.3.2/113 |
| C | 核心平台控制寄存器（MCM_CPCR）。 | 32 | R/W | 见本节 | 8.3.3/115 |
| 10 | 中断状态和控制寄存器（MCM_ISCR）。 | 32 | R | 0002_0000h | 8.3.4/118 |
| 30 | 进程ID寄存器（MCM_PID）。 | 32 | R/W | 0000_0000h | 8.3.5/121 |
| 40 | 计算操作控制寄存器（MCM_CPO）。 | 32 | R/W | 0000_0000h | 8.3.6/122 |
| 400 | 本地内存描述符寄存器（MCM_LMDR0）。 | 32 | R/W | 见本节 | 8.3.7/123 |
| 404 | 本地内存描述符寄存器（MCM_LMDR1）。 | 32 | R/W | 见本节 | 8.3.7/123 |
| 408 | 本地内存描述符寄存器2（MCM_LMDR2）。 | 32 | R/W | 8424_40A0h | 8.3.8/126 |
| 480 | LMEM奇偶校验和ECC控制寄存器（MCM_LMPECR）。 | 32 | R/W | 0000_0000h | 8.3.9/130 |
| 488 | LMEM奇偶校验和ECC中断寄存器（MCM_LMPEIR）。 | 32 | R/W | 0000_0000h | 8.3.10/130 |
| 490 | LMEM故障地址寄存器（MCM_LMFAR）。 | 32 | R | 0000_0000h | 8.3.11/132 |
| 494 | LMEM故障属性寄存器（MCM_LMFATR）。 | 32 | R | 0000_0000h | 8.3.12/133 |
| 4A0 | LMEM故障数据高位寄存器（MCM_LMFDHR）。 | 32 | R | 0000_0000h | 8.3.13/134 |
| 4A4 | LMEM故障数据低寄存器（MCM_LMFDLR）。 | 32 | R | 0000_0000h | 8.3.14/134 |

## 8.3.1 横梁开关（AXBS）从属配置 （MCM_PLASC）。

PLASC是一个16位的只读寄存器，用于识别总线从属连接到设备的横梁开关的存在/不存在。

地址。0h基数+8h偏移量=8h



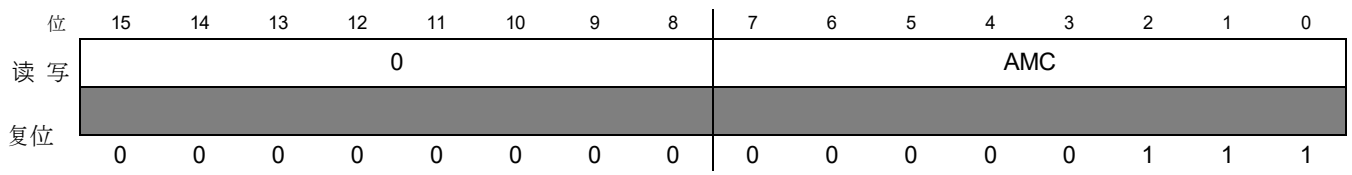

**MCM_PLASC字段描述**

| 场地 | 描述 |
|---|---|
| 15-8<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| ÃÃÃ | ASC字段中的每一位都表示是否有相应的连接到横梁开关的从属输入端口。<br><br>0 没有与AXBS输入端口*n*的总线从属连接<br>1 存在一个连接到AXBS输入端口*n*的总线从属连接 |

## 8.3.2 横梁式交换机（AXBS）主配置 （MCM_PLAMC）。

PLAMC是一个16位的只读寄存器，用于识别与设备的横梁开关的总线主控连接的存在/不存在。

地址。0h基数+Ah偏移量=Ah




**MCM_PLAMC字段描述**

| 场地 | 描述 |
|---|---|
| 15-8<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |

| | |
|---|---|
| AMC | AMC字段中的每一位都表示是否有相应的连接到AXBS主输入端口。 |

*表继续在下一页...*

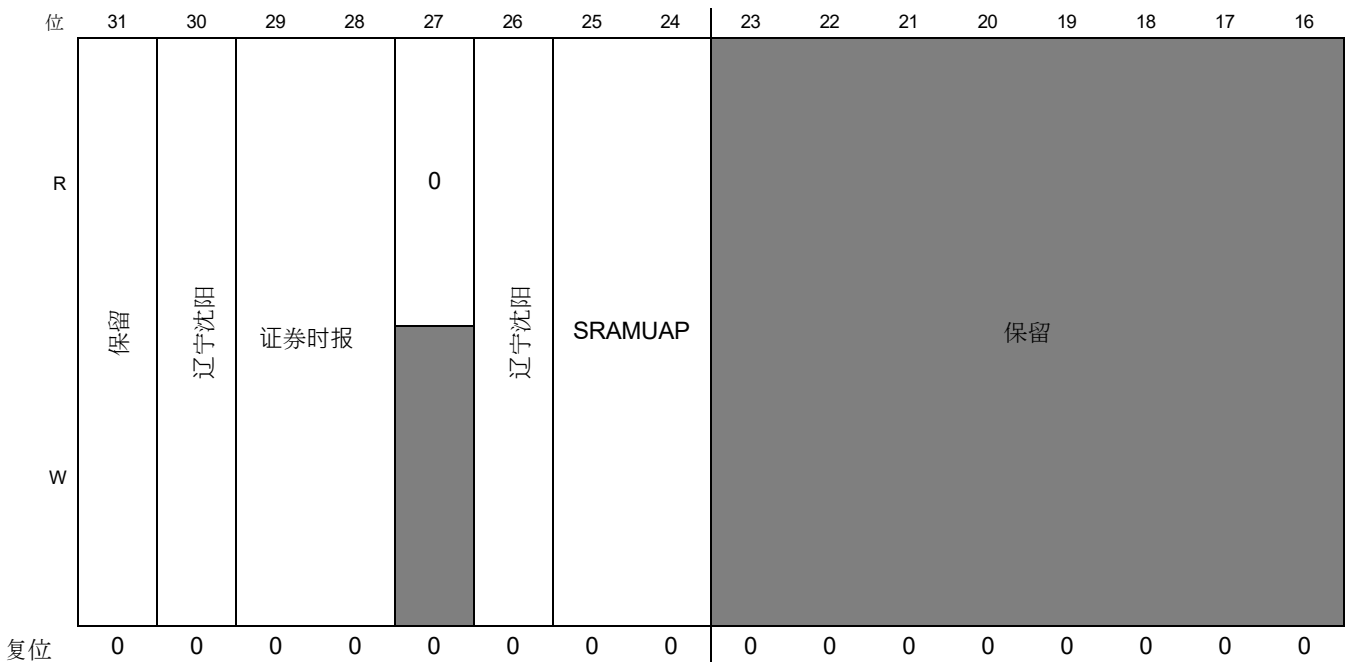
## MCM_PLAMC字段描述（续）

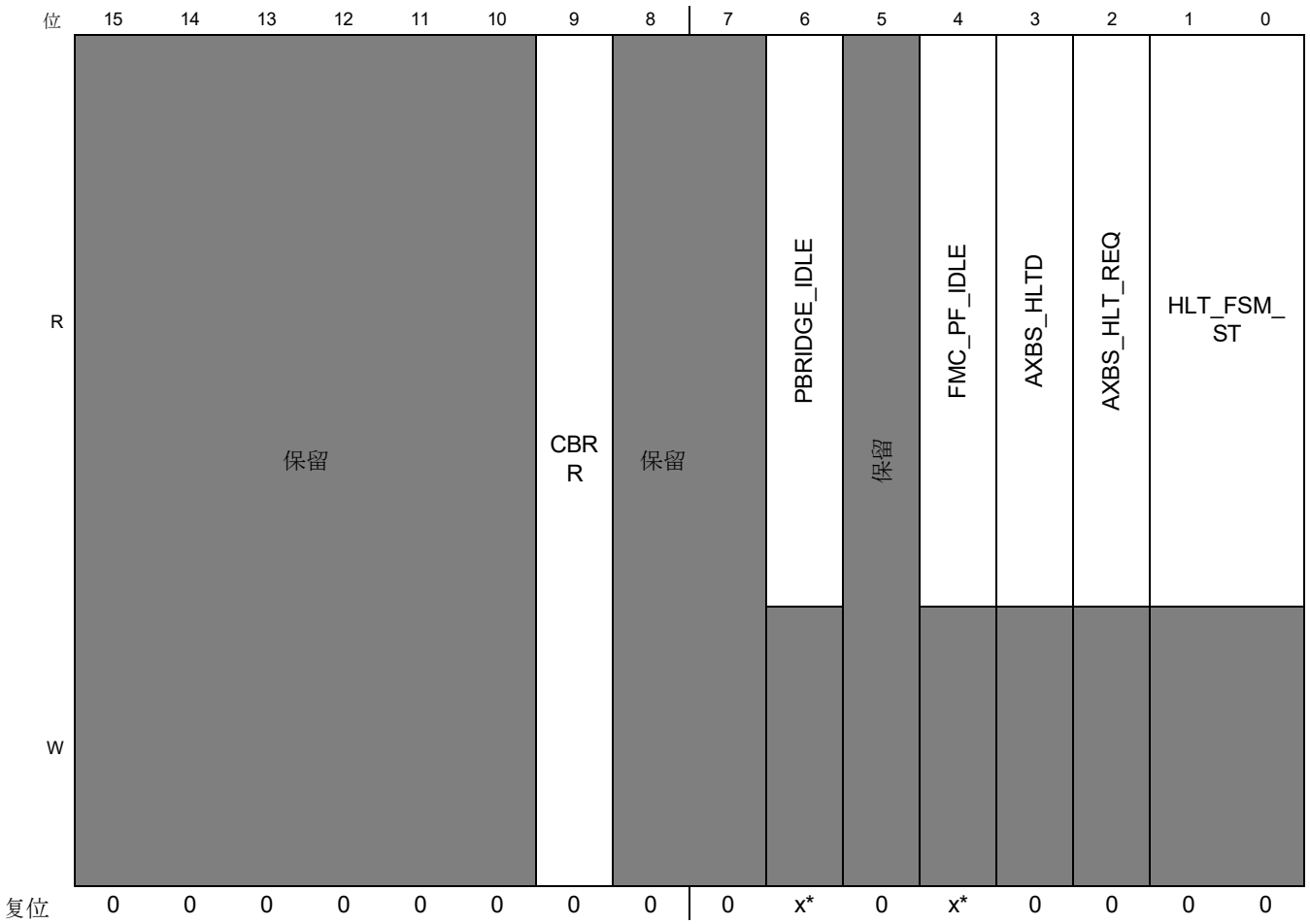
| 场地 | 描述 |
|---|---|
| | 0 没有与AXBS输入端口*n*的总线主连接<br>1 存在一个连接到AXBS输入端口*n*的总线主站 |

## 8.3.3 核心平台控制寄存器 （MCM_CPCR）。

CPCR 定义了两个系统 RAM 阵列的仲裁和保护方案。

地址。0h基数+Ch偏移量=Ch






* 注意事项。

- x = 重置时未定义。

## MCM_CPCR字段描述

| 场地 | 描述 |
|---|---|
| 31<br>保留 | 这个字段是保留的。 |
| 30<br>辽宁沈阳 | SRAM_L写保护<br><br>当这个字段被设置时，对SRAM_L阵列的写入会产生一个总线错误。 |
| 29-28<br>证券时报 | SRAM_L 仲裁优先级<br><br>定义了处理器和SRAM后门访问SRAM_L阵列的仲裁方案和优先级。<br><br>00 循环赛<br>01 特殊的轮回（偏向于SRAM后门访问，而不是处理器）。<br>10 固定优先级。处理器的优先级最高，后门的优先级最低<br>11 固定优先级。后门是最高的，处理器是最低的 |
| 27<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |

表继续在下一页...

## MCM_CPCR字段描述（续）

| 场地 | 描述 |
|---|---|
| 26<br>辽宁沈阳 | SRAM_U写保护<br><br>当这个字段被设置时，对SRAM_U阵列的写入会产生一个总线错误。 |
| 25-24<br>SRAMUAP | SRAM_U 仲裁优先级<br><br>定义了处理器和SRAM后门访问SRAM_U阵列的仲裁方案和优先级。<br><br>00 循环赛<br>01 特殊的轮回（偏向于SRAM后门访问，而不是处理器）。<br>10 固定优先级。处理器的优先级最高，后门的优先级最低<br>11 固定优先级。后门是最高的，处理器是最低的 |
| 23-10<br>保留 | 这个字段是保留的。 |
| 9<br>CBRR | 启用横梁式轮流仲裁<br><br>将横梁从属端口配置为固定优先级或轮流仲裁。<br><br>0 固定优先级的仲裁<br>1 循环仲裁 |
| 8-7<br>保留 | 这个字段是保留的。 |
| 6<br>PBRIDGE_IDLE | 外围网桥闲置<br><br>这个字段表示外围桥是否空闲。<br><br>0 PBRIDGE并没有闲置<br>1 PBRIDGE目前处于闲置状态 |
| 5<br>保留 | 这个字段是保留的。 |
| 4<br>FMC_PF_IDLE | 闪存控制器编程 闪存闲置<br><br>这个字段表示闪存的程序部分是否处于空闲状态。<br><br>0 FMC程序闪存没有闲置<br>1 FMC程序闪存目前处于空闲状态 |
| 3<br>AXBS_HLTD | AXBS停止了<br><br>该字段表示AXBS是否处于停止状态。<br><br>0 AXBS目前没有被停止<br>1 AXBS目前已停止运行 |

*表继续在下一页...*

| | |
|---|---|
| 2<br>AXBS_HLT_REQ | AXBS停止请求<br><br>这个字段表示AXBS是否收到了一个停止请求。<br><br>0 AXBS没有收到停顿请求<br>1 AXBS正在接收停顿请求 |
| HLT_FSM_ST | AXBS停止状态机状态<br><br>该字段表示AXBS停顿的状态。 |

MCM_CPCR字段描述（续）

| 场地 | 描述 |
|---|---|
| | 00 等待请求<br>01 等待平台闲置<br>11平台停滞不前<br>10未使用状态 |

## 8.3.4 中断状态和控制寄存器 (MCM_ISCR)

ISCR定义了一些与内核有关的中断异常条件的配置和报告状态。它包括与内核的浮点异常相关的使能和状态字段，以及与内核的高速缓存写缓冲器相关的总线错误。各个事件指示器首先与它们的异常使能一起被限定，然后在逻辑上相加，形成一个发送到内核NVIC的中断请求。

第15-
8位是基于处理器FPSCR的只读指示标志。对这些字段的尝试性写入会被忽略。在这些标志被设置后，它们会一直被引出，直到软件清除相应的FPSCR字段。

地址。0h基数+10h偏移量=10h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | FIDCE | 0 | | 锁定 | 保留 | FOFCE | FDZCE | FIOCE | 0 | | | 保留 | 0 | | 1 | 0 |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |

| 位 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | FIDC | 0 | | FIXC | 保留 | FOFC | FDZC | FIOC | 0 | | | 保留 | 0 | | | |
| W | | | | | | | | | | | | w1c | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

复位

## MCM_ISCR字段描述

| 场地 | 描述 |
|---|---|
| 31<br>FIDCE | FPU输入异常中断启用<br><br>0 禁用中断<br>1 启用中断 |
| 30-29<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 28<br>锁定 | FPU非精确中断启用<br><br>0 禁用中断<br>1 启用中断 |
| 27<br>孚孚孚孚孚孚孚孚<br>孚孚孚孚 | FPU下溢中断启用<br><br>0 禁用中断<br>1 启用中断 |
| 26<br>FOFCE | FPU溢出中断启用<br><br>0 禁用中断<br>1 启用中断 |
| 25<br>FDZCE | FPU除以零中断启用<br><br>0 禁用中断<br>1 启用中断 |
| 24<br>FIOCE | FPU无效操作中断启用<br><br>0 禁用中断<br>1 启用中断 |
| 23-21<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 20<br>保留 | 这个字段是保留的。 |
| 19-18<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 17<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为1。 |
| 16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 15<br>FIDC | FPU输入非正常中断状态<br><br>这个字段是内核的FPSCR[IDC]字段的副本，信号是在处理器的FPU中检测到输入的非正常化数字。在这个字段被设置后，它一直被设置，直到软件清除FPSCR[IDC]。<br><br>0 没有中断<br>1 发生中断 |

*表继续在下一页...*

| | |
|---|---|
| 14-13<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 12<br>FIXC | FPU非精确中断状态<br><br>这个字段是内核FPSCR[IXC]字段的副本，表示在处理器的FPU中检测到了一个不精确的数字。一旦被设置，这个字段就一直被设置，直到软件清除FPSCR[IXC]。 |

## MCM_ISCR字段描述（续）

| 场地 | 描述 |
| --- | --- |
| | 0 没有中断<br>1 发生中断 |
| 11<br>孚孚孚孚孚孚孚孚<br>孚孚孚孚 | FPU下溢中断状态<br><br>这个字段是内核FPSCR[UFC]字段的副本，表示在处理器的FPU中已经检测到了一个下溢。在这个字段被设置后，它一直被设置，直到软件清除FPSCR[UFC]。<br><br>0 没有中断<br>1 发生中断 |
| 10<br>FOFC | FPU溢出中断状态<br><br>这个字段是内核FPSCR[OFC]字段的副本，表示在处理器的FPU中检测到了一个溢出。在这个字段被设置后，它会一直被设置，直到软件清除FPSCR[OFC]。<br><br>0 没有中断<br>1 发生中断 |
| 9<br>FDZC | FPU除以零中断状态<br><br>这个字段是内核FPSCR[DZC]字段的副本，它表示在处理器的FPU中已经检测到了除以0。在这个字段被设置后，它一直被设置，直到软件清除FPSCR[DZC]。<br><br>0 没有中断<br>1 发生中断 |
| 8<br>FIOC | FPU无效操作中断状态<br><br>这个字段是内核的FPSCR[IOC]字段的副本，表示在处理器的FPU中检测到了一个非法操作。在这个字段被设置后，它会一直被设置，直到软件清除FPSCR[IOC]。<br><br>0 没有中断<br>1 发生中断 |
| 7-5<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 4<br>保留 | 这个字段是保留的。 |
| 保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |

*表继续在下一页...*

## 8.3.5 过程ID寄存器 （MCM_PID）。

该寄存器驱动内存保护单元（MPU）中的M0_PID和M1_PID值。系统软件在将控制权传递给一个给定的用户模式进程之前加载这个寄存器。如果进程的PID与该寄存器中的值不一致，就会发生总线错误。更多细节请参见MPU章节。

地址。0h基数+30h偏移量=30h

| 位 | 31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 | 15 14 13 12 11 10 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | PID | | | | | | | |
| W | | | | | | | | | | | |
| 复位 | 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 | 0 0 0 0 0 0 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

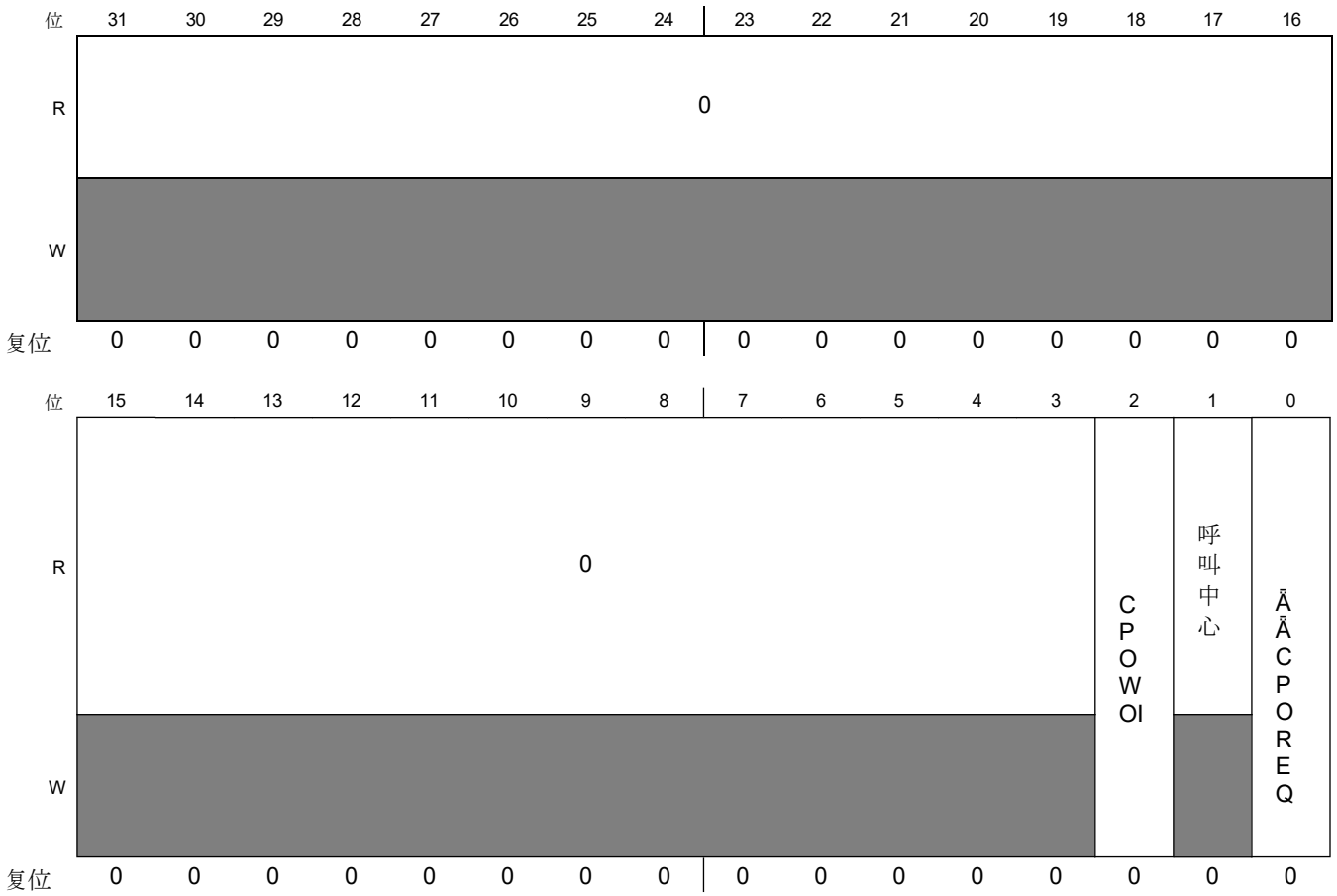
**MCM_PID字段描述**

| 场地 | 描述 |
|---|---|
| 31-8<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| PID | 用于MPU的M0_PID和M1_PID<br><br>驱动MPU中的M0_PID和M1_PID值。 |

## 8.3.6 计算操作控制寄存器 (MCM_CPO)

该寄存器控制计算操作。

地址。0h基数+40h偏移量=40h




**MCM_CPO字段描述**

| 场地 | 描述 |
|---|---|
| 31-3<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 2<br>CPOWOI | 计算操作在中断时唤醒<br>0 没有影响。<br>1 当设置时，CPOREQ在任何中断或异常向量的获取中被清零。 |
| 1<br>呼叫中心 | 计算操作确认<br>0 计算操作进入未完成或计算操作退出已完成。<br>1 计算操作进入已完成或计算操作退出未完成。 |

| | |
|---|---|
| 0<br>ÃÃCPOREQ | 计算操作请求<br>如果CPOWOI=1，这个字段会通过矢量取值自动清除。 |



*表继续在下一页...*

MCM_CPO字段描述（续）

| 场地 | 描述 |
| --- | --- |
| | 0 请求被清除了。<br>1 请求计算操作。 |

## 8.3.7 本地内存描述符寄存器(MCM_LMDRn )

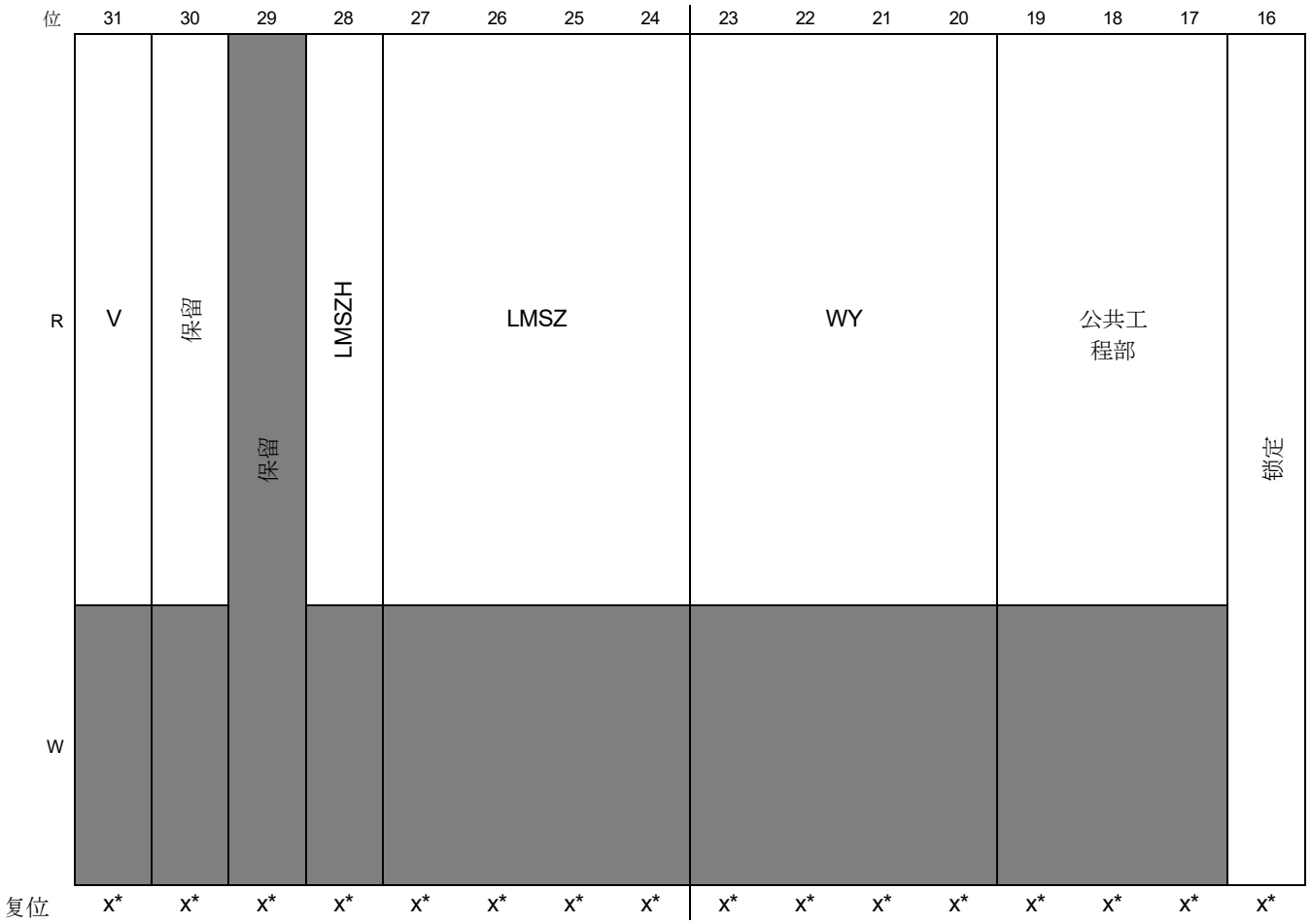
LMDRn寄存器与LMEM的映射关系如下。

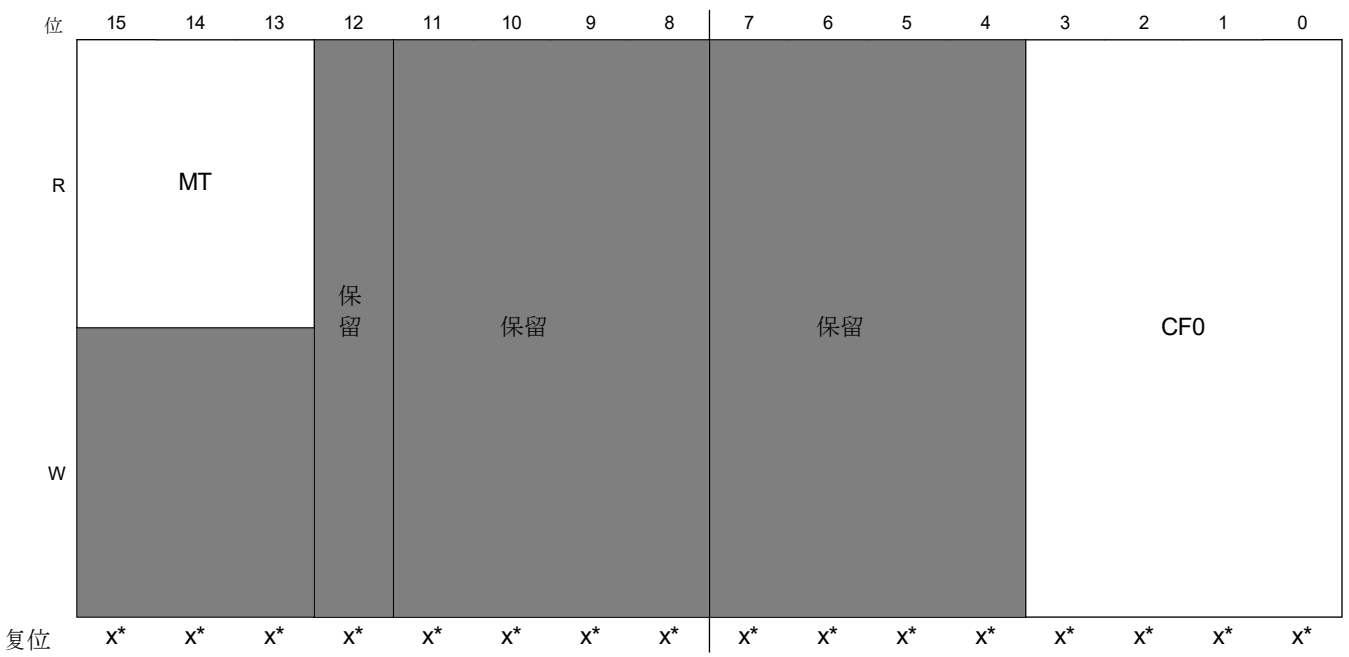
- LMDR0: SRAM_L
- LMDR1: SRAM_U

编程模型的这一部分是一个32位的通用片上存储器描述符寄存器阵列，提供所附存储器的静态信息以及可配置的控制（在适当的地方）。

从处理器内核或调试器的32位特权读数返回适当的处理器信息。从任何其他总线主站的读取会返回所有的零。从处理器内核或调试器到可写寄存器的特权写，更新适当的字段。来自其他总线主站的特权写入会被忽略。试图的用户模式访问或任何大小不超过32位的访问都以错误结束。

地址。0h基数+400h偏移量+（4d×i），其中i=0d至1d




* 注意事项。

- 各个LMDR寄存器的复位值是不同的。LMDR0：0x8804_0003；LMDR1：0x8804_2003。 x = 复位时未定义。

## MCM_LMDRn字段描述

| 场地 | 描述 |
|---|---|
| 31<br>V | 本地内存有效<br><br>这个字段定义了本地存储器的有效性（存在）。<br><br>0 LMEMn不存在。<br>1 LMEMn是存在的。 |
| 30<br>保留 | 这个字段是保留的。 |
| 29<br>保留 | 这个字段是保留的。 |
| 28<br>LMSZH | LMEM尺寸孔<br><br>对于没有完全填充的本地存储器，即在地址范围的上25%包括一个内存孔，这个字段被使用。<br><br>0 LMEMn是一个2次方的容量。<br>1 LMEMn不是2次方，其容量为0.75×LMSZ。 |
| 27-24<br>LMSZ | LMEM尺寸<br><br>这个字段提供了一个本地内存大小的编码值。内存的容量表示为大小[字节]=$2^{(9+LMSZ)}$，其中LMSZ为非零；LMSZ=0表示内存不存在。<br><br>0000no LMEMn (0 KB)<br>00011 KB LMEMn<br>00102 KB LMEMn<br>00114 KB LMEMn<br>01008 KB LMEMn<br>010116 KB LMEMn<br>011032 KB LMEMn<br>011164 KB LMEMn<br>1000128 KB LMEMn<br>1001256 KB LMEMn<br>1010512 KB LMEMn<br>10111024 KB LMEMn<br>11002048 KB LMEMn<br>11014096 KB LMEMn<br>11108192 KB LMEMn<br>111116384 KB LMEMn |
| 23-20<br>WY | 1级缓存方式<br><br>0000No Cache<br>00102-Way Set Associative<br>01004-Way Set Associative |
| 19-17<br>公共工程部 | LMEM数据路径宽度。这个字段定义了本地存储器的宽度。<br><br>000-001 保留 |

*表继续在下一页...*

**MCM_LMDRn字段描述（续）**

| 场地 | 描述 |
|---|---|
| | 10 LMEMn 32位宽<br>11 LMEMn 64位宽 100-<br>111 保留 |
| 16<br>锁定 | 锁定<br>锁定位。<br>该字段提供了一种机制来锁定由LMDRn[7:0]定义的配置状态。一旦被断定，对LMDRn[7:0]寄存器的尝试写入将被忽略，直到下一次复位清除该标志。与 LMDRn[7:0]一起，该位会自我锁定。一旦锁定，只有复位才能清除这个位。<br>0 允许对LMDRn[7:0]进行写入。<br>1 对LMDRn[7:0]的写入被忽略。 |
| 15-13<br>MT | 存储器类型<br>该字段定义了本地存储器的类型。000<br>SRAM_L<br>001 SRAM_U |
| 12<br>保留 | 这个字段是保留的。 |
| 11-8<br>保留 | 这个字段是保留的。 |
| 7-4<br>保留 | 这个字段是保留的。 |
| CF0 | 控制字段0<br>该字段用于TCM ECC控制功能。<br>• CF0[3] - 保留<br>• CF0[2] - 保留<br>• CF0[1] - EERC = ECC启用读取检查<br>• CF0[0] - EEWG = ECC使能写入生成 |

## 8.3.8 本地内存描述符寄存器2 (MCM_LMDR2)

LMDR2寄存器与代表PC CACHE的LMEM的映射。

**注意事项**

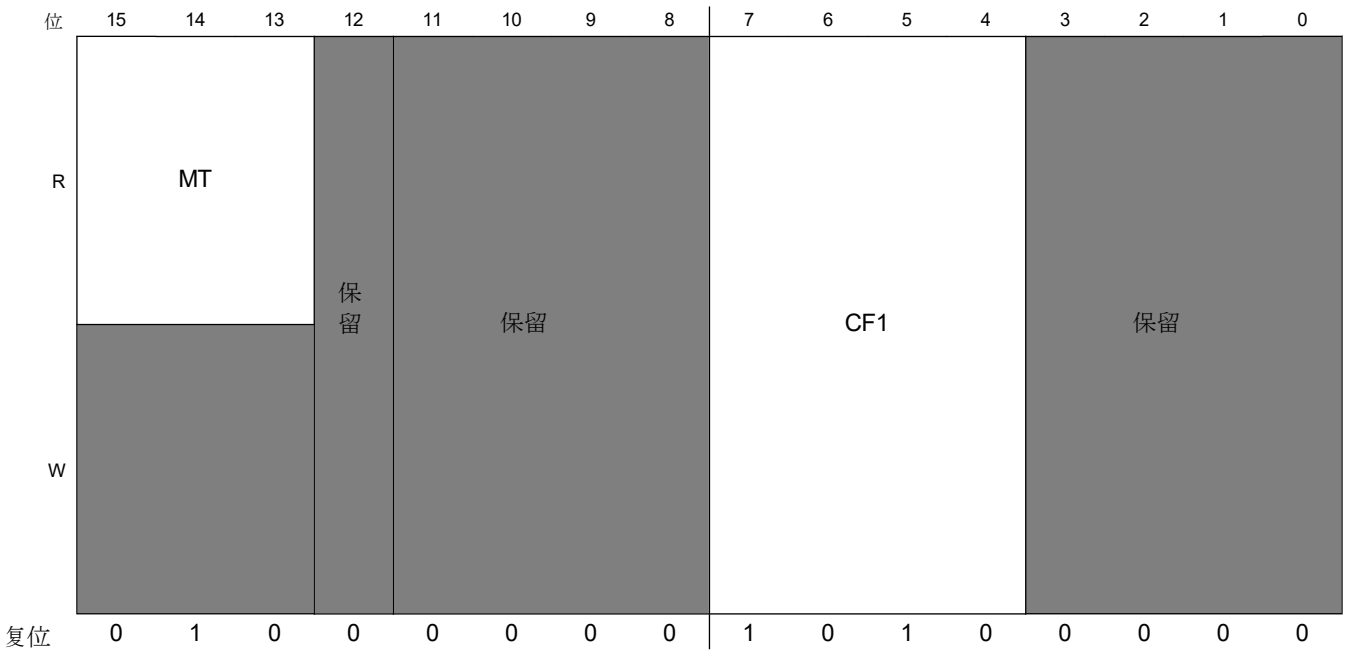
这个值代表了整个系列中最大的可用缓存。参见图2-3，查看特定设备的具体缓存量。

编程模型的这一部分是一个32位的通用片上存储器描述符寄存器阵列，提供所附存储器的静态信息以及可配置的控制（在适当的地方）。

从处理器内核或调试器的32位特权读数返回适当的处理器信息。从任何其他总线主站的读取会返回所有的零。从处理器内核或调试器到可写寄存器的特权写，更新适当的字段。来自其他总线主站的特权写入会被忽略。试图的用户模式访问或任何大小不超过32位的访问都以错误结束。

地址。0h base + 408h offset = 408h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | V | 保留 | 保留 | LMSZH | LMSZ | | | | WY | | | | 公共工程部 | | | 锁定 |
| W | | | | | | | | | | | | | | | | |
| 复位 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |




## MCM_LMDR2字段描述

| 场地 | 描述 |
|---|---|
| 31<br>V | 本地内存有效<br><br>这个字段定义了本地存储器的有效性（存在）。<br><br>0 LMEMn不存在。<br>1 LMEMn是存在的。 |
| 30<br>保留 | 这个字段是保留的。 |
| 29<br>保留 | 这个字段是保留的。 |
| 28<br>LMSZH | LMEM尺寸孔<br><br>对于没有完全填充的本地存储器，即在地址范围的上25%包括一个内存孔，这个字段被使用。<br><br>0 LMEMn是一个2次方的容量。<br>1 LMEMn不是2次方，其容量为0.75×LMSZ。 |
| 27-24<br>LMSZ | LMEM尺寸<br><br>这个字段提供了一个本地内存大小的编码值；LMSZ=0表示内存不存在。<br><br>01004 KB LMEMn |
| 23-20<br>WY | 1级缓存方式<br><br>0000No Cache<br>00102-Way Set Associative<br>01004-Way Set Associative |

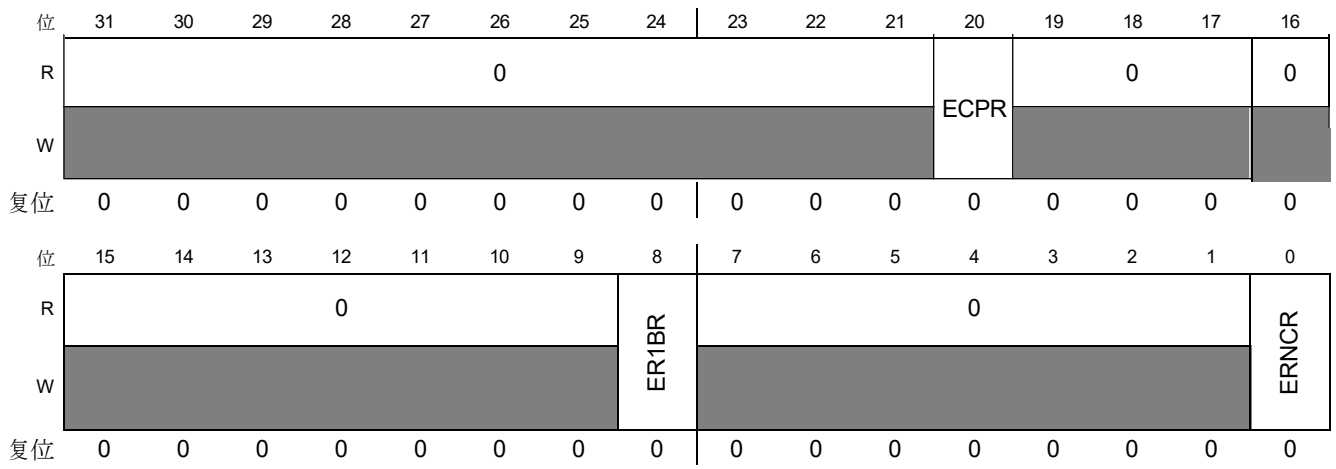
*表继续在下一页...*

## MCM_LMDR2字段描述（续）

| 场地 | 描述 |
|---|---|
| 19-17<br>公共工程部 | LMEM数据路径宽度。这个字段定义了本地存储器的宽度。<br><br>000-001 保留<br>10 LMEMn 32位宽<br>11 LMEMn 64位宽 100-<br>111 保留 |
| 16<br>锁定 | 锁定<br><br>锁定位<br><br>该字段提供了一种机制来锁定由LMDRn[7:0]定义的配置状态。一旦被断定，对LMDRn[7:0]寄存器的尝试写入将被忽略，直到下一次复位清除该标志。与 LMDRn[7:0]一起，该位会自我锁定。一旦锁定，只有复位才能清除这个位。<br><br>0 允许对LMDRn[7:0]进行写入。<br>1 对LMDRn[7:0]的写入被忽略。 |
| 15-13<br>MT | 存储器类型<br><br>该字段定义了本地存储器的类型。010PC<br><br>缓存 |
| 12<br>保留 | 这个字段是保留的。 |
| 11-8<br>保留 | 这个字段是保留的。 |
| 7-4<br>CF1 | 控制场1<br><br>该字段用于缓存奇偶校验控制功能。<br>• CF1[3]-PCPFE = PC 奇偶校验故障启用<br>• CF1[2]-保留<br>• CF1[1]-PCPME = PC奇偶校验缺失启用<br>• CF1[0]-保留 |
| 保留 | 这个字段是保留的。 |

## 8.3.9 LMEM 奇偶校验和 ECC 控制寄存器 (MCM_LMPECR)

地址。0h基数+480h偏移量=480h




**MCM_LMPECR字段描述**

| 场地 | 描述 |
|---|---|
| 31-21<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 20<br>ECPR | 启用缓存奇偶校验报告<br><br>0 报告失效<br>1 启用报告 |
| 19-17<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 15-9<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 8<br>ER1BR | 启用RAM ECC 1位报告<br><br>0 报告失效<br>1 启用报告 |
| 7-1<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 0<br>ERNCR | 启用RAM ECC不可更正报告<br><br>0 报告禁用 |

| | |
|---|---|
| | 1 启用报告 |

## 8.3.10 LMEM 奇偶校验和 ECC 中断寄存器 (MCM_LMPEIR)

## 注意事项

对LMPEIR[23:0]中的错误位写1可以清除中断标志。

地址。0h基数+488h偏移量=488h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | V | 0 | | 裴乐乐 | | | | | 塑胶跑道 | | | | | | | |
| W | | | | | | | | | w1c | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 位 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | E1B | | | | | | | | ENC | | | | | | | |
| W | w1c | | | | | | | | w1c | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

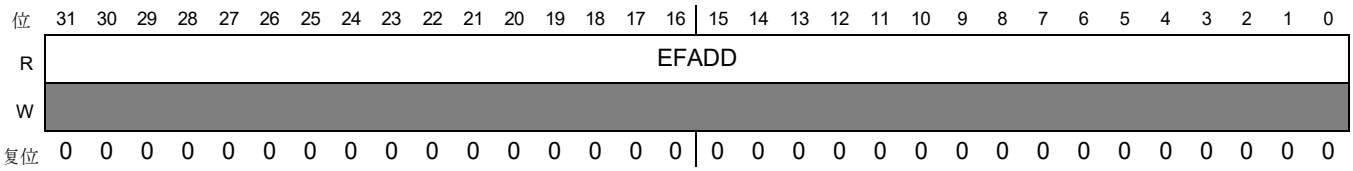
### MCM_LMPEIR字段描述

| 场地 | 描述 |
|---|---|
| 31<br>V | 有效位 |
| 30-29<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 28-24<br>裴乐乐 | 同位数或ECC错误位置<br>00 来自SRAM_L的不可纠正ECC事件 01 来自SRAM_U的不可纠正ECC事件 08 来自SRAM_L的1位可纠正ECC事件 09 来自SRAM_U的1位可纠正ECC事件<br>14 PC标签奇偶性错误<br>15 PC数据奇偶性错误 |
| 23-16<br>塑胶跑道 | 缓存奇偶校验错误<br>• [21] - PC数据奇偶性错误<br>• [20] - PC标签奇偶性错误<br>• [19] - 保留<br>• [18] - 保留 |
| 15-8<br>E1B | E1Bn = ECC 1位错误n<br>• PEIR[15:10] - 保留<br>• PEIR[9] - 在SRAM_U上检测到的1位错误<br>• PEIR[8] - 在SRAM_L上检测到的1位错误 |

| | |
|---|---|
| ENC | ENCn = ECC不可纠正的错误n<br>• PEIR[7:2] - 保留<br>• PEIR[1] - 在SRAM_U上检测到不可纠正的错误<br>• PEIR[0] - 在SRAM_L上检测到的不可纠正的错误 |

## 8.3.11 LMEM故障地址寄存器 (MCM_LMFAR)

地址。0h base + 490h offset = 490h

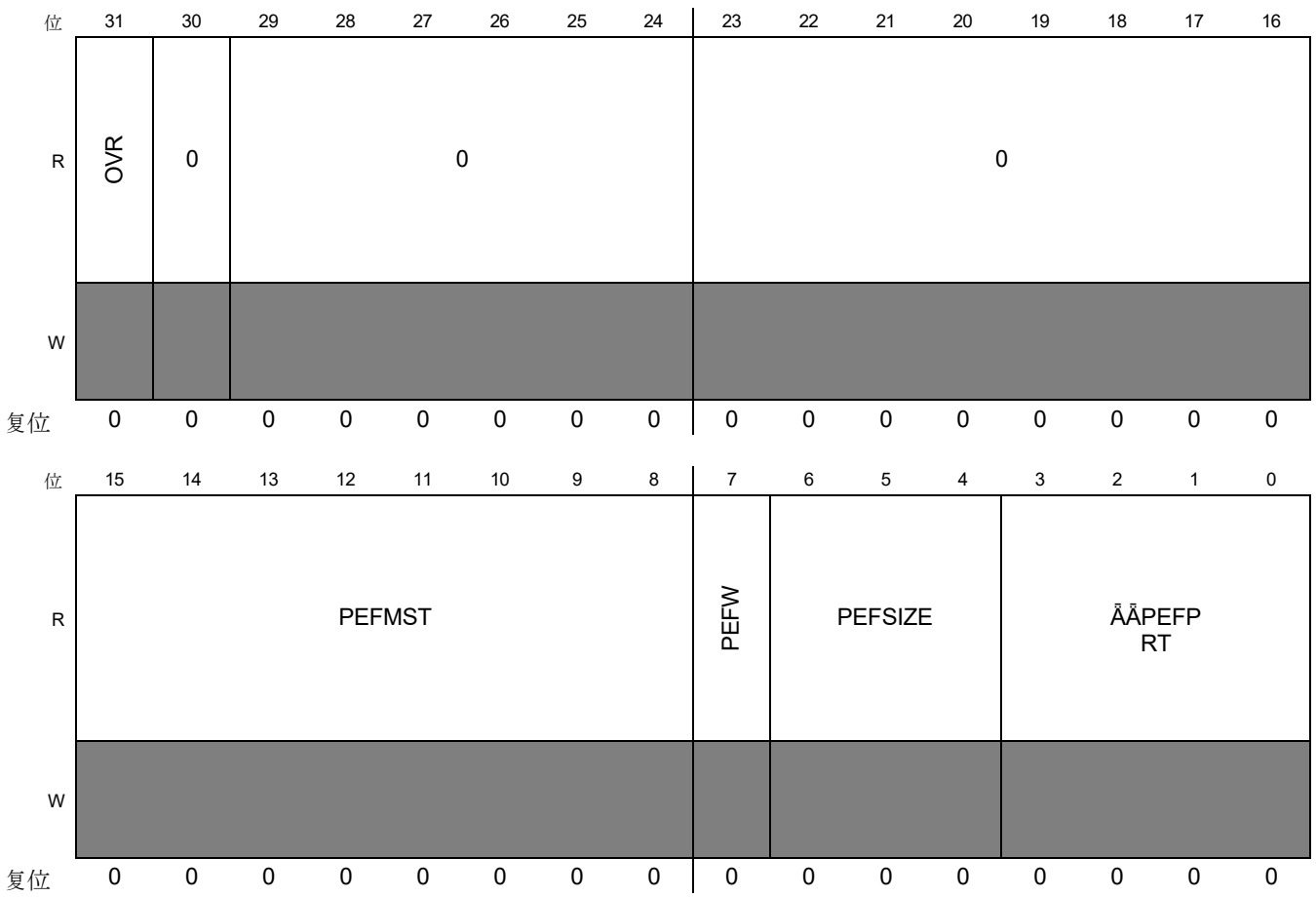



**MCM_LMFAR字段描述**

| 场地 | 描述 |
|---|---|
| EFADD | ECC故障地址 |

## 8.3.12 LMEM故障属性寄存器 (MCM_LMFATR)

地址。0h基数+494h偏移量=494h




**MCM_LMFATR字段描述**

| 场地 | 描述 |
|---|---|
| 31<br>OVR | 超额完成任务 |
| 30<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 29-24<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 23-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 15-8<br>PEFMST | 奇偶校验/ECC故障 主站号码 |
| 7<br>PEFW | 奇偶校验/ECC故障写入 |

表继续在下一页...

**MCM_LMFATR字段描述（续）**

| 场地 | 描述 |
|---|---|
| 6-4<br>PEFSIZE | 奇偶校验/ECC故障 主站尺寸<br><br>0008位访问<br>00116位访问<br>01032位访问<br>01164位访问<br>1xx 保留 |
| ÅÅPEFPRT | 奇偶校验/ECC故障保护<br><br>• FATR[3] - 可缓存。0=不可缓存，1=可缓存<br>• FATR[2] - 可缓冲。0=不可缓冲，1=可缓冲<br>• FATR[1] - 模式：0=用户模式，1=监管者模式<br>• FATR[0] - 类型。0=I-Fetch, 1=Data |

## 8.3.13 LMEM 故障数据高位寄存器 (MCM_LMFDHR)

地址。0h基数 + 4A0h偏移量 = 4A0h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | PEFDH | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**MCM_LMFDHR字段描述**

| 场地 | 描述 |
|---|---|
| PEFDH | 奇偶校验或ECC故障数据高 |

## 8.3.14 LMEM 故障数据低位寄存器 (MCM_LMFDLR)

地址。0h基数 + 4A4h偏移量 = 4A4h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | PEFDL | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

MCM_LMFDLR字段描述

| 场地 | 描述 |
|---|---|
| PEFDL | 奇偶校验或ECC故障数据低 |

## 8.4 功能 描述

本节介绍MCM模块的功能描述。

### 8.4.1 中断

如果以下任何一项为真，则产生MCM中断。

- 启用FPU输入正常化中断(FIDCE)，一个输入被正常化(FIDC)
- 启用FPU不精确中断(FIXCE)，并且一个数字是不精确的(FIXC)
- 启用FPU下溢中断（FUFCE），并发生下溢（FUFC）。
- FPU溢出中断被启用（FOFCE），并且发生溢出（FOFC）。
- 启用FPU除零中断（FDZCE）并发生除零（FDZC）。
- FPU无效操作中断被启用(FIOCE)，并且发生了无效操作(FIOC)
- SRAM_L可纠正（1位）ECC错误
- SRAM_L不可纠正的ECC错误
- SRAM_U可纠正（1位）ECC错误
- SRAM_U不可纠正的ECC错误
- PC数据奇偶性错误
- PC标签奇偶性错误
- 缓存写入缓冲区错误

#### 8.4.1.1 确定中断的来源

这些步骤可以用来确定中断的确切来源。

1. **中断状**态标志与相应的中断使能位的逻辑和。ISCR[31:16]和ISCR[15:0]。
2. 在结果中搜索表示确切中断源的断言位。

**注意事项**

ECC和奇偶性中断由LMPECR（中断使能）和LMPEIR（中断源）决定。

# 章 9
# 系统集成模块 （SIM）。

## 9.1 特定芯片的SIM卡 信息

S32K14x系列的产品中，SIM卡的某些方面有所不同。

### 9.1.1 SIM寄存器位域 实现

并非每个S32K1xx变体都能实现SIM内存图中的所有寄存器位域。寄存器/寄存器位域的可用性根据模块的可用性/一个变体中存在的模块实例的数量而变化。见图。S32K1xx产品系列比较，了解模块和实例信息的细节。

## 9.2 简介

系统集成模块（SIM）提供系统控制和芯片配置寄存器。

### 9.2.1 特点

该SIM卡的特点包括。

- 系统时钟配置
- 闪存和系统RAM大小配置
- FlexTimer时钟通道和配置
- ADC触发器选择
- LPO时钟源选择
- ENET时钟控制
- 闪存配置
- 系统设备识别（ID）。

## 9.3 内存地图和寄存器 定义

SIM包含许多字段，用于选择各种模块时钟的时钟源和分频器。请看章节。时钟分配一章，了解更多信息，包括方框图和时钟定义。

**注意事项**

SIM寄存器只能在监督员模式下被写入。在用户模式下，写访问被阻止，将导致总线错误。

**注意事项**

对只读寄存器的写入可能/可能不会产生传输错误。

### 9.3.1 SIM寄存器 描述

#### 9.3.1.1 SIM卡内存图

SIM基础地址。4004_8000h

| 偏移 | 注册 | 宽度 (单位：比特) | 访问 | 重置值 |
|---|---|---|---|---|
| 4h | 芯片控制寄存器（CHIPCTL） | 32 | RW | 0030_0000h |
| 丘吉尔 | FTM选项寄存器0（FTMOPT0）。 | 32 | RW | 0000_0000h |
| 10h | LPO时钟选择寄存器（LPOCLKS）。 | 32 | RW | 0000_0003h |
| 18h | ADC选项寄存器（ADCOPT） | 32 | RW | 0000_0000h |
| 1Ch | FTM选项寄存器1（FTMOPT1）。 | 32 | RW | 0000_0000h |
| 20h | 杂项控制寄存器0（MISCTRL0）。 | 32 | RW | 0000_0000h |
| 24h | 系统设备识别寄存器（SDID） | 32 | RO | 见描述。 |
| 40h | 平台时钟选通控制寄存器（PLATCGC）。 | 32 | RW | 0000_001Fh |
| 4Ch | 闪存配置寄存器1（FCFG1）。 | 32 | RW | 见描述。 |
| 54h | 唯一识别寄存器高位（UIDH）。 | 32 | RO | 见描述。 |
| 58h | 独特的识别注册中高（UIDMH）。 | 32 | RO | 见描述。 |

表继续在下一页...

| 偏移 | 注册 | 宽度<br>(单位：比特) | 访问 | 重置值 |
|---|---|---|---|---|
| 5Ch | 唯一的识别寄存器中低端（UIDML）。 | 32 | RO | 见描述。 |
| 60h | 唯一识别寄存器低电平（UIDL） | 32 | RO | 见描述。 |
| 68h | 系统时钟分频器寄存器4（CLKDIV4）。 | 32 | RW | 1000_0000h |
| 6Ch | 杂项控制寄存器1（MISCTRL1）。 | 32 | RW | 0000_0000h |

## 9.3.1.2 芯片控制寄存器（CHIPCTL）

### 9.3.1.2.1 偏移

| 注册 | 偏移 |
|---|---|
| 芯片 | 4h |

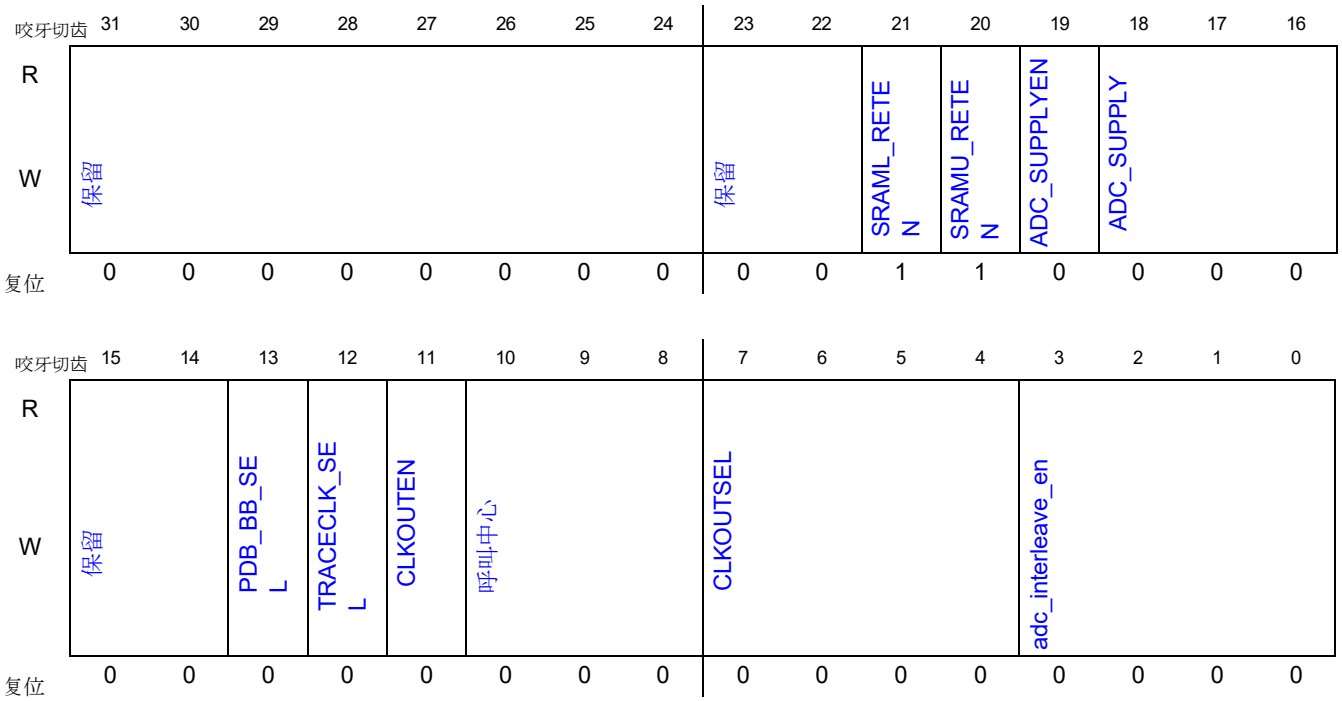
### 9.3.1.2.2 职能

SIM_CHIPCTL包含选择ADC
COCO触发器、跟踪时钟、时钟输出源、PDB背对背模式和ADC交错通道的控制
。

**注意事项**

第31:16位在POR时被重置。

### 9.3.1.2.3 图示




### 9.3.1.2.4 领域

| 场地 | 职能 |
|---|---|
| 31-24<br>- | 保留 |
| 23-22<br>- | 保留 |
| 21<br>SRAML_RETEN | SRAML_RETEN<br>SRAML保留<br>**注意：**对于S32K11x器件，SRAML指的是MTB RAM。<br>0b - 跨越复位保留SRAML内容 1b - 不保留<br>SRAML内容 |
| 20<br>SRAMU_RETEN | SRAMU_RETEN<br>SRAMU保留<br>0b - 跨越复位保留SRAMU内容 1b - 不保留<br>SRAMU内容 |
| 19<br>ADC_SUPPLYEN | ADC_SUPPLYEN<br>启用ADC0内部通道0的内部电源监控（通过选择ADC0_SC1n[ADCH]为010101b进行配置）。<br>0b - 禁用内部电源监控 1b - 启用内部电<br>源监控 |

| | |
|---|---|
| 18-16<br><br>ADC_SUPPLY | ADC_SUPPLY |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | 在ADC0内部通道0上监测内部电源（通过选择ADC0_SC1n[ADCH]为010101b来配置）。<br>000b - 5 V输入VDD电源（VDD） 001b<br>- 5 V输入模拟电源（VDDA） 010b -<br>ADC参考电源（VREFH）<br>011b - 3.3 V 振荡器稳压器输出 (VDD_3V) 100b -<br>3.3 V 闪存稳压器输出 (VDD_flash_3V) 101b - 1.2<br>V 核心稳压器输出 (VDD_LV)<br>110b - 保 留<br>111b - 保留 |
| 15-14<br>- | 保留 |
| 13<br>PDB_BB_SEL | PDB背对背选择<br>选择ADC COCO源为pdb背对背模式，详见背对背确认连接部分。<br>0b - PDB0通道0与ADC0 COCO[7:0]背对背操作，PDB1通道0与ADC1 COCO[7:0]背对背操作。<br>1b - PDB0和PDB1的通道0与ADC0和ADC1的COCO[7:0]背对背操作。 |
| 12<br>TRACECLK_SEL | 调试跟踪时钟选择<br>选择核心时钟或平台时钟作为跟踪时钟源。<br>0b - 核心时钟<br>1b - 平台时钟 |
| 11<br>CLKOUTEN | 启用CLKOUT<br>**注意：**在配置CLKOUT时应遵循以下顺序。<br>1. 配置SIM_CHIPCTL[CLKOUTSEL]<br>2. 配置SIM_CHIPCTL[CLKOUTDIV]<br>3. 启用SIM_CHIPCTL[CLKOUTEN]。<br>(在切换时，应首先清除CLKOUTEN，然后再按照上述顺序进行。）<br>0b - 禁用时钟输出 1b<br>- 启用时钟输出 |
| 10-8<br>呼叫中心 | CLKOUT 分割率<br>**注意：**在配置CLKOUT时应遵循以下顺序。<br>1. 配置SIM_CHIPCTL[CLKOUTSEL]<br>2. 配置SIM_CHIPCTL[CLKOUTDIV]<br>3. 启用SIM_CHIPCTL[CLKOUTEN]。<br>(在切换时，应首先清除CLKOUTEN，然后再按照上述顺序进行。）<br>000b - 除 以 1<br>001b - 除 以 2<br>010b - 除 以 3<br>011b - 除 以 4<br>100b - 除 以 5<br>101b - 除 以 6<br>110b - 除 以 7<br>111b - 除以8 |

| | |
|---|---|
| 7-4<br>CLKOUTSEL | CLKOUT选择<br><br>**注意：**在配置CLKOUT时应遵循以下顺序。<br>1. 配置SIM_CHIPCTL[CLKOUTSEL]<br>2. 配置SIM_CHIPCTL[CLKOUTDIV] |

*表继续在下一页...*

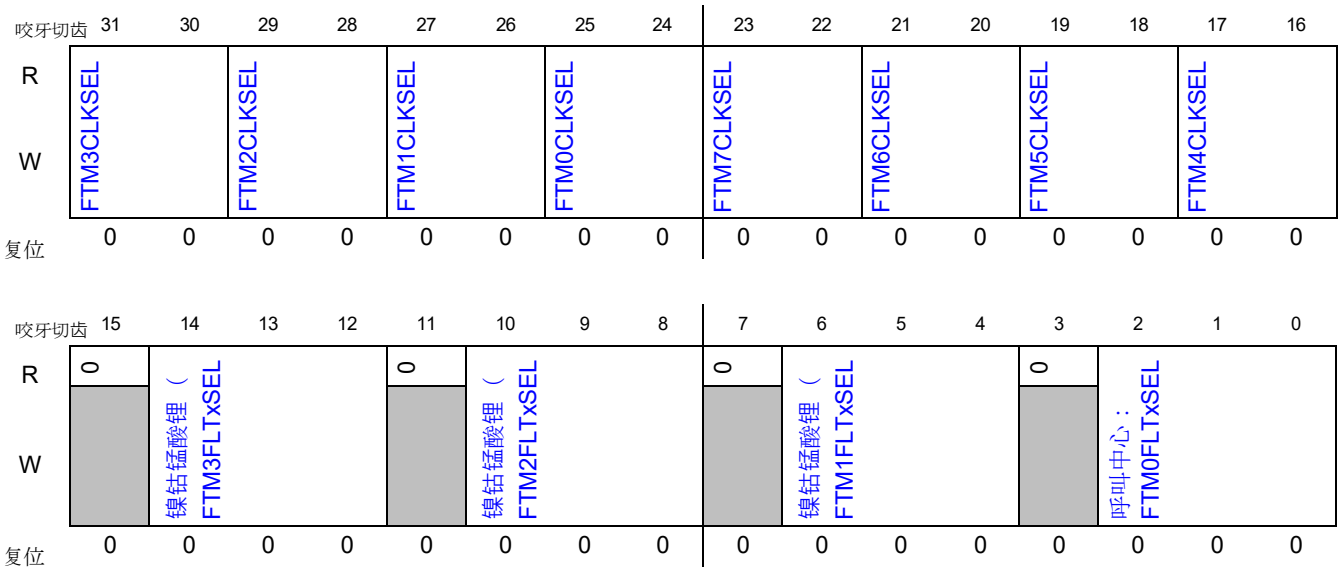
| 场地 | 职能 |
| --- | --- |
| | 3.启用SIM_CHIPCTL[CLKOUTEN]。<br>(在切换时，应首先清除CLKOUTEN，然后再按照上述顺序进行。）<br>注：对于QuadSPI时钟，见表 "外围模块时钟 "中的QuadSPI时钟图 选择在CLKOUT引脚上输出的时钟。<br>0000b - SCG CLKOUT<br>0001b - 保留<br>0010b - SOSC DIV2 CLK<br>0011b - 保留 0100b -<br>SIRC DIV2 CLK<br>0101b - 对于S32K148。QSPI_SFIF_CLK_HYP_PREMUX: 除以2的时钟（通过SCLKCONFIG[5]配置）用于HyperRAM进入sfif时钟到QSPI；对于其他。保留。<br>0110b - FIRC DIV2 CLK<br>0111b - HCLK<br>1000b - SPLL DIV2 CLK<br>1001b - BUS_CLK<br>1010b - LPO128K_CLK<br>1011b - 用于S32K148。QSPI_Module 时钟；对于其他。1100b - 由<br>SIM_LPOCLKS[LPOCLKSEL]选择的LPO_CLK 1101b - 对于<br>S32K148，QSPI_SFIF_CLK；对于其他的，保留。<br>QSPI_SFIF_CLK; 对于其他。1110b - 由<br>SIM_LPOCLKS[RTCCLKSEL]选择的RTC_CLK。<br>QSPI_2xSFIF_CLK; 对于其他。保留。 |
| 3-0<br>呼叫中心_interleave_en | ADC交错通道启用<br>选择ADC交错引脚。更多信息请参见ADC硬件交错通道部分。0000b - 禁用交织。没有通道对交错。交错通道单独连接到引脚。PTC0被连接到ADC0_SE8。PTC1连接到ADC0_SE9。PTB15连接到ADC1_SE14。PTB16连接到ADC1_SE15。PTB0连接到ADC0_SE4。PTB1连接到ADC0_SE5。PTB13连接到ADC1_SE8。PTB14连接到ADC1_SE9。<br>1xxxb - PTB14 到 ADC1_SE9 和 ADC0_SE9<br>x1xxb - PTB13 到 ADC1_SE8 和 ADC0_SE8<br>xx1xb - PTB1 到 ADC0_SE5 和 ADC1_SE15<br>xxx1b - PTB0到ADC0_SE4和ADC1_SE14 |

## 9.3.1.3 FTM选项寄存器0（FTMOPT0）。

### 9.3.1.3.1 偏移

| 注册 | 偏移 |
| --- | --- |
| FTMOPT0 | 丘吉尔 |

### 9.3.1.3.2 图示




### 9.3.1.3.3 领域

| 场地 | 职能 |
|---|---|
| 31-30<br>FTM3CLKSEL | FTM3外部时钟引脚选择<br>选择用于驱动FTM3模块时钟的外部引脚。<br>**注意：**选定的引脚也必须通过端口控制模块中适当的引脚控制寄存器配置为FTM3外部时钟功能。<br>00b - FTM3 的外部时钟由 TCLK0 引脚驱动。<br>01b - FTM3 的外部时钟由 TCLK1 引脚驱动。<br>10b - FTM3 的外部时钟由 TCLK2 引脚驱动。<br>11b - 无时钟输入 |
| 29-28<br>FTM2CLKSEL | FTM2外部时钟引脚选择<br>选择用于驱动FTM2模块时钟的外部引脚。<br>**注意：**选定的引脚也必须通过端口控制模块中适当的引脚控制寄存器配置为FTM2外部时钟功能。<br>00b - FTM2 的外部时钟由 TCLK0 引脚驱动。<br>01b - FTM2 的外部时钟由 TCLK1 引脚驱动。<br>10b - FTM2 的外部时钟由 TCLK2 引脚驱动。<br>11b - 无时钟输入 |

| | |
|---|---|
| 27-26<br>FTM1CLKSEL | FTM1外部时钟引脚选择<br>选择用于驱动FTM1模块时钟的外部引脚。<br>**注意：**选定的引脚也必须通过端口控制模块中适当的引脚控制寄存器配置为FTM1外部时钟功能。<br>00b - FTM1 的外部时钟由 TCLK0 引脚驱动。<br>01b - FTM1 的外部时钟由 TCLK1 引脚驱动。<br>10b - FTM1外部时钟由TCLK2引脚驱动。11b - 无时钟输入 |
| 25-24 | FTM0外部时钟引脚选择 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| FTM0CLKSEL | 选择用于驱动FTM0模块时钟的外部引脚。<br>**注意：**所选引脚也必须通过端口控制模块中适当的引脚控制寄存器配置为FTM0外部时钟功能。<br>00b - FTM0的外部时钟由TCLK0引脚驱动。01b - FTM0 的外部时钟由 TCLK1 引脚驱动。10b - FTM0 的外部时钟由 TCLK2 引脚驱动。11b - 无时钟输入 |
| 23-22<br>FTM7CLKSEL | FTM7外部时钟引脚选择<br>选择用于驱动FTM7模块时钟的外部引脚。<br>**注意：**选定的引脚也必须通过端口控制模块中适当的引脚控制寄存器配置为FTM7外部时钟功能。<br>00b - FTM7 的外部时钟由 TCLK0 引脚驱动。<br>01b - FTM7 的外部时钟由 TCLK1 引脚驱动。<br>10b - FTM7 的外部时钟由 TCLK2 引脚驱动。<br>11b - 无时钟输入 |
| 21-20<br>FTM6CLKSEL | FTM6外部时钟引脚选择<br>选择用于驱动FTM6模块时钟的外部引脚。<br>**注意：**选定的引脚也必须通过端口控制模块中适当的引脚控制寄存器配置为FTM6外部时钟功能。<br>00b - FTM6 的外部时钟由 TCLK0 引脚驱动。<br>01b - FTM6 的外部时钟由 TCLK1 引脚驱动。<br>10b - FTM6 的外部时钟由 TCLK2 引脚驱动。<br>11b - 无时钟输入 |
| 19-18<br>FTM5CLKSEL | FTM5外部时钟引脚选择<br>选择用于驱动FTM5模块时钟的外部引脚。<br>**注意：**所选引脚也必须通过端口控制模块中适当的引脚控制寄存器配置为FTM5外部时钟功能。<br>00b - FTM5 的外部时钟由 TCLK0 引脚驱动。<br>01b - FTM5 的外部时钟由 TCLK1 引脚驱动。<br>10b - FTM5 的外部时钟由 TCLK2 引脚驱动。<br>11b - 无时钟输入 |
| 17-16<br>FTM4CLKSEL | FTM4外部时钟引脚选择<br>选择用于驱动FTM4模块时钟的外部引脚。<br>**注意：**选定的引脚也必须通过端口控制模块中适当的引脚控制寄存器配置为FTM4外部时钟功能。<br>00b - FTM4 的外部时钟由 TCLK0 引脚驱动。<br>01b - FTM4 的外部时钟由 TCLK1 引脚驱动。<br>10b - FTM4 外部时钟由 TCLK2 引脚驱动。11b - 无时钟输入 |
| 15<br>- | 保留 |

| | |
|---|---|
| 14-12<br>镍钴锰酸锂（<br>FTM3FLTxSEL | FTM3 故障X选择<br>选择FTM3的故障源。每位分别代表一个故障输入。<br>**注意：**当故障来自外部引脚时，必须通过适当的PORT_PCRn字段为FTM3故障功能配置引脚来源。<br>TRGMUX_FTM3 SELx 对应于 FTM3 故障 x 输入。<br>000b - FTM3_FLTx 引脚 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | 001b - TRGMUX_FTM3输出 |
| 11<br>-<br> | 保留 |
| 10-8<br>镍钴锰酸锂（<br>FTM2FLTxSEL | FTM2故障X选择<br>选择FTM2的故障源。每位分别代表一个故障输入。<br>**注意：**当故障来自外部引脚时，必须通过适当的PORT_PCRn字段为FTM2故障功能配置引脚来源。<br>TRGMUX_FTM2 SELx 对应于 FTM2 故障 x 输入。<br>000b - FTM2_FLTx 引脚<br>001b - TRGMUX_FTM2输出 |
| 7<br>-<br> | 保留 |
| 6-4<br>镍钴锰酸锂（<br>FTM1FLTxSEL | FTM1 故障X选择<br>选择FTM1的故障源。每位分别代表一个故障输入。<br>**注意：**当故障来自外部引脚时，必须通过适当的PORT_PCRn字段为FTM1故障功能配置引脚来源。<br>TRGMUX_FTM1 SELx 对应于 FTM1 故障 x 输入。<br>000b - FTM1_FLTx 引脚<br>001b - TRGMUX_FTM1输出 |
| 3<br>-<br> | 保留 |
| 2-0<br>呼叫中心：<br>FTM0FLTxSEL | FTM0 故障X选择<br>选择FTM0的故障源。每位分别代表一个故障输入。<br>**注意：**当故障来自外部引脚时，必须通过适当的PORT_PCRn字段为FTM0故障功能配置引脚来源。<br>TRGMUX_FTM0 SELx 对应于 FTM0 故障 x 输入。<br>000b - FTM0_FLTx 引脚<br>001b - TRGMUX_FTM0输出 |

## 9.3.1.4 LPO时钟选择寄存器（LPOCLKS）。

### 9.3.1.4.1 偏移

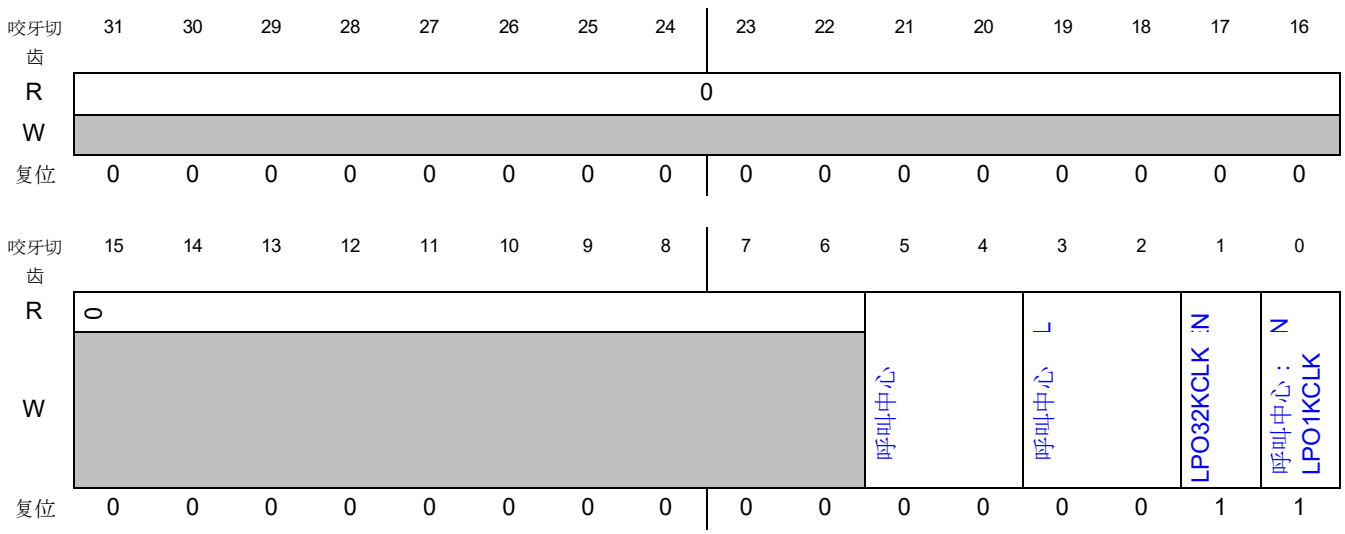
| 注册 | 偏移 |
| --- | --- |
| 呼叫中心 | 10h |

### 9.3.1.4.2 职能

**注意事项**

LPOCLKS寄存器是一个只写不回的寄存器，只有在POR或LVD时才会复位。

### 9.3.1.4.3 图示




### 9.3.1.4.4 领域

| 场地 | 职能 |
|---|---|
| 31-6<br>- | 保留 |
| 5-4<br>辽宁沈阳 | 32 kHz时钟源选择<br>选择外设的32kHz时钟源 00b - SOSCDIV1_CLK<br>01b - 32 kHz LPO_CLK<br>10b - RTC_CLKIN时钟<br>11b - FIRCDIV1_CLK |
| 3-2<br>辽宁省 | LPO时钟源选择<br>选择外围设备的LPO时钟源 00b - 128 kHz LPO_CLK<br>01b - 无时钟<br>10b - 32 kHz LPO_CLK，源自128 kHz LPO_CLK 11b - 1 kHz LPO_CLK，源自128 kHz LPO_CLK |

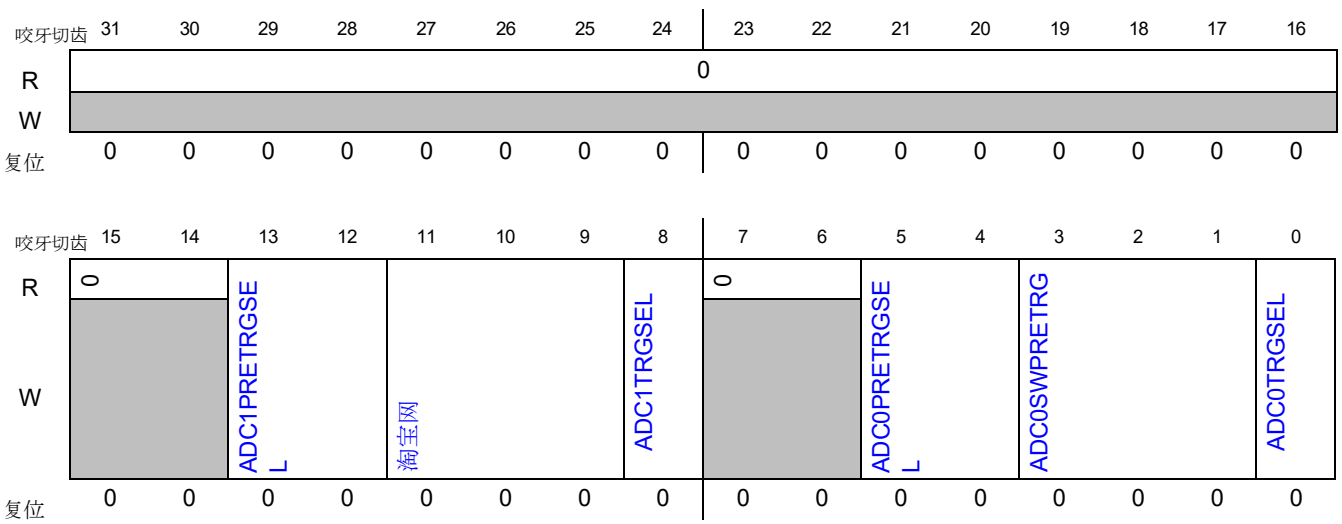
| | |
|---|---|
| 1<br>LPO32KCLKEN | 32 kHz LPO_CLK启用<br>0b - 禁用32kHz LPO_CLK输出 1b - 启用32kHz LPO_CLK输出 |
| 0<br>LPO1KCLKEN | 1 kHz LPO_CLK启用<br>0b - 禁用1kHz LPO_CLK输出 1b - 启用1kHz LPO_CLK输出 |

## 9.3.1.5 ADC选项寄存器（ADCOPT）

### 9.3.1.5.1 偏移

| 注册 | 偏移 |
|---|---|
| ÃÃÃ | 18h |

### 9.3.1.5.2 图示




### 9.3.1.5.3 领域

| 场地 | 职能 |
|---|---|
| 31-16<br>- | 保留 |
| 15-14<br>- | 保留 |
| 13-12<br>ADC1PRETRGSEL | ADC1预触发源选择<br>选择ADC1的预触发源。00b - PDB<br>预触发（默认） 01b -<br>TRGMUX预触发 10b - 软件 |

*表继续在下一页...*

| | |
|---|---|
| | 预触发<br>11b - 保留 |
| 11-9<br>ADC1SWPRETRG | ADC1软件预触发源<br>000b - 禁用软件预触发 001b - 保留（不使用）。<br>010b - 保留（不使用）。 |

| 场地 | 职能 |
| --- | --- |
| | 011b - 保留（不使用） 100b - 软件预触发 0 101b - 软件预触发 1 110b - 软件预触发 2 111b - 软件预触发 3 |
| 8<br>ADC1TRGSEL | ADC1触发源选择 选择ADC1的触发源。<br>**注意：**每个PDB支持两个ADC通道，每个通道有8个预触发器。如果需要，两个ADC通道的触发器被OR'ed在一起，以支持多达16个预触发器。<br>0b - PDB输出<br>1b - TRGMUX输出 |
| 7-6<br>- | 保留 |
| 5-4<br>ADC0PRETRGSEL | ADC0预触发源选择<br>选择ADC0的预触发源。00b - PDB预触发（默认） 01b - TRGMUX预触发 10b - 软件预触发<br>11b - 保留 |
| 3-1<br>ADC0SWPRETRG | ADC0软件预触发源<br>000b - 禁用软件预触发 001b - 保留（不使用）。<br>010b - 保留（不使用） 011b - 保留（不使用） 100b - 软件预触发0 101b - 软件预触发1 110b - 软件预触发2 111b - 软件预触发3 |
| 0<br>ADC0TRGSEL | ADC0触发源选择 选择ADC0的触发源。<br>**注意：**每个PDB支持两个ADC通道，每个通道有8个预触发器。如果需要，两个ADC通道的触发器被OR'ed在一起，以支持多达16个预触发器。<br>0b - PDB输出<br>1b - TRGMUX输出 |

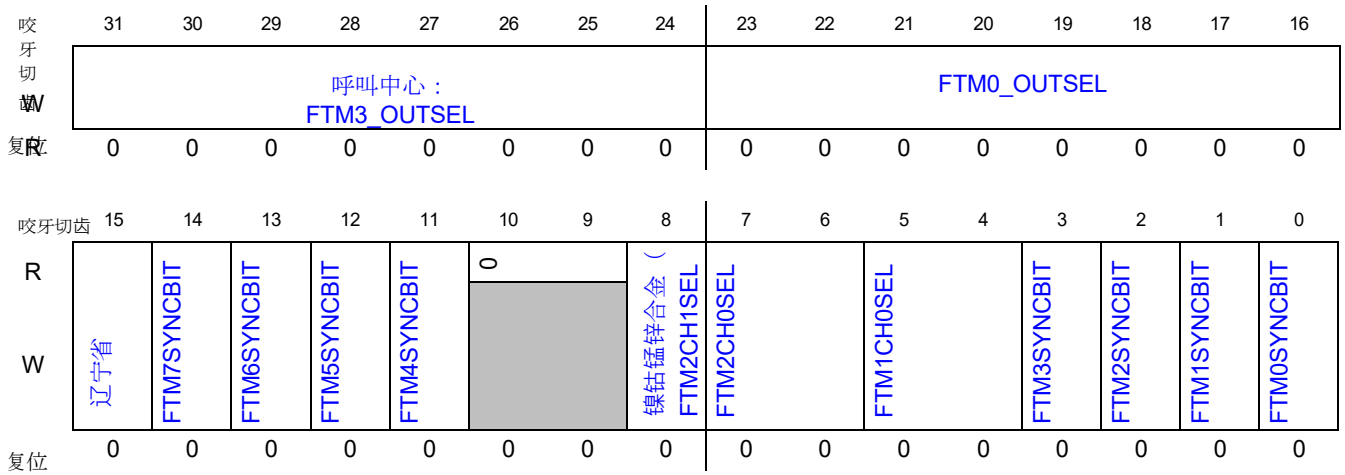
## 9.3.1.6 FTM选项寄存器1（FTMOPT1）。

*表继续在下一页...*

#### 9.3.1.6.1 偏移

| 注册 | 偏移 |
| --- | --- |
| FTMOPT1 | 1Ch |

### 9.3.1.6.2 图示




### 9.3.1.6.3 领域

| 场地 | 职能 |
|---|---|
| 31-24<br>呼叫中心：<br>FTM3_OUTSEL | FTM3通道调制选择与FTM2_CH1<br>该字段的第7至0位分别用于通道7至0。<br>00000000b - 不使用FTM2_CH1进行调制<br>00000001b - 使用FTM2_CH1进行调制 |
| 23-16<br>FTM0_OUTSEL | FTM0通道调制选择与FTM1_CH1<br>该字段的第7至0位分别用于通道7至0。<br>00000000b - 不使用FTM1_CH1调制 00000001b<br>- 使用FTM1_CH1进行调制 |
| 15<br>辽宁省 | 启用FTM全局负载<br>这个位不是自动清除的。对于后续的重载操作，它应该被清零，然后被设置。<br>0b - 禁用 FTM 全局负载机制。 1b - 启用<br>FTM全局负载机制 |
| 14<br>FTM7SYNCBIT | FTM7同步位<br>这被用作FTM7的触发源。关于FTM硬件触发的细节，请参见FTM硬件触发和同步一节。 |
| 13<br>FTM6SYNCBIT | FTM6同步位<br>这被用作FTM6的触发源。关于FTM硬件触发的细节，请参见FTM硬件触发和同步一节。 |
| 12<br>FTM5SYNCBIT | FTM5同步位<br>这被用作FTM5的触发源。关于FTM硬件触发的细节，请参见FTM硬件触发和同步一节。 |

*表继续在下一页...*

| | |
|---|---|
| 11<br>FTM4SYNCBIT | FTM4同步位<br>这被用作FTM4的触发源。关于FTM硬件触发的细节，请参见FTM硬件触发和同步一节。 |
| 10-9<br>- | 保留 |

| 9.3.1.6.2 场地 | 图示 | 职能 |
|---|---|---|
| 8<br>咬<br>镍钴锰锌合金（<br>FTM2CH1SEL<br>齿<br>R | | FTM2 CH1选择<br>选择FTM2 CH1输入 0b - FTM2_CH1输入<br>1b - FTM2_CH0,FTM2_CH1,和FTM1_CH1的排他性OR。 |
| 7-6<br>FTM2CH0SEL | | FTM2 CH0选择<br>选择FTM2 CH0输入 00b - FTM2_CH0输入 01b - CMP0输出 10b - 保留<br>11b - 保留 |
| 5-4<br>FTM1CH0SEL | | FTM1 CH0选择<br>选择FTM1 CH0输入 00b - FTM1_CH0输入 01b - CMP0输出 10b - 保留<br>11b - 保留 |
| 3<br>FTM3SYNCBIT | | FTM3同步位<br>这被用作FTM3的触发源。关于FTM硬件触发的细节，请参见FTM硬件触发和同步一节。 |
| 2<br>FTM2SYNCBIT | | FTM2同步位<br>这被用作FTM2的触发源。关于FTM硬件触发的细节，请参见FTM硬件触发和同步一节。 |
| 1<br>FTM1SYNCBIT | | FTM1同步位<br>这被用作FTM1的触发源。关于FTM硬件触发的细节，请参见FTM硬件触发和同步一节。 |
| 0<br>FTM0SYNCBIT | | FTM0同步位<br>这被用作FTM0的触发源。关于FTM硬件触发的细节，请参见FTM硬件触发和同步一节。 |

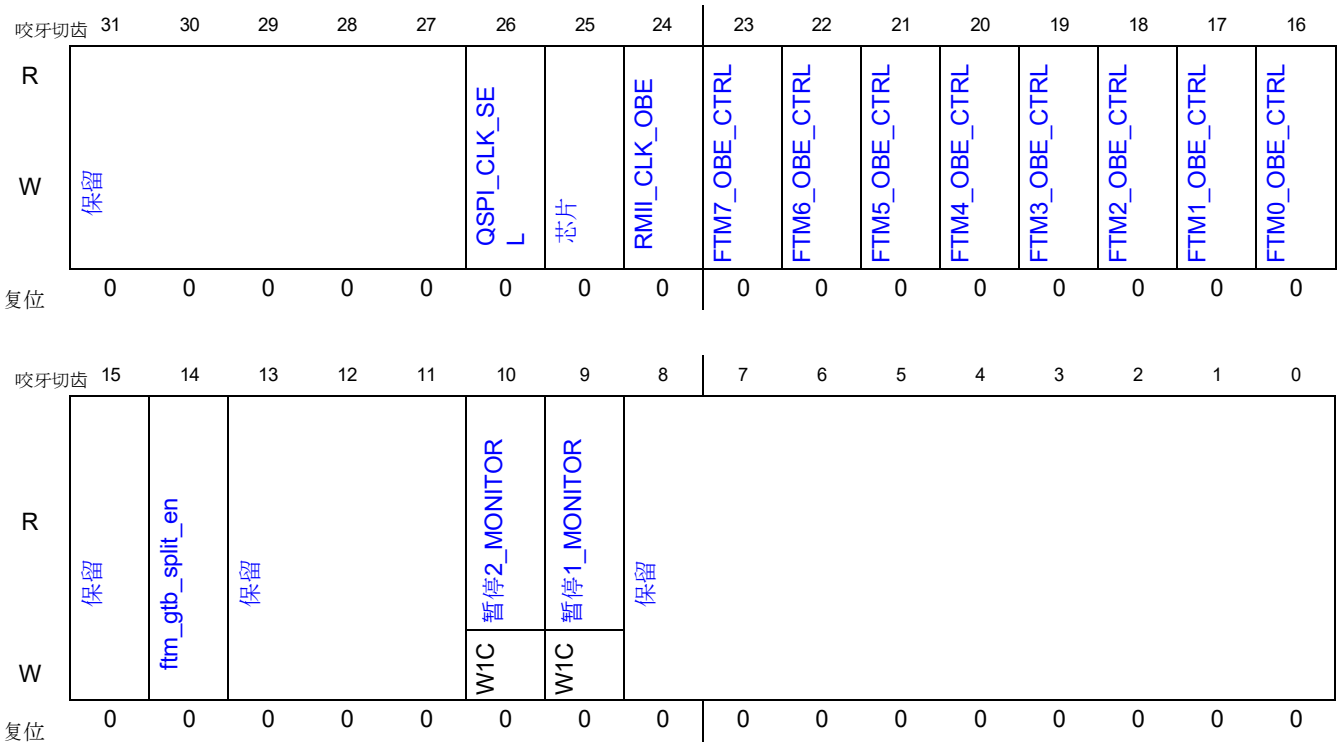
## 9.3.1.7 杂项控制寄存器0（MISCTRL0）。

### 9.3.1.7.1 偏移

| 注册 | 偏移 |
|---|---|
| MISCTRL0 | 20h |

*表继续在下一页...*

### 9.3.1.7.2 图示




### 9.3.1.7.3 领域

| 场地 | 职能 |
|---|---|
| 31-27<br>- | 保留 |
| 26<br>QSPI_CLK_SEL | QSPI CLK选择位<br>QSPI异步时钟门控启用<br>0b - QuadSPI内部参考时钟被门控。1b - QuadSPI内部参考时钟被启用。 |
| 25<br>芯片 | RMII CLK选择位<br>设置此位，在内部环回模式下，启用SOSCDIV1_CLK作为ENET RMII时钟。 |
| 24<br>RMII_CLK_OBE | RMII CLK OBE位<br>在内部环回模式下，ENET RMII时钟的输出缓冲器启用 |
| 23<br>FTM7_OBE_CTRL | FTM7 OBE CTRL位<br>0b - 当 FTM 计数器被启用，且 FTM 通道输出被故障控制（FTM_MODE[FAULTM]!=2'b00 和 FTM_FLTCTRL[FSTATE]=1'b0）和 PWM 被启用（FTM_SC[PWMENn] =1'b1）时，FTM 通道输出被置于安全状态。否则，该通道的输出是三态的。<br>1b - 当通道处于输出模式时，FTM 通道的输出状态被保留。当通道处于输入捕获[DECAPEN=1'b0, |

| | |
|---|---|
| | COMBINE=1'b0, MSnB:MSnA=2'b00]或双边缘捕获模式[DECAPEN=1'b1]时，输出通道处于三态。 |
| 22 | FTM6 OBE CTRL位 |

*表继续在下一页...*

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| FTM6_OBE_CTRL | 0b - 当 FTM 计数器被启用，且 FTM 通道输出被故障控制（FTM_MODE[FAULTM]!=2'b00 和 FTM_FLTCTRL[FSTATE]=1'b0）和 PWM 被启用（FTM_SC[PWMENn] =1'b1）时，FTM 通道输出被置于安全状态。否则，该通道的输出是三态的。<br>1b - 当通道处于输出模式时，FTM 通道的输出状态被保留。当通道处于输入捕获[DECAPEN=1'b0, COMBINE=1'b0, MSnB:MSnA=2'b00]或双边缘捕获模式[DECAPEN=1'b1]时，输出通道处于三态。 |
| 21<br>FTM5_OBE_CTRL | FTM5 OBE CTRL位<br>0b - 当 FTM 计数器被启用，且 FTM 通道输出被故障控制（FTM_MODE[FAULTM]!=2'b00 和 FTM_FLTCTRL[FSTATE]=1'b0）和 PWM 被启用（FTM_SC[PWMENn] =1'b1）时，FTM 通道输出被置于安全状态。否则，该通道的输出是三态的。<br>1b - 当通道处于输出模式时，FTM 通道的输出状态被保留。当通道处于输入捕获[DECAPEN=1'b0, COMBINE=1'b0, MSnB:MSnA=2'b00]或双边缘捕获模式[DECAPEN=1'b1]时，输出通道处于三态。 |
| 20<br>FTM4_OBE_CTRL | FTM4 OBE CTRL位<br>0b - 当 FTM 计数器被启用，且 FTM 通道输出被故障控制（FTM_MODE[FAULTM]!=2'b00 和 FTM_FLTCTRL[FSTATE]=1'b0）和 PWM 被启用（FTM_SC[PWMENn] =1'b1）时，FTM 通道输出被置于安全状态。否则，该通道的输出是三态的。<br>1b - 当通道处于输出模式时，FTM 通道的输出状态被保留。当通道处于输入捕获[DECAPEN=1'b0, COMBINE=1'b0, MSnB:MSnA=2'b00]或双边缘捕获模式[DECAPEN=1'b1]时，输出通道处于三态。 |
| 19<br>FTM3_OBE_CTRL | FTM3 OBE CTRL位<br>0b - 当 FTM 计数器被启用，且 FTM 通道输出被故障控制（FTM_MODE[FAULTM]!=2'b00 和 FTM_FLTCTRL[FSTATE]=1'b0）和 PWM 被启用（FTM_SC[PWMENn] =1'b1）时，FTM 通道输出被置于安全状态。否则，该通道的输出是三态的。<br>1b - 当通道处于输出模式时，FTM 通道的输出状态被保留。当通道处于输入捕获[DECAPEN=1'b0, COMBINE=1'b0, MSnB:MSnA=2'b00]或双边缘捕获模式[DECAPEN=1'b1]时，输出通道处于三态。 |
| 18<br>FTM2_OBE_CTRL | FTM2 OBE CTRL位<br>0b - 当 FTM 计数器被启用，且 FTM 通道输出被故障控制（FTM_MODE[FAULTM]!=2'b00 和 FTM_FLTCTRL[FSTATE]=1'b0）和 PWM 被启用（FTM_SC[PWMENn] =1'b1）时，FTM 通道输出被置于安全状态。否则，该通道的输出是三态的。<br>1b - 当通道处于输出模式时，FTM 通道的输出状态被保留。当通道处于输入捕获[DECAPEN=1'b0, COMBINE=1'b0, MSnB:MSnA=2'b00]或双边缘捕获模式[DECAPEN=1'b1]时，输出通道处于三态。 |
| 17<br>FTM1_OBE_CTRL | FTM1 OBE CTRL位<br>0b - 当 FTM 计数器被启用，且 FTM 通道输出被故障控制（FTM_MODE[FAULTM]!=2'b00 和 FTM_FLTCTRL[FSTATE]=1'b0）和 PWM 被启用（FTM_SC[PWMENn] =1'b1）时，FTM 通道输出被置于安全状态。否则，该通道的输出是三态的。<br>1b - 当通道处于输出模式时，FTM 通道的输出状态被保留。当通道处于输入捕获[DECAPEN=1'b0, COMBINE=1'b0, MSnB:MSnA=2'b00]或双边缘捕获模式[DECAPEN=1'b1]时，输出通道处于三态。 |

| | |
|---|---|
| 16<br><br>FTM0_OBE_CTRL | FTM0 OBE CTRL位<br>0b - 当 FTM 计数器被启用，且 FTM 通道输出被故障控制（FTM_MODE[FAULTM]!=2'b00 和 FTM_FLTCTRL[FSTATE]=1'b0）和 PWM 被启用（FTM_SC[PWMENn] =1'b1）时，FTM 通道输出被置于安全状态。否则，该通道的输出是三态的。<br>1b - 当通道处于输出模式时，FTM 通道的输出状态被保留。当通道处于输入捕获[DECAPEN=1'b0, COMBINE=1'b0, MSnB:MSnA=2'b00]或双边缘捕获模式[DECAPEN=1'b1]时，输出通道处于三态。 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| 15<br>-<br> | 保留 |
| 14<br>基金会_gtb_split_en<br> | FTM GTB分割的启用/禁用位<br>分割时间基础的启用/禁用。<br>0b - 所有的FTM都有一个全局时基<br>1b - FTM0-3 有一个共同的时基，其他有一个不同的共同时基。请参阅 FTM 章节中的 "FTM 全局时基 "以了解实施细节。 |
| 13-11<br>-<br> | 保留 |
| 10<br>STOP2_MONIT<br>或 | STOP2监控位<br>该状态位监测进入STOP2模式时的系统时钟状态，可用于监测STOP2进入是否成功或中止（在STOP2中，系统时钟被禁用，总线时钟被启用）。从STOP2模式唤醒后，该位需要变成W1C。<br>**注意：**该位在POR时被重置。<br>0b - 系统时钟启用或STOP2进入中止 1b - STOP2进入成功 |
| 9<br>STOP1_MONIT<br>或 | STOP1监控位<br>该状态位监控进入STOP1模式时的总线时钟状态，可用于监控STOP1进入是否成功或中止（在STOP1中，系统和总线时钟都被禁用）。从STOP1模式唤醒后，该位需要变成W1C。<br>**注意：**该位在POR时被重置。<br>0b - 总线时钟启用或STOP1进入中止 1b - STOP1进入成功 |
| 8-0<br>-<br> | 保留 |

## 9.3.1.8 系统设备识别寄存器（SDID）

### 9.3.1.8.1 偏移

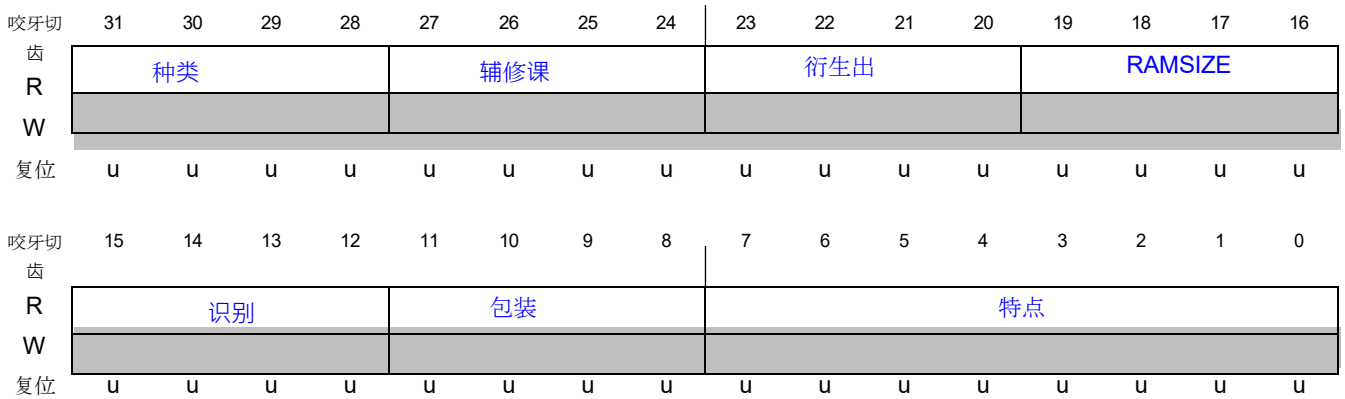
| 注册 | 偏移 |
| --- | --- |
| SDID | 24h |

### 9.3.1.8.2 职能

## 注意事项

这个寄存器的复位值在系统复位时从闪存IFR加载。

*表继续在下一页...*




## 9.3.1.8.4 领域

| 场地 | 职能 |
|---|---|
| 31-28<br>种类 | S32K产品系列的产生<br>指定S32K产品系列芯片的世代。该芯片的世代是1。 |
| 27-24<br>辅修课 | 子系列<br>指定芯片的子系列。其值为4（S32K14x芯片）或1（S32K11x芯片）。 |
| 23-20<br>衍生出 | 衍生品<br>指定芯片的导数。该值为8（S32K148）、6（S32K146）、4（S32K144）、2（S32K142）、8（S32118）或6（S32K116）。 |
| 19-16<br>RAMSIZE | 内存大小<br>这个字段指定了芯片上可用的系统RAM的总量，包括FlexRAM。<br>0000b - 保 留<br>0001b - 保 留<br>0010b - 保 留<br>0011b - 保 留<br>0100b - 保 留<br>0101b - 保 留<br>0110b - 保 留<br>0111b - 保 留<br>1000b - 保 留<br>1001b - 保 留<br>1010b - 保留<br>1011b - 192 KB (S32K148), 96 KB (S32K146), 保留 (其他)<br>1100b - 保留<br>1101b - 48 KB (S32K144), 保留（其他） 1110b - 保留<br>1111b - 256 KB（S32K148），128 KB（S32K146），64 KB（S32K144），32 KB（S32K142）。 |

### 9.3.1.8.3 图示

| | |
|---|---|
| 15-12<br>识别 | 设备修订号<br><br>指定芯片的硅实现号。该芯片的REV ID为0。 |
| 11-8<br>包装 | 包装<br><br>指定芯片的可用封装选项。<br><br>0000b - 保留 0001b - 保留 |

*表继续在下一页...*

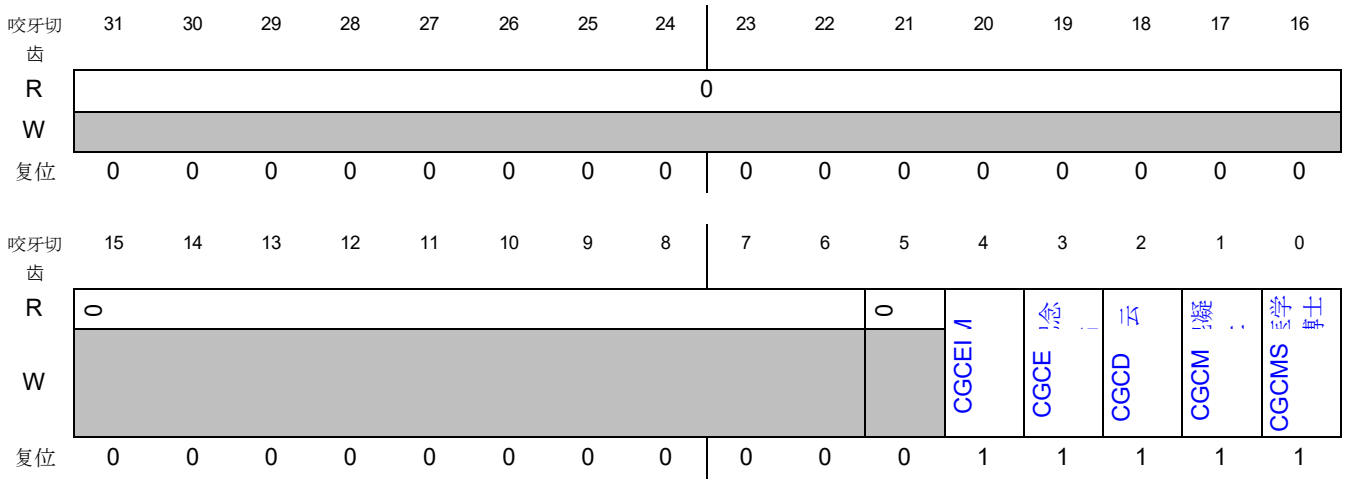
| 场地 | 职能 |
| --- | --- |
| | 0010b - 48 LQFP<br>0011b - 64 LQFP<br>0100b - 100 LQFP<br>0101b - 保留 0110b -<br>144 LQFP<br>0111b - 176 LQFP<br>1000b - 100 MAP BGA<br>1001b - 保留<br>1010b - 保留<br>1011b - 保留<br>1100b - 保留<br>1101b - 保留<br>1110b - 保留<br>1111b - 保留 |
| 7-0<br>特点 | 特点<br>指定芯片的支持功能。<br>**注意：**在通过PCC CGC位启用相应模块时钟的情况下，试图访问不可用功能的内存图区域时，不会出现任何传输错误终止。<br>1 特征是存在的<br>0 功能不存在<br>这个字段中的每一个比特都代表以下内容。<br>• 第7位：安全<br>• 第6位：ISO CAN-FD<br>• 位5：FlexIO<br>• Bit 4: QuadSPI<br>• 位3：ENET<br>• 位2：保留<br>• 位1：SAI<br>• 位0：保留 |

## 9.3.1.9 平台时钟选通控制寄存器（PLATCGC）。

### 9.3.1.9.1 偏移

| 注册 | 偏移 |
| --- | --- |
| PLATCGC | 40h |

### 9.3.1.9.2 图示




### 9.3.1.9.3 领域

| 场地 | 职能 |
|---|---|
| 31-6<br>- | 保留 |
| 5<br>- | 保留 |
| 4<br>CGCEIM | EIM时钟选通控制<br>控制通往EIM的时钟门控。<br>0b - 时钟禁用 1b - 时钟启用 |
| 3<br>CGCERM | ERM时钟选通控制<br>控制通往ERM的时钟门控。<br>0b - 时钟禁用 1b - 时钟启用 |
| 2<br>CGCDMA | DMA时钟选通控制<br>控制DMA模块的时钟门控。<br>0b - 时钟禁用 1b - 时钟启用 |
| 1<br>CGCMPU | MPU时钟选通控制<br>控制MPU模块的时钟门控。<br>0b - 时钟禁用 1b - 时钟启用 |

| | |
|---|---|
| 0<br>CGCMSCM | MSCM时钟选通控制<br>控制通往MSCMM的时钟门控。<br>0b - 时钟禁用 1b - 时钟启用 |

## 9.3.1.10 闪存配置寄存器1（FCFG1）。

### 9.3.1.10.1 偏移

| 注册 | 偏移 |
|---|---|
| FCFG1 | 4Ch |

### 9.3.1.10.2 职能

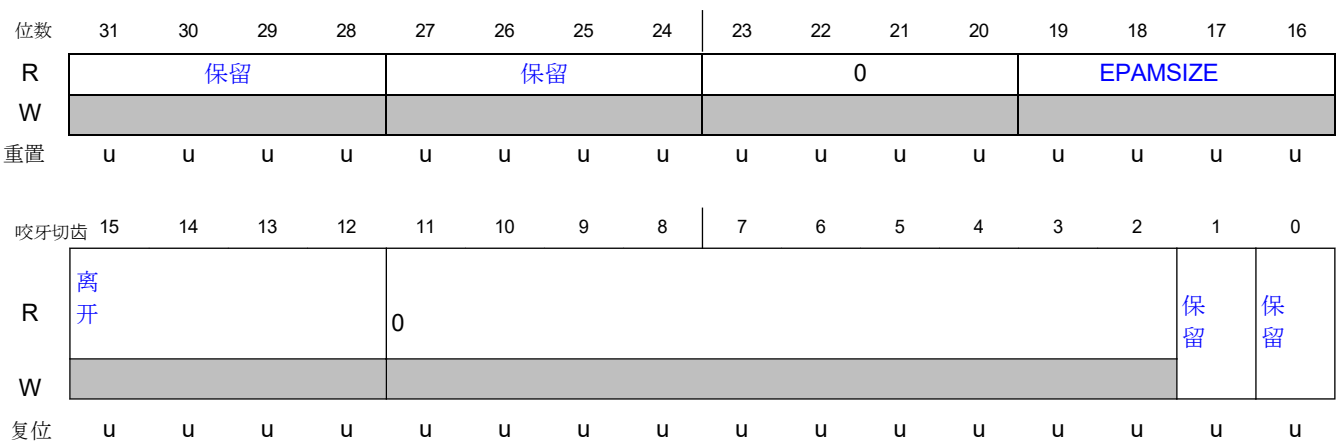
**注意事项**

该寄存器的DEPART字段的复位值在系统复位时从闪存IFR加载。

**注意事项**

对该寄存器的尝试性写入可能会导致不可预测的行为。

### 9.3.1.10.3 图示




### 9.3.1.10.4 领域

| 场地 | 职能 |
|---|---|
| 31-28<br>- | 保留 |
| 27-24<br>- | 保留 |

| | |
|---|---|
| 23-20 | 保留 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| - | |
| 19-16<br>EPAMSIZE | EEE SRAM SIZE<br>EEE SRAM数据大小。<br>0000b - 保留 0001b - 保留 0010b - 4 KB<br>0011b - 2 KB<br>0100b - 1 KB<br>0101b - 512 字节<br>0110b - 256 字节<br>0111b - 128 字节<br>1000b - 64 字节<br>1001b - 32 字节<br>1111b - 0 字节 |
| 15-12<br>离开 | FlexNVM分区<br>数据闪存/模拟EEPROM备份分割。见FTFC章节中的DEPART字段描述。 |
| 11-2<br>- | 保留 |
| 1<br>- | 保留 |
| 0<br>- | 保留 |

## 9.3.1.11 唯一识别寄存器高位（UIDH）。

### 9.3.1.11.1 偏移

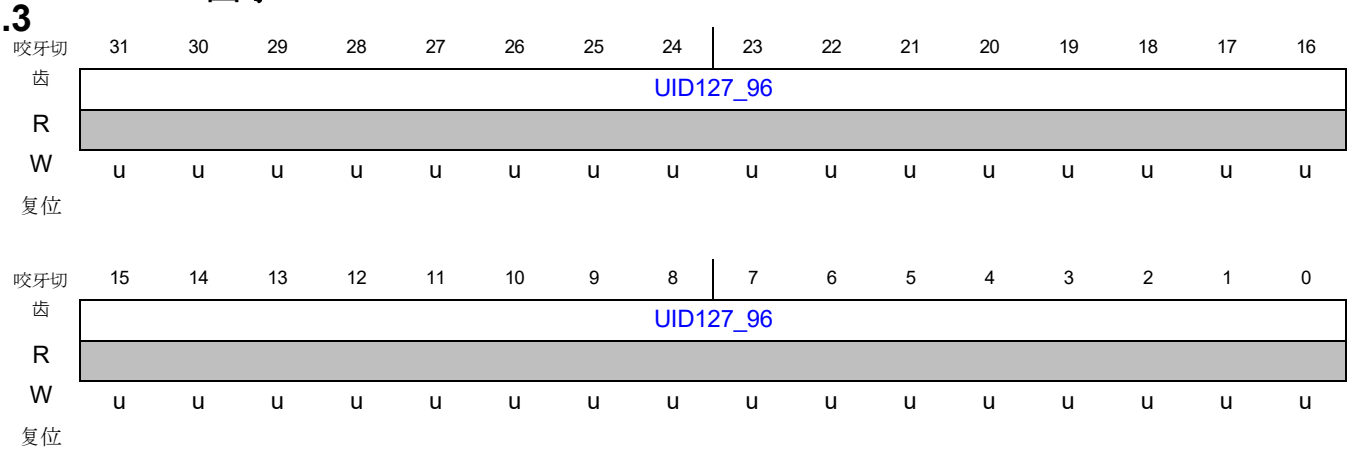
| 注册 | 偏移 |
|---|---|
| UIDH | 54h |

### 9.3.1.11.2 职能

注意事项

- UID127_96、UID95_64、UID63_32和UID31_0共同代表该设备的128位唯一识别号码。
- 这个寄存器的复位值在系统复位时从闪存IFR加载。

### 9.3.1.159.3 图示




### 9.3.1.11.4 领域

| 场地 | 职能 |
|---|---|
| 31-0<br>UID127_96 | 独特的识别<br><br>芯片的唯一标识。 |

## 9.3.1.12 独特的识别注册中高（UIDMH）。

### 9.3.1.12.1 偏移

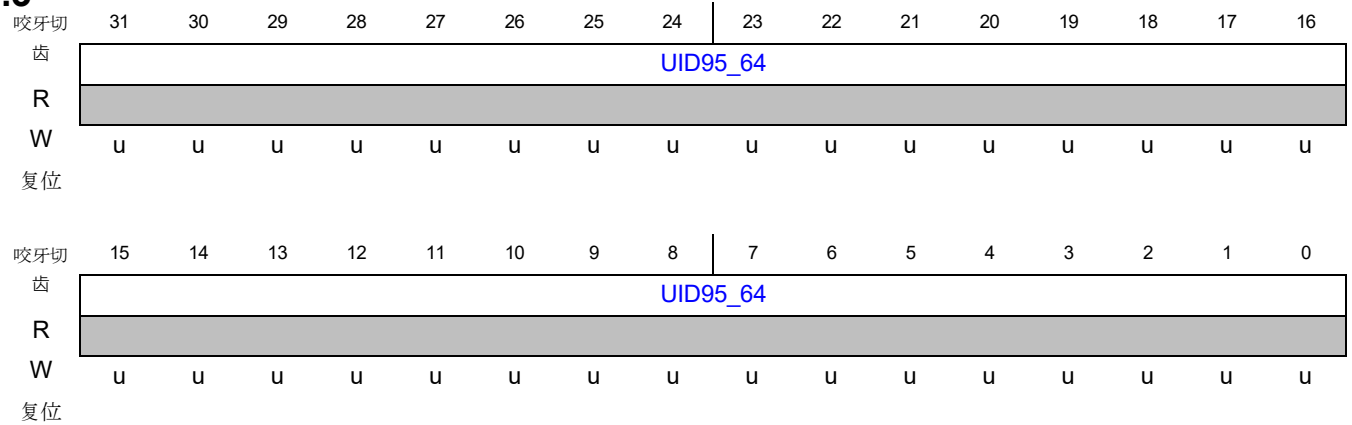
| 注册 | 偏移 |
|---|---|
| UIDMH | 58h |

### 9.3.1.12.2 职能

**注意事项**

这个寄存器的复位值在系统复位时从闪存IFR加载。

### 9.3.1.160.3 图示




### 9.3.1.12.4 领域

| 场地 | 职能 |
|---|---|
| 31-0<br>UID95_64 | 独特的识别<br>芯片的唯一标识。 |

## 9.3.1.13 唯一的识别寄存器中低端（UIDML）。

### 9.3.1.13.1 偏移

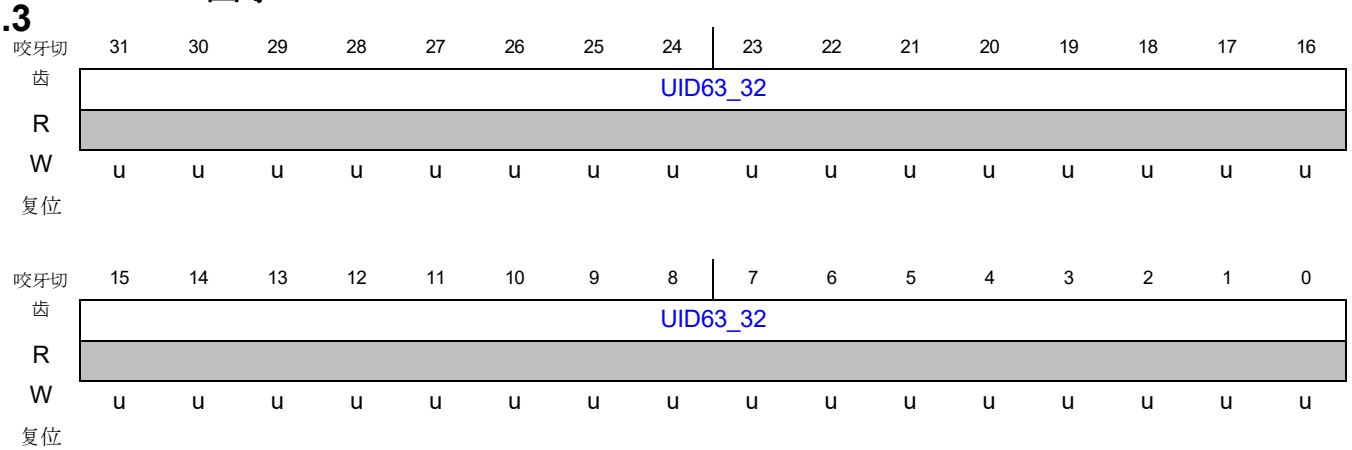
| 注册 | 偏移 |
|---|---|
| UIDML | 5Ch |

### 9.3.1.13.2 职能

**注意事项**

这个寄存器的复位值在系统复位时从闪存IFR加载。

### 9.3.1.161.3 图示




### 9.3.1.13.4 领域

| 场地 | 职能 |
|---|---|
| 31-0<br>UID63_32 | 独特的识别<br>芯片的唯一标识。 |

## 9.3.1.14 唯一识别寄存器低电平（UIDL）

### 9.3.1.14.1 偏移

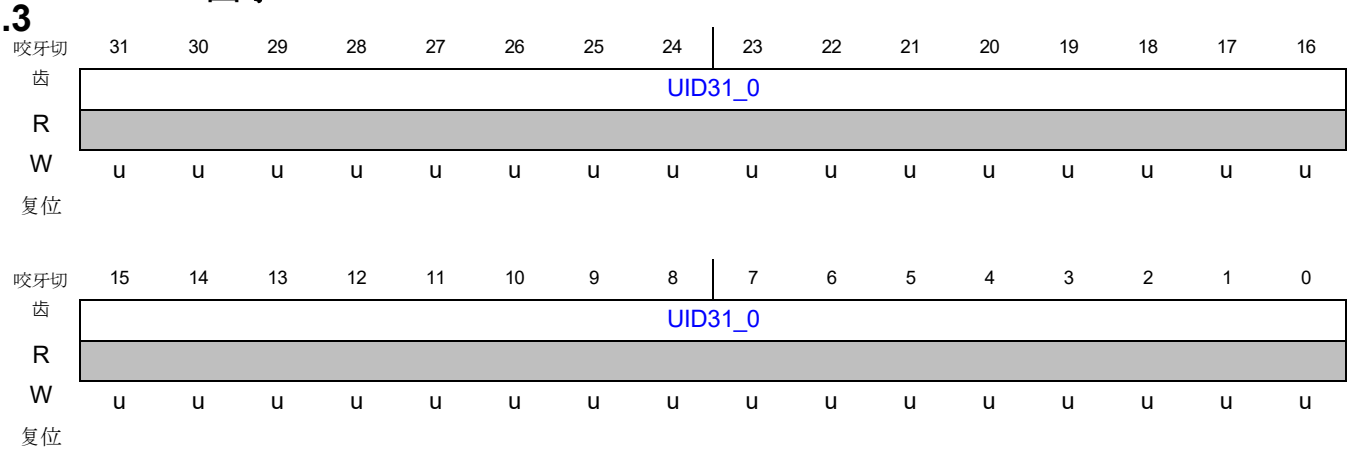
| 注册 | 偏移 |
|---|---|
| UIDL | 60h |

### 9.3.1.14.2 职能

**注意事项**

这个寄存器的复位值在系统复位时从闪存IFR加载。

### 9.3.1.162.3 图示




### 9.3.1.14.4 领域

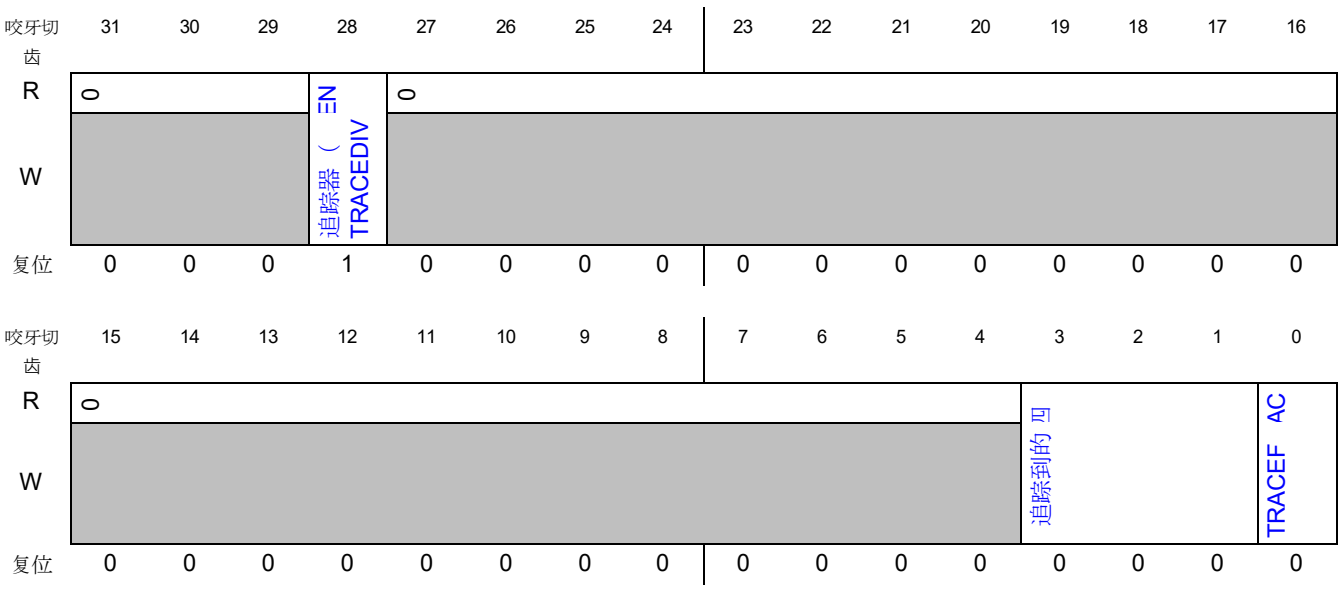
| 场地 | 职能 |
|---|---|
| 31-0<br>UID31_0 | 独特的识别<br><br>芯片的唯一标识。 |

## 9.3.1.15 系统时钟分频器寄存器4（CLKDIV4）。

### 9.3.1.15.1 偏移

| 注册 | 偏移 |
|---|---|
| 辽宁省 | 68h |

### 9.3.1.15.2 图示




### 9.3.1.15.3 领域

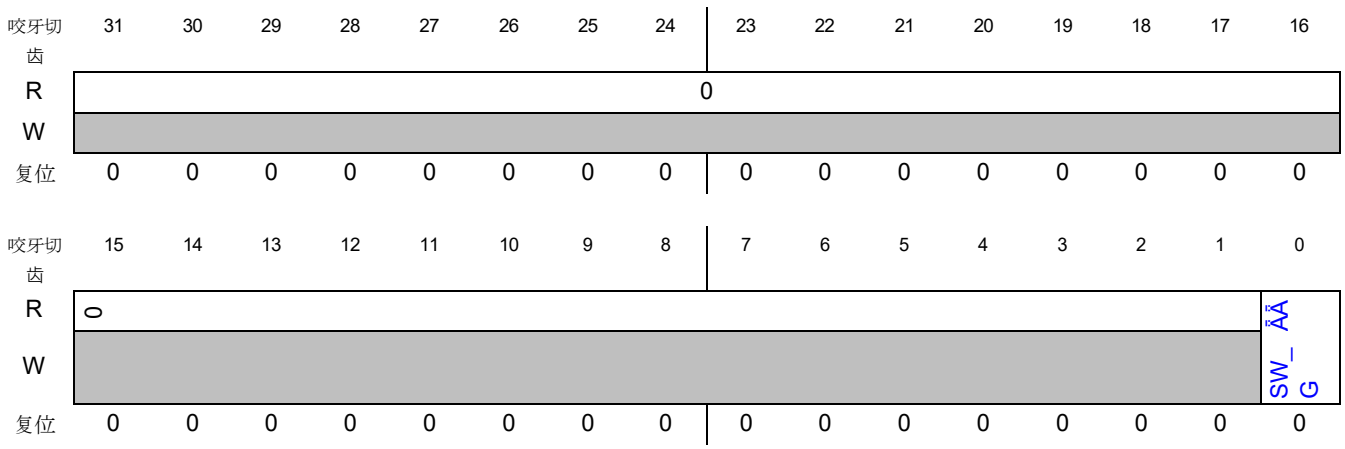
| 场地 | 职能 |
|---|---|
| 31-29<br>- | 保留 |
| 28<br>翠克迪文<br>(TRACEDIVEN) | 调试跟踪分割器控制<br>这个字段控制调试跟踪分割器。<br>0b - 禁用调试跟踪分配器 1b - 启用<br>调试跟踪分配器 |
| 27-4<br>- | 保留 |
| 3-1<br>追踪器（<br>TRACEDIV | 跟踪时钟分频器值 要配置TRACEDIV，必须先禁用TRACEDIVEN，然后在设置TRACEDIV后再启用它。<br>该字段设置作为跟踪时钟源的小数时钟分频器的除法值。跟踪时钟的源时钟由<br>SIM_CHIPCTL[TRACECLK_SEL]设置。分频器输出时钟=分频器输入时钟<br>*[(TRACEFRAC+1)/(TRACEDIV+1)]。 |
| 0<br>淘宝网 | 追踪时钟分频器分数 要配置TRACEDIV和TRACEFRAC，必须先清除TRACEDIVEN以禁用追踪时钟分频功能。<br>该字段设置作为跟踪时钟源的小数时钟分频器的除法值。跟踪时钟的源时钟由<br>SIM_CHIPCTL[TRACECLK_SEL]设置。分频器输出时钟=分频器输入时钟<br>*[(TRACEFRAC+1)/(TRACEDIV+1)]。 |

### 9.3.1.16 杂项控制寄存器1（MISCTRL1）。

## 9.3.1.16.1 偏移

| 注册 | 偏移 |
|---|---|
| 呼叫中心：MISCTRL1 | 6Ch |

## 9.3.1.16.2 图示




## 9.3.1.16.3 领域

| 场地 | 职能 |
|---|---|
| 31-1<br>- | 保留 |
| 0<br>SW_TRG | 软件触发到TRGMUX。写入该位可以通过TRGMUX产生软件触发到外设（参考图：触发器的互连性）。 |

# 章 10
# 端口控制和中断 (PORT)

## 10.1 芯片特定的PORT 信息

并非所有 PORT_PCRn
寄存器中提到的引脚控制设置对所有引脚都是可配置的。请参考《参考手册》所附 IO 信号描述输入复用表的 "IO 信号表 "标签中的 "可配置位 "字段。除了 "可配置位 "字段外，其他的位都是保留的，不应该从复位值中改变。

与复位焊盘相对应的PCR位是非粘性位，在功能复位时，该引脚的复位功能将被恢复。在进入任何ALT功能之前，相应焊盘的PCR应该被正确配置。

在进入模拟模式之前（对于有模拟功能的相应焊盘，ALT0对应PORT_PCRn[MUX]= 3'b000），应在相应的PCR寄存器中将PUE和PUS配置为0。

PORT_PCRn[PFE]只对 PTA5 和 PTD3
进行配置。参见《参考手册》所附的IO信号描述输入复用表。这些的 PFE 只应在 ALT7 模式下配置。对于其他模式，PFE应保持为0。

在配置PORT_DFER寄存器之前，应将相应的PAD配置为禁用模式（ALT0）。

任何在RUN/VLPR模式下完成的焊盘配置在低功耗模式（STOP1,STOP2/VLPS）下都会保留。

该设备不支持等待模式。

关于可用的电源模式的详情，请参见模块在可用的低功率模式下的操作。

### 10.1.1 PCRs的数量

在S32K14x系列的产品中，每个端口的PCR数量是不同的。下表显示了每个产品上每个端口的PCR数量。内存图和寄存器定义记录了实现的端口的超集数。

**表10-1.对每个产品的PCR**

| 港口 | PCR的数量1 | | | | | |
|---|---|---|---|---|---|---|
| | S32K116 | S32K118 | S32K142 | S32K144 | S32K146 | S32K148 |
| 端口A | 11 | 12 | 18 | 18 | 25 | 32 |
| B端口 | 9 | 10 | 18 | 18 | 27 | 32 |
| 端口C | 12 | 14 | 18 | 18 | 25 | 32 |
| 端口D | 7 | 10 | 18 | 18 | 27 | 32 |
| 端口E | 4 | 12 | 17 | 17 | 24 | 28 |

1.详见附件中的IO信号描述输入复用表。

### 10.1.2 I/O配置 顺序

1. 确保外设的引脚处于三态状态（缺省的复位状态）。
2. 在外围时钟控制器寄存器(PCC)中初始化外围时钟和外围特定的时钟配置。
3. 配置外围设备
4. 在外围时钟控制器寄存器（PCC_PORTx）中初始化外围引脚的端口时钟。
5. 在端口控制和中断寄存器（PORTx_PCRn）中配置外围引脚多路复用和功能。
6. 开始沟通

### 10.1.3 数字输入滤波器配置 序

1. 使用PORTx_DFCR和PORTx_DFWR为相应的引脚配置数字引脚过滤控制。
2. 启用PORTx_DFER[DFE]。
3. 配置PORTx_PCRn[MUX]为GPIO模式
4. 等待相当于PORTx_DFWR的延迟，以便滤波器使能突发事件传播
5. 通过配置PORTx_PCRn[MUX]，启用引脚的相应功能。

#### 10.1.3.1 使用GPIO中断时的数字输入滤波器配置顺序

1. 使用PORTx_DFCR和PORTx_DFWR为相应的引脚配置数字引脚过滤控制。
2. 启用PORTx_DFER[DFE]。
3. 配置PORTx_PCRn[MUX]为GPIO模式
4. 等待相当于PORTx_DFWR的延迟，以便滤波器使能突发事件传播
5. 通过配置PORTx_PCRn[IRQC]在相应的引脚上启用中断。

#### 10.1.3.2 使用NMI时的数字输入滤波器配置顺序

1. 使用PORTD_DFCR和PORTD_DFWR为NMI引脚配置数字引脚过滤控制。
2. 启用 PORTD_DFER[3]
3. 配置PORTD_PCR3[MUX]为GPIO模式
4. 等待相当于PORTD_DFWR的延迟，以使滤波器的故障传播。
5. 使用PORTD_PCR3[MUX]配置引脚为NMI模式。

在进入STOP/VLPS模式之前，应将滤波器时钟配置为LPO128K_CLK，并在唤醒时重新配置，如果需要的话。

如果在STOP/VLPS模式下不需要滤波，应使用PORTx_DFER禁用滤波，并在唤醒时重新配置，如果需要的话。如果在STOP/VLPS模式下不需要滤波，而需要在唤醒后不重新配置PORTx_DFER（用总线时钟作为滤波时钟），则应确保。

*cycles_tISR >> (PORTx_DFWR+3)*(SCG_xCCR[DIVBUS]+1)*

其中 *cycles_tISR*：从系统进入 RUN 模式到相应引脚的PORTx_PCRn[ISF]清零为止的中断服务程序（ISR）的核心时钟周期。

(如果需要，应在ISR中加入足够的延迟，以实现上述目标。）

## 10.2 简介

## 10.3 概述

端口控制和中断（PORT）模块提供对端口控制、数字滤波和外部中断功能的支持。

大多数功能可以为32位端口中的每个引脚独立配置，并影响引脚，而不考虑其引脚复用状态。

每个端口都有一个PORT模块的实例。并非每个端口内的所有引脚都在一个特定的设备上实现。

### 10.3.1 特点

端口模块有以下特点。

- 引脚中断
  - 每个引脚的中断标志和启用寄存器
  - 支持每个引脚的边沿敏感（上升、下降、两者）或电平敏感（低、高）配置
  - 支持每个引脚配置的中断或DMA请求
  - 低功耗模式下的异步唤醒
  - 引脚中断在所有的数字引脚复用模式中都是有效的
- 数字输入滤波器
  - 每个引脚的数字输入滤波器，可由任何数字外围设备混入引脚使用
  - 每个引脚都有单独的启用或旁路控制区域
  - 可选择数字输入滤波器的时钟源，滤波器尺寸的分辨率为5位
  - 在所有的数字引脚复用模式下都能发挥作用
- 港口控制
  - 支持上拉、下拉和停拉的独立拉动控制区域
  - 支持高和低驱动力的个体驱动力领域
  - 单个输入无源滤波器字段，支持单个输入无源滤波器的启用和停用。
  - 单独的多路复用控制领域，支持模拟或引脚禁用、GPIO和多达六个芯片特定的数字功能
  - 垫子配置字段在所有的数字引脚复用模式中都有作用。

### 10.3.2 的操作模式

#### 10.3.2.1 运行模式

在运行模式下，该端口正常运行。

#### 10.3.2.2 等待模式

在Wait模式下，PORT继续正常工作，如果检测到一个启用的中断，可以配置为退出低功耗模式。在等待模式下，仍会产生DMA请求，但不会导致退出低功率模式。

#### 10.3.2.3 停止模式

在停止模式下，如果检测到一个启用的中断，该端口可以被配置为通过一个异步唤醒信号退出低功耗模式。

在停止模式下，数字输入滤波器被绕过，除非它们被配置为从LPO时钟源运行。

#### 10.3.2.4 调试模式

在调试模式下，PORT正常运行。

## 10.4 外部信号 描述

这里找到的表格描述了PORT外部信号。

**表10-2.信号属性**

| 命名 | 职能 | 输入/输出 | 复位 | 拉动 |
|---|---|---|---|---|
| PORTx[31:0] | 外部中断 | 输入/输出 | 0 | - |

**注意事项**

并非每个端口内的所有引脚都在每个设备上实现。

## 10.5 详细的信号 描述

这里的表格包含了PORT接口的详细信号描述。

**表10-3.端口接口-详细的信号描述**

| 信号 | 输入/ | 描述 |
|---|---|---|

| | 输出 | |
|---|---|---|
| PORTx[31:0] | 输入/输出 | 外部中断。 |
| | | 国家含义 申明针脚为逻辑1。<br>负数引脚为逻辑0。 |

*表继续在下一页...*

表10-3.端口接口-详细的信号描述（续）

| 信号 | 输入/输出 | 描述 | |
|---|---|---|---|
| | | 时间安排 | 赋值--可在任何时候发生，并可与系统时钟异步赋值。<br>否定--可在任何时候发生，并可对系统时钟进行异步断言。 |

## 10.6 内存地图和寄存器 定义

任何在有效内存地图之外对端口内存空间的读或写访问都会导致总线错误。所有的寄存器访问都以零等待状态完成。

端口内存图

| 地址偏移（十六进制） | 注册名称 | 宽度（单位：比特） | 访问 | 重置值 | 节/页 |
|---|---|---|---|---|---|
| 0 | 引脚控制寄存器n（PORT_PCR0）。 | 32 | R/W | 见本节 | 10.6.1/172 |
| 4 | 引脚控制寄存器n (PORT_PCR1) | 32 | R/W | 见本节 | 10.6.1/172 |
| 8 | 引脚控制寄存器n（PORT_PCR2）。 | 32 | R/W | 见本节 | 10.6.1/172 |
| C | 引脚控制寄存器n（PORT_PCR3）。 | 32 | R/W | 见本节 | 10.6.1/172 |
| 10 | 引脚控制寄存器n（PORT_PCR4）。 | 32 | R/W | 见本节 | 10.6.1/172 |
| 14 | 引脚控制寄存器n（PORT_PCR5）。 | 32 | R/W | 见本节 | 10.6.1/172 |
| 18 | 引脚控制寄存器n（PORT_PCR6）。 | 32 | R/W | 见本节 | 10.6.1/172 |
| 1C | 引脚控制寄存器n（PORT_PCR7）。 | 32 | R/W | 见本节 | 10.6.1/172 |
| 20 | 引脚控制寄存器n（PORT_PCR8）。 | 32 | R/W | 见本节 | 10.6.1/172 |
| 24 | 引脚控制寄存器n（PORT_PCR9）。 | 32 | R/W | 见本节 | 10.6.1/172 |
| 28 | 引脚控制寄存器n（PORT_PCR10）。 | 32 | R/W | 见本节 | 10.6.1/172 |
| 2C | 引脚控制寄存器n（PORT_PCR11）。 | 32 | R/W | 见本节 | 10.6.1/172 |
| 30 | 引脚控制寄存器n（PORT_PCR12）。 | 32 | R/W | 见本节 | 10.6.1/172 |
| 34 | 引脚控制寄存器n（PORT_PCR13）。 | 32 | R/W | 见本节 | 10.6.1/172 |
| 38 | 引脚控制寄存器n（PORT_PCR14）。 | 32 | R/W | 见本节 | 10.6.1/172 |
| 3C | 引脚控制寄存器n（PORT_PCR15）。 | 32 | R/W | 见本节 | 10.6.1/172 |
| 40 | 引脚控制寄存器n（PORT_PCR16）。 | 32 | R/W | 见本节 | 10.6.1/172 |
| 44 | 引脚控制寄存器n（PORT_PCR17）。 | 32 | R/W | 见本节 | 10.6.1/172 |

| | | | | | |
|---|---|---|---|---|---|
| 48 | 引脚控制寄存器n（PORT_PCR18）。 | 32 | R/W | 见本节 | 10.6.1/172 |
| 4C | 引脚控制寄存器n（PORT_PCR19）。 | 32 | R/W | 见本节 | 10.6.1/172 |
| 50 | 引脚控制寄存器n（PORT_PCR20）。 | 32 | R/W | 见本节 | 10.6.1/172 |
| 54 | 引脚控制寄存器n（PORT_PCR21）。 | 32 | R/W | 见本节 | 10.6.1/172 |
| 58 | 引脚控制寄存器n（PORT_PCR22）。 | 32 | R/W | 见本节 | 10.6.1/172 |
| 5C | 引脚控制寄存器n（PORT_PCR23）。 | 32 | R/W | 见本节 | 10.6.1/172 |
| 60 | 引脚控制寄存器n（PORT_PCR24）。 | 32 | R/W | 见本节 | 10.6.1/172 |

*表继续在下一页...*

## 端口内存图（续）

| 地址偏移（十六进制） | 注册名称 | 宽度（单位：比特） | 访问 | 重置值 | 节/页 |
|---|---|---|---|---|---|
| 64 | 引脚控制寄存器n（PORT_PCR25）。 | 32 | R/W | 见本节 | 10.6.1/172 |
| 68 | 引脚控制寄存器n（PORT_PCR26）。 | 32 | R/W | 见本节 | 10.6.1/172 |
| 6C | 引脚控制寄存器n（PORT_PCR27）。 | 32 | R/W | 见本节 | 10.6.1/172 |
| 70 | 引脚控制寄存器n（PORT_PCR28）。 | 32 | R/W | 见本节 | 10.6.1/172 |
| 74 | 引脚控制寄存器n（PORT_PCR29）。 | 32 | R/W | 见本节 | 10.6.1/172 |
| 78 | 引脚控制寄存器n（PORT_PCR30）。 | 32 | R/W | 见本节 | 10.6.1/172 |
| 7C | 引脚控制寄存器n（PORT_PCR31）。 | 32 | R/W | 见本节 | 10.6.1/172 |
| 80 | 全局引脚控制低寄存器（PORT_GPCLR）。 | 32 | W（总是读作0） | 0000_0000h | 10.6.2/175 |
| 84 | 全局引脚控制高位寄存器（PORT_GPCHR）。 | 32 | W（总是读作0） | 0000_0000h | 10.6.3/175 |
| 88 | 全局中断控制低寄存器（PORT_GICLR）。 | 32 | W（总是读作0） | 0000_0000h | 10.6.4/176 |
| 8C | 全局中断控制高位寄存器（PORT_GICHR）。 | 32 | W（总是读作0） | 0000_0000h | 10.6.5/176 |
| A0 | 中断状态标志寄存器（PORT_ISFR）。 | 32 | w1c | 0000_0000h | 10.6.6/177 |
| C0 | 数字滤波器启用寄存器（PORT_DFER）。 | 32 | R/W | 0000_0000h | 10.6.7/178 |
| C4 | 数字滤波器时钟寄存器（PORT_DFCR）。 | 32 | R/W | 0000_0000h | 10.6.8/178 |
| C8 | 数字滤波宽度寄存器（PORT_DFWR）。 | 32 | R/W | 0000_0000h | 10.6.9/179 |

## 10.6.1 引脚控制寄存器n (PORT_PCRn )

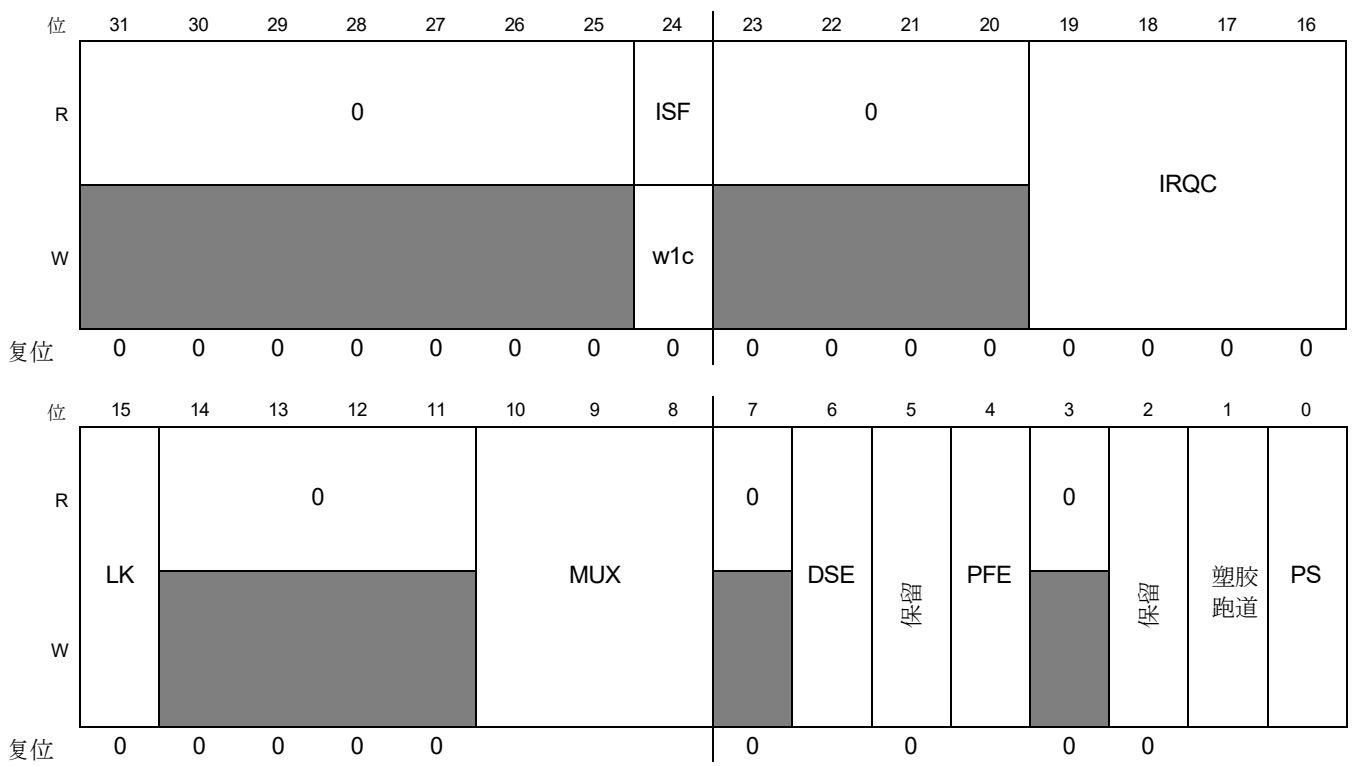
### 注意事项

关于该器件的复位值，请参见信号复用和引脚分配一章。

关于每个引脚的可用功能的详情，请参见GPIO配置部分。

不要修改与那些在缩小的引脚封装产品中没有的引脚相关的引脚配置寄存器。
封装中没有的未绑定引脚默认是禁用的，以防止它们消耗功率。

地址。0h基数+0h偏移量+（4d×i），其中i=0d至31d




* 注意事项。

- MUX字段。因端口而异。每个端口的重置值见信号复用和信号描述一章。
- DSE字段。因端口而异。关于每个端口的重置值，请参见信号复用和信号描述一章。
- PFE字段。因端口而异。每个端口的复位值见信号复用和信号描述一章。
- PE字段。因端口而异。每个端口的重置值见信号复用和信号描述一章。
- PS字段。因端口而异。每个端口的复位值见信号复用和信号描述一章。

## PORT_PCRn字段描述

| 场地 | 描述 |
|---|---|
| 31-25<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 24<br>ISF | 中断状态标志<br>引脚中断配置在所有数字引脚复用模式下都有效。<br>0 未检测到配置的中断。<br>1 配置的中断被检测到。如果该引脚被配置为产生一个DMA请求，那么相应的标志将在请求的DMA传输完成后自动清除。否则，该标志将保持设置，直到一个逻辑1被写入该标志。如果该引脚被配置为电平敏感的中断，并且该引脚保持断言状态，那么该标志在被清除后立即被再次设置。 |
| 23-20<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 19-16<br>IRQC | 中断配置<br>引脚中断配置在所有数字引脚复用模式下都有效。相应的引脚被配置为产生中断/DMA请求，如下所示。<br>0000中断 状态标志（ISF）被禁用。<br>0001ISF标志和DMA请求在上升沿。0010ISF标志<br>和DMA请求在下降沿。0011ISF标志和DMA请求<br>在任一边缘。0100 保留。<br>0101 保留。<br>0110 保留。<br>0111 保留。<br>1000ISF标志和逻辑0时的中断。1001ISF<br>标志和上升沿的中断。1010ISF标志<br>和下降沿的中断。1011ISF标志和中断在任一<br>边缘。1100ISF 标志和逻辑1时的中断。<br>1101 保留。<br>1110 保留。<br>1111 保留。 |
| 15<br>LK | 锁定寄存器<br>0 引脚控制寄存器字段[15:0]未被锁定。<br>1 引脚控制寄存器字段[15:0]被锁定，在下一次系统复位前不能被更新。 |
| 14-11<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |

| | |
|---|---|
| 10-8<br>MUX | 引脚多路复用控制<br><br>不是所有的引脚都支持所有的引脚复用槽。未实现的引脚多路复用槽是保留的，可能导致为不同的引脚多路复用槽进行配置。<br><br>相应的引脚配置在以下引脚复用槽中，如下所示。<br><br>00 引脚禁用（备选0）（模拟）。<br>01 替代方案1（GPIO）。<br>010替代方案2（芯片专用）。 |

*表继续在下一页...*

## PORT_PCRn字段描述（续）

| 场地 | 描述 |
|---|---|
| | 011替代方案3（芯片专用）。<br>100 替代方案4（特定芯片）。<br>101 替代方案5（特定芯片）。<br>110 替代方案6（特定芯片）。<br>111 替代方案7（特定芯片）。 |
| 7<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 6<br>DSE | 启用驱动强度<br>驱动强度配置在所有数字引脚复用模式下都有效。<br>0 如果引脚被配置为数字输出，则在相应的引脚上配置了低驱动强度。<br>1 如果引脚被配置为数字输出，则在相应的引脚上配置了高驱动强度。 |
| 5<br>保留 | 这个字段是保留的。 |
| 4<br>PFE | 启用无源滤波器<br>无源滤波器的配置在所有数字引脚复用模式下都有效。<br>0 无源输入滤波器在相应的引脚上被禁用。<br>1 如果引脚被配置为数字输入，则在相应的引脚上启用无源输入滤波器。关于滤波器的特性，请参考器件数据表。 |
| 3<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 2<br>保留 | 这个字段是保留的。 |
| 1<br>塑胶跑道 | 拉动启用<br>拉动配置在所有数字引脚复用模式下都有效。<br>0 在相应的引脚上没有启用内部上拉或下拉电阻。<br>1 如果引脚被配置为数字输入，则在相应的引脚上启用内部上拉电阻或下拉电阻。 |
| 0<br>PS | 拉动选择<br>拉动配置在所有数字引脚复用模式下都有效。<br>0 如果相应的PE字段被设置，则在相应的引脚上启用内部下拉电阻。<br>1 如果相应的PE字段被设置，则在相应的引脚上启用内部上拉电阻。 |

## 10.6.2 全局引脚控制低位寄存器 (PORT_GPCLR)

该寄存器只支持32位的写操作。

地址。0h base + 80h offset = 80h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | | 0 | | | | | | | | | | | | | | | |
| W | GPWE | | | | | | | | | | | | | | | | GPWD | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**PORT_GPCLR字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>GPWE | 全局引脚写入功能<br><br>选择哪些引脚控制寄存器（15到0）位[15:0]随GPWD中的值更新。如果选择的引脚控制寄存器被锁定，那么对该寄存器的写入将被忽略。<br><br>0 相应的引脚控制寄存器不随GPWD中的值更新。<br>1 相应的引脚控制寄存器被更新为GPWD中的值。 |
| GPWD | 全局引脚写入数据<br><br>写入由GPWE选择的所有引脚控制寄存器位[15:0]的数值。 |

## 10.6.3 全局引脚控制高位寄存器 (PORT_GPCHR)

该寄存器只支持32位的写操作。

地址。0h基数+84h偏移量=84h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | | 0 | | | | | | | | | | | | | | | |
| W | GPWE | | | | | | | | | | | | | | | | GPWD | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**PORT_GPCHR字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>GPWE | 全局引脚写入功能<br><br>选择哪些引脚控制寄存器（31至16）位[15:0]随GPWD中的值更新。如果选择的引脚控制寄存器被锁定，那么对该寄存器的写入将被忽略。<br><br>0 相应的引脚控制寄存器不随GPWD中的值更新。 |

| | |
|---|---|
| | 1 相应的引脚控制寄存器被更新为GPWD中的值。 |
| GPWD | 全局引脚写入数据 |

*表继续在下一页...*

**PORT_GPCHR字段描述（续）**

| 场地 | 描述 |
|---|---|
| | 写入由GPWE选择的所有引脚控制寄存器位[15:0]的数值。 |

## 10.6.4 全局中断控制低寄存器 (PORT_GICLR)

该寄存器只支持32位的写操作。

地址。0h基数+88h偏移量=88h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | | 0 | | | | | | | | | | | | | | | |
| W | GIWD | | | | | | | | | | | | | | | | GIWE | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**PORT_GICLR字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>GIWD | 全局中断写数据<br><br>写入由GIWE选择的所有引脚控制寄存器位[31:16]的数值。 |
| GIWE | 全局中断写启用<br><br>选择哪些引脚控制寄存器（15到0）位[31:16]随GIWD中的值更新。<br><br>0 相应的引脚控制寄存器不随GPWD中的值更新。<br>1 相应的引脚控制寄存器被更新为GPWD中的值。 |

## 10.6.5 全局中断控制高位寄存器 (PORT_GICHR)

该寄存器只支持32位的写操作。

地址。0h基数+8Ch偏移量=8Ch

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | | 0 | | | | | | | | | | | | | | | |
| W | GIWD | | | | | | | | | | | | | | | | GIWE | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**PORT_GICHR字段描述**

| 场地 | 描述 |
|---|---|

| | |
|---|---|
| 31-16<br>GIWD | 全局中断写数据<br><br>写入由GIWE选择的所有引脚控制寄存器位[31:16]的数值。 |

*表继续在下一页...*

**PORT_GICHR字段描述（续）**

| 场地 | 描述 |
|---|---|
| GIWE | 全局中断写启用<br><br>选择哪些引脚控制寄存器（31到16）位[31:16]随GIWD中的值更新。<br><br>0 相应的引脚控制寄存器不随GPWD中的值更新。<br>1 相应的引脚控制寄存器被更新为GPWD中的值。 |

## 10.6.6 中断状态标志寄存器 (PORT_ISFR)

引脚中断配置在所有数字引脚复用模式下都有效。每个引脚的中断状态标志在相应的引脚控制寄存器中也是可见的，每个标志可以在任一位置被清除。

地址。0h基数+A0h偏移量=A0h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | ISF | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| W | w1c | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**PORT_ISFR字段描述**

| 场地 | 描述 |
|---|---|
| ISF | 中断状态标志<br><br>字段中的每一位都表示对与字段相同编号的配置中断的检测。<br><br>0 未检测到配置的中断。<br>1 配置的中断被检测到。如果该引脚被配置为产生一个DMA请求，那么相应的标志将在请求的DMA传输完成后自动清除。否则，该标志将保持设置，直到一个逻辑1被写入该标志。如果该引脚被配置为电平敏感的中断，并且该引脚保持断言状态，那么该标志在被清除后立即被再次设置。 |

## 10.6.7 数字滤波器启用寄存器 (PORT_DFER)

数字滤波器的配置在所有的数字引脚复用模式下都有效。

地址。0h基数+C0h偏移量=C0h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R<br>W | DFE | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**PORT_DFER字段描述**

| 场地 | 描述 |
|---|---|
| DFE | 启用数字滤波器<br><br>数字滤波器的配置在所有的数字引脚复用模式下都有效。每个数字滤波器的输出在系统复位时和数字滤波器被禁用时都被重置为零。字段中的每一位都启用与该字段相同编号的数字滤波器。<br><br>0 数字滤波器在相应的引脚上被禁用，数字滤波器的输出被重置为零。<br>1 如果引脚被配置为数字输入，则在相应的引脚上启用数字滤波器。 |

## 10.6.8 数字滤波器时钟寄存器 (PORT_DFCR)

数字滤波器的配置在所有的数字引脚复用模式下都有效。

地址。0h基数 + C4h偏移量 = C4h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 位 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | CS |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**PORT_DFCR字段描述**

| 场地 | 描述 |
|---|---|
| 31-1<br>保留 | 这个字段是保留的。 |

| | |
|---|---|
| | 这个只读字段是保留的，其值始终为0。 |
| 0<br>CS | 时钟源 |

*表继续在下一页...*

**PORT_DFCR字段描述（续）**

| 场地 | 描述 |
|---|---|
| | 数字滤波器的配置在所有数字引脚复用模式下都有效。配置数字输入滤波器的时钟源。改变滤波器的时钟源必须在所有数字滤波器被禁用的情况下才能进行。<br><br>0 数字滤波器的时钟由总线时钟决定。<br>1 数字滤波器的时钟是由LPO时钟控制的。 |

## 10.6.9 数字滤波宽度寄存器 (PORT_DFWR)。

数字滤波器的配置在所有的数字引脚复用模式下都有效。

地址。0h基数 + C8h偏移量 = C8h

| 位 | 31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 | 15 14 13 12 11 10 9 8 | 7 6 5 4 3 2 1 0 |
|---|---|---|---|
| R | 0 | | FILT |
| W | | | |
| 复位 | 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 | 0 0 0 0 0 0 0 0 | 0 0 0 0 0 0 0 0 |

**PORT_DFWR字段描述**

| 场地 | 描述 |
|---|---|
| 31-5<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| FILT | 过滤器长度<br><br>数字滤波器的配置在所有数字引脚复用模式下都有效。配置数字滤波器吸收已启用的数字滤波器的最大突变大小（以时钟周期为单位）。长于该寄存器设置的突波将通过数字滤波器，而等于或小于该寄存器设置的突波将被过滤掉。改变滤波器的长度必须在所有的滤波器被禁用后才能进行。 |

# 10.7 功能 描述

## 10.7.1 引脚 控制

每个端口引脚都有一个相应的引脚控制寄存器PORT_PCRn，与之相关。

引脚控制寄存器的上半部分配置引脚的能力，在上升/下降沿或两个沿以及端口引脚上发生的逻辑电平上，中断CPU或请求DMA传输。它还包括一个标志，表

示中断已经发生。

引脚控制寄存器的下半部分为32位端口内的每个引脚配置以下功能。

- 上拉或下拉启用
- 驱动力
- 启用无源输入滤波器
- 引脚多路复用模式

这些功能适用于所有的数字引脚复用模式，个别外设不会覆盖引脚控制寄存器中的配置。例如，如果一个I2C功能在一个引脚上被启用，这不会覆盖该引脚的上拉配置。

当引脚多路复用模式被配置为模拟或被禁用时，该引脚的所有数字功能都被禁用。这包括上拉和下拉启用、输出缓冲器启用、输入缓冲器启用和无源滤波器启用。

LK位（引脚控制寄存器PCRn的第15位）允许每个引脚的配置被锁定，直到下次系统复位。锁定后，对该引脚控制寄存器下半部分的写入将被忽略，尽管在试图写入一个锁定的寄存器时不会产生总线错误。

当端口模块被禁用时，每个引脚控制寄存器的配置被保留。

每当一个引脚被配置在任何数字引脚复用模式中，该引脚的输入缓冲器就会被启用，允许通过相应的GPIO端口数据输入寄存器（GPIO_PDIR）读取该引脚的状态，或者允许产生一个引脚中断或DMA请求。如果一个引脚在其输入缓冲区被启用时曾经浮动，那么这会导致功耗的增加，必须避免。一个引脚的浮动可能是由于一个没有连接的输入引脚或一个已经三态的输出引脚（输出缓冲器被禁用）。

启用内部拉电阻（或实施外部拉电阻）将确保引脚在其输入缓冲器被启用时不会浮动；注意，只要输出缓冲器被启用，内部拉电阻就会自动禁用，允许拉动启用位保持设置。将引脚多路复用模式配置为禁用或模拟，将禁用引脚的输入缓冲器，从而实现最低的功耗。

## 10.7.2 全局引脚 控制

两个全局引脚控制寄存器允许一个寄存器的写入更新多达16个引脚控制寄存器的下半部分，所有的值都相同。被锁定的寄存器不能用全局引脚控制寄存器写入。

全局引脚控制寄存器的设计是为了使软件能够快速配置一个端口内的多个引脚，用于同一外设功能。然而，中断功能不能用全局引脚控制寄存器进行配置。

全局引脚控制寄存器是只写不读的寄存器，读数始终为0。

### 10.7.3 全局中断 控制

两个全局中断控制寄存器允许单一的寄存器写入来更新多达16个引脚控制寄存器的上半部分，所有的值都是一样的。

全局中断控制寄存器的设计是为了使软件能够快速配置一个端口内的多个引脚，以实现相同的中断配置。然而，引脚控制功能不能用全局中断控制寄存器来配置。

全局中断控制寄存器是只写不读的寄存器，总是读为0。

### 10.7.4 外部 中断

PORT模块的外部中断能力在所有数字引脚复用模式下都可用，前提是PORT模块被启用。

每个引脚都可以单独配置为以下任何一种外部中断模式。

- 中断禁用，复位后缺省
- 有源高电平敏感中断
- 有源低电平敏感中断
- 上升沿敏感中断
- 下降沿敏感中断
- 上升沿和下降沿敏感中断
- 上升沿敏感的DMA请求
- 下降沿敏感的DMA请求
- 对上升沿和下降沿敏感的DMA请求

当在引脚上或在数字输入滤波器的输出端（如果数字输入数字滤波器被启用）检测到配置的边沿或电平时，中断状态标志被设置。当不处于停止模式时，输入首先与总线时钟同步，以检测配置的电平或边缘转换。

端口模块产生一个单一的中断，当该端口的任何启用的中断的中断状态标志被设置时，该中断就会启动。在 PORT_ISFR 或 PORT_PCRn 寄存器中的 ISF 标志上写一个逻辑 1
后，所有启用的中断的中断状态标志都被清除，这时中断会被否定。

端口模块产生一个单一的DMA请求，当该端口中任何已启用的DMA请求的中断状态标志被设置时，该请求就会被提示。在DMA传输完成后，DMA请求会被否定，因为这将清除所有启用的DMA请求的中断状态标志。

在停止模式期间，如果检测到所需的电平或边沿，任何启用的中断的中断状态标志被异步设置。这也会产生一个异步唤醒信号以退出低功耗模式。

### 10.7.5 数字 滤波器

如果PORT模块被启用，PORT模块的数字滤波功能在所有数字Pin Muxing模式中都可用。

一个端口内所有数字滤波器使用的时钟可以在总线时钟或LPO时钟之间进行配置。只有当该端口的所有数字滤波器被禁用时，才能改变这一选择。如果一个端口的数字滤波器被配置为使用总线时钟，那么在停止模式期间，数字滤波器被旁路。当数字滤波器被旁路时，每个数字滤波器的输出总是等于输入引脚，但数字滤波器的内部状态保持不变，不会因为输入引脚的任何变化而更新。

在一个端口内，所有启用的数字滤波器的时钟大小的滤波宽度是相同的，只有在该端口的所有数字滤波器被禁用时才必须改变。

每个数字滤波器的输出在系统复位后和数字滤波器被禁用时为逻辑零。在一个数字滤波器被启用后，输入被同步到滤波器的时钟，总线时钟或LPO时钟。如果同步的输入和数字滤波器的输出在等于滤波器宽度寄存器配置的滤波器时钟周期内保持不同，那么数字滤波器的输出就会更新为等于同步的滤波器输入。

通过数字滤波器的最大延迟等于三个滤波器时钟周期加上滤波器宽度配置寄存器
。

# 章 11
# 通用输入/输出 (GPIO)

## 11.1 芯片特定的GPIO 信息

### 11.1.1 实例化 信息

支持端口控制和中断模块功能，每个32针的端口将支持一个中断。并非所有的引脚都能在所支持的每个封装上使用。参见《参考手册》所附的IO信号描述输入复用表，以了解不同封装支持哪些引脚的细节。参见端口控制和中断（PORT），了解如何控制端口的细节。

关于可用的电源模式的详情，请参见模块在可用的低功率模式下的操作。

### 11.1.2 GPIO端口内存 地图

该芯片实现了GPIOs
GPIOA至GPIOE的五个实例。本章中的GPIO寄存器描述是通用的。请参考下面的表格，看看地址是如何分配给设备内存图中的每个GPIO实例的。

**表11-1.GPIO端口内存图**

| 起始地址 | 港口 |
|---|---|
| 基数+0h | 港口A |
| 基数+40h | 港口B |
| 基数+80h | 港口C |
| 基数+C0h | 港口D |
| 基数+100h | 港口E |

**注意**

- 在S32K14x中，GPIO是被内核通过0x400F_F000的横条接口访问。
- 在S32K11x中，GPIO是多端口的，可以在零等待状态下由核心直接访问基地址0xF800_0000。它也可以被内核通过横条/AIPS接口在0x400F_F000和地址0x4000_F000的别名槽（15）访问。

## 11.2 简介

通用输入和输出（GPIO）模块通过零等待状态接口与处理器内核进行通信，以实现最大的引脚性能。GPIO寄存器支持8位、16位或32位访问。

当引脚被配置为GPIO功能时，GPIO数据方向和输出数据寄存器控制每个引脚的方向和输出数据。当引脚被配置为任何数字功能时，GPIO输入数据寄存器显示每个引脚的逻辑值，前提是该引脚的相应端口控制和中断模块被启用。

通过为每个端口输出数据寄存器增加设置、清除和切换的只写寄存器，支持对通用输出的高效位操作。

### 11.2.1 特点

GPIO模块的特点包括。

- 在所有数字引脚多路复用模式下，端口数据输入寄存器是可见的
- 带有相应设置/清除/切换寄存器的端口数据输出寄存器
- 端口数据方向寄存器

**注意事项**

GPIO模块的时钟由系统时钟提供。

### 11.2.2 的操作模式

下表描述了不同的操作模式以及GPIO模块在这些模式下的行为。

表11-2.操作模式

| 操作模式 | 描述 |
|---|---|
| 运行 | GPIO模块运行正常。 |
| 停止 | GPIO模块被禁用。 |
| 调试 | GPIO模块运行正常。 |

## 11.2.3 GPIO信号 描述

表11-3.GPIO信号描述

| GPIO信号描述 | 描述 | 输入/输出 |
|---|---|---|
| 波塔31-PORTA0 | 通用输入/输出 | 输入/输出 |
| PORTB31-PORTB0 | 通用输入/输出 | 输入/输出 |
| PORTC31-PORTC0 | 通用输入/输出 | 输入/输出 |
| PORTD31-PORTD0 | 通用输入/输出 | 输入/输出 |
| PORTE31-PORTE0 | 通用输入/输出 | 输入/输出 |

### 注意事项

并非每个端口内的所有引脚都在每个器件上实现。关于器件中可用的GPIO端口数量，请参见信号复用章节。

### 11.2.3.1 详细的信号描述

表11-4.GPIO接口-详细的信号描述

| 信号 | 输入/输出 | 描述 | |
|---|---|---|---|
| 波塔31-PORTA0<br>PORTB31-PORTB0<br>PORTC31-PORTC0<br>PORTD31-PORTD0<br>PORTE31-PORTE0 | 输入/输出 | 通用输入/输出 | |
| | | 国家含义 | 赋值。该引脚为逻辑1。<br>脱空。该引脚为逻辑0。 |
| | | 时间安排 | 赋值。当输出时，该信号发生在系统时钟的上升沿。对于输入，它可以在任何时候发生，并且输入可以与系统时钟异步进行。 |

| | | | |
|---|---|---|---|
| | 注意事项 | | 脱机。当输出时，该信号发生在系统时钟的上升沿。对于输入来说，它可以在任何时候发生，并且输入可以与系统时钟异步进行。 |

**注意事项**

并非每个端口内的所有引脚都在每个器件上实现。关于器件中可用的GPIO端口数量，请参见信号复用章节。

## 11.3 内存图和寄存器 定义

任何对GPIO内存空间的读或写访问如果超出了有效的内存地图，就会导致总线错误。

**注意事项**

为了简单起见，每个GPIO端口的寄存器的宽度都是32位，对应于32个引脚。每个端口的实际引脚数（以及因此每个端口寄存器的可用控制位数）是特定于芯片的。请参考特定芯片的GPIO信息，看看每个端口的确切控制位。

### 11.3.1 GPIO寄存器 描述

#### 11.3.1.1 GPIO 内存地图

GPIOA 基准地址。400F_F000h

GPIOB 基准地址。400F_F040h

GPIOC 基准地址。400F_F080h

GPIOD 基地地址。400F_F0C0h

GPIOE 基准地址。400F_F100h

| 偏移 | 注册 | 宽度 (单位：比特) | 访问 | 重置值 |
|---|---|---|---|---|
| 0h | 端口数据输出寄存器（PDOR） | 32 | RW | 0000_0000h |
| 4h | 端口设置输出寄存器（PSOR） | 32 | 钨钢 | 0000_0000h |

| | | | | |
|---|---|---|---|---|
| 8h | 端口清除输出寄存器（PCOR） | 32 | 钨钢 | 0000_0000h |
| 丘吉尔 | 端口切换输出寄存器（PTOR）。 | 32 | 钨钢 | 0000_0000h |
| 10h | 端口数据输入寄存器（PDIR） | 32 | RO | 0000_0000h |

*表继续在下一页...*

| 偏移 | 注册 | 宽度<br>(单位：比特) | 访问 | 重置值 |
|---|---|---|---|---|
| 14h | 端口数据方向寄存器（PDDR） | 32 | RW | 0000_0000h |
| 18h | 端口输入禁用寄存器（PIDR） | 32 | RW | 0000_0000h |

## 11.3.1.2 端口数据输出寄存器（PDOR）

### 11.3.1.2.1 偏移

| 注册 | 偏移 |
|---|---|
| PDOR | 0h |

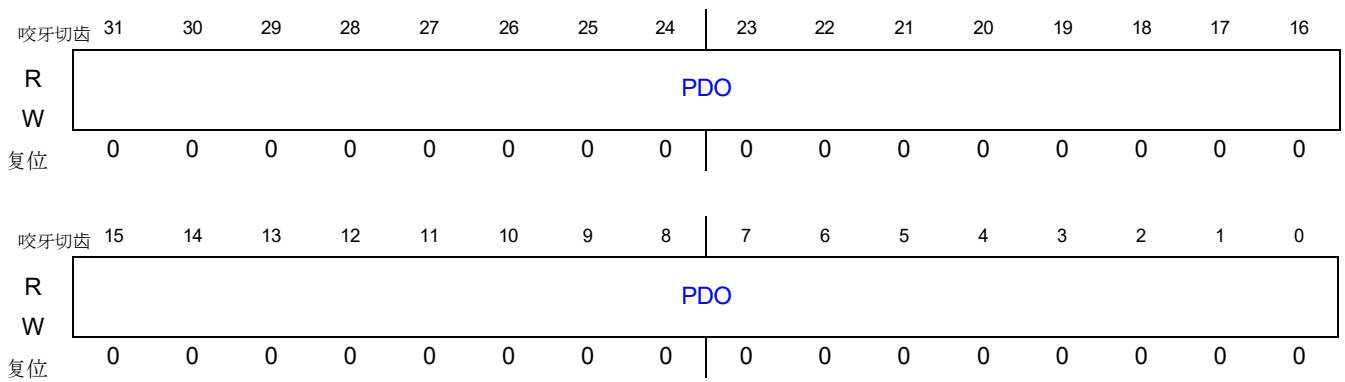
### 11.3.1.2.2 职能

该寄存器配置了每个通用输出引脚的逻辑电平。

**注意事项**

不要修改与你所选封装中没有的引脚相关的引脚配置寄存器。所有在你的封装中不可用的未绑定的引脚将默认为DISABLE状态，以达到最低的功耗。

### 11.3.1.2.3 图示



#### 11.3.1.2.4 领域

| 场地 | 职能 |
| --- | --- |
| 31-0<br>PDO | 端口数据输出<br>未绑定的引脚的寄存器位在读取时返回一个未定义的值。<br>0b - 逻辑电平0被驱动到引脚上，只要引脚被配置为通用输出。1b - 逻辑电平1在引脚上被驱动，前提是引脚被配置为通用输出。 |

### 11.3.1.3 端口设置输出寄存器（PSOR）

#### 11.3.1.3.1 偏移

| 注册 | 偏移 |
| --- | --- |
| 淘宝网 | 4h |

#### 11.3.1.3.2 职能

该寄存器配置是否设置PDOR的字段。

#### 11.3.1.3.3 图示

| 咬牙切齿 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| R | 0 | | | | | | | | | | | | | | | |
| W | PTSO | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 咬牙切齿 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| R | 0 | | | | | | | | | | | | | | | |
| W | PTSO | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

#### 11.3.1.3.4 领域

| 场地 | 职能 |
| --- | --- |

| | |
|---|---|
| 31-0<br>PTSO | 端口设置输出<br>对该寄存器的写入更新了PDOR中相应位的内容，如下所示。0b - PDORn中的对应位没有变化<br>。<br>1b - PDORn中的相应位被设置为逻辑1。 |

## 11.3.1.4 端口清除输出寄存器（PCOR）

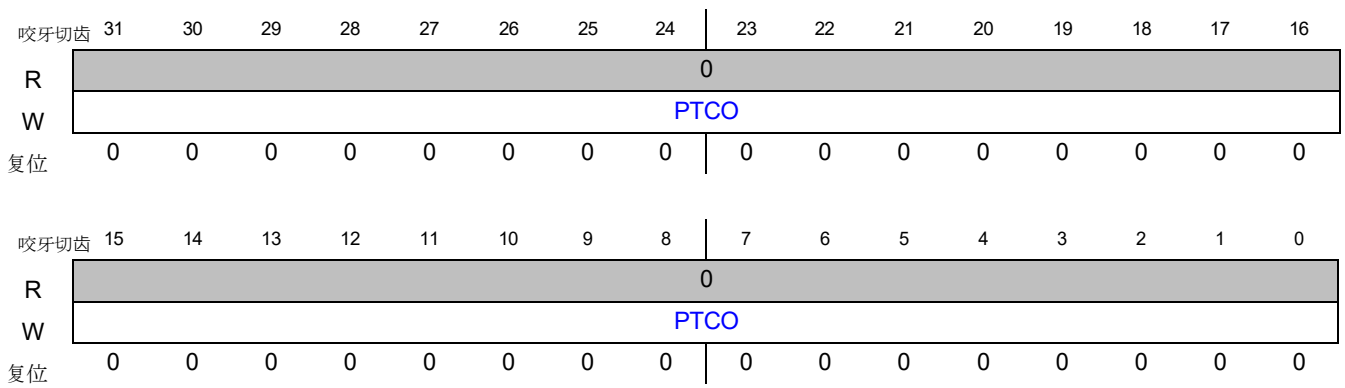
### 11.3.1.4.1 偏移

| 注册 | 偏移 |
|---|---|
| PCOR | 8h |

### 11.3.1.4.2 职能

该寄存器配置是否清除PDOR的字段。

### 11.3.1.4.3 图示




### 11.3.1.4.4 领域

| 场地 | 职能 |
|---|---|
| 31-0<br>PTCO | 端口清除输出<br>写入该寄存器，更新端口数据输出寄存器（PDOR）中相应位的内容，如下所示。<br>0b - PDORn中的对应位没有变化。1b - PDORn中的<br>对应位被清除为逻辑0。 |

## 11.3.1.5 端口切换输出寄存器（PTOR）。

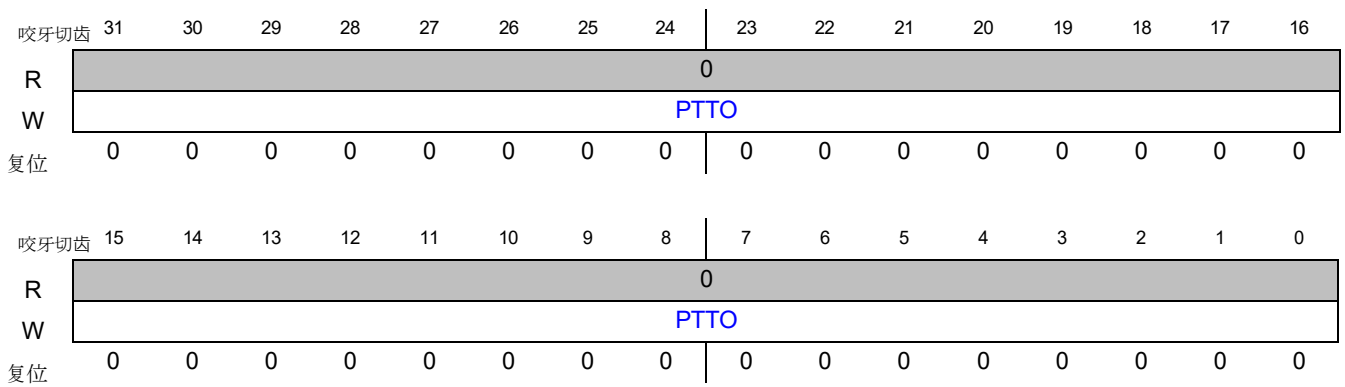
#### 11.3.1.5.1 偏移

| 注册 | 偏移 |
|---|---|
| 淘宝网 | 丘吉尔 |

#### 11.3.1.5.2 职能

这个寄存器可以切换每个通用输出引脚上驱动的逻辑电平。

#### 11.3.1.5.3 图示




#### 11.3.1.5.4 领域

| 场地 | 职能 |
|---|---|
| 31-0<br>PTTO | 端口切换输出<br>对该寄存器的写入更新了PDOR中相应位的内容，如下所示。0b - PDORn中的对应位没有变化<br>。<br>1b - PDORn中的相应位被设置为其现有逻辑状态的反值。 |

### 11.3.1.6 端口数据输入寄存器（PDIR）

#### 11.3.1.6.1 偏移

| 注册 | 偏移 |
|---|---|
| PDIR | 10h |

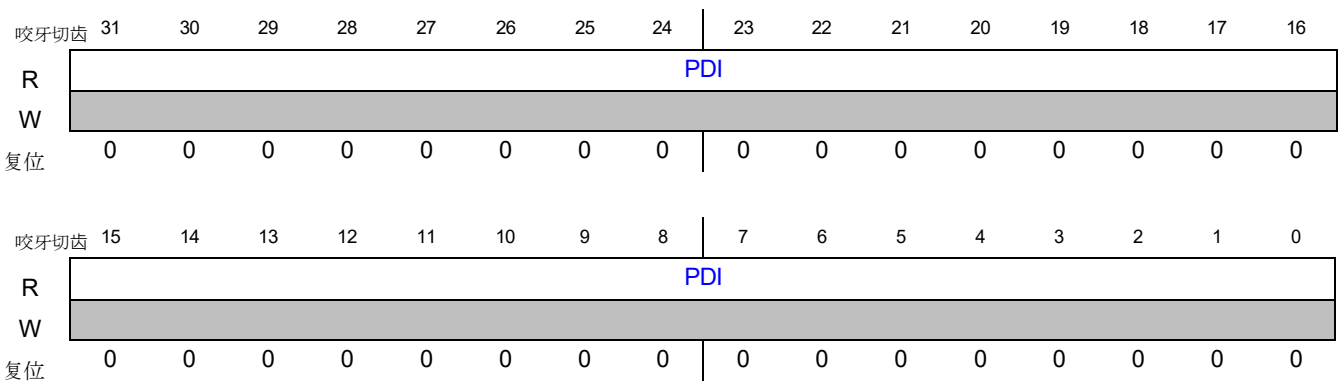
#### 11.3.1.6.2 职能

这个寄存器捕捉到了被驱动到每个通用输入引脚的逻辑电平。

**注意事项**

不要修改与你所选封装中没有的引脚相关的引脚配置寄存器。所有在你的封装中不可用的未绑定的引脚将默认为DISABLE状态，以达到最低的功耗。

#### 11.3.1.6.3 图示




#### 11.3.1.6.4 领域

| 场地 | 职能 |
|---|---|
| 31-0<br>PDI | 端口数据输入<br>在一个特定设备的未实现的引脚上读0。如果端口控制和中断模块被禁用，那么PDIR中的相应位就不会更新。<br>0b - 引脚逻辑电平为逻辑0，或未被配置为数字功能使用。1b - 引脚逻辑电平为逻辑1。 |

### 11.3.1.7 端口数据方向寄存器（PDDR）

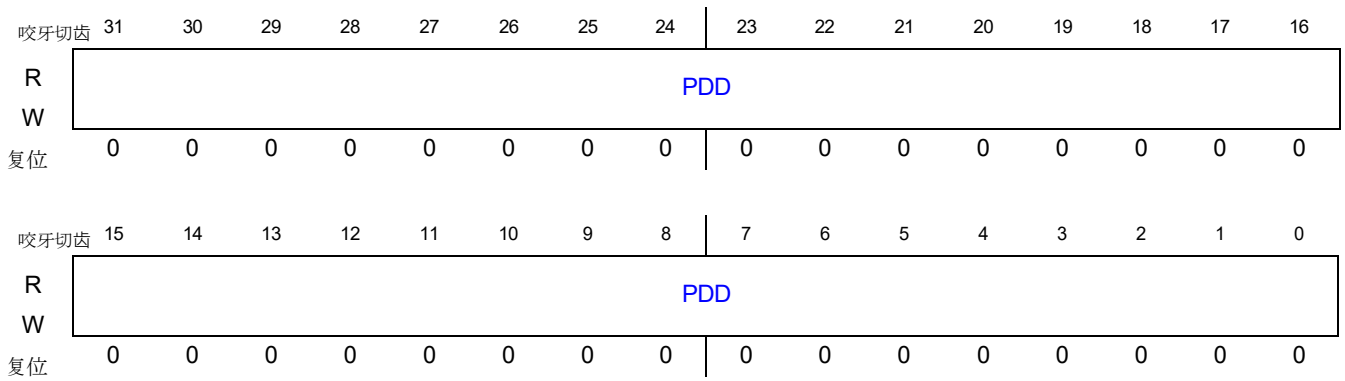
#### 11.3.1.7.1 偏移

| 注册 | 偏移 |
|---|---|
| PDDR | 14h |

#### 11.3.1.7.2 职能

PDDR将各个端口引脚配置为输入或输出。

#### 11.3.1.7.3 图示




#### 11.3.1.7.4 领域

| 场地 | 职能 |
|---|---|
| 31-0<br>PDD | 端口数据方向<br>配置单个端口引脚的输入或输出。<br>0b - 引脚被配置为通用输入，用于GPIO功能。如果在GPIOx_PIDR寄存器中禁用端口输入，该引脚将为高Z。<br>1b - 引脚被配置为通用输出，用于GPIO功能。 |

### 11.3.1.8 端口输入禁用寄存器（PIDR）

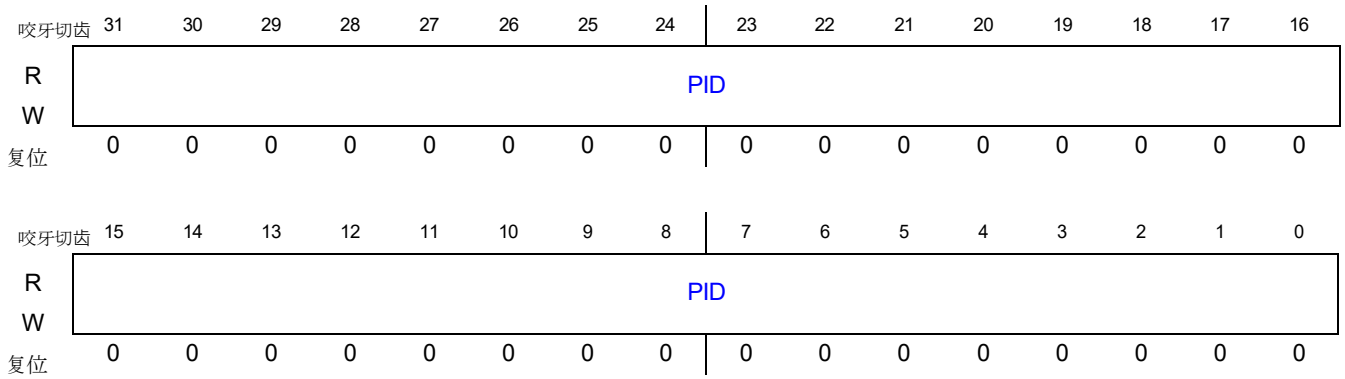
#### 11.3.1.8.1 偏移

| 注册 | 偏移 |
|---|---|
| PIDR | 18h |

#### 11.3.1.8.2 职能

这个寄存器使每个通用引脚不能作为输入。

#### 11.3.1.8.3 图示




#### 11.3.1.8.4 领域

| 场地 | 职能 |
|---|---|
| 31-0<br>PID | 端口输入禁用<br>0b - 引脚被配置为通用输入，只要引脚被配置为任何数字功能。<br>1b - 引脚未被配置为通用输入。相应的端口数据输入寄存器位将读作零。 |

## 11.4 功能 描述

### 11.4.1 通用的 输入

每个引脚的逻辑状态都可以通过端口数据输入寄存器获得，前提是端口输入使能寄存器中的相应位被设置，引脚被配置为数字功能，并且相应的端口控制和中断模块被启用。

输入引脚同步器与端口控制和中断模块共享，因此，如果相应的端口控制和中断模块被禁用，那么同步器也被禁用。当一个端口不需要通用输入功能时，这可以减少功耗。

### 11.4.2 通用的 输出

每个引脚的逻辑状态可以通过端口数据输出寄存器和端口数据方向寄存器来控制，前提是该引脚被配置为GPIO功能。下表描述了一个引脚被配置为输入/输出的条件。

| 如果 | 那么 |
| --- | --- |
| 一个引脚被配置为GPIO功能，相应的端口数据方向寄存器位被清除。 | 该引脚被配置为一个输入。 |
| 一个引脚被配置为GPIO功能，相应的端口数据方向寄存器位被设置。 | 该引脚被配置为输出，并且引脚的逻辑状态等于对应的端口数据输出寄存器。 |

为了便于对通用输出进行有效的位操作，存在引脚数据设置、引脚数据清除和引脚数据切换寄存器，以允许在一个端口内的一个或多个输出被设置、清除或从一个寄存器写入切换。

相应的端口控制和中断模块不需要启用来更新端口数据方向寄存器和端口数据输出寄存器的状态，包括设置/清除/切换寄存器。

# 章 12
# 轻型横梁开关 (AXBS- Lite)

## 12.1 特定芯片的AXBS-Lite 信息

### 12.1.1 横梁式交换机的主站 分配

下表列出了连接到横梁式交换机的主站以及它们在固定优先级模式下的优先级顺序。

**表12-1.固定优先级仲裁的主端口分配和优先级顺序**

| 薯片 | 主端口号0 | 主端口号1 | 主端口号2 | 主端口号3 |
|---|---|---|---|---|
| | 重点3 [1] | 第21项优先事项 | 第11项优先事项 | 优先级41 |
| S32K148 | ARM核心代码总线 | ARM核心系统总线 | DMA | ENET |
| S32K146 | ARM核心代码总线 | ARM核心系统总线 | DMA | - |
| S32K144 | ARM核心代码总线 | ARM核心系统总线 | DMA | |
| S32K142 | ARM核心代码总线 | ARM核心系统总线 | DMA | |
| S32K116 | ARM核心主总线 | - | DMA | |
| S32K118 | ARM核心主总线 | - | DMA | |

1.固定优先级模式下的优先级。MCM控制全局从属端口仲裁的模式选择。对于固定优先级，将MCM_CPCR[CBRR]设置为0；对于轮流仲裁，将MCM_CPCR[CBRR]设置为1；仲裁设置适用于所有从属端口。

### 12.1.2 横梁式开关的从属 分配。

下表列出了连接到横梁式交换机的从机以及系统MPU是否对其进行保护。

**表12-2.从属端口的分配和系统MPU的保护**

| 薯片 | 从属端口号0 | 从属端口号1 | 从属端口号2 | 从属端口号3 |
|---|---|---|---|---|
| S32K148 | 闪存控制器1 | SRAM控制器[1] | 外围桥0/GPIO [2] | QuadSPI [1] |

表继续在下一页...

职能描述

表12-2.从属端口的分配和系统MPU保护（续）

| 薯片 | 从属端口号0 | 从属端口号1 | 从属端口号2 | 从属端口号3 |
|---|---|---|---|---|
| S32K146 | 闪存控制器1 | SRAM控制器[1] | 外围桥0/GPIO [2] | - |
| S32K144 | 闪存控制器[1] | SRAM控制器[1] | 外围桥0/GPIO [2] | |
| S32K142 | 闪存控制器[1] | SRAM控制器[1] | 外围桥0/GPIO [2] | |
| S32K116 | 闪存控制器[1] | SRAM控制器[1] | 外围桥0/GPIO [2] | |
| S32K118 | 闪存控制器[1] | SRAM控制器[1] | 外围桥0/GPIO | |

1. 受系统MPU的保护
2. 不受系统MPU的保护。系统MPU不保护对外围寄存器的访问，包括系统MPU自己的寄存器。外围桥（AIPS-Lite）中内置了保护功能。

## 12.2 简介

这里的信息提供了有关横梁开关的布局、配置和编程的信息。

十字架开关使用十字架开关结构连接总线主站和总线从站。这种结构允许多达四个总线主站同时访问不同的总线从站，同时在总线主站访问同一从站时提供仲裁。

### 12.2.1 特点

横杆开关包括这些功能。

- 对称的横梁式总线交换机的实现
  - 允许从不同的主站对不同的从站进行并发访问
- 高达单时钟32位传输
- 可编程配置，用于固定优先级或轮流从属端口仲裁（见芯片特定信息）。

## 12.3 功能 说明

关于一般操作和仲裁的信息可以在这里找到。

### 12.3.1 一般 操作

当主站访问横梁交换机时，会立即进行访问。如果访问的目标从属端口可用，那么访问会立即在从属端口上呈现。单时钟或零等待状态的访问可以通过横梁进行。如果访问的目标从属端口繁忙或停在不同的主端口上，请求的主站只是看到插入的等待状态，直到目标从属端口可以为主站的请求服务。服务请求的延迟取决于每个主站的优先级和响应的从站的访问时间。

由于横梁开关看起来只是主设备的另一个从属设备，主设备不知道它是否真的拥有它所瞄准的从属端口。虽然主设备对它所瞄准的从属端口没有控制权，但它只是在等待。

在主站控制了它所瞄准的从属端口后，主站仍然控制着从属端口，直到它通过运行一个IDLE周期或为下一次访问瞄准一个不同的从属端口而放弃该从属端口。

如果另一个优先级更高的主站向从属端口发出请求，主站也可能失去对从属端口的控制。

横梁终止所有的主站IDLE传输，而不是让终止来自某个从属总线。此外，当没有主站要求访问从站端口时，尽管默认的主站可能被授予对从站端口的访问权，但横梁还是将IDLE传输驱动到从站总线上。

当从属总线被横梁闲置时，它仍然停在最后一个使用从属端口的主站上。这样做是为了节省最初的仲裁延迟时钟，否则，如果同一个主站必须进行仲裁以获得对从属端口的控制，就会出现这种延迟。

### 12.3.2 仲裁

横梁式交换机支持两种仲裁算法。

- 固定优先权
- 循环赛

全局从属端口仲裁算法的选择在横梁开关芯片特定信息中有所描述。

#### 12.3.2.1 在未定义长度的突发事件中进行仲裁

未定义长度的突发事件可以被打断。

#### 12.3.2.2 固定优先级的操作

在固定优先级模式下工作时，每个主站都被分配了一个独特的优先级，编号最高的主站拥有最高的优先级（例如，在一个有5个主站的系统中，主站1比主站3的优先级低）。如果两个主站请求访问同一个从属端口，具有最高优先权的主站获得对从属端口的控制。

**注意事项**

在这种仲裁模式下，优先级较高的主站可以垄断一个从站端口，阻止任何优先级较低的主站对该端口的访问。

当一个主站向一个从站端口发出请求时，从站端口会检查新请求的主站的优先级是否高于目前对从站端口有控制权的主站，除非从站端口处于停顿状态。从属端口在每个时钟边沿执行仲裁检查，以确保适当的主站（如果有的话）对从属端口拥有控制权。

下表描述了基于请求的主端口的可能情况。

**表12-3.横梁式交换机如何将从属端口的控制权授予主站**

| 当 | 然后，横梁开关将控制权授予提出请求的主站。 |
|---|---|
| 以下两点都是真的。<br>• 当前的主站没有在运行转移。<br>• 新的请求主站的优先级比当前主站的优先级高。 | 在下一个时钟沿 |
| 以下两点都是真的。<br>• 当前主站正在运行一个未定义长度的突发传输。<br>• 请求的主站的优先级高于当前主站的优先级。 | 在未定义长度突发传输的下一个仲裁点上 |
| 请求的主站的优先级低于当前主站。 | 在以下一个周期结束时。<br>• 一个IDLE循环<br>• 一个非IDLE循环到当前从属端口以外的位置 |

#### 12.3.2.3 轮流优先操作

在轮流模式下操作时，每个主站被分配一个基于主站端口号的相对优先级。这个相对优先权与最后一个在从属总线上进行传输的主站端口号（ID）进行比较。最高优先级的请求主站在下一个传输边界成为从属总线的所有者。优先权是基于请求主站的ID比上一个主站的ID领先多少。

在被授予对从属端口的访问权后，主站可以根据需要对该端口进行多次传输，直到另一个主站对同一从属端口发出请求。下一个排队的主站在下一个传输边界被授予对从属端口的访问权，如果当前主站没有未决的访问请求，也可能在下一个时钟周期被授予访问权。

作为轮流模式下仲裁的一个例子，假设横梁上实现的是主控端口0、1、4和5。如果从属端口的最后一个主站是主站1，而主站0、4和5同时发出请求，它们会按顺序得到服务。4然后5然后0。

循环仲裁模式通常能更公平地分配可用的从属端口带宽（与固定优先权相比），因为固定的主站优先权不影响主站选择。

## 12.4 初始化/应用 信息

横梁开关不需要初始化。

# 章 13
# 内存保护单元 (MPU)

## 13.1 特定芯片的MPU 信息

在该器件上，恩智浦的系统MPU实现了安全机制，以防止主控器访问受限的内存区域。该系统MPU在Crossbar开关的层面上提供内存保护。每个Crossbar主控器（Core、DMA、Ethernet）都可以对每个受保护的内存区域分配不同的访问权。该系列中的ARM
M4内核版本没有集成ARM内核MPU，它将同时监测只有内核发起的内存访问。在本文件中，术语MPU指的是恩智浦的系统MPU。

### 13.1.1 MPU从属端口 分配。

由MPU保护的内存映射资源是：。

**表13-1.S32K14x系列的MPU从属端口分配情况**

| 来源 | MPU从属端口的分配 | 目的地 |
|---|---|---|
| 交叉条从属端口0 | MPU从属端口0 | 闪存控制器 |
| 横梁式从属端口1 | MPU从属端口1 | SRAM后门 |
| 代码总线 | MPU从属端口2 | SRAM_L前门 |
| 系统总线 | MPU从属端口3 | SRAM_U前门 |
| 横梁式从属端口3 | MPU从属端口4 | QuadSPI |

**表13-2.S32K11x系列的MPU从属端口分配情况**

| 来源 | MPU从属端口的分配 | 目的地 |
|---|---|---|
| 交叉条从属端口0 | MPU从属端口0 | 闪存控制器 |
| 横梁式从属端口1 | MPU从属端口1 | SRAM控制器/MTB/DWT/MCM |

### 13.1.2 MPU逻辑总线主站 分配。

MPU的逻辑总线主站分配为：。

**表13-3.MPU逻辑总线主站的分配**

| MPU逻辑总线主站ID | 总线主站 |
|---|---|
| 0 | 核心 |
| 1 | 调试器 |
| 2 | DMA |
| 3 | ENET |

### 13.1.3 当前 PID

当前的PID被配置在杂项控制模块过程ID寄存器（MCM_PID）。

### 13.1.4 区域描述符和从属端口 配置

对于产品系列中的每个芯片，下表显示了区域描述符和从属端口的数量，以及反映这些数字的控制/错误状态寄存器（CESR）的复位值。

**表13-4.MPU的配置**

| 薯片 | 区域描述符 | 从属端口 | CESR复位值 |
|---|---|---|---|
| S32K116 | 8 | 2 | 0081_2001 |
| S32K118 | 8 | 2 | 0081_2001 |
| S32K142 | 8 | 4 | 0081_4001h |
| S32K144 | 8 | 4 | 0081_4001h |
| S32K146 | 8 | 4 | 0081_4001h |
| S32K148 | 16 | 5 | 0081_5201h |

## 13.2 简介

内存保护单元（MPU）为设备中产生的所有内存引用提供硬件访问控制。

## 13.3 概述

MPU同时监控所有的系统总线事务，并使用预先编程的区域描述符来评估它们是否合适，这些描述符定义了内存空间和它们的访问权限。具有足够访问控制权限的内存引用被允许完成，而没有映射到任何区域描述符或权限不足的引用则以保护错误响应终止。

### 13.3.1 块 图

下图是MPU模块的简化框图。

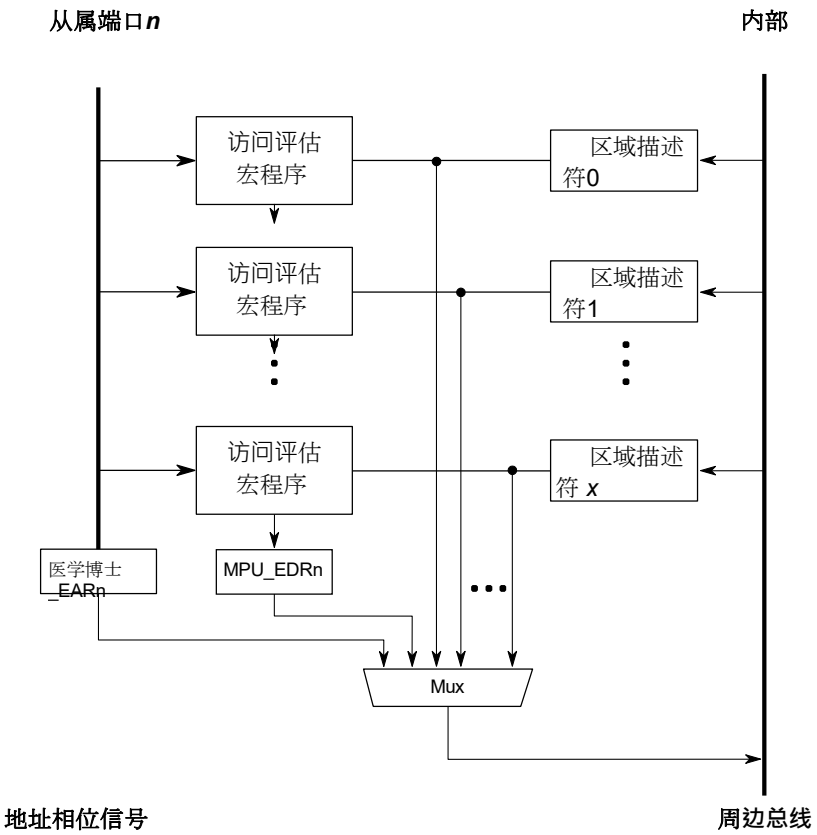



**图13-1.MPU框图**

硬件的二维连接矩阵清晰可见，基本访问评估宏显示为复制的子模块块。十字架开关的从属端口显示在左边，区域描述符寄存器在中间，外围总线接口在右边。评估宏包含两个幅度比较器，连接到每个区域描述器的起始和结束地址寄存器，以及组合逻辑块，以确定区域命中和访问保护错误。有关访问评估宏的细节，请

参见访问评估宏。

### 13.3.2 特点

MPU实现了一个内存区域描述符的二维硬件阵列和交叉条从属端口，以连续监测系统中每个总线主站产生的每个内存引用的合法性。

该功能集包括。

- 16个程序可见的128位区域描述符，每个描述符可由四个32位字访问
  - 每个区域描述符都定义了一个模数为32的字节空间，在内存的任何位置对齐
    - 区域大小可以从32字节到4G字节不等
  - 在一个描述符字中定义了两个访问控制权限
    - Master 0-3：主管和用户访问的读、写和执行属性
    - 大师4-7：读和写属性
  - 描述符有效位的硬件辅助维护将一致性问题降到最低。
  - 访问控制权限词的替代编程模型视图
  - 对于重叠的区域描述符，给予许可的优先权大于拒绝访问的优先权
- 如果一个内存引用在任何内存区域都没有命中，或者该引用在所有命中的内存区域都是非法的，则检测访问保护错误。如果发生了访问错误，引用就会被错误响应所终止，MPU就会抑制发送到目标从属设备的总线周期。
- 错误寄存器，每个从属端口，捕获最后的故障地址、属性和其他信息。
- 全局MPU启用/禁用控制位

## 13.4 MPU寄存器 描述

该编程模型被划分为三组。

- 控制/状态寄存器
- 包含区域描述符的数据结构
- 区域描述符访问控制值的另一种观点

编程模型只能用32位访问来引用。编程模型只能在监督员模式下访问。

以下类型的尝试性引用会产生一个错误终止。

- 非32位引用
- 在用户模式下的访问
- 对未定义的--也就是保留的--地址的引用
- 具有不支持的访问类型的引用，例如对只读寄存器的写访问或对只写寄存器的读访问

## 13.4.1 MPU内存 地图

MPU基础地址。4000_D000h

| 偏移 | 注册 | 宽度 (单位：比特) | 访问 | 重置值 |
|---|---|---|---|---|
| 0h | 控制/错误状态寄存器（CESR）。 | 32 | RW | 0081_5201h |
| 10h | 错误地址寄存器，从属端口0（EAR0）。 | 32 | RO | 0000_0000h |
| 14h | 错误详细信息寄存器，从属端口0（EDR0）。 | 32 | RO | 0000_0000h |
| 18h | 错误地址寄存器，从属端口1（EAR1）。 | 32 | RO | 0000_0000h |
| 1Ch | 错误详细信息寄存器，从属端口1（EDR1）。 | 32 | RO | 0000_0000h |
| 20h | 错误地址寄存器，从属端口2（EAR2）。 | 32 | RO | 0000_0000h |
| 24h | 错误详细信息寄存器，从属端口2（EDR2）。 | 32 | RO | 0000_0000h |
| 28h | 错误地址寄存器，从属端口3（EAR3）。 | 32 | RO | 0000_0000h |
| 2Ch | 错误详细信息寄存器，从属端口3（EDR3）。 | 32 | RO | 0000_0000h |
| 30h | 错误地址寄存器，从属端口4（EAR4）。 | 32 | RO | 0000_0000h |
| 34h | 错误详细信息寄存器，从属端口4（EDR4）。 | 32 | RO | 0000_0000h |
| 400h | 区域描述符0，字0（RGD0_WORD0）。 | 32 | RW | 0000_0000h |
| 404h | 区域描述符0，字1（RGD0_WORD1）。 | 32 | RW | FFFF_FFFFh |
| 408h | 区域描述符0，字2（RGD0_WORD2）。 | 32 | RW | 0061_F7DFh |
| 40Ch | 区域描述符0，字3（RGD0_WORD3）。 | 32 | RW | 0000_0001h |
| 410h | 区域描述符1，字0（RGD1_WORD0）。 | 32 | RW | 0000_0000h |
| 414h | 区域描述符1，字1（RGD1_WORD1）。 | 32 | RW | 0000_001Fh |
| 418h | 区域描述符1，字2（RGD1_WORD2）。 | 32 | RW | 0000_0000h |
| 41Ch | 区域描述符1，字3（RGD1_WORD3）。 | 32 | RW | 0000_0000h |

| | | | | |
|---|---|---|---|---|
| 420h | 区域描述符2，字0（RGD2_WORD0）。 | 32 | RW | 0000_0000h |
| 424h | 区域描述符2，字1（RGD2_WORD1）。 | 32 | RW | 0000_001Fh |

*表继续在下一页...*

| 偏移 | 注册 | 宽度<br>(单位：比特) | 访问 | 重置值 |
|---|---|---|---|---|
| 428h | 区域描述符2，字2（RGD2_WORD2）。 | 32 | RW | 0000_0000h |
| 42Ch | 区域描述符2，字3（RGD2_WORD3）。 | 32 | RW | 0000_0000h |
| 430h | 区域描述符3，字0（RGD3_WORD0）。 | 32 | RW | 0000_0000h |
| 434h | 区域描述符3，字1（RGD3_WORD1）。 | 32 | RW | 0000_001Fh |
| 438h | 区域描述符3，字2（RGD3_WORD2）。 | 32 | RW | 0000_0000h |
| 43Ch | 区域描述符3，字3（RGD3_WORD3）。 | 32 | RW | 0000_0000h |
| 440h | 区域描述符4，字0（RGD4_WORD0）。 | 32 | RW | 0000_0000h |
| 444h | 区域描述符4，字1（RGD4_WORD1）。 | 32 | RW | 0000_001Fh |
| 448h | 区域描述符4，字2（RGD4_WORD2）。 | 32 | RW | 0000_0000h |
| 44Ch | 区域描述符4，字3（RGD4_WORD3）。 | 32 | RW | 0000_0000h |
| 450h | 区域描述符5，字0（RGD5_WORD0）。 | 32 | RW | 0000_0000h |
| 454h | 区域描述符5，字1（RGD5_WORD1）。 | 32 | RW | 0000_001Fh |
| 458h | 区域描述符5，字2（RGD5_WORD2）。 | 32 | RW | 0000_0000h |
| 45Ch | 区域描述符5，字3（RGD5_WORD3）。 | 32 | RW | 0000_0000h |
| 460h | 区域描述符6，字0（RGD6_WORD0）。 | 32 | RW | 0000_0000h |
| 464h | 区域描述符6，字1（RGD6_WORD1）。 | 32 | RW | 0000_001Fh |
| 468h | 区域描述符6，字2（RGD6_WORD2）。 | 32 | RW | 0000_0000h |
| 46Ch | 区域描述符6，字3（RGD6_WORD3）。 | 32 | RW | 0000_0000h |
| 470h | 区域描述符7，字0（RGD7_WORD0）。 | 32 | RW | 0000_0000h |
| 474h | 区域描述符7，字1（RGD7_WORD1）。 | 32 | RW | 0000_001Fh |
| 478h | 区域描述符7，字2（RGD7_WORD2）。 | 32 | RW | 0000_0000h |
| 47Ch | 区域描述符7，字3（RGD7_WORD3）。 | 32 | RW | 0000_0000h |
| 480h | 区域描述符8，字0（RGD8_WORD0）。 | 32 | RW | 0000_0000h |
| 484h | 区域描述符8，字1（RGD8_WORD1）。 | 32 | RW | 0000_001Fh |
| 488h | 区域描述符8，字2（RGD8_WORD2）。 | 32 | RW | 0000_0000h |
| 48Ch | 区域描述符8，字3（RGD8_WORD3）。 | 32 | RW | 0000_0000h |
| 490h | 区域描述符9，字0（RGD9_WORD0）。 | 32 | RW | 0000_0000h |
| 494h | 区域描述符9，字1（RGD9_WORD1）。 | 32 | RW | 0000_001Fh |
| 498h | 区域描述符9，字2（RGD9_WORD2）。 | 32 | RW | 0000_0000h |
| 49Ch | 区域描述符9，字3（RGD9_WORD3）。 | 32 | RW | 0000_0000h |
| 4A0h | 区域描述符10，字0（RGD10_WORD0）。 | 32 | RW | 0000_0000h |
| 4A4h | 区域描述符10，字1（RGD10_WORD1）。 | 32 | RW | 0000_001Fh |
| 4A8h | 区域描述符10，字2（RGD10_WORD2）。 | 32 | RW | 0000_0000h |
| 4ACh | 区域描述符10，字3（RGD10_WORD3）。 | 32 | RW | 0000_0000h |
| 4B0h | 区域描述符11，字0（RGD11_WORD0）。 | 32 | RW | 0000_0000h |
| 4B4h | 区域描述符11，字1（RGD11_WORD1）。 | 32 | RW | 0000_001Fh |
| 4B8h | 区域描述符11，字2（RGD11_WORD2）。 | 32 | RW | 0000_0000h |
| 4BCh | 区域描述符11，字3（RGD11_WORD3）。 | 32 | RW | 0000_0000h |
| 4C0h | 区域描述符12，字0（RGD12_WORD0）。 | 32 | RW | 0000_0000h |
| 4C4h | 区域描述符12，字1（RGD12_WORD1）。 | 32 | RW | 0000_001Fh |

*表继续在下一页...*

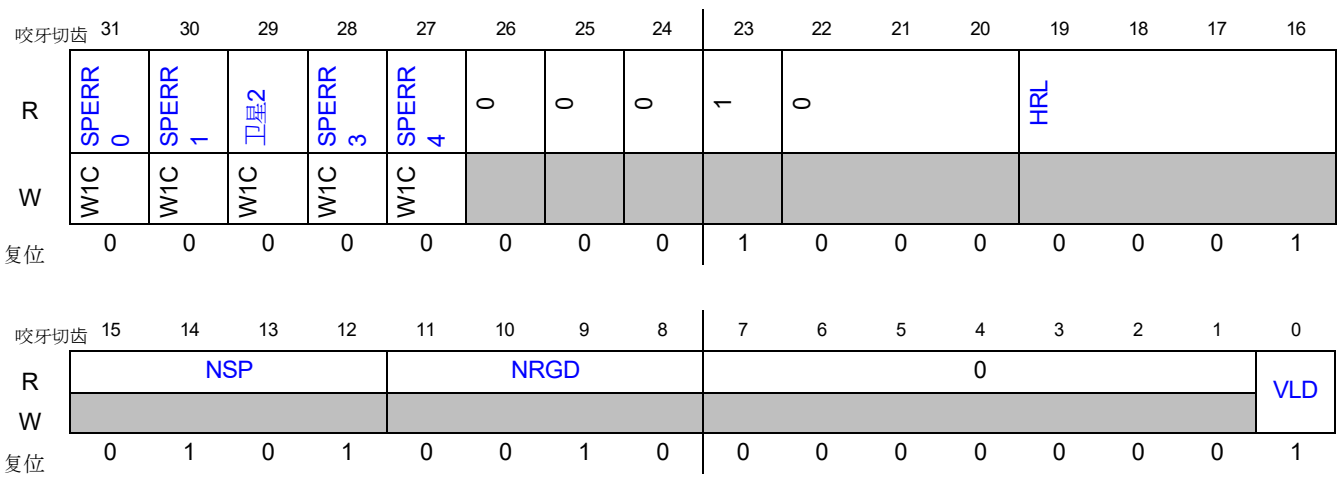
| 偏移 | 注册 | 宽度<br>(单位：比特) | 访问 | 重置值 |
|---|---|---|---|---|
| 4C8h | 区域描述符12，字2（RGD12_WORD2）。 | 32 | RW | 0000_0000h |
| 4CCh | 区域描述符12，字3（RGD12_WORD3）。 | 32 | RW | 0000_0000h |
| 4D0h | 区域描述符13，字0（RGD13_WORD0）。 | 32 | RW | 0000_0000h |
| 4D4h | 区域描述符13，字1（RGD13_WORD1）。 | 32 | RW | 0000_001Fh |
| 4D8h | 区域描述符13，字2（RGD13_WORD2）。 | 32 | RW | 0000_0000h |
| 4DCh | 区域描述符13，字3（RGD13_WORD3）。 | 32 | RW | 0000_0000h |
| 4E0h | 区域描述符14，字0（RGD14_WORD0）。 | 32 | RW | 0000_0000h |
| 4E4h | 区域描述符14，字1（RGD14_WORD1）。 | 32 | RW | 0000_001Fh |
| 4E8h | 区域描述符14，字2（RGD14_WORD2）。 | 32 | RW | 0000_0000h |
| 4ECh | 区域描述符14，字3（RGD14_WORD3）。 | 32 | RW | 0000_0000h |
| 4F0h | 区域描述符15，字0（RGD15_WORD0）。 | 32 | RW | 0000_0000h |
| 4F4h | 区域描述符15，字1（RGD15_WORD1）。 | 32 | RW | 0000_001Fh |
| 4F8h | 区域描述符15，字2（RGD15_WORD2）。 | 32 | RW | 0000_0000h |
| 4FCh | 区域描述符15，字3（RGD15_WORD3）。 | 32 | RW | 0000_0000h |
| 800h | 区域描述符备用访问控制0（RGDAAC0）。 | 32 | RW | 0061_F7DFh |
| 804h | 区域描述符替代访问控制1（RGDAAC1）。 | 32 | RW | 0000_0000h |
| 808h | 区域描述符替代访问控制2（RGDAAC2）。 | 32 | RW | 0000_0000h |
| 80Ch | 区域描述符替代访问控制3（RGDAAC3）。 | 32 | RW | 0000_0000h |
| 810h | 区域描述符替代访问控制4（RGDAAC4）。 | 32 | RW | 0000_0000h |
| 814h | 区域描述符替代访问控制5（RGDAAC5）。 | 32 | RW | 0000_0000h |
| 818h | 区域描述符替代访问控制6（RGDAAC6）。 | 32 | RW | 0000_0000h |
| 81Ch | 区域描述符替代访问控制7（RGDAAC7）。 | 32 | RW | 0000_0000h |
| 820h | 区域描述符替代访问控制8（RGDAAC8）。 | 32 | RW | 0000_0000h |
| 824h | 区域描述符替代访问控制9（RGDAAC9）。 | 32 | RW | 0000_0000h |
| 828h | 区域描述符替代访问控制10（RGDAAC10）。 | 32 | RW | 0000_0000h |
| 82Ch | 区域描述符替代访问控制11（RGDAAC11）。 | 32 | RW | 0000_0000h |
| 830h | 区域描述符替代访问控制12（RGDAAC12）。 | 32 | RW | 0000_0000h |
| 834h | 区域描述符替代访问控制13（RGDAAC13）。 | 32 | RW | 0000_0000h |
| 838h | 区域描述符替代访问控制14（RGDAAC14）。 | 32 | RW | 0000_0000h |
| 83Ch | 区域描述符替代访问控制15（RGDAAC15）。 | 32 | RW | 0000_0000h |

## 13.4.2 控制/错误状态寄存器 （CESR）。

### 13.4.2.1 偏移

| 注册 | 偏移 |
|---|---|
| CESR | 0h |

### 13.4.2.2 图示




### 13.4.2.3 领域

| 场地 | 职能 |
|---|---|
| 31<br>ĀĀĀ | 从属端口0错误<br>表示EAR0和EDR0中捕获的错误。当硬件检测到一个错误并记录故障地址和属性时，该位被设置。通过向其写一来清除它。如果在写的同一周期内捕获了另一个错误，该标志仍被设置。一个find-first-one指令或类似的指令可以检测到一个被捕获的错误的存在。<br>0b - 从属端口0没有发生错误。 1b - 从属端口0发生了错误。 |
| 30<br>SPERR1 | 从属端口1错误<br>表示EAR1和EDR1中捕获的错误。当硬件检测到一个错误并记录故障地址和属性时，该位被设置。通过向其写一来清除它。如果在写的同一周期内捕获了另一个错误，该标志仍被设置。一个find-first-one指令或类似的指令可以检测到一个被捕获的错误的存在。<br>0b - 从属端口1没有发生错误。1b - 从属端口1发生了错误。 |

| | |
|---|---|
| 29<br>SPERR2 | 从属端口2错误 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| | 表示EAR2和EDR2中捕获的错误。当硬件检测到一个错误并记录故障地址和属性时，该位被设置。通过向其写一来清除它。如果在写的同一周期内捕获了另一个错误，该标志仍被设置。一个find-first-one指令或类似的指令可以检测到一个被捕获的错误的存在。<br>0b - 从属端口2没有发生错误。1b - 从属端口2发生了错误。 |
| 28<br>SPERR3 | 从属端口3错误<br>表示EAR3和EDR3中捕获的错误。当硬件检测到一个错误并记录故障地址和属性时，该位被设置。通过向其写一来清除它。如果在写的同一周期内捕获了另一个错误，该标志仍被设置。一个find-first-one指令或类似的指令可以检测到一个被捕获的错误的存在。<br>0b - 从属端口3没有发生错误。1b - 从属端口3发生了错误。 |
| 27<br>SPERR4 | 从属端口4错误<br>表示EAR4和EDR4中捕获的错误。当硬件检测到一个错误并记录故障地址和属性时，该位被设置。通过向其写一来清除它。如果在写的同一周期内捕获了另一个错误，该标志仍被设置。一个find-first-one指令或类似的指令可以检测到一个被捕获的错误的存在。<br>0b - 从属端口4没有发生错误。1b - 从属端口4发生了错误。 |
| 26<br>- | 保留 |
| 25<br>- | 保留 |
| 24<br>- | 保留 |
| 23<br>- | 保留 |
| 22-20<br>- | 保留 |
| 19-16<br>HRL | 硬件版本级别<br>指定MPU的硬件和定义修订级别。它可以被软件读取以确定模块的功能定义。 |
| 15-12<br>NSP | 从属端口的数量<br>指定连接到MPU的从属端口的数量。 |
| 11-8<br>NRGD | 区域描述符的数量<br>表示MPU中实现的区域描述符的数量。<br>0000b - 8个区域描述符 0001b - 12个区域描述符 0010b - 16个区域描述符 |
| 7-1<br>- | 保留 |

| | |
|---|---|
| 0<br>VLD | 有效<br>MPU的全局启用/禁用。<br>0b - MPU被禁用。所有总线主站的所有访问都被允许。 |

| 场地 | 职能 |
|---|---|
| | 1b - MPU被启用 |

## 13.4.3 错误地址寄存器，从属端口n (EAR0 - EAR4)

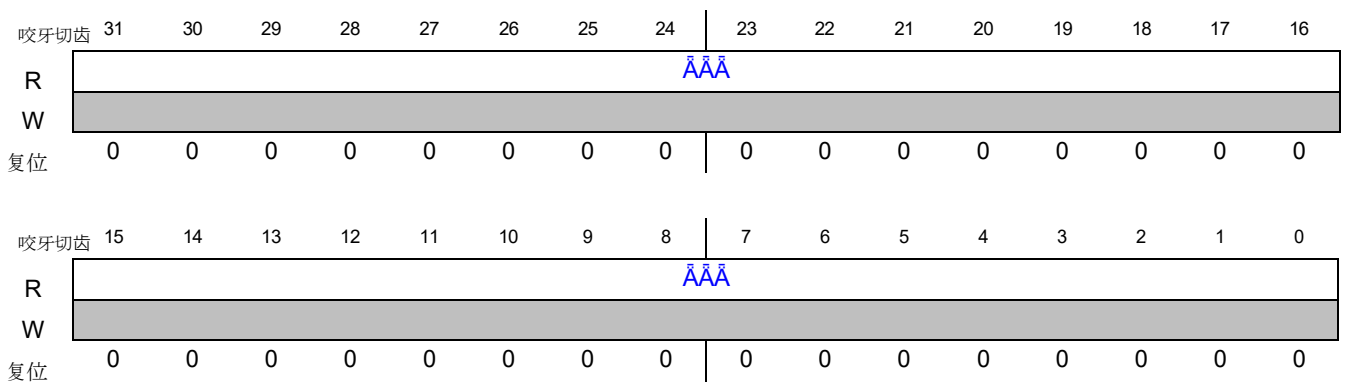
### 13.4.3.1 偏移

| 注册 | 偏移 |
|---|---|
| EAR0 | 10h |
| EAR1 | 18h |
| EAR2 | 20h |
| EAR3 | 28h |
| EAR4 | 30h |

### 13.4.3.2 职能

当MPU检测到从属端口n的访问错误时，32位参考地址被捕获到这个只读寄存器中，CESR[SPERRn]中的相应位被设置。同时在相应的EDRn中获取有关故障访问的其他信息。这个寄存器和相应的EDRn包含最近的访问错误；CESR[SPERRn]没有硬件互锁，因为错误寄存器总是在每次违反保护的情况发生时被加载。

### 13.4.3.3 图示



#### 13.4.3.4 领域

| 场地 | 职能 |
|---|---|
| 31-0<br>ÃÃÃ | 错误地址<br><br>表示产生访问错误的从属端口n的参考地址。 |

### 13.4.4 错误详细信息寄存器，从属端口n (EDR0 - EDR4)

#### 13.4.4.1 偏移

| 注册 | 偏移 |
|---|---|
| EDR0 | 14h |
| EDR1 | 1Ch |
| EDR2 | 24h |
| EDR3 | 2Ch |
| EDR4 | 34h |

#### 13.4.4.2 职能

当MPU检测到从属端口n的访问错误时，在这个只读寄存器中捕获32位错误细节，CESR[SPERRn]中的相应位被设置。同时，在相应的 EARn
寄存器中捕获故障地址的信息。这个寄存器和相应的EARn寄存器包含了最近的访问错误；由于错误寄存器总是在每次违反保护规定时被加载，所以与CESR[SPERRn]不存在硬件互锁关系。

### 13.4.4.3 图示




### 13.4.4.4 领域

| 场地 | 职能 |
|---|---|
| 31-16<br>EACD | 错误访问控制详情<br>表示有访问错误的区域描述符。<br>• 如果EDRn包含一个捕获的错误，并且EACD被清零，那么在任何区域描述符中都没有击中访问。<br>• 如果只有一个EACD位被设置，说明保护错误是由一个非重叠的区域描述符引起的。<br>• 如果两个或更多的EACD位被设置，保护错误是由一组重叠的区域描述符引起的。 |
| 15-8<br>EPID | 错误过程识别<br>记录出错参考的进程标识符。进程标识符通常只由处理器内核驱动；对于其他总线主站，这个字段被清除。 |
| 7-4<br>梦之城_梦之城娱乐_梦之城国际娱乐_梦之城国际娱乐平台 | 错误主控号码<br>表示产生访问错误的总线主站。 |
| 3-1<br>EATTR | 错误属性<br>表示关于故障参考的属性信息。<br>**注意**：所有其他编码都是保留的。000b - 用户模式，指令访问 001b - 用户模式，数据访问<br>010b - 监管者模式，指令访问 011b - 监管者模式，数据访问 |

| | |
|---|---|
| 0<br>ERW | 错误读/写<br><br>表示发生故障的参照物的访问类型。<br>0b - 读 1b - 写 |

# 13.4.5 区域描述符n，字0 (RGD0_WORD0 - RGD15_WO RD0)

## 13.4.5.1 偏移

对于n=0到15。

| 注册 | 偏移 |
|---|---|
| RGDn_WORD0 | 400h + (n × 10h) |

## 13.4.5.2 职能

区域描述符的第一个字定义了内存区域的0-modulo-32字节的起始地址。对该寄存器的写入将清除区域描述符的有效位（RGDn_WORD3[VLD]）。

## 13.4.5.3 图示




## 13.4.5.4 领域

| 场地 | 职能 |
|---|---|
| 31-5<br>呼叫中心 | 起始地址<br>定义内存区域的0-modulo-32字节起始地址的最有效位。 |
| 4-0<br>- | 保留 |

## 13.4.6 区域描述符0，字1 (RGD0_WORD1)

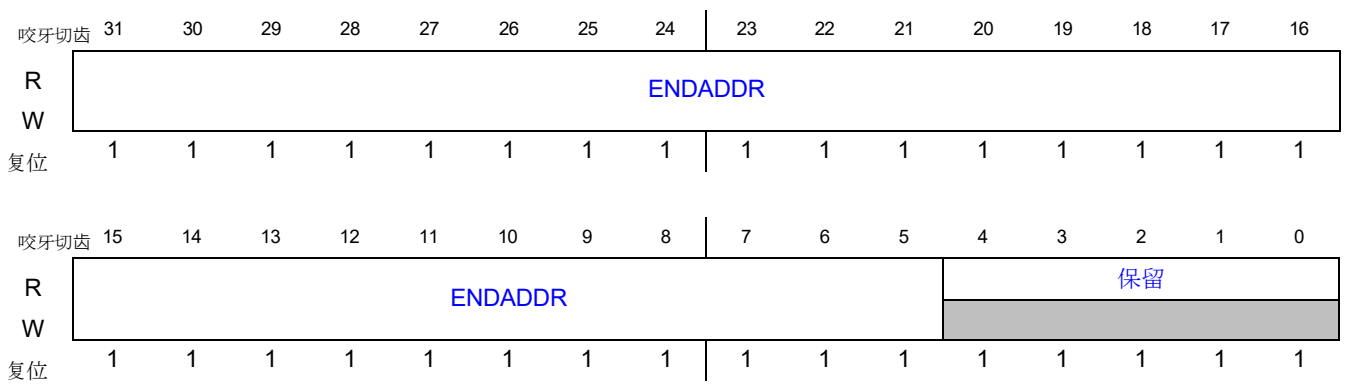
### 13.4.6.1 偏移

| 注册 | 偏移 |
|---|---|
| RGD0_WORD1 | 404h |

### 13.4.6.2 职能

区域描述符的第二个字定义了内存区域的31-模32字节的结束地址。对该寄存器的写入将清除区域描述符的有效位（RGDn_WORD3[VLD]）。

### 13.4.6.3 图示




### 13.4.6.4 领域

| 场地 | 职能 |
|---|---|
| 31-5<br>ENDADDR | 最终地址<br>定义内存区域的31-模数-32字节终端地址的最有效位。<br>**注意**：MPU不验证ENDADDR≥SRTADDR。 |

| | |
|---|---|
| 4-0<br>- | 保留 |

## 13.4.7 区域描述符0，字2 (RGD0_WORD2)

### 13.4.7.1 偏移

| 注册 | 偏移 |
| --- | --- |
| RGD0_WORD2 | 408h |

### 13.4.7.2 职能

区域描述符的第三个字定义了内存区域的访问控制权限。访问控制权限取决于总线主站的两个大分类。

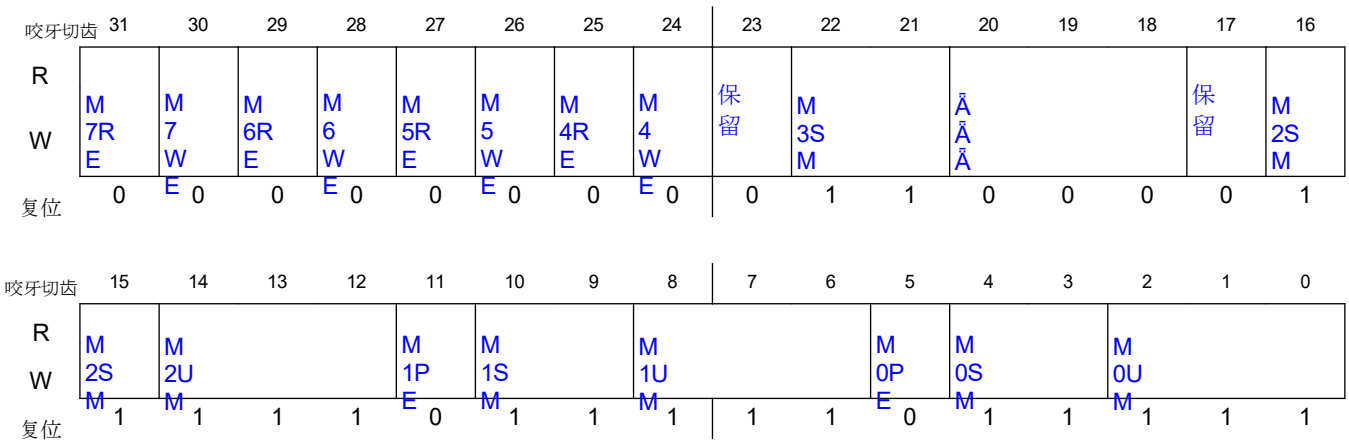
- 总线主站0-3有一个5位字段，定义了用户和监督员模式访问的单独权限。这些总线主站中的每一个都可以选择在定义中包括一个进程识别字段（如果为主站实施）。
- 总线主站4-7被限制为单独的读写权限。

对于总线主站0-3的特权权利，有三个标志与此功能相关。

- 读取（r）是指使用操作数（数据）获取来访问引用的内存地址
- 写（w）是指使用存储（数据）指令更新引用的内存地址
- 执行(x)指的是使用指令取值读取参考的内存地址

对RGDn_WORD2的写入会清除区域描述符的有效位（RGDn_WORD3[VLD]）。如果只是更新访问控制，可以写到RGDAACn，因为对这些位置的存储不会影响描述符的有效位。

## 13.4.7.3 图示




## 13.4.7.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>M7RE | 总线主站 7 读取功能<br>0b - 总线主站7的读数因访问错误而终止，不执行读数 1b - 总线主站7允许读数 |
| 30<br>M7WE | 总线主控 7 写入启用<br>0b - 总线主站7的写操作因访问错误而终止，不进行写操作 1b - 允许总线主站7写操作 |
| 29<br>M6RE | 总线主站 6 读取启用<br>0b - 总线主站6的读数因访问错误而终止，不进行读数 1b - 总线主站6允许读数 |
| 28<br>M6WE | 总线主控 6 写入启用<br>0b - 总线主站6的写入以访问错误结束，不进行写入 1b - 允许总线主站6的写入 |
| 27<br>M5RE | 总线主控 5 读取启用<br>0b - 总线主站5的读数因访问错误而终止，不进行读数 1b - 总线主站5允许读数 |
| 26<br>M5WE | 总线主控 5 写入启用<br>0b - 总线主站5的写入以访问错误结束，不进行写入 1b - 允许总线主站5的写入 |
| 25<br>M4RE | 总线主站 4 读取功能<br>0b - 总线主站4的读数因访问错误而终止，不执行读数 1b - 总线主站4允许读数 |
| 24<br>M4WE | 总线主站4写启用<br>0b - 总线主站4的写操作因访问错误而终止，不进行写操作 1b - 允许总线主站4写操作 |
| 23<br>- | 保留<br>此位必须写为零。 |
| 22-21 | 总线主控 3 监督者模式访问控制 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| M3SM | 定义监督员模式下的总线主站3的访问控制。<br>00b - r/w/x；允许读、写和执行<br>01b - r/x；允许读和执行，但不能写 10b - r/w；<br>允许读和写，但不能执行 11b - 与M3UM中定义<br>的用户模式一样 |
| 20-18<br>ÃÃÃ | 总线主控 3 用户模式访问控制<br>定义了用户模式下的总线主站3的访问控制。M3UM由三个独立的位组成，启用读（r）、写（w）和执行（x）的权限。在M3UM[2:0]。M3UM[2]控制读权限，M3UM[1]控制写权限，M3UM[0]控制执行权限。对于每个位。<br>0b - 该模式的尝试性访问可能以访问错误终止（如果其他描述符不允许），并且不执行访问。<br>1b - 允许给定的访问类型发生。 |
| 17<br>- | 保留<br>此位必须写为零。 |
| 16-15<br>M2SM | 总线主控 2 监督者模式访问控制<br>定义监督员模式下的总线主站2的访问控制。<br>00b - r/w/x；允许读、写和执行<br>01b - r/x；允许读和执行，但不能写 10b - r/w；<br>允许读和写，但不能执行 11b - 与M2UM中定义<br>的用户模式相同 |
| 14-12<br>M2UM | 总线主控 2 用户模式 访问控制<br>定义了用户模式下的总线主站2的访问控制。M2UM由三个独立的位组成，启用读（r）、写（w）和执行（x）的权限。在M2UM[2:0]中。M2UM[2]控制读权限，M2UM[1]控制写权限，M2UM[0]控制执行权限。对于每一个位。<br>0b - 该模式的尝试性访问可能以访问错误终止（如果其他描述符不允许），并且不执行访问。<br>1b - 允许给定的访问类型发生。 |
| 11<br>M1PE | 总线主站1进程标识符启用<br>0b - 在评价中不包括过程标识符<br>1b - 在区域命中评估中包括过程标识符和掩码（RGDn.RGDAAC）。 |
| 10-9<br>M1SM | 总线主站1监督员模式访问控制<br>定义监督员模式下总线主站1的访问控制。<br>00b - r/w/x；允许读、写和执行<br>01b - r/x；允许读和执行，但不能写 10b - r/w；<br>允许读和写，但不能执行 11b - 与 M1UM 中定义<br>的用户模式相同 |

| | |
|---|---|
| 8-6<br>M1UM | 总线主站1用户模式访问控制<br>定义了用户模式下总线主站1的访问控制。M1UM由三个独立的位组成，启用读（r）、写（w）和执行（x）的权限。在M1UM[2:0]中。M1UM[2]控制读权限，M1UM[1]控制写权限，M1UM[0]控制执行权限。对于每一个位。<br>0b - 该模式的尝试性访问可能以访问错误终止（如果其他描述符不允许），并且不执行访问。<br>1b - 允许给定的访问类型发生。 |
| 5<br>M0PE | 总线主站 0 进程标识符启用<br>0b - 在评价中不包括过程标识符<br>1b - 在区域命中评估中包括过程标识符和掩码（RGDn.RGDAAC）。 |
| 4-3 | 总线主控 0 监督者模式访问控制 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| M0SM | 定义了监督员模式下总线主站0的访问控制。<br>00b - r/w/x；允许读、写和执行<br>01b - r/x；允许读和执行，但不能写 10b - r/w；<br>允许读和写，但不能执行 11b - 与M0UM中定义<br>的用户模式相同 |
| 2-0<br>M0UM | 总线主站 0 用户模式访问控制<br>定义了用户模式下总线主站0的访问控制。M0UM由三个独立的位组成，启用读（r）、写（w）和执行（x）的权限。在M0UM[2:0]中。M0UM[2]控制读权限，M0UM[1]控制写权限，M0UM[0]控制执行权限。对于每一个位。<br>0b - 该模式的尝试性访问可能以访问错误终止（如果其他描述符不允许），并且不执行访问。<br>1b - 允许给定的访问类型发生。 |

## 13.4.8 区域描述符0，字3 (RGD0_WORD3)

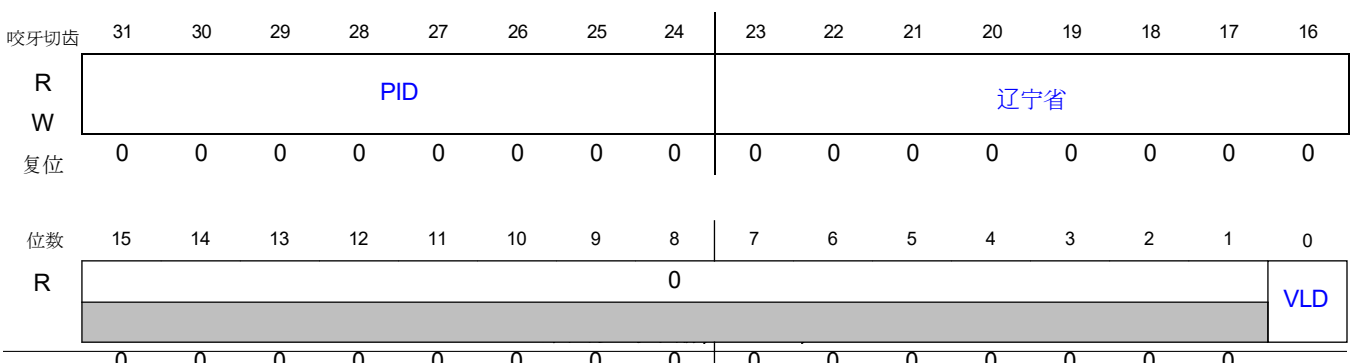
### 13.4.8.1 偏移

| 注册 | 偏移 |
|---|---|
| RGD0_WORD3 | 40Ch |

### 13.4.8.2 职能

区域描述符的第四个字包含可选的进程标识符和掩码，加上区域描述符的有效位。

### 13.4.8.3 图示



| | |
|---|---|
| W | |
| 重置 | 1 |

### 13.4.8.4 领域

| 场地 | 职能 |
|---|---|
| 31-24<br>PID | 过程标识符<br>如果RGDn_WORD2[MxPE]被设置，则指定包含在区域命中确定中的进程标识。PIDMASK可以屏蔽这个字段中的各个位。 |
| 23-16<br>辽宁省 | 过程标识符掩码<br>提供一个屏蔽功能，以便多个进程标识符可以被包括在区域命中确定的一部分。如果PIDMASK中的一个位被设置，那么相应的PID位在比较中被忽略。<br>如果RGDn_WORD2[MxPE]被设置，该字段和PID将包括在区域命中确定中。关于处理PID和PIDMASK的更多信息，请参见 "访问评估-命中确定"。 |
| 15-1<br>- | 保留 |
| 0<br>VLD | 有效<br>标志着区域描述符是有效的。对RGDn_WORD0-2的任何写入都会清除这个位。<br>0b - 区域描述符无效 1b - 区域描述符有效 |

## 13.4.9 区域描述符n，字1 (RGD1_WORD1 - RGD15_WO RD1)

### 13.4.9.1 偏移

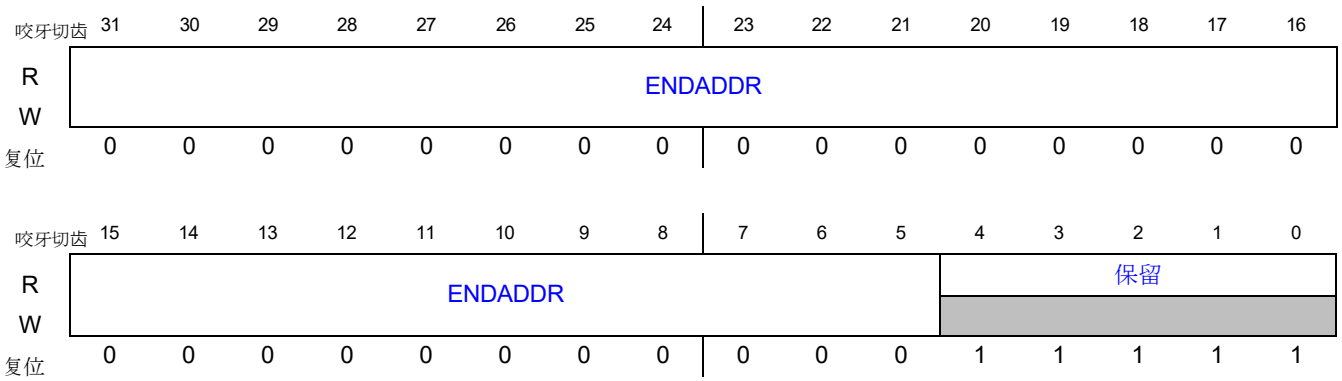
对于n=1到15。

| 注册 | 偏移 |
|---|---|
| 呼叫中心：<br>RGDn_WORD1 | 404h + (n × 10h) |

### 13.4.9.2 职能

区域描述符的第二个字定义了内存区域的31-
模32字节的结束地址。对该寄存器的写入将清除区域描述符的有效位（RGDn_WO RD3[VLD]）。

### 13.4.9.3 图示




### 13.4.9.4 领域

| 场地 | 职能 |
|---|---|
| 31-5<br>ENDADDR | 最终地址<br>定义内存区域的31-模数-32字节终端地址的最有效位。<br>**注意**：MPU不验证ENDADDR≥SRTADDR。 |
| 4-0<br>- | 保留 |

## 13.4.10 区域描述符n，字2（RGD1_WORD2 - RGD15_ WORD2）。

### 13.4.10.1 偏移

对于n=1到15。

| 注册 | 偏移 |
|---|---|
| 呼叫中心：<br>RGDn_WORD2 | 408h + (n × 10h) |

### 13.4.10.2 职能

区域描述符的第三个字定义了内存区域的访问控制权限。访问控制权限取决于总线主站的两个大分类。

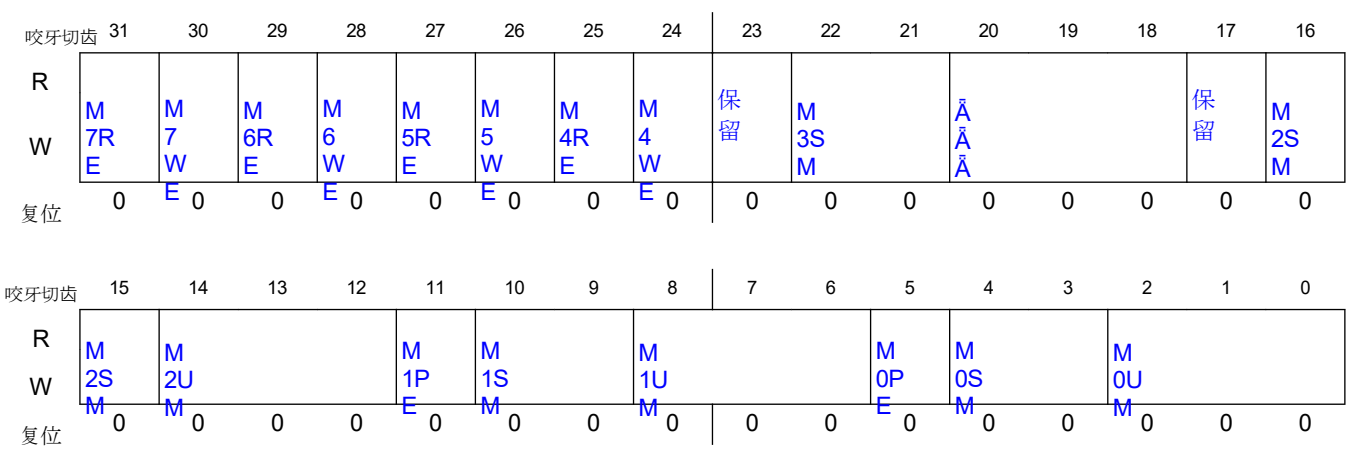
- 总线主站0-3有一个5位字段，定义了用户和监督员模式访问的单独权限。这些总线主站中的每一个都可以选择在定义中包括一个进程识别字段（如果为主站实施）。
- 总线主站4-7被限制为单独的读写权限。

对于总线主站0-3的特权权利，有三个标志与此功能相关。

- 读取（r）是指使用操作数（数据）获取来访问引用的内存地址
- 写（w）是指使用存储（数据）指令更新引用的内存地址
- 执行(x)指的是使用指令取值读取参考的内存地址

对RGDn_WORD2的写入会清除区域描述符的有效位（RGDn_WORD3[VLD]）。如果只是更新访问控制，可以写到RGDAACn，因为对这些位置的存储不会影响描述符的有效位。

### 13.4.10.3 图示



### 13.4.10.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>M7RE | 总线主站 7 读取功能<br>0b - 总线主站7的读数因访问错误而终止，不执行读数 1b - 总线主站7允许读数 |
| 30<br>M7WE | 总线主控 7 写入启用<br>0b - 总线主站7的写操作因访问错误而终止，不进行写操作 1b - 允许总线主站7写操作 |
| 29<br>M6RE | 总线主站 6 读取启用<br>0b - 总线主站6的读数因访问错误而终止，不进行读数 1b - 总线主站6允许读数 |
| 28<br>M6WE | 总线主控 6 写入启用<br>0b - 总线主站6的写入以访问错误结束，不进行写入 1b - 允许总线主站6的写入 |
| 27<br>M5RE | 总线主控 5 读取启用<br>0b - 总线主站5的读数因访问错误而终止，不进行读数 1b - 总线主站5允许读数 |
| 26<br>M5WE | 总线主控 5 写入启用<br>0b - 总线主站5的写操作因访问错误而终止，不进行写操作 1b - 允许总线主站5的写操作 |
| 25<br>M4RE | 总线主站 4 读取功能<br>0b - 总线主站4的读数因访问错误而终止，不执行读数 1b - 总线主站4允许读数 |
| 24<br>M4WE | 总线主站4写启用<br>0b - 总线主站4的写操作以访问错误结束，不进行写操作 1b - 总线主站4允许写操作 |
| 23<br>- | 保留<br>此位必须写为零。 |
| 22-21<br>M3SM | 总线主控 3 监督者模式访问控制<br>定义监督员模式下的总线主站3的访问控制。<br>00b - r/w/x；允许读、写和执行<br>01b - r/x；允许读和执行，但不能写 10b - r/w；<br>允许读和写，但不能执行 11b - 与M3UM中定义<br>的用户模式一样 |
| 20-18<br>ÃÃÃ | 总线主控 3 用户模式访问控制<br>定义了用户模式下的总线主站3的访问控制。M3UM由三个独立的位组成，启用读（r）、写（w）和执行（x）的权限。在M3UM[2:0]。M3UM[2]控制读权限，M3UM[1]控制写权限，M3UM[0]控制执行权限。对于每个位。<br>0b - 该模式的尝试性访问可能以访问错误终止（如果其他描述符不允许），并且不执行访问。<br>1b - 允许给定的访问类型发生。 |
| 17<br>- | 保留<br>此位必须写为零。 |

| | |
|---|---|
| 16-15<br>M2SM | 总线主控 2 监督者模式访问控制<br><br>定义监督员模式下的总线主站2的访问控制。<br>00b - r/w/x；允许读、写和执行<br>01b - r/x；允许读和执行，但不能写 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | 10b - r/w；允许读和写，但不能执行 11b - 与<br>M2UM中定义的用户模式相同 |
| 14-12<br>M2UM | 总线主控 2 用户模式 访问控制<br>定义了用户模式下的总线主站2的访问控制。M2UM由三个独立的位组成，启用读（r）、写（w）和执行（x）的权限。在M2UM[2:0]中。M2UM[2]控制读权限，M2UM[1]控制写权限，M2UM[0]控制执行权限。对于每一个位。<br>0b - 该模式的尝试性访问可能以访问错误终止（如果其他描述符不允许），并且不执行访问。<br>1b - 允许给定的访问类型发生。 |
| 11<br>M1PE | 总线主站1进程标识符启用<br>0b - 在评价中不包括过程标识符<br>1b - 在区域命中评估中包括过程标识符和掩码（RGDn.RGDAAC）。 |
| 10-9<br>M1SM | 总线主站1监督员模式访问控制<br>定义监督员模式下总线主站1的访问控制。<br>00b - r/w/x；允许读、写和执行<br>01b - r/x；允许读和执行，但不能写 10b - r/w；<br>允许读和写，但不能执行 11b - 与 M1UM 中定义<br>的用户模式相同 |
| 8-6<br>M1UM | 总线主站1用户模式访问控制<br>定义了用户模式下总线主站1的访问控制。M1UM由三个独立的位组成，启用读（r）、写（w）和执行（x）的权限。在M1UM[2:0]中。M1UM[2]控制读权限，M1UM[1]控制写权限，M1UM[0]控制执行权限。对于每一个位。<br>0b - 该模式的尝试性访问可能以访问错误终止（如果其他描述符不允许），并且不执行访问。<br>1b - 允许给定的访问类型发生。 |
| 5<br>M0PE | 总线主站 0 进程标识符启用<br>0b - 在评价中不包括过程标识符<br>1b - 在区域命中评估中包括过程标识符和掩码（RGDn.RGDAAC）。 |
| 4-3<br>M0SM | 总线主控 0 监督者模式访问控制<br>定义了监督员模式下总线主站0的访问控制。<br>00b - r/w/x；允许读、写和执行<br>01b - r/x；允许读和执行，但不能写 10b - r/w；<br>允许读和写，但不能执行 11b - 与M0UM中定义<br>的用户模式相同 |
| 2-0<br>M0UM | 总线主站 0 用户模式访问控制<br>定义了用户模式下总线主站0的访问控制。M0UM由三个独立的位组成，启用读（r）、写（w）和执行（x）的权限。在M0UM[2:0]中。M0UM[2]控制读权限，M0UM[1]控制写权限，M0UM[0]控制执行权限。对于每一个位。<br>0b - 该模式的尝试性访问可能以访问错误终止（如果其他描述符不允许），并且不执行访问。<br>1b - 允许给定的访问类型发生。 |

# 13.4.11 区域描述符n，字3（RGD1_WORD3 - RGD15_ WORD3）。

## 13.4.11.1 偏移

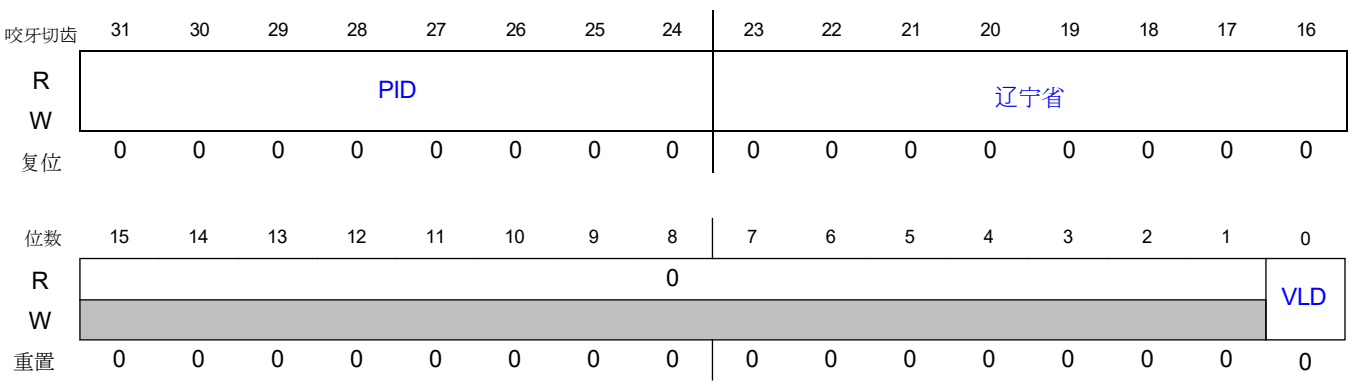
对于n=1到15。

| 注册 | 偏移 |
|---|---|
| 淘宝网 | 40Ch + (n × 10h) |

## 13.4.11.2 职能

区域描述符的第四个字包含可选的进程标识符和掩码，加上区域描述符的有效位。

## 13.4.11.3 图示




## 13.4.11.4 领域

| 场地 | 职能 |
|---|---|
| 31-24<br>PID | 过程标识符<br>如果RGDn_WORD2[MxPE]被设置，则指定包含在区域命中确定中的进程标识。PIDMASK可以屏蔽这个字段中的各个位。 |

| | |
|---|---|
| 23-16<br>辽宁省 | 过程标识符掩码<br>提供一个屏蔽功能，以便多个进程标识符可以被包括在区域命中确定的一部分。如果PIDMASK中的一个位被设置，那么相应的PID位在比较中被忽略。 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | 如果RGDn_WORD2[MxPE]被设置，该字段和PID将包括在区域命中确定中。关于处理PID和PIDMASK的更多信息，请参见 "访问评估-命中确定"。 |
| 15-1<br>- | 保留 |
| 0<br>VLD | 有效<br>标志着区域描述符是有效的。对RGDn_WORD0-2的任何写入都会清除这个位。<br>0b - 区域描述符无效 1b - 区域描述符有效 |

## 13.4.12 区域描述符备用访问控制0（RGDA AC0）。

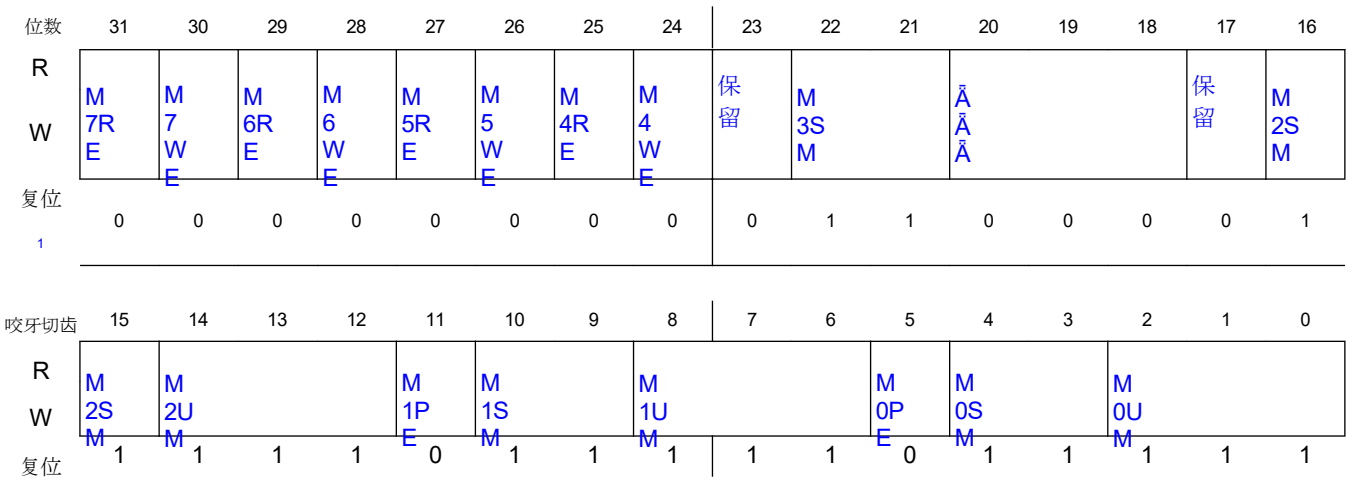
### 13.4.12.1 偏移

| 注册 | 偏移 |
| --- | --- |
| RGDAAC0 | 800h |

### 13.4.12.2 职能

因为软件可以在不同的任务执行过程中只调整区域描述符（RGDn_WORD2）内的访问控制，所以可以对这个32位的实体进行另一种编程视图。对该寄存器的写入不会影响描述符的有效位。

## 13.4.12.3 图示




1.RGDAAC0复位到61F7DFh。其他RGDAACn寄存器重置为0h。

## 13.4.12.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>M7RE | 总线主站 7 读取功能<br>0b - 总线主站7的读数因访问错误而终止，不执行读数 1b - 总线主站7允许读数 |
| 30<br>M7WE | 总线主控 7 写入启用<br>0b - 总线主站7的写操作因访问错误而终止，不进行写操作 1b - 允许总线主站7写操作 |
| 29<br>M6RE | 总线主站 6 读取启用<br>0b - 总线主站6的读数因访问错误而终止，不进行读数 1b - 总线主站6允许读数 |
| 28<br>M6WE | 总线主控 6 写入启用<br>0b - 总线主站6的写入以访问错误结束，不进行写入 1b - 允许总线主站6的写入 |
| 27<br>M5RE | 总线主控 5 读取启用<br>0b - 总线主站5的读数因访问错误而终止，不进行读数 1b - 总线主站5允许读数 |
| 26<br>M5WE | 总线主控 5 写入启用<br>0b - 总线主站5的写入以访问错误结束，不进行写入 1b - 允许总线主站5的写入。 |
| 25<br>M4RE | 总线主站 4 读取功能<br>0b - 总线主站4的读数因访问错误而终止，不执行读数 1b - 总线主站4允许读数 |
| 24<br>M4WE | 总线主站4写启用<br>0b - 总线主站4的写操作以访问错误结束，不进行写操作 1b - 总线主站4允许写操作 |

| | |
|---|---|
| 23<br><br>- | 保留<br><br>此位必须写为零。 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| 22-21<br>M3SM | 总线主控 3 监督者模式访问控制<br>定义监督员模式下的总线主站3的访问控制。<br>00b - r/w/x；允许读、写和执行<br>01b - r/x；允许读和执行，但不能写 10b - r/w；<br>允许读和写，但不能执行 11b - 与M3UM中定义<br>的用户模式一样 |
| 20-18<br>ÃÃÃ | 总线主控 3 用户模式访问控制<br>定义了用户模式下的总线主站3的访问控制。M3UM由三个独立的位组成，启用读（r）、写（w）和执行（x）的权限。在M3UM[2:0]。M3UM[2]控制读权限，M3UM[1]控制写权限，M3UM[0]控制执行权限。对于每个位。<br>0b - 该模式的尝试性访问可能以访问错误终止（如果其他描述符不允许），并且不执行访问。<br>1b - 允许给定的访问类型发生。 |
| 17<br>- | 保留<br>此位必须写为零。 |
| 16-15<br>M2SM | 总线主控 2 监督者模式访问控制<br>定义监督员模式下的总线主站2的访问控制。<br>00b - r/w/x；允许读、写和执行<br>01b - r/x；允许读和执行，但不能写 10b - r/w；<br>允许读和写，但不能执行 11b - 与M2UM中定义<br>的用户模式相同 |
| 14-12<br>M2UM | 总线主站2用户模式访问控制<br>定义了用户模式下的总线主站2的访问控制。M2UM由三个独立的位组成，启用读（r）、写（w）和执行（x）的权限。在M2UM[2:0]中。M2UM[2]控制读权限，M2UM[1]控制写权限，M2UM[0]控制执行权限。对于每一个位。<br>0b - 该模式的尝试性访问可能以访问错误终止（如果其他描述符不允许），并且不执行访问。<br>1b - 允许给定的访问类型发生。 |
| 11<br>M1PE | 总线主站1进程识别器启用<br>**注意：**对于RGDAAC0：只有总线主站1可以向M1PE写入。0b -<br>在评估中不包括过程标识符<br>1b - 在区域命中评估中包括过程标识符和掩码（RGDn.RGDAAC）。 |
| 10-9<br>M1SM | 总线主站1监督员模式访问控制<br>定义监督员模式下总线主站1的访问控制。<br>**注意：**对于RGDAAC0：只有总线主站1可以向M1SM写入。<br>00b - r/w/x；允许读、写和执行 |

*表继续在下一页...*

| | |
|---|---|
| | 01b - r/x；允许读和执行，但不能写 10b - r/w；<br>允许读和写，但不能执行 11b - 与 M1UM 中定义<br>的用户模式相同 |
| 8-6<br>M1UM | 总线主站1用户模式访问控制<br>定义了用户模式下总线主站1的访问控制。M1UM由三个独立的位组成，启用读（r）、写（w）和执行（x）的权限。在M1UM[2:0]中。M1UM[2]控制读权限，M1UM[1]控制写权限，M1UM[0]控制执行权限。<br>**注意：**对于RGDAAC0：只有总线主站1可以向M1UM写入。对于<br>每个位。<br>0b - 对该模式的尝试性访问可能会以访问错误而终止（如果不允许）。<br>另一个描述符），并且没有进行访问。 |

| 场地 | 职能 |
| --- | --- |
| | 1b - 允许给定的访问类型发生。 |
| 5<br>M0PE | 总线主站 0 进程识别器启用<br>0b - 在评价中不包括过程标识符<br>1b - 在区域命中评估中包括过程标识符和掩码（RGDn.RGDAAC）。 |
| 4-3<br>M0SM | 总线主控 0 监督者模式访问控制<br>定义了监督员模式下总线主站0的访问控制。<br>00b - r/w/x；允许读、写和执行<br>01b - r/x；允许读和执行，但不能写 10b - r/w；<br>允许读和写，但不能执行 11b - 与M0UM中定义<br>的用户模式相同 |
| 2-0<br>M0UM | 总线主站 0 用户模式访问控制<br>定义了用户模式下总线主站0的访问控制。M0UM由三个独立的位组成，启用读（r）、写（w）和执行（x）的权限。在M0UM[2:0]中。M0UM[2]控制读权限，M0UM[1]控制写权限，M0UM[0]控制执行权限。对于每一个位。<br>0b - 该模式的尝试性访问可能以访问错误终止（如果其他描述符不允许），并且不执行访问。<br>1b - 允许给定的访问类型发生。 |

## 13.4.13 区域描述符备用访问控制n (RGDA AC1 - RGDAAC15)

### 13.4.13.1 偏移

对于n=1到15。

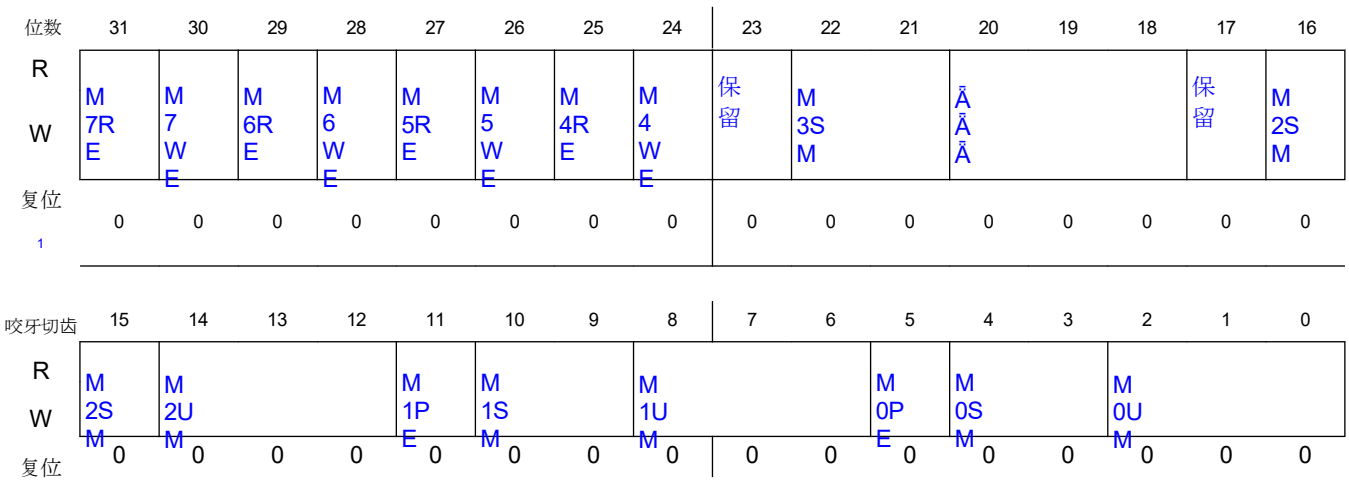
| 注册 | 偏移 |
| --- | --- |
| RGDAACn | 800h + (n × 4h) |

### 13.4.13.2 职能

因为软件可以在不同的任务执行过程中只调整区域描述符（RGDn_WORD2）内的访问控制，所以可以对这个32位的实体进行另一种编程视图。对该寄存器的写入不会影响描述符的有效位。

*表继续在下一页...*

## 13.4.13.3 图示




1.RGDAAC0复位到61F7DFh。其他RGDAACn寄存器重置为0h。

## 13.4.13.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>M7RE | 总线主站 7 读取功能<br>0b - 总线主站7的读数因访问错误而终止，不执行读数 1b - 总线主站7允许读数 |
| 30<br>M7WE | 总线主控 7 写入启用<br>0b - 总线主站7的写操作因访问错误而终止，不进行写操作 1b - 允许总线主站7的写操作 |
| 29<br>M6RE | 总线主站 6 读取启用<br>0b - 总线主站6的读数因访问错误而终止，不进行读数 1b - 总线主站6允许读数 |
| 28<br>M6WE | 总线主控 6 写入启用<br>0b - 总线主站6的写入以访问错误结束，不进行写入 1b - 允许总线主站6的写入 |
| 27<br>M5RE | 总线主控 5 读取启用<br>0b - 总线主站5的读数因访问错误而终止，不进行读数 1b - 总线主站5允许读数 |
| 26<br>M5WE | 总线主控 5 写入启用<br>0b - 总线主站5的写操作因访问错误而终止，不进行写操作 1b - 允许总线主站5的写操作 |
| 25<br>M4RE | 总线主站 4 读取功能<br>0b - 总线主站4的读数因访问错误而终止，不执行读数 1b - 总线主站4允许读数 |
| 24<br>M4WE | 总线主站4写启用<br>0b - 总线主站4的写操作以访问错误结束，不进行写操作 1b - 总线主站4允许写操作 |

| | |
|---|---|
| 23<br>-<br> | 保留<br><br>此位必须写为零。 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| 22-21<br>M3SM | 总线主控 3 监督者模式访问控制<br>定义监督员模式下的总线主站3的访问控制。<br>00b - r/w/x；允许读、写和执行<br>01b - r/x；允许读和执行，但不能写 10b - r/w；<br>允许读和写，但不能执行 11b - 与M3UM中定义<br>的用户模式一样 |
| 20-18<br>ÃÃÃ | 总线主控 3 用户模式访问控制<br>定义了用户模式下的总线主站3的访问控制。M3UM由三个独立的位组成，启用读（r）、写（w）和执行（x）的权限。在M3UM[2:0]。M3UM[2]控制读权限，M3UM[1]控制写权限，M3UM[0]控制执行权限。对于每个位。<br>0b - 该模式的尝试性访问可能以访问错误终止（如果其他描述符不允许），并且不执行访问。<br>1b - 允许给定的访问类型发生。 |
| 17<br>- | 保留<br>此位必须写为零。 |
| 16-15<br>M2SM | 总线主控 2 监督者模式访问控制<br>定义监督员模式下的总线主站2的访问控制。<br>00b - r/w/x；允许读、写和执行<br>01b - r/x；允许读和执行，但不能写 10b - r/w；<br>允许读和写，但不能执行 11b - 与M2UM中定义<br>的用户模式相同 |
| 14-12<br>M2UM | 总线主站2用户模式访问控制<br>定义了用户模式下的总线主站2的访问控制。M2UM由三个独立的位组成，启用读（r）、写（w）和执行（x）的权限。在M2UM[2:0]中。M2UM[2]控制读权限，M2UM[1]控制写权限，M2UM[0]控制执行权限。对于每一个位。<br>0b - 该模式的尝试性访问可能以访问错误终止（如果其他描述符不允许），并且不执行访问。<br>1b - 允许给定的访问类型发生。 |
| 11<br>M1PE | 总线主站1进程识别器启用<br>**注意：**对于RGDAAC0：只有总线主站1可以向M1PE写入。0b -<br>在评估中不包括过程标识符<br>1b - 在区域命中评估中包括过程标识符和掩码（RGDn.RGDAAC）。 |
| 10-9<br>M1SM | 总线主站1监督员模式访问控制<br>定义监督员模式下总线主站1的访问控制。<br>**注意：**对于RGDAAC0：只有总线主站1可以向M1SM写入。<br>00b - r/w/x；允许读、写和执行<br>01b - r/x；允许读和执行，但不能写 10b - r/w；<br>允许读和写，但不能执行 11b - 与 M1UM 中定义<br>的用户模式相同 |

| | |
|---|---|
| 8-6<br>M1UM | 总线主站1用户模式访问控制<br><br>定义了用户模式下总线主站1的访问控制。M1UM由三个独立的位组成，启用读（r）、写（w）和执行（x）的权限。在M1UM[2:0]中。M1UM[2]控制读权限，M1UM[1]控制写权限，M1UM[0]控制执行权限。<br><br>**注意：**对于RGDAAC0：只有总线主站1可以向M1UM写入。对于每个位。<br>0b - 对该模式的尝试性访问可能会以访问错误而终止（如果不允许）。<br>另一个描述符），并且没有进行访问。 |

*表继续在下一页...*

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | 1b - 允许给定的访问类型发生。 |
| 5<br>M0PE | 总线主站 0 进程识别器启用<br>0b - 在评价中不包括过程标识符<br>1b - 在区域命中评估中包括过程标识符和掩码（RGDn.RGDAAC）。 |
| 4-3<br>M0SM | 总线主控 0 监督者模式访问控制<br>定义了监督员模式下总线主站0的访问控制。<br>00b - r/w/x；允许读、写和执行<br>01b - r/x；允许读和执行，但不能写 10b - r/w；<br>允许读和写，但不能执行 11b - 与M0UM中定义<br>的用户模式相同 |
| 2-0<br>M0UM | 总线主站 0 用户模式访问控制<br>定义了用户模式下总线主站0的访问控制。M0UM由三个独立的位组成，启用读（r）、写（w）和执行（x）的权限。在M0UM[2:0]中。M0UM[2]控制读权限，M0UM[1]控制写权限，M0UM[0]控制执行权限。对于每一个位。<br>0b - 该模式的尝试性访问可能以访问错误终止（如果其他描述符不允许），并且不执行访问。<br>1b - 允许给定的访问类型发生。 |

## 13.5 功能 描述

在这一节中，详细介绍了MPU的功能操作，包括访问评估宏的操作和错误终止的总线周期的处理。

### 13.5.1 访问评价 宏

MPU的基本操作是在访问评估宏中进行的，这是一个复制在二维连接矩阵中的硬件结构。如下图所示，访问评估宏输入横梁总线地址相位信号和区域描述符（RGDn）的内容并执行两个主要功能。

- 区域命中率确定
- 检测到违反访问保护的行为

下图是一个功能框图。

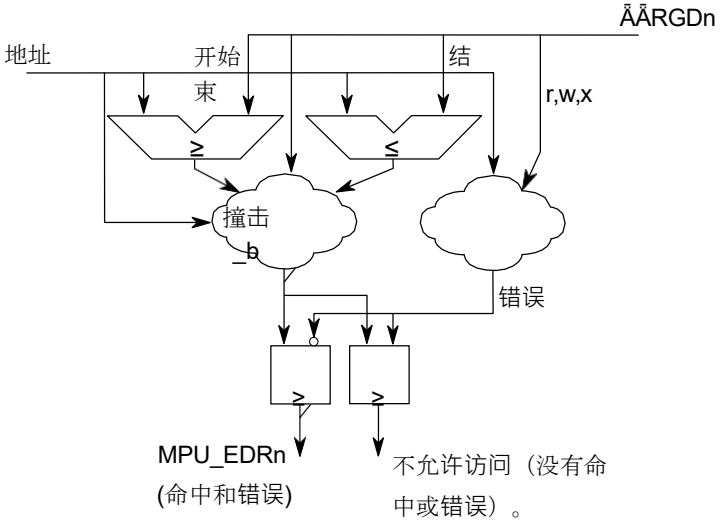



**图13-2.MPU访问评估宏**

### 13.5.1.1 命中率确定

为了确定当前参考是否命中给定区域，使用两个幅度比较器与该区域的开始和结束地址。这一部分确定命中的布尔公式是：。

```
region_hit = ((addr[31:5] >= RGDn_Word0[SRTADDR]) & (addr[31:5] <= RGDn_Word1[ENDADDR])) &
RGDn_Word3[VLD]
```

其中addr是当前参考地址，RGDn_Word0[SRTADDR]和RGDn_Word1[ENDADDR]是起始和结束地址，RGDn_Word3[VLD]是有效位。

**注意事项**

MPU没有验证ENDADDR≥SRTADDR。

除了将参考地址与区域描述符的起始和结束地址进行比较外，还将根据区域描述符的PID和PIDMASK字段检查可选进程标识符。进程标识符命中项的形成方式如下。

```
pid_hit = ~RGDn_Word2[MxPE]|
          ((current_pid |
          RGDn_Word3[PIDMASK]) == (RGDn_Word3[PID] | RGDn_Word3[PIDMASK])
```

其中current_pid是来自当前总线主站的选定进程标识符，RGDn_Word3[PID]和RGDn_Word3[PIDMASK]是来自区域描述符*n*的进程标识符字段。对于不输出进程标识符的总线主站，MPU强制pid_hit项断电。

#### 13.5.1.2 违反特权的确定

当访问评估宏确定区域命中时，该逻辑也在评估当前访问是否被区域描述符中定义的权限所允许。使用主控和监督者/用户模式信号，从区域描述符中的适当字段生成一组有效权限。然后，保护违反逻辑根据有效权限对访问进行评估，如下表所示。

**表13-5.保护违反定义**

| 描述 | MxUM | | | 违反保护规定？ |
|---|---|---|---|---|
| | r | w | x | |
| 指令获取读取 | - | - | 0 | 是的，没有执行许可 |
| | - | - | 1 | 不，允许进入 |
| 数据读取 | 0 | - | - | 是的，没有阅读许可 |
| | 1 | - | - | 不，允许进入 |
| 数据写入 | - | 0 | - | 是的，没有书写许可 |
| | - | 1 | - | 不，允许进入 |

### 13.5.2 把所有的东西放在一起，错误 终端

对于每个被监测的从属端口，MPU对每个访问评估宏中的所有单独条款进行还原-AND。然后这个表达式以错误终止总线周期，并报告三种情况的保护错误。

- 如果访问没有在任何区域描述符中命中，就会报告保护错误。
- 如果访问在一个区域描述符中命中，并且该区域发出了违反保护的信号，则报告一个保护错误。
- 如果访问在多个（重叠的）区域中命中，并且所有区域都有违反保护的信号，则报告保护错误。

如第三个条件所示，对于重叠的区域，授予权限比拒绝访问更优先。这种方法对系统软件在区域描述符的分配上更加灵活。关于使用重叠区域描述符的例子，见应用信息。

### 13.5.3 电源 管理

通过清除CESR[VLD]来禁用MPU，可以最大限度地减少功率耗散。为了最小化启用的MPU的功率耗散，通过清除相关的RGDn_Word3[VLD]位，使未使用的区域描述符失效。

## 13.6 初始化 信息

在系统启动时，加载适当数量的区域描述符，包括设置RGDn_Word3[VLD]。设置CESR[VLD]可以启用该模块。

如果系统要求同时启用所有加载的区域描述符，首先确保整个MPU被禁用（CESR[VLD]=0）。

**注意事项**

如果MPU寄存器受到MPU的保护，必须设置一个区域描述符，以便在需要进一步改变时访问MPU寄存器。

## 13.7 申请 信息

在一个操作系统中，与MPU的接口一般分为以下活动。

- 创建一个新的内存区域--
  将适当的区域描述符加载到一个可用的RGDn中，使用四个连续的32位写入。硬件协助维护有效位，所以如果采用这种方法，多周期描述符写入时不会出现一致性问题。(清除RGDn_Word3[VLD]会删除/移除一个现有的内存区域)。
- 只改变访问权限--
  为了不影响有效位，写入访问控制字的备用版本（RGDAACn），所以更新不涉及一致性问题。当写入完成后，内存区域的访问权限会瞬间切换到新的值。
- 改变一个区域的起始和结束地址--
  向区域描述符（RGDn_Word{0,1,3}）至少写入三个字。字0和1分别重新定义开始和结束地址。字3重新启用区域描述符的有效位。在大多数情况下，区域描述符的所有四个字都被重写。

- 访问MPU-
  分配一个区域描述符，以限制MPU从一个特定的主站访问监督模式。

- 检测访问错误--
  当前总线周期以错误响应终止，EARn和EDRn捕获故障参考的信息。错误终止的总线周期通常会在始发的总线主站中启动错误响应。例如，一个处理器内核可以用一个总线错误异常来响应，而一个数据移动总线主站可以用一个错误中断来响应。处理器可以通过读取E{A,D}Rn来检索捕获的错误地址和细节信息。CESR[SPERR]是哪个错误寄存器包含捕获的故障数据的信号。

- 重叠区域描述符-
  应用重叠区域通常可以减少给定的访问控制集所需的描述符的数量。在重叠的内存空间中，相应区域描述符的保护权限在逻辑上是相加的（布尔OR运算符）。

  下面的双核系统例子包含四个总线主站。

  - 这两个处理器。CP0、CP1
  - 两个DMA引擎。DMA1，一个传统的数据移动引擎，在RAM和外围设备之间传输数据；DMA2，第二个引擎，只在RAM之间传输数据。

  考虑以下区域描述符的分配。

**表13-6.重叠区域描述符示例**

| 区域描述 | ÃÃRGDn | | CP0 | CP1 | DMA1 | DMA2 | |
|---|---|---|---|---|---|---|---|
| CP0代码 | 0 | | rwx | r-- | - | - | 闪光灯 |
| CP1代码 | 1 | | r-- | rwx | - | - | |
| CP0数据和堆栈 | 2 | | rw- | - | - | - | 内存 |
| CP0 → CP1共享数据 | 2 | 3 | r-- | r-- | - | - | |
| CP1 → CP0共享数据 | 4 | | | | | | |
| CP1数据和堆栈 | 4 | | - | rw- | - | - | |
| 共享的DMA数据 | 5 | | rw- | rw- | rw | rw | |
| 蒙特利尔大学 | 6 | | rw- | rw- | - | - | 周边空间 |
| 外围设备 | 7 | | rw- | rw- | rw | - | |

在这个例子中，有8个描述符用于跨越系统内存地图的三个主要空间的9个区域：闪存、RAM和外围空间。每个区域都表示四个总线主站的具体权限，这个定义提供了一套适当的共享、私有和可执行的内存空间。

特别值得关注的是两个重叠的空间：区域描述符2和3以及3和4。

RGD2定义的没有重叠的空间是一个私有数据和堆栈区域，只提供对CP0的读/写访问。RGD2和RGD3之间的重叠空间定义了一个共享数据空间，用于从CP0到CP1传递数据，访问控制由两个区域描述符的逻辑OR定义。因此，CP0有(rw- | r-) = (rw-)的权限，而CP1在这个空间有(-- | r-) = (r-)
的权限。两个DMA引擎都被排除在这个共享的处理器数据区域之外。RGD3和RGD4之间的重叠空间定义了另一个共享数据空间，这个空间用于从CP1向CP0传递数据。对于这个重叠空间，CP0有(r--|--)=(r--)权限，而CP1有(rw-|r--)=(rw-)权限。RGD4的非重叠空间只为CP1定义了一个私有数据和堆栈区域。

RGD5定义的空间是一个共享的数据区域，所有四个总线主站都可以访问。最后，映射到IPS总线上的从属外设空间被划分为两个区域。

- 一个包含MPU的编程模型，只能由两个处理器核心访问
- 剩余的外设区域可供处理器和传统的DMA1主站访问。

这个例子显示了MPU的能力在一个典型系统中的一个可能的应用。

# 第十四章
# 外围桥（AIPS-Lite）

## 14.1 特定芯片的AIPS 信息

### 14.1.1 实例化 信息

该设备包含一个外设桥。外设访问保护是由AIPS支持的。MPU不会涵盖外设访问保护。

### 14.1.2 内存 地图

外围桥接器用于访问该设备上大多数模块的寄存器。关于内存插槽的分配，请参见《参考手册》所附的S32K1xx_memory_map.xlsx。

AIPS_PACR0 - PACR31指的是平台上的外设，其对应的AIPS外设桥槽号为0-31。AIPS_OPACR0 -
OPACR95指的是平台外设，其对应的AIPS外设桥接槽号为32。
-127.

#### 14.1.2.1 寄存器复位值

下表显示了AIPS寄存器的特定芯片复位值。

**表14-1.寄存器复位值**

| 注册 | S32K116 | S32K142 | S32K144 | S32K146 | S32K148 |
|---|---|---|---|---|---|
| MPRA | TBD | 7770_0000 | 7770_0000 | 7770_0000 | 7777_0000 |
| PACRA | TBD | 5400_0000 | 5400_0000 | 5400_0000 | 5400_0000 |
| PACRB | TBD | 4400_0400 | 4400_0400 | 4400_0400 | 4400_0400 |
| ÃÃÃ | TBD | 4400_0000 | 4400_0000 | 4400_0000 | 4400_0000 |

表继续在下一页...

表14-1.寄存器复位值（续）

| 注册 | S32K116 | S32K142 | S32K144 | S32K146 | S32K148 |
|---|---|---|---|---|---|
| OPACRA | TBD | 4400_4444 | 4400_4444 | 4400_4444 | 4400_4444 |
| 农村妇女委员会（OPACRB | TBD | 0000_4400 | 0000_4400 | 0000_4400 | 0004_4440 |
| OPACRC | TBD | 0440_0044 | 0440_0044 | 0440_0044 | 0440_0044 |
| OPACRD | TBD | 4444_0400 | 4444_0400 | 4444_0400 | 4444_0400 |
| OPACRE | TBD | 4000_0040 | 4000_0040 | 4000_0040 | 4000_0040 |
| OPACRF | TBD | 4444_4400 | 4444_4400 | 4444_4400 | 4444_4400 |
| OPACRG | TBD | 0040_0000 | 0040_0000 | 0040_0000 | 0040_4400 |
| OPACRH | TBD | 0040_0000 | 0040_0000 | 0040_0000 | 0040_0000 |
| 侨联 | TBD | 0404_4440 | 0404_4440 | 0404_4440 | 0404_4444 |
| OPACRJ | TBD | 0044_0000 | 0044_0000 | 0044_0000 | 0044_4044 |
| OPACRK | TBD | 0004_0000 | 0004_0000 | 0004_0000 | 4404_0040 |
| OPACRL | TBD | 0000_0444 | 0000_0444 | 0000_0444 | 0400_0444 |

## 14.2 简介

外设桥接器将横梁开关接口转换为可以访问该芯片上大多数从属外设的接口。

外设桥占用了64MB的地址空间，它被划分为4KB的外设插槽。(有可能所有的外设插槽都没有被使用。关于插槽分配的细节，请参见内存地图一章）。该桥包括每个插槽的独立时钟使能输入，以适应较慢的外设。

### 14.2.1 特点

周边桥梁的主要特点是。

- 支持具有8位、16位和32位数据通路宽度的外围插槽
- 编程模型提供内存保护功能

### 14.2.2 一般 操作

连接到外设桥的从属设备是包含控制和状态寄存器的编程模型的模块。系统主站通过外设桥接器读写这些寄存器。

外设的寄存器图位于4KB的边界上。每个外设被分配一个或多个4KB的内存图块。

## 14.3 内存图/寄存器 定义

32位外设桥接寄存器只能在监督者模式下由受信任的总线主站访问。此外，这些寄存器必须只能通过32位对齐的访问来读出或写入。外围桥接寄存器被映射到外围访问控制寄存器A PACRA[PACR0]地址空间。

### 14.3.1 AIPS寄存器 描述

#### 14.3.1.1 AIPS内存图

AIPS_Lite的基本地址。4000_0000h

| 偏移 | 注册 | 宽度 (单位：比特) | 访问 | 重置值 |
|---|---|---|---|---|
| 0h | 主权限登记簿A（MPRA）。 | 32 | RW | 2C36_0000h |
| 20h | 外围访问控制寄存器（PACRA）。 | 32 | RW | 5400_0000h |
| 24h | 外围访问控制寄存器（PACRB）。 | 32 | RW | 4400_0400h |
| 2Ch | 外围访问控制寄存器（PACRD）。 | 32 | RW | 4400_0000h |
| 40h | 非平台外围访问控制寄存器（OPACRA）。 | 32 | RW | 4400_4444h |
| 44h | 非平台外围访问控制寄存器（OPACRB）。 | 32 | RW | 0004_4440h |
| 48h | 非平台外围访问控制寄存器（OPACRC）。 | 32 | RW | 0440_0044h |
| 4Ch | 非平台外围访问控制寄存器（OPACRD）。 | 32 | RW | 4444_0400h |
| 50h | 非平台外围访问控制寄存器（OPACRE）。 | 32 | RW | 4000_0040h |
| 54h | 非平台外围访问控制寄存器（OPACRF）。 | 32 | RW | 4444_4400h |
| 58h | 非平台外围访问控制寄存器（OPACRG）。 | 32 | RW | 0040_4400h |
| 5Ch | 非平台外围访问控制寄存器（OPACRH）。 | 32 | RW | 0040_0000h |
| 60h | 非平台外围访问控制寄存器（OPACRI）。 | 32 | RW | 0404_4444h |
| 64h | 非平台外围访问控制寄存器（OPACRJ）。 | 32 | RW | 0044_4044h |
| 68h | 非平台外围访问控制寄存器（OPACRK）。 | 32 | RW | 4404_0040h |
| 6Ch | 非平台外围访问控制寄存器（OPACRL）。 | 32 | RW | 0400_0444h |

## 14.3.1.2 主权限登记簿A（MPRA）。

### 14.3.1.2.1 偏移

| 注册 | 偏移 |
|---|---|
| MPRA | 0h |

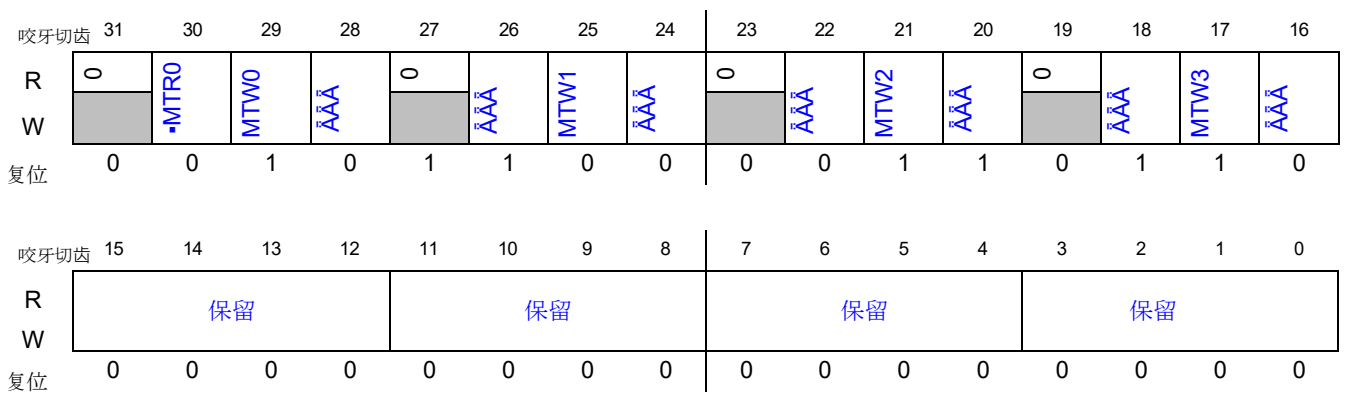
### 14.3.1.2.2 职能

MPRA规定了相同的4位字段，定义了与总线主站相关的对芯片上各种外围设备的访问权限级别。该寄存器为每个总线主站提供一个字段。

一个映射到未实现的主控或外设的寄存器字段表现为只读为零。

每个主站都被分配了一个从0到15的逻辑ID。见芯片特定AIPS信息中的主控逻辑ID分配表。

### 14.3.1.2.3 图示




### 14.3.1.2.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>- | 保留 |
| 30<br>▪MTR0 | 读取时主控0受信任<br>确定主站是否被信任用于读取访问。 |

| | |
|---|---|
| | 0b - 这个主站不被信任用于读取访问。1b - 该主站在读取访问方面是可信的。 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| 29<br>MTW0 | 写入时主站0受信任<br>决定主站是否被信任用于写访问。<br>0b - 这个主站不被信任用于写访问。1b - 这个主站在写访问方面是可信的。 |
| 28<br>ÃÃÃ | 主人 0 权限级别<br>指定如何确定主站的权限级别。0b - 从这个主站的访问被强制为用户模式。<br>1b - 该主站的访问不强制为用户模式。 |
| 27<br>- | 保留 |
| 26<br>ÃÃÃ | 主站1受托读取<br>确定主站是否被信任用于读取访问。<br>0b - 这个主站不被信任用于读取访问。1b - 该主站在读取访问方面是可信的。 |
| 25<br>MTW1 | 主站1受信于写作<br>决定主站是否被信任用于写访问。<br>0b - 这个主站不被信任用于写访问。1b - 这个主站在写访问方面是可信的。 |
| 24<br>ÃÃÃ | 主人1权限级别<br>指定如何确定主站的权限级别。0b - 从这个主站的访问被强制为用户模式。<br>1b - 该主站的访问不强制为用户模式。 |
| 23<br>- | 保留 |
| 22<br>ÃÃÃ | 主人 2 受托阅读<br>确定主站是否被信任用于读取访问。<br>0b - 这个主站不被信任用于读取访问。1b - 该主站在读取访问方面是可信的。 |
| 21<br>MTW2 | 主人2受托代写<br>决定主站是否被信任用于写访问。<br>0b - 这个主站不被信任用于写访问。1b - 这个主站在写访问方面是可信的。 |
| 20<br>ÃÃÃ | 主人2的权限级别<br>指定如何确定主站的权限级别。0b - 从这个主站的访问被强制为用户模式。<br>1b - 该主站的访问不强制为用户模式。 |
| 19<br>- | 保留 |

| | |
|---|---|
| 18<br>ĀĀĀ | 掌握3个受信任的阅读<br>确定主站是否被信任用于读取访问。<br>0b - 这个主站不被信任用于读取访问。1b - 该主站在读取访问方面是可信的。 |
| 17<br>MTW3 | 主人 3 被信任的写作<br>决定主站是否被信任用于写访问。 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| | 0b - 这个主站不被信任用于写访问。1b - 这个主站在写访问方面是可信的。 |
| 16<br>KKKK | 主3级权限级别<br>指定如何确定主站的权限级别。0b - 从这个主站的访问被强制为用户模式。<br>1b - 该主站的访问不强制为用户模式。 |
| 15-12<br>- | 保留 |
| 11-8<br>- | 保留 |
| 7-4<br>- | 保留 |
| 3-0<br>- | 保留 |

## 14.3.1.3 外围访问控制寄存器（PACRA）。

### 14.3.1.3.1 偏移

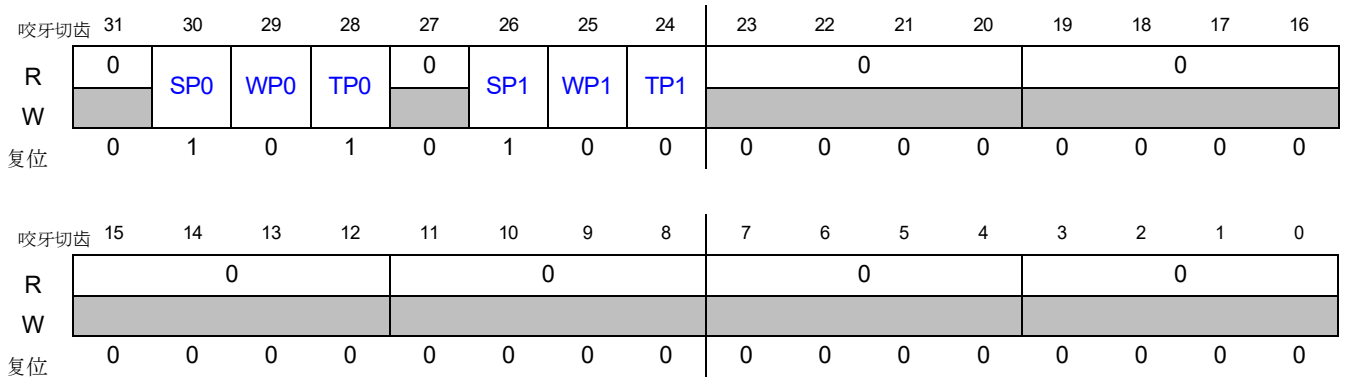
| 注册 | 偏移 |
|---|---|
| PACRA | 20h |

### 14.3.1.3.2 职能

每个PACR寄存器由8个4位PACR字段组成。每个PACR字段定义了一个特定平台上外设的访问级别。每个PACR字段的外设分配是由外设的内存映射槽定义的。关于特定外设的字段分配，请参见芯片特定的AIPS信息。

每个没有分配到外设的PACR字段都是保留的。对保留位置的读数返回0，而写数则被忽略。

### 14.3.1.3.3 图示




### 14.3.1.3.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>- | 保留 |
| 30<br>SP0 | 主管保护<br>确定外设是否需要监督员权限等级来进行访问。当这个字段被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制字段必须被设置。如果不这样做，访问就会以错误响应终止，并且没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |
| 29<br>WP0 | 写保护<br>确定外设是否允许写入访问。当这个字段被设置并尝试写访问时，访问以错误响应终止，没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |
| 28<br>TP0 | 值得信赖的保护<br>决定外设是否允许来自不受信任的主控的访问。当这个字段被设置，并且由一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |
| 27<br>- | 保留 |

| | |
|---|---|
| 26<br>SP1 | 主管保护<br>确定外设是否需要监督员权限等级来进行访问。当这个字段被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制字段必须被设置。如果不这样做，访问就会以错误响应终止，并且没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |
| 25 | 写保护 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| WP1 | 确定外设是否允许写入访问。当这个字段被设置并试图进行写访问时，访问将以错误响应终止，并且没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |
| 24<br>TP1 | 值得信赖的保护<br>决定外设是否允许来自不受信任的主控的访问。当这个字段被设置，并且由一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |
| 23-20<br>- | 保留 |
| 19-16<br>- | 保留 |
| 15-12<br>- | 保留 |
| 11-8<br>- | 保留 |
| 7-4<br>- | 保留 |
| 3-0<br>- | 保留 |

## 14.3.1.4 外围访问控制寄存器（PACRB）。

### 14.3.1.4.1 偏移

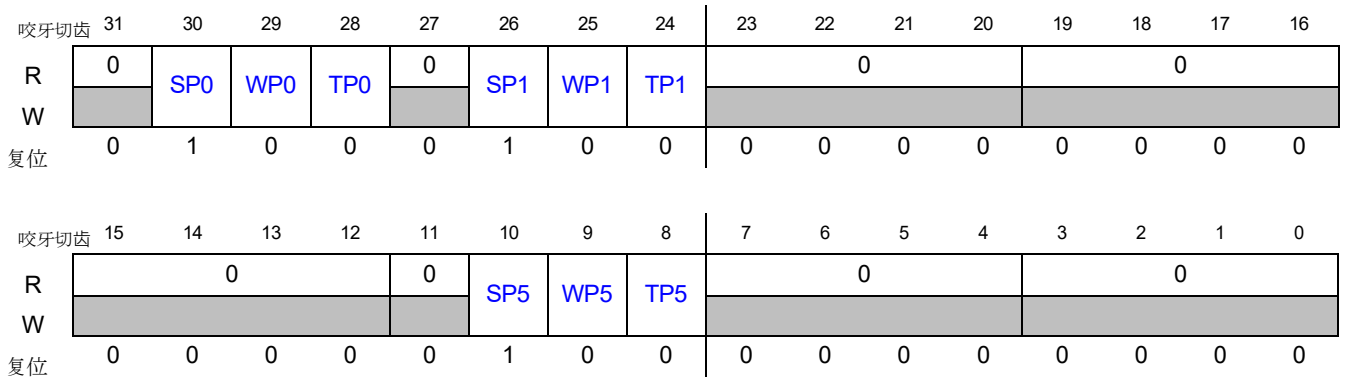
| 注册 | 偏移 |
| --- | --- |
| PACRB | 24h |

### 14.3.1.4.2 职能

每个PACR寄存器由8个4位PACR字段组成。每个PACR字段定义了一个特定平台上外设的访问级别。每个PACR字段的外设分配是由外设的内存映射槽定义的。关于特定外设的字段分配，请参见芯片特定的AIPS信息。

每个没有分配到外设的PACR字段都是保留的。对保留位置的读数返回0，而写数则被忽略。

### 14.3.1.4.3 图示




### 14.3.1.4.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>- | 保留 |
| 30<br>SP0 | 主管保护<br>确定外设是否需要监督员权限等级来进行访问。当这个字段被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制字段必须被设置。如果不这样做，访问就会以错误响应终止，并且没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |
| 29<br>WP0 | 写保护<br>确定外设是否允许写入访问。当这个字段被设置并尝试写访问时，访问以错误响应终止，没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |
| 28<br>TP0 | 值得信赖的保护<br>决定外设是否允许来自不受信任的主控的访问。当这个字段被设置，并且由一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |
| 27<br>- | 保留 |

*表继续在下一页...*

| | |
|---|---|
| 26<br>SP1 | 主管保护<br>确定外设是否需要监督员权限级别的访问。当这个字段被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制字段必须被设置。如果不这样做，访问将以错误响应终止，没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |

| 场地 | 职能 |
|---|---|
| 25<br>WP1 | 写保护<br>确定外设是否允许写入访问。当这个字段被设置并试图进行写访问时，访问将以错误响应终止，并且没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |
| 24<br>TP1 | 值得信赖的保护<br>决定外设是否允许来自不受信任的主控的访问。当这个字段被设置，并且由一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |
| 23-20<br>- | 保留 |
| 19-16<br>- | 保留 |
| 15-12<br>- | 保留 |
| 11<br>- | 保留 |
| 10<br>SP5 | 主管保护<br>确定外设是否需要监督员权限等级来进行访问。当这个字段被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制字段必须被设置。如果不这样做，访问就会以错误响应终止，并且没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |
| 9<br>WP5 | 写保护<br>确定外设是否允许写入访问。当这个字段被设置并试图进行写访问时，访问将以错误响应终止，并且没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |
| 8<br>TP5 | 值得信赖的保护<br>决定外设是否允许来自不受信任的主控的访问。当这个字段被设置，并且由一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |
| 7-4<br>- | 保留 |
| 3-0<br>- | 保留 |

*表继续在下一页...*

## 14.3.1.5 外围访问控制寄存器（PACRD）。

### 14.3.1.5.1 偏移

| 注册 | 偏移 |
|---|---|
| ÃÃÃ | 2Ch |

### 14.3.1.5.2 职能

每个PACR寄存器由8个4位PACR字段组成。每个PACR字段定义了一个特定平台上外设的访问级别。每个PACR字段的外设分配是由外设的内存映射槽定义的。关于特定外设的字段分配，请参见芯片特定的AIPS信息。

每个没有分配到外设的PACR字段都是保留的。对保留位置的读数返回0，而写数则被忽略。

### 14.3.1.5.3 图示

| 咬牙切齿 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | SP0 | WP0 | TP0 | 0 | SP1 | WP1 | TP1 | 0 | | | | 0 | | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 咬牙切齿 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | 0 | | | | 0 | | | | 0 | | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

### 14.3.1.5.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>- | 保留 |

| | |
|---|---|
| 30<br>SP0 | 主管保护<br>确定外设是否需要监督员权限等级来进行访问。当这个字段被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制字段必须被设置。如果不这样做，访问就会以错误响应终止，并且没有外设访问启动。<br>0b - 这个外设的访问不需要监督员的权限级别。 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | 1b - 这个外设需要主管权限级别来访问。 |
| 29<br>WP0 | 写保护<br>确定外设是否允许写入访问。当这个字段被设置并试图进行写访问时，访问将以错误响应终止，并且没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |
| 28<br>TP0 | 值得信赖的保护<br>决定外设是否允许来自不受信任的主控的访问。当这个字段被设置，并且由一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |
| 27<br>- | 保留 |
| 26<br>SP1 | 主管保护<br>确定外设是否需要监督员权限等级来进行访问。当这个字段被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制字段必须被设置。如果不这样做，访问就会以错误响应终止，并且没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |
| 25<br>WP1 | 写保护<br>确定外设是否允许写入访问。当这个字段被设置并试图进行写访问时，访问将以错误响应终止，并且没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |
| 24<br>TP1 | 值得信赖的保护<br>决定外设是否允许来自不受信任的主控的访问。当这个字段被设置，并且由一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |
| 23-20<br>- | 保留 |
| 19-16<br>- | 保留 |
| 15-12<br>- | 保留 |
| 11-8<br>- | 保留 |

| | |
|---|---|
| 7-4<br>-<br> | 保留 |
| 3-0<br>-<br> | 保留 |

*表继续在下一页...*

## 14.3.1.6 非平台外围访问控制寄存器（OPACRA）。

### 14.3.1.6.1 偏移

| 注册 | 偏移 |
|---|---|
| OPACRA | 40h |

### 14.3.1.6.2 职能

每个OPACR寄存器由8个4位OPACR字段组成。每个OPACR字段定义了一个特定的非平台外设的访问级别。每个OPACR字段的外设分配是由外设的内存映射槽定义的。关于特定外设的字段分配，请参见芯片的AIPS信息。

### 14.3.1.6.3 图示

| 咬牙切齿 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | SP0 | WP0 | TP0 | 0 | SP1 | WP1 | TP1 | 0 | | | | 0 | | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 咬牙切齿 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | SP4 | WP4 | TP4 | 0 | SP5 | WP5 | TP5 | 0 | SP6 | WP6 | TP6 | 0 | SP7 | WP7 | TP7 |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 |

### 14.3.1.6.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>- | 保留 |

| | |
|---|---|
| 30<br>SP0 | 主管保护<br>确定外设是否需要监督员权限等级来进行访问。当这个字段被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制字段必须被设置。如果不这样做，访问就会以错误响应终止，并且没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| 29<br>WP0 | 写保护<br>确定外设是否允许写入访问。当这个字段被设置并试图进行写访问时，访问将以错误响应终止，并且没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |
| 28<br>TP0 | 值得信赖的保护<br>决定外设是否允许来自不受信任的主控的访问。当这个位被设置，并且一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站进行访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |
| 27<br>- | 保留 |
| 26<br>SP1 | 主管保护<br>确定外设是否需要监督员权限等级来访问。当这个字段被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制字段必须被设置。如果不这样做，访问就会以一个错误响应终止，并且没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |
| 25<br>WP1 | 写保护<br>确定外设是否允许写入访问。当这个字段被设置并尝试写访问时，访问以错误响应终止，没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |
| 24<br>TP1 | 值得信赖的保护<br>决定外设是否允许来自不受信任的主控的访问。当这个字段被设置，并且由一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |
| 23-20<br>- | 保留 |
| 19-16<br>- | 保留 |
| 15<br>- | 保留 |

| | |
|---|---|
| 14<br>SP4 | 主管保护<br>确定外设是否需要监督员权限级别来访问。当该位被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制位必须被设置。如果没有，访问就会以错误响应终止，并且没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |
| 13<br>WP4 | 写保护<br>确定外设是否允许写入访问。当这个字段被设置并试图进行写访问时，访问将以错误响应终止，并且没有外设访问启动。 |

*表继续在下一页...*

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | 0b - 该外设允许写入访问。1b - 该外设有写保护。 |
| 12<br>TP4 | 值得信赖的保护<br>决定外设是否允许来自不受信任的主控的访问。当这个位被设置，并且一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |
| 11<br>- | 保留 |
| 10<br>SP5 | 主管保护<br>确定外设是否需要监督员权限等级来进行访问。当这个字段被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制字段必须被设置。如果不这样做，访问就会以错误响应终止，并且没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |
| 9<br>WP5 | 写保护<br>确定外设是否允许写入访问。当这个字段被设置并试图进行写访问时，访问将以错误响应终止，并且没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |
| 8<br>TP5 | 值得信赖的保护<br>决定外设是否允许来自不受信任的主控的访问。当这个字段被设置，并且由一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |
| 7<br>- | 保留 |
| 6<br>SP6 | 主管保护<br>确定外设是否需要监督员权限等级来进行访问。当这个字段被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制字段必须被设置。如果不这样做，访问就会以错误响应终止，并且没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |

| | |
|---|---|
| 5<br>WP6 | 写保护<br>确定外设是否允许写入访问。当这个字段被设置并试图进行写访问时，访问将以错误响应终止，并且没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |
| 4<br>TP6 | 值得信赖的保护<br>决定外设是否允许来自不受信任的主控的访问。当这个字段被设置，并且由一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| 3<br>- | 保留 |
| 2<br>SP7 | 主管保护<br>确定外设是否需要监督员权限级别的访问。当这个字段被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制字段必须被设置。如果不这样做，访问将以错误响应终止，没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |
| 1<br>WP7 | 写保护<br>决定外设是否允许来自不受信任的主控的访问。当这个字段被设置，并且由一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |
| 0<br>TP7 | 值得信赖的保护<br>决定外设是否允许来自不受信任的主控的访问。当这个字段被设置，并且由一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |

## 14.3.1.7 非平台外围访问控制寄存器（OPACRB）。

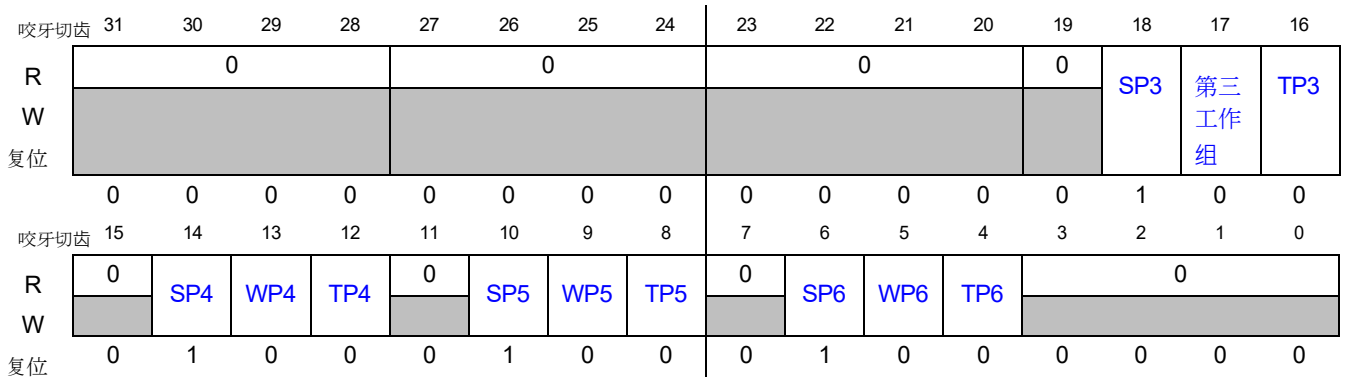
### 14.3.1.7.1 偏移

| 注册 | 偏移 |
|---|---|
| 农村妇女委员会（OPACRB | 44h |

### 14.3.1.7.2 职能

每个OPACR寄存器由8个4位OPACR字段组成。每个OPACR字段定义了一个特定的非平台外设的访问级别。每个OPACR字段的外设分配是由外设的内存映射槽定义的。关于特定外设的字段分配，请参见芯片的AIPS信息。

### 14.3.1.7.3 图示




### 14.3.1.7.4 领域

| 场地 | 职能 |
|---|---|
| 31-28<br>- | 保留 |
| 27-24<br>- | 保留 |
| 23-20<br>- | 保留 |
| 19<br>- | 保留 |
| 18<br>SP3 | 主管保护<br>确定外设是否需要监督员权限等级来进行访问。当这个字段被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制字段必须被设置。如果不这样做，访问就会以错误响应终止，并且没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |
| 17<br>第三工作组 | 写保护<br>确定外设是否允许写入访问。当该位被设置并试图进行写入访问时，访问将以错误响应终止，并且没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |

*表继续在下一页...*

| | |
|---|---|
| 16<br>TP3 | 值得信赖的保护<br>决定外设是否允许来自不受信任的主控的访问。当这个字段被设置，并且由一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |
| 15<br>- | 保留 |
| 14 | 主管保护 |

| 场地 | 职能 |
|---|---|
| SP4 | 确定外设是否需要监督员权限级别来访问。当该位被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制位必须被设置。如果没有，访问就会以错误响应终止，并且没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |
| 13<br>WP4 | 写保护<br>确定外设是否允许写入访问。当这个字段被设置并试图进行写访问时，访问将以错误响应终止，并且没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |
| 12<br>TP4 | 值得信赖的保护<br>决定外设是否允许来自不受信任的主控的访问。当这个位被设置，并且一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |
| 11<br>- | 保留 |
| 10<br>SP5 | 主管保护<br>确定外设是否需要监督员权限等级来进行访问。当这个字段被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制字段必须被设置。如果不这样做，访问就会以错误响应终止，并且没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |
| 9<br>WP5 | 写保护<br>确定外设是否允许写入访问。当这个字段被设置并试图进行写访问时，访问将以错误响应终止，并且没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |
| 8<br>TP5 | 值得信赖的保护<br>决定外设是否允许来自不受信任的主控的访问。当这个字段被设置，并且由一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |
| 7<br>- | 保留 |

*表继续在下一页...*

| | |
|---|---|
| 6<br>SP6 | 主管保护<br>确定外设是否需要监督员权限级别的访问。当这个字段被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制字段必须被设置。如果不这样做，访问将以错误响应终止，没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |
| 5<br>WP6 | 写保护 |

| 场地 | 职能 |
|---|---|
| | 确定外设是否允许写入访问。当这个字段被设置并试图进行写访问时，访问将以错误响应终止，并且没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |
| 4<br>TP6 | 值得信赖的保护<br>决定外设是否允许来自不受信任的主控的访问。当这个字段被设置，并且由一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |
| 3-0<br>- | 保留 |

## 14.3.1.8 非平台外围访问控制寄存器（OPACRC）。

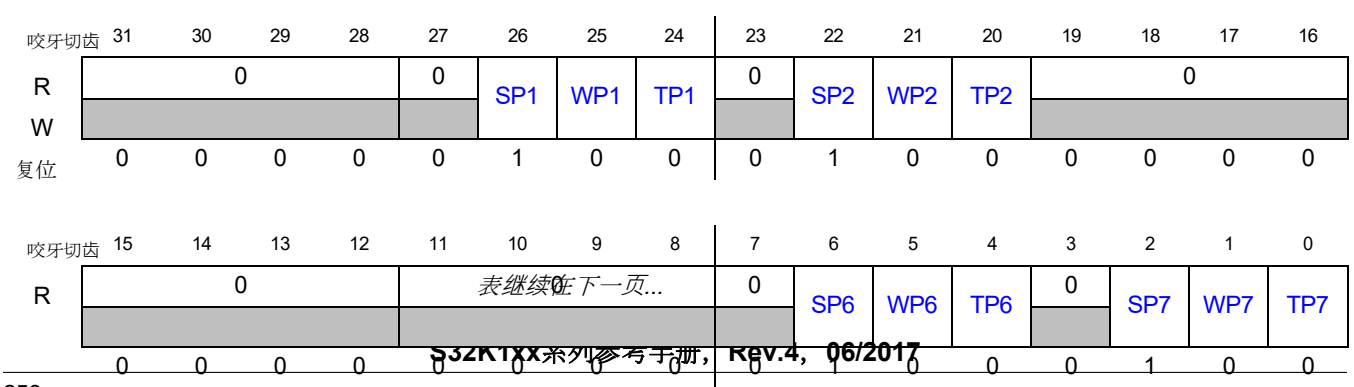
### 14.3.1.8.1 偏移

| 注册 | 偏移 |
|---|---|
| OPACRC | 48h |

### 14.3.1.8.2 职能

每个OPACR寄存器由8个4位OPACR字段组成。每个OPACR字段定义了一个特定的非平台外设的访问级别。每个OPACR字段的外设分配是由外设的内存映射槽定义的。关于特定外设的字段分配，请参见芯片的AIPS信息。

### 14.3.1.8.3 图示




表继续在下一页...

| | |
|---|---|
| W | 复位 |

### 14.3.1.8.4 领域

| 场地 | 职能 |
|---|---|
| 31-28<br>- | 保留 |
| 27<br>- | 保留 |
| 26<br>SP1 | 主管保护<br>确定外设是否需要监督员权限级别来访问。当这个字段被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制字段必须被设置。如果不这样做，访问就会以一个错误响应终止，并且没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |
| 25<br>WP1 | 写保护<br>确定外设是否允许写入访问。当这个字段被设置并试图进行写访问时，访问将以错误响应终止，并且没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |
| 24<br>TP1 | 值得信赖的保护<br>决定外设是否允许来自不受信任的主控的访问。当这个字段被设置，并且由一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |
| 23<br>- | 保留 |
| 22<br>SP2 | 主管保护<br>确定外设是否需要监督员权限级别来访问。当该位被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制位必须被设置。如果没有，访问就会以错误响应终止，并且没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |
| 21<br>WP2 | 写保护<br>确定外设是否允许写入访问。当这个字段被设置并试图进行写访问时，访问将以错误响应终止，并且没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |

*表继续在下一页...*

| | |
|---|---|
| 20<br>TP2 | 值得信赖的保护<br>决定外设是否允许来自不受信任的主控的访问。当这个位被设置，并且一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |
| 19-16<br>- | 保留 |

| 场地 | 职能 |
|---|---|
| 15-12<br>- | 保留 |
| 11-8<br>- | 保留 |
| 7<br>- | 保留 |
| 6<br>SP6 | 主管保护<br>确定外设是否需要监督员权限等级来进行访问。当这个字段被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制字段必须被设置。如果不这样做，访问就会以错误响应终止，并且没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |
| 5<br>WP6 | 写保护<br>确定外设是否允许写入访问。当这个字段被设置并试图进行写访问时，访问将以错误响应终止，并且没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |
| 4<br>TP6 | 值得信赖的保护<br>决定外设是否允许来自不受信任的主控的访问。当这个字段被设置，并且由一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |
| 3<br>- | 保留 |
| 2<br>SP7 | 主管保护<br>确定外设是否需要监督员权限等级来进行访问。当这个字段被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制字段必须被设置。如果不这样做，访问就会以错误响应终止，并且没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |
| 1<br>WP7 | 写保护<br>决定外设是否允许来自不受信任的主控的访问。当这个字段被设置，并且由一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |

*表继续在下一页...*

| | |
|---|---|
| 0<br>TP7 | 值得信赖的保护<br>决定外设是否允许来自不受信任的主控的访问。当这个字段被设置，并且由一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |

## 14.3.1.9 非平台外围访问控制寄存器（OPACRD）。

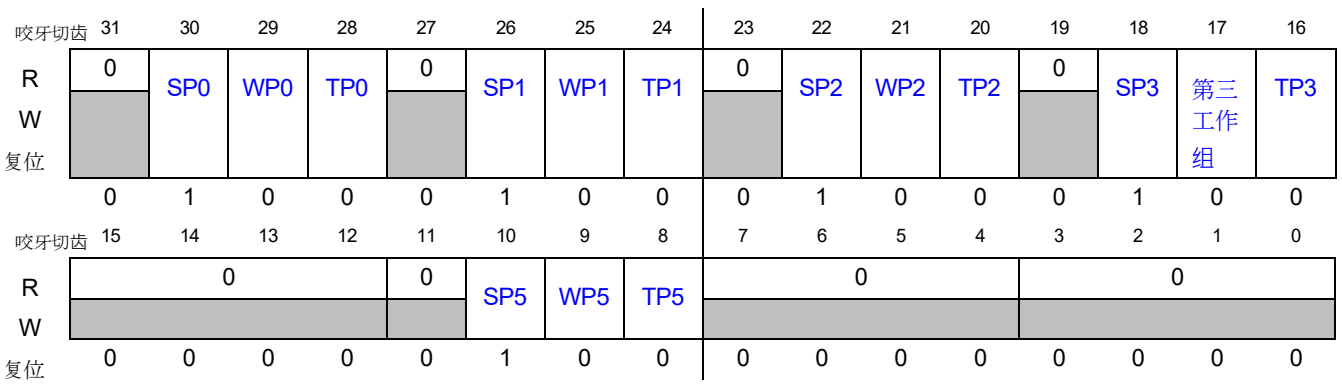
### 14.3.1.9.1 偏移

| 注册 | 偏移 |
|---|---|
| OPACRD | 4Ch |

### 14.3.1.9.2 职能

每个OPACR寄存器由8个4位OPACR字段组成。每个OPACR字段定义了一个特定的非平台外设的访问级别。每个OPACR字段的外设分配是由外设的内存映射槽定义的。关于特定外设的字段分配，请参见芯片的AIPS信息。

### 14.3.1.9.3 图示




### 14.3.1.9.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>- | 保留 |

*表继续在下一页...*

| | |
|---|---|
| 30<br>SP0 | 主管保护<br>确定外设是否需要监督员权限等级来进行访问。当这个字段被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制字段必须被设置。如果不这样做，访问将以错误响应终止，没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |

| 场地 | 职能 |
|---|---|
| 29<br>WP0 | 写保护<br>确定外设是否允许写入访问。当这个字段被设置并试图进行写访问时，访问将以错误响应终止，并且没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |
| 28<br>TP0 | 值得信赖的保护<br>决定外设是否允许来自不受信任的主控的访问。当这个位被设置，并且一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |
| 27<br>- | 保留 |
| 26<br>SP1 | 主管保护<br>确定外设是否需要监督员权限级别来访问。当这个字段被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制字段必须被设置。如果不这样做，访问就会以一个错误响应终止，并且没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |
| 25<br>WP1 | 写保护<br>确定外设是否允许写入访问。当这个字段被设置并试图进行写访问时，访问将以错误响应终止，并且没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |
| 24<br>TP1 | 值得信赖的保护<br>决定外设是否允许来自不受信任的主控的访问。当这个字段被设置，并且由一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |
| 23<br>- | 保留 |
| 22<br>SP2 | 主管保护<br>确定外设是否需要监督员权限级别来访问。当该位被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制位必须被设置。如果没有，访问就会以错误响应终止，并且没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |

*表继续在下一页...*

| | |
|---|---|
| 21<br>WP2 | 写保护<br>确定外设是否允许写入访问。当这个字段被设置并试图进行写访问时，访问将以错误响应终止，并且没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |
| 20<br>TP2 | 值得信赖的保护 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | 决定外设是否允许来自不受信任的主控的访问。当这个位被设置，并且一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |
| 19<br>- | 保留 |
| 18<br>SP3 | 主管保护<br>确定外设是否需要监督员权限等级来进行访问。当这个字段被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制字段必须被设置。如果不这样做，访问就会以错误响应终止，并且没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |
| 17<br>第三工作组 | 写保护<br>确定外设是否允许写入访问。当该位被设置并试图进行写入访问时，访问将以错误响应终止，并且没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |
| 16<br>TP3 | 值得信赖的保护<br>决定外设是否允许来自不受信任的主控的访问。当这个字段被设置，并且由一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |
| 15-12<br>- | 保留 |
| 11<br>- | 保留 |
| 10<br>SP5 | 主管保护<br>确定外设是否需要监督员权限等级来进行访问。当这个字段被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制字段必须被设置。如果不这样做，访问就会以错误响应终止，并且没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |

*表继续在下一页...*

| | |
|---|---|
| 9<br>WP5 | 写保护<br>确定外设是否允许写入访问。当这个字段被设置并试图进行写访问时，访问将以错误响应终止，并且没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |
| 8<br>TP5 | 值得信赖的保护<br>决定外设是否允许来自不受信任的主控的访问。当这个字段被设置，并且由一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |
| 7-4 | 保留 |

| 场地 | 职能 |
|---|---|
| - | |
| 3-0<br>- | 保留 |

## 14.3.1.10 非平台外围访问控制寄存器（OPACRE）。

### 14.3.1.10.1 偏移

| 注册 | 偏移 |
|---|---|
| OPACRE | 50h |

### 14.3.1.10.2 职能

每个OPACR寄存器由8个4位OPACR字段组成。每个OPACR字段定义了一个特定的非平台外设的访问级别。每个OPACR字段的外设分配是由外设的内存映射槽定义的。关于特定外设的字段分配，请参见芯片的AIPS信息。

### 14.3.1.10.3 图示

| 咬牙切齿 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | SP0 | WP0 | TP0 | 0 | | | | 0 | | | | 0 | | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 咬牙切齿 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | 0 | | | | 0 | SP6 | WP6 | TP6 | 0 | | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

### 14.3.1.10.4 领域

| 场地 | 职能 |
|---|---|

*表继续在下一页...*

| | |
|---|---|
| 31<br>-<br> | 保留 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| 30<br>SP0 | 主管保护<br>确定外设是否需要监督员权限等级来进行访问。当这个字段被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制字段必须被设置。如果不这样做，访问就会以错误响应终止，并且没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |
| 29<br>WP0 | 写保护<br>确定外设是否允许写入访问。当这个字段被设置并试图进行写访问时，访问将以错误响应终止，并且没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |
| 28<br>TP0 | 值得信赖的保护<br>决定外设是否允许来自不受信任的主控的访问。当这个位被设置，并且一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |
| 27-24<br>- | 保留 |
| 23-20<br>- | 保留 |
| 19-16<br>- | 保留 |
| 15-12<br>- | 保留 |
| 11-8<br>- | 保留 |
| 7<br>- | 保留 |
| 6<br>SP6 | 主管保护<br>确定外设是否需要监督员权限级别的访问。当这个字段被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制字段必须被设置。如果不这样做，访问将以错误响应终止，没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |

*表继续在下一页...*

| | |
|---|---|
| 5<br>WP6 | 写保护<br>确定外设是否允许写入访问。当这个字段被设置并试图进行写访问时，访问将以错误响应终止，并且没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |
| 4<br>TP6 | 值得信赖的保护<br>决定外设是否允许来自不受信任的主控的访问。当这个字段被设置，并且由一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。 |

| 场地 | 职能 |
|---|---|
| | 0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |
| 3-0<br>- | 保留 |

## 14.3.1.11 非平台外围访问控制寄存器（OPACRF）。

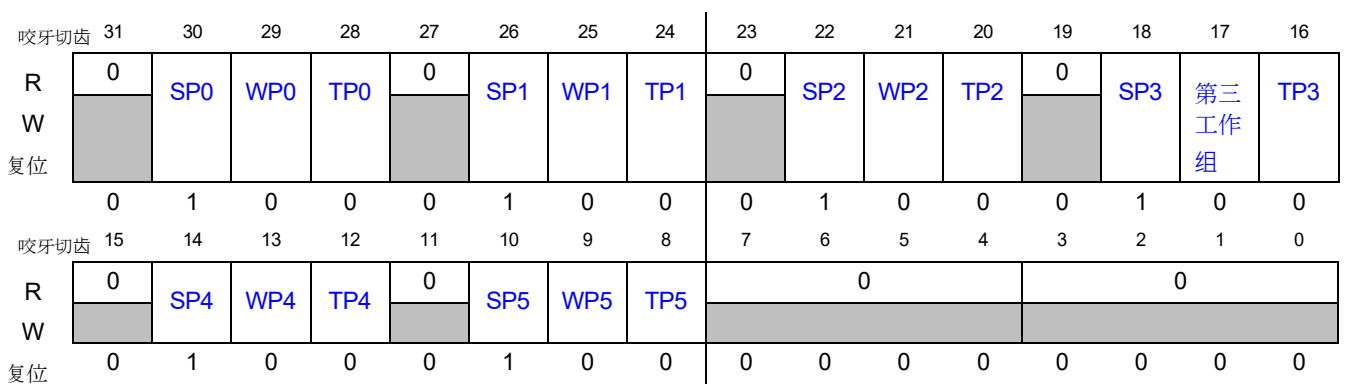
### 14.3.1.11.1 偏移

| 注册 | 偏移 |
|---|---|
| OPACRF | 54h |

### 14.3.1.11.2 职能

每个OPACR寄存器由8个4位OPACR字段组成。每个OPACR字段定义了一个特定的非平台外设的访问级别。每个OPACR字段的外设分配是由外设的内存映射槽定义的。关于特定外设的字段分配，请参见芯片的AIPS信息。

### 14.3.1.11.3 图示




### 14.3.1.11.4 领域

| 场地 | *表继续在下一页...* 职能 |
|---|---|

| | |
|---|---|
| 31 | 保留 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| - | |
| 30<br>SP0 | 主管保护<br>确定外设是否需要监督员权限等级来进行访问。当这个字段被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制字段必须被设置。如果不这样做，访问就会以错误响应终止，并且没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |
| 29<br>WP0 | 写保护<br>确定外设是否允许写入访问。当这个字段被设置并试图进行写访问时，访问将以错误响应终止，并且没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |
| 28<br>TP0 | 值得信赖的保护<br>决定外设是否允许来自不受信任的主控的访问。当这个位被设置，并且一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |
| 27<br>- | 保留 |
| 26<br>SP1 | 主管保护<br>确定外设是否需要监督员权限级别来访问。当这个字段被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制字段必须被设置。如果不这样做，访问就会以一个错误响应终止，并且没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |
| 25<br>WP1 | 写保护<br>确定外设是否允许写入访问。当这个字段被设置并试图进行写访问时，访问将以错误响应终止，并且没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |
| 24<br>TP1 | 值得信赖的保护<br>决定外设是否允许来自不受信任的主控的访问。当这个字段被设置，并且由一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |

*表继续在下一页...*

| | |
|---|---|
| 23<br>- | 保留 |
| 22<br>SP2 | 主管保护<br>确定外设是否需要监督员权限级别来访问。当该位被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制位必须被设置。如果没有，访问将以错误响应终止，并且没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |

| 场地 | 职能 |
| --- | --- |
| 21<br>WP2 | 写保护<br>确定外设是否允许写入访问。当这个字段被设置并试图进行写访问时，访问将以错误响应终止，并且没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |
| 20<br>TP2 | 值得信赖的保护<br>决定外设是否允许来自不受信任的主控的访问。当这个位被设置，并且一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |
| 19<br>- | 保留 |
| 18<br>SP3 | 主管保护<br>确定外设是否需要监督员权限等级来进行访问。当这个字段被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制字段必须被设置。如果不这样做，访问就会以错误响应终止，并且没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |
| 17<br>第三工作组 | 写保护<br>确定外设是否允许写入访问。当该位被设置并试图进行写入访问时，访问将以错误响应终止，并且没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |
| 16<br>TP3 | 值得信赖的保护<br>决定外设是否允许来自不受信任的主控的访问。当这个字段被设置，并且由一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |
| 15<br>- | 保留 |
| 14<br>SP4 | 主管保护<br>确定外设是否需要监督员权限级别来访问。当该位被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制位必须被设置。如果没有，访问就会以错误响应终止，并且没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |

*表继续在下一页...*

| | |
|---|---|
| 13<br>WP4 | 写保护<br>确定外设是否允许写入访问。当这个字段被设置并试图进行写访问时，访问将以错误响应终止，并且没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |
| 12<br>TP4 | 值得信赖的保护 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| | 决定外设是否允许来自不受信任的主控的访问。当这个位被设置，并且一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |
| 11<br>- | 保留 |
| 10<br>SP5 | 主管保护<br>确定外设是否需要监督员权限等级来进行访问。当这个字段被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制字段必须被设置。如果不这样做，访问就会以错误响应终止，并且没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |
| 9<br>WP5 | 写保护<br>确定外设是否允许写入访问。当这个字段被设置并试图进行写访问时，访问将以错误响应终止，并且没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |
| 8<br>TP5 | 值得信赖的保护<br>决定外设是否允许来自不受信任的主控的访问。当这个字段被设置，并且由一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |
| 7-4<br>- | 保留 |
| 3-0<br>- | 保留 |

## 14.3.1.12 非平台外围访问控制寄存器（OPACRG）。

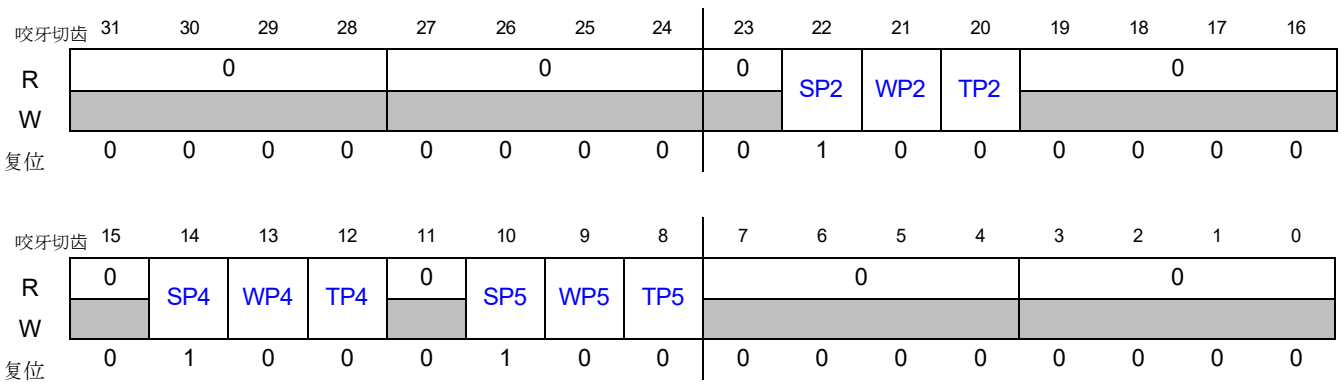
### 14.3.1.12.1 偏移

| 注册 | 偏移 |
|---|---|
| OPACRG | 58h |

### 14.3.1.12.2 职能

每个OPACR寄存器由8个4位OPACR字段组成。每个OPACR字段定义了一个特定的非平台外设的访问级别。每个OPACR字段的外设分配是由外设的内存映射槽定义的。关于特定外设的字段分配，请参见芯片的AIPS信息。

### 14.3.1.12.3 图示




### 14.3.1.12.4 领域

| 场地 | 职能 |
|---|---|
| 31-28<br>- | 保留 |
| 27-24<br>- | 保留 |
| 23<br>- | 保留 |
| 22<br>SP2 | 主管保护<br>确定外设是否需要监督员权限级别来访问。当该位被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制位必须被设置。如果没有，访问就会以错误响应终止，并且没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |
| 21<br>WP2 | 写保护<br>确定外设是否允许写入访问。当这个字段被设置并试图进行写访问时，访问将以错误响应终止，并且没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |

| | |
|---|---|
| 20<br>TP2 | 值得信赖的保护 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | 决定外设是否允许来自不受信任的主控的访问。当这个位被设置，并且一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |
| 19-16<br>- | 保留 |
| 15<br>- | 保留 |
| 14<br>SP4 | 主管保护<br>确定外设是否需要监督员权限级别来访问。当该位被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制位必须被设置。如果没有，访问就会以错误响应终止，并且没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |
| 13<br>WP4 | 写保护<br>确定外设是否允许写入访问。当这个字段被设置并试图进行写访问时，访问将以错误响应终止，并且没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |
| 12<br>TP4 | 值得信赖的保护<br>决定外设是否允许来自不受信任的主控的访问。当这个位被设置，并且一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |
| 11<br>- | 保留 |
| 10<br>SP5 | 主管保护<br>确定外设是否需要监督员权限等级来进行访问。当这个字段被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制字段必须被设置。如果不这样做，访问就会以错误响应终止，并且没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |
| 9<br>WP5 | 写保护<br>确定外设是否允许写入访问。当这个字段被设置并试图进行写访问时，访问将以错误响应终止，并且没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |

| | |
|---|---|
| 8<br>TP5 | 值得信赖的保护<br>决定外设是否允许来自不受信任的主控的访问。当这个字段被设置，并且由一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |
| 7-4 | 保留 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| - | |
| 3-0<br>- | 保留 |

## 14.3.1.13 非平台外围访问控制寄存器（OPACRH）。

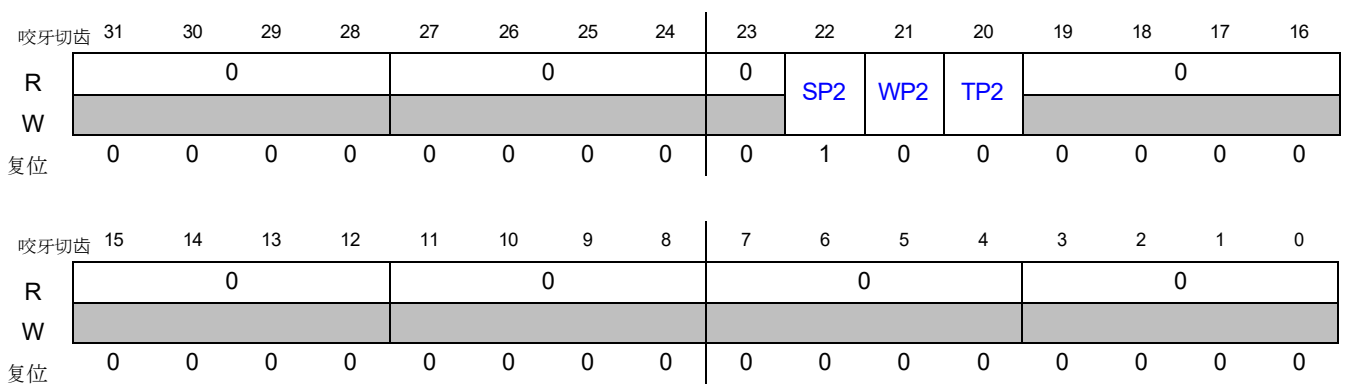
### 14.3.1.13.1 偏移

| 注册 | 偏移 |
|---|---|
| OPACRH | 5Ch |

### 14.3.1.13.2 职能

每个OPACR寄存器由8个4位OPACR字段组成。每个OPACR字段定义了一个特定的非平台外设的访问级别。每个OPACR字段的外设分配是由外设的内存映射槽定义的。关于特定外设的字段分配，请参见芯片的AIPS信息。

### 14.3.1.13.3 图示




### 14.3.1.13.4 领域

| 场地 | 职能 |
|---|---|

| | |
|---|---|
| 31-28<br>- | 保留 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| 27-24<br>- | 保留 |
| 23<br>- | 保留 |
| 22<br>SP2 | 主管保护<br>确定外设是否需要监督员权限级别来访问。当该位被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制位必须被设置。如果没有，访问将以错误响应终止，并且没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |
| 21<br>WP2 | 写保护<br>确定外设是否允许写入访问。当这个字段被设置并试图进行写访问时，访问将以错误响应终止，并且没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |
| 20<br>TP2 | 值得信赖的保护<br>决定外设是否允许来自不受信任的主控的访问。当这个位被设置，并且一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |
| 19-16<br>- | 保留 |
| 15-12<br>- | 保留 |
| 11-8<br>- | 保留 |
| 7-4<br>- | 保留 |
| 3-0<br>- | 保留 |

## 14.3.1.14 非平台外围访问控制寄存器（OPACRI）。

### 14.3.1.14.1 偏移

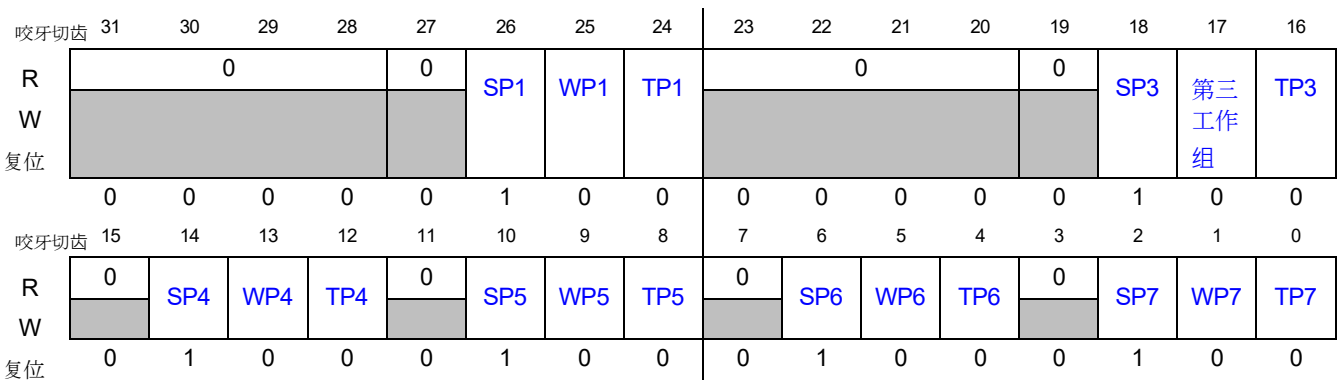
| 注册 | 偏移 |
| --- | --- |

| | |
|---|---|
| 侨联 | 60h |

### 14.3.1.14.2 职能

每个OPACR寄存器由8个4位OPACR字段组成。每个OPACR字段定义了一个特定的非平台外设的访问级别。每个OPACR字段的外设分配是由外设的内存映射槽定义的。关于特定外设的字段分配，请参见芯片的AIPS信息。

### 14.3.1.14.3 图示




### 14.3.1.14.4 领域

| 场地 | 职能 |
|---|---|
| 31-28<br>- | 保留 |
| 27<br>- | 保留 |
| 26<br>SP1 | 主管保护<br>确定外设是否需要监督员权限级别来访问。当这个字段被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制字段必须被设置。如果不这样做，访问就会以一个错误响应终止，并且没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |
| 25<br>WP1 | 写保护<br>确定外设是否允许写入访问。当这个字段被设置并试图进行写访问时，访问将以错误响应终止，并且没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |

| | |
|---|---|
| 24<br>TP1 | 值得信赖的保护<br>决定外设是否允许来自不受信任的主控的访问。当这个字段被设置，并且由一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | 0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |
| 23-20<br>- | 保留 |
| 19<br>- | 保留 |
| 18<br>SP3 | 主管保护<br>确定外设是否需要监督员权限级别的访问。当这个字段被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制字段必须被设置。如果不这样做，访问将以错误响应终止，没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |
| 17<br>第三工作组 | 写保护<br>确定外设是否允许写入访问。当该位被设置并试图进行写入访问时，访问将以错误响应终止，并且没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |
| 16<br>TP3 | 值得信赖的保护<br>决定外设是否允许来自不受信任的主控的访问。当这个字段被设置，并且由一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |
| 15<br>- | 保留 |
| 14<br>SP4 | 主管保护<br>确定外设是否需要监督员权限级别来访问。当该位被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制位必须被设置。如果没有，访问就会以错误响应终止，并且没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |
| 13<br>WP4 | 写保护<br>确定外设是否允许写入访问。当这个字段被设置并试图进行写访问时，访问将以错误响应终止，并且没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |

| | |
|---|---|
| 12<br>TP4 | 值得信赖的保护<br>决定外设是否允许来自不受信任的主控的访问。当这个位被设置，并且一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |
| 11<br>- | 保留 |
| 10 | 主管保护 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| SP5 | 确定外设是否需要监督员权限等级来进行访问。当这个字段被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制字段必须被设置。如果不这样做，访问就会以错误响应终止，并且没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |
| 9<br>WP5 | 写保护<br>确定外设是否允许写入访问。当这个字段被设置并试图进行写访问时，访问将以错误响应终止，并且没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |
| 8<br>TP5 | 值得信赖的保护<br>决定外设是否允许来自不受信任的主控的访问。当这个字段被设置，并且由一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |
| 7<br>- | 保留 |
| 6<br>SP6 | 主管保护<br>确定外设是否需要监督员权限等级来进行访问。当这个字段被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制字段必须被设置。如果不这样做，访问就会以错误响应终止，并且没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |
| 5<br>WP6 | 写保护<br>确定外设是否允许写入访问。当这个字段被设置并且尝试写访问时，访问会以一个错误响应终止，并且没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |
| 4<br>TP6 | 值得信赖的保护<br>决定外设是否允许来自不受信任的主控的访问。当这个字段被设置，并且由一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |
| 3<br>- | 保留 |

*表继续在下一页...*

| | |
|---|---|
| 2<br>SP7 | 主管保护<br>确定外设是否需要监督员权限级别的访问。当这个字段被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制字段必须被设置。如果不这样做，访问将以错误响应终止，没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |
| 1<br>WP7 | 写保护 |

| 场地 | 职能 |
|---|---|
| | 决定外设是否允许来自不受信任的主控的访问。当这个字段被设置，并且由一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |
| 0<br>TP7 | 值得信赖的保护<br>决定外设是否允许来自不受信任的主控的访问。当这个字段被设置，并且由一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |

## 14.3.1.15 非平台外围访问控制寄存器（OPACRJ）。

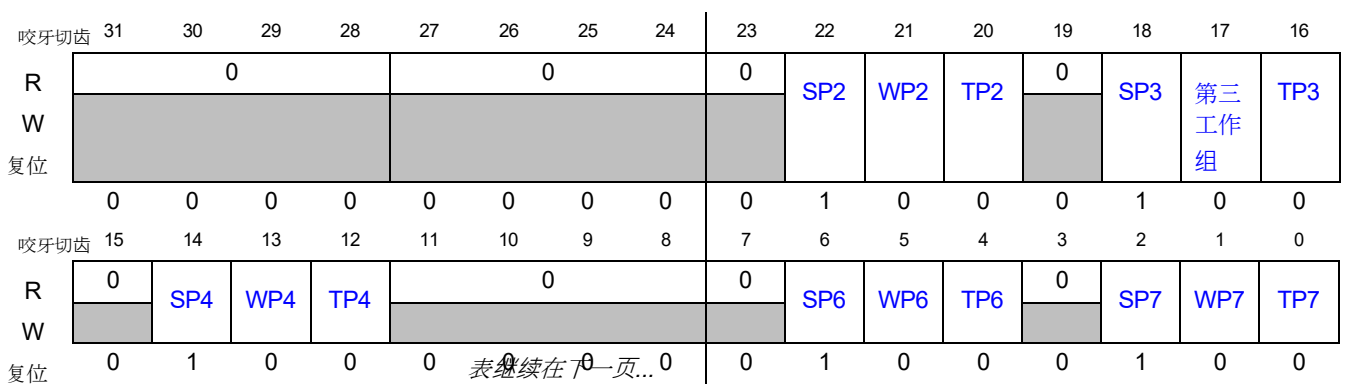
### 14.3.1.15.1 偏移

| 注册 | 偏移 |
|---|---|
| OPACRJ | 64h |

### 14.3.1.15.2 职能

每个OPACR寄存器由8个4位OPACR字段组成。每个OPACR字段定义了一个特定的非平台外设的访问级别。每个OPACR字段的外设分配是由外设的内存映射槽定义的。关于特定外设的字段分配，请参见芯片的AIPS信息。

### 14.3.1.15.3 图示



### 14.3.1.15.4 领域

| 场地 | 职能 |
|---|---|
| 31-28<br>- | 保留 |
| 27-24<br>- | 保留 |
| 23<br>- | 保留 |
| 22<br>SP2 | 主管保护<br>确定外设是否需要监督员权限级别来访问。当该位被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制位必须被设置。如果没有，访问就会以错误响应终止，并且没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |
| 21<br>WP2 | 写保护<br>确定外设是否允许写入访问。当这个字段被设置并试图进行写访问时，访问将以错误响应终止，并且没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |
| 20<br>TP2 | 值得信赖的保护<br>决定外设是否允许来自不受信任的主控的访问。当这个位被设置，并且一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |
| 19<br>- | 保留 |
| 18<br>SP3 | 主管保护<br>确定外设是否需要监督员权限级别的访问。当这个字段被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制字段必须被设置。如果不这样做，访问将以错误响应终止，没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |
| 17<br>第三工作组 | 写保护<br>确定外设是否允许写入访问。当该位被设置并试图进行写入访问时，访问将以错误响应终止，并且没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写 |

| | |
|---|---|
| | 保护。 |
| 16<br>TP3 | 值得信赖的保护<br>决定外设是否允许来自不受信任的主控的访问。当这个字段被设置，并且由一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |
| 15 | 保留 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| - | |
| 14<br>SP4 | 主管保护<br>确定外设是否需要监督员权限级别来访问。当该位被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制位必须被设置。如果没有，访问就会以错误响应终止，并且没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |
| 13<br>WP4 | 写保护<br>确定外设是否允许写入访问。当这个字段被设置并试图进行写访问时，访问将以错误响应终止，并且没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |
| 12<br>TP4 | 值得信赖的保护<br>决定外设是否允许来自不受信任的主控的访问。当这个位被设置，并且一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |
| 11-8<br>- | 保留 |
| 7<br>- | 保留 |
| 6<br>SP6 | 主管保护<br>确定外设是否需要监督员权限等级来进行访问。当这个字段被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制字段必须被设置。如果不这样做，访问就会以错误响应终止，并且没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |
| 5<br>WP6 | 写保护<br>确定外设是否允许写入访问。当这个字段被设置并尝试写访问时，访问以错误响应终止，没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |
| 4<br>TP6 | 值得信赖的保护<br>决定外设是否允许来自不受信任的主控的访问。当这个字段被设置，并且由一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |

| | |
|---|---|
| 3<br>- | 保留 |
| 2<br>SP7 | 主管保护 |

*表继续在下一页...*

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | 确定外设是否需要监督员权限等级来进行访问。当这个字段被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制字段必须被设置。如果不这样做，访问将以错误响应终止，没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |
| 1<br>WP7 | 写保护<br>决定外设是否允许来自不受信任的主控的访问。当这个字段被设置，并且由一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |
| 0<br>TP7 | 值得信赖的保护<br>决定外设是否允许来自不受信任的主控的访问。当这个字段被设置，并且由一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |

## 14.3.1.16 非平台外围访问控制寄存器（OPACRK）。

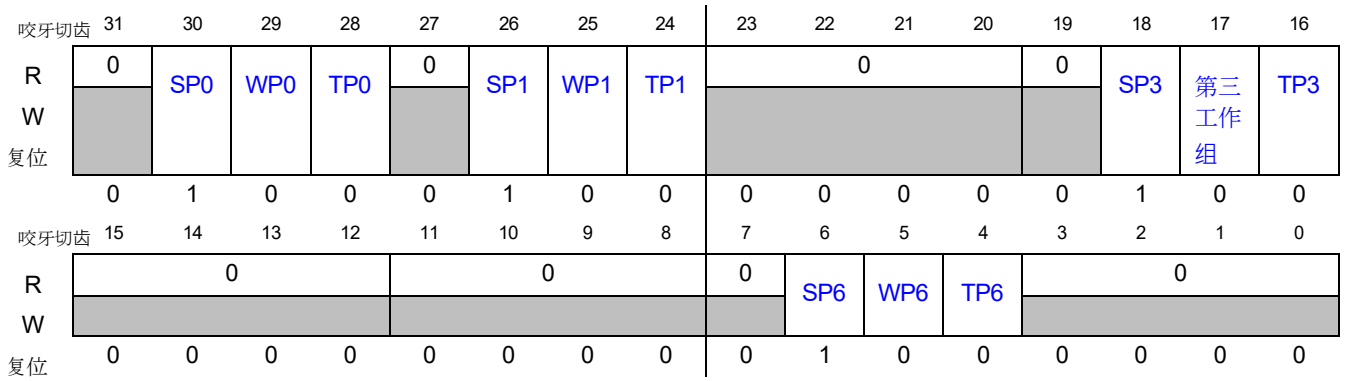
### 14.3.1.16.1 偏移

| 注册 | 偏移 |
| --- | --- |
| 呼叫中心（OPACRK | 68h |

### 14.3.1.16.2 职能

每个OPACR寄存器由8个4位OPACR字段组成。每个OPACR字段定义了一个特定的非平台外设的访问级别。每个OPACR字段的外设分配是由外设的内存映射槽定义的。关于特定外设的字段分配，请参见芯片的AIPS信息。

## 14.3.1.16.3 图示




## 14.3.1.16.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>- | 保留 |
| 30<br>SP0 | 主管保护<br>确定外设是否需要监督员权限等级来进行访问。当这个字段被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制字段必须被设置。如果不这样做，访问就会以错误响应终止，并且没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |
| 29<br>WP0 | 写保护<br>确定外设是否允许写入访问。当这个字段被设置并试图进行写访问时，访问将以错误响应终止，并且没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |
| 28<br>TP0 | 值得信赖的保护<br>决定外设是否允许来自不受信任的主控的访问。当这个位被设置，并且一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |
| 27<br>- | 保留 |

| | |
|---|---|
| 26<br>SP1 | 主管保护<br>确定外设是否需要监督员权限级别来访问。当这个字段被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制字段必须被设置。如果不这样做，访问就会以一个错误响应终止，并且没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |
| 25 | 写保护 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| WP1 | 确定外设是否允许写入访问。当这个字段被设置并试图进行写访问时，访问将以错误响应终止，并且没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |
| 24<br>TP1 | 值得信赖的保护<br>决定外设是否允许来自不受信任的主控的访问。当这个字段被设置，并且由一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |
| 23-20<br>- | 保留 |
| 19<br>- | 保留 |
| 18<br>SP3 | 主管保护<br>确定外设是否需要监督员权限等级来进行访问。当这个字段被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制字段必须被设置。如果不这样做，访问就会以错误响应终止，并且没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |
| 17<br>第三工作组 | 写保护<br>确定外设是否允许写入访问。当该位被设置并试图进行写入访问时，访问将以错误响应终止，并且没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |
| 16<br>TP3 | 值得信赖的保护<br>决定外设是否允许来自不受信任的主控的访问。当这个字段被设置，并且由一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |
| 15-12<br>- | 保留 |
| 11-8<br>- | 保留 |
| 7<br>- | 保留 |

| | |
|---|---|
| 6<br>SP6 | 主管保护<br>确定外设是否需要监督员权限级别的访问。当这个字段被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制字段必须被设置。如果不这样做，访问将以错误响应终止，没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |
| 5 | 写保护 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| WP6 | 确定外设是否允许写入访问。当这个字段被设置并尝试写访问时，访问以错误响应终止，没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |
| 4<br>TP6 | 值得信赖的保护<br>决定外设是否允许来自不受信任的主控的访问。当这个字段被设置，并且由一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |
| 3-0<br>- | 保留 |

## 14.3.1.17 非平台外围访问控制寄存器（OPACRL）。

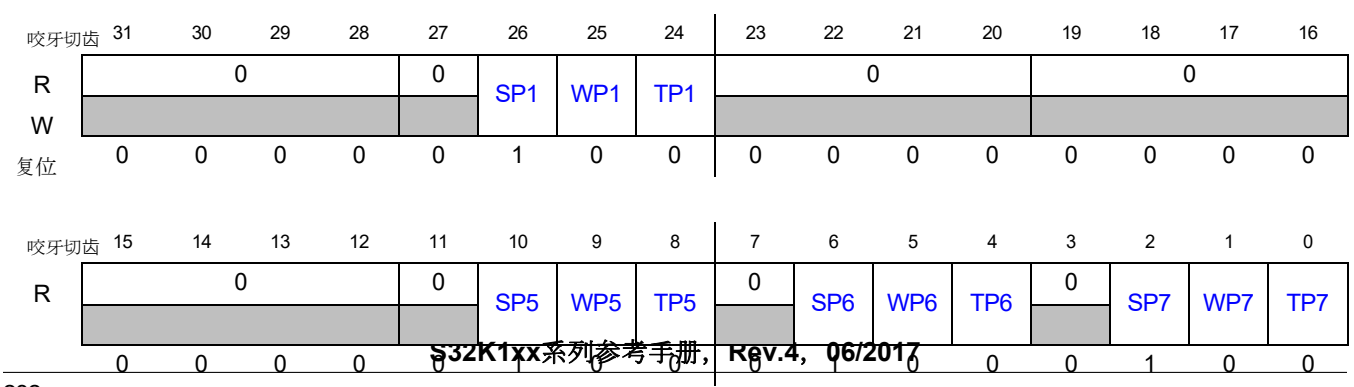
### 14.3.1.17.1 偏移

| 注册 | 偏移 |
|---|---|
| OPACRL | 6Ch |

### 14.3.1.17.2 职能

每个OPACR寄存器由8个4位OPACR字段组成。每个OPACR字段定义了一个特定的非平台外设的访问级别。每个OPACR字段的外设分配是由外设的内存映射槽定义的。关于特定外设的字段分配，请参见芯片的AIPS信息。

### 14.3.1.17.3 图示



| | |
|---|---|
| W | 复位 |

## 14.3.1.17.4 领域

| 场地 | 职能 |
|---|---|
| 31-28<br>- | 保留 |
| 27<br>- | 保留 |
| 26<br>SP1 | 主管保护<br>确定外设是否需要监督员权限级别来访问。当这个字段被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制字段必须被设置。如果不这样做，访问就会以一个错误响应终止，并且没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |
| 25<br>WP1 | 写保护<br>确定外设是否允许写入访问。当这个字段被设置并试图进行写访问时，访问将以错误响应终止，并且没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |
| 24<br>TP1 | 值得信赖的保护<br>决定外设是否允许来自不受信任的主控的访问。当这个字段被设置，并且由一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |
| 23-20<br>- | 保留 |
| 19-16<br>- | 保留 |
| 15-12<br>- | 保留 |
| 11<br>- | 保留 |
| 10<br>SP5 | 主管保护<br>确定外设是否需要监督员权限等级来进行访问。当这个字段被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制字段必须被设置。如果不这样做，访问就会以错误响应终止，并且没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |

| | |
|---|---|
| 9<br>WP5 | 写保护<br>确定外设是否允许写入访问。当这个字段被设置并试图进行写访问时，访问将以错误响应终止，并且没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| 8<br>TP5 | 值得信赖的保护<br>决定外设是否允许来自不受信任的主控的访问。当这个字段被设置，并且由一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |
| 7<br>- | 保留 |
| 6<br>SP6 | 主管保护<br>确定外设是否需要监督员权限等级来进行访问。当这个字段被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制字段必须被设置。如果不这样做，访问就会以错误响应终止，并且没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |
| 5<br>WP6 | 写保护<br>确定外设是否允许写入访问。当这个字段被设置并且尝试写访问时，访问会以一个错误响应终止，并且没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |
| 4<br>TP6 | 值得信赖的保护<br>决定外设是否允许来自不受信任的主控的访问。当这个字段被设置，并且由一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |
| 3<br>- | 保留 |
| 2<br>SP7 | 主管保护<br>确定外设是否需要监督员权限等级来进行访问。当这个字段被设置时，主控权限级别必须指示主管访问属性，并且主控的MPRx[MPLn]控制字段必须被设置。如果不这样做，访问就会以错误响应终止，并且没有外设访问启动。<br>0b - 这个外设的访问不需要主管权限级别。1b - 该外设需要主管权限级别来访问。 |
| 1<br>WP7 | 写保护<br>决定外设是否允许来自不受信任的主控的访问。当这个字段被设置，并且由一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 该外设允许写入访问。1b - 该外设有写保护。 |

| | |
|---|---|
| 0<br>TP7 | 值得信赖的保护<br>决定外设是否允许来自不受信任的主控的访问。当这个字段被设置，并且由一个不受信任的主站尝试访问时，访问将以一个错误响应终止，并且没有外设访问启动。<br>0b - 允许从一个不受信任的主站访问。<br>1b - 不允许从一个不受信任的主站进行访问。 |

## 14.4 功能 描述

外设桥的功能是在横梁开关和从属外设总线之间进行总线协议转换。

外设桥接器管理所有用于附加从属设备的事务，并通过解码附加地址空间内的访问，为外设总线上的模块生成选择信号。

### 14.4.1 访问 支持

对于32位外设，支持对齐和错位的32位、16位和字节访问。支持错位访问，以允许内存被放在从属外设总线上。外设寄存器不得错位，尽管外设桥没有进行明确的检查。所有的访问都是通过一次传输完成的。

对外设插槽的所有访问的尺寸必须小于或等于指定的外设插槽尺寸。如果试图进行大于目标端口的访问，将产生一个错误响应。

# 章 15
# 直接内存访问复用器 （DMAMUX）。

## 15.1 特定芯片的DMAMUX 信息

### 15.1.1 通道的数量

S32K1xx系列的通道数量在不同的变体中有所不同。见下表。

**表15-1.频道数**

| 薯片 | 通道数量 |
|---|---|
| S32K116 | 4 |
| S32K118 | 4 |
| S32K142 | 16 |
| S32K144 | 16 |
| S32K146 | 16 |
| S32K148 | 16 |

### 15.1.2 通过TRGMUX的DMA传输 触发器

来自TRGMUX模块的触发器可以触发前四个DMA通道的DMA传输，例如，LPIT可以通过TRGMUX触发DMA。模块的互连性细节见图17-2。LPIT/DMA的周期性触发分配详见LPIT/DMA周期性触发分配。

异步DMA操作不支持触发选项。

在多个DMA请求被路由到DMAMUX源的情况下，软件需要确保每次只有一个DMA请求被启用。例如，一个以太网定时器的DMA请求可以在某一特定时间启用，而其他定时器可以在该时间启用中断。

# 15.2 简介

## 15.2.1 概述

直接内存访问复用器（DMAMUX）将DMA源（称为插槽）路由到16个DMA通道中的任何一个。请参阅芯片的具体信息以了解详细的源编号。这个过程在下图中有所说明。

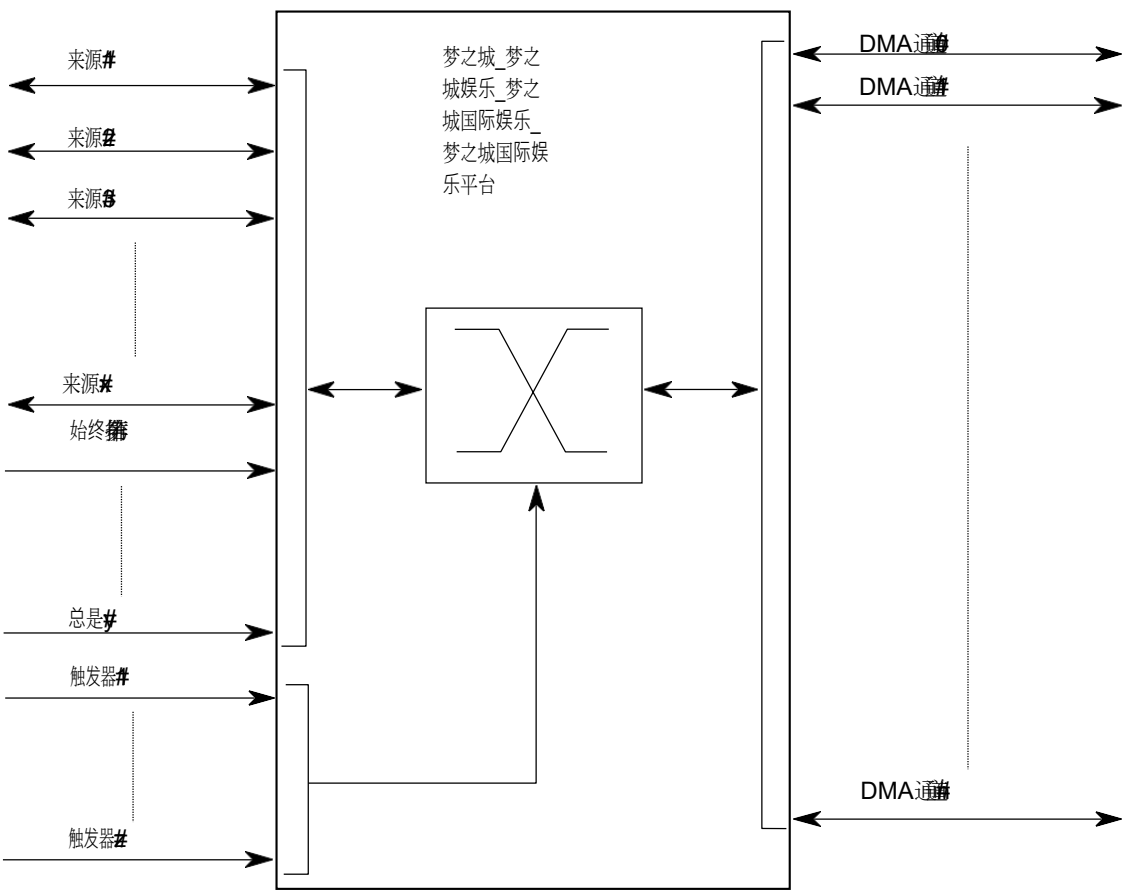



**图15-1.DMAMUX框图**

## 15.2.2 特点

DMAMUX模块提供这些功能。

- 多达61个外设插槽和最多2个始终在线的插槽可以被路由到16个通道。
- 16个可独立选择的DMA通道路由器。

  - 前4个通道还提供了一个触发功能。
- 每个通道路由器可以被分配到一个可能的外设DMA插槽或一个始终在线的插槽中。

### 15.2.3 的操作模式

有以下操作模式可供选择。

- 禁用模式

  在这种模式下，DMA通道被禁用。因为DMA通道的禁用和启用主要是通过DMA配置寄存器完成的，所以这种模式主要是作为DMA通道MUX中DMA通道的复位状态。它也可用于在系统重新配置时暂时中止一个DMA通道，例如，改变DMA触发器的周期。

- 正常模式

  在这种模式下，一个DMA源被直接路由到指定的DMA通道。在这种模式下，DMAMUX的操作对系统是完全透明的。

- 周期性触发模式

  在这种模式下，DMA源可能只请求一次DMA传输，比如当发送缓冲区变空或接收缓冲区变满时，周期性地请求DMA传输。

  周期的配置是由外部周期性中断定时器（例如PIT）来完成。这种模式只适用于通道0-3。

## 15.3 外部信号 描述

DMAMUX没有外部引脚。

## 15.4 内存图/寄存器 定义

本节对DMAMUX中所有内存映射的寄存器进行了详细描述。

## 15.4.1 DMAMUX 寄存器 描述

### 15.4.1.1 DMAMUX 内存图

DMAMUX基础地址。4002_1000h

| 偏移 | 注册 | 宽度<br>(单位：比特) | 访问 | 重置值 |
|---|---|---|---|---|
| 0h - Fh | 通道配置寄存器（CHCFG0 - CHCFG15）。 | 8 | RW | 00h |

### 15.4.1.2 通道配置寄存器（CHCFG0 - CHCFG15）。

#### 15.4.1.2.1 偏移

对于a=0到15。

| 注册 | 偏移 |
|---|---|
| CHCFGa | 0h + (a × 1h) |

#### 15.4.1.2.2 职能

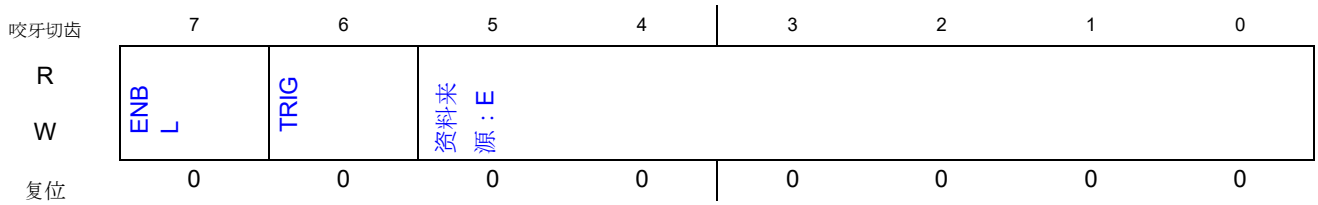
每个DMA通道都可以独立启用/禁用，并与系统中的一个DMA插槽（外围插槽或始终在线的插槽）相关联。

**注意事项**

用相同的源值设置多个CHCFG寄存器将导致不可预知的行为。即使一个通道被禁用（ENBL==0），也是如此。

在改变触发或源设置之前，必须通过CHCFGn[ENBL]禁用一个DMA通道。

#### 15.4.1.2.3 图示




#### 15.4.1.2.4 领域

| 场地 | 职能 |
|---|---|
| 7<br>ENBL | DMA通道启用<br>启用DMA通道。<br>0b - DMA通道被禁用。这种模式主要是在配置DMAMux的过程中使用。DMA有单独的通道启用/禁用功能，应该用来禁用或重新配置一个DMA通道。<br>1b - DMA通道被启用 |
| 6<br>TRIG | DMA通道触发器启用<br>启用被触发的DMA通道的定期触发功能。<br>0b - 触发被禁用。如果触发被禁用并且ENBL被设置，DMA通道将简单地将指定的源路由到DMA通道。(正常模式)<br>1b - 触发功能已启用。如果触发被启用并且ENBL被设置，DMAMUX处于周期性触发模式。 |
| 5-0<br>源头 | DMA通道源（槽位）<br>指定哪个DMA源，如果有的话，被路由到一个特定的DMA通道。有关外设及其槽号的细节，请参见芯片特定的DMAMUX信息。 |

## 15.5 功能 描述

DMAMUX的主要目的是为系统使用可用的DMA通道提供灵活性。

因此，DMAMUX的配置被认为是在执行系统启动代码期间完成的静态程序。然而，如果遵循 "启用和配置源
"中概述的程序，DMAMUX的配置可以在系统的正常运行中被改变。

在功能上，DMAMUX通道可以分为两类。

- 实现正常路由功能和定期触发能力的通道
- 只实现正常路由功能的通道

### 15.5.1 具有周期性触发的DMA通道 能力

除了正常的路由功能，DMAMUX的前4个通道提供了特殊的周期性触发能力，可以用来提供一种自动机制，在固定的时间间隔内传输字节、帧或数据包，而不需要处理器干预。

触发器是由外部周期性中断定时器（例如PIT）产生的；因此，周期性触发间隔的配置是通过配置外部周期性定时器完成的。

**注意事项**

由于系统的动态性质（由于DMA通道优先级、总线仲裁、中断服务程序长度等），不能保证触发和实际DMA传输之间的时钟周期数。

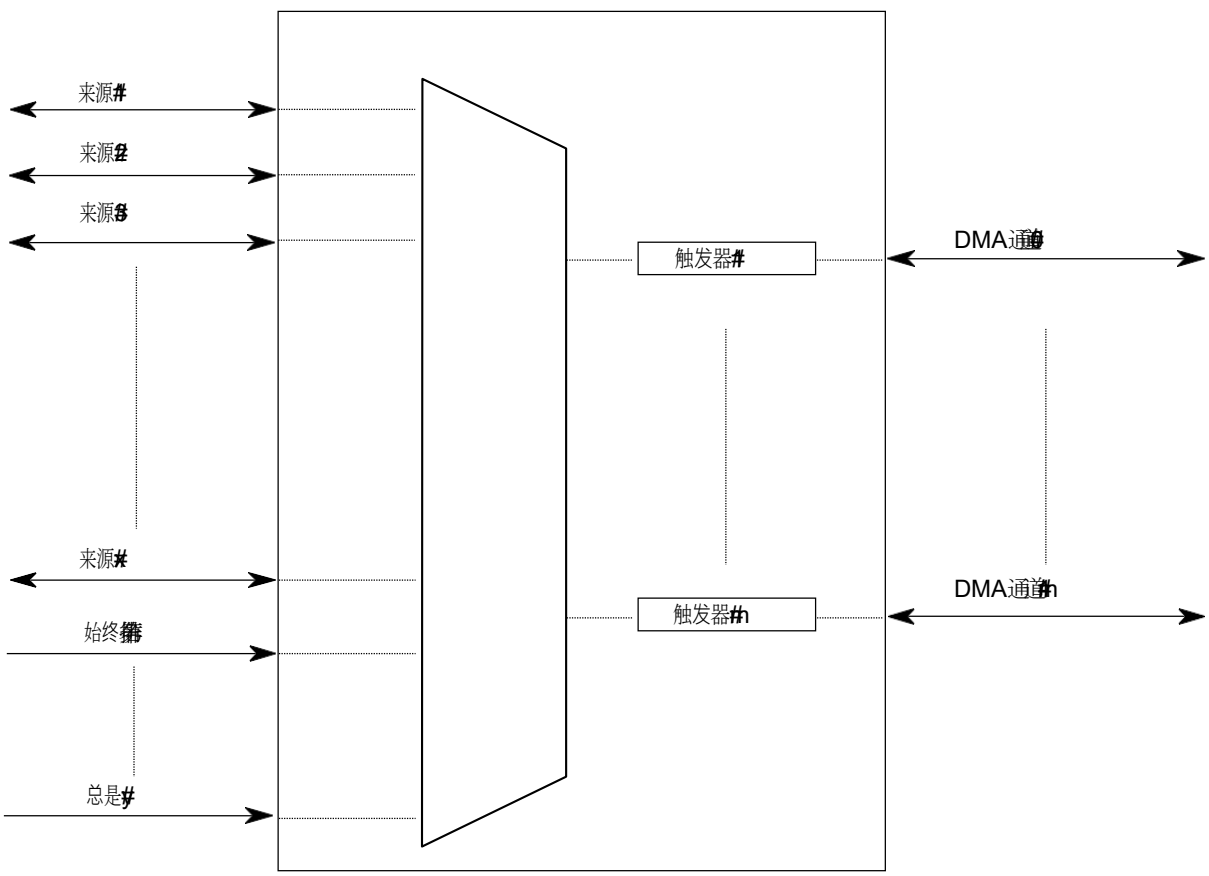



**图15-2.DMAMUX触发的通道**

DMA通道的触发能力允许系统定期安排DMA传输，通常是在某些外设的传输端，而不需要处理器的干预。这种触发的工作方式是将请求从外设转移到DMA，直到看到一个触发事件。这在下图中得到了说明。

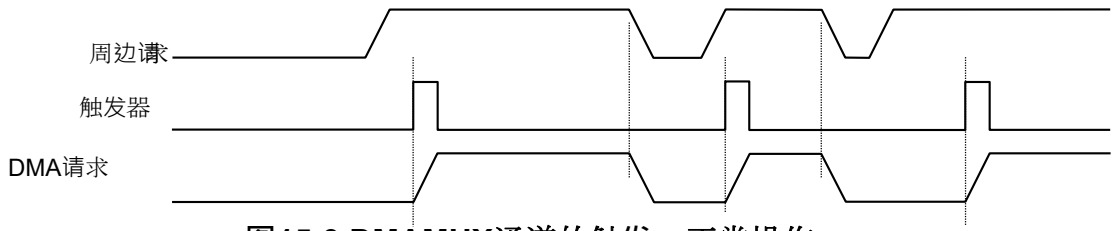



**图15-3.DMAMUX通道的触发：正常操作**

在DMA请求被服务后，外设将否定其请求，有效地重置了门控机制，直到外设重新发出请求并看到下一个触发事件。这意味着，如果看到了一个触发器，但外设没有请求传输，那么这个触发器将被忽略。这种情况在下图中得到了说明。

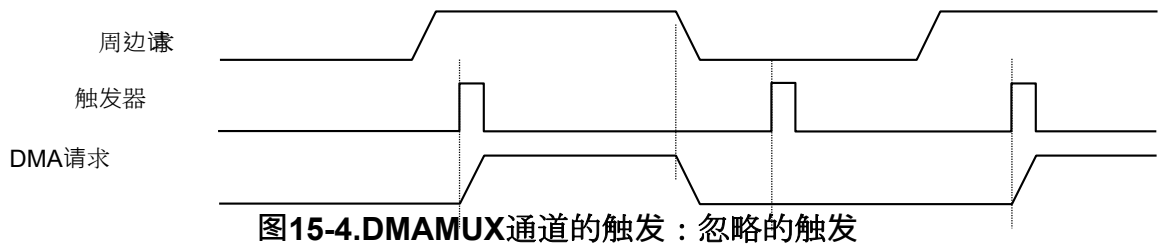



**图15-4.DMAMUX通道的触发：忽略的触发**

这种触发能力可用于任何支持DMA传输的外设，对两种情况最为有用。

- 定期轮询特定总线上的外部设备

  作为一个例子，一个SPI的发送端被分配到一个带有触发器的DMA通道，如上所述。在它被设置好之后，只要它的发送缓冲区是空的，SPI就会请求DMA传输，大概是来自内存。通过在这个通道上使用一个触发器，SPI的传输可以每5 µs自动执行一次（作为一个例子）。在SPI的接收端，SPI和DMA可以被配置为将接收数据传输到存储器中，有效地实现了一种定期从外部设备读取数据并将结果传输到存储器中而无需处理器干预的方法。

- 使用GPIO端口来驱动或采样波形

通过配置DMA将数据传输到一个或多个GPIO端口，有可能使用存储在片上存储器中的表格数据创建复杂的波形。反之，利用DMA定期从一个或多个GPIO端口传输数据，就有可能对复杂的波形进行采样，并将结果以表格形式存储在片上存储器中。

### 15.5.2 没有触发的DMA通道 能力

DMAMUX的其他通道提供正常的路由功能，如操作模式中所述。

### 15.5.3 始终启用的DMA 源

除了可以用作DMA源的外设外，还有2个额外的DMA源，它们总是被启用。与外设DMA源不同的是，外设在DMA传输过程中控制数据流，而始终启用的DMA源不提供这种数据传输的 "节流 "功能。这些源在以下情况下是最有用的。

- 向/从GPIO执行DMA传输-
  从/向一个或多个GPIO引脚移动数据，可以是无节制的（即尽可能快），也可以是定期的（使用DMA触发功能）。
- 执行从内存到内存的DMA传输-
  将数据从内存移至内存，通常是尽可能快，有时需要软件激活。
- 执行从内存到外部总线的DMA传输，或反之亦然--
  类似于内存到内存的传输，这通常要尽可能快地完成。
- 任何需要软件激活的DMA传输-任何应该由软件明确启动的DMA传输。

在软件应该启动DMA传输的情况下，可以使用一个始终启用的DMA源来提供最大的灵活性。当通过软件激活一个DMA通道时，小循环的后续执行需要发送一个新的启动事件。这可以是一个新的软件激活，也可以是一个来自DMA通道MUX的传输请求。这样做的选项有

- 在一个小循环中传输所有数据。

通过配置DMA在一个小循环中传输所有的数据（也就是主循环计数器=1），不需要重新激活通道。这个选项的缺点是在确定DMA传输将对系统造成的负载时减少了颗粒度。对于这个选项，DMA通道必须在DMA通道MUX中被禁用。

- 使用明确的软件重新激活。

  在这个选项中，DMA被配置为使用小循环和大循环来传输数据，但是处理器需要*在每个小循环之后*通过写入DMA寄存器来重新激活该通道。对于这个选项，DMA通道必须在DMA通道MUX中被禁用。

- 使用一个始终启用的DMA源。

  在这个选项中，DMA被配置为使用小循环和大循环传输数据，DMA通道MUX做通道的重新激活。对于这个选项，DMA通道应该被启用，并指向一个"总是启用"的源。注意，通道的重新激活可以是连续的（DMA触发被禁用），也可以使用DMA触发能力。通过这种方式，有可能执行从一个源到另一个源的数据包的定期传输，而不需要处理器的干预。

## 15.6 初始化/应用 信息

本节提供初始化DMA通道MUX的指令。

### 15.6.1 复位

每个单独位的复位状态显示在内存图/寄存器定义中。总之，复位后，所有通道都被禁用，使用前必须明确启用。

### 15.6.2 启用和配置 源

要启用一个具有周期性触发的源。

1. **确定**该源将与哪个**DMA通道相关**联。请注意，只有前**4个DMA通道具有定期触发能力**。
2. 清除DMA通道的CHCFG[ENBL]和CHCFG[TRIG]字段。

3. 确保DMA通道在DMA中被正确配置。此时DMA通道可能被启用。
4. 配置相应的定时器。
5. 选择要被路由到DMA通道的源。写入相应的CHCFG寄存器，确保CHCFG[ENBL]和CHCFG[TRIG]字段被设置。

**注意事项**

下面是一个例子。关于这个设备的DMA通道中具有触发能力的数量，请参见芯片的具体信息。

配置源码#5发送，与DMA通道1一起使用，具有定期触发能力。

1. 写0x00到CHCFG1。
2. 在DMA中配置通道1，包括启用该通道。
3. 为所需的触发时间间隔配置一个定时器。
4. 写0xC5到CHCFG1。

下面的代码例子说明了上述步骤1和4。

```
void DMAMUX_Init(uint8_t DMA_CH, uint8_t DMAMUX_SOURCE)
{
    dmamux_0.chcfg[dma_ch].b.source = dmamux_source;
    dmamux_0.chcfg[dma_ch].b.enbl=   1;
    dmamux_0.chcfg[dma_ch].b.trig= 1。
}
```

启用一个源，不定期触发。

1. 确定该源将与哪个DMA通道相关联。请注意，只有前4个DMA通道具有定期触发能力。
2. 清除DMA通道的CHCFG[ENBL]和CHCFG[TRIG]字段。
3. 确保DMA通道在DMA中被正确配置。此时DMA通道可能被启用。
4. 选择要被路由到DMA通道的源。写入相应的CHCFG寄存器，确保CHCFG[ENBL]被设置，同时CHCFG[TRIG]被清除。

**注意事项**

下面是一个例子。关于这个设备的DMA通道中具有触发能力的数量，请参见芯片配置细节。

配置源#5发送，用于DMA通道1，没有定期触发能力。

1. 写0x00到CHCFG1。

2. 在DMA中配置通道1，包括启用该通道。
3. 写0x85到CHCFG1。

下面的代码例子说明了上述步骤1和3。

```
在文件registers.h中。
#define DMAMUX_BASE_ADDR0x40021000/* 仅限示例!*/
/* 下面的例子假设char是8位的 */
volatile unsigned char *CHCFG0 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0003);
volatile    unsigned    char    *CHCFG1    =    (volatile    unsigned    char    *)
(DMAMUX_BASE_ADDR+0x0002);volatile unsigned char *CHCFG2 = (volatile unsigned char *)
(DMAMUX_BASE_ADDR+0x0001); volatile unsigned char *CHCFG3 = (volatile unsigned char *)
(DMAMUX_BASE_ADDR+0x0000)。volatile unsigned char *CHCFG4 = (volatile unsigned char *)
(DMAMUX_BASE_ADDR+0x0007); volatile unsigned char *CHCFG5 = (volatile unsigned char *)
(DMAMUX_BASE_ADDR+0x0006);volatile unsigned char *CHCFG6 = (volatile unsigned char *)
(DMAMUX_BASE_ADDR+0x0005); volatile unsigned char *CHCFG7 = (volatile unsigned char *)
(DMAMUX_BASE_ADDR+0x0004);volatile unsigned char *CHCFG8 = (volatile unsigned char *)
(DMAMUX_BASE_ADDR+0x000B); volatile unsigned char *CHCFG9 = (volatile unsigned char *)
(DMAMUX_BASE_ADDR+0x000A) 。volatile unsigned char *CHCFG10= (volatile unsigned char
*) (DMAMUX_BASE_ADDR+0x0009); volatile unsigned char *CHCFG11= (volatile unsigned char
*) (DMAMUX_BASE_ADDR+0x0008);volatile unsigned char *CHCFG12= (volatile unsigned char
*) (DMAMUX_BASE_ADDR+0x000F); volatile unsigned char *CHCFG13= (volatile unsigned char
*) (DMAMUX_BASE_ADDR+0x000E) 。volatile unsigned char *CHCFG14= (volatile unsigned
char *) (DMAMUX_BASE_ADDR+0x000D); volatile unsigned char *CHCFG15= (volatile unsigned
char *) (DMAMUX_BASE_ADDR+0x000C) 。

在文件main.c中。
#include "registers.h"
:
:
*CHCFG1 = 0x00。
*CHCFG1 = 0x85。
```

要停用一个源。

一个特定的DMA源可以通过不向任何CHCFG寄存器写入相应的源值而被禁用。此外，一些特定的模块配置可能是必要的。更多细节见相应章节。

切换一个DMA通道的来源。

1. 禁用DMA中的DMA通道，并为新的来源重新配置通道。
2. 清除DMA通道的CHCFG[ENBL]和CHCFG[TRIG]位。
3. 选择要被路由到DMA通道的源。写入相应的CHCFG寄存器，确保CHCFG[ENBL]和CHCFG[TRIG]字段被设置。

将DMA通道8从源5号传输切换到源7号传输。

1. 在DMA配置寄存器中，禁用DMA通道8，并重新配置它以处理向外设插槽7的传输。这个例子假设通道8没有触发能力。
2. 写0x00到CHCFG8。
3. 写0x87到CHCFG8。(在这个例子中，由于假设通道8不支持周期性触发功能，设置CHCFG[TRIG]将没有效果)。

下面的代码例子说明了上述步骤2和3。

在文件registers.h中。

```
#define DMAMUX_BASE_ADDR0x40021000/* 仅限示例!*/
/* 下面的例子假设char是8位的 */
volatile unsigned char *CHCFG0 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0003);
volatile unsigned char *CHCFG1 = (volatile unsigned char *)
(DMAMUX_BASE_ADDR+0x0002);volatile unsigned char *CHCFG2 = (volatile unsigned char *)
(DMAMUX_BASE_ADDR+0x0001); volatile unsigned char *CHCFG3 = (volatile unsigned char *)
(DMAMUX_BASE_ADDR+0x0000)。volatile unsigned char *CHCFG4 = (volatile unsigned char *)
(DMAMUX_BASE_ADDR+0x0007); volatile unsigned char *CHCFG5 = (volatile unsigned char *)
(DMAMUX_BASE_ADDR+0x0006);volatile unsigned char *CHCFG6 = (volatile unsigned char *)
(DMAMUX_BASE_ADDR+0x0005); volatile unsigned char *CHCFG7 = (volatile unsigned char *)
(DMAMUX_BASE_ADDR+0x0004);volatile unsigned char *CHCFG8 = (volatile unsigned char *)
(DMAMUX_BASE_ADDR+0x000B); volatile unsigned char *CHCFG9 = (volatile unsigned char *)
(DMAMUX_BASE_ADDR+0x000A) 。volatile unsigned char *CHCFG10= (volatile unsigned char
*) (DMAMUX_BASE_ADDR+0x0009); volatile unsigned char *CHCFG11= (volatile unsigned char
*) (DMAMUX_BASE_ADDR+0x0008);volatile unsigned char *CHCFG12= (volatile unsigned char
*) (DMAMUX_BASE_ADDR+0x000F); volatile unsigned char *CHCFG13= (volatile unsigned char
*) (DMAMUX_BASE_ADDR+0x000E) 。volatile unsigned char *CHCFG14= (volatile unsigned
char *) (DMAMUX_BASE_ADDR+0x000D); volatile unsigned char *CHCFG15= (volatile unsigned
char *) (DMAMUX_BASE_ADDR+0x000C) 。
```

在文件main.c中。#include
"registers.h"

```
:
:
*CHCFG8 = 0x00。
*CHCFG8 = 0x87。
```

# 章 16
# 增强型直接内存访问 (eDMA)

## 16.1 芯片特定的eDMA 信息

不支持等待模式。关于可用的电源模式的详细信息，请参见模块在可用的低功率模式下的操作。

### 16.1.1 通道的数量

S32K1xx系列的通道数量在不同的变体中有所不同。见下表。

**表16-1.频道数**

| 薯片 | 通道数量 |
|---|---|
| S32K116 | 4 |
| S32K118 | 4 |
| S32K142 | 16 |
| S32K144 | 16 |
| S32K146 | 16 |
| S32K148 | 16 |

## 16.2 简介

增强型直接存储器访问（eDMA）控制器是第二代模块，能够在主机处理器的最小干预下执行复杂的数据传输。该硬件微架构包括。

- 一个能执行的DMA引擎。

- 源地址和目的地址的计算
- 数据移动操作

• 包含16个通道中每个通道的传输控制描述符的本地存储器

## 16.2.1 eDMA系统块 图

图16-1说明了eDMA系统的组成部分，包括eDMA模块（"引擎"）。

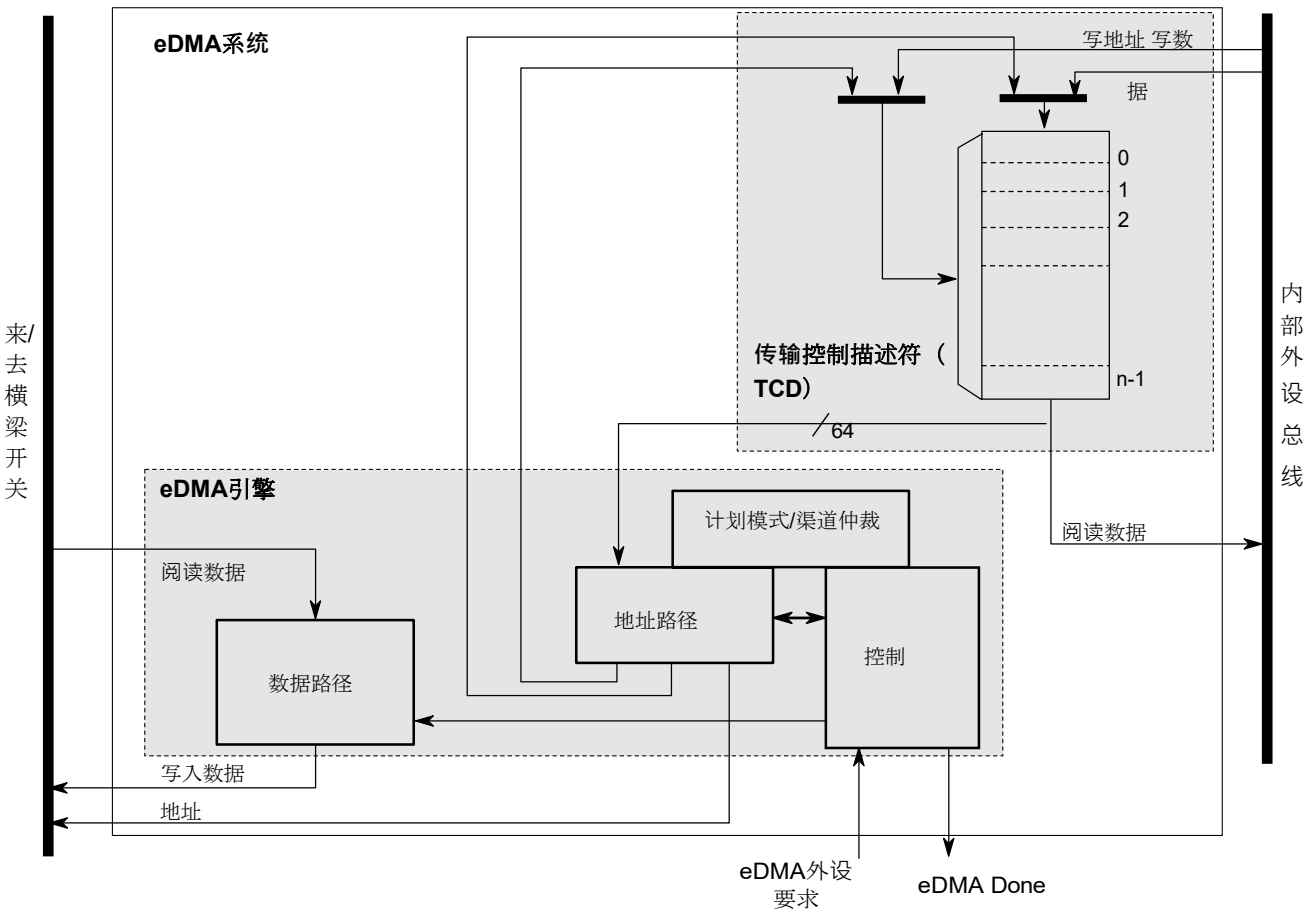



**图16-1. eDMA系统框图**

## 16.2.2 块 部分

eDMA模块被划分为两个主要模块：eDMA引擎和传输控制描述器本地存储器

。

eDMA引擎被进一步划分为四个子模块。

**表16-2. eDMA引擎子模块**

| 子模块 | 职能 |
| --- | --- |
| 地址路径 | 该块实现了两个通道传输控制描述符的注册版本，即通道x和通道y，并管理所有主总线地址的计算。所有的通道都提供相同的功能。这个结构允许在一个读/写序列完成后，如果在第一个通道激活时，一个更高的优先级的通道激活被断言，与一个通道相关的数据传输将被抢占。在一个通道被激活后，它一直运行到小循环完成，除非被更高优先级的通道抢占。这就提供了一个机制（由DCHPRIn[ECP]启用），一个大的数据移动操作可以被抢占，以减少另一个通道被阻止执行的时间。<br>当任何通道被选择执行时，其TCD的内容被从本地存储器中读取，并加载到地址路径通道x寄存器中，用于正常启动，并加载到通道y寄存器中，用于抢先启动。在小循环执行完毕后，地址路径硬件将TCDn_{SADDR, DADDR, CITER}的新值写回本地存储器。如果主要的迭代次数用完了，就会进行额外的处理，包括最后的地址指针更新，重新加载TCDn_CITER字段，以及可能从内存中获取下一个TCDn作为散点/聚集操作的一部分。 |
| 数据路径 | 该模块实现了总线主站的读/写数据路径。它包括一个数据缓冲器和必要的复用逻辑，以支持任何需要的数据排列。内部读数据总线是主要输入，而内部写数据总线是主要输出。<br>地址和数据路径模块直接支持2级流水线的内部总线。地址路径模块代表了总线流水线的第1阶段（地址阶段），而数据路径模块实现了流水线的第2阶段（数据阶段）。 |
| 程序模型/通道仲裁 | 该模块实现了eDMA编程模型的第一部分，以及通道仲裁逻辑。编程模型寄存器被连接到内部外设总线。eDMA外设请求输入和中断请求输出也被连接到这个块（通过控制逻辑）。 |
| 控制 | 该块为eDMA引擎提供所有的控制功能。对于源码和目的码大小相等的数据传输，eDMA引擎执行一系列源码读/目的码写操作，直到小循环字节数中指定的字节数移动完毕。对于大小不等的描述符，每次引用较大尺寸的数据时，都需要多次访问较小尺寸的数据。举例来说，如果源尺寸引用16位数据，而目的地是32位数据，则要进行两次读取，然后进行一次32位写入。 |

传输控制描述符的本地存储器被进一步划分为。

**表16-3.传输控制描述符内存**

| 子模块 | 描述 |
| --- | --- |
| 存储器控制器 | 这个逻辑实现了所需的双端口控制器，管理来自eDMA引擎的访问，以及来自内部外设总线的引用。如前所述，在同时访问的情况下，eDMA引擎被赋予优先权，而外设交易被停滞。 |
| 内存阵列 | 每个通道的传输配置文件的TCD存储。 |

### 16.2.3 特点

eDMA是一个高度可编程的数据传输引擎，经过优化，可以最大限度地减少主机处理器的任何必要干预。它的目的是用于要传输的数据大小是静态已知的，而不是在传输的数据本身中定义的应用。eDMA模块的特点是。

- 所有的数据移动都通过双地址传输：从源头读，写到目的地
  - 可编程的源地址和目的地址以及传输大小
  - 支持增强的寻址模式
- 16个通道的实现，在主机处理器的最小干预下执行复杂的数据传输
  - 内部数据缓冲器，用作临时存储，支持16和32字节的传输
  - 连接到用于总线主控数据移动的横梁开关
- 传输控制描述符（TCD）被组织起来，以支持两个深度的、嵌套的传输操作
  - 每个通道都有32字节的TCD存储在本地存储器中
  - 一个由小字节传输计数定义的内部数据传输循环
  - 一个由主要迭代次数定义的外部数据传输循环
- 通过三种方法中的一种激活通道。
  - 明确的软件启动
  - 通过一个通道对通道的链接机制启动，以实现连续传输
  - 以外设为节奏的硬件请求，每个通道一个
- 固定优先级和轮流通道仲裁
- 通过可编程的中断请求报告通道完成情况
  - 每个通道有一个中断，可在主要迭代次数完成时断言。
  - 每个通道都有可编程的错误终端，并在逻辑上相加，形成一个错误中断到中断控制器
- 可编程支持散射/聚集DMA处理
- 支持复杂的数据结构

在本模块的讨论中，*n*是用来指通道号的。

## 16.3 的操作模式

eDMA在以下模式下运行。

**表16-4.操作模式**

| 模式 | 描述 |
|---|---|
| 正常 | 在正常模式下，eDMA在源和目标之间传输数据。源点和终点可以是一个内存块或一个能够与eDMA操作的I/O块。<br>一个服务请求启动了传输控制描述符（TCD）中指定的特定字节数（NBYTES）的传输。小循环是为每个服务请求传输这些NBYTES的读写操作的序列。每个服务请求执行一次大循环的迭代，传输NBYTES的数据。 |
| 调试 | 在调试模式下，DMA操作可通过控制寄存器进行配置。<br>• 如果CR[EDBG]被清除，DMA继续运行。<br>• 如果CR[EDBG]被设置，eDMA停止传输数据。如果在通道处于活动状态时进入调试模式，eDMA继续运行，直到通道退出。 |
| 等待 | 在进入Wait模式之前，DMA试图完成其当前的传输。传输完成后，设备进入等待模式。 |

## 16.4 内存图/寄存器 定义

eDMA的编程模型被划分为两个区域。

- 第一个区域定义了一些提供控制功能的寄存器
- 第二个区域对应的是本地传输控制描述符（TCD）存储器

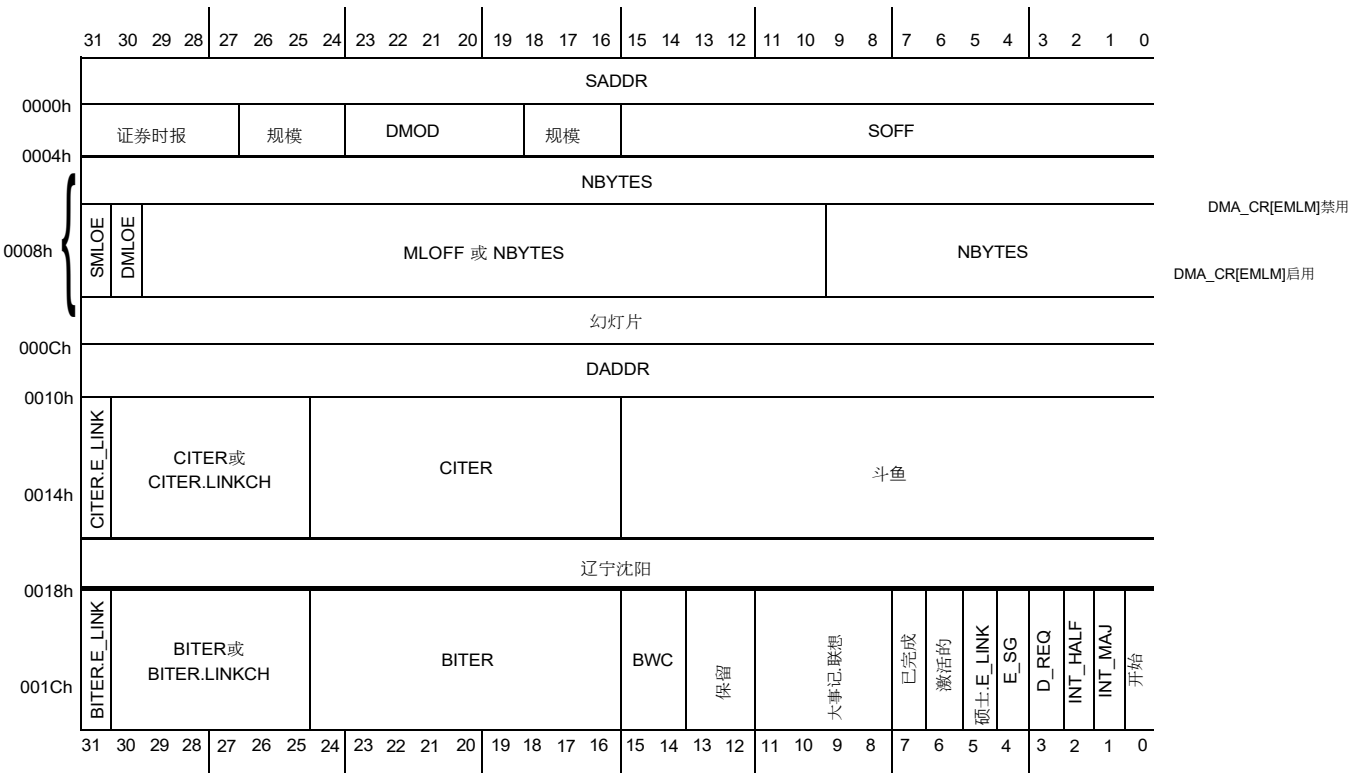
### 16.4.1 TCD 记忆

每个通道都需要一个32字节的传输控制描述符来定义所需的数据移动操作。通道描述符按顺序存储在本地存储器中：通道0，通道1，...通道15。每个TCDn定义以11个16或32位的寄存器形式呈现。

## 16.4.2 TCD 初始化

在激活一个通道之前，你必须用适当的传输配置文件来初始化其TCD。

## 16.4.3 TCD 结构




## 16.4.4 保留的内存和位 字段

- 读取寄存器中的保留位，返回值为0。
- 对寄存器中保留位的写入将被忽略。
- 读或写一个保留的内存位置会产生一个总线错误。

## 16.4.5 DMA寄存器 描述

### 16.4.5.1 DMA内存图

DMA基础地址。4000_8000h

| 偏移 | 注册 | 宽度<br>(单位：比特) | 访问 | 重置值 |
|---|---|---|---|---|
| 0h | 控制寄存器（CR） | 32 | RW | 0000_0000h |
| 4h | 错误状态寄存器（ES） | 32 | RO | 0000_0000h |
| 丘吉尔 | 启用请求寄存器（ERQ） | 32 | RW | 0000_0000h |
| 14h | 启用错误中断寄存器（EEI） | 32 | RW | 0000_0000h |
| 18h | 清除启用错误中断寄存器（CEEI）。 | 8 | 钨钢 | 00h |
| 19h | 设置启用错误中断寄存器（SEEI）。 | 8 | 钨钢 | 00h |
| 1Ah | 清除启用请求寄存器（CERQ） | 8 | 钨钢 | 00h |
| 1Bh | 设置启用请求寄存器（SERQ） | 8 | 钨钢 | 00h |
| 1Ch | 清除DONE状态位寄存器（CDNE）。 | 8 | 钨钢 | 00h |
| 1Dh | 设置START位寄存器（SSRT） | 8 | 钨钢 | 00h |
| 1Eh | 清除错误寄存器（CERR）。 | 8 | 钨钢 | 00h |
| 1Fh | 清除中断请求寄存器（CINT）。 | 8 | 钨钢 | 00h |
| 24h | 中断请求寄存器（INT） | 32 | W1C | 0000_0000h |
| 2Ch | 错误寄存器（ERR）。 | 32 | W1C | 0000_0000h |
| 34h | 硬件请求状态寄存器（HRS） | 32 | RO | 0000_0000h |
| 44h | 启用停止寄存器中的异步请求（EARS）。 | 32 | RW | 0000_0000h |
| 100h | 通道n优先级寄存器（DCHPRI3）。 | 8 | RW | 03h |
| 101h | 通道n优先级寄存器（DCHPRI2）。 | 8 | RW | 02h |
| 102h | 通道n优先级寄存器（DCHPRI1）。 | 8 | RW | 01h |
| 103h | 通道n优先级寄存器（DCHPRI0）。 | 8 | RW | 00h |
| 104h | 通道n优先级寄存器（DCHPRI7）。 | 8 | RW | 07h |
| 105h | 通道n优先级寄存器（DCHPRI6）。 | 8 | RW | 06h |
| 106h | 通道n优先级寄存器（DCHPRI5）。 | 8 | RW | 05h |
| 107h | 通道n优先级寄存器（DCHPRI4）。 | 8 | RW | 04h |
| 108h | 通道n优先级寄存器（DCHPRI11）。 | 8 | RW | 0Bh |
| 109h | 通道n优先级寄存器（DCHPRI10）。 | 8 | RW | 0Ah |
| 10Ah | 通道n优先级寄存器（DCHPRI9）。 | 8 | RW | 09h |
| 10Bh | 通道n优先级寄存器（DCHPRI8）。 | 8 | RW | 08h |
| 10Ch | 通道n优先级寄存器（DCHPRI15）。 | 8 | RW | 0Fh |
| 10Dh | 通道n优先级寄存器（DCHPRI14）。 | 8 | RW | 0Eh |
| 10Eh | 通道n优先级寄存器（DCHPRI13）。 | 8 | RW | 0Dh |
| 10Fh | 通道n优先级寄存器（DCHPRI12）。 | 8 | RW | 0Ch |
| 1000h | TCD源地址（TCD0_SADDR）。 | 32 | RW | 见描述。 |
| 1004h | TCD有符号的源地址偏移（TCD0_SOFF）。 | 16 | RW | 见描述。 |

| | | | | |
|---|---|---|---|---|
| 1006h | TCD传输属性（TCD0_ATTR）。 | 16 | RW | 见描述。 |
| 1008h | TCD小字节计数（小环路映射已禁用）（TCD0_NBY TES_MLNO）。 | 32 | RW | 见描述。 |

*表继续在下一页...*

| 偏移 | 注册 | 宽度<br>(单位：比特) | 访问 | 重置值 |
|---|---|---|---|---|
| 1008h | TCD有符号的小环路偏移（小环路映射已启用，偏移已禁用）（TCD0_NBYTES_MLOFFNO）。 | 32 | RW | 见描述。 |
| 1008h | TCD有符号的小环路偏移（小环路映射和偏移已启用）（TCD0_NBYTES_MLOFFYES）。 | 32 | RW | 见描述。 |
| 100Ch | TCD最后源地址调整（TCD0_SLAST）。 | 32 | RW | 见描述。 |
| 1010h | TCD目标地址（TCD0_DADDR）。 | 32 | RW | 见描述。 |
| 1014h | TCD有符号的目标地址偏移（TCD0_DOFF）。 | 16 | RW | 见描述。 |
| 1016h | TCD 当前小环路链接，大环路计数（通道链接已被禁用）（TCD0_CITER_ELINKNO）。 | 16 | RW | 见描述。 |
| 1016h | TCD 当前小环路链接，大环路计数（通道链接已启用）（TCD0_CITER_ELINKYES）。 | 16 | RW | 见描述。 |
| 1018h | TCD最后目的地地址调整/散点收集地址（TCD0_DLASTSGA）。 | 32 | RW | 见描述。 |
| 101Ch | TCD控制和状态（TCD0_CSR）。 | 16 | RW | 见描述。 |
| 101Eh | TCD开始小环路链接，大环路计数（通道链接被禁用）（TCD0_BITER_ELINKNO）。 | 16 | RW | 见描述。 |
| 101Eh | TCD开始小环路链接，大环路计数（通道链接已启用）（TCD0_BITER_ELINKYES）。 | 16 | RW | 见描述。 |
| 1020h | TCD源地址（TCD1_SADDR）。 | 32 | RW | 见描述。 |
| 1024h | TCD有符号的源地址偏移（TCD1_SOFF）。 | 16 | RW | 见描述。 |
| 1026h | TCD传输属性（TCD1_ATTR）。 | 16 | RW | 见描述。 |
| 1028h | TCD小字节计数（小环路映射已禁用）（TCD1_NBY TES_MLNO）。 | 32 | RW | 见描述。 |
| 1028h | TCD有符号的小环路偏移（小环路映射已启用，偏移已禁用）（TCD1_NBYTES_MLOFFNO）。 | 32 | RW | 见描述。 |
| 1028h | TCD有符号的小环路偏移（小环路映射和偏移已启用）（TCD1_NBYTES_MLOFFYES）。 | 32 | RW | 见描述。 |
| 102Ch | TCD最后源地址调整（TCD1_SLAST）。 | 32 | RW | 见描述。 |
| 1030h | TCD目标地址（TCD1_DADDR）。 | 32 | RW | 见描述。 |
| 1034h | TCD有符号的目标地址偏移（TCD1_DOFF）。 | 16 | RW | 见描述。 |
| 1036h | TCD 当前小环路链接，大环路计数（通道链接被禁用）（TCD1_CITER_ELINKNO）。 | 16 | RW | 见描述。 |
| 1036h | TCD 当前小环路链接，大环路计数（通道链接已启用）（TCD1_CITER_ELINKYES）。 | 16 | RW | 见描述。 |

| | | | | |
|---|---|---|---|---|
| 1038h | TCD最后目的地地址调整/散点收集地址（TCD1_DLASTSGA）。 | 32 | RW | 见描述。 |

*表继续在下一页...*

| 偏移 | 注册 | 宽度<br>(单位：比特) | 访问 | 重置值 |
|---|---|---|---|---|
| 103Ch | TCD控制和状态（TCD1_CSR）。 | 16 | RW | 见描述。 |
| 103Eh | TCD开始小环路链接，大环路计数（通道链接被禁用）（TCD1_BITER_ELINKNO）。 | 16 | RW | 见描述。 |
| 103Eh | TCD 开始小环路链接，大环路计数（通道链接已启用）（TCD1_BITER_ELINKYES）。 | 16 | RW | 见描述。 |
| 1040h | TCD源地址（TCD2_SADDR）。 | 32 | RW | 见描述。 |
| 1044h | TCD有符号的源地址偏移（TCD2_SOFF）。 | 16 | RW | 见描述。 |
| 1046h | TCD转移属性（TCD2_ATTR）。 | 16 | RW | 见描述。 |
| 1048h | TCD小字节计数（小环路映射已禁用）（TCD2_NBY TES_MLNO）。 | 32 | RW | 见描述。 |
| 1048h | TCD有符号的小环路偏移（小环路映射已启用，偏移已禁用）（TCD2_NBYTES_MLOFFNO）。 | 32 | RW | 见描述。 |
| 1048h | TCD有符号的小环路偏移（启用小环路映射和偏移）（TCD2_NBYTES_MLOFFYES）。 | 32 | RW | 见描述。 |
| 104Ch | TCD最后源地址调整（TCD2_SLAST）。 | 32 | RW | 见描述。 |
| 1050h | TCD目标地址（TCD2_DADDR）。 | 32 | RW | 见描述。 |
| 1054h | TCD有符号的目标地址偏移（TCD2_DOFF）。 | 16 | RW | 见描述。 |
| 1056h | TCD 当前小环路链接，大环路计数（通道链接被禁用）（TCD2_CITER_ELINKNO）。 | 16 | RW | 见描述。 |
| 1056h | TCD 当前小环路链接，大环路计数（通道链接已启用）（TCD2_CITER_ELINKYES）。 | 16 | RW | 见描述。 |
| 1058h | TCD最后目的地地址调整/散点收集地址（TCD2_DLASTSGA）。 | 32 | RW | 见描述。 |
| 105Ch | TCD控制和状态（TCD2_CSR）。 | 16 | RW | 见描述。 |
| 105Eh | TCD开始小环路链接，大环路计数（通道链接被禁用）（TCD2_BITER_ELINKNO）。 | 16 | RW | 见描述。 |
| 105Eh | TCD 开始小环路链接，大环路计数（通道链接已启用）（TCD2_BITER_ELINKYES）。 | 16 | RW | 见描述。 |
| 1060h | TCD源地址（TCD3_SADDR）。 | 32 | RW | 见描述。 |
| 1064h | TCD有符号的源地址偏移（TCD3_SOFF）。 | 16 | RW | 见描述。 |
| 1066h | TCD转移属性（TCD3_ATTR）。 | 16 | RW | 见描述。 |
| 1068h | TCD小字节计数（小环路映射已禁用）（TCD3_NBY TES_MLNO）。 | 32 | RW | 见描述。 |

| | | | | |
|---|---|---|---|---|
| 1068h | TCD有符号的小环路偏移（小环路映射已启用，偏移已禁用）（TCD3_NBYTES_MLOFFNO）。 | 32 | RW | 见描述。 |

*表继续在下一页...*

| 偏移 | 注册 | 宽度<br>(单位：比特) | 访问 | 重置值 |
|---|---|---|---|---|
| 1068h | TCD有符号的小环路偏移（启用小环路映射和偏移）（TCD3_NBYTES_MLOFFYES）。 | 32 | RW | 见描述。 |
| 106Ch | TCD最后源地址调整（TCD3_SLAST）。 | 32 | RW | 见描述。 |
| 1070h | TCD目标地址（TCD3_DADDR）。 | 32 | RW | 见描述。 |
| 1074h | TCD有符号的目标地址偏移（TCD3_DOFF）。 | 16 | RW | 见描述。 |
| 1076h | TCD 当前小环路链接，大环路计数（通道链接被禁用）（TCD3_CITER_ELINKNO）。 | 16 | RW | 见描述。 |
| 1076h | TCD 当前小环路链接，大环路计数（通道链接已启用）（TCD3_CITER_ELINKYES）。 | 16 | RW | 见描述。 |
| 1078h | TCD最后目的地地址调整/散点收集地址（TCD3_DLASTSGA）。 | 32 | RW | 见描述。 |
| 107Ch | TCD控制和状态（TCD3_CSR）。 | 16 | RW | 见描述。 |
| 107Eh | TCD开始小环路链接，大环路计数（通道链接被禁用）（TCD3_BITER_ELINKNO）。 | 16 | RW | 见描述。 |
| 107Eh | TCD 开始小环路链接，大环路计数（通道链接已启用）（TCD3_BITER_ELINKYES）。 | 16 | RW | 见描述。 |
| 1080h | TCD源地址（TCD4_SADDR）。 | 32 | RW | 见描述。 |
| 1084h | TCD有符号的源地址偏移（TCD4_SOFF）。 | 16 | RW | 见描述。 |
| 1086h | TCD转移属性（TCD4_ATTR）。 | 16 | RW | 见描述。 |
| 1088h | TCD小字节计数（小环路映射被禁用）（TCD4_NBY TES_MLNO）。 | 32 | RW | 见描述。 |
| 1088h | TCD有符号的小环路偏移（小环路映射已启用，偏移已禁用）（TCD4_NBYTES_MLOFFNO）。 | 32 | RW | 见描述。 |
| 1088h | TCD有符号的小环路偏移（启用小环路映射和偏移）（TCD4_NBYTES_MLOFFYES）。 | 32 | RW | 见描述。 |
| 108Ch | TCD最后源地址调整（TCD4_SLAST）。 | 32 | RW | 见描述。 |
| 1090h | TCD目标地址（TCD4_DADDR）。 | 32 | RW | 见描述。 |
| 1094h | TCD有符号的目标地址偏移（TCD4_DOFF）。 | 16 | RW | 见描述。 |
| 1096h | TCD 当前小环路链接，大环路计数（通道链接被禁用）（TCD4_CITER_ELINKNO）。 | 16 | RW | 见描述。 |
| 1096h | TCD 当前小环路链接，大环路计数（通道链接已启用）（TCD4_CITER_ELINKYES）。 | 16 | RW | 见描述。 |
| 1098h | TCD最后目的地地址调整/散点收集地址（TCD4_DLASTSGA）。 | 32 | RW | 见描述。 |

| | | | | |
|---|---|---|---|---|
| 109Ch | TCD控制和状态（TCD4_CSR）。 | 16 | RW | 见描述。 |

*表继续在下一页...*

| 偏移 | 注册 | 宽度<br>(单位：比特) | 访问 | 重置值 |
|---|---|---|---|---|
| 109Eh | TCD开始小环路链接，大环路计数（通道链接被禁用）（TCD4_BITER_ELINKNO）。 | 16 | RW | 见描述。 |
| 109Eh | TCD 开始小环路链接，大环路计数（通道链接已启用）（TCD4_BITER_ELINKYES）。 | 16 | RW | 见描述。 |
| 10A0h | TCD源地址（TCD5_SADDR）。 | 32 | RW | 见描述。 |
| 10A4h | TCD有符号的源地址偏移（TCD5_SOFF）。 | 16 | RW | 见描述。 |
| 10A6h | TCD转移属性（TCD5_ATTR）。 | 16 | RW | 见描述。 |
| 10A8h | TCD小字节计数（小环路映射被禁用）（TCD5_NBY TES_MLNO）。 | 32 | RW | 见描述。 |
| 10A8h | TCD有符号的小环路偏移（小环路映射已启用，偏移已禁用）（TCD5_NBYTES_MLOFFNO）。 | 32 | RW | 见描述。 |
| 10A8h | TCD有符号的小环路偏移（启用小环路映射和偏移）（TCD5_NBYTES_MLOFFYES）。 | 32 | RW | 见描述。 |
| 10ACh | TCD最后源地址调整（TCD5_SLAST）。 | 32 | RW | 见描述。 |
| 10B0h | TCD目标地址（TCD5_DADDR）。 | 32 | RW | 见描述。 |
| 10B4h | TCD有符号的目标地址偏移（TCD5_DOFF）。 | 16 | RW | 见描述。 |
| 10B6h | TCD 当前小环路链接，大环路计数（通道链接被禁用）（TCD5_CITER_ELINKNO）。 | 16 | RW | 见描述。 |
| 10B6h | TCD 当前小环路链接，大环路计数（通道链接已启用）（TCD5_CITER_ELINKYES）。 | 16 | RW | 见描述。 |
| 10B8h | TCD最后目的地地址调整/散点收集地址（TCD5_DLASTSGA）。 | 32 | RW | 见描述。 |
| 10BCh | TCD控制和状态（TCD5_CSR）。 | 16 | RW | 见描述。 |
| 10BEh | TCD开始小环路链接，大环路计数（通道链接被禁用）（TCD5_BITER_ELINKNO）。 | 16 | RW | 见描述。 |
| 10BEh | TCD 开始小环路链接，大环路计数（通道链接已启用）（TCD5_BITER_ELINKYES）。 | 16 | RW | 见描述。 |
| 10C0h | TCD源地址（TCD6_SADDR）。 | 32 | RW | 见描述。 |
| 10C4h | TCD有符号的源地址偏移（TCD6_SOFF）。 | 16 | RW | 见描述。 |
| 10C6h | TCD转移属性（TCD6_ATTR）。 | 16 | RW | 见描述。 |
| 10C8h | TCD小字节计数（小环路映射已禁用）（TCD6_NBY TES_MLNO）。 | 32 | RW | 见描述。 |
| 10C8h | TCD有符号的小环路偏移（小环路映射已启用，偏移已禁用）（TCD6_NBYTES_MLOFFNO）。 | 32 | RW | 见描述。 |

| | | | | |
|---|---|---|---|---|
| 10C8h | TCD有符号的小环路偏移（启用小环路映射和偏移）（TCD6_NBYTES_MLOFFYES）。 | 32 | RW | 见描述。 |

*表继续在下一页...*

| 偏移 | 注册 | 宽度<br>(单位：比特) | 访问 | 重置值 |
|---|---|---|---|---|
| 10CCh | TCD最后源地址调整（TCD6_SLAST）。 | 32 | RW | 见描述。 |
| 10D0h | TCD目标地址（TCD6_DADDR）。 | 32 | RW | 见描述。 |
| 10D4h | TCD有符号的目标地址偏移（TCD6_DOFF）。 | 16 | RW | 见描述。 |
| 10D6h | TCD 当前小环路链接，大环路计数（通道链接被禁用）（TCD6_CITER_ELINKNO）。 | 16 | RW | 见描述。 |
| 10D6h | TCD 当前小环路链接，大环路计数（通道链接已启用）（TCD6_CITER_ELINKYES）。 | 16 | RW | 见描述。 |
| 10D8h | TCD最后目的地地址调整/散点收集地址（TCD6_DLASTSGA）。 | 32 | RW | 见描述。 |
| 10DCh | TCD控制和状态（TCD6_CSR）。 | 16 | RW | 见描述。 |
| 10DEh | TCD开始小环路链接，大环路计数（通道链接被禁用）（TCD6_BITER_ELINKNO）。 | 16 | RW | 见描述。 |
| 10DEh | TCD 开始小环路链接，大环路计数（通道链接已启用）（TCD6_BITER_ELINKYES）。 | 16 | RW | 见描述。 |
| 10E0h | TCD源地址（TCD7_SADDR）。 | 32 | RW | 见描述。 |
| 10E4h | TCD有符号的源地址偏移（TCD7_SOFF）。 | 16 | RW | 见描述。 |
| 10E6h | TCD转移属性（TCD7_ATTR）。 | 16 | RW | 见描述。 |
| 10E8h | TCD小字节计数（小环路映射已禁用）（TCD7_NBY TES_MLNO）。 | 32 | RW | 见描述。 |
| 10E8h | TCD有符号的小环路偏移（小环路映射已启用，偏移已禁用）（TCD7_NBYTES_MLOFFNO）。 | 32 | RW | 见描述。 |
| 10E8h | TCD有符号的小环路偏移（启用小环路映射和偏移）（TCD7_NBYTES_MLOFFYES）。 | 32 | RW | 见描述。 |
| 10ECh | TCD最后源地址调整（TCD7_SLAST）。 | 32 | RW | 见描述。 |
| 10F0h | TCD目标地址（TCD7_DADDR）。 | 32 | RW | 见描述。 |
| 10F4h | TCD有符号的目标地址偏移（TCD7_DOFF）。 | 16 | RW | 见描述。 |
| 10F6h | TCD 当前小环路链接，大环路计数（通道链接被禁用）（TCD7_CITER_ELINKNO）。 | 16 | RW | 见描述。 |
| 10F6h | TCD 当前小环路链接，大环路计数（通道链接已启用）（TCD7_CITER_ELINKYES）。 | 16 | RW | 见描述。 |
| 10F8h | TCD最后目的地地址调整/散点收集地址（TCD7_DLASTSGA）。 | 32 | RW | 见描述。 |
| 10FCh | TCD控制和状态（TCD7_CSR）。 | 16 | RW | 见描述。 |

| | | | | |
|---|---|---|---|---|
| 10FEh | TCD开始小环路链接，大环路计数（通道链接被禁用）（TCD7_BITER_ELINKNO）。 | 16 | RW | 见描述。 |

*表继续在下一页...*

| 偏移 | 注册 | 宽度<br>(单位：比特) | 访问 | 重置值 |
|---|---|---|---|---|
| 10FEh | TCD 开始小环路链接，大环路计数（通道链接已启用）（TCD7_BITER_ELINKYES）。 | 16 | RW | 见描述。 |
| 1100h | TCD源地址（TCD8_SADDR）。 | 32 | RW | 见描述。 |
| 1104h | TCD有符号的源地址偏移（TCD8_SOFF）。 | 16 | RW | 见描述。 |
| 1106h | TCD传输属性（TCD8_ATTR）。 | 16 | RW | 见描述。 |
| 1108h | TCD小字节计数（小环路映射被禁用）（TCD8_NBY TES_MLNO）。 | 32 | RW | 见描述。 |
| 1108h | TCD有符号的小环路偏移（小环路映射已启用，偏移已禁用）（TCD8_NBYTES_MLOFFNO）。 | 32 | RW | 见描述。 |
| 1108h | TCD有符号的小环路偏移（启用小环路映射和偏移）（TCD8_NBYTES_MLOFFYES）。 | 32 | RW | 见描述。 |
| 110Ch | TCD最后源地址调整（TCD8_SLAST）。 | 32 | RW | 见描述。 |
| 1110h | TCD目标地址（TCD8_DADDR）。 | 32 | RW | 见描述。 |
| 1114h | TCD有符号的目标地址偏移（TCD8_DOFF）。 | 16 | RW | 见描述。 |
| 1116h | TCD 当前小环路链接，大环路计数（通道链接被禁用）（TCD8_CITER_ELINKNO）。 | 16 | RW | 见描述。 |
| 1116h | TCD 当前小环路链接，大环路计数（通道链接已启用）（TCD8_CITER_ELINKYES）。 | 16 | RW | 见描述。 |
| 1118h | TCD最后目的地地址调整/散点收集地址（TCD8_DLASTSGA）。 | 32 | RW | 见描述。 |
| 111Ch | TCD控制和状态（TCD8_CSR）。 | 16 | RW | 见描述。 |
| 111Eh | TCD开始小环路链接，大环路计数（通道链接被禁用）（TCD8_BITER_ELINKNO）。 | 16 | RW | 见描述。 |
| 111Eh | TCD 开始小环路链接，大环路计数（通道链接已启用）（TCD8_BITER_ELINKYES）。 | 16 | RW | 见描述。 |
| 1120h | TCD源地址（TCD9_SADDR）。 | 32 | RW | 见描述。 |
| 1124h | TCD有符号的源地址偏移（TCD9_SOFF）。 | 16 | RW | 见描述。 |
| 1126h | TCD转移属性（TCD9_ATTR）。 | 16 | RW | 见描述。 |
| 1128h | TCD小字节计数（小环路映射被禁用）（TCD9_NBY TES_MLNO）。 | 32 | RW | 见描述。 |
| 1128h | TCD有符号的小环路偏移（小环路映射已启用，偏移已禁用）（TCD9_NBYTES_MLOFFNO）。 | 32 | RW | 见描述。 |
| 1128h | TCD有符号的小环路偏移（启用小环路映射和偏移）（TCD9_NBYTES_MLOFFYES）。 | 32 | RW | 见描述。 |

| | | | | |
|---|---|---|---|---|
| 112Ch | TCD最后源地址调整（TCD9_SLAST）。 | 32 | RW | 见描述。 |

*表继续在下一页...*

| 偏移 | 注册 | 宽度<br>(单位：比特) | 访问 | 重置值 |
|---|---|---|---|---|
| 1130h | TCD目标地址（TCD9_DADDR）。 | 32 | RW | 见描述。 |
| 1134h | TCD有符号的目标地址偏移（TCD9_DOFF）。 | 16 | RW | 见描述。 |
| 1136h | TCD 当前小环路链接，大环路计数（通道链接被禁用）（TCD9_CITER_ELINKNO）。 | 16 | RW | 见描述。 |
| 1136h | TCD 当前小环路链接，大环路计数（通道链接已启用）（TCD9_CITER_ELINKYES）。 | 16 | RW | 见描述。 |
| 1138h | TCD最后目的地地址调整/散点收集地址（TCD9_DLASTSGA）。 | 32 | RW | 见描述。 |
| 113Ch | TCD控制和状态（TCD9_CSR）。 | 16 | RW | 见描述。 |
| 113Eh | TCD开始小环路链接，大环路计数（通道链接被禁用）（TCD9_BITER_ELINKNO）。 | 16 | RW | 见描述。 |
| 113Eh | TCD 开始小环路链接，大环路计数（通道链接已启用）（TCD9_BITER_ELINKYES）。 | 16 | RW | 见描述。 |
| 1140h | TCD源地址（TCD10_SADDR）。 | 32 | RW | 见描述。 |
| 1144h | TCD有符号的源地址偏移（TCD10_SOFF）。 | 16 | RW | 见描述。 |
| 1146h | TCD转移属性（TCD10_ATTR）。 | 16 | RW | 见描述。 |
| 1148h | TCD小字节计数（小环路映射已禁用）（TCD10_NB YTES_MLNO）。 | 32 | RW | 见描述。 |
| 1148h | TCD有符号的小环路偏移（小环路映射已启用，偏移已禁用）（TCD10_NBYTES_MLOFFNO）。 | 32 | RW | 见描述。 |
| 1148h | TCD有符号的小环路偏移（启用小环路映射和偏移）（TCD10_NBYTES_MLOFFYES）。 | 32 | RW | 见描述。 |
| 114Ch | TCD最后源地址调整（TCD10_SLAST）。 | 32 | RW | 见描述。 |
| 1150h | TCD目标地址（TCD10_DADDR）。 | 32 | RW | 见描述。 |
| 1154h | TCD有符号的目标地址偏移（TCD10_DOFF）。 | 16 | RW | 见描述。 |
| 1156h | TCD 当前小环路链接，大环路计数（通道链接已被禁用）（TCD10_CITER_ELINKNO）。 | 16 | RW | 见描述。 |
| 1156h | TCD 当前小环路链接，大环路计数（通道链接已启用）（TCD10_CITER_ELINKYES）。 | 16 | RW | 见描述。 |
| 1158h | TCD最后目的地地址调整/散点收集地址（TCD10_DLASTSGA）。 | 32 | RW | 见描述。 |
| 115Ch | TCD控制和状态（TCD10_CSR）。 | 16 | RW | 见描述。 |
| 115Eh | TCD开始小环路链接，大环路计数（通道链接被禁用）（TCD10_BITER_ELINKNO）。 | 16 | RW | 见描述。 |

| | | | | |
|---|---|---|---|---|
| 115Eh | TCD开始小环路链接，大环路计数（通道链接已启用）（TCD10_BITER_ELINKYES）。 | 16 | RW | 见描述。 |

*表继续在下一页...*

| 偏移 | 注册 | 宽度<br>(单位：比特) | 访问 | 重置值 |
|---|---|---|---|---|
| 1160h | TCD源地址（TCD11_SADDR）。 | 32 | RW | 见描述。 |
| 1164h | TCD有符号的源地址偏移（TCD11_SOFF）。 | 16 | RW | 见描述。 |
| 1166h | TCD转移属性（TCD11_ATTR）。 | 16 | RW | 见描述。 |
| 1168h | TCD小字节计数（小环路映射已禁用）（TCD11_NB YTES_MLNO）。 | 32 | RW | 见描述。 |
| 1168h | TCD有符号的小环路偏移（小环路映射已启用，偏移已禁用）（TCD11_NBYTES_MLOFFNO）。 | 32 | RW | 见描述。 |
| 1168h | TCD有符号的小环路偏移（启用小环路映射和偏移）（TCD11_NBYTES_MLOFFYES）。 | 32 | RW | 见描述。 |
| 116Ch | TCD最后源地址调整（TCD11_SLAST）。 | 32 | RW | 见描述。 |
| 1170h | TCD目标地址（TCD11_DADDR）。 | 32 | RW | 见描述。 |
| 1174h | TCD有符号的目标地址偏移（TCD11_DOFF）。 | 16 | RW | 见描述。 |
| 1176h | TCD 当前小环路链接，大环路计数（通道链接已被禁用）（TCD11_CITER_ELINKNO）。 | 16 | RW | 见描述。 |
| 1176h | TCD 当前小环路链接，大环路计数（通道链接已启用）（TCD11_CITER_ELINKYES）。 | 16 | RW | 见描述。 |
| 1178h | TCD最后目的地地址调整/散点收集地址（TCD11_DLASTSGA）。 | 32 | RW | 见描述。 |
| 117Ch | TCD控制和状态（TCD11_CSR）。 | 16 | RW | 见描述。 |
| 117Eh | TCD开始小环路链接，大环路计数（通道链接被禁用）（TCD11_BITER_ELINKNO）。 | 16 | RW | 见描述。 |
| 117Eh | TCD开始小环路链接，大环路计数（通道链接已启用）（TCD11_BITER_ELINKYES）。 | 16 | RW | 见描述。 |
| 1180h | TCD源地址（TCD12_SADDR）。 | 32 | RW | 见描述。 |
| 1184h | TCD有符号的源地址偏移（TCD12_SOFF）。 | 16 | RW | 见描述。 |
| 1186h | TCD转移属性（TCD12_ATTR）。 | 16 | RW | 见描述。 |
| 1188h | TCD小字节计数（小环路映射已禁用）（TCD12_NB YTES_MLNO）。 | 32 | RW | 见描述。 |
| 1188h | TCD有符号的小环路偏移（小环路映射已启用，偏移已禁用）（TCD12_NBYTES_MLOFFNO）。 | 32 | RW | 见描述。 |
| 1188h | TCD有符号的小环路偏移（启用小环路映射和偏移）（TCD12_NBYTES_MLOFFYES）。 | 32 | RW | 见描述。 |
| 118Ch | TCD最后源地址调整（TCD12_SLAST）。 | 32 | RW | 见描述。 |

| | | | | |
|---|---|---|---|---|
| 1190h | TCD目标地址（TCD12_DADDR）。 | 32 | RW | 见描述。 |

*表继续在下一页...*

| 偏移 | 注册 | 宽度<br>(单位：比特) | 访问 | 重置值 |
|---|---|---|---|---|
| 1194h | TCD有符号的目标地址偏移（TCD12_DOFF）。 | 16 | RW | 见描述。 |
| 1196h | TCD 当前小环路链接，大环路计数（通道链接被禁用）（TCD12_CITER_ELINKNO）。 | 16 | RW | 见描述。 |
| 1196h | TCD 当前小环路链接，大环路计数（通道链接已启用）（TCD12_CITER_ELINKYES）。 | 16 | RW | 见描述。 |
| 1198h | TCD最后目的地地址调整/散点收集地址（TCD12_DLASTSGA）。 | 32 | RW | 见描述。 |
| 119Ch | TCD控制和状态（TCD12_CSR）。 | 16 | RW | 见描述。 |
| 119Eh | TCD开始小环路链接，大环路计数（通道链接被禁用）（TCD12_BITER_ELINKNO）。 | 16 | RW | 见描述。 |
| 119Eh | TCD开始小环路链接，大环路计数（通道链接已启用）（TCD12_BITER_ELINKYES）。 | 16 | RW | 见描述。 |
| 11A0h | TCD源地址（TCD13_SADDR）。 | 32 | RW | 见描述。 |
| 11A4h | TCD有符号的源地址偏移（TCD13_SOFF）。 | 16 | RW | 见描述。 |
| 11A6h | TCD转移属性（TCD13_ATTR）。 | 16 | RW | 见描述。 |
| 11A8h | TCD小字节计数（小环路映射已禁用）（TCD13_NB YTES_MLNO）。 | 32 | RW | 见描述。 |
| 11A8h | TCD有符号的小环路偏移（小环路映射已启用，偏移已禁用）（TCD13_NBYTES_MLOFFNO）。 | 32 | RW | 见描述。 |
| 11A8h | TCD有符号的小环路偏移（启用小环路映射和偏移）（TCD13_NBYTES_MLOFFYES）。 | 32 | RW | 见描述。 |
| 11ACh | TCD最后源地址调整(TCD13_SLAST) | 32 | RW | 见描述。 |
| 11B0h | TCD目标地址（TCD13_DADDR）。 | 32 | RW | 见描述。 |
| 11B4h | TCD有符号的目标地址偏移（TCD13_DOFF）。 | 16 | RW | 见描述。 |
| 11B6h | TCD 当前小环路链接，大环路计数（通道链接被禁用）（TCD13_CITER_ELINKNO）。 | 16 | RW | 见描述。 |
| 11B6h | TCD 当前小环路链接，大环路计数（通道链接已启用）（TCD13_CITER_ELINKYES）。 | 16 | RW | 见描述。 |
| 11B8h | TCD最后目的地地址调整/散点收集地址（TCD13_DLASTSGA）。 | 32 | RW | 见描述。 |
| 11BCh | TCD控制和状态（TCD13_CSR）。 | 16 | RW | 见描述。 |
| 11BEh | TCD开始小环路链接，大环路计数（通道链接被禁用）（TCD13_BITER_ELINKNO）。 | 16 | RW | 见描述。 |
| 11BEh | TCD开始小环路链接，大环路计数（通道链接已启用）（ | 16 | RW | 见描述。 |

| | | | | |
|---|---|---|---|---|
| | TCD13_BITER_ELINKYES）。 | | | |
| 11C0h | TCD源地址（TCD14_SADDR）。 | 32 | RW | 见描述。 |

*表继续在下一页...*

| 偏移 | 注册 | 宽度<br>(单位：比特) | 访问 | 重置值 |
|---|---|---|---|---|
| 11C4h | TCD有符号的源地址偏移（TCD14_SOFF）。 | 16 | RW | 见描述。 |
| 11C6h | TCD转移属性（TCD14_ATTR）。 | 16 | RW | 见描述。 |
| 11C8h | TCD小字节计数（小环路映射已禁用）（TCD14_NB YTES_MLNO）。 | 32 | RW | 见描述。 |
| 11C8h | TCD有符号的小环路偏移（小环路映射已启用，偏移已禁用）（TCD14_NBYTES_MLOFFNO）。 | 32 | RW | 见描述。 |
| 11C8h | TCD有符号的小环路偏移（启用小环路映射和偏移）（TCD14_NBYTES_MLOFFYES）。 | 32 | RW | 见描述。 |
| 11CCh | TCD最后源地址调整（TCD14_SLAST）。 | 32 | RW | 见描述。 |
| 11D0h | TCD目标地址（TCD14_DADDR）。 | 32 | RW | 见描述。 |
| 11D4h | TCD有符号的目标地址偏移（TCD14_DOFF）。 | 16 | RW | 见描述。 |
| 11D6h | TCD 当前小环路链接，大环路计数（通道链接被禁用）（TCD14_CITER_ELINKNO）。 | 16 | RW | 见描述。 |
| 11D6h | TCD 当前小环路链接，大环路计数（通道链接已启用）（TCD14_CITER_ELINKYES）。 | 16 | RW | 见描述。 |
| 11D8h | TCD最后目的地地址调整/散点收集地址（TCD14_DLASTSGA）。 | 32 | RW | 见描述。 |
| 11DCh | TCD控制和状态（TCD14_CSR）。 | 16 | RW | 见描述。 |
| 11DEh | TCD开始小环路链接，大环路计数（通道链接被禁用）（TCD14_BITER_ELINKNO）。 | 16 | RW | 见描述。 |
| 11DEh | TCD开始小环路链接，大环路计数（通道链接已启用）（TCD14_BITER_ELINKYES）。 | 16 | RW | 见描述。 |
| 11E0h | TCD源地址（TCD15_SADDR）。 | 32 | RW | 见描述。 |
| 11E4h | TCD有符号的源地址偏移（TCD15_SOFF）。 | 16 | RW | 见描述。 |
| 11E6h | TCD传输属性（TCD15_ATTR）。 | 16 | RW | 见描述。 |
| 11E8h | TCD小字节计数（小环路映射已禁用）（TCD15_NB YTES_MLNO）。 | 32 | RW | 见描述。 |
| 11E8h | TCD有符号的小环路偏移（小环路映射已启用，偏移已禁用）（TCD15_NBYTES_MLOFFNO）。 | 32 | RW | 见描述。 |
| 11E8h | TCD有符号的小环路偏移（启用小环路映射和偏移）（TCD15_NBYTES_MLOFFYES）。 | 32 | RW | 见描述。 |
| 11ECh | TCD最后源地址调整（TCD15_SLAST）。 | 32 | RW | 见描述。 |
| 11F0h | TCD目标地址（TCD15_DADDR）。 | 32 | RW | 见描述。 |

| | | | | |
|---|---|---|---|---|
| 11F4h | TCD有符号的目标地址偏移（TCD15_DOFF）。 | 16 | RW | 见描述。 |

*表继续在下一页...*

| 偏移 | 注册 | 宽度<br>(单位：比特) | 访问 | 重置值 |
|---|---|---|---|---|
| 11F6h | TCD 当前小环路链接，大环路计数（通道链接已被禁用）（TCD15_CITER_ELINKNO）。 | 16 | RW | 见描述。 |
| 11F6h | TCD 当前小环路链接，大环路计数（通道链接已启用）（TCD15_CITER_ELINKYES）。 | 16 | RW | 见描述。 |
| 11F8h | TCD最后目的地地址调整/散点收集地址（TCD15_DLASTSGA）。 | 32 | RW | 见描述。 |
| 11FCh | TCD控制和状态（TCD15_CSR）。 | 16 | RW | 见描述。 |
| 11FEh | TCD开始小环路链接，大环路计数（通道链接被禁用）（TCD15_BITER_ELINKNO）。 | 16 | RW | 见描述。 |
| 11FEh | TCD开始小环路链接，大环路计数（通道链接已启用）（TCD15_BITER_ELINKYES）。 | 16 | RW | 见描述。 |

## 16.4.5.2 控制寄存器（CR）

### 16.4.5.2.1 偏移

| 注册 | 偏移 |
|---|---|
| CR | 0h |

### 16.4.5.2.2 职能

CR定义了DMA的基本操作配置。

仲裁可以被配置为使用固定优先级或轮流方案。对于固定优先级的仲裁，要求服务的最高优先级的通道被选择执行。通道优先级寄存器分配优先级；见DCHPRIn寄存器。对于轮流仲裁，通道的优先级被忽略，通道被循环使用（从高到低的通道编号）而不考虑优先级。

**注意事项**

为了正确操作，对CR寄存器的写入必须在DMA通道不活动时进行；也就是说，当TCDn_CSR[ACTIVE]位被清除时。

小循环偏移是在小循环完成时添加到最终源地址（TCDn_SADDR）或目标地址（

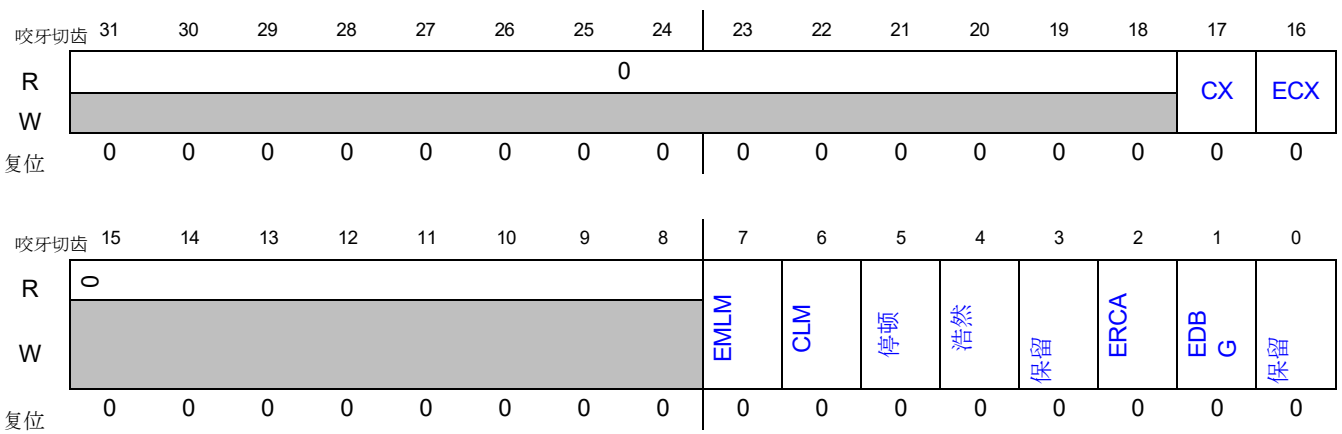
TCDn_DADDR）的地址偏移值。当小环路偏移被启用时，小环路偏移（MLOFF）被添加到最终源地址（TCDn_SADDR），到最终目标地址（TCDn_DADDR）。

或者在地址被写回TCD之前对两者都是如此。如果主循环完成，小循环偏移量被忽略，主循环地址偏移量（TCDn_SLAST和TCDn_DLAST_SGA）被用来计算下一个TCDn_SADDR和TCDn_DADDR值。

当小循环映射被启用时（EMLM为1），TCDn字2被重新定义。TCDn字2的一部分用于指定多个字段：一个源使能位（SMLOE），用于指定小环路偏移量应在小环路完成时应用于源地址（TCDn_SADDR）；一个目的使能位（DMLOE），用于指定小环路偏移量应在小环路完成时应用于目的地址（TCDn_DADDR）；以及符号扩展小环路偏移值（MLOFF）。相同的偏移值（MLOFF）被用于源和目的小循环偏移。当任一小环路偏移被启用时（SMLOE设置或DMLOE设置），NBYTES字段被减少到10位。当两个小环路偏移都被禁用时（SMLOE清除和DMLOE清除），NBYTES字段是一个30位的向量。

当小循环映射被禁用时（EMLM为0），TCDn字2的所有32位被分配到NBYTES字段。

### 16.4.5.2.3 图示




### 16.4.5.2.4 领域

| 场地 | 职能 |
|---|---|
| 31-18<br>- | 保留 |
| 17<br>CX | 取消转账<br>0b - 正常操作<br>1b - 取消剩余的数据传输。停止执行的通道，强制完成小循环。取消在当前读/写序列的最后一次写入后生效。CX位 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| | 在取消指令被执行后，将自身清除。这种取消使通道正常退场，就像小循环完成一样。 |
| 16<br>ECX | 错误 取消传输<br>0b - 正常操作<br>1b - 以与CX位相同的方式取消剩余的数据传输。停止正在执行的通道，强制完成小循环。取消在当前读/写序列的最后一次写入后生效。在取消生效后，ECX位自行清零。除了取消传输外，ECX还将取消视为错误条件，从而更新错误状态寄存器（DMAx_ES）并产生一个可选的错误中断。 |
| 15-8<br>- | 保留 |
| 7<br>EMLM | 启用小环路映射<br>0b - 禁用。TCDn.word2被定义为一个32位的NBYTES字段。<br>1b - 启用。TCDn.word2被重新定义为包括各个使能字段、一个偏移字段和NBYTES字段。各个使能字段允许将小环路偏移应用于源地址、目的地址或两者。当任何一个偏移被启用时，NBYTES字段都会减少。 |
| 6<br>CLM | 连续链接模式<br>**注意：**如果每个服务请求只有一个小循环迭代，例如，如果通道的NBYTES值与源或目标大小相同，则不要使用通道链接到自己的连续链接模式。只需增加NBYTES值就可以实现相同的数据传输情况，这样可以提供更有效、更快的处理。<br>0b - 对自己进行的小环路通道链接在再次激活前要经过通道仲裁。<br>1b - 在再次激活前，与自己进行的小循环通道链接不经过通道仲裁。在小循环完成后，如果该通道有一个启用的小循环通道链接，并且链接通道是它自己，则该通道再次激活。这有效地应用了小环路的偏移量，并重新启动了下一个小环路。 |
| 5<br>停顿 | 停止DMA操作<br>0b - 正常操作<br>1b - 拖延任何新通道的启动。正在执行的通道被允许完成。当这个位被清除时，通道的执行就会恢复。 |
| 4<br>浩然 | 错误时停止<br>0b - 正常操作<br>1b - 任何错误导致HALT位被设置。随后，所有的服务请求都被忽略，直到HALT位被清除。 |
| 3<br>- | 保留<br>保留 |
| 2<br>ERCA | 启用圆周率通道仲裁<br>0b - 使用固定的优先级仲裁来选择通道。1b - 循环仲裁用<br>于通道选择 . |
| 1<br>贸发局 | 启用调试<br>0b - 当处于调试模式时，DMA继续工作。<br>1b - 当处于调试模式时，DMA停滞新通道的启动。正在执行的通道被允许完成。当系统退出调试模式或EDBG位被清除时，通道的执行就会恢复。 |
| 0<br>- | 保留<br>保留 |

## 16.4.5.3 错误状态寄存器（ES）

### 16.4.5.3.1 偏移

| 注册 | 偏移 |
|---|---|
| ĀĀĀ | 4h |

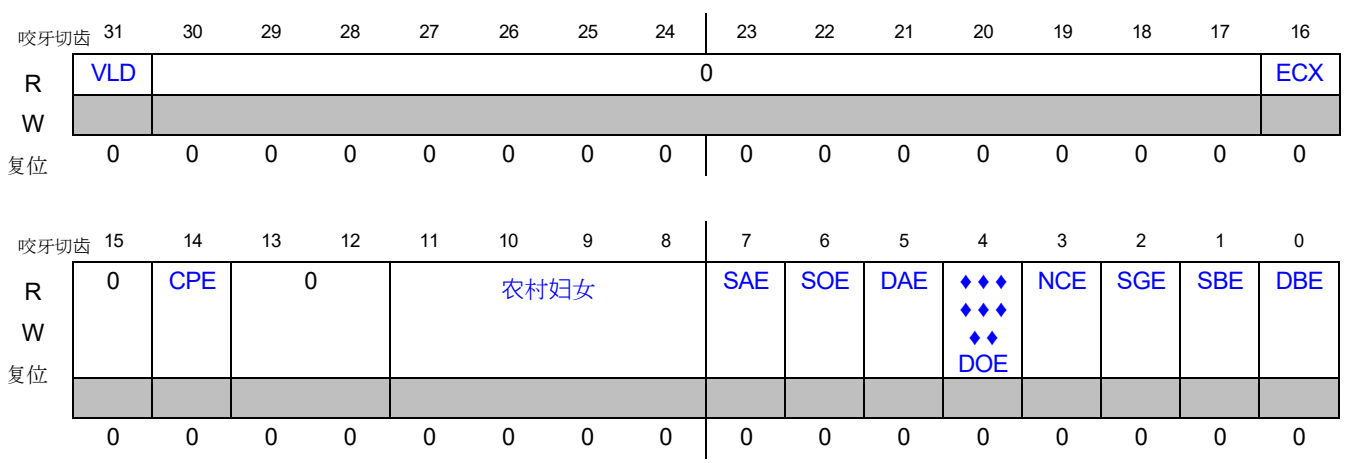
### 16.4.5.3.2 职能

ES提供有关最后记录的通道错误的信息。通道错误可能是由以下原因造成的。

- 一个配置错误，就是这样。
  - 传输控制描述符中的一个非法设置，或
  - 固定仲裁中的一个非法优先权寄存器设置
- 一个错误终止于总线主站的读或写周期
- 一个带错误的取消传输位，当通过相应的取消传输控制位取消传输时，该位将被设置。

更多细节见故障报告和处理。

### 16.4.5.3.3 图示




### 16.4.5.3.4 领域

| 场地 | 职能 |
|---|---|

| | |
|---|---|
| 31<br>VLD | VLD<br>所有ERR状态位的逻辑OR |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | 0b - 没有设置ERR位。<br>1b - 至少有一个ERR位被设置，表明存在一个有效的错误，并且没有被清除。 |
| 30-17<br>- | 保留 |
| 16<br>ECX | 转让取消了<br>0b - 没有取消的转让<br>1b - 最后记录的条目是由错误的取消转让输入的取消转让。 |
| 15<br>- | 保留 |
| 14<br>CPE | 通道优先级错误<br>0b - 没有通道优先级错误<br>1b - 最后记录的错误是通道优先级的配置错误。通道优先级不是唯一的。 |
| 13-12<br>- | 保留 |
| 11-8<br>农村妇女 | 错误的频道号码或取消的频道号码<br>最后记录的错误的通道号码，不包括CPE错误，或最后记录的错误取消的传输。 |
| 7<br>SAE | 源地址错误<br>0b - 没有源地址配置错误。<br>1b - 最后记录的错误是在TCDn_SADDR字段中检测到的配置错误。TCDn_SADDR与TCDn_ATTR[SSIZE]不一致。 |
| 6<br>SOE | 来源偏移错误<br>0b - 没有源偏移配置错误<br>1b - 最后记录的错误是在TCDn_SOFF字段中检测到的配置错误。TCDn_SOFF与TCDn_ATTR[SSIZE]不一致。 |
| 5<br>DAE | 目的地地址错误<br>0b - 没有目的地地址配置错误<br>1b - 最后记录的错误是在TCDn_DADDR字段中检测到的配置错误。TCDn_DADDR与TCDn_ATTR[DSIZE]不一致。 |
| 4<br>♦♦♦♦♦♦♦♦<br>DOE | 目的地偏移错误<br>0b - 没有目的地偏移配置错误<br>1b - 最后记录的错误是在TCDn_DOFF字段中检测到的配置错误。TCDn_DOFF与TCDn_ATTR[DSIZE]不一致。 |
| 3<br>NCE | NBYTES/CITER配置错误<br>0b - 没有NBYTES/CITER配置错误<br>1b - 最后记录的错误是在TCDn_NBYTES或TCDn_CITER字段中检测到的配置错误。TCDn_NBYTES不是TCDn_ATTR[SSIZE]和TCDn_ATTR[DSIZE]的倍数，或者TCDn_CITER[CITER]等于零，或者TCDn_CITER[ELINK]不等于TCDn_BITER[ELINK] |
| 2<br>SGE | 散射/聚集配置错误<br>0b - 没有散射/聚集配置错误<br>1b - 最后记录的错误是在 TCDn_DLASTSGA 字段中检测到的配置错误。如果 TCDn_CSR[ESG]被启用，则在主要循环完成后的散射/收集操作开始时检查该字段。TCDn_DLASTSGA 不在 32 字节边 |

| | |
|---|---|
| | 界上。 |
| 1<br>SBE | 源总线错误<br>0b - 没有源总线错误<br>1b - 最后记录的错误是源读的总线错误 |
| 0 | 目的地总线错误<br>0b - 没有目的地总线错误 |

| 场地 | 职能 |
| --- | --- |
| DBE | 1b - 最后记录的错误是目的地写入时的总线错误 |

## 16.4.5.4 启用请求寄存器（ERQ）

### 16.4.5.4.1 偏移

| 注册 | 偏移 |
| --- | --- |
| ERQ | 丘吉尔 |

### 16.4.5.4.2 职能

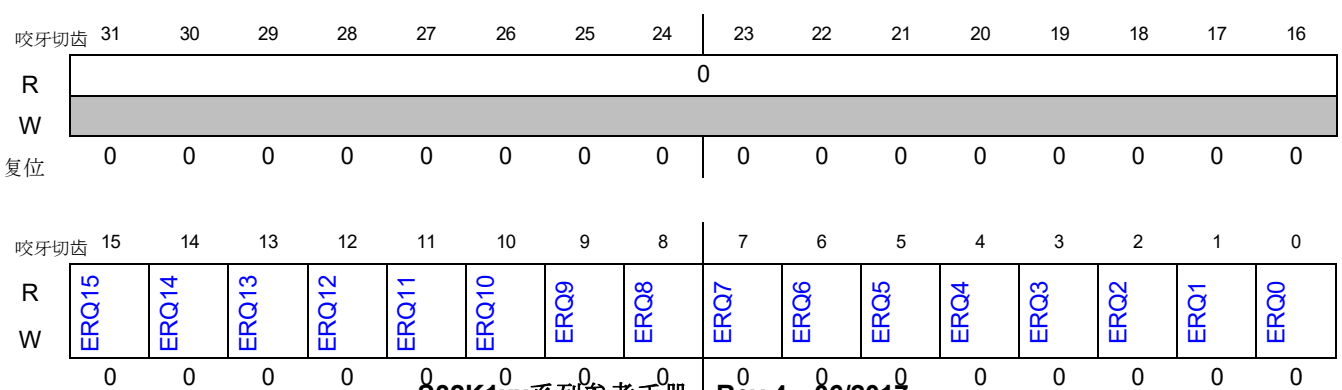
ERQ寄存器为16个通道提供一个位图，使每个通道的请求信号生效。任何给定的通道使能状态直接受写到这个寄存器的影响；它也受写到SERQ和CERQ寄存器的影响。提供这些寄存器是为了使单个通道的请求使能可以很容易地被修改，而不需要对这个寄存器进行读-修改-写的顺序。

在一个通道的硬件服务请求被接受之前，DMA请求输入信号和这个启用请求标志必须被断言。DMA使能请求标志的状态不影响通过软件明确提出的通道服务请求或链接通道请求。

**注意事项**

在清除通道的ERQ位之前，在源头上禁用一个通道的硬件服务请求。

### 16.4.5.4.3 图示



复位

### 16.4.5.4.4 领域

| 场地 | 职能 |
|---|---|
| 31-16<br>- | 保留 |
| 15<br>ERQ15 | 启用DMA请求 15<br>0b - 对应通道的DMA请求信号被禁用 1b - 对应通道的DMA请求信号被启用 |
| 14<br>ERQ14 | 启用DMA请求 14<br>0b - 对应通道的DMA请求信号被禁用 1b - 对应通道的DMA请求信号被启用 |
| 13<br>ERQ13 | 启用DMA请求 13<br>0b - 对应通道的DMA请求信号被禁用 1b - 对应通道的DMA请求信号被启用 |
| 12<br>ERQ12 | 启用DMA请求 12<br>0b - 对应通道的DMA请求信号被禁用 1b - 对应通道的DMA请求信号被启用 |
| 11<br>ERQ11 | 启用DMA请求 11<br>0b - 对应通道的DMA请求信号被禁用 1b - 对应通道的DMA请求信号被启用 |
| 10<br>ERQ10 | 启用DMA请求 10<br>0b - 对应通道的DMA请求信号被禁用 1b - 对应通道的DMA请求信号被启用 |
| 9<br>ERQ9 | 启用DMA请求 9<br>0b - 对应通道的DMA请求信号被禁用 1b - 对应通道的DMA请求信号被启用 |
| 8<br>ERQ8 | 启用DMA请求 8<br>0b - 对应通道的DMA请求信号被禁用 1b - 对应通道的DMA请求信号被启用 |
| 7<br>ERQ7 | 启用DMA请求7<br>0b - 对应通道的DMA请求信号被禁用 1b - 对应通道的DMA请求信号被启用 |
| 6<br>ERQ6 | 启用DMA请求6<br>0b - 对应通道的DMA请求信号被禁用 1b - 对应通道的DMA请求信号被启用 |
| 5<br>ERQ5 | 启用DMA请求5<br>0b - 对应通道的DMA请求信号被禁用 1b - 对应通道的DMA请求信号被启用 |
| 4<br>ERQ4 | 启用DMA请求4<br>0b - 对应通道的DMA请求信号被禁用 1b - 对应通道的DMA请求信号被启用 |

| | |
|---|---|
| 3<br>ERQ3 | 启用DMA请求3<br>0b - 对应通道的DMA请求信号被禁用 1b - 对应通道的DMA请求信号被启用 |
| 2<br>ERQ2 | 启用DMA请求2<br>0b - 对应通道的DMA请求信号被禁用 1b - 对应通道的DMA请求信号被启用 |
| 1 | 启用DMA请求1<br>0b - 相应通道的DMA请求信号被禁用。 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| ERQ1 | 1b - 相应通道的DMA请求信号被激活。 |
| 0<br>ERQ0 | 启用DMA请求0<br>0b - 对应通道的DMA请求信号被禁用 1b - 对应通道的DMA请求信号被启用 |

## 16.4.5.5 启用错误中断寄存器（EEI）

### 16.4.5.5.1 偏移

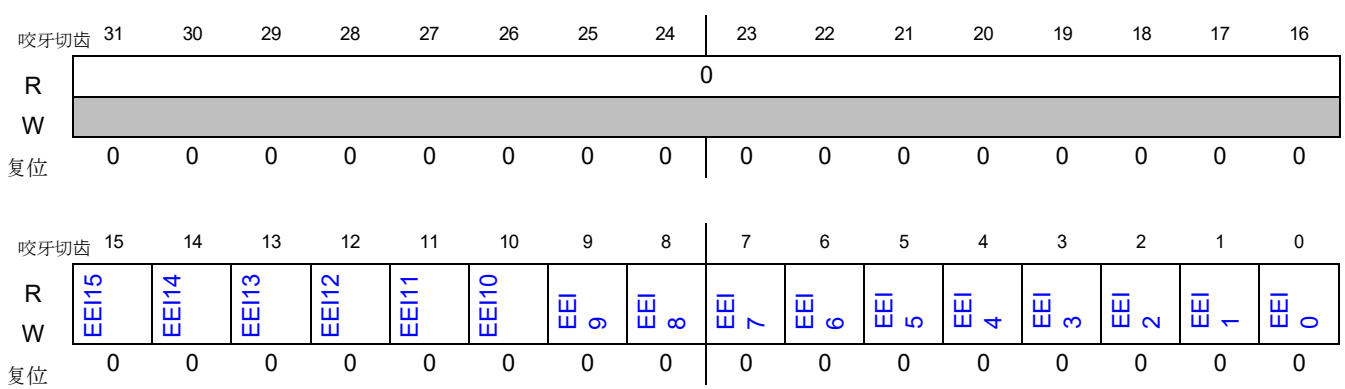
| 注册 | 偏移 |
|---|---|
| EEI | 14h |

### 16.4.5.5.2 职能

EEI寄存器为16个通道提供了一个位图，用于启用每个通道的错误中断信号。任何给定通道的错误中断使能状态直接受写入该寄存器的影响；它也受写入SEEI和CEEI的影响。提供这些寄存器是为了使单个通道的错误中断使能可以很容易地被修改，而不需要对EEI寄存器执行读-改-写的顺序。

DMA错误指示器和错误中断使能标志必须在一个给定通道的错误中断请求被断言到中断控制器之前被断言。

### 16.4.5.5.3 图示



### 16.4.5.5.4 领域

| 场地 | 职能 |
|---|---|
| 31-16<br>- | 保留 |
| 15<br>EEI15 | 启用错误中断 15<br>0b - 相应通道的错误信号不产生错误中断。<br>1b - 相应通道的错误信号的断言产生一个错误中断请求。 |
| 14<br>EEI14 | 启用错误中断 14<br>0b - 相应通道的错误信号不产生错误中断。<br>1b - 相应通道的错误信号的断言产生一个错误中断请求。 |
| 13<br>EEI13 | 启用错误中断 13<br>0b - 相应通道的错误信号不产生错误中断。<br>1b - 相应通道的错误信号的断言产生一个错误中断请求。 |
| 12<br>EEI12 | 启用错误中断 12<br>0b - 相应通道的错误信号不产生错误中断。<br>1b - 相应通道的错误信号的断言产生一个错误中断请求。 |
| 11<br>EEI11 | 启用错误中断 11<br>0b - 相应通道的错误信号不产生错误中断。<br>1b - 相应通道的错误信号的断言产生一个错误中断请求。 |
| 10<br>EEI10 | 启用错误中断 10<br>0b - 相应通道的错误信号不产生错误中断。<br>1b - 相应通道的错误信号的断言产生一个错误中断请求。 |
| 9<br>EEI9 | 启用错误中断 9<br>0b - 相应通道的错误信号不产生错误中断。<br>1b - 相应通道的错误信号的断言产生一个错误中断请求。 |
| 8<br>EEI8 | 启用错误中断 8<br>0b - 相应通道的错误信号不产生错误中断。<br>1b - 相应通道的错误信号的断言产生一个错误中断请求。 |
| 7<br>EEI7 | 启用错误中断7<br>0b - 相应通道的错误信号不产生错误中断。<br>1b - 相应通道的错误信号的断言产生一个错误中断请求。 |
| 6<br>EEI6 | 启用错误中断6<br>0b - 相应通道的错误信号不产生错误中断。<br>1b - 相应通道的错误信号的断言产生一个错误中断请求。 |
| 5<br>EEI5 | 启用错误中断5<br>0b - 相应通道的错误信号不产生错误中断。<br>1b - 相应通道的错误信号的断言产生一个错误中断请求。 |
| 4<br>EEI4 | 启用错误中断4<br>0b - 相应通道的错误信号不产生错误中断。 |

| | |
|---|---|
| | 1b - 相应通道的错误信号的断言产生一个错误中断请求。 |
| 3<br>EEI3 | 启用错误中断3<br>0b - 相应通道的错误信号不产生错误中断。<br>1b - 相应通道的错误信号的断言产生一个错误中断请求。 |
| 2<br>EEI2 | 启用错误中断2<br>0b - 相应通道的错误信号不产生错误中断。<br>1b - 相应通道的错误信号的断言产生一个错误中断请求。 |
| 1<br>EEI1 | 启用错误中断1<br>0b - 相应通道的错误信号不产生错误中断。<br>1b - 相应通道的错误信号的断言产生一个错误中断请求。 |
| 0 | 启用错误中断 0 |

| 场地 | 职能 |
|---|---|
| EEI0 | 0b - 相应通道的错误信号不产生错误中断。<br>1b - 相应通道的错误信号的断言产生一个错误中断请求。 |

## 16.4.5.6 清除启用错误中断寄存器（CEEI）。

### 16.4.5.6.1 偏移

| 注册 | 偏移 |
|---|---|
| CEEI | 18h |

### 16.4.5.6.2 职能

CEEI提供了一个简单的内存映射机制来清除EEI中的一个特定位，以禁用一个特定通道的错误中断。寄存器写入的数据值导致EEI中的相应位被清除。设置CAEE位提供了一个全局清除功能，迫使EEI内容被清除，禁用所有DMA请求输入。

如果NOP位被设置，该命令将被忽略。这允许你用32位的写操作来设置一个单一的、字节范围的寄存器，而不影响写操作中的其他寄存器。在这种情况下，字的其他三个字节都会被设置为NOP位，这样这些寄存器就不会受到写入的影响。

读取该寄存器返回所有的零。

### 16.4.5.6.3 图示

| 咬牙切齿 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | | | 0 | | | |
| W | NOP | CAEE | 0 | | CEEI | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

### 16.4.5.6.4 领域

| 场地 | 职能 |
|---|---|

| | |
|---|---|
| 7<br>NOP | 不启用操作<br>0b - 正常操作 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| | 1b - 无操作，忽略该寄存器中的其他位 |
| 6<br>CAEE | 清除所有启用错误的中断<br>0b - 只清除CEEI字段中指定的EEI位 1b - 清除EEI中的所有位 |
| 5-4<br>- | 保留 |
| 3-0<br>CEEI | 清除启用错误中断<br>清除EEI中的相应位 |

## 16.4.5.7 设置启用错误中断寄存器（SEEI）。

### 16.4.5.7.1 偏移

| 注册 | 偏移 |
|---|---|
| SEEI | 19h |

### 16.4.5.7.2 职能

SEEI提供了一个简单的内存映射机制来设置EEI中的一个特定位，以启用一个特定通道的错误中断。寄存器写入的数据值导致EEI中的相应位被设置。设置SAEE位提供了一个全局设置功能，迫使整个EEI内容被设置。

如果NOP位被设置，该命令将被忽略。这允许你用32位的写操作来设置一个单一的、字节范围的寄存器，而不影响写操作中的其他寄存器。在这种情况下，字的其他三个字节都会被设置为NOP位，这样这些寄存器就不会受到写入的影响。

读取该寄存器返回所有的零。

### 16.4.5.7.3 图示

| 咬牙切齿 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | | | 0 | | | |
| W | NOP | SAEE | 0 | | SE | EI | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

#### 16.4.5.7.4 领域

| 场地 | 职能 |
|---|---|
| 7<br>NOP | 不启用操作<br>0b - 正常操作<br>1b - 无操作，忽略该寄存器中的其他位 |
| 6<br>SAEE | 设置所有启用错误中断<br>0b - 只设置SEEI字段中指定的EEI位。1b - 设置EEI中的所有位 |
| 5-4<br>- | 保留 |
| 3-0<br>SEEI | 设置启用错误中断<br>设置EEI中的相应位 |

### 16.4.5.8 清除启用请求寄存器（CERQ）

#### 16.4.5.8.1 偏移

| 注册 | 偏移 |
|---|---|
| CERQ | 1Ah |

#### 16.4.5.8.2 职能

CERQ提供了一个简单的内存映射机制，以清除ERQ中的特定位，禁用特定通道的DMA请求。寄存器写入的数据值导致ERQ中的相应位被清空。设置CAER位提供了一个全局清除功能，迫使ERQ的全部内容被清除，禁用所有DMA请求输入。

如果NOP位被设置，该命令将被忽略。这允许你用32位的写操作来设置一个单一的、字节范围的寄存器，而不影响写操作中的其他寄存器。在这种情况下，字的其他三个字节都会被设置为NOP位，这样这些寄存器就不会受到写入的影响。

读取该寄存器返回所有的零。

**注意事项**

在清除通道的ERQ位之前，在源头上禁用一个通道的硬件服务请求。

### 16.4.5.8.3 图示

| 咬牙切齿 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | | | 0 | | | |
| W | NOP | CAER | 0 | | CERQ | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

### 16.4.5.8.4 领域

| 场地 | 职能 |
|---|---|
| 7<br>NOP | 不启用操作<br>0b - 正常操作<br>1b - 无操作，忽略该寄存器中的其他位 |
| 6<br>CAER | 清除所有启用请求<br>0b - 只清除CERQ字段中指定的ERQ位 1b - 清除ERQ中的所有位 |
| 5-4<br>- | 保留 |
| 3-0<br>CERQ | 清除启用请求<br>清除ERQ中的相应位。 |

## 16.4.5.9 设置启用请求寄存器（SERQ）

### 16.4.5.9.1 偏移

| 注册 | 偏移 |
|---|---|
| サイトからは、それがあります。 | 1Bh |

### 16.4.5.9.2 职能

SERQ提供了一个简单的内存映射机制来设置ERQ中的一个特定位，以启用一个特定通道的DMA请求。寄存器写入的数据值导致ERQ中的相应位被设置。设置SAE

R位提供了一个全局设置功能，迫使ERQ的全部内容被设置。

如果NOP位被设置，该命令将被忽略。这允许你用32位的写操作来设置一个单一的、字节范围的寄存器，而不影响写操作中的其他寄存器。在这种情况下，字的其他三个字节都会被设置为NOP位，这样这些寄存器就不会受到写入的影响。

读取该寄存器返回所有的零。

### 16.4.5.9.3 图示

| 咬牙切齿 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | | | 0 | | | |
| W | NOP | SAER | 0 | | SERQ | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

### 16.4.5.9.4 领域

| 场地 | 职能 |
|---|---|
| 7<br>NOP | 不启用操作<br>0b - 正常操作<br>1b - 无操作，忽略该寄存器中的其他位 |
| 6<br>SAER | 设置所有启用请求<br>0b - 只设置SERQ字段中指定的ERQ位 1b - 设置ERQ中的所有位 |
| 5-4<br>- | 保留 |
| 3-0<br>サイトからは、それがあります。 | 设置启用请求<br>设置ERQ中的相应位。 |

## 16.4.5.10 清除DONE状态位寄存器（CDNE）。

### 16.4.5.10.1 偏移

| 注册 | 偏移 |
|---|---|
| CDNE | 1Ch |

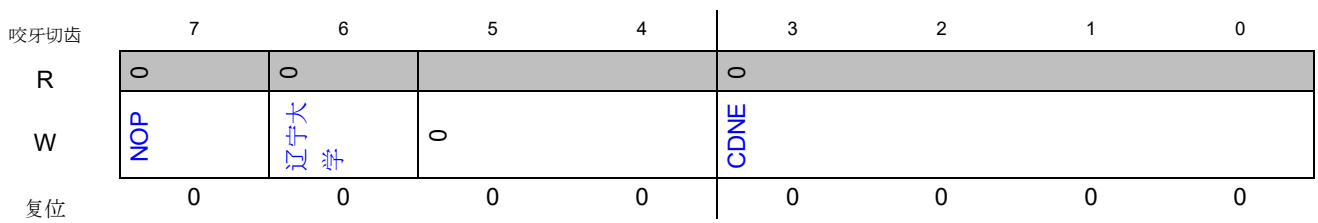
### 16.4.5.10.2 职能

CDNE提供了一个简单的内存映射机制来清除指定通道的TCD中的DONE位。寄存器写入的数据值导致相应的传输控制描述符中的DONE位被清空。设置CADN位提供了一个全局清除功能，迫使所有DONE位被清除。

如果NOP位被设置，该命令将被忽略。这允许你用32位的写操作来设置一个单一的、字节范围的寄存器，而不影响写操作中的其他寄存器。在这种情况下，字的其他三个字节都会被设置为NOP位，这样这些寄存器就不会受到写入的影响。

读取该寄存器返回所有的零。

### 16.4.5.10.3 图示




### 16.4.5.10.4 领域

| 场地 | 职能 |
|---|---|
| 7<br>NOP | 不启用操作<br>0b - 正常操作<br>1b - 无操作，忽略该寄存器中的其他位 |
| 6<br>辽宁大学 | 清除所有DONE位<br>0b - 只清除 CDNE 字段中指定的 TCDn_CSR[DONE] 位 1b - 清除 TCDn_CSR[DONE] 中的所有位 |
| 5-4<br>- | 保留 |
| 3-0<br>CDNE | 清除DONE位<br>清除TCDn_CSR[DONE]中的相应位。 |

## 16.4.5.11 设置START位寄存器（SSRT）

### 16.4.5.11.1 偏移

| 注册 | 偏移 |
|---|---|
| SSRT | 1Dh |

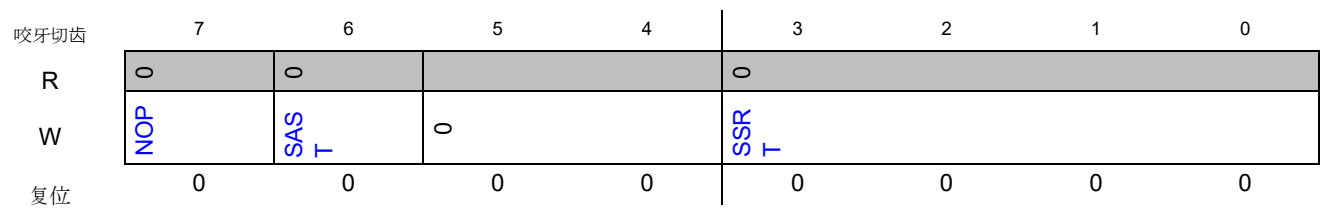
### 16.4.5.11.2 职能

SSRT提供了一个简单的内存映射机制来设置指定通道的TCD中的START位。寄存器写入的数据值导致相应的传输控制描述符中的START位被设置。设置SAST位提供了一个全局设置功能，迫使所有START位被设置。

如果NOP位被设置，该命令将被忽略。这允许你用32位的写操作来设置一个单一的、字节范围的寄存器，而不影响写操作中的其他寄存器。在这种情况下，字的其他三个字节都会被设置为NOP位，这样这些寄存器就不会受到写入的影响。

读取该寄存器返回所有的零。

### 16.4.5.11.3 图示




### 16.4.5.11.4 领域

| 场地 | 职能 |
|---|---|
| 7<br>NOP | 不启用操作<br>0b - 正常操作<br>1b - 无操作，忽略该寄存器中的其他位 |
| 6<br>SAST | 设置所有START位（激活所有通道）<br>0b - 只设置 SSRT 字段中指定的 TCDn_CSR[START] 位 1b - 设置 TCDn_CSR[START] 中的所有位 |
| 5-4<br>- | 保留 |
| 3-0<br>SSRT | 设置START位<br>设置 TCDn_CSR[START]中的相应位。 |

## 16.4.5.12 清除错误寄存器（CERR）。

### 16.4.5.12.1 偏移

| 注册 | 偏移 |
|---|---|
| CERR | 1Eh |

### 16.4.5.12.2 职能

CERR提供了一个简单的内存映射机制来清除ERR中的一个给定位，以禁用一个给定通道的错误条件标志。寄存器写入的给定值会导致ERR中的相应位被清除。设置CAEI位提供全局清除功能，迫使ERR内容被清除，清除所有通道错误指示。如果NOP位被设置，该命令被忽略。这允许你把多字节的寄存器写成一个32位的字。对该寄存器的读取会返回所有的零。

### 16.4.5.12.3 图示

| 咬牙切齿 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | | | 0 | | | |
| W | NOP | CAEI | 0 | | CERR | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

### 16.4.5.12.4 领域

| 场地 | 职能 |
|---|---|
| 7<br>NOP | 不启用操作<br>0b - 正常操作<br>1b - 无操作，忽略该寄存器中的其他位 |
| 6<br>CAEI | 清除所有错误指示器<br>0b - 只清除CERR字段中指定的ERR位 1b - 清除ERR中的所有位 |
| 5-4<br>- | 保留 |

| | |
|---|---|
| 3-0<br>CERR | 清除错误指示器<br><br>清除ERR中的相应位 |

## 16.4.5.13 清除中断请求寄存器（CINT）。

### 16.4.5.13.1 偏移

| 注册 | 偏移 |
|---|---|
| CINT | 1Fh |

### 16.4.5.13.2 职能

CINT提供了一个简单的、内存映射的机制来清除INT中的一个给定的位，以禁用一个给定通道的中断请求。寄存器写入的给定值导致INT中的相应位被清除。设置CAIR位提供了一个全局清除功能，迫使INT的全部内容被清除，禁用所有DMA中断请求。

如果NOP位被设置，该命令将被忽略。这允许你用32位的写操作来设置一个单一的、字节范围的寄存器，而不影响写操作中的其他寄存器。在这种情况下，字的其他三个字节都会被设置为NOP位，这样这些寄存器就不会受到写入的影响。

读取该寄存器返回所有的零。

### 16.4.5.13.3 图示

| 咬牙切齿 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | | | 0 | | | |
| W | NOP | CAIR | 0 | | CINT | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

### 16.4.5.13.4 领域

| 场地 | 职能 |
|---|---|
| 7<br>NOP | 不启用操作<br>0b - 正常操作<br>1b - 无操作，忽略该寄存器中的其他位 |

| | |
|---|---|
| 6<br>CAIR | 清除所有中断请求<br>0b - 只清除CINT字段中指定的INT位 1b - 清除INT中的所有位 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| 5-4<br>- | 保留 |
| 3-0<br>CINT | 清除中断请求<br>清除INT中的相应位 |

## 16.4.5.14 中断请求寄存器（INT）

### 16.4.5.14.1 偏移

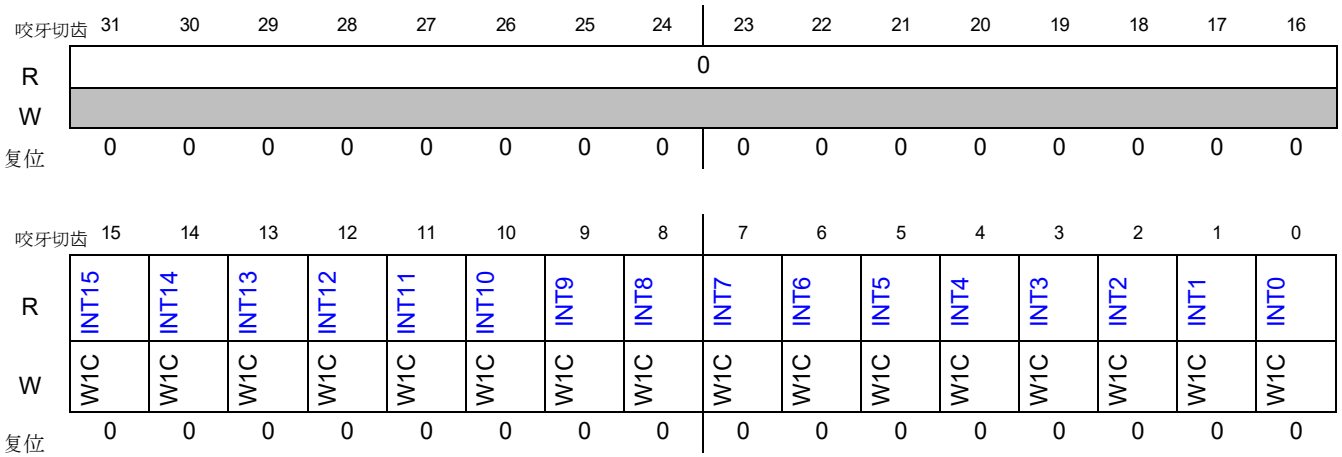
| 注册 | 偏移 |
| --- | --- |
| INT | 24h |

### 16.4.5.14.2 职能

INT寄存器提供了16个通道的位图，标志着每个通道的中断请求的存在。根据传输控制描述符中适当的位设置，eDMA引擎会在数据传输完成时产生一个中断。该寄存器的输出被直接路由到中断控制器。在与任何给定通道相关的中断服务程序中，软件有责任清除适当的位，否定中断请求。通常情况下，在中断服务例程中，对CINT寄存器的写入是用于这一目的。

任何一个通道的中断请求的状态直接受写到这个寄存器的影响；它也受写到CINT寄存器的影响。在写到INT的时候，任何位上的1都会清除相应通道的中断请求。任何位上的0都不会影响到相应通道的当前中断状态。提供CINT寄存器是为了方便清除单个通道的中断请求，而不需要对INT寄存器进行读-修改-写的顺序。

### 16.4.5.14.3 图示




### 16.4.5.14.4 领域

| 场地 | 职能 |
|---|---|
| 31-16<br>- | 保留 |
| 15<br>INT15 | 中断请求 15<br>0b - 相应通道的中断请求被清除 1b - 相应通道的中断请求被激活 |
| 14<br>INT14 | 中断请求 14<br>0b - 相应通道的中断请求被清除 1b - 相应通道的中断请求被激活 |
| 13<br>INT13 | 中断请求 13<br>0b - 相应通道的中断请求被清除 1b - 相应通道的中断请求被激活 |
| 12<br>INT12 | 中断请求 12<br>0b - 相应通道的中断请求被清除 1b - 相应通道的中断请求被激活 |
| 11<br>INT11 | 中断请求 11<br>0b - 相应通道的中断请求被清除 1b - 相应通道的中断请求被激活 |
| 10<br>INT10 | 中断请求 10<br>0b - 相应通道的中断请求被清除 1b - 相应通道的中断请求被激活 |
| 9<br>INT9 | 中断请求 9<br>0b - 相应通道的中断请求被清除 1b - 相应通道的中断请求被激活 |

| | |
|---|---|
| 8<br>INT8 | 中断请求 8<br>0b - 相应通道的中断请求被清除 1b - 相应通道的中断请求被激活 |
| 7<br>INT7 | 中断请求 7<br>0b - 相应通道的中断请求被清除 1b - 相应通道的中断请求被激活 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| 6<br>INT6 | 中断请求 6<br>0b - 相应通道的中断请求被清除 1b - 相应通道的中断请求被激活 |
| 5<br>INT5 | 中断请求5<br>0b - 相应通道的中断请求被清除 1b - 相应通道的中断请求被激活 |
| 4<br>INT4 | 中断请求4<br>0b - 相应通道的中断请求被清除 1b - 相应通道的中断请求被激活 |
| 3<br>INT3 | 中断请求3<br>0b - 相应通道的中断请求被清除 1b - 相应通道的中断请求被激活 |
| 2<br>INT2 | 中断请求2<br>0b - 相应通道的中断请求被清除 1b - 相应通道的中断请求被激活 |
| 1<br>INT1 | 中断请求1<br>0b - 相应通道的中断请求被清除 1b - 相应通道的中断请求被激活 |
| 0<br>INT0 | 中断请求0<br>0b - 相应通道的中断请求被清除 1b - 相应通道的中断请求被激活 |

## 16.4.5.15 错误寄存器（ERR）。

### 16.4.5.15.1 偏移

| 注册 | 偏移 |
| --- | --- |
| 紧急情况 | 2Ch |

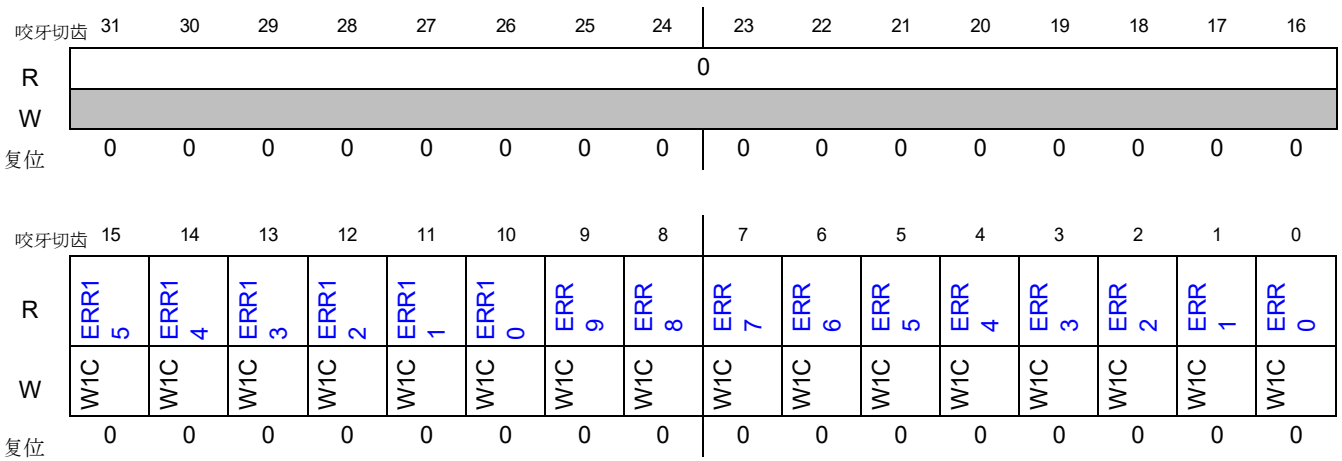
### 16.4.5.15.2 职能

ERR寄存器提供了16个通道的位图，标志着每个通道的错误存在。eDMA引擎通过设置该寄存器中相应的位来提示错误情况的发生。该寄存器的输出被EEI寄存器的内容所启用，然后被路由到中断控制器。在执行与任何DMA错误相关的中断服务程序时，软件有责任清除相应的位，否定错误中断请求。通常情况下，在中断服

务例程中对CERR的写入是用于这一目的。当检测到错误时，正常的DMA通道完成指标（设置传输控制描述符DONE标志和可能的中断请求断言）不受影响。

这个寄存器的内容也可以被轮询，因为非零值表示通道错误的存在，与EEI字段的状态无关。任何一个通道的错误指示器的状态都会受到写到这个寄存器的影响；它也会受到写到CERR的影响。在写到ERR时，任何位上的1都会清除相应通道的错误状态。任何位上的零对相应通道的当前错误状态没有影响。提供CERR是为了使单个通道的错误指示器能够容易地被清除。

### 16.4.5.15.3 图示




### 16.4.5.15.4 领域

| 场地 | 职能 |
|---|---|
| 31-16<br>- | 保留 |
| 15<br>ERR15 | 通道15中的错误<br>0b - 这个通道没有发生错误 1b - 这个通道发生了错误 |
| 14<br>ERR14 | 通道14中的错误<br>0b - 这个通道没有发生错误 1b - 这个通道发生了错误 |
| 13<br>ERR13 | 频道13中的错误<br>0b - 这个通道没有发生错误 1b - 这个通道发生了错误 |
| 12<br>ERR12 | 通道12中的错误<br>0b - 这个通道没有发生错误 1b - 这个通道发生了错误 |

| | |
|---|---|
| 11<br>ERR11 | 通道11中的错误<br>0b - 这个通道没有发生错误 1b - 这个通道发生了错误 |
| 10 | 通道10中的错误 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| ERR10 | 0b - 这个通道没有发生错误 1b - 这个通道发生了错误 |
| 9<br>ERR9 | 频道9中的错误<br>0b - 这个通道没有发生错误 1b - 这个通道发生了错误 |
| 8<br>ERR8 | 通道8中的错误<br>0b - 这个通道没有发生错误 1b - 这个通道发生了错误 |
| 7<br>ERR7 | 通道7中的错误<br>0b - 这个通道没有发生错误 1b - 这个通道发生了错误 |
| 6<br>ERR6 | 通道6中的错误<br>0b - 这个通道没有发生错误 1b - 这个通道发生了错误 |
| 5<br>ERR5 | 频道5中的错误<br>0b - 这个通道没有发生错误 1b - 这个通道发生了错误 |
| 4<br>ERR4 | 第4频道中的错误<br>0b - 这个通道没有发生错误 1b - 这个通道发生了错误 |
| 3<br>ERR3 | 通道3中的错误<br>0b - 这个通道没有发生错误 1b - 这个通道发生了错误 |
| 2<br>ERR2 | 通道2中的错误<br>0b - 这个通道没有发生错误 1b - 这个通道发生了错误 |
| 1<br>ERR1 | 通道1中的错误<br>0b - 这个通道没有发生错误 1b - 这个通道发生了错误 |
| 0<br>ERR0 | 通道0中的错误<br>0b - 这个通道没有发生错误 1b - 这个通道发生了错误 |

## 16.4.5.16 硬件请求状态寄存器（HRS）

## 16.4.5.16.1 偏移

| 注册 | 偏移 |
| --- | --- |
| HRS | 34h |

### 16.4.5.16.2 职能

HRS寄存器为DMA通道提供了一个位图，标志着每个通道的硬件请求的存在。硬件请求状态位反映了寄存器的当前状态和DMA的仲裁逻辑所看到的合格的（通过ERQ字段）DMA请求信号。这种对硬件请求信号的看法可用于调试目的。

**注意事项**

这些位反映了仲裁逻辑所看到的请求的状态。因此，该状态受到ERQ位的影响。

### 16.4.5.16.3 图示

| 咬牙切齿 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 咬牙切齿 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | HRS15 | HRS14 | HRS13 | HRS12 | HRS11 | HRS10 | HRS 9 | HRS 8 | HRS 7 | HRS 6 | HRS 5 | HRS 4 | HRS 3 | HRS 2 | HRS 1 | HRS 0 |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

### 16.4.5.16.4 领域

| 场地 | 职能 |
|---|---|
| 31-16<br>- | 保留 |
| 15<br>HRS15 | 硬件请求状态通道15<br>当通道上有硬件请求时，其各自的通道的HRS位在这段时间内一直处于正向状态。在请求完成和通道自由之后，HRS位被硬件自动清除。<br>0b - 不存在通道15的硬件服务请求 1b - 存在通道15的硬件服务请求 |
| 14<br>HRS14 | 硬件请求状态通道14<br>当通道上有硬件请求时，其各自的通道的HRS位在这段时间内保持正向状态。在请求完成后，通道自由了，HRS位被硬件自动清除。<br>0b - 不存在通道14的硬件服务请求 1b - 存在通道14的硬件服务 |

| | |
|---|---|
| | 请求 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| 13<br>HRS13 | 硬件请求状态通道13<br>当通道上有硬件请求时，其各自的通道的HRS位在这段时间内一直处于正向状态。在请求完成和通道自由之后，HRS位被硬件自动清除。<br>0b - 通道13的硬件服务请求不存在 1b - 通道13的硬件服务请求存在 |
| 12<br>HRS12 | 硬件请求状态通道12<br>当通道上有硬件请求时，其各自的通道的HRS位在这段时间内保持正向状态。在请求完成后，通道自由了，HRS位被硬件自动清除。<br>0b - 通道12的硬件服务请求不存在 1b - 通道12的硬件服务请求存在 |
| 11<br>HRS11 | 硬件请求状态通道11<br>当通道上有硬件请求时，其各自的通道的HRS位在这段时间内保持正向状态。在请求完成后，通道自由了，HRS位被硬件自动清除。<br>0b - 不存在通道11的硬件服务请求 1b - 存在通道11的硬件服务请求 |
| 10<br>HRS10 | 硬件请求状态通道10<br>当通道上有硬件请求时，其各自的通道的HRS位在这段时间内一直处于正向状态。在请求完成和通道自由之后，HRS位被硬件自动清除。<br>0b - 通道10的硬件服务请求不存在 1b - 通道10的硬件服务请求存在 |
| 9<br>HRS9 | 硬件请求状态通道9<br>当通道上有硬件请求时，其各自的通道的HRS位在这段时间内一直处于正向状态。在请求完成和通道自由之后，HRS位被硬件自动清除。<br>0b - 通道9的硬件服务请求不存在 1b - 通道9的硬件服务请求存在 |
| 8<br>HRS8 | 硬件请求状态通道8<br>当通道上有硬件请求时，其各自的通道的HRS位在这段时间内一直处于正向状态。在请求完成和通道自由之后，HRS位被硬件自动清除。<br>0b - 不存在通道8的硬件服务请求 1b - 存在通道8的硬件服务请求 |
| 7<br>HRS7 | 硬件请求状态通道7<br>当通道上有硬件请求时，其各自的通道的HRS位在这段时间内保持正向状态。在请求完成后，通道自由了，HRS位被硬件自动清除。<br>0b - 不存在通道7的硬件服务请求 1b - 存在通道7的硬件服务请求 |

| | |
|---|---|
| 6<br>HRS6 | 硬件请求状态通道6<br>当通道上有硬件请求时，其各自的通道的HRS位在这段时间内一直处于正向状态。在请求完成和通道自由之后，HRS位被硬件自动清除。<br>0b - 不存在通道6的硬件服务请求 1b - 存在通道6的硬件服务请求 |
| 5 | 硬件请求状态通道5 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| HRS5 | 当通道上有硬件请求时，其各自的通道的HRS位在这段时间内保持正向状态。在请求完成后，通道自由了，HRS位被硬件自动清除。<br>0b - 不存在通道5的硬件服务请求 1b - 存在通道5的硬件服务请求 |
| 4<br>HRS4 | 硬件请求状态通道4<br>当通道上有硬件请求时，其各自的通道的HRS位在这段时间内保持正向状态。在请求完成后，通道自由了，HRS位被硬件自动清除。<br>0b - 不存在通道4的硬件服务请求 1b - 存在通道4的硬件服务请求 |
| 3<br>HRS3 | 硬件请求状态通道3<br>当通道上有硬件请求时，其各自的通道的HRS位在这段时间内一直处于正向状态。在请求完成后，通道自由了，HRS位被硬件自动清除。<br>0b - 不存在通道3的硬件服务请求 1b - 存在通道3的硬件服务请求 |
| 2<br>HRS2 | 硬件请求状态通道2<br>当通道上有硬件请求时，其各自的通道的HRS位在这段时间内保持正向状态。在请求完成后，通道自由了，HRS位被硬件自动清除。<br>0b - 不存在通道2的硬件服务请求 1b - 存在通道2的硬件服务请求 |
| 1<br>HRS1 | 硬件请求状态通道1<br>当通道上有硬件请求时，其各自的通道的HRS位在这段时间内一直处于正向状态。在请求完成后，通道自由了，HRS位被硬件自动清除。<br>0b - 通道1的硬件服务请求不存在 1b - 通道1的硬件服务请求存在 |
| 0<br>HRS0 | 硬件请求状态通道0<br>当通道上有硬件请求时，其各自的通道的HRS位在这段时间内保持正向状态。在请求完成后，通道自由了，HRS位被硬件自动清除。<br>0b - 通道0的硬件服务请求不存在 1b - 通道0的硬件服务请求存在 |

## 16.4.5.17 启用停止寄存器中的异步请求（EARS）。

### 16.4.5.17.1 偏移

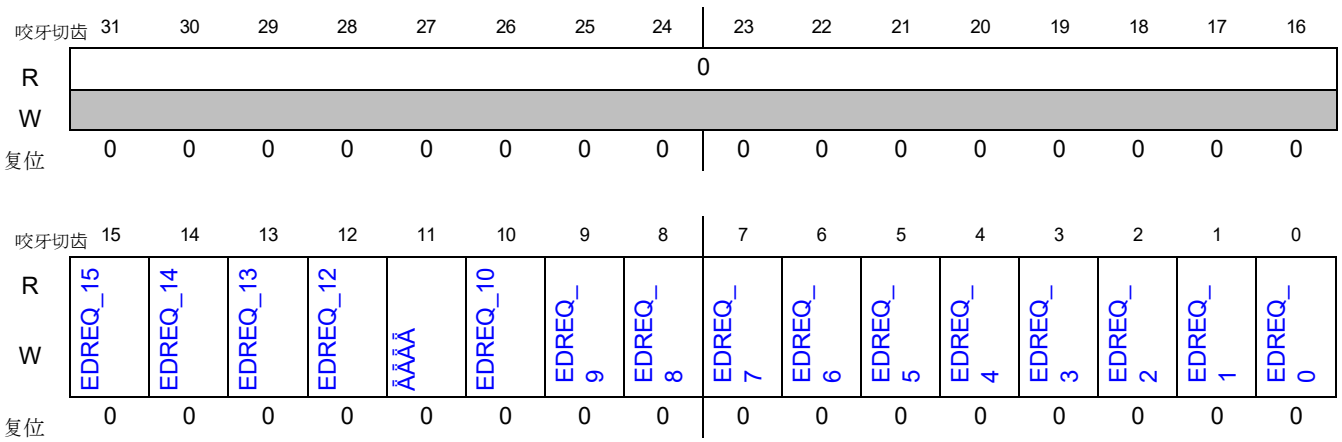
| 注册 | 偏移 |
| --- | --- |

| | |
|---|---|
| 耳朵 | 44h |

### 16.4.5.17.2 职能

EARS寄存器用于通过与这两个寄存器的位相加来启用或禁用启用请求寄存器（ERQ）中的DMA请求。

### 16.4.5.17.3 图示




### 16.4.5.17.4 领域

| 场地 | 职能 |
|---|---|
| 31-16<br>- | 保留 |
| 15<br>EDREQ_15 | 在停止模式下为通道15启用异步DMA请求 0b - 禁用通道15的异步DMA请求。1b - 启用通道15的异步DMA请求。 |
| 14<br>EDREQ_14 | 在停止模式下为通道14启用异步DMA请求 0b - 禁用通道14的异步DMA请求。1b - 启用通道14的异步DMA请求。 |
| 13<br>EDREQ_13 | 在停止模式下启用通道13的异步DMA请求 0b - 禁用通道13的异步DMA请求。1b - 启用通道13的异步DMA请求。 |
| 12<br>EDREQ_12 | 在停止模式下为通道12启用异步DMA请求 0b - 禁用通道12的异步DMA请求。1b - 启用通道12的异步DMA请求。 |
| 11<br>ÅÅÅÅ | 在停止模式下启用通道11的异步DMA请求 0b - 禁用通道11的异步DMA请求。1b - 启用通道11的异步DMA请求。 |
| 10<br>EDREQ_10 | 在停止模式下启用通道10的异步DMA请求 0b - 禁用通道10的异步DMA请求。1b - 启用通道10的异步DMA请求。 |
| 9<br>EDREQ_9 | 在停止模式下启用通道9的异步DMA请求 0b - 禁用通道9的异步DMA请求。1b - 启用通道9的异步DMA请求。 |
| 8 | 在停止模式下启用通道8的异步DMA请求 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| EDREQ_8 | 0b - 禁用通道8的异步DMA请求。1b - 启用通道8的异步DMA请求。 |
| 7<br>EDREQ_7 | 在停止模式下启用通道7的异步DMA请求 0b - 禁用通道7的异步DMA请求。1b - 启用通道7的异步DMA请求。 |
| 6<br>EDREQ_6 | 在停止模式下启用通道6的异步DMA请求 0b - 禁用通道6的异步DMA请求。1b - 启用通道6的异步DMA请求。 |
| 5<br>EDREQ_5 | 在停止模式下启用通道5的异步DMA请求 0b - 禁用通道5的异步DMA请求。1b - 启用通道5的异步DMA请求。 |
| 4<br>EDREQ_4 | 在停止模式下启用通道4的异步DMA请求 0b - 禁用通道4的异步DMA请求。1b - 启用通道4的异步DMA请求。 |
| 3<br>EDREQ_3 | 启用通道3的停止模式下的异步DMA请求。<br>0b - 禁用通道3的异步DMA请求。1b - 启用通道3的异步DMA请求。 |
| 2<br>EDREQ_2 | 启用通道2的停止模式下的异步DMA请求。<br>0b - 禁用通道2的异步DMA请求。1b - 启用通道2的异步DMA请求。 |
| 1<br>EDREQ_1 | 启用通道1的停止模式下的异步DMA请求。<br>0b - 禁用通道1的异步DMA请求 1b - 启用通道1的异步DMA请求。 |
| 0<br>EDREQ_0 | 启用通道0的停止模式下的异步DMA请求。<br>0b - 禁用通道0的异步DMA请求。1b - 启用通道0的异步DMA请求。 |

## 16.4.5.18 通道n优先级寄存器（DCHPRI3 - DCHPRI12）。

### 16.4.5.18.1 偏移

对于n=0到15。

| 注册 | 偏移 |
|---|---|
| DCHPRIn | FDh + (n × 1h) |

### 16.4.5.18.2 职能

当固定优先级的通道仲裁被启用时（CR[ERCA] =
0），这些寄存器的内容定义了与每个通道相关的唯一优先级。通道的优先级是通过数值来评估的；例如，0是最低的优先级，1是次高的优先级。

频道的优先级，然后是2，3，等等。软件必须用唯一的值对通道优先级进行编程，否则会报告一个配置错误。优先权值的范围被限制在0到15的值。

### 16.4.5.18.3 图示

| 咬牙切齿 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | ECP | DPA | 0 | | CHPRI | | | |
| W | | | | | | | | |
| 复位 | 见表16-4 。 | | | | | | | |

### 16.4.5.18.4 寄存器复位值

| 注册 | 复位 |
|---|---|
| DCHPRI3 | 03h |
| DCHPRI2 | 02h |
| DCHPRI1 | 01h |
| DCHPRI0 | 00h |
| DCHPRI7 | 07h |
| DCHPRI6 | 06h |
| DCHPRI5 | 05h |
| DCHPRI4 | 04h |
| DCHPRI11 | 0Bh |
| DCHPRI10 | 0Ah |
| DCHPRI9 | 09h |
| DCHPRI8 | 08h |
| DCHPRI15 | 0Fh |
| DCHPRI14 | 0Eh |
| DCHPRI13 | 0Dh |
| DCHPRI12 | 0Ch |

### 16.4.5.18.5 领域

| 场地 | 职能 |
|---|---|
| 7<br>ECP | 启用通道优先权。这个字段重设为0。<br>0b - 通道n不能被更高优先级通道的服务请求暂停。<br>1b - 通道n可以被更高优先级的通道的服务请求暂时中止。 |

| | |
|---|---|
| 6 | 禁用抢占能力。这个字段重设为0。<br>0b - 通道n可以暂停一个低优先级的通道。 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| DPA | 1b - 通道n不能暂停任何通道，无论通道的优先级如何。 |
| 5-4<br>- | 保留 |
| 3-0<br>CHPRI | 通道 n 仲裁优先权<br>启用固定优先级仲裁时的通道优先级<br>**注意：**通道优先级字段CHPRI的复位值等于每个优先级寄存器的相应通道号，即DCHPRI15[CHPRI]=0b1111。 |

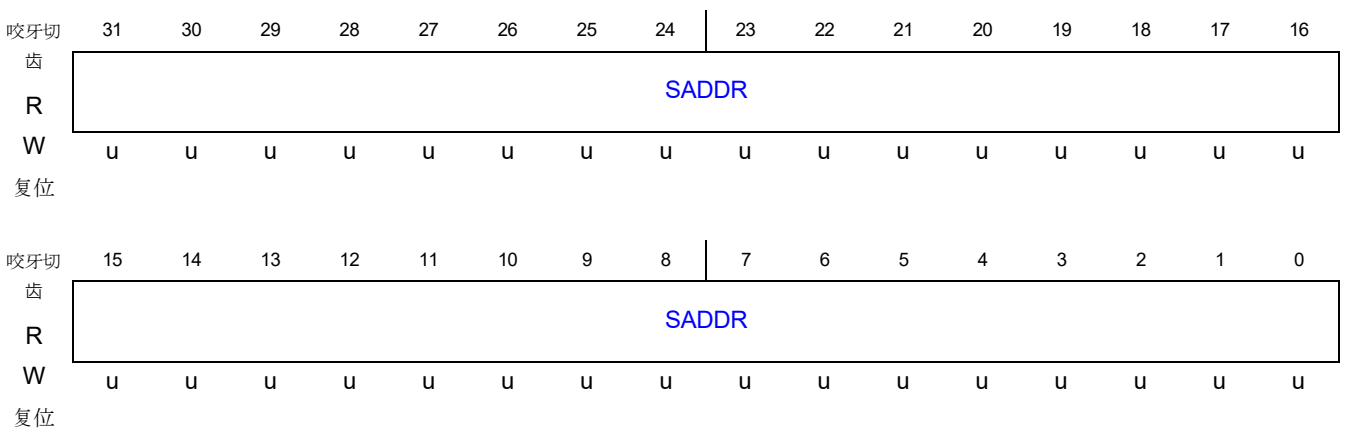
## 16.4.5.19 TCD源地址（TCD0_SADDR - TCD15_SADDR）。

### 16.4.5.19.1 偏移

对于n=0到15。

| 注册 | 偏移 |
|---|---|
| TCDn_SADDR | 1000h + (n × 20h) |

### 16.4.5.19.2 图示




### 16.4.5.19.3 领域

| 场地 | 职能 |
|---|---|

| | |
|---|---|
| 31-0<br>SADDR | 来源地址<br>指向源数据的内存地址。 |

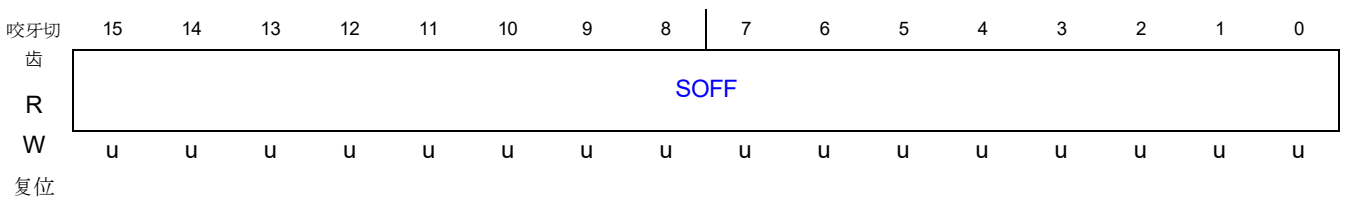
## 16.4.5.20 TCD有符号的源地址偏移（TCD0_SOFF - TCD15_ SOFF）。

### 16.4.5.20.1 偏移

对于n=0到15。

| 注册 | 偏移 |
|---|---|
| TCDn_SOFF | 1004h + (n × 20h) |

### 16.4.5.20.2 图示




### 16.4.5.20.3 领域

| 场地 | 职能 |
|---|---|
| 15-0<br>SOFF | 源地址带符号的偏移量<br>符号扩展的偏移量，应用于当前源地址，在每个源读取完成后形成下一状态值。 |

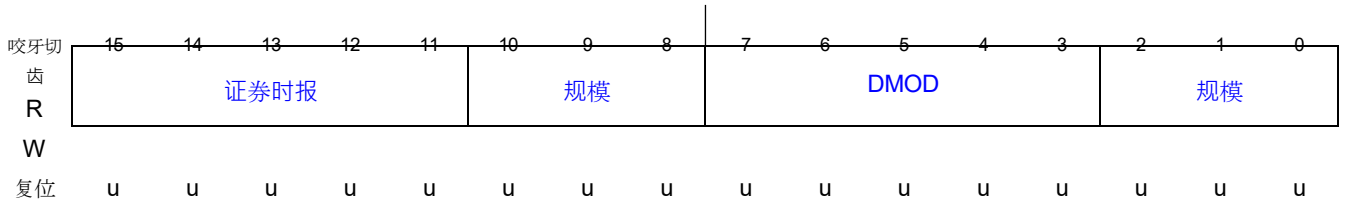
## 16.4.5.21 TCD传输属性（TCD0_ATTR - TCD15_ATTR）。

### 16.4.5.21.1 偏移

对于n=0到15。

| 注册 | 偏移 |
|---|---|
| TCDn_ATTR | 1006h + (n × 20h) |

### 16.4.5.21.2 图示




### 16.4.5.21.3 领域

| 场地 | 职能 |
|---|---|
| 15-11<br>证券时报 | 源地址模数<br>00000b - 源地址调制功能被禁用<br>00001-11111b - 这个值定义了一个特定的地址范围，指定为SADDR之后的值。<br>+ SOFF计算是在原始寄存器值上进行的。设置这个字段提供了轻松实现循环数据队列的能力。对于需要2次方大小字节的数据队列，队列应从0次方大小的地址开始，SMOD字段应设置为队列的适当值，冻结所需的上地址位数。编入该字段的值指定允许改变的下层地址位的数量。对于循环队列应用，SOFF通常被设置为传输大小，以实现增量后寻址，SMOD功能将地址限制在0-modulo-size范围。 |
| 10-8<br>规模 | 源数据传输大小<br>**注意：**使用一个保留值会导致配置错误。<br>000b - 8位 001b<br>- 16位 010b - 32<br>位 011b - 保留<br>100b - 16字节突发事<br>件 101b - 32字节突<br>发事件 110b - 保留<br>111b - 保留 |
| 7-3<br>DMOD | 目的地地址模数<br>见SMOD定义 |
| 2-0<br>规模 | 目的地数据传输大小<br>见SSIZE定义 |

## 16.4.5.22 TCD有符号的小环路偏移（小环路映射已启用，偏移已禁用）（TCD0_NBYTES_MLOFFNO - TCD15_NBYTES_MLOFFNO）。

#### 16.4.5.22.1 偏移

对于n=0到15。

| 注册 | 偏移 |
| --- | --- |
| TCDn_NBYTES_MLOFFNO | 1008h + (n × 20h) |

### 16.4.5.22.2 职能

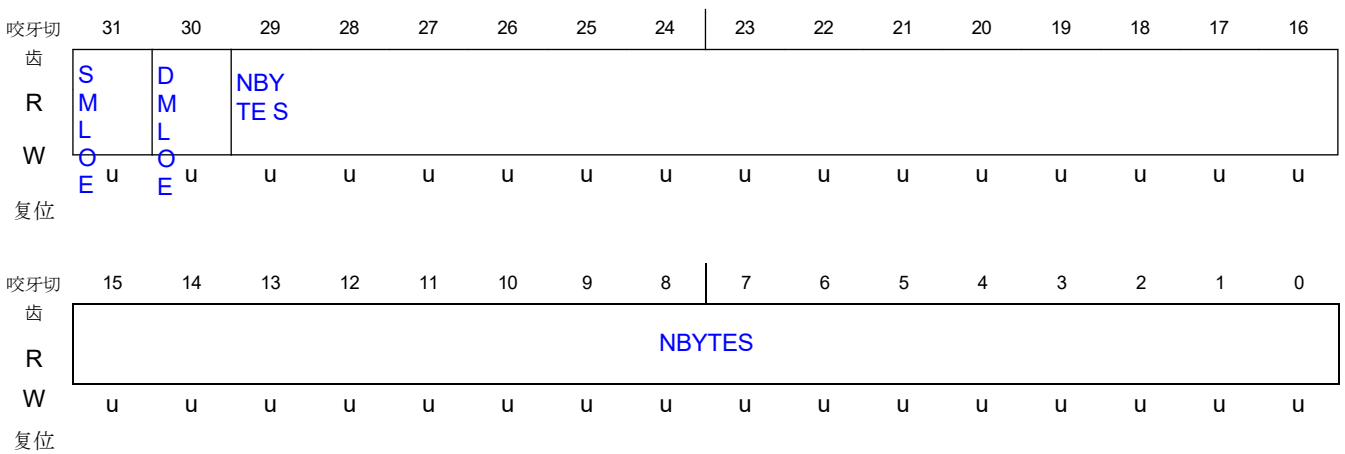
三个寄存器之一（这个寄存器，TCD_NBYTES_MLNO，或TCD_NBYTES_MLOFFYES），定义了每个请求要传输的字节数。使用哪个寄存器取决于小循环映射是否被禁用，启用但不用于该通道，还是启用并使用。

TCD字2的定义如下，如果。

- 小循环映射被启用（CR[EMLM] = 1），并且
- SMLOE = 0和DMLOE = 0

如果小循环映射被启用并且SMLOE或DMLOE被设置，那么请参考TCD_NBYTES_MLOFFYES寄存器的描述。如果小循环映射被禁用，那么请参考TCD_NBYTES_MLNO寄存器的描述。

### 16.4.5.22.3 图示




### 16.4.5.22.4 领域

| 场地 | 职能 |
| --- | --- |

| | |
|---|---|
| 31<br>SMLOE | 源小环路偏移启用<br>选择是否在小循环完成时将小循环偏移量应用于源地址。<br>0b - 小循环偏移不应用于SADDR 1b - 小循环偏移应用于SADDR |
| 30 | 启用目的地小环路偏移 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| DMLOE | 选择是否在小循环完成时将小循环偏移量应用于目标地址。<br>0b - 小循环偏移不应用于DADDR 1b - 小循环偏移应用于DADDR |
| 29-0<br>NBYTES | 小字节传输计数<br>通道的每个服务请求中要传输的字节数。<br>当一个通道激活时，适当的TCD内容加载到eDMA引擎中，适当的读和写执行，直到小字节传输计数转移完毕。这是一个不可分割的操作，不能被停止。然而，它可以通过使用带宽控制字段或通过抢占来停滞。在次要计数用完后，SADDR和DADDR值被写回TCD存储器，主要迭代计数被减去并恢复到TCD存储器中。如果主要迭代次数用完了，就会进行额外的处理。 |

## 16.4.5.23 TCD有符号的小环路偏移（小环路映射和偏移已启用）（TCD0_NBYTES_MLOFFYES - TCD15_NBYTES_MLOFFYES）。

### 16.4.5.23.1 偏移

对于n=0到15。

| 注册 | 偏移 |
| --- | --- |
| TCDn_NBYTES_MLOFFYES | 1008h + (n × 20h) |

### 16.4.5.23.2 职能

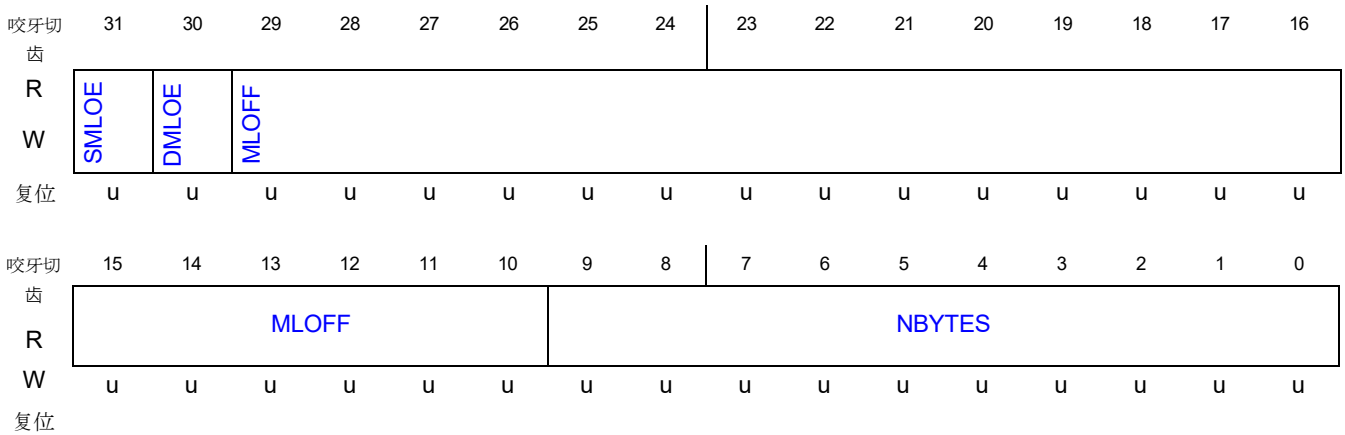
三个寄存器之一（这个寄存器，TCD_NBYTES_MLNO，或TCD_NBYTES_MLOFFNO），定义了每个请求要传输的字节数。使用哪个寄存器取决于小循环映射是否被禁用，启用但不用于该通道，还是启用并使用。

TCD字2的定义如下，如果。

- 小循环映射被启用（CR[EMLM] = 1），并且
- 小循环偏移被启用（SMLOE或DMLOE=1）。

如果小循环映射被启用并且SMLOE和DMLOE被清除，那么请参考TCD_NBYTES_MLOFFNO寄存器的描述。如果小循环映射被禁用，那么请参考TCD_NBYTES_MLNO寄存器的描述。

### 16.4.5.23.3 图示




### 16.4.5.23.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>SMLOE | 源小环路偏移启用<br>选择是否在小循环完成时将小循环偏移量应用于源地址。<br>0b - 小循环偏移不应用于SADDR 1b - 小循环偏移应用于SADDR |
| 30<br>DMLOE | 启用目的地小环路偏移<br>选择是否在小循环完成时将小循环偏移量应用于目标地址。<br>0b - 小循环偏移不应用于DADDR 1b - 小循环偏移应用于DADDR |
| 29-10<br>MLOFF | 如果设置了SMLOE或DMLOE，该字段表示在小循环完成后，应用于源地址或目的地址的符号扩展偏移，形成下一状态值。 |
| 9-0<br>NBYTES | 小字节传输计数<br>通道的每个服务请求中要传输的字节数。<br>当一个通道激活时，适当的TCD内容加载到eDMA引擎中，适当的读和写执行，直到小字节传输计数转移完毕。这是一个不可分割的操作，不能被停止。然而，它可以通过使用带宽控制字段或通过抢占来停滞。在次要计数用完后，SADDR和DADDR值被写回TCD存储器，主要迭代计数被减去并恢复到TCD存储器中。如果主要迭代次数用完了，就会进行额外的处理。 |

## 16.4.5.24 TCD小字节计数（小环路映射已禁用）（TCD0_NBYTES_MLNO - TCD15_NBYTES_MLNO）。

### 16.4.5.24.1 偏移

对于n=0到15。

| 注册 | 偏移 |
| --- | --- |
| TCDn_NBYTES_MLNO | 1008h + (n × 20h) |

### 16.4.5.24.2 职能

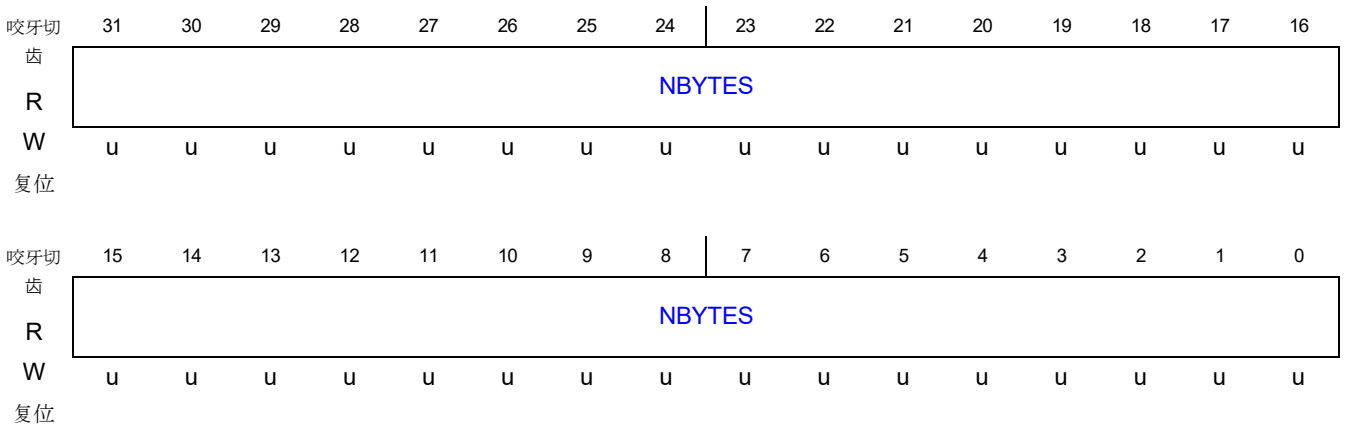
这个寄存器或后面两个寄存器之一（TCD_NBYTES_MLOFFNO, TCD_NBYTES_MLOFFYES），定义了每个请求要传输的字节数。使用哪个寄存器取决于小循环映射是否被禁用，是否被启用但不用于该通道，还是被启用并使用。

TCD字2的定义如下，如果。

- 小循环映射被禁用（CR[EMLM] = 0）。

如果启用了小循环映射，请参阅TCD_NBYTES_MLOFFNO和TCD_NBYTES_MLOFFYES寄存器的描述，了解TCD字2的定义。

### 16.4.5.24.3 图示




### 16.4.5.24.4 领域

| 场地 | 职能 |
| --- | --- |

| | |
|---|---|
| 31-0<br>NBYTES | 小字节传输计数<br>通道的每个服务请求中要传输的字节数。当一个通道激活时，适当的TCD内容加载到eDMA引擎中，适当的读和写执行，直到小字节传输计数传输完毕。这是一个不可分割的操作，不能被停止。然而，它可以通过使用带宽控制字段或通过抢占来停滞。在次要计数用完后，SADDR和DADDR值被写回TCD存储器，主要迭代计数被减去并恢复到TCD存储器中。如果主要迭代次数用完了，就会进行额外的处理。<br>**注意：**NBYTES值为0x0000_0000，被解释为4GB的传输。 |

## 16.4.5.25 TCD最后源地址调整（TCD0_SLAST - TCD15_SLAST）

。

### 16.4.5.25.1 偏移

对于n=0到15。

| 注册 | 偏移 |
|---|---|
| TCDn_SLAST | 100Ch + (n × 20h) |

### 16.4.5.25.2 图示

| | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 咬牙切齿 | 幻灯片 | | | | | | | | | | | | | | | |
| R | | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | u | u | u | u | u | u | u | u | u | u | u | u | u | u | u | u |

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 咬牙切齿 | 幻灯片 | | | | | | | | | | | | | | | |
| R | | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | u | u | u | u | u | u | u | u | u | u | u | u | u | u | u | u |

### 16.4.5.25.3 领域

| 场地 | 职能 |
|---|---|
| 31-0<br>幻灯片 | 最后的源地址调整<br>在主要迭代次数完成时添加到源地址的调整值。这个值可以用来将源地址恢复到初始值，或者调整地址以引用下一个数据结构。<br>该寄存器使用二补符号；溢出位被丢弃。 |

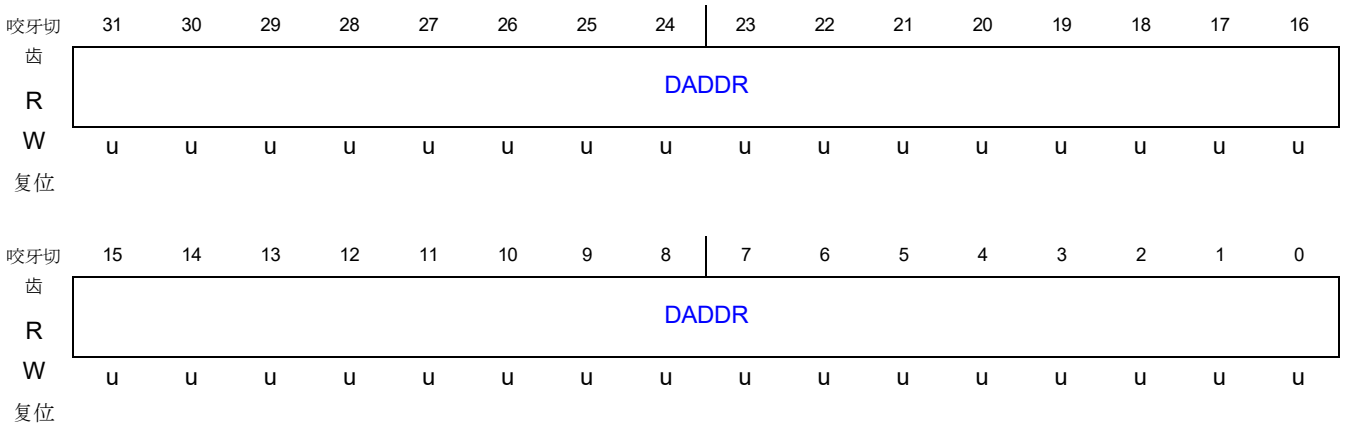
## 16.4.5.26 TCD目标地址（TCD0_DADDR - TCD15_DADDR）。

### 16.4.5.26.1 偏移

对于n=0到15。

| 注册 | 偏移 |
| --- | --- |
| TCDn_DADDR | 1010h + (n × 20h) |

### 16.4.5.26.2 图示




### 16.4.5.26.3 领域

| 场地 | 职能 |
| --- | --- |
| 31-0<br>DADDR | 目的地地址<br>指向目标数据的内存地址。 |

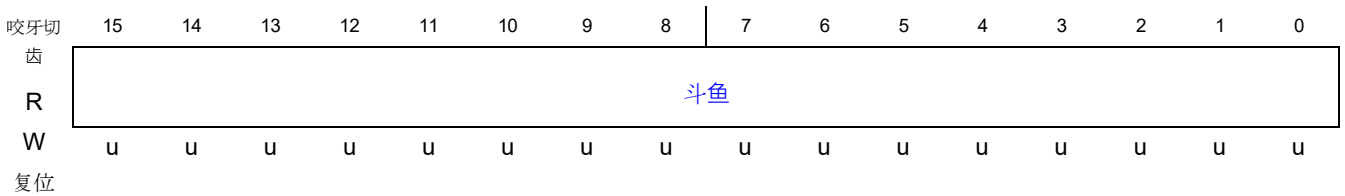
## 16.4.5.27 TCD有符号的目标地址偏移（TCD0_DOFF - TCD15_DOFF）。

### 16.4.5.27.1 偏移

对于n=0到15。

| 注册 | 偏移 |
| --- | --- |
| TCDn_DOFF | 1014h + (n × 20h) |

### 16.4.5.27.2 图示




### 16.4.5.27.3 领域

| 场地 | 职能 |
|---|---|
| 15-0<br>斗鱼 | 目的地地址有符号的偏移量<br>符号扩展的偏移量，应用于当前目标地址，在每个目标写入完成时形成下一个状态值。 |

## 16.4.5.28 TCD 当前小环路链接，大环路计数（通道链接已禁用）（TCD0_CITER_ELINKNO - TCD15_CITER_ELINKNO）。

### 16.4.5.28.1 偏移

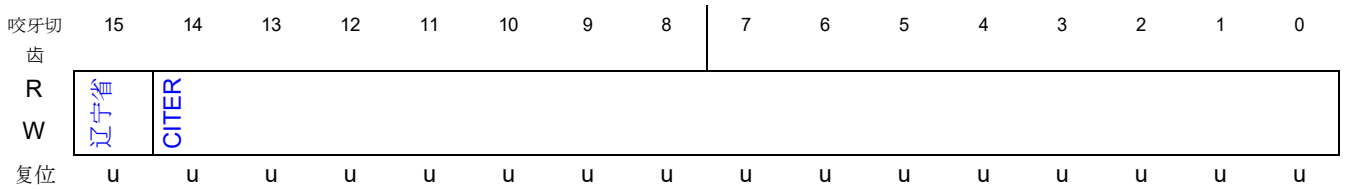
对于n=0到15。

| 注册 | 偏移 |
|---|---|
| TCDn_CITER_ELINKNO | 1016h + (n × 20h) |

### 16.4.5.28.2 职能

该寄存器包含小环路通道连接配置和通道的当前迭代计数。它与 TCD 当前小环路链接，大环路计数（通道链接已启用）（TCD0_CITER_ELINKYES - TCD15_CITER_ELINKYES）是同一个寄存器，但它的字段是根据 ELINK 字段的状态来定义的。如果ELINK字段被清零，这个寄存器的定义如下。

### 16.4.5.28.3 图示




### 16.4.5.28.4 领域

| 场地 | 职能 |
|---|---|
| 15<br>辽宁省 | 在小环路完成时启用通道与通道之间的链接<br>当通道完成小循环时，这个标志使链接到另一个通道，由LINKCH字段定义。链接目标通道通过内部机制启动通道服务请求，设置指定通道的TCDn_CSR[START]位。<br>如果通道链接被禁用，CITER值被扩展到15位，以代替链接通道号。如果主要环路被耗尽，这种链接机制被抑制，而采用MAJORELINK通道链接。<br>**注意：**该位必须等于BITER[ELINK]位；否则，将报告配置错误。<br>0b - 通道与通道之间的连接被禁用 1b - 通道与通道之间的连接被启用 |
| 14-0<br>CITER | 当前主要迭代次数<br>这个字段是该通道当前的主要循环计数。它在每次小循环完成时被递减，并在传输控制描述符存储器中被更新。在大循环计数用完后，通道执行一些操作，例如，最终的源地址和目的地址计算，可选择产生一个中断以示通道完成，然后再从起始迭代计数（BITER）字段重新加载CITER字段。<br>**注意：**当CITER字段最初由软件加载时，它必须被设置为与BITER字段中包含的数值相同。<br>**注意：**如果通道被配置为执行单个服务请求，BITER 和 CITER 的初始值应该是 0x0001。 |

## 16.4.5.29 TCD 当前小环路链接，大环路计数（通道链接已启用）（TCD0_CITER_ELINKYES - TCD15_CI TER_ELINKYES）。

### 16.4.5.29.1 偏移

对于n=0到15。

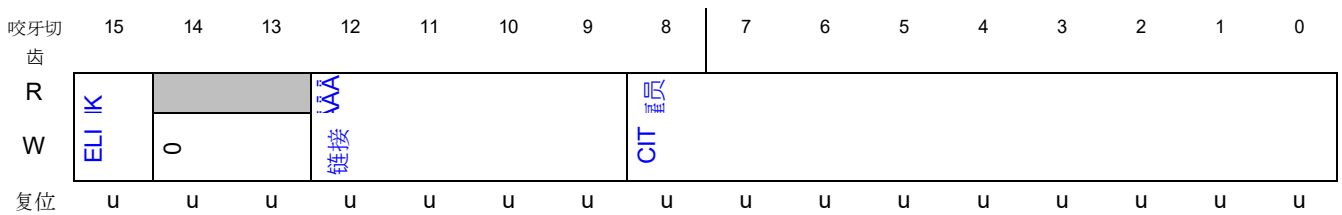
| 注册 | 偏移 |
|---|---|

| | |
|---|---|
| TCDn_CITER_ELINKYES | 1016h + (n × 20h) |

### 16.4.5.29.2 职能

该寄存器包含小环路通道连接配置和通道的当前迭代计数。它与 TCD 当前小环路链接、大环路计数（通道链接禁用）（TCD0_CITER_ELINKNO - TCD15_CITER_ELINKNO）是同一个寄存器，但其字段的定义根据 ELINK 字段的状态而不同。如果ELINK字段被设置，该寄存器的定义如下。

### 16.4.5.29.3 图示




### 16.4.5.29.4 领域

| 场地 | 职能 |
|---|---|
| 15<br>辽宁省 | 在小环路完成时启用通道与通道之间的链接<br>当通道完成小循环时，这个标志使链接到另一个通道，由LINKCH字段定义。链接目标通道通过内部机制启动通道服务请求，设置指定通道的TCDn_CSR[START]位。<br>如果通道链接被禁用，CITER值被扩展到15位，以代替链接通道号。如果主要环路被耗尽，这种链接机制被抑制，而采用MAJORELINK通道链接。<br>**注意：**该位必须等于BITER[ELINK]位；否则，将报告配置错误。<br>0b - 通道与通道之间的连接被禁用 1b - 通道与通道之间的连接被启用 |
| 14-13<br>- | 保留 |
| 12-9<br>LINKCH | 小环路链接通道号<br>如果通道与通道之间的链接被启用（ELINK = 1），那么在小循环用完后，eDMA引擎通过设置该通道的TCDn_CSR[START]位，向该字段定义的通道发起通道服务请求。 |

| | |
|---|---|
| 8-0<br>CITER | 当前主要迭代次数<br>这个字段是该通道当前的主要循环计数。它在每次小循环完成时被递减，并在传输控制描述符存储器中被更新。在大循环计数用完后，通道执行一些操作，例如，最终的源地址和目的地址计算，可选择产生一个中断以示通道完成，然后再从起始迭代计数（BITER）字段重新加载CITER字段。<br>**注意：**当CITER字段最初由软件加载时，它必须被设置为与BITER字段中包含的数值相同。 |

| 场地 | 职能 |
| --- | --- |
| | **注意：**如果通道被配置为执行单个服务请求，BITER 和 CITER 的初始值应该是 0x0001。 |

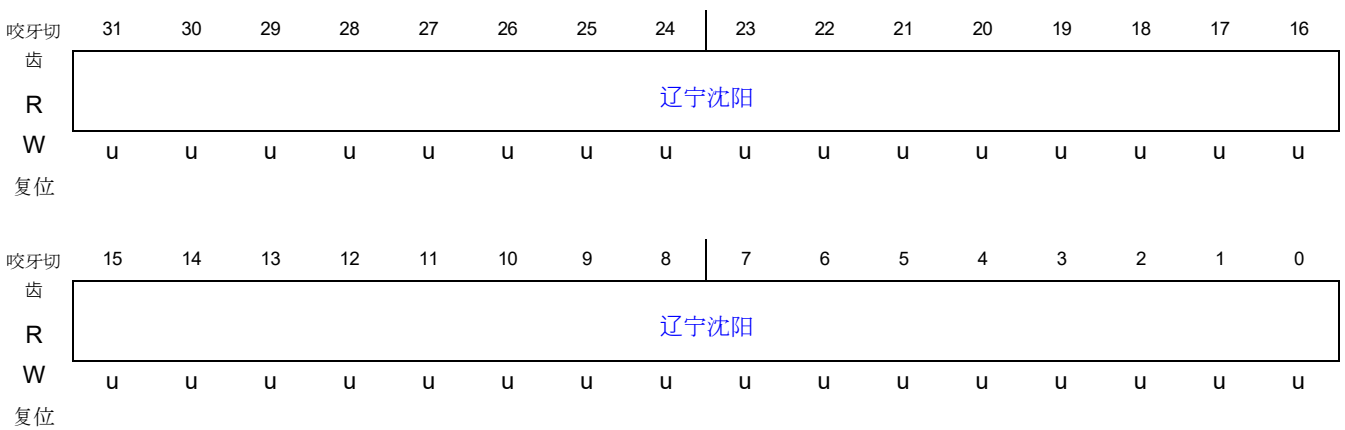
## 16.4.5.30 TCD最后目的地地址调整/散点收集地址（TCD0_DLASTSGA - TCD15_DLASTSGA）。

### 16.4.5.30.1 偏移

对于n=0到15。

| 注册 | 偏移 |
| --- | --- |
| TCDn_DLASTSGA | 1018h + (n × 20h) |

### 16.4.5.30.2 图示




### 16.4.5.30.3 领域

| 场地 | 职能 |
| --- | --- |

| | |
|---|---|
| 31-0<br>辽宁沈阳 | 辽宁沈阳<br>目的地最后的地址调整或下一个传输控制描述符的内存地址，将被加载到这个通道（分散/聚集）。<br>如果（TCDn_CSR[ESG]=0），那么。<br>• 在主要迭代次数完成时添加到目标地址的调整值。这个值可以应用于将目标地址恢复到初始值或调整地址以引用下一个数据结构。<br>• 这个字段使用二进制补码来调整最终目标地址。<br>否则。 |

| 场地 | 职能 |
| --- | --- |
| | • 这个地址指向一个0-modulo-32字节的区域的开始，这个区域包含下一个要加载到这个通道的传输控制描述符。这个通道的重新加载是在主要迭代计数完成后进行的。散射/聚集地址必须是0-modulo-32-byte，否则会报告配置错误。 |

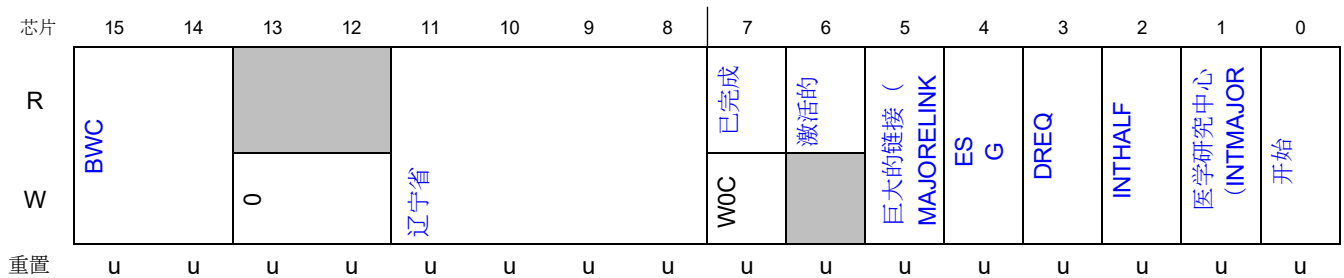
## 16.4.5.31 TCD控制和状态（TCD0_CSR - TCD15_CSR）。

### 16.4.5.31.1 偏移

对于n=0到15。

| 注册 | 偏移 |
| --- | --- |
| TCDn_CSR | 101Ch + (n × 20h) |

### 16.4.5.31.2 图示




### 16.4.5.31.3 领域

| 场地 | 职能 |
| --- | --- |

| | |
|---|---|
| 15-14<br>BWC | 带宽控制<br><br>节制eDMA所消耗的总线带宽量。一般来说，当eDMA处理小循环时，它不断产生读/写序列，直到小循环计数用完。这个字段迫使eDMA在每次读/写访问完成后停顿，以控制横梁开关看到的总线请求带宽。<br><br>**注意：**如果源端和目的端大小相等，在第一次和第二次传输之间以及每个小循环的最后一次写入之后，这个字段会被忽略。这种行为是减少启动延迟的一个副作用。<br>00b - 没有eDMA引擎停顿。01b - 保留<br>10b - eDMA引擎在每个R/W之后停顿4个周期。11b - eDMA引擎在每个R/W之后停顿8个周期。 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| 13-12<br>- | 保留 |
| 11-8<br>辽宁省 | 主要环路链接通道号码 如果（<br>MAJORELINK = 0），那么。<br>• 在主要的循环计数器用完后，不进行通道与通道之间的连接，也不进行链式连接。<br>否则。<br>• 在主循环计数器耗尽后，eDMA引擎通过设置该通道的TCDn_CSR[START]位，在该字段定义的通道发起通道服务请求。 |
| 7<br>已完成 | 频道完成<br>该标志表示eDMA已经完成了主要循环。eDMA引擎在CITER计数达到0时设置该标志。软件清除它，或者在通道被激活时硬件清除。<br>**注意：**该位必须被清除才能写入MAJORELINK或ESG位。 |
| 6<br>激活的 | 渠道活跃<br>这个标志表明通道目前正在执行中。它在通道服务开始时被设置，并在小循环完成或检测到任何错误情况时被eDMA清除。 |
| 5<br>巨大的链接（<br>MAJORELINK | 在主要环路上启用通道与通道之间的链接，完成后的通道与通道之间的链接<br>当通道完成主循环时，这个标志使链接到另一个通道，由MAJORLINKCH定义。链接目标通道通过内部机制启动通道服务请求，设置指定通道的TCDn_CSR[START]位。<br>**注意：**为了支持动态链接一致性模型，当TCDn_CSR[DONE]位被设置时，该字段被强制为零。<br>0b - 通道与通道之间的连接被禁用。 1b - 通道与<br>通道之间的连接被启用。 |
| 4<br>ESG | 启用散射/聚集处理<br>当通道完成主循环时，这个标志在当前通道中启用散射/聚集处理。如果启用，eDMA引擎使用DLASTSGA作为内存指针，指向一个包含32字节数据结构的0-模32地址，作为传输控制描述符加载到本地内存。<br>**注意：**为了支持动态散射/聚集一致性模型，当TCDn_CSR[DONE]位被设置时，该字段被强制写为零。<br>0b - 当前通道的TCD为正常格式。<br>1b - 当前通道的TCD指定了一个散点收集格式。DLASTSGA字段提供了一个内存指针，用于在主要循环完成后将下一个TCD加载到该通道。 |
| 3<br>DREQ | 禁用请求<br>如果这个标志被设置，当当前主要迭代次数达到零时，eDMA硬件自动清除相应的ERQ位。<br>0b - 该通道的ERQ位不受影响。<br>1b - 当主要循环完成时，通道的ERQ位被清零。 |

| | |
|---|---|
| 2<br>INTHALF | 当主要计数器完成一半时，启用一个中断。<br>如果这个标志被设置，当当前主要的迭代次数达到一半时，通道通过设置INT寄存器中适当的位来产生一个中断请求。具体来说，eDMA引擎执行的比较是（CITER == (BITER >> 1)）。这个半点中断请求是为了支持双缓冲，也被称为乒乓，方案或其他类型的数据移动，其中处理器需要一个传输进度的早期指示。<br>**注意：**如果BITER=1，不要使用INTHALF。使用INTMAJOR代替。<br>0b - 半点中断被禁用。 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | 1b - 半点中断被启用。 |
| 1<br>医学研究中心（<br>INTMAJOR | 当主要迭代计数完成时，启用一个中断。<br>如果这个标志被设置，当当前主要的迭代次数达到零时，通道通过设置INT中的相应位来产生一个中断请求。<br>0b - 主要循环结束的中断被禁用。1b - 主要循环结束的中断被激活。 |
| 0<br>开始 | 频道启动<br>如果这个标志被设置，说明该通道正在请求服务。eDMA硬件在通道开始执行后自动清除这个标志。<br>0b - 该通道没有明确启动。<br>1b - 该通道通过软件发起的服务请求明确启动。 |

## 16.4.5.32 TCD开始小环路链接，大环路计数（通道链接已启用）（TCD0_BITER_ELINKYES - TCD1 5_BITER_ELINKYES）。

### 16.4.5.32.1 偏移

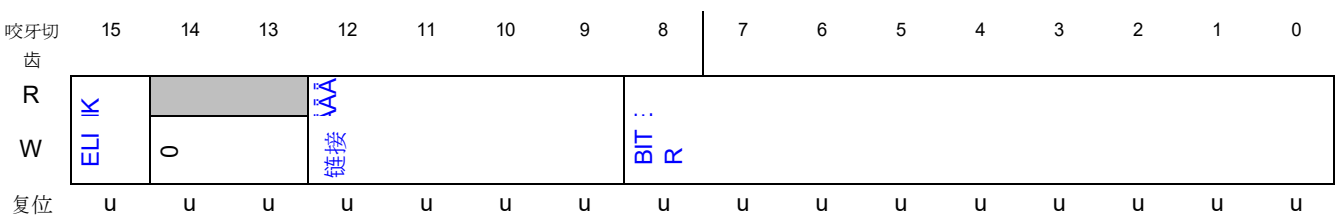
对于n=0到15。

| 注册 | 偏移 |
| --- | --- |
| TCDn_BITER_ELINKYES | 101Eh + (n × 20h) |

### 16.4.5.32.2 职能

如果TCDn_BITER[ELINK]位被设置，TCDn_BITER寄存器的定义如下。

### 16.4.5.32.3 图示



#### 16.4.5.32.4 领域

| 场地 | 职能 |
|---|---|
| 15<br>辽宁省 | 在小循环完成时，启用通道与通道之间的链接<br>当通道完成小循环时，这个标志使链接到另一个通道，由BITER[LINKCH]定义。链接目标通道通过内部机制启动通道服务请求，设置指定通道的 TCDn_CSR[START] 位。如果通道链接禁用，BITER值扩展到15位，以代替链接通道号。如果主要环路被耗尽，该链接机制被抑制，而采用MAJORELINK通道链接。<br>**注意：**当软件加载TCD时，该字段必须被设置为等于相应的CITER字段；否则，将报告配置错误。随着主要迭代次数的耗尽，该字段的内容被重新加载到CITER字段。<br>0b - 通道与通道之间的连接被禁用 1b - 通道与通道之间的连接被启用 |
| 14-13<br>- | 保留 |
| 12-9<br>LINKCH | 链接通道号<br>如果通道与通道之间的链接被启用（ELINK = 1），那么在小循环用完后，eDMA引擎通过设置该通道的TCDn_CSR[START]位，在该字段定义的通道发起通道服务请求。<br>**注意：**当软件加载TCD时，该字段必须被设置为等于相应的CITER字段；否则，将报告配置错误。随着主要迭代次数的耗尽，该字段的内容被重新加载到CITER字段。 |
| 8-0<br>BITER | 开始的主要迭代数<br>当传输控制描述符第一次被软件加载时，这个9位（ELINK=1）或15位（ELINK=0）字段必须等于CITER字段的值。随着主要迭代次数的耗尽，这个字段的内容被重新加载到CITER字段。<br>**注意：**当软件加载TCD时，该字段必须被设置为等于相应的CITER字段；否则，将报告配置错误。随着主要迭代次数的耗尽，这个字段的内容被重新加载到CITER字段。如果通道被配置为执行单个服务请求，BITER和CITER的初始值应该是0x0001。 |

## 16.4.5.33 TCD开始小环路链接，大环路计数（通道链接被禁用）（TCD0_BITER_ELINKNO - TCD1 5_BITER_ELINKNO）。

### 16.4.5.33.1 偏移

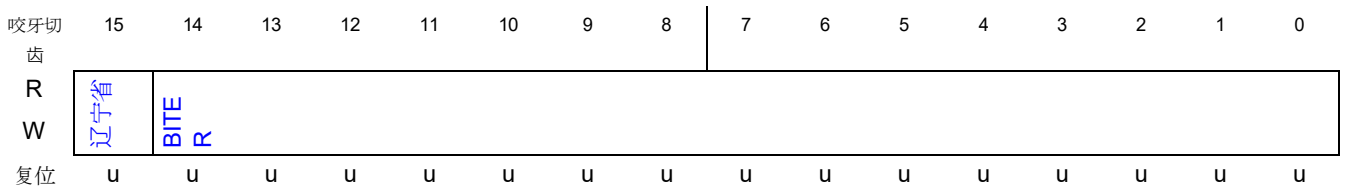
对于n=0到15。

| 注册 | 偏移 |
|---|---|
| TCDn_BITER_ELINKNO | 101Eh + (n × 20h) |

### 16.4.5.33.2 职能

如果TCDn_BITER[ELINK]位被清除，TCDn_BITER寄存器的定义如下。

### 16.4.5.33.3 图示




### 16.4.5.33.4 领域

| 场地 | 职能 |
|---|---|
| 15<br>辽宁省 | 在小循环完成时，启用通道与通道之间的链接<br>当通道完成小循环时，这个标志使链接到另一个通道，由BITER[LINKCH]定义。链接目标通道通过内部机制启动通道服务请求，设置指定通道的 TCDn_CSR[START] 位。如果通道链接被禁用，BITER值扩展到15位，以代替链接通道号。如果主要环路被耗尽，该链接机制被抑制，而采用MAJORELINK通道链接。<br>**注意：**当软件加载TCD时，该字段必须被设置为等于相应的CITER字段；否则，将报告配置错误。随着主要迭代次数的耗尽，该字段的内容被重新加载到CITER字段。<br>0b - 通道与通道之间的连接被禁用 1b - 通道与通道之间的连接被启用 |
| 14-0<br>BITER | 开始主要迭代次数<br>当传输控制描述符第一次被软件加载时，这个9位（ELINK=1）或15位（ELINK=0）字段必须等于CITER字段的值。随着主要迭代次数的耗尽，这个字段的内容被重新加载到CITER字段。<br>**注意：**当软件加载TCD时，该字段必须被设置为等于相应的CITER字段；否则，将报告配置错误。随着主要迭代次数的耗尽，该字段的内容被重新加载到CITER字段。如果通道被配置为执行单个服务请求，BITER和CITER的初始值应该是0x0001。 |

## 16.5 功能 描述

eDMA的操作将在以下几个小节中描述。

### 16.5.1 eDMA基本数据 流程

数据传输的基本流程可以划分为三个部分。

如下图所示，第一段涉及通道的激活。

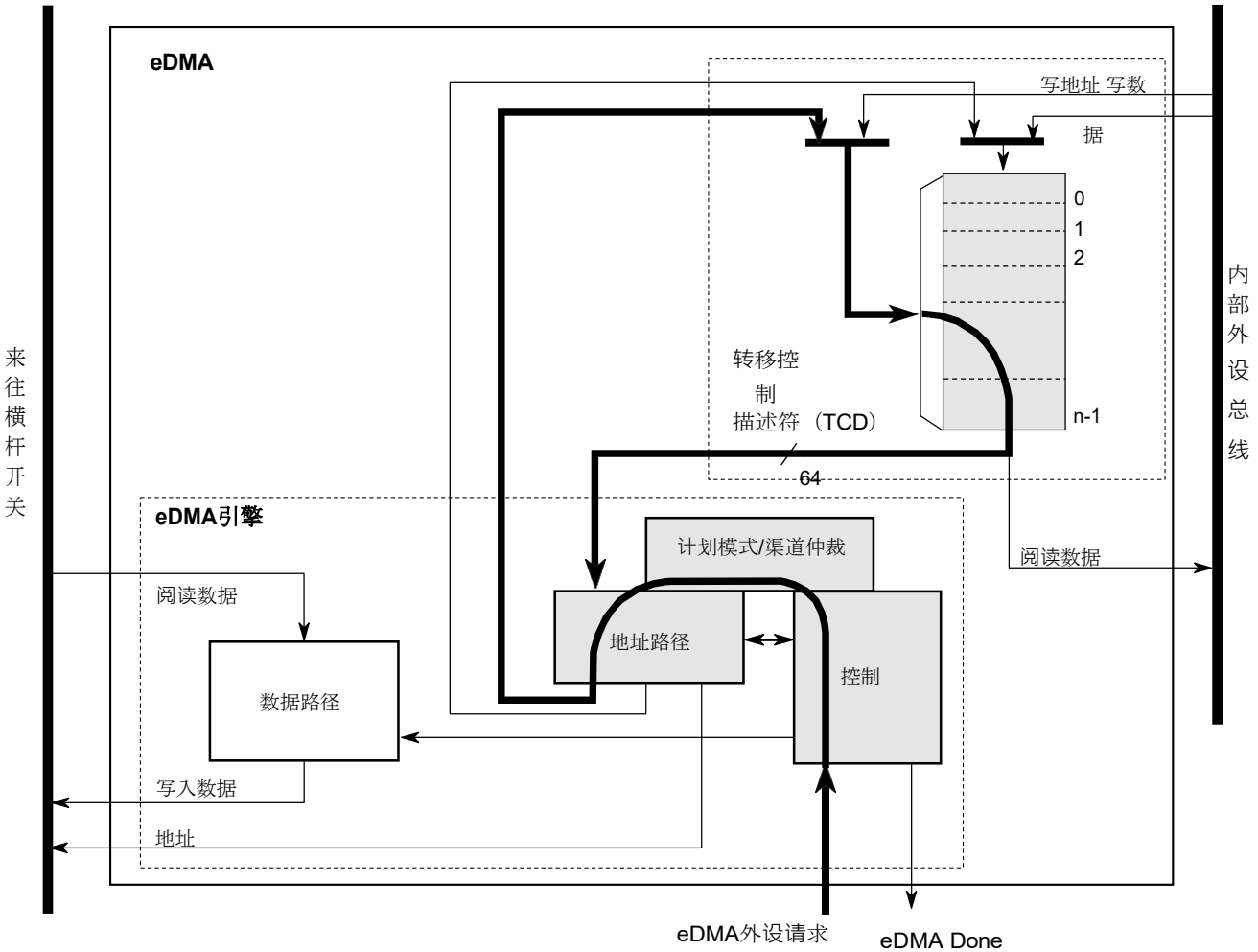



**图16-2. eDMA操作，第一部分**

这个例子使用eDMA外设请求信号的断言来请求通道*n*的服务。通过软件和TCDn_CSR[START]位激活的通道与外设请求的基本流程相同。eDMA请求输入信号在内部注册，然后通过eDMA引擎路由：首先通过控制模块，然后进入程序模型和通道仲裁。在下一个周期，通道仲裁执行，使用固定优先级或轮流算法。仲裁完成后，激活的通道号通过地址路径发送，并转换为所需的地址以访问TCDn的本地存储器。接下来，TCD存储器被访问，所需的描述符从本地存储器中读取，并加载到eDMA引擎地址路径通道x或y寄存器中。TCD存储器的宽度为64位，以尽量减少获取激活的通道描述符并将其加载到地址路径通道x或y寄存器所需的时间。

下图说明了基本数据流的第二部分。

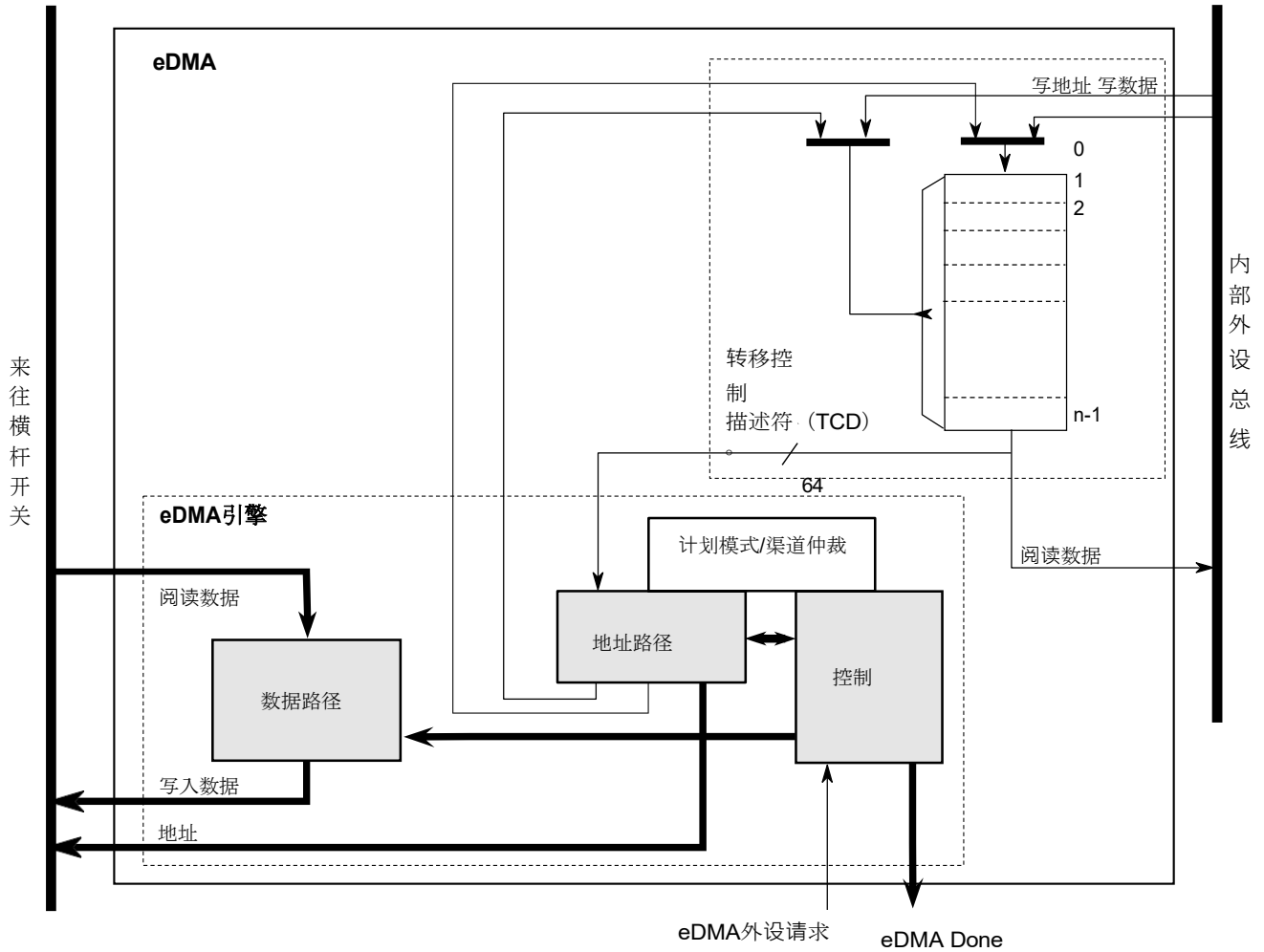



**图16-3. eDMA操作，第二部分**

与数据传输相关的模块（地址路径、数据路径和控制）通过所需的源读和目的写进行排序，以执行实际的数据移动。源读取被启动，获取的数据被暂时存储在数据路径块中，直到它在目的地写入时被门控到内部总线上。这种源读/目的写处理一直持续到小字节数转移完毕。

在小字节数移动后，进行基本数据流的最后阶段。在这个环节中，地址路径逻辑对适当的TCD中的某些字段进行所需的更新，例如，SADDR、DADDR、CITER。如果主要的迭代次数用完了，就会进行额外的操作。这些操作包括最后的地址调整和将BITER字段重新加载到CITER中。这时也会发生一个可选的中断请求，以及可能使用描述符中的散射/聚集地址指针从内存中获取一个新的TCD（如果散射/聚集被启用）。对TCD存储器的更新和对中断请求的断言显示在下图中。

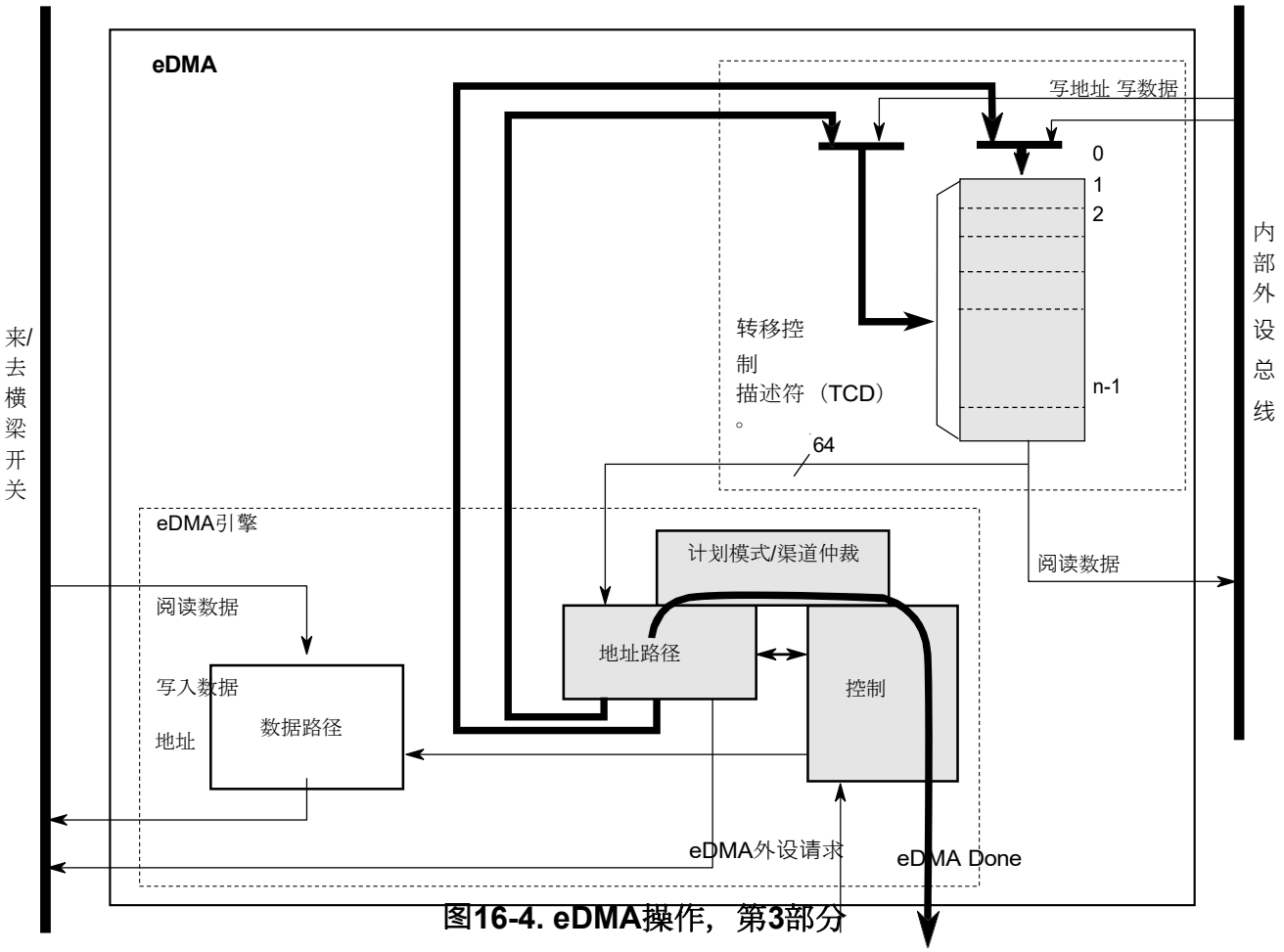



**图16-4. eDMA操作，第3部分**

## 16.5.2 故障报告和 处理

通道错误在错误状态寄存器(DMAx_ES)中报告，可由以下原因引起。

- 配置错误，即传输控制描述符中的非法设置或固定仲裁模式中的非法优先级寄存器设置，或
- 一个错误终止于总线主站的读或写周期

当起始源地址或目标地址、源或目标偏移量、小循环字节数或传输大小代表不一致的状态时，将报告配置错误。这些可能的原因都在下面详细说明。

- 地址和偏移量必须在0-modulo-transfer-size边界上对齐。
- 小循环字节数必须是源和目的传输大小的倍数。

- 所有的源读和目的写必须分别配置到编程的传输大小的自然边界。
- 在固定仲裁模式下，任何两个通道优先级相等都会造成配置错误。启用固定仲裁模式时，所有通道的优先级必须是唯一的。

**注意事项**

当两个通道具有相同的优先级时，存在一个通道优先级错误，将在错误状态寄存器中报告。但是，通道号码不会被报告在错误状态寄存器中。当一个组内所有的通道优先级不唯一时，通过仲裁选择的通道号是不确定的。

为了帮助通道优先级错误（CPE）的调试，在DMA的控制寄存器中设置出错时停止位。如果一个组内的所有通道优先级不是唯一的，在记录CPE错误后，DMA将被停止。DMA将保持停止状态，不处理任何通道服务请求。一旦所有的通道优先级被设置为唯一的数字，DMA可以通过清除Halt位再次被启用。

- 如果在通道完成时启用了散射/收集操作，如果散射/收集地址（DLAST_SGA）没有在32字节边界上对齐，则报告配置错误。
- 如果在通道完成时启用了小循环通道链接，当尝试链接时，如果TCDn_CITER[E_LINK]位不等于TCDn_BITER[E_LINK]位，则报告配置错误。

如果启用，所有的配置错误条件，除了散射/收集和小环路链接错误，都会在通道激活并发出错误中断请求时报告。当正确启用时，在主要环路完成时开始进行散射/收集操作，报告散射/收集配置错误。当链接操作在小循环完成时被服务时，报告一个小循环通道链接配置错误。

如果系统总线的读或写因错误而终止，数据传输将被停止，并设置相应的总线错误标志。在这种情况下，通道的传输控制描述符的状态由eDMA引擎用当前的源地址、目标地址和故障点的当前迭代计数来更新。当系统总线错误发生时，该通道在下一次传输后终止。由于流水线效应，当eDMA收到总线错误时，下一次传输已经在进行中。如果一个总线错误

如果在开始写序列之前的最后一次读发生总线错误，则使用在总线错误期间捕获的数据执行写。如果在切换到下一个读序列之前的最后一次写发生了总线错误，那么在通道由于目标总线错误而终止之前，读序列就会执行。

软件可以通过CR[CX]位来取消一个传输。当取消传输请求被识别时，DMA引擎停止处理该通道。当前的读写序列被允许完成。如果取消发生在一个大循环或小循环的最后一个读写序列上，则取消请求被丢弃，通道正常退场。

除了错误状态寄存器（DMAx_ES）被更新为已取消的通道编号和ECX被设置外，错误取消传输与取消传输是一样的。被取消的通道的TCD包含保存在TCD中的最后一次传输的源地址和目的地址。如果通道需要重新启动，你必须重新初始化TCD，因为上述的字段不再代表原来的参数。当一个传输被错误取消传输机制取消时，通道号被加载到DMA_ES[ERRCHN]，ECX和VLD被设置。此外，如果启用，可能会产生一个错误中断。

**注意事项**

取消传输请求允许用户在不再需要全部数据传输的情况下停止大型数据传输。取消传输位不会中止通道。它只是停止数据的传输，然后通过正常的关机顺序使通道退役。应用软件必须处理取消的背景。如果需要（或不需要）一个中断，那么在取消请求之前应该启用（或禁用）中断。应用软件必须清理传输控制描述符，因为完整的传输并没有发生。

任何错误的发生都会导致eDMA引擎立即停止对活动通道的正常处理（它进入错误处理状态，对系统总线的事务仍有管道效应），eDMA错误寄存器中的相应通道位被断言。同时，错误状态的细节被加载到错误状态寄存器（DMAx_ES）。当检测到错误时，主要的循环完成指标、设置传输控制描述符DONE标志和可能的中断请求断言，都不会受到影响。在错误状态被更新后，eDMA引擎通过服务下一个合适的通道继续运行。遇到错误状态的通道不会被自动禁用。如果一个通道被错误终止，然后在错误被修复之前发出另一个服务请求，该通道就会执行并以同样的错误状态终止。

### 16.5.3 频道 抢占先机

通道抢占是通过设置DCHPRIn[ECP]位在每个通道基础上启用的。通道抢占允许执行通道的数据传输暂时中止，以利于启动一个更优先的通道。在抢占的通道完成其所有的小循环数据传输后，被抢占的通道被恢复并继续执行。在恢复的通道完成了一个读/写序列后，它又有资格被抢占。如果有更高优先级的通道请求服务，被恢复的通道将被暂停，更高优先级的通道将得到服务。不支持嵌套式抢占，即试图抢占一个抢占通道，。在一个抢占的通道开始执行后，它不能被抢占。只有当选择固定仲裁时，抢占才是可用的。

一个通道抢占另一个通道的能力可以通过设置DCHPRIn[DPA]来禁用。当一个通道的抢占能力被禁用时，该通道不能暂停低优先级通道的数据传输，无论低优先级通道的ECP设置如何。这允许定义一个低优先级、大数据移动的通道池。这些低优先级的通道可以被配置为不互相抢占，从而防止低优先级的通道消耗真正的高优先级通道通常可用的抢占时间。

## 16.6 初始化/应用 信息

下面几节讨论eDMA的初始化和编程注意事项。

### 16.6.1 eDMA 初始化

要初始化eDMA。

1. 如果需要除默认值以外的配置，请写到CR。
2. 如果需要一个非默认的配置，请将通道优先级写到DCHPRIn寄存器中。
3. 如果需要，在EEI寄存器中启用错误中断。
4. 为每个可能请求服务的通道写入32字节的TCD。
5. 通过ERQ寄存器启用任何硬件服务请求。

6. 通过以下两种方式请求通道服务。
   - 软件：设置TCDn_CSR[START]。
   - 硬件：从属设备发出其eDMA外设请求信号

在任何通道请求服务后，根据写进程序员模型的仲裁和优先级，选择一个通道进行执行。eDMA引擎读取整个TCD，包括TCD控制和状态字段，如下表所示，用于选定的通道进入其内部地址路径模块。

当TCD被读取时，除非检测到配置错误，否则将在内部总线上启动第一次传输。从TCDn_SADDR定义的源端向TCDn_DADDR定义的目的端传输，一直持续到TCDn_NBYTES指定的字节数被传输完毕。

当传输完成后，eDMA引擎的本地TCDn_SADDR、TCDn_DADDR和TCDn_CITER被写回主TCD存储器，如果启用，则执行任何小循环通道链接。如果主循环被耗尽，则执行进一步的后处理，如中断、主循环通道链接和散射/收集操作（如果启用）。

**表16-5.TCD控制和状态字段**

| TCDn_CSR字段名称 | 描述 |
|---|---|
| 开始 | 当使用软件启动的DMA服务时，明确启动通道的控制位（由硬件自动清除）。 |
| 激活的 | 表示通道当前正在执行的状态位 |
| 已完成 | 指示主要循环完成的状态位（当使用软件启动的DMA服务时，由软件清除）。 |
| D_REQ | 当使用硬件启动的DMA服务时，控制位用于在主要循环结束时禁用DMA请求 |
| BWC | 用于通道的节流带宽控制的控制位 |
| E_SG | 启用散点收集功能的控制位 |
| INT_HALF | 控制位，当主要循环完成一半时启用中断 |
| INT_MAJ | 控制位，当主要循环完成时启用中断 |

下图显示了每个DMA请求如何在没有CPU干预的情况下启动一个小环路传输，或者说迭代。DMA仲裁可以在每个小循环之后发生，并且允许一个级别的小循环DMA抢占。一个大循环中的小循环的数量由开始迭代计数（BITER）指定。

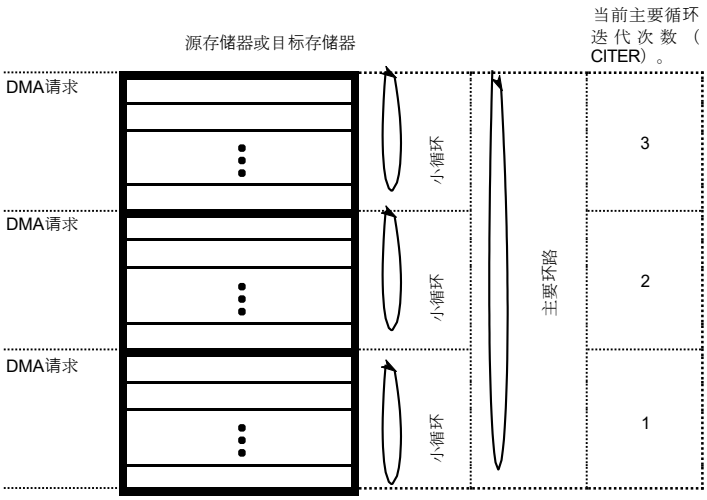



**图16-5.多次循环迭代的例子**

下图列出了内存阵列术语和TCD设置的相互关系。

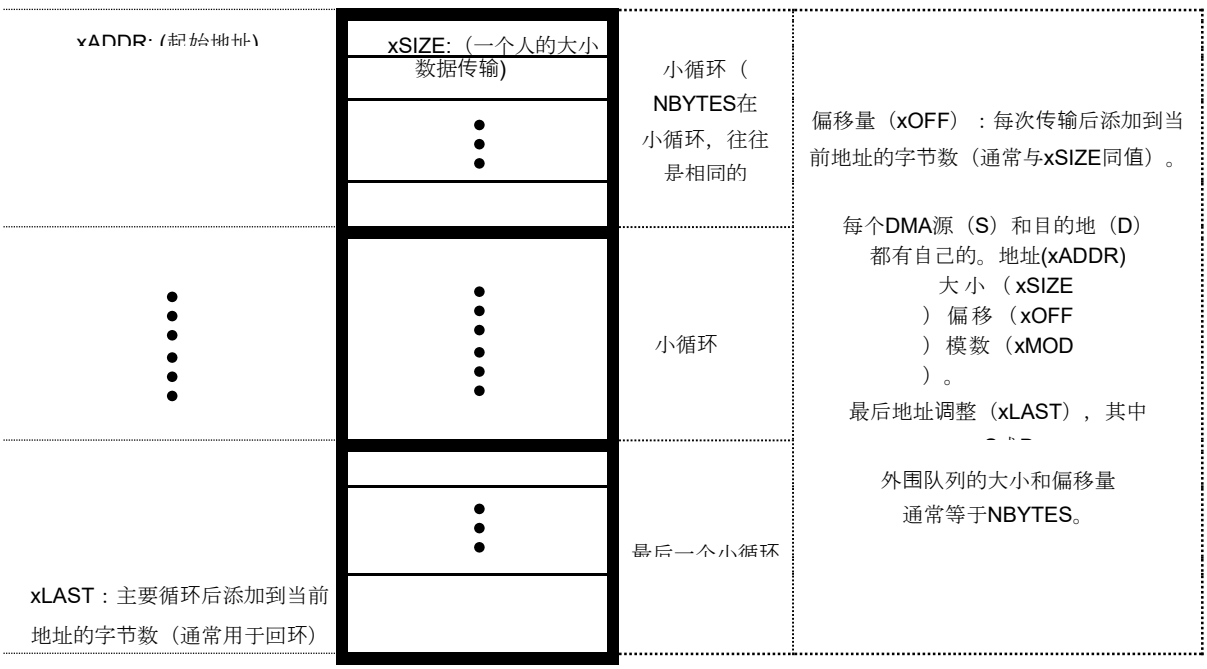



**图16-6.存储器阵列术语**

## 16.6.2 编程 错误

eDMA对传输控制描述符进行各种测试以验证描述符数据的一致性。除了通道优先级错误（ES[CPE]）外，大多数编程错误都是以每个通道为单位报告的。

对于除通道优先级错误以外的所有错误类型，引起错误的通道号被记录在错误状态寄存器（DMAx_ES）中。如果在下一次激活有问题的通道之前，错误源没有被清除，那么错误会被检测并再次记录。

如果优先级不唯一，当任何通道请求服务时，会报告通道优先级错误。具有活动请求的最高通道优先级被选中，但具有该优先级的最低编号的通道被仲裁选中并由eDMA引擎执行。硬件服务请求握手信号、错误中断和错误报告与所选通道相关。

## 16.6.3 仲裁模式 考虑因素

本节讨论eDMA的仲裁注意事项。

### 16.6.3.1 固定通道仲裁

在这种模式下，来自最高优先级通道的通道服务请求被选择执行。

### 16.6.3.2 轮流通道仲裁

频道的服务从最高的频道号开始，轮流到最低的频道号，不考虑频道的优先级。

## 16.6.4 执行DMA 传输

本节介绍了如何用eDMA进行DMA传输的例子。

### 16.6.4.1 单一请求

要通过一次激活执行n个字节的简单数据传输，请将主循环设置为1（TCDn_CITER = TCDn_BITER =
1）。在通道服务请求被确认并且通道被选择执行后，数据传输开始。传输完成后，TCDn_CSR[DONE]位被设置，如果正确启用，则产生一个中断。

例如，下面的TCD条目被配置为传输16字节的数据。eDMA被编程为主要循环的一次迭代，每次迭代传输16字节。源内存有一个字节宽的内存端口，位于0x1000。目标存储器有一个32位的端口，位于0x2000。地址偏移是以增量编程的，以匹配传输的大小：源端为一个字节，目的端为四个字节。最终的源地址和目的地址被调整为返回到它们的起始值。

```
TCDn_CITER = TCDn_BITER = 1
TCDn_NBYTES = 16
TCDn_SADDR = 0x1000
TCDn_SOFF = 1
TCDn_ATTR[SSIZE] = 0
TCDn_SLAST = -16
TCDn_DADDR = 0x2000
TCDn_DOFF = 4
TCDn_ATTR[DSIZE] = 2
TCDn_DLAST_SGA= -16
TCDn_CSR[INT_MAJ] = 1
TCDn_CSR[START] = 1 （应在所有其他字段初始化后最后写入） 所有其他TCDn字段 = 0
```

这产生了以下事件序列。

1. 用户对TCDn_CSR[START]位的写入请求通道服务。
2. 该通道是通过仲裁选择的，用于服务。
3. eDMA引擎写入。TCDn_CSR[DONE] = 0，TCDn_CSR[START] = 0，TCDn_CSR[ACTIVE] = 1。
4. eDMA引擎读取：通道TCD数据从本地存储器到内部寄存器文件。
5. 源头到目的地的转移是按以下方式执行的。
   a. 从位置0x1000读字节，从位置0x1001读字节，从0x1002读字节，从0x1003读字节。
   b. 写32位到位置0x2000→小循环的第一次迭代。
   c. 从位置0x1004读取字节，从位置0x1005读取字节，从0x1006读取字节，从0x1007读取字节。
   d. 写32位到位置0x2004→小循环的第二次迭代。
   e. 从位置0x1008读字节，从位置0x1009读字节，从0x100A读字节，从0x100B读字节。
   f. 写32位到位置0x2008→小循环的第三次迭代。
   g. 从位置0x100C读取字节，从位置0x100D读取字节，从0x100E读取字节，从0x100F读取字节。

h. 写32位到位置0x200C→小循环的最后一次迭代→大循环完成。

6. eDMA引擎写入。TCDn_SADDR = 0x1000，TCDn_DADDR = 0x2000，TCDn_CITER = 1（TCDn_BITER）。
7. eDMA引擎写入。TCDn_CSR[ACTIVE] = 0，TCDn_CSR[DONE] = 1，INT[*n*] = 1。
8. 该通道退场，eDMA进入空闲状态或为下一个通道提供服务。

### 16.6.4.2 多个请求

下面的例子通过两个硬件请求传输32个字节，但其他方面与前面的例子相同。唯一改变的字段是主循环迭代计数和最终地址偏移。eDMA被编程为主要循环的两次迭代，每次迭代传输16字节。在ERQ寄存器中启用通道的硬件请求后，从属设备启动通道服务请求。

```
TCDn_CITER = TCDn_BITER = 2
TCDn_SLAST = -32
TCDn_DLAST_SGA = -32
```

这将产生以下的事件序列。

1. 首先是硬件，也就是eDMA外设，要求提供通道服务。
2. 该通道是通过仲裁选择的，用于服务。
3. eDMA引擎写入。TCDn_CSR[DONE] = 0，TCDn_CSR[START] = 0，TCDn_CSR[ACTIVE] = 1。
4. eDMA引擎读取：通道TCDn数据从本地存储器到内部寄存器文件。
5. 源头到目的地的转移按以下方式执行。
   a. 从位置0x1000读字节，从位置0x1001读字节，从0x1002读字节，从0x1003读字节。
   b. 写32位到位置0x2000→小循环的第一次迭代。
   c. 从位置0x1004读字节，从位置0x1005读字节，从0x1006读字节，从0x1007读字节。
   d. 写32位到位置0x2004→小循环的第二次迭代。
   e. 从位置0x1008读字节，从位置0x1009读字节，从0x100A读字节，从0x100B

读字节。

f. 写32位到位置0x2008→小循环的第三次迭代。

g. 从位置0x100C读取字节，从位置0x100D读取字节，从0x100E读取字节，从0x100F读取字节。

h. 写32位到位置0x200C→小循环的最后一次迭代。

6. eDMA引擎写入。TCDn_SADDR = 0x1010, TCDn_DADDR = 0x2010, TCDn_CITER = 1。
7. eDMA引擎写入。TCDn_CSR[ACTIVE] = 0。
8. 信道退场→大循环的一次迭代。eDMA进入空闲状态或为下一个通道服务。
9. 第二个硬件，也就是eDMA外设，请求通道服务。
10. 该通道是通过仲裁选择的，用于服务。
11. eDMA引擎写入。TCDn_CSR[DONE] = 0，TCDn_CSR[START] = 0，TCDn_CSR[ACTIVE] = 1。
12. eDMA引擎读取：通道TCD数据从本地存储器到内部寄存器文件。
13. 源头到目的地的转移按以下方式执行。

    a. 从位置0x1010读字节，从位置0x1011读字节，从0x1012读字节，从0x1013读字节。

    b. 写32位到位置0x2010→小循环的第一次迭代。

    c. 从位置0x1014读字节，从位置0x1015读字节，从0x1016读字节，从0x1017读字节。

    d. 写32位到位置0x2014→小循环的第二次迭代。

    e. 从位置0x1018读字节，从位置0x1019读字节，从0x101A读字节，从0x101B读字节。

    f. 写32位到位置0x2018→小循环的第三次迭代。

    g. 从位置0x101C读字节，从位置0x101D读字节，从0x101E读字节，从0x101F读字节。

    h. 写32位到位置0x201C→小循环的最后一次迭代→大循环完成。

14. eDMA引擎写入。TCDn_SADDR = 0x1000, TCDn_DADDR = 0x2000,

TCDn_CITER = 2 (TCDn_BITER)。

15. eDMA引擎写入。TCDn_CSR[ACTIVE] = 0, TCDn_CSR[DONE] = 1, INT[n] = 1.
16. 通道退场→主要循环完成。eDMA进入空闲状态或服务于下一个通道。

#### 16.6.4.3 使用乘法功能

eDMA的modulo功能提供了实现循环数据队列的能力，其中队列的大小是2的幂。MOD是TCD中源和目的的5位字段，它指定了在地址+偏移计算后，哪些下位地址位从其原始值递增。所有的上层地址位保持与原始值相同。该字段的设置为0，则禁用模数功能。

下表显示了如何根据MOD字段的设置来指定传输地址。这里创建了一个循环缓冲区，地址被包裹到原始值，而上面的28个地址位（0x1234567x）保留其原始值。在这个例子中，源地址被设置为0x12345670，偏移量被设置为4个字节，MOD字段被设置为4，允许一个$^{24}$字节（16字节）大小的队列。

**表16-6.模数的例子**

| 转账号码 | 地址 |
|---|---|
| 1 | 0x12345670 |
| 2 | 0x12345674 |
| 3 | 0x12345678 |
| 4 | 0x1234567C |
| 5 | 0x12345670 |
| 6 | 0x12345674 |

### 16.6.5 监控传输描述符 状态

本节讨论了如何监控eDMA状态。

#### 16.6.5.1 测试小循环的完成情况

在使用软件发起的服务请求时，有两种方法可以测试小循环的完成情况。第一种是读取TCDn_CITER字段并测试是否有变化。
另一种方法可以从以下所示的序列中提取。第二种方法是

来测试TCDn_CSR[START]位和TCDn_CSR[ACTIVE]位。在TCDn_CSR[START]被设置后，两个位的读数均为零，则表示小环路完成状态。轮询TCDn_CSR[ACTIVE]位可能是不确定的，因为如果通道执行的时间很短，可能会错过活动状态。

对于一个软件激活的通道，TCD状态位执行以下顺序。

| 舞台 | TCDn_CSR位 | | | 国家 |
|---|---|---|---|---|
| | 开始 | 激活的 | 已完成 | |
| 1 | 1 | 0 | 0 | 通过软件进行渠道服务请求 |
| 2 | 0 | 1 | 0 | 频道正在执行中 |
| 3a | 0 | 0 | 0 | 通道已完成小循环，处于空闲状态 |
| 3b | 0 | 0 | 1 | 通道已完成主要循环，处于空闲状态 |

在使用硬件（即外设）发起的服务请求时，测试小环路完成的最佳方法是读取TCDn_CITER字段并测试是否有变化。硬件请求和确认握手信号在程序员的模型中是不可见的。

对于一个硬件激活的通道，TCD状态位执行以下顺序。

| 舞台 | TCDn_CSR位 | | | 国家 |
|---|---|---|---|---|
| | 开始 | 激活的 | 已完成 | |
| 1 | 0 | 0 | 0 | 通过硬件的通道服务请求（外设请求被断言）。 |
| 2 | 0 | 1 | 0 | 频道正在执行中 |
| 3a | 0 | 0 | 0 | 通道已完成小循环，处于空闲状态 |
| 3b | 0 | 0 | 1 | 通道已完成主要循环，处于空闲状态 |

对于这两种激活类型，主要环路的完成状态通过TCDn_CSR[DONE]位明确指示。

无论通道如何激活，当通道开始执行时，TCDn_CSR[START]位将自动清零。

### 16.6.5.2 读取活动通道的传输描述符

如果在通道执行时读到TCDn_SADDR、TCDn_DADDR和TCDn_NBYTES的真实值，eDMA会读回来。SADDR、DADDR和NBYTES的真实值是eDMA引擎当前在其内部寄存器文件中使用的值，而不是该通道的TCD本地存储器中的值。这些地址

，SADDR和

DADDR和NBYTES在传输过程中递减为零，可以显示出传输的进度。所有其他的值都是从TCD本地存储器中读回来的。

#### 16.6.5.3 检查通道抢占状态

只有当固定仲裁被选为通道仲裁模式时，抢占才可用。抢占情况是指启用抢占的信道运行时，更高的优先级请求成为活动的。当eDMA引擎不在固定信道仲裁模式下运行时，确定主动运行的相对优先级未完成的请求变得不确定。当选择Round-Robin仲裁模式时，信道优先级被视为平等的，也就是不断轮换的。

在整个抢占过程中，被抢占通道的TCDn_CSR[ACTIVE]位仍被引爆。当抢占通道执行一个主要的循环迭代时，被抢占的通道被暂时中止。如果在全局 TCD 地图中，两个
TCDn_CSR[ACTIVE]位同时被设置，则表示一个较高优先级的通道正在主动抢占一个较低优先级的通道。

### 16.6.6 通道 链接

通道连接（或连锁）是一种机制，一个通道设置另一个通道（或其本身）的TCDn_CSR[START]位，因此启动该通道的服务请求。当正确启用时，EDMA引擎在主要或次要循环完成时自动执行这一操作。

小环路通道链接在小环路（或大环路的一次迭代）完成时发生。TCDn_CITER[E_LINK]字段决定是否要求进行小循环链接。当启用时，除最后一次外，在主循环的每次迭代后进行通道链接。当主循环耗尽时，只有主循环通道链接字段被用来决定是否要做通道链接。例如，初始字段为。

```
TCDn_CITER[E_LINK] = 1
TCDn_CITER[LINKCH] = 0xC
TCDn_CITER[CITER]值 = 0x4
TCDn_CSR[MAJOR_E_LINK] = 1
TCDn_CSR[MAJOR_LINKCH] = 0x7
```

执行为。

1. 小循环完成→设置TCD12_CSR[START]位

2. 小循环完成→设置TCD12_CSR[START]位
3. 小循环完成→设置TCD12_CSR[START]位
4. 小循环完成，大循环完成→设置TCD7_CSR[START]位

当启用小循环链接时（TCDn_CITER[E_LINK] = 1），TCDn_CITER[CITER]字段使用一个 9 位矢量来构成当前的迭代计数。当小循环链接被禁用时（TCDn_CITER[E_LINK] = 0），TCDn_CITER[CITER]字段使用一个 15 位矢量来构成当前的迭代计数。与 TCDn_CITER[LINKCH]字段相关的位被串联到 CITER 值上以增加 CITER 的范围。

**注意事项**

TCDn_CITER[E_LINK]位和TCDn_BITER[E_LINK]位必须相等，否则会报告配置错误。CITER和BITER矢量宽度必须相等，以计算主要循环、半程中断点。

下表总结了一个DMA通道如何在一个循环的末端链接到另一个DMA通道，即使用另一个通道的TCD。

**表16-7.通道连接参数**

| 希望的链接行为 | TCD控制字段名称 | 描述 |
|---|---|---|
| 小环线末端的链接 | CITER[E_LINK] | 在小循环完成时启用通道到通道的链接（当前迭代）。 |
| | CITER[LINKCH] | 在小循环结束时链接通道的编号（当前迭代）。 |
| 大环线末端的链接 | csr[main_e_link] | 在主要环路完成时启用通道与通道之间的链接 |
| | csr[major_linkch] | 在主要环路末端链接时，链接通道号码 |

## 16.6.7 动态 编程

本节提供了在通道执行期间改变编程模型的推荐方法。

### 16.6.7.1 动态改变频道优先级

以下两个选项被推荐用于动态改变信道优先级水平。

1. 切换到Round-Robin通道仲裁模式，改变通道优先级，然后切换回固定仲裁模式。
2. 禁用所有通道，改变通道优先级，然后启用适当的通道。

### 16.6.7.2 动态通道连接

动态通道连接是在通道执行过程中设置TCDn_CSR[MAJORELINK]位的过程（见TCD结构图）。该位在通道执行结束时从TCD本地存储器中读出，从而允许用户在通道执行期间启用该功能。

因为允许用户在执行过程中改变配置，所以需要一个一致性模型。考虑这样的情况：用户在eDMA引擎检索通道的同时，试图通过启用TCDn_CSR[MAJORELINK]位来执行动态通道链接。在程序员的模型中，TCDn_CSR[MAJORELINK]将被设置，但不清楚实际的链接是否在通道退役前进行。

在执行动态通道链接请求时，建议采用以下一致性模型。

1. 向TCDn_CSR[MAJORELINK]位写1。
2. 回读TCDn_CSR[MAJORELINK]位。
3. 测试TCDn_CSR[MAJORELINK]请求状态。
   - 如果TCDn_CSR[MAJORELINK]=1，说明动态链接尝试成功。
   - 如果TCDn_CSR[MAJORELINK]=0，说明尝试的动态链接没有成功（该通道已经退役）。

对于这个请求，TCD本地内存控制器在某个通道的TCD.word7位被设置后，在对该通道的TCD.word7的任何写操作上强制将TCDn_CSR[MAJORELINK]位设为零，表示主循环已经完成。

**注意事项**

在写入 TCDn_CSR[MAJORELINK] 位之前，用户必须清除 TCDn_CSR[DONE] 位。TCDn_CSR[DONE]位在一个通道开始执行后被 eDMA 引擎自动清除。

### 16.6.7.3 动态散射/聚集

Scatter/gather是自动加载一个新的TCD到通道中的过程。它允许一个DMA通道使用多个TCD；这使一个DMA通道能够将DMA数据分散到多个目的地或从多个来源收集数据。当分散/收集被启用并且通道已经完成了它的主要循环，一个新的TCD被从系统内存中获取并加载到eDMA程序员模型中该通道的描述符位置，从而替代了当前的描述符。

因为允许用户在执行过程中改变配置，所以需要一个一致性模型。考虑这样的情况：用户在eDMA引擎检索信道的同时，试图通过启用TCDn_CSR[ESG]位来执行一个动态的分散/聚集操作。在程序员的模型中，ESG位将被设置，但不清楚在信道退役前实际的分散/聚集请求是否被兑现。

这种一致性模型的两种方法在下面的小节中显示。方法1的优点是用一次读取MAJORLINKCH字段和ESG位。对于动态通道链接和分散/收集请求，如果一个通道的TCD.DONE位被设置，表明主要循环已经完成，那么TCD本地内存控制器在对该通道的TCD字7进行任何写入时，强制将TCD MAJOR.E_LINK和E_SG位设为0。

**注意事项**

在写入 MAJORELINK 或 ESG 位之前，用户必须清除 TCDn_CSR[DONE]位。TCDn_CSR[DONE]位在一个通道开始执行后被eDMA引擎自动清除。

#### 16.6.7.3.1 方法1（不使用主要循环通道连接的通道）

对于不使用主要循环通道连接的通道，这里描述的一致性模型可用于动态散射/收集请求。

当 TCDn_CSR[MAJORELINK]位为零时，TCDn_CSR[MAJORLINKCH] 字段不被 eDMA
使用。在这种情况下，MAJORLINKCH字段可以用于其他目的。这种方法使用MAJORLINKCH字段作为TCD的识别（ID）。

1. 当描述符建立后，在TCDn_CSR[MAJORLINKCH]字段中为每个与通道相关的TCD写一个唯一的TCD ID，使用动态散射/收集。
2. 向TCDn_CSR[DREQ]位写1b。

如果一个动态散射/收集尝试失败，设置DREQ位将阻止这个通道未来的硬件激活。这将阻止通道以目标地址（DADDR）执行，该地址是用散射/聚集地址（在下一步写入）而不是dlast最终偏移值计算的。

3. 用散射/聚集地址写入TCDn_DLASTSGA寄存器。
4. 向 TCDn_CSR[ESG]位写 1b。
5. 读回16位TCD控制/状态字段。
6. 测试TCDn_CSR寄存器中的ESG请求状态和MAJORLINKCH值。如果ESG = 1b，说明动态链接的尝试是成功的。

   如果ESG=0b并且MAJORLINKCH(ID)没有变化，那么尝试的动态链接没有成功（通道已经退役）。

   如果ESG=0b并且MAJORLINKCH(ID)改变了，那么动态链接尝试是成功的（新TCD的E_SG值清除了ESG位）。

#### 16.6.7.3.2 方法2（使用主要循环通道连接的通道）

对于使用主循环信道连接的信道，这里描述的一致性模型可用于动态散射/收集请求。这种方法使用TCD.DLAST_SGA字段作为TCD标识（ID）。

1. 向TCDn_CSR[DREQ]位写1b。

   如果一个动态散射/收集尝试失败，设置DREQ位将阻止这个通道未来的硬件激活。这将阻止通道以目标地址（DADDR）执行，该地址是用分散/聚集地址（在下一步写入）而不是dlast最终偏移值计算的。

2. 用散射/聚集地址写入TCDn_DLASTSGA寄存器。
3. 向TCDn_CSR[ESG]位写1b。
4. 回读ESG位。
5. 测试ESG请求状态。

   如果ESG = 1b，说明动态链接尝试成功。如果ESG = 0b，则读取32位TCDn_DLASTSGA字段。

   如果ESG=0b且TCDn_DLASTSGA没有变化，则尝试的动态链接没有成功（通道

已经退役）。

如果ESG=0b且TCDn_DLASTSGA发生变化，则说明动态链接尝试成功（新TCD的E_SG值清除了ESG位）。

## 16.6.8 暂停/恢复一个有活动的DMA通道 硬件服务请求

DMA允许用户将数据从内存或外设寄存器移动到内存或外设寄存器的另一个位置，而不需要CPU交互。一旦DMA和外设被配置好并处于活动状态，动态地中止外设的服务请求是很罕见的。在这种情况下，停用DMA硬件服务请求有一定的限制。对于连贯性，必须遵循一个特定的程序。本节提供了指导，当DMA被一个从属模块如串行外设接口（DSPI）、Sigma Delta模数转换器（SDADC）或其他模块触发时，如何连贯地暂停和恢复直接内存访问（DMA）通道。

### 16.6.8.1 暂停一个活动的DMA通道

要暂停一个活动的DMA通道。

1. 首先停止外设的DMA服务请求。通过读回外设中相应的寄存器来确认它已经被禁用。
2. 检查DMA的硬件请求状态寄存器（DMA_HRSn），以确保没有对被暂停的DMA通道的服务请求。然后通过清除相应DMA通道上的ERQ位来禁用硬件服务请求。

### 16.6.8.2 恢复一个DMA通道

要恢复一个DMA通道。

1. 通过设置其ERQ位来启用相应通道上的DMA服务请求。
2. 启用外设的DMA服务请求。

例如，假设当DSPI_TXFIFO有一个空槽时，DSPI被设置为通过DMA服务请求来传输数据的主站。DMA会在请求后将下一个命令和数据传输到TXFIFO。如果用户需要暂停DMA/DSPI传输循环，请执行以下步骤。

1. 通过向DSPI_RSER[TFFF_RE]写0，在源端禁用DMA服务请求。确认DSPI_RSER[TFFF_RE]为0。

2. 通过验证相应通道的DMA_HRS[HRSn]为0，确保没有来自DSPI的DMA服务请求。如果没有服务请求，通过清除该通道的ERQ位来禁用DMA通道。如果有服务请求，请等待，直到请求被处理，HRS位读数为0。

# 章 17
# 触发器MUX控制 (TRGMUX)

## 17.1 特定芯片的TRGMUX 信息

### 17.1.1 模块 互联性

TRGMUX提供了一个极其灵活的机制，将各种触发源连接到多个引脚/外设。

可用的输入触发器见图17-2和图17-3。

通过TRGMUX，每个接受外部触发器的外设通常有一个特定的32位触发器控制寄存器。每个控制寄存器最多支持四个触发器，每个触发器可以从可用的输入触发器中选择。

下图显示了TRGMUX的主要结构，以模块_A为例。

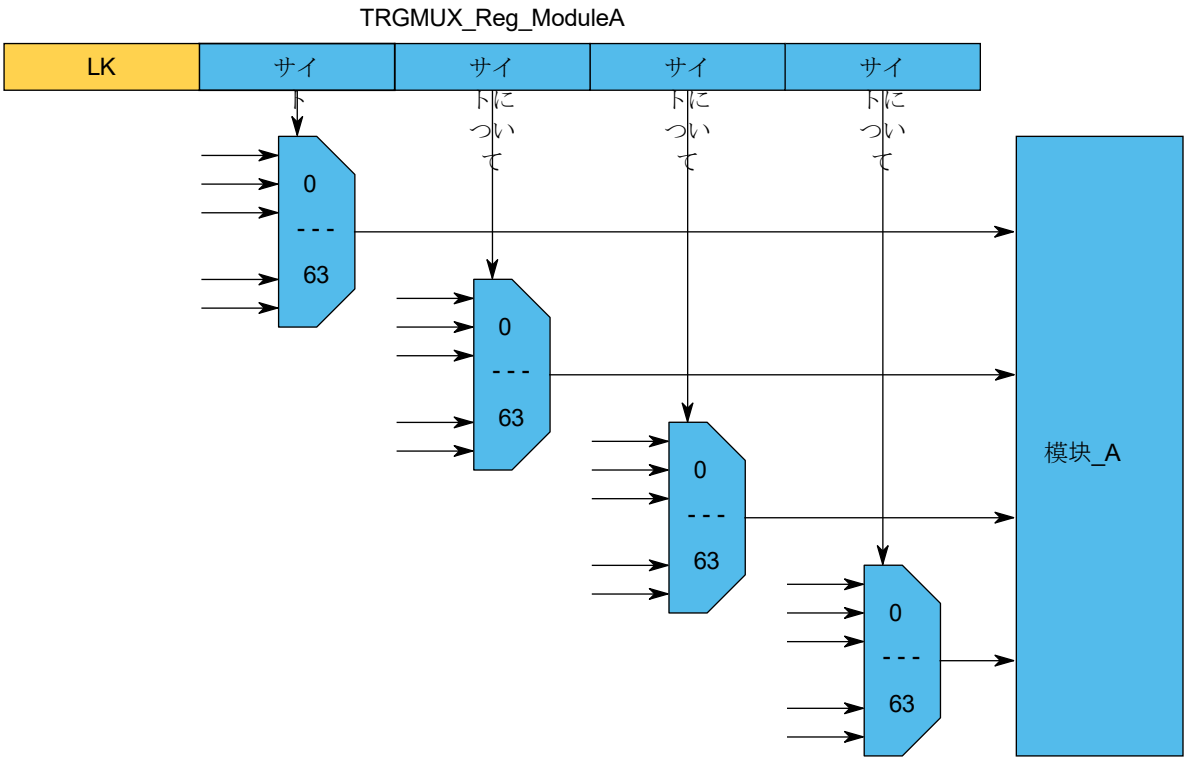
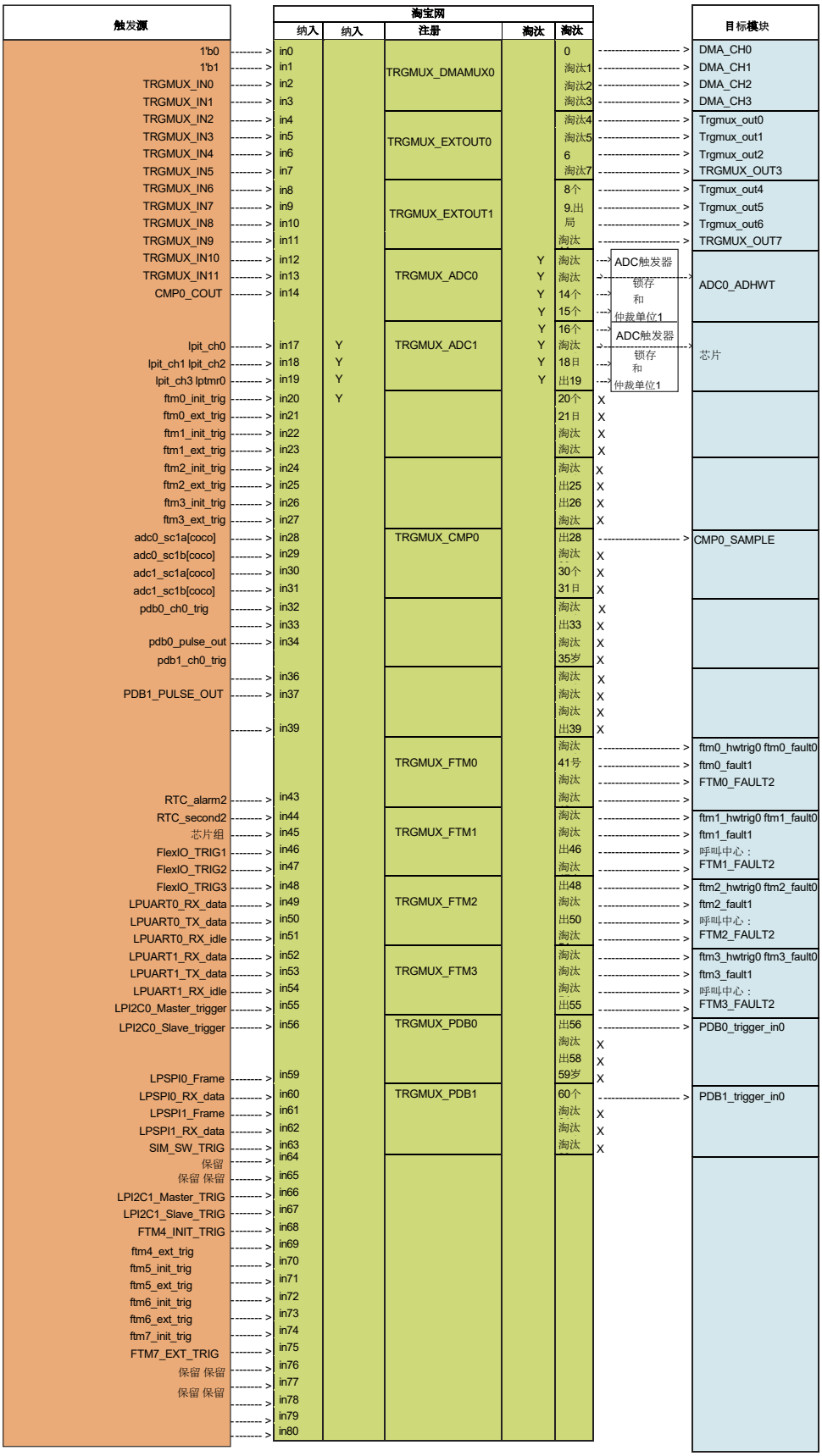



**图17-1.TRGMUX结构示例（64个输入）。**

### 注意事项

每个TRGMUX控制寄存器最多支持四个触发通道，但每个模块不一定实现所有四个触发器。需要四个以上触发输入的模块（如用于外部输出）有多个控制寄存器。

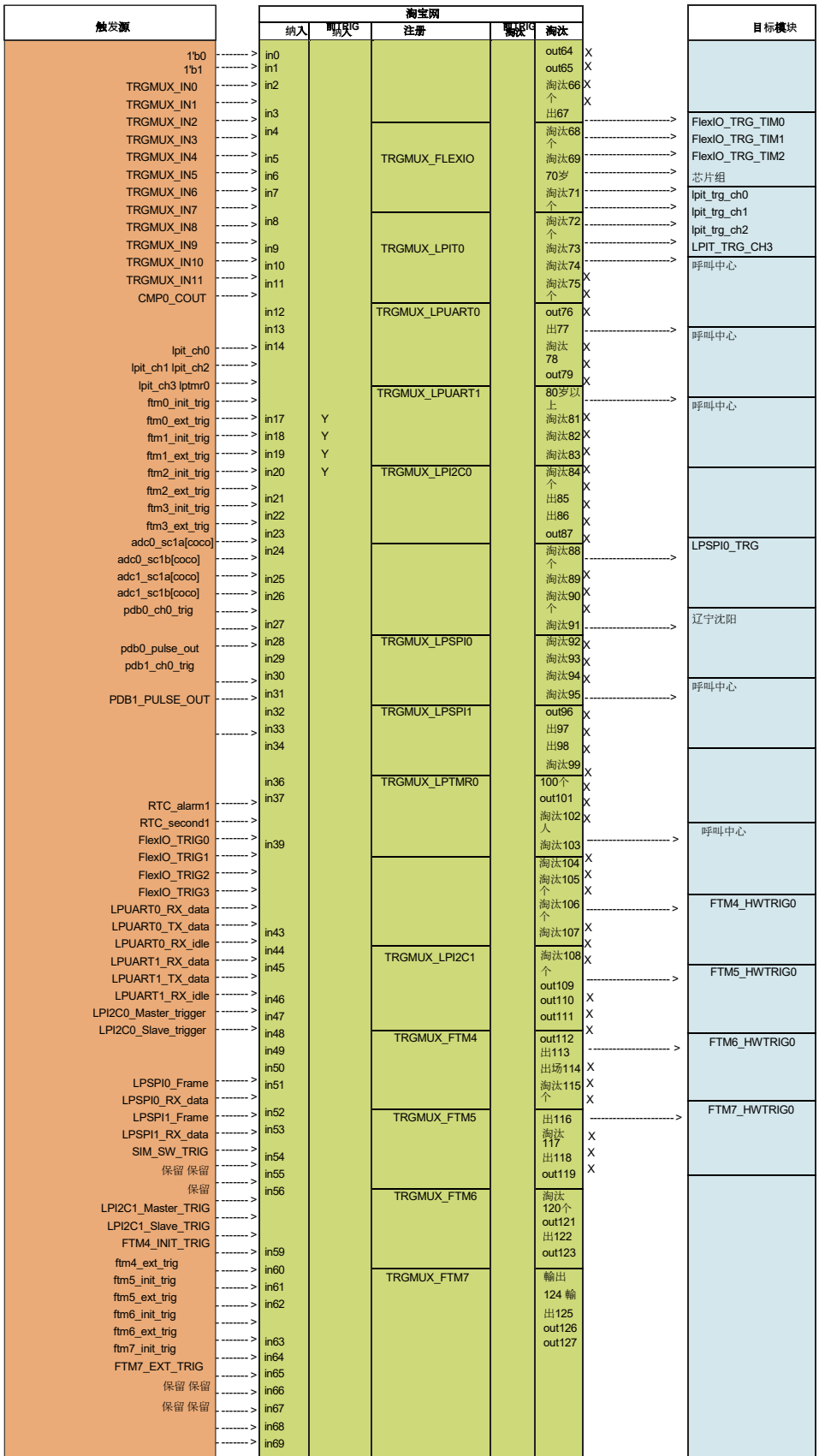
下图显示了S32K1xx系列的触发输入、输出和控制寄存器的超集。关于使用触发方案时必须遵守的建议，请参见附件S32K1xx_Trigger_Muxing.xlsx的细节。




1 在ADC配置一章中，详见 "触发器锁存和仲裁 "部分。

2 在期待TRGMUX的RTC_alarm和RTC_second触发之前，需要配置中断使能。

芯片特定的TRGMUX信息 图17-2S 3T2Kri1gxgxeSreimietserRceofenrenneccetiMvaitnyua(pl, a. 4o, f062/: 20o1u7tputs 0-63)




1 在期待TRGMUX的RTC_alarm和RTC 显示了所有的连接。在某一特定部分没有实例的插槽是保留的。

图17-3.触发器的互连性（第2部分：输出64-127）。

**注意事项**

关于ADC触发器的其他细节包括。

- 对于每个ADC来说，四个触发器被OR在一起，为每个ADC的硬件触发提供一个灵活的方案。预触发器没有被OR。
- 只有LPIT支持预触发器。
- 如果其他外设需要预触发器，必须使用软件预触发器。参见SIM_ADCOPT[ADCxSWPRETRG]以配置软件预触发器。
- 关于ADC触发器实现的更多信息，包括不涉及TRGMUX的PDB预触发方案，请参阅ADC触发源。

### 17.1.2 芯片特定的TRGMUX 寄存器

一些TRGMUX寄存器和输入连接仅存在于某些S32K1xx产品上。在特定变体中不可用的寄存器和输入连接被保留。

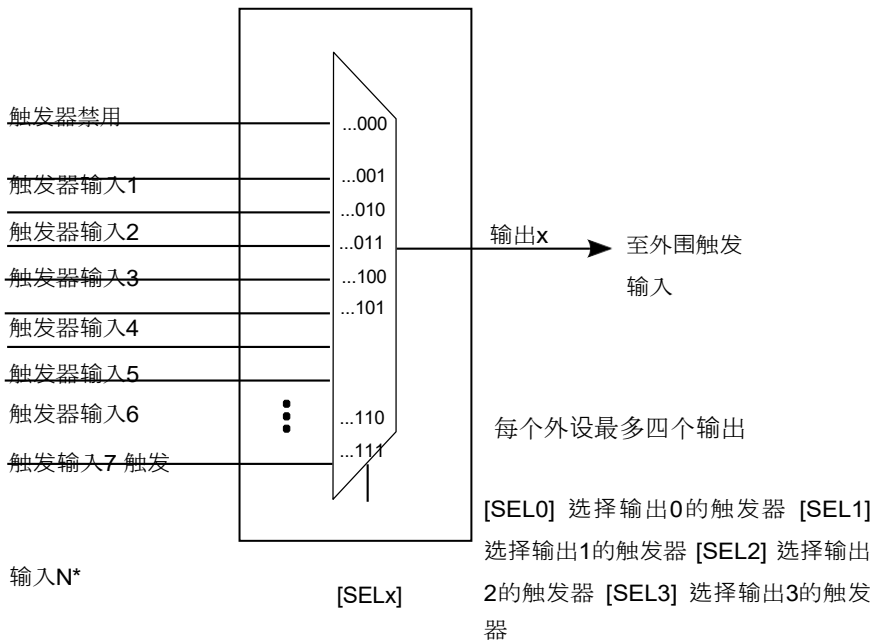
## 17.2 简介

触发器多路复用器（TRGMUX）模块允许软件为各种外设配置触发输入。

## 17.3 特点

TRGMUX模块允许软件选择外围设备的触发源。下面的框图显示了TRGMUX模块的触发选择逻辑。

淘宝网




* 可能有多达127个可选择的触发输入。请参见芯片特定的TRGMUX信息，该设备支持的最大触发器输入数。

**图17-4.TRGMUX框图**

每个外设都有自己的专用TRGMUX寄存器。详见每个外设的TRGMUX寄存器。

## 17.4 内存地图和寄存器 定义

TRGMUX寄存器包含为外围模块选择触发源的字段。

TRGMUX寄存器只能在监督员模式下被写入。

### 17.4.1 TRGMUX 寄存器 描述

#### 17.4.1.1 TRGMUX 内存图

**表17-1.选择位字段**

| 场地 | 职能 |
| --- | --- |
| サイトについて | 该读/写字段用于配置外围触发输入的MUX选择。000_0000 - (0x00) -<br><br>000_0001 - (0x01) VDD |

表17-1 选择位字段

| 场地 | 职能 |
| --- | --- |
| | 000_0010 - (0x02) TRGMUX_IN0<br>000_0011 - (0x03) TRGMUX_IN1<br>000_0100 - (0x04) TRGMUX_IN2<br>000_0101 - (0x05) TRGMUX_IN3<br>000_0110 - (0x06) TRGMUX_IN4<br>000_0111 - (0x07) TRGMUX_IN5<br>000_1000 - (0x08) TRGMUX_IN6<br>000_1001 - (0x09) TRGMUX_IN7<br>000_1010 - (0x0A) TRGMUX_IN8<br>000_1011 - (0x0B) TRGMUX_IN9<br>000_1100 - (0x0C) TRGMUX_IN10<br>000_1101 - (0x0D) TRGMUX_IN11<br>000_1110 - (0x0E) CMP0_OUT<br>000_1111 - (0x0F) 保留<br>001_0000 - (0x10) 保留<br>001_0001 - (0x11) LPIT_CH0<br>001_0010 - (0x12) LPIT_CH1<br>001_0011 - (0x13) LPIT_CH2<br>001_0100 - (0x14) LPIT_CH3<br>001_0101 - (0x15) LPTMR0<br>001_0110 - (0x16) FTM0_INIT_TRIG<br>001_0111 - (0x17) FTM0_EXT_TRIG<br>001_1000 - (0x18) FTM1_INIT_TRIG<br>001_1001 - (0x19) FTM1_EXT_TRIG<br>001_1010 - (0x1A) FTM2_INIT_TRIG<br>001_1011 - (0x1B) FTM2_EXT_TRIG<br>001_1100 - (0x1C) FTM3_INIT_TRIG<br>001_1101 - (0x1D) FTM3_EXT_TRIG<br>001_1110 - (0x1E) ADC0_SC1A[COCO]。<br>001_1111 - (0x1F) ADC0_SC1B[COCO]。<br>010_0000 - (0x20) ADC1_SC1A[COCO]<br>010_0001 - (0x21) ADC1_SC1B[COCO]。<br>010_0010 - (0x22) PDB0_CH0_TRIG<br>010_0011 - (0x23) 保留<br>010_0100 - (0x24) PDB0_PULSE_OUT<br>010_0101 - (0x25) PDB1_CH0_TRIG<br>010_0110 - (0x26) 保留<br>010_0111 - (0x27) PDB1_PULSE_OUT |

表17-1.选择位字段

| 场地 | 职能 |
| --- | --- |
| | 010_1000 - (0x28) 保留<br>010_1001 - (0x29) 保留<br>010_1010 - (0x2A) 保留<br>010_1011 - (0x2B) RTC_alarm<br>010_1100 - (0x2C) RTC_second<br>010_1101 - (0x2D) FlexIO_TRIG0<br>010_1110 - (0x2E) FlexIO_TRIG1<br>010_1111 - (0x2F) FlexIO_TRIG2<br>011_0000 - (0x30) FlexIO_TRIG3<br>011_0001 - (0x31) LPUART0_RX_data<br>011_0010 - (0x32) LPUART0_TX_data<br>011_0011 - (0x33) LPUART0_RX_idle<br>011_0100 - (0x34) LPUART1_RX_data<br>011_0101 - (0x35) LPUART1_TX_data<br>011_0110 - (0x36) LPUART1_RX_idle<br>011_0111 - (0x37) LPI2C0_Master_trigger<br>011_1000 - (0x38) LPI2C0_Slave_trigger<br>011_1001 - (0x39) Reserved<br>011_1010 - (0x3A) 保留 011_1011 -<br>(0x3B) LPSPI0_Frame<br>011_1100 - (0x3C) LPSPI0_RX_data<br>011_1101 - (0x3D) LPSPI1_Frame<br>011_1110 - (0x3E) LPSPI1_RX_data<br>011_1111 - (0x3F) SIM_SW_TRIG<br>100_0000 - (0x40) 保 留<br>100_0001 - (0x41) 保 留<br>100_0010 - (0x42) 保留<br>100_0011 - (0x43) LPI2C1_Master_trigger<br>100_0100 - (0x44) LPI2C1_Slave_trigger<br>100_0101 - (0x45) FTM4_INIT_TRIG<br>100_0110 - (0x46) FTM4_EXT_TRIG<br>100_0111 - (0x47) FTM5_INIT_TRIG<br>100_1000 - (0x48) FTM5_EXT_TRIG<br>100_1001 - (0x49) FTM6_INIT_TRIG<br>100_1010 - (0x4A) FTM6_EXT_TRIG<br>100_1011 - (0x4B) FTM7_INIT_TRIG<br>100_1100 - (0x4C) FTM7_EXT_TRIG<br>100_1101 - (0x4D) 保留 |

表17-1 选择位字段

| 场地 | 职能 |
| --- | --- |
| | 100_1110 - (0x4E) 保 留 |
| | 100_1111 - (0x4F) 保 留 |
| | 101_0000 - (0x50) 保 留 |
| | 101_0001 - (0x51) 保 留 |
| | 101_0010 - (0x52) 保 留 |
| | 101_0011 - (0x53) 保 留 |
| | 101_0100 - (0x54) 保 留 |
| | 101_0101 - (0x55) 保 留 |
| | 101_0110 - (0x56) 保 留 |
| | 101_0111(0x57) 保 留 |
| | 101_1000 - (0x58) 保 留 |
| | 101_1001 - (0x59) 保 留 |
| | 101_1010 - (0x5A) 保 留 |
| | 101_1011 - (0x5B) 保 留 |
| | 101_1100 - (0x5C) 保 留 |
| | 101_1101 - (0x5D) 保 留 |
| | 101_1110 - (0x5E) 保 留 |
| | 101_1111 - (0x5F) 保 留 |
| | 110_0000(0x60) 保 留 |
| | 110_0001 - (0x61) 保 留 |
| | 110_0010 - (0x62) 保 留 |
| | 110_0011 - (0x63) 保 留 |
| | 110_0100 - (0x64) 保 留 |
| | 110_0101 - (0x65) 保 留 |
| | 110_0110 - (0x66) 保 留 |
| | 110_0111 - (0x67) 保 留 |
| | 110_1000 - (0x68) 批 准 |
| | 110_1001(0x69) 保 留 |
| | 110_1010 - (0x6A) 保 留 |
| | 110_1011 - (0x6B) 保 留 |
| | 110_1100 - (0x6C) 保 留 |
| | 110_1101 - (0x6D) 保 留 |
| | 110_1110 - (0x6E) 保 留 |
| | 110_1111 - (0x6F) 保 留 |

表17-1 选择位字段

| | |
|---|---|
| | 111_0000 - (0x70) 保 留<br>111_0001 - (0x71) 保 留<br>111_0010 - (0x72) 保留<br>111_0011 - (0x73) 保留 |

表17-1.选择位字段

| 场地 | 职能 |
|---|---|
| | 111_0100 - (0x74) 保 留<br>111_0101 - (0x75) 保 留<br>111_0110 - (0x76) 保 留<br>111_0111 - (0x77) 保 留<br>111_1000 - (0x78) 保 留<br>111_1001 - (0x79) 保 留<br>111_1010 - (0x7A) 保 留<br>111_1011 - (0x7B) 保 留<br>111_1100 - (0x7C) 保 留<br>111_1101 - (0x7D) 保 留<br>11_1110 - (0x7E) 保留<br>111_1111 - (0x7F) 保留 |

TRGMUX基础地址。4006_3000h

| 偏移 | 注册 | 宽度<br>(单位：比特) | 访问 | 重置值 |
|---|---|---|---|---|
| 0h | TRGMUX DMAMUX0寄存器（DMAMUX0）。 | 32 | RW | 0000_0000h |
| 4h | TRGMUX EXTOUT0寄存器（EXTOUT0）。 | 32 | RW | 0000_0000h |
| 8h | TRGMUX EXTOUT1寄存器（EXTOUT1）。 | 32 | RW | 0000_0000h |
| 丘吉尔 | TRGMUX ADC0寄存器（ADC0）。 | 32 | RW | 0000_0000h |
| 10h | TRGMUX ADC1寄存器（ADC1） | 32 | RW | 0000_0000h |
| 1Ch | TRGMUX CMP0寄存器（CMP0）。 | 32 | RW | 0000_0000h |
| 28h | TRGMUX FTM0寄存器（FTM0）。 | 32 | RW | 0000_0000h |
| 2Ch | TRGMUX FTM1寄存器（FTM1）。 | 32 | RW | 0000_0000h |
| 30h | TRGMUX FTM2寄存器（FTM2）。 | 32 | RW | 0000_0000h |
| 34h | TRGMUX FTM3寄存器（FTM3）。 | 32 | RW | 0000_0000h |
| 38h | TRGMUX PDB0寄存器（PDB0）。 | 32 | RW | 0000_0000h |
| 3Ch | TRGMUX PDB1寄存器（PDB1）。 | 32 | RW | 0000_0000h |
| 44h | TRGMUX FLEXIO寄存器（FLEXIO）。 | 32 | RW | 0000_0000h |
| 48h | TRGMUX LPIT0寄存器（LPIT0）。 | 32 | RW | 0000_0000h |
| 4Ch | TRGMUX LPUART0寄存器（LPUART0） | 32 | RW | 0000_0000h |
| 50h | TRGMUX LPUART1寄存器（LPUART1） | 32 | RW | 0000_0000h |
| 54h | TRGMUX LPI2C0寄存器（LPI2C0）。 | 32 | RW | 0000_0000h |
| 5Ch | TRGMUX LPSPI0寄存器（LPSPI0）。 | 32 | RW | 0000_0000h |
| 60h | TRGMUX LPSPI1寄存器（LPSPI1）。 | 32 | RW | 0000_0000h |
| 64h | TRGMUX LPTMR0寄存器（LPTMR0）。 | 32 | RW | 0000_0000h |

表17-1.选择位字段

| | | | | |
|---|---|---|---|---|
| 6Ch | TRGMUX LPI2C1寄存器（LPI2C1）。 | 32 | RW | 0000_0000h |

*表继续在下一页...*

| 偏移 | 注册 | 宽度<br>(单位：比特) | 访问 | 重置值 |
|---|---|---|---|---|
| 70h | TRGMUX FTM4寄存器（FTM4）。 | 32 | RW | 0000_0000h |
| 74h | TRGMUX FTM5寄存器（FTM5）。 | 32 | RW | 0000_0000h |
| 78h | TRGMUX FTM6寄存器（FTM6）。 | 32 | RW | 0000_0000h |
| 7Ch | TRGMUX FTM7寄存器（FTM7）。 | 32 | RW | 0000_0000h |

## 17.4.1.2 TRGMUX DMAMUX0寄存器（DMAMUX0）。

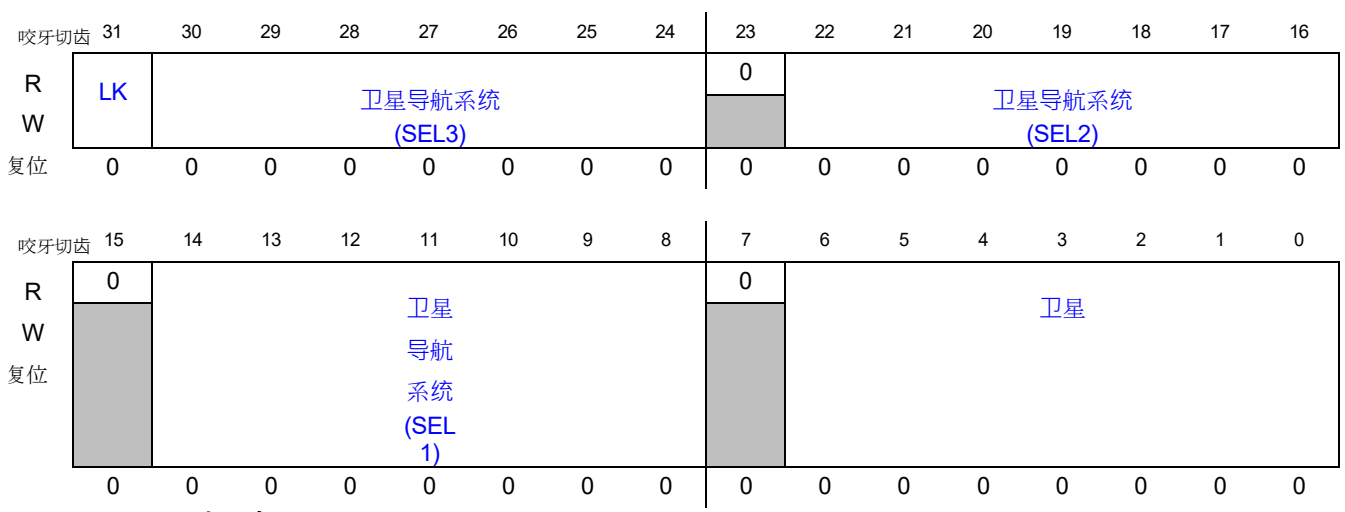
### 17.4.1.2.1 偏移

| 注册 | 偏移 |
|---|---|
| DMAMUX0 | 0h |

### 17.4.1.2.2 职能

TRGMUX 寄存器

### 17.4.1.2.3 图示




### 17.4.1.2.4 领域

| 场地 | 职能 |
|---|---|

| | |
|---|---|
| 31<br>LK | TRGMUX寄存器锁定。<br>这个位显示寄存器是否可以被写入。LK位在任何系统复位后只能被写入一次。一旦LK被设置，该TRGMUX寄存器中的SELx位不能被改变，直到下一次系统复位清除LK。<br>0b - 寄存器可以被写入。 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | 1b - 在下一次系统复位前不能写入寄存器。 |
| 30-24<br>卫星导航系统(SEL3) | 触发器MUX输入3源选择<br>该读/写位域用于配置外设触发输入3的MUX选择。 关于该域的设置定义，请参见内存图和寄存器定义。 |
| 23<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 22-16<br>卫星导航系统(SEL2) | 触发器MUX输入2来源选择<br>这个读/写位字段用于配置外设触发输入2的MUX选择。 关于字段设置的定义，见内存图和寄存器定义。 |
| 15<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 14-8<br>卫星导航系统(SEL1) | 触发器MUX输入1来源选择<br>这个读/写位字段用于配置外设触发输入1的MUX选择。 关于字段设置的定义，见内存图和寄存器定义。 |
| 7<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 6-0<br>卫星 | 触发器MUX输入0来源选择<br>这个读/写位字段用于配置外设触发输入0的MUX选择。 关于字段设置的定义，见内存图和寄存器定义。 |

## 17.4.1.3 TRGMUX EXTOUT0寄存器（EXTOUT0）。

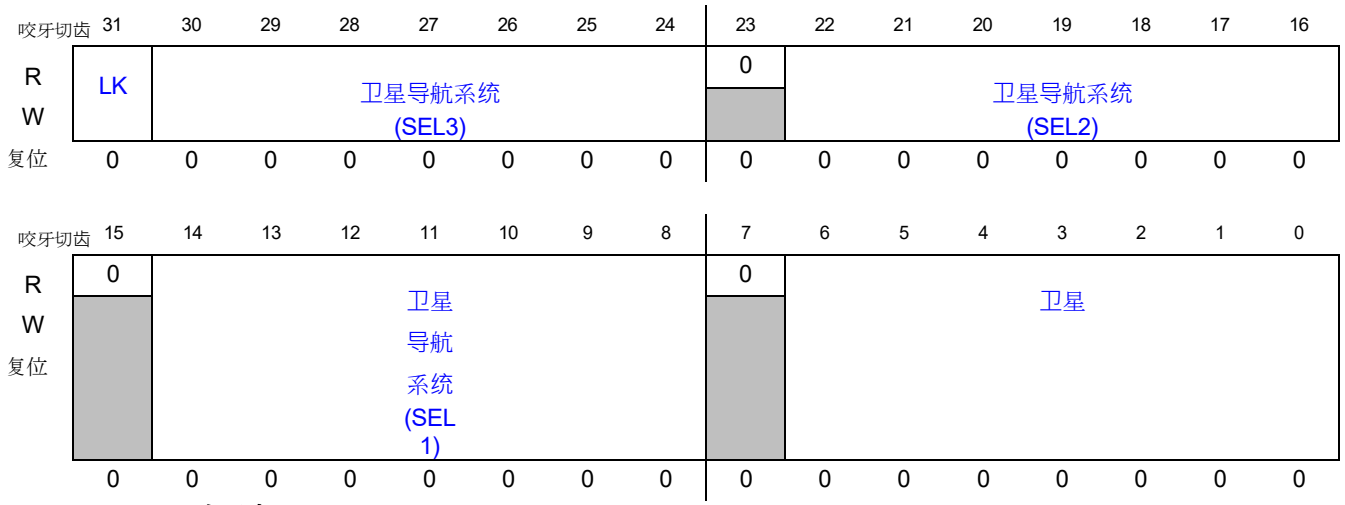
### 17.4.1.3.1 偏移

| 注册 | 偏移 |
| --- | --- |
| EXTOUT0 | 4h |

### 17.4.1.3.2 职能

TRGMUX 寄存器

### 17.4.1.3.3 图示




### 17.4.1.3.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>LK | TRGMUX寄存器锁定。<br>这个位显示寄存器是否可以被写入。LK位在任何系统复位后只能被写入一次。一旦LK被设置，该TRGMUX寄存器中的SELx位不能被改变，直到下一次系统复位清除LK。<br>0b - 寄存器可以被写入。<br>1b - 在下一次系统复位前不能写入寄存器。 |
| 30-24<br>卫星导航系统 (SEL3) | 触发器MUX输入3源选择<br>该读/写位域用于配置外设触发输入3的MUX选择。 关于该域的设置定义，请参见内存图和寄存器定义。 |
| 23<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 22-16<br>卫星导航系统 (SEL2) | 触发器MUX输入2来源选择<br>这个读/写位字段用于配置外设触发输入2的MUX选择。 关于字段设置的定义，见内存图和寄存器定义。 |
| 15<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 14-8<br>卫星导航系统 (SEL1) | 触发器MUX输入1来源选择<br>这个读/写位字段用于配置外设触发输入1的MUX选择。 关于字段设置的定义，见内存图和寄存器定义。 |
| 7<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 6-0<br>卫星 | 触发器MUX输入0来源选择<br>这个读/写位字段用于配置外设触发输入0的MUX选择。 关于字段设置的定义，见内存图和寄存器定义。 |

# 17.4.1.4 TRGMUX EXTOUT1寄存器（EXTOUT1）。

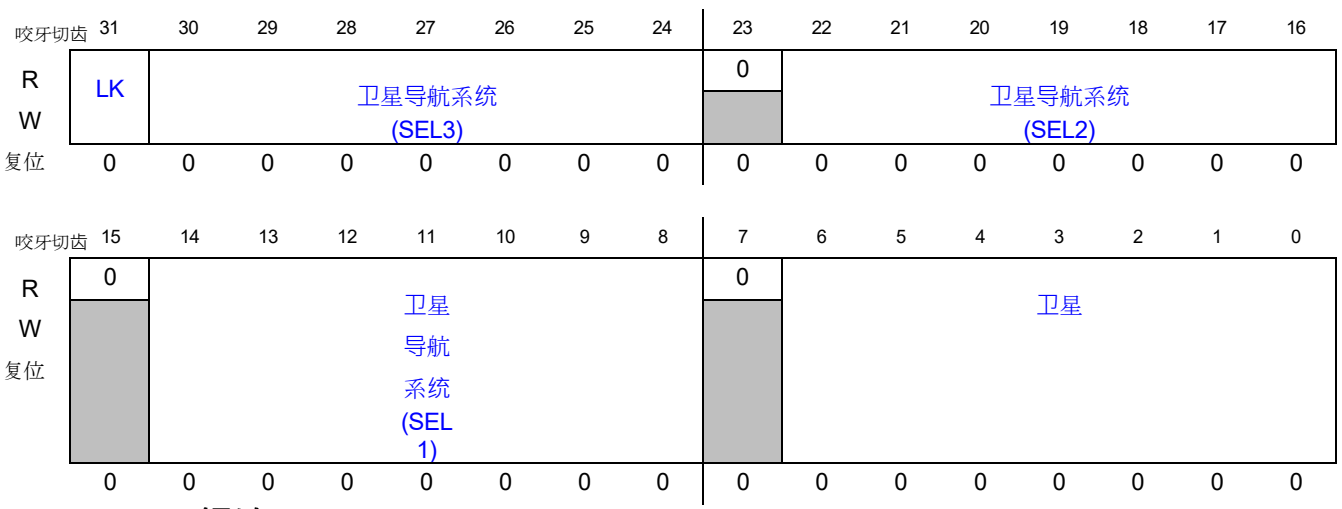
## 17.4.1.4.1 偏移

| 注册 | 偏移 |
|---|---|
| EXTOUT1 | 8h |

## 17.4.1.4.2 职能

TRGMUX 寄存器

## 17.4.1.4.3 图示




## 17.4.1.4.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>LK | TRGMUX寄存器锁定。<br>这个位显示寄存器是否可以被写入。LK位在任何系统复位后只能被写入一次。一旦LK被设置，该TRGMUX寄存器中的SELx位不能被改变，直到下一次系统复位清除LK。<br>0b - 寄存器可以被写入。<br>1b - 在下一次系统复位前不能写入寄存器。 |
| 30-24<br>卫星导航系统 (SEL3) | 触发器MUX输入3源选择<br>该读/写位域用于配置外设触发输入3的MUX选择。 关于该域的设置定义，请参见内存图和寄存器定义。 |
| 23<br>- | 这个只读的位域是保留的，其值始终为0。 |

| | |
|---|---|
| 22-16<br>卫星导航系统(SEL2) | 触发器MUX输入2来源选择<br>这个读/写位字段用于配置外设触发输入2的MUX选择。 关于字段设置的定义，见内存图和寄存器定义。 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| 15<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 14-8<br>卫星导航系统(SEL1) | 触发器MUX输入1来源选择<br>这个读/写位字段用于配置外设触发输入1的MUX选择。 关于字段设置的定义，见内存图和寄存器定义。 |
| 7<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 6-0<br>卫星 | 触发器MUX输入0来源选择<br>这个读/写位字段用于配置外设触发输入0的MUX选择。 关于字段设置的定义，见内存图和寄存器定义。 |

## 17.4.1.5 TRGMUX ADC0寄存器（ADC0）。

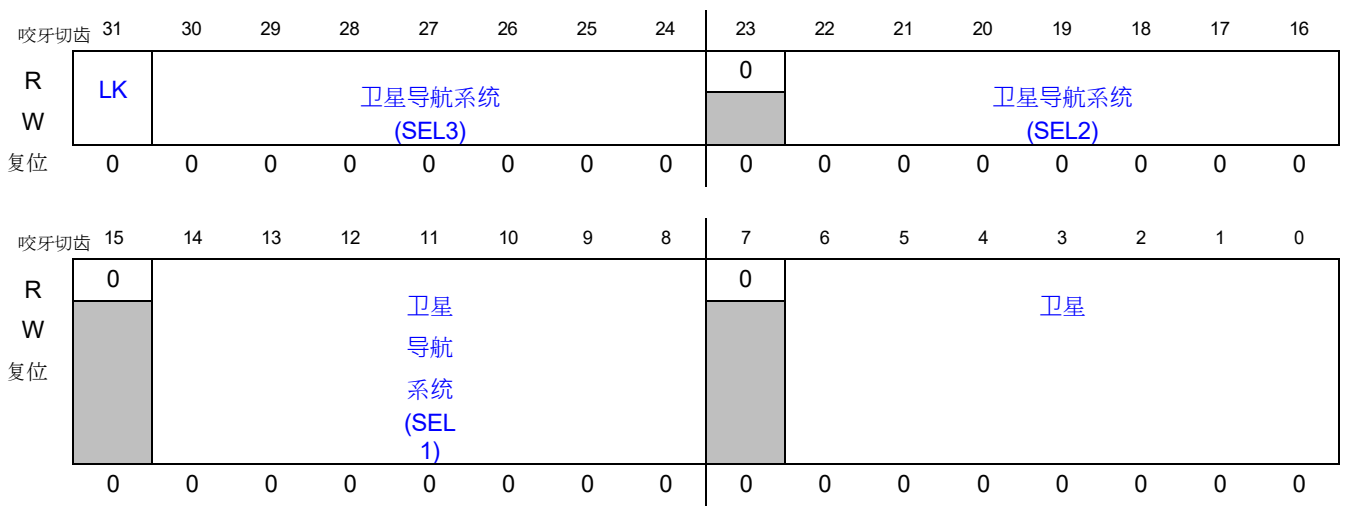
### 17.4.1.5.1 偏移

| 注册 | 偏移 |
|---|---|
| ADC0 | 丘吉尔 |

### 17.4.1.5.2 职能

TRGMUX 寄存器

### 17.4.1.5.3 图示



#### 17.4.1.5.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>LK | TRGMUX寄存器锁定。<br>这个位显示寄存器是否可以被写入。LK位在任何系统复位后只能被写入一次。一旦LK被设置，该TRGMUX寄存器中的SELx位不能被改变，直到下一次系统复位清除LK。<br>0b - 寄存器可以被写入。<br>1b - 在下一次系统复位前不能写入寄存器。 |
| 30-24<br>卫星导航系统(SEL3) | 触发器MUX输入3源选择<br>该读/写位域用于配置外设触发输入3的MUX选择。 关于该域的设置定义，请参见内存图和寄存器定义。 |
| 23<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 22-16<br>卫星导航系统(SEL2) | 触发器MUX输入2来源选择<br>这个读/写位字段用于配置外设触发输入2的MUX选择。 关于字段设置的定义，见内存图和寄存器定义。 |
| 15<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 14-8<br>卫星导航系统(SEL1) | 触发器MUX输入1来源选择<br>这个读/写位字段用于配置外设触发输入1的MUX选择。 关于字段设置的定义，见内存图和寄存器定义。 |
| 7<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 6-0<br>卫星 | 触发器MUX输入0来源选择<br>这个读/写位字段用于配置外设触发输入0的MUX选择。 关于字段设置的定义，见内存图和寄存器定义。 |

## 17.4.1.6 TRGMUX ADC1寄存器（ADC1）

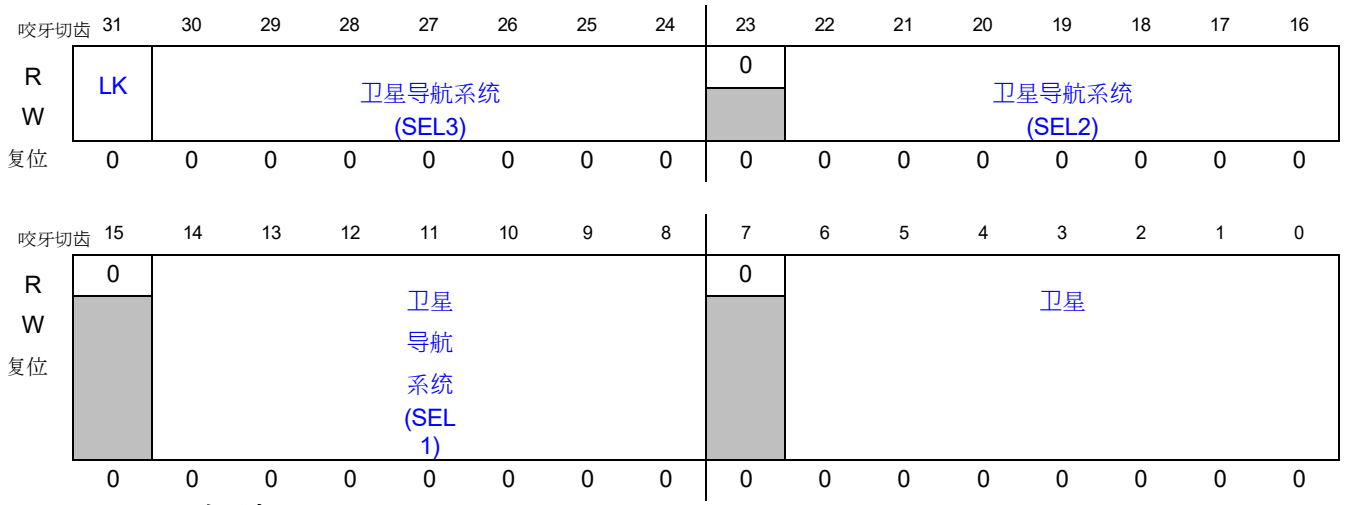
### 17.4.1.6.1 偏移

| 注册 | 偏移 |
|---|---|
| ADC1 | 10h |

### 17.4.1.6.2 职能

TRGMUX 寄存器

### 17.4.1.6.3 图示




### 17.4.1.6.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>LK | TRGMUX寄存器锁定。<br>这个位显示寄存器是否可以被写入。LK位在任何系统复位后只能被写入一次。一旦LK被设置，该TRGMUX寄存器中的SELx位不能被改变，直到下一次系统复位清除LK。<br>0b - 寄存器可以被写入。<br>1b - 在下一次系统复位前不能写入寄存器。 |
| 30-24<br>卫星导航系统 (SEL3) | 触发器MUX输入3源选择<br>该读/写位域用于配置外设触发输入3的MUX选择。 关于该域的设置定义，请参见内存图和寄存器定义。 |
| 23<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 22-16<br>卫星导航系统 (SEL2) | 触发器MUX输入2来源选择<br>这个读/写位字段用于配置外设触发输入2的MUX选择。 关于字段设置的定义，见内存图和寄存器定义。 |
| 15<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 14-8<br>卫星导航系统 (SEL1) | 触发器MUX输入1来源选择<br>这个读/写位字段用于配置外设触发输入1的MUX选择。 关于字段设置的定义，见内存图和寄存器定义。 |
| 7<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 6-0<br>卫星 | 触发器MUX输入0来源选择<br>这个读/写位字段用于配置外设触发输入0的MUX选择。 关于字段设置的定义，见内存图和寄存器定义。 |

# 17.4.1.7 TRGMUX CMP0寄存器（CMP0）。

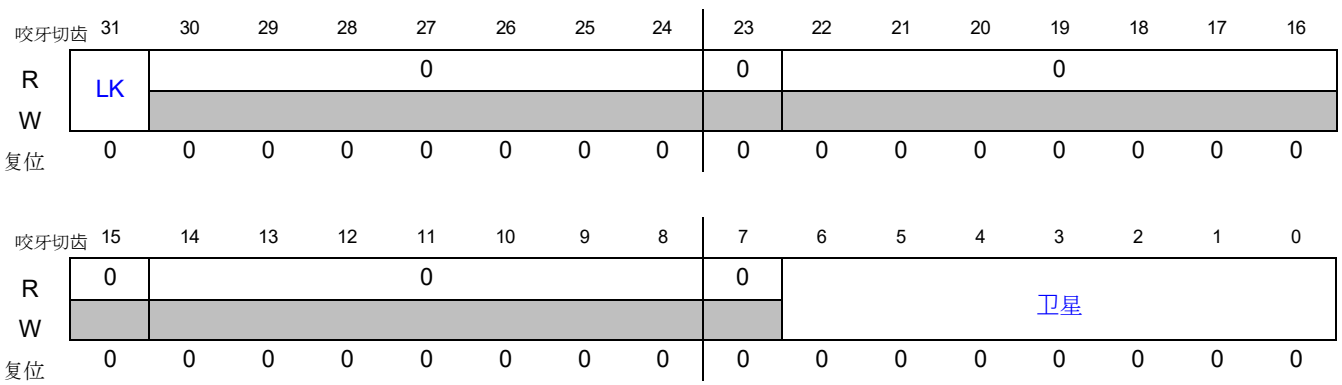
## 17.4.1.7.1 偏移

| 注册 | 偏移 |
|---|---|
| CMP0 | 1Ch |

## 17.4.1.7.2 职能

TRGMUX 寄存器

## 17.4.1.7.3 图示




## 17.4.1.7.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>LK | TRGMUX寄存器锁定。<br>这个位显示寄存器是否可以被写入。LK位在任何系统复位后只能被写入一次。一旦LK被设置，该TRGMUX寄存器中的SELx位不能被改变，直到下一次系统复位清除LK。<br>0b - 寄存器可以被写入。<br>1b - 在下一次系统复位前不能写入寄存器。 |
| 30-24<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 23<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 22-16<br>- | 这个只读的位域是保留的，其值始终为0。 |

| | |
|---|---|
| 15<br>- | 这个只读的位域是保留的，其值始终为0。 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| 14-8<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 7<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 6-0<br>卫星 | 触发器MUX输入0来源选择<br>这个读/写位字段用于配置外设触发输入0的MUX选择。 关于字段设置的定义，见内存图和寄存器定义。 |

## 17.4.1.8 TRGMUX FTM0寄存器（FTM0）。

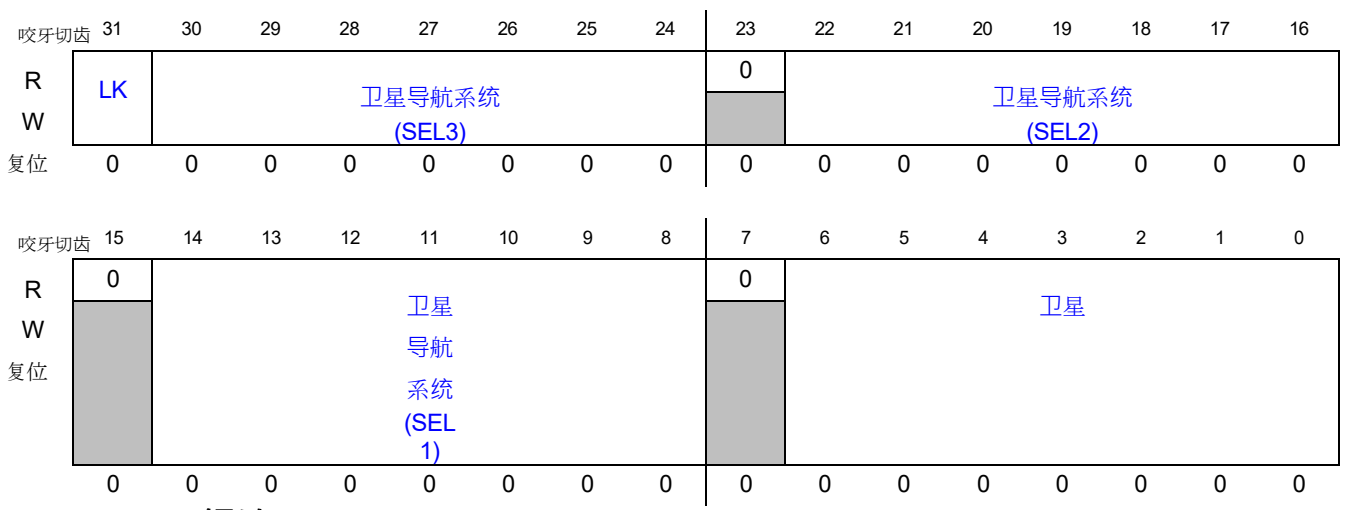
### 17.4.1.8.1 偏移

| 注册 | 偏移 |
|---|---|
| FTM0 | 28h |

### 17.4.1.8.2 职能

TRGMUX 寄存器

### 17.4.1.8.3 图示




### 17.4.1.8.4 领域

| 场地 | 职能 |
|---|---|

| | |
|---|---|
| 31<br>LK | TRGMUX寄存器锁定。 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| | 这个位显示寄存器是否可以被写入。在任何系统复位后，LK位只能被写入一次。一旦LK被设置，该TRGMUX寄存器中的SELx位不能被改变，直到下一次系统复位清除LK。<br>0b - 寄存器可以被写入。<br>1b - 在下一次系统复位前不能写入寄存器。 |
| 30-24<br>卫星导航系统(SEL3) | 触发器MUX输入3源选择<br>该读/写位域用于配置外设触发输入3的MUX选择。 关于该域的设置定义，请参见内存图和寄存器定义。 |
| 23<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 22-16<br>卫星导航系统(SEL2) | 触发器MUX输入2来源选择<br>这个读/写位字段用于配置外设触发输入2的MUX选择。 关于字段设置的定义，见内存图和寄存器定义。 |
| 15<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 14-8<br>卫星导航系统(SEL1) | 触发器MUX输入1来源选择<br>这个读/写位字段用于配置外设触发输入1的MUX选择。 关于字段设置的定义，见内存图和寄存器定义。 |
| 7<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 6-0<br>卫星 | 触发器MUX输入0来源选择<br>这个读/写位字段用于配置外设触发输入0的MUX选择。 关于字段设置的定义，见内存图和寄存器定义。 |

## 17.4.1.9 TRGMUX FTM1寄存器（FTM1）。

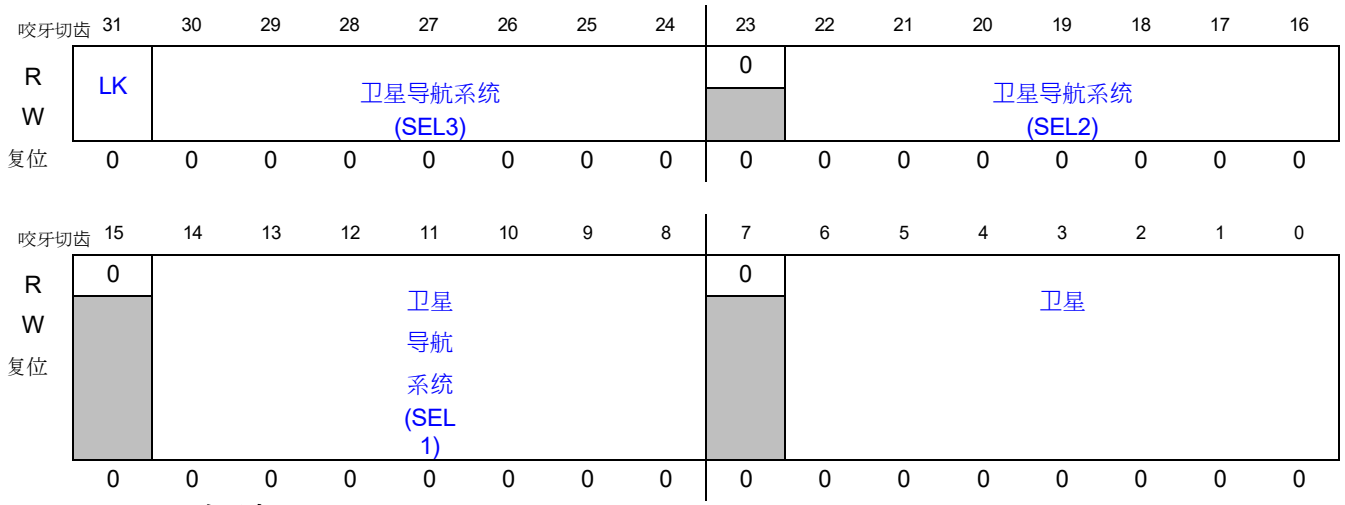
### 17.4.1.9.1 偏移

| 注册 | 偏移 |
|---|---|
| FTM1 | 2Ch |

### 17.4.1.9.2 职能

TRGMUX 寄存器

### 17.4.1.9.3 图示




### 17.4.1.9.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>LK | TRGMUX寄存器锁定。<br>这个位显示寄存器是否可以被写入。在任何系统复位后，LK位只能被写入一次。一旦LK被设置，该TRGMUX寄存器中的SELx位不能被改变，直到下一次系统复位清除LK。<br>0b - 寄存器可以被写入。<br>1b - 在下一次系统复位前不能写入寄存器。 |
| 30-24<br>卫星导航系统 (SEL3) | 触发器MUX输入3源选择<br>该读/写位域用于配置外设触发输入3的MUX选择。 关于该域的设置定义，请参见内存图和寄存器定义。 |
| 23<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 22-16<br>卫星导航系统 (SEL2) | 触发器MUX输入2来源选择<br>这个读/写位字段用于配置外设触发输入2的MUX选择。 关于字段设置的定义，见内存图和寄存器定义。 |
| 15<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 14-8<br>卫星导航系统 (SEL1) | 触发器MUX输入1来源选择<br>这个读/写位字段用于配置外设触发输入1的MUX选择。 关于字段设置的定义，见内存图和寄存器定义。 |
| 7<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 6-0<br>卫星 | 触发器MUX输入0来源选择<br>这个读/写位字段用于配置外设触发输入0的MUX选择。 关于字段设置的定义，见内存图和寄存器定义。 |

# 17.4.1.10 TRGMUX FTM2寄存器（FTM2）。

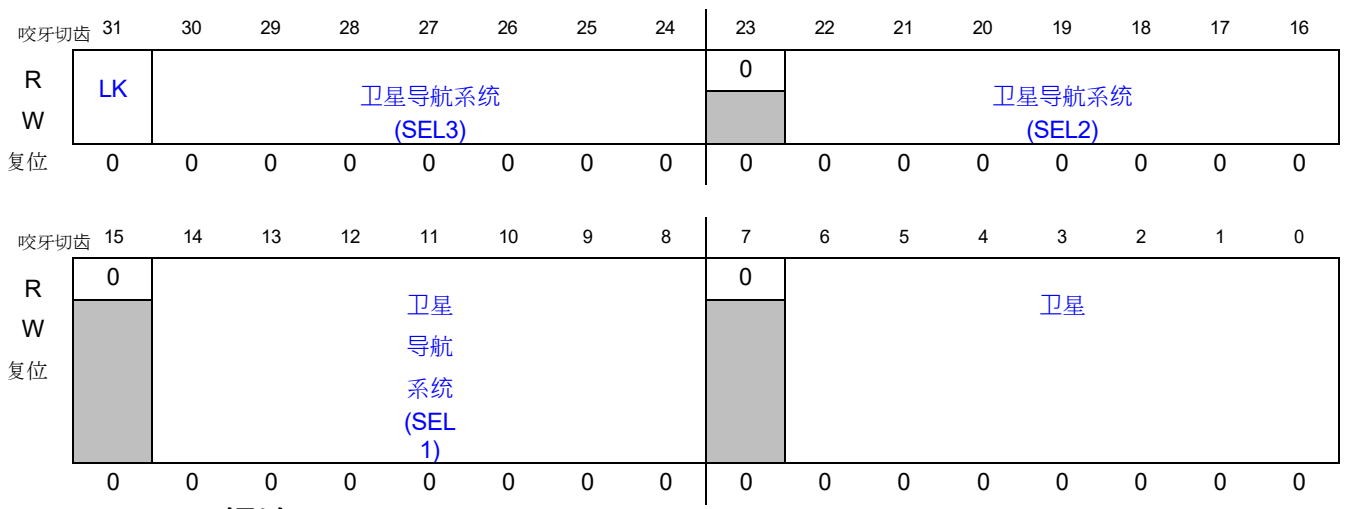
## 17.4.1.10.1 偏移

| 注册 | 偏移 |
|---|---|
| FTM2 | 30h |

## 17.4.1.10.2 职能

TRGMUX 寄存器

## 17.4.1.10.3 图示




## 17.4.1.10.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>LK | TRGMUX寄存器锁定。<br>这个位显示寄存器是否可以被写入。LK位在任何系统复位后只能被写入一次。一旦LK被设置，该TRGMUX寄存器中的SELx位不能被改变，直到下一次系统复位清除LK。<br>0b - 寄存器可以被写入。<br>1b - 在下一次系统复位前不能写入寄存器。 |
| 30-24<br>卫星导航系统 (SEL3) | 触发器MUX输入3源选择<br>该读/写位域用于配置外设触发输入3的MUX选择。 关于该域的设置定义，请参见内存图和寄存器定义。 |
| 23<br>- | 这个只读的位域是保留的，其值始终为0。 |

| | |
|---|---|
| 22-16<br>卫星导航系统(SEL2) | 触发器MUX输入2来源选择<br>这个读/写位字段用于配置外设触发输入2的MUX选择。 关于字段设置的定义，见内存图和寄存器定义。 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| 15<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 14-8<br>卫星导航系统(SEL1) | 触发器MUX输入1来源选择<br>这个读/写位字段用于配置外设触发输入1的MUX选择。 关于字段设置的定义，见内存图和寄存器定义。 |
| 7<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 6-0<br>卫星 | 触发器MUX输入0来源选择<br>这个读/写位字段用于配置外设触发输入0的MUX选择。 关于字段设置的定义，见内存图和寄存器定义。 |

## 17.4.1.11 TRGMUX FTM3寄存器（FTM3）。

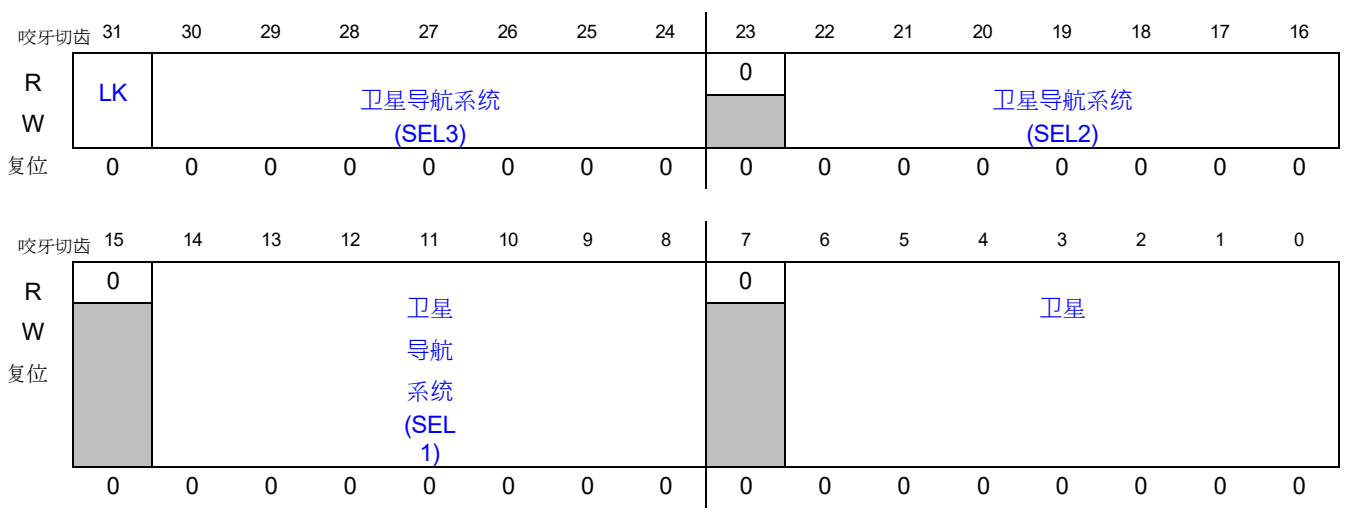
### 17.4.1.11.1 偏移

| 注册 | 偏移 |
|---|---|
| FTM3 | 34h |

### 17.4.1.11.2 职能

TRGMUX 寄存器

### 17.4.1.11.3 图示



#### 17.4.1.11.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>LK | TRGMUX寄存器锁定。<br>这个位显示寄存器是否可以被写入。LK位在任何系统复位后只能被写入一次。一旦LK被设置，该TRGMUX寄存器中的SELx位不能被改变，直到下一次系统复位清除LK。<br>0b - 寄存器可以被写入。<br>1b - 在下一次系统复位前不能写入寄存器。 |
| 30-24<br>卫星导航系统(SEL3) | 触发器MUX输入3源选择<br>该读/写位域用于配置外设触发输入3的MUX选择。 关于该域的设置定义，请参见内存图和寄存器定义。 |
| 23<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 22-16<br>卫星导航系统(SEL2) | 触发器MUX输入2来源选择<br>这个读/写位字段用于配置外设触发输入2的MUX选择。 关于字段设置的定义，见内存图和寄存器定义。 |
| 15<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 14-8<br>卫星导航系统(SEL1) | 触发器MUX输入1来源选择<br>这个读/写位字段用于配置外设触发输入1的MUX选择。 关于字段设置的定义，见内存图和寄存器定义。 |
| 7<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 6-0<br>卫星 | 触发器MUX输入0来源选择<br>这个读/写位字段用于配置外设触发输入0的MUX选择。 关于字段设置的定义，见内存图和寄存器定义。 |

## 17.4.1.12 TRGMUX PDB0寄存器（PDB0）。

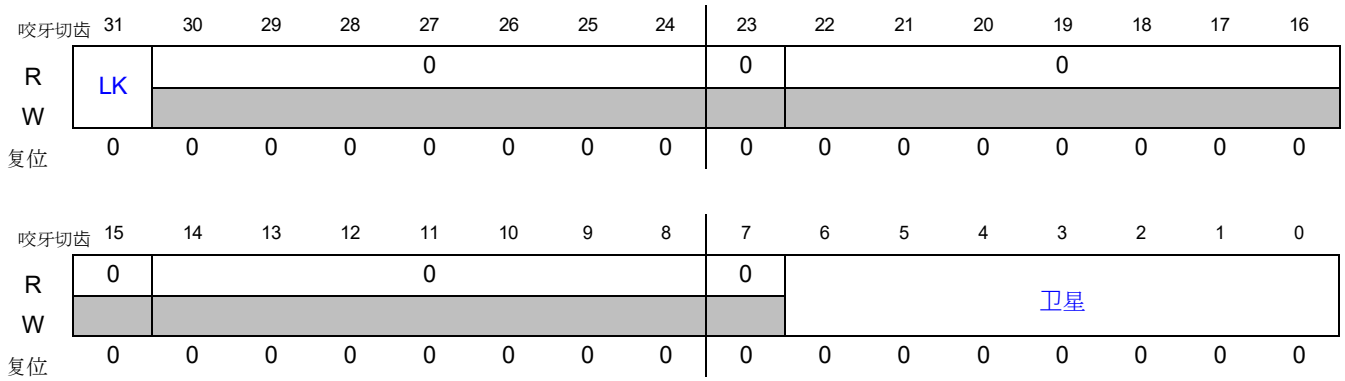
### 17.4.1.12.1 偏移

| 注册 | 偏移 |
|---|---|
| PDB0 | 38h |

### 17.4.1.12.2 职能

TRGMUX 寄存器

### 17.4.1.12.3 图示




### 17.4.1.12.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>LK | TRGMUX寄存器锁定。<br>这个位显示寄存器是否可以被写入。在任何系统复位后，LK位只能被写入一次。一旦LK被设置，该TRGMUX寄存器中的SELx位不能被改变，直到下一次系统复位清除LK。<br>0b - 寄存器可以被写入。<br>1b - 在下一次系统复位前不能写入寄存器。 |
| 30-24<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 23<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 22-16<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 15<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 14-8<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 7<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 6-0<br>卫星 | 触发器MUX输入0来源选择<br>这个读/写位字段用于配置外设触发输入0的MUX选择。 关于字段设置的定义，见内存图和寄存器定义。 |

## 17.4.1.13 TRGMUX PDB1寄存器（PDB1）。

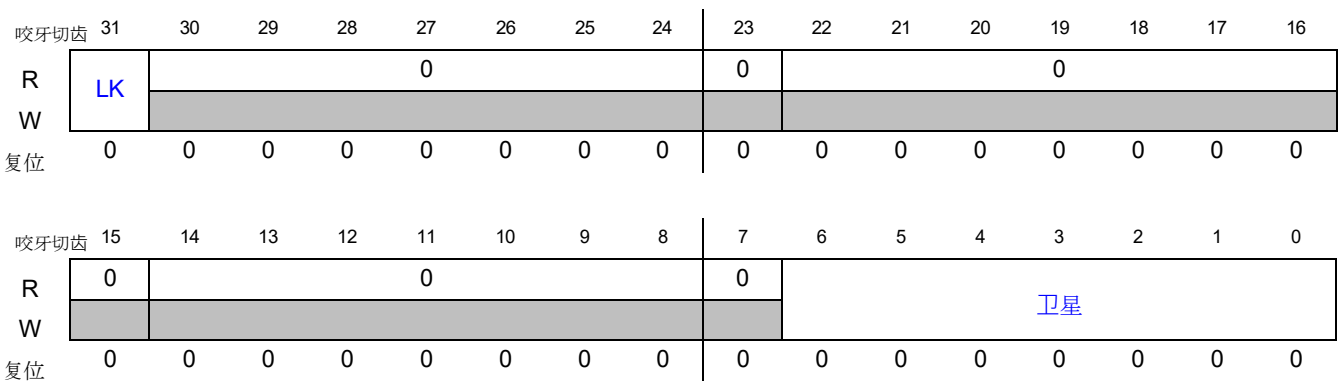
## 17.4.1.13.1 偏移

| 注册 | 偏移 |
|---|---|
| PDB1 | 3Ch |

## 17.4.1.13.2 职能

TRGMUX 寄存器

## 17.4.1.13.3 图示




## 17.4.1.13.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>LK | TRGMUX寄存器锁定。<br>这个位显示寄存器是否可以被写入。LK位在任何系统复位后只能被写入一次。一旦LK被设置，该TRGMUX寄存器中的SELx位不能被改变，直到下一次系统复位清除LK。<br>0b - 寄存器可以被写入。<br>1b - 在下一次系统复位前不能写入寄存器。 |
| 30-24<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 23<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 22-16<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 15<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 14-8<br>- | 这个只读的位域是保留的，其值始终为0。 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| 7<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 6-0<br>卫星 | 触发器MUX输入0来源选择<br>这个读/写位字段用于配置外设触发输入0的MUX选择。 关于字段设置的定义，见内存图和寄存器定义。 |

## 17.4.1.14 TRGMUX FLEXIO寄存器（FLEXIO）。

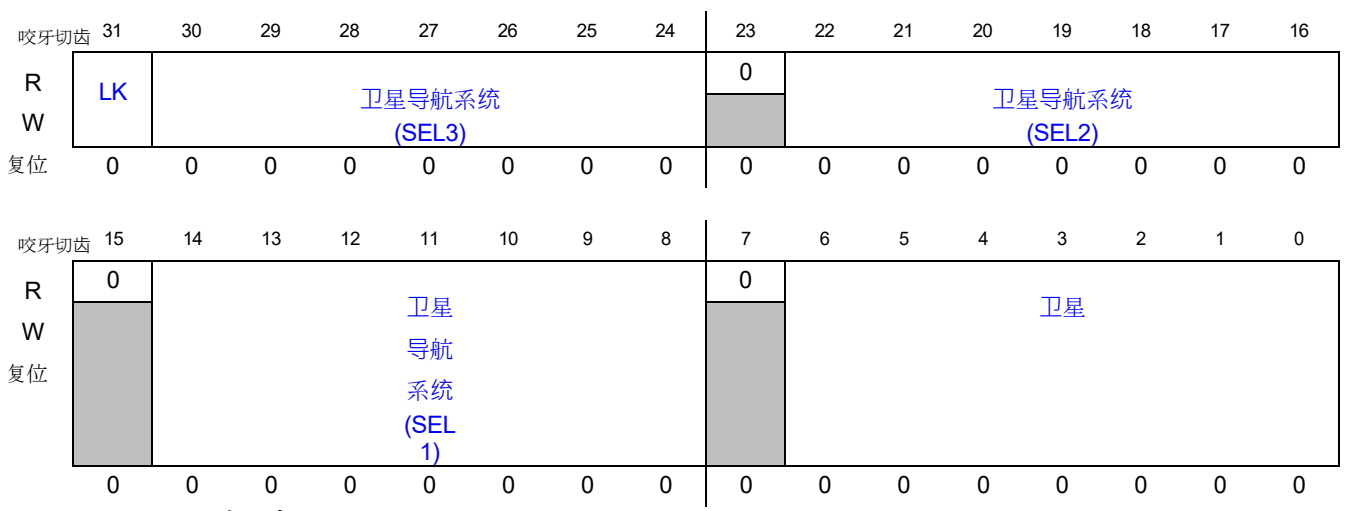
### 17.4.1.14.1 偏移

| 注册 | 偏移 |
| --- | --- |
| 弹性 | 44h |

### 17.4.1.14.2 职能

TRGMUX 寄存器

### 17.4.1.14.3 图示




### 17.4.1.14.4 领域

| 场地 | 职能 |
| --- | --- |

| | |
|---|---|
| 31<br>LK | TRGMUX寄存器锁定。<br>这个位显示寄存器是否可以被写入。LK位在任何系统复位后只能被写入一次。一旦LK被设置，该TRGMUX寄存器中的SELx位不能被改变，直到下一次系统复位清除LK。<br>0b - 寄存器可以被写入。 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | 1b - 在下一次系统复位前不能写入寄存器。 |
| 30-24<br>卫星导航系统(SEL3) | 触发器MUX输入3源选择<br>该读/写位域用于配置外设触发输入3的MUX选择。 关于该域的设置定义，请参见内存图和寄存器定义。 |
| 23<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 22-16<br>卫星导航系统(SEL2) | 触发器MUX输入2来源选择<br>这个读/写位字段用于配置外设触发输入2的MUX选择。 关于字段设置的定义，见内存图和寄存器定义。 |
| 15<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 14-8<br>卫星导航系统(SEL1) | 触发器MUX输入1来源选择<br>这个读/写位字段用于配置外设触发输入1的MUX选择。 关于字段设置的定义，见内存图和寄存器定义。 |
| 7<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 6-0<br>卫星 | 触发器MUX输入0来源选择<br>这个读/写位字段用于配置外设触发输入0的MUX选择。 关于字段设置的定义，见内存图和寄存器定义。 |

## 17.4.1.15 TRGMUX LPIT0寄存器（LPIT0）。

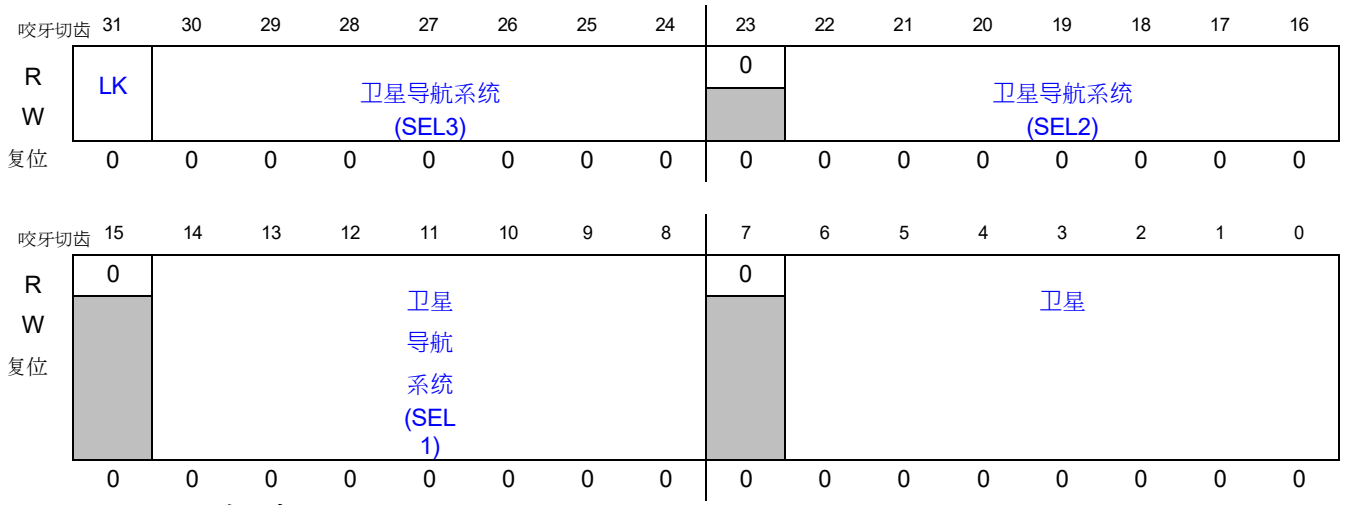
### 17.4.1.15.1 偏移

| 注册 | 偏移 |
| --- | --- |
| LPIT0 | 48h |

### 17.4.1.15.2 职能

TRGMUX 寄存器

### 17.4.1.15.3 图示




### 17.4.1.15.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>LK | TRGMUX寄存器锁定。<br>这个位显示寄存器是否可以被写入。在任何系统复位后，LK位只能被写入一次。一旦LK被设置，该TRGMUX寄存器中的SELx位不能被改变，直到下一次系统复位清除LK。<br>0b - 寄存器可以被写入。<br>1b - 在下一次系统复位前不能写入寄存器。 |
| 30-24<br>卫星导航系统 (SEL3) | 触发器MUX输入3源选择<br>该读/写位域用于配置外设触发输入3的MUX选择。 关于该域的设置定义，请参见内存图和寄存器定义。 |
| 23<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 22-16<br>卫星导航系统 (SEL2) | 触发器MUX输入2来源选择<br>这个读/写位字段用于配置外设触发输入2的MUX选择。 关于字段设置的定义，见内存图和寄存器定义。 |
| 15<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 14-8<br>卫星导航系统 (SEL1) | 触发器MUX输入1来源选择<br>这个读/写位字段用于配置外设触发输入1的MUX选择。 关于字段设置的定义，见内存图和寄存器定义。 |
| 7<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 6-0<br>卫星 | 触发器MUX输入0来源选择<br>这个读/写位字段用于配置外设触发输入0的MUX选择。 关于字段设置的定义，见内存图和寄存器定义。 |

# 17.4.1.16 TRGMUX LPUART0寄存器（LPUART0）

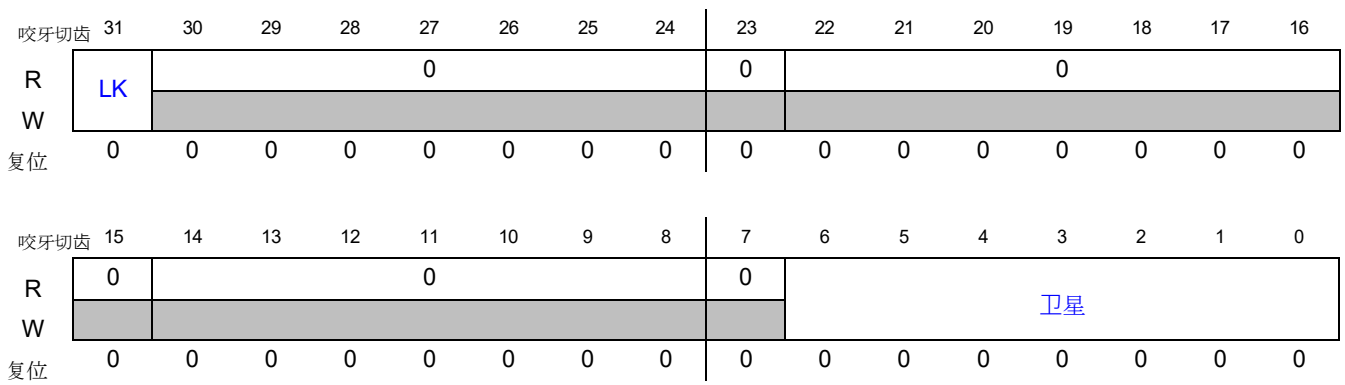
## 17.4.1.16.1 偏移

| 注册 | 偏移 |
|---|---|
| 呼叫中心 | 4Ch |

## 17.4.1.16.2 职能

TRGMUX 寄存器

## 17.4.1.16.3 图示




## 17.4.1.16.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>LK | TRGMUX寄存器锁定。<br>这个位显示寄存器是否可以被写入。LK位在任何系统复位后只能被写入一次。一旦LK被设置，该TRGMUX寄存器中的SELx位不能被改变，直到下一次系统复位清除LK。<br>0b - 寄存器可以被写入。<br>1b - 在下一次系统复位前不能写入寄存器。 |
| 30-24<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 23<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 22-16<br>- | 这个只读的位域是保留的，其值始终为0。 |

| | |
|---|---|
| 15<br>-<br> | 这个只读的位域是保留的，其值始终为0。 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| 14-8<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 7<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 6-0<br>卫星 | 触发器MUX输入0来源选择<br>这个读/写位字段用于配置外设触发输入0的MUX选择。 关于字段设置的定义，见内存图和寄存器定义。 |

## 17.4.1.17 TRGMUX LPUART1寄存器（LPUART1）

### 17.4.1.17.1 偏移

| 注册 | 偏移 |
|---|---|
| 卫星定位系统(LPUART1) | 50h |

### 17.4.1.17.2 职能

TRGMUX 寄存器

### 17.4.1.17.3 图示

| 咬牙切齿 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | LK | 0 | | | | | | | 0 | 0 | | | | | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 咬牙切齿 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | | | | | | | 0 | 卫星 | | | | | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

### 17.4.1.17.4 领域

| 场地 | 职能 |
|---|---|

| | |
|---|---|
| 31<br>LK | TRGMUX寄存器锁定。 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | 这个位显示寄存器是否可以被写入。LK位在任何系统复位后只能被写入一次。一旦LK被设置，该TRGMUX寄存器中的SELx位不能被改变，直到下一次系统复位清除LK。<br>0b - 寄存器可以被写入。<br>1b - 在下一次系统复位前不能写入寄存器。 |
| 30-24<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 23<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 22-16<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 15<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 14-8<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 7<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 6-0<br>卫星 | 触发器MUX输入0来源选择<br>这个读/写位字段用于配置外设触发输入0的MUX选择。 关于字段设置的定义，见内存图和寄存器定义。 |

## 17.4.1.18 TRGMUX LPI2C0寄存器（LPI2C0）。

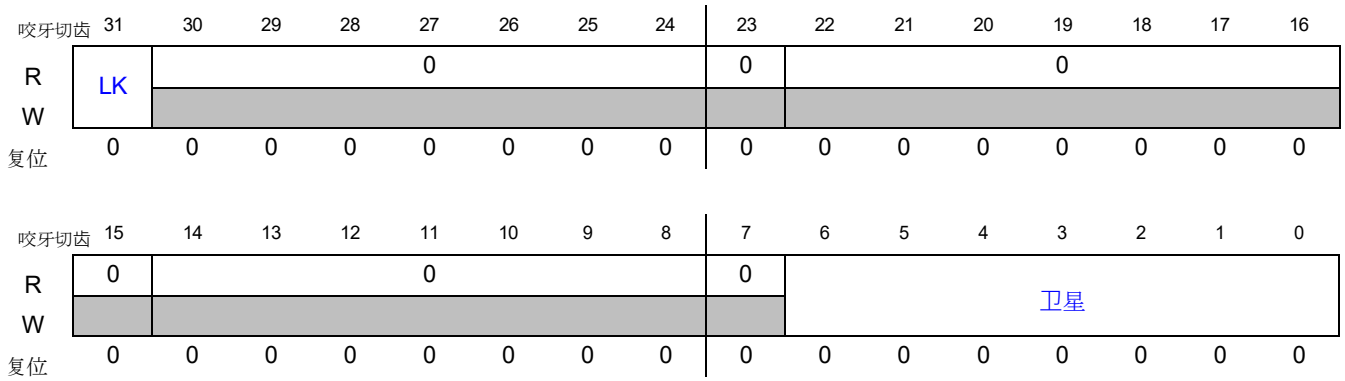
### 17.4.1.18.1 偏移

| 注册 | 偏移 |
| --- | --- |
| LPI2C0 | 54h |

### 17.4.1.18.2 职能

TRGMUX 寄存器

### 17.4.1.18.3 图示




### 17.4.1.18.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>LK | TRGMUX寄存器锁定。<br>这个位显示寄存器是否可以被写入。LK位在任何系统复位后只能被写入一次。一旦LK被设置，该TRGMUX寄存器中的SELx位不能被改变，直到下一次系统复位清除LK。<br>0b - 寄存器可以被写入。<br>1b - 在下一次系统复位前不能写入寄存器。 |
| 30-24<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 23<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 22-16<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 15<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 14-8<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 7<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 6-0<br>卫星 | 触发器MUX输入0来源选择<br>这个读/写位字段用于配置外设触发输入0的MUX选择。 关于字段设置的定义，见内存图和寄存器定义。 |

## 17.4.1.19 TRGMUX LPSPI0寄存器（LPSPI0）。

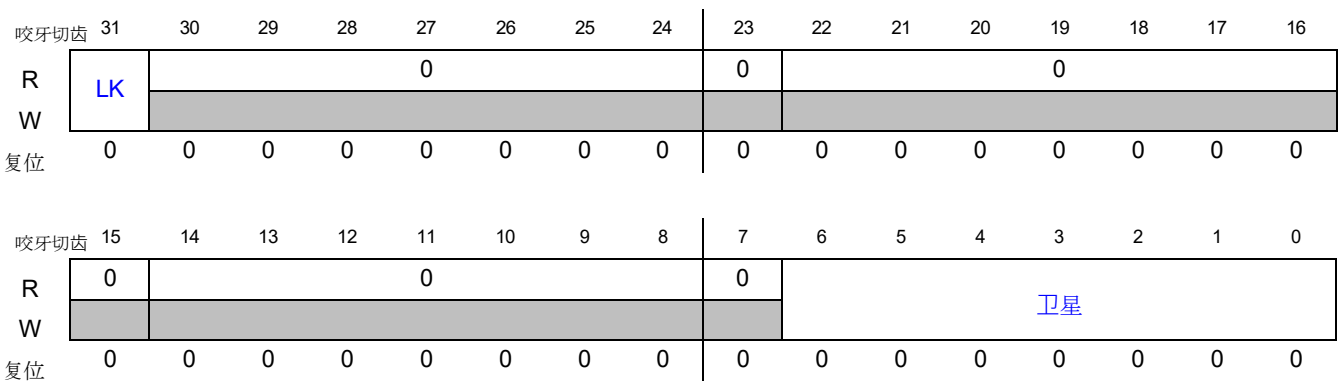
### 17.4.1.19.1 偏移

| 注册 | 偏移 |
|---|---|
| LPSPI0 | 5Ch |

### 17.4.1.19.2 职能

TRGMUX 寄存器

### 17.4.1.19.3 图示




### 17.4.1.19.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>LK | TRGMUX寄存器锁定。<br>这个位显示寄存器是否可以被写入。LK位在任何系统复位后只能被写入一次。一旦LK被设置，该TRGMUX寄存器中的SELx位不能被改变，直到下一次系统复位清除LK。<br>0b - 寄存器可以被写入。<br>1b - 在下一次系统复位前不能写入寄存器。 |
| 30-24<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 23<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 22-16<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 15<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 14-8<br>- | 这个只读的位域是保留的，其值始终为0。 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| 7<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 6-0<br>卫星 | 触发器MUX输入0来源选择<br>这个读/写位字段用于配置外设触发输入0的MUX选择。 关于字段设置的定义，见内存图和寄存器定义。 |

## 17.4.1.20 TRGMUX LPSPI1寄存器（LPSPI1）。

### 17.4.1.20.1 偏移

| 注册 | 偏移 |
|---|---|
| 锂离子交换器1 | 60h |

### 17.4.1.20.2 职能

TRGMUX 寄存器

### 17.4.1.20.3 图示

| 咬牙切齿 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | LK | 0 | | | | | | | 0 | 0 | | | | | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 咬牙切齿 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | | | | | | | 0 | 卫星 | | | | | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

### 17.4.1.20.4 领域

| 场地 | 职能 |
|---|---|

| | |
|---|---|
| 31<br>LK | TRGMUX寄存器锁定。<br>这个位显示寄存器是否可以被写入。LK位在任何系统复位后只能被写入一次。一旦LK被设置，该TRGMUX寄存器中的SELx位不能被改变，直到下一次系统复位清除LK。<br>0b - 寄存器可以被写入。 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | 1b - 在下一次系统复位前不能写入寄存器。 |
| 30-24<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 23<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 22-16<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 15<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 14-8<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 7<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 6-0<br>卫星 | 触发器MUX输入0来源选择<br>这个读/写位字段用于配置外设触发输入0的MUX选择。 关于字段设置的定义，见内存图和寄存器定义。 |

## 17.4.1.21 TRGMUX LPTMR0寄存器（LPTMR0）。

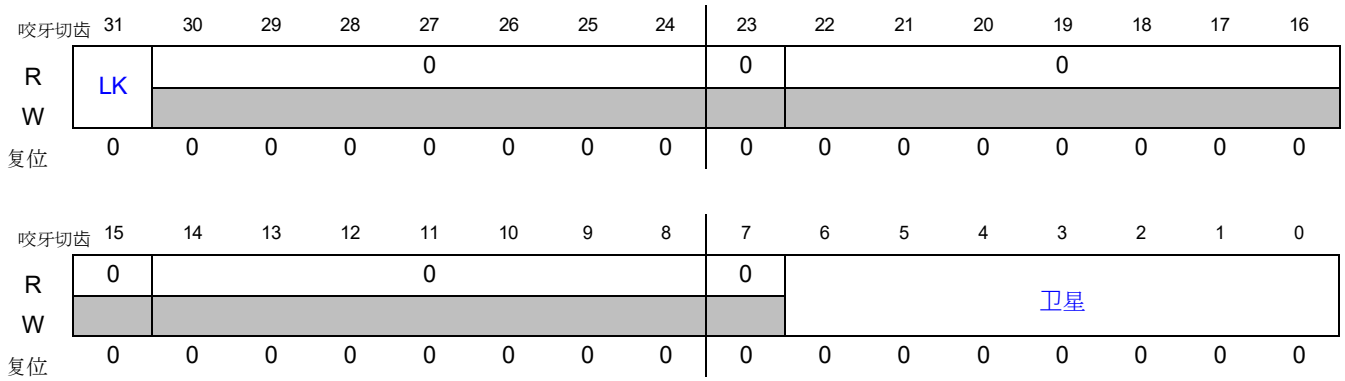
### 17.4.1.21.1 偏移

| 注册 | 偏移 |
| --- | --- |
| 呼叫中心 | 64h |

### 17.4.1.21.2 职能

TRGMUX 寄存器

### 17.4.1.21.3 图示




### 17.4.1.21.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>LK | TRGMUX寄存器锁定。<br>这个位显示寄存器是否可以被写入。LK位在任何系统复位后只能被写入一次。一旦LK被设置，该TRGMUX寄存器中的SELx位不能被改变，直到下一次系统复位清除LK。<br>0b - 寄存器可以被写入。<br>1b - 在下一次系统复位前不能写入寄存器。 |
| 30-24<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 23<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 22-16<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 15<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 14-8<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 7<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 6-0<br>卫星 | 触发器MUX输入0来源选择<br>这个读/写位字段用于配置外设触发输入0的MUX选择。 关于字段设置的定义，见内存图和寄存器定义。 |

## 17.4.1.22 TRGMUX LPI2C1寄存器（LPI2C1）。

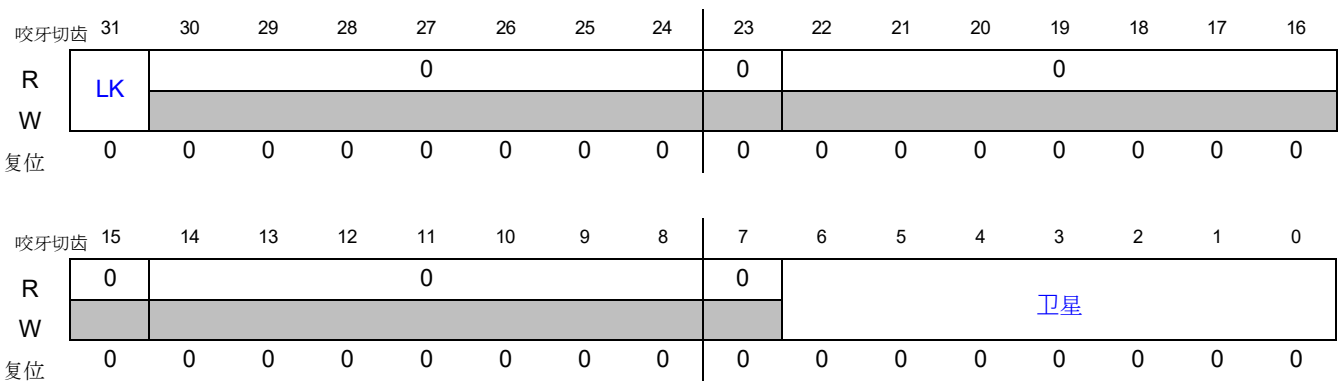
## 17.4.1.22.1 偏移

| 注册 | 偏移 |
|---|---|
| LPI2C1 | 6Ch |

## 17.4.1.22.2 职能

TRGMUX 寄存器

## 17.4.1.22.3 图示




## 17.4.1.22.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>LK | TRGMUX寄存器锁定。<br>这个位显示寄存器是否可以被写入。LK位在任何系统复位后只能被写入一次。一旦LK被设置，该TRGMUX寄存器中的SELx位不能被改变，直到下一次系统复位清除LK。<br>0b - 寄存器可以被写入。<br>1b - 在下一次系统复位前不能写入寄存器。 |
| 30-24<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 23<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 22-16<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 15<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 14-8<br>- | 这个只读的位域是保留的，其值始终为0。 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| 7<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 6-0<br>卫星 | 触发器MUX输入0来源选择<br>这个读/写位字段用于配置外设触发输入0的MUX选择。 关于字段设置的定义，见内存图和寄存器定义。 |

## 17.4.1.23 TRGMUX FTM4寄存器（FTM4）。

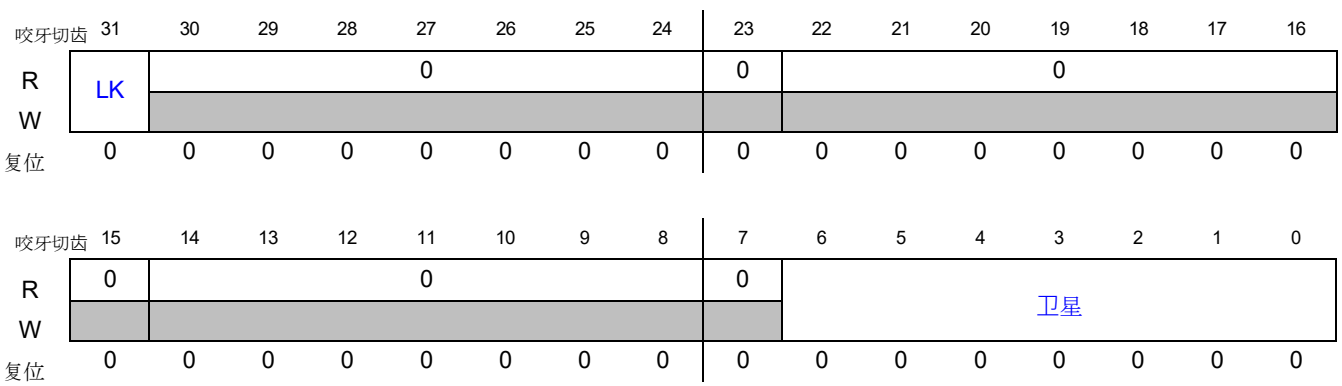
### 17.4.1.23.1 偏移

| 注册 | 偏移 |
|---|---|
| FTM4 | 70h |

### 17.4.1.23.2 职能

TRGMUX 寄存器

### 17.4.1.23.3 图示




### 17.4.1.23.4 领域

| 场地 | 职能 |
|---|---|

| | |
|---|---|
| 31<br>LK | TRGMUX寄存器锁定。<br>这个位显示寄存器是否可以被写入。在任何系统复位后，LK位只能被写入一次。一旦LK被设置，该TRGMUX寄存器中的SELx位不能被改变，直到下一次系统复位清除LK。<br>0b - 寄存器可以被写入。 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | 1b - 在下一次系统复位前不能写入寄存器。 |
| 30-24<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 23<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 22-16<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 15<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 14-8<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 7<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 6-0<br>卫星 | 触发器MUX输入0来源选择<br>这个读/写位字段用于配置外设触发输入0的MUX选择。 关于字段设置的定义，见内存图和寄存器定义。 |

## 17.4.1.24 TRGMUX FTM5寄存器（FTM5）。

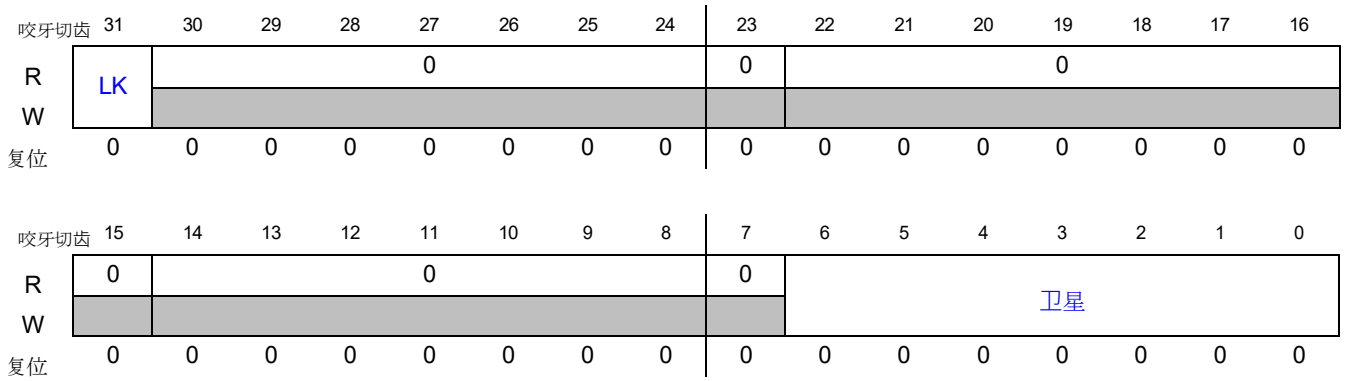
### 17.4.1.24.1 偏移

| 注册 | 偏移 |
| --- | --- |
| FTM5 | 74h |

### 17.4.1.24.2 职能

TRGMUX 寄存器

### 17.4.1.24.3 图示




### 17.4.1.24.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>LK | TRGMUX寄存器锁定。<br>这个位显示寄存器是否可以被写入。LK位在任何系统复位后只能被写入一次。一旦LK被设置，该TRGMUX寄存器中的SELx位不能被改变，直到下一次系统复位清除LK。<br>0b - 寄存器可以被写入。<br>1b - 在下一次系统复位前不能写入寄存器。 |
| 30-24<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 23<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 22-16<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 15<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 14-8<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 7<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 6-0<br>卫星 | 触发器MUX输入0来源选择<br>这个读/写位字段用于配置外设触发输入0的MUX选择。 关于字段设置的定义，见内存图和寄存器定义。 |

## 17.4.1.25 TRGMUX FTM6寄存器（FTM6）。

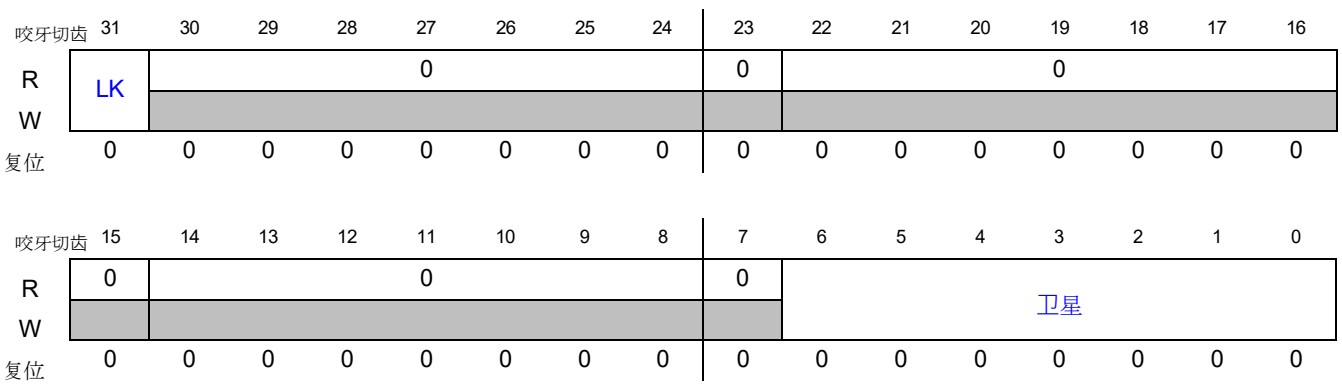
## 17.4.1.25.1 偏移

| 注册 | 偏移 |
|---|---|
| FTM6 | 78h |

## 17.4.1.25.2 职能

TRGMUX 寄存器

## 17.4.1.25.3 图示




## 17.4.1.25.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>LK | TRGMUX寄存器锁定。<br>这个位显示寄存器是否可以被写入。LK位在任何系统复位后只能被写入一次。一旦LK被设置，该TRGMUX寄存器中的SELx位不能被改变，直到下一次系统复位清除LK。<br>0b - 寄存器可以被写入。<br>1b - 在下一次系统复位前不能写入寄存器。 |
| 30-24<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 23<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 22-16<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 15<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 14-8<br>- | 这个只读的位域是保留的，其值始终为0。 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| 7<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 6-0<br>卫星 | 触发器MUX输入0来源选择<br>这个读/写位字段用于配置外设触发输入0的MUX选择。 关于字段设置的定义，见内存图和寄存器定义。 |

## 17.4.1.26 TRGMUX FTM7寄存器（FTM7）。

### 17.4.1.26.1 偏移

| 注册 | 偏移 |
| --- | --- |
| FTM7 | 7Ch |

### 17.4.1.26.2 职能

TRGMUX 寄存器

### 17.4.1.26.3 图示

| 咬牙切齿 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| R | LK | 0 | | | | | | | 0 | 0 | | | | | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 咬牙切齿 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| R | 0 | 0 | | | | | | | 0 | 卫星 | | | | | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

### 17.4.1.26.4 领域

| 场地 | 职能 |
| --- | --- |

| | |
|---|---|
| 31<br>LK | TRGMUX寄存器锁定。<br>这个位显示寄存器是否可以被写入。LK位在任何系统复位后只能被写入一次。一旦LK被设置，该TRGMUX寄存器中的SELx位不能被改变，直到下一次系统复位清除LK。<br>0b - 寄存器可以被写入。 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | 1b - 在下一次系统复位前不能写入寄存器。 |
| 30-24<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 23<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 22-16<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 15<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 14-8<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 7<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 6-0<br>卫星 | 触发器MUX输入0来源选择<br>这个读/写位字段用于配置外设触发输入0的MUX选择。 关于字段设置的定义，见内存图和寄存器定义。 |

# 章 18
# 外部看门狗监控 (EWM)

## 18.1 特定芯片的EWM 信息

S32K14x有一个EWM的实例。S32K11x系列的产品没有EWM。

### 18.1.1 EWM_OUT信号 配置

EWM_out信号与其他I/O共享其焊盘。要知道该焊盘的默认状态，请参见《参考手册》所附的IO信号描述输入复用表。

### 18.1.2 EWM内存地图 访问

只支持8位访问。

### 18.1.3 EWM低功率 模式

该表显示了EWM的低功耗模式和相应的芯片低功耗模式。

不支持等待模式和断电模式。关于可用的电源模式的详情，请参见模块在可用的低功率模式下的操作。

## 18.2 简介

为了安全起见，设计了一个冗余的看门狗系统，即外部看门狗监控器（EWM），用于监控外部电路以及MCU软件流。这为复位MCU的CPU和外设的内部看门狗提供了一个备份机制。

看门狗一般用于监控MCU内嵌入式软件的流程和执行。看门狗由一个计数器组成，如果允许其溢出，则强制对所有片上外设进行内部复位（异步），并可选择对RESET_B引脚进行复位外部器件/电路。如果软件代码工作良好，并为看门狗提供服务以重新启动实际的计数器，看门狗计数器的溢出一定不会发生。

EWM与内部看门狗不同，它不重置MCU的CPU和外围设备。EWM提供一个独立的EWM_OUT_b信号，当它被断言时，会重置或将外部电路放入安全模式。EWM_OUT_b信号在EWM计数器超时时被引爆。提供一个可选的外部输入EWM_in，以允许对EWM_OUT_b信号的断言进行额外控制。

### 18.2.1 特点

EWM模块的特点包括。

- 独立的LPO_CLK时钟源
- 以EWM LPO_CLK时钟周期的数量指定的可编程超时周期。
- 窗口式刷新选项
  - 提供强有力的检查，确保程序流比预期的快。
  - 可编程窗口。
  - 刷新外部窗口导致EWM_OUT_b的断言。
- 健全的刷新机制
  - 在15个外围总线时钟周期内向EWM刷新寄存器写入0xB4和0x2C的值。
- 一个输出端口，EWM_OUT_b，当被断言时，用于复位或将外部电路放入安全模式。
- 一个输入端口，EWM_in，允许外部电路控制EWM_OUT_b信号的宣称。

### 18.2.2 的操作模式

本节介绍了该模块的操作模式。

#### 18.2.2.1 停止模式

当EWM处于停止模式时，CPU对EWM的刷新不能发生。在进入停止模式时，EWM的计数器会冻结。

有两种可能的方式来退出停止模式。

- 在通过复位从停止模式退出时，EWM仍然被禁用。
- 在通过中断退出停止模式时，EWM被重新启用，计数器继续从进入停止模式前的相同值开始计时。

如果EWM在CPU刷新机制中进入停止模式，请注意以下几点。在通过中断退出停止模式时，刷新机制状态机从以前的状态开始，这意味着，如果第一个刷新指令被正确写入，EWM立即进入停止模式，则下一个指令必须在退出停止模式后的*15*个外围总线时钟内写入。用户必须在执行EWM刷新指令之前屏蔽所有中断。

#### 18.2.2.2 调试模式

进入调试模式对EWM没有影响。

- 如果EWM在进入调试模式之前被启用，它将继续被启用。
- 如果EWM在进入调试模式前被禁用，它将一直被禁用。

### 18.2.3 块 图

该图显示了EWM的方框图。

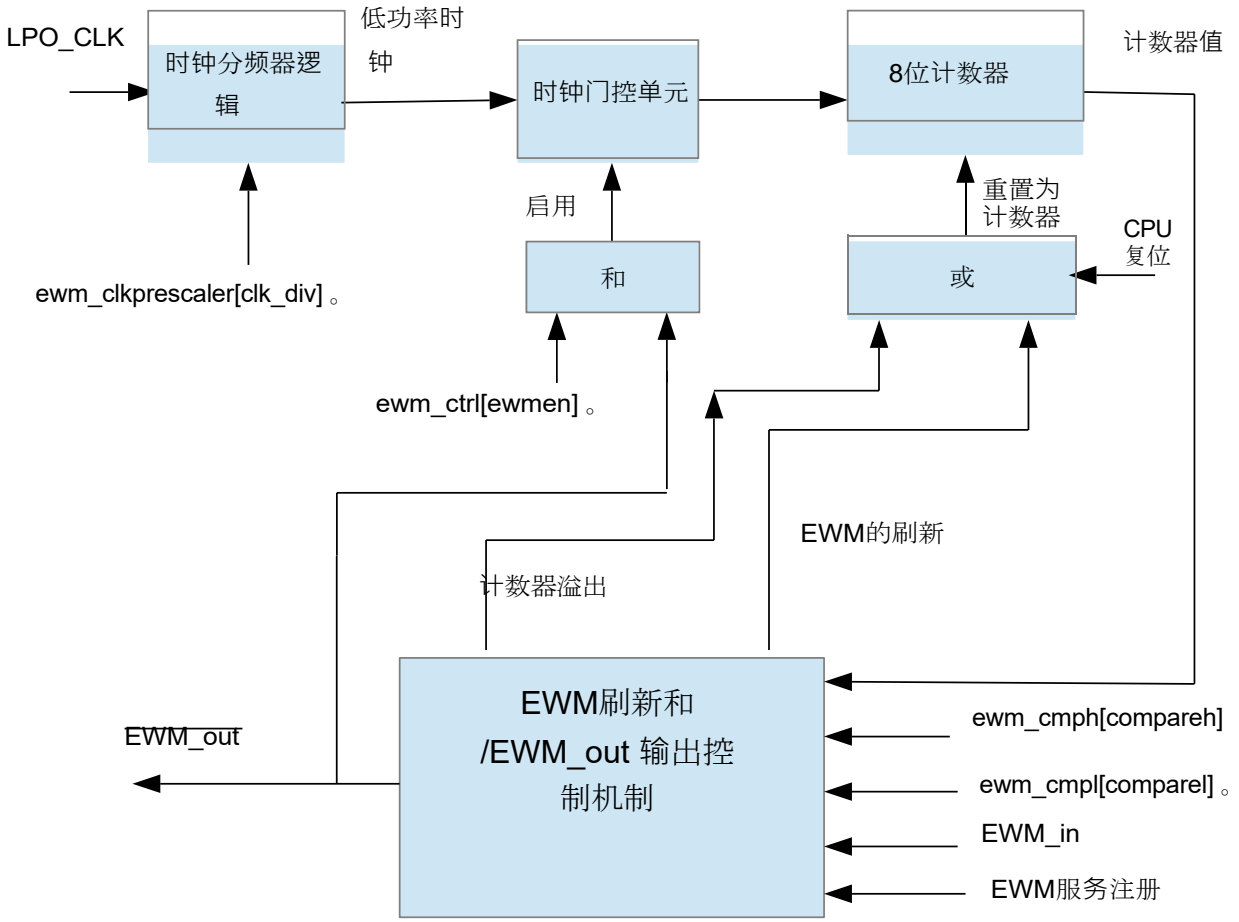



**图18-1.EWM方框图**

## 18.3 EWM信号 说明

EWM有两个外部信号，如下表所示。

**注意事项**

所有的有源低电平信号现在都用后缀"_b"来表示，贯穿本章。

**表18-1.EWM信号描述**

| 信号 | 描述 | 输入/输出 |
|---|---|---|
| EWM_in | EWM输入用于外部安全电路的安全状态。EWM_in的极性可以通过EWM_CTRL[ASSIN]位来编程。默认极性为低电平。 | I |
| EWM_OUT_b | EWM复位输出信号 | O |

## 18.4 内存地图/寄存器 定义

本节包含了模块内存图和寄存器。

**注意事项**

EWM只支持8位寄存器访问。16位和32位访问是不可能的。

### 18.4.1 EWM寄存器 描述

#### 18.4.1.1 EWM内存地图

EWM基础地址。4006_1000h

| 偏移 | 注册 | 宽度 (单位：比特) | 访问 | 重置值 |
|---|---|---|---|---|
| 0h | 控制寄存器（CTRL） | 8 | RW | 00h |
| 1h | 服务寄存器（SERV） | 8 | 钨钢 | 00h |
| 2h | 比较低位寄存器（CMPL） | 8 | RWONCE | 00h |
| 3h | 比较高位寄存器（CMPH） | 8 | RWONCE | FFh |
| 5h | 时钟预分频器寄存器（CLKPRESCALER）。 | 8 | RWONCE | 00h |

#### 18.4.1.2 控制寄存器（CTRL）

##### 18.4.1.2.1 偏移

| 注册 | 偏移 |
|---|---|
| CTRL | 0h |

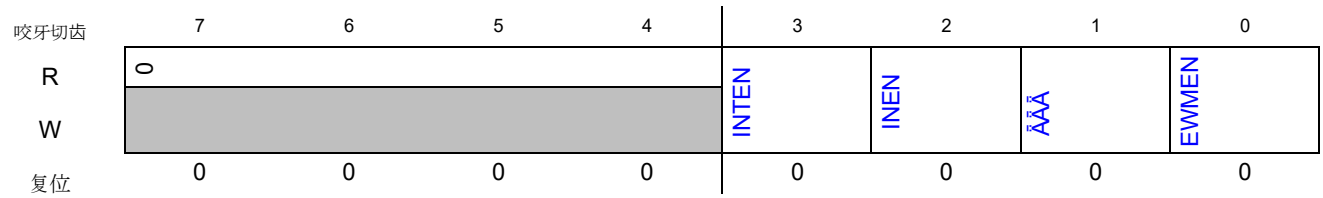
#### 18.4.1.2.2 职能

CTRL寄存器被任何复位所清除。

**注意事项**

INEN、ASSIN和EWMEN位在CPU复位后可以被写入一次。修改这些位超过一次，会产生一个总线传输错误。

### 18.4.1.2.3 图示




### 18.4.1.2.4 领域

| 场地 | 职能 |
|---|---|
| 7-4<br>- | 保留 |
| 3<br>INTEN | 中断启用。<br>当该位被设置且EWM_OUT_b被断言时，将产生一个中断请求。要取消中断请求，用户应该通过写0来清除这个位。 |
| 2<br>INEN | 输入启用。<br>该位被设置时，启用EWM_in端口。 |
| 1<br>欣欣 | EWM_in的断言状态选择。<br>EWM_in信号的默认断言状态是逻辑零。设置ASSIN位可以将EWM_in信号的断言状态反转为逻辑1。 |
| 0<br>EWMEN | EWM启用。<br>该位被设置时，启用EWM模块。这将使EWM计数器复位为零，并使EWM_OUT_b信号失效。当该位未被设置时，保持EWM模块被禁用。由于该位的一次性写入特性，在下一次复位之前，它不能被重新启用。 |

## 18.4.1.3 服务寄存器（SERV）

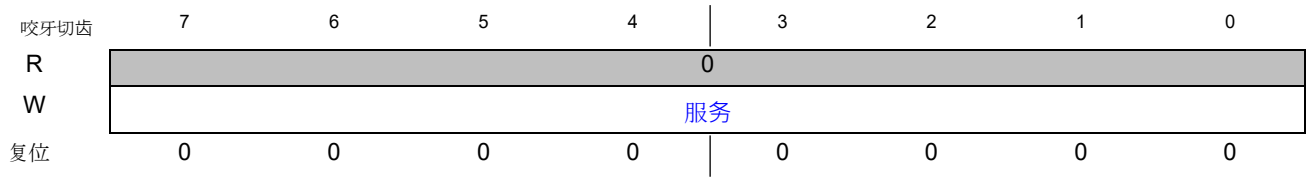
### 18.4.1.3.1 偏移

| 注册 | 偏移 |
|---|---|
| 中介服务 | 1h |

#### 18.4.1.3.2 职能

SERV寄存器提供了从CPU到EWM模块的接口。它是只写的，读这个寄存器返回0。

#### 18.4.1.3.3 图示




#### 18.4.1.3.4 领域

| 场地 | 职能 |
|---|---|
| 7-0<br>服务 | 服务<br>EWM刷新机制要求CPU向SERV寄存器写入两个值：第一个数据字节为0xB4，然后是第二个数据字节为0x2C。如果以下任何一个条件为真，EWM刷新是无效的。<br>• 第一个或第二个数据字节没有正确写入。<br>• 第二个数据字节不会在第一个数据字节的固定数量的外围总线周期内被写入。这个固定的周期数被称为*EWM_refresh_time*。<br>*EWM_refresh_time*需要15个外围总线时钟周期。 |

### 18.4.1.4 比较低位寄存器（CMPL）

#### 18.4.1.4.1 偏移

| 注册 | 偏移 |
|---|---|
| CMPL | 2h |

#### 18.4.1.4.2 职能

在CPU复位后，CMPL寄存器被重置为零。这没有为CPU提供刷新EWM计数器的最短时间。

**注意事项**

在CPU复位后，这个寄存器只能被写入一次。多次写入该寄存器会产生一个总线传输错误。

#### 18.4.1.4.3 图示

| 咬牙切齿 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 比较 | | | | | | | |
| W | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

#### 18.4.1.4.4 领域

| 场地 | 职能 |
|---|---|
| 7-0<br>比较 | 比较<br>为了防止失控的代码改变这个字段，即使需要（默认的）最小刷新时间，软件也应该在CPU复位后写到这个字段。 |

### 18.4.1.5 比较高位寄存器（CMPH）

#### 18.4.1.5.1 偏移

| 注册 | 偏移 |
|---|---|
| CMPH | 3h |

#### 18.4.1.5.2 职能

在CPU复位后，CMPH寄存器被重置为0xFF。这为CPU刷新EWM计数器提供了最多256个时钟的时间。

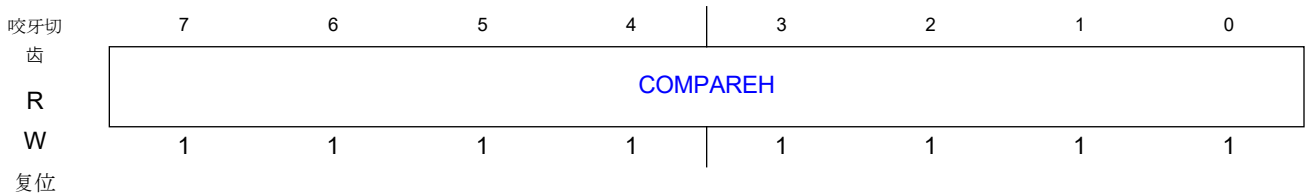
**注意事项**

在CPU复位后，这个寄存器只能被写入一次。多次写入该寄存器会产生一个总线传输错误。

### 注意事项

CMPH的有效值最高为0xFE，因为当CMPH=0xFF时，EWM计数器不会过期。只有当EWM计数器大于CMPH时，才会发生失效。

#### 18.4.1.5.3 图示




#### 18.4.1.5.4 领域

| 场地 | 职能 |
|---|---|
| 7-0<br>COMPAREH | COMPAREH<br>为了防止失控的代码改变这个字段，即使需要（默认的）最大刷新时间，软件也应该在CPU复位后写到这个字段。 |

## 18.4.1.6 时钟预分频器寄存器（CLKPRESCALER）。

#### 18.4.1.6.1 偏移

| 注册 | 偏移 |
|---|---|
| 循环冷却器(CLKPRESCALER) | 5h |

#### 18.4.1.6.2 职能

这个CLKPRESCALER寄存器在CPU复位后被重置为0x00。

### 注意事项

在CPU复位后，这个寄存器只能被写入一次。多次写入该

寄存器会产生一个总线传输错误。

**注意事项**

在启用EWM之前写入所需的预分频器值。

**注意事项**

这个寄存器的实现是特定于芯片的。参见芯片配置的细节
。

#### 18.4.1.6.3 图示




#### 18.4.1.6.4 领域

| 场地 | 职能 |
|---|---|
| 7-0<br>脉冲发生器（CLK_DIV | 脉冲发生器（CLK_DIV<br>选择用于运行EWM计数器的低功耗时钟源，可以按以下方式进行预调。<br>• 预标定的时钟频率 = 低功率时钟源频率 / ( 1 + CLK_DIV ) |

## 18.5 功能 说明

以下章节描述了EWM模块的功能细节。

**注意事项**

当BUS_CLK丢失时，EWM模块不产生EWM_OUT_b信号
，不可能有刷新操作。

### 18.5.1 EWM_OUT_b 信号。

EWM_OUT_b是一个数字输出信号，用来给控制关键安全功能的外部电路（具体应用）把关。例如，EWM_OUT_b可以连接到控制大型电器中交流电机的高压晶体管电路。

当EWM在可编程的刷新窗口内被CPU定期刷新时，EWM_OUT_b信号保持去导

通，表明应用程序代码按预期执行。

EWM_OUT_b信号在以下任何一种情况下都被断言。

- 当计数器值小于CMPL值时，EWM刷新发生。
- EWM计数器值达到CMPH值，并且没有发生EWM刷新。
- 如果EWM_in引脚的功能被启用，并且EWM_in引脚在刷新EWM时被断电。
- 在任何复位后（凭借EWM_OUT_b引脚上的外部下拉机制）。

EWM_OUT_b在任何复位后，凭借EWM_OUT_b信号的外部下拉机制而被确认。然后，为了取消对EWM_OUT_b信号的断言，在CTRL寄存器中设置EWMEN位来启用EWM。

如果EWM_OUT_b信号与一个数字I/O引脚共享它的焊盘，那么在复位时，这个实际的焊盘将被延迟为一个输入信号。只有在EWM被CTRL寄存器中的EWMEN位启用后，焊盘状态才会被EWM_OUT_b信号控制。

**注意事项**

当使用EWM功能和EWM处于复位状态时，EWM_OUT_b焊盘必须处于下拉状态。

## 18.5.2 EWM_OUT_b引脚在低功率 模式下的状态

在等待、停止和断电模式下，EWM_OUT_b引脚保持其进入等待或停止模式前的状态。当CPU从等待或停止恢复进入运行模式时，该引脚恢复到进入等待或停止模式之前的状态。当CPU从断电模式进入运行模式时，该引脚恢复到其复位状态。

## 18.5.3 EWM_in 信号

EWM_in是一个用于外部安全电路的安全状态的数字输入信号，它允许外部电路控制EWM_OUT_b信号的断言。例如，在应用中，一个外部电路监测一个关键的安全功能，如果安全功能出现故障，外部电路就可以主动启动控制门控电路的EWM_OUT_b信号。

如果EWM被禁用，或者CTRL寄存器的INEN位被清除，如任何复位后，EWM_in

信号被忽略。

在启用EWM（设置CTRL[EWMEN]位）和启用EWM_in功能（设置CTRL[INEN]位）时，在CPU开始刷新EWM之前，EWM_in信号必须处于非置位状态。这可以确保EWM_OUT_b保持在去顶点的状态；否则，EWM_OUT_b的输出信号就会被顶点。

**注意事项**

用户必须在启用EWM之前更新CMPH和CMPL寄存器。启用EWM后，计数器复位为零，因此用户应在上电复位后提供合理的时间让外部监测电路稳定下来。
用户还应该确保EWM_in引脚被去掉。

### 18.5.4 EWM 计数器

它是一个8位纹波计数器，由独立于外围总线时钟源的时钟源提供。由于首选的超时是在1ms和100ms之间，实际的时钟源应该在kHz范围内。

在CPU复位后，或在EWM刷新动作完成后，或在计数器溢出时，计数器被重置为零。计数器的值不能被CPU访问。

### 18.5.5 EWM比较 寄存器

比较寄存器CMPL和CMPH在CPU复位后只写一次，在另一次CPU复位之前不能被修改。

EWM比较寄存器被用来创建一个刷新窗口，以刷新EWM模块。

将CMPL和CMPH编程为相同的值是非法的。在这种情况下，一旦计数器达到(CMPL+1)，EWM_OUT_b就被断言。

### 18.5.6 EWM刷新 机制

除了EWM的初始配置外，CPU只能通过EWM服务寄存器访问EWM。CPU必须在CMPL和CMPH寄存器确定的窗口时间范围内以正确的写入唯一数据的方式访问EWM服务寄存器，以实现正确的EWM刷新操作。因此，可能会出现三种情况。

表18-2.EWM刷新机制

| 状况 | 机制 |
| --- | --- |
| 当一个EWM刷新动作完成时。CMPL < Counter < CMPH。 | 软件的行为符合预期，EWM计数器被重置为零。如果在EWM刷新动作期间，EWM_in输入一直处于非置位状态，则EWM_OUT_b输出信号仍然处于非置位状态。 |
| 当Counter < CMPL时，EWM刷新动作完成。 | 软件在窗口时间段之前刷新EWM，计数器被重置为零，EWM_OUT_b输出信号被断言，与输入的EWM_in信号无关。 |
| 计数器值在EWM刷新动作完成前达到CMPH。 | 软件没有刷新EWM。EWM计数器被重置为零，无论输入的EWM_in信号如何，EWM_OUT_b输出信号都被引爆。 |

## 18.5.7 EWM 中断

当EWM_OUT_b被断言时，会产生一个中断请求，指示EWM复位输出信号的断言。当CTRL[INTEN]被设置时，这个中断被启用。清除这个位可以清除中断请求，但不影响EWM_OUT_b。EWM_OUT_b信号只有通过强制系统复位才能被解除。

## 18.5.8 计数器时钟 预分频器

通过编程CLKPRESCALER[CLK_DIV]，EWM计数器的时钟源可以被一个时钟分频器预调。这个分频的时钟用于运行EWM计数器。

**注意事项**

用于运行EWM计数器的分割时钟必须不超过总线时钟频率的一半。

# 章 19
# 错误注入模块 (EIM)

## 19.1 特定芯片的EIM 信息

S32K1xx有一个EIM的实例。

### 19.1.1 EIM通道 分配

每个EIM通道对应于一个特定的RAM阵列目标。EIM通道可以在对该特定RAM阵列的访问中注入错误。下表显示了通道的分配情况。

**表19-1.EIM通道的RAM阵列目标**

| EIM渠道 | RAM阵列目标 |
|---|---|
| 0 | Cortex-M4 TCM较低 |
| 1 | Cortex-M4 TCM上部 |

## 19.2 简介

错误注入模块主要用于诊断目的。它提供了一种对外围存储器进行诊断的方法。请参阅特定芯片的EIM信息，以确定这种方法支持哪些外围存储器。

### 19.2.1 概述

错误注入模块（EIM）提供了对访问外围RAM时在读数据上诱发单比特和多比特反转的支持。在存储器访问上注入故障可以用来锻炼相关系统的SEC-DED ECC功能。

**注意事项**

下图显示了一个具有64位读数据总线和8位校验位总线的EIM实施实例。

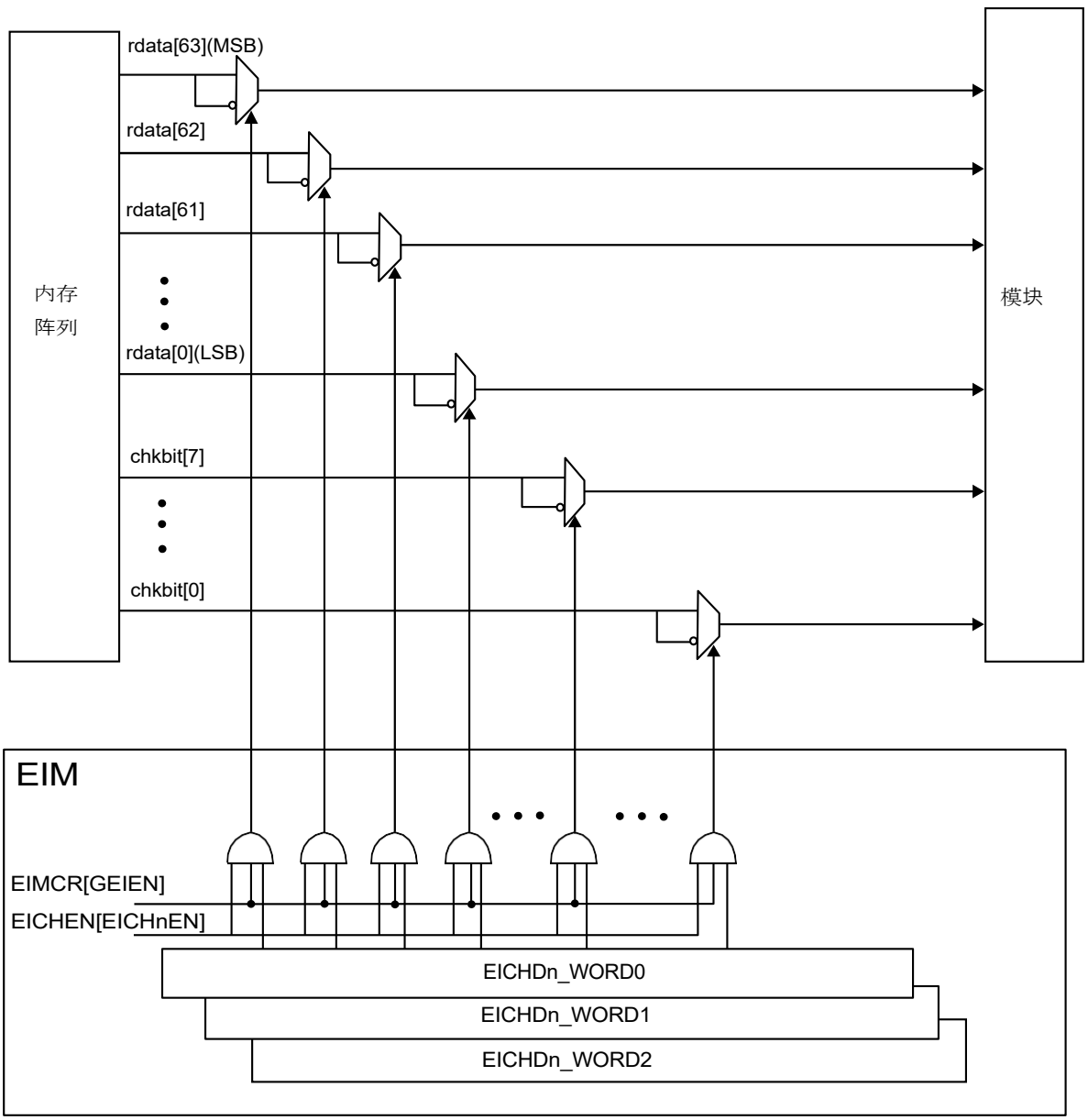



**图19-1.EIM功能框图（64位读数据总线和8位检查位总线）**

## 19.2.2 特点

EIM包括这些功能。

- 支持2个错误注入通道
- 通过两级启用机制防止意外启用和重新配置错误注入功能

## 19.3 EIM寄存器 描述

EIM提供了一个映射到平台上外设插槽的IPS编程模型。

### 编程模型访问

所有系统总线主站都可以访问编程模型。

- 仅在监管者模式下
- 只使用32位（字）访问

以下任何试图引用编程模型的行为都会产生一个IPS错误终止。

- 在用户模式下
- 使用非32位的访问大小
- 对未定义（保留）的地址

当EIM处于操作过程中时，试图更新编程模型会导致非确定性的行为。

### 错误注入通道描述符：功能和结构

每个错误注入通道描述符。

- 指定一个掩码，定义目标RAM的读数据和校验位总线的哪些位在读访问中被反转。
- 由一个128位（16字节）的结构组成，由四个32位的字组成，在EIM编程模型中。
  - 第一个字，Word0（EICHDn_WORD0），定义了检查位掩码。详见错误注入通道描述符*n*，字0。
  - 其余的字，即字1-3（EICHDn_WORD1-3），定义了数据掩码。字2和字3仅在通道数据掩码的总宽度需要时使用。参见错误注入通道描述符。DATA_MASK细节和各个寄存器的描述。

多通道描述符是按顺序组织的。

**错误注入通道描述符。DATA_MASK详情**

对于每个通道。下表显示了DATA_MASK的总宽度和它的位在WORD寄存器中的分布。

| 频道 | DATA_MASK总宽度(比特) | 在DATA_MASK中的特定位 | | |
|---|---|---|---|---|
| | | 词条1 | 词条2 | 语录3 |
| 0 | 32 | 31-0 | - | - |
| 1 | 32 | 31-0 | - | - |

## 19.3.1 EIM内存 地图

EIM基础地址。4001_9000h

| 偏移 | 注册 | 宽度 (单位：比特) | 访问 | 重置值 |
|---|---|---|---|---|
| 0h | 错误注入模块配置寄存器（EIMCR）。 | 32 | RW | 0000_0000h |
| 4h | 错误注入通道启用寄存器（EICHEN）。 | 32 | RW | 0000_0000h |
| 100h | 错误注入通道描述符n，字0（EICHD0_WORD0）。 | 32 | RW | 0000_0000h |
| 104h | 错误注入通道描述符n，字1（EICHD0_WORD1）。 | 32 | RW | 0000_0000h |
| 200h | 错误注入通道描述符n，字0（EICHD1_WORD0）。 | 32 | RW | 0000_0000h |
| 204h | 错误注入通道描述符n，字1（EICHD1_WORD1）。 | 32 | RW | 0000_0000h |

## 19.3.2 错误注入模块配置寄存器 （EIMCR）。

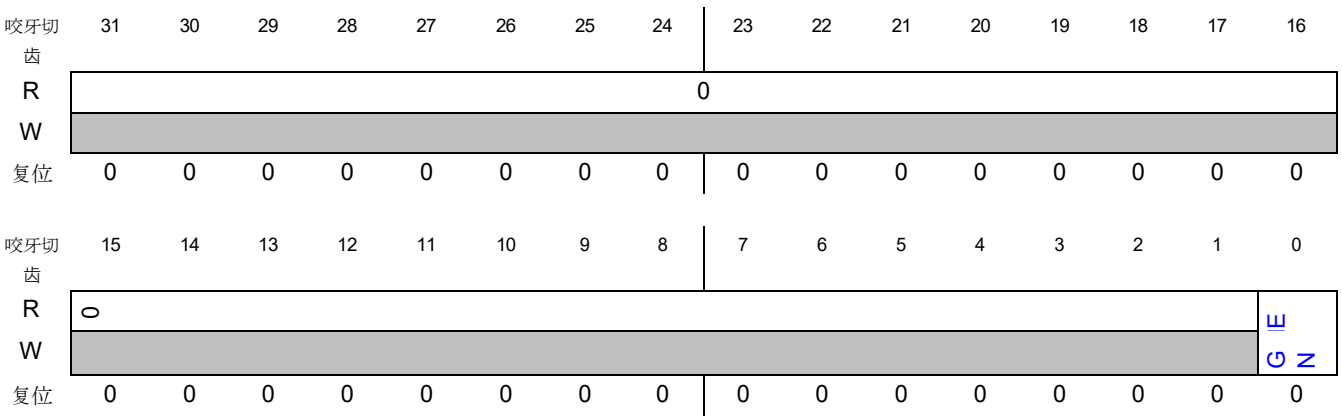
### 19.3.2.1 偏移

| 注册 | 偏移 |
|---|---|
| ÃÃÃ | 0h |

### 19.3.2.2 职能

EIM配置寄存器用于全局启用/禁用错误注入功能。

### 19.3.2.3 图示




### 19.3.2.4 领域

| 场地 | 职能 |
|---|---|
| 31-1<br>- | 保留 |
| 0<br>纪恩 | 启用全局错误注入<br>该位全局地启用或禁用 EIM 的错误注入功能。这个字段由硬件复位初始化。<br>0b - 禁用 1b - 启用 |

## 19.3.3 错误注入通道启用寄存器 (EICHEN)

### 19.3.3.1 偏移

| 注册 | 偏移 |
|---|---|
| 尹恩惠 | 4h |

### 19.3.3.2 职能

错误注入通道启用寄存器（EICHEN）的每个字段都用来启用或禁用相应的错误注入通道。

**注意事项**

要启用一个错误注入通道，全局错误注入启用（EIMCR[GEIEN]）字段也必须被主张。

### 19.3.3.3 图示

| 咬牙切齿 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | EICH0EN | EICH1EN | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 咬牙切齿 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

### 19.3.3.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>EICH0EN | 错误注入通道0启用<br>该字段启用相应的错误注入通道。全局错误注入启用（EIMCR[GEIEN]）字段也必须被断言以启用错误注入。<br>启用错误注入后，所有后续的读访问都会产生一个或多个位反转，这在相应的EICHDn_WORD寄存器中定义。错误注入一直有效，直到通过软件手动禁用错误注入通道。<br>对相应的 EICHDn_WORD 寄存器的任何写入都会清除相应的 EICHEN[EICHnEN] 字段，禁用错误注入通道。<br>0b - 错误注入通道0上的错误注入被禁用 1b - 错误注入通道0上的错误注入被启用 |

| | |
|---|---|
| 30<br>EICH1EN | 错误注入通道1启用<br>该字段启用相应的错误注入通道。全局错误注入启用（EIMCR[GEIEN]）字段也必须被断言以启用错误注入。<br>启用错误注入后，所有后续的读访问都会产生一个或多个位反转，这在相应的EICHDn_WORD寄存器中定义。错误注入一直有效，直到通过软件手动禁用错误注入通道。<br>对相应的 EICHDn_WORD 寄存器的任何写入都会清除相应的 EICHEN[EICHnEN] 字段，禁用错误注入通道。<br>0b - 错误注入通道1的错误注入被禁用 1b - 错误注入通道1的错误注入被启用 |
| 29 | 保留 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| - | |
| 28<br>- | 保留 |
| 27<br>- | 保留 |
| 26<br>- | 保留 |
| 25<br>- | 保留 |
| 24<br>- | 保留 |
| 23<br>- | 保留 |
| 22<br>- | 保留 |
| 21<br>- | 保留 |
| 20<br>- | 保留 |
| 19<br>- | 保留 |
| 18<br>- | 保留 |
| 17<br>- | 保留 |
| 16<br>- | 保留 |
| 15<br>- | 保留 |
| 14<br>- | 保留 |
| 13<br>- | 保留 |
| 12<br>- | 保留 |
| 11<br>- | 保留 |
| 10<br>- | 保留 |
| 9 | 保留 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| - | |
| 8<br>- | 保留 |
| 7<br>- | 保留 |
| 6<br>- | 保留 |
| 5<br>- | 保留 |
| 4<br>- | 保留 |
| 3<br>- | 保留 |
| 2<br>- | 保留 |
| 1<br>- | 保留 |
| 0<br>- | 保留 |

## 19.3.4 错误注入通道描述符n，字0 (EICHD0_W ORD0 - EICHD1_WORD0)

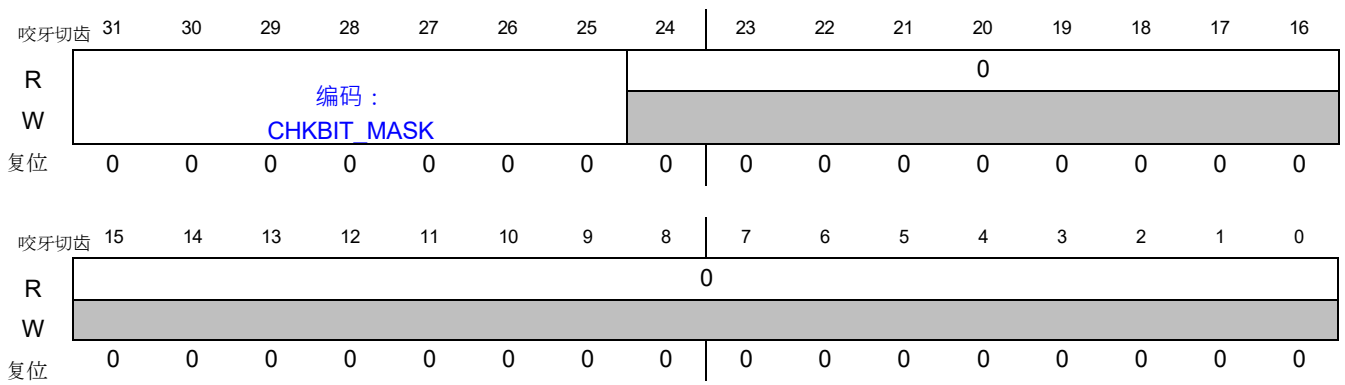
### 19.3.4.1 偏移

| 注册 | 偏移 |
|---|---|
| EICHD0_WORD0 | 100h |
| EICHD1_WORD0 | 200h |

### 19.3.4.2 职能

错误注入通道描述符的第一个字定义了一个左对齐的屏蔽字段。CHKBIT_MASK。CHKBIT_MASK的每一位都指定来自目标RAM的检查位总线的相应位是否应该被反转或者在读访问中保持不被修改。对这个字段的成功写入会清除相应的错误注入通道有效位，EICHEN[EICHnEN]。

### 19.3.4.3 图示




### 19.3.4.4 领域

| 场地 | 职能 |
|---|---|
| 31-25<br>编码：<br>CHKBIT_MASK | 检查位掩码<br>这个字段定义了一个位映射的掩码，指定来自目标RAM的检查位总线的相应位是否应该被反转或保持不被修改。<br>关于CHKBIT_MASK的位与通道的目标RAM的映射的任何独特细节，请参见芯片特定的EIM信息。<br>下表显示了每个通道的CHKBIT_MASK的宽度和位范围。 |

| 频道 | CHKBIT_MASK宽度（位）。 | CHKBIT_MASK位范围 |
|---|---|---|
| 0 | 7 | 6-0 |
| 1 | 7 | 6-0 |

**注意：**因为CHKBIT_MASK是左对齐的，所以位范围内的最高位总是在最有效位的位置。对于宽度为7位的CHKBIT_MASK，CHKBIT_MASK[6]位于最有效位的位置。对于任何宽度较小的CHKBIT_MASK，该宽度的位范围内的最高位是在最有意义的位的位置。

0b - 检查位总线的相应位保持不被修改。

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| | 1b - 检查位总线的相应位被反转。 |
| 24-0<br>- | 保留。 |

## 19.3.5 错误注入通道描述符n，字1 (EICHD0_W ORD1 - EICHD1_WORD1)

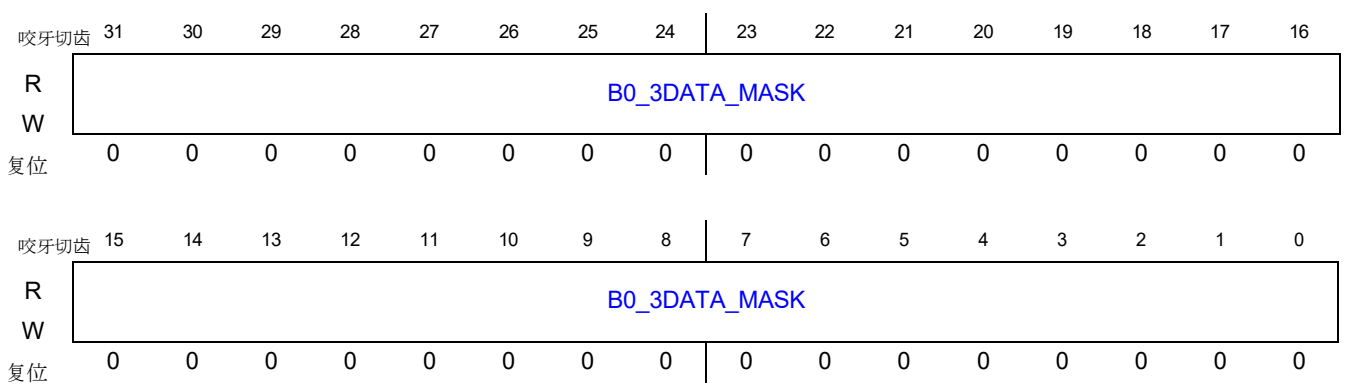
### 19.3.5.1 偏移

| 注册 | 偏移 |
|---|---|
| EICHD0_WORD1 | 104h |
| EICHD1_WORD1 | 204h |

### 19.3.5.2 职能

错误注入通道描述符的第二个字定义了一个右对齐的屏蔽字段。B0_3DATA_MASK中的位对应于读数据总线的0-3字节。每个位指定目标RAM的读数据总线的相应位是否应该被反转或者在读访问中保持不被修改。对这个字段的成功写入会清除相应的错误注入通道有效字段，EICHEN[EICHnEN]。

### 19.3.5.3 图示



### 19.3.5.4 领域

| 场地 | 职能 |
|---|---|
| 31-0<br>B0_3DATA_MASK | 数据掩码 字节0-3<br>这个字段定义了一个位映射的掩码，指定从目标RAM读取数据总线的相应位是否应该被反转或保持不被修改。<br>**注意：**对于每个通道。关于B0_3DATA_MASK的确切宽度和B0_3DATA_MASK所对应的具体DATA_MASK位，见错误注入通道描述符。DATA_MASK详情。<br>0b - 读取数据总线上的字节0-3的相应位保持不动。1b - 读取数据总线上的字节0-3的相应位被反转。 |

## 19.4 功能 描述

EIM通过执行两阶段的启用机制，为防止错误注入功能的意外启用和重新配置提供保护。为了正确启用一个通道的错误注入机制。

- 将1写入EICHEN[EICHnEN]字段，其中*n*表示通道号。
- 写1到EIMCR[GEIEN]。

**注意事项**

当一个通道的用例需要写入任何EICHDn_WORD寄存器时，在执行两阶段使能机制之前写入EICHDn_WORD寄存器。对任何EICHDn_WORD寄存器的成功写入会清除相应的EICHEN[EICHnEN]字段。

EIM支持2个错误注入通道。每个通道。

- 被分配到一个内存阵列接口。
- 拦截指定的存储器读取数据总线和校验位总线，并通过反转每个总线线路上选定位的传输值来注入错误。

在存储器的读取访问中，适用的EICHDn_WORD寄存器定义了读取数据和/或检查位总线的哪些位要反转。

图19-

1描述了一个示例存储器阵列的64位读数据总线和8位校验位数据总线的拦截和覆盖。

### 19.4.1 错误注入 场景

EIM支持这些错误注入的情况。

- 要产生一个单比特错误，只需反转EICHDn_WORD寄存器中CHKBIT_MASK或DATA_MASK的1位。
- 要产生一个多比特错误，只需反转EICHDn_WORD寄存器中CHKBIT_MASK或DATA_MASK的2位。

**注意事项**

试图在一次操作中反转2个以上的比特可能导致未定义的行为。

# 章 20
# 错误报告模块 (ERM)

## 20.1 特定芯片的ERM 信息

S32K1xx有一个ERM的实例。

### 20.1.1 内存错误的来源 事件

每个ERM通道*n*对应一个潜在内存错误事件的来源。下表显示了通道的分配情况。

**表20-1.ERM通道的内存错误事件源**

| ERM渠道*n* | 内存*n*事件源1 |
|---|---|
| 0 | Cortex-M4 TCM较低 |
| 1 | Cortex-M4 TCM上部 |

1.在本章中，"Memory n "表示由ERM通道n提供的内存阵列。

## 20.2 简介

### 20.2.1 概述

错误报告模块（ERM）提供与ECC（纠错码）有关的内存错误事件的信息和可选的中断通知。ERM收集平台本地存储器阵列的存储器访问的ECC事件，如闪存、系统RAM或外围RAM。关于支持的内存源和具体的内存通道分配的细节，请参见特定芯片的ERM信息。

### 20.2.2 特点

ERM包括这些特点。

- 捕获单比特纠正和不可纠正的ECC事件的地址信息
- 捕获ECC事件时可选择中断通知
- 支持内存源的ECC事件捕获，每个内存通道有单独的报告字段和中断配置

## 20.3 ERM寄存器 描述

ERM提供一个映射到平台上外设插槽的IPS编程模型。所有系统总线主站都可以访问该编程模型。

- 仅在监管者模式下
- 只使用32位（字）访问

以下任何试图引用编程模型的行为都会产生一个IPS错误终止。

- 在用户模式下
- 使用非32位的访问大小
- 对未定义（保留）的地址

ERM内存地图中一些32位宽的位置是只读的，并且总是读为0。对这些位置的读访问不会产生IPS错误终止。

- 104h
- 114h

当ERM处于操作过程中时，试图更新编程模型会导致非确定性的行为。

### 20.3.1 ERM内存 地图

ERM基础地址。4001_8000h

| 偏移 | 注册 | 宽度<br>(单位：比特) | 访问 | 重置值 |
|---|---|---|---|---|
| 0h | ERM配置寄存器0（CR0） | 32 | RW | 0000_0000h |
| 10h | ERM状态寄存器0（SR0） | 32 | W1C | 0000_0000h |
| 100h | ERM内存n错误地址寄存器（EAR0）。 | 32 | RO | 0000_0000h |
| 110h | ERM内存n错误地址寄存器（EAR1）。 | 32 | RO | 0000_0000h |

## 20.3.2 ERM配置寄存器0 (CR0)

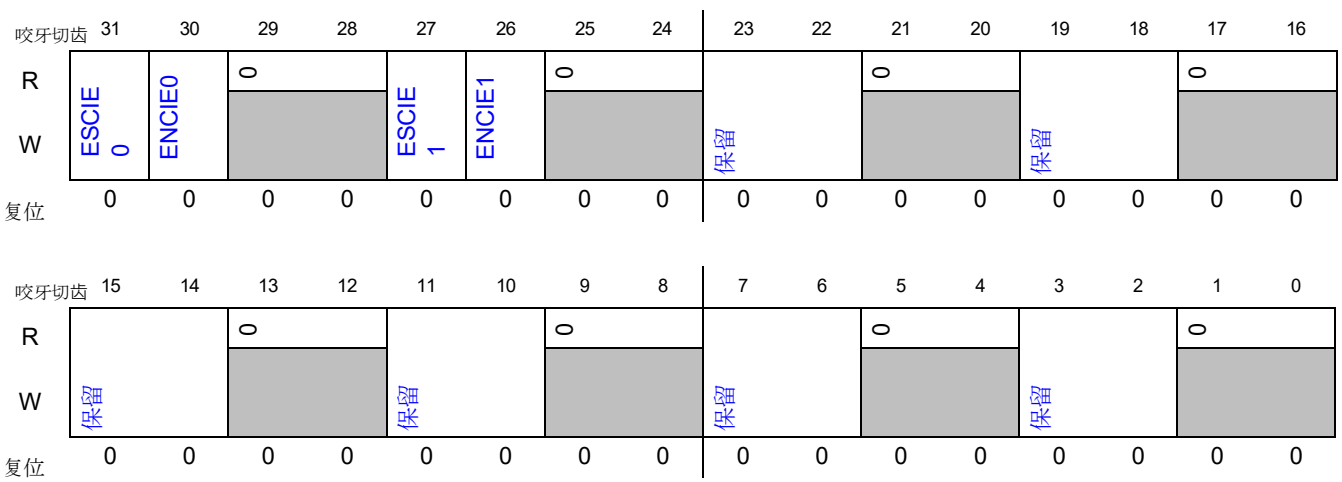
### 20.3.2.1 偏移

| 注册 | 偏移 |
|---|---|
| CR0 | 0h |

### 20.3.2.2 职能

这个32位的控制寄存器为每个支持的内存通道配置0到7的中断通知能力。

### 20.3.2.3 图示



### 20.3.2.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>ĀĀĀ | ĀĀĀ<br>Enable Memory 0 Single Correction Interrupt Notification<br>该字段通过硬件复位进行初始化。<br>**注意：**关于内存0映射的细节，请参考芯片特定的ERM信息。0b - 内存0单比特校正事件的中断通知被禁用。1b - 启用内存0单比特校正事件的中断通知。 |
| 30<br>ENCIE0 | ENCIE0<br>Enable Memory 0 Non-Correctable Interrupt Notification<br>该字段通过硬件复位进行初始化。<br>**注意：**关于内存0映射的细节，请参考芯片特定的ERM信息。0b - 内存0不可纠正的错误事件的中断通知被禁用。1b - 启用内存0不可纠正的错误事件的中断通知。 |
| 29-28<br>- | 保留 |
| 27<br>ĀĀĀ | ĀĀĀ<br>Enable Memory 1 Single Correction Interrupt Notification<br>该字段通过硬件复位进行初始化。<br>**注意：**关于内存1映射的细节，请参考芯片特定的ERM信息。0b - 内存1单比特校正事件的中断通知被禁用。1b - 启用内存1单比特校正事件的中断通知。 |
| 26<br>ENCIE1 | ENCIE1<br>Enable Memory 1 Non-Correctable Interrupt Notification<br>该字段通过硬件复位进行初始化。<br>**注意：**关于内存1映射的细节，请参考芯片特定的ERM信息。0b - 内存1不可纠正的错误事件的中断通知被禁用。1b - 启用内存1不可纠正的错误事件的中断通知。 |
| 25-24<br>- | 保留 |
| 23-22<br>- | 保留 |
| 21-20<br>- | 保留 |
| 19-18<br>- | 保留 |
| 17-16<br>- | 保留 |
| 15-14<br>- | 保留 |
| 13-12 | 保留 |

表继续在下一页...

| 场地 | 职能 |
|---|---|
| - | |
| 11-10<br>- | 保留 |
| 9-8<br>- | 保留 |
| 7-6<br>- | 保留 |
| 5-4<br>- | 保留 |
| 3-2<br>- | 保留 |
| 1-0<br>- | 保留 |

## 20.3.3 ERM状态寄存器0 (SR0)

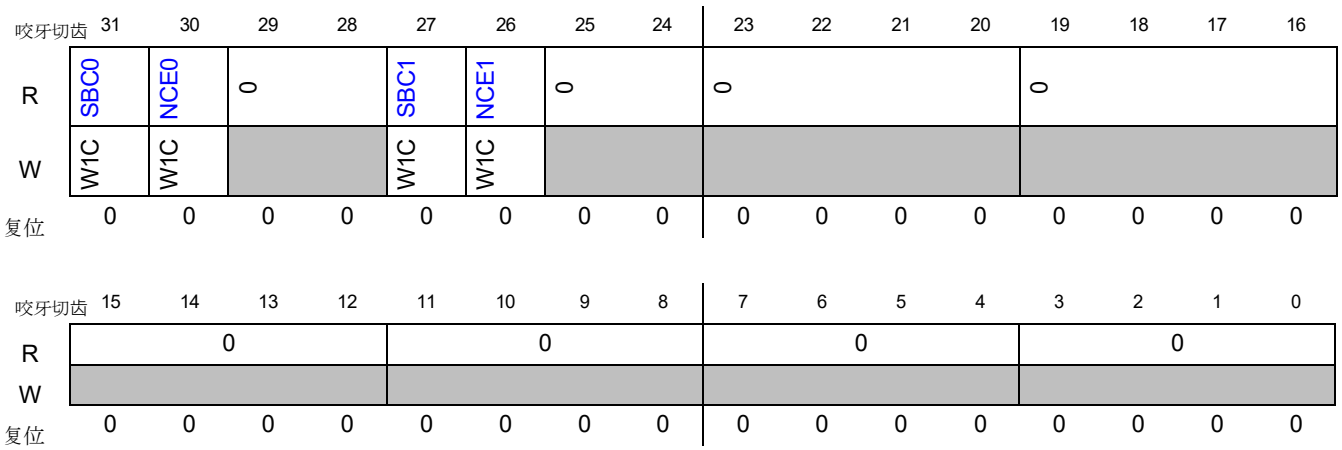
### 20.3.3.1 偏移

| 注册 | 偏移 |
|---|---|
| SR0 | 10h |

### 20.3.3.2 职能

这个32位的寄存器为每个内存通道检测到的ECC事件的类型提供信号。该寄存器为每个支持的内存通道在0到7之间检测到的最后一个内存事件提供信号。

### 20.3.3.3 图示




### 20.3.3.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>SBC0 | SBC0<br>存储器0单比特校正事件 该字段由硬件复<br>位初始化。<br>写1来清除这个字段。如果CR0[ESCIE0]被启用，该写法也将清除相应的中断通知。<br>**注意：**关于内存0映射的细节，请参考特定芯片的ERM信息。<br>0b - 没有检测到内存0的单比特校正事件 1b - 检测到内存0<br>的单比特校正事件 |
| 30<br>NCE0 | NCE0<br>内存0不可纠正的错误事件 该字段由硬件复<br>位初始化。<br>写1来清除这个字段。如果CR0[ENCIE0]被启用，这个写入也会清除相应的中断通知。<br>**注意：**关于内存0映射的细节，请参考特定芯片ERM信息。<br>0b - 检测到内存0上没有不可纠正的错误事件 1b - 检测到内<br>存0上有不可纠正的错误事件 |
| 29-28<br>- | 保留 |

| | |
|---|---|
| 27<br>SBC1 | SBC1<br>内存1单比特校正事件 该字段通过硬件复<br>位初始化。<br>写1来清除这个字段。如果CR0[ESCIE1]被启用，该写法也将清除相应的中断通知。<br>**注意：**关于内存1映射的细节，请参考特定芯片ERM信息。 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | 0b - 没有检测到内存1的单比特校正事件 1b - 检测到内存1<br>的单比特校正事件 |
| 26<br>NCE1 | NCE1<br>内存1不可纠正的错误事件 该字段由硬件复<br>位初始化。<br>写1来清除这个字段。如果CR0[ENCIE1]被启用，该写法也清除了相应的中断通知。<br>**注意：**关于内存1映射的细节，请参考特定芯片ERM信息。<br>0b - 检测到内存1上没有不可纠正的错误事件 1b - 检测到内<br>存1上有不可纠正的错误事件 |
| 25-24<br>- | 保留 |
| 23-20<br>- | 保留 |
| 19-16<br>- | 保留 |
| 15-12<br>- | 保留 |
| 11-8<br>- | 保留 |
| 7-4<br>- | 保留 |
| 3-0<br>- | 保留 |

## 20.3.4 ERM内存n错误地址寄存器（EAR0 - EAR1）。

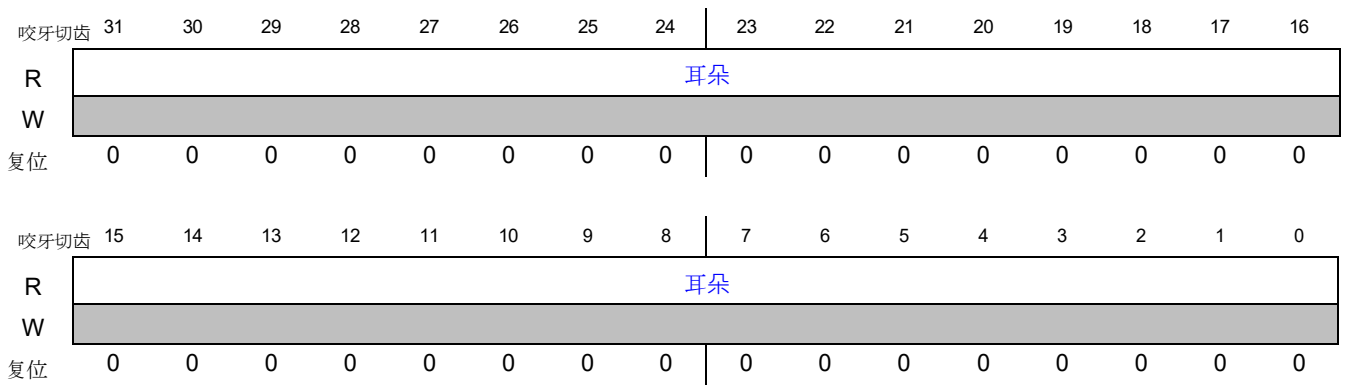
### 20.3.4.1 偏移

| 注册 | 偏移 |
| --- | --- |
| EAR0 | 100h |
| EAR1 | 110h |

### 20.3.4.2 职能

每个EARn是一个32位的寄存器，用于捕获Memory中最后一个ECC事件的地址。n，其中*n*表示内存通道。任何试图写入EARn的行为都会被忽略。

### 20.3.4.3 图示




### 20.3.4.4 领域

| 场地 | 职能 |
|---|---|
| 31-0<br>耳朵 | 耳朵<br>存储器*n*错误地址 - 该字段包含最后记录的ECC事件的故障系统地址<br>关于内存*n*。<br>**注意**：关于Memory *n*映射的细节，请参考特定芯片的ERM信息。 |

## 20.4 功能 描述

### 20.4.1 单位校正 事件

当检测到存储器*n*上的一个单比特修正事件时，ERM。

- 通过将适用的状态寄存器位的值改为1来记录该事件。SRx[SBCn]
- 在内存中记录启动事件的相应访问地址 *n*

错误地址寄存器。EARn（如果该通道有此寄存器）

ERM只保存最后报告的事件的信息。

要清除一个事件的记录，向SRx[SBCn]写1，将其值改为0。

#### 20.4.1.1 单位校正事件的可选中断通知

ERM提供了一个选项，在报告单比特校正事件时产生一个中断通知。本序列描述了如何使用该选项。

1. 要启用内存*n*上的单比特校正事件的中断通知，将CRx[ESCIEn]设置为1。
2. 随后，当检测到存储器*n*上的单比特修正事件时，ERM。
   - 像往常一样记录事件和地址
   - 另外，发送一个与事件对应的中断通知
3. 要同时清除一个事件的记录和相应的中断通知，请向SRx[SBCn]写1，将其值改为0。

### 20.4.2 不可纠正的错误 事件

当检测到内存*n*上有一个不可纠正的ECC错误事件时，ERM。

- 通过将适用的状态寄存器位的值改为1来记录该事件。SRx[NCEn]
- 在内存中记录启动事件的相应访问地址 *n*
  错误地址寄存器。EARn（如果该通道有此寄存器）。

ERM只保存最后报告的事件的信息。

要清除一个事件的记录，向SRx[NCEn]写1，将其值改为0。

#### 20.4.2.1 对不可纠正的错误事件可选择中断通知

ERM提供了一个选项，在报告不可纠正的ECC事件时产生一个中断通知。本序列描述了如何使用该选项。

1. 要启用内存*n*上不可纠正的错误事件的中断通知，将CRx[ENCIEn]设置为1。
2. 随后，当检测到存储器*n*上有不可纠正的错误事件时，ERM。
   - 像往常一样记录事件和地址
   - 另外，发送一个与事件对应的中断通知
3. 要同时清除一个事件的记录和相应的中断通知，请向SRx[NCEn]写1，将其值改为0。

## 20.5 初始化

对于每个ERM通道，在启用关于该内存错误的ERM中断之前，准备好相应的内存阵列。

1. 将存储器初始化为一个已知值，以便存储正确的相应ECC码。
2. 在存储器的初始化过程中，如果ERM捕捉到任何ECC错误事件的信息，则清除存储该事件记录的相应SRx[SBCn]或SRx[NCEn]字段。
3. 对适用的CRx[ESCIEn]和CRx[ENCIEn]字段进行编程，根据需要启用ERM中断。

# 第二十一章
# 看门狗定时器（WDOG）

## 21.1 特定芯片的WDOG 信息

### 21.1.1 WDOG clocks

WDOG模块有以下可选择的时钟源。

- 内部低功率振荡器(LPO_CLK)
- 内部慢速IRC时钟（SIRC）。
- 系统振荡器时钟(SOSC)
- 总线时钟

**注意事项**

WDOG_CNT的重置读取值会因时间戳的不同而不同，因为它是一个默认的运行计数器。

### 21.1.2 WDOG低功率 模式

该表显示了WDOG的低功耗模式和相应的芯片低功耗模式。

该设备不支持等待模式。关于可用的电源模式的详情，请参见模块在可用的低功率模式下的操作。

### 21.1.3 默认看门狗 超时

超时的时间取决于看门狗计数器的时钟源。在初始上电后，看门狗由LPO128K_CLK提供时钟，并在1024个周期后发出超时。这导致看门狗在大约8ms后产生超时，这将迫使MCU复位。为了避免这种情况，请确保看门狗在1024个周期结束前被配置或刷新。

## 21.2 简介

看门狗定时器（WDOG）模块是一个独立的定时器，可供系统使用。它提供了一个安全功能，以确保软件按计划执行，CPU不会陷入无限循环或执行非预期的代码。如果WDOG模块在一定时间内没有得到服务（刷新），它将重置MCU。

### 21.2.1 特点

WDOG模块的特点包括。

- 独立于总线时钟的可配置时钟源输入
  - 总线时钟（慢速时钟）
  - LPO时钟
  - INTCLK（内部时钟）
  - ERCLK（外部参考时钟）
- 可编程的超时时间
  - 可编程的16位超时值
  - 当需要更长的超时时间时，可选择固定256时钟预分频器
- 用于计数器刷新的稳健的写入序列
  - 写0xA602然后写0xB480的刷新顺序
- 刷新机制的窗口模式选项
  - 可编程的16位窗口值
  - 提供强有力的检查，确保程序流比预期的要快
  - 早期的刷新尝试会触发重置。
- 可选的超时中断，允许后处理诊断
  - 向CPU发出中断请求，并为中断服务程序（ISR）提供中断向量。
  - 强制复位发生在获取中断向量后的128个总线时钟。

- 配置位在复位后一次写入，以确保看门狗配置不会被错误地改变。
- 用于解锁一次性写入的配置位的强健写入序列
  - 先写0xC520，再写0xD928的解锁顺序，允许更新一次性写入的配置位
  - 软件必须在解锁后和WDOG关闭解锁窗口前的128个总线时钟内进行更新。

## 21.2.2 块 图

下图是WDOG模块的方框图。

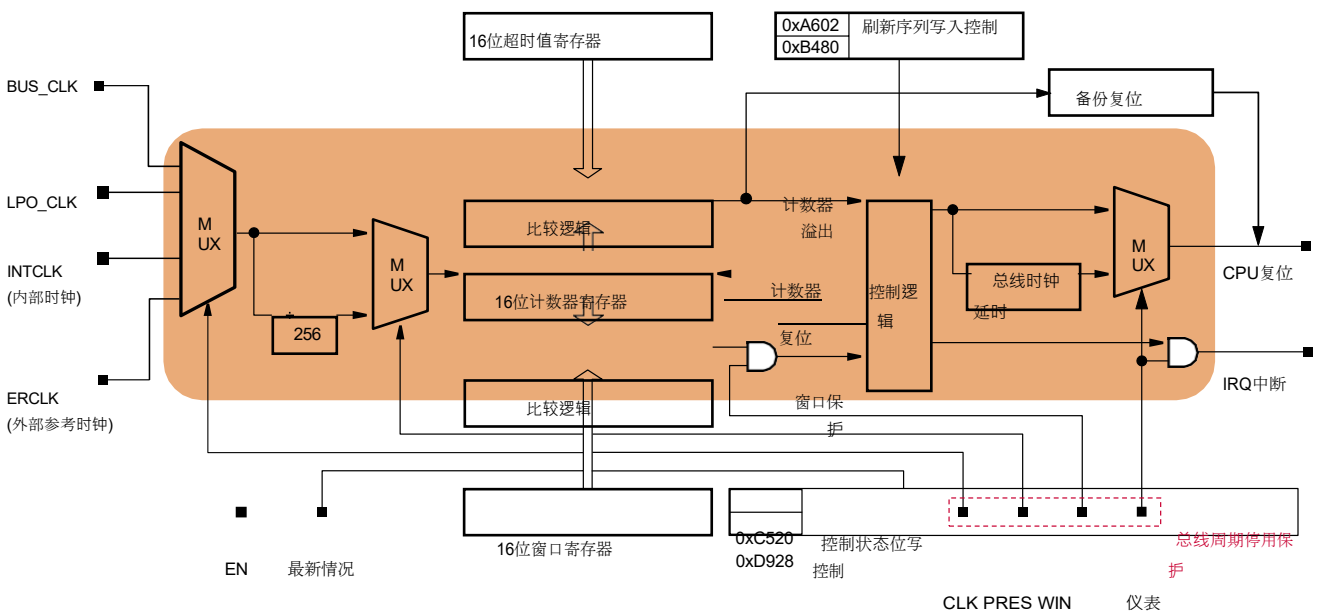



图21-1.WDOG方框图

# 21.3 内存图和寄存器 定义

## 21.3.1 WDOG寄存器 描述

### 21.3.1.1 WDOG 内存图

WDOG基础地址。4005_2000h

| 偏移 | 注册 | 宽度 (单位：比特) | 访问 | 重置值 |
|---|---|---|---|---|
| 0h | 看门狗控制和状态寄存器（CS） | 32 | RW | 0000_2980h |
| 4h | 看门狗计数器寄存器（CNT） | 32 | RW | 0000_0000h |
| 8h | 看门狗超时值寄存器（TOVAL）。 | 32 | RW | 0000_0400h |
| 丘吉尔 | 看门狗窗口寄存器（WIN） | 32 | RW | 0000_0000h |

### 21.3.1.2 看门狗控制和状态寄存器（CS）

#### 21.3.1.2.1 偏移

| 注册 | 偏移 |
|---|---|
| CS | 0h |

#### 21.3.1.2.2 职能

本节介绍看门狗控制和状态寄存器的功能。

**注意事项**

TST仅在POR时被清零（0:0）。任何其他的复位都不会影响这个字段的值。

#### 21.3.1.2.3 图示

| 咬牙切齿 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 咬牙切齿 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | WIN | FLG | CMD32EN | 初步统计 | ULK | RCS | 循环器 | | EN | INT | 重新 | TST | | DBG | 等待 | 停止 |
| | | W1C | | | | | | | | | | | | | | |
| | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 初步 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

W

复位

### 21.3.1.2.4 领域

| 场地 | 职能 |
|---|---|
| 31-16<br>- | 保留 |
| 15<br>WIN | 看门狗窗口<br>这个只写一次的位启用窗口模式。参见窗口模式部分。<br>0b - 关闭窗口模式。1b - 启用窗口模式。 |
| 14<br>FLG | 看门狗中断标志<br>当INT在控制和状态寄存器1中被设置时，该位是一个中断指示器。写1来清除它。<br>0b - 没有发生中断。1b - 发生了一个中断。 |
| 13<br>CMD32EN | 启用或禁用WDOG对32位（否则为16位或8位）刷新/解锁命令写入字的支持<br>这是一次性写入的字段，用户在写入这个字段后需要解锁WDOG进行重新配置。<br>0b - 禁用对32位刷新/解锁命令写入字的支持。只支持16位或8位。<br>1b - 启用对32位刷新/解锁命令写入字的支持。不支持16位或8位。 |
| 12<br>预先声明 | 看门狗预分频器<br>这个一次性写入的位可以使看门狗计数器的参考时钟有一个固定的256预分频。(方框图显示了这个时钟分频器选项)。<br>0b - 禁用256预分频器。1b - 启用256预分频器。 |
| 11<br>ULK | 解锁状态<br>这个只读位表示WDOG是否被解锁。<br>0b - WDOG被锁定。<br>1b - WDOG已被解锁。 |
| 10<br>RCS | 重新配置成功<br>这个只读的位表示重新配置是否成功。默认复位值为0.当新的配置生效时，该位被设置，并通过成功的解锁命令被清除。<br>0b - 重新配置WDOG。<br>1b - 重新配置成功。 |
| 9-8<br>循环器 | 看门狗时钟<br>这个只写一次的字段表示为看门狗计数器提供的时钟源。参见时钟源部分。<br>00b - 总线时钟<br>01b - LPO时钟<br>10b - INTCLK（内部时钟）。<br>11b - ERCLK（外部参考时钟）。 |

| | |
|---|---|
| 7<br>EN | 看门狗启用<br>这个一次性写入的位使看门狗计数器开始计数。<br>0b - 看门狗禁用。<br>1b - 启用看门狗。 |
| 6<br>INT | 看门狗中断 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | 在复位触发事件（超时或对看门狗的非法写入）发生时，该写位将看门狗配置为立即产生中断请求，然后强制复位。在中断向量获取之后（在复位触发事件之后），复位在128个总线时钟的延迟之后发生。<br>0b - 看门狗中断被禁用。看门狗复位没有延迟。<br>1b - 看门狗中断被启用。看门狗复位的时间比中断向量获取的时间延迟128个总线时钟。 |
| 5<br>更新 | 允许更新<br>这个一次性写入的位允许软件在没有复位的情况下重新配置看门狗。<br>0b - 不允许更新。在初始配置后，如果不强制复位，以后就不能再修改看门狗。<br>1b - 允许更新。软件可以在执行解锁写入序列后的128个总线时钟内修改看门狗配置寄存器。 |
| 4-3<br>TST | 看门狗测试<br>启用快速测试模式。该测试模式允许软件行使计数器的所有位，以证明看门狗的功能正常。参见看门狗的快速测试部分。<br>这个一次性写入的字段只在POR时被清零（0:0）。任何其他的复位都不会影响这个字段的值。<br>00b - 看门狗测试模式禁用。<br>01b - 看门狗用户模式启用。(看门狗测试模式禁用。)在测试完看门狗后，软件应该用这个设置来表示看门狗在用户模式下运行正常。10b - 看门狗测试模式启用，只有低字节被使用。CNT[CNTLOW]与TOVAL[TOVALLOW]比较。<br>11b - 看门狗测试模式启用，只使用高字节。CNT[CNTHIGH]与TOVAL[TOVALHIGH]比较。 |
| 2<br>DBG | 调试启用<br>当芯片处于调试模式时，这个只写一次的位使看门狗能够工作。<br>0b - 在芯片调试模式下禁用看门狗。1b - 在芯片调试模式下启用看门狗。 |
| 1<br>等待 | 等待启用<br>当芯片处于等待模式时，这个一写即逝的位使看门狗工作。<br>0b - 在芯片等待模式下禁用看门狗。1b - 在芯片等待模式下启用看门狗。 |
| 0<br>停止 | 停止启用<br>这个一写的位子使看门狗在芯片处于停止模式时能够工作。<br>0b - 在芯片停止模式下禁用看门狗。1b - 在芯片停止模式下启用看门狗。 |

## 21.3.1.3 看门狗计数器寄存器（CNT）

### 21.3.1.3.1 偏移

| 注册 | 偏移 |
| --- | --- |
| CNT | 4h |

### 21.3.1.3.2 职能

本节介绍看门狗计数器寄存器。

看门狗计数器寄存器提供了对自由运行的看门狗计数器值的访问。软件可以在任何时候读取该计数器寄存器。

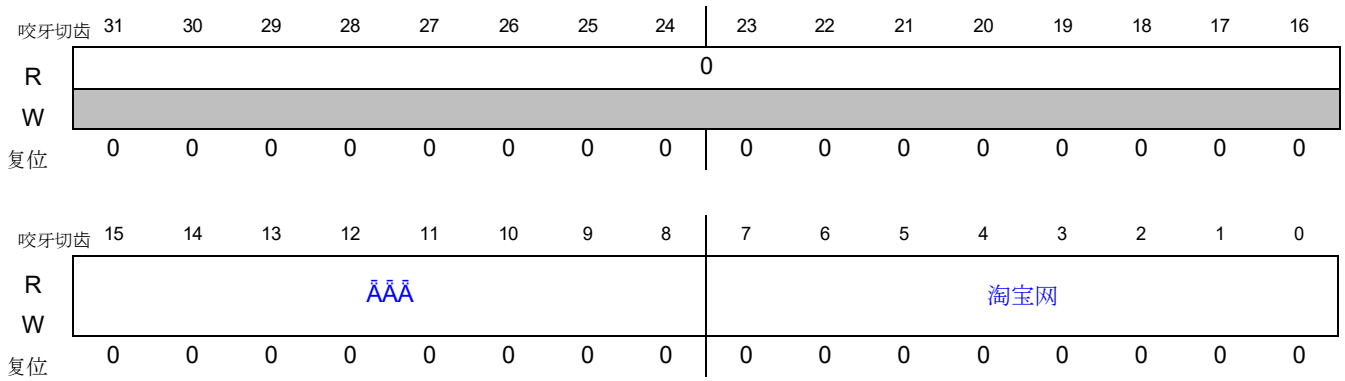
软件不能直接写入看门狗计数器；但是，对这些寄存器的两个写入序列有特殊的功能。

1. *刷新序列*将看门狗计数器复位到0x0000。见 "刷新看门狗 "一节。
2. *解锁序列*允许在不强制复位的情况下对看门狗进行重新配置（当CS[UPDATE] = 1 时）。参见 "可重新配置的配置 "部分。

**注意事项**

所有其他对该寄存器的写入都是非法的，并强制复位。

### 21.3.1.3.3 图示




### 21.3.1.3.4 领域

| 场地 | 职能 |
|---|---|
| 31-16<br>- | 保留 |
| 15-8<br>ĂĂĂ | 看门狗计数器的高字节 |
| 7-0<br>淘宝网 | 看门狗计数器的低字节 |

## 21.3.1.4 看门狗超时值寄存器（TOVAL）。

### 21.3.1.4.1 偏移

| 注册 | 偏移 |
|---|---|
| 托瓦尔 | 8h |

### 21.3.1.4.2 职能

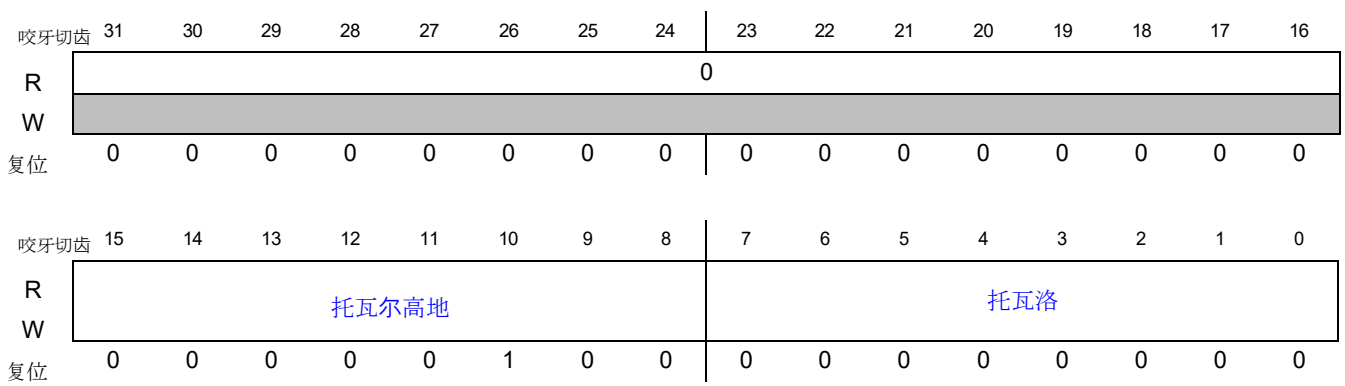
本节介绍看门狗超时值寄存器。TOVAL包含用于设置看门狗超时周期的16位数值。

看门狗计数器（CNT）不断与超时值（TOVAL）进行比较。如果计数器达到超时值，看门狗会强制复位触发事件。

**注意事项**

不要在看门狗超时值寄存器中写0，否则，看门狗总是会产生复位。

### 21.3.1.4.3 图示




### 21.3.1.4.4 领域

| 场地 | 职能 |
|---|---|
| 31-16<br>- | 保留 |
| 15-8<br>托瓦尔高地 | 超时值的高字节 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| 7-0<br>托瓦洛 | 超时值的低字节 |

## 21.3.1.5 看门狗窗口寄存器（WIN）

### 21.3.1.5.1 偏移

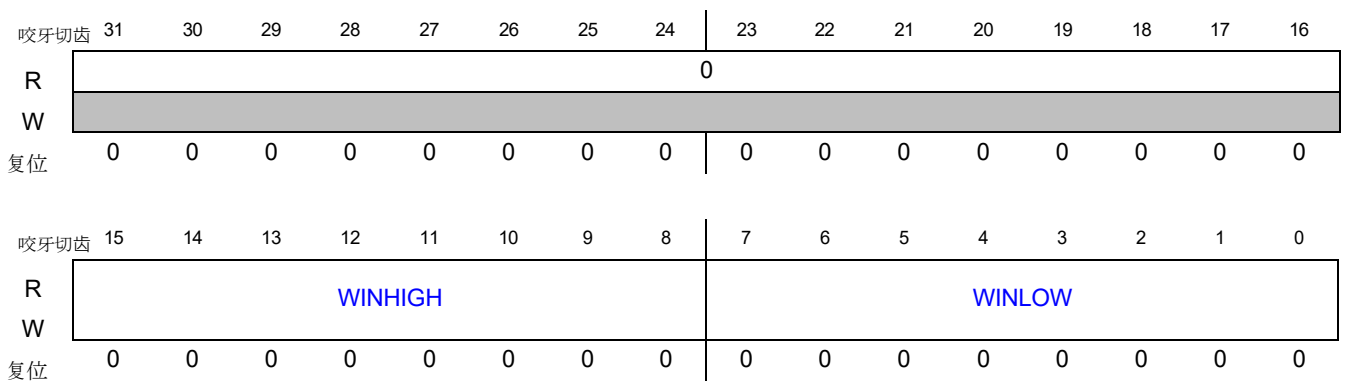
| 注册 | 偏移 |
| --- | --- |
| WIN | 丘吉尔 |

### 21.3.1.5.2 职能

本节介绍看门狗窗口寄存器。当窗口模式启用时（CS[WIN]被设置），WIN寄存器决定了刷新序列被认为有效的最早时间。参见看门狗刷新机制部分。

WIN寄存器的值必须小于TOVAL寄存器的值。

### 21.3.1.5.3 图示




### 21.3.1.5.4 领域

| 场地 | 职能 |
| --- | --- |

| | |
|---|---|
| 31-16<br>- | 保留 |

*表继续在下一页...*

职能描述

| 场地 | 职能 |
| --- | --- |
| 15-8<br>WINHIGH | 看门狗窗口的高字节 |
| 7-0<br>WINLOW | 看门狗窗口的低字节 |

## 21.4 功能 描述

WDOG模块提供了一个故障安全机制，以确保在系统故障的情况下，如CPU时钟停止或软件代码出现失控的情况下，系统可以被重置到一个已知的操作状态。看门狗计数器通过一个可选择的时钟源连续运行，并期望定期得到服务（刷新）。如果没有，它将产生一个复位触发事件。

**超时周期、窗口模式和时钟源都是可编程的，但必须在复位后的128个总线时钟内进行配置。**

### 21.4.1 时钟 源

通过对CS[CLK]的编程，看门狗计数器有如下的时钟源选择。

- 总线时钟
- 低功率振荡器时钟（LPO_CLK）（这是*默认*来源）。
- 内部时钟
- 外部时钟

这些选项允许软件选择独立于总线时钟的时钟源，用于需要满足更强大的安全要求的应用。使用总线时钟以外的时钟源可以确保看门狗计数器在总线时钟停止的情况下继续运行；见备份复位。

所有时钟源的可选固定预分频器允许更长的超时周期。当CS[PRES]被设置时，时钟源在为看门狗计数器提供时钟前被预分频为256。

下表总结了不同的看门狗超时周期，作为一个例子，这些周期可能是可用的。

表21-1.看门狗超时的可用性（示例）

| 参考时钟 | 预分频器 | 看门狗超时可用性 |
|---|---|---|
| LPO_CLK | 通过 | ~1 ms-65.5 s（如果LPO_CLK = 1 kHz）；（~1 ms-65.5 s）/128（如果LPO_CLK = 128 kHz）。[1] |
| | ÷256 | ~256 ms-16,777.2 s（如果LPO_CLK = 1 kHz）；~2 ms-131.1 s（如果LPO_CLK = 128 kHz）。 |
| 内部时钟8MHz | 通过 | 125 ns-8.1925 ms |
| | ÷256 | 32微秒-2.09728秒 |
| 1 MHz (来自总线或外部) | 通过 | 1 µs-65.54 ms |
| | ÷256 | 256微秒-16.777秒 |
| 20 MHz (来自总线或外部) | 通过 | 50 ns-3.277 ms |
| | ÷256 | 12.8 µs-838.8 ms |

1.复位后的默认超时值约为1024 ms（如果LPO_CLK = 1 kHz），或1024/128 ms（如果LPO_CLK = 128 kHz）。

**注意事项**

当程序员在重新配置过程中切换时钟源时，看门狗硬件会在配置时间段（128个总线时钟）结束后的2.5个周期内将计数器保持在零位，并在新的时钟源上保持2.5个周期。这个延迟保证了在用新的配置重新启动计数器之前的平稳过渡。

## 21.4.2 看门狗刷新 机制

如果看门狗计数器没有被刷新，看门狗会重置MCU。一个强大的刷新机制使得看门狗被失控代码刷新的可能性很小。

要刷新看门狗计数器，软件必须在超时期结束前执行刷新写入序列。此外，如果使用窗口模式，软件必须在WIN寄存器中设置的时间值之后才能开始刷新序列。见下图。

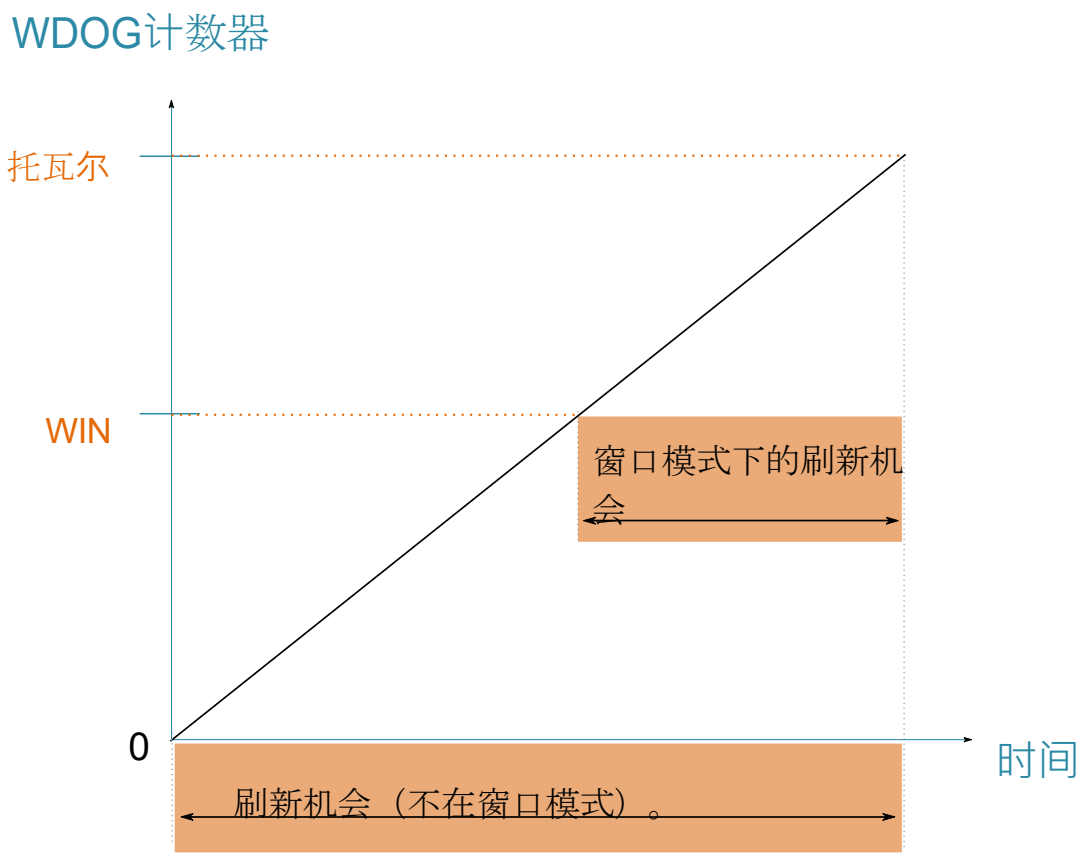



**图21-2.看门狗计数器的刷新机会**

### 21.4.2.1 窗口模式

软件完成其主控制循环的速度比预期的要快，这可能是一个问题的迹象。根据应用的要求，WDOG可以被编程为当刷新尝试过早时强制复位。

当窗口模式被启用时，看门狗必须在计数器达到最小预期时间值后被刷新；否则，看门狗会重置MCU。最小预期时间值在WIN寄存器中指定。设置CS[WIN]可以启用窗口模式。

### 21.4.2.2 刷新看门狗

刷新写入顺序可以是

- 如果WDOG_CS[CMD32EN]为0，则为两个16位的写入（0xA602, 0xB480）或四个8位的写入（0xA6, 0x02, 0xB4, 0x80；适用于Cortex-M核心的情况）。

- 如果WDOG_CS[CMD32EN]为1，则有一个32位的写入（0xB480_A602）。

到CNT寄存器。这两种方法都必须在WDG超时前发生，否则，看门狗会重置MCU。

**注意事项**

在开始刷新序列之前，禁用全局中断。否则，如果中断发生在刷新写入完成之前，可能会有效地使刷新序列失效。在序列完成后，恢复全局中断控制状态。

示例代码可在本章末尾找到。

## 21.4.3 配置 看门狗

### 21.4.3.1 配置看门狗一次

**所有的看门狗控制位、超时值和窗口值在*128个总线时钟内*复位后都是一次写入。这意味着，在写完之后，除非发生复位，否则不能改变。**这可以通过用户先配置窗口和超时值，然后再配置其他控制位，并确保CS[UPDATE]也被设置为0来保证。

这提供了一个强大的机制来配置看门狗，并确保在配置后，失控的情况下不能错误地禁用或修改看门狗配置。

新的配置只有在复位后写入除CNT以外的所有寄存器后才会生效。否则，WDOG默认使用复位值。如果不使用窗口模式（CS[WIN]为0），则不需要写入WIN来使新配置生效。

### 21.4.3.2 重新配置看门狗

在某些情况下（如支持引导程序功能时），你可能想重新配置或禁用看门狗，*而不必先强制复位*。

- 通过在复位后初始配置看门狗时将CS[UPDATE]设置为1，你可以在任何时候通过执行解锁序列重新配置看门狗。
- 反之，如果CS[UPDATE]保持0，重新配置看门狗的唯一方法就是启动复位。

解锁序列与刷新序列相似，但使用不同的值。

#### 21.4.3.2.1 解锁看门狗

解锁顺序是在配置看门狗后的任何时候，在16个总线时钟内向CNT寄存器写入0xC520和0xD928。在完成解锁序列后，用户必须在128个总线时钟内重新配置看门狗；否则，看门狗将关闭解锁窗口。

**注意事项**

由于重新配置看门狗需要128个总线时钟，在重新配置看门狗后执行STOP或WAIT指令前必须插入一些延时。这可以确保看门狗的新配置在MCU进入低功耗模式之前生效。否则，MCU可能无法从低功耗模式下被唤醒。

示例代码可在本章末尾找到。

### 21.4.4 使用中断来延迟 复位。

- **当中断被启用时（CS[INT]=1）。**在一个复位触发事件（如计数器超时或无效的刷新尝试）发生后，看门狗首先产生一个中断请求。接下来，看门狗在强制复位前会延迟128个总线时钟（从中断向量获取，而不是复位触发事件），以允许中断服务程序（ISR）执行任务（如分析堆栈以调试代码）。
- **当中断被禁止时（CS[INT]=0）：**看门狗不会延迟强制复位。

### 21.4.5 备份 重置

**注意事项**

必须使用总线时钟以外的时钟源作为计数器的参考时钟；否则，备份复位功能就无法使用。

备用复位功能是一种保障功能，在主WDOG逻辑失去其时钟（总线时钟）而不能再监视计数器时，独立地产生复位。如果看门狗计数器连续溢出两次（没有中间的复位），备份复位功能就会生效并产生复位。

当中断使能且看门狗时钟来自非总线时钟时，备份复位有效。如果启用了中断，一旦总线时钟在退出中断程序前被切断，正常的看门狗复位将被阻断。在这种情况下，第二次溢出将直接导致备份复位。

### 21.4.6 调试和低功耗模式下的功能

默认情况下，看门狗在Debug模式、Wait模式或Stop模式下是不工作的。然而，看门狗可以在这些模式下保持功能，如下所示。

- **对于Debug模式，** 设置CS[DBG]。(这样看门狗在Debug模式下是有功能的，即使CPU被Debug模块所控制)。
- **对于等待模式，**设置CS[WAIT]。
- **对于停止模式，**设置CS[STOP]、CS[WAIT]，并确保时钟源在停止模式下是有效的。

**注意事项**

看门狗可以在停止模式下产生中断。

对于调试模式和停止模式，除了上述配置外，还必须使用总线时钟以外的时钟源作为计数器的参考时钟；否则，看门狗不能发挥作用。

### 21.4.7 看门狗的快速测试

在安全关键应用中执行应用程序代码之前，用户需要测试看门狗是否按预期工作并重置MCU。通过让它运行到溢出值来测试一个16位计数器的每一位，需要相对较长的时间（64kHz时钟）。

为了帮助尽量减少复位后应用程序代码的启动延迟，看门狗有一个功能，即通过将计数器分割成其组成的字节范围的阶段来更快地测试看门狗。低字节和高字节是独立运行的，并根据超时值寄存器的相应字节进行超时测试。(对于完整的覆盖
当测试计数器的高字节时，测试功能通过低字节的第8位输入时钟，从而确保从低字节到高字节的溢出连接被测试）。

使用这个测试功能可以将测试时间减少到512个时钟（不包括开销，如用户配置和复位向量的获取）。为了进一步加快测试速度，使用更快的时钟（如总线时钟）作为计数器参考。

上电复位时，系统复位寄存器中的POR位被设置，表明用户应该执行WDOG快速测试。

#### 21.4.7.1 测试计数器的每个字节

测试程序遵循以下步骤。

1. 在看门狗配置时间段内，在TOVAL寄存器中编程首选看门狗超时值。
2. 使用CS[TST]=10b的低字节选择计数器的一个字节进行测试；CS[TST]=11b的高字节。
3. 等待看门狗超时。可以选择在空闲循环中，增加RAM的位置，作为一个并行的软件计数器，以便以后比较。因为RAM不受看门狗复位的影响，看门狗计数器的超时周期可以与软件计数器进行比较，以验证超时周期是否如预期的那样发生。
4. 看门狗计数器超时并强制复位。
5. 确认系统复位寄存器中的WDOG标志被设置，表明是看门狗引起的复位。(POR标志保持清零)。
6. 确认CS[TST]显示进行了一次测试（10b或11b）。

如果确认，计数和比较功能对所选字节起作用。重复该程序，在步骤2中选择另一个字节。

**注意事项**

CS[TST]只被POR清零，不受其他复位的影响。

#### 21.4.7.2 进入用户模式

在成功测试看门狗计数器的低字节和高字节后，用户可以将CS[TST]配置为01b，表示看门狗可以在应用用户模式下使用。因此，如果再次发生复位，软件可以将复位触发器识别为由失控或错误的应用代码引起的真正的看门狗复位。

在使用默认的LPO时钟源时，作为一个持续的测试，软件可以定期读取CNT寄存器

，以确保计数器被递增。

## 21.5 申请 信息

看门狗在复位后默认是启用的。要禁用或重新配置看门狗，最好在第一次看门狗超时前完成。建议在软件代码的最开始禁用或重新配置看门狗，例如在启动或主功能的开始。

**注意事项**

当用户配置看门狗时，需要至少2.5个周期的看门狗时钟才能生效。这意味着用户两次配置的时间间隔必须大于2.5个时钟。

**注意事项**

当芯片从BOOT ROM启动并跳转到Flash时，看门狗将在启动程序开始时被激活，并在启动程序退出时被禁用。如果在flash程序中有任何代码想重新配置看门狗，必须在bootloader退出后2.5个看门狗时钟后运行。

为了在不强制复位的情况下禁用或重新配置看门狗，WDOG_CS[UPDATE]位必须在WDOG模块的初始配置中被设置。然后，在超时限制内的任何时间都可以使用解锁序列来重新配置看门狗。

### 21.5.1 禁用 看门狗

要禁用看门狗，首先进行解锁程序，然后解除WDOG_CS[EN]位。 下面的代码片段显示了一个32位写入的例子。

```
DisableInterrupts; //禁用全局中断 WDOG_CNT =
0xD928C520; //解锁看门狗 WDOG_CS &=
~WDOG_CS_EN_MASK; //停用看门狗 EnableInterrupts;
//启用全局中断
```

### 21.5.2 复位后禁用看门狗

所有的看门狗寄存器在复位时被解锁。因此，解锁序列是不必要的，但它需要写入所有的看门狗寄存器以使新的配置生效。下面的代码片段显示了一个复位后禁用看

门狗的例子。

```
DisableInterrupts; //禁用全局中断 WDOG_CS &=
~WDOG_CS_EN_MASK; //禁用看门狗 WDOG_TOVAL= 0xFFFF。
while(WDOG_CS[ULK]);   // 等待锁的到来
while(~WDOG_CS[RCS]); //等待新配置生效 EnableInterrupts; //启用全局中断
```

## 21.5.3 配置 看门狗

看门狗可以通过设置WDOG_CS[UPDATE]=0来配置一次，此后看门狗不能被重新配置，直到复位。如果在配置看门狗时设置WDOG_CS[UPDATE]=1，看门狗可以被重新配置而不需要强制复位。下面的示例代码显示了如何配置看门狗而不使用窗口模式，时钟源为LPO，中断启用，超时值为256时钟。下面的代码片段显示了一个32位写入的例子。

*配置一次*

```
DisableInterrupts; //禁用全局中断 WDOG_CNT =
0xD928C520; //解锁看门狗
while(WDOG_CS[ULK]==0);  //等待直到寄存器被解锁 WDOG_TOVAL =
256; //设置超时值
wdog_cs = wdog_cs_en(1) | wdog_cs_clk(1) | wdog_cs_int(1) |
          wdog_cs_win(0) | wdog_cs_update(0) 。
while(WDOG_CS[RCS]==0); //等待直到新的配置生效 EnableInterrupts; //启用全局中断
```

*为可重新配置的配置*

```
DisableInterrupts; //禁用全局中断 WDOG_CNT =
0xD928C520; //解锁看门狗
while(WDOG_CS[ULK]==0);  //等待直到寄存器被解锁 WDOG_TOVAL =
256; //设置超时值
wdog_cs = wdog_cs_en(1) | wdog_cs_clk(1) | wdog_cs_int(1) |
          wdog_cs_win(0) | wdog_cs_update(1) 。
while(WDOG_CS[RCS]==0); //等待直到新的配置生效 EnableInterrupts; //启用全局中断
```

## 21.5.4 刷新 看门狗

为了刷新看门狗并将看门狗计数器重置为零，需要一个刷新序列。下面的代码片段显示了一个32位写入的例子。

```
DisableInterrupts; //禁用全局中断 WDOG_CNT =
0xB480A602; //刷新看门狗 EnableInterrupts; //启用全局中
断
```

# 章 22
# 循环冗余检查 (CRC)

## 22.1 芯片特定的CRC 信息

**注意事项**

所有的输入字节都需要进行字节转置。任何额外的转置将受到CRC协议要求的制约。

## 22.2 简介

循环冗余检查（CRC）模块生成16/32位CRC码用于错误检测。

CRC模块提供一个可编程的多项式和其他参数，以实现16位或32位的CRC标准。

16/32位编码是针对每次32位的数据计算的。

### 22.2.1 特点

CRC模块的特点包括。

- 使用16位或32位可编程移位寄存器的硬件CRC发生器电路
- 可编程的初始种子值和多项式
- 对输入数据或输出数据（CRC结果）进行位数或字节数移位的选项。这个选项对于某些CRC标准是必需的。当通过8位访问来访问CRC数据寄存器时，不可能进行顺时针转置操作。在这种情况下，用户的软件必须执行bytewise转置功能。
- 可选择反转CRC的最终结果
- 32位CPU寄存器编程接口

## 22.2.2 块 图

以下是CRC的方框图。

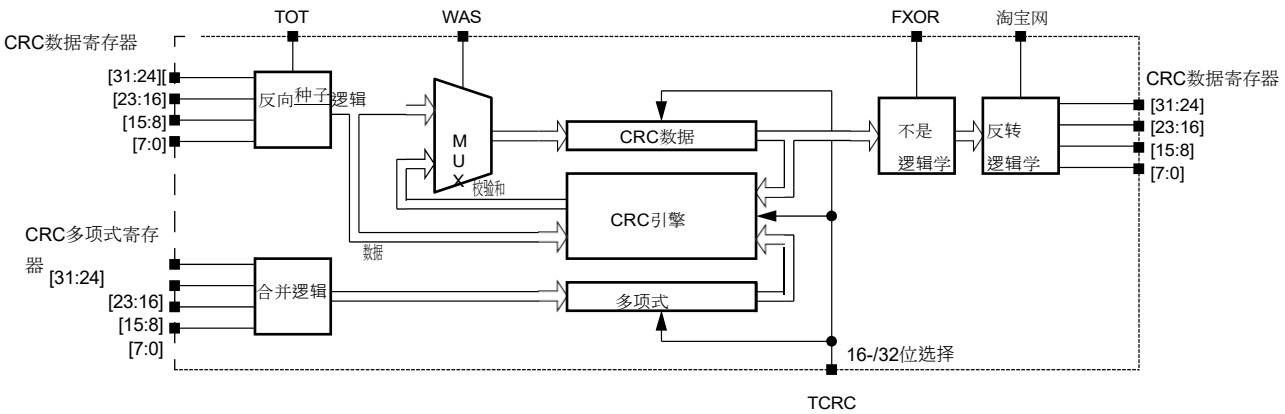



**图22-1.可编程循环冗余检查（CRC）框图**

## 22.2.3 的操作模式

各种MCU模式影响CRC模块的功能。

### 22.2.3.1 运行模式

这是基本的操作模式。

### 22.2.3.2 低功耗模式（停止）。

当MCU进入禁用模块时钟的低功耗模式时，任何正在进行的CRC计算都会停止。在时钟启用后或通过系统复位退出低功耗模式后，它又恢复了。该模块的时钟门控取决于MCU。

# 22.3 内存图和寄存器 描述

### 22.3.1 CRC寄存器 说明

### 22.3.1.1 CRC 内存图

CRC基础地址。4003_2000h

| 偏移 | 注册 | 宽度<br>(单位：比特) | 访问 | 重置值 |
|---|---|---|---|---|
| 0h | CRC数据寄存器（DATA）。 | 32 | RW | FFFF_FFFFh |
| 4h | CRC多项式寄存器（GPOLY）。 | 32 | RW | 0000_1021h |
| 8h | CRC控制寄存器（CTRL）。 | 32 | RW | 0000_0000h |

### 22.3.1.2 CRC数据寄存器（DATA）。

#### 22.3.1.2.1 偏移

| 注册 | 偏移 |
|---|---|
| 数据 | 0h |

#### 22.3.1.2.2 职能

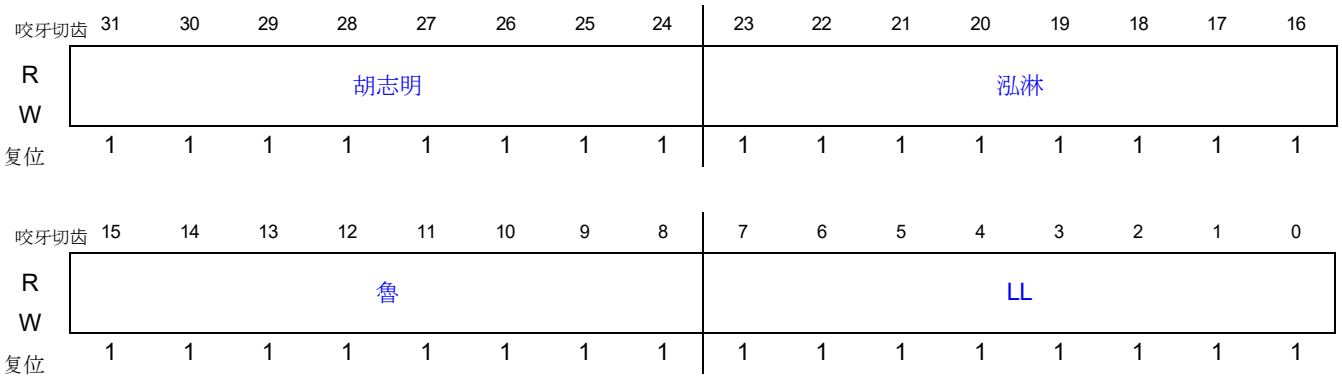
CRC数据寄存器包含了种子、数据和校验和的值。当CTRL[WAS]被设置时，任何对数据寄存器的写入都被认为是种子值。当CTRL[WAS]被清零时，任何对数据寄存器的写入都被看作是一般CRC计算的数据。

在16位CRC模式下，HU和HL字段不用于编程种子值，对这些字段的读取会返回一个不确定的值。在32位CRC模式下，所有字段都用于编程种子值。

在对数据值进行编程时，每次可以写入8位、16位或32位的数据值，前提是所有字节是连续的；数据值的MSB先写入。

所有数据值写入后，可以从这个数据寄存器中读取CRC结果。在16位CRC模式下，CRC结果可在LU和LL字段中获得。在32位CRC模式下，所有字段都包含结果。只要配置了CRC模块，在任何时候读这个寄存器都会返回中间的CRC值。

### 22.3.1.2.3 图示




### 22.3.1.2.4 领域

| 场地 | 职能 |
|---|---|
| 31-24<br>胡志明 | CRC高位字节<br>在16位CRC模式下（CTRL[TCRC]为0），该字段不用于编程种子值。在32位CRC模式下（CTRL[TCRC]为1），当CTRL[WAS]为1时，写入该字段的值是种子值的一部分。<br>1.当CTRL[WAS]为0时，在16位和32位CRC模式下，写入该字段的数据都用于生成CRC校验和。 |
| 23-16<br>泓淋 | CRC 高低字节<br>在16位CRC模式下（CTRL[TCRC]为0），该字段不用于编程种子值。在32位CRC模式下（CTRL[TCRC]为1），当CTRL[WAS]为1时，写入该字段的值是种子值的一部分。<br>1.当CTRL[WAS]为0时，在16位和32位CRC模式下，写入该字段的数据都用于生成CRC校验和。 |
| 15-8<br>鲁 | CRC低位上字节<br>当CTRL[WAS]为1时，写到这个字段的数值是种子值的一部分。当CTRL[WAS]为0时，写入该字段的数据用于生成CRC校验。 |
| 7-0<br>LL | CRC低位字节<br>当CTRL[WAS]为1时，写到这个字段的数值是种子值的一部分。当CTRL[WAS]为0时，写入该字段的数据用于生成CRC校验。 |

## 22.3.1.3 CRC多项式寄存器（GPOLY）。

### 22.3.1.3.1 偏移

| 注册 | 偏移 |
|---|---|

| | |
|---|---|
| 朴槿惠 | 4h |

#### 22.3.1.3.2 职能

该寄存器包含用于CRC计算的多项式的值。HIGH字段包含CRC多项式的前16位，仅在32位CRC模式下使用。在16位CRC模式下，对HIGH字段的写入被忽略。LOW字段包含CRC多项式的低16位，在16位和32位CRC模式中都会使用。

#### 22.3.1.3.3 图示

| 咬牙切齿 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R<br>W | 高 | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 咬牙切齿 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R<br>W | 低 | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 |

#### 22.3.1.3.4 领域

| 场地 | 职能 |
|---|---|
| 31-16<br>高 | 高位多义词半字法<br>在32位CRC模式下可写可读（CTRL[TCRC]为1）。在16位CRC模式下（CTRL[TCRC]为0），该字段不可写。 |
| 15-0<br>低 | 低位多义词半字<br>在32位和16位CRC模式下均可写入和读取。 |

### 22.3.1.4 CRC控制寄存器（CTRL）。

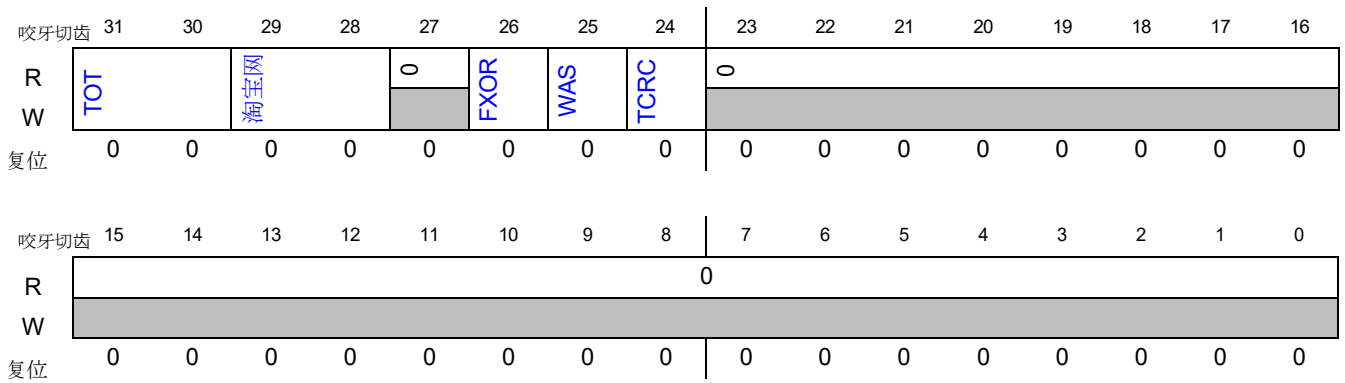
#### 22.3.1.4.1 偏移

| 注册 | 偏移 |
|---|---|
| CTRL | 8h |

### 22.3.1.4.2 职能

该寄存器控制CRC模块的配置和工作。在开始一个新的CRC计算之前，必须设置适当的位。一个新的CRC计算是通过断言CTRL[WAS]，然后将种子写入CRC数据寄存器来初始化的。

### 22.3.1.4.3 图示




### 22.3.1.4.4 领域

| 场地 | 职能 |
|---|---|
| 31-30<br>TOT | 写作的移调类型<br>定义写到CRC数据寄存器中的数据的转置配置。关于可用的转置选项，请参见转置功能的描述。<br>00b - 没有转位。<br>01b - 字节中的位被转置，字节不被转置。10b - 字节中的位和字节都被移位。<br>11b - 只有字节被转置，字节中没有比特被转置。 |
| 29-28<br>淘宝网 | 读取时的移位类型<br>确定从CRC数据寄存器中读取的数值的转置配置。有关可用的转置选项，请参见转置功能的描述。<br>00b - 没有转位。<br>01b - 字节中的位被转置，字节不被转置。10b - 字节中的位和字节都被移位。<br>11b - 只有字节被转置，字节中没有比特被转置。 |
| 27<br>- | 保留 |
| 26<br>FXOR | CRC数据寄存器的补读<br>一些CRC协议要求最终的校验和与0xFFFFFFFF或0xFFFFFFFF进行XOR。对该位的赋值可以使读取的数据得到即时的补充。<br>0b - 读取时没有XOR。 |

| | |
|---|---|
| | 1b - 反转或补充CRC数据寄存器的读取值。 |
| 25<br>WAS | 写入CRC数据寄存器作为种子<br>当断电时，写入CRC数据寄存器的值被认为是一个种子值。当断电时，写到CRC数据寄存器的值被作为CRC计算的数据。 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | 0b - 对CRC数据寄存器的写入是数据值。1b - 对CRC数据寄存器的写入是种子值。 |
| 24<br>TCRC | TCRC<br>CRC协议的宽度。<br>0b - 16位CRC协议。1b - 32位CRC协议。 |
| 23-0<br>- | 保留 |

## 22.4 功能 描述

### 22.4.1 CRC 初始化/重新初始化

为了实现CRC计算，用户必须在相应的寄存器中对CRC_CTRL[WAS]、CRC_GPOLY、转置和CRC结果反转的必要参数进行编程。干预CRC_CTRL[WAS]可以将种子值编入CRC_DATA寄存器。

在完成CRC计算后，通过重新确认CRC_CTRL[WAS]和编程一个新的或以前使用过的种子值，可以重新初始化该模块以进行新的CRC计算。所有其他参数必须在编程种子值和后续数据值之前设置。

### 22.4.2 CRC 计算

在16位和32位CRC模式下，只要所有字节是连续的，数据值可以一次编程为8位、16位或32位。不连续的字节会导致CRC计算的不正确。

#### 22.4.2.1 16位CRC

要计算一个16位的CRC。

1. 清除CRC_CTRL[TCRC]以启用16位CRC模式。
2. 根据CRC计算的需要，在CTRL寄存器中对转置和补码选项进行编程。详见转置功能和CRC结果补充。

3. 向CRC_GPOLY[LOW]字段写入一个16位多项式。CRC_GP OLY[HIGH]字段在16位CRC模式下不能使用。
4. 设置CRC_CTRL[WAS]来编程种子值。
5. 向CRC_DATA[LU:LL]写入一个16位种子。CRC_DATA[HU:HL]不使用。
6. 清除CRC_CTRL[WAS]，开始写入数据值。
7. 将数据值写入CRC_DATA[HU:HL:LU:LL]中。对每一个写入的数据值进行CRC计算，并将中间的CRC结果存回CRC_DATA[LU:LL]中。
8. 当所有的值都被写入后，从CRC_DATA[LU:LL]中读取最终的CRC结果。

在读或写数值的时候，转置和补码操作是即时进行的。详见转置功能和CRC结果补码。

#### 22.4.2.2 32位CRC

要计算一个32位的CRC。

1. 设置CRC_CTRL[TCRC]以启用32位CRC模式。
2. 根据CRC计算的需要，在CTRL寄存器中对转置和补码选项进行编程。详见转置功能和CRC结果补充。
3. 向CRC_GPOLY[HIGH:LOW]写入一个32位多项式。
4. 设置CRC_CTRL[WAS]来编程种子值。
5. 向CRC_DATA[HU:HL:LU:LL]写入一个32位的种子。
6. 清除CRC_CTRL[WAS]，开始写入数据值。
7. 将数据值写入CRC_DATA[HU:HL:LU:LL]中。对每一个写入的数据值进行CRC计算，并将中间的CRC结果存回CRC_DATA[HU:HL:LU:LL]中。
8. 当所有的值都写完后，从CRC_DATA[HU:HL:LU:LL]读取最终的CRC结果。CRC的计算是逐次进行的，完成一次CRC计算需要两个时钟。

在读或写数值的时候，转置和补码操作是即时进行的。详见转置功能和CRC结果补码。

### 22.4.3 转位 功能

默认情况下，转置功能是不启用的。然而，一些CRC标准要求对输入数据和/或最终校验和进行转置。用户软件可以根据CRC标准的要求，选择单独配置每个转置操作。数据在读或写的过程中被转置。

一些协议使用小字节格式的数据流来计算CRC。在这种情况下，转置功能可以有效地翻转这些位。这种转置选项是CRC模块所支持的类型之一。

### 22.4.3.1 转置的类型

CRC模块提供了几种类型的转置功能来翻转比特和/或字节，用于写入输入数据和读取CRC结果，根据正在使用的CRC计算，分别使用CTRL[TOT]或CTRL[TOTR]字段。

以下类型的转置功能可用于写入和读出CRC数据寄存器。

1. CTRL[TOT]或CTRL[TOTR]为00。

   没有发生移位。

2. CTRL[TOT]或CTRL[TOTR]为01

   字节中的位被转置，而字节则不被转置。

   reg[31:0]变成{reg[24:31], reg[16:23], reg[8:15], reg[0:7]}。

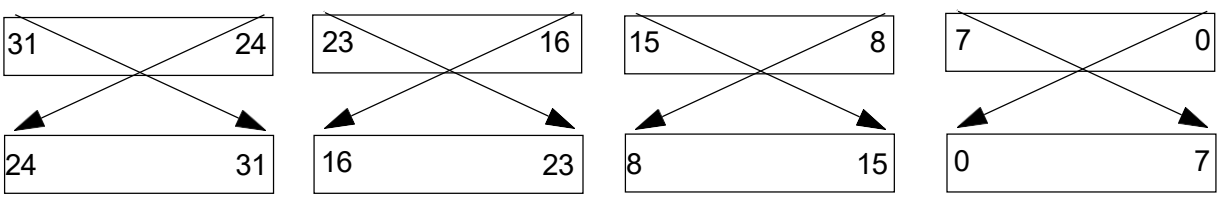



**图22-2.转位类型01**

3. CTRL[TOT]或CTRL[TOTR]为10。

   字节中的比特和字节都是转置的。

   reg[31:0]变成={reg[0:7], reg[8:15],reg[16:23], reg[24:31]}。

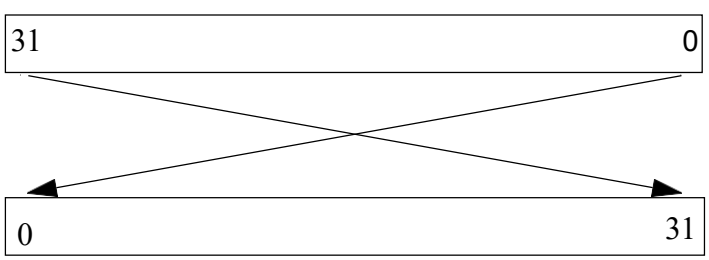



**图22-3.转置类型10**

4. CTRL[TOT]或CTRL[TOTR]为11。

字节是转置的，但比特是不转置的。

reg[31:0]变成{reg[7:0], reg[15:8], reg[23:16], reg[31:24]}。

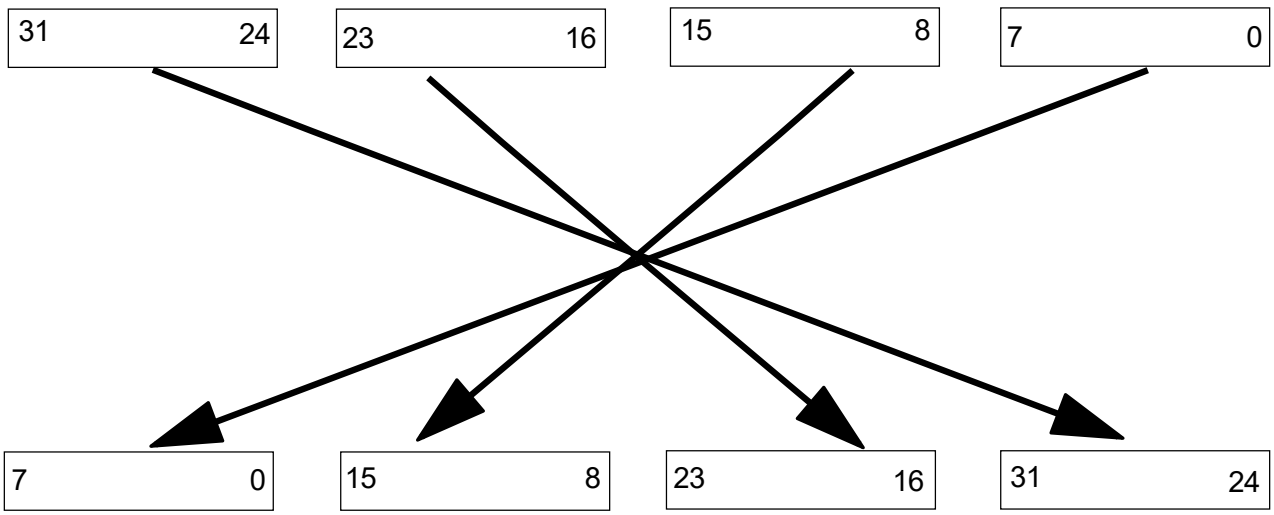



**图22-4.转置类型11**

**注意事项**

对于8位和16位的CRC数据寄存器的写访问，数据在未使用的一个或多个字节上用0进行转置（将32位作为一个整体），但CRC只在有效的字节上计算。当读取CRC数据寄存器中的
16位CRC结果并使用转置选项10和11，转置后的结果值停留在CRC[HU:HL]字段中。用户软件在读取16位CRC结果时必须考虑到这种情况，所以最好是读取32位。

## 22.4.4 CRC结果 补码

当CTRL[FXOR]被设置时，校验值被补充。每次读取CRC数据寄存器时，CRC结果补码功能会输出存储在CRC数据寄存器中的校验和值的补码。当CTRL[FXOR]被清零时，读取CRC数据寄存器访问原始校验值。

# 第23章 重置和 启动

## 23.1 简介

该表显示了所支持的复位源。

**表23-1.复位源**

| 重置来源 | 描述 |
| --- | --- |
| POR复位 | • 通电复位（POR） |
| 系统复位 | • 外部引脚复位（PIN）<br>• 低电压检测（LVD）。<br>• 看门狗复位<br>• 时钟丢失（LOC）复位<br>• 失锁（LOL）复位<br>• 停止模式确认错误（SACKERR）。<br>• 软件复位（SW）<br>• 锁定复位（LOCKUP）。<br>• MDM DAP系统重置 |
| 调试复位 | • JTAG复位 |

## 23.2 复位

本节讨论了基本的复位机制和来源。一些引起复位的外设模块可以被配置为引起中断来代替。更多信息请参见各个模块的章节。

每个复位源在复位控制模块（RCM）系统复位状态（RCM_SRS）寄存器中都有一个相关字段。除了即时的系统复位，RCM还支持可配置的延迟系统复位，同时产生一个中断。这为软件提供了一个执行优雅关机的选项。

MCU在RUN模式下退出复位，CPU正在执行代码。更多细节见启动。

职能描述

### 23.2.1 开机复位 (POR)

当电源最初应用于MCU时，或者当电源电压低于POR检测电压($V_{POR}$)时，POR电路会引起一个POR复位条件。

随着电源电压的上升，LVD电路使MCU处于复位状态，直到电源上升到LVD低电压检测阈值（$V_{LVD}$）以上。在POR之后，MCU向RCM_SRS寄存器的POR和LVD字段写1。

在POR之外，RCM_SRS[POR]和RCM_SRS[LVD]都被MCU配置为1。

在正常运行时，如果设备电源低于VPOR，MCU会向RCM_SRS[POR]和RCM_SRS[LVD]写1。在正常工作时，如果器件电源低于VLVD（前提是LVD被启用以产生复位）或VLVR，但高于VPOR，MCU会向RCM_SRS[LVD]写1。

### 23.2.2 系统复位 来源

重置MCU提供了一种从一组已知的初始条件开始处理的方法。系统复位开始时，片上稳压器处于完全调节状态，系统时钟由内部参考产生。当处理器退出复位时，它

。

- 从向量表的偏移量0读取起始堆栈指针（SP）（SP_main）。
- 从矢量表偏移量4中读取起始程序计数器（PC）。
- 将0xFFFF_FFFF写入链接寄存器（LR）。

片上外围模块被禁用，非模拟I/O引脚最初被配置为禁用。具有模拟功能的引脚在复位后被配置为其模拟功能。

在复位期间和复位之后，JTAG引脚的相关输入引脚被配置为。

- TDI作为上拉（PU）。
- TCK作为下拉线（PD）。
- TMS作为上拉

和相关的输出引脚配置为：。

- 没有下拉或上拉的TDO

#### 23.2.2.1 外部引脚复位

RESET_B是一个共享引脚（参见《参考手册》所附的IO信号描述输入复用表）。用户必须在使用该引脚作为RESET_B之前配置相应的PORT_PCR寄存器。在任何复位事件中，MCU会将相应的焊盘配置为RESET_B焊盘。对RESET_B的赋值将MCU从任何模式中唤醒。在引脚复位期间，MCU向RCM_SRS[PIN]写1。

##### 23.2.2.1.1 RESET_B引脚滤波器

RESET_B引脚滤波器支持从128kHz的LPO时钟和总线时钟中进行过滤。RCM_RPC[RSTFLTSS]、RCM_RPC[RSTFLTSRW]和RCM_RPC[RSTFLTSEL]控制这个功能。滤波器在芯片POR时被异步复位。每个滤波器的复位值假定RESET_B引脚被否定。

对于LPO时钟仍然有效的所有停止模式，唯一的过滤选项是LPO时钟滤波器。滤波逻辑要么切换到旁路操作，要么有持续的滤波操作，这取决于RCM_RPC寄存器中所选择的滤波模式。

LPO滤波器有一个固定的滤波值。2个LPO128K_CLK周期以内的脉冲将总是被过滤掉。长于3个LPO128K_CLK周期的脉冲将总是被通过并产生一个RESET。在2到3个LPO128K_CLK周期之间的脉冲可能会产生复位，也可能不会产生复位，这取决于复位与时钟沿的一致性。

#### 23.2.2.2 低电压检测（LVD）。

该芯片包括一个管理低电压条件的系统，以便在电源电压变化期间保护存储器内容和控制MCU系统状态。该系统由一个POR电路和一个LVD电路组成。LVD系统在HSRUN和正常运行时总是被启用。当进入VLPx模式时，LVD系统被禁用。

LVD系统可以被配置为在检测到低电压条件时通过向PMC_LVDSC1[LVDRE]写1来产生复位。在LVD复位发生后，LVD系统使MCU处于复位状态，直到电源电压上升到低电压检测阈值以上。在LVD复位或POR之后，MCU写1到RCM_SRS[LVD]位被设置。

除了LVD操作，该器件还支持LVR（低电压复位）操作。如果电源电压低于复位跳闸点（VLVR），将产生一个系统复位。LVR系统在所有模式下都被启用。LVDRE仅对LVD操作有影响。

在POR之外，RCM_SRS[POR]和RCM_SRS[LVD]都被MCU配置为1。

在正常运行时，如果设备电源低于VPOR，MCU会向RCM_SRS[POR]和RCM_SRS[LVD]写1。在正常工作时，如果器件电源低于VLVD（前提是LVD被启用以产生复位）或VLVR，但高于VPOR，MCU会向RCM_SRS[LVD]写1。

#### 23.2.2.3 看门狗复位

看门狗定时器（WDOG）通过期待来自软件的定期通信来监控系统的运行。这种通信通常被称为看门狗的服务或刷新。如果这种定期刷新没有发生，看门狗会发出系统复位。看门狗复位会导致MCU向RCM_SRS[WDOG]写1。

#### 23.2.2.4 时钟丢失（LOC）复位

当SOSC时钟监控被启用（通过向SOSCCM写1）并被配置为出错时产生复位（通过向SOSCCMRE写1）时，LOC复位被启用。如果错误位（SOSCERR）被写为1，并且LOC复位被启用（如上所述），MCU会复位。MCU将1写到LOC字段以指示这个复位源。

**注意事项**

为了防止时钟复位事件的意外丢失，在进入任何低功率模式（包括VLPR）之前，应禁用所有的时钟监视器。

#### 23.2.2.5 失锁（LOL）复位

当PLL时钟监控被启用（通过向SCG_SPLLCSR[SPLLCM]写1）并被配置为出错时产生复位（通过向SCG_SPLLCSR[SPLLCMRE]写1），LOL复位被启用。如果错误位（SCG_SPLLCSR[SPLLERR]）被写为1，并且LOL复位被启用（如上所述），MCU会复位。MCU写1到RCM_SRS[LOL]字段被设置，以表示这个复位源。

**注意事项**

如果芯片处于任何停止模式，这个复位源不会引起复位。

如果SPLL的参考发生故障，SPLL也会标志着LOL。因此，在使用SPLL时，任何参考时钟源的故障，即SOSC可能导致LOC或LOL事件。

#### 23.2.2.6 停止模式确认错误（SACKERR）。

如果内核试图进入停止模式，但在LPO128K_CLK的（2^16+1（相当于0.5s））周期内，不是所有的模块都确认了停止模式，就会产生停止模式确认错误（SACKERR）复位。

如果发生错误情况，一个模块可能不承认进入停止模式。该错误可能是由于模块的外部时钟输入失败造成的。

RCM_SRS[SACKERR]位被写为1，表示这个复位源。

#### 23.2.2.7 软件复位（SW）

NVIC应用中断和复位控制寄存器中的SYSESETREQ位可以被写为1，以强制进行软件复位。(有关寄存器字段的完整描述，特别是VECTKEY字段要求，请参见ARM的NVIC文档）。向 SYSRESETREQ 写 1
会产生一个软件复位请求。这种复位强制所有主要部件的系统复位，除了调试模块。软件复位导致MCU向RCM_SRS[SW]字段写1。

#### 23.2.2.8 锁定复位（LOCKUP）。

LOCKUP对严重错误的内核软件给出了直接的指示。这是由于处理器内置的系统状态保护硬件被激活后，出现了不可恢复的异常，导致内核被锁定。

LOCKUP条件会导致系统复位，也会导致MCU向RCM_SRS[LOCKUP]字段写1。

#### 23.2.2.9 MDM-AP系统重置请求

向MDM-AP控制寄存器中的系统复位请求字段写1，启动系统复位。这是通过JTAG/SWD接口进行复位的主要方法。系统复位一直保持到这个字段被写成0。

在MDM-AP控制寄存器中的核心保持复位位上写1，当芯片的其他部分从系统复位中出来时，核心处于复位状态。

### 23.2.3 MCU 复位

各种复位是由MCU产生的，用于复位不同的模块。

#### 23.2.3.1 仅限POR

POR

Only复位只对POR复位源发出信号。它重置了PMC寄存器。只有POR复位也会导致所有其他的复位类型发生。

#### 23.2.3.2 芯片POR

芯片POR在POR、LVR或LVD上发出指令。

芯片POR也会导致芯片复位（包括早期芯片复位）的发生。

#### 23.2.3.3 早期芯片复位

早期的芯片复位是对所有的复位源发出信号。它只对闪存模块进行复位。它在闪存初始化开始前（早于芯片复位的解除）就解除了信号。

#### 23.2.3.4 芯片复位

芯片复位在所有的复位源上都会发出信号，只有在闪存初始化完成并且RESET_B引脚也被否定后才会发出信号。

#### 23.2.3.5 磁芯复位

在芯片复位的一个周期后，内核就会脱离复位状态。当芯片的其他部分从系统复位中出来时，可以使用MDM-
AP控制寄存器中的核心保持复位位来保持核心的复位。

### 23.2.4 复位 引脚

对于所有的复位源，RESET_B引脚被MCU驱动为低电平，至少持续128个总线时钟周期，直到闪存初始化完成。

在闪存初始化完成后，RESET_B引脚被释放，内部芯片复位解除。保持RESET_B引脚的外部断言会延迟内部芯片复位的否定。

### 23.2.5 调试 复位

下面的章节详细介绍了调试复位的情况。

#### 23.2.5.1 JTAG复位

当选择某些IR代码时，JTAG模块产生一个系统复位。当EXTEST、HIGHZ和CLAMP指令处于激活状态时，这个功能复位被断言。当选择任何其他IR代码时，来自JTAG模块的复位源被释放。JTAG复位导致MCU向RCM_SRS[JTAG]字段写1。

#### 23.2.5.2 重置调试子系统

使用SWJ-DP
CTRL/STAT寄存器中的CDBGRSTREQ字段来重置调试模块。但是CDBGRSTREQ字段并不能重置所有与调试有关的寄存器。

CDBGRSTREQ只重置下列模块中与调试有关的寄存器。

- SWJ-DP
- AHB-AP
- TPIU
- MDM-AP（MDM控制和状态寄存器）。

CDBGRSTREQ并不重置这些模块中与调试有关的寄存器。

- CM4核心（核心调试寄存器：DHCSR、DCRSR、DCRDR、DEMCR）。
- FPB
- DWT
- ITM
- NVIC
- 横梁式总线开关
- 私人外围总线

## 23.3 靴子

本节描述了启动顺序，包括来源和选项。

### 23.3.1 启动 源

该MCU支持从内部闪存冷启动。

当MCU从内部闪存启动时，复位向量位于地址0x0（初始SP_main）和0x4（初始PC）。

该MCU还通过将异常向量表重新定位到RAM来支持从RAM启动。这是通过NVIC模块中一个可编程的向量表偏移寄存器（VTOR）实现的。

该MCU支持通过以下方式下载串行代码

- CAN
- LIN
- SPI等。

应用程序开发人员可以编写他们自己的基于闪存的引导装载程序，或者改编在http://www.nxp.com。然后，这些启动加载器被用来配置和使用通信协议来进行大规模的串行下载。例如，CAN模块可以用来将数据传输到它自己的消息缓冲区，然后DMA模块可以用来将数据移入主系统RAM。一旦下载完成，可以使用内核将新下载的代码转移到闪存区。

### 23.3.2 FOPT启动 选项

闪存模块中的FOPT寄存器允许用户在启动时定制MCU的操作。该寄存器包含只读位，这些位从NVM的闪存配置字段的选项字节中加载。FTFC_FOPT寄存器中所有数值的默认设置是逻辑1，因为它是从驻留在闪存中的选项字节复制过来的，在闪存擦除状态下，所有位都是逻辑1。
要配置替代的设置，请在NVM选项字节中对适当的位进行编程。新的设置在随后的POR和任何系统复位时生效。关于选项字节编程的更多细节，请参见闪存章节。

在复位时，MCU使用FOPT来配置MCU，如该表所示。

**表23-2.FTFC_FOPT定义**

| 位号 | 场地 | 价值 | 定义 |
|---|---|---|---|
| 7-6 | 保留 | 为未来的扩展而保留 | |
| 5 | 保留 | 保留 | |
| 4 | 保留 | 为未来的扩展而保留 | |
| 3 | RESET_PIN_CFG | 启用/禁用对RESET引脚的控制。 | |
| | | 0 | 在POR之后，RESET_B引脚被禁用，不能被启用为复位。<br>功能。当选择这个选项时，可能会有一个短暂的争论期<br>在一个POR斜坡期间，MCU在建立一个新系统之前将引脚驱动为低电平。<br>该选项的设置和释放引脚上的复位功能。<br>该位在系统复位和低功耗模式下被保留下来。当<br>RESET_B引脚的功能被禁用，它不能被用作低功耗的来源。<br>模式唤醒。 |
| | | 1 | 该端口被配置为启用上拉，启用无源滤波器。 |
| 2 | NMI_PIN_CFG | 启用/禁用对NMI功能的控制。 | |
| | | 0 | NMI中断总是被阻止的。相关引脚继续默认为NMI_b引脚控制，并启用内部上拉。<br>当NMI_b引脚功能被禁用时，它不能被用作低功耗模式唤醒的来源。<br>如果不需要NMI功能，无论是中断还是唤醒源，建议通过向NMI_PIN_CFG写0来禁用NMI功能。 |
| | | 1 | NMI_b引脚/中断重置默认为启用。 |
| 1 | 保留 | 为未来的扩展而保留。 | |
| 0 | 笔记本电脑 | 控制IRC 48 Mhz的时钟分频器的复位值，以供给Core和平台时钟。较大的分频值选择在POR和复位排序期间以及复位退出后产生较低的平均功耗。 | |
| | | 0 | 核心和系统时钟分频器（OUTDIV1）为0x1（除以2）。 |
| | | 1 | 核心和系统时钟分频器（OUTDIV1）是0x0（除以1）。 |

## 23.3.3 启动序

上电时，片上稳压器将系统保持在POR状态，直到输入电源高于POR阈值。系统继续保持这种静态状态，直到内部调节的电源达到由LVD确定的安全工作电压。然后，模式控制器的复位逻辑控制这个序列，退出复位。

1. 系统复位被保持在内部逻辑上，RESET_B引脚被驱动出低电平，SCG在其默认的时钟模式下被启用。

2. 所需的时钟被启用（核心时钟、系统时钟、闪存时钟，以及任何没有将时钟门控制重置为禁用的总线时钟）。

3. 内部逻辑上的系统复位继续保持，但闪存控制器从复位中释放出来并开始初始化操作，同时复位控制逻辑继续驱动RESET_B引脚为低电平。
4. 在复位顺序的早期，NVM选项字节被读取并存储到闪存模块的FOPT寄存器。如果LPBOOT被编程为一个备用的时钟分频器复位值，系统/内核时钟将被切换到一个较慢的时钟速度。
5. 当闪存初始化完成后，RESET_B引脚被释放。如果RESET_B继续被断言（表明RESET_B引脚的上升时间较慢或外部驱动为低电平），系统将继续保持在复位状态。一旦RESET_B引脚被检测到高电平，核心时钟将被启用，系统将从复位中释放。
6. 当系统退出复位时，处理器设置了堆栈、程序计数器（PC）和链接寄存器（LR）。处理器从矢量表的偏移量0读取起始SP（SP_main），内核从矢量表的偏移量4读取起始PC。LR被写到0xFFFF_FFFF。
7. 如果FlexNVM被启用，闪存控制器将继续恢复FlexNVM的数据。这个数据在复位后不能立即使用，在闪存控制器完成这个初始化步骤之前，系统不应该访问这个数据，如EEERDY标志所指示的。

随后的系统复位也遵循这个复位流程。

# 第二十四章
# 复位控制模块（RCM）

## 24.1 特定芯片的RCM信息

该器件不支持高电压检测复位。SACKERR的时间是0.5秒。

在该器件上，LVD（低电压检测）和LVR（低电压复位）事件触发LVD复位，从而触发芯片POR。芯片POR是由POR、LVR或LVD事件触发的。PMC_LVDSC1[LVDRE]仅用于LVD的门控，对LVR的产生没有影响。在禁用PMC_SC1[LVDRE]的情况下，LVR仍然可以产生对系统的复位。参见低电压检测（LVD）系统。

该设备不支持等待模式。关于可用的电源模式的详细信息，请参见模块在可用的低功率模式下的操作。

LPO128K_CLK被用作复位引脚滤波器的低功耗时钟。

## 24.2 在STOP1/2模式下的复位引脚过滤操作

在STOP1/2模式下，复位引脚的过滤情况如下。

- STOP1：RUN模式和STOP模式的滤波器都可以工作。脉冲持续时间较短的滤波器占主导地位。
- STOP2。RUN模式过滤器运行。

## 24.3 简介

这里的信息描述了复位控制模块（RCM）的寄存器。RCM实现了芯片的许多复位功能。更多信息请参见芯片的复位章节。

请参阅AN4503。Kinetis和ColdFire+ MCU的电源管理，以了解关于使用RCM的进一步详情。

## 24.4 复位内存图和寄存器描述

RCM的内存图/寄存器定义可以在这里找到。

复位控制模块（RCM）寄存器提供复位状态信息和复位滤波器控制。

**注意事项**

RCM寄存器只能在监督员模式下被写入。在用户模式下的写访问被阻止，并将导致总线错误。

**RCM内存图**

| 地址偏移（十六进制） | 注册名称 | 宽度（单位：比特） | 访问 | 重置值 | 节/页 |
|---|---|---|---|---|---|
| 0 | 版本标识寄存器（RCM_VERID）。 | 32 | R | 0300_0003h | 24.4.1/502 |
| 4 | 参数寄存器（RCM_PARAM）。 | 32 | R | 见本节 | 24.4.2/503 |
| 8 | 系统复位状态寄存器（RCM_SRS）。 | 32 | R | 0000_0082h | 24.4.3/505 |
| C | 复位引脚控制寄存器（RCM_RPC）。 | 32 | R/W | 0000_0000h | 24.4.4/508 |
| 18 | 粘性系统复位状态寄存器（RCM_SSRS）。 | 32 | R/W | 0000_0082h | 24.4.5/510 |
| 1C | 系统复位中断使能寄存器（RCM_SRIE）。 | 32 | R/W | 0000_0000h | 24.4.6/512 |

## 24.4.1 版本标识寄存器（RCM_VERID）。

地址。0h base + 0h offset = 0h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 主要的 | | | | | | | | 小学 | | | | | | | | 特写 | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |

**RCM_VERID字段描述**

| 场地 | 描述 |
|---|---|
| 31-24<br>主要的 | 主要版本号<br><br>这个只读字段返回该规范的主要版本号。 |
| 23-16<br>小学 | 次要版本号<br><br>这个只读字段返回规范的次要版本号。 |

| | |
|---|---|
| 特写 | 特点规格号 |

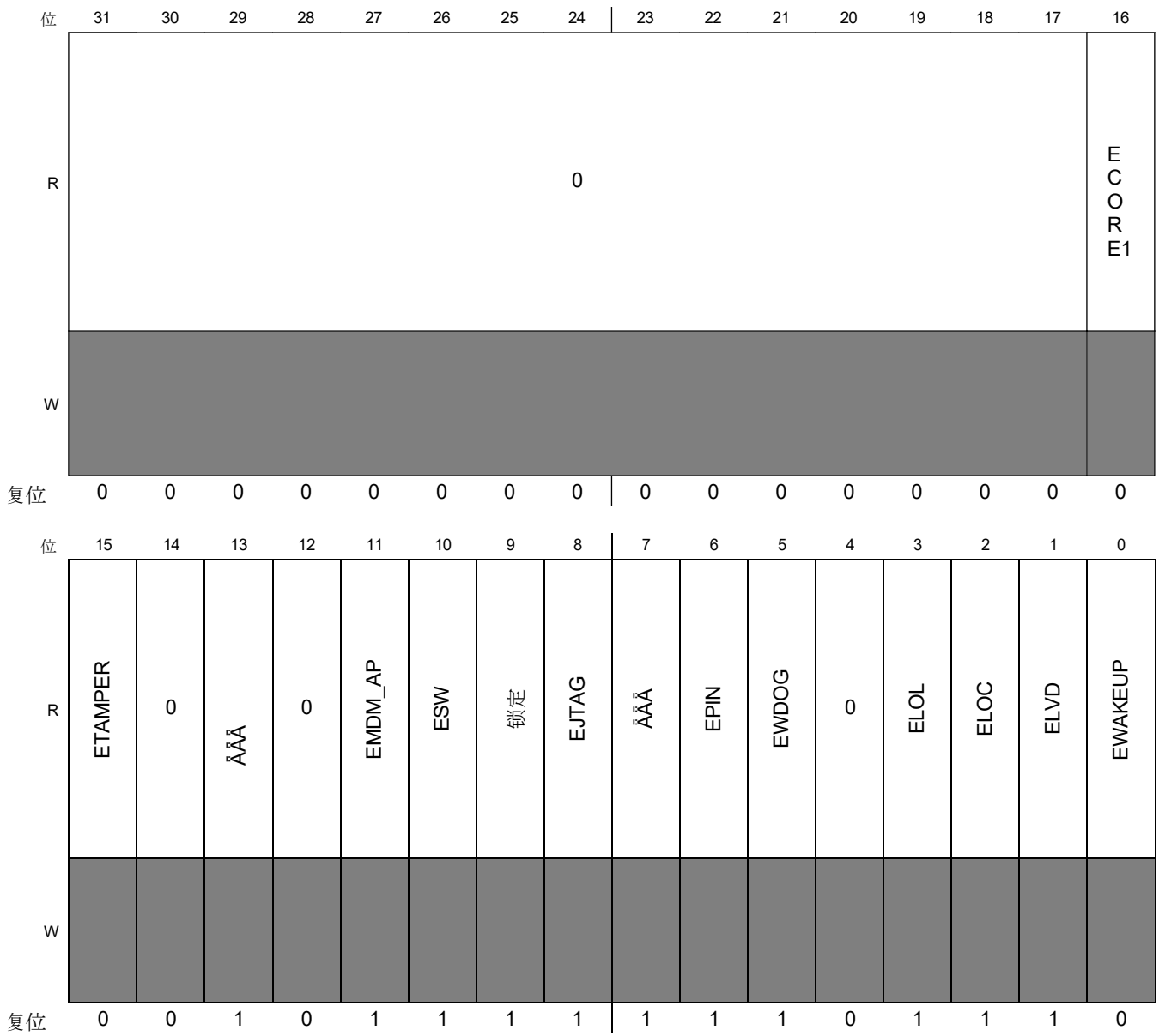
*表继续在下一页...*

RCM_VERID字段描述（续）

| 场地 | 描述 |
|---|---|
| | 这个只读字段返回特征集编号。<br><br>0x0003标准特征集。 |

## 24.4.2 参数寄存器（RCM_PARAM）。

地址。0h基数+4h偏移量=4h



## RCM_PARAM字段描述

| 场地 | 描述 |
|---|---|
| 31-17<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 16<br>ECORE1 | 存在SRS[CORE1]的状态指示功能<br>这个静态位说明该功能在设备上是否可用。<br>0该功能不可用。<br>1该功能是可用的。 |
| 15<br>ETAMPER | 存在SRS[TAMPER]状态指示功能<br>这个静态位说明该功能在设备上是否可用。<br>0该功能不可用。<br>1该功能是可用的。 |
| 14<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 13<br>ÃÃÃ | 存在SRS[SACKERR]状态指示功能<br>这个静态位说明该功能在设备上是否可用。<br>0该功能不可用。<br>1该功能是可用的。 |
| 12<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 11<br>EMDM_AP | 存在SRS[MDM_AP]的状态指示功能<br>这个静态位说明该功能在设备上是否可用。<br>0该功能不可用。<br>1该功能是可用的。 |
| 10<br>ESW | 存在SRS[SW]状态指示功能<br>这个静态位说明该功能在设备上是否可用。<br>0该功能不可用。<br>1该功能是可用的。 |
| 9<br>锁定 | 存在SRS[LOCKUP]状态指示功能<br>这个静态位说明该功能在设备上是否可用。<br>0该功能不可用。<br>1该功能是可用的。 |
| 8<br>EJTAG | 存在SRS[JTAG]状态指示功能<br>这个静态位说明该功能在设备上是否可用。<br>0该功能不可用。<br>1该功能是可用的。 |

| | |
|---|---|
| 7<br>ÃÃÃ | 存在SRS[POR]状态指示功能<br><br>这个静态位说明该功能在设备上是否可用。 |

*表继续在下一页...*

**RCM_PARAM字段描述（续）**

| 场地 | 描述 |
|---|---|
| | 0 该功能不可用。<br>1 该功能是可用的。 |
| 6<br>EPIN | 存在SRS[PIN]状态指示功能<br>这个静态位说明该功能在设备上是否可用。<br>0该功能不可用。<br>1该功能是可用的。 |
| 5<br>EWDOG | 存在SRS[WDOG]状态指示功能<br>这个静态位说明该功能在设备上是否可用。<br>0该功能不可用。<br>1该功能是可用的。 |
| 4<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 3<br>ELOL | 存在SRS[LOL]状态指示功能<br>这个静态位说明该功能在设备上是否可用。<br>0该功能不可用。<br>1该功能是可用的。 |
| 2<br>ELOC | 存在SRS[LOC]状态指示功能<br>这个静态位说明该功能在设备上是否可用。<br>0该功能不可用。<br>1该功能是可用的。 |
| 1<br>ELVD | 存在SRS[LVD]状态指示功能<br>这个静态位说明该功能在设备上是否可用。<br>0该功能不可用。<br>1该功能是可用的。 |
| 0<br>EWAKEUP | 存在SRS[WAKEUP]状态指示功能<br>这个静态位说明该功能在设备上是否可用。<br>0该功能不可用。<br>1该功能是可用的。 |

## 24.4.3 系统复位状态寄存器（RCM_SRS）。

这个寄存器包括只读的状态标志，以指示最近一次复位的来源。注意，如果多个复位事件同时发生，可以设置多个标志。这些位的复位状态取决于导致MCU复位

的原因。

## 注意事项

该寄存器的复位值取决于复位源。

- POR（包括LVD） - 0x82
- LVD（无POR） - 0x02
- 其他复位 -
  如果一个位的相应复位源引起了复位，则该位被设置。

地址。0h基数+8h偏移量=8h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | 0 |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 位 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | ÃÃÃ | 0 | MDM_AP | 紫外线 | 锁定 | JTAG | POR | 密码 | 纬度 | 0 | 大笑 | 辽宁省 | LVD | 0 |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |

### RCM_SRS字段描述

| 场地 | 描述 |
|---|---|
| 31-17<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 15<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 14<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |

| | |
|---|---|
| 13<br>ÃÃÃ | 停止确认错误 |

*表继续在下一页...*

## RCM_SRS字段描述（续）

| 场地 | 描述 |
|---|---|
| | 表示在试图进入停止模式后，由于一个或多个外设未能在大约一秒钟内确认进入停止模式而导致复位。<br><br>0 复位不是因为外设未能确认进入停止模式的尝试而引起的<br>1 由于外设未能确认进入停止模式的尝试而导致复位 |
| 12<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 11<br>MDM_AP | MDM-AP系统复位请求<br><br>表示由主机调试器系统设置MDM-AP控制寄存器中的系统复位请求位引起的复位。<br><br>0 复位不是由主机调试器系统设置的系统复位请求位引起的<br>1 复位是由主机调试器系统设置的系统复位请求位引起的 |
| 10<br>紫外线 | 软件<br><br>表示通过软件设置ARM内核中应用中断和复位控制寄存器中的SYSESETREQ位，引起了复位。<br><br>0 复位不是由软件设置 SYSRESETREQ 位引起的<br>1 由软件设置 SYSRESETREQ 位引起的复位 |
| 9<br>锁定 | 核心锁定<br><br>表示由于ARM核指示LOCKUP事件而引起的复位。<br><br>0 复位不是由核心LOCKUP事件引起的<br>1 由核心LOCKUP事件引起的复位 |
| 8<br>JTAG | JTAG产生的复位<br><br>表示由于JTAG选择了某些IR代码而造成了复位。EXTEST, HIGHZ, 和 CLAMP。<br><br>0 复位不是由JTAG引起的<br>1 由JTAG引起的复位 |
| 7<br>POR | 开机复位<br><br>表明上电检测逻辑已经引起了复位。因为当时内部电源电压正在上升，所以低电压复位（LVD）状态位也被设置，表示复位发生在内部电源低于LVD阈值的时候。<br><br>0 复位不是由POR引起的<br>1 由POR引起的复位 |
| 6<br>密码 | 外部复位引脚<br><br>表示外部RESET(RESET_b)引脚上的有效低电平已经引起了复位。<br><br>0 复位不是由外部复位引脚引起的<br>1 由外部复位引脚引起的复位 |

| | |
|---|---|
| 5<br>纬度 | 看门狗<br><br>表示复位是由看门狗定时器超时引起的。这个复位源可以通过禁用看门狗来阻断。 |

*表继续在下一页...*

**RCM_SRS字段描述（续）**

| 场地 | 描述 |
|---|---|
| | 0 复位不是由看门狗超时引起的<br>1 由看门狗超时引起的复位 |
| 4<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 3<br>大笑 | 失锁复位<br><br>表示由于SCG PLL/FLL失去锁定而导致复位。<br><br>0 复位不是由PLL/FLL的失锁引起的<br>1 因PLL/FLL失去锁定而引起的复位 |
| 2<br>辽宁省 | 失时钟复位<br><br>表明外部时钟丢失导致了复位。必须启用SCG的SOSC时钟监视器才能检测到时钟的丢失。关于启用时钟监视器的信息，请参考详细的SCG描述。<br><br>0 不是由外部时钟丢失引起的复位。<br>1 由外部时钟丢失引起的复位。 |
| 1<br>LVD | 低电压检测复位或高电压检测复位<br><br>如果PMC_LVDSC1[LVDRE]被设置，且电源下降到LVD跳闸电压以下，则发生LVD复位。如果PMC_HVDSC1[HVDRE]被设置，并且电源上升到HVD跳闸电压以上，则发生HVD复位。<br>这个字段也是由POR设置的。<br><br>0 非由LVD跳闸、HVD跳闸或POR引起的复位<br>1 由LVD跳闸、HVD跳闸或POR引起的复位 |
| 0<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |

## 24.4.4 复位引脚控制寄存器（RCM_RPC）。

**注意事项**

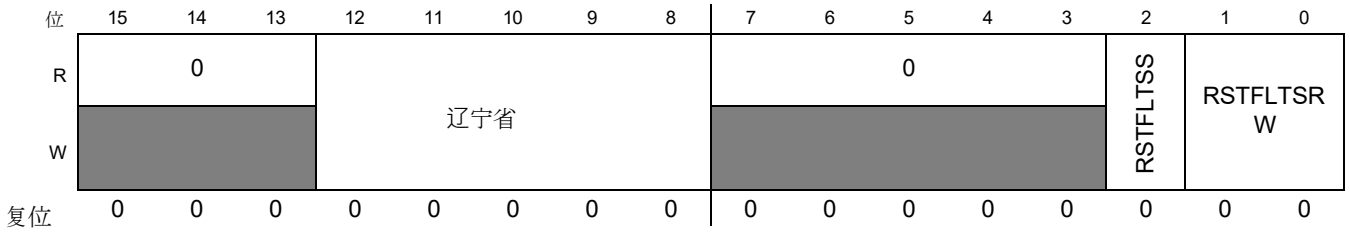
这个寄存器只在芯片POR时被复位，不受其他复位类型的影响。

**注意事项**

总线时钟滤波器在禁用时或进入停止模式时被重置。LPO滤波器在禁用时被重置。

地址。0h基数+Ch偏移量=Ch

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |




## RCM_RPC字段描述

| 场地 | 描述 |
|---|---|
| 31-13<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 12-8<br>辽宁省 | 复位引脚 滤波器 总线时钟选择<br><br>选择复位引脚总线时钟过滤宽度。<br>• 小于RSTFLTSEL周期的过渡长度被过滤。<br>• RSTFLTSEL和（RSTFLTSEL+1）周期（包括）之间的过渡长度可以被过滤。<br>• 大于(RSTFLTSEL+1)周期的转换长度不被过滤。 |
| 7-3<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 2<br>RSTFLTSS | 在停止模式下的复位引脚过滤器选择<br><br>选择在任何停止模式下如何启用复位引脚过滤器。<br><br>0 禁用所有过滤功能<br>1 启用LPO时钟滤波器 |
| RSTFLTSRW | 运行和等待模式下的复位引脚过滤器选择<br>选择在运行和等待模式下如何启用复位引脚过滤。00所有<br>过滤功能被禁用<br>01总线时钟滤波器启用，用于正常操作<br>10 正常操作时启用LPO时钟滤波器<br>11 保留 |

## 24.4.5 粘性系统复位状态寄存器（RCM_SSRS）。

该寄存器包括状态标志，用于指示自上一次POR或LVD以来，所有未被软件清除的复位源。软件可以通过给一个标志写一个逻辑1来清除状态标志。

地址。0h基数+18h偏移量=18h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | 0 |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 位 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | 0 | ÃÃÃ | 0 | SMDM_AP | ÃÃÃ | 锁定 | SJTAG | 淘宝网 | 延伸阅读 | 窦文涛 | 0 | 傻子 | SLOC | SLVD | 0 |
| W | | | w1c | | w1c | w1c | w1c | w1c | w1c | w1c | w1c | | w1c | w1c | w1c | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |

**RCM_SSRS字段描述**

| 场地 | 描述 |
|---|---|
| 31-17<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 15<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 14<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 13<br>ÃÃÃ | 粘滞停止确认错误 |

表继续在下一页...

## RCM_SSRS字段描述（续）

| 场地 | 描述 |
|---|---|
| | 表示在试图进入停止模式后，由于一个或多个外设未能在大约一秒钟内确认进入停止模式而导致复位。<br><br>0 复位不是因为外设未能确认进入停止模式的尝试而引起的<br>1 由于外设未能确认进入停止模式的尝试而引起的复位 |
| 12<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 11<br>SMDM_AP | 粘性MDM-AP系统重置请求<br><br>表示由主机调试器系统设置MDM-AP控制寄存器中的系统复位请求位引起的复位。<br><br>0 复位不是由主机调试器系统设置的系统复位请求位引起的<br>1 复位是由主机调试器系统设置的系统复位请求位引起的 |
| 10<br>ĀĀĀ | 粘性软件<br><br>表示通过软件设置ARM内核中应用中断和复位控制寄存器中的SYSESETREQ位引起的复位。<br><br>0 复位不是由软件设置 SYSRESETREQ 位引起的<br>1 由软件设置 SYSRESETREQ 位引起的复位 |
| 9<br>锁定 | 粘性核心锁定<br><br>表示由于ARM核指示LOCKUP事件而引起的复位。<br><br>0 复位不是由核心LOCKUP事件引起的<br>1 由核心LOCKUP事件引起的复位 |
| 8<br>SJTAG | 粘性JTAG产生的复位<br><br>表示由于JTAG选择了某些IR代码而造成了复位。EXTEST, HIGHZ, 和 CLAMP。<br><br>0 复位不是由JTAG引起的<br>1 由JTAG引起的复位 |
| 7<br>淘宝网 | 粘性开机复位<br><br>表明上电检测逻辑已经引起了复位。因为当时内部电源电压正在上升，所以低电压复位（LVD）状态位也被设置，表示复位发生在内部电源低于LVD阈值的时候。<br><br>0 复位不是由POR引起的<br>1 由POR引起的复位 |
| 6<br>延伸阅读 | 粘性外部复位引脚<br><br>表示外部RESET(RESET_b)引脚上的有效低电平已经引起了复位。<br><br>0 复位不是由外部复位引脚引起的<br>1 由外部复位引脚引起的复位 |

| | |
|---|---|
| 5<br>窦文涛 | 粘性看门狗<br><br>表示复位是由看门狗定时器超时引起的。这个复位源可以通过禁用看门狗来阻断。 |

*表继续在下一页...*

RCM_SSRS字段描述（续）

| 场地 | 描述 |
| --- | --- |
| | 0 复位不是由看门狗超时引起的<br>1 由看门狗超时引起的复位 |
| 4<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 3<br>傻子 | 粘性失锁复位<br><br>表示由于SCG PLL/FLL的失锁而引起的复位。有关失锁事件的信息，请参见SCG的描述。<br><br>0 复位不是由PLL/FLL的失锁引起的<br>1 因PLL/FLL失去锁定而引起的复位 |
| 2<br>SLOC | 粘性失时钟复位<br><br>表明外部时钟丢失导致了复位。必须启用SCG的SOSC时钟监视器才能检测到时钟的丢失。关于启用时钟监视器的信息，请参考详细的SCG描述。<br><br>0 不是由外部时钟丢失引起的复位。<br>1 由外部时钟丢失引起的复位。 |
| 1<br>SLVD | 粘性低电压检测复位<br><br>如果PMC_LVDSC1[LVDRE]被设置，且电源下降到LVD跳闸电压以下，则发生LVD复位。这个字段也被POR设置。<br><br>0 复位不是由LVD跳闸或POR引起的<br>1 由LVD跳闸或POR引起的复位 |
| 0<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |

## 24.4.6 系统复位中断使能寄存器（RCM_SRIE）。

这个寄存器提供了一个选项，在产生中断的同时，将系统复位的断言延迟一段时间（DELAY字段）。当一个复位源的中断被启用时，软件有时间来执行优雅的关机。芯片POR源不能被这个功能延迟。SRS只在系统复位发生后更新。

**注意事项**

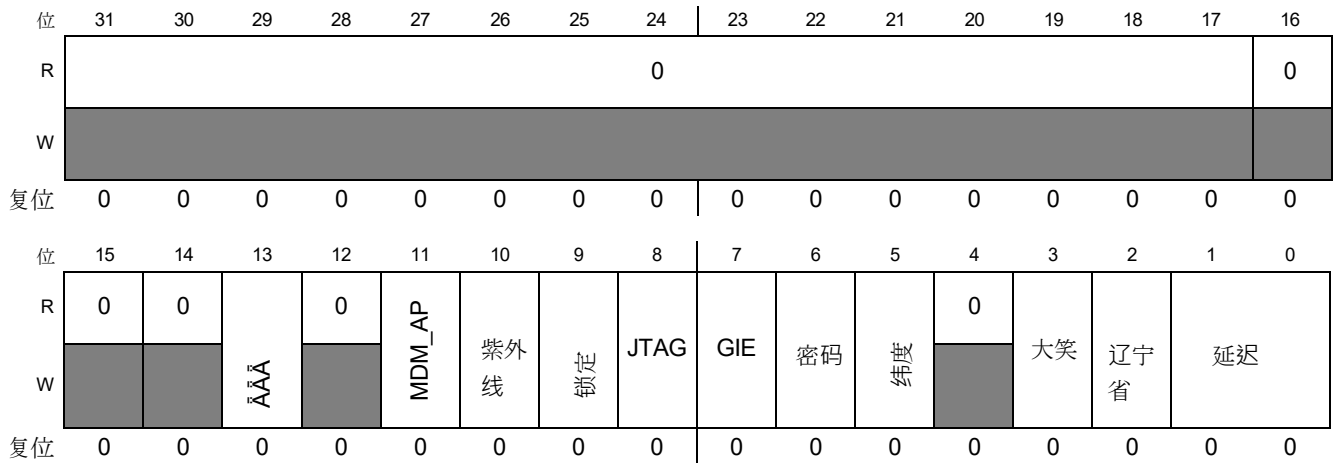
复位延迟功能要求LPO时钟保持有效。

**注意事项**

这个寄存器只在芯片POR时被复位，不受其他复位类型的

影响。

地址。0h基数+1Ch偏移量=1Ch




## RCM_SRIE字段描述

| 场地 | 描述 |
|---|---|
| 31-17<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 15<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 14<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 13<br>ÃÃÃ | 停止确认错误中断<br><br>0 中断禁用。<br>1 中断启用。 |
| 12<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 11<br>MDM_AP | MDM-AP系统复位请求<br><br>0 中断禁用。<br>1 中断启用。 |
| 10<br>紫外线 | 软件中断<br><br>0 中断禁用。<br>1 中断启用。 |
| 9<br>锁定 | 核心锁定中断<br><br>**注意：**LOCKUP位只对具有一个以上核心处理器的设备有用。<br><br>0 中断禁用。<br>1 中断启用。 |

| | |
|---|---|
| 8<br>JTAG | JTAG产生的复位<br><br>0 中断禁用。<br>1 中断启用。 |

*表继续在下一页...*

## RCM_SRIE字段描述（续）

| 场地 | 描述 |
|---|---|
| 7<br>GIE | 全局中断启用<br><br>0 所有中断源被禁用。<br>1 所有的中断源都被启用。请注意，各个中断使能位仍然需要被设置以产生中断。 |
| 6<br>密码 | 外部复位引脚中断<br><br>0 复位不是由外部复位引脚引起的<br>1 由外部复位引脚引起的复位 |
| 5<br>纬度 | 看门狗中断<br><br>0 中断禁用。<br>1 中断启用。 |
| 4<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 3<br>大笑 | 失锁中断<br><br>0 中断禁用。<br>1 中断启用。 |
| 2<br>辽宁省 | 失时钟中断<br><br>0 中断禁用。<br>1 中断启用。 |
| 延迟 | 复位延迟时间<br><br>配置从中断断言到系统复位发生时的最大复位延迟时间。<br><br>00 10个LPO周期<br>01 34个LPO周期<br>10 130个LPO周期<br>11 514个LPO周期 |

# 第二十五章
# 时钟分布

## 25.1 简介

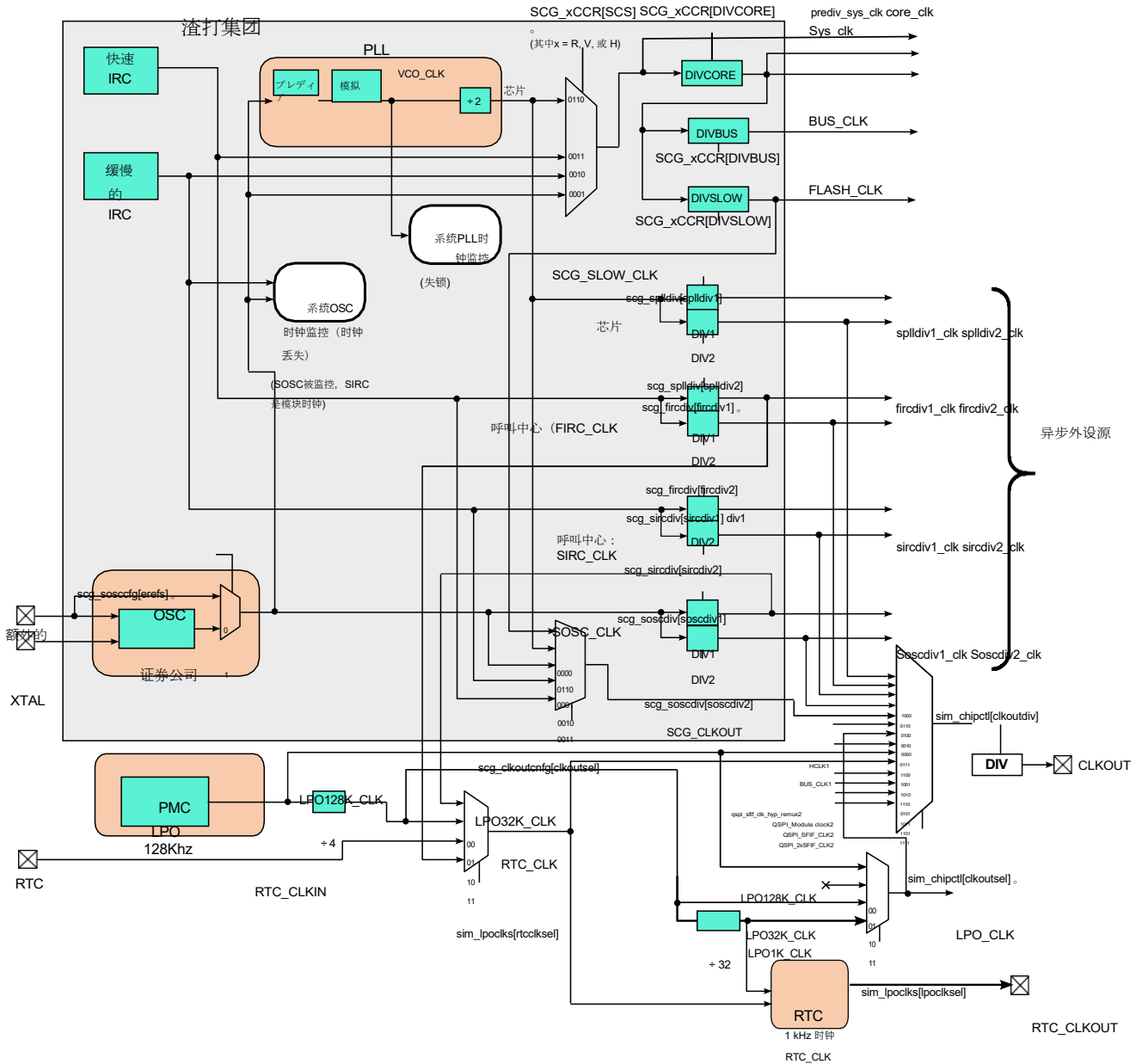
本章介绍了该器件的时钟结构概述、时钟分布、模块时钟和时钟术语。系统时钟发生器（SCG）模块用于生成该器件使用的大部分时钟。SCG模块控制哪个时钟源（内部参考、外部晶体、外部时钟）用来产生系统时钟。SCG还将选定的时钟源划分为各种时钟域，包括系统总线主站、系统总线从站和闪存的时钟。

时钟生成电路提供了几个时钟分频器和选择器，允许不同的模块以该模块的特定频率进行时钟控制。时钟生成逻辑还实现了特定模块的时钟门控，允许模块被单独禁用。因此，允许对性能或低功率进行优化。

各种模块都有特定的时钟，可以从FIRC_CLK、SIRC_CLK、SOSC_CLK、SPLL_CLK或电源管理控制器（PMC）时钟信号（LPO128K_CLK）产生。此外，模块的特定时钟可以从其他来源进行配置。大多数模块的时钟选择是由PCC模块控制的。

## 25.2 高水平的时钟图

下图显示了该器件的高层时钟结构和各种时钟源。




1 HCLK（对ARM® Cortex®模块）来自CORE_CLK。关于CORE_CLK和BUS_CLK，请参见 "时钟定义 "部分。

2 QSPI时钟只适用于S32K148，其他的保留。见 "外围模块时钟 "表中的QuadSPI点击图。

**图25-1.时钟图**

## 25.3 时钟的定义

下表描述了图25-1和本文件其他章节中显示的时钟。

## 表25-1.时钟描述

| 钟表名称 | 相关的时钟选择器 | 相关的时钟分频器 | 描述 |
|---|---|---|---|
| 预测_sys_c lk | SCG_xCCR[SCS] | - | 在HSRUN 80模式下为QSPI存储器提供时钟。 |
| CORE_CLK1 | SCG_xCCR[SCS] | SCG_xCCR[DIVCORE] (÷1...16) | 为ARM内核提供时钟，由SCG内部的DIVCORE位划分。 |
| 呼叫中心：SYS_CLK1 | SCG_xCCR[SCS] | SCG_xCCR[DIVCORE] (÷1...16) | 为Crossbar、NVIC、Flash控制器、FTM、PDB等提供时钟。 |
| BUS_CLK | SCG_xCCR[SCS] | SCG_xCCR[DIVBUS] (÷ 1...16) | 为芯片外围设备提供时钟。 |
| FLASH_CLK [2] (SCG_SLOW_CLK in SCG) | SCG_xCCR[SCS] | SCG_xCCR[DIVSLOW] (÷1...8) | 为闪光模块计时。 |
| 芯片 | - | - | PLL的输出(VCO_CLK÷2) |
| 呼叫中心：SIRC_CLK | - | - | 慢速IRC的输出时钟。 |
| 呼叫中心（FIRC_CLK | - | - | 快速IRC的输出时钟。 |
| SOSC_CLK | scg_sosccfg[erefs]。 | - | 系统振荡器时钟。可以是EXTAL引脚或振荡器的输出（SOSC）。<br>**注意：**ERCLK/OSCERCLK代表着为同一时钟源，在一些模块的章节中。 |
| RTC_CLKOUT | - | - | RTC振荡器输出驱动外部引脚。 |
| LPO32K_CLK | sim_lpoclks[clk32sel] | LPO_CLK的固定除以4驱动RTC_CLK复用器的01b输入。 | RTC的源时钟。 |
| LPO128K_CLK | - | - | 由PMC产生的低功耗振荡器时钟始终处于开启状态。 |
| SCG_CLKOUT | scg_clkoutcnfg[clkout sel] | - | SCG输出时钟，可由SOSC_CLK、SIRC_CLK、FIRC_CLK、SPLL_CLK或SCG_SLOW_CLK（FLASH_CLK）驱动。 |
| LPO_CLK | sim_lpoclks[lpoclksel] | - | 从三个LPO时钟源（LPO128K_CLK、LPO32K_CLK、LPO1K_CLK）中的一个产生时钟输出。 |
| CLKOUT | sim_chipctl[clkoutsel]。 | sim_chipctl[clkoutdiv] (÷1...8) | 从八个内部时钟源中选择一个。 |
| VCO_CLK | - | - | PLL内的VCO输出时钟。其频率=SPLL_CLK × 2。 |

| | | | |
|---|---|---|---|
| SPLLDIV1_CLK | - | SCG_SPLLDIV[SPLLDIV1] (÷ 1, 2, 4, 8, 16, 32, 64, 或输出禁用) | 除以SPLL_CLK<br>在RUN模式下应配置为80MHz或更低,<br>在HSRUN模式下应配置为112MHz或更低。 |
| SPLLDIV2_CLK | - | SCG_SPLLDIV[SPLLDIV2] (÷ 1, 2, 4, 8, 16, 32, 64, 或输出禁用) | 除以SPLL_CLK<br>在RUN模式下应配置为40MHz或更低，在HSRUN模式下应配置为56MHz或更低。 |

*表继续在下一页...*

表25-1.时钟描述（续）

| 钟表名称 | 相关的时钟选择器 | 相关的时钟分频器 | 描述 |
|---|---|---|---|
| FIRCDIV1_CLK | - | SCG_FIRCDIV[FIRCDIV1] (÷ 1, 2, 4, 8, 16, 32, 64, 或输出禁用) | 除去FIRC_CLK<br>在RUN/HSRUN模式下应配置为48MHz或更低。 |
| FIRCDIV2_CLK | - | SCG_FIRCDIV[FIRCDIV2] (÷ 1, 2, 4, 8, 16, 32, 64, 或输出禁用) | 除去FIRC_CLK<br>在RUN/HSRUN模式下应配置为48MHz或更低。 |
| SIRCDIV1_CLK | - | SCG_SIRCDIV[SIRCDIV1] (÷ 1, 2, 4, 8, 16, 32, 64, 或输出禁用) | 被分割的SIRC_CLK<br>在RUN/HSRUN模式下应配置为8MHz或更少，在VLPR/VLPS模式下应配置为4MHz或更少。 |
| SIRCDIV2_CLK | - | SCG_SIRCDIV[SIRCDIV2](÷ 1, 2, 4, 8, 16, 32, 64, 或输出禁用<br>) | 被分割的SIRC_CLK<br>在RUN/HSRUN模式下应配置为8MHz或更少，在VLPR/VLPS模式下应配置为4MHz或更少。 |
| SOSCDIV1_CLK | - | SCG_SPLLDIV[SOSCDIV1] (÷ 1, 2, 4, 8, 16, 32, 64, 或输出禁用) | 被分割的SOSC_CLK<br>在RUN/HSRUN模式下应配置为40MHz或更低。 |
| SOSCDIV2_CLK | - | SCG_SPLLDIV[SOSCDIV2] (÷ 1, 2, 4, 8, 16, 32, 64, 或输出禁用) | 被分割的SOSC_CLK<br>在RUN/HSRUN模式下应配置为40MHz或更低。 |

1. SYS_CLK/CORE_CLK不能被配置为小于BUS_CLK。
2. 关于并行启动期间FLASH_CLK的工作范围，请参见设备启动模式

## 25.4 内部时钟要求

时钟分频器是通过SCG模块的时钟分频器寄存器进行编程的。在为该芯片配置时钟时，必须满足以下要求。

- CORE_CLK和SYS_CLK的时钟频率在HSRUN模式下必须是112 MHz或更低，在正常RUN模式下必须是80 MHz（但不能配置为低于BUS_CLK）。
- BUS_CLK频率在HSRUN中必须编程为56 MHz或更低，在RUN中必须编程为48 MHz或更低（当使用PLL作为系统时钟源时，最大总线时钟频率为40 MHz。参见下面的选项1和3），以及CORE_CLK的一个整数除法。

- FLASH_CLK频率在HSRUN中必须被编程为28 MHz或更低，在RUN中必须被编程为26.67 MHz或更低，并且是CORE_CLK的整数倍。核心时钟与FLASH时钟的比率被限制在8的最大值内。

下面是该芯片在四种时钟模式下的一些常见时钟配置。

选项1：慢速RUN（通常使用未分割的FIRC）[1]，使用以下内存地图/寄存器定义寄存器设置。

- SCG_RCCR[SCS] = 0011b
- SCG_RCCR[DIVCORE] = 0000b
- SCG_RCCR[DIVBUS] = 0000b
- SCG_RCCR[DIVSLOW] = 0001b

**表25-2.慢速RUN示例**

| 时钟 | 频率 |
|---|---|
| CORE_CLK | 48兆赫 |
| 呼叫中心 | 48兆赫 |
| BUS_CLK | 48 MHz (RUN模式下的最大频率) |
| FLASH_CLK | 24兆赫 |

选项2：正常运行（VCO_CLK = 320 MHz，SPLL_CLK = 160 MHz），使用以下内存地图/寄存器定义寄存器设置。

- SCG_RCCR[SCS] = 0110b
- SCG_RCCR[DIVCORE] = 0001b
- SCG_RCCR[DIVBUS] = 0001b
- SCG_RCCR[DIVSLOW] = 0010b

**表25-3.正常运行示例**

| 钟表 | 频率 |
|---|---|
| CORE_CLK | 80兆赫 |
| 呼叫中心 | 80兆赫 |
| BUS_CLK | 40兆赫 |
| FLASH_CLK | 26.67 MHz (RUN模式下的最大频率) |

选项3：正常运行（VCO_CLK = 256 MHz，SPLL_CLK = 128 MHz），使用以下内存映射/寄存器定义寄存器设置。

- SCG_RCCR[SCS] = 0110b
- SCG_RCCR[DIVCORE] = 0001b
- SCG_RCCR[DIVBUS] = 0001b
- SCG_RCCR[DIVSLOW] = 0010b

1. 复位后的默认配置。FIRC_CLK = 48 MHz。

表25-4.正常运行示例

| 钟表 | 频率 |
|---|---|
| CORE_CLK | 64兆赫 |
| 呼叫中心 | 64兆赫 |
| BUS_CLK | 32 MHz |
| FLASH_CLK | 21.33 MHz |

选项4：高速运行（VCO_CLK = 224 MHz，SPLL_CLK = 112 MHz），使用以下内存地图/寄存器定义寄存器设置。

- SCG_HCCR[SCS] = 0110b
- SCG_HCCR[DIVCORE] = 0000b
- SCG_HCCR[DIVBUS] = 0001b
- SCG_HCCR[DIVSLOW] = 0011b

表25-5.高速RUN示例

| 时钟 | 频率 |
|---|---|
| CORE_CLK | 112兆赫 |
| 呼叫中心 | 112兆赫 |
| BUS_CLK | 56兆赫 |
| FLASH_CLK | 28兆赫 |

## 注意事项

上表中列出的所有频率都是HSRUN模式的最大频率。

选项5：高速RUN 80（VCO_CLK = 320 MHz，SPLL_CLK = 160 MHz），使用以下内存地图/寄存器定义寄存器设置。

- SCG_HCCR[SCS] = 0110b
- SCG_HCCR[DIVCORE] = 0001b
- SCG_HCCR[DIVBUS] = 0001b
- SCG_HCCR[DIVSLOW] = 0010b

表25-6.高速RUN 80示例（仅QuadSPI使用的模式）

| 钟表 | 频率 |
|---|---|
| CORE_CLK | 80兆赫 |
| 呼叫中心 | 80兆赫 |
| BUS_CLK | 40兆赫 |
| FLASH_CLK | 26.67 MHz（RUN模式下的最大频率） |

选项6：极低功率运行，VLPR（与SIRC_CLK = 8 MHz），使用以下内存映射/寄存器定义寄存器设置。

- SCG_VCCR[SCS] = 0010b (SIRC_CLK)
- SCG_VCCR[DIVCORE] = 0001b
- SCG_VCCR[DIVBUS] = 0000b
- SCG_VCCR[DIVSLOW] = 0011b

**表25-7.极低功率RUN示例**

| 时钟 | 频率 |
|---|---|
| CORE_CLK | 4兆赫 |
| 呼叫中心 | 4兆赫 |
| BUS_CLK | 4兆赫 |
| FLASH_CLK | 1兆赫 |

**注意事项**

上表中列出的所有频率都是VLPR模式的最大频率。

**注意事项**

在SCG_SIRCDIV中配置的VLPR/VLPS模式下，所有异步时钟源也将被限制在4MHz。

### 25.4.1 复位后的时钟分频器值

复位后的默认配置是CPU由快速IRC（FIRC_CLK）提供时钟。时钟（例如，CORE_CLK、FLASH_CLK和BUS_CLK）是在SCG模块中配置的（见内存图/寄存器定义）。

### 25.4.2 HSRUN模式的时钟

当芯片工作在HSRUN模式时，不应修改时钟分频器。它们必须在进入HSRUN模式之前进行配置，以保证。

- CORE_CLK/SYS_CLK小于或等于112MHz
- BUS_CLK小于或等于56 MHz
- FLASH_CLK小于或等于28MHz

### 25.4.3 VLPR模式的时钟

当芯片工作在VLPR模式时，不应修改时钟分频器。它们必须在进入VLPR模式之前进行配置，以保证。

- CORE_CLK/SYS_CLK和BUS_CLK小于或等于4 MHz
- FLASH_CLK小于或等于1MHz

**注意事项**

在SCG_SIRCDIV中配置的VLPR/VLPS模式下，所有异步时钟源也将被限制在4MHz。

### 25.4.4 VLPR/VLPS模式条目

当进入VLPR/VLPS模式时，在进行任何模式转换之前，必须通过软件在RUN模式下禁用PLL。

## 25.5 时钟门控

每个模块的时钟可以通过PCC模块单独开启和关闭。在任何复位后，PCC会禁用相应模块的时钟，以节省电能。在初始化一个模块之前，在PCC寄存器中设置相应的时钟门控位以启用时钟。在关闭时钟之前，确保禁用该模块。

任何对一个时钟被禁用的外设的总线访问都会产生一个错误终止。

## 25.6 模块时钟

下表总结了每个模块可以使用的时钟。

**表25-8.周边时钟汇总**

| 模块名称 | 总线接口时钟[1] | 总线接口时钟1门控 | 外围功能时钟 | 附加时钟 | 评论和最高频率 |
|---|---|---|---|---|---|
| | | 由PCC的[CGC]把关 | 由PCC的[PCS]控制的时钟 | | |
| 通讯 | | | | | |

表25-8.外围时钟汇总（续）

| | | | | | |
|---|---|---|---|---|---|
| 朴槿惠 | BUS_CLK | 是 | spll div2_clk, fircdiv2_clk, sircdiv2_clk, soscdiv2_clk | - | 由BUS_CLK管理的最大频率 |

*表继续在下一页...*

| 模块名称 | 总线接口时钟 [1] | 总线接口时钟1门控 | 外围功能时钟 | 附加时钟 | 评论和最高频率 |
|---|---|---|---|---|---|
| | | 由PCC的[CGC]把关 | 由PCC的[PCS]控制的时钟 | | |
| LPSPI | BUS_CLK | 是 | spll div2_clk, firc div2_clk, sirc div2_clk, sosc div2_clk | - | 由BUS_CLK管理的最大频率 |
| LPI2C | BUS_CLK | 是 | spll div2_clk, firc div2_clk, sirc div2_clk, sosc div2_clk | - | 由BUS_CLK管理的最大频率 |
| FlexIO [3] | BUS_CLK | 是 | spll div2_clk, firc div2_clk, sirc div2_clk, sosc div2_clk | - | 由BUS_CLK管理的最大频率 |
| 弹性CAN | 呼叫中心 | 是 | - | sys_clk, sosc div2_clk | 支持来自OSC的40MHz；SYS_CLK必须是>1.5倍的协议时钟；而同步操作（当协议时钟选择为SYS_CLK时）可以在1：1的时钟频率下进行。 |
| **计时器** | | | | | |
| LPTMR | BUS_CLK | 是 | spll div2_clk, firc div2_clk, sirc div2_clk, sosc div2_clk | rtc_clk2, sirc div2_clk, lpo1k_clk | 由BUS_CLK管理的最大频率 |
| LPIT | BUS_CLK | 是 | spll div2_clk, firc div2_clk, sirc div2_clk, sosc div2_clk | - | 由BUS_CLK管理的最大频率 |
| RTC | BUS_CLK | 是 | - | rtc_clk2, lpo1k_clk | 由BUS_CLK管理的最大频率 |
| PDB | 呼叫中心 | 是 | - | - | 由SYS_CLK控制的最大频率 |
| 伸缩式定时器 | 呼叫中心 | 是 | spll div1_clk, firc div1_clk, sirc div1_clk, sosc div1_clk | RTC_CLK2, SYS_CLK, TCLKx | 由SYS_CLK控制的最大频率 |
| **系统模块** | | | | | |

*表继续在下一页...*

## 表25-8.外围时钟汇总（续）

| 模块名称 | 总线接口时钟[1] | 总线接口时钟1门控 | 外围功能时钟 | 附加时钟 | 评论和最高频率 |
|---|---|---|---|---|---|
| | | 由PCC的[CGC]把关 | 由PCC的[PCS]控制的时钟 | | |
| 纬度 | BUS_CLK | 没有 | - | lpo_clk[4], sosc_clk, sirc_clk | 由BUS_CLK管理的最大频率 |
| EWM | BUS_CLK | 是 | - | LPO_CLK4 | 由BUS_CLK管理的最大频率 |
| PMC | BUS_CLK | 没有 | - | - | 由BUS_CLK管理的最大频率 |
| 卫星电话 | BUS_CLK | 没有 | - | - | 由BUS_CLK管理的最大频率 |
| RCM | BUS_CLK | 没有 | - | LPO128K_CLK | 由BUS_CLK管理的最大频率 |
| 儿童权利委员会 | BUS_CLK | 是 | - | - | 由BUS_CLK管理的最大频率 |
| 港口 | BUS_CLK | 是 | - | LPO128K_CLK | 由BUS_CLK管理的最大频率 |
| 芯片 | 呼叫中心 | 没有 | - | - | 由SYS_CLK控制的最大频率 |
| 淘宝网 | BUS_CLK | 没有 | - | - | 由BUS_CLK管理的最大频率 |
| 梦之城_梦之城娱乐_梦之城国际娱乐_梦之城国际娱乐平台 | BUS_CLK | 是 | - | - | 由BUS_CLK管理的最大频率 |
| DMA | 呼叫中心 | 没有 | - | - | 由SYS_CLK控制的最大频率 |
| 蒙特利尔大学 | 呼叫中心 | 没有 | - | - | 最大频率 |

*表继续在下一页...*

**表25-8.外围时钟汇总（续）**

| 模块名称 | 总线接口时钟[1] | 总线接口时钟1门控 | 外围功能时钟 | 附加时钟 | 评论和最高频率 |
|---|---|---|---|---|---|
| | | 由PCC的[CGC]把关 | 由PCC的[PCS]控制的时钟 | | |
| | | | | | 受SYS_CLK支配 |
| EIM | 呼叫中心 | 没有 | - | - | 由SYS_CLK控制的最大频率 |
| ERM | 呼叫中心 | 没有 | - | - | 由SYS_CLK控制的最大频率 |
| ÃÃÃ | 呼叫中心 | 没有 | - | - | 由SYS_CLK控制的最大频率 |
| **记忆模块** | | | | | |
| FTFC5 | FLASH_CLK | 是 | - | - | 由FLASH_CLK控制的最大频率 |
| 系统RAM | 呼叫中心 | 没有 | - | - | 由SYS_CLK控制的最大频率 |
| **模拟模块** | | | | | |
| ADC | BUS_CLK | 是 | splldiv2_clk, fircdiv2_clk, sircdiv2_clk, soscdiv2_clk | - | 50 MHz6 |
| CMP | BUS_CLK | 是 | - | - | 由BUS_CLK管理的最大频率 |
| QuadSPI | Bus_clk/ Sys_clk | 是 | SPLLDIV1_CLK FIRCDIV1_CLK | - | - |
| SAI | BUS_CLK | 是 | - | MCLK（来自PAD） BCLK（来自PAD） | - |
| ENET7 | 呼叫中心 | 是 | splldiv1_clk sircdiv1_clk fircdiv1_clk SOSCDIV1_CLK | enet_mii_rmii_tx_clk (来自PAD) | MAC时间戳的时钟周期必须是整数纳秒。 |

1. 指的是模块的接口时钟，不应误解为BUS_CLK
2. RTC_CLK是多路复用器的输出，它使用SIM_LPOCLKS[RTCCLKSEL]进行源选择（详见图1）。
3. FlexIO外设时钟不应超过FlexIO总线接口时钟的两倍。对于快速访问FlexIO寄存器（FLEXIO_CTRL[FASTACC]=1），

表25-8 外围时钟汇总（续）

| | | |
|---|---|---|
| FlexIO外围时钟可以是FlexIO总线接口时钟的两倍。 | | |

4. LPO_CLK是多路复用器的输出，它使用SIM_LPOCLKS[LPOCLKSEL]进行源选择（详见图1）。
5. 不要在访问闪存的同时改变闪存的时钟控制设置。相反，当需要改变频率时，建议从SRAM执行。
6. 最大的转换时钟频率是50MHz。在任何时候，转换时钟频率都应小于ADC总线接口的时钟频率。关于ADC的规格，请参见器件的数据手册。
7. 关于时钟限制，请参考外部信号描述。

## 注意事项

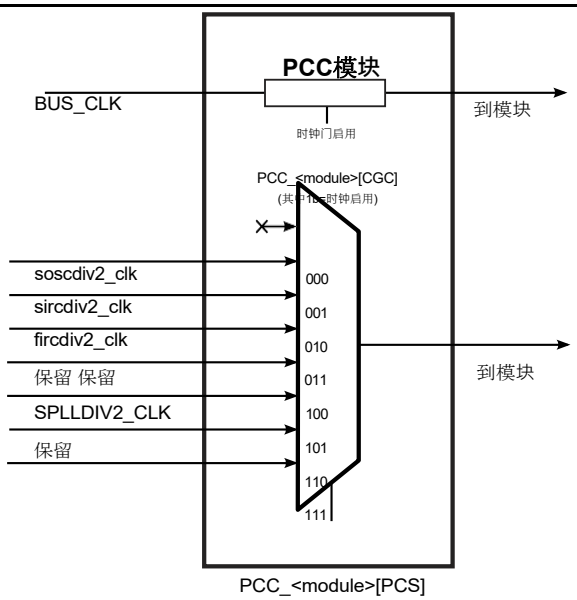
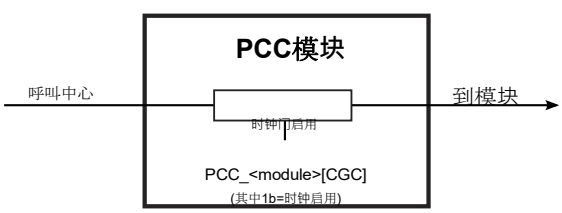
上述时钟选择是在PCC模块中控制的（表25-8第4列）。

## 注意事项

- SIRC和LPO_CLK是VLP*模式的有效时钟源。
- SPLL和FIRC是HSRUN模式的有效时钟源。

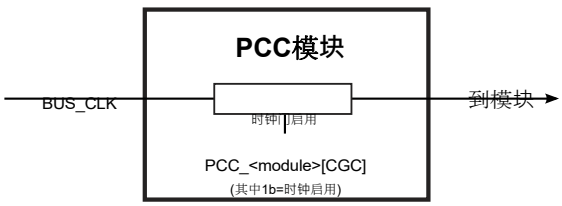
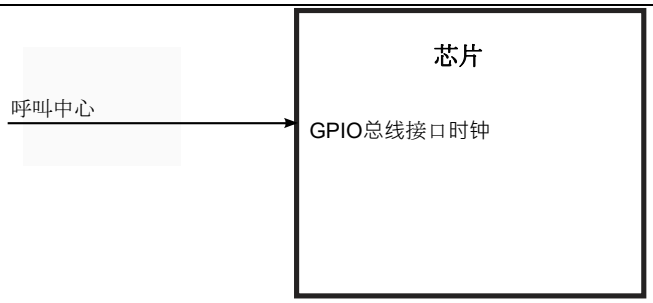
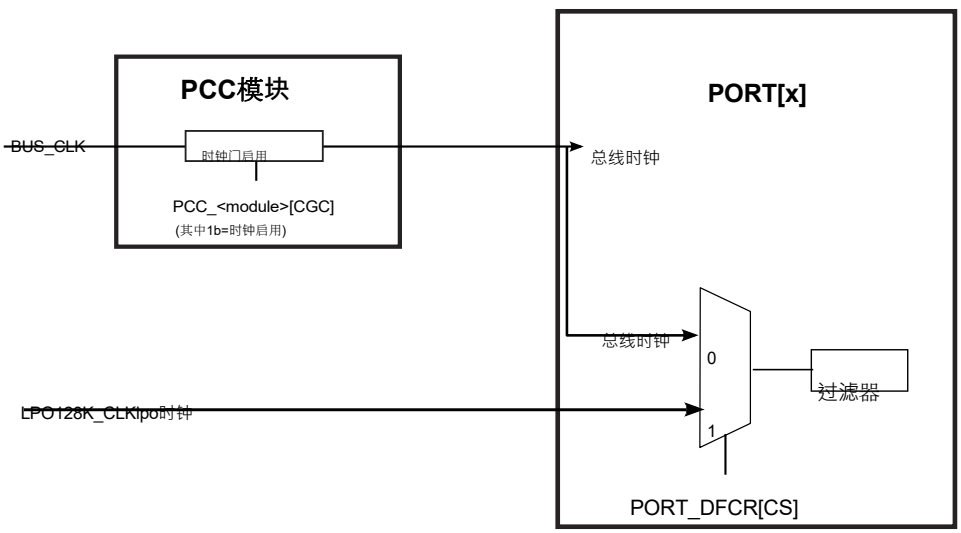
下表总结了每个模块可以使用的时钟。

**表25-9.外围模块的时钟**

| 模块 | PCC多路复用器 |
|---|---|
| LPSPI<br>LPIT<br>弹性IO<br>LPI2C<br>朴槿惠 |  |
| PDB |  |

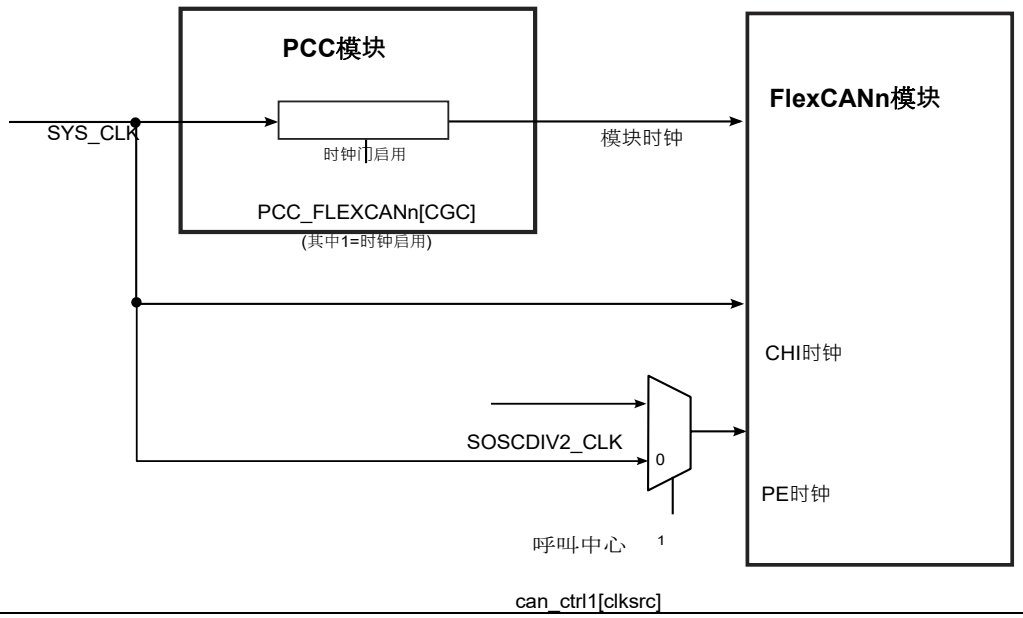
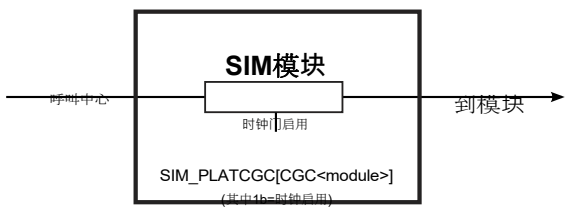
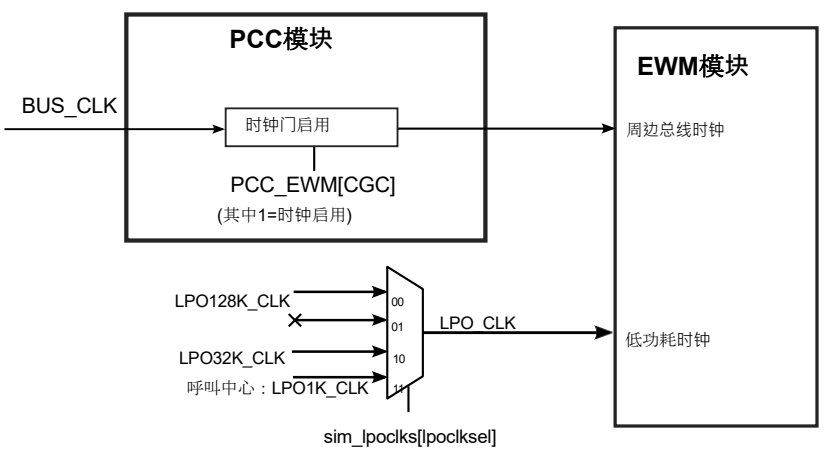
*表继续在下一页...*

表25-9 外围模块的时钟（续）

| 模块 | PCC多路复用器 |
| --- | --- |
| dmamux<br>cmp0 crc |  |
| 芯片 |  |
| 港口 |  |

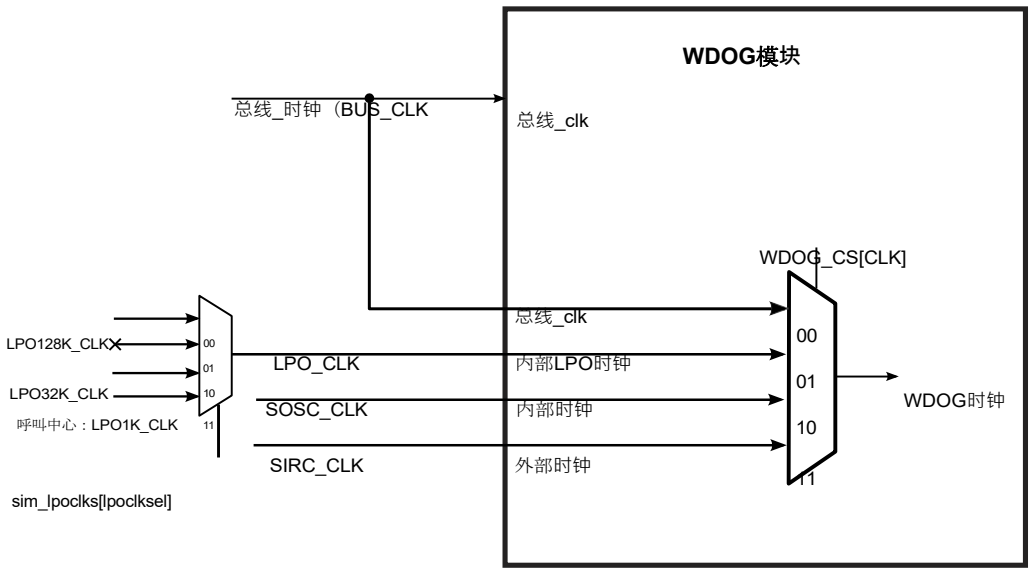
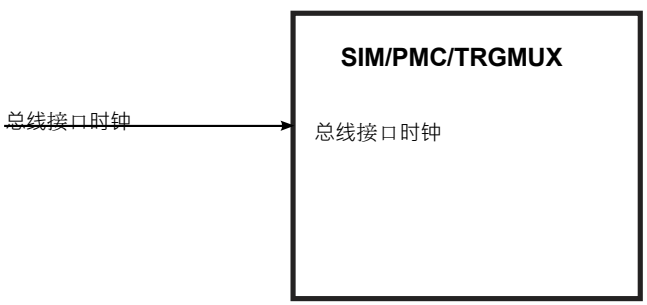
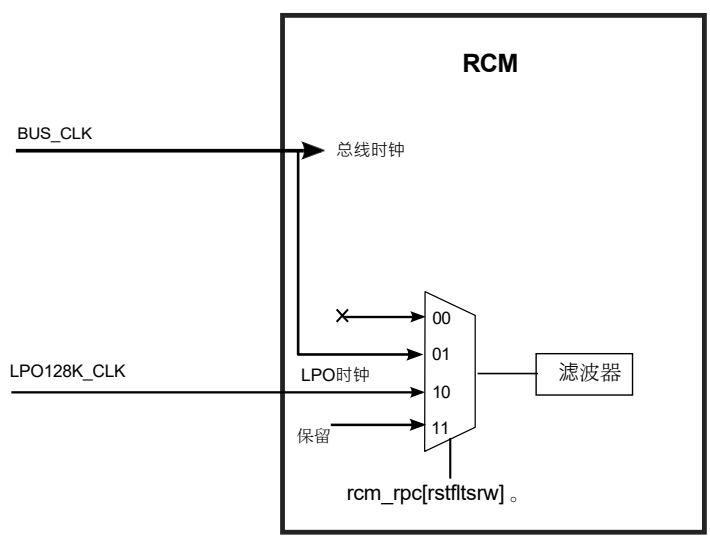
*表继续在下一页...*

表25-9.外围模块的时钟（续）

| 模块 | PCC多路复用器 |
|---|---|
| 弹性CAN |  |
| eim<br>erm<br>mscm<br>dma<br>mpu |  |
| EWM |  |

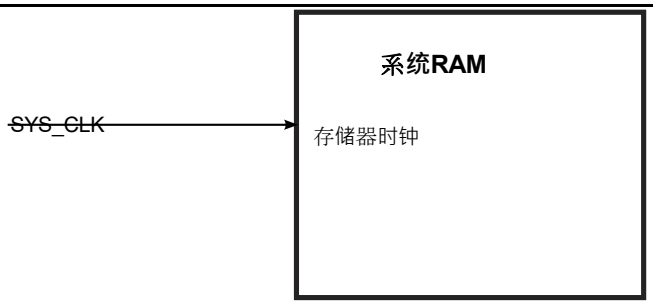
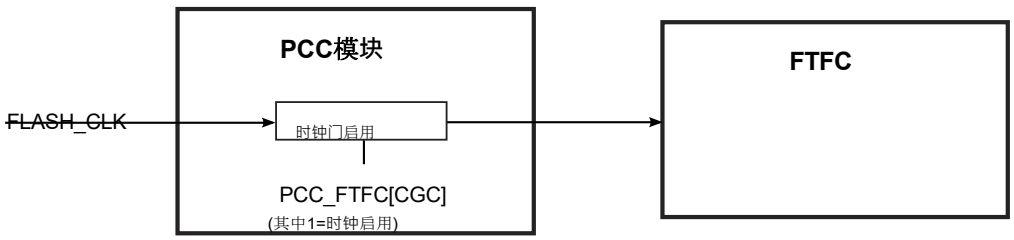
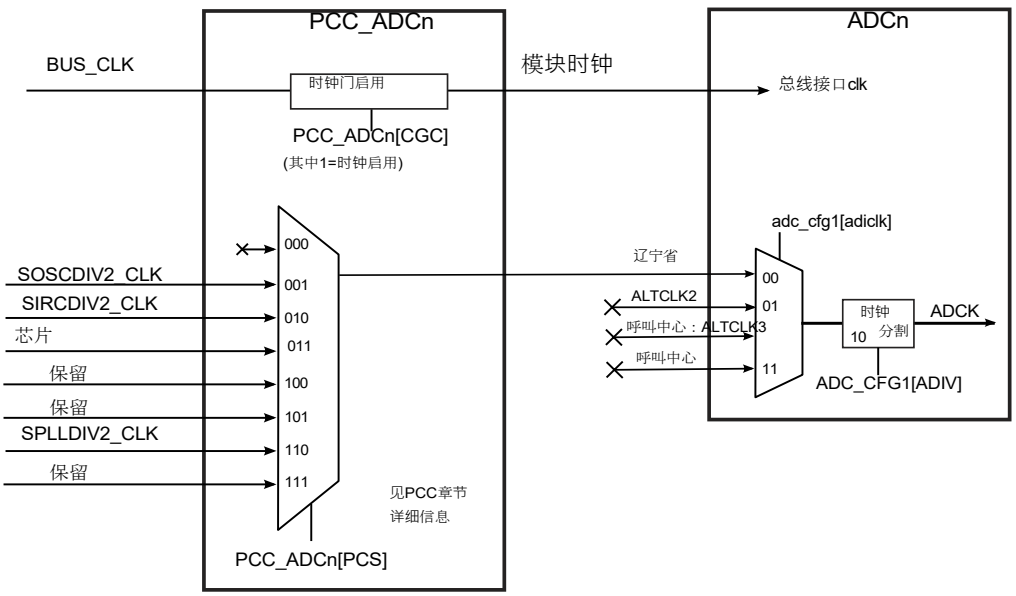
*表继续在下一页...*

表25-9.外围模块的时钟（续）

| 模块 | PCC多路复用器 |
| --- | --- |
| 纬度 |  |
| 卫星电话<br><br>PMC<br>TRGMUX |  |
| RCM |  |

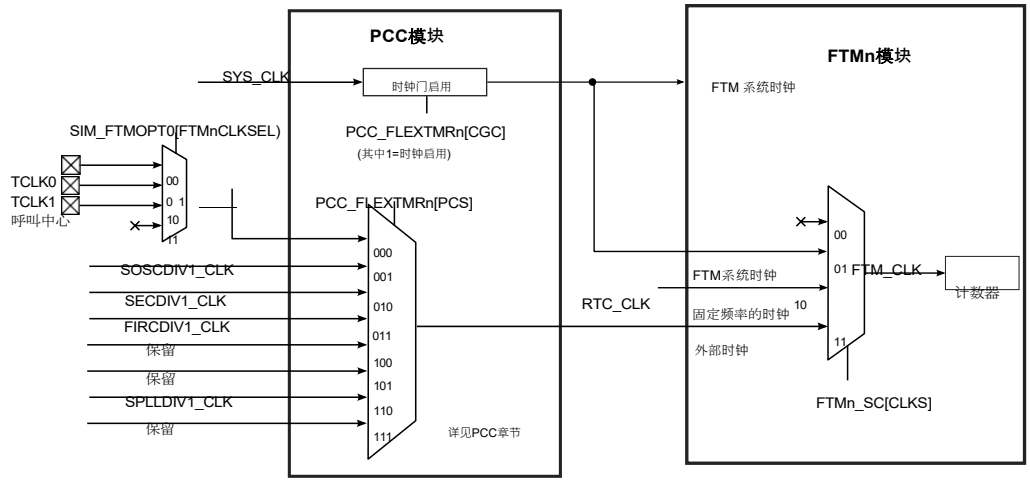
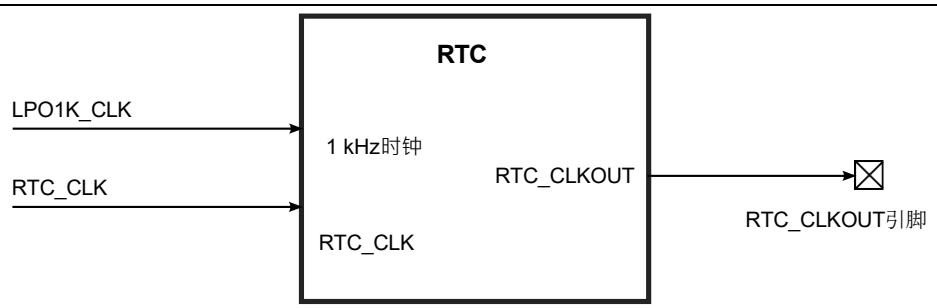
*表继续在下一页...*

表25-9.外围模块的时钟（续）

| 模块 | PCC多路复用器 |
|---|---|
| 系统RAM |  |
| 融资租赁公司 |  |
| ADC |  <br>ADC有多个时钟源。选择是由PCC_ADCx[PCS]的配置决定的。分频器的配置应使ADC转换的时钟频率位于ADC要求的有效范围内（见数据表）。 |
| 基金会 | |

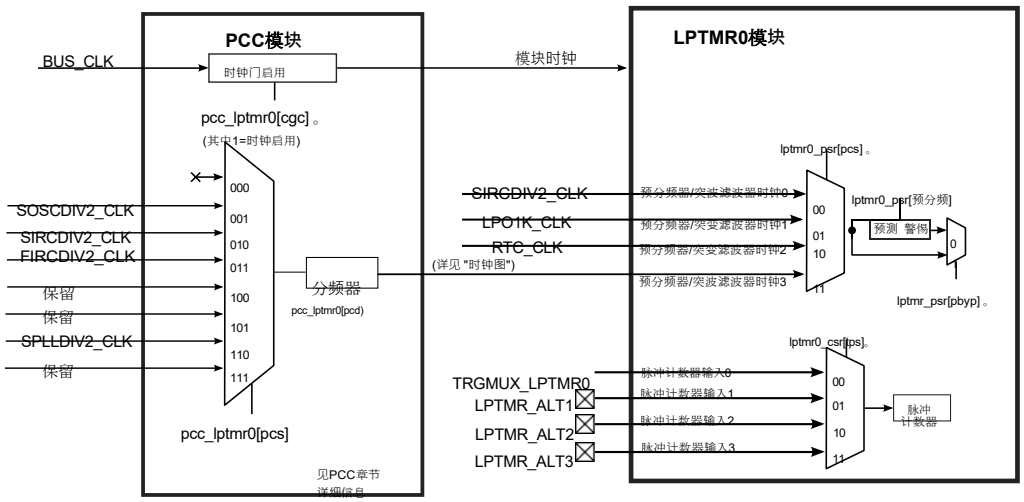
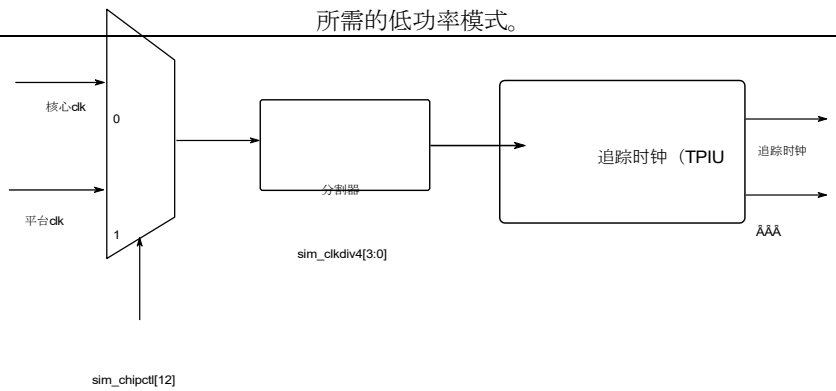
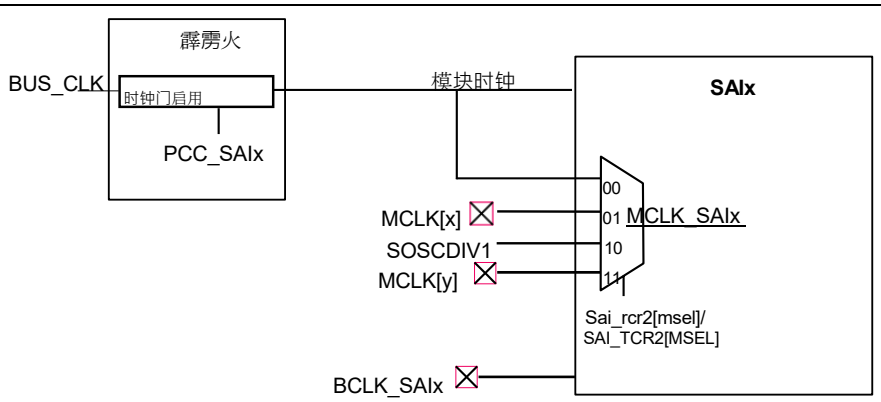
*表继续在下一页...*

表25-9 外围模块的时钟（续）

| 模块 | PCC多路复用器 |
|---|---|
| |  <br>FTM模块由内部SYS_CLK（FTM模块称其为系统时钟）时钟，它可以达到CPU的频率，但FTM计数器允许由三个时钟源时钟。<br>• 来自PCC的系统时钟（SYS_CLK）。<br>• 内部固定频率的时钟<br>• 来自引脚的外部时钟（TCLK[2:0]）。<br>计数器时钟源的选择由 FTMn_SC 寄存器中的 CLKS 位控制。FTM_CLK由PCC模块控制，可以运行到CPU的频率，为FTM定时器提供更高的分辨率。<br>固定频率的时钟是一个由RTC_CLK驱动的固定时钟。SIM_LPOCLKS[RTCCLKSEL]用于选择RTC_CLK源。<br>有三个时钟可以从外部接入芯片（TCLK0、TCLK1、TCLK2）。通过配置SIM_FTMOPT0[FTMnCLKSEL]，选择其中一个时钟来驱动PCC中复用器的000b输入（PCC_FLEXTMRn[PCS] = 000b）。<br>固定频率的时钟和外部时钟为用户提供了更多FTM计数器的时钟选择。 |
| RTC |  |

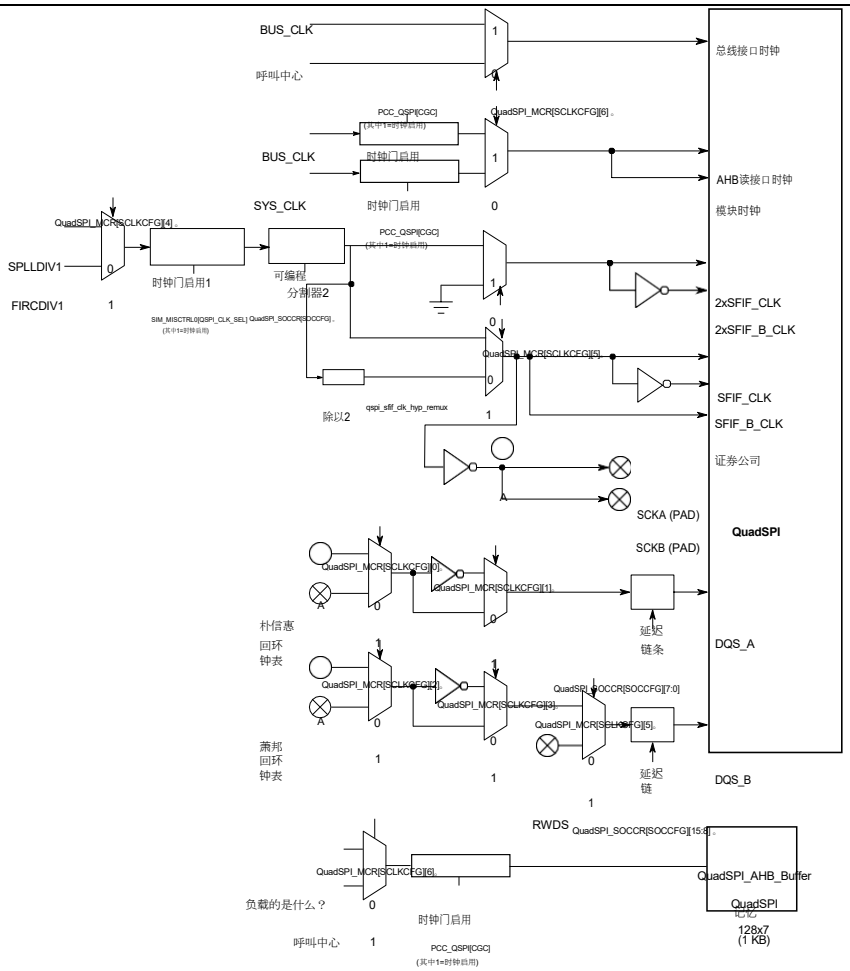
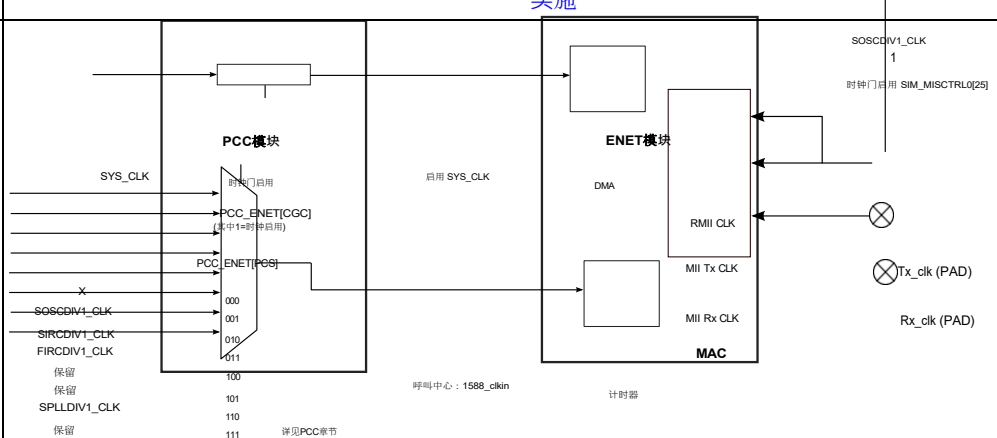
*表继续在下一页...*

表25-9.外围模块的时钟（续）

| 模块 | PCC多路复用器 |
|---|---|
| LPTMR |  <br>LPTMR0模块的预分频器和突变滤波器可以从PCC_LPTMR[PCS]控制位域和LPTMR0_PSR[PCS]位域确定的混杂源中获得时钟。LPTMR_PSR[PCS]位域用于内部LPTMR选择时钟源。该器件支持的时钟源如下图所示。<br>**注意。** 如果要使LPTMR在所有情况下继续工作，所选择的时钟必须保持启用。所需的低功率模式。 |
| TPIU |  |
| SAI |  <br>BCLK可以通过BCLK引脚从外部提供，也可以通过配置SAIn_RCR2[MSEL]/SAIn_TCR2[MSEL]和SAIn_RCR2[DIV]/SAIn_TCR2[DIV]，在内部位时钟模式下利用SAI内部分频器内部产生。 |

*表继续在下一页...*

表25-9.外围模块的时钟（续）

| 模块 | PCC多路复用器 |
|---|---|
| | **注意。** BCLK绝对不能大于总线接口的时钟，因为它将导致欠载或不合格。<br>SAI内FIFO的溢出。 |
| QuadSPI |  <br>**注意事项** 1. 时钟门启用。sim_misctrl0[qspi_clk_sel]。<br>2. 关于可编程分压器的配置，见QuadSPI_SOCCR[SOCCFG]实施 |
| ENET |  <br>**注意。** SIM_MISCTRL0的第24和25位用于控制哪个时钟（外部或内部）。<br>产生的）仅在RMII模式下被MAC使用。 |

### 注意事项

在改变外围时钟源时，应禁用相应的模块。

# 第二十六章
# 系统时钟发生器（SCG）

## 26.1 芯片特定的SCG信息

该设备不支持等待模式。关于可用的电源模式的详细信息，请参见模块在可用的低功率模式下的操作。

**注意事项**

S32K11x系列器件中没有SPLL。

### 26.1.1 支持的频率范围

本节介绍了支持的频率范围。

1. **scg_sosccfg[范围]**
   - 00 保留
   - 01 保留
   - 10 为晶体振荡器选择的中等频率范围为4 MHz至8 MHz。
   - 11 为晶体振荡器选择的高频率范围为8 MHz至40 MHz。
2. **scg_firccfg[范围]**
   - 00 快速IRC被修剪到48MHz
   - 01 保留
   - 10 保留
   - 11 保留

**注意事项**

软件不应该将SCG_FIRCCFG[RANGE]配置为00以外的任何值。

3. **scg_sirccfg[范围]**
   - 0 保留
   - 1个慢速IRC高范围时钟（8MHz）。

### 26.1.2 振荡器限制

如果使用PLL，那么振荡器只需要在高范围内，SCG_SOSCCFG[RANGE]在11上，作为参考时钟使用。

如果当前的系统时钟是SOSC，那么下面的工作应该由软件来完成。

- 通过RCM_SRIE将所有复位源配置为 "中断"（而不是 "复位"）。
- 应通过SCG_SOSCCSR[SOSCEN]禁用SOSC。
- 在禁用SOSC后，通过RCM_SRIE将复位源配置回复位，见IO信号表中的

注释。

### 26.1.3 通过SCG_FIRCCSR[FIRCEN]启用/禁用FIRC的指南。

在启动模式转换请求到任何低功耗模式之前，在RUN模式下禁用FIRC。

在RUN模式下禁用FIRC的程序。

- 可配置的SIRC作为系统时钟
- 通过RCM_SRIE将所有复位源配置为 "复位"（而非中断）。
- 通过RCM_SRIE将每个复位源编程为中断，最小延迟时间为10 LPO。
- 禁用FIRC
- 执行少量的nops以确保45ns的时间间隔
- 通过RCM_SRIE将每个复位源配置回原定的复位配置（中断或复位）。

### 26.1.4 系统时钟切换

对于系统时钟的任何时钟切换，请遵循以下步骤。

- 在进行时钟切换之前，通过RCM_SRIE将所有复位源配置为 "复位"（而不是中断）。
- 通过RCM_SRIE将每个复位源编程为中断，最小延迟时间为10 LPO。
- 执行时钟切换
- 等待/查询时钟切换完成
- 通过RCM_SRIE将每个复位源配置回原定的复位配置（中断或复位）。

## 26.2 简介

系统时钟发生器（SCG）模块提供MCU的系统时钟。SCG包含一个系统锁相环（SPLL），一个慢速内部参考时钟（SIRC），一个快速内部参考时钟（FIRC），以及系统振荡器时钟（SOSC）。SPLL是由SOSC参考时钟提供的。SCG可以选择SPLL的输出时钟或SCG参考时钟（SIRC、FIRC和SOSC）作为MCU系统时钟的来源。单片机还支持与晶体振荡器一起工作，这允许外部晶体、陶瓷谐振器或另一个外部时钟源产生外部参考时钟（它们也可作为MCU系统时钟的时钟源）。

### 26.2.1 特点

SCG模块的主要特点是。

- 系统锁相环（SPLL）。
  - 电压控制的振荡器（VCO）
  - 外部参考时钟被用作PLL源
  - Modulo VCO分频器
  - 相位/频率检测器
  - 集成环路滤波器
  - 可以被选为MCU系统时钟的时钟源
  - 2个可编程的后分频器时钟输出，可作为其他片上外设的时钟源使用
- 2个内部参考时钟（IRC）生成器。
  - 快速IRC时钟，具有可编程的高、低频率范围
  - 可以选择慢速或快速时钟作为MCU系统时钟的时钟源。
  - 每个IRC有2个可编程的后分频器时钟输出，可作为其他片上外设的时钟源使用
- 系统晶体振荡器。

- 用作系统PLL的来源
- 可以被选为MCU系统时钟的时钟源

- 具有复位和中断请求能力的SPLL、SOSC、时钟的时钟监视器
- 具有中断请求能力的锁定检测器，与SPLL一起使用
- 每个时钟源都有参考分频器，用于为片上模块和外围设备提供时钟，即。
  - spll div1_clk / spll div2_clk
  - fircdiv1_clk / scg_fircdiv2_clk
  - sircdiv1_clk / sircdiv2_clk
  - Soscdiv1_clk / Soscdiv2_clk

更多信息请参见时钟分配章节。

## 26.3 存储器地图/寄存器的定义

本节包括内存图和寄存器定义。

SCG寄存器只有在监督员模式下才能被写入。在用户模式下进行写操作将导致总线传输错误。读取访问可以在监管者和用户模式下进行。

**SCG内存图**

| 地址偏移（十六进制） | 注册名称 | 宽度（单位：比特） | 访问 | 重置值 | 节/页 |
|---|---|---|---|---|---|
| 0 | 版本ID寄存器（SCG_VERID）。 | 32 | R | 0100_0000h | 26.3.1/539 |
| 4 | 参数寄存器（SCG_PARAM）。 | 32 | R | 见本节 | 26.3.2/540 |
| 10 | 时钟状态寄存器（SCG_CSR）。 | 32 | R | 见本节 | 26.3.3/540 |
| 14 | 运行时钟控制寄存器（SCG_RCCR）。 | 32 | R/W | 见本节 | 26.3.4/543 |
| 18 | VLPR时钟控制寄存器（SCG_VCCR）。 | 32 | R/W | 见本节 | 26.3.5/545 |
| 1C | HSRUN时钟控制寄存器（SCG_HCCR）。 | 32 | R/W | 见本节 | 26.3.6/547 |
| 20 | SCG CLKOUT配置寄存器（SCG_CLKOUTCNFG）。 | 32 | R/W | 0300_0000h | 26.3.7/549 |
| 100 | 系统OSC控制状态寄存器（SCG_SOSCCSR）。 | 32 | R/W | 见本节 | 26.3.8/551 |
| 104 | 系统OSC分频寄存器（SCG_SOSCDIV）。 | 32 | R/W | 0000_0000h | 26.3.9/553 |
| 108 | 系统振荡器配置寄存器（SCG_SOSCCFG）。 | 32 | R/W | 0000_0010h | 26.3.10/554 |
| 200 | 慢速IRC控制状态寄存器（SCG_SIRCCSR）。 | 32 | R/W | 0100_0005h | 26.3.11/556 |

*表继续在下一页...*

SCG内存图（续）

| 地址偏移（十六进制） | 注册名称 | 宽度（单位：比特） | 访问 | 重置值 | 节/页 |
|---|---|---|---|---|---|
| 204 | 慢速IRC除法寄存器（SCG_SIRCDIV）。 | 32 | R/W | 0000_0000h | 26.3.12/557 |
| 208 | 慢速IRC配置寄存器（SCG_SIRCCFG）。 | 32 | R/W | 0000_0001h | 26.3.13/558 |
| 300 | 快速IRC控制状态寄存器（SCG_FIRCCSR）。 | 32 | R/W | 见本节 | 26.3.14/559 |
| 304 | 快速IRC除法寄存器（SCG_FIRCDIV）。 | 32 | R/W | 0000_0000h | 26.3.15/561 |
| 308 | 快速IRC配置寄存器（SCG_FIRCCFG）。 | 32 | R/W | 0000_0000h | 26.3.16/562 |
| 600 | 系统PLL控制状态寄存器（SCG_SPLLCSR）。 | 32 | R/W | 0000_0000h | 26.3.17/563 |
| 604 | 系统PLL除法寄存器（SCG_SPLLDIV）。 | 32 | R/W | 0000_0000h | 26.3.18/565 |
| 608 | 系统PLL配置寄存器（SCG_SPLLCFG）。 | 32 | R/W | 0000_0000h | 26.3.19/566 |

## 26.3.1 版本ID寄存器（SCG_VERID）。

注意：对该寄存器的写入将导致一个传输错误。

地址。0h base + 0h offset = 0h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 版本说明 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SCG_VERID字段描述

| 场地 | 描述 |
|---|---|
| 版本说明 | SCG版本号 |

## 26.3.2 参数寄存器（SCG_PARAM）。

注意：对该寄存器的写入将导致一个传输错误。

地址。0h基数+4h偏移量=4h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | DIVPRES | | | | | 0 | | | | | | | | | | | 0 | | | | | | | | 辽宁省 | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | | | | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | | | | | | | * |

* 注意事项。

- DIVPRES字段。复位值由Soc使用哪些SCG系统分频器来控制。
- CLKPRES字段。复位值由Soc使用的SCG时钟源控制，请参考参考手册时钟章节。

**SCG_PARAM字段描述**

| 场地 | 描述 |
|---|---|
| 31-27<br>DIVPRES | 分割器存在<br>表示在SCG的这个实例中存在哪些系统时钟分频器。DIVPRES[27]=1 系统<br>DIVSLOW存在。<br>DIVPRES[28]=1 系统DIVBUS存在<br>DIVPRES[31]=1 系统DIVCORE存在 |
| 26-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 15-8<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 辽宁省 | 钟表现况<br>表示在SCG的这个实例中存在哪些时钟源。任何未在此位域中定义的位都是保留位，读取时总是数值为0。<br>CLKPRES[0] 保留 保留<br>CLKPRES[1]=1System OSC (SOSC) is present<br>CLKPRES[2]=1Slow IRC (SIRC) is present<br>CLKPRES[3]=1Fast IRC (FIRC) is present<br>CLKPRES[6]=1System PLL (SPLL) is present |

## 26.3.3 时钟状态寄存器（SCG_CSR）。

该寄存器返回当前配置的系统时钟源以及核心（DIVCORE）和外围接口时钟（DIVSLOW）的系统时钟分频器。SCG_CSR反映了由三个时钟控制寄存器SCG_RCC

R、SCG_VCCR、SCG_HCCR之一设置的配置。

注意：对该寄存器的写入将导致一个传输错误。

地址。0h基数+10h偏移量=10h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | SCS | | | | 0 | | | | DIVCORE | | | | 0 | | | | 0 | | | | DIVBUS | | | | DIVSLOW | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | | | * | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

* 注意事项。

- DIVCORE字段。**复位**值由复位时上传的用户FOPT**位控制。当复位到RUN模式**时，有效的复位值是div-by-1或div-by-2；**当复位到VLPR模式**时，有效的复位值是div-by-4或div-by-8。

## SCG_CSR字段描述

| 场地 | 描述 |
|---|---|
| 31-28<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 27-24<br>SCS | 系统时钟源<br>返回当前配置的产生系统时钟的时钟源。0000 保留<br>0001系统OSC (SOSC_CLK)<br>0010Slow IRC (SIRC_CLK)<br>0011快速IRC (FIRC_CLK)<br>0100 保留<br>0101 保留<br>0110系统PLL (SPLL_CLK)<br>0111 保留 |
| 23-20<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 19-16<br>DIVCORE | 核心时钟分割率<br>如果选择SPLL作为系统时钟源，最大DIVCORE值为除以4。0000 除以1<br>0001 除以2<br>0010 除以3<br>0011 除以4<br>0100 除以5<br>0101 除以6<br>0110 除以7<br>0111 除以8<br>1000 除以9<br>1001 除以10<br>1010 除以11<br>1011 除以12<br>1100 除以13<br>1101 除以14 |

*表继续在下一页...*

## SCG_CSR字段描述（续）

| 场地 | 描述 |
|---|---|
| | 1110 除以15<br>1111 除以16 |
| 15-12<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 11-8<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 7-4<br>DIVBUS | 总线时钟分频率<br><br>0000 除以1<br>0001 除以2<br>0010 除以3<br>0011 除以4<br>0100 除以5<br>0101 除以6<br>0110 除以7<br>0111 除以8<br>1000 除以9<br>1001 除以10<br>1010 除以11<br>1011 除以12<br>1100 除以13<br>1101 除以14<br>1110 除以15<br>1111 除以16 |
| DIVSLOW | 慢速时钟分频比<br><br>0000 除以1<br>0001 除以2<br>0010 除以3<br>0011 除以4<br>0100 除以5<br>0101 除以6<br>0110 除以7<br>0111 除以8<br>1000 保留<br>1001 保留<br>1010 保留<br>1011 保留<br>1100 保留<br>1101 保留<br>1110 保留<br>1111 保留 |

## 26.3.4 运行时钟控制寄存器（SCG_RCCR）。

该寄存器控制系统时钟源和系统时钟分频器，用于核心、平台、外部和总线时钟域，仅在运行模式下。这个寄存器只能用32位的写入方式来写。在RUN模式下选择不同的时钟源需要首先启用该时钟源，并在系统时钟切换到该时钟源之前有效。如果在RUN模式下选择不同的时钟模式时，系统时钟分频比也发生了变化，新的系统时钟分频比将不会生效，直到新的时钟源有效。

地址。0h基数+14h偏移量=14h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | SCS | | | | 0 | | | | DIVCORE | | | | 保留 | | | | | | | 0 | DIVBUS | | | | DIVSLOW | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | * | | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

* 注意事项。

- DIVCORE字段。**复位**值由用户FOPT**位控制**，**在复位**过程中得到上传。两个有效的复位值是div-by-1和div-by-2。

**SCG_RCCR字段描述**

| 场地 | 描述 |
|---|---|
| 31-28<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 27-24<br>SCS | 系统时钟源<br><br>选择在运行模式下产生系统时钟的时钟源。试图选择一个无效的时钟将被忽略。在运行模式下选择一个不同的时钟源需要首先启用该时钟源，并且在系统时钟被允许切换到该时钟源之前是有效的。<br><br>0000 保留<br>0001系统OSC (SOSC_CLK)<br>0010Slow IRC (SIRC_CLK)<br>0011快速IRC (FIRC_CLK)<br>0100 保留<br>0101 保留<br>0110系统PLL (SPLL_CLK)<br>0111 保留 |
| 23-20<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 19-16<br>DIVCORE | 核心时钟分割率<br><br>0000 除以1<br>0001 除以2<br>0010 除以3<br>0011 除以4 |

*表继续在下一页...*

## SCG_RCCR字段描述（续）

| 场地 | 描述 |
|---|---|
| | 0100 除以5<br>0101 除以6<br>0110 除以7<br>0111 除以8<br>1000 除以9<br>1001 除以10<br>1010 除以11<br>1011 除以12<br>1100 除以13<br>1101 除以14<br>1110 除以15<br>1111 除以16 |
| 15-12<br>保留 | 这个字段是保留的。软件应该把这些位写成0，以保持兼容性。<br><br>这个字段是保留的。 |
| 11-8<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 7-4<br>DIVBUS | 总线时钟分频率<br><br>0000 除以1<br>0001 除以2<br>0010 除以3<br>0011 除以4<br>0100 除以5<br>0101 除以6<br>0110 除以7<br>0111 除以8<br>1000 除以9<br>1001 除以10<br>1010 除以11<br>1011 除以12<br>1100 除以13<br>1101 除以14<br>1110 除以15<br>1111 除以16 |

| | |
|---|---|
| DIVSLOW | 慢速时钟分频比<br><br>0000 除以1<br>0001 除以2<br>0010 除以3<br>0011 除以4<br>0100 除以5<br>0101 除以6<br>0110 除以7<br>0111 除以8<br>1000 保留<br>1001 保留 |

*表继续在下一页...*

**SCG_RCCR字段描述（续）**

| 场地 | 描述 |
|---|---|
| | 1010 保留<br>1011 保留<br>1100 保留<br>1101 保留<br>1110 保留<br>1111 保留 |

## 26.3.5 VLPR时钟控制寄存器（SCG_VCCR）。

该寄存器仅在VLPR模式下控制系统时钟源和核心、平台、外部和总线时钟域的系统时钟分压器。这个寄存器只能用32位的写入方式来写。在VLPR模式下，选择不同的时钟源需要首先启用该时钟源，并且在系统时钟切换到该时钟源之前有效。如果在VLPR模式下选择不同的时钟模式时，系统时钟分频比也发生了变化，新的系统时钟分频比将不会生效，直到新的时钟源有效。

地址。0h基数+18h偏移量=18h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | SCS | | | | 0 | | | | DIVCORE | | | | 保留 | | | | 0 | | | | DIVBUS | | | | DIVSLOW | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | * | | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

* 注意事项。

- DIVCORE字段。**复位**值由用户FOPT**位控制**，**在复位**过程中得到上传。两个选项的复位值是div-by-4和div-by-8。

**SCG_VCCR字段描述**

| 场地 | 描述 |
|---|---|
| 31-28<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 27-24<br>SCS | 系统时钟源<br><br>选择在VLPR模式下产生系统时钟的时钟源。试图选择一个无效的时钟将被忽略。选择产生系统时钟的时钟源。在VLPR模式下选择一个不同的时钟源，需要在系统时钟切换到该时钟源之前首先启用该时钟源并使其有效。<br>0000 保留<br>0001 保留<br>0010Slow IRC (SIRC_CLK)<br>0011 保留<br>0100 保留 |

*表继续在下一页...*

## SCG_VCCR字段描述（续）

| 场地 | 描述 |
|---|---|
| | 0101 保留 保留<br>0110 保留<br>0111 |
| 23-20<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 19-16<br>DIVCORE | 核心时钟分割率<br><br>0000 除以1<br>0001 除以2<br>0010 除以3<br>0011 除以4<br>0100 除以5<br>0101 除以6<br>0110 除以7<br>0111 除以8<br>1000 除以9<br>1001 除以10<br>1010 11除以11<br>1011 除以12<br>1100 13除法<br>1101 除以14<br>1110 除以15<br>1111 除以16 |
| 15-12<br>保留 | 这个字段是保留的。软件应该把这些位写成0，以保持兼容性。<br><br>这个字段是保留的。 |
| 11-8<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 7-4<br>DIVBUS | 总线时钟分频率<br><br>0000 除以1<br>0001 除以2<br>0010 除以3<br>0011 除以4<br>0100 除以5<br>0101 除以6<br>0110 除以7<br>0111 除以8<br>1000 除以9<br>1001 除以10<br>1010 11除以11<br>1011 除以12<br>1100 13除法<br>1101 除以14<br>1110 除以15<br>1111 除以16 |

| | |
|---|---|
| DIVSLOW | 慢速时钟分频比 |

*表继续在下一页...*

**SCG_VCCR字段描述（续）**

| 场地 | 描述 |
|---|---|
| | 0000 除以1<br>0001 除以2<br>0010 除以3<br>0011 除以4<br>0100 除以5<br>0101 除以6<br>0110 除以7<br>0111 除以8<br>1000 保留<br>1001 保留<br>1010 保留<br>1011 保留<br>1100 保留<br>1101 保留<br>1110 保留<br>1111 保留 |

## 26.3.6 HSRUN时钟控制寄存器（SCG_HCCR）。

该寄存器仅在HSRUN模式下控制系统时钟源和核心、平台、外部和总线时钟域的系统时钟分压器。这个寄存器只能用32位的写法来写。在HSRUN模式下，选择不同的时钟源需要首先启用该时钟源，并且在系统时钟切换到该时钟源之前有效。如果在HSRUN模式下选择不同的时钟模式时，系统时钟分频比也发生了变化，新的系统时钟分频比将不会生效，直到新的时钟源有效。

地址。0h基数+1Ch偏移量=1Ch

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | SCS | | | | 0 | | | | DIVCORE | | | | 保留 | | | | | | | 0 | DIVBUS | | | | DIVSLOW | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

**SCG_HCCR字段描述**

| 场地 | 描述 |
|---|---|
| 31-28<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 27-24<br>SCS | 系统时钟源<br><br>选择在HSRUN模式下产生系统时钟的时钟源。试图选择一个无效的时钟将被忽略。在HSRUN模式下选择一个不同的时钟源将启用该时钟源，并在其有效时切换到该时钟模式。 |

*表继续在下一页...*

## SCG_HCCR字段描述（续）

| 场地 | 描述 |
| --- | --- |
| | 0000 保留<br>0001 保留<br>0010 保留<br>0011快速IRC (FIRC_CLK)<br>0100 保留<br>0101 保留<br>0110系统PLL (SPLL_CLK)<br>0111 保留 |
| 23-20<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 19-16<br>DIVCORE | 核心时钟分割率<br><br>0000 除以1<br>0001 除以2<br>0010 除以3<br>0011 除以4<br>0100 除以5<br>0101 除以6<br>0110 除以7<br>0111 除以8<br>1000 除以9<br>1001 除以10<br>1010 除以11<br>1011 除以12<br>1100 除以13<br>1101 除以14<br>1110 除以15<br>1111 除以16 |
| 15-12<br>保留 | 这个字段是保留的。软件应该把这些位写成0，以保持兼容性。<br><br>这个字段是保留的。 |
| 11-8<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |

| | |
|---|---|
| 7-4<br>DIVBUS | 总线时钟分频率<br><br>0000 除以1<br>0001 除以2<br>0010 除以3<br>0011 除以4<br>0100 除以5<br>0101 除以6<br>0110 除以7<br>0111 除以8<br>1000 除以9<br>1001 除以10<br>1010 除以11<br>1011 除以12 |

*表继续在下一页...*

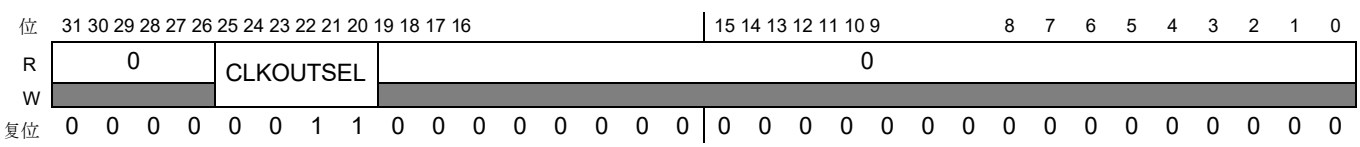
**SCG_HCCR字段描述（续）**

| 场地 | 描述 |
|---|---|
| | 1100 除以13<br>1101 除以14<br>1110 除以15<br>1111 除以16 |
| DIVSLOW | 慢速时钟分频比<br><br>0000 除以1<br>0001 除以2<br>0010 除以3<br>0011 除以4<br>0100 除以5<br>0101 除以6<br>0110 除以7<br>0111 除以8<br>1000 保留<br>1001 保留<br>1010 保留<br>1011 保留<br>1100 保留<br>1101 保留<br>1110 保留<br>1111 保留 |

## 26.3.7 SCG CLKOUT配置寄存器（SCG_CLKOUTCNFG）。

该寄存器控制哪一个SCG时钟源被选择移植到CLKOUT引脚。

地址。0h基数+20h偏移量=20h




**SCG_CLKOUTCNFG字段描述**

| 场地 | 描述 |
|---|---|
| 31-28<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 27-24<br>CLKOUTSEL | SCG Clkout选择<br><br>选择SCG系统时钟。<br><br>0000SCG 慢速时钟 |

| | |
|---|---|
| | 0001系统OSC (SOSC_CLK) |

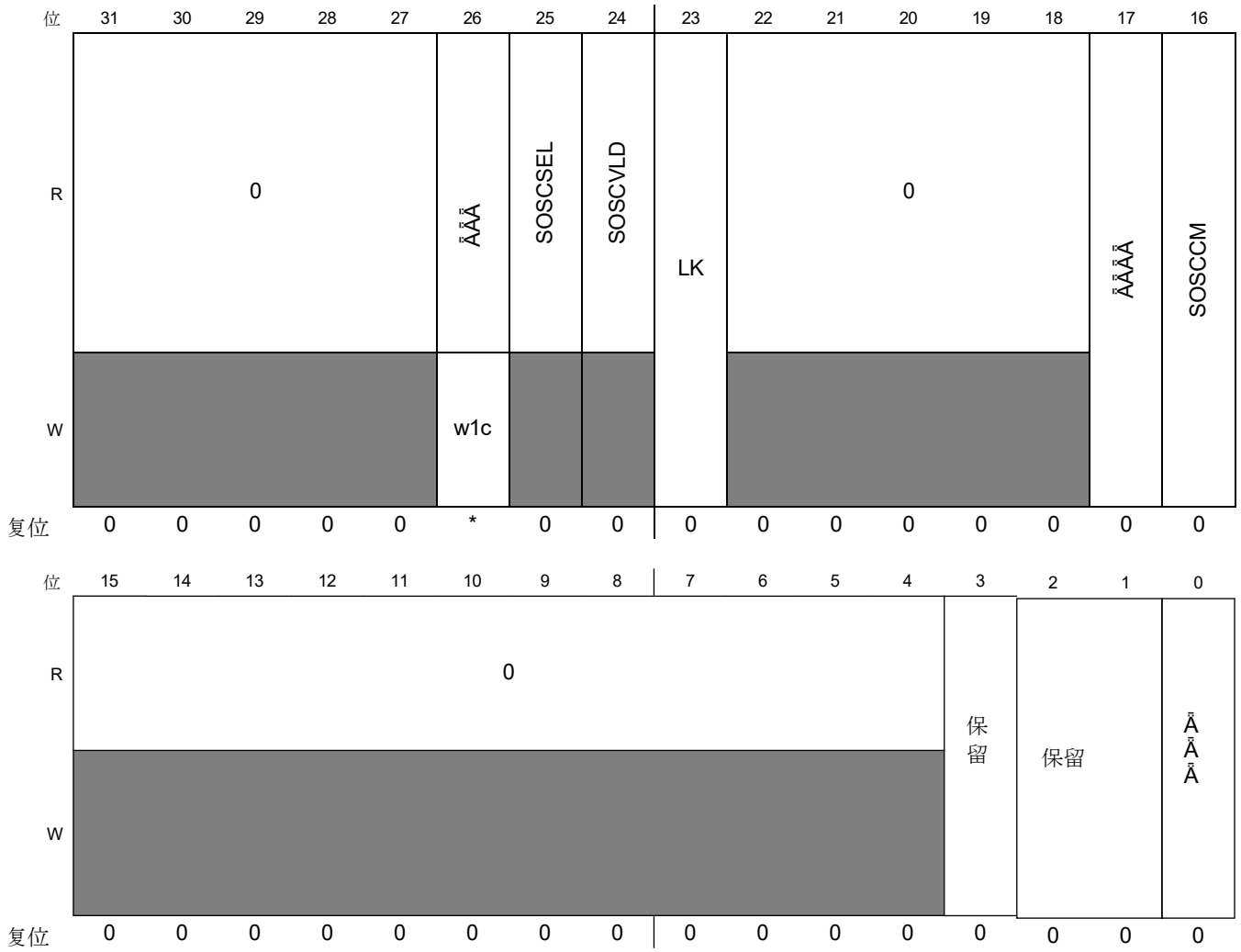
*表继续在下一页...*

## SCG_CLKOUTCNFG字段描述（续）

| 场地 | 描述 |
| --- | --- |
| | 0010Slow IRC (SIRC_CLK)<br>0011快速IRC (FIRC_CLK)<br>0100 保留<br>0101 保留<br>0110系统PLL (SPLL_CLK)<br>0111 保留<br>1111 保留 |
| 保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |

## 26.3.8 系统OSC控制状态寄存器（SCG_SOSCCSR）。

地址。0h基数+100h偏移量=100h




* 注意事项。

- SOSCERR字段。该标志仅在芯片POR时被重置

### SCG_SOSCCSR字段描述

| 场地 | 描述 |
|---|---|
| 31-27<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |

| | |
|---|---|
| 26<br>ÃÃÃ | 系统OSC时钟错误<br>这个标志只在芯片POR时被重置，软件也可以通过写一个逻辑1来清除这个标志。<br>0 系统OSC时钟监视器被禁用或未检测到错误<br>1 系统OSC时钟监控器被启用并检测到一个错误 |

*表继续在下一页...*

## SCG_SOSCCSR字段描述（续）

| 场地 | 描述 |
|---|---|
| 25<br>SOSCSEL | 选择的系统OSC<br><br>0 系统OSC不是系统时钟源<br>1 系统OSC是系统时钟源 |
| 24<br>SOSCVLD | 系统OSC有效<br><br>SOSC在4096个xtal计数后被认为是有效的。<br><br>0 系统OSC未启用或时钟无效<br>1 系统OSC被启用，输出时钟有效 |
| 23<br>LK | 锁定寄存器<br><br>这个位域可以在任何时候被清除/设置。<br><br>0 这个控制状态寄存器可以被写入。<br>1 该控制状态寄存器不能被写入。 |
| 22-18<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 17<br>ÃÃÃÃ | 系统OSC时钟监控复位启用<br><br>0 当检测到错误时，时钟监视器会产生中断<br>1 检测到错误时，时钟监控器产生复位 |
| 16<br>SOSCCM | 系统OSC时钟监控<br><br>当SOSCVLD被设置时，启用时钟监视器。如果时钟源在低功耗模式下被禁用，那么时钟监视器在低功耗模式下也被禁用。当时钟监视器在低功率模式下被禁用时，它将一直被禁用，直到退出低功率模式后时钟有效标志被设置。<br><br>0 系统OSC时钟监控被禁用<br>1 系统OSC时钟监控被启用 |
| 15-4<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 3<br>保留 | 这个字段是保留的。<br>这个字段是保留的。软件应将这些位写入0以保持兼容性。 |
| 2-1<br>保留 | 这个字段是保留的。<br>这个字段是保留的。软件应将这些位写入0以保持兼容性。 |
| 0<br>ÃÃÃ | 系统OSC启用<br><br>如果该位在时钟切换期间写入，应在继续进行之前读回并确认。<br><br>0 系统OSC被禁用 |

| | |
|---|---|
| | 1 系统OSC被启用 |

## 26.3.9 系统OSC分频寄存器（SCG_SOSCDIV）。

SCG_SOSCDIV寄存器提供了对2个时钟树的控制，可以用来提供可选的外围功能时钟，或替代模块时钟。每个时钟树都有输入SOSC时钟的可选分频器。对SOSCDIV的修改应该在系统OSC被禁用时进行，以防止对输出分频的时钟产生突变。

地址。0h基数+104h偏移量=104h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | 保留 | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 位 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | SOSCDIV2 | | | 0 | | | | | SOSCDIV1 | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**SCG_SOSCDIV字段描述**

| 场地 | 描述 |
|---|---|
| 31-19<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 18-16<br>保留 | 这个字段是保留的。<br>这个位域是保留的。软件应该写0到这个位域以保持兼容性。 |
| 15-11<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 10-8<br>SOSCDIV2 | 系统OSC时钟划分2<br><br>用于系统OSC的时钟分频器2。由需要异步时钟源的模块使用。000输出 禁用<br>001除以1<br>10 除以2<br>11 除以4<br>100 除以8<br>101 除以16<br>110 除以32<br>111 除以64 |
| 7-3<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| SOSCDIV1 | 系统OSC时钟分频1<br><br>用于系统OSC的时钟分配器1。用于为需要异步时钟源的模块生成系统时钟源。 |

| | |
|---|---|
| | 000输出禁用 |

*表继续在下一页...*

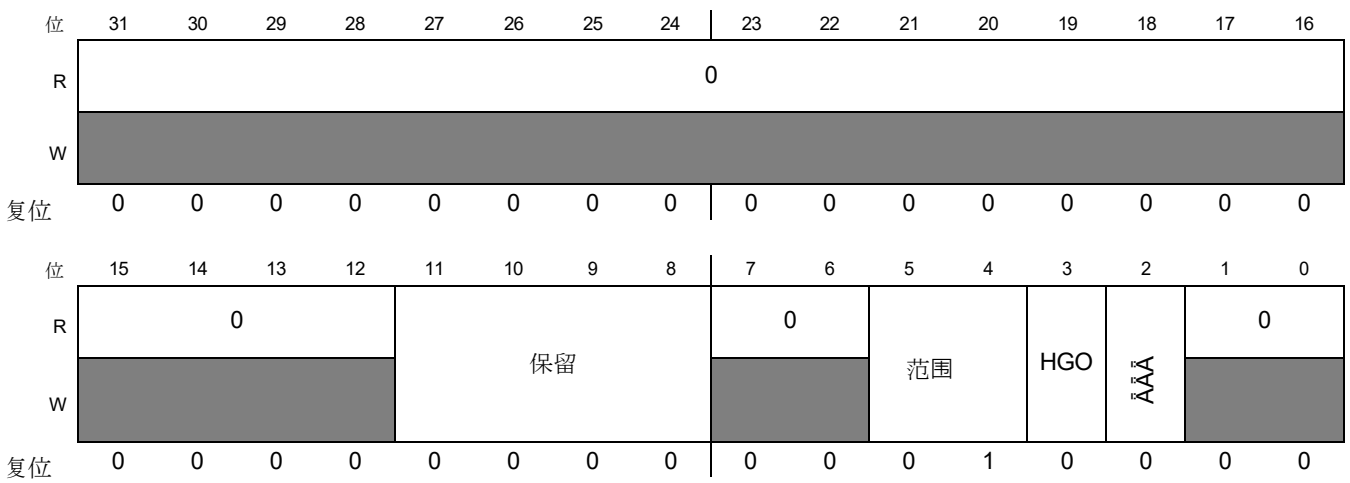
**SCG_SOSCDIV字段描述（续）**

| 场地 | 描述 |
|---|---|
| | 001除以1<br>10 除以2<br>11 除以4<br>100 除以8<br>101 除以16<br>110 除以32<br>111 除以64 |

## 26.3.10 系统振荡器配置寄存器（SCG_SOSCCFG）。

SOSCCFG 寄存器在系统 OSC
启用时不能被改变。当系统OSC被启用时，对该寄存器的写入将被忽略，并且不会出现传输错误。

地址。0h base + 108h offset = 108h




**SCG_SOSCCFG字段描述**

| 场地 | 描述 |
|---|---|
| 31-12<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 11-8<br>保留 | 这个字段是保留的。<br>此位是保留的。软件应将此位写入0。 |
| 7-6<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |

| | |
|---|---|
| 5-4<br>范围 | 系统OSC范围选择<br><br>选择系统晶体振荡器（OSC）的频率范围<br><br>**关于支持的晶体振荡器范围，请参见芯片特定信息。**<br><br>00 保留<br>01 为晶体振荡器选择的低频范围 |

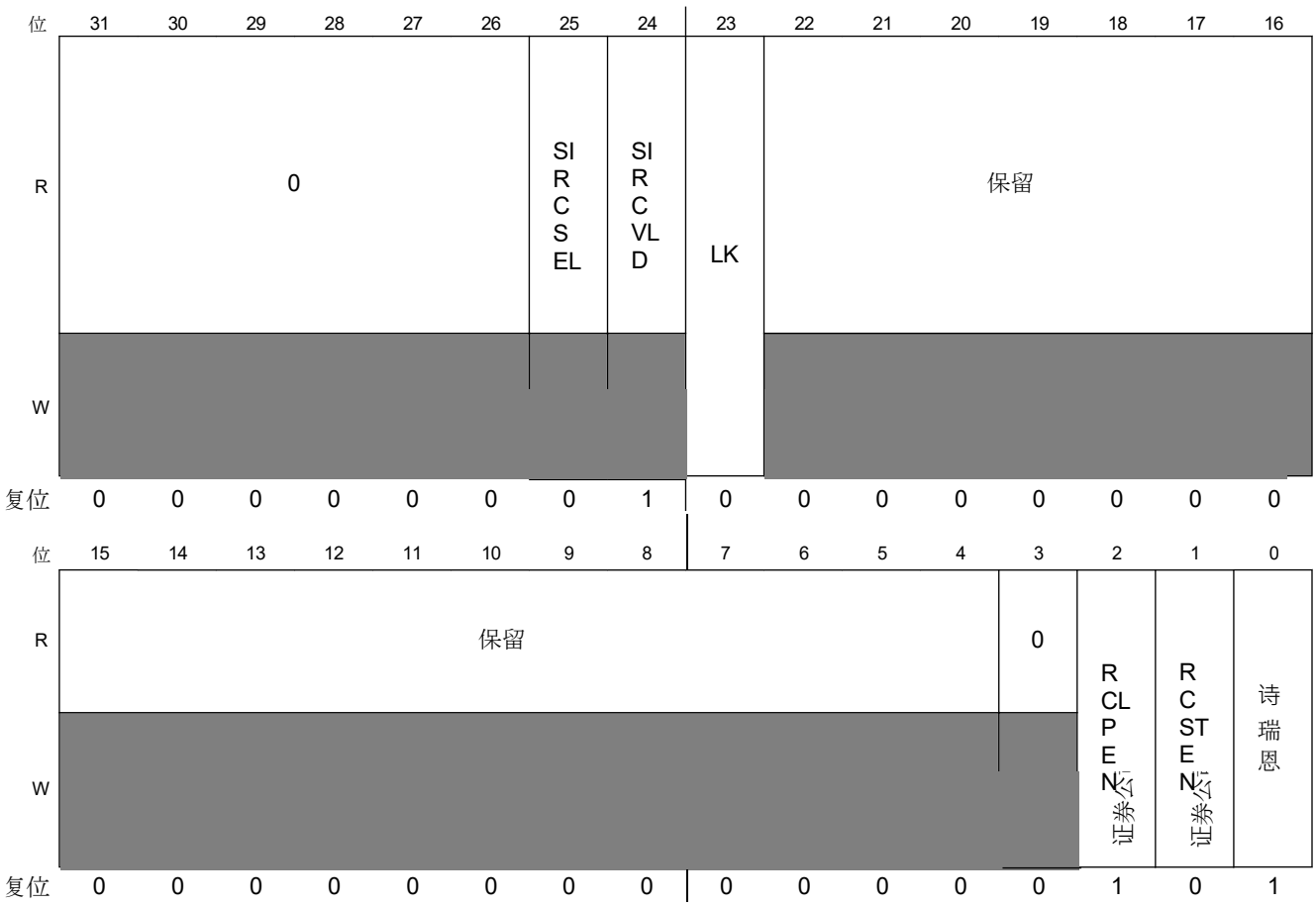
*表继续在下一页...*

## SCG_SOSCCFG字段描述（续）

| 场地 | 描述 |
|---|---|
| | 10 为晶体振荡器选择的中等频率范围<br>11 为晶体振荡器选择的高频率范围 |
| 3<br>HGO | 高增益振荡器选择<br><br>控制晶体振荡器的工作电源模式。<br><br>0 配置晶体振荡器进行低增益操作<br>1 为高增益操作配置晶体振荡器 |
| 2<br>ÅÅÅ | 外部基准选择<br><br>选择外部参考时钟的来源。该位选择从系统振荡器（SOSC）输出到SCG的时钟，因此可以选择晶体振荡器或从外部时钟输入。<br><br>0 选择外部参考时钟<br>1 选择OSC的内部晶体振荡器。 |
| 保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |

## 26.3.11 慢速IRC控制状态寄存器（SCG_SIRCCSR）。

地址。0h基数+200h偏移量=200h




**SCG_SIRCCSR字段描述**

| 场地 | 描述 |
|---|---|
| 31-26<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 25<br>SIRCSEL | 选定的慢速IRC<br><br>0 慢速IRC不是系统时钟源<br>1 慢速IRC是系统时钟源 |
| 24<br>SIRCVLD | 慢速IRC有效<br><br>0 慢速IRC未启用或时钟无效<br>1 慢速IRC被启用，输出时钟有效 |
| 23<br>LK | 锁定寄存器<br><br>这个位域可以在任何时候被清除/设置。 |

表继续在下一页...

**SCG_SIRCCSR字段描述（续）**

| 场地 | 描述 |
|---|---|
| | 0 控制状态寄存器可以被写入。<br>1 控制状态寄存器不能被写入。 |
| 22-4<br>保留 | 这个字段是保留的，其值始终为0。<br><br>这个字段是保留的。 |
| 3<br>保留 | 这个字段是保留的，其值始终为0。<br><br>这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 2<br>辽宁沈阳 | 慢速IRC低功率启用<br><br>0 在VLP模式下，慢速IRC被禁用<br>1 在VLP模式下，慢速IRC被启用 |
| 1<br>循环器 | 慢速IRC停止启用<br><br>0 在支持的停止模式下，慢速IRC被禁用<br>1 慢速IRC在支持的停止模式中被启用 |
| 0<br>诗瑞恩 | 启用慢速IRC<br><br>如果该位在时钟切换期间写入，应在继续进行之前读回并确认。<br><br>0Slow IRC被禁用<br>1慢速IRC已启用 |

## 26.3.12 慢速IRC除法寄存器（SCG_SIRCDIV）。

为了防止输出分频时钟的突变，在禁用慢速IRC时要改变SIRDIV。

地址。0h base + 204h offset = 204h

| 位 | 31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 | | | | | | | | | | | | | | | | 15 14 13 12 11 10 9 | | | | | | | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | | 保留 | | | | | | | 0 | | | | | | | SIRCDIV 2 | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 位 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | SIRCDIV 1 | | |
| W | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**SCG_SIRCDIV字段描述**

| 场地 | 描述 |
|---|---|
| 31-19<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 18-16<br>保留 | 这个字段是保留的。<br>这个位域是保留的。软件应该写0到这个位域以保持兼容性。 |
| 15-11<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |

| | |
|---|---|
| 10-8<br>SIRCDIV2 | 慢速IRC时钟除以2 |

*表继续在下一页...*

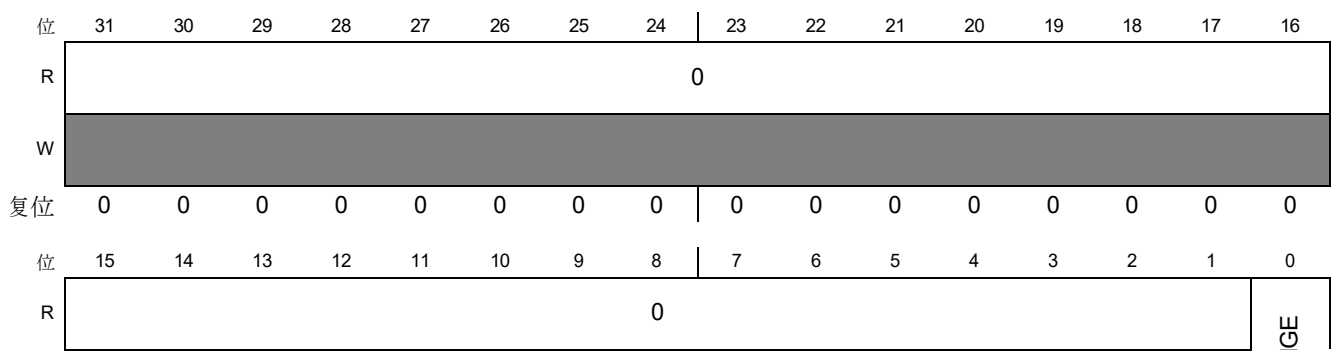
SCG_SIRCDIV字段描述（续）

| 场地 | 描述 |
|---|---|
| | 用于慢速IRC的时钟分频器2。由需要异步时钟源的模块使用。<br><br>00 输出禁用<br>01 除以1<br>10 除以2<br>11 除以4<br>100 除以8<br>101 除以16<br>110 除以32<br>111 除以64 |
| 7-3<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| SIRCDIV1 | 慢速IRC时钟分频器1<br><br>用于慢速IRC的时钟分频器1。用于为需要异步时钟源的模块生成系统时钟源。<br><br>00 输出禁用<br>01 除以1<br>10 除以2<br>11 除以4<br>100 除以8<br>101 除以16<br>110 除以32<br>111 除以64 |

## 26.3.13 慢速IRC配置寄存器（SCG_SIRCCFG）。

当慢速IRC时钟被启用时，SIRCCFG寄存器不能被改变。当慢速IRC时钟被启用时，对该寄存器的写入被忽略，并且没有传输错误。

地址。0h base + 208h offset = 208h



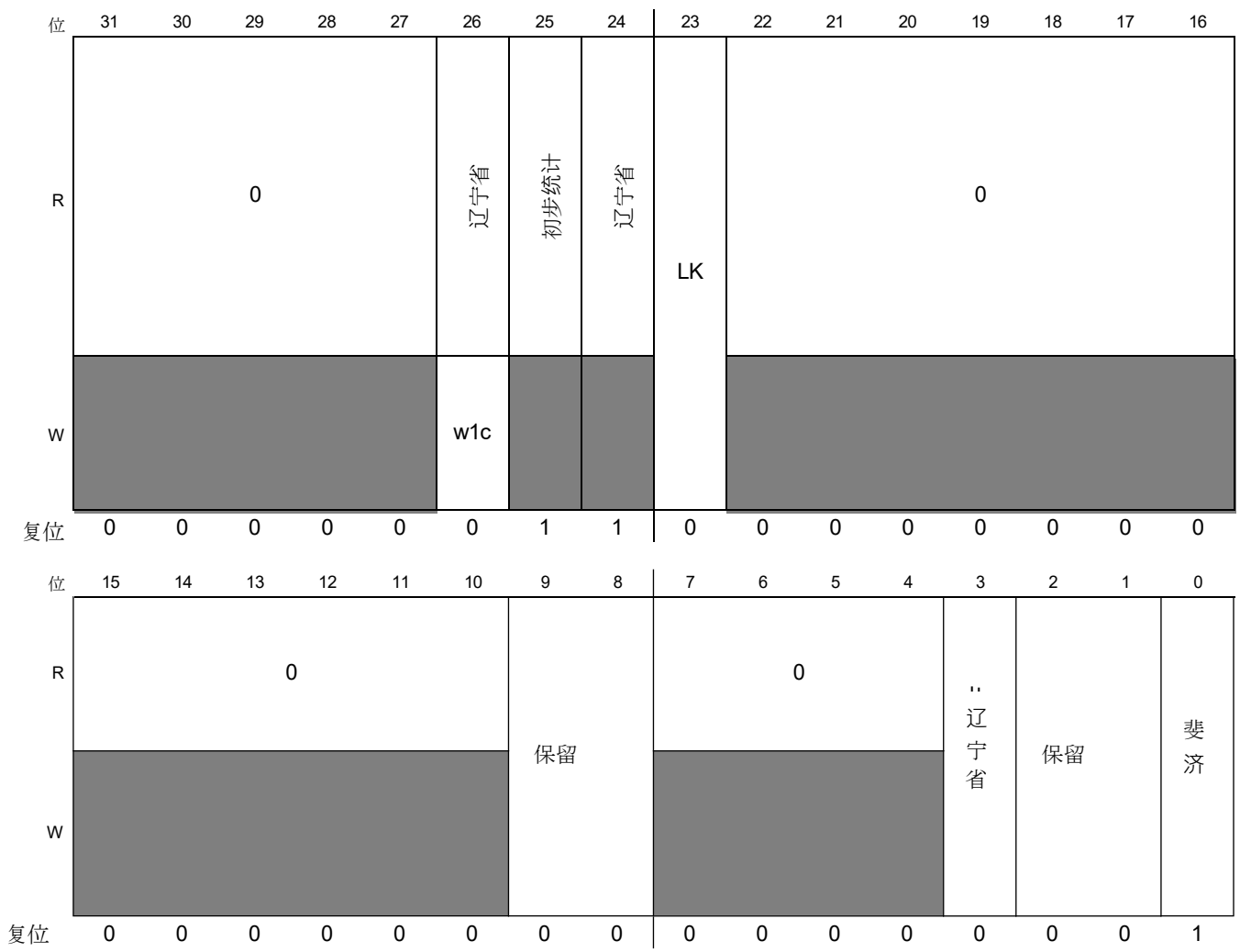

W
RA
复位
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1

**SCG_SIRCCFG字段描述**

| 场地 | 描述 |
|---|---|
| 31-1<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 0<br>范围 | 频率范围<br>**关于支持的频率范围，请参见芯片的具体信息。**<br>0低速IRC低范围时钟(2MHz)<br>1慢速IRC高范围时钟（8MHz）。 |

## 26.3.14 快速IRC控制状态寄存器（SCG_FIRCCSR）。

地址。0h基数+300h偏移量=300h



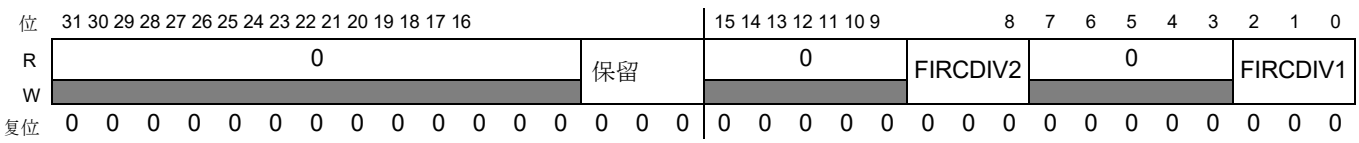
## SCG_FIRCCSR字段描述

| 场地 | 描述 |
|---|---|
| 31-27<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 26<br>辽宁省 | 快速IRC时钟错误<br><br>这个标志只在芯片POR时被重置，软件也可以通过写一个逻辑一来清除这个标志。<br><br>0 在快速IRC修剪中未检测到错误。<br>1 在快速IRC修剪中检测到错误。 |
| 25<br>初步统计 | 快速IRC选定状态<br><br>0 快速IRC不是系统时钟源<br>1 快速IRC是系统时钟源 |
| 24<br>辽宁省 | 快速IRC有效状态<br><br>0 快速IRC未启用或时钟无效。<br>1 快速IRC被启用，输出时钟有效。一旦有来自FIRC模拟的输出时钟，该时钟就有效。 |
| 23<br>LK | 锁定寄存器<br><br>这个位域可以在任何时候被清除/设置。<br><br>0 控制状态寄存器可以被写入。<br>1 控制状态寄存器不能被写入。 |
| 22-10<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 9-8<br>保留 | 这个字段是保留的。<br>这个字段是保留的。软件应将这些位写入0以保持兼容性。 |
| 7-4<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 3<br>基金会 | 快速IRC调节器启用<br><br>0 快速IRC调节器被启用。<br>1 快速IRC调节器被禁用。 |
| 2-1<br>保留 | 这个字段是保留的。软件应该把这些位写成0，以保持兼容性。<br><br>这个字段是保留的。 |
| 0<br>斐济 | 快速IRC启用<br><br>如果该位在时钟切换期间写入，应在继续进行之前读回并确认。<br><br>0 快速IRC被禁用<br>1 启用了快速IRC |

## 26.3.15 快速IRC除法寄存器（SCG_FIRCDIV）。

对FIRCDIV的修改应该在FAST
IRC被禁用时进行，以防止对输出分频时钟的突发事件。

地址。0h基数+304h偏移量=304h




**SCG_FIRCDIV字段描述**

| 场地 | 描述 |
|---|---|
| 31-19<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 18-16<br>保留 | 这个字段是保留的。<br>这个位域是保留的。软件应该写0到这个位域以保持兼容性。 |
| 15-11<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 10-8<br>FIRCDIV2 | 快速IRC时钟分频器2<br>用于快速IRC的时钟分频器2。由需要异步时钟源的模块使用。000输出　禁用<br>001除以1<br>10　除以2<br>11　除以4<br>100　除以8<br>101　除以16<br>110　除以32<br>111　除以64 |
| 7-3<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| FIRCDIV1 | 快速IRC时钟分频器1<br>用于快速IRC的时钟分频器1。用于为需要异步时钟源的模块生成系统时钟源。<br>00　输出禁用<br>01　除以1<br>10　除以2<br>11　除以4<br>100　除以8<br>101　除以16<br>110　除以32<br>111　除以64 |

## 26.3.16 快速IRC配置寄存器（SCG_FIRCCFG）。

当快速IRC被启用时，FIRCCFG寄存器不能被改变。当快速IRC被启用时，对该寄存器的写入被忽略，并且没有传输错误。

地址。0h base + 308h offset = 308h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

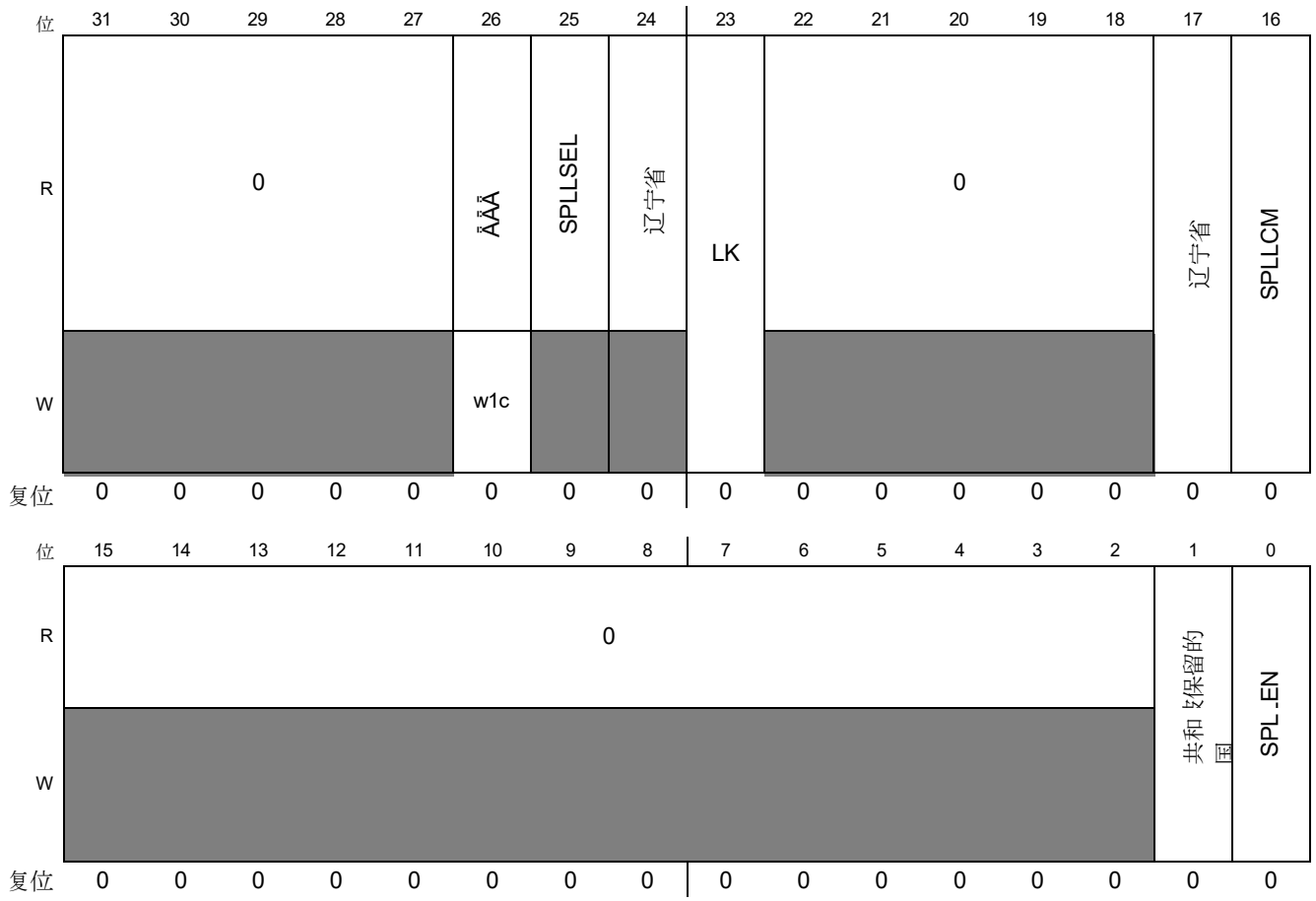
| 位 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | 范围 | |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**SCG_FIRCCFG字段描述**

| 场地 | 描述 |
|---|---|
| 31-2<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 范围 | 频率范围<br>**关于支持的频率范围，见芯片的具体信息。**<br>00快速IRC被微调到48MHz 01快速<br>IRC被微调到52MHz<br>10 快速IRC被修剪到56MHz<br>11 快速IRC被修剪到60MHz |

## 26.3.17 系统PLL控制状态寄存器（SCG_SPLLCSR）。

地址。0h base + 600h offset = 600h




### SCG_SPLLCSR字段描述

| 场地 | 描述 |
|---|---|
| 31-27<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 26<br>ÅÃÅ | 系统PLL时钟错误<br><br>这个标志只在芯片POR时被重置，软件也可以通过写一个逻辑一来清除这个标志。<br><br>0 系统PLL时钟监视器被禁用或未检测到错误<br>1 系统PLL时钟监控器被启用并检测到一个错误。当系统OSC被选为其来源且SOSCERR已设定时，系统PLL时钟错误标志将不被设定。 |
| 25<br>SPLLSEL | 选择的系统PLL<br><br>0 系统PLL不是系统时钟源<br>1 系统PLL是系统时钟源 |

| | |
|---|---|
| 24<br>辽宁省 | 系统PLL有效 |

*表继续在下一页...*

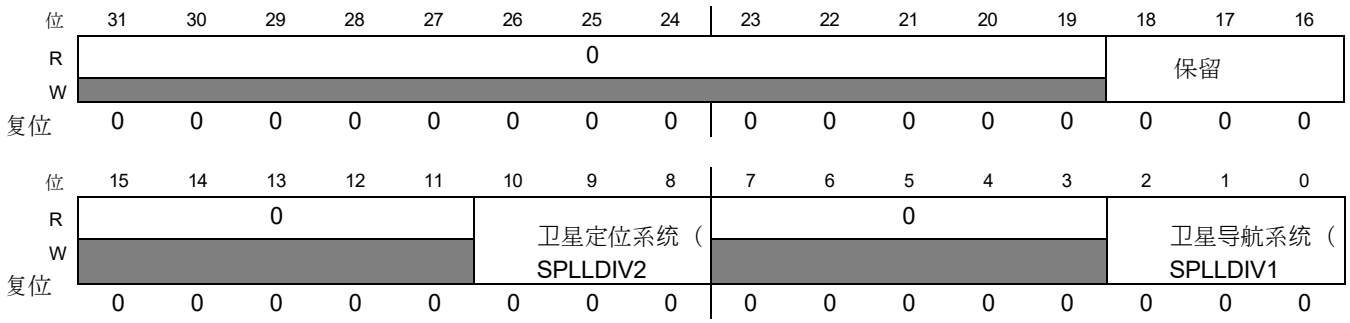
## SCG_SPLLCSR字段描述（续）

| 场地 | 描述 |
|---|---|
| | 表明SPLL时钟何时有效。当选择SPLL作为SPLL的参考时钟时，禁用SPLL或出现SOSC错误将导致SPLLVLD清零而不设置SPLLERROR。<br>锁定检测是由一个锁定检测电路决定的。锁定检测的三个样本决定了时钟是否有效。<br>0 系统PLL未启用或时钟无效<br>1 系统PLL被启用，输出时钟有效 |
| 23<br>LK | 锁定寄存器<br>这个位域可以在任何时候被清除/设置。<br>0 控制状态寄存器可以被写入。<br>1 控制状态寄存器不能被写入。 |
| 22-18<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 17<br>辽宁省 | 系统PLL时钟监控器复位启用<br>0 当检测到错误时，时钟监视器会产生中断<br>1 检测到错误时，时钟监控器产生复位 |
| 16<br>SPLLCM | 系统PLL时钟监控<br>启用时钟监视器，如果时钟源在低功耗模式下被禁用，那么时钟监视器在低功耗模式下也被禁用。当时钟监视器在低功率模式下被禁用时，它将一直被禁用，直到退出低功率模式后时钟有效标志被设置。<br>0 系统PLL时钟监控被禁用<br>1 系统PLL时钟监控被启用 |
| 15-2<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 1<br>保留 | 这个字段是保留的。<br>这个字段是保留的。软件应将此位写入0以保持兼容性。 |
| 0<br>スプリン | 系统PLL启用<br>**注意：**如果该位在时钟切换期间被写入，应在继续前读回并确认。<br>当器件退出复位时，SCG_RCCR 寄存器应根据器件支持的频率范围进行配置，然后再启用SPLL（SPLLEN=1）。<br>0 系统PLL被禁用<br>1 系统PLL被启用 |

## 26.3.18 系统PLL除法寄存器（SCG_SPLLDIV）。

对SPLLDIV的修改应该在系统PLL被禁用时进行，以防止对输出分频时钟的突变。

地址。0h base + 604h offset = 604h




**SCG_SPLLDIV字段描述**

| 场地 | 描述 |
|---|---|
| 31-19<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 18-16<br>保留 | 这个字段是保留的。<br>这个位域是保留的。软件应该写0到这个位域以保持兼容性。 |
| 15-11<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 10-8<br>卫星导航系统（ SPLLDIV2 | 系统PLL时钟分频2<br>系统PLL的时钟分频器2。由需要异步时钟源的模块使用。000Clock disabled<br>001除以1<br>10 除以2<br>11 除以4<br>100 除以8<br>101 除以16<br>110 除以32<br>111 除以64 |
| 7-3<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 卫星导航系统（ SPLLDIV1 | 系统PLL时钟分频1<br>系统PLL的时钟分频器1。用于为需要异步时钟源的模块生成系统时钟源。<br>00 暂停使用时钟<br>01 除以1<br>10 除以2<br>11 除以4<br>100除以8 |

表继续在下一页...

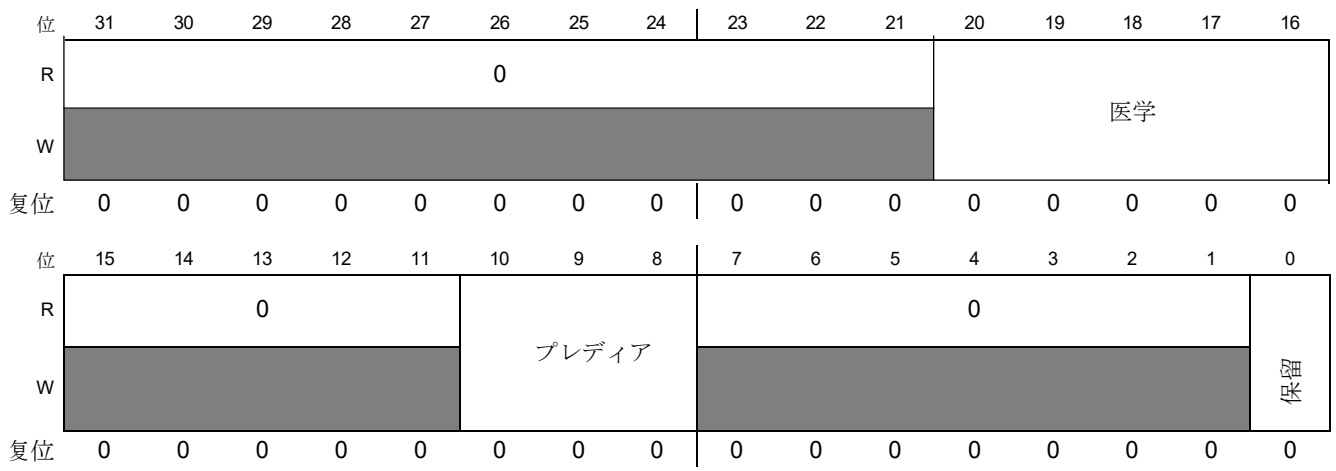
**SCG_SPLLDIV字段描述（续）**

| 场地 | 描述 |
|---|---|
| | 101除以16<br>110 除以32<br>111 除以64 |

## 26.3.19 系统PLL配置寄存器（SCG_SPLLCFG）。

当系统PLL被启用时，SPLLCFG寄存器不能被改变。当系统 PLL 启用时，对该寄存器的写入将被忽略，并且不会出现传输错误。下面的信息适用于VCO_CLK。SPLL_CLK=（VCO_CLK）/2。VCO_CLK = SOSC_CLK/(PREDIV + 1) X (MULT + 16)

地址。0h base + 608h offset = 608h




**SCG_SPLLCFG字段描述**

| 场地 | 描述 |
|---|---|
| 31-21<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 20-16<br>医学 | 系统PLL乘法器<br><br>系统PLL的乘法器。MULT位建立应用于PLL参考时钟频率的乘法系数。 |

**表26-1.PLL VCO倍增系数**

| 医学 | 乘法系数 | | 医学 | 乘法系数 | | 医学 | 乘法系数 | | 医学 | 乘法系数 |
|---|---|---|---|---|---|---|---|---|---|---|
| 00000 | 16 | | 01000 | 24 | | 10000 | 32 | | 11000 | 40 |
| 00001 | 17 | | 01001 | 25 | | 10001 | 33 | | 11001 | 41 |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 00010 | 18 | | 01010 | 26 | | 10010 | 34 | | 11010 | 42 |

*表继续在下一页...*

**SCG_SPLLCFG字段描述（续）**

| 场地 | 描述 |
|---|---|
| | **表26-1.PLL VCO乘法系数（续）**<br>00011 19 01011 27 10011 35 11011 43<br>00100 20 01100 28 10100 36 11100 44<br>00101 21 01101 29 10101 37 11101 45<br>00110 22 01110 30 10110 38 11110 46<br>00111 23 01111 31 10111 39 11111 47 |
| 15-11<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 10-8<br>プレディア | PLL参考时钟分配器<br>选择用于系统PLL的参考时钟的除数。产生的频率必须在数据手册中规定的范围内。<br>**表26-2.系统PLL参考除法系数**<br>プレディア / 除法系数<br>000 / 1<br>001 / 2<br>010 / 3<br>011 / 4<br>100 / 5<br>101 / 6<br>110 / 7<br>111 / 8 |
| 7-1<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 0<br>保留 | 这个字段是保留的。<br>这个字段是保留的。软件应将此位写入0以保持兼容性。 |

# 26.4 职能描述

## 26.4.1 SCG时钟模式转换

下图显示了SCG支持的有效时钟模式转换。该设备不支持慢速IRC（SIRC

）启动模式。

SCG有效模式转换

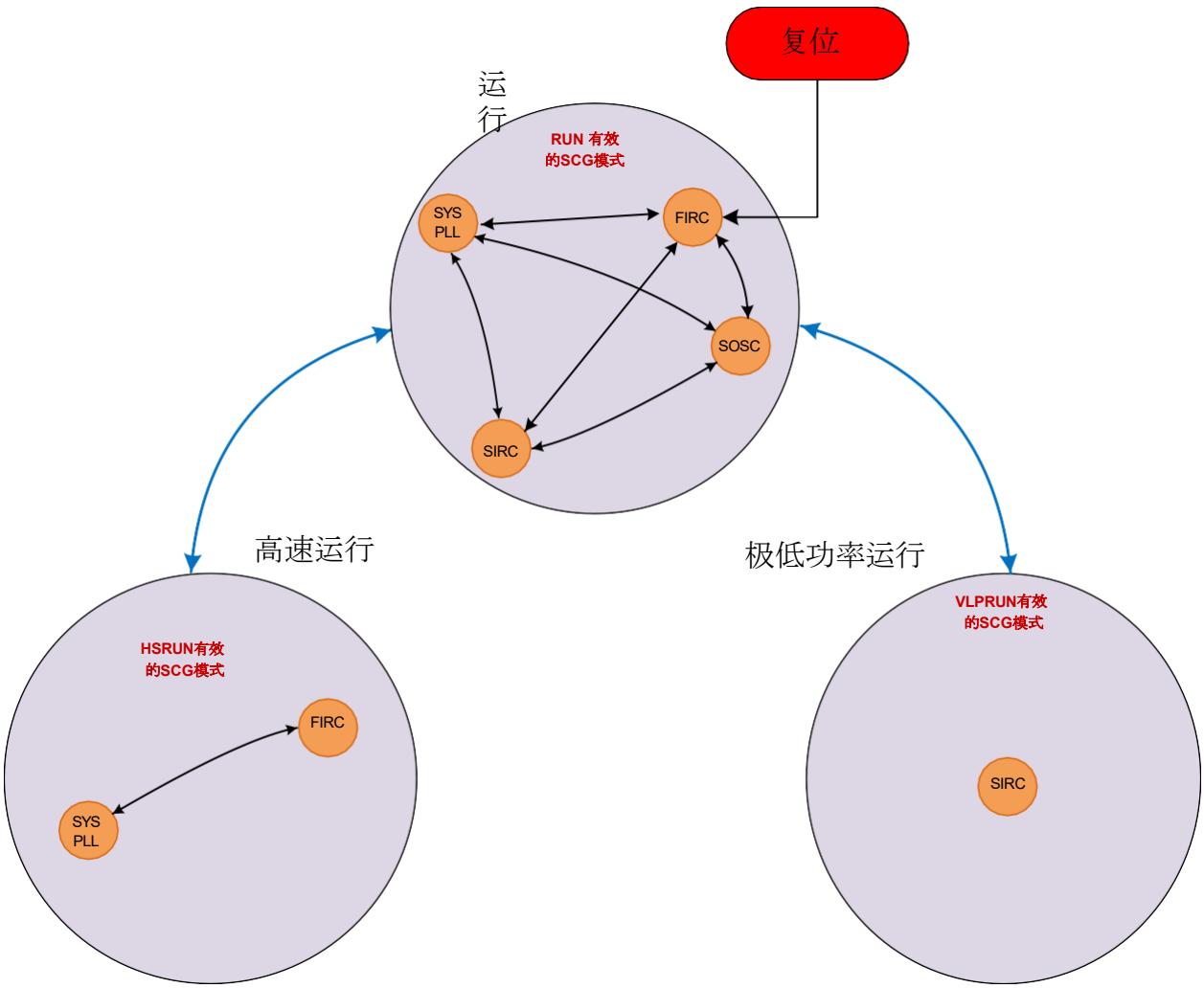



**图26-1.SCG有效模式转换图**

## 注意事项

当需要在运行模式（RUN、HSRUN、VLRUN）之间转换时，SCG应首先完成对SCG时钟控制寄存器中定义的时钟模式的切换。一旦完成了时钟模式的切换，系统就可以启动对所选运行模式的请求。

例如，如果需要从RUN模式过渡到VLRUN，首先完成任何需要的时钟变化。在时钟变化完成后启动VLRUN请求。

电源模式是针对芯片的。关于电源模式分配的更多细节，请参阅电源管理和系统模式控制信息。

下表所列的操作模式是SCG这种实现的有效模式。

**表26-3.SCG的操作模式**

| 模式 | 描述 |
| --- | --- |
| 系统振荡器时钟（SOSC） | 当所有下列条件发生时，进入系统振荡器时钟（SOSC）模式。<br>• RUN MODE：0001被写入RCCR[SCS]。<br>HSRUN MODE：0001被写入HCCR[SCS]。<br>• SOSCEN = 1<br>• SOSCVLD = 1<br><br>在SOSC模式下，SCGCLKOUT和系统时钟来自外部系统振荡器时钟（SOSC）。 |
| 慢速内部参考时钟（SIRC） | 当所有下列条件发生时，进入慢速内部参考时钟（SIRC）模式。<br>• RUN MODE：0010被写入RCCR[SCS]。<br>VLRUN模式：0010被写入VCCR[SCS]，1被写入SIRCCSR[SIRCLPEN]。HSRUN模式：0010被写入HCCR[SCS]。<br>• SIRCEN = 1<br>• SIRCVLD = 1<br><br>在SIRC模式下，SCGCLKOUT和系统时钟来自缓慢的内部参考时钟。如SIRCCFG[RANGE]寄存器定义中所述，SIRC时钟有两个频率范围。当SIRC时钟被启用时，对SIRC范围设置的改变将被忽略。<br>有关SIRC在正常和低功率停止模式下的操作信息可在本表的 "停止 "一行找到。 |
| 快速内部参考时钟（FIRC） | 快速内部参考时钟（FIRC）模式是默认的操作时钟模式，当所有下列条件出现时，就进入该模式。<br>• RUN MODE：0011被写入RCCR[SCS]。<br>HSRUN MODE：0011被写入HCCR[SCS]。<br>• FIRCEN = 1<br>• FIRCVLD = 1<br><br>在FIRC模式下，SCGCLKOUT和系统时钟来自快速内部参考时钟。如FIRC[RANGE]寄存器定义中所述，FIRC时钟有四个频率范围设置。当FIRC时钟被启用时，对FIRC范围设置的改变将被忽略。 |
| 系统PLL (SPLL) | 当所有下列条件发生时，进入系统PLL（SPLL）模式。<br>• RUN MODE：0110被写入RCCR[SCS]。<br>HSRUN MODE：0110被写入HCCR[SCS]。<br>• SPLLEN = 1<br>• SPLLVLD = 1 |

*表继续在下一页...*

## 表26-3.SCG的操作模式（续）

| 模式 | 描述 |
| --- | --- |
| | 在SPLL模式下，SCGCLKOUT和系统时钟来自PLL的输出，PLL由系统振荡器（SOSC）时钟控制。选定的PLL时钟频率锁定在一个乘法系数上，由其对应的SCG_SPLLCFG[MULT]指定，乘以选定的PLL参考频率。必须对PLL的可编程参考分频器进行配置，以产生一个有效的PLL参考时钟。这个除法值由SCG_SPLLCFG[PREDIV]位定义。 |
| 停止 | 只要MCU进入停止状态就会进入。电源模式是针对芯片的。关于电源模式的分配，请参阅描述模块如何配置和停止恢复期间SCG行为的章节。进入停止模式，所有的SCG时钟信号都是静态的，除了以下的时钟可以继续运行，并在以下情况下保持启用。<br>在正常停止和VLPS模式下，当以下所有条件成为现实时，SIRCCLK是可用的。<br>• sirccsr[sircen] = 1<br>• sirccsr[sircsten] = 1<br>• 在VLPS中SIRCCSR[SIRCLPEN] = 1 |

# 第二十七章
# 外围时钟控制器（PCC）

## 27.1 芯片特定的PCC信息

关于特定模块的时钟，见模块时钟。

**注意事项**

在改变外设时钟源时，应禁用相应的模块。

- 时钟应该是CGC门控的
- 如果有的话，模块应通过MDIS位禁用。

在时钟切换后，如果有的话，应使用软复位位（模块软件复位位）对模块进行软复位。

### 27.1.1 芯片特定的PCC寄存器信息

PCC内存图显示了为S32K1xx系列实现的寄存器的超集。在一个特定的变体中不可用的实例的寄存器被保留。

## 27.2 简介

外围时钟控制（PCC）模块为片上外设提供时钟控制和配置。每个外设都有自己的时钟控制和配置寄存器。

## 27.3 特点

PCC模块使软件可以为每个外设配置以下时钟选项。

- 接口时钟门控
- 功能性时钟源选择
- 功能性时钟分频值

下面是PCC模块的方框图。

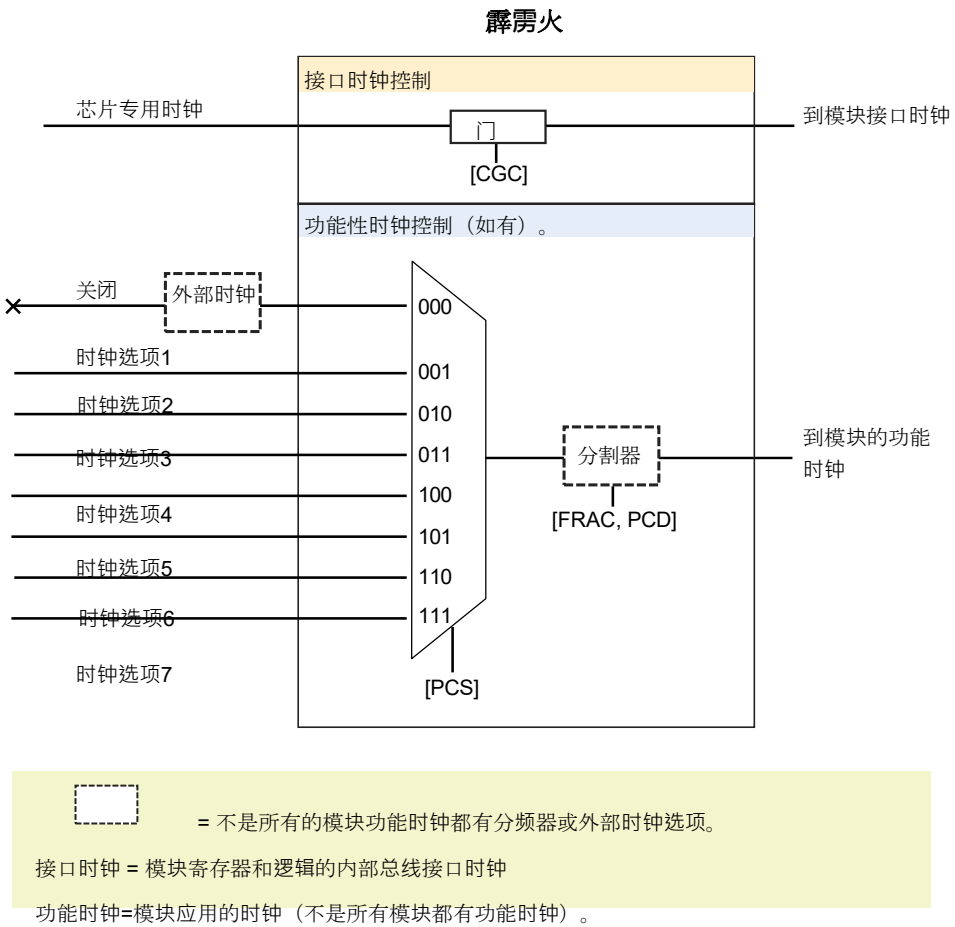



**图27-1.PCC方框图**

## 27.4 职能描述

PCC模块为片上外设（模块）提供自己的专用PCC寄存器，用于时钟门控和配置选项。每个模块的PCC寄存器包含一个用于模块接口时钟的时钟门控位（CGC）。在使用一个模块之前，其接口时钟必须在模块的PCC寄存器中被启用（CGC=1）。

如果一个模块有一个功能时钟，其PCC寄存器可以提供时钟源的选项，通过编程外围时钟选择（PCS）字段进行选择。可选的是，一个模块也可以有一个时钟分频器，通过编程外围时钟分频器（PCD）字段和分数（FRAC）字段来选择。在配置功能时钟之前，模块的接口时钟必须被禁用（CGC=0）。

## 27.5 内存图和寄存器定义

每个模块都有自己的专用PCC寄存器，它控制该特定模块的时钟门控、时钟源和分频器（如适用）。详情见每个模块的PCC寄存器。

PCC寄存器只能在监督员模式下被写入。

**注意事项**

要配置一个特定模块可用的时钟选项或修改现有的配置，首先将模块的CGC位写成0，禁用模块的接口时钟。

## 27.6 PCC寄存器描述

### 27.6.1 PCC内存地图

PCC的基本地址。4006_5000h

| 偏移 | 注册 | 宽度 (单位：比特) | 访问 | 重置值 |
|---|---|---|---|---|
| 80h | PCC FTFC寄存器（PCC_FTFC）。 | 32 | RW | C000_0000h |
| 84h | PCC DMAMUX 寄存器（PCC_DMAMUX）。 | 32 | RW | 8000_0000h |
| 90h | PCC FlexCAN0寄存器（PCC_FlexCAN0） | 32 | RW | 8000_0000h |
| 94h | PCC FlexCAN1寄存器（PCC_FlexCAN1）。 | 32 | RW | 8000_0000h |
| 98h | PCC FTM3寄存器（PCC_FTM3）。 | 32 | RW | 8000_0000h |
| 9Ch | PCC ADC1寄存器（PCC_ADC1）。 | 32 | RW | 8000_0000h |
| ACh | PCC FlexCAN2寄存器（PCC_FlexCAN2）。 | 32 | RW | 8000_0000h |
| B0h | PCC LPSPI0寄存器（PCC_LPSPI0）。 | 32 | RW | 8000_0000h |

*表继续在下一页...*

| | | | | |
|---|---|---|---|---|
| B4h | PCC LPSPI1寄存器（PCC_LPSPI1）。 | 32 | RW | 8000_0000h |
| B8h | PCC LPSPI2寄存器（PCC_LPSPI2）。 | 32 | RW | 8000_0000h |

| 偏移 | 注册 | 宽度<br>(单位：比特) | 访问 | 重置值 |
|---|---|---|---|---|
| C4h | PCC PDB1寄存器（PCC_PDB1）。 | 32 | RW | 8000_0000h |
| C8h | PCC CRC寄存器（PCC_CRC）。 | 32 | RW | 8000_0000h |
| D8h | PCC PDB0寄存器（PCC_PDB0）。 | 32 | RW | 8000_0000h |
| DCh | PCC LPIT寄存器（PCC_LPIT）。 | 32 | RW | 8000_0000h |
| E0h | PCC FTM0寄存器（PCC_FTM0）。 | 32 | RW | 8000_0000h |
| E4h | PCC FTM1寄存器（PCC_FTM1）。 | 32 | RW | 8000_0000h |
| E8h | PCC FTM2寄存器（PCC_FTM2）。 | 32 | RW | 8000_0000h |
| ECh | PCC ADC0寄存器（PCC_ADC0）。 | 32 | RW | 8000_0000h |
| F4h | PCC RTC寄存器（PCC_RTC）。 | 32 | RW | 8000_0000h |
| 100h | PCC LPTMR0寄存器（PCC_LPTMR0）。 | 32 | RW | 8000_0000h |
| 124h | PCC PORTA寄存器（PCC_PORTA）。 | 32 | RW | 8000_0000h |
| 128h | PCC PORTB寄存器（PCC_PORTB） | 32 | RW | 8000_0000h |
| 12Ch | PCC PORTC寄存器（PCC_PORTC）。 | 32 | RW | 8000_0000h |
| 130h | PCC PORTD寄存器（PCC_PORTD）。 | 32 | RW | 8000_0000h |
| 134h | PCC PORTE寄存器（PCC_PORTE）。 | 32 | RW | 8000_0000h |
| 150h | PCC SAI0寄存器（PCC_SAI0）。 | 32 | RW | 8000_0000h |
| 154h | PCC SAI1寄存器（PCC_SAI1）。 | 32 | RW | 8000_0000h |
| 168h | PCC FlexIO寄存器（PCC_FlexIO）。 | 32 | RW | 8000_0000h |
| 184h | PCC EWM寄存器（PCC_EWM）。 | 32 | RW | 8000_0000h |
| 198h | PCC LPI2C0寄存器（PCC_LPI2C0）。 | 32 | RW | 8000_0000h |
| 19Ch | PCC LPI2C1寄存器（PCC_LPI2C1）。 | 32 | RW | 8000_0000h |
| 1A8h | PCC LPUART0寄存器（PCC_LPUART0） | 32 | RW | 8000_0000h |
| 1ACh | PCC LPUART1寄存器（PCC_LPUART1）。 | 32 | RW | 8000_0000h |
| 1B0h | PCC LPUART2寄存器（PCC_LPUART2）。 | 32 | RW | 8000_0000h |
| 1B8h | PCC FTM4寄存器（PCC_FTM4）。 | 32 | RW | 8000_0000h |
| 1BCh | PCC FTM5寄存器（PCC_FTM5）。 | 32 | RW | 8000_0000h |
| 1C0h | PCC FTM6寄存器（PCC_FTM6）。 | 32 | RW | 8000_0000h |
| 1C4h | PCC FTM7寄存器（PCC_FTM7）。 | 32 | RW | 8000_0000h |
| 1CCh | PCC CMP0寄存器（PCC_CMP0）。 | 32 | RW | 8000_0000h |
| 1D8h | PCC QSPI寄存器（PCC_QSPI）。 | 32 | RW | 8000_0000h |
| 1E4h | PCC ENET寄存器（PCC_ENET）。 | 32 | RW | 8000_0000h |

## 27.6.2 PCC FTFC寄存器（PCC_FTFC）。

*表继续在下一页...*

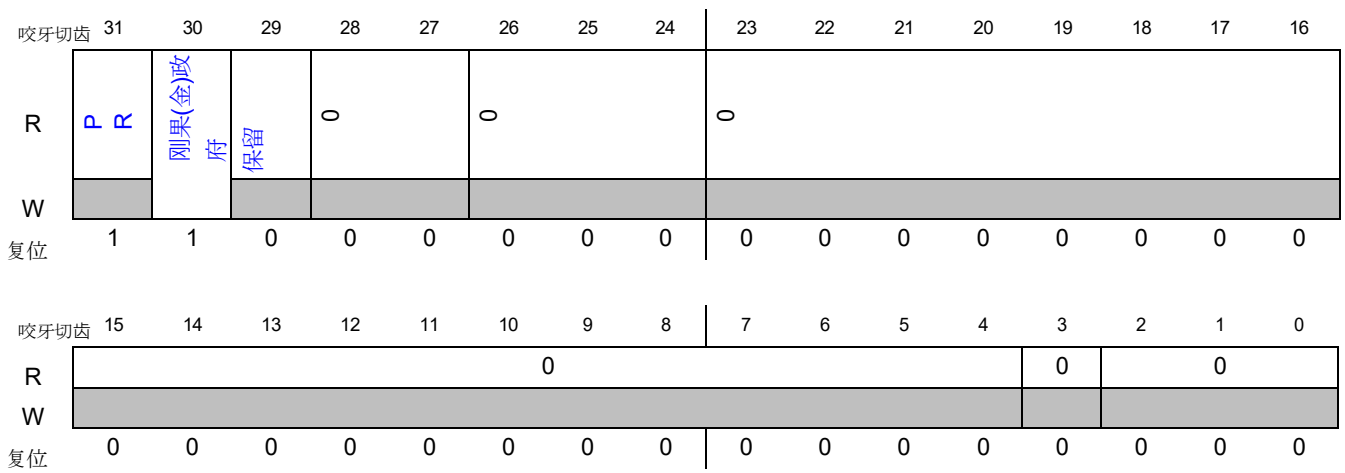
## 27.6.2.1 偏移

| 注册 | 偏移 |
|---|---|
| PCC_FTFC | 80h |

## 27.6.2.2 职能

PCC登记簿

## 27.6.2.3 图示




## 27.6.2.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>公关 | 目前<br>该位显示外设是否存在于该设备上。<br>0b - 周边设备不存在。1b - 外围存在。 |

| | |
|---|---|
| 30<br>刚果(金)政府 | 时钟门控制<br>该读/写位启用外设的时钟。<br>0b - 禁用时钟<br>1b - 时钟启用。当前的时钟选择和分频器选项被锁定。 |
| 29<br>- | 这个只读的位域是保留的。这个位可以改变数值，但却是不关心的。 |
| 28-27 | 这个只读的位域是保留的，其值始终为0。 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| - | |
| 26-24<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 23-4<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 3<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 2-0<br>- | 这个只读的位域是保留的，其值始终为0。 |

## 27.6.3 PCC DMAMUX 寄存器（PCC_DMAMUX）。

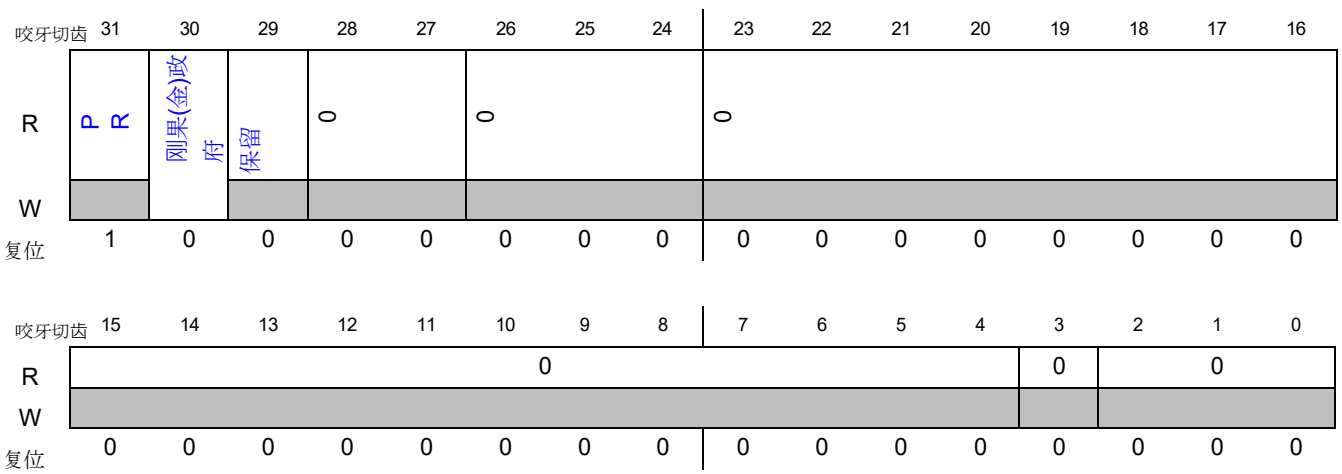
### 27.6.3.1 偏移

| 注册 | 偏移 |
|---|---|
| PCC_DMAMUX | 84h |

### 27.6.3.2 职能

PCC登记簿

### 27.6.3.3 图示



### 27.6.3.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>公关 | 目前<br>该位显示外设是否存在于该设备上。<br>0b - 周边设备不存在。1b - 外围存在。 |
| 30<br>刚果(金)政府 | 时钟门控制<br>该读/写位启用外设的时钟。<br>0b - 禁用时钟<br>1b - 时钟启用。当前的时钟选择和分频器选项被锁定。 |
| 29<br>- | 这个只读的位域是保留的。这个位可以改变数值，但却是不关心的。 |
| 28-27<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 26-24<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 23-4<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 3<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 2-0<br>- | 这个只读的位域是保留的，其值始终为0。 |

## 27.6.4 PCC FlexCAN0寄存器（PCC_FlexCAN0）

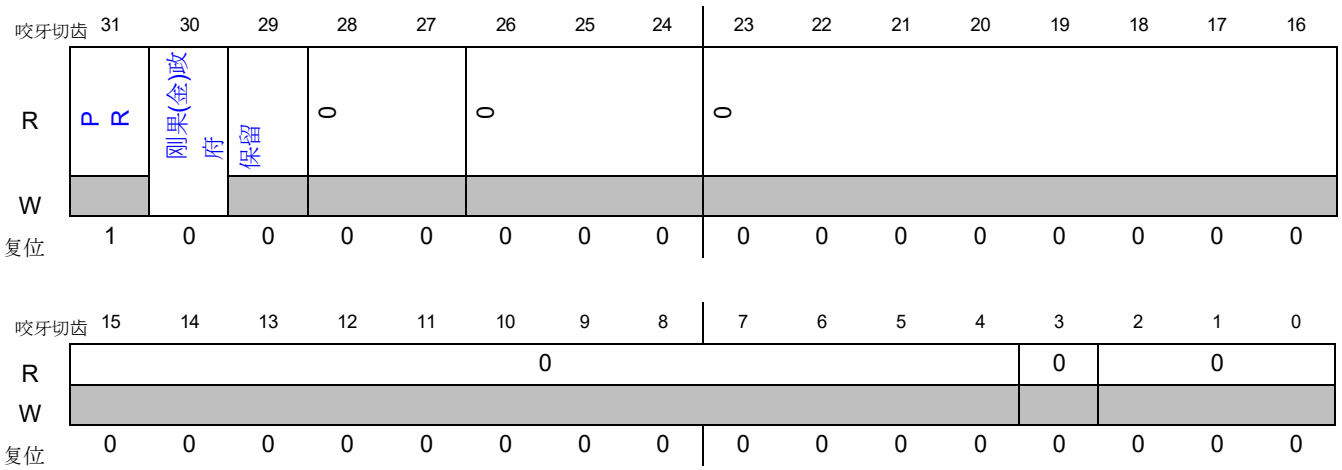
### 27.6.4.1 偏移

| 注册 | 偏移 |
|---|---|
| PCC_FlexCAN0 | 90h |

### 27.6.4.2 职能

# PCC登记簿

### 27.6.4.3 图示




### 27.6.4.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>公关 | 目前<br>该位显示外设是否存在于该设备上。<br>0b - 周边设备不存在。1b - 外围存在。 |
| 30<br>刚果(金)政府 | 时钟门控制<br>该读/写位启用外设的时钟。<br>0b - 禁用时钟<br>1b - 时钟启用。当前的时钟选择和分频器选项被锁定。 |
| 29<br>- | 这个只读的位域是保留的。这个位可以改变数值，但却是不关心的。 |
| 28-27<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 26-24<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 23-4<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 3<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 2-0<br>- | 这个只读的位域是保留的，其值始终为0。 |

# 27.6.5 PCC FlexCAN1寄存器（PCC_FlexCAN1）。

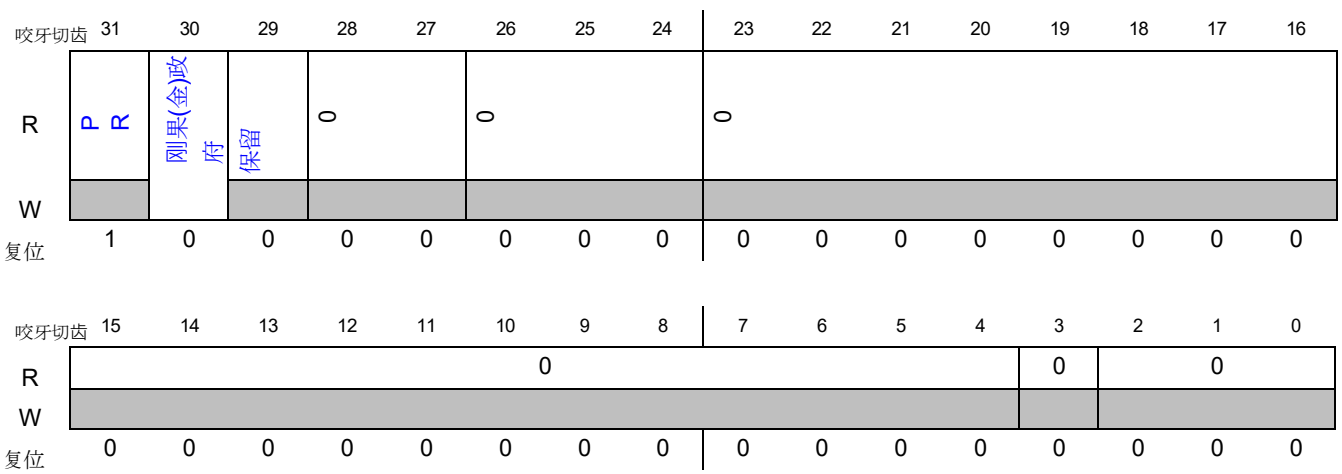
## 27.6.5.1 偏移

| 注册 | 偏移 |
|---|---|
| PCC_FlexCAN1 | 94h |

## 27.6.5.2 职能

PCC登记簿

## 27.6.5.3 图示




## 27.6.5.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>公关 | 目前<br>该位显示外设是否存在于该设备上。<br>0b - 周边设备不存在。1b - 外围存在。 |

| | |
|---|---|
| 30<br>刚果(金)政府 | 时钟门控制<br>该读/写位启用外设的时钟。 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | 0b - 禁用时钟<br>1b - 时钟启用。当前的时钟选择和分频器选项被锁定。 |
| 29<br>- | 这个只读的位域是保留的。这个位可以改变数值，但却是不关心的。 |
| 28-27<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 26-24<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 23-4<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 3<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 2-0<br>- | 这个只读的位域是保留的，其值始终为0。 |

## 27.6.6 PCC FTM3寄存器（PCC_FTM3）。

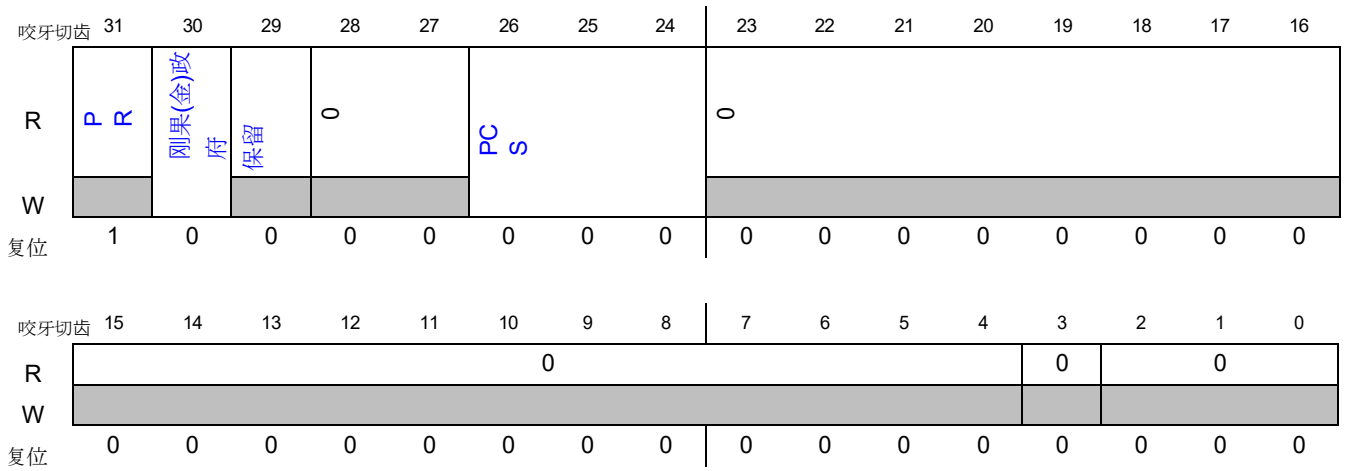
### 27.6.6.1 偏移

| 注册 | 偏移 |
| --- | --- |
| PCC_FTM3 | 98h |

### 27.6.6.2 职能

PCC登记簿

### 27.6.6.3 图示




### 27.6.6.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>公关 | 目前<br>该位显示外设是否存在于该设备上。<br>0b - 周边设备不存在。1b - 外围存在。 |
| 30<br>刚果(金)政府 | 时钟门控制<br>该读/写位启用外设的时钟。<br>0b - 禁用时钟<br>1b - 时钟启用。当前的时钟选择和分频器选项被锁定。 |
| 29<br>- | 这个只读的位域是保留的。这个位可以改变数值，但却是不关心的。 |
| 28-27<br>- | 这个只读的位域是保留的，其值始终为0。 |

| | |
|---|---|
| 26-24<br><br>辽宁省 | 周边时钟源选择<br><br>这个读/写位域用于支持各种时钟选择的外设。该字段只有在时钟被禁用（CGC=0）<br><br>时才能被写入。<br><br>000b - 时钟关闭。可以为这个外设启用一个外部时钟。001b - 时钟选项 1<br>010b - 时钟选项2<br>011b - 时钟选项3<br>100b - 时钟选项4<br>101b - 时钟选项5<br>110b - 时钟选项6<br>111b - 时钟选项7 |
| 23-4 | 这个只读的位域是保留的，其值始终为0。 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| - | |
| 3<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 2-0<br>- | 这个只读的位域是保留的，其值始终为0。 |

## 27.6.7 PCC ADC1寄存器（PCC_ADC1）。

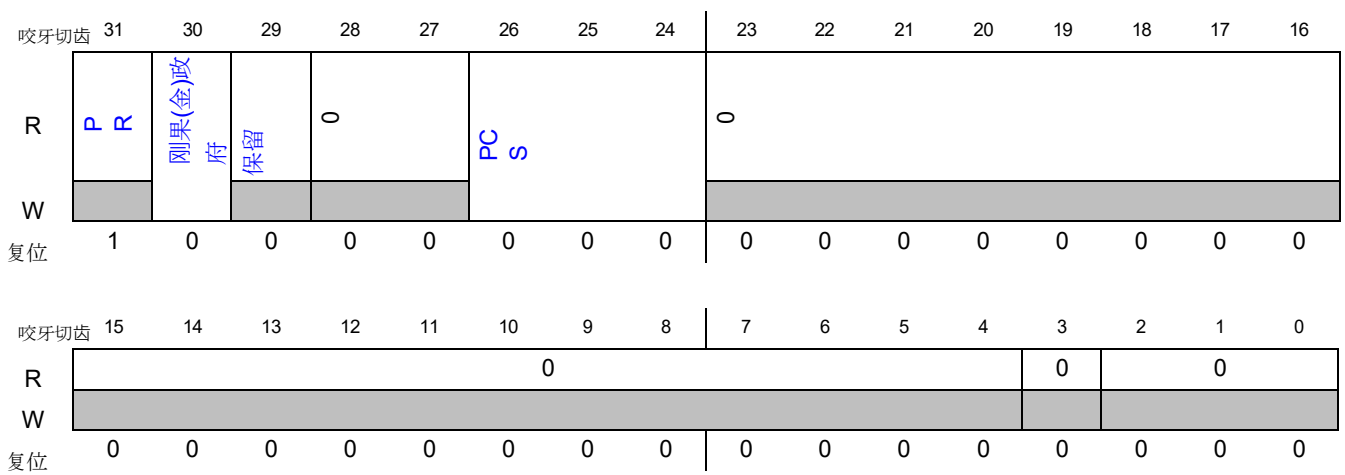
### 27.6.7.1 偏移

| 注册 | 偏移 |
|---|---|
| PCC_ADC1 | 9Ch |

### 27.6.7.2 职能

PCC登记簿

### 27.6.7.3 图示



### 27.6.7.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>公关 | 目前<br>该位显示外设是否存在于该设备上。<br>0b - 周边设备不存在。1b - 外围存在。 |
| 30<br>刚果(金)政府 | 时钟门控制<br>该读/写位启用外设的时钟。<br>0b - 禁用时钟<br>1b - 时钟启用。当前的时钟选择和分频器选项被锁定。 |
| 29<br>- | 这个只读的位域是保留的。这个位可以改变数值，但却是不关心的。 |
| 28-27<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 26-24<br>辽宁省 | 周边时钟源选择<br>这个读/写位域用于支持各种时钟选择的外设。该字段只有在时钟被禁用（CGC=0）时才能被写入。<br>000b - 时钟已关闭。<br>001b - 时钟选项1<br>010b - 时钟选项2<br>011b - 时钟选项3<br>100b - 时钟选项4<br>101b - 时钟选项5<br>110b - 时钟选项6<br>111b - 时钟选项7 |
| 23-4<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 3<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 2-0<br>- | 这个只读的位域是保留的，其值始终为0。 |

## 27.6.8 PCC FlexCAN2寄存器（PCC_FlexCAN2）。

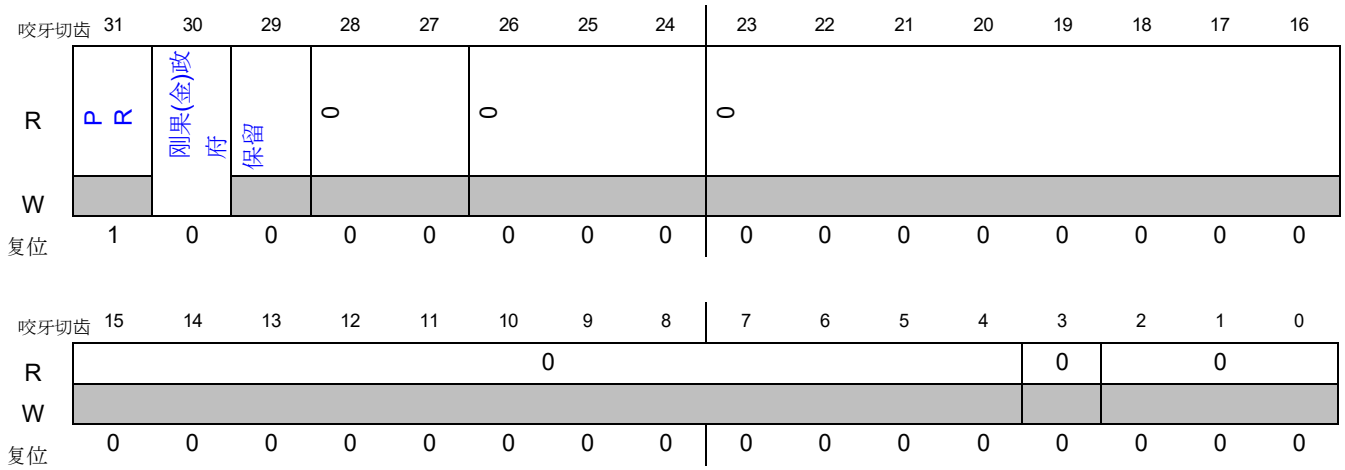
### 27.6.8.1 偏移

| 注册 | 偏移 |
| --- | --- |
| PCC_FlexCAN2 | ACh |

## 27.6.8.2 职能

PCC登记簿

## 27.6.8.3 图示




## 27.6.8.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>公关 | 目前<br>该位显示外设是否存在于该设备上。<br>0b - 周边设备不存在。1b - 外围存在。 |
| 30<br>刚果(金)政府 | 时钟门控制<br>该读/写位启用外设的时钟。<br>0b - 禁用时钟<br>1b - 时钟启用。当前的时钟选择和分频器选项被锁定。 |
| 29<br>- | 这个只读的位域是保留的。这个位可以改变数值，但却是不关心的。 |
| 28-27<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 26-24<br>- | 这个只读的位域是保留的，其值始终为0。 |

| | |
|---|---|
| 23-4<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 3 | 这个只读的位域是保留的，其值始终为0。 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| - | |
| 2-0<br>- | 这个只读的位域是保留的，其值始终为0。 |

## 27.6.9 PCC LPSPI0寄存器（PCC_LPSPI0）。

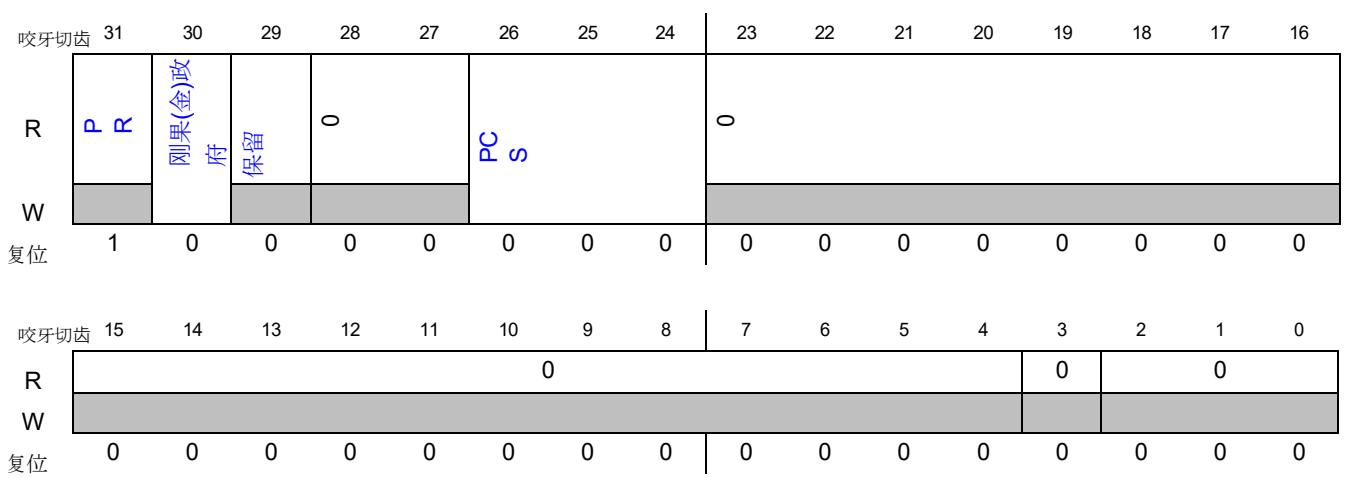
### 27.6.9.1 偏移

| 注册 | 偏移 |
|---|---|
| PCC_LPSPI0 | B0h |

### 27.6.9.2 职能

PCC登记簿

### 27.6.9.3 图示




### 27.6.9.4 领域

| 场地 | 职能 |
| --- | --- |
| 31 | 目前 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| 公关 | 该位显示外设是否存在于该设备上。<br>0b - 周边设备不存在。1b - 外围存在。 |
| 30<br>刚果(金)政府 | 时钟门控制<br>该读/写位启用外设的时钟。<br>0b - 禁用时钟<br>1b - 时钟启用。当前的时钟选择和分频器选项被锁定。 |
| 29<br>- | 这个只读的位域是保留的。这个位可以改变数值，但却是不关心的。 |
| 28-27<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 26-24<br>辽宁省 | 周边时钟源选择<br>这个读/写位域用于支持各种时钟选择的外设。该字段只有在时钟被禁用（CGC=0）时才能被写入。<br>000b - 时钟已关闭。<br>001b - 时钟选项1<br>010b - 时钟选项2<br>011b - 时钟选项3<br>100b - 时钟选项4<br>101b - 时钟选项5<br>110b - 时钟选项6<br>111b - 时钟选项7 |
| 23-4<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 3<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 2-0<br>- | 这个只读的位域是保留的，其值始终为0。 |

## 27.6.10 PCC LPSPI1寄存器（PCC_LPSPI1）。

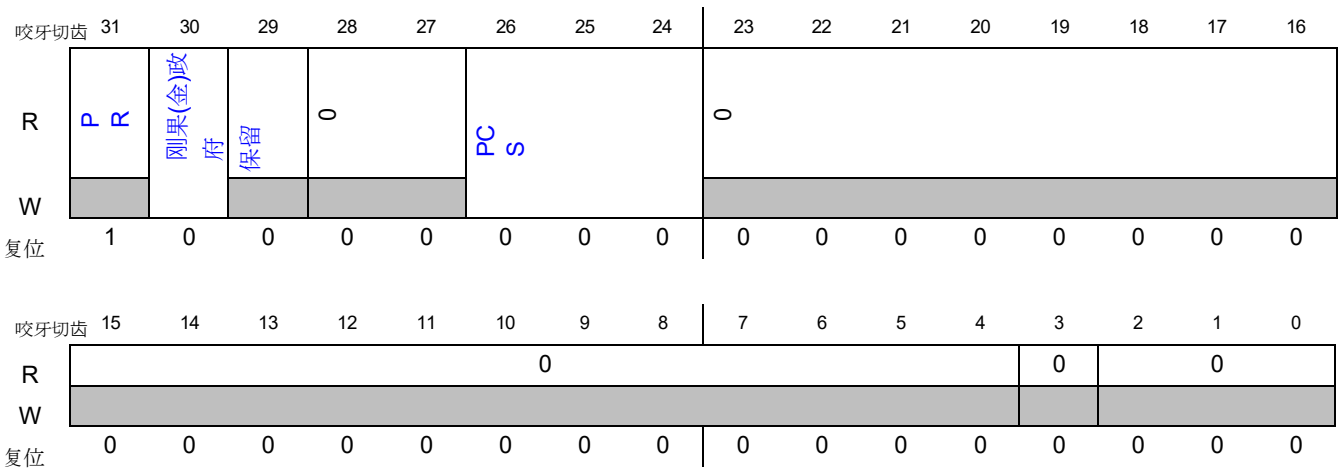
### 27.6.10.1 偏移

| 注册 | 偏移 |
|---|---|
| PCC_LPSPI1 | B4h |

## 27.6.10.2 职能

PCC登记簿

## 27.6.10.3 图示




## 27.6.10.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>公关 | 目前<br>该位显示外设是否存在于该设备上。<br>0b - 周边设备不存在。1b - 外围存在。 |
| 30<br>刚果(金)政府 | 时钟门控制<br>该读/写位启用外设的时钟。<br>0b - 禁用时钟<br>1b - 时钟启用。当前的时钟选择和分频器选项被锁定。 |
| 29<br>- | 这个只读的位域是保留的。这个位可以改变数值，但却是不关心的。 |
| 28-27<br>- | 这个只读的位域是保留的，其值始终为0。 |

| | |
|---|---|
| 26-24<br><br>辽宁省 | 周边时钟源选择<br><br>这个读/写位域用于支持各种时钟选择的外设。该字段只有在时钟被禁用（CGC=0）时<br><br>才能被写入。<br><br>000b - 时钟关闭。<br>001b - 时钟选项1<br>010b - 时钟选项2<br>011b - 时钟选项3 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| | 100b- 时 钟 选 项 4<br>101b- 时 钟 选 项 5<br>110b- 时 钟 选 项 6<br>111b-时钟选项7 |
| 23-4<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 3<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 2-0<br>- | 这个只读的位域是保留的，其值始终为0。 |

## 27.6.11 PCC LPSPI2寄存器（PCC_LPSPI2）。

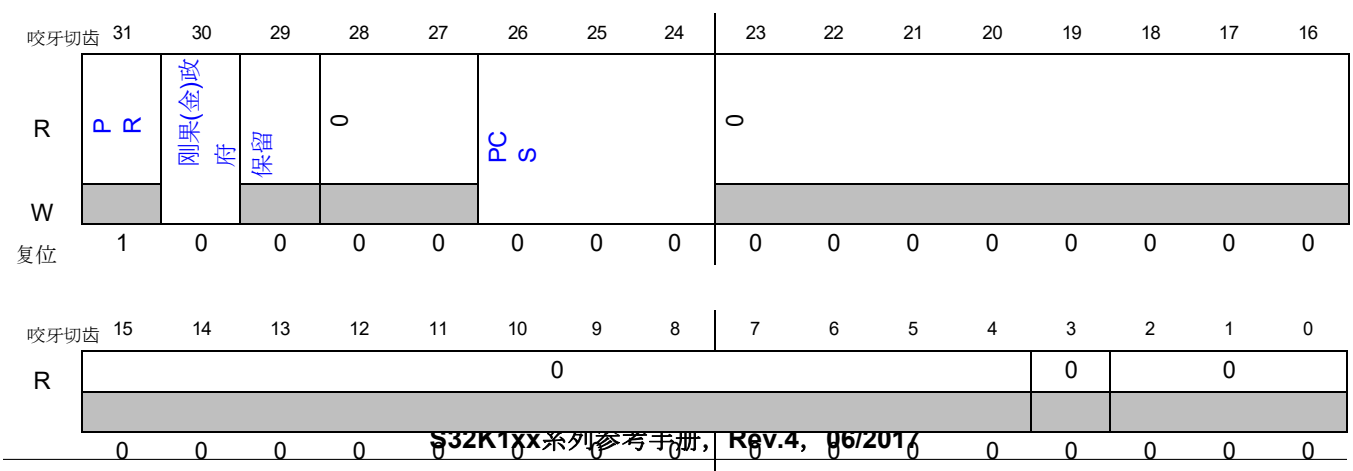
### 27.6.11.1 偏移

| 注册 | 偏移 |
|---|---|
| PCC_LPSPI2 | B8h |

### 27.6.11.2 职能

PCC登记簿

### 27.6.11.3 图示



W

复位

### 27.6.11.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>公关 | 目前<br>该位显示外设是否存在于该设备上。<br>0b - 周边设备不存在。1b - 外围存在。 |
| 30<br>刚果(金)政府 | 时钟门控制<br>该读/写位启用外设的时钟。<br>0b - 禁用时钟<br>1b - 时钟启用。当前的时钟选择和分频器选项被锁定。 |
| 29<br>- | 这个只读的位域是保留的。这个位可以改变数值，但却是不关心的。 |
| 28-27<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 26-24<br>辽宁省 | 周边时钟源选择<br>这个读/写位域用于支持各种时钟选择的外设。该字段只有在时钟被禁用（CGC=0）时才能被写入。<br>000b - 时钟已关闭。<br>001b - 时钟选项1<br>010b - 时钟选项2<br>011b - 时钟选项3<br>100b - 时钟选项4<br>101b - 时钟选项5<br>110b - 时钟选项6<br>111b - 时钟选项7 |
| 23-4<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 3<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 2-0<br>- | 这个只读的位域是保留的，其值始终为0。 |

## 27.6.12 PCC PDB1寄存器（PCC_PDB1）。

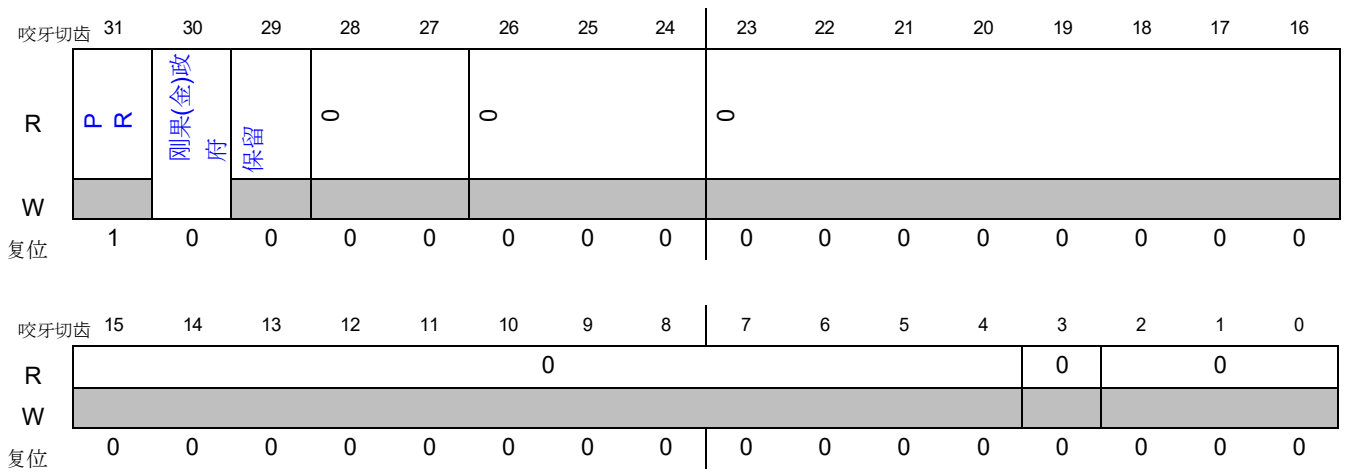
### 27.6.12.1 偏移

| 注册 | 偏移 |
| --- | --- |
| PCC_PDB1 | C4h |

## 27.6.12.2 职能

PCC登记簿

## 27.6.12.3 图示




## 27.6.12.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>公关 | 目前<br>该位显示外设是否存在于该设备上。<br>0b - 周边设备不存在。1b - 外围存在。 |
| 30<br>刚果(金)政府 | 时钟门控制<br>该读/写位启用外设的时钟。<br>0b - 禁用时钟<br>1b - 时钟启用。当前的时钟选择和分频器选项被锁定。 |
| 29<br>- | 这个只读的位域是保留的。这个位可以改变数值，但却是不关心的。 |
| 28-27<br>- | 这个只读的位域是保留的，其值始终为0。 |

**PCC寄存器描述**

| | |
|---|---|
| 26-24<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 23-4<br>- | 这个只读的位域是保留的，其值始终为0。 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| 3<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 2-0<br>- | 这个只读的位域是保留的，其值始终为0。 |

## 27.6.13 PCC CRC寄存器（PCC_CRC）。

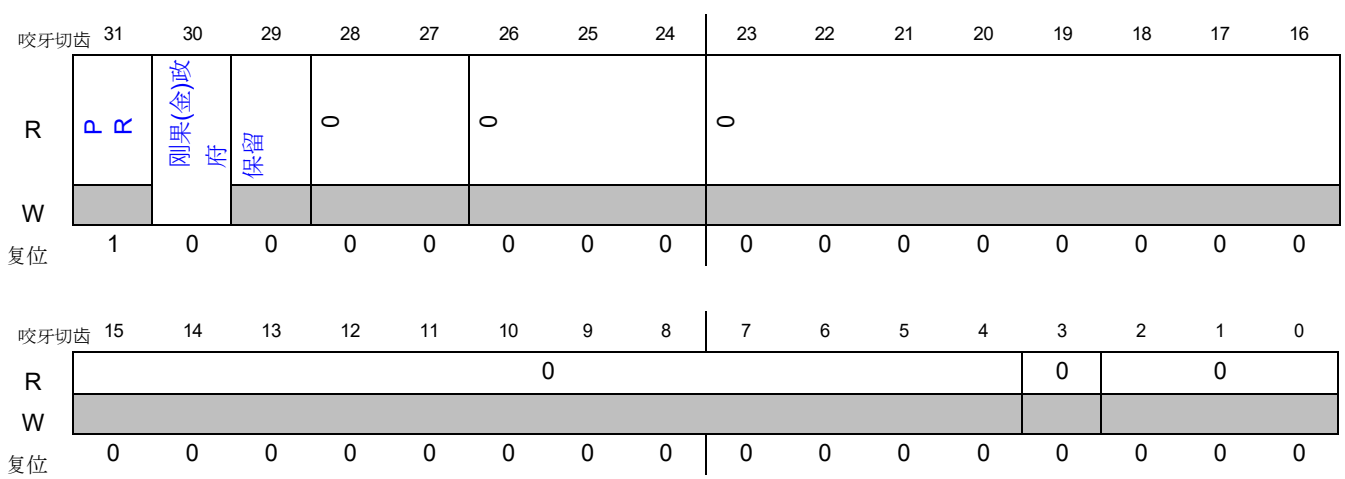
### 27.6.13.1 偏移

| 注册 | 偏移 |
| --- | --- |
| PCC_CRC | C8h |

### 27.6.13.2 职能

PCC登记簿

### 27.6.13.3 图示



### 27.6.13.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>公关 | 目前<br>该位显示外设是否存在于该设备上。<br>0b - 周边设备不存在。1b - 外围存在。 |
| 30<br>刚果(金)政府 | 时钟门控制<br>该读/写位启用外设的时钟。<br>0b - 禁用时钟<br>1b - 时钟启用。当前的时钟选择和分频器选项被锁定。 |
| 29<br>- | 这个只读的位域是保留的。这个位可以改变数值，但却是不关心的。 |
| 28-27<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 26-24<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 23-4<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 3<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 2-0<br>- | 这个只读的位域是保留的，其值始终为0。 |

## 27.6.14 PCC PDB0寄存器（PCC_PDB0）。

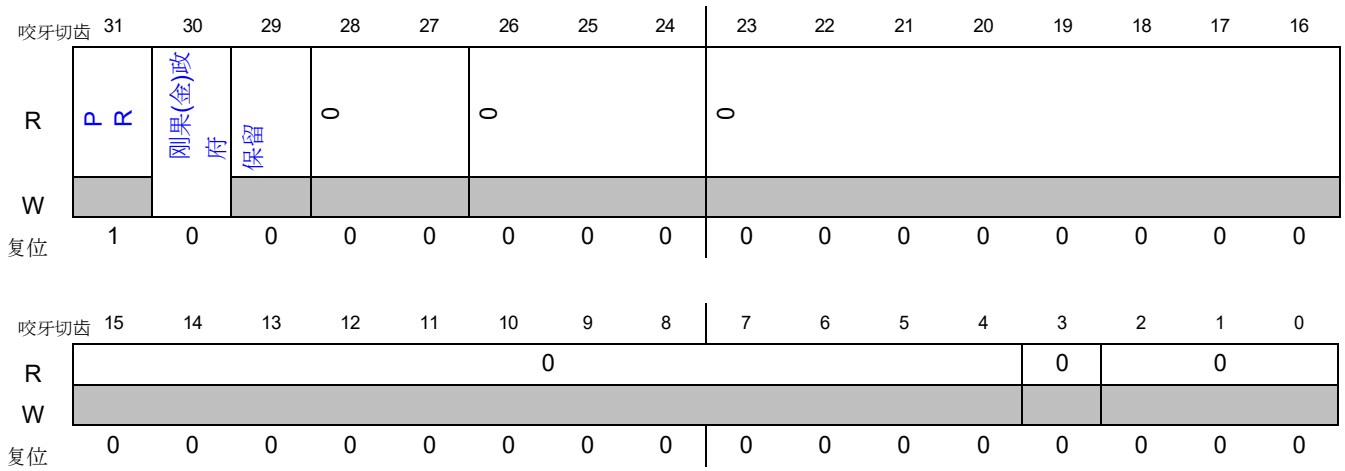
### 27.6.14.1 偏移

| 注册 | 偏移 |
|---|---|
| PCC_PDB0 | D8h |

### 27.6.14.2 职能

# PCC登记簿

## 27.6.14.3 图示




## 27.6.14.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>公关 | 目前<br>该位显示外设是否存在于该设备上。<br>0b - 周边设备不存在。1b - 外围存在。 |
| 30<br>刚果(金)政府 | 时钟门控制<br>该读/写位启用外设的时钟。<br>0b - 禁用时钟<br>1b - 时钟启用。当前的时钟选择和分频器选项被锁定。 |
| 29<br>- | 这个只读的位域是保留的。这个位可以改变数值，但却是不关心的。 |
| 28-27<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 26-24<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 23-4<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 3<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 2-0<br>- | 这个只读的位域是保留的，其值始终为0。 |

# 27.6.15 PCC LPIT寄存器（PCC_LPIT）。

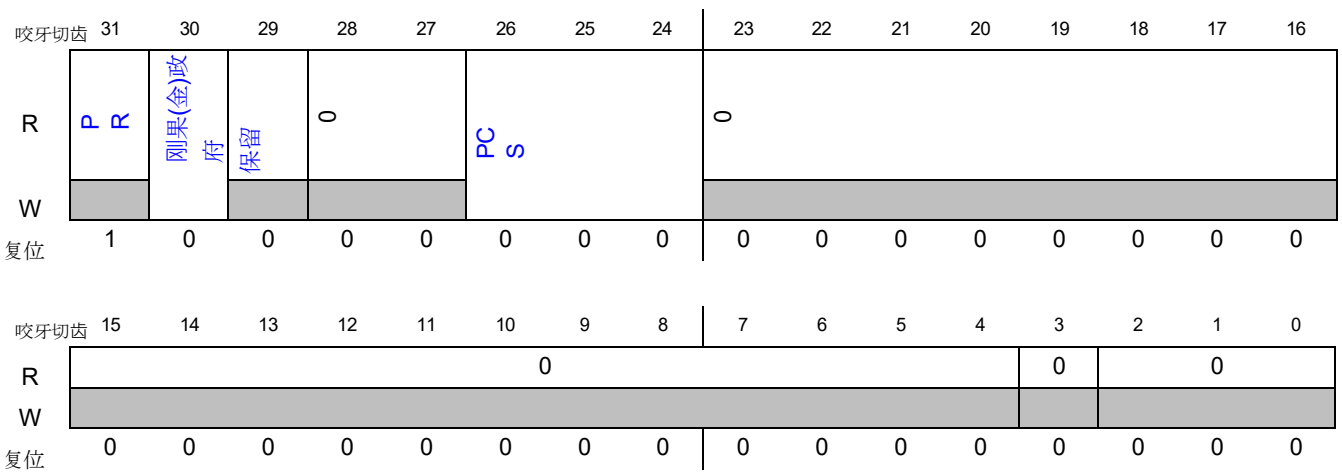
## 27.6.15.1 偏移

| 注册 | 偏移 |
|---|---|
| PCC_LPIT | DCh |

## 27.6.15.2 职能

PCC登记簿

## 27.6.15.3 图示




## 27.6.15.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>公关 | 目前<br>该位显示外设是否存在于该设备上。<br>0b - 周边设备不存在。1b - 外围存在。 |

| | |
|---|---|
| 30<br><br>刚果(金)政府 | 时钟门控制<br><br>该读/写位启用外设的时钟。 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| | 0b - 禁用时钟<br>1b - 时钟启用。当前的时钟选择和分频器选项被锁定。 |
| 29<br>- | 这个只读的位域是保留的。这个位可以改变数值，但却是不关心的。 |
| 28-27<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 26-24<br>辽宁省 | 周边时钟源选择<br>这个读/写位域用于支持各种时钟选择的外设。<br>这个字段只有在时钟被禁用（CGC=0）时才能写入。<br>000b - 时钟已关闭。<br>001b - 时钟选项1<br>010b - 时钟选项2<br>011b - 时钟选项3<br>100b - 时钟选项4<br>101b - 时钟选项5<br>110b - 时钟选项6<br>111b - 时钟选项7 |
| 23-4<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 3<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 2-0<br>- | 这个只读的位域是保留的，其值始终为0。 |

## 27.6.16 PCC FTM0寄存器（PCC_FTM0）。

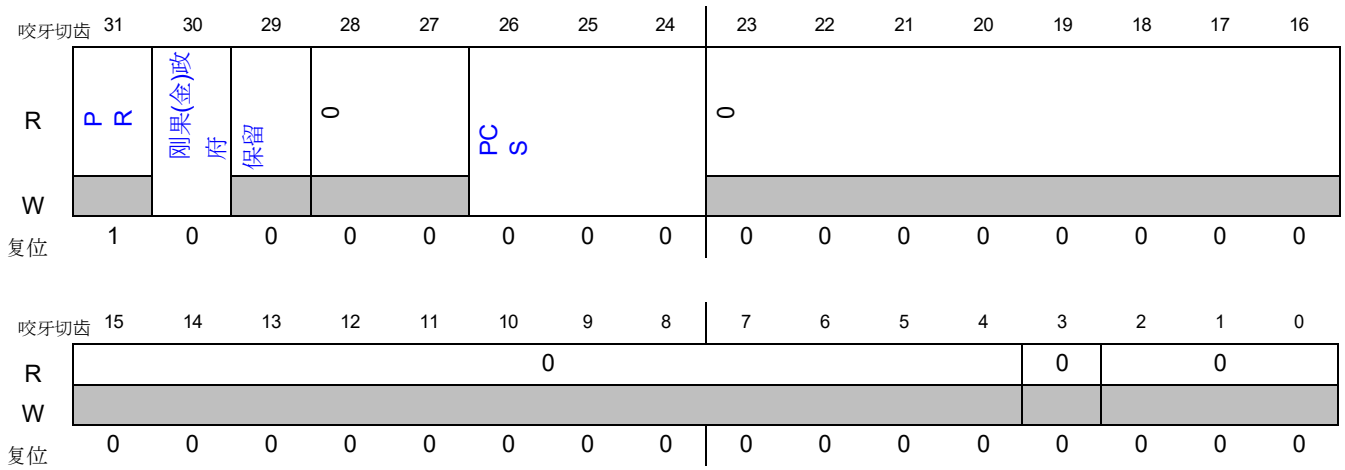
### 27.6.16.1 偏移

| 注册 | 偏移 |
|---|---|
| PCC_FTM0 | E0h |

### 27.6.16.2 职能

PCC登记簿

## 27.6.16.3 图示




## 27.6.16.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>公关 | 目前<br>该位显示外设是否存在于该设备上。<br>0b - 周边设备不存在。1b - 外围存在。 |
| 30<br>刚果(金)政府 | 时钟门控制<br>该读/写位启用外设的时钟。<br>0b - 禁用时钟<br>1b - 时钟启用。当前的时钟选择和分频器选项被锁定。 |
| 29<br>- | 这个只读的位域是保留的。这个位可以改变数值，但却是不关心的。 |
| 28-27<br>- | 这个只读的位域是保留的，其值始终为0。 |

| | |
|---|---|
| 26-24<br>辽宁省 | 周边时钟源选择<br>这个读/写位域用于支持各种时钟选择的外设。该字段只有在时钟被禁用（CGC=0）<br>时才能被写入。<br>000b - 时钟关闭。可以为这个外设启用一个外部时钟。001b - 时钟选项1<br>010b - 时钟选项2<br>011b - 时钟选项3<br>100b - 时钟选项4<br>101b - 时钟选项5<br>110b - 时钟选项6<br>111b - 时钟选项7 |
| 23-4 | 这个只读的位域是保留的，其值始终为0。 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| - | |
| 3<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 2-0<br>- | 这个只读的位域是保留的，其值始终为0。 |

## 27.6.17 PCC FTM1寄存器（PCC_FTM1）。

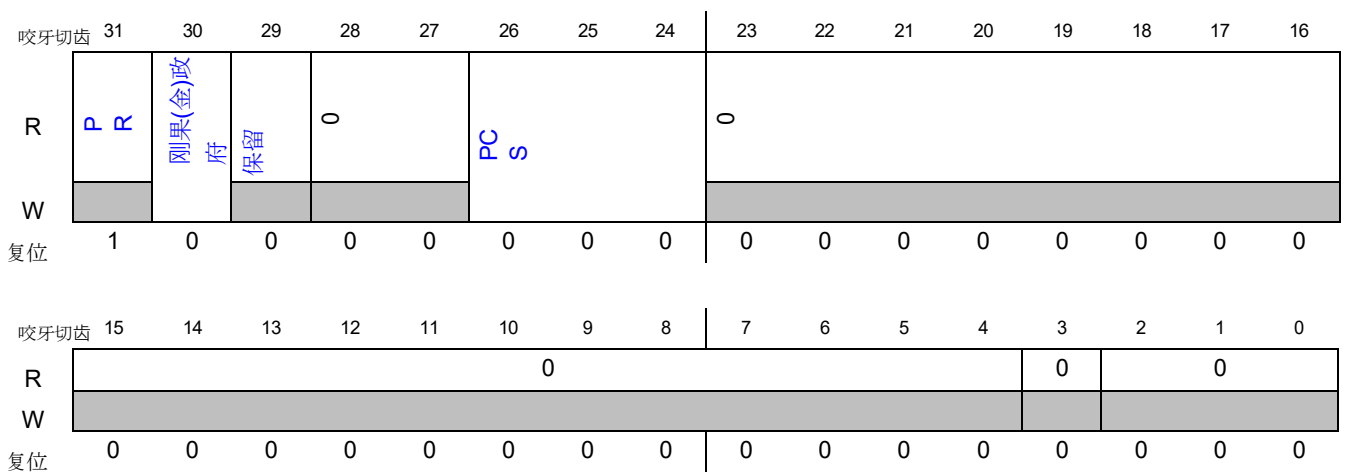
### 27.6.17.1 偏移

| 注册 | 偏移 |
|---|---|
| PCC_FTM1 | E4h |

### 27.6.17.2 职能

PCC登记簿

### 27.6.17.3 图示



### 27.6.17.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>公关 | 目前<br>该位显示外设是否存在于该设备上。<br>0b - 周边设备不存在。1b - 外围存在。 |
| 30<br>刚果(金)政府 | 时钟门控制<br>该读/写位启用外设的时钟。<br>0b - 禁用时钟<br>1b - 时钟启用。当前的时钟选择和分频器选项被锁定。 |
| 29<br>- | 这个只读的位域是保留的。这个位可以改变数值，但却是不关心的。 |
| 28-27<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 26-24<br>辽宁省 | 周边时钟源选择<br>这个读/写位域用于支持各种时钟选择的外设。该字段只有在时钟被禁用（CGC=0）<br>时才能被写入。<br>000b - 时钟关闭。可以为这个外设启用一个外部时钟。001b - 时钟选项1<br>010b - 时钟选项2<br>011b - 时钟选项3<br>100b - 时钟选项4<br>101b - 时钟选项5<br>110b - 时钟选项6<br>111b - 时钟选项7 |
| 23-4<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 3<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 2-0<br>- | 这个只读的位域是保留的，其值始终为0。 |

## 27.6.18 PCC FTM2寄存器（PCC_FTM2）。

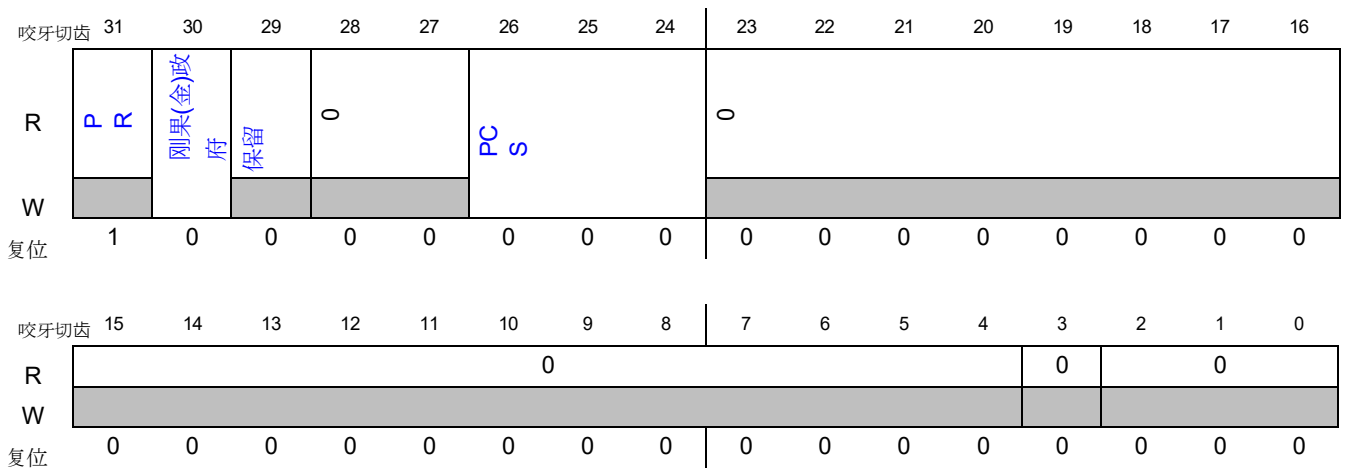
## 27.6.18.1 偏移

| 注册 | 偏移 |
| --- | --- |
| PCC_FTM2 | E8h |

## 27.6.18.2 职能

PCC登记簿

## 27.6.18.3 图示




## 27.6.18.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>公关 | 目前<br>该位显示外设是否存在于该设备上。<br>0b - 周边设备不存在。1b - 外围存在。 |
| 30<br>刚果(金)政府 | 时钟门控制<br>该读/写位启用外设的时钟。<br>0b - 禁用时钟<br>1b - 时钟启用。当前的时钟选择和分频器选项被锁定。 |
| 29<br>- | 这个只读的位域是保留的。这个位可以改变数值，但却是不关心的。 |
| 28-27<br>- | 这个只读的位域是保留的，其值始终为0。 |

| | |
|---|---|
| 26-24<br><br>辽宁省 | 周边时钟源选择<br><br>这个读/写位域用于支持各种时钟选择的外设。该字段只有在时钟被禁用（CGC=0）时<br><br>才能被写入。<br><br>000b - 时钟关闭。可以为这个外设启用一个外部时钟。<br>001b - 时钟选项1 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | 010b - 时钟选项2<br>011b - 时钟选项3<br>100b - 时钟选项4<br>101b - 时钟选项5<br>110b - 时钟选项6<br>111b - 时钟选项7 |
| 23-4<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 3<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 2-0<br>- | 这个只读的位域是保留的，其值始终为0。 |

## 27.6.19 PCC ADC0寄存器（PCC_ADC0）。

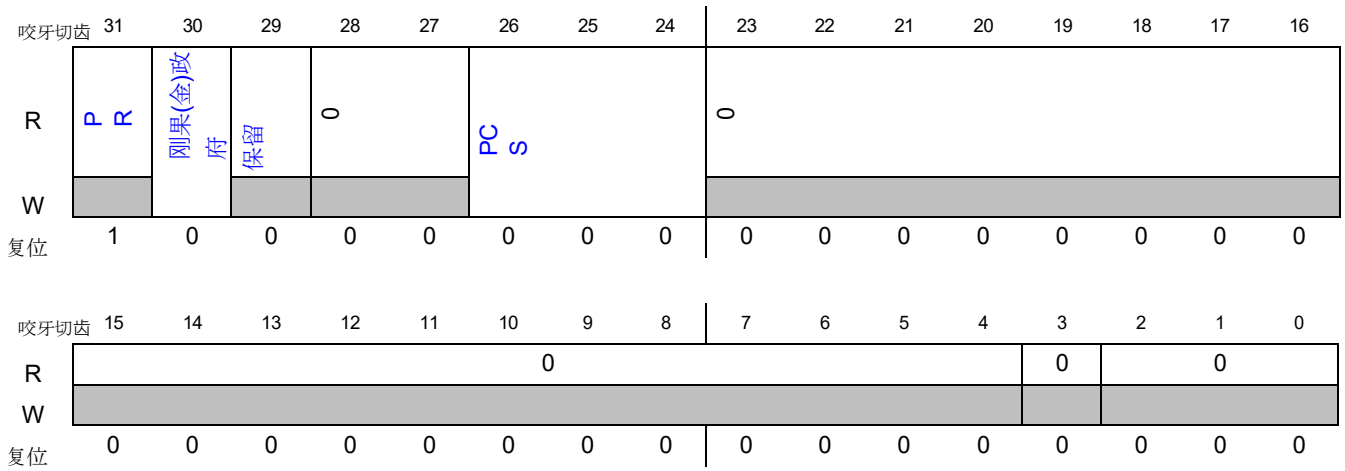
### 27.6.19.1 偏移

| 注册 | 偏移 |
| --- | --- |
| PCC_ADC0 | ECh |

### 27.6.19.2 职能

PCC登记簿

## 27.6.19.3 图示




## 27.6.19.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>公关 | 目前<br>该位显示外设是否存在于该设备上。<br>0b - 周边设备不存在。1b - 外围存在。 |
| 30<br>刚果(金)政府 | 时钟门控制<br>该读/写位启用外设的时钟。<br>0b - 禁用时钟<br>1b - 时钟启用。当前的时钟选择和分频器选项被锁定。 |
| 29<br>- | 这个只读的位域是保留的。这个位可以改变数值，但却是不关心的。 |
| 28-27<br>- | 这个只读的位域是保留的，其值始终为0。 |

| | |
|---|---|
| 26-24<br>辽宁省 | 周边时钟源选择<br>这个读/写位域用于支持各种时钟选择的外设。该字段只有在时钟被禁用（CGC=0）时<br>才能被写入。<br>000b - 时钟已关闭。<br>001b - 时钟选项1<br>010b - 时钟选项2<br>011b - 时钟选项3<br>100b - 时钟选项4<br>101b - 时钟选项5<br>110b - 时钟选项6<br>111b - 时钟选项7 |
| 23-4 | 这个只读的位域是保留的，其值始终为0。 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| - | |
| 3<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 2-0<br>- | 这个只读的位域是保留的，其值始终为0。 |

## 27.6.20 PCC RTC寄存器（PCC_RTC）。

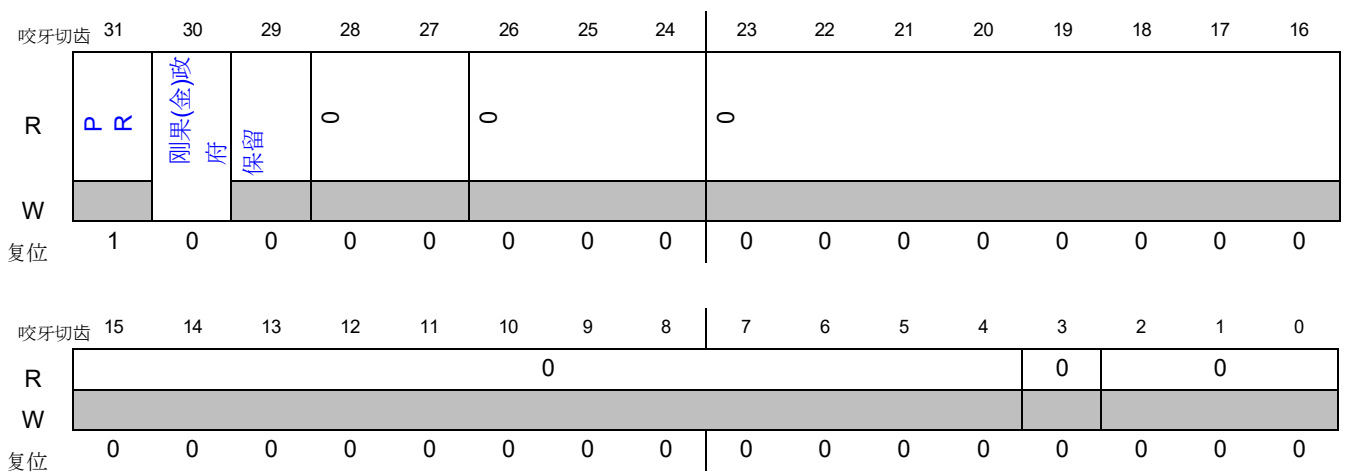
### 27.6.20.1 偏移

| 注册 | 偏移 |
|---|---|
| PCC_RTC | F4h |

### 27.6.20.2 职能

PCC登记簿

### 27.6.20.3 图示



### 27.6.20.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>公关 | 目前<br>该位显示外设是否存在于该设备上。<br>0b - 周边设备不存在。1b - 外围存在。 |
| 30<br>刚果(金)政府 | 时钟门控制<br>该读/写位启用外设的时钟。<br>0b - 禁用时钟<br>1b - 时钟启用。当前的时钟选择和分频器选项被锁定。 |
| 29<br>- | 这个只读的位域是保留的。这个位可以改变数值，但却是不关心的。 |
| 28-27<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 26-24<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 23-4<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 3<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 2-0<br>- | 这个只读的位域是保留的，其值始终为0。 |

## 27.6.21 PCC LPTMR0寄存器（PCC_LPTMR0）。

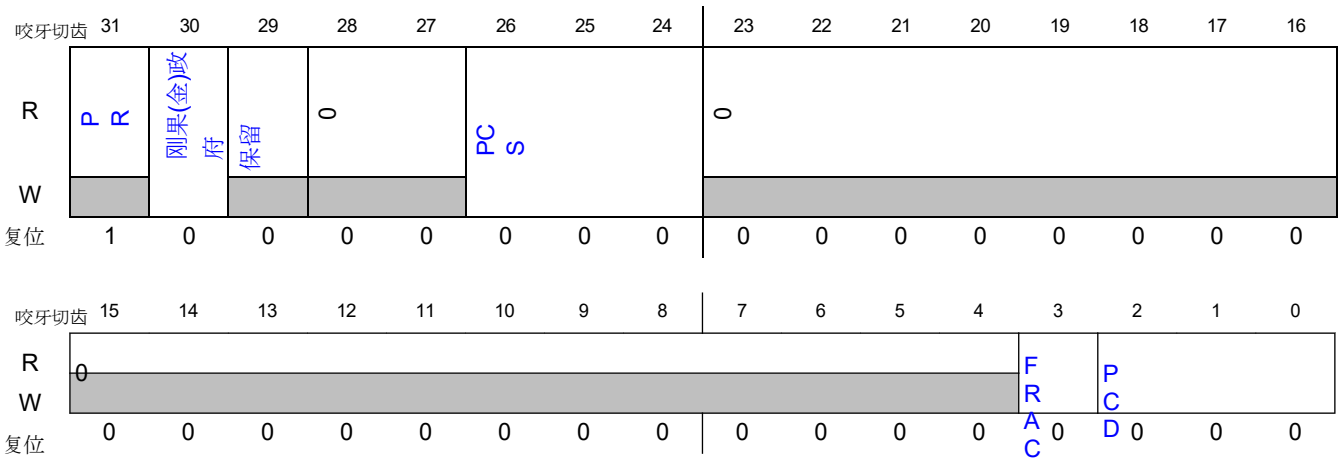
### 27.6.21.1 偏移

| 注册 | 偏移 |
|---|---|
| PCC_LPTMR0 | 100h |

### 27.6.21.2 职能

# PCC登记簿

## 27.6.21.3 图示




## 27.6.21.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>公关 | 目前<br>该位显示外设是否存在于该设备上。<br>0b - 周边设备不存在。1b - 外围存在。 |
| 30<br>刚果(金)政府 | 时钟门控制<br>该读/写位启用外设的时钟。<br>0b - 禁用时钟<br>1b - 时钟启用。当前的时钟选择和分频器选项被锁定。 |
| 29<br>- | 这个只读的位域是保留的。这个位可以改变数值，但却是不关心的。 |
| 28-27<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 26-24<br>辽宁省 | 周边时钟源选择<br>这个读/写位域用于支持各种时钟选择的外设。该字段只有在时钟被禁用（CGC=0）时才能被写入。<br>000b - 时钟已关闭。<br>001b - 时钟选项1<br>010b - 时钟选项2<br>011b - 时钟选项3 |

| | |
|---|---|
| | 100b - 时钟选项4<br>101b - 时钟选项5<br>110b - 时钟选项6<br>111b - 时钟选项7 |
| 23-4 | 这个只读的位域是保留的，其值始终为0。 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| 27.6.21.3 图示 | |
| 3<br>咬牙切齿<br>FRAC | 外围时钟分频器分数<br>这个读/写位域为用作时钟源的小数时钟分频器设置分数乘法值。分频器输出时钟=分频器输入时钟x [(FRAC+1)/(PCD+1)]。<br>这个字段只有在时钟被禁用（CGC=0）时才能写入。<br>**注意：**当除以1（PCD=000）时，不要设置FRAC位，否则，输出时钟被禁用。<br>0b - 分数值为0。 1b - 分数<br>值为1。 |
| 2-0<br>PCD | 周边时钟分频器选择<br>这个读/写位域用于需要时钟分频器的外设。分频器输出时钟=分频器输入时钟x [(FRAC+1)/(PCD+1)]。<br>这个字段只有在时钟被禁用（CGC=0）时才能写入。<br>000b - 除以 1.<br>001b - 除以 2.<br>010b - 除以 3.<br>011b - 除以 4.<br>100b - 除以 5.<br>101b - 除以 6.<br>110b - 除以 7.<br>111b - 除以 8. |

## 27.6.22 PCC PORTA寄存器（PCC_PORTA）。

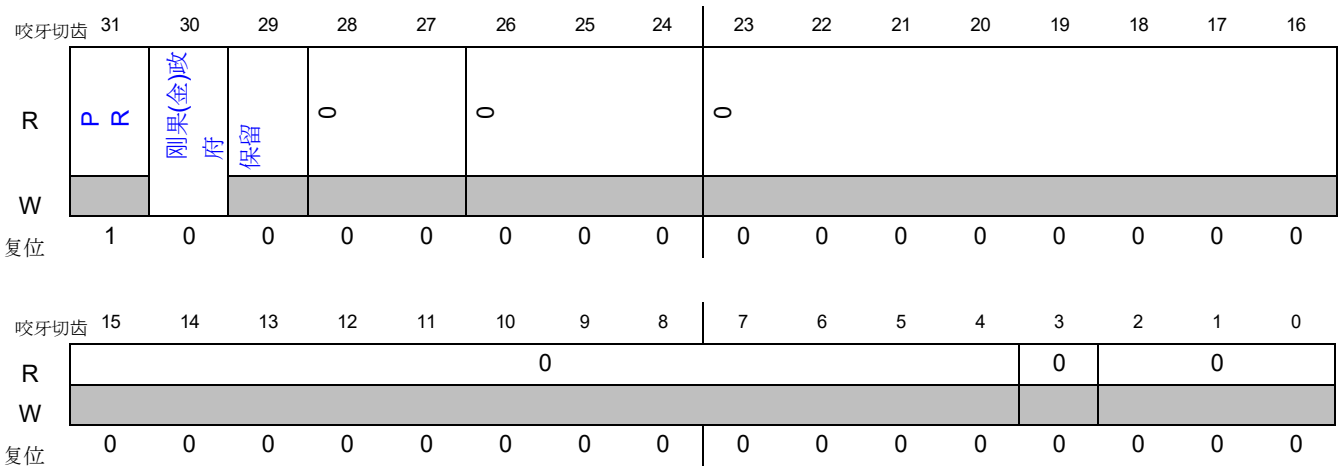
### 27.6.22.1 偏移

| 注册 | 偏移 |
|---|---|
| PCC_PORTA | 124h |

### 27.6.22.2 职能

PCC登记簿

## 27.6.22.3 图示




## 27.6.22.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>公关 | 目前<br>该位显示外设是否存在于该设备上。<br>0b - 周边设备不存在。1b - 外围存在。 |
| 30<br>刚果(金)政府 | 时钟门控制<br>该读/写位启用外设的时钟。<br>0b - 禁用时钟<br>1b - 时钟启用。当前的时钟选择和分频器选项被锁定。 |
| 29<br>- | 这个只读的位域是保留的。这个位可以改变数值，但却是不关心的。 |
| 28-27<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 26-24<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 23-4<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 3<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 2-0<br>- | 这个只读的位域是保留的，其值始终为0。 |

# 27.6.23 PCC PORTB寄存器（PCC_PORTB）

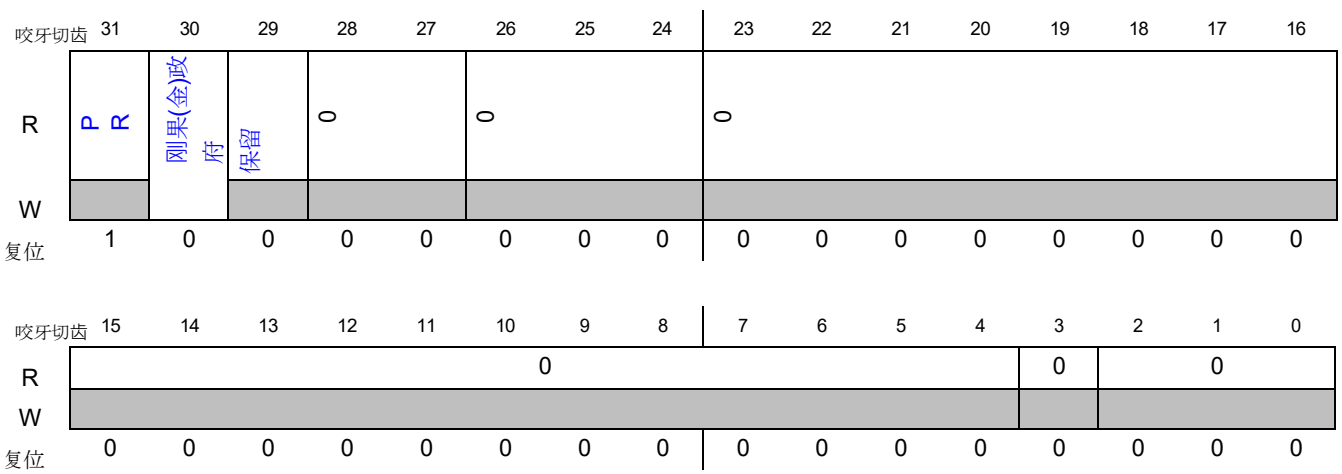
## 27.6.23.1 偏移

| 注册 | 偏移 |
|---|---|
| PCC_PORTB | 128h |

## 27.6.23.2 职能

PCC登记簿

## 27.6.23.3 图示




## 27.6.23.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>公关 | 目前<br>该位显示外设是否存在于该设备上。<br>0b - 周边设备不存在。1b - 外围存在。 |

| | |
|---|---|
| 30<br><br>刚果(金)政府 | 时钟门控制<br><br>该读/写位启用外设的时钟。 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | 0b - 禁用时钟<br>1b - 时钟启用。当前的时钟选择和分频器选项被锁定。 |
| 29<br>- | 这个只读的位域是保留的。这个位可以改变数值，但却是不关心的。 |
| 28-27<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 26-24<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 23-4<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 3<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 2-0<br>- | 这个只读的位域是保留的，其值始终为0。 |

## 27.6.24 PCC PORTC寄存器（PCC_PORTC）。

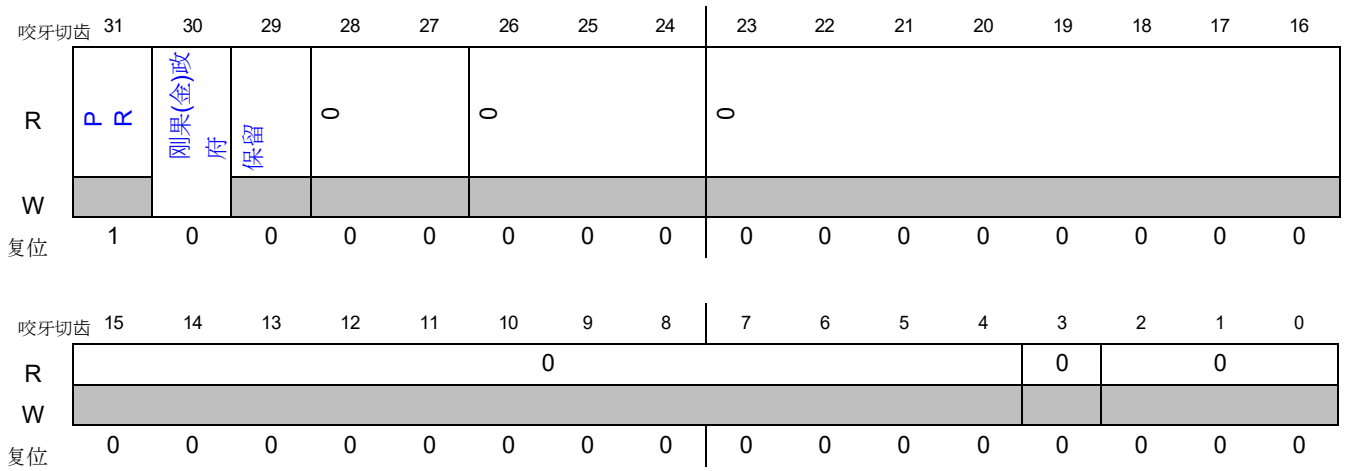
### 27.6.24.1 偏移

| 注册 | 偏移 |
| --- | --- |
| PCC_PORTC | 12Ch |

### 27.6.24.2 职能

PCC登记簿

## 27.6.24.3 图示




## 27.6.24.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>公关 | 目前<br>该位显示外设是否存在于该设备上。<br>0b - 周边设备不存在。1b - 外围存在。 |
| 30<br>刚果(金)政府 | 时钟门控制<br>该读/写位启用外设的时钟。<br>0b - 禁用时钟<br>1b - 时钟启用。当前的时钟选择和分频器选项被锁定。 |
| 29<br>- | 这个只读的位域是保留的。这个位可以改变数值，但却是不关心的。 |
| 28-27<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 26-24<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 23-4<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 3<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 2-0<br>- | 这个只读的位域是保留的，其值始终为0。 |

# 27.6.25 PCC PORTD寄存器（PCC_PORTD）。

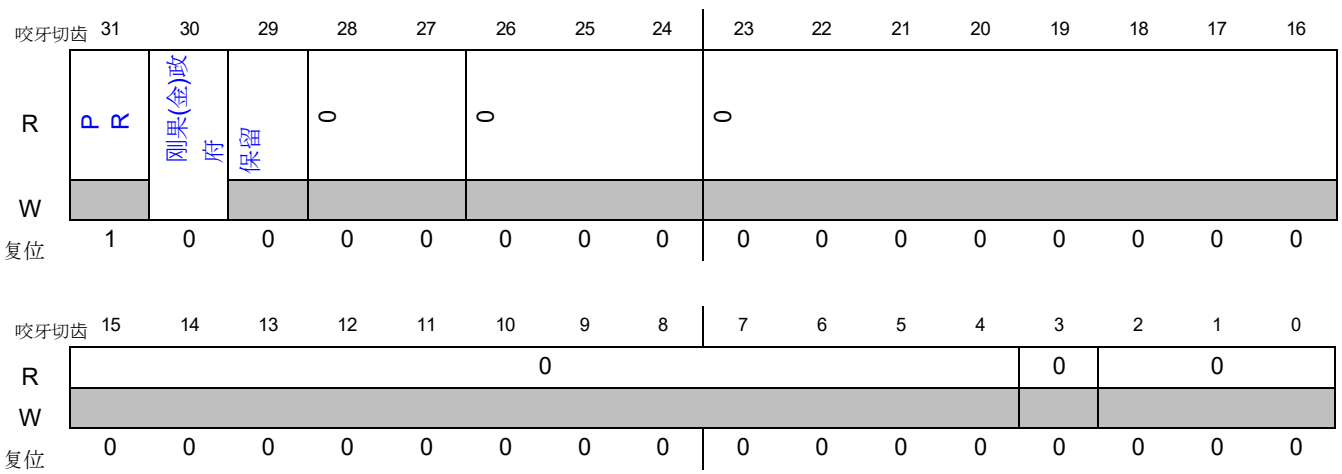
## 27.6.25.1 偏移

| 注册 | 偏移 |
|---|---|
| PCC_PORTD | 130h |

## 27.6.25.2 职能

PCC登记簿

## 27.6.25.3 图示




## 27.6.25.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>公关 | 目前<br>该位显示外设是否存在于该设备上。<br>0b - 周边设备不存在。1b - 外围存在。 |

| | |
|---|---|
| 30<br><br>刚果(金)政府 | 时钟门控制<br><br>该读/写位启用外设的时钟。 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | 0b - 禁用时钟<br>1b - 时钟启用。当前的时钟选择和分频器选项被锁定。 |
| 29<br>- | 这个只读的位域是保留的。这个位可以改变数值，但却是不关心的。 |
| 28-27<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 26-24<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 23-4<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 3<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 2-0<br>- | 这个只读的位域是保留的，其值始终为0。 |

## 27.6.26 PCC PORTE寄存器（PCC_PORTE）。

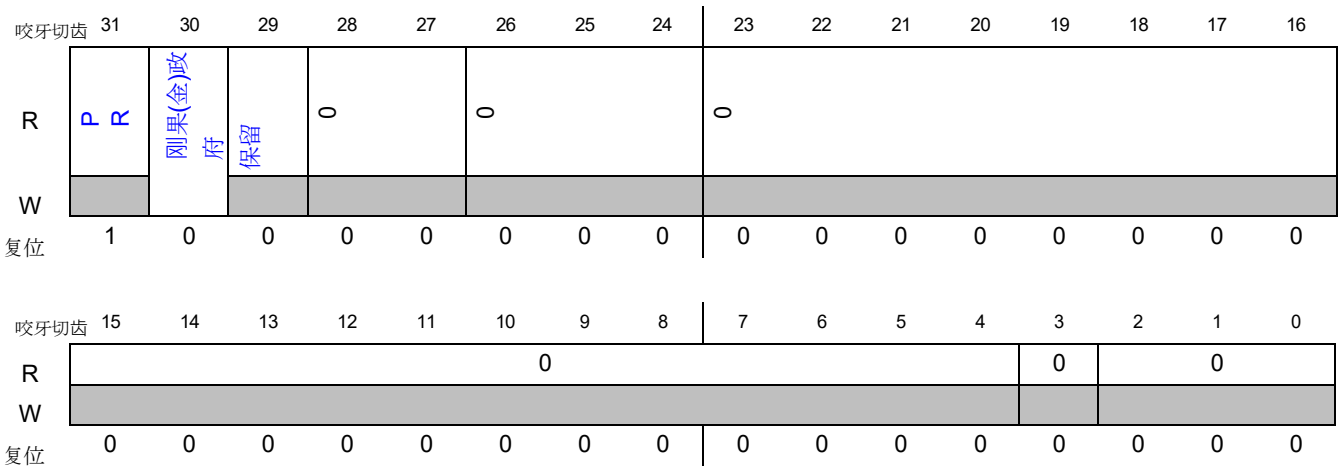
### 27.6.26.1 偏移

| 注册 | 偏移 |
| --- | --- |
| 呼叫中心：PCC_PORTE | 134h |

### 27.6.26.2 职能

PCC登记簿

## 27.6.26.3 图示




## 27.6.26.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>公关 | 目前<br>该位显示外设是否存在于该设备上。<br>0b - 周边设备不存在。1b - 外围存在。 |
| 30<br>刚果(金)政府 | 时钟门控制<br>该读/写位启用外设的时钟。<br>0b - 禁用时钟<br>1b - 时钟启用。当前的时钟选择和分频器选项被锁定。 |
| 29<br>- | 这个只读的位域是保留的。这个位可以改变数值，但却是不关心的。 |
| 28-27<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 26-24<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 23-4<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 3<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 2-0<br>- | 这个只读的位域是保留的，其值始终为0。 |

# 27.6.27 PCC SAI0寄存器（PCC_SAI0）。

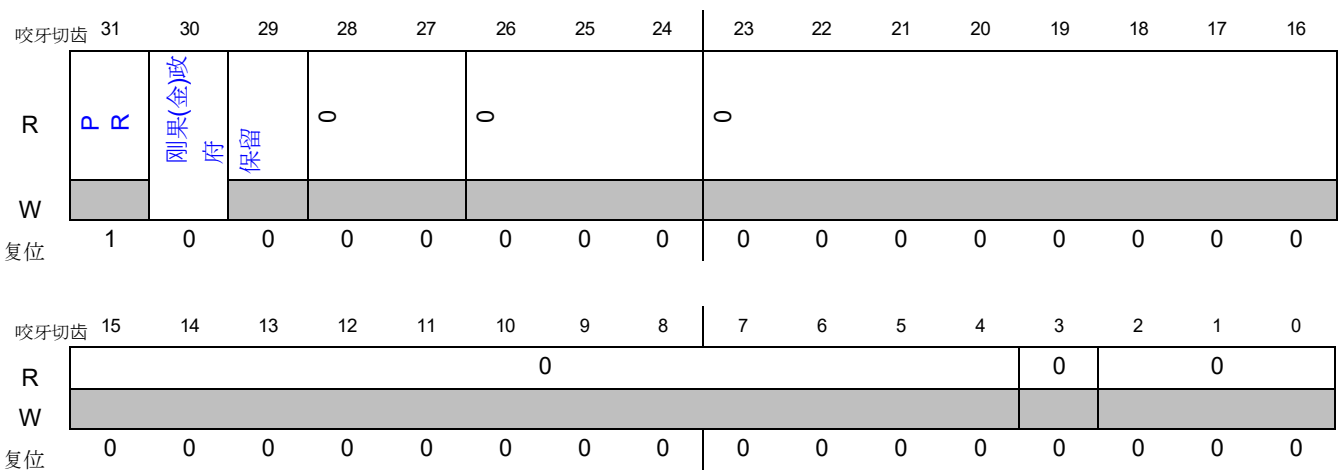
## 27.6.27.1 偏移

| 注册 | 偏移 |
|---|---|
| PCC_SAI0 | 150h |

## 27.6.27.2 职能

PCC登记簿

## 27.6.27.3 图示




## 27.6.27.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>公关 | 目前<br>该位显示外设是否存在于该设备上。<br>0b - 周边设备不存在。1b - 外围存在。 |

| | |
|---|---|
| 30<br><br>刚果(金)政府 | 时钟门控制<br><br>该读/写位启用外设的时钟。 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | 0b - 禁用时钟<br>1b - 时钟启用。当前的时钟选择和分频器选项被锁定。 |
| 29<br>- | 这个只读的位域是保留的。这个位可以改变数值，但却是不关心的。 |
| 28-27<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 26-24<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 23-4<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 3<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 2-0<br>- | 这个只读的位域是保留的，其值始终为0。 |

## 27.6.28 PCC SAI1寄存器（PCC_SAI1）。

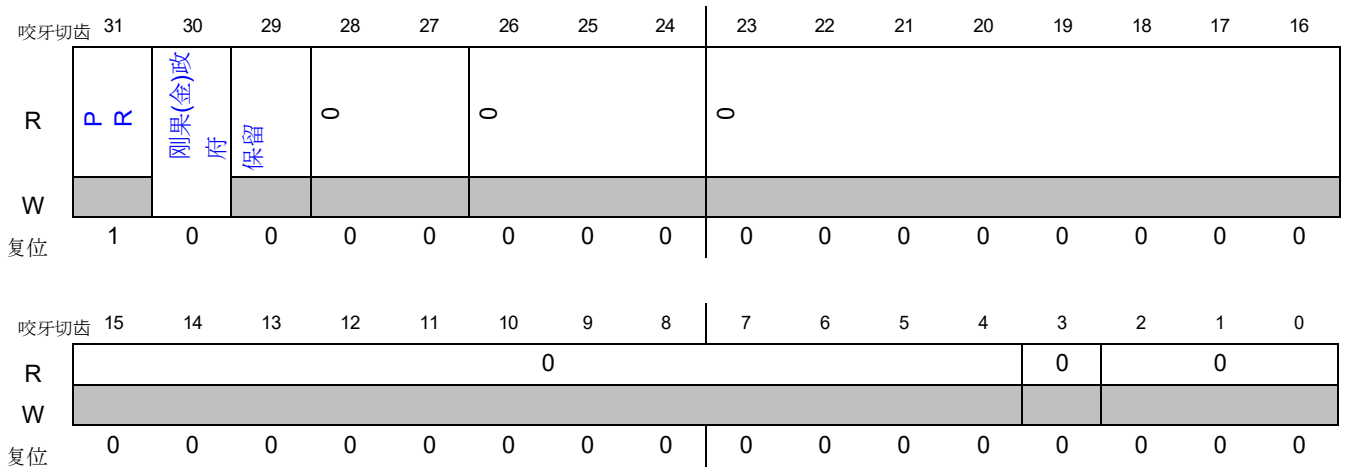
### 27.6.28.1 偏移

| 注册 | 偏移 |
| --- | --- |
| PCC_SAI1 | 154h |

### 27.6.28.2 职能

PCC登记簿

## 27.6.28.3 图示




## 27.6.28.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>公关 | 目前<br>该位显示外设是否存在于该设备上。<br>0b - 周边设备不存在。1b - 外围存在。 |
| 30<br>刚果(金)政府 | 时钟门控制<br>该读/写位启用外设的时钟。<br>0b - 禁用时钟<br>1b - 时钟启用。当前的时钟选择和分频器选项被锁定。 |
| 29<br>- | 这个只读的位域是保留的。这个位可以改变数值，但却是不关心的。 |
| 28-27<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 26-24<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 23-4<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 3<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 2-0<br>- | 这个只读的位域是保留的，其值始终为0。 |

# 27.6.29 PCC FlexIO寄存器（PCC_FlexIO）。

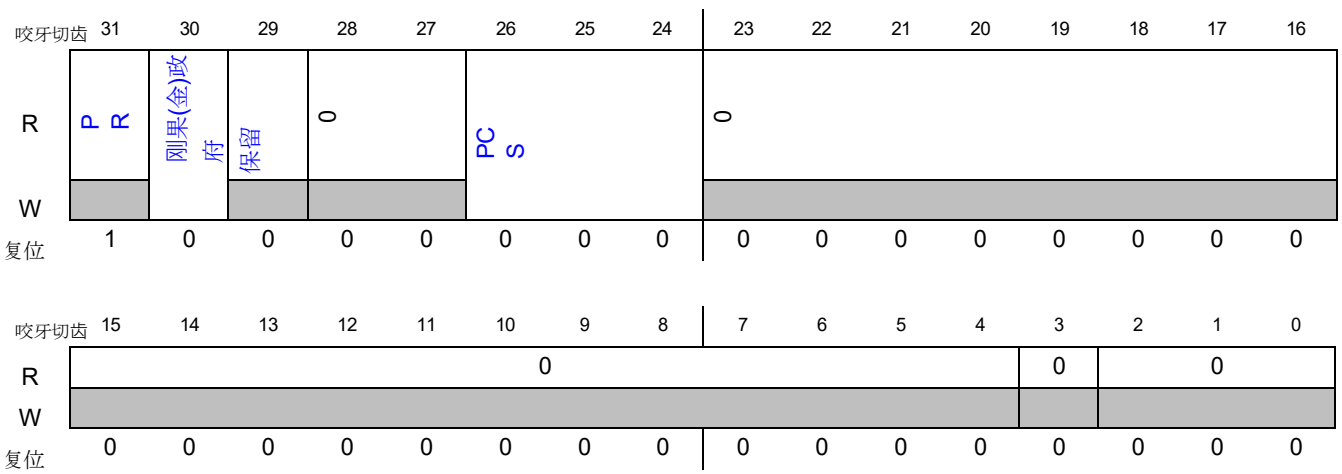
## 27.6.29.1 偏移

| 注册 | 偏移 |
|---|---|
| PCC_FlexIO | 168h |

## 27.6.29.2 职能

PCC登记簿

## 27.6.29.3 图示




## 27.6.29.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>公关 | 目前<br>该位显示外设是否存在于该设备上。<br>0b - 周边设备不存在。1b - 外围存在。 |

| | |
|---|---|
| 30<br><br>刚果(金)政府 | 时钟门控制<br><br>该读/写位启用外设的时钟。 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | 0b - 禁用时钟<br>1b - 时钟启用。当前的时钟选择和分频器选项被锁定。 |
| 29<br>- | 这个只读的位域是保留的。这个位可以改变数值，但却是不关心的。 |
| 28-27<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 26-24<br>辽宁省 | 周边时钟源选择<br>这个读/写位域用于支持各种时钟选择的外设。<br>这个字段只有在时钟被禁用（CGC=0）时才能写入。<br>000b - 时钟已关闭。<br>001b - 时钟选项1<br>010b - 时钟选项2<br>011b - 时钟选项3<br>100b - 时钟选项4<br>101b - 时钟选项5<br>110b - 时钟选项6<br>111b - 时钟选项7 |
| 23-4<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 3<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 2-0<br>- | 这个只读的位域是保留的，其值始终为0。 |

## 27.6.30 PCC EWM寄存器（PCC_EWM）。

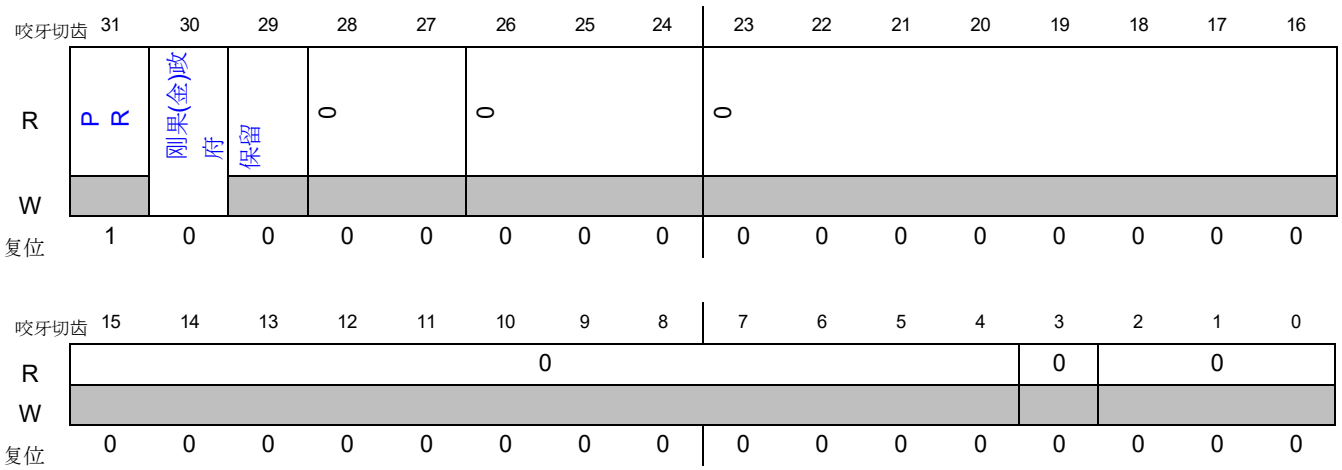
### 27.6.30.1 偏移

| 注册 | 偏移 |
| --- | --- |
| PCC_EWM | 184h |

### 27.6.30.2 职能

PCC登记簿

## 27.6.30.3 图示




## 27.6.30.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>公关 | 目前<br>该位显示外设是否存在于该设备上。<br>0b - 周边设备不存在。1b - 外围存在。 |
| 30<br>刚果(金)政府 | 时钟门控制<br>该读/写位启用外设的时钟。<br>0b - 禁用时钟<br>1b - 时钟启用。当前的时钟选择和分频器选项被锁定。 |
| 29<br>- | 这个只读的位域是保留的。这个位可以改变数值，但却是不关心的。 |
| 28-27<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 26-24<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 23-4<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 3<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 2-0<br>- | 这个只读的位域是保留的，其值始终为0。 |

# 27.6.31 PCC LPI2C0寄存器（PCC_LPI2C0）。

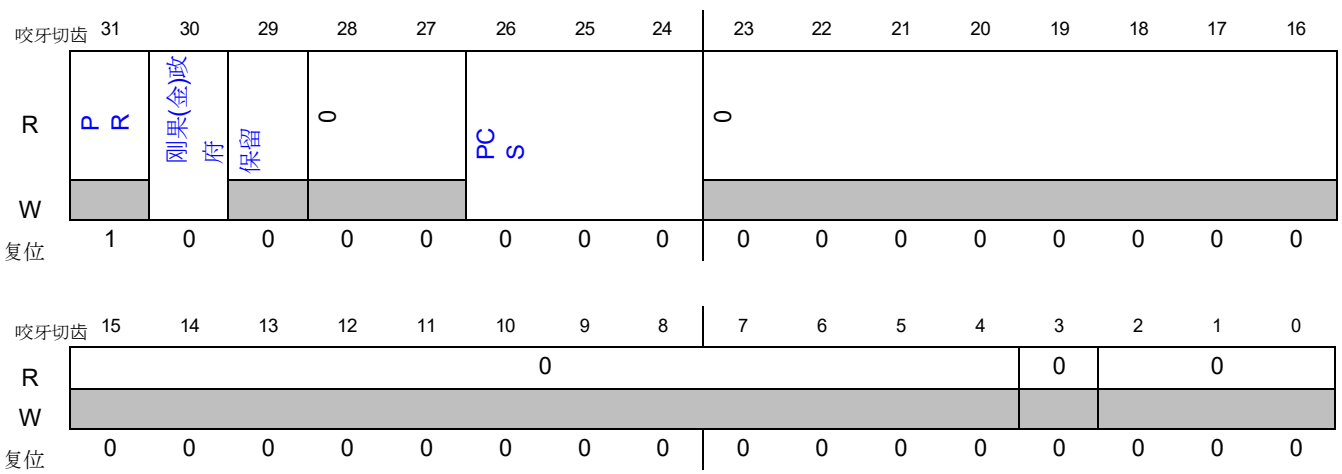
## 27.6.31.1 偏移

| 注册 | 偏移 |
|---|---|
| PCC_LPI2C0 | 198h |

## 27.6.31.2 职能

PCC登记簿

## 27.6.31.3 图示




## 27.6.31.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>公关 | 目前<br>该位显示外设是否存在于该设备上。<br>0b - 周边设备不存在。1b - 外围存在。 |

| | |
|---|---|
| 30<br><br>刚果(金)政府 | 时钟门控制<br><br>该读/写位启用外设的时钟。 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | 0b - 禁用时钟<br>1b - 时钟启用。当前的时钟选择和分频器选项被锁定。 |
| 29<br>- | 这个只读的位域是保留的。这个位可以改变数值，但却是不关心的。 |
| 28-27<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 26-24<br>辽宁省 | 周边时钟源选择<br>这个读/写位域用于支持各种时钟选择的外设。<br>这个字段只有在时钟被禁用（CGC=0）时才能写入。<br>000b - 时钟已关闭。<br>001b - 时钟选项1<br>010b - 时钟选项2<br>011b - 时钟选项3<br>100b - 时钟选项4<br>101b - 时钟选项5<br>110b - 时钟选项6<br>111b - 时钟选项7 |
| 23-4<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 3<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 2-0<br>- | 这个只读的位域是保留的，其值始终为0。 |

## 27.6.32 PCC LPI2C1寄存器（PCC_LPI2C1）。

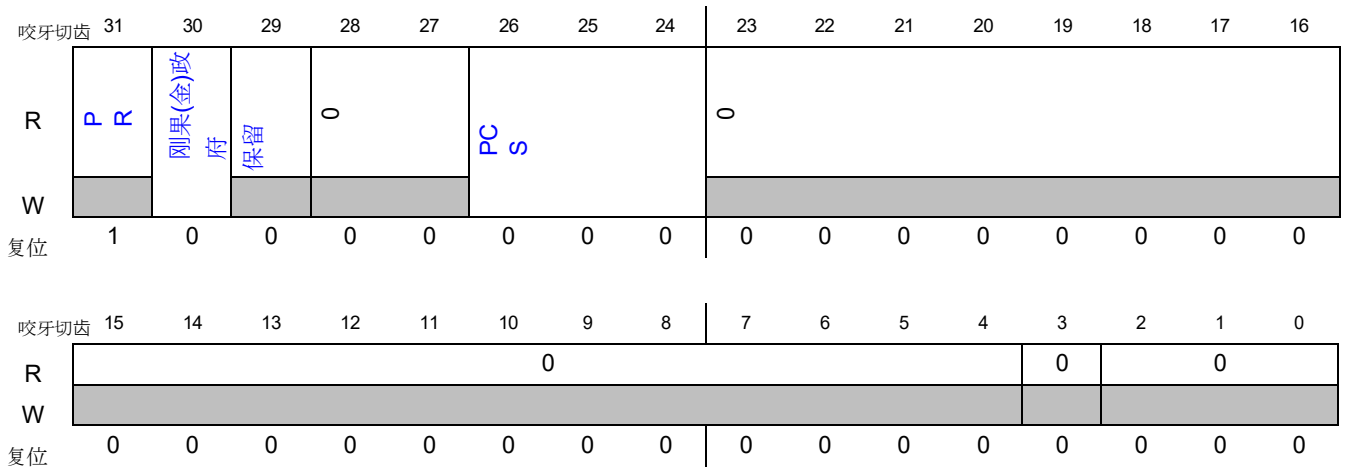
### 27.6.32.1 偏移

| 注册 | 偏移 |
| --- | --- |
| PCC_LPI2C1 | 19Ch |

### 27.6.32.2 职能

PCC登记簿

## 27.6.32.3 图示




## 27.6.32.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>公关 | 目前<br>该位显示外设是否存在于该设备上。<br>0b - 周边设备不存在。1b - 外围存在。 |
| 30<br>刚果(金)政府 | 时钟门控制<br>该读/写位启用外设的时钟。<br>0b - 禁用时钟<br>1b - 时钟启用。当前的时钟选择和分频器选项被锁定。 |
| 29<br>- | 这个只读的位域是保留的。这个位可以改变数值，但却是不关心的。 |
| 28-27<br>- | 这个只读的位域是保留的，其值始终为0。 |

| | |
|---|---|
| 26-24<br>辽宁省 | 周边时钟源选择<br>这个读/写位域用于支持各种时钟选择的外设。该字段只有在时钟被禁用（CGC=0）时<br>才能被写入。<br>000b - 时钟已关闭。<br>001b - 时钟选项1<br>010b - 时钟选项2<br>011b - 时钟选项3<br>100b - 时钟选项4<br>101b - 时钟选项5<br>110b - 时钟选项6<br>111b - 时钟选项7 |
| 23-4 | 这个只读的位域是保留的，其值始终为0。 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| - | |
| 3<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 2-0<br>- | 这个只读的位域是保留的，其值始终为0。 |

## 27.6.33 PCC LPUART0寄存器（PCC_LPUART0）

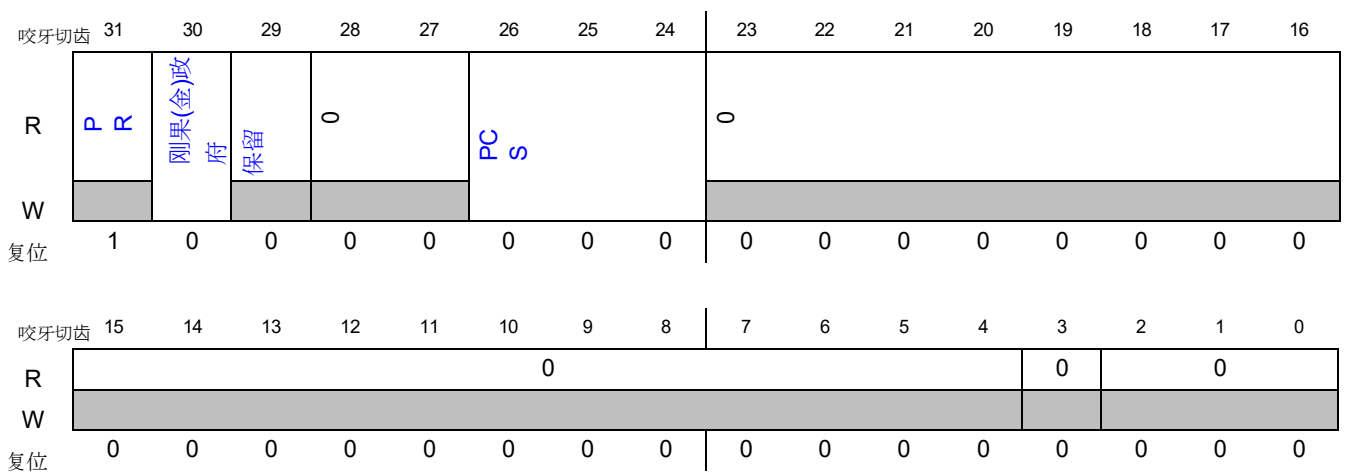
### 27.6.33.1 偏移

| 注册 | 偏移 |
|---|---|
| PCC_LPUART0 | 1A8h |

### 27.6.33.2 职能

PCC登记簿

### 27.6.33.3 图示



### 27.6.33.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>公关 | 目前<br>该位显示外设是否存在于该设备上。<br>0b - 周边设备不存在。1b - 外围存在。 |
| 30<br>刚果(金)政府 | 时钟门控制<br>该读/写位启用外设的时钟。<br>0b - 禁用时钟<br>1b - 时钟启用。当前的时钟选择和分频器选项被锁定。 |
| 29<br>- | 这个只读的位域是保留的。这个位可以改变数值，但却是不关心的。 |
| 28-27<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 26-24<br>辽宁省 | 周边时钟源选择<br>这个读/写位域用于支持各种时钟选择的外设。该字段只有在时钟被禁用（CGC=0）时才能被写入。<br>000b - 时钟已关闭。<br>001b - 时钟选项1<br>010b - 时钟选项2<br>011b - 时钟选项3<br>100b - 时钟选项4<br>101b - 时钟选项5<br>110b - 时钟选项6<br>111b - 时钟选项7 |
| 23-4<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 3<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 2-0<br>- | 这个只读的位域是保留的，其值始终为0。 |

## 27.6.34 PCC LPUART1寄存器（PCC_LPUART1）。

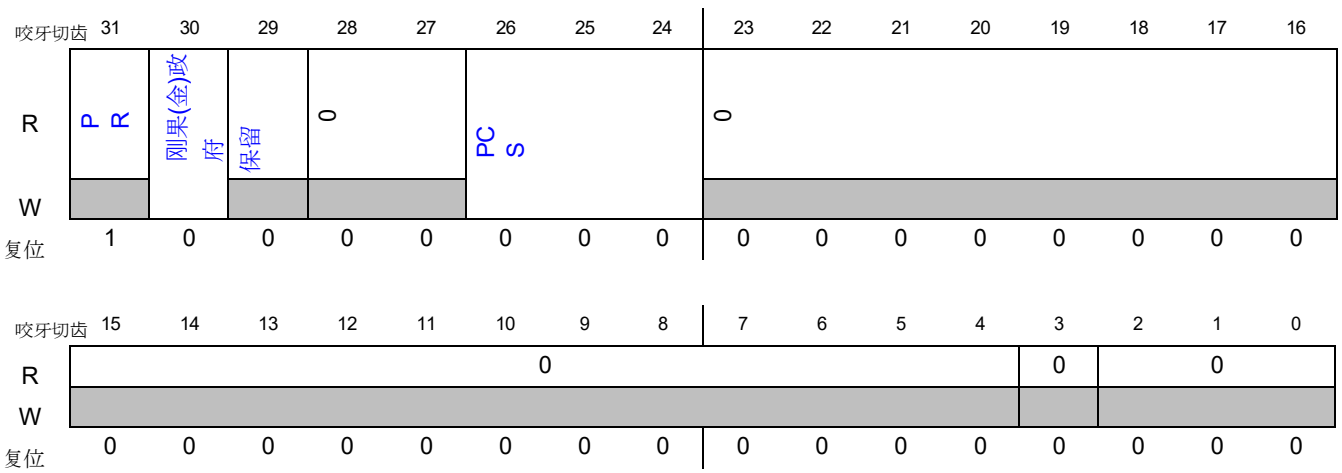
### 27.6.34.1 偏移

| 注册 | 偏移 |
| --- | --- |
| PCC_LPUART1 | 1ACh |

## 27.6.34.2 职能

PCC登记簿

## 27.6.34.3 图示




## 27.6.34.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>公关 | 目前<br>该位显示外设是否存在于该设备上。<br>0b - 周边设备不存在。1b - 外围存在。 |
| 30<br>刚果(金)政府 | 时钟门控制<br>该读/写位启用外设的时钟。<br>0b - 禁用时钟<br>1b - 时钟启用。当前的时钟选择和分频器选项被锁定。 |
| 29<br>- | 这个只读的位域是保留的。这个位可以改变数值，但却是不关心的。 |
| 28-27<br>- | 这个只读的位域是保留的，其值始终为0。 |

| | |
|---|---|
| 26-24<br><br>辽宁省 | 周边时钟源选择<br><br>这个读/写位域用于支持各种时钟选择的外设。该字段只有在时钟被禁用（CGC=0）时<br><br>才能被写入。<br><br>000b - 时钟关闭。<br>001b - 时钟选项1 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | 010b - 时钟选项2<br>011b - 时钟选项3<br>100b - 时钟选项4<br>101b - 时钟选项5<br>110b - 时钟选项6<br>111b - 时钟选项7 |
| 23-4<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 3<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 2-0<br>- | 这个只读的位域是保留的，其值始终为0。 |

## 27.6.35 PCC LPUART2寄存器（PCC_LPUART2）。

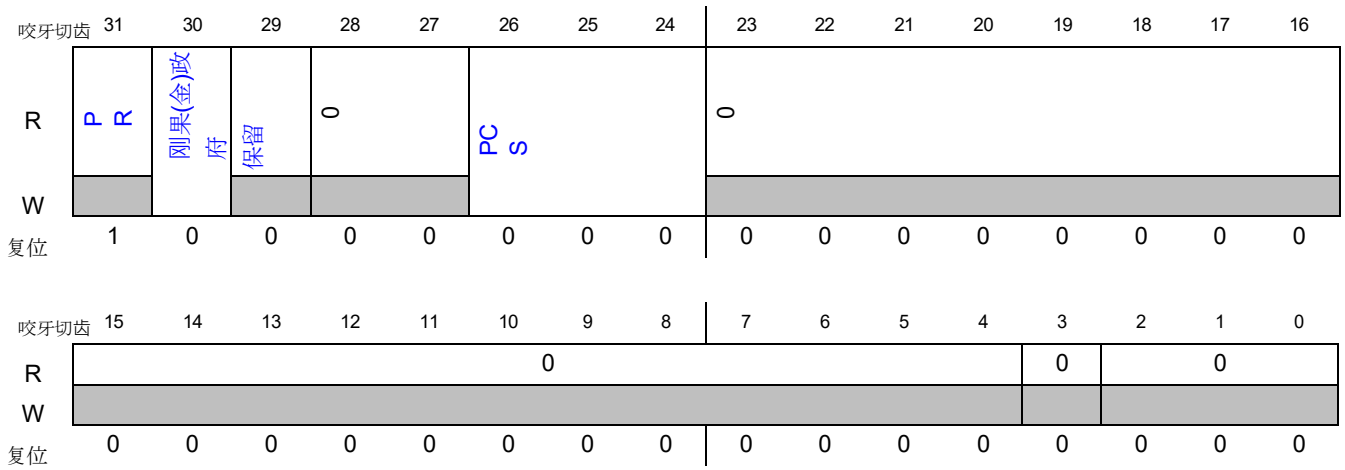
### 27.6.35.1 偏移

| 注册 | 偏移 |
| --- | --- |
| PCC_LPUART2 | 1B0h |

### 27.6.35.2 职能

PCC登记簿

## 27.6.35.3 图示




## 27.6.35.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>公关 | 目前<br>该位显示外设是否存在于该设备上。<br>0b - 周边设备不存在。1b - 外围存在。 |
| 30<br>刚果(金)政府 | 时钟门控制<br>该读/写位启用外设的时钟。<br>0b - 禁用时钟<br>1b - 时钟启用。当前的时钟选择和分频器选项被锁定。 |
| 29<br>- | 这个只读的位域是保留的。这个位可以改变数值，但却是不关心的。 |
| 28-27<br>- | 这个只读的位域是保留的，其值始终为0。 |

| | |
|---|---|
| 26-24<br>辽宁省 | 周边时钟源选择<br>这个读/写位域用于支持各种时钟选择的外设。该字段只有在时钟被禁用（CGC=0）时<br>才能被写入。<br>000b - 时钟已关闭。<br>001b - 时钟选项1<br>010b - 时钟选项2<br>011b - 时钟选项3<br>100b - 时钟选项4<br>101b - 时钟选项5<br>110b - 时钟选项6<br>111b - 时钟选项7 |
| 23-4 | 这个只读的位域是保留的，其值始终为0。 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| - | |
| 3<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 2-0<br>- | 这个只读的位域是保留的，其值始终为0。 |

# 27.6.36 PCC FTM4寄存器（PCC_FTM4）。

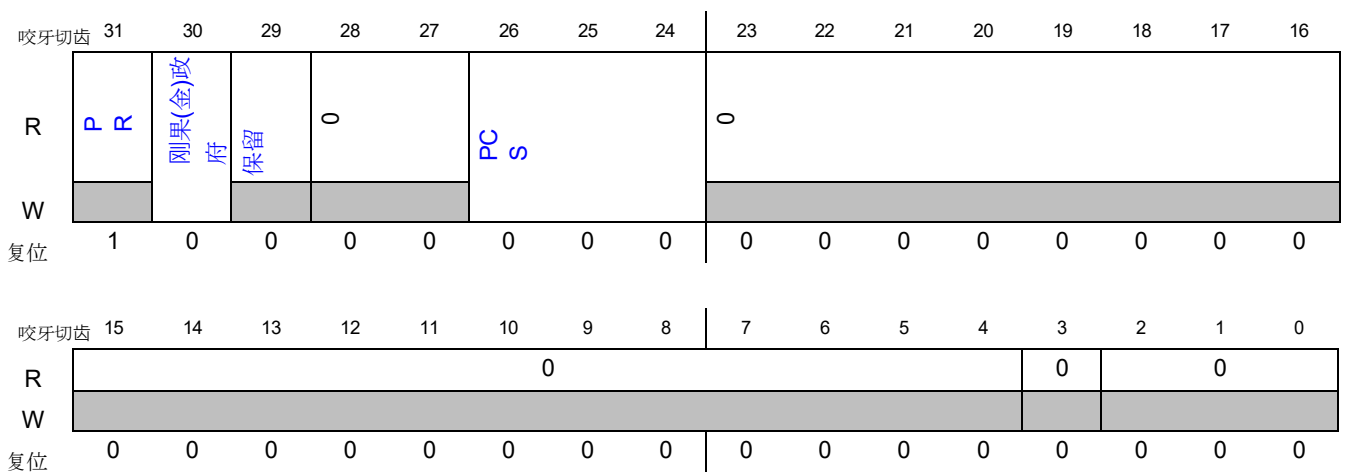
## 27.6.36.1 偏移

| 注册 | 偏移 |
|---|---|
| PCC_FTM4 | 1B8h |

## 27.6.36.2 职能

PCC登记簿

## 27.6.36.3 图示



### 27.6.36.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>公关 | 目前<br>该位显示外设是否存在于该设备上。<br>0b - 周边设备不存在。1b - 外围存在。 |
| 30<br>刚果(金)政府 | 时钟门控制<br>该读/写位启用外设的时钟。<br>0b - 禁用时钟<br>1b - 时钟启用。当前的时钟选择和分频器选项被锁定。 |
| 29<br>- | 这个只读的位域是保留的。这个位可以改变数值，但却是不关心的。 |
| 28-27<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 26-24<br>辽宁省 | 周边时钟源选择<br>这个读/写位域用于支持各种时钟选择的外设。该字段只有在时钟被禁用（CGC=0）<br>时才能被写入。<br>000b - 时钟关闭。可以为这个外设启用一个外部时钟。001b - 时钟选项1<br>010b - 时钟选项2<br>011b - 时钟选项3<br>100b - 时钟选项4<br>101b - 时钟选项5<br>110b - 时钟选项6<br>111b - 时钟选项7 |
| 23-4<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 3<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 2-0<br>- | 这个只读的位域是保留的，其值始终为0。 |

## 27.6.37 PCC FTM5寄存器（PCC_FTM5）。

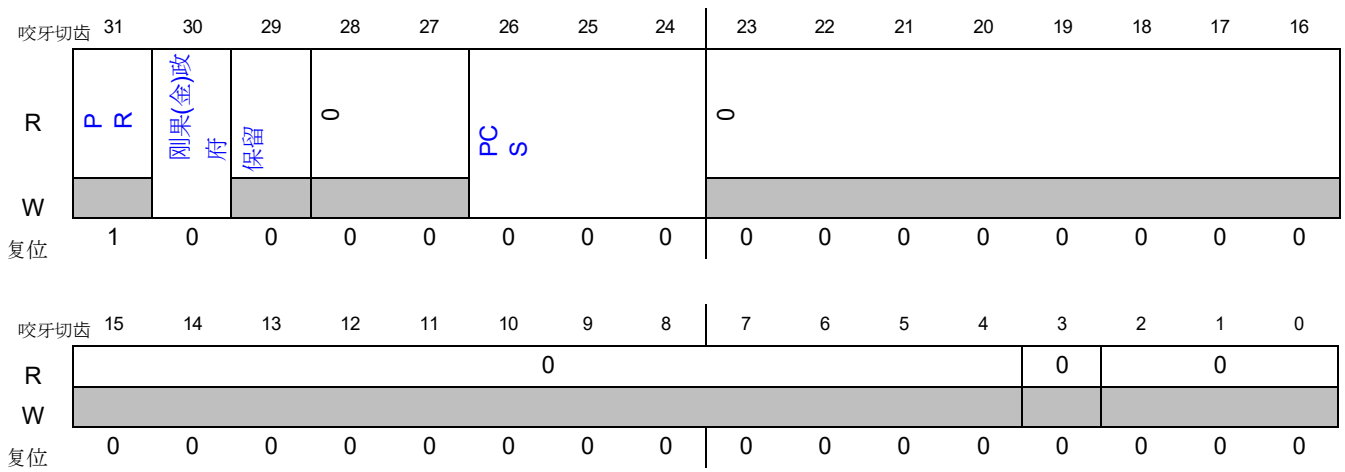
## 27.6.37.1 偏移

| 注册 | 偏移 |
| --- | --- |
| PCC_FTM5 | 1BCh |

## 27.6.37.2 职能

PCC登记簿

## 27.6.37.3 图示




## 27.6.37.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>公关 | 目前<br>该位显示外设是否存在于该设备上。<br>0b - 周边设备不存在。1b - 外围存在。 |
| 30<br>刚果(金)政府 | 时钟门控制<br>该读/写位启用外设的时钟。<br>0b - 禁用时钟<br>1b - 时钟启用。当前的时钟选择和分频器选项被锁定。 |
| 29<br>- | 这个只读的位域是保留的。这个位可以改变数值，但却是不关心的。 |
| 28-27<br>- | 这个只读的位域是保留的，其值始终为0。 |

| | |
|---|---|
| 26-24<br><br>辽宁省 | 周边时钟源选择<br><br>这个读/写位域用于支持各种时钟选择的外设。该字段只有在时钟被禁用（CGC=0）时<br><br>才能被写入。<br><br>000b - 时钟关闭。可以为这个外设启用一个外部时钟。<br>001b - 时钟选项1 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | 010b - 时钟选项2<br>011b - 时钟选项3<br>100b - 时钟选项4<br>101b - 时钟选项5<br>110b - 时钟选项6<br>111b - 时钟选项7 |
| 23-4<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 3<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 2-0<br>- | 这个只读的位域是保留的，其值始终为0。 |

## 27.6.38 PCC FTM6寄存器（PCC_FTM6）。

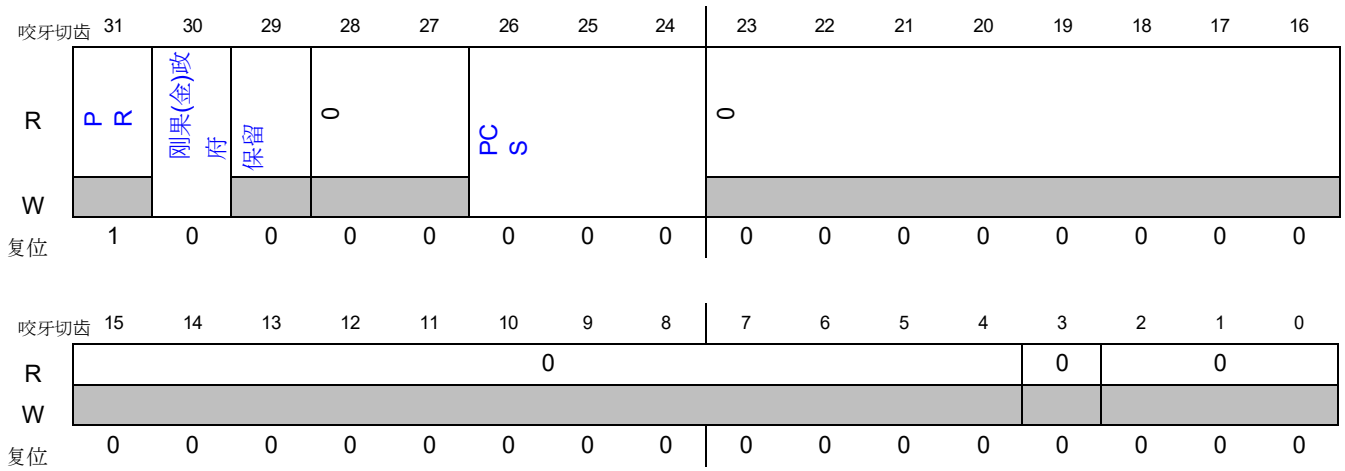
### 27.6.38.1 偏移

| 注册 | 偏移 |
| --- | --- |
| PCC_FTM6 | 1C0h |

### 27.6.38.2 职能

PCC登记簿

## 27.6.38.3 图示




## 27.6.38.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>公关 | 目前<br>该位显示外设是否存在于该设备上。<br>0b - 周边设备不存在。1b - 外围存在。 |
| 30<br>刚果(金)政府 | 时钟门控制<br>该读/写位启用外设的时钟。<br>0b - 禁用时钟<br>1b - 时钟启用。当前的时钟选择和分频器选项被锁定。 |
| 29<br>- | 这个只读的位域是保留的。这个位可以改变数值，但却是不关心的。 |
| 28-27<br>- | 这个只读的位域是保留的，其值始终为0。 |

| | |
|---|---|
| 26-24<br>辽宁省 | 周边时钟源选择<br>这个读/写位域用于支持各种时钟选择的外设。该字段只有在时钟被禁用（CGC=0）<br>时才能被写入。<br>000b - 时钟关闭。可以为这个外设启用一个外部时钟。001b - 时钟选项1<br>010b - 时钟选项2<br>011b - 时钟选项3<br>100b - 时钟选项4<br>101b - 时钟选项5<br>110b - 时钟选项6<br>111b - 时钟选项7 |
| 23-4 | 这个只读的位域是保留的，其值始终为0。 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| - | |
| 3<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 2-0<br>- | 这个只读的位域是保留的，其值始终为0。 |

## 27.6.39 PCC FTM7寄存器（PCC_FTM7）。

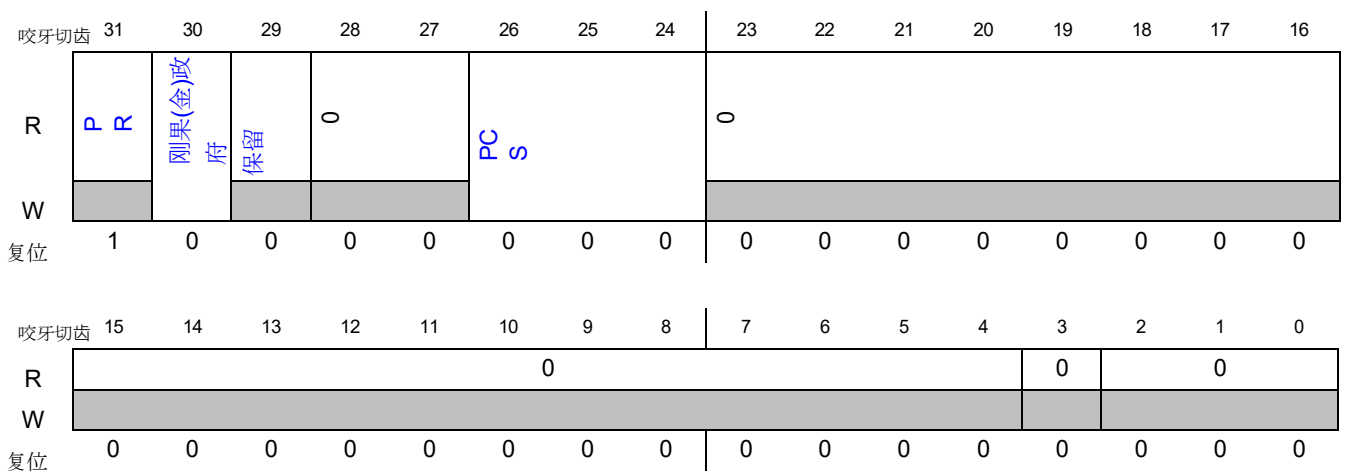
### 27.6.39.1 偏移

| 注册 | 偏移 |
|---|---|
| PCC_FTM7 | 1C4h |

### 27.6.39.2 职能

PCC登记簿

### 27.6.39.3 图示



### 27.6.39.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>公关 | 目前<br>该位显示外设是否存在于该设备上。<br>0b - 周边设备不存在。1b - 外围存在。 |
| 30<br>刚果(金)政府 | 时钟门控制<br>该读/写位启用外设的时钟。<br>0b - 禁用时钟<br>1b - 时钟启用。当前的时钟选择和分频器选项被锁定。 |
| 29<br>- | 这个只读的位域是保留的。这个位可以改变数值，但却是不关心的。 |
| 28-27<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 26-24<br>辽宁省 | 周边时钟源选择<br>这个读/写位域用于支持各种时钟选择的外设。该字段只有在时钟被禁用（CGC=0）<br>时才能被写入。<br>000b - 时钟关闭。可以为这个外设启用一个外部时钟。001b - 时钟选项1<br>010b - 时钟选项2<br>011b - 时钟选项3<br>100b - 时钟选项4<br>101b - 时钟选项5<br>110b - 时钟选项6<br>111b - 时钟选项7 |
| 23-4<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 3<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 2-0<br>- | 这个只读的位域是保留的，其值始终为0。 |

## 27.6.40 PCC CMP0寄存器（PCC_CMP0）。

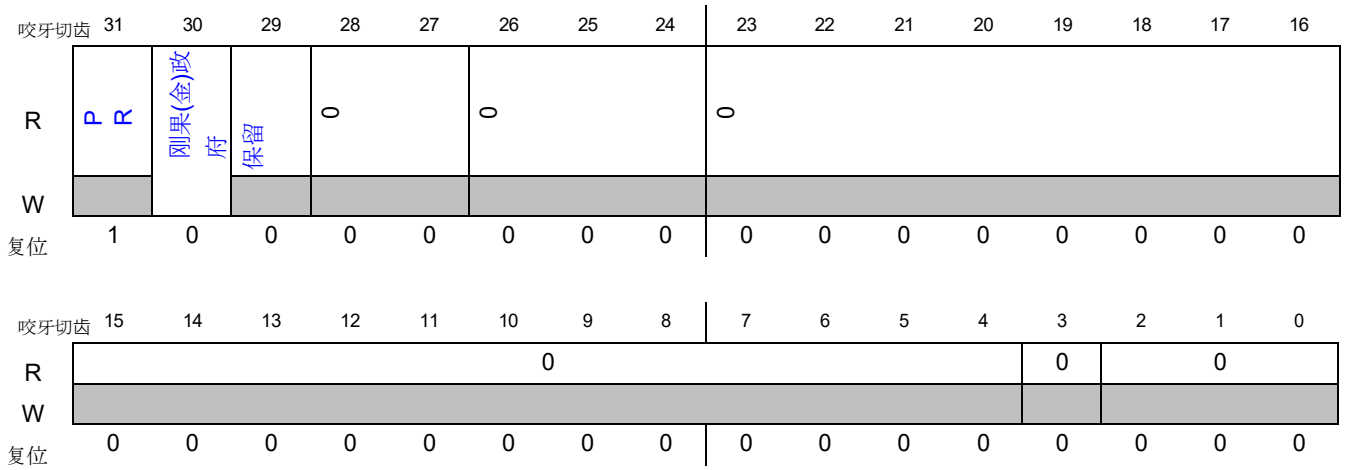
## 27.6.40.1 偏移

| 注册 | 偏移 |
| --- | --- |
| PCC_CMP0 | 1CCh |

## 27.6.40.2 职能

PCC登记簿

## 27.6.40.3 图示




## 27.6.40.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>公关 | 目前<br>该位显示外设是否存在于该设备上。<br>0b - 周边设备不存在。1b - 外围存在。 |
| 30<br>刚果(金)政府 | 时钟门控制<br>该读/写位启用外设的时钟。<br>0b - 禁用时钟<br>1b - 时钟启用。当前的时钟选择和分频器选项被锁定。 |
| 29<br>- | 这个只读的位域是保留的。这个位可以改变数值，但却是不关心的。 |
| 28-27<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 26-24<br>- | 这个只读的位域是保留的，其值始终为0。 |

| | |
|---|---|
| 23-4<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 3 | 这个只读的位域是保留的，其值始终为0。 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| - | |
| 2-0<br>- | 这个只读的位域是保留的，其值始终为0。 |

## 27.6.41 PCC QSPI寄存器（PCC_QSPI）。

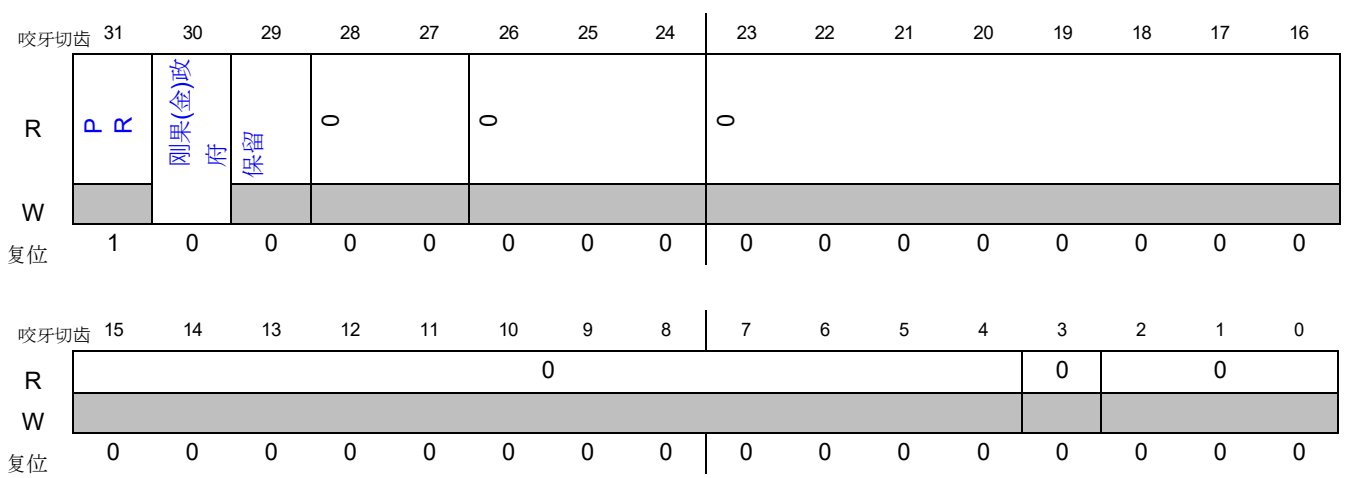
### 27.6.41.1 偏移

| 注册 | 偏移 |
|---|---|
| PCC_QSPI | 1D8h |

### 27.6.41.2 职能

PCC登记簿

### 27.6.41.3 图示




### 27.6.41.4 领域

**PCC寄存器描述**

| 场地 | 职能 |
| --- | --- |
| 31 | 目前 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| 公关 | 该位显示外设是否存在于该设备上。<br>0b - 周边设备不存在。1b - 外围存在。 |
| 30<br>刚果(金)政府 | 时钟门控制<br>该读/写位启用外设的时钟。<br>0b - 禁用时钟<br>1b - 时钟启用。当前的时钟选择和分频器选项被锁定。 |
| 29<br>- | 这个只读的位域是保留的。这个位可以改变数值，但却是不关心的。 |
| 28-27<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 26-24<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 23-4<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 3<br>- | 这个只读的位域是保留的，其值始终为0。 |
| 2-0<br>- | 这个只读的位域是保留的，其值始终为0。 |

## 27.6.42 PCC ENET寄存器（PCC_ENET）。

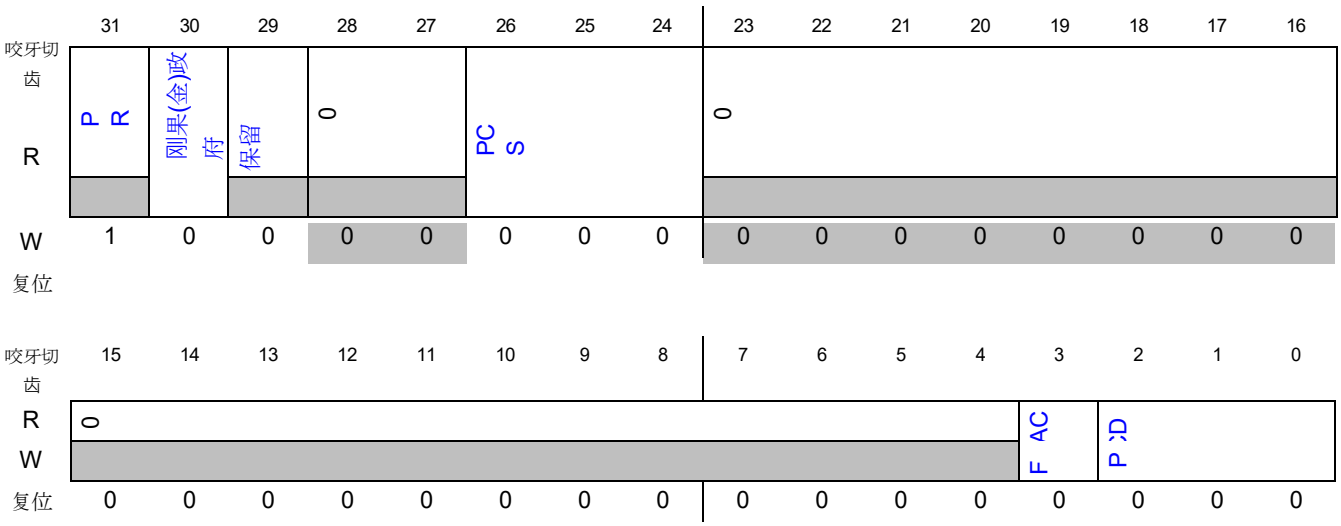
### 27.6.42.1 偏移

| 注册 | 偏移 |
|---|---|
| PCC_ENET | 1E4h |

### 27.6.42.2 职能

PCC登记簿

## 27.6.42.3 图示




## 27.6.42.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>公关 | 目前<br>该位显示外设是否存在于该设备上。<br>0b - 周边设备不存在。1b - 外围存在。 |
| 30<br>刚果(金)政府 | 时钟门控制<br>该读/写位启用外设的时钟。<br>0b - 禁用时钟<br>1b - 时钟启用。当前的时钟选择和分频器选项被锁定。 |
| 29<br>- | 这个只读的位域是保留的。这个位可以改变数值，但却是不关心的。 |
| 28-27<br>- | 这个只读的位域是保留的，其值始终为0。 |

| | |
|---|---|
| 26-24<br>辽宁省 | 周边时钟源选择<br>这个读/写位域用于支持各种时钟选择的外设。该字段只有在时钟被禁用（CGC=0）时<br>才能被写入。<br>000b - 时钟已关闭。<br>001b - 时钟选项1<br>010b - 时钟选项2<br>011b - 时钟选项3<br>100b - 时钟选项4<br>101b - 时钟选项5<br>110b - 时钟选项6<br>111b - 时钟选项7 |
| 23-4 | 这个只读的位域是保留的，其值始终为0。 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| - | |
| 3<br>FRAC | 外围时钟分频器分数<br>这个读/写位域为用作时钟源的小数时钟分频器设置分数乘法值。分频器输出时钟=分频器输入时钟x [(FRAC+1)/(PCD+1)]。<br>这个字段只有在时钟被禁用（CGC=0）时才能写入。<br>**注意：**当除以1（PCD=000）时，不要设置FRAC位，否则，输出时钟被禁用。<br>0b - 分数值为0。 1b - 分数值为1。 |
| 2-0<br>PCD | 周边时钟分频器选择<br>这个读/写位域用于需要时钟分频器的外设。分频器输出时钟=分频器输入时钟x [(FRAC+1)/(PCD+1)]。<br>这个字段只有在时钟被禁用（CGC=0）时才能写入。<br>000b - 除以 1.<br>001b - 除以 2.<br>010b - 除以 3.<br>011b - 除以 4.<br>100b - 除以 5.<br>101b - 除以 6.<br>110b - 除以 7.<br>111b - 除以 8. |

# 第二十八章
# 时钟监测单元（CMU）

## 28.1 简介

频率检查模块检查输入监控时钟的频率是否在用户指定的频率范围内。该模块需要另一个时钟，作为频率检查操作的基础参考。这个参考时钟需要一直在文件规定的范围内，以便模块的正确操作。
对于有问题的参考时钟，不能保证模块的行为。

该模块在参考时钟的'X'个时钟周期的时间内对被监测时钟的时钟周期进行计数。参考时钟的'X'个周期是用户可编程的。在这段时间后，被监测的时钟的最终计数与用户编程的上限和下限进行比较，以检查计数是否在限制之内。如果被监测时钟的最终计数小于低阈值，模块将产生FLL中断。同样地，如果被监测的时钟的最终计数大于高阈值，模块就会产生FHH中断。

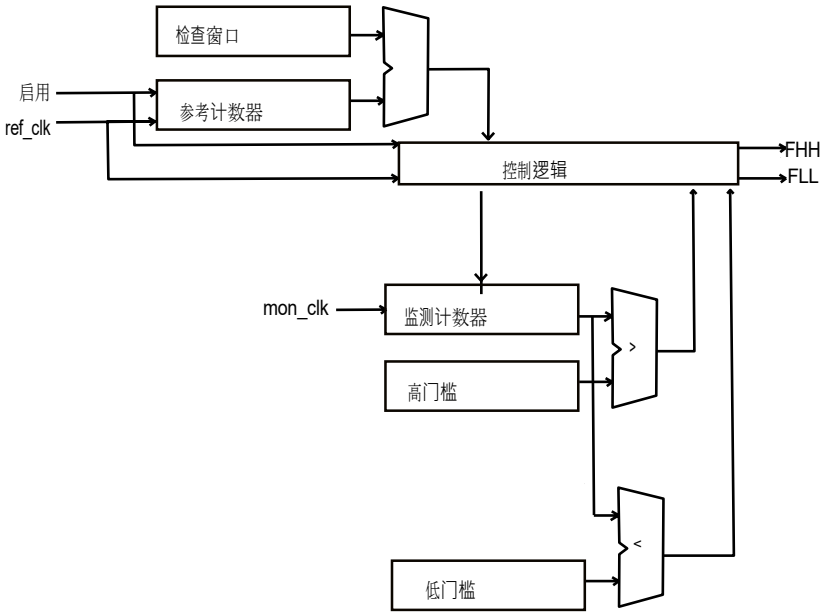



图28-1.框图

## 28.2 寄存器描述

### 28.2.1 CMU内存地图

本节描述了所有CMU寄存器的地址顺序。每个描述包括一个标准的寄存器图和相关的位域描述。

DEFAULT_NICKNAME基础地址。0h

| 偏移 | 注册 | 宽度 (单位：比特) | 访问 | 重置值 |
|---|---|---|---|---|
| 0h | CMU频率检查全局配置寄存器（CMU_FCGC R）。 | 32 | RW | 0000_0000h |
| 4h | CMU频率检查参考计数配置寄存器（CMU_FCRCCR）。 | 32 | RW | 0000_0000h |
| 8h | CMU频率检查高阈值配置寄存器（CMU_FCHTCR）。 | 32 | RW | 00FF_FFFFh |
| 丘吉尔 | CMU频率检查低阈值配置寄存器（CMU_FCLTCR）。 | 32 | RW | 0000_0000h |
| 10h | CMU频率检查状态寄存器（CMU_FCSR）。 | 32 | W1C | 0000_0000h |

| | | | | |
|---|---|---|---|---|
| 14h | CMU频率检查中断/事件启用寄存器(CMU_FCIE R) | 32 | RW | 0000_0000h |

# 28.2.2 CMU频率检查全局配置寄存器（CMU_FCGCR）。

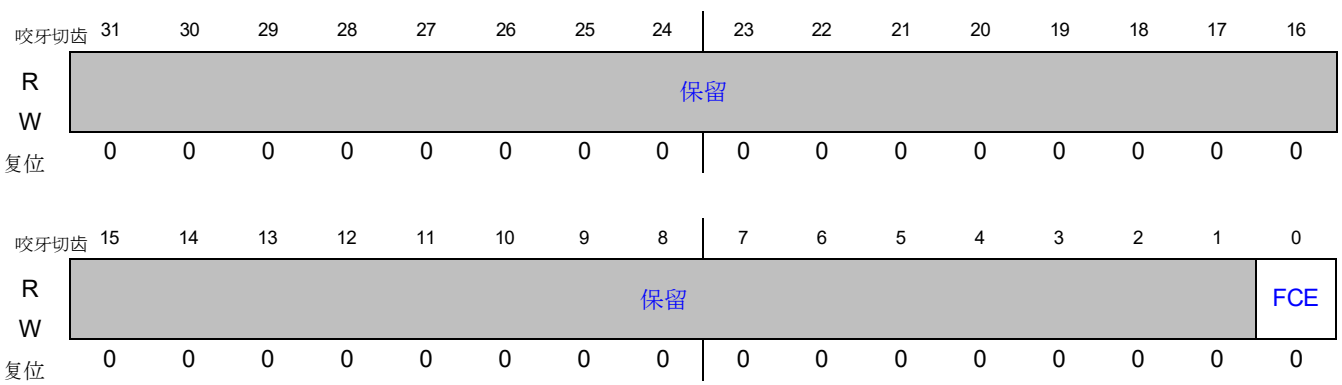
## 28.2.2.1 偏移

| 注册 | 偏移 |
|---|---|
| CMU_FCGCR | 0h |

## 28.2.2.2 职能

该寄存器用于控制模块级的配置。

## 28.2.2.3 图示




## 28.2.2.4 领域

| 场地 | 职能 |
|---|---|
| 31-1<br>- | 保留。 |
| 0<br>FCE | 启用频率检查<br>当频率检查被启用时，向其他配置寄存器的写入被禁止。首先禁用频率检查以写入其他配置寄存器。<br>0b - 禁用频率检查 1b - 启用频率<br>检查 |

## 28.2.3 CMU频率检查参考计数配置寄存器（CMU_FCRCCR）。

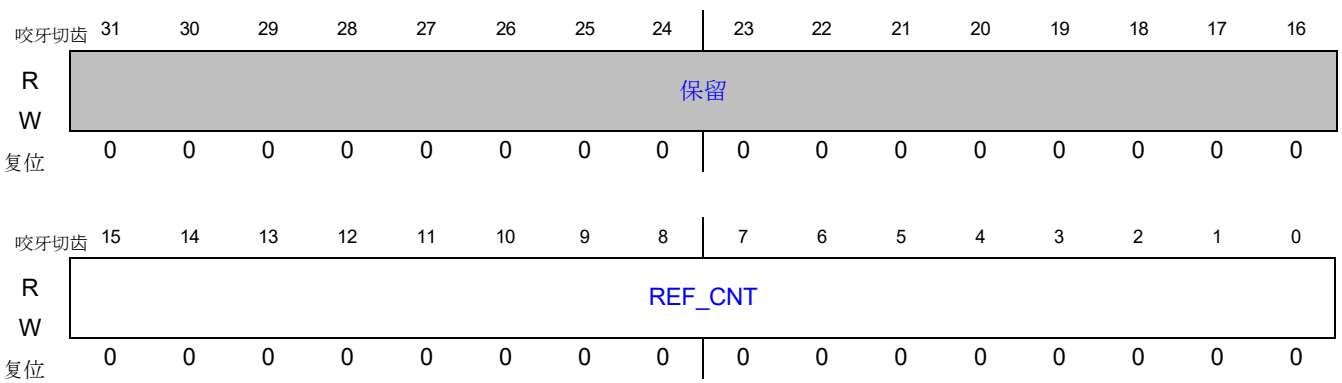
### 28.2.3.1 偏移

| 注册 | 偏移 |
|---|---|
| CMU_FCRCCR | 4h |

### 28.2.3.2 职能

该寄存器用于编程频率检查窗口的参考计数持续时间。注意：只有当CMU_FCGCR[FCE]='0'时才能写入该寄存器。

### 28.2.3.3 图示




### 28.2.3.4 领域

| 场地 | 职能 |
|---|---|
| 31-16<br>- | 保留。 |
| 15-0<br>REF_CNT | 参考时钟计数<br><br>频率检查运行的参考时钟的总计数。这些位定义了一个频率检查窗口的持续时间。 |

## 28.2.4 CMU频率检查高阈值配置寄存器（CMU_FCHTCR）。

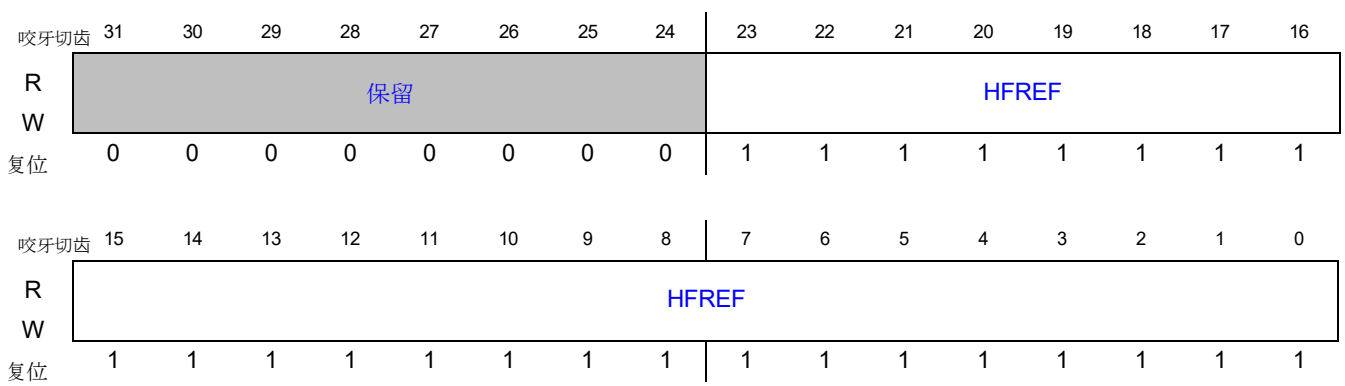
### 28.2.4.1 偏移

| 注册 | 偏移 |
|---|---|
| CMU_FCHTCR | 8h |

### 28.2.4.2 职能

该寄存器用于对受监控的时钟计数器的高阈值进行编程。注意：只有当CMU_FCGCR[FCE]='0'时才能写入该寄存器。

### 28.2.4.3 图示




### 28.2.4.4 领域

| 场地 | 职能 |
|---|---|
| 31-24<br>- | 保留。 |
| 23-0<br>HFREF | 高频参考阈值<br>这些位决定了被监控时钟频率的高参考值。该参考值由以下因素给出。(被监测的时钟频率/参考时钟频率 |

| | |
|---|---|
| | )*REF_CNT + 高门限值 |

## 28.2.5 CMU频率检查低阈值配置寄存器（CMU_FCLTCR）。

### 28.2.5.1 偏移

| 注册 | 偏移 |
|---|---|
| CMU_FCLTCR | 丘吉尔 |

### 28.2.5.2 职能

该寄存器用于对受监控的时钟计数器的低门限值进行编程。注意：只有当CMU_FCGCR[FCE]='0'时才能写入该寄存器。

### 28.2.5.3 图示

| 咬牙切齿 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R<br>W | 保留 | | | | | | | | LFREF | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 咬牙切齿 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R<br>W | LFREF | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

### 28.2.5.4 领域

| 场地 | 职能 |
|---|---|
| 31-24<br>- | 保留。 |
| 23-0<br>LFREF | 低频参考阈值<br>这些位决定CLKMN1频率的低参考值。该参考值由以下因素给出。(被监测时钟的频率/参考时钟的频率 |

| | |
|---|---|
| | )*REF_CNT - 低门限值 |

# 28.2.6 CMU频率检查状态寄存器（CMU_FCSR）。

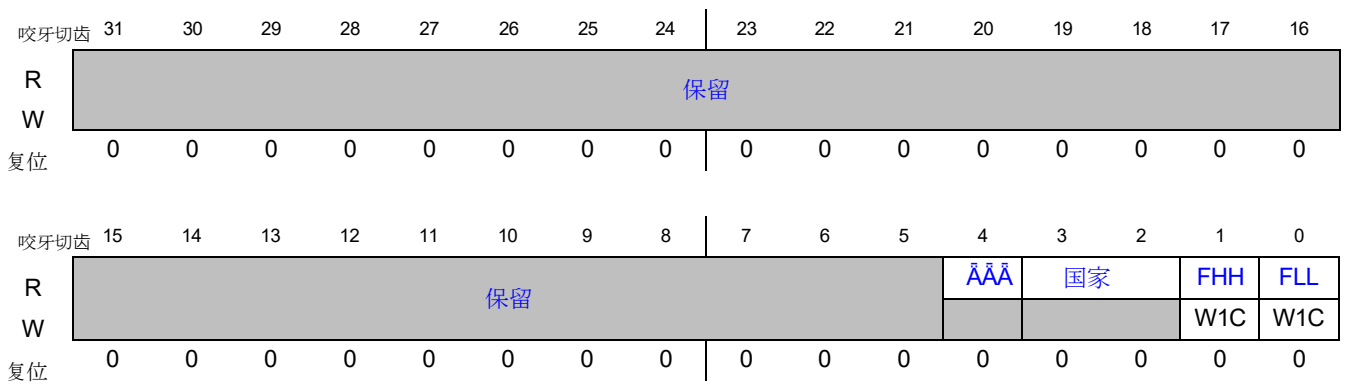
## 28.2.6.1 偏移

| 注册 | 偏移 |
|---|---|
| CMU_FCSR | 10h |

## 28.2.6.2 职能

该寄存器给出了模块的内部状态

## 28.2.6.3 图示




## 28.2.6.4 领域

| 场地 | 职能 |
|---|---|
| 31-5<br>- | 保留。 |

| | |
|---|---|
| 4<br>ĀĀĀ | 运行状态<br>该位给出了内部模块逻辑的状态。在启用频率检查操作（CMU_FCGCR[FCE}=1）后，需要10个总线时钟周期+10个参考时钟周期的延迟，该位才显示运行状态。<br>0b - 频率检查已停止 1b - 频率检查正在运行 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| 3-2<br>国家 | 这些位提供了内部状态信息。注意：如果系统要禁用频率检查操作，那么软件应该读取这个寄存器，当这些位为 "11 "时，改变CMU_FCGCR[FCE]="0"。<br>00b - 配置状态 01b - 初始化状态<br>10b - 初始化等待状态 11b - 频率检查状态 |
| 1<br>FHH | 频率高于高频参考阈值 事件状态<br>当CLKMN1的频率高于HFREF值时，该位由硬件设置，并且CLKMN1在FCE的信号下被 "打开"。它可以通过软件写入 "1 "来清除。<br>0b - 没有FHH事件<br>1b - 发生了FHH事件 |
| 0<br>FLL | 频率低于低频参考阈值 事件状态<br>当CLKMN1的频率低于LFREF值时，该位由硬件设置，并且在FCE的信号下，CLKMN1被 "打开"。它可以通过软件写入 "1 "来清除。<br>0b - 没有FLL活动<br>1b - 发生了FLL事件 |

## 28.2.7 CMU频率检查中断/事件启用寄存器(CMU_FCIER)

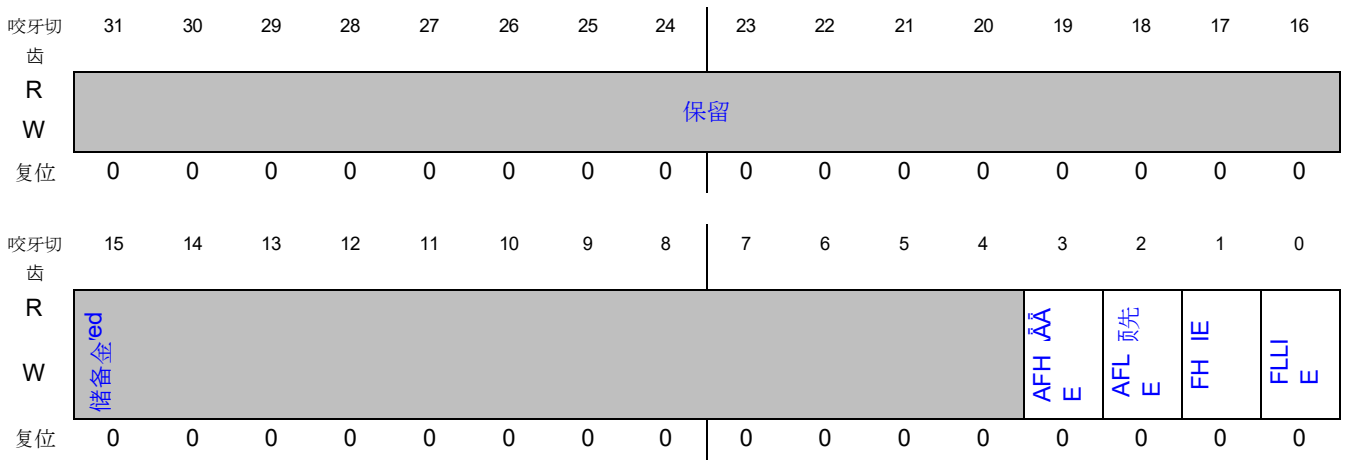
### 28.2.7.1 偏移

| 注册 | 偏移 |
| --- | --- |
| CMU_FCIER | 14h |

### 28.2.7.2 职能

该寄存器掩盖/不掩盖模块的事件状态。注意：只有当CMU_FCGCR[FCE]='0'时才能写入该寄存器。

### 28.2.7.3 图示




### 28.2.7.4 领域

| 场地 | 职能 |
|---|---|
| 31-4<br>- | 保留。 |
| 3<br>辽宁省 | 异步频率高于高频参考阈值事件启用 0b - 异步FHH事件被禁用<br>1b - 启用了异步FHH事件 |
| 2<br>AFLLEE | 异步频率低于低频参考阈值事件启用 0b - 异步FLL事件被禁用<br>1b - 启用异步FLL事件 |
| 1<br>FHHIE | 频率高于高频参考阈值的中断启用 0b - FHH中断被禁用<br>1b - FHH中断被启用 |
| 0<br>辽宁省 | 频率低于低频参考阈值的中断启用 0b - FLL中断被禁用<br>1b - FLL中断被启用 |

## 28.3 功能描述

通过编程CMU_FCGCR[FCE]=1启动频率检查操作。

该模块对被监测的时钟进行连续的定期时钟检查。每个检查窗口涉及以下两个操作。

- 第一阶段。参考时钟计数器在参考时钟的CMU_FCRCCR[REF_CNT]周期内运行，被监测的时钟计数器在与参考时钟计数器相同的时间长度内并行

运行。

- 第二阶段：在参考计数检查窗口结束时，模块需要4个参考时钟周期+4个监测时钟周期来评估FHH和FLL事件。
- 如果监测到的时钟计数大于由字段CMU_FCHTCR[HFREF]决定的高阈值，则产生FHH事件。

如果被监测的时钟频率大于由字段CMU_FCHTCR[HFREF]决定的高阈值，那么模块会设置CMU_FCISR[FHH]位并触发一个可屏蔽的中断。

如果监测到的时钟计数低于由字段CMU_FCLTCR[LFREF]确定的低阈值，就会产生FLL事件。

一旦模块内部产生了FHH/FLL事件，在模块边界上产生FHH/FLL中断之前，还需要3个总线时钟周期。

一旦FHH/FLL事件在模块内部产生，在FHH/FLL异步事件在模块边界产生之前还需要一个参考时钟周期。

### 监测的时钟丢失

如果被监测的时钟在FLL/FHH评估（阶段2）之前丢失，模块通过延长阶段2，等待REF_CNT参考时钟周期的持续时间，然后标记一个FLL事件。

在软件启用后，监测的时钟完全丢失，模块最坏的情况下的FLL响应时间是（2*REF_CNT+8）ref_clk周期+8个总线时钟周期。

## 28.4 编程指南

### 为HFREF和LFREF编程

HFREF/LFREF编程必须考虑可容忍的监测时钟频率变化、最大参考时钟频率变化和固有的模块不精确性。

该模块与用户预期的理想监测时钟周期相比，最大偏差为+/-2个监测时钟周期。

#### 例子

fReference_Clock = 20 MHz

fmonitored_clock = 200 MHz

cmu_fcrccr[ref_cnt] =10

理想的监测时钟计数 = (fmonitored_clock/ fReference_Clock) * REF_CNT = 100

现在，考虑参考时钟和监测时钟的组合规格变化 =
+/- 5%.

模块的固有误差为2个周期。

最小监控时钟计数=（理想监控时钟计数*0.95）-2=93（LFREF）。

最大监控时钟数=（理想监控时钟数*1.05）+2=107（HFREF）。

### 编程 CMU_FCRCCR[REF_CNT]

REF_CNT的编程必须在考虑应用对中断响应时间的要求和对模块的整体精度要求后进行。REF_CNT的值越高，检查窗口越长，因此FLL/FHH中断响应越慢；REF_CNT的值越低，检查窗口越短，FLL/FHH响应越快。

**注意事项**

对REF_CNT进行编程，以确保在该时间段内监测的时钟至少有10个计数。

### 模块编程顺序

建议模块的编程顺序如下。

1. 对CMU_FCRCCR[REF_CNT]进行编程，定义频率检查窗口的持续时间，单位为ref_clk周期。
2. 对CMU_FCHTCR[HFREF]和CMU_FCLTCR[LFREF]进行编程，以设置被监控时钟的允许频率范围。
3. 对CMU_FCIER寄存器进行编程，启用/禁用FHH和FLL事件。
4. 上述3个步骤可以按任何顺序进行。
5. 一旦完成上述步骤，对CMU_FCGCR[FCE]=1进行编程，开始频率检查操作。
   注意：在CMU_FCRCCR、CMU_FCHTCR和CMU_FCLTCR被设置后，向CMU_FCGCR[FCE]的写操作被禁止。

**注意事项**

CMU_FCGCR[FCE]=0只有在CMU_FCSR[RS]=1时才能编程。

# 第29章
# 存储器和存储器接口

## 29.1 简介

本章讨论了存储器和存储器接口的配置，如闪存、闪存控制器和SRAM。其他相关信息可以在以下章节中找到。

- 蒙特利尔大学
- 梦之城_梦之城娱乐_梦之城国际娱乐_梦之城国际娱乐平台
- eDMA
- EIM
- ERM

## 29.2 闪存控制器和闪存模块

关于这些模块的详情，请参见。

- 闪存控制器（FMC）
- 闪存模块（FTFC）

## 29.3 SRAM配置

本节总结了该模块在芯片中的配置情况。

**表29-1.相关信息的参考链接**

| 主题 | 相关模块 | 参考资料 |
|---|---|---|
| 系统内存图 | | 见附件S32K1xx_memory_map.xlsx |
| 计时 | 系统时钟发生器 | 时钟分布 |
| ARM Cortex-M4内核 | | ARM Cortex-M4F内核 |

表继续在下一页...

表29-1.相关信息的参考链接（续）

| 主题 | 相关模块 | 参考资料 |
|---|---|---|
| 系统MPU | | 内存保护单元（MPU） |

## 29.3.1 SRAM尺寸

该芯片包含与ARM Cortex-M4F内核紧密耦合的SRAM。片上SRAM在内存图中被分成连续的SRAM_L和SRAM_U区域。

- SRAM_L被锚定在1FFF_FFFFh，并占据了这个结束地址之前的空间。
- SRAM_U被锚定在2000_0000h，并占据了这个起始地址之后的空间。

**注意事项**

在ARM Cortex-M4F架构中不支持跨越2000_0000h边界的错位访问。

在SIM中，SDID[RAMSIZE]标识了一个设备上的SRAM总量。对于该系列的每个芯片，下表显示了该总数中每个SRAM区域的大小。

表29-2.片上SRAM尺寸

| 芯片 | SRAM_L (KB)[1] | SRAM_U (KB)[1] | FlexRAM (KB)[1] | 总SRAM（KB） |
|---|---|---|---|---|
| S32K116 | [1][2] | 14 | [2][2] | 17 |
| S32K142 | [1][2] | 22 | [2][2] | 25 |
| S32K142 | 16 | 12 | 4 | 32 |
| S32K144 | 32 | 28 | 4 | 64 |
| | 28 | 16 | 4 | 48 |
| S32K146 | 64 | 60 | 4 | 128 |
| | 48 | 44 | 4 | 96 |
| S32K148 | 128 | 124 | 4 | 256 |
| | 96 | 92 | 4 | 192 |

1. 具体地址见附件S32K1xx_memory_map.xlsx
2. 1KB用于MTB。如果不用于MTB，可以作为系统RAM使用。不受ECC保护

**注意事项**

对于用作系统RAM的FlexRAM，不产生ECC或访问错误。在SRAM_U结束地址之后的大小等于FlexRAM（4KB）的

区域被保留，但对这个区域不产生访问错误。

## 29.3.2 SRAM的可访问性

SRAM_L和SRAM_U都有一个前门端口和一个后门端口。

下表总结了每个端口的主控访问和访问的性能。下图说明了主控访问的情况。

**表29-3.SRAM的可访问性和性能**

| SRAM区域 | 前门端口 | | 后门端口 | |
|---|---|---|---|---|
| | 主人访问 | 业绩 | 主人访问 | 业绩 |
| SRAM_L | Cortex-M4F核心代码总线 | 单周期访问 | 非核心总线主站 | 读取时至少两个周期，写入时至少一个周期 |
| SRAM_U | Cortex-M4F核心系统总线 | 单周期访问 | 非核心总线主站 | 读取时至少两个周期，写入时至少一个周期 |

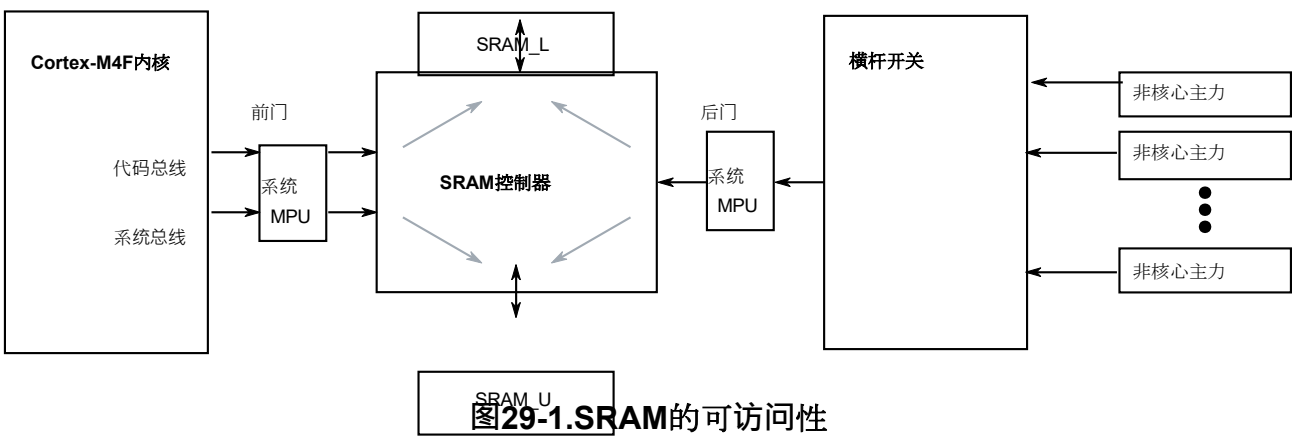



**图29-1.SRAM的可访问性**

以下对SRAM_L和SRAM_U的同时访问可能发生。

- 核心代码总线对SRAM_L的访问和核心系统总线对SRAM_U的访问
- 核心代码总线访问SRAM_L，非核心总线主站访问SRAM_U
- 核心系统总线访问SRAM_U，非核心总线主站访问SRAM_L

下表说明了这些情况。

**表29-4.SRAM的同时访问量**

| 核心代码总线访问 | 核心系统总线访问 | 非核心总线主控访问 |
|---|---|---|
| SRAM_L | SRAM_U | - |

*表继续在下一页...*

表29-4.SRAM的同时访问（续）

| 核心代码总线访问 | 核心系统总线访问 | 非核心总线主控访问 |
|---|---|---|
| SRAM_L | - | SRAM_U |
| - | SRAM_U | SRAM_L |

**注意事项**

突发访问不能发生在分隔两个SRAM阵列的2000_0000h边界上。对于突发访问，这两个阵列应被视为独立的内存范围。

为了补充主系统RAM，FlexRAM可以作为系统RAM使用。然而，FlexRAM的访问涉及额外的系统延迟。

### 29.3.3 SRAM仲裁和优先级控制

核心平台控制寄存器（MCM_CPCR）控制两个SRAM阵列的仲裁和优先级方案。

### 29.3.4 SRAM保留：电源模式和复位

在所有的电源模式下，SRAM的电源都能保持。

为了在功能复位期间保留和访问SRAM的内容，编写应用程序代码来执行以下序列。

1. 配置系统复位中断使能寄存器(RCM_SRIE)以延迟复位，而在每次复位请求时产生一个中断。
2. 在复位请求的中断中，应用代码将芯片配置为进入运行模式，然后将芯片控制寄存器（CHIPCTL）中的SRAMU_RETEN和SRAML_RETEN都写成0。这个逻辑确保在这些位被写入后，任何对SRAM的尝试性访问都会被阻止，从而防止在复位断言期间的内存损坏。
3. 芯片保持在运行模式下，直到复位指令。

**注意事项**

在软件初始化过程中，确保不对保留内容的RAM进行访问。

4. 退出复位后，读取芯片控制寄存器（CHIPCTL）中的SRAMU_RETEN和SRA

ML_RETEN，并将其写入1，以允许对SRAM的访问。

# 第30章
# 本地存储器控制器（LMEM）

## 30.1 特定芯片的LMEM信息

S32K11x系列的变体没有LMEM。

### 30.1.1 LMEM区域描述

关于LMEM区域，见附件S32K1xx_memory_map.xlsx。

**注意事项**

S32K14x上不支持缓存写缓冲区，因为可缓存的内存区域是只读的。

### 30.1.2 LMEM SRAM尺寸

在S32K14x系列的产品中，SRAM的尺寸各不相同。在本章中，SRAM阵列记录了超集SRAM的尺寸。

关于每个S32K14x产品的SRAM尺寸和其他细节，见SRAM尺寸。

## 30.2 简介

本地存储器控制器为处理器提供紧密耦合的处理器本地存储器和通往所有从属存储器空间的总线路径。

### 30.2.1 框图

该处理器有一个改进的32位哈佛总线结构。使用32位地址空间，低阶地址（0x0000_0000到0x1FFF_FFFF）使用处理器代码（PC）总线，而高阶地址（0x2000_0000到0xFFFF_FFFF）使用处理器系统（PS）总线。正如总线名称所暗示的，正常操作时，代码访问在PC总线上，数据访问在PS总线上。

该设备已经为PC和PS总线增加了紧密耦合的存储器。这些存储器包括RAM和缓冲器。这些本地存储器提供对RAM和可缓存地址空间的零等待状态访问。

本地存储器控制器包括三个存储器控制器及其附属存储器。

- 通过PC总线的SRAM下位（SRAM_L）控制器
- 通过PS总线的SRAM上部（SRAM_U）控制器
- 通过PC总线的高速缓冲存储器控制器

本地内存控制器有以下32位AMBA_AHB总线。

- 两个输入是用于处理器的改进型哈佛总线--
  处理器代码（PC）和处理器系统（PS）总线。
- 一个输入是用于所有其他总线主站访问SRAM控制器空间的 "后门
  "端口。
- 两个输出端口是CCM（核心代码主控）总线，用于不撞击PC缓存或SRAM_L或不可缓存的PC访问，以及CSM（核心系统主控）总线，用于不撞击SRAM_U的PS引用。

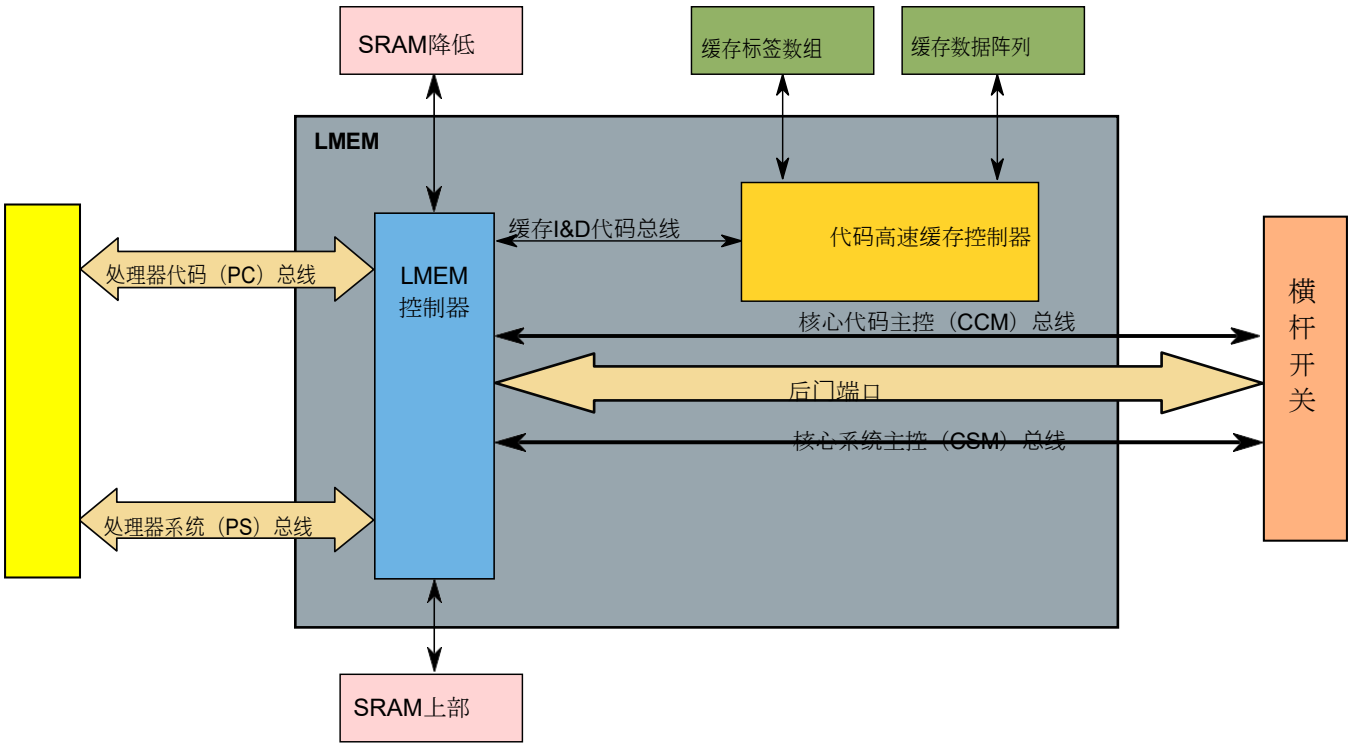



**图30-1.本地存储器控制器框图**

**注意事项**

SRAM和高速缓存控制器位于LMEM内，但这些控制器所使用的单端口同步RAM阵列是外部的。

LMEM包含PC和PS总线的地址解码逻辑。该逻辑将内核对各种系统资源的访问进行路由。这些地址空间是特定于器件的。有关地址空间解码的细节，请参见特定芯片的LMEM信息。

## 30.2.2 缓存功能

缓存是一个包含地址信息（通常被称为标签）和相关数据的高速内存位置块。其目的是为了减少内存访问的平均时间。缓存的运作有两个定位原则。

- 空间位置性--
  对一个位置的访问可能会有相邻位置的访问（例如，顺序指令的执行或数据结构的使用）。
- 时间定位 -
  对内存区域的访问可能会在短时间内重复进行（例如，执行代码循环）。

为了最大限度地减少控制信息的存储量，空间定位属性被用来将几个位置集中在同一个标签下。这种逻辑块通常被称为高速缓存线。

当数据被加载到高速缓存中时，随后的加载和存储的访问时间就会减少，从而带来整体的性能优势。对已经在高速缓存中的信息的访问被称为高速缓存命中，其他访问被称为高速缓存缺失。

通常情况下，缓存是自我管理的，更新是自动发生的。每当处理器想要访问一个可缓存的位置时，都会检查缓存。如果访问是一个高速缓存命中，访问就立即发生。否则，将分配一个位置，并从内存中加载缓存行。不同的缓冲区拓扑结构和访问策略是可能的。然而，它们必须符合底层架构的内存一致性模型。

缓存引入了一些潜在的问题，主要是因为。

- 在程序员通常预期之外的时间发生的内存访问。
- 存在多个物理位置，可以保存一个数据项。本地存储器控制器支持

以下操作模式。

1. Write-through - 对具有这种缓存模式的地址空间的访问是可以缓存的。
   - 如果所有的可缓存空间都是只读空间，缓存将包含只读数据，所有对缓存的写入都会出现故障。请看特定芯片的可缓冲空间信息。
   - 在输入总线上的写通读缺失会导致在输出总线上对一个包含所需地址的16字节对齐的内存地址进行行读。这个漏读数据被加载到高速缓存中，并被标记为有效且未修改。
   - 对一个有效的缓存位置的写通读命中，从缓存中返回数据，没有输出总线访问。
   - 穿透式写失误会绕过缓存并写到输出总线上（对于穿透式模式的空间，写失误时没有分配政策）。
   - 穿透式写命中更新缓存命中数据并写入输出总线。
   - 缓存是处理器本地的，不支持硬件缓存的一致性。如果处理器已经访问了写通区域，而外部总线主控器（如DMA）需要更新这些区域，软件必须首先对任何需要的缓存内存范围进行显式缓存清除，以确保所有被修改的缓存行在被外部主控器修改之前更新其相关的内存，随后的处理器访问将获得更新的内存。
2. 不可缓存 -
   对具有这种缓存模式的地址空间的访问是不可缓存的。这些访问会绕过缓冲区，访问输出总线。

## 30.3 存储器地图/寄存器的定义

缓存程序员模型提供了各种用于配置和控制缓存的寄存器，以及对所有缓存标记和数据存储的间接访问路径。

**注意事项**

**LMEM寄存器只能在监督员模式下访问。**

### 30.3.1 LMEM寄存器描述

#### 30.3.1.1 LMEM内存图

LMEM基础地址。E008_2000h

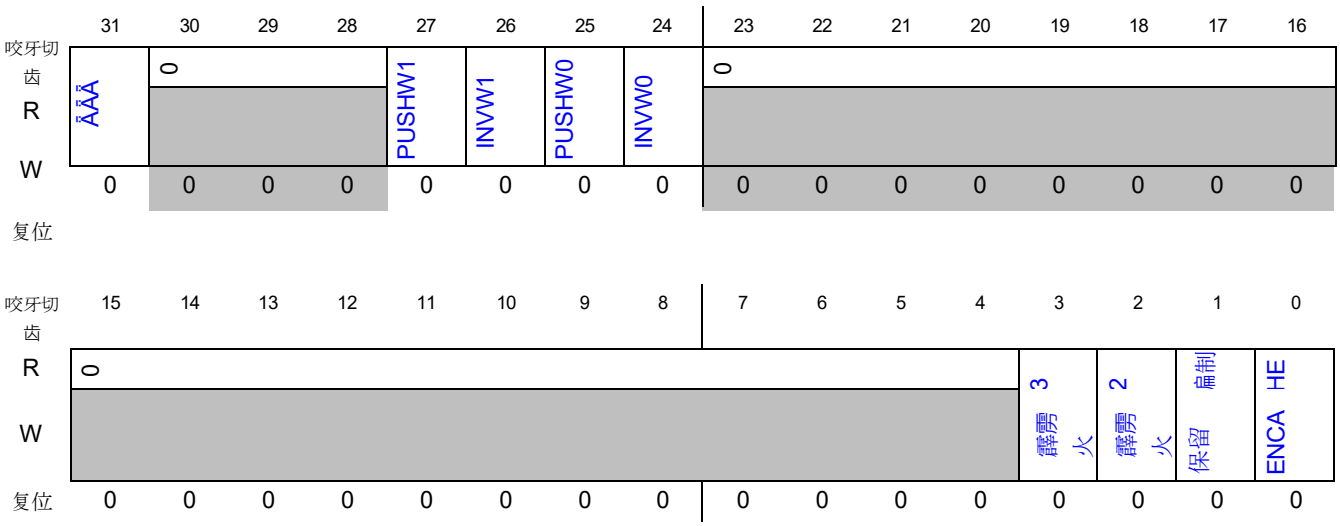
| 偏移 | 注册 | 宽度<br>(单位：比特) | 访问 | 重置值 |
|---|---|---|---|---|
| 0h | 缓存控制寄存器（PCCCR） | 32 | RW | 0000_0000h |
| 4h | 缓存线控制寄存器（PCCLCR） | 32 | RW | 0000_0000h |
| 8h | 缓存搜索地址寄存器（PCCSAR） | 32 | RW | 0000_0000h |
| 丘吉尔 | 缓存读/写值寄存器（PCCCVR） | 32 | RW | 0000_0000h |
| 20h | 缓存区域模式寄存器（PCCRMR）。 | 32 | RW | AA0F_A000h |

#### 30.3.1.2 缓存控制寄存器（PCCCR）

##### 30.3.1.2.1 偏移

| 注册 | 偏移 |
|---|---|
| PCCCR | 0h |

### 30.3.1.2.2 图示




### 30.3.1.2.3 领域

| 场地 | 职能 |
|---|---|
| 31<br>ÃÃÃ | 启动缓存命令<br>设置该位可以启动由第27-24位指示的高速缓存命令。读取该位表明一个命令是否被激活<br>**注意：**在命令完成之前，该位一直保持设置。写为零没有任何作用。<br>0b - 写：无影响。读取：没有激活缓存命令。<br>1b - 写：启动由第27-24位指示的命令。读取：缓存命令激活。 |
| 30-28<br>- | 保留 |
| 27<br>PUSHW1 | 推动方式1<br>0b - 无操作<br>1b - 当设置GO位时，以1的方式推动所有修改过的线。 |
| 26<br>INVW1 | 无效方式1<br>**注意：**如果PUSHW1和INVW1位被设置，那么在设置GO位后，推送方式1中所有修改的行，并使方式1中的所有行无效（清除方式1）。<br>0b - 无操作<br>1b - 当设置GO位时，在方式1中使所有行无效。 |
| 25<br>PUSHW0 | 推动方式0<br>0b - 无操作<br>1b - 当设置GO位时，以0的方式推送所有修改的线路 |
| 24<br>INVW0 | 无效方式0<br>**注意：**如果PUSHW0和INVW0位被设置，那么在设置GO位后，将所有修改过的行推到方式0，并使方式0的所有行无效（清除方式0）。<br>0b - 无操作 |

| | |
|---|---|
| | 1b - 当设置GO位时，使方式0中的所有行无效。 |
| 23-4<br>- | 保留 |
| 3 | 强制在缓存缺失时不分配（必须同时断言PCCR2）。 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| PCCR3 | |
| 2<br>PCCR2 | 强制所有可缓存的空间通过以下方式写入 |
| 1<br>- | 这个字段是保留的。软件应将此位写入0以保持兼容性。 |
| 0<br>ÃÃÃ | 启用缓存<br>0b - 禁用缓存 1b - 启<br>用缓存 |

## 30.3.1.3 缓存线控制寄存器（PCCLCR）

### 30.3.1.3.1 偏移

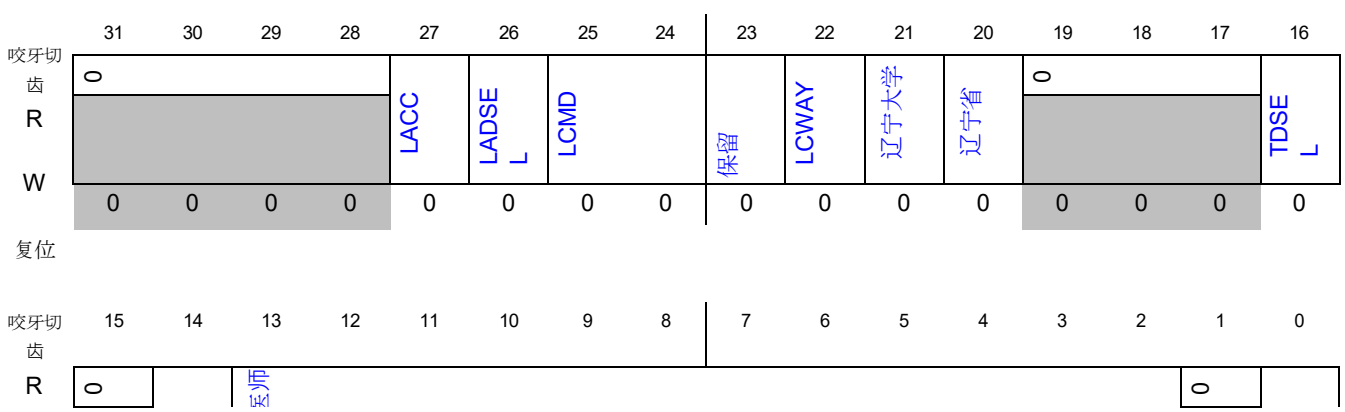
| 注册 | 偏移 |
|---|---|
| PCCLCR | 4h |

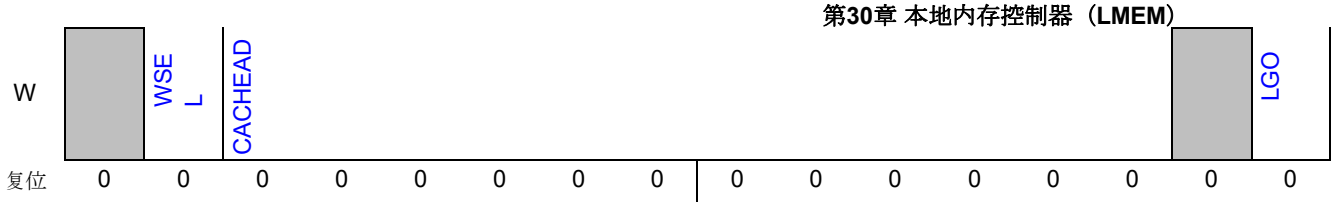
### 30.3.1.3.2 职能

该寄存器定义了特定的行大小的缓存操作，使用特定的缓存行地址或物理地址来执行。

如果指定了一个物理地址，那么缓存的两种方式都会被搜索到，而命令只在命中的方式上执行。

### 30.3.1.3.3 图示




W
WSEL
CACHEAD
LGO
复位
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

## 30.3.1.3.4 领域

| 场地 | 职能 |
|---|---|
| 31-28<br>- | 保留 |
| 27<br>LACC | 线路接入类型<br>0b - 读 1b<br>- 写 |
| 26<br>LADSEL | 线路地址选择<br>当使用缓存地址时，还必须在CLCR[WSEL]中指定方式。<br>当使用物理地址时，两种方式都会被搜索到，只有在命中时才会执行命令。<br>0b - 缓存地址 1b - 物<br>理地址 |
| 25-24<br>LCMD | 线路命令<br>00b - 搜索和读或写 01b - 无效<br>10b - 推动<br>11b - 清除 |
| 23<br>- | 保留 |
| 22<br>LCWAY | 线路命令方式<br>表示行命令所使用的方式。仅在有效位<br>LCIVB=1时适用。 |
| 21<br>辽宁大学 | 行指令初始修改位<br>如果命令使用了缓存地址和方式，那么该位显示了修改后的初始状态<br>如果命令使用的是物理地址，并且是命中，那么这个位显示修改后的初始状态。如果是未命中，则该位读作0。 |
| 20<br>辽宁省 | 行指令初始有效位<br>如果命令使用了缓存地址和方式，那么该位显示有效位的初始状态<br>如果命令使用的是物理地址，并且是命中，那么这个位显示有效位的初始状态。如果未命中，该位的读数为零。 |
| 19-17<br>- | 保留 |
| 16<br>TDSEL | 标签/数据选择<br>选择标签或数据进行搜索和读或写命令。<br>0b - 数据<br>1b - 标签 |

| | |
|---|---|
| 15<br>- | 保留 |
| 14<br>WSEL | 方式选择<br>选择行命令的方式。<br>0b - 方式 0 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| | 1b - 方式1 |
| 13-2<br>CACHEADDR | 缓存地址<br>CLCR[10:4]位用于访问标签数组 CLCR[10:2]位用于访问数据<br>数组。 |
| 1<br>- | 保留 |
| 0<br>LGO | 启动缓存线命令<br>设置该位可以启动由第27-24位指示的高速缓存行命令。读取这个位表示一个行命令是否被激活<br>**注意：**在命令完成之前，该位一直保持设置。写为零没有任何作用。<br>**注意：**该位与CSAR[LGO]共享。<br>0b - 写：无影响。读取：没有行命令生效。<br>1b - 写：启动由第27-24位指示的行命令。读取：行命令激活。 |

## 30.3.1.4 缓存搜索地址寄存器（PCCSAR）

### 30.3.1.4.1 偏移

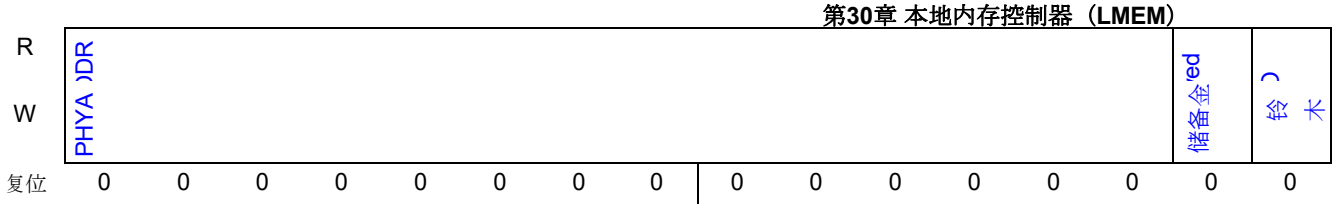
| 注册 | 偏移 |
|---|---|
| PCCSAR | 8h |

### 30.3.1.4.2 职能

CSAR寄存器用于定义显式缓存地址或CLCR[LADSEL]位中指定的行大小命令的物理地址。

### 30.3.1.4.3 图示

| 咬牙切齿 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R<br>W | PHYADDR | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 咬牙切齿 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |


R
W
PHYA )DR
储备金'ed
铃 木
复位
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

#### 30.3.1.4.4 领域

| 场地 | 职能 |
|---|---|
| 31-2<br>PHYADDR | 物理地址<br>PHYADDR代表系统地址的[31:2]位。CSAR[31:11]位用于标<br>签比较<br>CSAR[10:4]位用于访问标签数组<br>CSAR[10:2]位用于访问数据数组 |
| 1<br>- | 保留 |
| 0<br>LGO | 启动缓存线命令<br>设置该位可以启动由第27-24位指示的高速缓存行命令。读取这个位表示一个行命令是否被激活<br>**注意：**在命令完成之前，该位一直保持设置。写为零没有任何作用。<br>**注意：**该位与CLCR[LGO]共享。<br>0b - 写：无影响。读取：没有行命令生效。<br>1b - 写：启动由CLCR[27:24]位指示的行命令。读取：行命令激活。 |

### 30.3.1.5 缓存读/写值寄存器（PCCCVR）

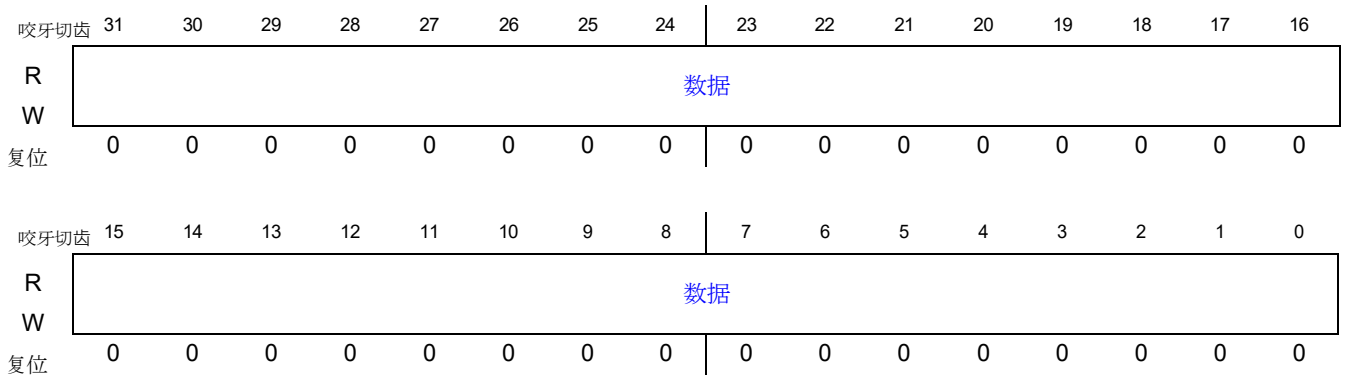
#### 30.3.1.5.1 偏移

| 注册 | 偏移 |
|---|---|
| PCCCVR | 丘吉尔 |

#### 30.3.1.5.2 职能

CCVR寄存器用于为CLCR寄存器中指定的命令提供写数据或返回读数据。

### 30.3.1.5.3 图示




### 30.3.1.5.4 领域

| 场地 | 职能 |
|---|---|
| 31-0<br>数据 | 缓存读/写数据<br>对于标签搜索，读或写。<br>• CCVR[31:11]位用于标签阵列的R/W值<br>• CCVR[10:4]位在读取时用于标签设置地址；在写入时未使用。<br>• CCVR[3:2]位被保留。<br>• CCVR[1]标签修改位<br>• CCVR[0] 标签有效位<br>用于数据搜索、读或写。<br>• CCVR[31:0]位用于数据阵列的R/W值 |

## 30.3.1.6 缓存区域模式寄存器（PCCRMR）。

### 30.3.1.6.1 偏移

| 注册 | 偏移 |
|---|---|
| PCCRMR | 20h |

### 30.3.1.6.2 职能

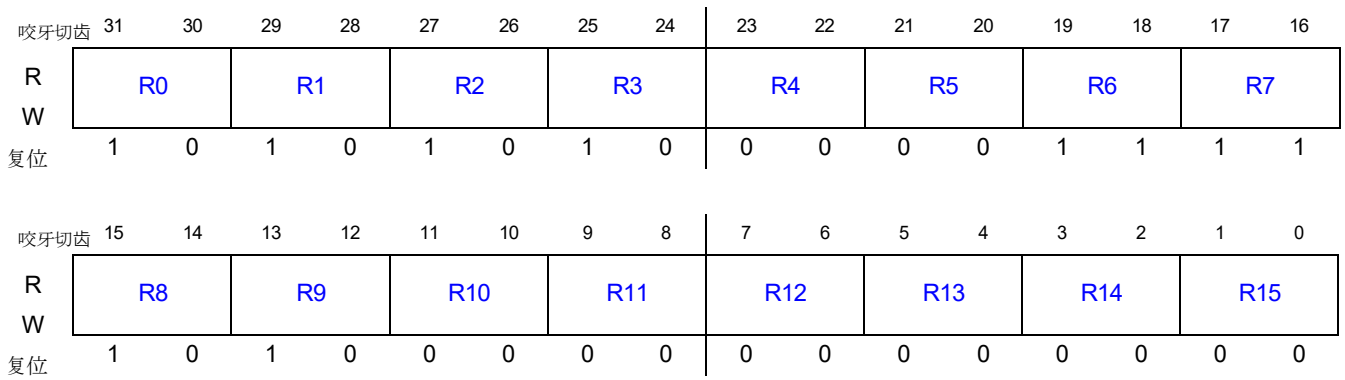
CRMR寄存器允许你在设备的内存地图中降低各种子区域的缓存模式。降级缓存模式将应用于一个内存区域的缓存功能从回写到穿写再到不可缓存。在一个区域被降级后，它的缓存模式只能通过复位来提高，复位后它会恢复到默认状态。

为了保持高速缓存的一致性，对高速缓存模式的改变应该在被改变的地址空间没有被访问或高速缓存被禁用时完成。在改变高速缓存模式之前，完成高速缓存清除所有的命令，以推送并使任何可能已经改变的高速缓存条目无效。

**注意事项**

16个子区域的地址/模块分配是特定于芯片的。有关这些细节，请参见特定芯片的LMEM信息。有些区域可能不被使用（不可缓存），有些区域可能不能回写。

### 30.3.1.6.3 图示




### 30.3.1.6.4 领域

| 场地 | 职能 |
|---|---|
| 31-30<br>R0 | 区域0模式<br>控制区域0的缓存模式 00b - 不可缓存<br>01b - 不可缓存 10b -<br>直写 11b - 回写 |
| 29-28<br>R1 | 区域1模式<br>控制区域1的缓存模式 00b - 不可缓存<br>01b - 不可缓存 10b -<br>直写 11b - 回写 |
| 27-26<br>R2 | 第2区模式<br>控制区域2的缓存模式 00b - 不可缓存<br>01b - 不可缓存 10b -<br>穿透写入 |

表继续在下一页...

| 场地 | 职能 |
| --- | --- |
| | 11b - 回写 |
| 25-24<br>R3 | 第3区模式<br>控制区域3的缓存模式 00b - 不可缓存<br>01b - 不可缓存 10b -<br>直写 11b - 回写 |
| 23-22<br>R4 | 第4区模式<br>控制区域4的缓存模式 00b - 不可缓存<br>01b - 不可缓存 10b -<br>直写 11b - 回写 |
| 21-20<br>R5 | 第5区模式<br>控制区域5的缓存模式 00b - 不可缓存<br>01b - 不可缓存 10b -<br>直写 11b - 回写 |
| 19-18<br>R6 | 第6区模式<br>控制区域6的缓存模式 00b - 不可缓存<br>01b - 不可缓存 10b -<br>直写 11b - 回写 |
| 17-16<br>R7 | 第7区模式<br>控制区域7的缓存模式 00b - 不可缓存<br>01b - 不可缓存 10b -<br>直写 11b - 回写 |
| 15-14<br>R8 | 第8区模式<br>控制区域8的缓存模式 00b - 不可缓存<br>01b - 不可缓存 10b -<br>直写 11b - 回写 |
| 13-12<br>R9 | 第9区模式<br>控制区域9的缓存模式 00b - 不可缓存<br>01b - 不可缓存 10b -<br>直写 11b - 回写 |
| 11-10<br>R10 | 10区模式<br>控制区域10的缓存模式 00b - 不可缓存 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | 01b - 不可缓存 10b - 直写 11b - 回写 |
| 9-8<br>R11 | 第11区模式<br>控制区域11的缓存模式 00b - 不可缓存<br>01b - 不可缓存 10b - 直写 11b - 回写 |
| 7-6<br>R12 | 12区模式<br>控制区域12的缓存模式 00b - 不可缓存<br>01b - 不可缓存 10b - 直写 11b - 回写 |
| 5-4<br>R13 | 第13区模式<br>控制区域13的缓存模式 00b - 不可缓存<br>01b - 不可缓存 10b - 直写 11b - 回写 |
| 3-2<br>R14 | 第14区模式<br>控制区域14的缓存模式 00b - 不可缓存<br>01b - 不可缓存 10b - 直写 11b - 回写 |
| 1-0<br>R15 | 15区模式<br>控制区域15的缓存模式 00b - 不可缓存<br>01b - 不可缓存 10b - 直写 11b - 回写 |

# 30.4 功能描述

## 30.4.1 LMEM功能

本地存储器控制器接收以下请求。

- 处理器代码（PC）总线上的核心主总线请求。

- 处理器系统（PS）总线上的核心主总线请求，以及
- 后门端口上所有其他总线主站的SRAM控制器请求。

本地存储器控制器的地址解码逻辑对这些访问进行路由，并且还提供任何横杠开关从属目标逻辑。最后，本地存储器控制器提供所需的MPU连接，以检查所有SRAM控制器和可缓存访问。

代码缓存的编程模型是通过内核的私有外设总线（PPB）访问的。

#### 30.4.1.1 处理器代码访问

如果处理器代码访问被映射到SRAM_L空间，则被路由到该空间。所有其他的PC访问都被路由到代码高速缓冲存储器控制器。该控制器然后根据需要处理可缓存的访问，同时绕过不可缓存的、缓存写过的、缓存缺失的和缓存维护的访问，使用Master0端口访问CCM总线和横梁开关。

#### 30.4.1.2 处理器系统访问量

如果处理器系统访问被映射到SRAM_U空间，则被路由到该空间。所有其他的PS访问被路由到CCM总线和使用Master1端口的crossbar开关。

#### 30.4.1.3 后门端口访问

所有LMEM后门端口的访问都是针对SRAM控制器的。这些访问根据其具体地址进入SRAM_L或SRAM_U。

### 30.4.2 SRAM功能

#### 30.4.2.1 SRAM配置

下图显示了SRAM控制器是如何配置的。

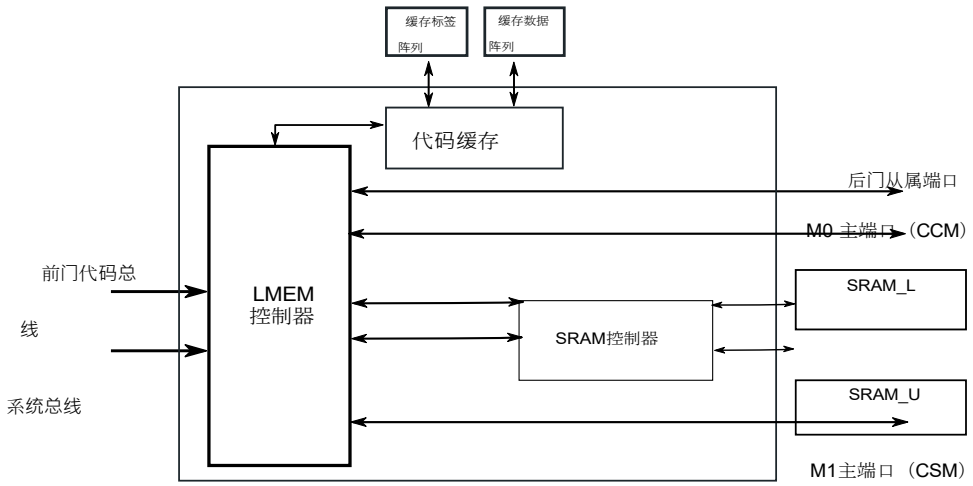



**图30-2.LMEM与SRAM的接口**

### 30.4.2.2 SRAM阵列

片上SRAM被分成两个逻辑阵列：SRAM_L和SRAM_U。SRAM_L的大小为128 KBytes。

SRAM_U的大小为128 KBytes。

那么SRAM_L和SRAM_U的有效地址范围定义为：。

- SRAM_L = (0x20000_0000 - 128 KBytes) 至 0x1fff_ffff
- SRAM_U = 0x2000_0000 到 (0x2000_0000 + 128 KBytes)

SRAML_SIZE和SRAMU_SIZE不一定相等。

### 30.4.2.3 SRAM访问

SRAM被分成两个逻辑阵列，宽度为32位。

- SRAM_L - 可通过内核的代码总线和后门端口访问。
- SRAM_U - 可通过内核的系统总线和后门端口访问。

后门端口使非核心总线主站（如DMA）可以访问SRAM。

图30-3说明了器件内的SRAM访问情况。

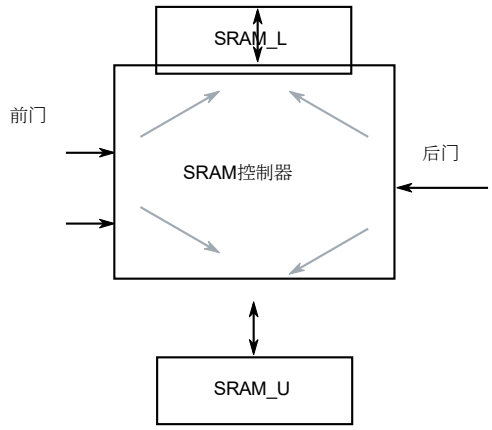



**图30-3.SRAM访问图**

可以对SRAM的不同逻辑半区进行以下的同时访问。

- 核心代码和核心系统
- 核心代码和非核心主控
- 核心系统和非核心主控

**注意事项**

两个非核心主站不能同时访问SRAM。所需的仲裁和序列化是由横梁开关提供的。SRAM_L和SRAM_U的仲裁是由SRAM控制器根据MCM模块中的配置位控制的。

**注意事项**

突发访问不能发生在分隔两个SRAM阵列的0x2000_0000边界上。对于突发访问，这两个数组应被视为独立的内存范围。

## 30.4.3 缓存功能

该设备上的高速缓存结构如下。该高速缓存有一个2路设置关联的高速缓存结构，代码高速缓存的总大小为4KBytes。该高速缓存有32位的地址和数据路径以及16字节的行大小。缓存标记和数据存储使用单端口、同步RAM。

对于Code 4-KByte高速缓存，每个高速缓存TAG功能使用两个128 x 23位的RAM阵列，高速缓存DATA功能使用两个512 x 32位的RAM阵列。缓存TAG条目存储了21位的上层地址，以及每个缓存行的修改位和有效位。
缓存中的DATA条目存储了四个字节的代码或数据。

所有正常的缓冲区访问都使用物理地址。这导致了以下缓存地址的使用。

代码缓存 - 4 KByte大小=（128组）x（16字节行）x（2路集合关联）。

**表30-1.标签缓存地址的使用**

| 标签缓存地址使用 | 代码缓存 |
|---|---|
| 标签命中地址范围 | 地址[31:11] |
| 标签集选择地址范围 | 地址[10:4]用于选择128组中的1组 |
| 未使用 | 地址[3:0] |

**表30-2.数据缓存地址的使用**

| 数据缓存地址使用 | 代码缓存 |
|---|---|
| 未使用 | 地址[31:11] |
| 数据集选择地址范围 | 地址[10:4]用于选择128组中的一组 |
| 32位字选择 | 地址[3:2]用于选择一组中的四个32位字之一 |
| 字节选择 | 地址[1:0]用于选择32位字中的字节。 |

## 30.4.4 缓存控制

代码缓存在复位时被禁用。缓存的标签和数据数组在复位时不被清除。因此，要启用缓存，必须执行缓存命令来清除和初始化所需的标签阵列位，并配置和启用缓存。

### 30.4.4.1 缓存设置命令

缓存设置命令可以对以下内容进行操作。

- 所有的方式0。
- 所有的方式1，或
- 所有的两种方式（完整的缓存）。

缓存设置命令是通过CCR寄存器的上位发起的。缓存设置命令在缓存上执行的操作与缓存使能位CCR[ENCACHE]无关。

缓存设置命令是通过设置CCR[GO]位启动的。这个位也是设置命令的一个繁忙位。当命令处于激活状态时，该位一直被设置，当设置命令完成后，硬件会将其清除。

表30-3中给出了支持的高速缓存设置命令。设置命令的工作方式如下。

- Invalidate - 无条件地清除缓存条目的有效位和修改位。
- 推 -
  如果一个缓存条目是有效的并且被修改过，就推它，然后清除修改位。如果条目无效或未修改，则保持原样。
- 清除 -
  如果一个缓存条目是有效的并且被修改过，则推送它，然后清除有效位和修改位。如果条目无效或未修改，则清除有效位。

**表30-3.缓存设置命令**

| CCR[27:24] | | | | 指挥部 |
|---|---|---|---|---|
| PUSHW1 | INVW1 | PUSHW0 | INVW0 | |
| 0 | 0 | 0 | 0 | NOP |
| 0 | 0 | 0 | 1 | 所有方式无效 0 |
| 0 | 0 | 1 | 0 | 全部推到0 |
| 0 | 0 | 1 | 1 | 清除所有方式0 |
| 0 | 1 | 0 | 0 | 所有方式无效 1 |
| 0 | 1 | 0 | 1 | 无效所有方式1；无效所有方式0（无效缓存）。 |
| 0 | 1 | 1 | 0 | 无效所有方式1；推送所有方式0 |
| 0 | 1 | 1 | 1 | 无效所有方式1；清除所有方式0 |
| 1 | 0 | 0 | 0 | 全部推入1 |
| 1 | 0 | 0 | 1 | 推进所有方式1；使所有方式0无效 |
| 1 | 0 | 1 | 0 | 全程推送1；全程推送0（推送缓存）。 |
| 1 | 0 | 1 | 1 | 推进所有方式1；清除所有方式0 |
| 1 | 1 | 0 | 0 | 清除所有方式1 |
| 1 | 1 | 0 | 1 | 清除所有方式1；使所有方式0无效 |
| 1 | 1 | 1 | 0 | 清除所有方式1；推动所有方式0 |
| 1 | 1 | 1 | 1 | 清除所有方式1；清除所有方式0（清除缓存）。 |

重置后，在使用高速缓存之前，要完成一个无效的高速缓存命令。可以将缓存无效命令与缓存启用结合起来。也就是说，将CCR设置为0x8500_0001将使缓存失效并启用缓存。

### 30.4.4.2 缓存行命令

缓存行命令一次只对缓存中的一个行进行操作。缓存行命令可以使用物理地址或缓存地址执行。

- 一个缓存地址由一个设定地址和一个方式选择组成。line命令作用于指定的高速缓存行。
- 有物理地址的高速缓存行命令首先搜索由以下物理地址位指定的高速缓存集的两路。如果它们命中了，命令就在命中的那条路上执行它们的行动。
  - 对于代码缓存 - [10:4]。

缓存行命令是使用CLCR寄存器中的高位指定的。缓存行命令在缓存上执行的操作与缓存使能位（CCR[ENCACHE]）无关。使用高速缓存地址，命令可以通过CLCR寄存器完全指定。使用物理地址，命令也必须使用CSAR寄存器来指定物理地址。

行缓存命令是通过设置行命令运行位（CLCR[LGO]或CSAR[LGO]）启动的。这个位也是行命令的繁忙位。当命令处于活动状态时，该位一直被设置，当命令完成后，硬件会将其清除。

CLCR[27:24]位选择行命令，如下所示。

**表30-4.缓存线命令**

| CLCR[27:24] | | | 指挥部 |
|---|---|---|---|
| LACC | LADSEL | LCMD | |
| 0 | 0 | 00 | 按缓存地址和方式搜索 |
| 0 | 0 | 01 | 按缓存地址和方式进行无效处理 |
| 0 | 0 | 10 | 按缓存地址和方式推送 |
| 0 | 0 | 11 | 按缓存地址和方式清除 |
| 0 | 1 | 00 | 按实际地址搜索 |
| 0 | 1 | 01 | 通过物理地址进行无效处理 |
| 0 | 1 | 10 | 按实际地址推送 |
| 0 | 1 | 11 | 按实际地址清除 |
| 1 | 0 | 00 | 按缓存地址和方式写入 |
| 1 | 0 | 01 | 保留，NOP |
| 1 | 0 | 10 | 保留，NOP |
| 1 | 0 | 11 | 保留，NOP |
| 1 | 1 | *xx* | 保留，NOP |

#### 30.4.4.2.1 使用缓存地址执行一系列的行命令

只需写入CLCR，就可以执行一系列具有增量缓存地址的行命令。

- 将命令放在CLCR[27:24]中。
- 根据需要设置方式（CLCR[WSEL]）和标签/数据（CLCR[TDSEL]）控制。
- 将缓存地址放在CLCR[CACHEADDR]中，并且
- 设置行指令去位（CLCR[LGO]）。

当一个行命令完成后，按照以下步骤启动下一个命令。

- 递增缓存地址（在第2位递增数据，或在第4位递增行），并且
- 设置行指令去位（CLCR[LGO]）。

#### 30.4.4.2.2 使用物理地址执行一系列的行命令

使用以下步骤执行一系列具有增量物理地址的行命令。

- 写给CLCR。
  - 将命令放在CLCR[27:24]中。
  - 设置标签/数据（CLCR[TDSEL]）控制
- 将物理地址放在CSAR[PHYADDR]中，并设置行指令去位（CSAR[LGO]）。

当一个行命令完成后，按照以下步骤启动下一个命令。

- 递增物理地址（在第2位递增数据，或在第4位递增线），以及
- 设置行指令去位（CSAR[LGO]）。

行命令去位在CLCR和CSAR寄存器之间共享，因此上述步骤可以在对CSAR寄存器的一次写入中完成。

#### 30.4.4.2.3 行命令的结果

在行命令完成后，CLCR寄存器包含了命令所针对的行的初始状态信息。对于带有高速缓存地址的行命令，在从目标高速缓存行执行行命令动作之前读取这一信息。对于带有物理地址的行命令，在行命令动作从命中的高速缓存行执行之前，该信息被读取，或者有初始的有效位

如果命令失误，则清空。一般来说，如果有效指示器（CLCR[LCIVB]）被清零，说明目标行在行命令开始时是无效的，没有进行行操作。

**表30-5.行命令结果**

| CLCR[22:20] | | | 对于高速缓存地址命令 | 对于物理地址命令 |
|---|---|---|---|---|
| LCWAY | 辽宁大学 | 辽宁省 | | |
| 0 | 0 | 0 | 第0条线路是无效的 | 没有击中 |
| 0 | 0 | 1 | 方式0有效，未修改 | 方式0有效，未修改 |
| 0 | 1 | 0 | 第0条线路是无效的 | 没有击中 |
| 0 | 1 | 1 | 方式0有效并修改 | 方式0有效并修改 |
| 1 | 0 | 0 | 方式1行无效 | 没有击中 |
| 1 | 0 | 1 | 方式1有效，未修改 | 方式1有效，未修改 |
| 1 | 1 | 0 | 方式1行无效 | 没有击中 |
| 1 | 1 | 1 | 方式1有效并经过修改 | 方式1有效并经过修改 |

在完成除写之外的行命令时，CCVR（缓存R/W值寄存器）包含了命令所针对的行标记或数据的初始状态信息。对于行命令，CLCR[TDSEL]在标签和数据之间进行选择。如果行命令使用了一个物理地址并错过了，那么数据就不要管了。对于写命令，CCVR持有写数据。

# 第31章
# 杂项系统控制模块（MSCM）。

## 31.1 特定芯片的MSCM信息

### 31.1.1 特定芯片的TMLSZ/TMUSZ信息

MSCM的CPxCFG2寄存器的TMLSZ和TMUSZ字段分别记录了器件的TCMU（SRAM_U）和TCML（SRAM_L）大小。然而，对于总的SRAM大小小于64KB的器件，这些字段显示了该器件部件号的最大可能的SRAM分区大小。比如说。在S32K144上，TMUSZ显示SRAM_U大小为32KB，但实际的SRAM_U大小为28KB或16KB。在任何给定的设备上。关于总的SRAM的确切数量，见SIM的SDID[RAMSIZE]。关于这个总数中SRAM_L和SRAM_U的确切数量，见SRAM大小。

**注意事项**

TCMU和TCML术语可与SRAM_U和SRAM_L互换。

### 31.1.2 芯片特定的寄存器信息

在S32K1xx系列的产品中，一些MSCMM寄存器的复位值有所不同。下表显示了每个产品的这些复位值。

**表31-1.MSCM寄存器的复位值**

| 注册 | S32K116 | S32K118 | S32K142 | S32K144 | S32K146 | S32K148 |
|---|---|---|---|---|---|---|
| CPxCFG2 | TBD | TBD | 0601_0601 | 0701_0701 | 0801_0801 | 0901_0901 |
| CP0CFG2 | TBD | TBD | 0601_0601 | 0701_0701 | 0801_0801 | 0901_0901 |
| OCMDR0 | TBD | TBD | C906_9000 | CA08_9000 | CB08_9000 | DC08_9000 |
| OCMDR1 | TBD | TBD | C706_B000 | C706_B000 | C706_B000 | CA08_B000 |

**注意事项**

上述S32K148的DFLASH值是用于非EEE模式，即完整的FlexNVM（512KB）被用作DFLASH。对于EEE模式，这个大小将是448KB。

## 31.2 概述

杂项系统控制模块（MSCM）包含CPU配置寄存器和片上存储器控制器寄存器。

## 31.3 芯片配置和启动

器件的逻辑定义是通过芯片特定的配置位、支持的内存大小和包装选项来控制。这些配置位共同定义了一个复位配置值（RCON）。

一旦内核获取了所需的复位向量，预计内核和系统配置信息将从一个全局可访问的从属外设中读取，该外设将信息正确转换为更合适的值。更具体地说，内核从一组通用的外设地址访问配置信息，芯片配置逻辑根据请求的处理器进行适当的评估，并为给定的处理器返回适当的值，包括内核识别。

举个例子，有一个32位的只读位置用于识别核心。从这个位置读32位，会返回一个4字符的ASCII字符串。0x434D3401（"Cortex- M4"）。

与核心配置信息相关的编程模型被列为杂项系统控制模块（MSCM）的一部分。它具体包括处理器配置的多个视图；一个视图是对内核通用的，其他视图是对系统中的任何总线主站可用的。

## 31.4 MSCM内存图/寄存器的定义

### 31.4.1 CPU配置内存图和寄存器

MSCM模块的CPU配置部分提供了一组内存映射的只读地址来定义处理器的设置。MSCM编程模型的这一部分只能用特权模式的32位读引用来访问；任何其他的访问类型或大小都以错误结束。如果处理器在逻辑上不包括在芯片配置中，那么对其配置寄存器的读取就会返回0。

CPU配置寄存器是根据逻辑处理器号（不是任何类型的物理端口号）组织的，并被划分为以下相等的部分。

**表31-2.CPU配置寄存器部分**

| 偏移地址 | 职能 |
|---|---|
| 0x000 - 0x01F | 定义了通用的处理器 "x "配置。这个区域只有处理器内核可以访问；非内核总线主站的读取被当作零读（RAZ）访问。 |
| 0x020 - 0x03F | 定义了处理器0（CP0）的配置信息。任何总线主站都可以访问这个区域。 |

试图进行的用户模式或写入访问被错误地终止。

### 31.4.2 MSCM寄存器描述

#### 31.4.2.1 MSCM内存地图

MSCM基础地址。4000_1000h

| 偏移 | 注册 | 宽度<br>(单位：比特) | 访问 | 重置值 |
|---|---|---|---|---|
| 0h | 处理器X类型寄存器（CPxTYPE）。 | 32 | RO | 见描述。 |
| 4h | 处理器X号码寄存器（CPxNUM）。 | 32 | RO | 见描述。 |
| 8h | 处理器X主寄存器（CPxMASTER）。 | 32 | RO | 见描述。 |
| 丘吉尔 | 处理器X计数寄存器（CPxCOUNT）。 | 32 | RO | 0000_0000h |
| 10h | 处理器X配置寄存器0（CPxCFG0）。 | 32 | RO | 见描述。 |

| | | | | |
|---|---|---|---|---|
| 14h | 处理器X配置寄存器1（CPxCFG1）。 | 32 | RO | 见描述。 |
| 18h | 处理器X配置寄存器2（CPxCFG2）。 | 32 | RO | 见描述。 |

*表继续在下一页...*

| 偏移 | 注册 | 宽度<br>(单位：比特) | 访问 | 重置值 |
|---|---|---|---|---|
| 1Ch | 处理器X配置寄存器3（CPxCFG3）。 | 32 | RO | 见描述。 |
| 20h | 处理器0类型寄存器（CP0TYPE）。 | 32 | RO | 434D_3401h |
| 24h | 处理器0号寄存器（CP0NUM）。 | 32 | RO | 0000_0000h |
| 28h | 处理器0主寄存器（CP0MASTER）。 | 32 | RO | 0000_0000h |
| 2Ch | 处理器0计数寄存器（CP0COUNT）。 | 32 | RO | 0000_0000h |
| 30h | 处理器0配置寄存器0（CP0CFG0）。 | 32 | RO | 0400_0000h |
| 34h | 处理器0配置寄存器1（CP0CFG1）。 | 32 | RO | 0000_0000h |
| 38h | 处理器0配置寄存器2（CP0CFG2）。 | 32 | RO | 0901_0901h |
| 3Ch | 处理器0配置寄存器3（CP0CFG3）。 | 32 | RO | 0000_0101h |
| 400h | 片上存储器描述符寄存器（OCMDR0）。 | 32 | RW | DC08_9000h |
| 404h | 片上存储器描述符寄存器（OCMDR1）。 | 32 | RW | CA08_B000h |
| 408h | 片上存储器描述符寄存器（OCMDR2）。 | 32 | RW | C304_D000h |

## 31.4.2.2 处理器X类型寄存器（CPxTYPE）。

### 31.4.2.2.1 偏移

| 注册 | 偏移 |
|---|---|
| CPxTYPE | 0h |

### 31.4.2.2.2 职能

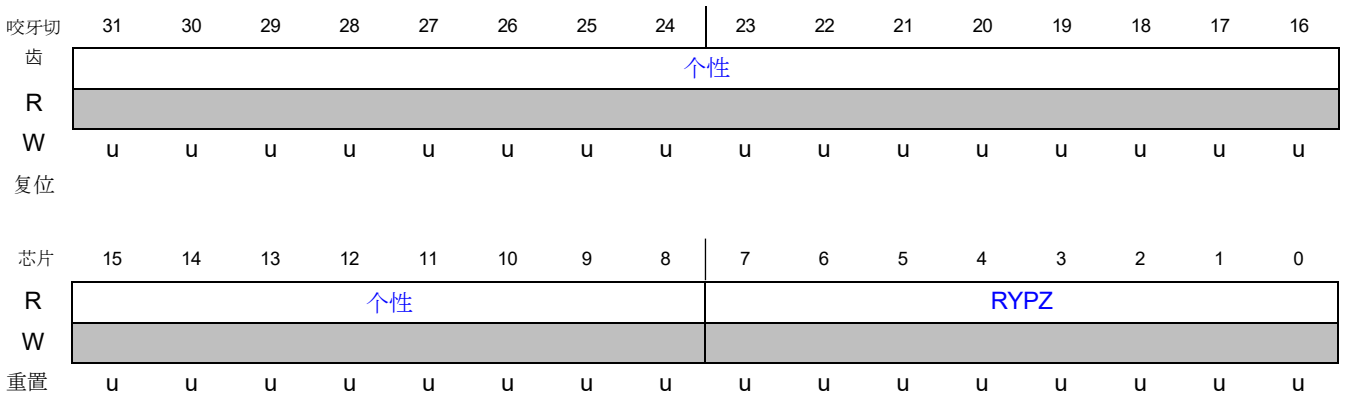
该寄存器提供了一个CPU特定的响应，表明进行访问的内核的个性。32位响应包括3个定义CPU类型的ASCII字符，以及一个定义逻辑修订号的字节。逻辑修订号遵循ARM的rYpZ命名法。

**注意事项**

该设备的CPxTYPE值。

- 如果CPU0正在进行访问 = 0x434D3401
- 如果读访问不是来自CPU，那么读的值是0x00000000

#### 31.4.2.2.3 图示




#### 31.4.2.2.4 领域

| 场地 | 职能 |
|---|---|
| 31-8<br>个性 | 处理器 x 个性<br>这个只读字段定义了CPUx的处理器个性。<br>如果CPUx = Cortex-M0+，那么PERSONALITY = 0x43_4D_30 ("CM0+")。如果CPUx = Cortex-M4，那么PERSONALITY = 0x43_4D_34 ("CM4")。 |
| 7-0<br>RYPZ | 处理器x修订版<br>这个只读字段定义了CPUx的处理器版本：0x00对应于r0p0内<br>核的发布。<br>0x01对应的是r0p1核心版本。<br>... |

### 31.4.2.3 处理器X号码寄存器（CPxNUM）。

#### 31.4.2.3.1 偏移

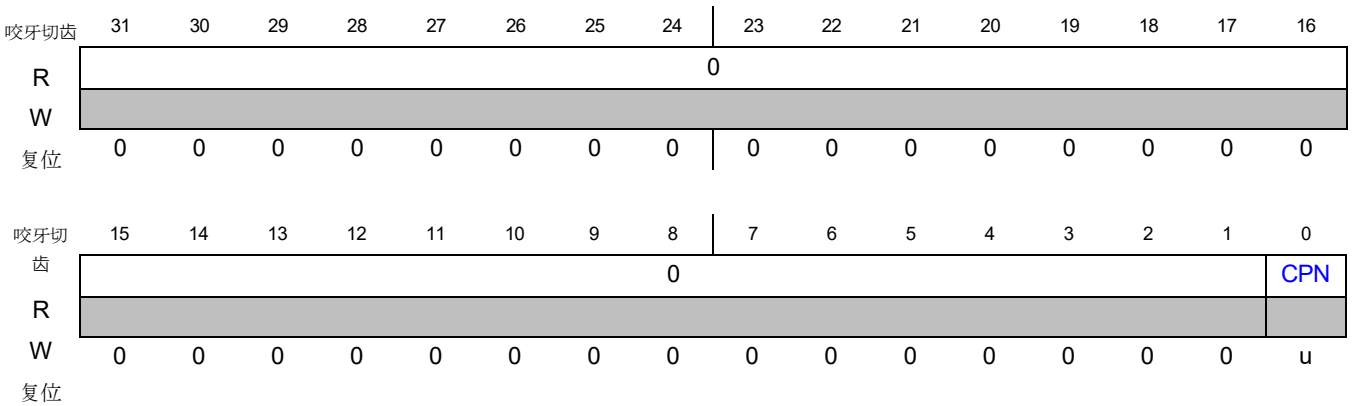
| 注册 | 偏移 |
|---|---|
| CPxNUM | 4h |

### 31.4.2.3.2 职能

该寄存器提供了一个CPU特定的响应，表明进行访问的内核的逻辑处理器编号。逻辑处理器的编号总是0。

如果读访问不是来自CPU，那么读的值是0x00000000。

#### 31.4.2.3.3 图示




#### 31.4.2.3.4 领域

| 场地 | 职能 |
|---|---|
| 31-1<br>- | 保留 |
| 0<br>CPN | 处理器 x 数量<br>这个填零的字定义了CPUx的逻辑处理器编号 如果是单核配置，那么<br>CPN=0 |

### 31.4.2.4 处理器X主寄存器（CPxMASTER）。

#### 31.4.2.4.1 偏移

| 注册 | 偏移 |
|---|---|
| ÃÃÃÃ | 8h |

#### 31.4.2.4.2 职能

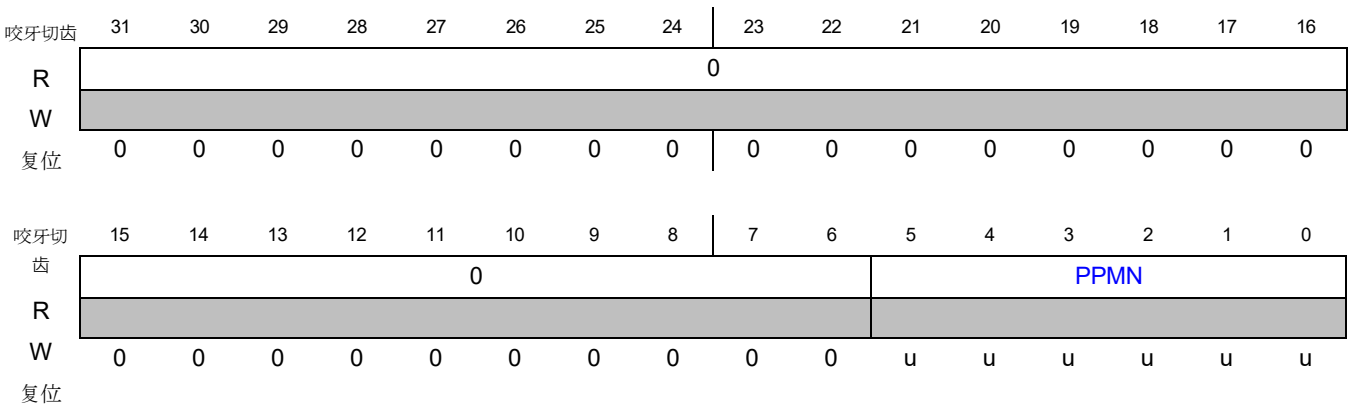
该寄存器提供了一个CPU特定的响应，表明正在进行访问的内核的物理总线主站号码。32位响应定义了处理器x的物理主控编号。

- CPxMASTER = 0x00000000

**注意事项**

从CPU的特权读取会返回相应的处理器信息。从非CPU总线主站的读取会返回所有的零。尝试的用户模式或写访问会以错误结束。

### 31.4.2.4.3 图示




### 31.4.2.4.4 领域

| 场地 | 职能 |
|---|---|
| 31-6<br>- | 保留 |
| 5-0<br>PPMN | 处理器x物理主控编号<br>这个只读字段定义了CPUx的物理总线主站号码。PPMN = 0x00 |

## 31.4.2.5 处理器X计数寄存器（CPxCOUNT）。

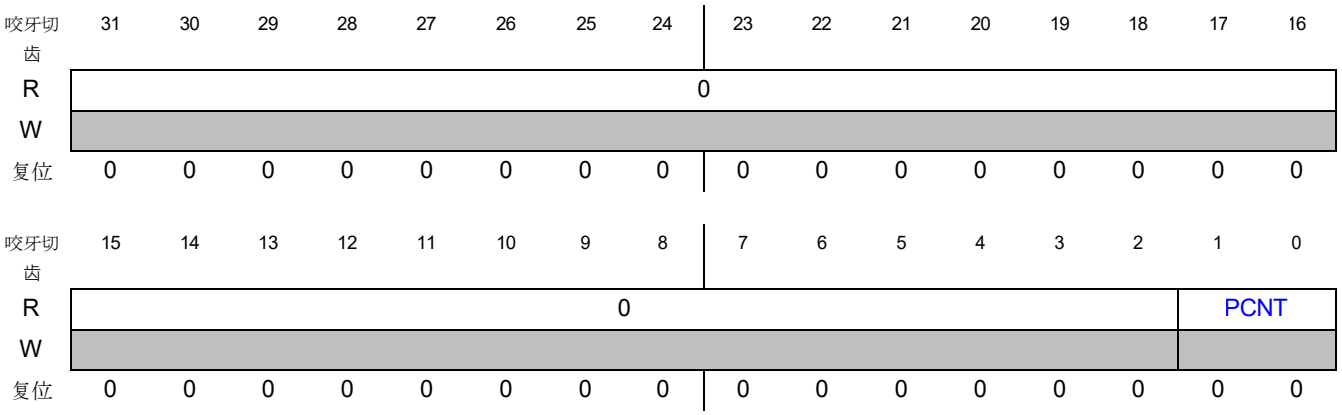
### 31.4.2.5.1 偏移

| 注册 | 偏移 |
|---|---|
| CPxCOUNT | 丘吉尔 |

#### 31.4.2.5.2 职能

该寄存器显示芯片配置中处理器内核的总数。

#### 31.4.2.5.3 图示




#### 31.4.2.5.4 领域

| 场地 | 职能 |
|---|---|
| 31-2<br>- | 保留 |
| 1-0<br>PCNT | 处理器数量<br>这个只读字段定义了芯片配置的处理器数量。PCNT = 00 (单核) |

### 31.4.2.6 处理器X配置寄存器0（CPxCFG0）。

#### 31.4.2.6.1 偏移

| 注册 | 偏移 |
|---|---|
| CPxCFG0 | 10h |

#### 31.4.2.6.2 职能

CPxCFG0寄存器提供了一个针对CPU的响应，详细说明了配置信息，在这种情况下，是关于1级缓存的信息（如果存在）。

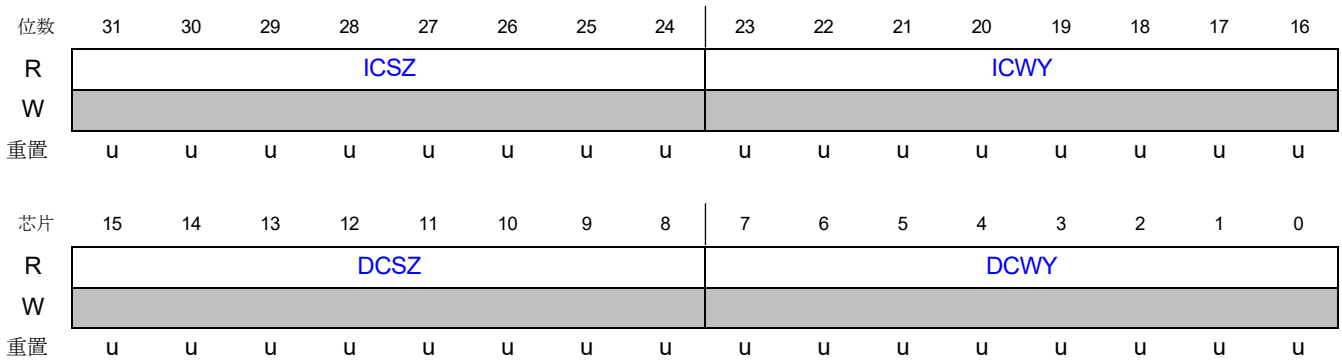
访问权限。特权只读

## 注意事项

处理器X配置寄存器0的复位值。

- 对于CPU0 - CPxCFG0 = 0x04000000
- 如果读访问不是来自CPU，那么读的值是0x00000000

### 31.4.2.6.3 图示




### 31.4.2.6.4 领域

| 场地 | 职能 |
|---|---|
| 31-24<br>ICSZ | 第1级指令缓存大小<br>这个只读字段提供了一个指令缓存大小的编码值。内存的容量表示为大小[字节]=$2^{(9+ICSZ)}$，其中ICSZ为非零；ICSZ=0表示内存不存在。<br>• 如果没有指令缓存，那么 ICSZ = 0x00<br>• 如果是1K字节的指令缓存，那么ICSZ = 0x01<br>• 如果是2K字节的指令缓存，那么ICSZ = 0x02<br>• 如果是4K字节的指令缓存，那么ICSZ = 0x03<br>• 如果是8K字节的指令缓存，那么ICSZ = 0x04<br>• 如果是16K字节的指令缓存，那么ICSZ = 0x05<br>• 如果是32K字节的指令缓存，那么ICSZ = 0x06<br>• 如果是64K字节的指令缓存，那么ICSZ = 0x07<br>• 如果是128K字节的指令缓存，那么ICSZ = 0x08 |
| 23-16<br>ICWY | 1级指令缓存方式<br>这个只读字段提供了指令缓存的缓存方式的数量。ICWY=0x00表示不存在。 |

| | |
|---|---|
| 15-8<br>DCSZ | 1级数据缓存大小<br><br>这个只读字段提供了一个数据缓存容量的编码值。内存的容量表示为大小[字节]=$2^{(9+DCSZ)}$，其中DCSZ为非零；DCSZ=0表示内存不存在。<br><br>• 如果没有数据缓存，那么DCSZ = 0x00<br>• 如果是1K字节的数据缓存，那么DCSZ = 0x01<br>• 如果是2K字节的数据缓存，那么DCSZ = 0x02<br>• 如果是4K字节的数据缓存，那么DCSZ = 0x03<br>• 如果是8K字节的数据缓存，那么DCSZ = 0x04<br>• 如果是16K字节的数据缓存，那么DCSZ = 0x05 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | • 如果是32K字节的数据缓存，那么DCSZ = 0x06<br>• 如果是64K字节的数据缓存，那么DCSZ = 0x07<br>• 如果是128K字节的数据缓存，那么DCSZ = 0x08 |
| 7-0<br>DCWY | 1级数据缓存方式<br>这个只读字段提供了数据缓存的缓存方式的数量。DCWY=0x00表示不存在。 |

## 31.4.2.7 处理器X配置寄存器1（CPxCFG1）。

### 31.4.2.7.1 偏移

| 注册 | 偏移 |
| --- | --- |
| CPxCFG1 | 14h |

### 31.4.2.7.2 职能

CPxCFG1寄存器提供了一个针对CPU的响应，详细说明了配置信息，在这种情况下，是关于2级缓存的信息（如果存在）。

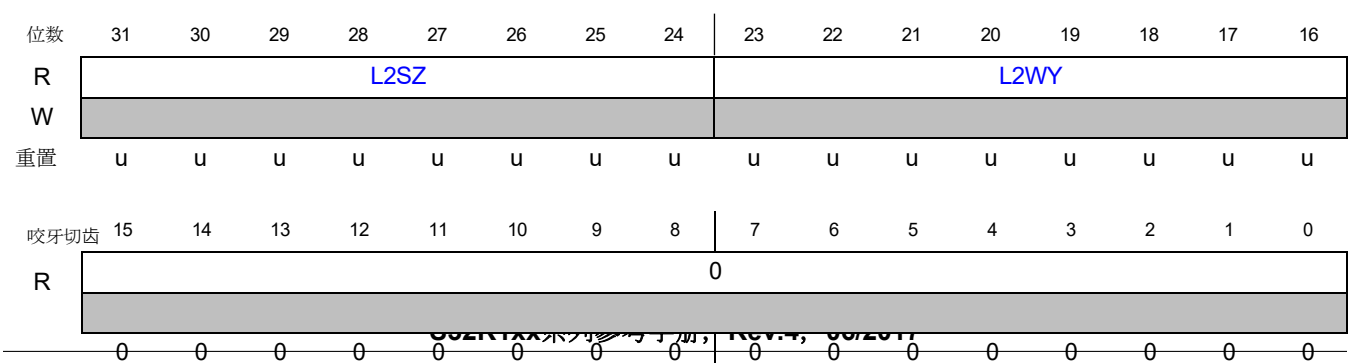
访问权限。特权只读

**注意事项**

处理器X配置寄存器1的复位值。

- 对于CPU0 - CPxCFG1 = 0x00000000
- 如果读取访问不是来自CPU，那么读取的值是0x00000000

### 31.4.2.7.3 图示



W

复位

#### 31.4.2.7.4 领域

| 场地 | 职能 |
|---|---|
| 31-24<br>L2SZ | 第2级指令缓存大小<br>这个只读字段提供了一个指令缓存容量的编码值。存储器的容量表示为大小[字节]=$2^{(9+L2SZ)}$，其中L2SZ为非零；L2SZ=0表示不存在该存储器。<br>• 如果没有二级缓存，则L2SZ = 0x00<br>• 如果是1K字节的2级缓存，那么L2SZ = 0x01<br>• 如果是2K字节的2级缓存，那么L2SZ = 0x02<br>• 如果是4K字节的2级缓存，那么L2SZ = 0x03<br>• 如果是8K字节的2级缓存，那么L2SZ = 0x04<br>• 如果是16K字节的2级缓存，那么L2SZ = 0x05<br>• 如果是32K字节的2级缓存，那么L2SZ = 0x06<br>• 如果是64K字节的2级缓存，那么L2SZ = 0x07<br>• 如果是128K字节的2级缓存，那么L2SZ = 0x08 |
| 23-16<br>L2WY | 2级指令缓存方式<br>这个只读字段提供了指令缓存的缓存方式的数量。L2WY=0x00表示不存在。 |
| 15-0<br>- | 保留 |

## 31.4.2.8 处理器X配置寄存器2（CPxCFG2）。

### 31.4.2.8.1 偏移

| 注册 | 偏移 |
|---|---|
| CPxCFG2 | 18h |

### 31.4.2.8.2 职能

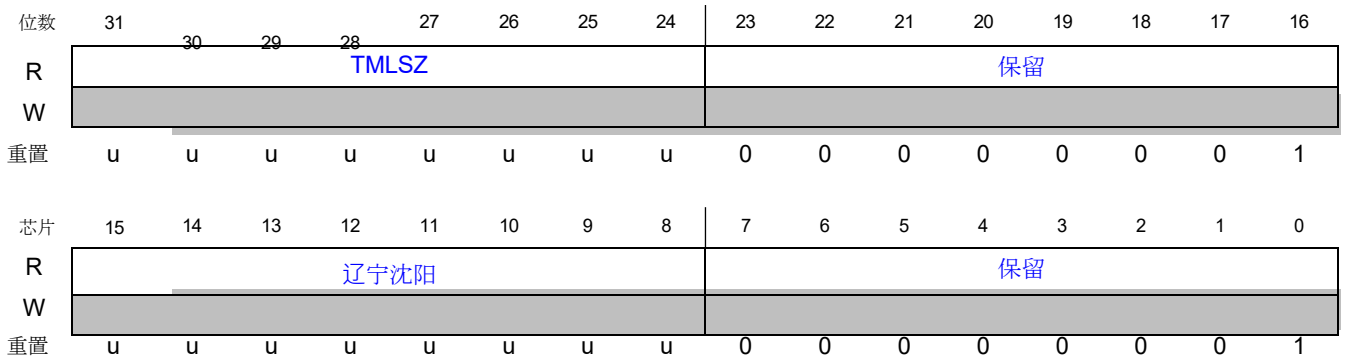
CPxCFG2寄存器提供了一个CPU特定的响应，详细说明了配置信息，在这种情况下，紧耦合的本地存储器的信息（如果存在的话）。

**注意事项**

处理器X配置寄存器2的复位值。

- 对于CPU0 - CPxCFG2 = 0x09010901
- 如果读访问不是来自CPU，那么读的值是0x00000000

### 31.4.2.8.3 图示




### 31.4.2.8.4 领域

| 场地 | 职能 |
|---|---|
| 31-24<br>TMLSZ | 紧密耦合的内存较低的尺寸<br>这个字段提供了一个紧耦合本地存储器下限大小的编码值。内存的容量表示为大小[字节]=$2^{(9+TMLSZ)}$，其中TMLSZ为非零；TMLSZ=0表示内存不存在。<br>• 如果没有TCML，那么TMLSZ = 0x00<br>• 如果是1K字节的TCML，那么TMLSZ = 0x01<br>• 如果是2K字节的TCML，那么TMLSZ = 0x02<br>• 如果是4K字节的TCML，那么TMLSZ = 0x03<br>• 如果是8K字节的TCML，那么TMLSZ = 0x04<br>• 如果是16K字节的TCML，那么TMLSZ = 0x05<br>• 如果是32K字节的TCML，那么TMLSZ = 0x06<br>• 如果是64K字节的TCML，那么TMLSZ = 0x07<br>• 如果是128K字节的TCML，那么TMLSZ = 0x08<br>• 如果是256K字节的TCML，那么TMLSZ = 0x09 |
| 23-16<br>- | 保留。 |
| 15-8<br>辽宁沈阳 | 紧密耦合的内存上限尺寸<br>这个字段提供了一个紧耦合本地存储器上层尺寸的编码值。内存的容量表示为大小[字节]=$2^{(9+TMUSZ)}$，其中TMUSZ为非零；TMUSZ=0表示内存不存在。<br>• 如果没有TCMU，那么TMUSZ = 0x00<br>• 如果是1K字节的TCMU，那么TMUSZ = 0x01<br>• 如果是2K字节的TCMU，那么TMUSZ = 0x02<br>• 如果是4K字节的TCMU，那么TMUSZ = 0x03<br>• 如果是8K字节的TCMU，那么TMUSZ = 0x04<br>• 如果是16K字节的TCMU，那么TMUSZ = 0x05<br>• 如果是32K字节的TCMU，那么TMUSZ = 0x06 |

| | |
|---|---|
| | • 如果是64K字节的TCMU，那么TMUSZ = 0x07<br>• 如果是128K字节的TCMU，那么TMUSZ = 0x08<br>• 如果是256K字节的TCMU，那么TMUSZ = 0x09 |
| 7-0<br>- | 保留。 |

## 31.4.2.9 处理器X配置寄存器3（CPxCFG3）。

### 31.4.2.9.1 偏移

| 注册 | 偏移 |
|---|---|
| CPxCFG3 | 1Ch |

### 31.4.2.9.2 职能

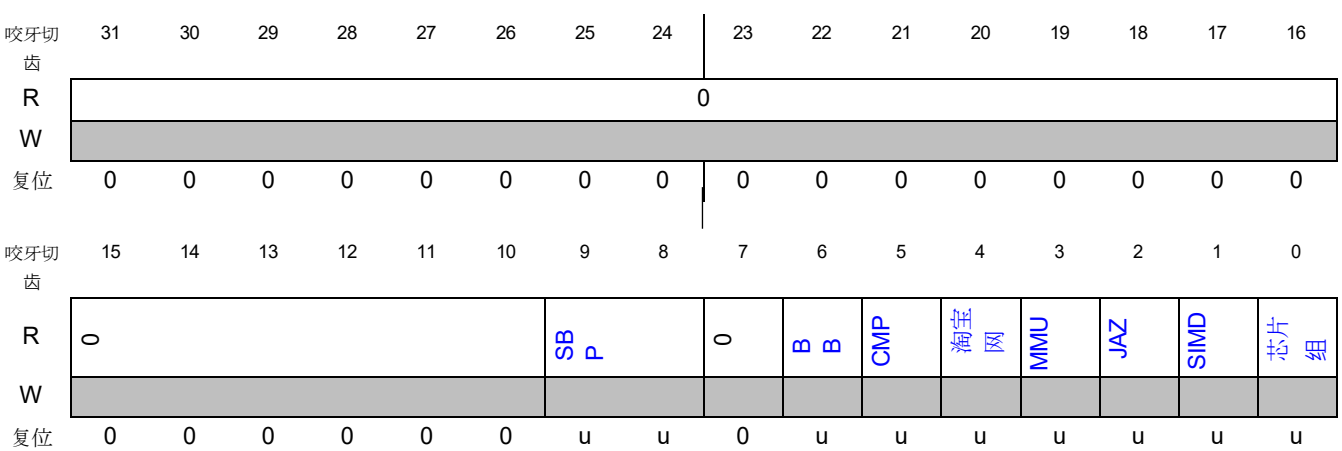
CPxCFG3寄存器提供一个CPU特定的响应，详细说明配置信息，在这种情况下，是关于处理器选项的信息。

**注意事项**

处理器X配置寄存器3的复位值。

- 对于CPU0 - CPxCFG3 = 0x00000101
- 如果读取访问不是来自CPU，那么读取的值是0x00000000

### 31.4.2.9.3 图示




### 31.4.2.9.4 领域

| 场地 | 职能 |
|---|---|

| | |
|---|---|
| 31-10<br>- | 保留 |
| 9-8 | 系统总线端口 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| SBP | 这个字段定义了这个处理器与系统总线结构的物理连接数。 |
| 7<br>- | 保留 |
| 6<br>ÃÃÃ | 位带化<br>这个字段定义了处理器是否支持 "位带"。<br>0b - 不支持位带化。1b - 支持位带化。 |
| 5<br>CMP | 核心存储器保护单元<br>这个字段表示处理器中是否包括核心内存保护硬件。<br>0b - 不包括核心内存保护。1b - 包括核心内存保护。 |
| 4<br>淘宝网 | 信任区<br>这个字段表示处理器中是否支持信任区功能。<br>0b - 不包括信任区支持。1b - 包括信任区支持。 |
| 3<br>MMU | 内存管理单元<br>内存管理单元。这个字段表示处理器中是否支持虚拟内存管理能力。<br>0b - 不包括MMU支持。1b - 包括MMU支持。 |
| 2<br>JAZ | 杰泽尔支持<br>这个字段表示处理器中是否支持Jazelle硬件。<br>0b - 不包括Jazelle支持。1b - 包括Jazelle支持。 |
| 1<br>SIMD | 支持SIMD/NEON指令<br>这个字段表示处理器中是否支持支持SIMD和/或NEON功能的指令集扩展。<br>0b - 不包括SIMD/NEON支持。1b - 包括SIMD/NEON支持。 |
| 0<br>芯片组 | 浮点单元<br>这个字段表示处理器中是否支持浮点功能的硬件支持。<br>0b - 不包括FPU支持。1b - 包括FPU支持。 |

## 31.4.2.10 处理器0类型寄存器（CP0TYPE）。

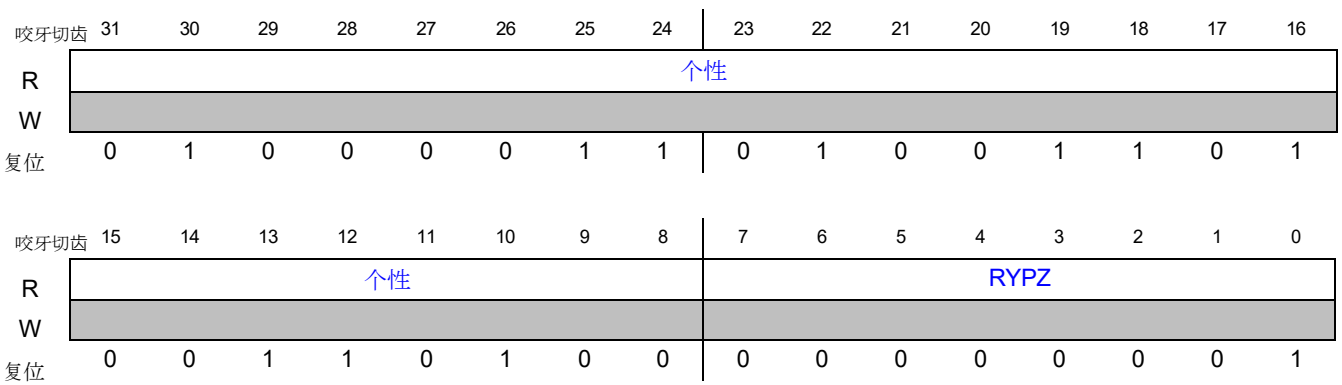
### 31.4.2.10.1 偏移

| 注册 | 偏移 |
| --- | --- |
| CP0TYPE | 20h |

### 31.4.2.10.2 职能

该寄存器提供处理器0的个性。
32位响应包括定义CPU类型的3个ASCII字符，以及定义逻辑修订号的一个字节。
逻辑修订号遵循ARM的rYpZ命名法。

### 31.4.2.10.3 图示




### 31.4.2.10.4 领域

| 场地 | 职能 |
|---|---|
| 31-8<br>个性 | 处理器0的个性<br>这个只读字段定义了CP0的处理器个性 CP0 = Cortex-M4，那么<br>PERSONALITY = 0x43_4D_34（"CM4"）。 |
| 7-0<br>RYPZ | 处理器0修订版<br>这个只读字段定义了CPU0的处理器版本：0x00对应于r0p0<br>内核的发布。<br>0x01对应的是r0p1核心版本。<br>... |

## 31.4.2.11 处理器0号寄存器（CP0NUM）。

### 31.4.2.11.1 偏移

| 注册 | 偏移 |
| --- | --- |
| CP0NUM | 24h |

### 31.4.2.11.2 职能

该寄存器提供了0号处理器的逻辑处理器编号，逻辑处理器编号始终为0。

### 31.4.2.11.3 图示

| 咬牙切齿 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 咬牙切齿 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | CPN |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

### 31.4.2.11.4 领域

| 场地 | 职能 |
|---|---|
| 31-1<br>- | 保留 |
| 0<br>CPN | 处理器0号<br>这个填零的字定义了CPU0的逻辑处理器编号 如果是单核配置，那么<br>CPN=0 |

## 31.4.2.12 处理器0主寄存器（CP0MASTER）。

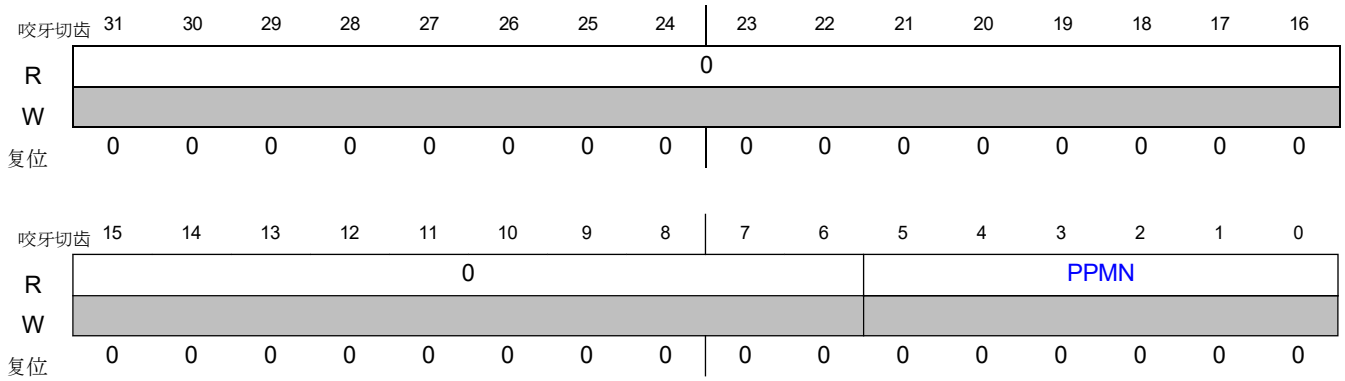
### 31.4.2.12.1 偏移

| 注册 | 偏移 |
|---|---|
| CP0MASTER | 28h |

### 31.4.2.12.2 职能

该寄存器提供了处理器0的物理总线主站编号。

### 31.4.2.12.3 图示




### 31.4.2.12.4 领域

| 场地 | 职能 |
|---|---|
| 31-6<br>- | 保留 |
| 5-0<br>PPMN | 处理器0的物理主控编号<br>这个只读字段定义了CPU0的物理总线主站编号。 |

## 31.4.2.13 处理器0计数寄存器（CP0COUNT）。

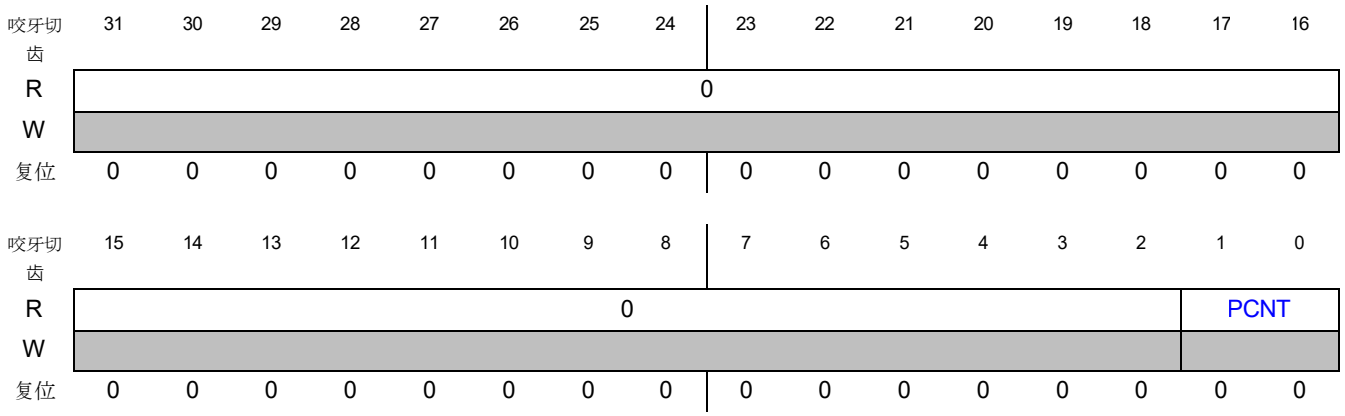
### 31.4.2.13.1 偏移

| 注册 | 偏移 |
|---|---|
| CP0COUNT | 2Ch |

### 31.4.2.13.2 职能

该寄存器表示芯片配置中处理器内核的总数。

### 31.4.2.13.3 图示




### 31.4.2.13.4 领域

| 场地 | 职能 |
|---|---|
| 31-2<br>- | 保留 |
| 1-0<br>PCNT | 处理器数量<br>这个只读字段定义了芯片配置的处理器数量。PCNT = 00 (单核) |

## 31.4.2.14 处理器0配置寄存器0（CP0CFG0）。

### 31.4.2.14.1 偏移

| 注册 | 偏移 |
|---|---|
| CP0CFG0 | 30h |

### 31.4.2.14.2 职能

CP0CFG0寄存器提供关于CPU0一级缓存的信息（如果存在）。访问权限。特权只读

**注意事项**

处理器0配置寄存器0的复位值。

- CP0CFG0 = 0x04000000

### 31.4.2.14.3 图示

| 咬牙切齿 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | ICSZ | | | | | | | | ICWY | | | | | | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 咬牙切齿 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | DCSZ | | | | | | | | DCWY | | | | | | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

### 31.4.2.14.4 领域

| 场地 | 职能 |
|---|---|
| 31-24<br>ICSZ | 第1级指令缓存大小<br>这个只读字段提供了一个指令缓存大小的编码值。内存的容量表示为大小[字节]=$2^{(9+ICSZ)}$，其中ICSZ为非零；ICSZ=0表示内存不存在。<br>• 如果没有指令缓存，那么 ICSZ = 0x00<br>• 如果是1K字节的指令缓存，那么ICSZ = 0x01<br>• 如果是2K字节的指令缓存，那么ICSZ = 0x02<br>• 如果是4K字节的指令缓存，那么ICSZ = 0x03<br>• 如果是8K字节的指令缓存，那么ICSZ = 0x04<br>• 如果是16K字节的指令缓存，那么ICSZ = 0x05<br>• 如果是32K字节的指令缓存，那么ICSZ = 0x06<br>• 如果是64K字节的指令缓存，那么ICSZ = 0x07<br>• 如果是128K字节的指令缓存，那么ICSZ = 0x08 |
| 23-16<br>ICWY | 1级指令缓存方式<br>这个只读字段提供了指令缓存的缓存方式的数量。ICWY=0x00表示不存在。 |
| 15-8<br>DCSZ | 1级数据缓存大小<br>这个只读字段提供了一个数据缓存容量的编码值。内存的容量表示为大小[字节]=$2^{(9+DCSZ)}$，其中DCSZ为非零；DCSZ=0表示内存不存在。<br>• 如果没有数据缓存，那么DCSZ = 0x00<br>• 如果是1K字节的数据缓存，那么DCSZ = 0x01<br>• 如果是2K字节的数据缓存，那么DCSZ = 0x02<br>• 如果是4K字节的数据缓存，那么DCSZ = 0x03<br>• 如果是8K字节的数据缓存，那么DCSZ = 0x04<br>• 如果是16K字节的数据缓存，那么DCSZ = 0x05 |

| | |
|---|---|
| | • 如果是32K字节的数据缓存，那么DCSZ = 0x06<br>• 如果是64K字节的数据缓存，那么DCSZ = 0x07<br>• 如果是128K字节的数据缓存，那么DCSZ = 0x08 |
| 7-0<br>DCWY | 1级数据缓存方式<br>这个只读字段提供了数据缓存的缓存方式的数量。DCWY=0x00表示不存在。 |

## 31.4.2.15 处理器0配置寄存器1（CP0CFG1）。

### 31.4.2.15.1 偏移

| 注册 | 偏移 |
|---|---|
| CP0CFG1 | 34h |

### 31.4.2.15.2 职能

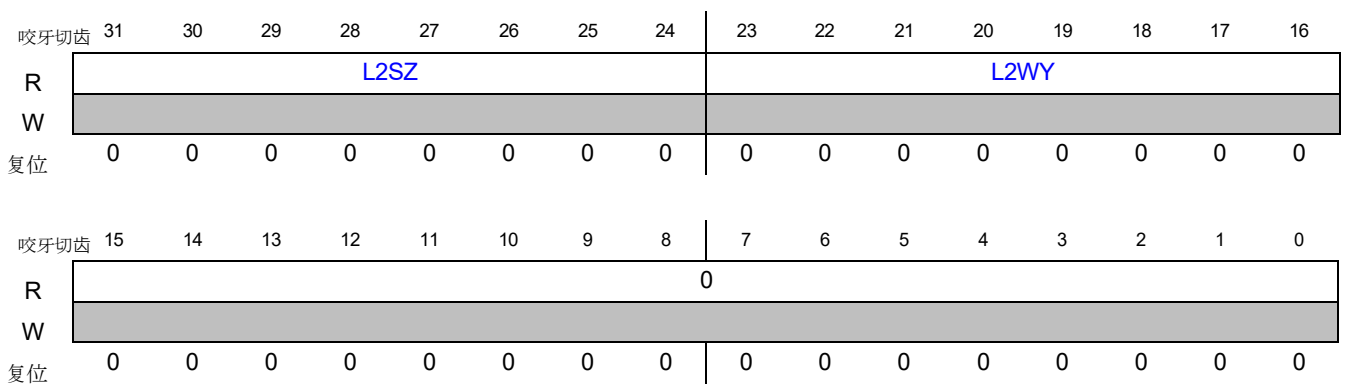
CP0CFG1寄存器提供关于CPU0二级缓存的信息（如果存在的话）。访问权限。特权只读

**注意事项**

处理器0配置寄存器1的复位值。

- CP0CFG1 = 0x00000000

### 31.4.2.15.3 图示




### 31.4.2.15.4 领域

| 场地 | 职能 |
|---|---|

| | |
|---|---|
| 31-24<br>L2SZ | 第2级指令缓存大小<br>这个只读字段提供了一个指令缓存容量的编码值。存储器的容量表示为大小[字节]=$2^{(9+L2SZ)}$，其中L2SZ为非零；L2SZ=0表示不存在该存储器。<br>• 如果没有二级缓存，则L2SZ = 0x00<br>• 如果是1K字节的2级缓存，那么L2SZ = 0x01<br>• 如果是2K字节的2级缓存，那么L2SZ = 0x02<br>• 如果是4K字节的2级缓存，那么L2SZ = 0x03 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| | • 如果是8K字节的2级缓存，那么L2SZ = 0x04<br>• 如果是16K字节的2级缓存，那么L2SZ = 0x05<br>• 如果是32K字节的2级缓存，那么L2SZ = 0x06<br>• 如果是64K字节的2级缓存，那么L2SZ = 0x07<br>• 如果是128K字节的2级缓存，那么L2SZ = 0x08 |
| 23-16<br>L2WY | 2级指令缓存方式<br>这个只读字段提供了指令缓存的缓存方式的数量。L2WY=0x00表示不存在。 |
| 15-0<br>- | 保留 |

## 31.4.2.16 处理器0配置寄存器2（CP0CFG2）。

### 31.4.2.16.1 偏移

| 注册 | 偏移 |
|---|---|
| CP0CFG2 | 38h |

### 31.4.2.16.2 职能

CP0CFG2寄存器提供关于CPU0紧耦合本地存储器的信息（如果存在的话）。

**注意事项**

处理器0配置寄存器2的复位值。

- CP0CFG2 = 0x09010901

### 31.4.2.16.3 图示

| 咬牙切齿 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | TMLSZ | | | | | | | | 保留 | | | | | | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

| 咬牙切齿 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 辽宁沈阳 | | | | | | | | 保留 | | | | | | | |
| W | | | | | | | | | | | | | | | | |
| | 0 | 0 | 0 | 0 | S32K1xx系列参考手册, | | | | Rev.4, 06/2017 | | | 0 | 0 | 0 | 0 | 1 |

复位

### 31.4.2.16.4 领域

| 场地 | 职能 |
|---|---|
| 31-24<br>TMLSZ | 紧密耦合的内存较低的尺寸<br>这个字段提供了一个紧耦合本地存储器下限大小的编码值。内存的容量表示为大小[字节]=$2^{(9+TMLSZ)}$，其中TMLSZ为非零；TMLSZ=0表示内存不存在。<br>• 如果没有TCML，那么TMLSZ = 0x00<br>• 如果是1K字节的TCML，那么TMLSZ = 0x01<br>• 如果是2K字节的TCML，那么TMLSZ = 0x02<br>• 如果是4K字节的TCML，那么TMLSZ = 0x03<br>• 如果是8K字节的TCML，那么TMLSZ = 0x04<br>• 如果是16K字节的TCML，那么TMLSZ = 0x05<br>• 如果是32K字节的TCML，那么TMLSZ = 0x06<br>• 如果是64K字节的TCML，那么TMLSZ = 0x07<br>• 如果是128K字节的TCML，那么TMLSZ = 0x08<br>• 如果是256K字节的TCML，那么TMLSZ = 0x09 |
| 23-16<br>- | 保留。 |
| 15-8<br>辽宁沈阳 | 紧密耦合的内存上限尺寸<br>这个字段提供了一个紧耦合本地存储器上层尺寸的编码值。内存的容量表示为大小[字节]=$2^{(9+TMUSZ)}$，其中TMUSZ为非零；TMUSZ=0表示内存不存在。<br>• 如果没有TCMU，那么TMUSZ = 0x00<br>• 如果是1K字节的TCMU，那么TMUSZ = 0x01<br>• 如果是2K字节的TCMU，那么TMUSZ = 0x02<br>• 如果是4K字节的TCMU，那么TMUSZ = 0x03<br>• 如果是8K字节的TCMU，那么TMUSZ = 0x04<br>• 如果是16K字节的TCMU，那么TMUSZ = 0x05<br>• 如果是32K字节的TCMU，那么TMUSZ = 0x06<br>• 如果是64K字节的TCMU，那么TMUSZ = 0x07<br>• 如果是128K字节的TCMU，那么TMUSZ = 0x08<br>• 如果是256K字节的TCMU，那么TMUSZ = 0x09 |
| 7-0<br>- | 保留。 |

## 31.4.2.17 处理器0配置寄存器3（CP0CFG3）。

### 31.4.2.17.1 偏移

| 注册 | 偏移 |
| --- | --- |
| CP0CFG3 | 3Ch |

### 31.4.2.17.2 职能

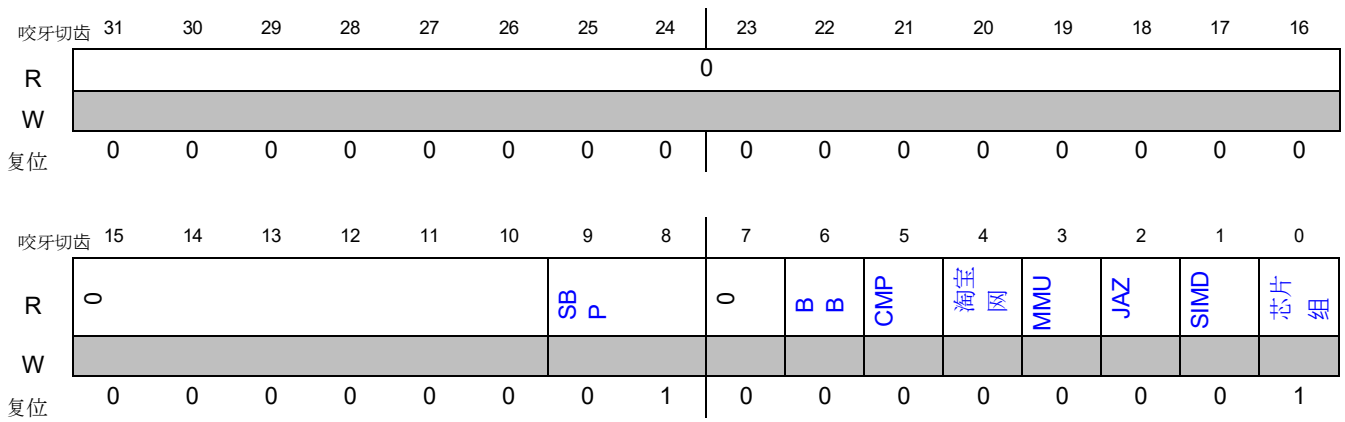
CP0CFG3寄存器提供关于处理器0选项的信息。

**注意事项**

处理器0配置寄存器3的复位值。

- CP0CFG3 = 0x00000101

### 31.4.2.17.3 图示




### 31.4.2.17.4 领域

| 场地 | 职能 |
|---|---|
| 31-10<br>- | 保留 |
| 9-8<br>SBP | 系统总线端口<br>这个字段定义了这个处理器与系统总线结构的物理连接数。 |
| 7<br>- | 保留 |
| 6<br>ÃÃÃ | 位带化<br>这个字段定义了处理器是否支持 "位带"。<br>0b - 不支持位带化。 1b - 支持位带化。 |

*表继续在下一页...*

| | |
|---|---|
| 5<br>CMP | 核心存储器保护单元<br>这个字段表示处理器中是否包括核心内存保护硬件。<br>0b - 不包括核心内存保护。 1b - 包括核心内存保护。 |
| 4<br>淘宝网 | 信任区<br>这个字段表示处理器中是否支持信任区功能。<br>0b - 不包括信任区支持。 1b - 包括信任区支持。 |

| 场地 | 职能 |
|---|---|
| 3<br>MMU | 内存管理单元<br>内存管理单元。这个字段表示处理器中是否支持虚拟内存管理能力。<br>0b - 不包括MMU支持。1b - 包括MMU支持。 |
| 2<br>JAZ | 杰泽尔支持<br>这个字段表示处理器中是否支持Jazelle硬件。<br>0b - 不包括Jazelle支持。1b - 包括Jazelle支持。 |
| 1<br>SIMD | 支持SIMD/NEON指令<br>这个字段表示处理器中是否支持支持SIMD和/或NEON功能的指令集扩展。<br>0b - 不包括SIMD/NEON支持。1b - 包括SIMD/NEON支持。 |
| 0<br>芯片组 | 浮点单元<br>这个字段表示处理器中是否支持浮点功能的硬件支持。<br>0b - 不包括FPU支持。1b - 包括FPU支持。 |

## 31.4.2.18 片上存储器描述符寄存器（OCMDR0）。

### 31.4.2.18.1 偏移

| 注册 | 偏移 |
|---|---|
| OCMDR0 | 400h |

### 31.4.2.18.2 职能

编程模型的这一部分是一个32位的通用片上存储器描述符寄存器阵列，提供关于所附存储器的静态信息，以及可配置的控制（在适当的地方）。

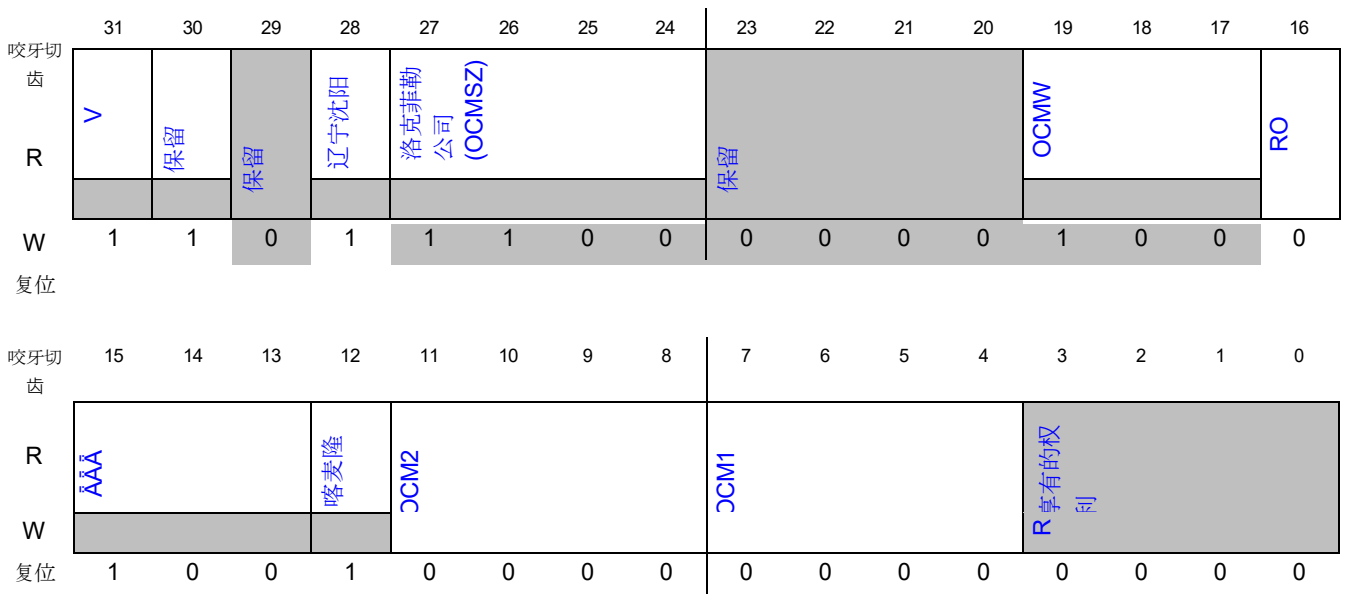
- 从处理器内核或调试器中的特权32位读数会返回适当的处理器信息。
- 从任何其他总线主站的读取都会返回所有的零。
- 从处理器内核或调试器到可写寄存器的特权写入会更新相应的字段。
- 来自其他总线主站的特权写入会被忽略。
- 试图的用户模式访问或任何大小不超过32位的访问都会以错误结束。

*表继续在下一页...*

下表描述了OCMDRn的复位值和相关的内存类型。

| OCMDRn | 重置值 | 片上存储器类型 |
|---|---|---|
| OCMDR0 | 0xDC089000 | 程序闪电 |
| OCMDR1 | 0xCA08B000 | 数据闪存 |
| OCMDR2 | 0xC304D000 | EXPAЙ KOPAЙ |
| OCMDR3 | 0x40000000 | OCMEM3不存在 |

## 31.4.2.18.3 图示




## 31.4.2.18.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>V | V<br>OCMEM 有效位。这个只读字段定义了片上存储器的有效性（存在） 0b - OCMEMn不存在。<br>1b - OCMEMn存在。 |
| 30<br>- | 保留<br>这个保留字段的值始终为1。 |
| 29<br>- | 保留 |

| | |
|---|---|
| 28<br>辽宁沈阳 | 辽宁沈阳<br>OCMEM大小 "洞"。对于没有完全填充的片上存储器，也就是在地址范围的上25%包括一个存储器 "洞"，这个位被使用。<br>0b - OCMEMn是一个2次方的容量。 |

*表继续在下一页...*

**MSCM内存图/寄存器的定义**

| 场地 | 职能 |
|---|---|
| | 1b - OCMEMn不是2次方，其容量为0.75*OCMSZ。 |
| 27-24<br>洛克菲勒公司(OCMSZ) | 洛克菲勒公司(OCMSZ)<br>OCMEM大小。这个只读字段提供了一个片上存储器大小的编码值。<br>0000b - 没有OCMEMn<br>0001b - 1KB OCMEMn<br>0010b - 2KB OCMEMn<br>0011b - 4KB OCMEMn<br>0100b - 8KB OCMEMn<br>0101b - 16KB OCMEMn<br>0110b - 32KB OCMEMn<br>0111b - 64KB OCMEMn<br>1000b - 128KB OCMEMn<br>1001b - 256KB OCMEMn<br>1010b - 512KB OCMEMn<br>1011b - 1MB OCMEMn<br>1100b - 2MB OCMEMn<br>1101b - 4MB OCMEMn<br>1110b - 8MB OCMEMn<br>1111b - 16MB OCMEMn |
| 23-20<br>- | 保留 |
| 19-17<br>OCMW | OCMW<br>OCMEM数据通路宽度。这个只读字段定义了片上存储器的宽度。000-001b - 保留<br>010b - OCMEMn 32位宽 011b<br>- OCMEMn 64位宽 100b -<br>OCMEMn 128位宽 101b -<br>OCMEMn 256位宽 110-111b -<br>保留 |
| 16<br>RO | RO<br>只读。该寄存器位提供了一种机制，用于 "锁定 "OCMDRn[11:0]所定义的配置状态。一旦断定，对OCMDRn[11:0]寄存器的尝试性写入将被忽略，直到下一次复位清除该标志。<br>0b - 允许对OCMDRn[11:0]进行写入 1b - 忽略<br>对OCMDRn[11:0]的写入。 |
| 15-13<br>ÃÃÃ | ÃÃÃ<br>OCMEM类型。这个字段定义了片上存储器的类型。000b - 保留<br>001b - 保 留<br>010b - 保 留<br>011b - 保留<br>100b - OCMEMn是一个程序闪存。<br>101b - OCMEMn是一个数据闪存。<br>110b - OCMEMn是一个EEE。<br>111b - 保留 |
| 12<br>喀麦隆 | 喀麦隆 |

*表继续在下一页...*

| | |
|---|---|
| | OCMEM内存保护单元。这个字段是为这个设备保留的。 |
| 11-8<br>OCM2 | OCMEM控制字段2 |

<table>
<tr><th>场地</th><th>职能</th></tr>
<tr><td></td><td>在向与闪存相关的OCMDRn寄存器写入时，闪存控制器需要处于空闲状态。在活动状态下改变控制器的配置会导致不希望看到的结果。<br>CF2 = 控制字段2 - 用于闪存高速缓存<br>• CF2[3] - 禁用闪存高速缓存<br>• CF2[2] - 禁用指令缓存<br>• CF2[1] - 禁用闪存数据缓存<br>• CF2[0] - 清除闪存缓存</td></tr>
<tr><td>7-4<br>OCM1</td><td>OCMEM控制字段1<br>在向与闪存相关的OCMDRn寄存器写入时，闪存控制器需要处于空闲状态。在活动状态下改变控制器的配置会导致不希望看到的结果。<br>OCMDRn[4]或OCMDRn[5]位控制是否启动预取（或推测性访问）以响应指令获取或数据引用，见FMC推测性读取部分。值0表示启用，值1表示禁用。
<table>
<tr><th>闪存推测<br>(OCMDRn第5位)</th><th>数据预取<br>(OCMDRn第4位)</th><th>结果</th></tr>
<tr><td>失能</td><td>失能</td><td rowspan="2">所有投机功能被禁用，投机缓冲区被清空</td></tr>
<tr><td>失能</td><td>已启用</td></tr>
<tr><td>已启用</td><td>失能</td><td>启用指令猜测，禁用数据猜测</td></tr>
<tr><td>已启用</td><td>已启用</td><td>对指令和数据启用的推测</td></tr>
</table></td></tr>
<tr><td>3-0<br>-</td><td>保留</td></tr>
</table>

## 31.4.2.19 片上存储器描述符寄存器（OCMDR1）。

### 31.4.2.19.1 偏移

| 注册 | 偏移 |
|---|---|
| OCMDR1 | 404h |

### 31.4.2.19.2 职能

编程模型的这一部分是一个32位的通用片上存储器描述符寄存器阵列，提供关于所附存储器的静态信息，以及可配置的控制（在适当的地方）。

表继续在下一页...

- 从处理器内核或调试器中的特权32位读数会返回适当的处理器信息。

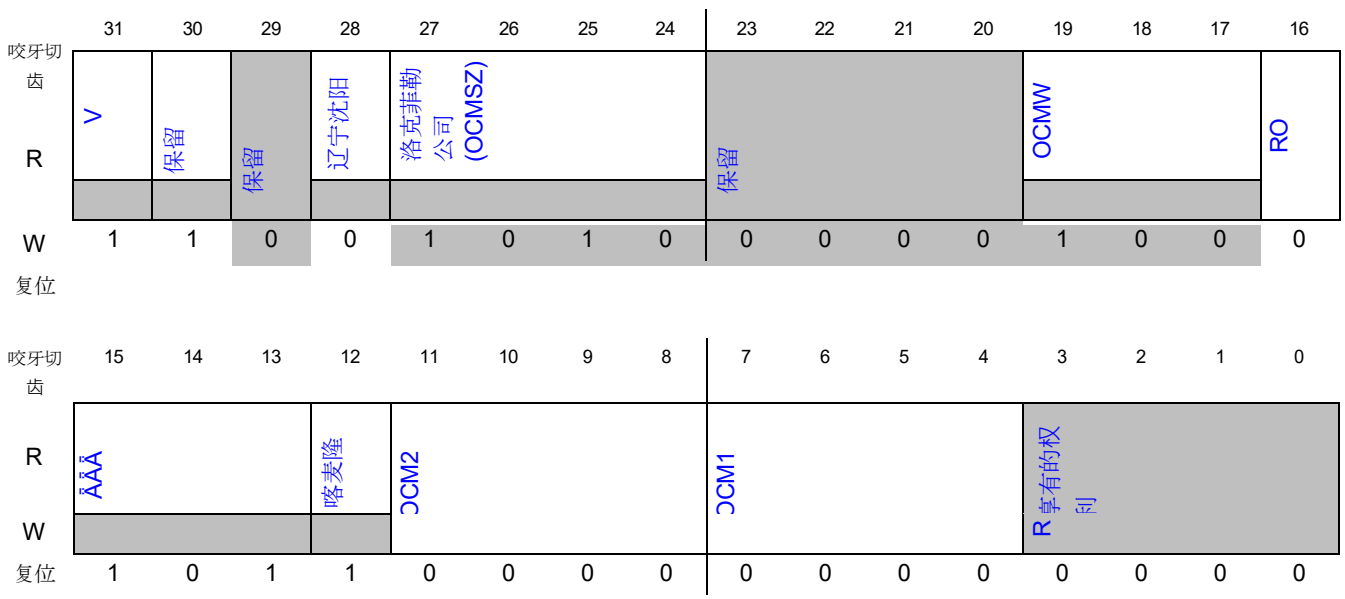
- 从任何其他总线主站的读取都会返回所有的零。

- 从处理器内核或调试器到可写寄存器的特权写入会更新相应的字段。
- 来自其他总线主站的特权写入会被忽略。
- 试图的用户模式访问或任何大小不超过32位的访问都会以错误结束。

下表描述了OCMDRn的复位值和相关的内存类型。

| OCMDRn | 重置值 | 片上存储器类型 |
|---|---|---|
| OCMDR0 | 0xDC089000 | 程序闪电 |
| OCMDR1 | 0xCA08B000 | 数据闪存 |
| OCMDR2 | 0xC304D000 | EXPAЙ KOPAЙ |
| OCMDR3 | 0x40000000 | OCMEM3不存在 |

### 31.4.2.19.3 图示




### 31.4.2.19.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>V | V<br>OCMEM 有效位。这个只读字段定义了片上存储器的有效性（存在） 0b - OCMEMn不存在。<br>1b - OCMEMn存在。 |
| 30<br>- | 保留<br>这个保留字段的值始终为1。 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| 29<br>- | 保留 |
| 28<br>辽宁沈阳 | 辽宁沈阳<br>OCMEM大小 "洞"。对于没有完全填充的片上存储器，也就是在地址范围的上25%包括一个存储器 "洞"，这个位被使用。<br>0b - OCMEMn是一个2次方的容量。<br>1b - OCMEMn不是2次方，其容量为0.75*OCMSZ。 |
| 27-24<br>洛克菲勒公司(OCMSZ) | 洛克菲勒公司(OCMSZ)<br>OCMEM大小。这个只读字段提供了一个片上存储器大小的编码值。<br>0000b - 没有OCMEMn<br>0001b - 1KB OCMEMn<br>0010b - 2KB OCMEMn<br>0011b - 4KB OCMEMn<br>0100b - 8KB OCMEMn<br>0101b - 16KB OCMEMn<br>0110b - 32KB OCMEMn<br>0111b - 64KB OCMEMn<br>1000b - 128KB OCMEMn<br>1001b - 256KB OCMEMn<br>1010b - 512KB OCMEMn<br>1011b - 1MB OCMEMn<br>1100b - 2MB OCMEMn<br>1101b - 4MB OCMEMn<br>1110b - 8MB OCMEMn<br>1111b - 16MB OCMEMn |
| 23-20<br>- | 保留 |
| 19-17<br>OCMW | OCMW<br>OCMEM数据通路宽度。这个只读字段定义了片上存储器的宽度。000-001b - 保留<br>010b - OCMEMn 32位宽 011b<br>- OCMEMn 64位宽 100b -<br>OCMEMn 128位宽 101b -<br>OCMEMn 256位宽 110-111b -<br>保留 |
| 16<br>RO | RO<br>只读。该寄存器位提供了一种机制，用于 "锁定 "OCMDRn[11:0]所定义的配置状态。一旦断定，对OCMDRn[11:0]寄存器的尝试性写入将被忽略，直到下一次复位清除该标志。<br>0b - 允许对OCMDRn[11:0]进行写入 1b - 忽略<br>对OCMDRn[11:0]的写入。 |

| | |
|---|---|
| 15-13<br>ÃÃÃ | ÃÃÃ<br>OCMEM类型。这个字段定义了片上存储器的类型。000b - 保留<br>001b - 保 留<br>010b - 保 留<br>011b - 保留<br>100b - OCMEMn是一个程序闪存。<br>101b - OCMEMn是一个数据闪存。 |

*表继续在下一页...*

*表继续在下一页...*

<table>
<tr><th>场地</th><th>职能</th></tr>
<tr><td></td><td>110b - OCMEMn是一个EEE。<br>111b - 保留</td></tr>
<tr><td>12<br>喀麦隆</td><td>喀麦隆<br>OCMEM内存保护单元。这个字段是为这个设备保留的。</td></tr>
<tr><td>11-8<br>OCM2</td><td>OCMEM控制字段2<br>在向与闪存相关的OCMDRn寄存器写入时，闪存控制器需要处于空闲状态。在活动状态下改变控制器的配置会导致不希望看到的结果。<br>CF2 = 控制字段2 - 用于闪存高速缓存<br>• CF2[3] - 禁用闪存高速缓存<br>• CF2[2] - 禁用指令缓存<br>• CF2[1] - 禁用闪存数据缓存<br>• CF2[0] - 清除闪存缓存</td></tr>
<tr><td>7-4<br>OCM1</td><td>OCMEM控制字段1<br>在向与闪存相关的OCMDRn寄存器写入时，闪存控制器需要处于空闲状态。在活动状态下改变控制器的配置会导致不希望看到的结果。<br>OCMDRn[4]或OCMDRn[5]位控制是否启动预取（或推测性访问）以响应指令获取或数据引用，见FMC推测性读取部分。值0表示启用，值1表示禁用。
<table>
<tr><th>闪光推测<br>(OCMDRn第5位)</th><th>数据预取<br>(OCMDRn第4位)</th><th>结果</th></tr>
<tr><td>失能</td><td>失能</td><td rowspan="2">所有投机功能被禁用，投机缓冲区被清空</td></tr>
<tr><td>失能</td><td>已启用</td></tr>
<tr><td>已启用</td><td>失能</td><td>启用指令猜测，禁用数据猜测</td></tr>
<tr><td>已启用</td><td>已启用</td><td>对指令和数据启用的推测</td></tr>
</table>
</td></tr>
<tr><td>3-0<br>-</td><td>保留</td></tr>
</table>

## 31.4.2.20 片上存储器描述符寄存器（OCMDR2）。

### 31.4.2.20.1 偏移

| 注册 | 偏移 |
|---|---|
| OCMDR2 | 408h |

### 31.4.2.20.2 职能

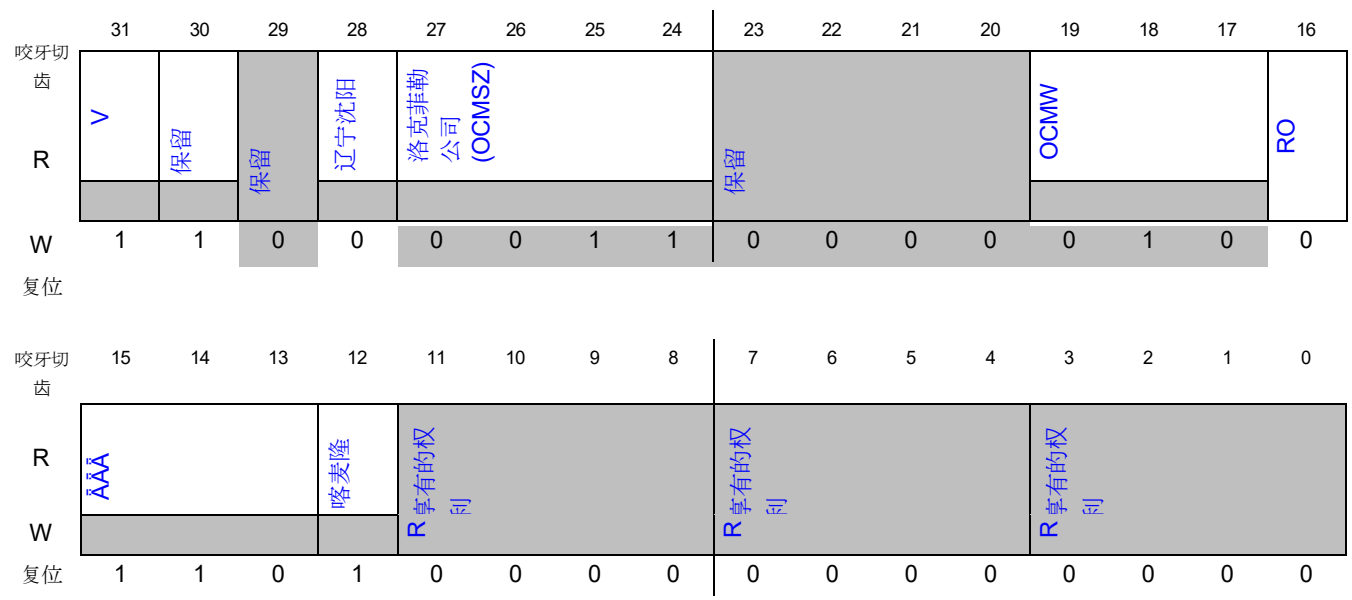
编程模型的这一部分是一个32位的通用片上存储器描述符寄存器阵列，提供关于所附存储器的静态信息，以及可配置的控制（在适当的地方）。

- 从处理器内核或调试器中的特权32位读数会返回适当的处理器信息。
- 从任何其他总线主站读出的数据都会返回所有的零。
- 从处理器内核或调试器到可写寄存器的特权写入会更新相应的字段。
- 来自其他总线主站的特权写入会被忽略。
- 试图的用户模式访问或任何大小不超过32位的访问都会以错误结束。

下表描述了OCMDRn的复位值和相关的内存类型。

| OCMDRn | 重置值 | 片上存储器类型 |
|---|---|---|
| OCMDR0 | 0xDC089000 | 程序闪电 |
| OCMDR1 | 0xCA08B000 | 数据闪存 |
| OCMDR2 | 0xC304D000 | EXPAЙ KOPAЙ |
| OCMDR3 | 0x40000000 | OCMEM3不存在 |

### 31.4.2.20.3 图示



## 31.4.2.20.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>V | V<br>OCMEM 有效位。这个只读字段定义了片上存储器的有效性（存在） 0b - OCMEMn不存在。<br>1b - OCMEMn存在。 |
| 30<br>- | 保留<br>这个保留字段的值始终为1。 |
| 29<br>- | 保留 |
| 28<br>辽宁沈阳 | 辽宁沈阳<br>OCMEM大小 "洞"。对于没有完全填充的片上存储器，也就是在地址范围的上25%包括一个存储器 "洞"，这个位被使用。<br>0b - OCMEMn是一个2次方的容量。<br>1b - OCMEMn不是2次方，其容量为0.75*OCMSZ。 |
| 27-24<br>洛克菲勒公司(OCMSZ) | 洛克菲勒公司(OCMSZ)<br>OCMEM大小。这个只读字段提供了一个片上存储器大小的编码值。<br>0000b - 没有OCMEMn<br>0001b - 1KB OCMEMn<br>0010b - 2KB OCMEMn<br>0011b - 4KB OCMEMn<br>0100b - 8KB OCMEMn<br>0101b - 16KB OCMEMn<br>0110b - 32KB OCMEMn<br>0111b - 64KB OCMEMn<br>1000b - 128KB OCMEMn<br>1001b - 256KB OCMEMn<br>1010b - 512KB OCMEMn<br>1011b - 1MB OCMEMn<br>1100b - 2MB OCMEMn<br>1101b - 4MB OCMEMn<br>1110b - 8MB OCMEMn<br>1111b - 16MB OCMEMn |
| 23-20<br>- | 保留 |
| 19-17<br>OCMW | OCMW<br>OCMEM数据通路宽度。这个只读字段定义了片上存储器的宽度。000-001b - 保留<br>010b - OCMEMn 32位宽 011b<br>- OCMEMn 64位宽 100b -<br>OCMEMn 128位宽 101b -<br>OCMEMn 256位宽 110-111b -<br>保留 |
| 16<br>RO | RO<br>只读。该寄存器位提供了一种机制，用于 "锁定 "OCMDRn[11:0]所定义的配置状态。一旦断定，对OCMDRn[11:0]寄存器的尝试性写入将被忽略，直到下一次复位清除该标志。 |

表继续在下一页...

| 场地 | 职能 |
| --- | --- |
| | 0b - 允许对OCMDRn[11:0]进行写入 1b - 忽略<br>对OCMDRn[11:0]的写入。 |
| 15-13<br>ÃÃÃ | ÃÃÃ<br>OCMEM类型。这个字段定义了片上存储器的类型。000b - 保留<br>001b - 保 留<br>010b - 保 留<br>011b - 保留<br>100b - OCMEMn是一个程序闪存。<br>101b - OCMEMn是一个数据闪存。<br>110b - OCMEMn是一个EEE。<br>111b - 保留 |
| 12<br>喀麦隆 | 喀麦隆<br>OCMEM内存保护单元。这个字段是为这个设备保留的。 |
| 11-8<br>- | 保留 |
| 7-4<br>- | 保留 |
| 3-0<br>- | 保留 |

# 第32章
# 闪存控制器（FMC）

## 32.1 特定芯片的FMC信息

本节总结了该模块在芯片中的配置情况。

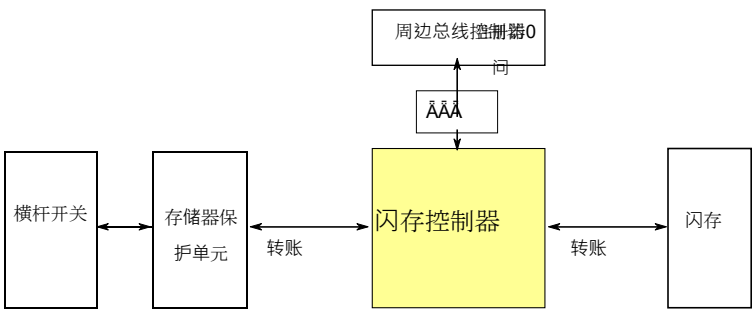



**图32-1.闪存控制器配置**

该设备不支持Wait/VLPW模式。关于可用的电源模式的详细信息，请参见模块在可用的低功率模式下的操作。

**表32-1.相关信息的参考链接**

| 主题 | 相关模块 | 参考资料 |
|---|---|---|
| 系统内存图 | | 见附件S32K1xx_memory_map.xlsx |
| 计时 | 系统时钟发生器 | 时钟分布 |
| 转移 | 闪存 | 闪存模块（FTFC） |
| 转移 | 系统MPU | 内存保护单元（MPU） |
| 转移 | 横杆开关 | 横梁开关(AXBS-Lite) |
| 登记访问 | 周边桥梁 | 外围桥（AIPS-Lite） |
| 登记控制 | ÃÃÃ | MSCM寄存器的OCMC1字段 片上存储器描述符寄存器（OCMDR0）。 |

### 32.1.1 FMC大师

关于主控端口的分配，请参见Crossbar Switch主控分配。

## 32.2 简介

闪存控制器（FMC）是一个内存加速单元，提供。

- 是设备和双组非易失性存储器之间的接口。银行0由程序闪存组成，银行1由FlexNVM组成。
- 缓冲器，可以加速闪存和FlexNVM数据传输。

### 32.2.1 概述

闪存控制器管理着设备和双组闪存之间的接口。程序闪存被称为Bank 0，数据闪存被称为Bank 1。FMC接收详细说明存储器配置的状态信息，并使用这些信息来确保正确的接口。下表显示了支持的读/写操作。

| 闪存类型 | 阅读 | 撰写 |
|---|---|---|
| 对闪存进行编程 | 8位、16位和32位读数 | -1 |
| 作为数据闪存使用的FlexNVM | 8位、16位和32位读数 | -1 |
| FlexNVM和FlexRAM被用作模拟的EEPROM | 8位、16位和32位读数 | 8位、16位和32位写法 |

1.对编程闪存或作为数据闪存的FlexNVM的写操作会导致总线错误。

此外，对于银行0和银行1，FMC提供了单独的机制来加速器件和闪存之间的接口。一个128（Bank1，数据闪存为64）位的投机缓冲器可以预取下一个128（Bank1，数据闪存为64）位的闪存位置，而一个单次进入的128（Bank1，数据闪存为64）位缓冲器可以存储以前访问过的闪存或FlexNVM数据，以获得快速访问时间。

### 32.2.2 特点

FMC的功能包括。

- 设备与双组闪存和FlexMemory之间的接口。
  - 8位、16位和32位的读操作，对闪存和作为数据闪存的FlexNVM进行编程。

  - 对作为仿真EEPROM的FlexNVM和FlexRAM进行8位、16位和32位读写操作。
  - 对于0号库（程序闪存）。存储器返回128位，因此对存储器中连续的32位空间的读取访问会返回第二、第三和第四个读取数据，没有等待状态。对于库1（数据闪存）。存储器返回64位，因此对存储器中连续的32位空间的读访问返回第二个读数据，没有等待状态。
- 对于银行0和银行1：加速从程序闪存和FlexMemory到设备的数据传输。
  - 128（Bank1为64，数据闪存）位预取投机缓冲器，可控制每个主站和银行的指令/数据访问
  - 每个银行的单一入口缓冲器

## 32.3 操作模式

FMC只在总线主站访问闪存或FlexMemory时工作。

在设备电源模式方面，FMC只在运行模式下工作，包括VLPR模式。

对于任何不能访问闪存或FlexMemory的设备电源模式，FMC被禁用。

## 32.4 外部信号描述

FMC没有外部信号。

## 32.5 职能描述

FMC是一个闪存加速单元，具有灵活的缓冲区供用户配置。每当预取投机缓冲区或单次进入缓冲区发生命中时，所要求的数据就会在一个系统时钟内传输。

### 32.5.1 默认配置

在系统复位时，FMC被配置成为闪存或FlexMemory的传输提供相当程度的缓冲。

- 对于银行0和银行1。

- 对数据和指令的预取支持已经启用。
- 单一入口的缓冲区被启用。

## 32.5.2 投机性读物

FMC有一个单一的缓冲器，如果有一个空闲周期，它可以提前读到闪存中的下一个字。使用MSCM_OCMDR0[5]和MSCM_OCMDR1[5]（值为0意味着"启用预取（或推测性访问）"），对每个库的指令和/或数据访问的推测性预取是可编程的。因为许多代码访问是连续的，在大多数情况下，使用推测性预取缓冲器可以提高性能。

当推测性读取被启用时，FMC在读取完成后立即请求下一个顺序地址。通过立即请求下一个字，推测性读取可以帮助减少甚至消除访问顺序代码和/或数据时的等待状态。

例如，考虑以下情况。

- 假设一个系统的内核与闪存的时钟比为4:1，并启用了推测性读取。
- 内核以背对背的方式请求4个（对于数据闪存，Bank 1）或8个（对于程序闪存，Bank 0）连续的长字，这意味着除了等待闪存数据返回的停顿之外，没有任何内核周期的延迟。
- 没有一个数据已经存储在缓存或投机缓冲区中。

在这种情况下，访问四个（对于数据闪存，库1）或八个（对于程序闪存，库0）长字的事件顺序如下。

1. 第一个长字的读取需要4到7个核心时钟。
2. 由于闪存的128位数据总线，第二个长字的读取只需要1个核心时钟，因为数据已经在FMC内可用。出于同样的原因，第三个和第四个长字的读取也只需要一个核心时钟。
3. 每64位的数据闪存（bank 1），访问第三个长字需要3个核心时钟周期。闪存的读取本身需要4个时钟，但第一个时钟与第二个长字的读取重叠了。
4. 每128位的程序闪存（Bank 0），访问第五个长字需要1个核心时钟周期。闪存的读取本身需要4个时钟，但是访问在第一次读取后立即开始。因此，这个访问的3个时钟与内核的第二、第三和第四个长字的读取重叠。
5. 对于数据闪存（Bank

1）的每64位，读取第四个长字，就像第二个长字一样，由于64位闪存的数据总线，只需要1个时钟。

6. 每128位的程序闪存（Bank 0），读取第六、第七和第八个长字只需要1个时钟，因为数据已经在FMC内可用。

## 32.6 初始化和应用信息

FMC不需要用户初始化。闪存加速功能在默认情况下是启用的。

# 第三十三章
# 闪存模块（FTFC）

## 33.1 芯片特有的FTFC信息

本节总结了该模块在芯片中的配置情况。

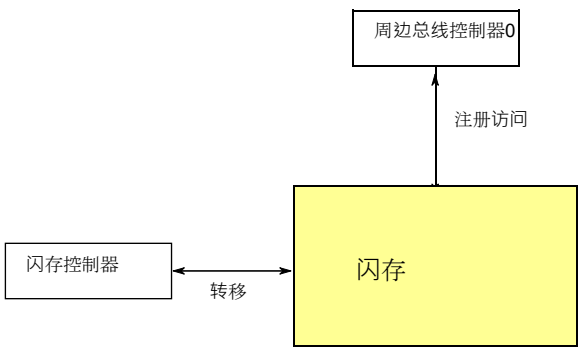



**图33-1.闪存配置**

该设备不支持等待模式。关于电源模式的详细信息，请参见模块在可用的低功耗模式下的操作。

**表33-1.相关信息的参考链接**

| 主题 | 相关模块 | 参考资料 |
|---|---|---|
| 系统内存图 | | 见附件S32K1xx_memory_map.xlsx |
| 计时 | 系统时钟发生器 | 时钟分布 |
| 转移 | 闪存控制器 | 闪存控制器（FMC） |
| 注册访问 | 周边桥梁 | 外围桥（AIPS-Lite） |

### 33.1.1 闪存类型

该芯片包含这些类型的闪存。

- 程序闪存：可执行程序代码的非易失性闪存
- FlexMemory，它包含了以下内存类型。
  - FlexNVM：非易失性闪存，可以
    - 执行程序代码
    - 存储数据
    - 备份模拟的EEPROM数据
  - FlexRAM。具备以下特点的RAM
    - 可作为SRAM或作为高端保险仿真EEPROM存储使用
    - 加速闪存编程

**注意事项**

PCC_FTFC[CGC]在安全启动期间和FlexRAM作为系统RAM工作时必须为1。

## 33.1.2 闪存的尺寸

芯片上的闪存类型的尺寸是。

| 薯片 | 程序闪电 | 弹性虚拟机（FlexNVM | 弹性RAM |
|---|---|---|---|
| S32K116 | 128 KB | 32 KB | 2 KB |
| S32K118 | 256 KB | 32 KB | 2 KB |
| S32K142 | 256 KB | 64 KB | 4 KB |
| S32K144 | 512 KB | 64 KB | 4 KB |
| S32K146 | 1 M | 64 KB | 4 KB |
| S32K148 | 高达2米 | 高达512KB1 | 4 KB |

1.可用的程序闪存被FlexNVM的大小所减少。当程序闪存被完全使用时，FlexNVM就不能使用。

## 33.1.3 闪存地图

如下图所示，各种类型的闪存和闪存的寄存器有不同的基址。每个基地址都在所附文件S32K1xx_memory_map.xlsx中注明。

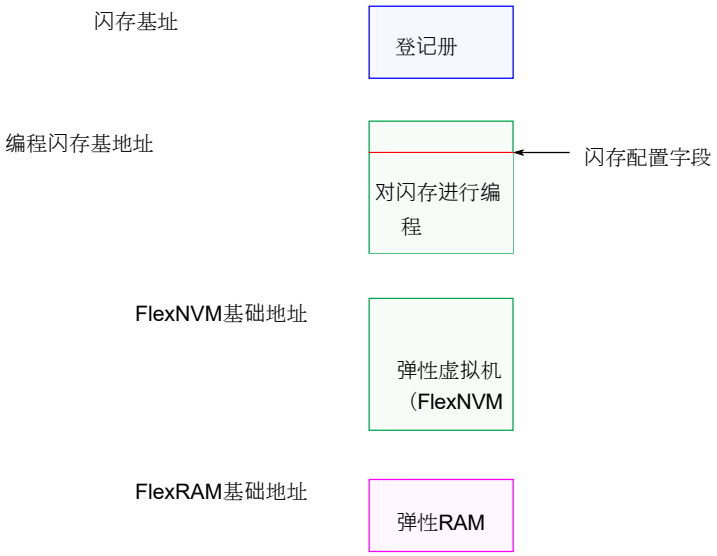



**图33-2.闪存地图**

## 33.1.4 闪存的安全性

见闪存安全。

## 33.1.5 闪存编程的电源模式限制

在芯片工作时，不应该对闪存进行编程或擦除。

- VLPR模式
- HSRUN模式

当芯片处于这些模式时，任何类型的FTFC命令，包括CSE命令（用于CSEc部件），都不可用。

## 33.1.6 闪存模式

FTFC模块有两种工作模式。NVM正常和NVM特殊。

- 在这个芯片上，FTFC总是被配置在NVM正常模式下。
- 芯片没有配置为NVM特殊模式的FTFC的工作条件。

### 33.1.7 擦除闪存中的所有内容

擦除所有区块的命令可以由软件通过一系列外围总线写到FTFC寄存器来启动。

此外，根据这个顺序，可以从外部的SWJ-DP调试端口擦除整个闪存。

1. 设置MDM-AP控制[0]。
2. MDM-AP状态[0]设置，表示已接受大规模擦除命令。
3. MDM-AP状态[0]在大规模擦除完成后被清除。

### 33.1.8 通过FTFC_FOPT寄存器定制MCU操作

Flash选项寄存器（FOPT）允许你在启动时自定义MCU的操作。关于FOPT值的含义和如何编程替代配置选项的细节，请参见FOPT启动选项。

### 33.1.9 在PFLASH读分区上同时操作

当从一个特定的PFLASH读取分区执行时，FTFC命令（除了平行启动）不能在该PFLASH读取分区运行。下面是每个设备中的PFLASH读取分区的数量。

**表33-2.PFLASH分区**

| 芯片 | PFLASH读取分区的数量 |
|---|---|
| S32K116 | 1 (128 KB) |
| S32K118 | 1 (256 KB) |
| S32K142 | 1 (256 KB) |
| S32K144 | 1 (512 KB) |
| S32K146 | 2个(每个512KB) |
| S32K148 | 3个(每个512KB) |

## 33.2 简介

FTFC模块包括以下可访问的内存区域。

- 用于矢量空间和代码存储的程序闪存

- 用于数据存储和额外代码存储的FlexNVM
- 用于高端耐力数据存储或传统RAM的FlexRAM

闪存是单电源应用的理想选择，允许在现场进行擦除和重新编程操作，不需要任何外部高压电源。

FTFC模块包括一个内存控制器，它执行命令来修改闪存内容。被擦除的位读作"1"，被编程的位读作
"0"。编程操作是单向的；它只能将位从'1'状态（擦除）移动到'0'状态（编程）。只有擦除操作才能将比特从'0'恢复到'1'；比特不能从'0'编程到'1'。

**注意事项**

闪存位置在被编程之前必须处于擦除状态。不允许在闪存位置内进行比特的累积编程（在没有擦除的情况下进行背对背的编程操作）。不允许将现有的0重新编程为0，因为这样做会对器件造成过大压力。

闪存的标准运输条件是在禁用安全的情况下进行擦除。由于擦除（'1'）状态和/或编程（'0'）状态的退化，随着时间的推移可能会出现数据丢失。因此，建议在工厂编程前立即重新擦除每个闪存块或扇区，以确保实现完整的数据保留能力。

配置的具体信息

FTFC模块有多种配置。每个配置由一个或多个读取分区组成。一个给定的读取分区在同一时间只能被一个任务占用。拥有多个读取分区可以进行并发操作，如边写边读（RWW）。必须特别考虑到有FlexNVM的分区，因为FlexMEM功能是由几个方面组成的，包括数据闪存、模拟EEPROM备份、CSEc功能（需要模拟EEPROM被启用）。因此，一个单一的FlexMEM读取分区可以被要求读取，编程，擦除，仿真EEPROM更新，或CSEc加密操作。但一次只能进行其中一项。

对于除2MB
FTFC_2048K_256K6_256K2_4K（用于S32K148）以外的所有现有配置，包括一个单独的FlexMEM/
DFlash区域的读取分区（见下文更多细节）。FTFC_2048K_256K6_256K2_4K宏有四个读分区，其中最后一个分区包括448kB PFlash和64kB
DFlash/FlexMEM。这最后一个

分区是一个共享的单读分区，需要考虑的是，在64kB的DFlash/FlexNVM区域被访问的同时，代码可能不会从448kB的PFlash区域执行，因为任何原因。

S32K116/FTFC_128K_128K_32K_2K模块包括两个NVM读取分区，以及一个2kB的FlexRAM。一个128kB的PFlash读取分区（非交错式1128kB），和一个32kB的FlexNVM读取分区（非交错式132kB）。

S32K118/FTFC_256K_256K_32K_2K模块由两个NVM读取分区和一个2kB FlexRAM组成。一个256kB的PFlash读取分区（非交错式1x256kB），和一个32kB的FlexNVM读取分区（非交错式1x32kB）。

S32K142/FTFC_256K_256K_64K_4K模块由两个NVM读取分区和一个4kB FlexRAM组成。一个256kB的PFlash读取分区（非交错式1x256kB），和一个64kB的FlexNVM读取分区（非交错式1x64kB）。

S32K144/FTFC_512K_512K_64K_4K模块由两个NVM读取分区和一个4kB FlexRAM组成。一个512kB的PFlash读取分区（交错的2x256kB），一个64kB的FlexNVM读取分区（非交错的1x64kB）。

S32K146FTFC_1024K_256K4_64K_4K模块包括三个NVM读取分区，以及4kB的FlexRAM上。两个PFlash读取分区（交错的2x256kB），和一个64kB的FlexNVM读取分区（非交错的1x64kB）。

S32K148/FTFC_2048K_256K6_256K2_4K模块包括四个NVM读取分区，和一个4kB的FlexRAM。三个PFlash读取分区（交错2x256kB）和一个共享的P/DFlash（FlexNVM）读取分区（交错2x256kB）。这个共享的P/DFlash读取分区包括448kB的PFlash（交错2x224kB）和64kB的FlexNVM（交错2x32kB）区域。

交错块的数据宽度为128位，而非交错块的数据宽度为64位。

### 33.2.1 特点

FTFC模块包括以下特点。

**注意事项**

关于你的设备上可用的闪存的确切数量，请参见设备的芯片配置细节。

#### 33.2.1.1 编程flash存储器功能

- 对于非交错配置（即256kB及以下），扇区大小为2KB。对于交错配置（512kB及以上）为4KB
- 程序闪存保护方案可防止意外编程或擦除存储的数据
- 自动、内置、带验证的编程和擦除算法
- 分段编程，加快批量编程时间
- 在对数据闪存块或FlexRAM中的数据进行编程或擦除时，可以对程序闪存块进行读取访问。

#### 33.2.1.2 FlexNVM内存的特点

当FlexNVM对数据闪存进行分区时。

- 对于非交错配置（即2MB配置以外的所有配置），扇区大小为2KB。交错配置（2MB）为4KB
- 保护方案可防止意外编程或删除存储的数据
- 自动的、内置的程序和擦除算法，带有验证功能
- 分段编程，加快批量编程时间
- 在编程或擦除程序闪存块中的数据时，可以对数据闪存块进行读取访问。

#### 33.2.1.3 FlexRAM的特点

- 可作为传统RAM或高端保险仿真EEPROM存储的存储器
- 配置多达4KB的FlexRAM，用于模拟EEPROM或传统RAM操作
- 当被配置为仿真EEPROM时。
  - 保护方案可防止为仿真EEPROM写入的数据被意外编程或擦除
  - 内置硬件仿真方案，实现仿真EEPROM记录维护功能的自动化
  - 可编程的仿真EEPROM数据集大小和FlexNVM分区代码促进了仿真EEPROM内存耐久性的权衡
  - 支持每次1、2或4个字节的FlexRAM对齐写操作
  - 在对程序或数据闪存中的数据进行编程或擦除时，可以对FlexRAM进行读取访问
- 当配置为传统RAM时。
  - 在对程序或数据闪存中的数据进行编程或擦除时，可以对FlexRAM进行读和写访问

#### 33.2.1.4 加密服务引擎（CSEc）模块的特点

FTFC模块包括对SHE（安全硬件扩展）规范的功能支持。

- 安全的加密密钥存储（从3到17个用户密钥不等）。
- AES-128加密和解密
- AES-128 CMAC（基于密码的消息认证码）计算和认证
- ECB（电子密码本）模式 - 加密和解密
- CBC（密码块链）模式 - 加密和解密
- 真实和伪随机数生成
- 宫口-普雷内尔压缩函数
- 安全启动模式（用户可配置）
  - 顺序启动模式
  - 并行启动模式
  - 严格的顺序启动模式（一旦设置就无法改变）。

#### 33.2.1.5 其他FTFC模块的特点

- 用于闪存编程和擦除操作的内部高压电源发生器
- 闪存命令完成后可选择生成中断
- 支持MCU安全机制，防止未经授权访问闪存内容
- ECC逻辑纠正单比特故障并检测每个NVM闪存短语中的多比特故障

### 33.2.2 框图

FTFC模块的方框图如下图所示。

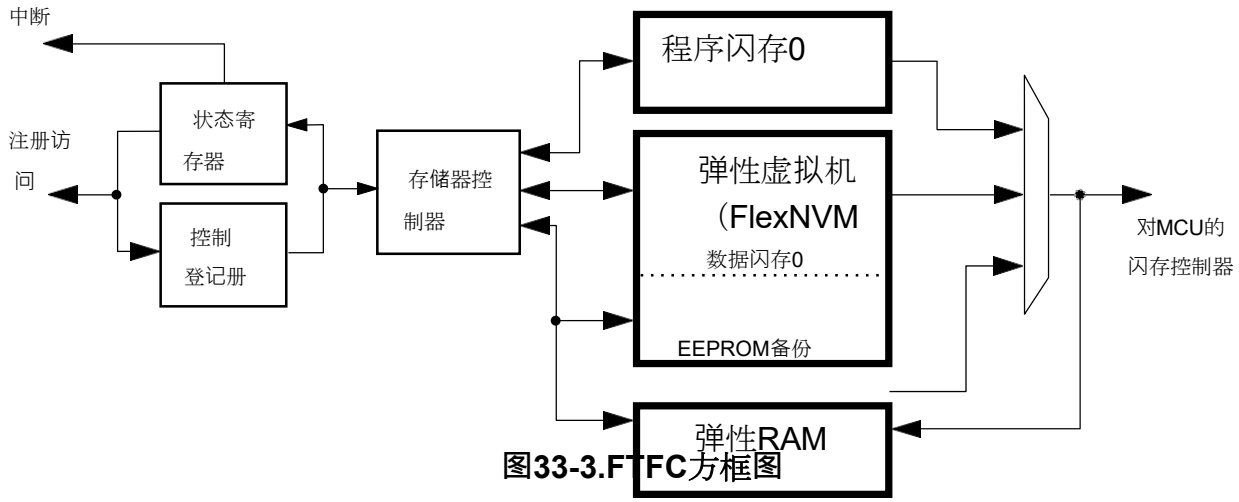
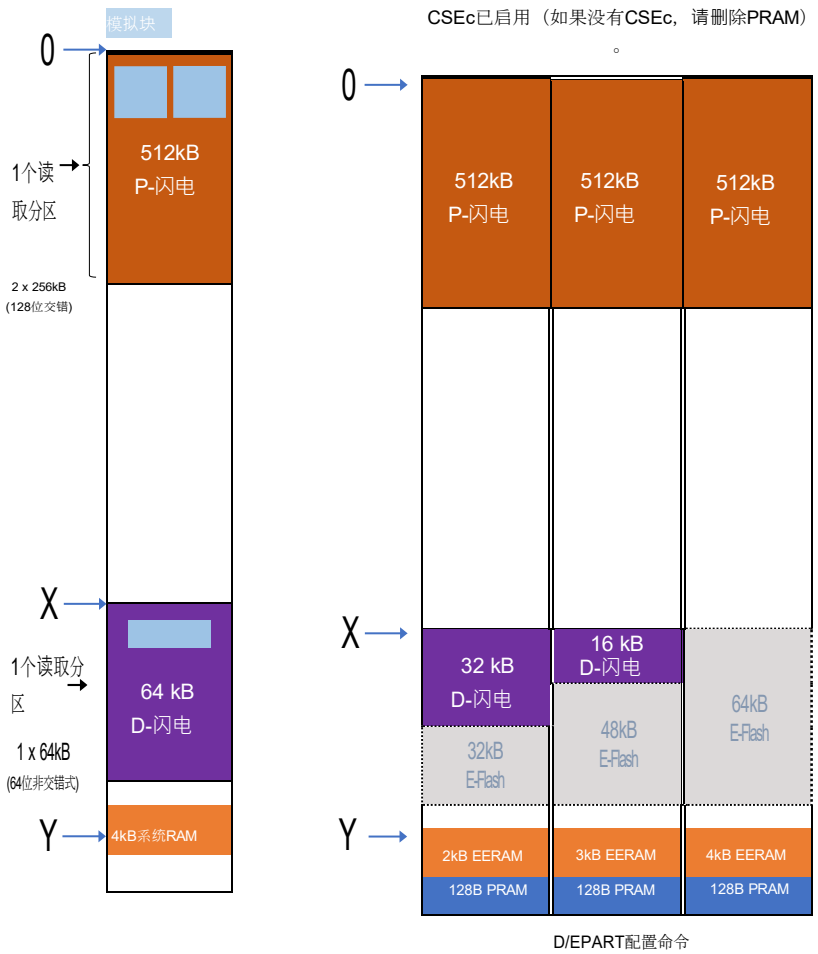



**图33-3.FTFC方框图**

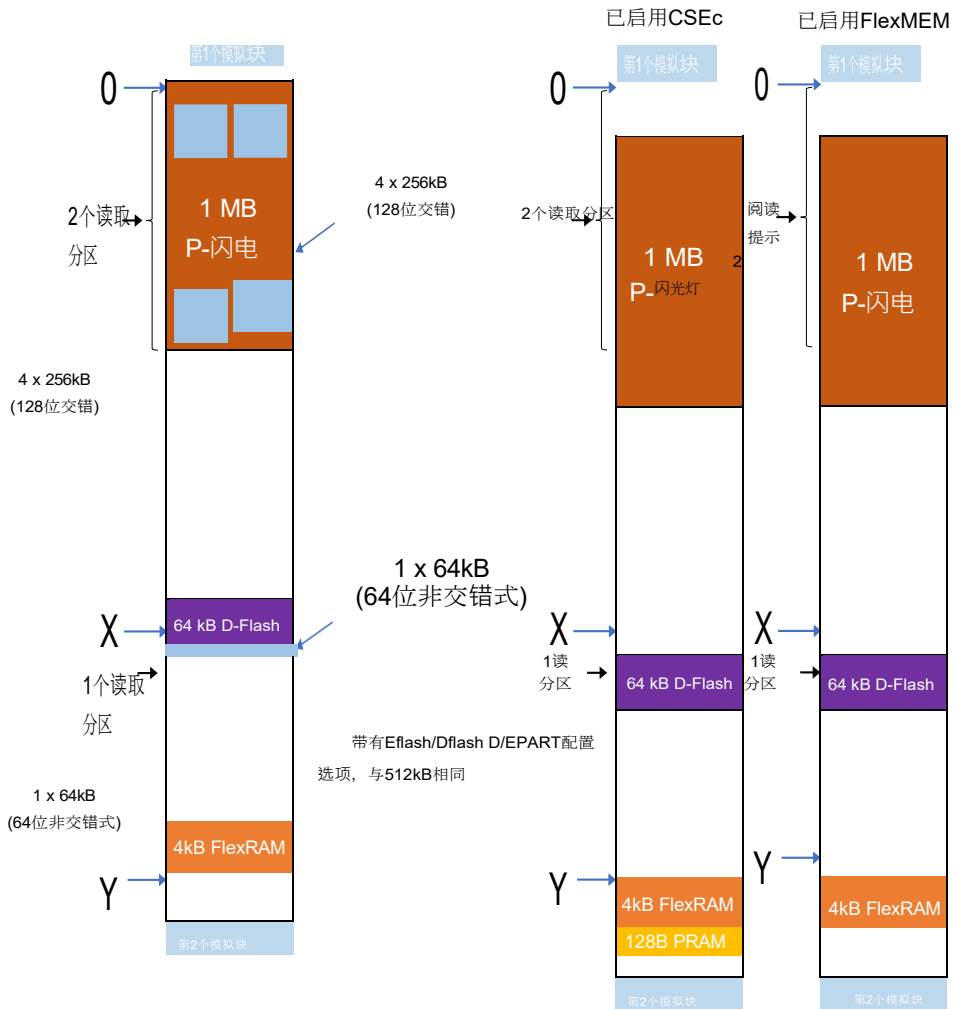
S32K144/512kB宏的方框图，显示各种FlexNVM DEPART配置。S32K142/256kB是一样的，只是用一个256kB的PFlash块（单一的1 x256kB非交错块，数据路径64位）减少了512kB的PFlash块（交错2x256kB）。由于FlexNVM被配置为仿真EEPROM，相关的EFLash从内存图中消失，如图所示。

P/D-闪存映射(S32K144/512kB)




S32K146/1MB宏的方框图，显示了各种FlexNVM DEPART配置 FlexMEM DEPART规则在1MB宏上与上述512kB宏相同。这里显示了PFlash中额外的读取分区。

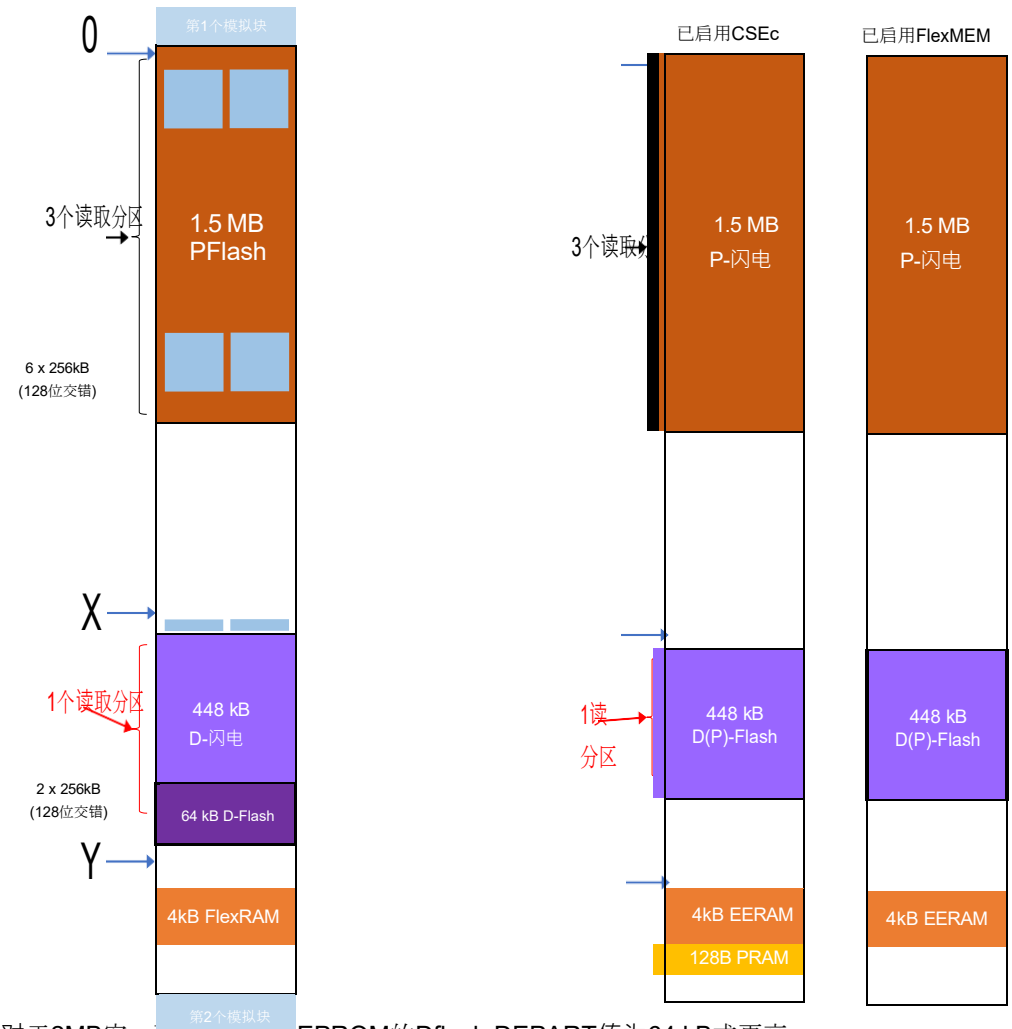
## P/D-闪存映射(S32K146/1M)




对于1MB的宏，DEPART规则遵循上面所示的512kB配置

S32K148/2MB宏的方框图，显示了各种FlexNVM DEPART配置
由于FlexNVM在2MB宏上配置为仿真EEPROM，所以与其他配置有点不同。选择是全部64kB或没有。

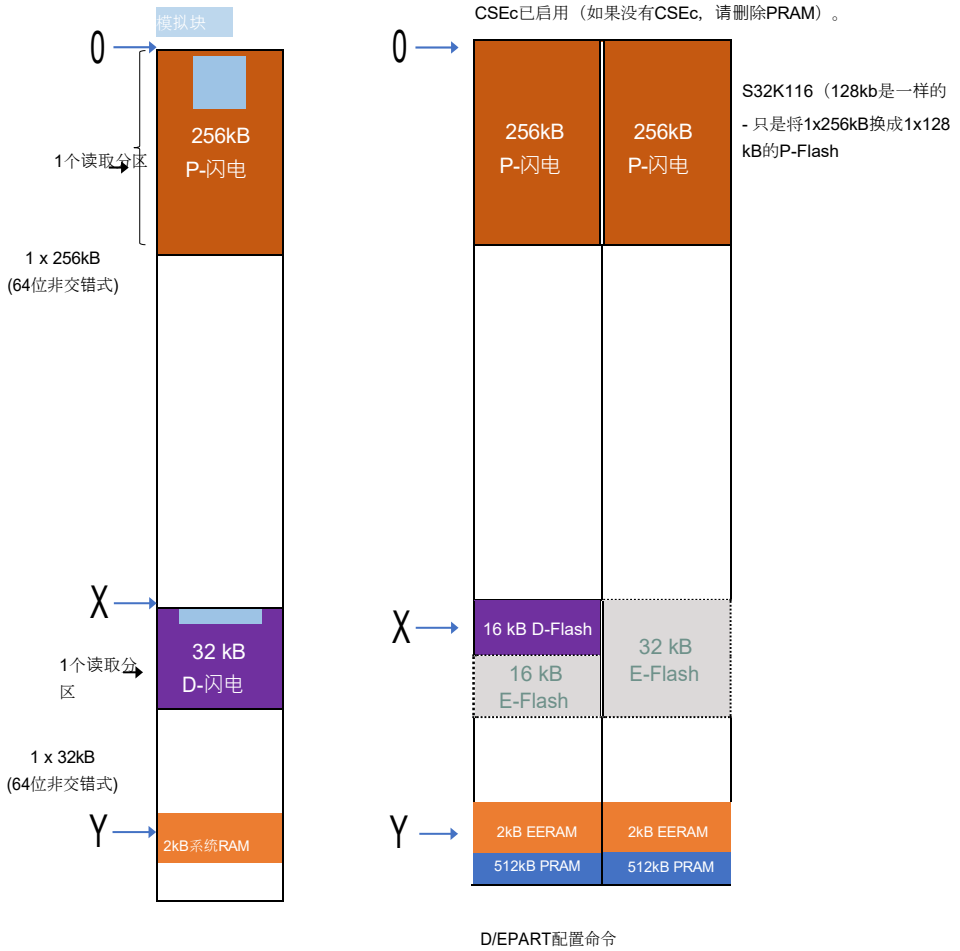
P/D-闪存映射(S32K148/2M)




对于2MB宏，配置为仿真EEPROM的Dflash DEPART值为64 kB或更高

S32K116/118 128kB/256kB宏的方框图，显示了各种FlexNVM DEPART配置由于FlexNVM被配置为仿真EEPROM，相关的EFLASH从内存图中消失，如图所示，与其他宏（不包括2MB）相同，但DFLASH更小。S32K116/128kB宏与118相同，只是PFlash被减少到128kB。

P/D-闪存映射（S32K118/256kB）




P/D-闪存映射(S32K116/128kB)是相同的，只是将256kB与128kB交换。

## 33.2.3 词汇表

**命令写入序列** - 一系列MCU对Flash
FCCOB寄存器组的写入，启动并控制内置于FTFC模块的Flash算法的执行。

**加密服务引擎--CSEc--**
为满足SHE规范而建立的各种加密、解密和认证算法的功能集。这（CSEc）是CSE模块的一个 "紧凑 "版本

**CSE_PRAM** -
专门用于CSEc参数空间的RAM。用于所有CSEc/SHE命令请求和相关的数据输入/输

**数据闪存** -
从FlexNVM块中划分出来，数据闪存为用户数据、启动代码和附加代码存储提供非易失性存储。

**数据闪存扇区** - 数据闪存扇区是数据闪存中可被擦除的最小部分。

**双短语** - 128位数据的对齐双短语，其字节地址[0:3] == 0000。

**EEPROM** -
使用一个内置的文件系统，FTFC模块通过有效地提供一个高端保险、字节可写（编程和擦除）的NVM来模拟EEPROM的特性。

**EEPROM备份数据头** -
仿真EEPROM备份数据头包括一个在仿真EEPROM备份数据存储器中发现的64位字段，它包含了仿真EEPROM归档系统用来确定特定仿真EEPROM备份闪存扇区状态的信息。

**EEPROM备份数据记录** -
仿真EEPROM备份数据记录包括一个7位的状态字段，一个13位的地址字段，以及一个在仿真EEPROM备份数据存储器中发现的32位数据字段，该数据字段被仿真EEPROM存档系统使用。如果状态字段表明记录是有效的，那么数据字段就会在FlexRAM中的一个由地址字段决定的位置上进行镜像。

**EEPROM备份数据存储器** -
从FlexNVM块分区，仿真EEPROM备份数据存储器为仿真EEPROM备案系统提供非易失性存储，代表写入需要最高耐久性的FlexRAM的数据。

**EEPROM备份数据扇区** -
仿真EEPROM备份数据扇区包含一个仿真EEPROM头和最多255条仿真EEPROM备份数据记录，这些记录被仿真EEPROM归档系统使用。

**耐久性（循环）**--闪存位置可以被擦除和重新编程的次数。

**FCCOB (Flash Common Command Object)** -
一组闪存寄存器，用于将命令、地址、数据和任何相关参数传递给FTFC模块的内存控制器。

**闪存块** - FTFC模块内的一个宏，提供非易失性存储器存储。

**FlexMemory** - FTFC配置，支持数据闪存、模拟EEPROM和FlexRAM。

**FlexNVM块** -
FlexNVM块可以被配置为数据闪存、仿真EEPROM备份闪存或两者的组合。

**FlexRAM** -
FlexRAM指的是FTFC模块专用的RAM，它可以被配置为存储仿真EEPROM数据或作为传统RAM。当配置为仿真EEPROM时，对FlexRAM的有效写入会产生一个新的仿真EEPROM备份数据记录，存储在仿真EEPROM备份闪存中。

**FTFC模块** -
所有闪存块加上一个闪存管理单元，提供高级控制和MCU总线的接口。

**HSRUN** -
一种MCU电源模式，能够临时高速访问FTFC模块中的内存资源。当MCU处于HSRUN模式时，用户无法访问Flash命令集（没有FCCOB或CSEc命令，也没有模拟EEPROM功能，也没有FlexRAM访问--只有对NVM阵列的系统读取）。

**IFR** - 在每个闪存块中发现的非易失性信息寄存器，与主存储器阵列分开。

**交错/非交错**--
交错（D或P闪存）是指内部数据路径为128位宽，地址在两个64位宽的数据路径模块之间交替进行。而非交错式（D或P闪存）指的是内部数据路径为64位宽。交错式DFlash（2MB宏）与非交错式宏有不同的DEPART选项。

**长字** - 32位的数据，有一个对齐的长字，字节地址[1:0] = 00。

**NVM** -
非易失性存储器。一种存储器技术，在断电期间保持存储的数据。闪存阵列是一种采用NOR型闪存技术的非易失性存储器。

**NVM正常模式** -
一种NVM模式，提供用户对FTFC资源的基本访问。CPU或其他总线主站使用对FTFC模块中FCCOB寄存器组的写入来启动闪存编程和擦除操作（或其他闪存命令）。

**NVM特殊模式** -
一种NVM模式，使外部、片外访问FTFC模块中的内存资源。当MCU处于安全状态时，一个减少的闪存命令集是可用的。有关何时使用该模式的信息，请参见芯片配置详情。

**短语** - 64位的数据，有一个对齐的短语，字节地址[2:0] = 000。

**程序闪存** - 程序闪存为向量和代码存储提供非易失性存储。

**程序闪存扇区** - 程序闪存（连续地址）中可被擦除的最小部分。

**四元组** - 256比特的数据，其对齐的四元组的字节地址[4:0]=00000。

**RWW-**
边读边写。在另一个内存资源中进行指令操作时，同时从一个内存资源中读取的能力。

部分程序缓冲区 -
FlexRAM的下四分之一，分配用于存储大量数据，以便通过编程部分命令进行编程。

安全 -
传递给FTFC模块的MCU状态，如该设备的芯片配置细节中所述。在安全状态下，读取和改变NVM内容受到限制。

字 - 16位的数据，有一个对齐的字，字节地址[0]=0。

## 33.3 外部信号描述

FTFC模块不包含连接片外的信号。

## 33.4 存储器地图和寄存器

本节介绍了FTFC模块的内存图和寄存器。从FTFC模块中未实现的内存空间读取的数据是未定义的。对FTFC模块中未实现的或保留的内存空间（寄存器）的写入是被忽略的。

### 33.4.1 闪存配置字段描述

程序闪存包含一个16字节的闪存配置字段，存储默认保护设置（复位时加载）和安全信息，允许MCU限制对FTFC模块的访问。

| 闪存配置字段偏移地址 | 尺寸（字节） | 领域描述 |
|---|---|---|
| 0x0_0400 - 0x0_0407 | 8 | 后门比较密钥。请参考验证后门访问密钥命令和使用后门密钥访问解除对MCU的安全保护。 |
| 0x0_0408 - 0x0_040B | 4 | 编程闪存保护字节。请参考编程闪存保护寄存器（FPROT0-3）的描述。 |
| 0x0_040F | 1 | 数据闪存保护字节。请参考数据闪存保护寄存器（FDPROT）的描述。 |
| 0x0_040E | 1 | EEPROM保护字节。请参考EEPROM保护寄存器（FEPROT）的描述。 |

*表继续在下一页...*

| 闪存配置字段偏移地址 | 尺寸（字节） | 领域描述 |
|---|---|---|
| 0x0_040D | 1 | 闪存非易失性选项字节。请参考闪存选项寄存器（FOPT）的描述。 |
| 0x0_040C | 1 | 闪存安全字节。请参考闪存安全寄存器（FSEC）的描述。 |

## 33.4.2 程序闪光0 IFR地图

程序闪存0
IFR是一个1K字节的非易失性信息存储器，可以自由读取，但用户没有擦除和有限的编程能力（参见Read Once, and Program Once命令中的Read Once命令，和 Program Once命令）。下表总结了程序闪存0 IFR的内容，并在随后的段落中进一步描述。

程序闪存0的IFR位于程序闪存0内存块内。

| 地址范围 | 尺寸（字节） | 领域描述 |
|---|---|---|
| 0x000 - 0x3BF | 960 | 保留 |
| 0x3C0 - 0x3FF | 64 | 计划 一次 性领域 |

### 33.4.2.1 程序一次字段

程序闪存0 IFR中的Program
Once字段提供了64字节的用户数据存储，与程序闪存0主阵列分开。用户只能对"一次编程
"字段进行一次编程，因为用户没有可用的程序闪存IFR擦除机制。一次编程字段可以被读取任何次数。使用Read Once和Program Once命令（见Read Once命令和Program Once命令），可以用8个字节的记录来访问程序闪存0 IFR的这一部分。

## 33.4.3 数据闪存0 IFR地图

数据闪存0
IFR是一个1K字节的非易失性信息存储器，可以被读取和擦除，但用户在数据闪存0

IFR中的编程能力有限（参见《编程分区》中的《编程分区命令》，《擦除所有块》中的《擦除所有块》命令）。 数据闪存0
IFR的内容在下表中进行了总结，并在随后的段落中进一步描述。

数据闪存0的IFR位于数据闪存0内存块内。

| 地址范围 | 尺寸（字节） | 领域描述 |
|---|---|---|
| 0x00 - 0x3FB, 0x3FE - 0x3FF | 1022 | 保留 |
| 0x3FD | 1 | EEPROM数据集大小 |
| 0x3FC | 1 | FlexNVM 分区代码 |

#### 33.4.3.1 EEPROM数据集大小

数据闪存0
IFR中的仿真EEPROM数据集大小字节提供的信息决定了每个可用的仿真EEPROM子系统中使用的FlexRAM的数量，并指示FlexRAM是否在闪存复位序列中加载了有效的仿真EEPROM数据。要对EEERST、EEESIZE值进行编程，请参见Program Partition命令中描述的Program Partition命令。

请参考SoC的SIM（系统集成模块），了解该信息的读取值（EEERST，EEESIZE）。

#### 33.4.3.2 FlexNVM分区代码

数据闪存0
IFR中的FlexNVM分区代码字节提供了一个代码，指定如何在数据闪存和支持仿真EEPROM功能的仿真EEPROM备份存储器之间分割FlexNVM块。要对DEPART值进行编程，请参见Program Partition命令中描述的Program Partition命令。

请参考SoC SIM（系统集成模块），了解该信息的读取值（DEPART）。

### 33.4.4 登记簿描述

FTFC模块包含一组内存映射的控制和状态寄存器。

**注意事项**

当一个命令正在运行时（FSTAT[CCIF]=0），除了FCNFG和FSTAT，不接受对任何寄存器的写入。在启动复位序列中，在CCIF初始上升之前，禁写规则被放松。在这个初始化期间，用户可以写入任何寄存器。当擦除暂停请求激活时（FCNFG[ERSSUSP]=1），所有寄存器的写入也被禁止

（除了寄存器FCNFG和FSTAT）。

## 33.4.4.1 FTFC寄存器描述

### 33.4.4.1.1 FTFC内存图

FTFC的基础地址。4002_0000h

| 偏移 | 注册 | 宽度 (单位：比特) | 访问 | 重置值 |
|---|---|---|---|---|
| 0h | 闪存状态寄存器（FSTAT） | 8 | W1C | 80h |
| 1h | 闪存配置寄存器（FCNFG） | 8 | RW | 02h |
| 2h | 闪存安全寄存器（FSEC） | 8 | RO | 见描述。 |
| 3h | 闪存选项寄存器（FOPT） | 8 | RO | 见描述。 |
| 4h | 闪存公共命令对象寄存器（FCCOB3） | 8 | RW | 00h |
| 5h | 闪存公共命令对象寄存器（FCCOB2） | 8 | RW | 00h |
| 6h | 闪存公共命令对象寄存器（FCCOB1） | 8 | RW | 00h |
| 7h | 闪存公共命令对象寄存器（FCCOB0） | 8 | RW | 00h |
| 8h | 闪存公共命令对象寄存器（FCCOB7） | 8 | RW | 00h |
| 9h | 闪存公共命令对象寄存器（FCCOB6） | 8 | RW | 00h |
| 啊 | 闪存公共命令对象寄存器（FCCOB5） | 8 | RW | 00h |
| ÃÃÃ | 闪存公共命令对象寄存器（FCCOB4） | 8 | RW | 00h |
| 丘吉尔 | 闪存公共命令对象寄存器（FCCOBB）。 | 8 | RW | 00h |
| ÃÃÃ | 闪存公共命令对象寄存器（FCCOBA）。 | 8 | RW | 00h |
| Eh | 闪存公共命令对象寄存器（FCCOB9）。 | 8 | RW | 00h |
| Fh | 闪存公共命令对象寄存器（FCCOB8） | 8 | RW | 00h |
| 10h | 编程闪存保护寄存器（FPRT3）。 | 8 | RW | 见描述。 |
| 11h | 编程闪存保护寄存器（FPRT2）。 | 8 | RW | 见描述。 |
| 12h | 编程闪存保护寄存器（FPROT1）。 | 8 | RW | 见描述。 |
| 13h | 编程闪存保护寄存器（FPROT0）。 | 8 | RW | 见描述。 |
| 16h | EEPROM保护寄存器（FEPROT）。 | 8 | RW | 见描述。 |
| 17h | 数据闪存保护寄存器（FDPROT）。 | 8 | RW | 见描述。 |

| | | | | |
|---|---|---|---|---|
| 2Ch | 闪存CSEc状态寄存器（FCSESTAT）。 | 8 | RO | 00h |
| 2Eh | 闪存错误状态寄存器（FERSTAT）。 | 8 | W1C | 00h |
| 2Fh | 闪存错误配置寄存器（FERCNFG）。 | 8 | RW | 00h |

## 33.4.4.1.2 闪存状态寄存器（FSTAT）

### 33.4.4.1.2.1 偏移

| 注册 | 偏移 |
|---|---|
| FSTAT | 0h |

### 33.4.4.1.2.2 职能

FSTAT寄存器报告FTFC模块的运行状态。

CCIF、RDCOLERR、ACCERR和FPVIOL位是可读和可写的。MGSTAT0位是只读的。未分配的位读为0，不可写。

**注意事项**

当设置时，该寄存器中的访问错误(ACCERR)和闪存保护违规(FPVIOL)位会阻止向FlexRAM发起任何更多的命令或写入(当EEERDY被设置时)，直到标志被清除(通过向其写一)。

### 33.4.4.1.2.3 图示

| 咬牙切齿 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | CCIF | 辽宁沈阳 | ACCERR | 羟基苯丙胺 (FPVIOL) | 0 | | | MGSTAT0 |
| W | W1C | W1C | W1C | W1C | | | | |
| 复位 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

### 33.4.4.1.2.4 领域

| 场地 | 职能 |
|---|---|

*表继续在下一页...*

| | |
|---|---|
| 7<br>CCIF | 命令完成中断标志<br><br>CCIF标志表示一个FTFC命令或仿真EEPROM文件系统操作已经完成。CCIF标志通过向CCIF写1来启动一个命令而被清除，CCIF保持低电平直到命令完成或命令违反。CCIF标志也会在EEE启用时通过对FlexRAM的成功写入而被清除，并且CCIF保持低电平，直到仿真EEPROM文件系统创建了相关的EEPROM数据记录。CCIF标志也会在执行任何CSEc命令时被清除，并在完成后被设置。 |

| 场地 | 职能 |
|---|---|
| | CCIF位被重置为0，但在重置初始化序列结束时被内存控制器设置为1。取决于复位释放后的读取速度，用户可能看到也可能看不到0的硬件复位值。<br>0b - FTFC命令或模拟的EEPROM文件系统操作正在进行中<br>1b - FTFC命令或模拟的EEPROM文件系统操作已经完成。 |
| 6<br>辽宁沈阳，辽宁省沈阳市，辽宁省沈阳市，辽宁省沈阳市，辽宁省沈阳市。 | FTFC 读取碰撞错误标志<br>RDCOLERR错误位表明MCU试图从FTFC资源中读取，而该资源正在被FTFC命令操纵（CCIF=0）。任何同时的访问都会被块仲裁逻辑检测为碰撞错误。这种情况下的读取数据不能得到保证。向RDCOLERR位写1，可以清除它。向RDCOLERR写0没有效果。<br>0b - 没有检测到碰撞错误 1b - 检测到碰撞错误 |
| 5<br>ACCERR | 闪存访问错误标志<br>ACCERR错误位表示对FTFC资源发生了非法访问，原因是违反了命令写入顺序或发出了非法FTFC命令。当 ACCERR 被设置时，CCIF 标志不能被清除以启动一个命令。在CCIF被设置时，通过向ACCERR写1来清除ACCERR位。把0写到ACCERR位上没有任何作用。<br>0b - 没有检测到访问错误 1b - 检测到访问错误 |
| 4<br>羟基苯丙胺(FPVIOL) | 违反闪光保护标志<br>FPVIOL错误位表明在命令写入序列中试图对程序闪存或数据闪存的受保护区域的地址进行编程或擦除，或者在为仿真EEPROM启用时试图对FlexRAM的受保护区域进行写入。当FPVIOL被设置时，CCIF标志不能被清除以启动一个命令。当CCIF被设置时，通过向FPVIOL写1来清除FPVIOL位。把0写到FPVIOL位上没有任何作用。<br>0b - 未检测到违反保护规定 1b - 检测到违反保护规定 |
| 3-1<br>- | 保留 |
| 0<br>MGSTAT0 | 内存控制器命令完成状态标志<br>如果在执行FTFC命令期间或闪存重置序列期间检测到错误，MGSTAT0状态标志将被设置。作为一个状态标志，这个位不能（也不需要）像这个寄存器中的其他错误标志一样被用户清除。<br>命令-N "的MGSTAT0位的值只有在CCIF=1时 "命令-N "执行结束，下一条命令启动之前才有效。在 "命令-N+1 "的执行过程中的某个时刻，之前的结果被丢弃，之前的错误被清除。 |

## 33.4.4.1.3 闪存配置寄存器（FCNFG）

### 33.4.4.1.3.1 偏移

*表继续在下一页...*

| 注册 | 偏移 |
| --- | --- |
| FCNFG | 1h |

### 33.4.4.1.3.2 职能

该寄存器提供了关于FTFC模块当前功能状态的信息。

擦除控制位（ERSAREQ和ERSSUSP）有写入限制。RAMRDY和EEERDY是只读的状态位。RAMRDY和EEERDY位的复位值是在复位程序中确定的。

### 33.4.4.1.3.3 图示

| 咬牙切齿 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | CCIE | 辽宁沈阳 | ERSAREQ | 鄂尔多斯市P | 0 | 0 | 镭迪 | EEERDY |
| W | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |

### 33.4.4.1.3.4 领域

| 场地 | 职能 |
|---|---|
| 7<br>CCIE | 命令完成中断启用<br>CCIE位控制FTFC命令完成后的中断产生。<br>0b - 禁用命令完成中断<br>1b - 启用命令完成中断。只要FSTAT[CCIF]标志被设置，就会产生一个中断请求。 |
| 6<br>辽宁沈阳 | 读取碰撞错误中断启用<br>RDCOLLIE位控制FTFC读取碰撞错误发生时的中断产生。<br>0b - 禁用读取碰撞错误中断<br>1b - 启用读碰撞错误中断。每当检测到FTFC读取碰撞错误时，就会产生中断请求（见FSTAT[RDCOLERR]的描述）。 |
| 5<br>ERSAREQ | 删除所有请求<br>这个位向内存控制器发出请求，以执行擦除所有块的命令并释放安全。ERSAREQ不能直接写入，而是由用户间接控制。关于如何请求这个命令，请参考设备的芯片配置细节。<br>当FTFC外部触发擦除所有请求且CCIF被设置时，ERSAREQ位被设置（当前没有命令被执行）。当操作完成后，ERSAREQ被FTFC清零。<br>0b - 没有请求或请求完成<br>1b - 要求。(1)运行擦除所有块的命令，(2)验证擦除状态，(3)将Flash配置字段中的安全字节编程为 |

*表继续在下一页...*

| | |
|---|---|
| | 非安全状态，(4)通过将FSEC[SEC]字段设置为非安全状态来释放MCU安全。 |
| 4<br>ĀĀĀ | 擦除 暂停<br>ERSSUSP位允许用户在执行擦除Flash扇区的命令时暂停（中断）该命令。 |

| 场地 | 职能 |
|---|---|
| | 0b - 不要求暂停<br>1b - 暂停当前擦除Flash扇区的命令执行。 |
| 3<br>- | 保留 |
| 2<br>- | 保留 |
| 1<br>镭迪 | RAM就绪<br>该标志表示FlexRAM的当前状态。<br>RAMRDY标志的状态通常由Set FlexRAM Function命令控制。在复位序列中，如果FlexNVM块被划分为仿真EEPROM，并选择在复位序列中加载FlexRAM，则RAMRDY标志被清除；如果FlexNVM块没有被划分为仿真EEPROM，或者FlexNVM块被划分为仿真EEPROM，并选择在复位序列中不加载FlexRAM，则RAMRDY标志将被设置。如果运行Program Partition命令对FlexNVM块进行仿真EEPROM的分区，RAMRDY标志被清除。在完成擦除所有块的命令或执行FTFC外部触发的擦除所有操作后，RAMRDY标志被设置。<br>0b - FlexRAM不能用于传统的RAM访问<br>1b - FlexRAM仅作为传统的RAM使用；对FlexRAM的写入不会触发EEPROM操作。 |
| 0<br>EPDY | EPDY<br>这个标志表明EEPROM的备份数据是否已经被复制到FlexRAM中，因此可供读取访问。在复位序列中，当CCIF=0时，EEERDY标志保持清零，只有当FlexNVM块被分割为仿真EEPROM时才会设置。<br>0b - FlexRAM不能用于模拟EEPROM操作<br>1b - FlexRAM可用于EEPROM操作，其中。(1)从FlexRAM中读出的数据会返回之前在模拟EEPROM模式下写入FlexRAM的数据，(2)写出的数据会启动EEPROM操作，将写入的数据存储在FlexRAM和EEPROM备份中。 |

## 33.4.4.1.4 闪存安全寄存器（FSEC）

### 33.4.4.1.4.1 偏移

| 注册 | 偏移 |
|---|---|
| FSEC | 2h |

### 33.4.4.1.4.2 职能

这个只读寄存器持有与MCU和FTFC模块安全相关的所有位。

在复位序列中，该寄存器被加载了位于程序闪存中的闪存配置字段的闪存安全字

*表继续在下一页...*

节的内容。

### 33.4.4.1.4.3 图示

| 咬牙切齿 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | KEYEN | | MEEN | | FSLACC | | SEC | |
| W | | | | | | | | |
| 重置 | u | u | u | u | u | u | u | u |

### 33.4.4.1.4.4 领域

| 场地 | 职能 |
|---|---|
| 7-6<br>KEYEN | 启用后门钥匙安全<br>这些位启用和禁用对FTFC模块的后门密钥访问。<br>00b - 禁用后门钥匙访问<br>01b - 禁用后门钥匙访问（首选KEYEN状态，禁用后门钥匙访问） 10b - 启用后门钥匙访问<br>11b - 禁用后门钥匙访问 |
| 5-4<br>MEEN | 大规模擦除启用位<br>启用和禁用FTFC模块的大规模擦除能力。MEEN位的状态只有在NVM正常模式之外的SEC位被设置为安全时才有意义。当SEC字段被设置为不安全时，MEEN设置并不重要。<br>00b - 启用大规模擦除 01b - 启用大规模擦除 10b - 禁止大规模擦除 11b - 启用大规模擦除 |
| 3-2<br>FSLACC | 工厂故障分析访问代码<br>这些位在工厂进行返回的部件故障分析时启用或禁用对闪存内容的访问。当SEC是安全的，FSLACC被拒绝时，对程序闪存内容的访问被拒绝，在工厂测试期间进行的任何故障分析必须从完全擦除开始，以解除零件的安全。<br>当访问被授予时（SEC为不安全，或SEC为安全且FSLACC被授予），工厂测试对当前闪存内容具有可见性。FSLACC位的状态只有在SEC位被设置为安全时才有意义。当SEC字段被设置为不安全时，FSLACC设置并不重要。<br>00b - 准许工厂访问 01b - 拒绝工厂访问 10b - 拒绝工厂访问<br>11b - 准许工厂访问 |

| | |
|---|---|
| 1-0<br>SEC | 闪存安全<br>这些位定义了MCU的安全状态。在安全状态下，MCU限制对FTFC模块资源的访问。这些限制是按设备定义的，并在芯片配置细节中详细说明。如果FTFC模块使用后门密钥访问不安全，则SEC位被强制为10b。<br>00b - MCU安全状态是安全的 01b - MCU安全状态是安全的<br>10b - MCU安全状态为不安全（FTFC的标准运输条件为不安全） 11b - MCU安全状态为安全 |

### 33.4.4.1.5 闪存选项寄存器（FOPT）

#### 33.4.4.1.5.1 偏移

| 注册 | 偏移 |
|---|---|
| 基金会 | 3h |

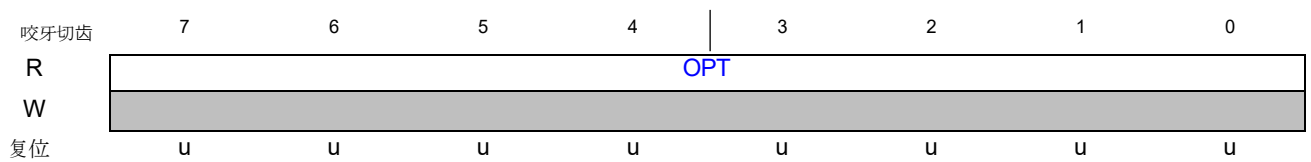
#### 33.4.4.1.5.2 职能

闪存选项寄存器允许MCU通过检查这些只读位的状态来定制其操作，这些位在复位时从NVM加载。这些位的功能在器件的芯片配置细节中定义。

该寄存器中的所有位都是只读的。

在复位序列中，该寄存器从位于程序闪存中的闪存配置字段中的闪存非易失性选项字节加载。

#### 33.4.4.1.5.3 图示




#### 33.4.4.1.5.4 领域

| 场地 | 职能 |
|---|---|
| 7-0<br>OPT | 非挥发性选项<br>这些位在复位时从闪存加载到这个寄存器。关于这些位的定义和使用，请参考器件的芯片配置细节。 |

### 33.4.4.1.6 闪存公共命令对象寄存器（FCCOB3 - FCCO B8）。

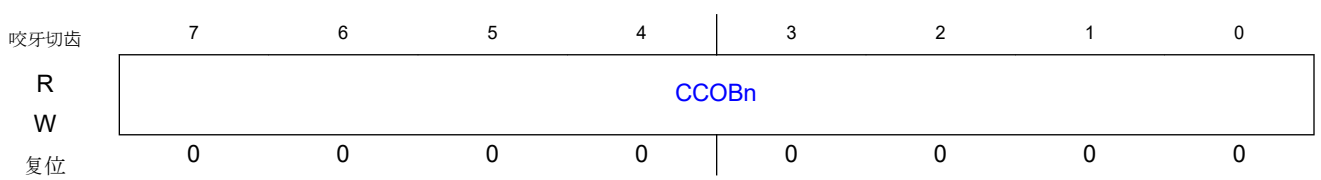
#### 33.4.4.1.6.1 偏移

| 注册 | 偏移 |
|---|---|
| FCCOB3 | 4h |
| FCCOB2 | 5h |
| FCCOB1 | 6h |
| FCCOB0 | 7h |
| FCCOB7 | 8h |
| FCCOB6 | 9h |
| FCCOB5 | 啊 |
| FCCOB4 | 卜 |
| ÃÃÃ | 丘吉尔 |
| ÃÃÃ | ÃÃÃ |
| FCCOB9 | Eh |
| FCCOB8 | Fh |

#### 33.4.4.1.6.2 职能

FCCOB寄存器组为命令代码和参数提供12个字节。该组中的各个字节在FCCOB寄存器名称后附加一个0-B的十六进制标识符：FCCOB0, FCCOB1, ..., FCCOBB。

#### 33.4.4.1.6.3 图示




#### 33.4.4.1.6.4 领域

| 场地 | 职能 |
|---|---|
| 7-0<br>CCOBn | CCOBn<br>FCCOB寄存器向内存控制器提供一个命令代码和相关参数。组成FCCOB数据集的各个寄存器可以按任何顺序写入，但你必须提供所有需要的值，这些值因命令而异。首先，设置所有需要的FCCOB字段，然后通过向FSTAT[CCIF]位写1来启动命令的执行。这样就清除了CCIF位，它锁定了所有FCCOB参数字段，在命令完成之前，用户不能改变它们（CCIF返回到1）。没有提供命令缓冲或队列，只有在当前命令完成后才能加载下一条命令。<br>一些命令将信息返回到FCCOB寄存器中。任何返回到FCCOB的值都可以在内存控制器的FSTAT[CCIF]标 |

| | |
|---|---|
| | 志返回到1之后进行读取。 |

<table>
<tr><th>场地</th><th>职能</th></tr>
<tr><td></td><td>
下表显示了一个通用的FTFC命令格式。第一个FCCOB寄存器，FCCOB0，总是包含命令代码。这个8位值定义了要执行的命令。命令代码后面是该特定FTFC命令所需的参数，通常是地址和/或数据值。<br><br>
注意：命令参数表是以FCCOB编号（相当于字节号）来写的。这个数字是对FCCOB寄存器名称的参考，而不是寄存器的地址。
<table>
<tr><th>FCCOB号码</th><th>典型命令参数内容 [7:0]</th></tr>
<tr><td>0</td><td>FCMD（一个定义FTFC命令的代码）。</td></tr>
<tr><td>1</td><td>闪存地址[23:16]</td></tr>
<tr><td>2</td><td>闪存地址[15:8]</td></tr>
<tr><td>3</td><td>闪存地址 [7:0]</td></tr>
<tr><td>4</td><td>数据字节0</td></tr>
<tr><td>5</td><td>数据字节1</td></tr>
<tr><td>6</td><td>数据字节2</td></tr>
<tr><td>7</td><td>数据字节3</td></tr>
<tr><td>8</td><td>数据字节4</td></tr>
<tr><td>9</td><td>数据字节5</td></tr>
<tr><td>A</td><td>数据字节6</td></tr>
<tr><td>B</td><td>数据字节7</td></tr>
</table>
1.指的是FCCOB寄存器名称，而不是寄存器地址<br><br>
FCCOB端性和多字节访问。<br><br>
FCCOB 寄存器组使用大 endian 寻址惯例。对于所有大于1字节的命令参数字段，最重要的数据驻留在最低的FCCOB寄存器编号中。FCCOB寄存器组可以作为单个字节、对齐的字（2字节）或对齐的长字（4字节）进行读写。
</td></tr>
</table>

1.指的是FCCOB寄存器名称，而不是寄存器地址

## 33.4.4.1.7 编程闪存保护寄存器（FPROT3 - FPROT0）。

### 33.4.4.1.7.1 偏移

对于a=0到3。

| 注册 | 偏移 |
|---|---|
| FPROTa | Dh + (a × 1h) |

#### 33.4.4.1.7.2 职能

FPROT寄存器定义了哪些编程闪存区域被保护起来，免于编程和擦除操作。受保护的闪存区域不能改变其内容；也就是说，这些区域不能被编程，也不能被任何FTFC命令擦除。
不受保护的区域可以通过编程和擦除操作来改变。

如果配置的PFLash大小不是2^n的倍数，就把region_size取整到下一个2^n大小。例如，2MB的宏是用1.5MB的PFLash加上512kB的混合P/DFlash（448kB/64kB）建立的。在这个计算中，PFLASH应该使用2MB作为 region_size。

四个FPROT寄存器允许多达32个相同内存大小的可保护区域。FPROT寄存器的位被映射到可保护区域，如下所示。

- *region_size* = total_program_flash_size / 32
- 从程序闪存基地址开始，区域0指的是第一个*区域_大小*字节，region 1指的是下一个*region_size*字节，以此类推。

换句话说。
- region_n_start_address = program_flash_base_address + (n) * *region_size*
- region_n_end_address = program_flash_base_address + ((n+1) * *region_size*) -1

例如，如果程序闪存的总容量是512MBytes，*region_size*是16KB。如果最低的程序闪存地址是0x000A_0000，区域的定义如下。

| 地区 | 起始地址 | 最终地址 |
|---|---|---|
| 0 | 0x000A_0000 | 0x000A_3FFF |
| 1 | 0x000A_4000 | 0x000A_7FFF |
| 2 | 0x000A_8000 | 0x000A_BFFF |
| 3 | 0x000A_C000 | 0x000A_FFFF |
| .. . | .. . | .. . |
| 32 | 0x0011_C000 | 0x0011_FFFF |

- FPROT[PROT]位中的'0'表示其对应的闪存保护区域未受保护。
- FPROT[PROT]位中的'1'表示其对应的闪存保护区域受到保护。

| 编程闪光保护寄存器 | 编程闪存保护位 |
|---|---|
| FPROT0 | PROT[31:24] |

*表继续在下一页...*

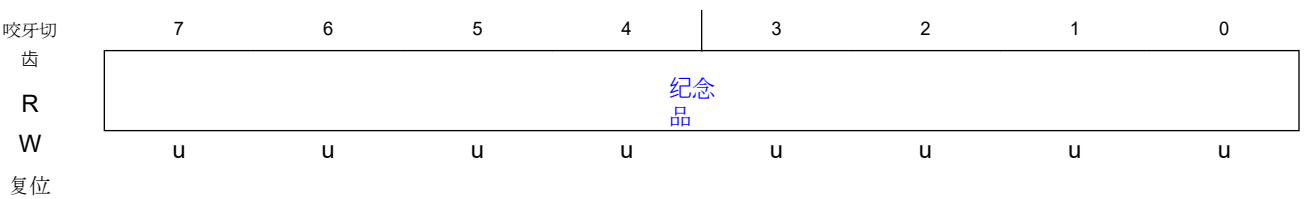
| 编程闪光保护寄存器 | 编程闪存保护位 |
|---|---|
| FPROT1 | PROT[23:16] |
| FPROT2 | PROT[15:8] |
| FPROT3 | PROT[7:0] |

在复位过程中，FPROT寄存器被加载闪存配置字段中的程序闪存保护字节的内容，如下表所示。

| 编程闪光保护寄存器 | 闪存配置字段偏移地址 |
|---|---|
| FPROT0 | 0x000B |
| FPROT1 | 0x000A |
| FPROT2 | 0x0009 |
| FPROT3 | 0x0008 |

要改变复位序列中加载的程序闪存保护，请解除对包含闪存配置字段的程序闪存扇区的保护。然后，重新对程序闪存保护字节进行编程。

### 33.4.4.1.7.3 图示




### 33.4.4.1.7.4 领域

| 场地 | 职能 |
|---|---|
| 7-0<br>纪念品 | 程序闪光区域保护<br>通过将相关的PROT位设置为保护状态，可以保护每个编程闪存区域免受编程和擦除操作。<br>**在NVM正常模式下。**保护只能增加，也就是说，当前未受保护的内存可以被保护，但当前受保护的内存不能被解除保护。由于未受保护的区域用1标记，受保护的区域用0标记，所以只接受将1变为0的写入。这种1到0的转换检查是在逐位进行的。那些具有1-0转换的FPROT位被接受，而所有具有0-1转换的位被忽略。<br>**在NVM特殊模式下。**FPROT的所有位都可以不受限制地写入。未受保护的区域可以被保护，受保护的区域可以不受保护。<br>**限制条件。**用户决不能在命令运行时对任何FPROT寄存器进行写入（CCIF=0）。 |

| 场地 | 职能 |
| --- | --- |
| | 试图改变程序闪存中任何受保护区域的数据会导致违反保护的错误，并设置FSTAT[FPVIOL]位。如果一个程序闪存块包含任何受保护的区域，就不可能对其进行全块擦除。 |

## 33.4.4.1.8 EEPROM保护寄存器（FEPROT）。

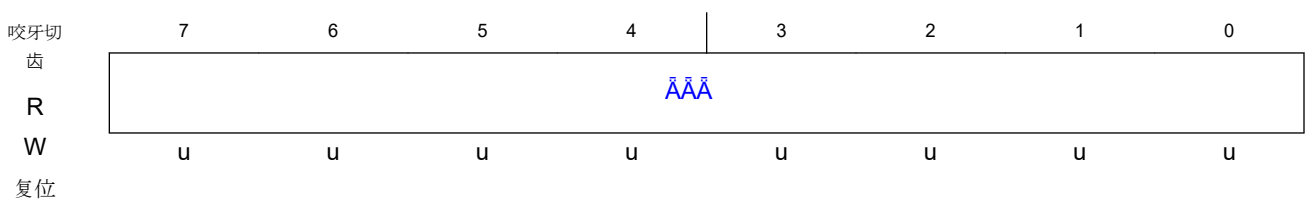
### 33.4.4.1.8.1 偏移

| 注册 | 偏移 |
| --- | --- |
| 基金会 | 16h |

### 33.4.4.1.8.2 职能

FEPROT寄存器定义了FlexRAM的哪些仿真EEPROM区域被保护，以防止编程和擦除操作。受保护的EEPROM区域不能通过写入它来改变其内容。不受保护的区域可以通过对FlexRAM的写入来改变。

### 33.4.4.1.8.3 图示




### 33.4.4.1.8.4 领域

| 场地 | 职能 |
| --- | --- |

| | |
|---|---|
| 7-0<br>ĂĂĂ | EEPROM区域保护<br>通过将相关的EPROT位设置为保护状态，可以保护单个模拟EEPROM区域不被改变。当FlexNVM分区代码只设置为数据闪存时，不使用EPROT位。当FlexNVM分区代码被设置为数据闪存和EEPROM或仅EEPROM时，每个EPROT位覆盖了配置的EEPROM数据的八分之一（参见EEPROM数据集大小参数描述）。<br>**在NVM正常模式下。**只能增加保护。这意味着当前未受保护的内存可以被保护，但当前受保护的内存不能被解除保护。由于未受保护的区域用1标记，受保护的区域用0标记，所以只接受将1变为0的写入。这种1到0的转换检查是在一个位的基础上进行的。那些具有1-0转换的FEPROT位<br>被接受，而所有从0到1的转换的位都被忽略。 |

| 场地 | 职能 |
| --- | --- |
| | **在NVM特殊模式下。**FEPROT寄存器的所有位都可以不受限制地写入。未受保护的区域可以被保护，受保护的区域可以不受保护。<br>**限制条件。**当一个命令正在运行时，不要写到FEPROT寄存器（CCIF=0）。<br>**复位。**在复位序列中，FEPROT寄存器被加载了位于程序闪存中的闪存配置字段的FlexRAM保护字节的内容。复位值的闪存基础在寄存器图中用X表示。要改变将在复位序列中加载的EEPROM保护，包含闪存配置字段的程序闪存扇区必须被解除保护；然后EEPROM保护字节必须被擦除并重新编程。<br>试图通过写入仿真EEPROM中的任何受保护区域来改变数据会导致违反保护的错误，并设置FSTAT[FPVIOL]位。<br>00000000b - EEPROM区域被保护<br>00000001b - EEPROM区域未被保护 |

## 33.4.4.1.9 数据闪存保护寄存器（FDPROT）。

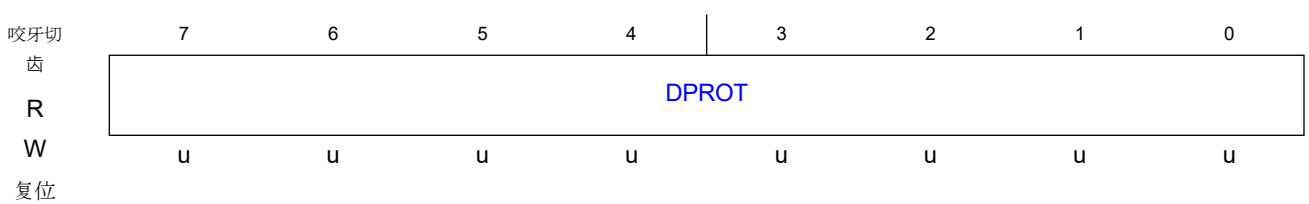
### 33.4.4.1.9.1 偏移

| 注册 | 偏移 |
| --- | --- |
| 辽宁省 | 17h |

### 33.4.4.1.9.2 职能

FDPROT寄存器定义了哪些数据闪存区域被保护起来，以防止编程和擦除操作。受保护的闪存区域不能改变其内容，也就是说，这些区域不能被编程，也不能被任何FTFC命令擦除。
不受保护的区域可以通过编程和擦除操作来改变。

### 33.4.4.1.9.3 图示



### 33.4.4.1.9.4 领域

| 场地 | 职能 |
| --- | --- |
| 7-0 | 数据闪存区域保护 |

| 场地 | 职能 |
| --- | --- |
| DPROT | 通过将相关的DPROT位设置为保护状态，可以保护单个数据闪存区域免受编程和擦除操作。每个DPROT位保护八分之一的分区数据闪存空间。数据闪存保护的颗粒度不能小于数据闪存扇区的大小。如果一个未使用的DPROT位被设置为保护状态，擦除所有区块的命令就不会执行，并设置FSTAT[FPVIOL]位。<br>**在NVM正常模式下。**保护只能增加，也就是说，当前未受保护的内存可以被保护，但当前受保护的内存不能被解除保护。由于未受保护的区域用1标记，受保护的区域用0标记，所以只接受将1变为0的写入。这种1到0的转换检查是在逐位进行的。那些具有1-0转换的FDPROT位被接受，而所有具有0-1转换的位被忽略。<br>**在NVM特殊模式下。**FDPROT寄存器的所有位都可以不受限制地写入。未受保护的区域可以被保护，受保护的区域可以不受保护。<br>**限制条件。**当一个命令正在运行时，用户决不能对FDPROT寄存器进行写入（CCIF=0）。<br>**复位。**在复位过程中，FDPROT寄存器被加载位于程序闪存中的闪存配置字段的数据闪存保护字节的内容。复位值的闪存基础在寄存器图中用X表示。要改变复位过程中加载的数据闪存保护，请解除对包含闪存配置域的程序闪存扇区的保护。然后，擦除并重新编程数据闪存的保护字节。<br>试图用编程和擦除命令改变数据闪存中任何受保护区域的数据会导致违反保护的错误，并设置FSTAT[FPVIOL]位。如果数据闪存块包含任何受保护的区域，或者FlexNVM内存已经被划分为仿真EEPROM，则不可能对任何数据闪存块进行块擦除（见擦除闪存块命令描述）。<br>00000000b - 数据闪存区域受到保护 00000001b - 数据闪存区域不受保护 |

## 33.4.4.1.10 闪存CSEc状态寄存器（FCSESTAT）。

### 33.4.4.1.10.1 偏移

| 注册 | 偏移 |
| --- | --- |
| FCSESTAT | 2Ch |

### 33.4.4.1.10.2 职能

FCSESTAT寄存器报告CSEc加密相关功能集操作的状态。

以下是FCSESTAT寄存器中CSEc状态标志的快速参考映射。

**表33-3.启动/MAC映射到CSEc状态标志**

| | SB | BIN | BFN | BOK | <错误代码 |
| --- | --- | --- | --- | --- | --- |

| | | | | | |
|---|---|---|---|---|---|
| 没有靴子类型 | 0 | 0 | 0 | 0 | 0 |

*表继续在下一页...*

表33-3.启动/MAC映射到CSEc状态标志（续）

| | SB | BIN | BFN | BOK | <错误代码 |
|---|---|---|---|---|---|
| MAC是空的 | 1 | 1 | 1 | 0 | NO_ERR |
| MAC不匹配 | 1 | 0 | 1 | 0 | NO_ERR |
| MAC比赛 | 1 | 0 | 0 | 1 | NO_ERR |
| MAC_KEY为空 | 0 | 0 | 1 | 0 | NO_SECURE_BOOT |

### 33.4.4.1.10.3 图示

| 咬牙切齿 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | IDB | EDB | RIN | BOK | BFN | BIN | SB | ÃÃÃ |
| W | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

### 33.4.4.1.10.4 领域

| 场地 | 职能 |
|---|---|
| 7<br>IDB | 内部调试<br>当没有存储用户钥匙时，IDB位由DBG_AUTH命令设置。它在复位时和LOAD_KEY命令中被清除。<br>0b - 内部调试功能被禁用 1b - 内部调试器功能被启用 |
| 6<br>EDB | 外部调试<br>当CPU调试端口（BDM/JTAG/等）被激活时，EDB位被设置。它在复位时被清除。<br>0b - 未连接外部调试器 1b - 已连接外部调试器 |
| 5<br>RIN | 初始化的随机数发生器<br>RIN位由INIT_RNG命令设置。它在复位和DBG_AUTH命令中被清除。<br>0b - 随机数发生器未被初始化。1b - 随机数发生器已被初始化。 |
| 4<br>BOK | 安全启动OK<br>BOK位在安全启动过程成功完成后被设置。它在复位或 BOOT_FAILURE 命令下被清除。如果没有BOOT_MAC/BOOT_MAC_KEY/boot flavor的选择，它将保持清零。<br>0b - 安全启动未完成，或安全启动失败 1b - 安全启动成功。 |

| | |
|---|---|
| 3<br>BFN | 安全启动完成<br><br>BFN 位由 BOOT_FAILURE 或 BOOT_OK 命令设置，或者当 BIN 位被设置，遇到错误，或者 BOOT_MAC 值不匹配时，安全启动过程也会设置 BFN。它假定BOOT_MAC_KEY是存在的，并且启动味道被配置为安全启动类型（串行、并行、严格）。它在复位时被清除。 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| | 0b - 安全启动未完成。1b - 安全<br>启动已经完成 |
| 2<br>BIN | 安全启动初始化<br>如果BOOT_MAC内存插槽是空的，BIN位会被安全启动过程设置。它在复位时被清空。假设BOOT_MAC_KEY是存在的。(根据SHE第10.3节自主启动)<br>0b - 安全启动个性化定制未完成。1b - 安全启动个<br>性化设置已经完成 |
| 1<br>SB | 安全启动<br>如果BOOT_MAC_KEY槽不是空的，SB位会被安全启动过程设置。它在复位时被清空。假设安全引导的味道被配置了（串行、并行或严格）。<br>0b - 未激活安全启动 1b - 激活<br>了安全启动 |
| 0<br>ÃÃÃ | 忙碌的<br>当命令发出时，BSY位被设置。当命令处理完成时，它被清零 0b - 命令处理已完成<br>1b - 命令处理正在进行中 |

## 33.4.4.1.11 闪存错误状态寄存器（FERSTAT）。

### 33.4.4.1.11.1 偏移

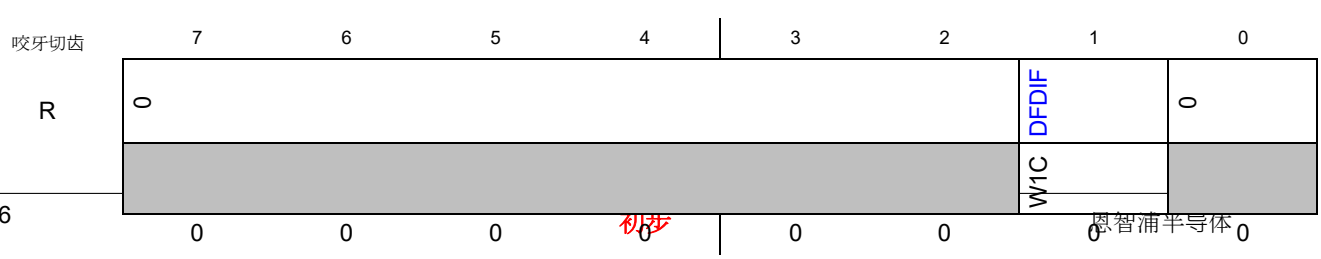
| 注册 | 偏移 |
|---|---|
| 实验数据 | 2Eh |

### 33.4.4.1.11.2 职能

该寄存器报告在读访问FTFC模块期间检测到的未纠正的ECC错误。

DFDIF标志是可读和可写的。未分配的位读为0，不可写。

### 33.4.4.1.11.3 图示





初步
恩智浦半导体

W

复位

#### 33.4.4.1.11.4 领域

| 场地 | 职能 |
|---|---|
| 7-2<br>- | 保留 |
| 1<br>DFDIF | 双位故障检测中断标志<br>DFDIF标志表示在从平台闪存控制器进行有效的闪存读取访问期间检测到一个不可纠正的ECC故障。<br>DFDIF标志通过向其写1来清除。向DFDIF写0没有效果。<br>0b - 在从平台闪存控制器进行有效闪存读取访问期间未检测到双位故障 1b - 在从平台闪存控制器进行有效闪存读取访问期间检测到双位故障（或 FERCNFG[FDFD]被设置）。 |
| 0<br>- | 保留 |

### 33.4.4.1.12 闪存错误配置寄存器（FERCNFG）。

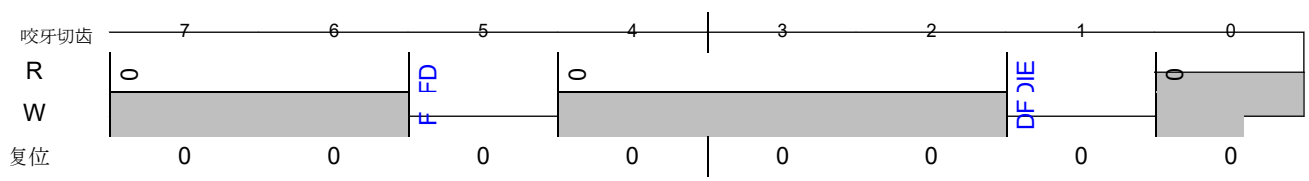
#### 33.4.4.1.12.1 偏移

| 注册 | 偏移 |
|---|---|
| FERCNFG | 2Fh |

#### 33.4.4.1.12.2 职能

该寄存器使在读访问FTFC模块时检测到的未纠正的ECC错误得到强制和中断。FDFD和DFDIE位是可读和可写的。未分配的位读为0，不可写。

#### 33.4.4.1.12.3 图示



#### 33.4.4.1.12.4 领域

| 场地 | 职能 |
|---|---|
| 7-6<br>- | 保留 |
| 5<br>基金会 | 强制双比特故障检测<br>FDFD位使用户能够模拟FERSTAT[DFDIF]标志的设置，以检查相关的中断程序。FDFD位通过向FDFD写0而被清除。<br>0b - FERSTAT[DFDIF]仅在从平台闪存控制器读访问期间检测到双位故障时设置。<br>1b - FERSTAT[DFDIF]在任何来自平台闪存控制器的有效闪存读取访问中被设置。如果DFDIE位被设置，将产生一个中断请求。 |
| 4-2<br>- | 保留 |
| 1<br>DFDIE | 双位故障检测中断启用<br>当从平台闪存控制器进行有效的闪存读取访问期间检测到一个不可纠正的ECC故障时，DFDIE位控制中断的产生。<br>0b - 禁用双位故障检测中断<br>1b - 启用双位故障检测中断。只要FERSTAT[DFDIF]标志被设置，就会产生一个中断请求。 |
| 0<br>- | 保留 |

## 33.5 职能描述

以下章节描述了FTFC模块的功能细节。

### 33.5.1 闪光保护

闪存中的个别区域可以被保护起来，以防止编程和擦除操作。保护是由下列寄存器控制的。

- FPROTn - 四个寄存器保护程序闪存的32个区域，如下图所示

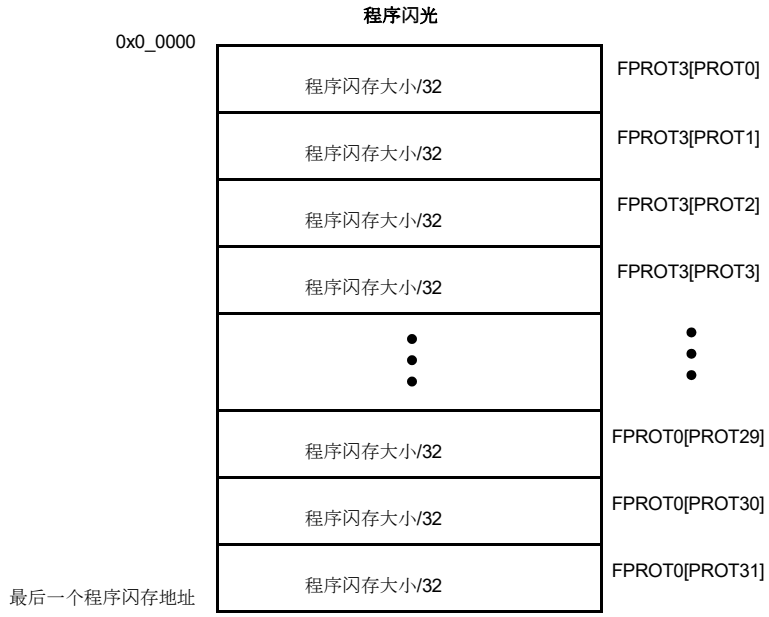



**图33-4.程序闪光保护**

- FDPROT -
  - 对于2n个数据闪存大小，保护数据闪存的八个区域，如下图所示

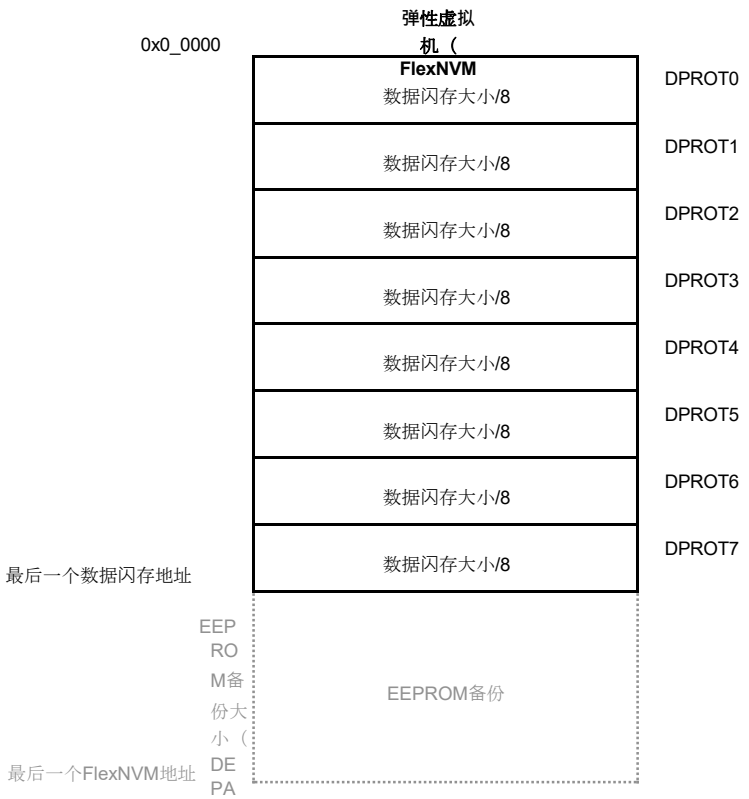



**图33-5.数据闪存保护（2n个数据闪存尺寸）**

- FEPROT - 保护仿真EEPROM存储器的八个区域，如下图所示

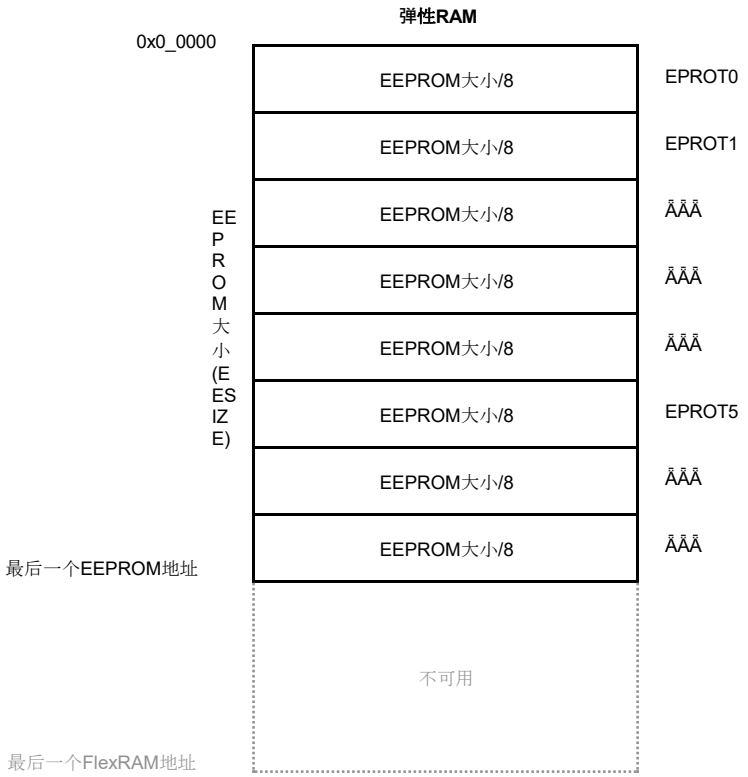



**图33-6.EEPROM保护**

## 33.5.2 FlexNVM描述

本节介绍FlexNVM内存。

### 33.5.2.1 用于FlexRAM的FlexNVM块分区

用户可以将FlexNVM块配置为。

- 基本数据闪光。
- EEPROM闪存记录，以支持内置模拟EEPROM功能，或
- 两者的结合。

当使用CSEc功能集时，必须选择EEE闪存配置。

用户的FlexNVM配置选择是通过Program Partition命令指定的，详见Program Partition命令。

**注意事项**

虽然FlexNVM块的不同分区是可用的，但目的是在特定应用的整个生命周期内使用单一的分区选择。FlexNVM分区代码的选择会影响设备的耐用性和数据保留特性。

### 33.5.2.2 EEPROM用户视角

下图显示了仿真的EEPROM系统。为了处理不同的客户要求，FlexRAM和FlexNVM块可以被分割成不同的分区，如图所示。

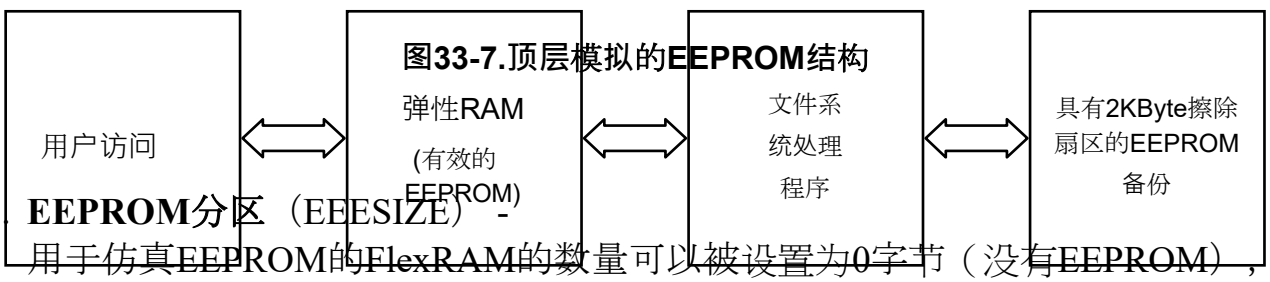



**图33-7.顶层模拟的EEPROM结构**

1. **EEPROM分区**（EEESIZE）-
   用于仿真EEPROM的FlexRAM的数量可以被设置为0字节（没有EEPROM），或者设置为最大的FlexRAM大小。仿真EEPROM分区从FlexRAM地址空间的底部开始增长。利用较少的记录将减少RAM与NVM的比例，从而增加W/E的耐用性。对于EEPROM以外的FlexRAM使用（见Set FlexRAM Function命令。
2. **数据闪存分区**（DEPART）--
   用于数据闪存的FlexNVM内存的数量可以从0字节（所有的FlexNVM块都可用于模拟EEPROM备份）编程到FlexNVM块的最大尺寸。
3. **FlexNVM EEPROM分区** -
   用于仿真EEPROM备份的FlexNVM内存的数量（见图33-3）。为了达到指定的w/e周期的耐久性，仿真EEPROM的备份大小必须是FlexRAM中仿真EEPROM分区大小的至少16倍。对于2MB配置以外的部分（有交错的DFlash），FlexNVM可以根据程序分区命令进行分割，允许部分EFlash和部分DFlash。例如，一个64kB的FlexNVM可以被分割为32kB的仿真EEPROM和32kB的DFlash。为了在仿真EEPROM系统中保持全部指定的w/e耐久性，应该限制存储在FlexRAM中的记录数量，以保持RAM与NVM的1:16比例。记录总数的管理由用户决定。

分区信息（EEESIZE，DEPART）存储在数据闪存IFR中，并通过编程分区命令进行编程（见编程分区命令）。通常情况下，编程分区命令在设备的生命周期内只执行一次。

数据闪存对于需要快速存储大量数据或存储静态数据的应用非常有用。FlexRAM中的模拟EEPROM分区对于存储较小数量的、会经常改变的数据是有用的。

### 33.5.2.3 EEPROM实施概述

在FSTAT[CCIF]位清零的情况下，分区设置（EEESIZE，DEPART）从数据闪存IFR中被读取，仿真EEPROM文件系统被相应地初始化。仿真EEPROM文件系统在EEPROM备份中定位所有有效的EEPROM数据记录，并将最新的数据复制到FlexRAM。在所有有效EEPROM数据记录的数据被复制到FlexRAM之后，FSTAT[CCIF]和FCNFG[EEERDY]位被设置。在CCIF位被设置后，FlexRAM可用于读或写访问。

当配置为仿真EEPROM使用时，写到FlexRAM中不受保护的位置会调用仿真EEPROM文件系统，以轮流的方式在仿真EEPROM备份存储器中编程新的EEPROM数据记录。根据需要，仿真EEPROM文件系统识别正在被擦除的仿真EEPROM备份扇区，以便将来使用，并部分擦除该仿真EEPROM备份扇区。在对FlexRAM进行写入后，FlexRAM不能被访问，直到FSTAT[CCIF]位被设置。FCNFG[EEERDY]位也将被设置。如果启用，与FSTAT[CCIF]位相关的中断可以被用来确定FlexRAM何时可以被读或写访问。

在仿真EEPROM备份中的一个扇区充满了EEPROM数据记录之后，持有最古老数据的扇区的EEPROM数据记录被逐渐复制到一个先前被擦除的仿真EEPROM备份扇区。当扇区复制完成后，持有最旧数据的仿真EEPROM备份扇区被标记为擦除。

**注意事项**

中断仿真EEPROM操作（由于断电、复位、电源超出指定的操作范围或任何其他原因）将使内容处于不确定的状态。用户必须采取适当的应对措施，以防止在模拟EEPROM操作中断的情况下出现数据丢失。

#### 33.5.2.4 对模拟EEPROM的FlexRAM的写入耐久性

当FlexNVM分区代码没有设置为全数据闪存时，模拟的EEPROM数据集大小可以设置为几个非零值中的任何一个。

没有通过FlexNVM分区代码分配给数据闪存的字节被FTFC用来为EEPROM数据获得有效的耐久性增长。内置的仿真EEPROM记录管理系统通过在更大的仿真EEPROM
NVM存储空间中循环使用EEPROM数据，提高了在器件损耗之前可以达到的编程/擦除周期的数量。

虽然有不同的FlexNVM分区，但目的是在特定应用的整个生命周期内，对FlexNVM分区代码和模拟EEPROM数据集大小进行单一选择。

### 33.5.3 中断

FTFC模块可以在各种FTFC事件发生时向MCU产生中断请求。这些中断事件及其相关的状态和控制位显示在下表中。

**表33-4.FTFC中断源**

| FTFC活动 | 可读性<br>状态位 | 中断<br>启用位 |
|---|---|---|
| FTFC指令完成 | FSTAT[CCIF] | FCNFG[CCIE] |
| FTFC读取碰撞错误 | fstat[rdcolerr] | fcnfg[rdcollie]。 |
| FTFC ECC错误检测 | FERSTAT[DFDIF] | FERCNFG[DFDIE] |

**注意事项**

矢量地址和它们的相对中断优先级是在MCU层面确定的。

### 33.5.4 低功耗模式下的闪存操作

#### 33.5.4.1 等待模式

当MCU进入等待模式时，FTFC模块不受影响。FTFC模块可以通过命令完成中断将MCU从等待中恢复过来（见中断）。

#### 33.5.4.2 停止模式

当MCU请求停止模式时，如果FTFC命令处于激活状态（CCIF=0），那么在MCU被允许进入停止模式之前，命令的执行已经完成。

**注意事项**

在任何FTFC命令运行时，MCU决不能进入停止模式（CCIF = 0）。

**注意事项**

当MCU处于极低功率模式（VLPR、VLPW、VLPS）时，FTFC模块不接受闪存命令。

### 33.5.5 运作的功能模式

FTFC模块有两种工作模式。NVM正常和NVM特殊。工作模式会影响命令集的可用性（见表33-5）。关于如何激活每种模式，请参考该设备的芯片配置细节。

### 33.5.6 闪存的读取和忽略的写入

FTFC模块只需要闪存地址来执行闪存读取。单片机对所有闪存都有读取权限。

当命令在该块上运行时（如CCIF=0所示），MCU不得从闪存中读取数据。当任何命令在该块内处理时，不能保证从闪存块中读取数据。区块仲裁逻辑会检测到任何同时的访问，并将其报告为读取碰撞错误（见FSTAT[RDCOLERR]位）。

### 33.5.7 边写边读（RWW）

对于带有FlexNVM的设备，允许以下同时访问。

- 当数据闪存和FlexRAM内存空间中的命令（通常是编程和擦除操作）处于活动状态时，用户可以从程序闪存中读取。
- 在数据闪存编程和擦除操作期间，MCU可以从程序闪存中获取指令，同时模拟的EEPROM备份由EEPROM指令维持。
- 反之，用户可以在程序闪存上执行编程和擦除命令时从数据闪存和FlexRAM中读取数据。
- 当配置为传统RAM时，在数据闪存操作期间，允许对FlexRAM进行写入。

当FlexRAM被用于EEE时，不可能同时进行数据闪存操作和FlexRAM写入。

同时操作在允许同时闪光操作中进一步讨论。

### 33.5.8 Flash编程和擦除

除了读取之外，所有的闪存功能都需要用户通过一系列的外围总线写入来设置和启动一个FTFC命令。用户不能再启动任何进一步的FTFC命令，直到被通知当前的命令已经完成。FTFC命令的结构和操作详见FTFC命令操作。

**注意事项**

中断编程或擦除操作（由于断电、复位、电源超出指定的操作范围或任何其他原因）将使内容处于不确定的状态。用户必须采取适当的应对措施，在程序或擦除操作中断的情况下防止数据丢失。

### 33.5.9 FTFC的指挥业务

FTFC命令（与CSEc无关）操作通常用于修改闪存内容。接下来的章节将描述。

- 用于设置FTFC命令参数和启动执行的命令写入顺序
- 对所有可用的FTFC命令的描述

**注意事项**

关于CSEc/SHE相关的命令，请参考加密服务引擎（CSEc）。

#### 33.5.9.1 命令写入顺序

FTFC的命令是使用图中所示的命令写入顺序来指定的。
33-
8.FTFC模块对命令（FCCOB）内容进行各种检查，如果所有要求都得到满足，就继续执行命令。

在启动一个命令之前，FSTAT寄存器中的ACCERR和FPVIOL位必须为零，CCIF标志必须读为1，以验证之前的任何命令已经完成。如果CCIF为零，先前的命令执行仍然有效，新的命令写入序列不能启动，所有对FCCOB寄存器的写入都被忽略。

试图在VLP模式下启动FTFC命令的行为将被忽略。试图在HSRUN模式下启动任何FTFC命令(包括SHE命令、FlexRAM/EEPROM访问)，将被设置为ACCERR标志。

##### 33.5.9.1.1 加载FCCOB寄存器

用户必须在FCCOB寄存器中加载所需的FTFC命令的所有参数。组成FCCOB数据集的各个寄存器可以按任何顺序写入。用户必须确保提供所有需要的参数（这些参数可能因命令不同而不同）。

##### 33.5.9.1.2 通过清除CCIF来启动命令

一旦所有相关的命令参数被加载，用户通过向FSTAT[CCIF]位写 "1 "来清零，从而启动命令。CCIF标志保持为零，直到FTFC命令完成。

FSTAT寄存器包含一个阻塞机制，如果前一个命令导致访问错误（FSTAT[ACCERR]=1）或违反保护规定（FSTAT[FPVIOL]=1），则阻止新命令的启动（不能清除CCIF）。在出错的情况下，需要对FSTAT进行两次写操作来启动下一条命令：第一次写清除错误标志，第二次写清除CCIF。

##### 33.5.9.1.3 命令执行和错误报告

命令处理有几个步骤。

1. FTFC读取命令代码并执行一系列参数检查和保护检查（如果适用），这些检查对每条命令都是独一无二的。

   如果参数检查失败，FSTAT[ACCERR]（访问错误）标志被设置。ACCERR报告无效的指令代码和越界的地址。通常情况下，访问错误表明指令在FCCOB寄存器组中没有设置有效参数。

   编程和擦除命令也会检查地址，以确定操作是否被要求在受保护区域执行。如果保护检查失败，FSTAT[FPVIOL]（保护错误）标志被设置。

   当参数或保护步骤失败时，命令处理绝不会继续执行。相反，在设置FSTAT[CCIF]位后，命令处理被终止。

2. 如果参数和保护检查通过，命令继续执行。在执行阶段可能会发生运行时错误，比如擦除验证失败。运行时错误会在FSTAT[MGSTAT0]位上报告。一个命令可能有访问错误、保护错误和运行时错误，但在所有访问和保护错误被纠正之前，运行时错误是看不到的。
3. 命令的执行结果，如果适用，将通过FCCOB和FSTAT寄存器反馈给用户。
4. FTFC设置FSTAT[CCIF]位，标志着命令已经完成。下图显示了一个通用的命令写入序列的流程。

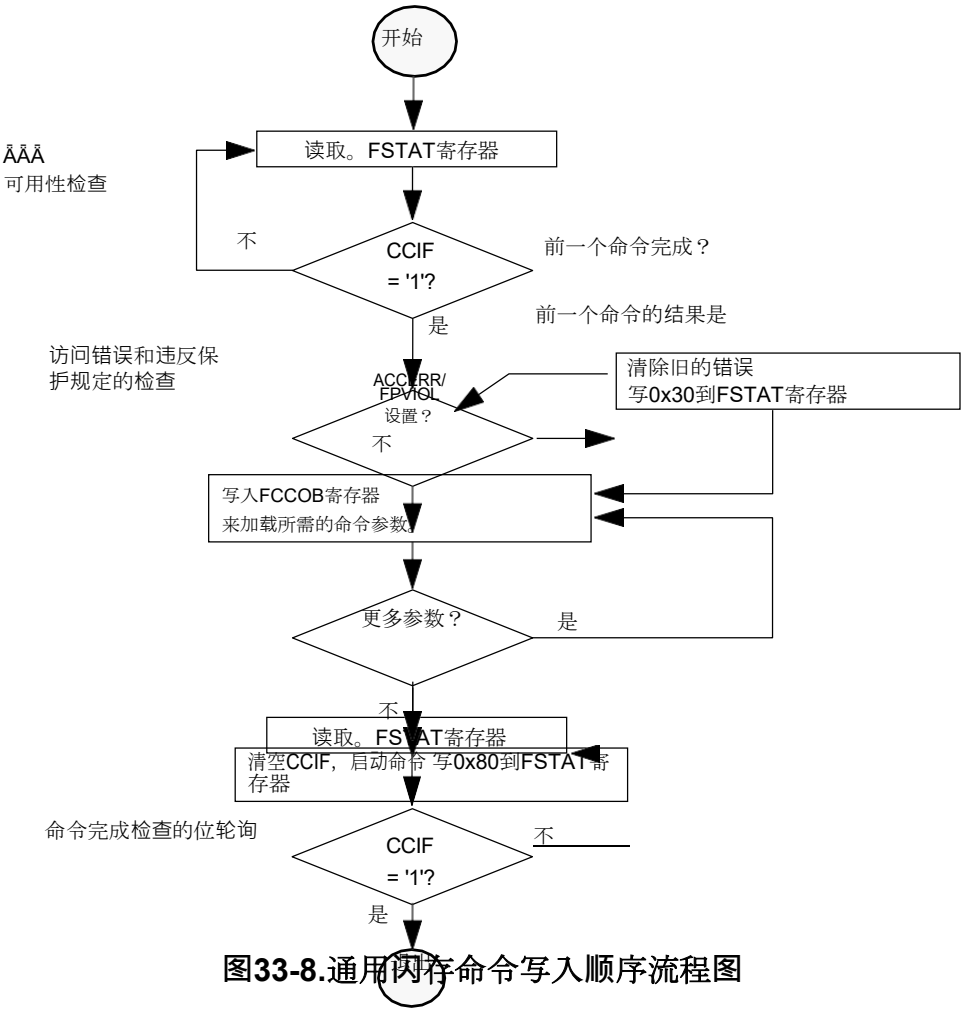



**图33-8.通用闪存命令写入顺序流程图**

## 33.5.9.2 闪存命令

下表总结了所有闪存命令的功能。如果任何一列标有"X"，则该闪存命令与该特定的内存资源有关。

| FCMD | 指挥部 | 程序闪存0 | 弹性RAM | 职能 | |
|---|---|---|---|---|---|
| 0x00 | 读取1s块 | × | × | | 验证一个程序闪存或数据闪存块是否被擦除。FlexNVM |

*表继续在下一页...*

| FCMD | 指挥部 | 程序闪存0 | 弹性RAM | 职能 | |
|---|---|---|---|---|---|
| | | | | | 块不能是仿真EEPROM的分区。 |
| 0x01 | 阅读1部分 | × | × | | 验证从起始地址开始的一定数量的程序闪存或数据闪存位置被擦除。 |
| 0x02 | 程序检查 | × | × | | 测试先前编程的短语，在边缘阅读水平。 |
| 0x03 | 保留 | X | X | | 保留 |
| 0x07 | 程序短语 | × | × | | 在一个程序闪存块或一个数据闪存块中编程8个字节。 |
| 0x08 | 擦除闪存块 | × | × | | 擦除一个程序闪存块或数据闪存块。只有在不受保护的情况下才可以擦除任何闪存块。FlexNVM块不得为仿真EEPROM进行分区。 |
| 0x09 | 擦除闪存区 | × | × | | 擦除一个程序闪存或数据闪存扇区的所有字节。 |
| 0x0B | 计划科 | × | × | × | 将部分程序缓冲区的数据编程到程序闪存或数据闪存块。 |
| 0x40 | 读取1的所有区块 | × | × | | 确认所有的程序闪存、数据闪存块、仿真EEPROM备份数据记录和数据闪存IFR都被擦除，然后释放MCU的安全性。 |
| 0x41 | 阅读一次 | IFR | | | 读取程序闪存0 IFR中一个64字节的专用字段的8个字节。 |

表继续在下一页...

表继续在下一页...

| FCMD | 指挥部 | 程序闪存0 | 弹性RAM | 职能 | |
|---|---|---|---|---|---|
| 0x43 | 计划一次 | IFR | | | 对程序闪存0 IFR中一个64字节的专用字段的8字节进行一次性编程。 |
| 0x44 | 擦除所有区块 | × | × | × | 擦除所有的程序闪存块、数据闪存块、FlexRAM、仿真EEPROM备份数据记录和数据闪存IFR。然后，验证-擦除并释放MCU安全。<br>**注意：**只有当所有内存位置都不受保护时，才有可能进行擦除。 |
| 0x45 | 验证后门访问密钥 | × | | | 在比较了一组用户提供的安全密钥和存储在程序闪存中的安全密钥后，释放MCU的安全性。 |
| 0x49 | 擦除所有块状物 不安全 | × | × | × | 擦除所有程序闪存块、数据闪存块、FlexRAM、仿真EEPROM备份数据记录和数据闪存IFR。然后，验证-擦除，将安全字节编程到不安全状态，并释放MCU安全。 |
| 0x80 | 程序分区 | | IFR, × | × | 将FlexNVM分区代码和仿真EEPROM数据集大小编程到数据闪存IFR中。格式化所有仿真EEPROM备份数据扇区 |

d.

| FCMD | 指挥部 | 程序闪存0 | 弹性RAM | 职能 | |
|---|---|---|---|---|---|
| | | | | | 分配给仿真的EEPROM，初始化FlexRAM。 |
| 0x81 | 设置FlexRAM功能 | | × | × | 在RAM和模拟EEPROM之间切换FlexRAM功能。当切换到模拟的EEPROM。当有效的数据记录从模拟的EEPROM备份复制到FlexRAM时，FlexNVM不可用。 |

## 33.5.9.3 按模式划分的闪光命令

下表显示了在每种闪存操作模式下可以执行的闪存命令。

**表33-5.按模式划分的Flash命令**

| FCMD | 指挥部 | NVM正常 | | | NVM专项 | | |
|---|---|---|---|---|---|---|---|
| | | 不安全 | 安全 | MEEN=10 | 不安全 | 安全 | MEEN=10 |
| 0x00 | 读取1s块 | × | × | × | × | - | - |
| 0x01 | 阅读1部分 | × | × | × | × | - | - |
| 0x02 | 程序检查 | × | × | × | × | - | - |
| 0x03 | 保留 | × | × | × | × | - | - |
| 0x07 | 程序短语 | × | × | × | × | - | - |
| 0x08 | 擦除闪存块 | × | × | × | × | - | - |
| 0x09 | 擦除闪存区 | × | × | × | × | - | - |
| 0x0B | 计划科 | × | × | × | × | - | - |
| 0x40 | 读取1的所有区块 | × | × | × | × | × | - |
| 0x41 | 阅读一次 | × | × | × | × | - | - |
| 0x43 | 计划一次 | × | × | × | × | - | - |
| 0x44 | 擦除所有区块 | × | × | × | × | × | - |
| 0x45 | 验证后门访问密钥 | × | × | × | × | - | - |
| 0x49 | 擦除所有块状物 不安全 | × | × | - | × | × | - |
| 0x80 | 程序分区 | × | × | × | × | - | - |
| 0x81 | 设置FlexRAM功能 | × | × | × | × | - | - |

表继续在下一页...

### 33.5.9.4 允许同时进行闪光操作

只有下表中标记为 "OK
"的操作才允许在程序闪存、数据闪存和FlexRAM存储器上同时运行。有些操作不能同时执行，因为某些硬件资源是由存储器共享的。在FlexNVM和FlexRAM上执行编程和擦除操作时，优先允许程序闪存的读取。这就提供了读（程序闪存）而写（FlexNVM、FlexRAM）的功能。一般来说，这些CCOB命令不能与任何CSEc命令同时执行，反之亦然。详见CSEc章节。

**表33-6.允许的同时进行的内存操作**

| | | 程序闪光 | | | 数据闪光 | | | 弹性RAM | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | 阅读 | 程序短语 | 擦除 Flash Sector 1 | 阅读 | 程序短语 | 擦除 Flash Sector 1 | 阅读 | 电子笔2 | R-Write3 |
| 程序闪光 | 阅读 | | | | | 认可 | 认可 | | 认可 | |
| | 程序短语 | | | | 认可 | | | 认可 | | 认可 |
| | 擦除 Flash Sector 1 | | | | 认可 | | | 认可 | | 认可 |
| 数据闪光 | 阅读 | | 认可 | 认可 | | | | | | |
| | 程序短语 | 认可 | | | | | | 认可 | | 认可 |
| | 擦除 Flash Sector 1 | 认可 | | | | | | 认可 | | 认可 |
| 弹性RAM | 阅读 | | 认可 | 认可 | | 认可 | 认可 | | | |
| | 电子笔2 | 认可 | | | | | | | | |
| | R-Write3 | | 认可 | 认可 | | 认可 | 认可 | | | |

1. 也适用于擦除闪存块
2. 当FlexRAM配置为仿真EEPROM（EEERDY=1）时。
3. 当FlexRAM配置为传统RAM（RAMRDY=1）时；单周期操作。

## 33.5.10 保证金读取命令

Read-1s命令（Read 1s All Blocks, Read 1s Block, Read 1s Section）和Program Check命令有一个余量选择参数，允许用户对这些命令进行的程序闪存和数据闪存阵列的读取应用非标准的读取参考水平。使用预设的 "用户 "和 "工厂 "余量水平，这些命令以比 "正常 "更严格的公差执行相关的读操作。

读取。这些非标准的读取水平只在命令执行期间应用。所有简单的（无命令的）闪存阵列对MCU的读取总是使用标准的、无边际的读取参考电平。

在正常的闪存使用过程中，只应采用 "正常 "的读取水平。非标准、"用户 "和 "工厂
"的余量水平应该只在特殊情况下使用。它们可以在特殊的诊断程序中使用，以获得对设备没有遭受闪存设备常见的寿命结束时的数据损失的信心。

擦除（'1'）和编程（'0'）的位状态会因为时间的推移和数据的循环（一个位被擦除和重新编程的次数）而退化。擦除状态的寿命是相对于最后一次擦除操作而言的。编程状态的寿命是从最后一次编程时间开始测量的。

用户 "和 "工厂 "水平实际上成为一个最小的余量，即，如果读数在 "用户 "和 "工厂 "余量的较小公差下通过，那么 "正常 "读数在经历数据损失之前至少有这么大的余量。

用户 "余量是正常读取参考电平的一个小的delta。用户
"余量水平可以用来检查闪存内容是否有足够的余量用于正常水平的读取操作。如果在'用户'余量水平上检查闪存内容时遇到了意外的读取结果，那么在'正常'读出时可能很快就会发生信息丢失。

出厂
"余量是对标准的更大偏离，是更严格的读取标准，只应在完成擦除或编程命令后立即（或很快）尝试，在循环使用的早期。工厂
"的余量水平可以用来检查闪存内容在正常水平设置下是否有足够的余量用于长期的数据保留。如果在'工厂'余量水平下检查闪存内容时，遇到了意外的结果，应该对闪存内容进行擦除和重新编程。

**注意事项**

工厂余量水平必须只在验证初始工厂编程时使用。

## 33.5.11 Flash命令描述

本节介绍了所有可以通过命令写入序列启动的闪存命令。如果出现以下非法情况，FTFC会设置FSTAT[ACCERR]位并中止命令的执行。

- 在FCCOB FCMD字段中有一个未被识别的命令代码。
- 在特定命令的FCCOB字段中存在错误。请参考为每个命令提供的错误处理表。

确保FSTAT寄存器中的ACCERR和FPVIOL位在启动命令写入序列前被清除。如清空CCIF启动命令中所述，当这些错误标志被设置时，不能启动新的命令。

当FTFC在同一块上运行命令(CCIF=0)时，不要试图读取闪存块。FTFC可能会向MCU返回无效数据，并设置碰撞错误标志（FSTAT[RDCOLERR]）。

当命令要求时，地址位23在程序闪存（=0）和数据闪存（=1）之间进行选择。

**注意事项**

闪存数据在被编程前必须处于擦除状态。不允许对比特进行累积编程（添加更多的零）。

### 33.5.11.1 读取1s区块指令

读取1s块命令检查整个程序闪存或数据闪存块是否已经被擦除到指定的余量水平。FCCOB闪存地址位决定哪个块被擦除验证。

**表33-7.读取1s块命令FCCOB要求**

| FCCOB号码 | FCCOB内容 [7:0] |
|---|---|
| 0 | 0x00 (RD1BLK) |
| 1 | 要验证的闪存块中的闪存地址[23:16]。 |
| 2 | 要验证的闪存块中的闪存地址[15:8]。 |
| 3 | 要验证的闪存块中的闪存地址[7:0][1] |
| 4 | 读取-1保证金选择 |

1.对于程序闪存必须是128位对齐（闪存地址[3:0]=0000），对于数据闪存必须是64位对齐（闪存地址[2:0]=000）。

在清空CCIF启动读1s块命令后，FTFC根据表33-8设置1s的读取余量，然后读取所选程序闪存或数据闪存块中的所有位置。

当以数据闪存为目标时，DEPART必须设置为无EEPROM，否则读取1s块的命令会中止设置FSTAT[ACCERR]位。如果FTFC未能读取所有的1（即闪存块没有被完全擦除），FSTAT[MGSTAT0]位被设置。在读1s块操作完成后，CCIF标志被设置。

**表33-8.读取1s块的保证金水平选择**

| 阅读保证金的选择 | 保证金水平描述 |
|---|---|
| 0x00 | 使用 "正常 "读取水平为1s |
| 0x01 | 将 "用户 "保证金应用于正常的读-1水平 |
| 0x02 | 将 "工厂 "保证金应用于正常的读-1水平 |

**表33-9.读取1s块命令的错误处理**

| 错误情况 | 错误位 |
|---|---|
| 在当前模式/安全中无法使用的命令 | FSTAT[ACCERR] |
| 指定了一个无效的保证金选择 | FSTAT[ACCERR] |
| 选择了程序闪存，而地址超出了程序闪存的范围 | FSTAT[ACCERR] |
| 选择了数据闪存，而地址超出了数据闪存的范围 | FSTAT[ACCERR] |
| 在启用模拟EEPROM的情况下选择数据闪存 | FSTAT[ACCERR] |
| 闪存地址对于程序闪存来说不是128位对齐的，对于数据闪存来说是64位对齐的 | FSTAT[ACCERR] |
| 读取-1s失败 | FSTAT[MGSTAT0] |

## 33.5.11.2 读取1s部分命令

读取1s段命令检查程序闪存或数据闪存的某一段是否被擦除到指定的读取余量水平。读取1s段命令定义了起始地址，对于程序闪存来说，需要验证的双字节数，对于数据闪存来说，需要验证的短语数。

**表33-10.读取1s段命令FCCOB的要求**

| FCCOB号码 | FCCOB内容 [7:0] |
|---|---|
| 0 | 0x01 (RD1SEC) |
| 1 | 对于程序闪存，要验证的第一个双短语的闪存地址[23:16]，对于数据闪存，短语要验证 |
| 2 | 对于程序闪存，要验证的第一个双短语的闪存地址[15:8]，对于数据闪存，短语要验证 |
| 3 | 闪存地址[7:0][1]，对于程序闪存要验证的第一个双字节，对于数据闪存要验证的短语 |

| | |
|---|---|
| 4 | 程序闪存需要验证的双字节数，数据闪存需要验证的短语数 [15:8] 。 |

*表继续在下一页...*

表33-10.读取1s段命令FCCOB的要求（续）

| FCCOB号码 | FCCOB内容 [7:0] |
|---|---|
| 5 | 程序闪存需要验证的双字节数，数据闪存需要验证的短语数[7:0]。 |
| 6 | 读取-1保证金选择 |

1.对于程序闪存必须是128位对齐（闪存地址[3:0]=0000），对于数据闪存必须是64位对齐（闪存地址[2:0]=000）。

在清空CCIF启动读1s段命令后，FTFC根据表33-11设置1s的读取余量，然后读取闪存指定区域内的所有位置。

如果FTFC未能读取所有的1（即闪存部分未被擦除），FSTAT(MGSTAT0)位被设置。在读1s部分操作完成后，CCIF标志被设置。

表33-11.读取1的保证金水平选择部分

| 阅读保证金的选择 | 保证金水平描述 |
|---|---|
| 0x00 | 使用 "正常 "读取水平为1s |
| 0x01 | 将 "用户 "保证金应用于正常的读-1水平 |
| 0x02 | 将 "工厂 "保证金应用于正常的读-1水平 |

表33-12.读取1s部分命令的错误处理

| 错误情况 | 错误位 |
|---|---|
| 在当前模式/安全中无法使用的命令 | FSTAT[ACCERR] |
| 提供了一个无效的保证金代码 | FSTAT[ACCERR] |
| 提供了一个无效的闪存地址 | FSTAT[ACCERR] |
| 闪存地址对于程序闪存来说不是128位对齐的，对于数据闪存来说是64位对齐的 | FSTAT[ACCERR] |
| 要求的部分跨越了闪存块的边界 | FSTAT[ACCERR] |
| 要求的程序闪存的双短语数量，数据闪存的短语数量为零 | FSTAT[ACCERR] |
| 读取-1s失败 | FSTAT[MGSTAT0] |

### 33.5.11.3 程序检查命令

程序检查命令测试一个先前编程的闪存或数据闪存长字，看它在指定的余量水平下是否能正确读取。

表33-13.程序检查命令FCCOB的要求

| FCCOB号码 | FCCOB内容 [7:0] |
|---|---|
| 0 | 0x02 (PGMCHK) |
| 1 | 闪存地址[23:16] |
| 2 | 闪存地址[15:8] |
| 3 | 闪存地址[7:0][1] |
| 4 | 保证金选择 |
| 8 | 字节0预期数据 |
| 9 | 字节1预期数据 |
| A | 字节2预期数据 |
| B | 字节3预期数据 |

1.必须是长字对齐（闪存地址[1:0]=00）。

在清空CCIF以启动程序检查命令时，FTFC根据表33-14提供的余量选择，设置读余量为1s。然后，程序检查操作读取指定的长字，并将实际读取的数据与FCCOB提供的预期数据进行比较。如果在margin-1处比较失败，MGSTAT0位被设置。

然后，FTFC将根据所提供的余量选择，为0s设置读取余量。然后，程序检查操作将读取指定的长字，并将实际读取的数据与FCCOB提供的预期数据进行比较。如果在margin-0处比较失败，MGSTAT0位将被设置。在程序检查操作完成后，CCIF标志将被设置。

起始地址必须是长字对齐的（字节地址的最低两个位必须是00）。

- 字节0的数据预计在提供的32位对齐的地址。
- 字节1数据预计在指定的字节地址+0b01。
- 字节2的数据预计在指定的字节地址+0b10，并且
- 字节3数据预计在指定的字节地址+0b11。

### 注意事项

见保证金读取的描述，保证金读取命令

表33-14.程序检查的保证金级别选择

| 阅读保证金的选择 | 保证金水平描述 |
|---|---|
| 0x01 | 在 "用户 "边际-1和 "用户 "边际-0处读取 |
| 0x02 | 在 "工厂 "边距-1和 "工厂 "边距-0处读取 |

表33-15.程序检查命令的错误处理

| 错误情况 | 错误位 |
|---|---|
| 在当前模式/安全中无法使用的命令 | FSTAT[ACCERR] |
| 提供了一个无效的闪存地址 | FSTAT[ACCERR] |
| 闪存地址不是长字对齐的 | FSTAT[ACCERR] |
| 提供了一个无效的保证金选择 | FSTAT[ACCERR] |
| 任意一个保证金读数与预期数据不一致 | FSTAT[MGSTAT0] |

### 33.5.11.4 程序短语命令

编程短语命令使用一种嵌入式算法对程序闪存或数据闪存中的八个先前删除的字节进行编程。

**注意事项**

一个闪存位置在被编程之前必须处于擦除状态。不允许在一个闪存位置上进行比特的累积编程（在没有擦除的情况下进行背靠背的编程操作）。不允许将现有的0重新编程为0，因为这样做会对器件造成过大的压力。

表33-16.程序短语命令FCCOB要求

| FCCOB号码 | FCCOB内容 [7:0] |
|---|---|
| 0 | 0x07 (PGM8) |
| 1 | 闪存地址[23:16] |
| 2 | 闪存地址[15:8] |
| 3 | 闪存地址[7:0][1] |
| 4 | 字节0程序值 |
| 5 | 字节1程序值 |
| 6 | 字节2程序值 |
| 7 | 字节3程序值 |
| 8 | 字节4程序值 |
| 9 | 字节5 程序值 |
| A | 字节6 程序值 |
| B | 字节7 程序值 |

1.必须是64位对齐（闪存地址[2:0] = 000）。

在清空CCIF以启动编程短语命令后，FTFC使用提供的地址将数据字节编程到闪存中。保护状态总是被检查的。目标闪存位置必须是当前未受保护的（参见FPRT寄存器的描述），以允许执行编程短语操作。

编程操作是单向的。它只能将NVM位从擦除状态（'1'）移到编程状态（'0'）。未能编程到'0'状态的擦除位在MGSTAT0中被标记为错误。CCIF标志在编程短语操作完成后被设置。

起始地址必须是64位对齐的（闪存地址[2:0] = 000）。

- 字节0的数据被写到起始地址（'start'）。
- 字节1的数据被编程到字节地址start+0b01。
- 字节2的数据被编程到字节地址start+0b10，并且
- 字节3的数据被编入字节地址start+0b11，等等。

**表33-17.程序短语命令错误处理**

| 错误情况 | 错误位 |
|---|---|
| 在当前模式/安全中无法使用的命令 | FSTAT[ACCERR] |
| 提供了一个无效的闪存地址 | FSTAT[ACCERR] |
| 闪存地址不是64位对齐的 | FSTAT[ACCERR] |
| Flash地址指向一个保护区 | FSTAT[FPVIOL] |
| 在验证操作中遇到了任何错误。 | FSTAT[MGSTAT0] |

### 33.5.11.5 擦除闪存块命令

擦除闪存块操作可以擦除单个编程闪存或数据闪存块中的所有地址。

**表33-18.擦除闪存块命令FCCOB的要求**

| FCCOB号码 | FCCOB内容 [7:0] |
|---|---|
| 0 | 0x08 (ERSBLK) |
| 1 | 要擦除的闪存块中的闪存地址[23:16]。 |
| 2 | 要擦除的闪存块中的闪存地址[15:8]。 |
| 3 | 要擦除的闪存块中的闪存地址[7:0][1] |

1.对于程序闪存必须是128位对齐（闪存地址[3:0]=0000），对于数据闪存必须是64位对齐（闪存地址[2:0]=000）。

在清空CCIF以启动Erase Flash Block命令时，FTFC会擦除所选Flash块的主阵列并验证其是否被擦除。当以数据闪存为目标时，DEPART必须设置为无EEPROM（见FlexNVM分区代码），否则Erase Flash Block命令会中止，设置FSTAT[ACCERR]位。如果闪存块内的任何区域被保护，擦除闪存块命令就会中止，并设置FSTAT[FPVIOL]位（参见程序闪存保护（FPRT

）寄存器的描述和

数据闪存保护（FDPROT）寄存器）。如果擦除验证失败，FSTAT中的MGSTAT0位被设置。在擦除闪存块操作完成后，CCIF标志将被设置。

**注意事项**

对于启用了CSEc的部件，任何（1个或多个）用户密钥的存在将阻止该命令的执行。请参考命令CMD_DBG_CHAL和CMD_DBG_AUTH，以了解通过挑战/认证过程的程序，这将首先删除密钥，然后（假设成功执行）允许执行此命令。

**表33-19.擦除闪存块命令的错误处理**

| 错误情况 | 错误位 |
|---|---|
| 在当前模式/安全中无法使用的命令 | FSTAT[ACCERR] |
| 选择了程序闪存，而地址超出了程序闪存的范围 | FSTAT[ACCERR] |
| 选择了数据闪存，而地址超出了数据闪存的范围 | FSTAT[ACCERR] |
| 在启用模拟EEPROM的情况下选择数据闪存 | FSTAT[ACCERR] |
| 闪存地址对于程序闪存来说不是128位对齐的，对于数据闪存来说是64位对齐的 | FSTAT[ACCERR] |
| 所选闪存块的任何区域都受到保护 | FSTAT[FPVIOL] |
| 在验证操作中遇到了任何错误1 | FSTAT[MGSTAT0] |

1.可以用读1s块的命令运行用户余量读，以验证所有的位都被擦除。

## 33.5.11.6 擦除闪存扇区命令

擦除闪存扇区的操作是擦除闪存扇区的所有地址。

**表33-20.擦除闪存扇区命令FCCOB要求**

| FCCOB号码 | FCCOB内容 [7:0] |
|---|---|
| 0 | 0x09 (ERSSCR) |
| 1 | 要擦除的闪存扇区的闪存地址[23:16]。 |
| 2 | 要擦除的闪存扇区的闪存地址[15:8]。 |
| 3 | 要擦除的闪存扇区的闪存地址[7:0][1] |

1.对于程序闪存必须是128位对齐（闪存地址[3:0]=0000），对于数据闪存必须是64位对齐（闪存地址[2:0]=000）。

在清空CCIF以启动擦除闪存扇区命令后，FTFC会擦除所选的程序闪存或数据闪存扇区，然后验证其是否被擦除。如果所选扇区受到保护，擦除闪存扇区的命令就会中止（见 "保护 "的描述）。

FPROT寄存器）。如果擦除验证失败，FSTAT[MGSTAT0]位被设置。在Erase Flash
Sector操作完成后，CCIF标志被设置。擦除闪存扇区的命令是可以暂停的（见FCNFG[ERSSUSP]位和图33-9）。

**表33-21.擦除闪存扇区命令的错误处理**

| 错误情况 | 错误位 |
|---|---|
| 在当前模式/安全中无法使用的命令 | FSTAT[ACCERR] |
| 提供了一个无效的Flash地址 | FSTAT[ACCERR] |
| 闪存地址对于程序闪存来说不是128位对齐的，对于数据闪存来说是64位对齐的 | FSTAT[ACCERR] |
| 所选的程序闪存或数据闪存扇区受到保护 | FSTAT[FPVIOL] |
| 在验证操作中遇到了任何错误1 | FSTAT[MGSTAT0] |

1.用户余量读取可以使用读取1s部分的命令来验证所有的位都被擦除。

#### 33.5.11.6.1 暂停擦除闪存区的操作

当CCIF、ACCERR和FPVIOL清零，并且CCOB命令字段包含擦除闪存扇区命令的代码时，要暂停擦除闪存扇区的操作，设置FCNFG[ERSSUSP]位。在擦除闪存扇区的操作中（见擦除闪存扇区命令），闪存会在方便的时候对ERSSUSP位的状态进行采样。如果FTFC检测到ERSSUSP位被设置，擦除闪存扇区的操作就会暂停，FTFC会设置CCIF。当ERSSUSP被设置时，除了对FSTAT和FCNFG寄存器的写入，所有对闪存寄存器的写入都被忽略。

如果在FTFC检测到有暂停请求之前，擦除闪存扇区操作有效完成，FTFC在设置CCIF之前清除ERSSUSP位。当擦除闪存扇区操作成功暂停后，FTFC设置CCIF并保持ERSSUSP位的设置。当CCIF被设置时，ERSSUSP位只能被清零，以防止在FTFC确认之前撤回暂停请求。

#### 33.5.11.6.2 恢复暂停的擦除闪存扇区操作

如果在清空CCIF启动下一条命令时，ERSSUSP位仍然被设置，那么之前的擦除闪存扇区操作将恢复。FTFC通过清除ERSSUSP位来确认恢复暂停操作的请求。然后可以通过设置ERSSUSP来提出新的暂停请求。一个单一的擦除闪存扇区操作可以被暂停和恢复多次。

在请求恢复擦除闪存扇区操作（CCIF被清除）和请求再次暂停操作（ERSSUSP被设置）之间有一个最小的时间限制，即4.3毫秒。这个最短的时间段是为了确保擦

除闪存扇区的操作最终完成。如果这个最短的时间段是持续的

违反，即暂停请求反复出现且速度过快，擦除闪存扇区的算法就不会取得任何进展。恢复/暂停序列无限期地运行，而没有完成擦除。

#### 33.5.11.6.3 中止暂停的擦除闪存扇区操作

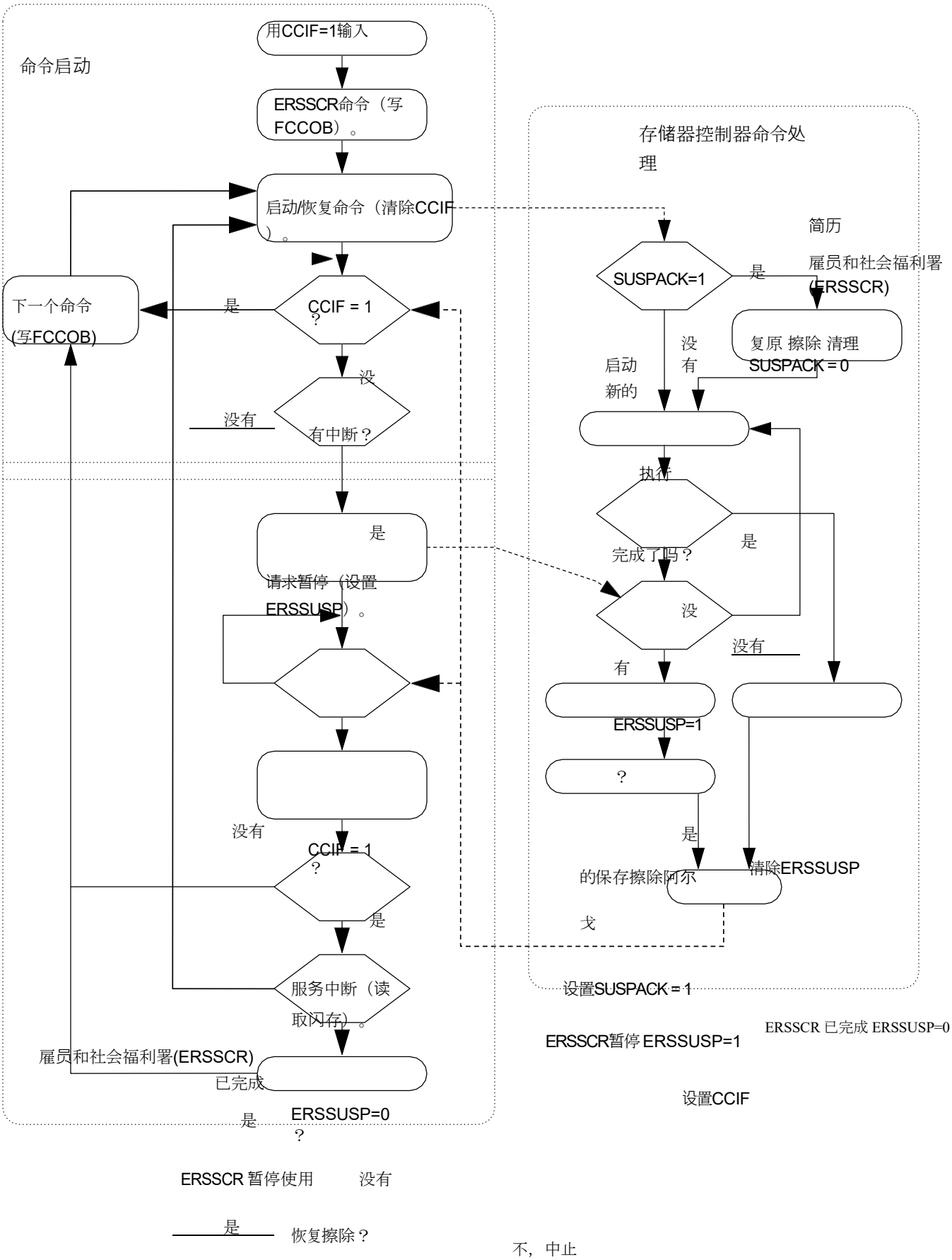
用户可以选择在为下一个命令启动清空CCIF之前，通过清空ERSSUSP位来中止暂停的擦除Flash扇区操作。当暂停的操作被中止时，FTFC会使用新的FCCOB内容启动新的命令。

当FCNFG[ERSSUSP]被设置时，当FCNFG[EEERDY]被设置时，对FlexRAM的写操作将清除ERSSUSP并中止暂停的操作。FlexRAM的写操作由FTFC执行。

**注意事项**

终止擦除会使位元组处于不确定的、部分擦除的状态。在一个新的擦除命令完全完成之前，这个扇区的数据是不可靠的。

下图显示了如何暂停和恢复Erase Flash Sector操作。


命令启动
用CCIF=1输入
ERSSCR命令（写FCCOB）。
启动/恢复命令（清除CCIF）。
下一个命令(写FCCOB)
是
CCIF = 1 ?
没有
没有中断？
是
请求暂停（设置ERSSUSP）。
没有
CCIF = 1 ?
是
服务中断（读取闪存）。
雇员和社会福利署(ERSSCR)已完成
是
ERSSUSP=0 ?
ERSSCR 暂停使用
是
恢复擦除？
存储器控制器命令处理
SUSPACK=1
是
简历雇员和社会福利署(ERSSCR)
没有
启动新的
复原 擦除 清理
SUSPACK = 0
执行
完成了吗？
是
没有
ERSSUSP=1
?
是
的保存擦除阿尔戈
清除ERSSUSP
设置SUSPACK = 1
ERSSCR暂停 ERSSUSP=1
ERSSCR 已完成 ERSSUSP=0
设置CCIF
不，中止

ERSUSP。FCNFG寄存器中的位SUSPAGR：中断暂停确认

清除ERSSUSP

用户指令中断/暂停

**图33-9.暂停和恢复擦除Flash扇区的操作**

### 33.5.11.7 程序部分命令

编程部分操作使用一种嵌入式算法将在部分程序缓冲器中找到的数据编程到闪存中先前被清除的位置。当FlexRAM被设置为编程加速RAM的功能时，数据被预装到部分程序缓冲器中（见闪存扇区编程）。

部分程序缓冲区被限制在编程加速RAM的下四分之一（字节地址0x0000-0x<1/4 * FlexRAM大小>）。写到编程加速RAM其余部分的数据被忽略，在执行编程部分命令时可能被覆盖。

**注意事项**

闪存位置在被编程之前必须处于擦除状态。不允许在闪存位置内进行比特的累积编程（在没有擦除的情况下进行背对背的编程操作）。不允许将现有的0重新编程为0，因为这样做会对器件造成过大压力。

**表33-22.程序部分命令FCCOB要求**

| FCCOB号码 | FCCOB内容 [7:0] |
|---|---|
| 0 | 0x0B (PGMSEC) |
| 1 | 闪存地址[23:16] |
| 2 | 闪存地址[15:8] |
| 3 | 闪存地址[7:0][1] |
| 4 | 双重短语（用于交错块）、短语（用于非交错块）的编程数量[15:8]。 |
| 5 | 要编程的双语句（用于交错块）、语句（用于非交错块）的数目 [7:0] |

1.对于程序闪存必须是128位对齐（闪存地址[3:0]=0000），对于数据闪存必须是64位对齐（闪存地址[2:0]=000）。

在清空CCIF启动Program Section命令后，FTFC将阻止对FlexRAM的访问，并将驻留在Section Program Buffer中的数据从提供的Flash地址开始编程到Flash存储器中。

起始地址必须是不受保护的（见FPRT寄存器的描述），以允许执行编程部分的操作。编程是不允许跨越闪存扇区边界的，一直到所有要求的双短语（对于交错块）和短语（对于非交错块）都被编程完毕。

程序部分操作完成后，CCIF标志将被设置，对FlexRAM的正常访问被恢复。部分程序缓冲区的内容不会因为程序部分操作而改变。

表33-23.程序节命令错误处理

| 错误情况 | 错误位 |
| --- | --- |
| 在当前模式/安全中无法使用的命令 | FSTAT[ACCERR] |
| 提供了一个无效的闪存地址 | FSTAT[ACCERR] |
| 闪存地址对于程序闪存来说不是128位对齐的，对于数据闪存来说是64位对齐的 | FSTAT[ACCERR] |
| 要求的部分跨越了程序闪存扇区的边界 | FSTAT[ACCERR] |
| 要求的程序闪存的双短语数量，数据闪存的短语数量为零 | FSTAT[ACCERR] |
| 对于程序闪存来说，存储所要求的双语句数量的数据所需的空间，对于数据闪存来说，短语是超过四分之一的FlexRAM的大小 | FSTAT[ACCERR] |
| FlexRAM没有被设置为传统RAM的功能，即如果RAMRDY=0，则设置为 | FSTAT[ACCERR] |
| 闪存地址属于受保护区域 | FSTAT[FPVIOL] |
| 在验证操作中遇到了任何错误 | FSTAT[MGSTAT0] |

#### 33.5.11.7.1 闪存扇区编程

使用Program Section命令对整个闪存扇区进行编程的过程如下。

1. 如果需要，执行Set FlexRAM Function命令，使FlexRAM作为传统RAM可用，并将FlexRAM初始化为所有的1。
2. 启动Erase Flash Sector命令，擦除要编程的Flash扇区。
3. 从FlexRAM的起始地址开始，按顺序向RAM写入足够的数据，以填满整个闪存扇区。RAM的这一区域作为部分程序缓冲区。

**注意事项**

在步骤1中，部分程序缓冲区被初始化为所有的1，即闪存的擦除状态。

在步骤2中启动的操作正在执行时，即在CCIF=0时，可以写入该段程序缓冲区。

4. 执行Program Section命令，将部分程序缓冲区的内容编入选定的Flash扇区。
5. 如果一个Flash扇区大于FlexRAM的一半，重复步骤3和4，直到该扇区被完全编程。
6. 要对其他闪光区进行编程，请重复步骤2至4。
7. 为了恢复仿真EEPROM的功能，执行Set FlexRAM Function命令，使FlexRAM可用于仿真EEPROM。

### 33.5.11.8 读取1s所有区块的命令

读取1s所有区块命令检查程序闪存区块、数据闪存区块、模拟EEPROM备份记录和数据闪存IFR是否已经被擦除到指定的读取余量水平（如果适用），如果读出通过，即所有数据读为 "1"，则释放安全。

**表33-24.读取1s所有块的命令FCCOB要求**

| FCCOB号码 | FCCOB内容 [7:0] |
|---|---|
| 0 | 0x40 (RD1ALL) |
| 1 | 读取-1保证金选择 |

在清空CCIF启动读1s所有区块命令后，FTFC。

- 根据表33-25设置1s的读取余量。
- 检查程序闪存、数据闪存、仿真EEPROM备份记录和数据闪存IFR的内容是否处于擦除状态。

如果FTFC确认这些内存资源被擦除，则通过将FSEC[SEC]字段设置为不安全状态来释放安全。闪存配置字段中的安全字节（见闪存配置字段描述）仍然不受读取1s所有区块命令的影响。如果读取失败，即所有闪存资源没有处于完全擦除状态，FSTAT[MGSTAT0]位被设置。

EEERDY和RAMRDY位在读1s所有块操作期间被清除，并在读1s所有块操作结束后被恢复。

CCIF标志在读1s所有区块操作完成后设置。

**表33-25.读取1s所有区块的保证金水平选择**

| 阅读保证金的选择 | 保证金水平描述 |
|---|---|
| 0x00 | 使用 "正常 "读取水平为1s |
| 0x01 | 将 "用户 "保证金应用于正常的读-1水平 |
| 0x02 | 将 "工厂 "保证金应用于正常的读-1水平 |

**表33-26.读取1s所有区块命令的错误处理**

| 错误情况 | 错误位 |
|---|---|
| 指定了一个无效的保证金选择 | FSTAT[ACCERR] |
| 读取-1s失败 | FSTAT[MGSTAT0] |

### 33.5.11.9 读取一次命令

读取一次命令提供了对位于程序闪存IFR中的一个保留的64字节字段的读取权限（参见程序闪存0 IFR图和程序一次字段）。对Program Once字段的访问是通过8条记录进行的，每条记录长8个字节。使用Program Once命令对Program Once字段进行编程，详见Program Once命令。

**表33-27.读一遍命令FCCOB的要求**

| FCCOB号码 | FCCOB内容 [7:0] |
|---|---|
| 0 | 0x41 (RDONCE) |
| 1 | 程序一次记录索引 (0x00 - 0x07 ) |
| 返回的值 | |
| 4 | 程序 一旦字节0值 |
| 5 | 程序 一旦字节1的值 |
| 6 | 程序 一旦字节2的值 |
| 7 | 程序 一旦字节3的值 |
| 8 | 程序 一旦字节4的值 |
| 9 | 程序 一旦字节5的值 |
| A | 程序 一旦字节6的值 |
| B | 程序 一旦字节7的值 |

在清除CCIF以启动Read Once命令后，一个8字节的Program Once记录被从程序闪存IFR中读取并存储在FCCOB寄存器中。在一次读操作完成后，CCIF标志被设置。Read Once命令的有效记录索引值范围是0x00到0x07。在执行一读命令期间，任何试图读取包含这个64字节字段的程序闪存块内的地址都会返回无效的数据。读一遍命令可以被执行任意次数。

**表33-28.一次读命令的错误处理**

| 错误情况 | 错误位 |
|---|---|
| 在当前模式/安全中无法使用的命令 | FSTAT[ACCERR] |
| 提供了一个无效的记录索引 | FSTAT[ACCERR] |

## 33.5.11.10 程序一旦命令

Program Once命令可以对程序闪存IFR中一个保留的64字节的字段进行编程（参见Program flash 0 IFR map and Program Once field）。对Program Once字段的访问是通过8条记录进行的，每条记录长8个字节。Program Once字段可以用Read Once命令来读取（见Read Once命令）。每个Program Once记录只能被编程一次，因为Program flash IFR不能被擦除。

**表33-29.程序一旦命令FCCOB的要求**

| FCCOB号码 | FCCOB内容 [7:0] |
|---|---|
| 0 | 0x43 (PGMONCE) |
| 1 | 程序一次记录索引 (0x00 - 0x07 ) |
| 2 | 未使用 |
| 3 | 未使用 |
| 4 | 程序一次 字节0值 |
| 5 | 程序一次 字节1值 |
| 6 | 程序一次 字节2值 |
| 7 | 程序一次 字节3值 |
| 8 | 程序一次 字节4值 |
| 9 | 程序一次 字节5值 |
| A | 程序一次 字节6值 |
| B | 程序一次 字节7值 |

在清空CCIF以启动一次编程命令后，FTFC首先验证所选的记录是否被擦除。如果被擦除，则使用所提供的值对所选记录进行编程。一次编程命令还验证了编程值是否正确读回。在 "一次编程 "操作完成后，CCIF标志被设置。

由Program Once命令访问的保留的程序闪存IFR位置不能被擦除，当现有值没有被Fs（擦除）时，任何试图对这些记录之一进行编程的行为都是不允许的。Program Once命令的有效记录索引值范围是0x00到0x07。在执行Program Once命令期间，任何试图读取程序闪存内的地址都会返回无效的数据。

**表33-30.程序一旦命令错误处理**

| 错误情况 | 错误位 |
|---|---|
| 在当前模式/安全中无法使用的命令 | FSTAT[ACCERR] |
| 提供了一个无效的记录索引 | FSTAT[ACCERR] |
| 所请求的记录已经被编程为一个非删除值1 | FSTAT[ACCERR] |

| | |
|---|---|
| 在验证操作中遇到了任何错误。 | FSTAT[MGSTAT0] |

1.如果一个Program Once记录最初被编程为0xFFFF_FFFF_FFFF_FFFF，则允许在同一记录上再次执行Program Once命令。

## 33.5.11.11 擦除所有区块命令

擦除所有区块操作擦除所有闪存，初始化FlexRAM，验证所有内存内容，并释放MCU安全。

**表33-31.擦除所有区块命令FCCOB要求**

| FCCOB号码 | FCCOB内容 [7:0] |
|---|---|
| 0 | 0x44 (ERSALL) |

在清空CCIF启动擦除所有区块的命令后，FTFC会擦除所有的程序闪存、数据闪存、数据闪存IFR空间、仿真EEPROM备份存储器和FlexRAM，然后验证是否全部被擦除。

如果FTFC验证了所有的闪存和FlexRAM都被正确地擦除，则通过将FSEC[SEC]字段设置为不安全状态和FCNFG[RAMRDY]位来释放安全。如果任何闪存或FlexRAM区域被保护，擦除所有块的命令就会中止。安全字节和闪存配置字段的所有其他内容（见闪存配置字段描述）都被Erase All Blocks命令擦除。如果擦除验证失败，FSTAT[MGSTAT0]位被设置。在Erase All Blocks操作完成后，CCIF标志被设置。

**表33-32.擦除所有区块命令的错误处理**

| 错误情况 | 错误位 |
|---|---|
| 在当前模式/安全中无法使用的命令 | FSTAT[ACCERR] |
| 程序闪存、数据闪存或FlexRAM的任何区域都受到保护 | FSTAT[FPVIOL] |
| 在验证操作中遇到了任何错误1 | FSTAT[MGSTAT0] |

1.可以用读1s所有区块的命令运行用户余量读，以验证所有的位都被擦除。

### 33.5.11.11.1 触发闪存模块外部的所有擦除工作

擦除所有区块/擦除所有区块不安全命令的功能也可以在闪存外以非命令的方式使用。有关这一功能的信息，请参考器件的芯片配置细节。

在调用外部全部擦除功能之前，FCCOB0寄存器必须不包含0x44。当调用时，全

部擦除功能会擦除所有的程序闪存、数据闪存、数据闪存IFR空间、仿真EEPROM备份和FlexRAM，而不考虑FSTAT[ACCERR和FPVIOL]标志的状态或保护设置。如果

擦除验证通过后，该程序通过将FSEC[SEC]字段寄存器设置为不安全状态和FCNFG[RAMRDY]位设置来释放安全。闪存配置字段中的安全字节也被编程为不安全状态。擦除请求的状态反映在FCNFG[ERSAREQ]位。一旦操作完成，FCNFG[ERSAREQ]位就会被清空，并且正常的FSTAT错误报告（FPVIOL除外）可以使用，如Erase All Blocks命令中所述。

### 33.5.11.12 验证后门访问密钥命令

只有在满足模式和安全条件的情况下，才会执行验证后门访问密钥的命令（见按模式划分的Flash命令）。FSEC[KEYEN]位进一步限定了验证后门访问密钥命令的执行。如果FCCOB中用户提供的密钥与存储在闪存配置字段的后门比较密钥字节中的密钥相匹配，则验证后门访问密钥命令会释放安全。标有Flash Configuration Field偏移地址的一列显示了Flash Configuration Field中匹配字节的位置。

**表33-33.验证后门访问密钥命令FCCOB要求**

| FCCOB号码 | FCCOB内容 [7:0] | 闪存配置字段偏移地址 |
|---|---|---|
| 0 | 0x45 (VFYKEY) | |
| 1-3 | 未使用 | |
| 4 | 关键字节0 | 0x0_0003 |
| 5 | 关键字节1 | 0x0_0002 |
| 6 | 关键字节2 | 0x0_0001 |
| 7 | 关键字节3 | 0x0_0000 |
| 8 | 关键字节4 | 0x0_0007 |
| 9 | 关键字节5 | 0x0_0006 |
| A | 关键字节6 | 0x0_0005 |
| B | 关键字节7 | 0x0_0004 |

在清除CCIF以启动验证后门访问密钥命令后，FTFC检查FSEC[KEYEN]位以验证该命令是否被启用。如果没有启用，FTFC会设置FSTAT[ACCERR]位并终止。如果该命令被启用，FTFC将FCCOB中提供的密钥与Flash配置字段中的后门比较密钥进行比较。如果后门密钥匹配，FSEC[SEC]字段被改变为不安全状态，安全被释放。如果后门钥匙不匹配，安全就不会被释放，所有未来执行验证后门访问钥匙命令的尝试都会立即中止，FSTAT[ACCERR]位（再次）被设置为1，直到重置

FTFC模块发生。如果整个8字节的密钥都是0或1，则验证后门访问密钥命令失败，出现访问错误。在验证后门访问密钥操作完成后，CCIF标志被设置。

**表33-34.验证后门访问密钥命令错误处理**

| 错误情况 | 错误位 |
|---|---|
| 在当前模式/安全中无法使用的命令 | FSTAT[ACCERR] |
| 提供的钥匙是全0或全F的。 | FSTAT[ACCERR] |
| 提供了一个不正确的后门钥匙 | FSTAT[ACCERR] |
| 未启用后门钥匙访问（见FSEC寄存器的描述）。 | FSTAT[ACCERR] |
| 该命令启动后，自上次断电复位后，后门钥匙不匹配。 | FSTAT[ACCERR] |

### 33.5.11.13 擦除所有区块 不安全的命令

擦除所有区块不安全操作擦除所有闪存，初始化FlexRAM，验证所有内存内容，将闪存配置字段中的安全字节编程为不安全状态，并释放MCU安全。

**表33-35.擦除所有区块不安全命令FCCOB要求**

| FCCOB号码 | FCCOB内容 [7:0] |
|---|---|
| 0 | 0x49 (ERSALLU) |

在清除CCIF启动擦除所有区块的不安全命令后，FTFC会擦除所有的程序闪存、数据闪存、数据闪存IFR空间、仿真EEPROM备份内存和FlexRAM，然后验证所有都被擦除。

如果FTFC验证所有闪存和FlexRAM被正确擦除，则通过将FSEC[SEC]字段设置为不安全状态来释放安全，通过Erase All Blocks Unsecure命令将安全字节（见闪存配置字段描述）编程为不安全状态，并且FCNFG[RAMRDY]位被设置。如果擦除或编程验证失败，FSTAT[MGSTAT0]位被设置。在Erase All Blocks Unsecure操作完成后，CCIF标志被设置。

**表33-36.擦除所有区块 不安全命令的错误处理**

| 错误情况 | 错误位 |
|---|---|
| 在当前模式/安全中无法使用的命令 | FSTAT[ACCERR] |
| 在擦除或程序验证操作中遇到了任何错误 | FSTAT[MGSTAT0] |

### 33.5.11.14 程序分区命令

Program
Partition命令准备FlexNVM块作为数据闪存、模拟EEPROM备份或两者的组合使用，并初始化FlexRAM。编程分区命令不能从闪存中启动，因为在编程分区命令执行期间不能访问闪存资源。
与执行Program Partition命令有关的变化在下次复位后生效。

**注意事项**

虽然FlexNVM有不同的分区，但目的是在特定应用的整个生命周期内使用单一的分区选择。FlexNVM分区代码的选择会影响设备的耐用性和数据保留特性。

**表33-37.程序分区命令FCCOB要求**

| FCCOB号码 | FCCOB内容 [7:0] |
|---|---|
| 0 | 0x80 (PGMPART) |
| 1 | CSEc密钥大小 |
| 2 | SFE |
| 3 | 复位期间的FlexRAM加载选项（仅使用位0）。<br>0 - 复位序列中，FlexRAM加载了有效的EEPROM数据<br>1 - 复位程序中未加载FlexRAM |
| 4 | EEPROM数据集大小代码1 |
| 5 | FlexNVM 分区代码2 |

1. 见表5和EEPROM数据集大小
2. 见表6和FlexNVM分区代码

**表33-38.CSEc/SHE用户密钥大小（KEY<n>）。**

| FCCOB1 [1:0] | 用户钥匙的数量（{MASTER_ECU_KEY, BOOT_MAC_KEY, BOOT_MAC, KEY<n>}。 | 字节数（从4K EEERAM总空间中减去）。 |
|---|---|---|
| 2'b00 | 零度 | 0 |
| 2'b01 | 1至5键 | 128 字节 |
| 2'b10 | 1至10键 | 256字节 |
| 2'b11 | 1至20键 | 512 字节 |

对于未启用CSEc的部件，用户钥匙的数量必须配置为2'b00（无钥匙）。

对于启用了 CSEc 的部件，用户键的数量是用户可配置的，但空间将假定 MASTER_ECU_KEY、BOOT_MAC_KEY 和 BOOT_MAC 都在那里（如果有任何键被启用），因此将占用可用的 20 个键槽空间中的 3 个键槽。这就导致了为1到17个用户键<n>键的范围留下了键位。

对于未启用CSEc的部件，键的分配必须设置为零键（2'b00），否则该命令将返回一个错误。

**注意事项**

对于支持CSEc的部件，一旦用户密钥被分配（无论是否被初始化），SHE规范中关于未经认证不得擦除用户密钥的要求将适用。这意味着在擦除DFlash之前，必须运行并通过认证（DBG_CHAL和DBG_AUTH）命令（删除所有的用户钥匙）（所有的用户钥匙都备份在DFlash中的模拟EEPROM中）。因此，擦除所有的区块不会起作用，如果所选的区块/扇区包括储存钥匙的DFlash，擦除Flash区块或扇区也不会起作用。此外，如果任何用户的密钥是受写保护的，它们就不能被擦除/删除，因此DFlash不能被擦除，认证过程也就不能通过。

**表33-39.安全标志扩展（SFE）**

| SFE | 结果 |
|---|---|
| SFE == 0x01 | 启用用户密钥 "仅验证 "属性 |
| SFE == 0x00 | 未启用用户密钥 "仅验证 "属性 |

FCCOB[2]字段用于增强的安全标志扩展（SFE）是在一个8位字段中，但只有位[0]被使用（位[7：1]被忽略）。这个功能是对SHE规范的一个扩展。对于标准的SHE使用模式，用户应该将其配置为 "未启用"（SFE == 0x00）。这个新增功能是为了增加标记一个密钥为 "VERIFY_ONLY "的能力。也就是说，用户可以配置SFE == 0x01，然后使用LOAD_KEY命令给密钥添加一个新的属性，告诉GENERATE_MAC命令该密钥只能由VERIFY_MAC命令使用（通过配置SFE为 "启用"，然后使用LOAD_KEY给该密钥设置新的VERIFY_ONLY属性）。

如果KEY_USAGE属性为'0'，VERIFY_ONLY属性属性就没有作用。关于这个新

属性的更多信息，请参见CMD_LOAD_KEY命令。

## 表33-40.有效的模拟EEPROM数据集大小代码：2kB FlexRAM配置

| EEPROM数据集大小代码（FCCOB4）[1] | EEPROM数据集大小（字节） |
|---|---|
| eesize (fccob4[3:0]) | |
| 0xF | 0[2] |
| 0x3 | 2k |

1. FCCOB4[7:4] = 0000
2. 当FlexNVM分区代码被设置为无EEPROM时，EEPROM数据集大小必须被设置为0字节。

## 表33-41.有效的仿真EEPROM数据集大小代码 :4kB FlexRAM配置

| EEPROM数据集大小代码（FCCOB4）[1] | EEPROM数据集大小（字节） |
|---|---|
| eesize (fccob4[3:0]) | |
| 0xF | 0[2] |
| 0x2 | 4k |

1. FCCOB4[7:4] = 0000
2. 当FlexNVM分区代码被设置为无EEPROM时，EEPROM数据集大小必须被设置为0字节。

## 表33-42.有效的FlexNVM分区代码 :带有32kB FlexNVM的128kB和256kB配置

| FlexNVM分区代码DEPART（FCCOB5[3:0]）[1] | 数据闪存大小（KB） | EEPROM-备份大小(KB) |
|---|---|---|
| 0000 | 32 | 0 |
| 0011 | 0 | 32 |
| 1000 | 0 | 32 |
| 1001 | 8 | 24 |
| 1011 | 32 | 0 |

1. FCCOB5[7:4] = 0000

## 表33-43.有效的FlexNVM分区代码：使用64kB FlexNVM的256KB、512kB和1MB配置

| FlexNVM分区代码DEPART（FCCOB5[3:0]）[1] | 数据闪存大小（KB） | EEPROM-备份大小(KB) |
|---|---|---|
| 0000 | 64 | 0 |
| 0011 | 32 | 32 |
| 0100 | 0 | 64 |
| 1000 | 0 | 64 |
| 1010 | 16 | 48 |
| 1011 | 32 | 32 |
| 1100 | 64 | 0 |

1. FCCOB5[7:4] = 0000

## 注意事项

对于2M配置（DFlash在多个块中实现），FlexNVM只能被配置为：所有的数据闪存，或者64k字节的仿真EEPROM备份，其余部分作为数据闪存。因此，DEPART（FCCOB5[3:0]）只有三个有效值：4'b0100（64kB仿真EEPROM，剩余部分为DFlash）或4'b0000，4'b1111（0kB仿真EEPROM/全部DFlash）。

**表33-44.有效的FlexNVM分区代码：2M配置**

| FlexNVM分区代码DEPART（FCCOB5[3:0]）[1] | 数据闪存大小（KB） | EEPROM-备份大小(KB) |
|---|---|---|
| 0000 | DFlash大小（512kB） | 0 |
| 0100 | DFlash大小 - EEPROM大小（512kB - 64kB = 448）。 | 64 |
| 1111 | DFlash大小（512kB） | 0 |

1. FCCOB5[7:4] = 0000

在清除CCIF以启动Program Partition命令后，FTFC首先验证数据闪存IFR中的EEPROM数据集大小代码和FlexNVM分区代码是否被擦除。如果被擦除了，那么Program Partition命令将擦除FlexNVM内存的内容。如果FlexNVM要为仿真EEPROM备份进行分区，那么所分配的仿真EEPROM备份扇区将被格式化以用于仿真EEPROM。最后，使用提供的数值将分区代码编程到数据闪存IFR中。编程分区的命令也验证了分区代码在编程后是否能正确读回。在编程分区操作完成后，CCIF标志被设置。

在启动Program Partition命令之前，数据闪存IFR必须处于擦除状态，这可以通过执行Erase All Blocks命令或外部请求来完成（参见Erase All Blocks命令）。

**表33-45.程序分区命令的错误处理**

| 错误情况 | 错误位 |
|---|---|
| 在当前模式/安全中无法使用的命令 | FSTAT[ACCERR] |
| EEPROM的数据大小和FlexNVM的分区代码字节最初不是0xFFFF | FSTAT[ACCERR] |
| 输入了无效的EEPROM数据集大小代码（有效代码见表33-41）。 | FSTAT[ACCERR] |
| 输入了无效的FlexNVM分区代码（有效代码见表33-42）。 | FSTAT[ACCERR] |
| FlexNVM分区代码=全数据闪存（无仿真EEPROM）和EEPROM数据集大小代码为仿真EEPROM分配FlexRAM | FSTAT[ACCERR] |

| | |
|---|---|
| FlexNVM分区代码为仿真EEPROM备份分配空间，但EEPROM数据集大小代码没有为仿真EEPROM分配FlexRAM | FSTAT[ACCERR] |

*表继续在下一页...*

表33-45.程序分区命令错误处理（续）

| 错误情况 | 错误位 |
|---|---|
| FCCOB4[7:6] != 00 | FSTAT[ACCERR] |
| FCCOB5[7:4] !=0000 | FSTAT[ACCERR] |
| 在验证操作中遇到了任何错误 | FSTAT[MGSTAT0] |

## 33.5.11.15 设置FlexRAM功能的命令

Set FlexRAM Function命令改变FlexRAM的功能。

- 当不为仿真EEPROM分区时，FlexRAM通常作为传统RAM使用。
- 当对仿真EEPROM进行分区时，FlexRAM通常用于存储EEPROM数据。

表33-46.设置FlexRAM功能命令FCCOB要求

| FCCOB号码 | FCCOB内容 [7:0] |
|---|---|
| 0 | 0x81 (SETRAM) |
| 1 | FlexRAM功能控制代码<br>(见表33-47) |
| 2 | 保留 |
| 3 | 保留 |
| 4 | 为EEPROM快速写入分配的FlexRAM字节数 [15:8] |
| 5 | 为EEPROM快速写入分配的FlexRAM字节数 [7:0] |
| 5 | 断电（BO）检测代码<br>• 0x00 - 没有检测到EEPROM问题<br>• 0x01 - 在完成EEPROM快写维护之前检测到BO<br>• 0x02 - 在完成EEPROM快速写入前检测到BO<br>• 0x04 - 在正常的EEPROM写入活动中检测到BO |
| 6 | 需要维护的EEPROM快写记录的数量 [15:8] |
| 7 | 需要维护的EEPROM快写记录的数量 [7:0] |
| 8 | EEPROM扇区擦除计数 [15:8] |
| 9 | EEPROM扇区擦除计数 [7:0] |

表33-47.FlexRAM 功能控制

| FlexRAM功能控制代码 | 行动 |
|---|---|

| | |
|---|---|
| 0xFF | 让FlexRAM作为RAM使用。<br>• 清除FCNFG[RAMRDY]和FCNFG[EEERDY]标志。<br>• 向所有的FlexRAM位置写一个1的背景<br>• 设置FCNFG[RAMRDY]标志 |
| 0xAA | 完成中断的EEPROM快速写入过程 |

*表继续在下一页...*

表33-47.FlexRAM功能控制（续）

| FlexRAM功能控制代码 | 行动 |
|---|---|
| | • 清除FCNFG.EEERDY标志（和FCNFG.RAMRDY标志）。<br>• 完成对中断的EEPROM快速写入操作的维护<br>• 设置FCNFG.EEERDY标志 |
| 0x77 | EEPROM快速写入状态查询<br>• 清除FCNFG.EEERDY标志（和FCNFG.RAMRDY标志）。<br>• 报告模拟的EEPROM状态<br>• 设置FCNFG.EERDY标志 |
| 0x55 | 使FlexRAM可用于EEPROM的快速写入<br>• 清除FCNFG.EERDY闪存（和FCNFG.RAMRDY标志）。<br>• 启用模拟的EEPROM系统进行EEPROM快速写入<br>• 设置FCNFG.EEERDY标志 |
| 0x00 | 使FlexRAM可用于仿真EEPROM。<br>• 清除FCNFG[RAMRDY]和FCNFG[EEERDY]标志。<br>• 向所有的FlexRAM位置写一个1的背景<br>• 将现有的EEPROM数据复制到FlexRAM上<br>• 设置FCNFG[EEERDY]标志 |

在清除CCIF启动设置FlexRAM功能的命令后，FTFC根据FlexRAM功能控制代码设置FlexRAM的功能。

当使FlexRAM作为传统RAM可用时，FTFC清除FCNFG[EEERDY]和FCNFG[RAMRDY]标志，用所有1的背景模式覆盖整个FlexRAM的内容，并设置FCNFG[RAMRDY]标志。EPROT寄存器的状态并不能阻止FlexRAM被覆盖。当FlexRAM被设置为RAM功能时，可以对FlexRAM进行正常的读和写访问。当需要对大段的闪存进行编程时。
例如，在工厂编程过程中，FlexRAM可以作为Program Section命令的Section Program Buffer使用（见Program Section命令）。

在清除EEERDY标志后，CSEc将处理所有先前未完成的EEPROM正常或快速写入活动。如果EEPROM快速写入活动没有完成写入所有要求的记录，那么写入的记录将被清除，有效地恢复以前的有效记录。如果所有的EEPROM快写记录都被写入，但是维护活动没有完成，维护将在剩余的快写记录上完成。CCIF和EEERDY标志将一直被否定，直到所有的EEPROM维护活动都完成。

在清除EEERDY标志后，CSEc将询问仿真EEPROM系统并报告EEPROM清理要求和EEPROM扇区擦除计数。如果EEPROM的正常写入活动在最终完成记录之前被重置或断电所打断，将返回BO检测代码0x04。如果EEPROM快速写入活动在写入

所有控制代码要求的快速写入记录之前被复位中断

0x55，BO检测代码0x02将被返回。如果中断发生在所有要求的快速写入之后，但在快速写入维护完成之前，BO检测代码0x01将被返回。CCIF和EEERDY标志将保持负值，直到EEPROM状态被加载到FCCOB寄存器。

当使FlexRAM可用于EEPROM快速写入时，CSEc将保持FlexRAM的内容不变，并为快速写入活动准备好仿真EEPROM系统。在FCNFG寄存器中的EEERDY标志被设置后，仿真的EEPROM系统已经准备好进行快速写入。当FlexRAM被设置为接受EEPROM快速写入时，对FlexRAM的读写访问是可用的，但是对FlexRAM的每一个32位的写入都会调用EEPROM快速写入活动。CCIF和EEERDY标志将在每一个32位的写入时去掉，并在相关的EEPROM快速写入活动完成后去掉。在最后一次对FlexRAM的32位写入之后，由ht
FCCOB4和FCCOB5的内容指定，仿真EEPROM系统将创建最后一次写入的数据记录（~150uSec）并开始对整个快速写入数据块进行EEPROM维护活动。CCIF和EEERDY标志将保持负值，直到所有EEPROM快写维护活动完成。

当使FlexRAM可用于仿真EEPROM时，FTFC清除FCNFG[RAMRDY]和FCNFG[EEERDY]标志，用所有1的背景图案覆盖分配给仿真EEPROM的FlexRAM的内容，并将现有的EEPROM数据从EEPROM备份记录空间复制到FlexRAM中。在完成EEPROM的复制后，FCNFG[EEERDY]标志被设置。当FlexRAM被设置为模拟EEPROM的功能时，对FlexRAM的正常读写访问是可用的，但是对FlexRAM的写入也会调用模拟EEPROM活动。

在设置FlexRAM功能操作完成后，CCIF标志将被设置。

**表33-48.设置FlexRAM功能命令的错误处理**

| 错误情况 | 错误位 |
|---|---|
| 在当前模式/安全中无法使用的命令 | FSTAT[ACCERR] |
| 未定义FlexRAM功能控制代码 | FSTAT[ACCERR] |
| FlexRAM功能控制代码被设置为使FlexRAM可用于仿真EEPROM，但FlexNVM没有为仿真EEPROM进行分区。 | FSTAT[ACCERR] |
| 如果FlexRAM功能控制代码被设置为使FlexRAM可用于EEPROM快速写入，但FCNFG.[EEERDY]=0（FlexRAM不在仿真EEPROM模式下），则设置。 | FSTAT[ACCERR] |
| 如果FlexRAM处于EEPROM快速写入模式，即所有对FlexRAM的EEPROM快速写入还没有完成，则设置该值 | FSTAT[ACCERR] |
| 如果FlexRAM功能控制代码被设置为使FlexRAM可用于EEPROM的快速写入，但分配给快速写入的字节数小于16，大于512，或不能被4整除，则设置该值（只允许32位快速写入）。 | FSTAT[ACCERR] |
| 如果模拟的EEPROM系统处于快速写入模式，并且对FlexRAM的写入不是32位，则设置该值 | FSTAT[ACCERR] |

## 33.5.12 安全问题

FTFC模块根据FSEC安全寄存器的内容向MCU提供安全信息。然后，MCU限制对FTFC资源的访问，如设备的芯片配置细节中所定义。在复位过程中，FTFC模块使用从闪存配置字段的安全字节读取的数据初始化FSEC寄存器（见闪存配置字段描述）。

以下字段在FSEC寄存器中可用。设置的细节在FSEC寄存器的描述中有所说明。

**表33-49.FSEC字段**

| FSEC领域 | 描述 |
|---|---|
| KEYEN | 后门钥匙访问 |
| MEEN | 大规模擦除能力 |
| FSLACC | 工厂访问 |
| SEC | 单片机安全 |

### 33.5.12.1 按模式和安全访问FTFC

下表总结了对FTFC模块的访问如何受安全和操作模式的影响。

**表33-50.FTFC访问摘要**

| 操作模式 | 芯片安全状态 | |
|---|---|---|
| | 不安全 | 安全 |
| NVM正常 | 完整的命令集 | |
| NVM专项 | 完整的命令集 | 只有Erase All Blocks、Erase All Blocks Unsecure和Read 1s All Blocks命令。 |

### 33.5.12.2 改变安全状态

复位后的安全状态可以通过编程闪存配置字段的安全字节来永久改变。这假定你是从一个模式开始的，在这个模式下，必要的编程闪存擦除和编程命令是可用的，并且

包含闪存配置字段的程序闪存区域是不受保护的。如果闪存安全字节被成功编程，其新值在下一次MCU复位后生效。

#### 33.5.12.2.1 利用后门钥匙访问解除对MCU的安全保护

MCU可以通过使用后门钥匙访问功能来实现不安全，这需要了解存储在闪存配置字段中的8字节后门钥匙值的内容（见闪存配置字段说明）。如果FSEC[KEYEN]位处于启用状态，可以运行验证后门访问密钥命令(见验证后门访问密钥命令)，该命令允许用户提出潜在的密钥与存储的密钥进行比较。如果钥匙匹配，FSEC[SEC]位将被改变，以解除MCU的安全。整个8字节的密钥不能全是0或全是1，即0x0000_0000_0000_0000和0xFFFF_FFFF_FFFF_FFFF不被验证后门访问密钥命令接受为有效的比较值。当验证后门访问密钥命令处于激活状态时，程序闪存不能用于读取访问，并返回无效的数据。

存储在程序闪存中的用户代码必须有一种从外部刺激接收后门钥匙的方法。这种外部刺激通常是通过片上的一个串行端口。

如果KEYEN位处于启用状态，MCU可以通过下面描述的后门密钥访问序列来实现不安全状态。

1. 按照验证后门访问密钥命令的命令顺序，如验证后门访问密钥命令中所解释的。
2. 如果验证后门访问密钥命令成功，MCU是不安全的，FSEC[SEC]位被强制到不安全状态。

提供给验证后门访问密钥命令的非法密钥，禁止今后使用验证后门访问密钥命令。当比较失败时，MCU的复位是重新启用验证后门访问密钥命令的唯一方法。

在后门密钥被正确匹配后，MCU通过改变FSEC[SEC]位来实现不安全状态。验证后门访问密钥命令的成功执行仅改变FSEC寄存器中的安全性。它不改变安全字节或存储在闪存配置字段中的密钥（闪存配置字段描述）。在MCU下次复位后，FTFC模块的安全状态又恢复到闪存配置域中的闪存安全字节。验证后门访问密钥命令序列对程序闪存保护寄存器中定义的编程和擦除保护没有影响。

如果后门钥匙成功匹配，不安全的MCU就能完全控制闪存配置域的内容。MCU可以擦除包含闪存配置字段的扇区，并将闪存安全字节重新编程为不安全状态，并将后门密钥改为任何想要的值。

## 33.5.13 加密服务引擎(CSEc)

FTFC模块增加了符合SHE1规范的功能。假设用户熟悉SHE规范，因此我们不在此复制该文件。

通过使用嵌入式处理器、固件和硬件辅助的AES-128子块，FTFC宏实现了安全信息应用的加密、解密、信息生成和认证算法。此外，TRNG和Miyaguchi-Prenell压缩子块实现了真正的随机数生成（AES子块的PRNG的熵发生器）。

要访问该命令功能集，必须使用PGMPART命令将该零件配置为EEE操作。更多内容请参见Program Partition命令。通过启用安全功能和配置一些用户钥匙，4 KByte EEERAM的总大小将因存储用户钥匙所需的空间而减少。然后，用户钥匙空间将有效地成为EEERAM中不可寻址的空间。

CCOB命令集被有效地扩展，包括与ECB、CBC和CMAC功能相关的SHE命令。在FTFE模块内启用了安全密钥的安全存储，包括多达17个用户密钥（MASTER_ECU_KEY、BOOT_MAC_KEY、BOOT_MAC被假定分配）。这些命令被称为CSEc命令集。CSEc命令集要求输入数据和命令，如用户CSEc命令界面和命令集中所述。

**注意事项**

1. 不可能同时执行：EEE操作，与编程、擦除（或其他标准FTFC闪存命令）有关的CCOB命令，CSEc命令。在任何时候都只能进行一个操作（任何类型）。
2. 在Erase Suspend（ERSSUSP）中执行CSEc命令将导致暂停的Erase操作被中止（不能恢复，同样中止的Erase命令将使内容处于不确定的状态）。

1. HIS SHE Specification - Secure Hardware Extension Functional Specification; Version 1.0.1; Rev429 from 23.03.2009

3. 也不可能在一个正在进行的CSEc命令的延续过程中执行一个不同的CSEc命令。
4. 可以在一个正在进行的CSEc命令的延续过程中执行FCCOB命令，但结果是现有的CSEc命令将被取消。
5. 开始执行CCOB命令或CSEc命令将锁定CCOB接口、EEERAM和PRAM。在所请求的命令完成之前，该锁一直处于状态。

**注意事项**

中断CSEc键的更新操作（由于断电、复位、电源超出规定的操作范围或任何其他原因）将使内容处于不确定的状态。用户必须采取适当的应对措施，以防止在CSEc密钥更新操作中断的情况下发生数据丢失。

出于安全考虑，在任何类型的调试器（JTAG、DP、SDB等）连接到SoC时，根据用户配置的SHE规范要求，一些功能将被禁用。

所有的信息数据块都要以128位块的形式提供。最小块大小为1。如果数据报文中包含的比特数少于128比特，在将数据传递给FTFC模块之前，必须将报文填充（根据SHE规范）到所需的128比特块大小。

在FTFC模块的内部存储着所需的密匙和唯一ID。两者都在不可寻址和不可更改的空间内，并在工厂最终测试中被编程。此外，还支持所述的用户密钥。用户密钥被存储在EEERAM空间中，用户密钥的空间量可以配置。这个空间是从4KB EEERAM的末端地址范围中减去的。从1到20（17个用户钥匙）的任何地方的用户钥匙都可以按照用户钥匙表中的描述进行配置。假设前三个键被MASTER_ECU_KEY、BOOT_MAC_KEY和BOOT_MAC占用。留下任何1到17个KEY<n>键来配置。

**表33-51.用户键和RAM_KEY**

| 关键名称 | 关键ID {KBS, Key IDx} | | 类型 | 尺寸（字节） | 评论 |
|---|---|---|---|---|---|
| 秘密密钥（SECRET_KEY | X | 0X0 | 身份证 | 16 | |
| UID | X | 0x0 | 身份证 | 15 | |
| MASTER_ECU_KEY | X | 0X1 | 非挥发性 | 16 | |
| 淘宝网 | X | 0x2 | 非挥发性 | 16 | |
| 淘宝网 | X | 0x3 | 非挥发性 | 16 | |

*表继续在下一页...*

表33-51.用户键和RAM_KEY（续）

| 关键名称 | 关键ID {KBS, Key IDx} | | 类型 | 尺寸（字节） | 评论 |
|---|---|---|---|---|---|
| key_01 - key_10 | 1'b0 | 0x4 - 0xD | 非挥发性 | 16 | 最少3个用户键 - 最多17个 |
| 键_11 - 键_17 | 1'b1 | 0x4 - 0xA | 非挥发性 | 16 | |
| 保留 | 1'b1 | 0xB - 0xE | 保留 | 16 | |
| 保留 | 1'b0 | 0xE | 保留 | 16 | |
| RAM_KEY | X | 0xF | 挥发性（RAM_KEY） | 16 | |

表33-52.其他挥发性键

| 关键名称 | 地址 | 类型 | 尺寸（字节） | 评论 |
|---|---|---|---|---|
| PRNG_KEY | 不适用 | 内存 | 16 | |
| PRNG_STATE | 不适用 | 内存 | 16 | |

表33-53.ROM键

| 关键名称 | 地址 | 类型 | 尺寸（字节） | 评论 |
|---|---|---|---|---|
| 秘密密钥（SECRET_KEY | 不适用 | 身份证 | 16 | |
| UID | 不适用 | 身份证 | 15 | 可通过GET_ID命令访问 |

### 33.5.13.1 密钥/种子/随机数生成

这是初始化和生成随机数的高层流程。

1. 运行CMD_INIT_RNG，从内部TRNG初始化一个随机种子。

**注意事项**

a. CMD_INIT_RNG必须在每次POR之后，以及第一次执行CMD_RND之前运行。
b. 如果在PRNG中使用种子之前对种子进行了压缩，那么CSESTAT[RIN]将被设置为

c. 注意，如果在没有初始化种子的情况下运行下一步（运行CMD_RND），将设置一个错误（ERC_RNG_SEED）。
d. n

2. 运行CMD_RND来生成一个随机数
   PRNG使用PRNG_STATE/KEY和SHE规格和AIS20标准的种子，如下所示。

- PRNG_STATEi = $ENC_{ECB, PRNG\_KEY}$ (PRNG_$STATE_{i-1}$ )
- rnd = $prng\_state_i$

3. 对于额外的随机数，用户可以继续执行CMD_RND，除非发生POR事件。每次发生POR事件时，在试图重新生成新的随机数之前，必须首先运行CMD_INT_RND命令

### 33.5.13.2 安全启动模式

支持三种用户可选择的启动模式（如SHE规范中所述）。启动模式在"CMD_BOOT_DEFINE "命令中被配置。

- 顺序启动模式
  a. 闪存系统从RESET中出来，主核心保持在RESET中，或可能从ROM代码中执行。
  b. FTFC验证客户固件块（通过CMAC计算）。
  c. 根据验证结果（CMAC验证通过/失败），FTFC和安全密钥将可用于安全任务或不可用于安全任务（根据通过/失败结果和个别密钥属性）。
  d. 主核心从复位中出来，开始执行客户的固件
- 严格的顺序启动模式
  a. 闪存系统从RESET中出来，主核心保持在RESET中，或者可能从ROM中执行。
  b. FTFC验证客户固件块（通过CMAC计算）。
  c. 根据固件验证结果（CMAC验证通过/失败），FTFC和安全密钥是否可用于安全任务（根据通过/失败结果和个别密钥属性）。
  d. 如果MAC比较失败，那么主核保持在RESET状态（不执行客户代码），或者可能执行ROM代码。

**注意事项**

1. 在选择 "Strict "安全启动模式之前，必须存储正确的BOOT_MAC。否则启动检查会失败，部件会被卡在复位状态。
2. 如果任何键被写保护，那么DBG_CHALLANGE/AUTH将不会通过，因为被写保护的键阻止了所有键的擦除。这也导致否定了以下的能力

失败分析返回是永远适用的。如果在应用中使用写保护任何键，必须小心谨慎。

- 并行启动模式
  a. FTFC系统和主核心从RESET中出来。
  b. 主核心开始执行客户的固件
  c. 在执行主核心代码的同时，FTFC模块验证了客户的固件块
  d. 根据固件验证的结果（CMAC通过/失败），FTFC和安全密钥是否可用于安全相关任务（根据通过/失败结果和个别密钥属性）。

当安全启动检测到软件或安全启动过程被操纵时，非易失性钥匙可能被禁用（每个钥匙都有一个属性来决定该钥匙的反应）。对启动失败敏感的内存插槽只有在下一次成功启动时才会被重新激活。此外，非易失性钥匙可以在连接调试器（JTAG、BDM等）时被禁用（同样，每个钥匙都有一个属性来决定这种行为），并且只有在下一次成功启动时才会重新激活。

### 33.5.13.3 用户CSEc命令界面和命令集

下文将描述CSEc命令集中实现的符合SHE标准的命令、命令接口、数据结构和与所有操作有关的协议。

这些命令遵循与CCOB命令类似的协议，一旦一个命令开始，CCOB接口将被锁定，在完成正在进行的命令之前，不得启动其他命令。

这导致一个SHE命令（CMD_CANCEL）没有在本模块的CSE命令功能集中实现。另外，CMD_STATUS并不被严格支持，因为我们可能不会中断一个正在进行的命令。但是CMD_STATUS命令的意图可以通过一个用户可访问的状态（FCSESTAT）寄存器来满足

所有加密功能都由AES-128引擎处理。数据的加密和解密由电子密码本（ECB）模式支持，用于单块（128位）数据，由密码块链（CBC）模式支持128位块大小的整数倍数据。

基于密码的信息验证码（CMAC）的生成和验证也由AES-128引擎支持，此外还使用了宫口-普雷内尔压缩功能。

CSEc命令集的使用是通过将数据（纯文本或密码文本）写入CSE_PRAM，然后按照下面指定的地址偏移量将命令字写入CSE_PRAM。向CSE_PRAM写入 "命令头 "字的行为将触发CCOB接口被锁定并开始执行命令操作。

**注意事项**

1. 使用模式要求先写CSE_PRAM数据内容，再写命令头内容。最后再写命令头!
2. 对命令头中的任何字节（字节0-3）的写入将锁定接口。
3. 一般来说，以命令头的字节[2]开始的命令（continue），但却是对该命令的第一次调用，将导致一个序列错误（ERC_SEQUENCE_ERROR）。
4. 对于持续的命令（命令头字节[2]）。=0x01）将忽略字节[1]（数据类型）、字节[3]（键槽）和任何MESSAGE_LENGTH、PAGE_LENGTH中的任何数据变化，因为这些数据只在第一次执行命令时被捕获。

CSEc命令的执行可以被认为是将系统内存内容块转移到CSE_PRAM空间以执行命令，然后将CSE_PRAM空间的内容转移回系统内存以使用。每次传输都是由一些128位的数据块组成，后面是命令头。数字'n'可以从0到7不等，因为CSE_PRAM中最多可以分配7个128位数据块的空间。此外，128位的命令头和命令的相关参数（如消息长度）。该格式在下面的每个命令描述中都有图形显示。

SHE规范列出了一组错误代码，以及哪个错误条件可能适用于哪个键。

下面是错误代码在这个宏中实现的总体上的优先顺序（从最高优先级到最低）。此外，还有一个粗略的例子列表，可能是错误的来源（注意，这个例子列表不是详尽的）。请参考每个命令，看看哪些特定的ERC_*代码适用于它。

- 误差。例子。FuncID是无效的
- 顺序错误。例子。CalSeq无效：延续一个命令错误--改变命令类型
- 误差。例子。信息长度为==0

- erc_key_invalid。例子：键槽不符合命令的每个键映射；KeyID[7:5]被设置；kBS==0 & KeyID==0xE
- 键不可用。例子：键有DBG附加标志，并且调试器处于活动状态
- ERC_KEY_EMPTY。例子：钥匙槽是空的（未初始化）/不存在或比分区更高的槽（未分区）；对于RAM_KEY，如果设置了平原属性
- erc_no_secure_boot。对于FTFC不适用 - BOOT_DEFINE一旦配置好，将自动运行安全启动
- erc_key_write_protected。例子：更新一个受写保护的键槽，或用受写保护的键激活调试器。
- erc_key_update_error。例如：密钥更新认证失败；对于LOAD_KEY : M3 =!M3*或UID为空，且AuthID不是同一个密钥槽
- ERC_RNG_SEED。例子：没有使用TRNG初始化种子，即CSESTAT[RIN] !=1
- erc_no_debugging。例子。DEBUG命令认证失败
- ERC_BUSY。对FTFC不适用
- ERC_memory_failure。例子：一般的内存技术故障（多比特ECC错误，检测到普通故障）。
- erc_general_error。例子：对上面没有定义的错误进行一般的捕捉；如果信息长度会导致序列越过读分区边界，也适用于'指针法GENERATE_MAC（最大CMAC可以在512KB程序闪存块上计算）。
- ERC_NO_ERROR。例子。没有错误 - 命令将执行

### 33.5.13.3.1 CSEc命令头

每条命令可能有不同数量的数据与之相关，但每条命令都有一个标准的命令字头。命令字头如下。

**图33-10.CSEc命令的标题**

| 字节数 | | | | | |
|---|---|---|---|---|---|
| 0 | 1 | 2 | 3 | 4 | 5 |
| 指挥词 | | | | 错误位 | |
| 编码（FuncID | 格式化功能 | 呼叫Seq | 关键ID | | |

**FuncID**（功能ID）是8位长。有效值范围是0x01到0x16。这些是CSEc命令，它们

遵循与SHE命令定义相同的值。

**FuncFormat**（函数格式）规定了数据如何被传送到/从CSEc中。有两种使用情况。一种（最常见的）是复制所有数据和命令函数调用方法，另一种是指针和函数调用方法。在 "复制
"方法中，主核或DMA将首先复制数据，然后将函数调用复制到CSE_PRAM中。对于'指针式'的情况（对于两个CSEc
MAC命令），主核或DMA将提供指针信息，然后是命令函数调用。

复制方法：0x00

指针方法：0x01

**CallSeq**（调用序列）指定了该信息是第一个还是后面的函数调用。使用情况是在不能将全部数据集与一个命令函数调用一起提供给CSE_PRAM空间的情况下提供持续的数据。例如，一个CMAC验证了一组128位的块，其中有超过7个128位的块。

0x00 : 第一个函数调用

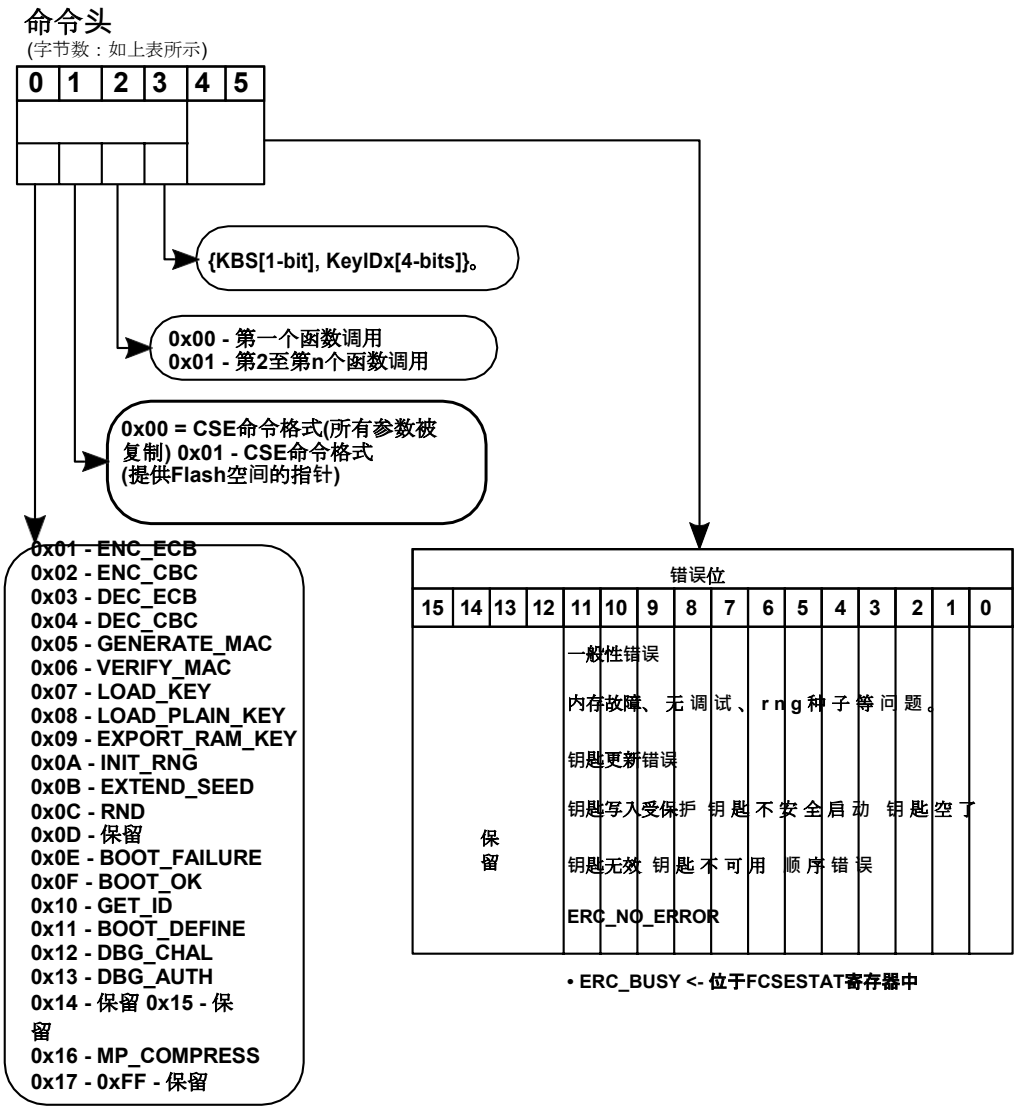
0x01 : 第2至第n个函数调用

**KeyID** {KeyIDx,
KBS}分为两部分。KeyIDx是一个4位的值，代表SHE规格的密钥索引。然后是KBS (Key Block Select -
不是在所有的命令中使用)，这是一个1位的值，允许客户在钥匙库之间切换。

**注意事项**

KeyIDx是一个4位的值。在上面的4位写1可能会导致寻址
到不正确的钥匙索引/槽。

最后，命令头还包含16位的错误信息。




## 33.5.13.4 通用的PRAM接口描述

PRAM接口可以被认为是一个128位宽的SRAM，有8个128位的页面。图中以大端数显示。第一个 "页
"总是从命令头开始，这必须是写到PRAM的数据序列中的最后一个数据。这是因为对命令头的写入（对任何一个字节0-
3的写入）会触发宏来锁定PRAM的接口，这样CSEc操作就可以开始了。

**图33-11.通用的PRAM接口**

职能描述

| 咬牙切齿 | [127:0] | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 咬牙切齿 | 31:24 | 23:17 | 15:8 | 7:0 | 31:24 | 23:17 | 15:8 | 7:0 | 31:24 | 23:17 | 15:8 | 7:0 | 31:24 | 23:17 | 15:8 | 7:0 |
| 纬度 | 字0 | | | | 字1 | | | | 字2 | | | | 字3 | | | |
| | 字节 | | | | | | | | | | | | | | | |
| 页码 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
| 0 | 趣味性D | 在的Func表格 | 呼叫Seq | 关键ID | 错误位 | | 具体命令 | | | | | | | | | |
| 1 | 输入到CSEc的数据或从CSEc输出的数据 | | | | | | | | | | | | | | | |
| 2 | | | | | | | | | | | | | | | | |
| 3 | | | | | | | | | | | | | | | | |
| 4 | | | | | | | | | | | | | | | | |
| 5 | | | | | | | | | | | | | | | | |
| 6 | | | | | | | | | | | | | | | | |
| 7 | | | | | | | | | | | | | | | | |

要设置CSEc命令，用户应首先按要求输入数据，以128位块为单位，根据需要输入多个块（在一次允许的7页内）。随后是任何相关的控制信息，如"MESSAGE_LENGTH"。
后面是最后一次写，是对命令头的写。

当CCIF从1过渡到0时，操作将开始。在这一点上，用户可以读取PRAM来验证或转移结果，如适用。

如果要继续执行相同的命令，用户应根据情况写入剩余的数据，然后写上更新的命令头信息（相同的命令，但字节2有所改变，以表明这是同一命令的延续）。对于任何命令的延续，KeyID、CSEc格式和信息长度只在写入的第一条命令中捕获。

## 注意事项

1. 如果在继续执行命令时，CMD字段发生变化，就会发生顺序错误。在整个命令的延续过程中，CMD字段必须保持一致。
2. 关于所有MAC和ENC/DEC命令的一般声明。
   根据SHE规范，用户/应用必须管理所有ENC/DEC命令的任何潜在填充（位长小于128位块长的块）。而CSEc模块将管理MAC命令的所有潜在填充。

### 33.5.13.5 呼叫中心：CMD_ENC_ECB

Encrypt
ECB命令对一个128位的数据块进行操作。该函数用KEY_ID（5位）标识的密钥对指定的PLAIN_TEXT（128位）进行加密，并返回CIPHER_TEXT（128位）。PAGE_LENGTH
"用于告诉操作要处理多少个128位页面。所有的数据都必须以128位的块呈现（所有的填充必须由应用程序完成）。这个命令也可以通过使用CallSeq字段扩展到0x01来继续处理数据的页面。

**表33-54.加密ECB命令详情**

| 参数 | 方向 | 宽度 |
|---|---|---|
| KEY_ID | 纳入 | 5 |
| 页面长度（PAGE_LENGTH | 纳入 | 16 |
| PLAIN_TEXT | 纳入 | n * 128 |
| CIPHER_TEXT | 淘汰 | n * 128 |
| cipher_text = enc ecb,key, key_id (plain_text) | | |
| 错误代码。没有错误, 顺序错误, 密钥不可用, 密钥无效, 密钥空, 内存错误, 忙乱, 一般错误。 | | |

用于CSE_PRAM内容写入/读取的CSEc命令和数据结构如下。

**图33-12.加密ECB输入参数格式**

| | 字节 | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 页码 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
| 0 | 0x01 | 0x00 | 0x00 | 关键ID | 错误位 | | 保留 | | | | | | | | PAGE_LENGTH | |
| 1 | 平行文本 1 [0:15] | | | | | | | | | | | | | | | |
| 2 | 平行文本 2 [0:15] | | | | | | | | | | | | | | | |
| 3 | 平行文本 3 [0:15] | | | | | | | | | | | | | | | |
| 4 | 平行文本 4 [0:15] | | | | | | | | | | | | | | | |
| 5 | plain_text 5 [0:15] | | | | | | | | | | | | | | | |
| 6 | plain_text 6 [0:15] | | | | | | | | | | | | | | | |
| 7 | plain_text 7 [0:15] | | | | | | | | | | | | | | | |

**图33-13.加密ECB输出参数格式**

职能描述

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0x01 | 0x00 | 0x00 | 关键 ID | 错误位 | | 保留 | | | | | | | | | |
| 1 | cipher_text 1 [0:15] | | | | | | | | | | | | | | | |

*表继续在下一页...*

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 2 | cipher_text 2 [0:15] | | | | | | | | | | | | | | | |
| 3 | cipher_text 3 [0:15] | | | | | | | | | | | | | | | |
| 4 | cipher_text 4 [0:15] | | | | | | | | | | | | | | | |
| 5 | cipher_text 5 [0:15] | | | | | | | | | | | | | | | |
| 6 | cipher_text 6 [0:15] | | | | | | | | | | | | | | | |
| 7 | cipher_text 7 [0:15] | | | | | | | | | | | | | | | |

### 33.5.13.6 CMD_ENC_CBC

Encrypt
CBC命令对多个128位数据块进行操作。该函数用由KEY_ID（5位）和初始化向量（IV）标识的密钥对若干PLAIN_TEXT（128位）块进行加密，然后为每个输入块返回CIPHER_TEXT（128位）块。

所有提交的数据必须是128位块（任何填充必须由应用程序完成）。

**表33-55.加密CBC命令详情**

| 参数 | 方向 | 宽度 |
|---|---|---|
| KEY_ID | 纳入 | 5 |
| 页面长度（PAGE_LENGTH | 纳入 | 16 |
| PLAIN_TEXT | 纳入 | n * 128 |
| CIPHER_TEXT | 淘汰 | n* 128 |
| cipher_text = enc cbc,key, key_id,iv (plain_text) | | |
| 错误代码。没有错误, 顺序错误, 密钥不可用, 密钥无效, 密钥空, 内存错误, 忙乱, 一般错误。 | | |

**图33-14.加密CBC输入参数格式**

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0x02 | 0x00 | 0x00 | 关键ID | 错误位 | | 保留 | | | | | | | | PAGE_LENGTH | |
| 1 | 四 [0:15] | | | | | | | | | | | | | | | |
| 2 | 纯文本 1 [0:15] | | | | | | | | | | | | | | | |
| 3 | 平行文本 2 [0:15] | | | | | | | | | | | | | | | |
| 4 | 平行文本 3 [0:15] | | | | | | | | | | | | | | | |
| 5 | 平行文本 4 [0:15] | | | | | | | | | | | | | | | |
| 6 | plain_text 5 [0:15] | | | | | | | | | | | | | | | |
| 7 | plain_text 6 [0:15] | | | | | | | | | | | | | | | |

**图33-15.加密CBC输出参数格式**

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0x02 | 0x00 | 0x00 | 关键ID | 错误位 | | 保留 | | | | | | | | | |
| 1 | 四 [0:15] | | | | | | | | | | | | | | | |
| 2 | cipher_text 1 [0:15] | | | | | | | | | | | | | | | |
| 3 | cipher_text 2 [0:15] | | | | | | | | | | | | | | | |
| 4 | cipher_text 3 [0:15] | | | | | | | | | | | | | | | |
| 5 | cipher_text 4 [0:15] | | | | | | | | | | | | | | | |
| 6 | cipher_text 5 [0:15] | | | | | | | | | | | | | | | |
| 7 | cipher_text 6 [0:15] | | | | | | | | | | | | | | | |

提供后续的信息页，CSEc命令头的字节2字段表示正在进行的操作要继续（即把CallSeq字段从0x00改为0x01）。

**图33-16.继续输入参数**

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0x02 | 0x00 | 0x01 | 关键ID | 错误位 | | 保留 | | | | | | | | | |
| 1 | plain_text 7 [0:15] | | | | | | | | | | | | | | | |
| 2 | plain_text 8 [0:15] | | | | | | | | | | | | | | | |
| 3 | plain_text 9 [0:15] | | | | | | | | | | | | | | | |
| 4 | 平行文本 10 [0:15] | | | | | | | | | | | | | | | |
| 5 | 平行文本 11 [0:15] | | | | | | | | | | | | | | | |
| 6 | 平行文本 12 [0:15] | | | | | | | | | | | | | | | |
| 7 | plain_text 13 [0:15] | | | | | | | | | | | | | | | |

**图33-17.输出参数的延续**

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0x02 | 0x00 | 0x01 | 关键ID | 错误位 | | 保留 | | | | | | | | | |
| 1 | cipher_text 7 [0:15] | | | | | | | | | | | | | | | |
| 2 | cipher_text 8 [0:15] | | | | | | | | | | | | | | | |
| 3 | cipher_text 9 [0:15] | | | | | | | | | | | | | | | |
| 4 | cipher_text 10 [0:15] | | | | | | | | | | | | | | | |
| 5 | cipher_text 11 [0:15] | | | | | | | | | | | | | | | |
| 6 | cipher_text 12 [0:15] | | | | | | | | | | | | | | | |
| 7 | cipher_text 13 [0:15] | | | | | | | | | | | | | | | |

然后继续传入和传出数据，直到完成完整的CBC操作。

### 33.5.13.7 呼叫中心：CMD_DEC_ECB

解密ECB命令对一个128位的数据块进行操作。该函数用KEY_ID（5位）标识的密钥解密给定的CYPHER_TEXT（128位），并返回PLAIN_TEXT（128位）。这个命令也可以通过使用CallSeq字段扩展到0x01，以获得持续的数据页。

所有提交的数据必须是128位块（任何填充必须由应用程序完成）。

**表33-56.解密ECB命令详情**

| 参数 | 方向 | 宽度 |
|---|---|---|
| KEY_ID | 纳入 | 5 |
| 页面长度（PAGE_LENGTH | 纳入 | 16 |
| CIPHER_TEXT | 纳入 | n * 128 |
| PLAIN_TEXT | 淘汰 | n * 128 |
| plain_text = dececb,key, key_id (cipher_text) | | |
| 错误代码。没有错误, 顺序错误, 密钥不可用, 密钥无效, 密钥已空, 内存失效, 忙乱, 一般错误 | | |

**图33-18.解密ECB输入参数格式**

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0x03 | 0x00 | 0x00 | 关键ID | 错误位 | | 保留 | | | | | | | | PAGE_LENGTH | |
| 1 | cipher_text 1 [0:15] | | | | | | | | | | | | | | | |
| 2 | cipher_text 2 [0:15] | | | | | | | | | | | | | | | |
| 3 | cipher_text 3 [0:15] | | | | | | | | | | | | | | | |
| 4 | cipher_text 4 [0:15] | | | | | | | | | | | | | | | |
| 5 | cipher_text 5 [0:15] | | | | | | | | | | | | | | | |
| 6 | cipher_text 6 [0:15] | | | | | | | | | | | | | | | |
| 7 | cipher_text 7 [0:15] | | | | | | | | | | | | | | | |

**图33-19.解密ECB输出参数格式**

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0x03 | 0x00 | 0x00 | 关键ID | 错误位 | | 保留 | | | | | | | | | |
| 1 | 纯文本 1 [0:15] | | | | | | | | | | | | | | | |
| 2 | 平行文本 2 [0:15] | | | | | | | | | | | | | | | |
| 3 | 平行文本 3 [0:15] | | | | | | | | | | | | | | | |

*表继续在下一页...*

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 4 | 平行文本 4 [0:15] | | | | | | | | | | | | | | | |
| 5 | plain_text 5 [0:15] | | | | | | | | | | | | | | | |
| 6 | plain_text 6 [0:15] | | | | | | | | | | | | | | | |
| 7 | plain_text 7 [0:15] | | | | | | | | | | | | | | | |

### 33.5.13.8 呼叫中心：CMD_DEC_CBC

解密CBC命令对多个128位数据块进行操作。该函数用KEY_ID（5位）和初始化向量（IV）标识的密钥解密若干CIPHER_TEXT（128位）块，然后为每个输入块返回PLAIN_TEXT（128位）块。

所有的数据都必须以128位块的形式呈现（任何填充必须由应用程序完成）。

**表33-57.解密CBC命令详情**

| 参数 | 方向 | 宽度 |
|---|---|---|
| KEY_ID | 纳入 | 5 |
| 页面长度（PAGE_LENGTH | 纳入 | 16 |
| 四 | 纳入 | 128 |
| CIPHER_TEXT | 纳入 | n * 128 |
| 页面长度（PAGE_LENGTH | 纳入 | 32 |
| plain_text = deccbc,key, key_id,iv （cipher_text）。 | | |
| 错误代码:错误代码：CER_no_error, CER_sequence_error, CER_key_not_available, CER_key_invalid,CER_key_empty, CER_memory_failure, CER_busy, CER_general_error | | |

**图33-20.解密CBC输入参数**

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0x04 | 0x00 | 0x00 | 关键ID | 错误位 | | 保留 | | | | | | | | PAGE_LENGTH | |
| 1 | 四 [0:15] | | | | | | | | | | | | | | | |
| 2 | cipher_text 1 [0:15] | | | | | | | | | | | | | | | |
| 3 | cipher_text 2 [0:15] | | | | | | | | | | | | | | | |
| 4 | cipher_text 3 [0:15] | | | | | | | | | | | | | | | |
| 5 | cipher_text 4 [0:15] | | | | | | | | | | | | | | | |
| 6 | cipher_text 5 [0:15] | | | | | | | | | | | | | | | |
| 7 | cipher_text 6 [0:15] | | | | | | | | | | | | | | | |

**图33-21.解密CBC输出参数**

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0x04 | 0x00 | 0x00 | 关键ID | 错误位 | | 保留 | | | | | | | | | |
| 1 | 四 [0:15] | | | | | | | | | | | | | | | |
| 2 | 纯文本 1 [0:15] | | | | | | | | | | | | | | | |
| 3 | 平行文本 2 [0:15] | | | | | | | | | | | | | | | |
| 4 | plain_text 3 [0:15] | | | | | | | | | | | | | | | |
| 5 | 平行文本 4 [0:15] | | | | | | | | | | | | | | | |
| 6 | plain_text 5 [0:15] | | | | | | | | | | | | | | | |
| 7 | plain_text 6 [0:15] | | | | | | | | | | | | | | | |

提供后续的信息页，CSEc命令头的字节2字段表示正在进行的操作要继续（即把CallSeq字段从0x00改为0x01）。

**图33-22.继续输入参数**

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0x04 | 0x00 | 0x01 | 关键ID | 错误位 | | 保留 | | | | | | | | | |
| 1 | cipher_text 7 [0:15] | | | | | | | | | | | | | | | |
| 2 | cipher_text 8 [0:15] | | | | | | | | | | | | | | | |
| 3 | cipher_text 9 [0:15] | | | | | | | | | | | | | | | |
| 4 | cipher_text 10 [0:15] | | | | | | | | | | | | | | | |
| 5 | cipher_text 11 [0:15] | | | | | | | | | | | | | | | |
| 6 | cipher_text 12 [0:15] | | | | | | | | | | | | | | | |
| 7 | cipher_text 13 [0:15] | | | | | | | | | | | | | | | |

**图33-23.输出参数的延续**

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0x04 | 0x00 | 0x01 | 关键ID | 错误位 | | 保留 | | | | | | | | | |
| 1 | plain_text 7 [0:15] | | | | | | | | | | | | | | | |
| 2 | plain_text 8 [0:15] | | | | | | | | | | | | | | | |
| 3 | plain_text 9 [0:15] | | | | | | | | | | | | | | | |
| 4 | 平行文本 10 [0:15] | | | | | | | | | | | | | | | |
| 5 | 平行文本 11 [0:15] | | | | | | | | | | | | | | | |
| 6 | 平行文本 12 [0:15] | | | | | | | | | | | | | | | |
| 7 | plain_text 13 [0:15] | | | | | | | | | | | | | | | |

### 33.5.13.9 cmd_generate_mac

生成MAC命令对一个长度为MESAGE_LENGTH的MESSAGE进行操作。该函数使用一个密钥（KEY_ID）为整个消息体编码一个MAC值（128-位）。

数据通过MESSAGE_LENGH的位数以128位的块进行处理，如果不是整数除以128位，则在最后一个块上进行内部填充。

如图33-27所示，在处理完全部信息之前，输出不会反映CMAC值，此时它将覆盖第二页的数据。

**表33-58.生成MAC命令详情**

| 参数 | 方向 | 宽度 |
|---|---|---|
| KEY_ID | 纳入 | 5 |
| 訊息長度 | 纳入 | 64 |
| 讯息 | 纳入 | n * 128 |
| 介质管理 (MAC) | 淘汰 | 128 |
| mac = cmackey, key_id (message, message_length) | | |
| 错误代码。没有错误, 顺序错误, 密钥不可用, 密钥无效, 密钥空, 内存错误, 忙乱, 一般错误。 | | |

**图33-24.生成MAC输入参数**

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0x05 | 0x00 | 0x00 | 关键ID | 错误位 | | 保留 | | | | | | 訊息長度 | | | |
| 1 | 数据1 [0:15] | | | | | | | | | | | | | | | |
| 2 | 数据2 [0:15] | | | | | | | | | | | | | | | |
| 3 | 数据3 [0:15] | | | | | | | | | | | | | | | |
| 4 | 数据4 [0:15] | | | | | | | | | | | | | | | |
| 5 | 数据5 [0:15] | | | | | | | | | | | | | | | |
| 6 | 数据6 [0:15] | | | | | | | | | | | | | | | |
| 7 | 数据7 [0:15] | | | | | | | | | | | | | | | |

**图33-25.生成MAC输出参数**

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0x05 | 0x00 | 0x00 | 关键 | 错误位 | | 保留 | | | | | | 訊息長度 | | | |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | | | | ID | | | |
| 1 | 数据1 [0:15] | | | | | | |
| 2 | 数据2 [0:15] | | | | | | |

*表继续在下一页...*

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 3 | 数据3 [0:15] | | | | | | | | | | | | | | | |
| 4 | 数据4 [0:15] | | | | | | | | | | | | | | | |
| 5 | 数据5 [0:15] | | | | | | | | | | | | | | | |
| 6 | 数据6 [0:15] | | | | | | | | | | | | | | | |
| 7 | 数据7 [0:15] | | | | | | | | | | | | | | | |

**图33-26.继续输入参数**

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0x05 | 0x00 | 0x01 | 关键 ID | 错误位 | | 保留 | | | | | | 訊息長度 | | | |
| 1 | 数据8[0:15] | | | | | | | | | | | | | | | |
| 2 | 数据9[0:15] | | | | | | | | | | | | | | | |
| 3 | 数据10 [0:15] | | | | | | | | | | | | | | | |
| 4 | 数据11 [0:15] | | | | | | | | | | | | | | | |
| 5 | 数据12 [0:15] | | | | | | | | | | | | | | | |
| 6 | 数据13 [0:15] | | | | | | | | | | | | | | | |
| 7 | 数据14 [0:15] | | | | | | | | | | | | | | | |

**图33-27.继续输出参数**

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0x05 | 0x00 | 0x01 | 关键 ID | 错误位 | | 保留 | | | | | | 訊息長度 | | | |
| 1 | 数据8[0:15] | | | | | | | | | | | | | | | |
| 2 | CMAC[0:15] | | | | | | | | | | | | | | | |
| 3 | 数据10 [0:15] | | | | | | | | | | | | | | | |
| 4 | 数据11 [0:15] | | | | | | | | | | | | | | | |
| 5 | 数据12 [0:15] | | | | | | | | | | | | | | | |
| 6 | 数据13 [0:15] | | | | | | | | | | | | | | | |
| 7 | 数据14 [0:15] | | | | | | | | | | | | | | | |

## 33.5.13.10 CSEc格式的CMD_GENERATE_MAC（指针法）。

对于完全包含在NVM宏中的数据，使用 "指针
"方法进行通信。对于内部NVM空间，数据的最大尺寸被限制为不超过一个读分区（最大512KB），或者如果起始地址不是

读取分区的开始（也就是说，地址序列不能跨越读取分区的边界）。闪存起始地址"必须是128位对齐的（对于VERIFY_MAC指针法也是如此）。

**图33-28.CSEc格式的GENERATE_MAC输入参数**

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0x05 | 0x01 | 0x00 | 关键ID | 错误位 | | 保留 | | | | | | 訊息長度 | | | |
| 1 | 闪存起始地址 | | | | 保留 | | | | | | | | | | | |
| 2 | 保留 | | | | | | | | | | | | | | | |
| 3 | | | | | | | | | | | | | | | | |
| 4 | | | | | | | | | | | | | | | | |
| 5 | | | | | | | | | | | | | | | | |
| 6 | | | | | | | | | | | | | | | | |
| 7 | | | | | | | | | | | | | | | | |

**图33-29.CMD_GENERATE_MAC输出参数的CSEc格式**

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0x05 | 0x01 | 0x00 | 关键ID | 错误位 | | 保留 | | | | | | 訊息長度 | | | |
| 1 | 闪存起始地址 | | | | 保留 | | | | | | | | | | | |
| 2 | CMAC [0:15] | | | | | | | | | | | | | | | |
| 3 | 保留 | | | | | | | | | | | | | | | |
| 4 | | | | | | | | | | | | | | | | |
| 5 | | | | | | | | | | | | | | | | |
| 6 | | | | | | | | | | | | | | | | |
| 7 | | | | | | | | | | | | | | | | |

## 33.5.13.11 CMD_VERIFY_MAC

验证MAC命令将给定的MESSAGE（可以是多个128位的数据块）的MAC与提供的MAC进行验证。要比较的MESSAGE，从MAC最左边的位开始，在MESSAGE_LENGTH中给出。值0是不允许的，SHE将其解释为比较MAC的所有位。如果提供的信息的长度与MESSAGE_LENGTH中所述的长度不同，则该函数返回错误。输入DATA1-
DATAn，直到所有信息都输入完毕，预期的MAC是所有数据输入后的最后一个条目，即MAC将是数据块N+1的位置，为'N'个数据块。

所有的填充都要按照SHE的规范使用MESSAGE_LENGTH来完成。

n = CIEL2(MESSAGE_LENGTH / 128)

第一个例子显示了6个DATA块，然后是预期的MAC，输出在操作结束时用MAC验证状态覆盖了DATA1块。

继续的例子显示了DATA1-10，从第一个例子继续，但是DATA7将被写在第一个例子的CMAC的位置上，接着是继续的DATA8，在下一页。最后在本例中以DATA10结束，在DATA10的条目后写上预期的CMAC值。

**表33-59.验证MAC命令详情**

| 参数 | 方向 | 宽度 |
|---|---|---|
| KEY_ID | 纳入 | 5 |
| 訊息長度 | 纳入 | 32 |
| 讯息 | 纳入 | n * 128 |
| 介质管理 (MAC) | 纳入 | 128 |
| MAC_LENGTH | 纳入 | 7 |
| 验证状态 | 淘汰 | 1 |
| $mac_{calc}$ = truncate($cmac_{KEY, key\_id}$ (message, message_length), mac_length) $mac_{ref}$ = truncate (mac, mac_length)<br>verification_status = (0 ~= ( $mac_{calc}$ - $mac_{ref}$ )) | | |
| 错误代码。没有错误, 顺序错误, 密钥不可用, 密钥无效, 密钥空, 内存错误, 忙乱, 一般错误。 | | |

**图33-30.验证MAC输入参数**

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0x06 | 0x00 | 0x00 | 关键ID | 错误位 | | 保留 | | MAC长度 | | 保留 | | 訊息長度 | | | |
| 1 | 数据1 [0:15] | | | | | | | | | | | | | | | |
| 2 | 数据2 [0:15] | | | | | | | | | | | | | | | |
| 3 | 数据3 [0:15] | | | | | | | | | | | | | | | |
| 4 | 数据4 [0:15] | | | | | | | | | | | | | | | |
| 5 | 数据5 [0:15] | | | | | | | | | | | | | | | |
| 6 | 数据6 [0:15] | | | | | | | | | | | | | | | |
| 7 | MAC [0:15] | | | | | | | | | | | | | | | |

**图33-31.验证MAC输出参数**

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|

职能描述

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 0x06 | 0x00 | 0x00 | 关键 ID | 错误位 | 保留 | MAC长度 | 保留 | 訊息長度 |

*表继续在下一页...*

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | | | | | 验证状态 | | 数据1 [6:15] | | | | | | | | | |
| 2 | 数据2 [0:15] | | | | | | | | | | | | | | | |
| 3 | 数据3 [0:15] | | | | | | | | | | | | | | | |
| 4 | 数据4 [0:15] | | | | | | | | | | | | | | | |
| 5 | 数据5 [0:15] | | | | | | | | | | | | | | | |
| 6 | 数据6 [0:15] | | | | | | | | | | | | | | | |
| 7 | MAC [0:15] | | | | | | | | | | | | | | | |

**图33-32.继续输入参数**

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0x06 | 0x00 | 0x01 | 关键ID | 错误位 | | 保留 | | MAC长度 | | 保留 | | 訊息長度 | | | |
| 1 | 数据8 [0:15] | | | | | | | | | | | | | | | |
| 2 | 数据9 [0:15] | | | | | | | | | | | | | | | |
| 3 | 数据10 [0:15] | | | | | | | | | | | | | | | |
| 4 | MAC [0:15] | | | | | | | | | | | | | | | |
| 5 | | | | | | | | | | | | | | | | |
| 6 | | | | | | | | | | | | | | | | |
| 7 | | | | | | | | | | | | | | | | |

**图33-33.继续输出参数**

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0x06 | 0x00 | 0x01 | 关键ID | 错误位 | | 保留 | | MAC长度 | | 保留 | | 訊息長度 | | | |
| 1 | | | | | 验证状态 | | 数据 7 [ 6:15] | | | | | | | | | |
| 2 | 数据9 [0:15] | | | | | | | | | | | | | | | |
| 3 | 数据10 [0:15] | | | | | | | | | | | | | | | |
| 4 | MAC [0:15] | | | | | | | | | | | | | | | |
| 5 | | | | | | | | | | | | | | | | |
| 6 | | | | | | | | | | | | | | | | |
| 7 | | | | | | | | | | | | | | | | |

## 33.5.13.12 CMD_VERIFY_MAC - CSEc格式（指针法）。

图33-34.验证MAC（指针法）输入参数

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0x06 | 0x00 | 0x01 | 关键ID | 错误位 | | 保留 | | MAC长度 | | 保留 | | 訊息長度 | | | |
| 1 | 闪存起始地址 | | | | 保留 | | | | | | | | | | | |
| 2 | MAC [0:15] | | | | | | | | | | | | | | | |
| 3 | 保留 | | | | | | | | | | | | | | | |
| 4 | | | | | | | | | | | | | | | | |
| 5 | | | | | | | | | | | | | | | | |
| 6 | | | | | | | | | | | | | | | | |
| 7 | | | | | | | | | | | | | | | | |

**图33-35.验证MAC（指针法）输出参数**

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0x06 | 0x00 | 0x01 | 关键ID | 错误位 | | 保留 | | MAC长度 | | 保留 | | 訊息長度 | | | |
| 1 | 闪存起始地址 | | | | 验证状态 | | 保留 | | | | | | | | | |
| 2 | MAC [0:15] | | | | | | | | | | | | | | | |
| 3 | 保留 | | | | | | | | | | | | | | | |
| 4 | | | | | | | | | | | | | | | | |
| 5 | | | | | | | | | | | | | | | | |
| 6 | | | | | | | | | | | | | | | | |
| 7 | | | | | | | | | | | | | | | | |

## 33.5.13.13 CMD_LOAD_KEY

LOAD KEY命令根据SHE规范更新一个内部密钥。如果一个受保护的密钥被加载到RAM_KEY中，该函数必须禁用普通密钥的状态位。

**表33-60.LOAD_KEY命令详情**

| 参数 | 方向 | 宽度 |
|---|---|---|
| **KEY_ID** | **纳入** | **5** |
| M1 | 纳入 | 128 |
| M2 | 纳入 | 256 |
| M3 | 纳入 | 128 |
| M4 | 淘汰 | 256 |
| M5 | 淘汰 | 128 |

| 职能描述 |
| --- |
| 错误代码。没有错误，序列错误，钥匙不可用，钥匙无效，钥匙受写保护，钥匙更新错误，钥匙空，内存错误，忙乱，一般错误。 |

## 注意事项

1. 由于扩大了钥匙的最大数量（超出SHE规范），并使用KBS来识别高阶和低阶钥匙，M1输入将遵循传统的SHE规范（要求两个4位ID和AUTHID条目以及UID），但命令将需要更多的信息来显示要求的KBS（0或1）。这将通过输入KEY_ID（{KBS,KeyID[3:0]}的额外要求来传达。最后，M1中的KeyID[3:0]必须与命令头中5位KEY_ID条目中的KeyID[3:0]一致。
2. 如果钥匙AuthID是空的，钥匙更新可以用AuthID = ID，Auth_value = 所有Fs；否则返回ERC_KEY_UPDATE_ERROR。
3. 当更新BOOT_MAC时，KBS必须被设置为0（KBS=1会产生错误）。

PGMPART配置（见程序分区命令）允许对SHE规范进行扩展，用户可以启用VERIFY_ONLY标志，以便GENERATE_MAC考虑到。对于VERIFY_ONLY属性，如果设置了，相关的内存插槽键将被CMD_GENERATE_MAC命令视为无效（仅对CMD_VERIFY_MAC有效）。如果KEY_USAGE==0，VERIFY_ONLY属性就没有作用。

SHE规范将M2定义为。

M2 - $ENC_{CBD,K1,IV=0}$(CID'|FID'|"0...0"95|KID') :SFE == 0x00

M2 - $ENC_{CBD,K1,IV=0}$(CID'|FID'|"0...0"94|KID')。SFE == 0x01

**表33-61.M2中的FID和零点填充，含SFE**

| PGMPART SFE字段 | LOAD_KEY的M2方程式变化 | FID |
|---|---|---|
| SFE == 0x00 (VERIFY_ONLY标志未启用，默认SHE) | FID（5位）\|"0...0"95 | { 写入保护\|启动保护\|调试器保护\|钥匙使用\|通配符 } |
| SFE == 0x01 (VERIFY_ONLY标志是启用，增强SHE) | FID（6位）\|"0...0"94 | { 写入保护\|启动保护\|调试器保护\|密钥使用\|通配符\|仅验证 } |

当在PGMPART命令中设置可选的SFE位以启用用户钥匙的VERIFY_ONLY标志时，FID字段将因VERIFY_ONLY标志而延长1位，相邻的补零字段减少1位，总共为94位。

如果启用了SFE，那么VERIFY_ONLY标志应该被设置为1'b1，以使VERIFY_ONLY属性被启用，并被清除为1'b0，以使特定密钥的VERIFY_ONLY属性被禁用。

**图33-36.加载按键输入参数**

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0x07 | 0x00 | 0x01 | 关键ID | 错误位 | | 保留 | | | | | | | | | |
| 1 | M1 [0:15] | | | | | | | | | | | | | | | |
| 2 | M2 [0:15] | | | | | | | | | | | | | | | |
| 3 | | | | | | | | | | | | | | | | |
| 4 | M3 [0:15] | | | | | | | | | | | | | | | |
| 5 | 保留 | | | | | | | | | | | | | | | |
| 6 | | | | | | | | | | | | | | | | |
| 7 | | | | | | | | | | | | | | | | |

**图33-37.加载关键输出参数**

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0x07 | 0x00 | 0x01 | 关键ID | 错误位 | | 保留 | | | | | | | | | |
| 1 | M1 [0:15] | | | | | | | | | | | | | | | |
| 2 | M2 [0:15] | | | | | | | | | | | | | | | |
| 3 | | | | | | | | | | | | | | | | |
| 4 | M3 [0:15] | | | | | | | | | | | | | | | |
| 5 | M4 [0:31] | | | | | | | | | | | | | | | |
| 6 | | | | | | | | | | | | | | | | |
| 7 | M5 [0:15] | | | | | | | | | | | | | | | |

### 33.5.13.14 cmd_load_plain_key

加载普通密钥命令，加载一个没有加密的密钥
"KEY"，并对该密钥进行验证。钥匙是以纯文本形式递交的。普通钥匙只能被装入RAM_KEY槽（根据SHE规范）。该命令设置RAM_KEY的纯文本密钥状态位。没有返回值。字节[1,3]中的命令头字段被忽略，而将字节[2]设置为0x01将产生一个序列错误。

**表33-62.LOAD_PLAIN_KEY命令详情**

| 参数 | 方向 | 宽度 |
|---|---|---|
| 关键 | 纳入 | 128 |
| | | |
| KEYRAM_KEY = key<br>ram_key_plain = 1 | | |
| 错误代码。错误代码：CER_no_error, CER_sequence_error, CER_busy, CER_general_error | | |

**图33-38.加载普通键的输入参数**

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0x08 | 0x00 | 0x00 | 关键ID | 错误位 | | 保留 | | | | | | | | | |
| 1 | 普通键[0:15] | | | | | | | | | | | | | | | |
| 2 | 保留 | | | | | | | | | | | | | | | |
| 3 | | | | | | | | | | | | | | | | |
| 4 | | | | | | | | | | | | | | | | |
| 5 | | | | | | | | | | | | | | | | |
| 6 | | | | | | | | | | | | | | | | |
| 7 | | | | | | | | | | | | | | | | |

该命令没有返回值

### 33.5.13.15 cmd_export_ram_key

Export RAM
Key命令将RAM_KEY导出为受SECRET_KEY保护的格式。该钥匙可以通过CMD_LOAD_KEY命令再次导入。

只有当RAM_KEY被写成纯文本时才能被导出，即通过CMD_LOAD_PLAIN_KEY。

这个命令是用来加载和卸载RAM_KEY的。安全标志和计数器的保留字段被设置

为 "0"。

除M1外，没有其他值，可以离开SHE/CSEc。

## 注意事项

由于在RAM_KEY更新的情况下，根据定义将计数器和标志设置为'0'，加密信息$M_2$
的第一个块将成为一个恒定的'0'块。

### 表33-63.导出RAM_KEY命令详情

| 参数 | 方向 | 宽度 |
|---|---|---|
| M1 | 淘汰 | 128 |
| M2 | 淘汰 | 256 |
| M3 | 淘汰 | 128 |
| M4 | 淘汰 | 256 |
| M5 | 淘汰 | 128 |
| K1 = KDF(KEYSECRET_KEY, KEY_UPDATE_ENC_) K2 = KDF(KEYSECRET_KEY, KEY_UPDATE_MAC_C)<br>CID = 0 (28-位)<br>FID = 0 (5-位)<br>m1 = uid \| idram_key \| idsecret_key<br>m2 = enccbc$_{k1,iv=0}$ (cid \| fid \| "0...0" 95 \| keyram_key) = enc $_{CBC,K1,\ IV=0}$ ("0 ...0" 128 \| keyram_key) m3 = cmac$_{K2}$ (m1 \| m2)<br>k3 = kdf (keyram_key , key_update_enc_c)i k4 = kdf (keyram_key , key_update_mac_c)i<br>m4 = uid\| idram_key \| idsecret_key \| enc $_{ebc,\ k2}$, (cid)<br>M5 = cmac$_{K4}$ (M4) | | |
| 错误代码。没有错误, 顺序错误, , 钥匙无效, 钥匙空, 内存错误, 忙乱, 一般错误。 | | |

### 图33-39.输出RAM_KEY输入参数

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0x09 | 0x00 | 0x00 | 关键ID | 错误位 | | 保留 | | | | | | | | | |
| 1 | 保留 | | | | | | | | | | | | | | | |
| 2 | | | | | | | | | | | | | | | | |
| 3 | | | | | | | | | | | | | | | | |
| 4 | | | | | | | | | | | | | | | | |
| 5 | | | | | | | | | | | | | | | | |
| 6 | | | | | | | | | | | | | | | | |
| 7 | | | | | | | | | | | | | | | | |

图33-40.导出RAM_KEY输出参数

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0x09 | 0x00 | 0x00 | 关键ID | 错误位 | | 保留 | | | | | | | | | |
| 1 | M1 [0:15] | | | | | | | | | | | | | | | |
| 2 | M2 [0:31] | | | | | | | | | | | | | | | |
| 3 | | | | | | | | | | | | | | | | |
| 4 | M3 [0:15] | | | | | | | | | | | | | | | |
| 5 | M4 [ 0:31] | | | | | | | | | | | | | | | |
| 6 | | | | | | | | | | | | | | | | |
| 7 | M5 [ 0:15[ | | | | | | | | | | | | | | | |

### 33.5.13.16 呼叫中心： 呼叫中心： 呼叫中心： 呼叫中心： 呼叫中心： 呼叫中心： 呼叫中心： 呼叫中心：

INIT_RNG命令为PRNG初始化并得出一个密钥。它必须在CMD_RND命令之前和每次电源循环或复位之后被调用。命令头的[1:3]字节被忽略。

**表33-64.INIT_RNG命令详情**

| 参数 | 方向 | 宽度 |
|---|---|---|
| 无 | | |
| 该命令必须忽略SECRET_KEY上的活动调试器保护或安全启动保护标志。<br><br>**注意：**该命令可能需要几百毫秒才能返回。 | | |
| 错误代码:没有错误, 顺序错误, 内存错误, 忙音, 一般错误 | | |

**图33-41.INIT_RNG输入参数**

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0x0A | 0x00 | 0x00 | 关键ID | 错误位 | | 保留 | | | | | | | | | |
| 1 | 保留 | | | | | | | | | | | | | | | |
| 2 | | | | | | | | | | | | | | | | |
| 3 | | | | | | | | | | | | | | | | |
| 4 | | | | | | | | | | | | | | | | |
| 5 | | | | | | | | | | | | | | | | |
| 6 | | | | | | | | | | | | | | | | |
| 7 | | | | | | | | | | | | | | | | |

INIT_RNG命令没有返回值。

### 33.5.13.17 cmd_extend_seed

EXTEND_SEED命令通过将以前的种子值和提供的ENTROPY压缩成一个新的种子来扩展PRNG的种子。这个新的种子将通过调用CMD_RND命令来生成一个随机数。在扩展种子之前，必须通过CMD_INIT_RNG初始化随机数发生器。

**表33-65.EXTEND_SEED命令详情**

| 参数 | 方向 | 宽度 |
|---|---|---|
| ENTROPY | 纳入 | 128 |
| 该命令必须忽略SECRET_KEY上的活动调试器保护或安全启动保护标志。<br><br>**注意：**该命令可能需要几百毫秒才能返回。 | | |
| 错误代码:错误代码：CER_no_error, CER_sequence_error, CER_rng_seed, CER_memory_failure, CER_busy, CER_general_error | | |

**图33-42.EXTEND_SEED输入参数**

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0x0B | 0x00 | 0x00 | 关键ID | 错误位 | | 保留 | | | | | | | | | |
| 1 | ENTROPY [0:15] | | | | | | | | | | | | | | | |
| 2 | 保留 | | | | | | | | | | | | | | | |
| 3 | | | | | | | | | | | | | | | | |
| 4 | | | | | | | | | | | | | | | | |
| 5 | | | | | | | | | | | | | | | | |
| 6 | | | | | | | | | | | | | | | | |
| 7 | | | | | | | | | | | | | | | | |

EXTEND_SEED命令没有返回值。

### 33.5.13.18 呼叫中心

CMD_RND命令返回一个由128个随机比特组成的向量。

在提供随机数之前，随机数发生器必须由CMD_INIT_RNG初始化。

字节[1,3]中的命令头字段被忽略，而将字节[2]设置为0x01将产生一个序列错误。

**表33-66.CMD_RND命令详情**

| 参数 | 方向 | 宽度 |
|---|---|---|
| RND | 淘汰 | 128 |
| 如果(sregrnd_init == 0)<br>PRNG_STATEi = ENCECB,KEY PRNG_KEY (PRNG_STATE i-1)<br>RND = PRNG_STATE i<br>END IF | | |
| 错误代码:错误代码：CER_no_error, CER_sequence_error, CER_rng_seed, CER_memory_failure, CER_busy, CER_general_error | | |

**图33-43.CMD_RND输入参数**

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0x0C | 0x00 | 0x00 | 关键ID | 错误位 | | 保留 | | | | | | | | | |
| 1 | 保留 | | | | | | | | | | | | | | | |
| 2 | | | | | | | | | | | | | | | | |
| 3 | | | | | | | | | | | | | | | | |
| 4 | | | | | | | | | | | | | | | | |
| 5 | | | | | | | | | | | | | | | | |
| 6 | | | | | | | | | | | | | | | | |
| 7 | | | | | | | | | | | | | | | | |

**表33-67.CMD_RND输出参数**

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0x0C | 0x00 | 0x00 | 关键ID | 错误位 | | 保留 | | | | | | | | | |
| 1 | RND [0:15] | | | | | | | | | | | | | | | |
| 2 | 保留 | | | | | | | | | | | | | | | |
| 3 | | | | | | | | | | | | | | | | |
| 4 | | | | | | | | | | | | | | | | |
| 5 | | | | | | | | | | | | | | | | |
| 6 | | | | | | | | | | | | | | | | |
| 7 | | | | | | | | | | | | | | | | |

### 33.5.13.19 cmd_secure_boot

这个命令不被FTFC/CSEc模块所支持。请看新的命令"CMD_BOOT_DEFINE"，它配置了要执行的安全启动类型。启用了CSE功能的部件在复位时将始终执行配置的安全启动类型。

CSEc功能自主运行，从闪存中读取自定义参数，并验证以字节为单位的SIZE数据中提供的引导程序的MAC。在支持CSEc的部件上，安全启动是通过对BOOT_MAC_KEY的编程来启用。

安全启动参数是通过CMD_BOOT_DEFINE命令输入的。

### 33.5.13.20 cmd_boot_failure

在自主安全引导完成后检查额外的信息块时，可以使用此命令（根据CMD_BOOT_DEFINE配置）。

**表33-68.CMD_BOOT_FAILURE 命令细节**

| 参数 | 方向 | 宽度 |
|---|---|---|
| 无 | | |
| 该命令将施加与CMD_SECURE_BOOT会检测到失败一样的制裁，但可以在启动过程的后期阶段使用。<br>CMD_BOOT_FAILURE只能在每次通电/复位后被调用一次，并且只有在SECURE_BOOT之前没有检测到任何错误，并且CME_BOOT_OK没有被调用的情况下，才可以被调用。<br>注意：CSEc模块提供了安全启动过程的第一个自主阶段（如果用BOOT_DEFINE配置的话），因此这个函数可以在以下阶段使用，向CSEc提供启动状态。<br>如果(sreg $_{secureBOOT}$ == 1 and sreg $_{BOOTOK}$ == 1 and sreg$_{BOOT\ finished}$ == 0)sreg $_{boot\ finished}$ =<br>1<br>sreg $_{boot\ ok}$ = 0<br>END IF | | |
| 错误代码:没有错误，没有顺序错误，没有安全启动，没有忙乱，没有一般错误。 | | |

**表33-69.CMD_BOOT_FAILURE 输入参数**

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0x0E | 0x00 | 0x00 | 关键ID | 错误位 | | 保留 | | | | | | | | | |
| 1 | 保留 | | | | | | | | | | | | | | | |
| 2 | | | | | | | | | | | | | | | | |

| 职能描述 |
| --- |
| 3 |
| 4 |

*表继续在下一页...*

表33-69.CMD_BOOT_FAILURE输入参数（续）

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 5 | | | | | | | | | | | | | | | | |
| 6 | | | | | | | | | | | | | | | | |
| 7 | | | | | | | | | | | | | | | | |

## 33.5.13.21 CMD_BOOT_OK

在自主安全启动完成后检查额外的信息块时，可以使用此命令（根据CMD_BOOT_DEFINE配置）。

表33-70.CMD_BOOT_OK命令详情

| 参数 | 方向 | 宽度 |
|---|---|---|
| 无 | | |
| 该命令用于标记成功的启动验证，用于比自主的 SECURE_BOOT（由 CMD_BOOT_DEFINE 定义的操作）晚的阶段。特别是它是为了锁定命令CMD_BOOT_FAILURE。<br>CMD_BOOT_OK只能在每次通电/复位后被调用一次，并且只有在SECURE_BOOT之前没有检测到任何错误，并且CME_BOOT_FAILIRE没有被调用的情况下，才可以被调用。<br>注意：CSEc模块提供了安全启动过程的第一个自主阶段（如果用BOOT_DEFINE配置的话），因此这个函数可以在以下阶段使用，向CSEc提供启动状态。<br>如果($sreg_{secureBOOT}$ == 1 and $sreg_{BOOTOK}$ == 1 and $sreg_{BOOT\ finished}$ == 0) $sreg_{boot\ finished}$ =<br>1<br>END IF | | |
| 错误代码:没有错误，没有顺序错误，没有安全启动，忙乱，一般错误。 | | |

表33-71.CMD_BOOT_OK输入参数

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0x0F | 0x00 | 0x00 | 关键ID | 错误位 | | 保留 | | | | | | | | | |
| 1 | 保留 | | | | | | | | | | | | | | | |
| 2 | | | | | | | | | | | | | | | | |
| 3 | | | | | | | | | | | | | | | | |
| 4 | | | | | | | | | | | | | | | | |
| 5 | | | | | | | | | | | | | | | | |
| 6 | | | | | | | | | | | | | | | | |

| | |
|---|---|
| 7 | |

### 33.5.13.22 CMD_GET_STATUS

该命令通过STATUS寄存器FCSESTAT实现。

### 33.5.13.23 CMD_GET_ID

GET_ID命令返回UID和状态寄存器的值（由挑战和数据的MAC保护）。如果MASTER_ECU_KEY是空的，那么返回的MAC值就被设置为'0'。

在这个命令中，"KeyID" 的输入被忽略了，因为 MASTER_ECU_KEY 是唯一需要考虑的有效密钥。

**表33-72.GET_ID命令详情**

| 参数 | 方向 | 宽度 |
|---|---|---|
| 面临的挑战 | 纳入 | 128 |
| 身份证 | 淘汰 | 120 |
| 证券交易委员会(SREG) | 淘汰 | 8 |
| 介质管理 (MAC) | 淘汰 | 128 |
| ID = UID<br>mac = cmackey_master_ecu_key(challenge \| id \| sreg) | | |
| 错误代码:错误代码：CER_no_error, CER_sequence_error, CER_key_not_available, CER_memory_failure, CER_general_error | | |

**图33-44.GET_ID输入参数**

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0x10 | 0x00 | 0x00 | 关键ID | 错误位 | | 储备的 | 保留 | | | | | | | | |
| 1 | 挑战 [0:15] | | | | | | | | | | | | | | | |
| 2 | 保留 | | | | | | | | | | | | | | | |
| 3 | | | | | | | | | | | | | | | | |
| 4 | | | | | | | | | | | | | | | | |
| 5 | | | | | | | | | | | | | | | | |
| 6 | | | | | | | | | | | | | | | | |
| 7 | | | | | | | | | | | | | | | | |

**图33-45.GET_ID输出参数**

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0x10 | 0x00 | 0x00 | 关键 ID | 错误位 | | 保留 | | | | | | | | | |
| 1 | 挑战 [0:15] | | | | | | | | | | | | | | | |
| 2 | ID [0:14] | | | | | | | | | | | | | | | SREG |
| 3 | MAC [0:15] | | | | | | | | | | | | | | | |
| 4 | 保留 | | | | | | | | | | | | | | | |
| 5 | | | | | | | | | | | | | | | | |
| 6 | | | | | | | | | | | | | | | | |
| 7 | | | | | | | | | | | | | | | | |

## 33.5.13.24 CMD_CANCEL

这个命令并不被FTFC模块严格支持。用户应该相应地管理软件驱动的命令。

对于任何在一系列持续命令中执行的CSEc命令，该命令可以通过以下几种机制之一被 "取消"。

例如，通过PRAM七块输入缓冲区输入数据的连续处理的CBC编码数据流，可以通过给出下一段数据的非法延续来
"取消"。如清除命令头中的CallSeq字段，或将FuncID改为不同的命令。每一个都会产生一个序列错误，这将杀死/取消正在进行的命令。

此外，在一个CSEc命令继续执行的暂停中启动任何FCCOB命令，都会导致FCCOB命令按预期执行完毕，但会杀死/取消当前的CSEc命令。

## 33.5.13.25 cmd_boot_define

这是一个非SHE指定的命令，在CSEc的实现中，允许用户定义用户BOOT_SIZE（引导代码的数据大小在信息长度中的位数，以便在每个POR上运行CMAC），以及引导的 "风味"（严格、串行、并行、无）。

字节[1:3]中的命令头字段被忽略。

## 注意事项

1. **警告**。如果选择了 "Strict Boot
   "选项，而提供的CMAC与计算出来的CMAC不一致，那么这个部件在复位时就会被挂起。此外，由于一旦选择了
   "严格"，启动方式就不能改变，这个错误将不能被扭转。请确保在选择 "严格
   "启动方式之前，验证CMAC和完整的启动过程是否完成并通过预期。

   此外，尽管严格启动一旦设定就不能改变，但可以改变CODE内容和相关的BOOT_MAC和SIZE。但是，如果在启用 "严格启动
   "选项的情况下，在更新CODE和匹配SIZE、BOOT_MAC之间发生复位或任何部件的中断，其结果将是BOOT_MAC与计算的CMAC不匹配，从而使部件陷入复位状态。

2. 自主的安全启动在PFlash块上运行，从地址'0'开始，在BOOT_SIZE
   #位之后结束。在其他地址的后续引导代码检查可以在以后通过CMD_VEFIFY_MAC命令运行。在安全启动过程结束后，会出现CMD_BOOT_OK或CMD_BOOT_FAILURE。

**表33-73.BOOT_DEFINE命令详情**

| 参数 | 方向 | 宽度 |
|---|---|---|
| BOOT_SIZE (# Bytes * 8, or Number of Bits : Maximum size is 512kBytes) | 纳入 | 32 |
| 靴子的味道<br>2b'00 : 严格 (SHE可选的启动 - OTP : 一旦设置就不能改变)<br>2'b01 : 串行启动方式 2'b10 :并<br>行启动方式<br>2'b11 : 没有定义Boot (MASTER_ECU_KEY)，还没有初始化，或者没有启用CSEc的部分。 | 纳入 | 8 |
| 错误代码。没有错误, 顺序错误, , 钥匙无效, , 记忆错误, 忙乱, 一般错误。 | | |

## 图33-46.BOOT DEFINE输入参数格式

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0x11 | 0x00 | 0x00 | 关键ID | 错误位 | | 保留 | | | | | | | | | |
| 1 | 保留 | | | | | | | | | | | 靴子的味道 | 编码：BOOT_SIZE | | | |

*表继续在下一页...*

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 2 | 保留 | | | | | | | | | | | | | | | |
| 3 | | | | | | | | | | | | | | | | |
| 4 | | | | | | | | | | | | | | | | |
| 5 | | | | | | | | | | | | | | | | |
| 6 | | | | | | | | | | | | | | | | |
| 7 | | | | | | | | | | | | | | | | |

### 33.5.13.26 呼叫中心： 呼叫中心： 呼叫中心： 呼叫中心： 呼叫中心： 呼叫中心： 呼叫中心： 呼叫中心：

SHE
CMD_DEBUG命令被修改，在CSEc命令中被分成两个步骤。CMD_DBG_CHAL之后是CMD_DBG_AUTH。在CMD_DBG_CHAL/AUTH命令对之后，在PGMPART命令重新发出之前，必须进行后续的复位。

发出 CMD_DBG_CHAL 命令是为了请求一个随机数，用户将使用这个随机数与 MASTER_ECU_KEY 和 UID 一起使用 CMD_DBG_AUTH 命令返回一个授权请求。

**注意事项**

CMD_DBG_CHAL命令之后必须有CMD_DBG_AUTH命令，否则在继续之前需要重新发布CMD_DBG_CHAL命令。

**表33-74.DEBUG命令详情**

| 参数 | 方向 | 宽度 |
|---|---|---|
| 面临的挑战 | 淘汰 | 128 |
| 错误代码:错误代码：CER_no_error，CER_sequence_error，CER_write_protected，CER_rng_seed，CER_no_debugging，CER_memory_failure，CER_busy，CER_general_error | | |

**图33-47.DEBUG输入参数**

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0x12 | 0x00 | 0x00 | 关键ID | 错误位 | | 保留 | | | | | | | | | |
| 1 | 保留 | | | | | | | | | | | | | | | |
| 2 | | | | | | | | | | | | | | | | |
| 3 | | | | | | | | | | | | | | | | |

| 职能描述 |
| --- |
| 4 |
| 5 |

*表继续在下一页...*

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 6 | | | | | | | | | | | | | | | | |
| 7 | | | | | | | | | | | | | | | | |

**图33-48.DEBUG输出参数**

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0x12 | 0x00 | 0x00 | 关键ID | 错误位 | | 保留 | | | | | | | | | |
| 1 | 挑战 [0:15] | | | | | | | | | | | | | | | |
| 2 | 保留 | | | | | | | | | | | | | | | |
| 3 | | | | | | | | | | | | | | | | |
| 4 | | | | | | | | | | | | | | | | |
| 5 | | | | | | | | | | | | | | | | |
| 6 | | | | | | | | | | | | | | | | |
| 7 | | | | | | | | | | | | | | | | |

### 33.5.13.27 呼叫中心： 呼叫中心： 呼叫中心： 呼叫中心： 呼叫中心： 呼叫中心： 呼叫中心： 呼叫中心： 呼叫中心

CMD_DBG_AUTH命令擦除存储在NVM存储器中的所有密钥（实际的和过时的）。

**注意事项**

如果在任何一个键上设置了WRITE_PROTECTION标志，CMD_DEBUG将不执行任何键的擦除。这也意味着ERS ALL命令将不能执行，ERS Block/Sector命令也不能在DFlash块上执行。

在CSEc确认认证后，用户密钥被擦除。如果有任何密钥设置了WRITE_PR OTECTION标志，该命令将停止，不提供任何故障分析模式。

在过程的SHE认证通过后，命令将擦除用户密钥（DFlash）， 将零件放回出厂状态。

如果认证失败，则不能执行该命令。

该命令忽略了活动调试器保护和安全启动保护标志以及空键。

在成功认证后，用户IFR，（用户）密钥（包括DFLASH和EERAM空间）已经被擦除。在这一点上，FlexRAM不再被配置为EEE，现在可以作为系统RAM（EEE RDY ==0，RAMRDY ==1）。

**表33-75.CMD_DBG_AUTH**

| 参数 | 方向 | 宽度 |
|---|---|---|
| 授权书 | 纳入 | 128 |
| SHE & 后台。<br>k = kdf(keymaster_ecu_key, debug_key_c) 她。<br>挑战 = cmd_rnd()<br>后台。<br>授权 = $cmac_K$ (challenge \| uid) | | |
| 错误代码。没有错误，有错误的序列，有错误的写法，有错误的种子，有错误的调试，有错误的内存失败，有错误的忙，有错误的一般错误。 | | |

**图33-49.DBG_AUTH输入参数**

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0x13 | 0x00 | 0x00 | 关键ID | 错误位 | | 保留 | | | | | | | | | |
| 1 | 授权 [0:15] | | | | | | | | | | | | | | | |
| 2 | 保留 | | | | | | | | | | | | | | | |
| 3 | | | | | | | | | | | | | | | | |
| 4 | | | | | | | | | | | | | | | | |
| 5 | | | | | | | | | | | | | | | | |
| 6 | | | | | | | | | | | | | | | | |
| 7 | | | | | | | | | | | | | | | | |

## 33.5.13.28 cmd_mp_compress

MP_COMPRESS命令访问CSEc功能集中的Miyaguchi-Prenell压缩功能。这个功能是作为区块密码的AES提供的，被用作一个压缩功能。如果信息不符合完整的128位块大小要求，必须按照SHE规范进行预处理，也就是说，如果信息的长度还没有达到128位，则由用户/应用程序负责进行填充。

对于这条命令，PAGE_LENGTH的输入是128位的页数，而KeyID的输入是不关心的。

## 表33-76.MP压缩命令详情

| 参数 | 方向 | 宽度 |
|---|---|---|
| 数据<n>。 | 纳入 | 128 |
| 页面长度（PAGE_LENGTH | 纳入 | 16 |
| 错误代码:错误代码：CER_no_error, CER_sequence_error, CER_general_error | | |

## 图33-50.MP压缩的输入参数

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0x16 | 0x00 | 0x00 | 关键ID | 错误位 | | 保留 | | | | | | | | PAGE_LENGTH | |
| 1 | 数据1 [0:15] | | | | | | | | | | | | | | | |
| 2 | 数据2 [0:15] | | | | | | | | | | | | | | | |
| 3 | 数据3 [0:15] | | | | | | | | | | | | | | | |
| 4 | 数据4 [0:15] | | | | | | | | | | | | | | | |
| 5 | 数据5 [0:15] | | | | | | | | | | | | | | | |
| 6 | 数据6 [0:15] | | | | | | | | | | | | | | | |
| 7 | 数据7 [0:15] | | | | | | | | | | | | | | | |

## 图33-51.MP压缩输出参数

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0x16 | 0x00 | 0x00 | 关键ID | 错误位 | | 保留 | | | | | | | | 消息_length | |
| 1 | mp_压缩 [0:15] | | | | | | | | | | | | | | | |
| 2 | 数据2 [0:15] | | | | | | | | | | | | | | | |
| 3 | 数据3 [0:15] | | | | | | | | | | | | | | | |
| 4 | 数据4 [0:15] | | | | | | | | | | | | | | | |
| 5 | 数据5 [0:15] | | | | | | | | | | | | | | | |
| 6 | 数据6 [0:15] | | | | | | | | | | | | | | | |
| 7 | 数据7 [0:15] | | | | | | | | | | | | | | | |

## 图33-52.继续输入参数

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0x16 | 0x00 | 0x01 | 关键ID | 错误位 | | 保留 | | | | | | | | 消息_length | |
| 1 | 数据8[0:15] | | | | | | | | | | | | | | | |
| 2 | 数据9[0:15] | | | | | | | | | | | | | | | |

| 职能描述 | |
|---|---|
| 3 | 数据10 [0:15] |

*表继续在下一页...*

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 4 | 数据11 [0:15] | | | | | | | | | | | | | | | |
| 5 | 数据12 [0:15] | | | | | | | | | | | | | | | |
| 6 | 数据13 [0:15] | | | | | | | | | | | | | | | |
| 7 | 数据14 [0:15] | | | | | | | | | | | | | | | |

**图33-53.输出参数的延续**

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0x16 | 0x00 | 0x01 | 关键ID | 错误位 | | 保留 | | | | | | | | 消息_l ength | |
| 1 | mp_compress[0:15]。 | | | | | | | | | | | | | | | |
| 2 | 数据9 [0:15] | | | | | | | | | | | | | | | |
| 3 | 数据10 [0:15] | | | | | | | | | | | | | | | |
| 4 | 数据11 [0:15] | | | | | | | | | | | | | | | |
| 5 | 数据12 [0:15] | | | | | | | | | | | | | | | |
| 6 | 数据13 [0:15] | | | | | | | | | | | | | | | |
| 7 | 数据14 [0:15] | | | | | | | | | | | | | | | |

## 33.5.14 复位顺序

在每次系统复位时，FTFC模块会执行一个序列，为闪存块配置参数、FPROT、FDPROT、FEPROT、FOPT和FSEC寄存器以及FCNFG[RAMRDY, EEERDY]位建立初始值。

启动将根据BOOT_DEFINE命令设置的配置（是否安全启动，以及什么类型的安全启动）进行。

在整个复位过程中，CCIF被清零。FTFC模块在复位序列的一部分时间内暂缓所有的CPU访问。当保持被移除时，可以进行闪存读取。复位序列的完成是以设置CCIF为标志的，它使闪存用户命令得以实现。

如果在任何FTFC命令进行时发生复位，该命令将立即被中止。正在编程的字或正在擦除的扇区/块的状态不被保证。退出复位后，命令和操作不会自动恢复。

# 第34章
# 四个串行外设接口（QuadSPI）。

## 34.1 芯片特定的QuadSPI信息

### 34.1.1 概述

S32K148有一个QuadSPI的实例。S32K1xx系列的其他产品没有QuadSPI。

有关目标频率的详情，请参见设备数据表。

四个串行外设接口（QuadSPI）块作为外部串行闪存设备的一个接口。它支持SDR和HyperRAM模式，分别达到4和8条双向数据线。

**注意事项**

以下情况不被支持。
- AHB写
- 数据学习功能

### 34.1.2 内存大小要求

QuadSPI对AHB缓冲区的内存大小要求是：128x64，即1KB。

### 34.1.3 QuadSPI寄存器重置值

**表34-1.QuadSPI寄存器的复位值**

| 注册 | 重置值 |
|---|---|
| 模块配置寄存器（QuadSPI_MCR）。 | 000F_400C |
| 缓冲器0配置寄存器（QuadSPI_BUF0CR）。 | 0000_0003 |

*表继续在下一页...*

表34-1.QuadSPI寄存器的复位值（续）

| 注册 | 重置值 |
|---|---|
| 缓冲器1配置寄存器（QuadSPI_BUF1CR）。 | 0000_0002 |
| 缓冲器2配置寄存器（QuadSPI_BUF2CR）。 | 0000_0001 |
| 缓冲器3配置寄存器（QuadSPI_BUF3CR）。 | 0000_0000 |
| 串行闪存A1顶部地址（QuadSPI_SFA1AD）。 | 6C00_0000 |
| 串行闪存A2顶部地址（QuadSPI_SFA2AD）。 | 6C00_0000 |
| 串行闪存B1顶部地址（QuadSPI_SFB1AD）。 | 7000_0000 |
| 串行闪存B2顶部地址(QuadSPI_SFB2AD) | 7000_0000 |

## 34.1.4 使用案例

QuadSPI和外部存储器的主要应用情况是：。

- 手写识别 - 外部闪存作为字体/字符数据的存储器。
- RDS广播 - 用于临时存储RDS数据的外部RAM。

## 34.1.5 支持的读取模式

| 阅读模式 | | | QuadSPI_MCR[DDR_EN] | QuadSPI_MCR[DQS_EN] | 支持的模式 | 数据学习支持 |
|---|---|---|---|---|---|---|
| SDR模式（闪光灯A） | 内部采样(N/1) | | 0 | 0 | 是 | 没有 |
| | DQS抽样方法 | 内部产生的DQS | 0 | 1 | 是 | 没有 |
| | | 外部DQS | 0 | 1 | 没有 | 没有 |
| | | 回路DQS | 0 | 1 | 是 | 没有 |
| SDR模式（闪光灯B） | 内部采样(N/1) | | 0 | 0 | 是 | 没有 |
| | DQS抽样方法 | 内部产生的DQS | 0 | 1 | 没有 | 没有 |
| | | 外部DQS | 0 | 1 | 没有 | 没有 |
| | | 回路DQS | 0 | 1 | 没有 | 没有 |
| HyperRAM (DDR) | 内部采样（4倍法） | | 1 | 0 | 没有 | 没有 |
| | DQS抽样方法 | 内部产生的DQS | 1 | 1 | 没有 | 没有 |
| | | 外部DQS | 1 | 1 | 是 | 没有 |
| | | 回路DQS | 1 | 1 | 没有 | 没有 |

## 34.1.6 外部存储器选项

QuadSPI支持一个A端和一个B端。QuadSPI的A端连接到快速焊盘（80 mA），而B端连接到20 mA焊盘。操作值见数据表。

在任何特定的应用中，只支持一个外部存储器，不允许同时运行QuadSPI的A端和B端。因此，可以支持下列外部存储器选项。

- 在A面或B面有一个单一的Quad Flash。
- 在B面的单个HyperRAM或
- 在B面有一个单一的Quad Flash。

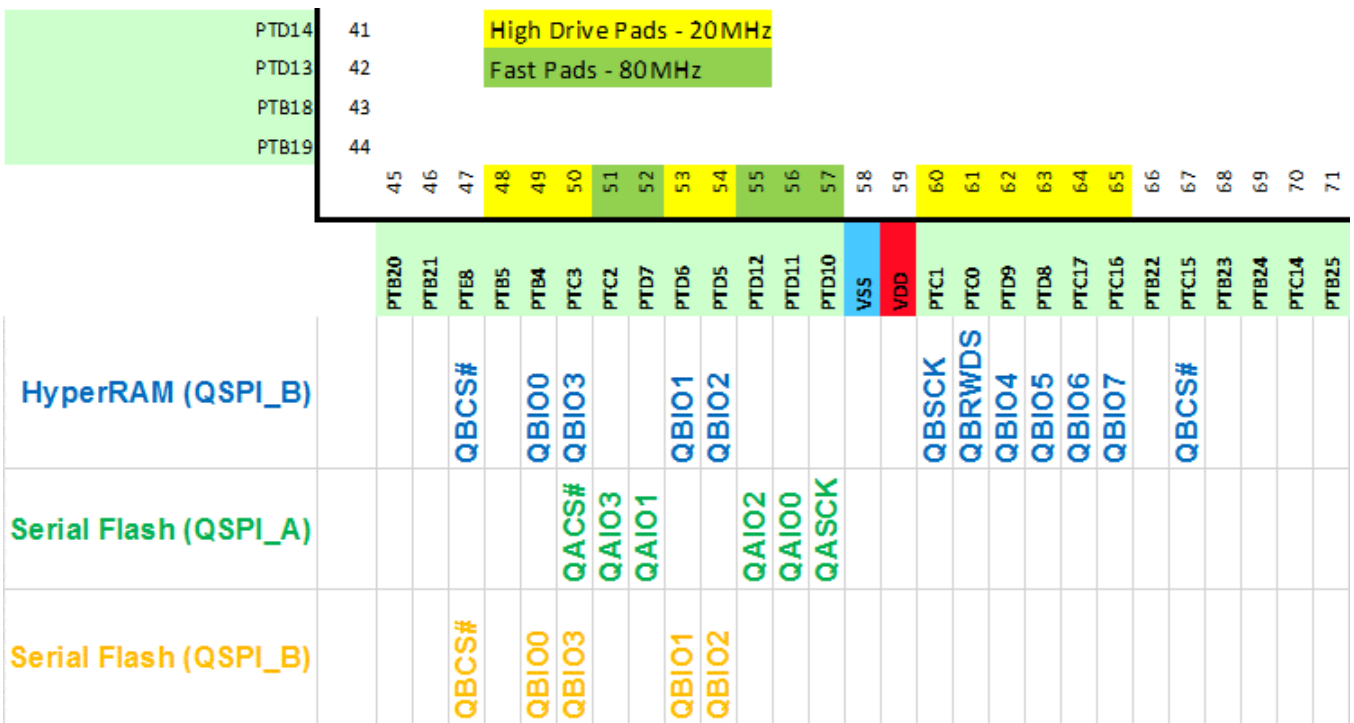



**图34-1.外部存储器选项**

## 34.1.7 推荐的软件配置

- QuadSPI的操作被限制在系统时钟作为SPLL。
- 在HSRUN和RUN之间切换系统模式时，应禁用QuadSPI，然后重新启用。

- 在启用QuadSPI的时钟之前，应通过寄存器SOC_CONFIG和SCLK_CONFIG为QuadSPI提供软件配置选项。
- 在配置选择QuadSPI焊盘的ALT模式选择之前，先配置QuadSPI模块。

**注意事项**

- 用户应确保在开始进入低功耗模式之前，QuadSPI上的通信已经完成。
- 可编程分频器的分频值（详见模块时钟中的QuadSPI时钟图）应在启用前进行编程。分频器默认是启用的。

## 34.1.8 推荐的编程顺序

以下是推荐的编程顺序。

1. 启用 PCC_QSPI[CGC]
2. 对 QuadSPI_MCR[SCLKCFG]和 QuadSPI_SOCCR[SOCCFG]进行编程
3. 将QuadSPI_MCR[MDIS]编程为0

## 34.1.9 QuadSPI时钟之间的时钟比率

在QuadSPI的sfif（闪存内部参考时钟）和ahb（总线时钟）之间应保持以下时钟关系。

从闪存中获取下一个n位数据（n可以是64、32或16）所需的时钟周期的总时间段> 2.5个总线时钟周期。

**比如说。**

在八进制DDR模式下，获取64位数据所需的时钟周期数将是4。因此，需要考虑的频率比率将是 。

4xInternal_reference_clock_period > 2.5xbus时钟周期

## 34.1.10 QuadSPI_MCR[SCLKCFG]的实现

**表34-2.QuadSPI_MCR[SCLKCFG]位字段描述**

| 比特字段名称 | 位字段描述 |
|---|---|

| | |
|---|---|
| SCLKCFG[7] | 为SDR和HyperRAM模式启用QSPI焊盘的输入缓冲器 |

*表继续在下一页...*

表34-2.QuadSPI_MCR[SCLKCFG]位字段描述（续）

| 比特字段名称 | 位字段描述 |
| --- | --- |
| | 0：禁用所有QSPI焊盘的ibe<br>1: 启用所有QSPI焊盘的ibe |
| SCLKCFG[6] | **Quadspi时钟模式选择。**在配置模式选择时，一定要对SCLKCFG[6]编程两次。<br>0: 选择SYS_CLK作为AHB读接口时钟和模块时钟。首选用于HSRUN 80/RUN 64/RUN 80配置。<br>1: 选择BUS_CLK作为AHB读取接口时钟和模块时钟。HSRUN 112配置时必须使用。 |
| SCLKCFG[5] | **用于闪存B的超RAM选择 - 用于闪存B的DQS的参考时钟选择**<br>0：来自SCLKCFG [3]的时钟被选为DQS（HyperRAM被禁用）。<br>1：来自闪存B的外部RWDS被选为DQS（启用了HyperRAM）。<br>**注意：**当HyperRAM被启用时，除了sfif时钟外，QuadSPI还需要2个内部参考时钟，因为HyperRAM是一个DDR协议。如果用户将PLAT_CLK配置为80MHz，那么用于Quadspi异步时钟选择的PCC分频器必须被编程为divby2，SCLKCFG[5]必须被编程为1，以便HyperRAM功能运行。 |
| SCLKCFG[4] | **Quadspi的内部参考时钟（异步时钟域）来源选择**<br>0: PLL_DIV1 是 Quadspi 内部参考时钟的时钟源 1: FIRC_DIV1 是 Quadspi 内部参考时钟的时钟源 |
| SCLKCFG[3] | **为Flash-B的DQS选择参考时钟**<br>0：从SCLKCFG [2]选择的倒置时钟作为DQS 1：从SCLKCFG [2]选择的时钟作为DQS |
| SCLKCFG[2] | **为Flash-B的DQS选择参考时钟**<br>0: 内部参考时钟选择为DQS<br>1: 来自PAD SCKB的环回时钟被选为DQS |
| SCLKCFG[1] | **为Flash-A的DQS选择参考时钟**<br>0：从SCLKCFG [0]选择的倒置时钟作为DQS 1：从SCLKCFG [0]选择的时钟作为DQS |
| SCLKCFG[0] | **为Flash-A的DQS选择参考时钟**<br>0: 内部参考时钟选择为DQS<br>1: 来自PAD SCKA的环回时钟被选为DQS |

## 34.1.11 QuadSPI_SOCCR[SOCCFG]实现

表34-3.QuadSPI_SOCCR[SOCCFG]位字段描述

| 比特字段名称 | 位字段描述 |
| --- | --- |

| | |
|---|---|
| SOCCFG[7:0] | **SOC配置。闪存A的精细延迟链配置** |

*表继续在下一页...*

## 表34-3.QuadSPI_SOCCR[SOCCFG]位字段描述（续）

| 比特字段名称 | 位字段描述 |
| --- | --- |
| | 7F :127个缓冲区和128个复用器的延迟<br>7E :126个缓冲区和127个复用器的延迟<br>7D :125个缓冲区和126个复用器的延迟<br>-<br>-<br>-<br>3F：63个缓冲区和64个复用器的延迟<br>3E：62个缓冲区和63个复用器的延迟<br>3D：61个缓冲区和62个复用器的延迟<br>-<br>-<br>-<br>0：0号缓冲器和1号复用器的延迟 |
| SOCCFG[15:8] | **SOC配置。闪存B的精细延迟链配置**<br>7F :127个缓冲区和128个复用器的延迟<br>7E :126个缓冲区和127个复用器的延迟<br>7D :125个缓冲区和126个复用器的延迟<br>-<br>-<br>-<br>3F：63个缓冲区和64个复用器的延迟<br>3E：62个缓冲区和63个复用器的延迟<br>3D：61个缓冲区和62个复用器的延迟<br>-<br>-<br>-<br>0：0号缓冲器和1号复用器的延迟 |
| SOCCFG[16] | **待读启用控制总线垫圈对待读事务的处理。**<br>0: 禁用待读数据。<br>1: 启用待读，产生通过垫圈的性能改进。 |

| | |
|---|---|
| SOCCFG[17] | **突发读启用控制总线垫圈对突发读事务的处理。**<br>0：突发读数在垫圈的从属侧被转换为一系列的单一事务。1: 突发读取被优化为最佳系统性能。 |
| SOCCFG[18] | **突发写启用控制总线垫圈对突发读事务的处理。**<br>0：突发写入在垫圈的从属侧被转换为一系列的单一事务。<br>1: 突发写入被优化为最佳系统性能。注意这个设置将写入处理为 "不精确"，这样在突发的任何节拍上的错误响应会在最后一个节拍上报告。 |

*表继续在下一页...*

表34-3.QuadSPI_SOCCR[SOCCFG]位字段描述（续）

| 比特字段名称 | 位字段描述 |
| --- | --- |
| SOCCFG[28] | **可编程分割器禁用。**<br>0: 启用除法器<br>1: 分频器禁用 |
| SOCCFG[31:29] | **可编程分压器配置选择。**<br>000：除以1<br>001：除以2<br>010 :除以3<br>011 : DIV by 4<br>100：除以5<br>101: DIV by 6<br>110: DIV by 7<br>111:DIV by 8 |

## 34.2 简介

四个串行外设接口（QuadSPI）块作为一个接口连接一个或两个外部串行闪存设备，每个设备有多达8条双向数据线。

### 34.2.1 特点

QuadSPI支持以下功能。

- 灵活的序列引擎，支持各种闪存供应商的设备。由于没有特定的标准，该模块支持不同供应商的各种闪存。请参考串行闪存设备的例子序列。
- 单、双、四和八的工作模式。
- 双倍数据率（DDR）/双倍传输率（DTR）模式，其中数据是在串行闪存时钟的每个边沿产生的。
- 支持闪存数据选通信号，用于DDR和单数据率（SDR）模式下的数据采样。
- 对HyperRAM的支持。

- AHB主站通过AMBA
  AHB总线（64位宽度接口）或IP寄存器空间（32位访问）读取RX Buffer数据，并通过IPS寄存器空间（32位访问）填充TX Buffer。
  - AHB主站可以是一个具有可配置内循环大小的DMA。
- 有优先权的多主机访问
  - 为每个主站提供灵活和可配置的缓冲器
- 多个中断条件（见表34-12）。
- 所有对闪存设备的AHB访问都是直接内存映射到芯片系统内存。
- 可编程序列引擎，以迎合未来的命令/协议变化，并能够支持所有现有的供应商命令和操作。软件需要根据连接的闪存设备来选择相应的序列。
  - 支持所有类型的寻址。

### 34.2.2 框图

下图是四路串行外设接口（QuadSPI）模块的框图。

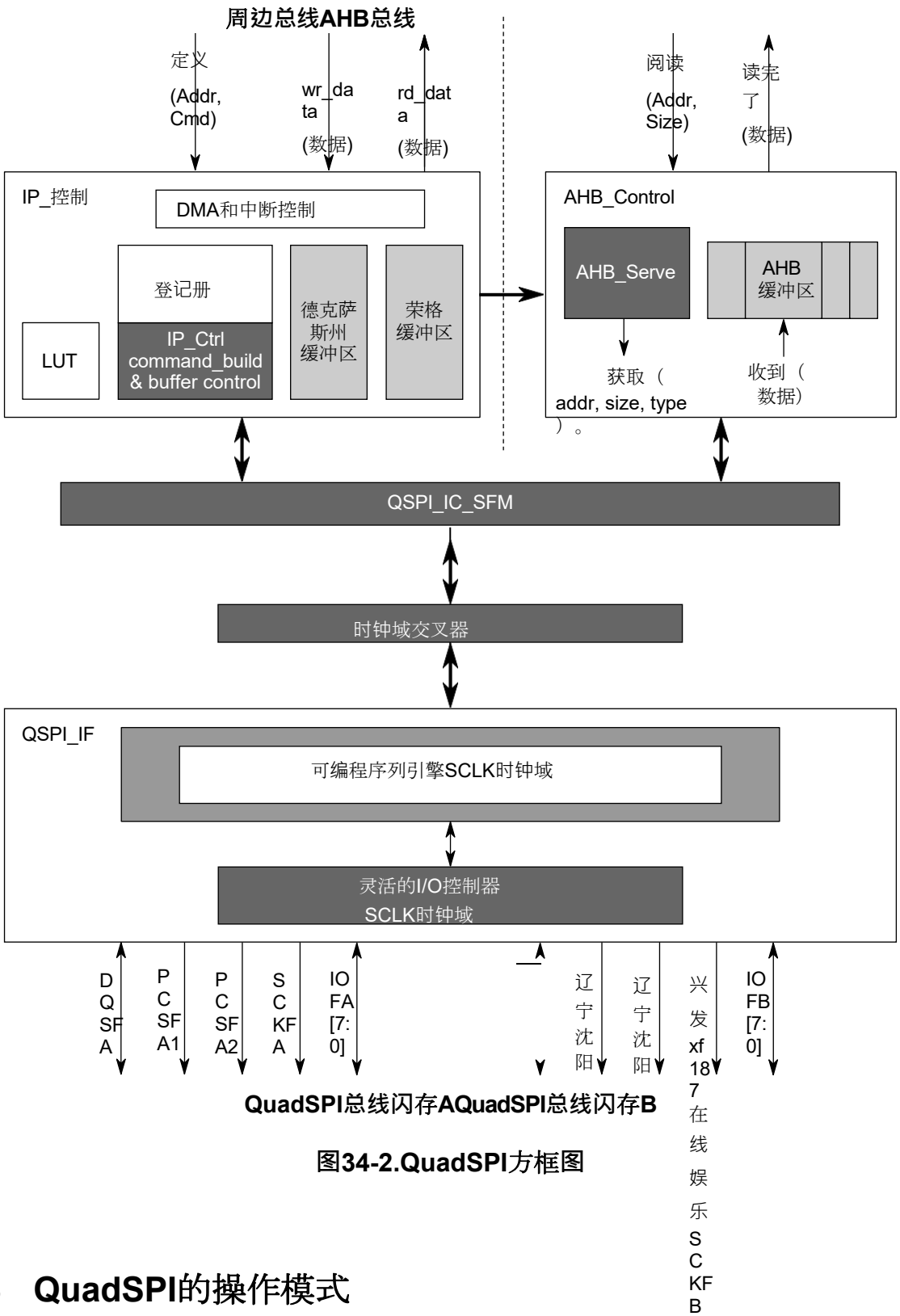



图34-2.QuadSPI方框图

## 34.2.3 QuadSPI的操作模式

为了通过IPS接口访问和正确的配置寄存器编程顺序进行电源管理，QuadSPI支持三种模式：正常、模块禁用和停止模式。

- 正常模式可用于对外部串行闪存设备进行写或读访问。
  - 串行闪存写入。数据只能通过IP接口被编程到闪存中。更多细节请参考闪存编程。
  - 串行闪存读取。读取串行闪存设备的内容。通过RX Buffer和AHB Buffer有两个独立的读取通道，见Flash Read。
- 停止模式。该模式用于电源管理。当提出进入停止模式的请求时，QuadSPI块确认该请求并完成正在进行的SFM命令，然后可以关闭通往QuadSPI块的系统时钟
- 模块停用模式。该模式用于电源管理。在模块禁用模式下，QuadSPI的非内存映射逻辑的时钟可以被停止。通过设置QSPI_MCR[MDIS]或当QSPI_MCR[DOZE]被设置时外部控制器发出请求，模块进入该模式。

### 34.2.4 首字母缩写词和缩略语

下表包含了本文件中使用的首字母缩写和缩略语。

**表34-4.首字母缩写词和缩略语**

| 条款 | 描述 |
|---|---|
| AHB | 先进的高性能总线，AMBA的版本 |
| AMBA | 先进的微控制器总线结构 |
| BE | 大尾数字节排序 |
| CS | 芯片选择 |
| DMA | 直接内存访问 |
| MB | 兆字节。每个MB是1024*1024字节 |
| IFM | 单独闪光模式 |
| LSB | 最小有效位 |
| MSB | 最显著位 |
| 辽宁省 | 周边芯片选择 |
| QSPI, QuadSPI | 四个串行外设接口 |
| 证券公司 | 串行通信时钟 |
| w1c | 写1来清除，向这个字段写1会重置标志。 |
| 输入/输出 | 输入输出。在本文件中，I/O线也被称为垫子。 |

## 34.2.5 QuadSPI模块的术语

表34-5.词汇表

| 术语 | 定义 |
| --- | --- |
| AHB命令 | AHB命令是由对属于表34-10中定义的内存映射访问的地址范围的读访问触发的SFM命令。详情请参考AHB命令。 |
| 已断言 | 一个被断言的信号处于其有效状态。一个有源低电平信号在断言时从逻辑电平1变为逻辑电平0，而一个有源高电平信号从逻辑电平0变为逻辑电平1。 |
| 清楚 | 清除一个位或多个位是指在该位或多个位上建立逻辑电平零。 |
| 时钟相位 | 决定何时相对于SCK的有效边沿对数据进行采样。 |
| 时钟极性 | 确定SCK信号的空闲状态。 |
| 沥水 | 通过软件或硬件从FIFO中删除条目。 |
| 内联性 | 字节排序方案。 |
| 场地 | 两个或多个寄存器位组合在一起。 |
| 填充 | 通过软件或硬件向FIFO添加条目。 |
| 宿主 | 指的是包含QuadSPI模块的设备中的另一个功能块 |
| 指示代码 | 8位，定义了要执行的命令的类型。 |
| IP指令 | IP命令是一个通过向QSPI_IPCR[SEQID]字段写入而触发的SFM命令。 |
| 逻辑电平一 | 与布尔真（1）状态相对应的电压。 |
| 逻辑电平为零 | 与布尔假（0）状态相对应的电压。 |
| 被否定 | 一个被否定的信号处于非活动状态。一个有效的低信号在被否定时从逻辑电平0变为逻辑电平1，一个有效的高信号从逻辑电平1变为逻辑电平0。 |
| QSPI_AMBA_BASE | 提交给QuadSPI控制器的串行闪存设备的第一个地址。这可能是系统地址图中串行闪存的基数，也可能是系统的重映射，例如重映射到0x0，由系统完成。请参考系统地址图。 |
| QSPI_ARDB_BASE | 系统内存地图上QuadSPI Rx Buffer的第一个地址。 |
| 设置 | 设置一个位或多个位意味着在该位或多个位上建立逻辑电平一。 |
| RX缓冲区PUSH事件 | 将有效条目添加到RX缓冲区。在默认情况下，每个Buffer PUSH事件在RX Buffer中增加2个条目，因为与串行时钟域的接口宽度为64位。根据从串行闪存设备读取的字节数，最后一个Buffer PUSH事件有可能只增加一个条目。<br>QSPI_RBSR[RDBFL]字段被添加到RX缓冲区的条目数量所增加。 |
| RX缓冲区POP事件 | 从RX缓冲区移除有效条目。每个缓冲区POP事件都会从缓冲区移除（QSPI_RBCT[WMRK] + 1）个有效条目。QSPI_RBSR[RDBFL]字段以相同的数字递减，QSPI_RBSR[RDCTR]字段相应递增。 |
| 单独闪光模式 | 访问一个单独的串行闪存设备。详情请参考《串行闪存访问方案》。 |
| SFM指令 | 串行闪存指令。一个SFM命令包括一个指令代码和该特定指令代码所需的所有其他参数（例如，大小或模式字节）。触发一个指令要么启动外部串行闪存上的事务，要么导致一个错误。关于错误的细节，请参考表34-19。 |

| | |
|---|---|
| 单/双/四/八指令 | 根据连接到QuadSPI模块的串行闪存设备，会有使用不同数量数据线的指令。<br>• 单垫。单线I/O，有一条数据输出线和一条数据输入线来/自串行闪存设备。<br>• 双垫。双线I/O有两个双向I/O线，由串行闪存交替驱动器件或QuadSPI模块 |

*表继续在下一页...*

表34-5.词汇表（续）

| 术语 | 定义 |
| --- | --- |
| | • Quad pad:四线I/O有4条双向I/O线，由串行闪存设备或QuadSPI交替驱动。<br>• 八进制垫。八进制线I/O，有8条双向I/O线，可由串行闪存设备或QuadSPI驱动。 |
| 交易 | 一个事务由所有的标志、数据和信号组成，在任何一个方向上都可以执行一个连接的串行闪存设备的命令。它是芯片选择、时钟、指令代码、地址、模式和/或虚拟字节、发送和/或接收数据的组合。 |
| LUT | 查询表。 |

## 34.3 外部信号描述

本节提供QuadSPI模块的外部信号信息。

下表列出了属于模块的外部信号与不同的操作模式。

表34-6.信号属性

| 信号名称 | 职能 | 方向 | 描述 |
| --- | --- | --- | --- |
| PCSFA1 | 外围芯片选择闪存A1 | O | 这个信号是串行闪存设备A1的芯片选择。A1代表两个共享IOFA的闪存器件中的第一个。 |
| PCSFA2 | 外围芯片选择闪存A2 | O | 这个信号是串行闪存设备A2的芯片选择。A2代表两个共享IOFA的闪存器件中的第二个。 |
| 辽宁沈阳 | 周边芯片选择 闪存B1 | O | 这个信号是串行闪存设备B1的芯片选择。B1代表两个共享IOFB的闪存器件中的第一个。 |
| 辽宁沈阳 | 外围芯片选择闪存B2 | O | 这个信号是串行闪存设备B2的芯片选择。B2代表两个共享IOFB的闪存器件中的第二个。 |
| SCKFA | 串行时钟闪存A | O | 这个信号是输出到串行闪存设备A的串行时钟。 |
| 兴发xf187在线娱乐SCKFB | 串行时钟闪存B | O | 该信号是输出给串行闪存设备B的串行时钟。 |
| IOFA[7:0] | 串行I/O闪存A | 输入/输出 | 这些信号是通往/来自串行闪存设备A的数据I/O线，关于信号驱动和时序行为的细节，请参考驱动外部信号。请注意，串行闪存设备的信号引脚可能会根据执行的SFM指令而改变其功能，在执行单指令和双指令时，它们会被留作控制输入。该模块支持将这些输入驱动到专用值。在单I/O模式下，QuadSPI驱动IOFA[0]上的数据，期望IOFA[1]上的数据。 |

| | | | |
|---|---|---|---|
| IOFB[7:0] | 串行I/O闪存B | 输入/输出 | 这些信号是通往/来自串行闪存设备B的数据I/O线，关于信号驱动和时序行为的细节，请参考驱动外部信号。请注意，串行闪存设备的信号引脚可能会根据执行的SFM指令而改变其功能，在执行单指令和双指令时，将其作为控制输入。该模块支持将这些输入驱动到专用值。在单I/O模式下，QuadSPI驱动IOFB[0]上的数据，期望IOFB[1]上的数据。 |

*表继续在下一页...*

表34-6.信号属性（续）

| 信号名称 | 职能 | 方向 | 描述 |
|---|---|---|---|
| DQSFA | 数据选通信号 Flash A | 输入/输出 | 一些闪存供应商提供了DQS信号，在DDR模式下，读取数据与之对齐。在写数据阶段，它也被作为一个输出信号提供。 |
| DQSFB | 数据选通信号 Flash B | 输入/输出 | 一些闪存供应商提供了DQS信号，在DDR模式下，读取数据与之对齐。在写数据阶段，它也被作为一个输出信号提供。 |

### 34.3.1 驱动外部信号

串行闪存访问方案的不同阶段如下图所示。

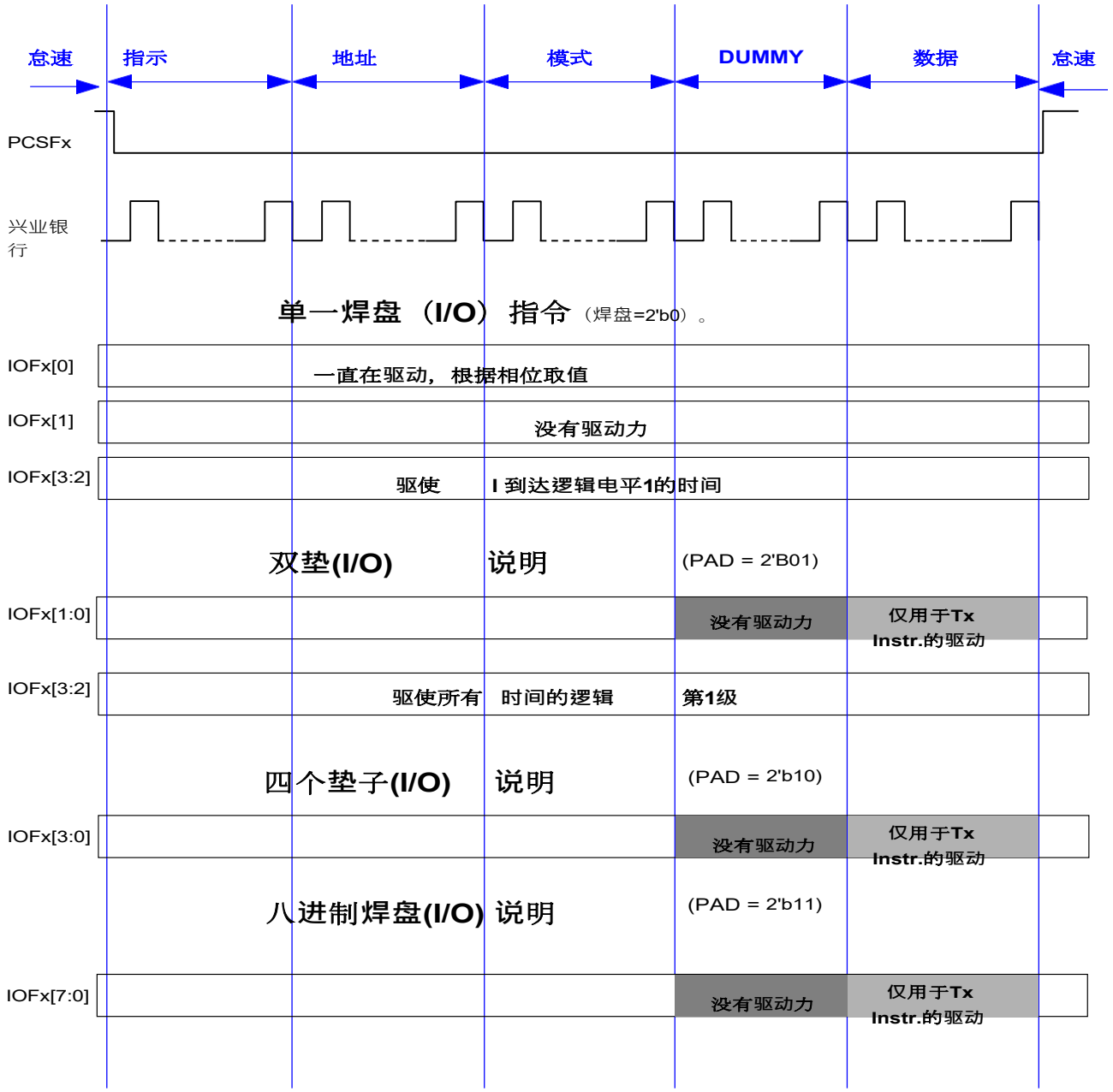



**图34-3.串行闪存访问方案**

QuadSPI模块的不同阶段和I/O驱动特性的特点如下。

- IDLE:
  未选择串行闪存设备。与串行闪存设备没有互动。所有IOFx信号被驱动。
- 指令：选择串行闪存设备。该指令被发送到串行闪存设备上。所有IOFx信号被驱动。

- ADDRESS:串行闪存地址被发送至设备。所有IOFx信号被驱动。注意，这个阶段并不适用于所有的SFM命令。
- MODE。模式字节被发送至串行闪存设备。所有IOFx信号被驱动。注意，这个阶段并不适用于所有的SFM命令。
- DUMMY：为串行闪存设备提供假时钟。关于驱动的IOFx信号，请参考图34-3。对于数据读取命令，执行SFM命令所需的实际数据线不被驱动。请注意，这一阶段并不适用于所有的SFM命令。
- 数据：串行闪存数据被发送到或从串行闪存设备接收。关于驱动的IOFx信号，请参考上图。对于数据读取命令，执行SFM命令所需的实际数据线不被驱动。注意，这个阶段并不适用于所有的SFM命令。

PCSFx和SCKFx信号在所有阶段都被永久驱动。在单独闪存模式下，这适用于选定的闪存设备。

## 34.4 内存地图和寄存器的定义

本节提供了QuadSPI模块的内存图和寄存器的定义。

### 34.4.1 寄存器写访问

本节描述了适用于所有寄存器的写访问限制条款，可以是以下的一种。

- **寄存器写访问限制**

  对于每个寄存器位和寄存器字段，写访问条件在详细的寄存器描述中规定。写入条件的描述在下表中给出。如果对于一个特定的寄存器位或字段，没有一个给定的写入条件被满足，任何对这个寄存器位或字段的写入尝试都会被忽略而没有任何通知。这些位或字段的值不会被改变。

  条件项[A或B]表示，如果至少有一个条件得到满足，就可以将寄存器或字段写入。

表34-7.寄存器写访问
限制条件

| 状况 | 描述 |
| --- | --- |
| 任何时候 | 没有写访问限制。 |
| 禁用模式 | 只有在*QSPI_MCR[MDIS]=1*的情况下才可以进行写操作。 |
| 正常模式 | 只有当模块处于*正常模式*时，才能进行写操作。 |

- **寄存器写访问要求**

  所有的寄存器都可以用8位、16位和32位宽的操作来访问。对于一些寄存器，至少需要16/32位宽的写访问以确保正确的操作。这种写访问要求在每个受影响的寄存器的详细描述中都有说明。

## 34.4.2 周边总线寄存器描述

本节提供了QuadSPI模块的内存图和寄存器的定义。

**注意事项**

对位置和0x190的访问不会出现传输错误。

**QuadSPI内存图**

| 地址偏移（十六进制） | 注册名称 | 宽度（单位：比特） | 访问 | 重置值 | 节/页 |
| --- | --- | --- | --- | --- | --- |
| 0 | 模块配置寄存器（QuadSPI_MCR）。 | 32 | R/W | 见本节 | 34.4.2.1/856 |
| 8 | IP配置寄存器（QuadSPI_IPCR）。 | 32 | R/W | 0000_0000h | 34.4.2.2/860 |
| C | 闪存配置寄存器（QuadSPI_FLSHCR）。 | 32 | R/W | 0000_0303h | 34.4.2.3/861 |
| 10 | 缓冲器0配置寄存器（QuadSPI_BUF0CR）。 | 32 | R/W | 见本节 | 34.4.2.4/862 |
| 14 | 缓冲器1配置寄存器（QuadSPI_BUF1CR）。 | 32 | R/W | 见本节 | 34.4.2.5/863 |
| 18 | 缓冲器2配置寄存器（QuadSPI_BUF2CR）。 | 32 | R/W | 见本节 | 34.4.2.6/863 |
| 1C | 缓冲器3配置寄存器（QuadSPI_BUF3CR）。 | 32 | R/W | 见本节 | 34.4.2.7/864 |

| | | | | | |
|---|---|---|---|---|---|
| 20 | 缓冲器通用配置寄存器（QuadSPI_BFGENCR）。 | 32 | R/W | 0000_0000h | 34.4.2.8/ 865 |

**QuadSPI内存图（续）**

*表继续在下一页...*

| 地址偏移（十六进制） | 注册名称 | 宽度（单位：比特） | 访问 | 重置值 | 节/页 |
|---|---|---|---|---|---|
| 24 | SOC配置寄存器（QuadSPI_SOCCR）。 | 32 | R/W | 0000_0000h | 34.4.2.9/866 |
| 30 | 缓冲区0顶部索引寄存器（QuadSPI_BUF0IND）。 | 32 | R/W | 0000_0000h | 34.4.2.10/867 |
| 34 | 缓冲区1顶部索引寄存器（QuadSPI_BUF1IND）。 | 32 | R/W | 0000_0000h | 34.4.2.11/867 |
| 38 | 缓冲区2顶部索引寄存器（QuadSPI_BUF2IND）。 | 32 | R/W | 0000_0000h | 34.4.2.12/868 |
| 100 | 串行闪存地址寄存器（QuadSPI_SFAR）。 | 32 | R/W | 0000_0000h | 34.4.2.13/869 |
| 104 | 串行闪存地址配置寄存器（QuadSPI_SFACR）。 | 32 | R/W | 0000_0000h | 34.4.2.14/869 |
| 108 | 采样寄存器（QuadSPI_SMPR）。 | 32 | R/W | 0000_0000h | 34.4.2.15/870 |
| 10C | RX缓冲器状态寄存器（QuadSPI_RBSR）。 | 32 | R | 0000_0000h | 34.4.2.16/872 |
| 110 | RX缓冲器控制寄存器（QuadSPI_RBCT）。 | 32 | R/W | 0000_0000h | 34.4.2.17/872 |
| 150 | TX缓冲器状态寄存器（QuadSPI_TBSR）。 | 32 | R | 0000_0000h | 34.4.2.18/874 |
| 154 | TX缓冲器数据寄存器（QuadSPI_TBDR）。 | 32 | R/W | 0000_0000h | 34.4.2.19/874 |
| 158 | Tx缓冲器控制寄存器（QuadSPI_TBCT）。 | 32 | R/W | 0000_0000h | 34.4.2.20/875 |
| 15C | 状态寄存器（QuadSPI_SR）。 | 32 | R | 0200_3800h | 34.4.2.21/876 |
| 160 | 标志寄存器（QuadSPI_FR）。 | 32 | w1c | 0800_0000h | 34.4.2.22/879 |
| 164 | 中断和DMA请求选择和启用寄存器（QuadSPI_RSER）。 | 32 | R/W | 0000_0000h | 34.4.2.23/882 |
| 168 | 序列暂停状态寄存器（QuadSPI_SPNDST）。 | 32 | R | 0000_0000h | 34.4.2.24/885 |
| 16C | 序列指针清除寄存器（QuadSPI_SPTRCLR）。 | 32 | R/W | 0000_0000h | 34.4.2.25/887 |
| 180 | 串行闪存A1顶部地址（QuadSPI_SFA1AD）。 | 32 | R/W | 见本节 | 34.4.2.26/888 |
| 184 | 串行闪存A2顶部地址（QuadSPI_SFA2AD）。 | 32 | R/W | 见本节 | 34.4.2.27/888 |
| 188 | 串行闪存B1顶部地址（QuadSPI_SFB1AD）。 | 32 | R/W | 见本节 | 34.4.2.28/889 |
| 18C | 串行闪存B2顶部地址(QuadSPI_SFB2AD) | 32 | R/W | 见本节 | 34.4.2.29/889 |
| 200 | RX缓冲器数据寄存器（QuadSPI_RBDR0）。 | 32 | R | 0000_0000h | 34.4.2.30/890 |

*表继续在下一页...*

## QuadSPI内存图（续）

| 地址偏移（十六进制） | 注册名称 | 宽度（单位：比特） | 访问 | 重置值 | 节/页 |
|---|---|---|---|---|---|
| 204 | RX缓冲器数据寄存器（QuadSPI_RBDR1）。 | 32 | R | 0000_0000h | 34.4.2.30/890 |
| 208 | RX缓冲器数据寄存器（QuadSPI_RBDR2）。 | 32 | R | 0000_0000h | 34.4.2.30/890 |
| 20C | RX缓冲器数据寄存器（QuadSPI_RBDR3）。 | 32 | R | 0000_0000h | 34.4.2.30/890 |
| 210 | RX缓冲器数据寄存器（QuadSPI_RBDR4）。 | 32 | R | 0000_0000h | 34.4.2.30/890 |
| 214 | RX缓冲器数据寄存器（QuadSPI_RBDR5）。 | 32 | R | 0000_0000h | 34.4.2.30/890 |
| 218 | RX缓冲器数据寄存器（QuadSPI_RBDR6）。 | 32 | R | 0000_0000h | 34.4.2.30/890 |
| 21C | RX缓冲器数据寄存器（QuadSPI_RBDR7）。 | 32 | R | 0000_0000h | 34.4.2.30/890 |
| 220 | RX缓冲器数据寄存器（QuadSPI_RBDR8）。 | 32 | R | 0000_0000h | 34.4.2.30/890 |
| 224 | RX缓冲器数据寄存器（QuadSPI_RBDR9）。 | 32 | R | 0000_0000h | 34.4.2.30/890 |
| 228 | RX缓冲器数据寄存器（QuadSPI_RBDR10）。 | 32 | R | 0000_0000h | 34.4.2.30/890 |
| 22C | RX缓冲器数据寄存器（QuadSPI_RBDR11）。 | 32 | R | 0000_0000h | 34.4.2.30/890 |
| 230 | RX缓冲器数据寄存器（QuadSPI_RBDR12）。 | 32 | R | 0000_0000h | 34.4.2.30/890 |
| 234 | RX缓冲器数据寄存器（QuadSPI_RBDR13）。 | 32 | R | 0000_0000h | 34.4.2.30/890 |
| 238 | RX缓冲器数据寄存器（QuadSPI_RBDR14）。 | 32 | R | 0000_0000h | 34.4.2.30/890 |
| 23C | RX缓冲器数据寄存器（QuadSPI_RBDR15）。 | 32 | R | 0000_0000h | 34.4.2.30/890 |
| 240 | RX缓冲器数据寄存器（QuadSPI_RBDR16）。 | 32 | R | 0000_0000h | 34.4.2.30/890 |
| 244 | RX缓冲器数据寄存器（QuadSPI_RBDR17）。 | 32 | R | 0000_0000h | 34.4.2.30/890 |
| 248 | RX缓冲器数据寄存器（QuadSPI_RBDR18）。 | 32 | R | 0000_0000h | 34.4.2.30/890 |
| 24C | RX缓冲器数据寄存器（QuadSPI_RBDR19）。 | 32 | R | 0000_0000h | 34.4.2.30/890 |
| 250 | RX缓冲器数据寄存器（QuadSPI_RBDR20）。 | 32 | R | 0000_0000h | 34.4.2.30/890 |
| 254 | RX缓冲器数据寄存器（QuadSPI_RBDR21）。 | 32 | R | 0000_0000h | 34.4.2.30/890 |
| 258 | RX缓冲器数据寄存器（QuadSPI_RBDR22）。 | 32 | R | 0000_0000h | 34.4.2.30/890 |

*表继续在下一页...*

## QuadSPI内存图（续）

| 地址偏移（十六进制） | 注册名称 | 宽度（单位：比特） | 访问 | 重置值 | 节/页 |
|---|---|---|---|---|---|
| 25C | RX缓冲器数据寄存器（QuadSPI_RBDR23）。 | 32 | R | 0000_0000h | 34.4.2.30/890 |
| 260 | RX缓冲器数据寄存器（QuadSPI_RBDR24）。 | 32 | R | 0000_0000h | 34.4.2.30/890 |
| 264 | RX缓冲器数据寄存器（QuadSPI_RBDR25）。 | 32 | R | 0000_0000h | 34.4.2.30/890 |
| 268 | RX缓冲器数据寄存器（QuadSPI_RBDR26）。 | 32 | R | 0000_0000h | 34.4.2.30/890 |
| 26C | RX缓冲器数据寄存器（QuadSPI_RBDR27）。 | 32 | R | 0000_0000h | 34.4.2.30/890 |
| 270 | RX缓冲器数据寄存器（QuadSPI_RBDR28）。 | 32 | R | 0000_0000h | 34.4.2.30/890 |
| 274 | RX缓冲器数据寄存器（QuadSPI_RBDR29）。 | 32 | R | 0000_0000h | 34.4.2.30/890 |
| 278 | RX缓冲器数据寄存器（QuadSPI_RBDR30）。 | 32 | R | 0000_0000h | 34.4.2.30/890 |
| 27C | RX缓冲器数据寄存器（QuadSPI_RBDR31）。 | 32 | R | 0000_0000h | 34.4.2.30/890 |
| 300 | LUT键寄存器（QuadSPI_LUTKEY）。 | 32 | R/W | 5AF0_5AF0h | 34.4.2.31/891 |
| 304 | LUT锁定配置寄存器（QuadSPI_LCKCR）。 | 32 | R/W | 0000_0002h | 34.4.2.32/891 |
| 310 | 查找表寄存器（QuadSPI_LUT0）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 314 | 查找表寄存器（QuadSPI_LUT1）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 318 | 查找表寄存器（QuadSPI_LUT2）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 31C | 查找表寄存器（QuadSPI_LUT3）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 320 | 查找表寄存器（QuadSPI_LUT4）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 324 | 查找表寄存器（QuadSPI_LUT5）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 328 | 查找表寄存器（QuadSPI_LUT6）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 32C | 查找表寄存器（QuadSPI_LUT7）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 330 | 查找表寄存器（QuadSPI_LUT8）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 334 | 查找表寄存器（QuadSPI_LUT9）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 338 | 查找表寄存器（QuadSPI_LUT10）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |

*表继续在下一页...*

## QuadSPI内存图（续）

| 地址偏移（十六进制） | 注册名称 | 宽度（单位：比特） | 访问 | 重置值 | 节/页 |
|---|---|---|---|---|---|
| 33C | 查找表寄存器（QuadSPI_LUT11）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 340 | 查找表寄存器（QuadSPI_LUT12）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 344 | 查找表寄存器（QuadSPI_LUT13）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 348 | 查找表寄存器（QuadSPI_LUT14）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 34C | 查找表寄存器（QuadSPI_LUT15）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 350 | 查找表寄存器（QuadSPI_LUT16）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 354 | 查找表寄存器（QuadSPI_LUT17）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 358 | 查找表寄存器（QuadSPI_LUT18）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 35C | 查找表寄存器（QuadSPI_LUT19）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 360 | 查找表寄存器（QuadSPI_LUT20）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 364 | 查找表寄存器（QuadSPI_LUT21）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 368 | 查找表寄存器（QuadSPI_LUT22）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 36C | 查找表寄存器（QuadSPI_LUT23）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 370 | 查找表寄存器（QuadSPI_LUT24）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 374 | 查找表寄存器（QuadSPI_LUT25）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 378 | 查找表寄存器（QuadSPI_LUT26）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 37C | 查找表寄存器（QuadSPI_LUT27）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 380 | 查找表寄存器（QuadSPI_LUT28）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 384 | 查找表寄存器（QuadSPI_LUT29）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 388 | 查找表寄存器（QuadSPI_LUT30）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 38C | 查找表寄存器（QuadSPI_LUT31）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 390 | 查找表寄存器（QuadSPI_LUT32）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |

*表继续在下一页...*

## QuadSPI内存图（续）

| 地址偏移（十六进制） | 注册名称 | 宽度（单位：比特） | 访问 | 重置值 | 节/页 |
|---|---|---|---|---|---|
| 394 | 查找表寄存器（QuadSPI_LUT33）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 398 | 查找表寄存器（QuadSPI_LUT34）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 39C | 查找表寄存器（QuadSPI_LUT35）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 3A0 | 查找表寄存器（QuadSPI_LUT36）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 3A4 | 查找表寄存器（QuadSPI_LUT37）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 3A8 | 查找表寄存器（QuadSPI_LUT38）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 3AC | 查找表寄存器（QuadSPI_LUT39）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 3B0 | 查找表寄存器（QuadSPI_LUT40）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 3B4 | 查找表寄存器（QuadSPI_LUT41）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 3B8 | 查找表寄存器（QuadSPI_LUT42）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 3BC | 查找表寄存器（QuadSPI_LUT43）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 3C0 | 查找表寄存器（QuadSPI_LUT44）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 3C4 | 查找表寄存器（QuadSPI_LUT45）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 3C8 | 查找表寄存器（QuadSPI_LUT46）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 3CC | 查找表寄存器（QuadSPI_LUT47）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 3D0 | 查找表寄存器（QuadSPI_LUT48）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 3D4 | 查找表寄存器（QuadSPI_LUT49）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 3D8 | 查找表寄存器（QuadSPI_LUT50）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 3DC | 查找表寄存器（QuadSPI_LUT51）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 3E0 | 查找表寄存器（QuadSPI_LUT52）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 3E4 | 查找表寄存器（QuadSPI_LUT53）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 3E8 | 查找表寄存器（QuadSPI_LUT54）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |

*表继续在下一页...*

QuadSPI内存图（续）

| 地址偏移（十六进制） | 注册名称 | 宽度（单位：比特） | 访问 | 重置值 | 节/页 |
|---|---|---|---|---|---|
| 3EC | 查找表寄存器（QuadSPI_LUT55）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 3F0 | 查找表寄存器（QuadSPI_LUT56）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 3F4 | 查找表寄存器（QuadSPI_LUT57）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 3F8 | 查找表寄存器（QuadSPI_LUT58）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 3FC | 查找表寄存器（QuadSPI_LUT59）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 400 | 查找表寄存器（QuadSPI_LUT60）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 404 | 查找表寄存器（QuadSPI_LUT61）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 408 | 查找表寄存器（QuadSPI_LUT62）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |
| 40C | 查找表寄存器（QuadSPI_LUT63）。 | 32 | R/W | 见本节 | 34.4.2.33/892 |

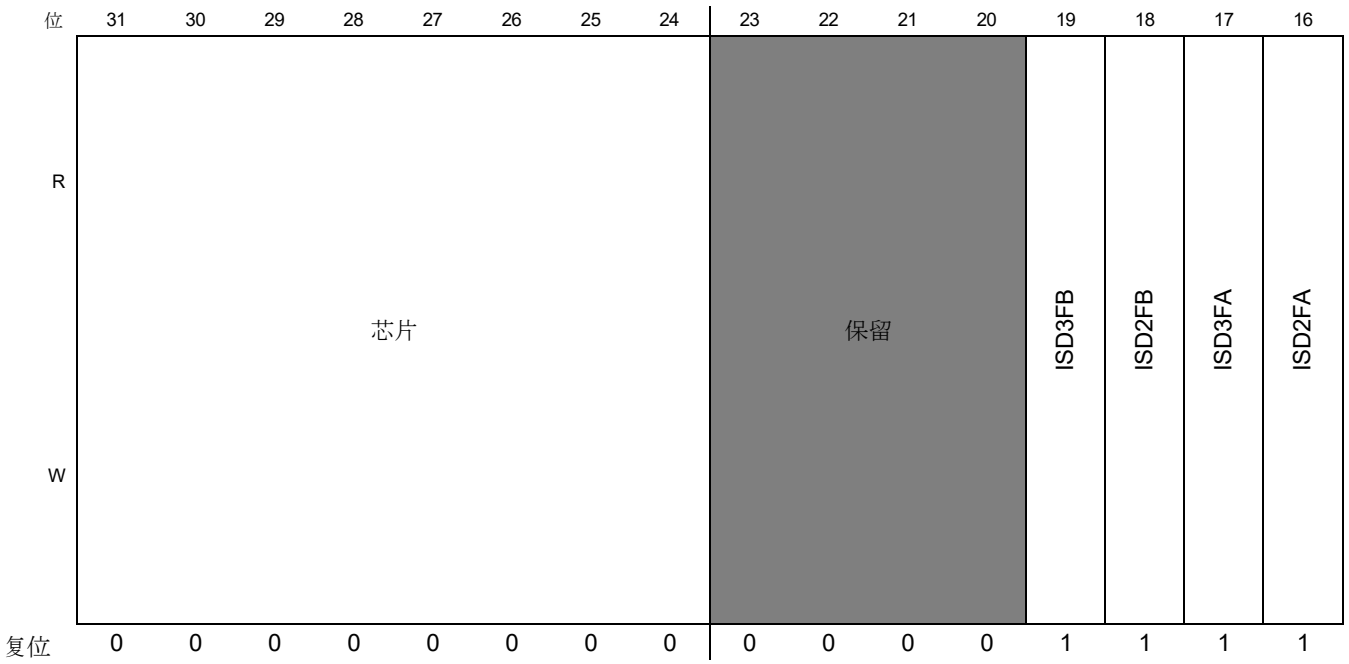
### 34.4.2.1 模块配置寄存器（QuadSPI_MCR）。

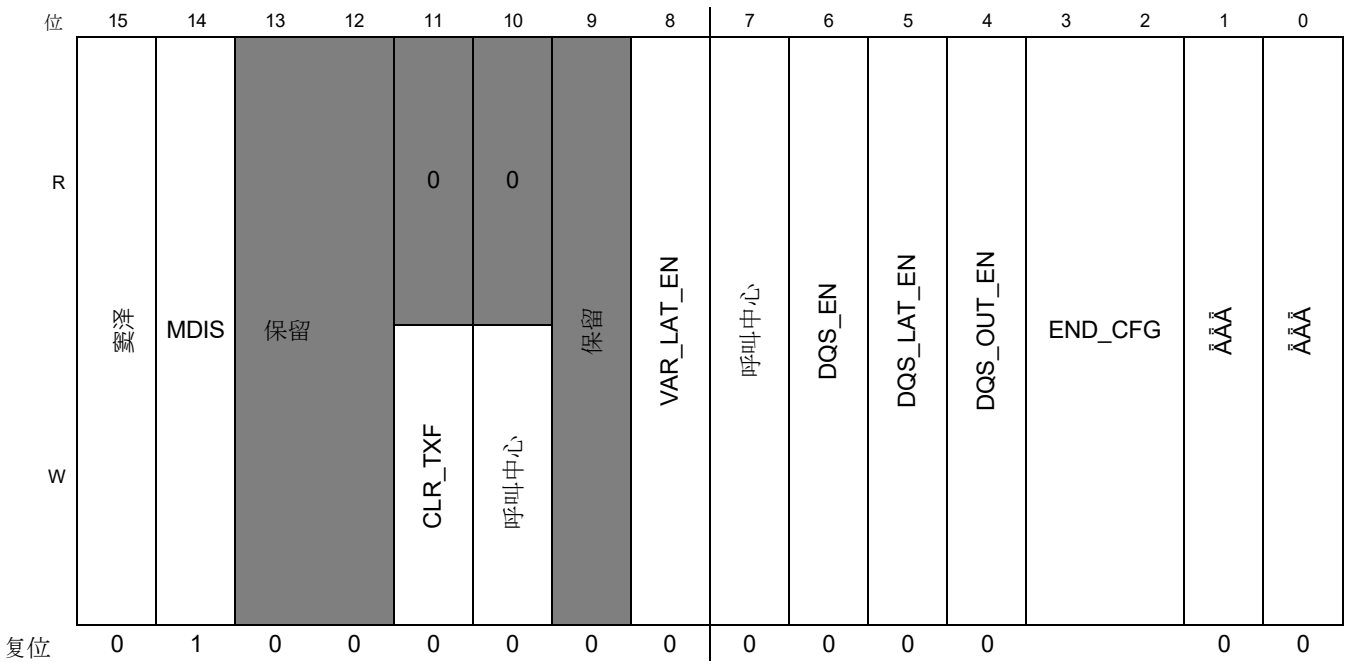
QuadSPI_MCR持有与QuadSPI操作相关的配置数据。

*写。*

- *SCLKCFG: 禁用模式*
- *ISD3FB, ISD2FB,ISD3FA, ISD2FA: 禁用模式*
- *所有其他领域。任何时候*

地址。0h base + 0h offset = 0h




* 注意事项。

- END_CFG字段。有关设备特定的复位值，请参见模块配置。

## QuadSPI_MCR字段描述

| 场地 | 描述 |
|---|---|
| 31-24<br>芯片 | 串行时钟配置。这个字段的配置是针对芯片的。详情请参考特定芯片的QuadSPI信息。它可用于划分时钟。 |

*表继续在下一页...*

QuadSPI_MCR字段描述（续）

| 场地 | 描述 |
|---|---|
| 23-20<br>保留 | 这个字段是保留的。 |
| 19<br>ISD3FB | 闲置信号驱动 IOFB[3] 闪存 B。该位决定了 QuadSPI 模块的 IOFB[3] 输出在非活动状态下的逻辑电平。请参考《在单、双模式下驱动闪存控制信号》。<br><br>0 IOFB[3]被驱动到逻辑L<br>1 IOFB[3]被驱动到逻辑H |
| 18<br>ISD2FB | 闲置信号驱动IOFB[2] 闪存B。该位决定了QuadSPI模块的IOFB[2]输出在非活动状态下被驱动到的逻辑电平。请参考《在单、双模式下驱动闪存控制信号》。<br><br>0 IOFB[2]被驱动到逻辑L<br>1 IOFB[2]被驱动到逻辑H |
| 17<br>ISD3FA | 闲置信号驱动 IOFA[3] 闪存 A. 该位决定了 QuadSPI 模块的 IOFA[3] 输出在非活动状态下被驱动到的逻辑电平。请参考《在单、双模式下驱动闪存控制信号》。<br><br>0 IOFA[3]被驱动到逻辑L<br>1 IOFA[3]被驱动到逻辑H |
| 16<br>ISD2FA | 闲置信号驱动 IOFA[2] 闪存 A. 该位决定了 QuadSPI 模块的 IOFA[2] 输出在非活动状态下被驱动到的逻辑电平。请参考《在单、双模式下驱动闪存控制信号》。<br><br>0 IOFA[2]被驱动到逻辑L<br>1 IOFA[2]被驱动到逻辑H |
| 15<br>斗泽 | 休眠启用。DOZE位提供了对外部控制的Doze模式省电机制的支持。<br><br>0 QuadSPI模块将忽略一个打盹请求。<br>1 暂停请求将由QuadSPI模块来处理 |
| 14<br>MDIS | 模块停用。MDIS位允许停止QuadSPI中非内存映射逻辑的时钟，使QuadSPI处于软件控制的省电状态。<br><br>0 启用QuadSPI时钟。<br>1 允许外部逻辑停用QuadSPI时钟。 |
| 13-12<br>保留 | 这个字段是保留的。 |
| 11<br>CLR_TXF | 清除TX FIFO/缓冲区。使TX缓冲区的内容无效。这是一个<br>自我清除的字段。<br><br>0 没有行动。<br>1 TX 缓冲器的读写指针被重置为 0。QSPI_TBSR[TRCTR]被重置为0。 |
| 10<br>呼叫中心 | 清除RX FIFO。使RX缓冲器失效。这是一个<br>自我清除的字段。<br><br>0 没有行动。 |

| | |
|---|---|
| | 1　RX 缓冲区的读和写指针被重置为 0。QSPI_RBSR[RDBFL]被重置为0。 |
| 9<br>保留 | 这个字段是保留的。 |
| 8<br>VAR_LAT_EN | 该字段用于在控制器中启用可变延迟功能。这个字段对HyperRAM有效，在读或写的命令和地址（CA）周期中，数据选通作为存储器的输出。 |

*表继续在下一页...*

QuadSPI_MCR字段描述（续）

QuadSPI_MCR字段描述（续）

| 场地 | 描述 |
|---|---|
| | 写交易，以表明是否需要额外的初始访问延迟来执行动态内存刷新操作。更多细节，请参考HyperRAM支持。<br>0 固定延时。两倍+1的延迟启用<br>1 可变延迟："一次 "或 "两次+1 "的初始延迟，基于CA阶段的数据选通。如果启用，还需要确保QSPI_FLSHCR[TCSS]必须>=2。 |
| 7<br>呼叫中心 | 启用DDR模式<br>0 2倍时钟仅对SDR指令无效<br>1 启用2倍时钟，支持SDR和DDR指令。 |
| 6<br>DQS_EN | DQS启用。这个字段对SDR和DDR模式都有效。更多细节请参考数据选通（DQS）采样方法。<br>0 DQS禁用。<br>1 DQS 启用。当启用时，当QSPI_MCR[DDR_EN]被设置时，传入的数据在DQS输入的两个边沿被采样，否则，当QSPI_MCR[DDR_EN]为0时，只在一个边沿被采样。 QSPI_SMPR[DDR_SMP]的值被忽略。 |
| 5<br>DQS_LAT_EN | DQS 延迟启用。当QSPI_MCR[DQS_EN]为1时，该字段在闪存的读访问之间包含延迟时有效。 更多细节请参考数据选通（DQS）采样方法。<br>0 禁用DQS延时<br>1 启用包含延迟的DQS功能 |
| 4<br>DQS_OUT_EN | 当Data Strobe在写数据阶段也被用作控制器的输出时，该字段有效。这对数据选通作为读写数据选通（RWDS）的HyperRAM有效。更多细节，请参考HyperRAM支持。<br>0 DQS作为控制器的一个输出被禁用<br>1 DQS作为控制器的一个输出被启用 |
| 3-2<br>END_CFG | 定义了QuadSPI模块的字节序数。更多细节请参考字节排序的端性 |
| 1<br>ÃÃÃ | AHB域的软件复位<br>0 没有行动<br>1 AHB域瓣被重置。不重置配置寄存器。<br>建议同时重置串行闪存域和AHB域。只重置一个域可能会导致副作用。<br>**注意：**软件复位需要时钟的运行来传播到设计中。因此，当软件复位位被断言时，MCR[MDIS]应该被设置为0。另外，在它们再次被解除指令之前（通过将MCR[SWRSTHD]设置为0），建议将MCR[MDIS]位设置为1。一旦软件复位被解除，通过将MCR[MDIS]位设置为0，就可以开始正常的操作。 |

| | |
|---|---|
| 0<br>ÃÃÃ | 串行闪存域的软件复位<br><br>0 没有行动<br>1 串行闪存域瓣被重置。不重置配置寄存器。<br>建议同时重置串行闪存域和AHB域。只重置一个域可能会导致副作用。 |

*表继续在下一页...*

**QuadSPI_MCR字段描述（续）**

| 场地 | 描述 |
|---|---|
| | **注意：**软件复位需要时钟的运行来传播到设计中。因此，当软件复位位被断言时，MCR[MDIS]应该被设置为0。另外，在它们再次被去势之前（通过设置MCR[SWRSTSD]为0），建议将MCR[MDIS]位设置为1。一旦软件复位被解除，通过将MCR[MDIS]位设置为0，就可以开始正常的操作。 |

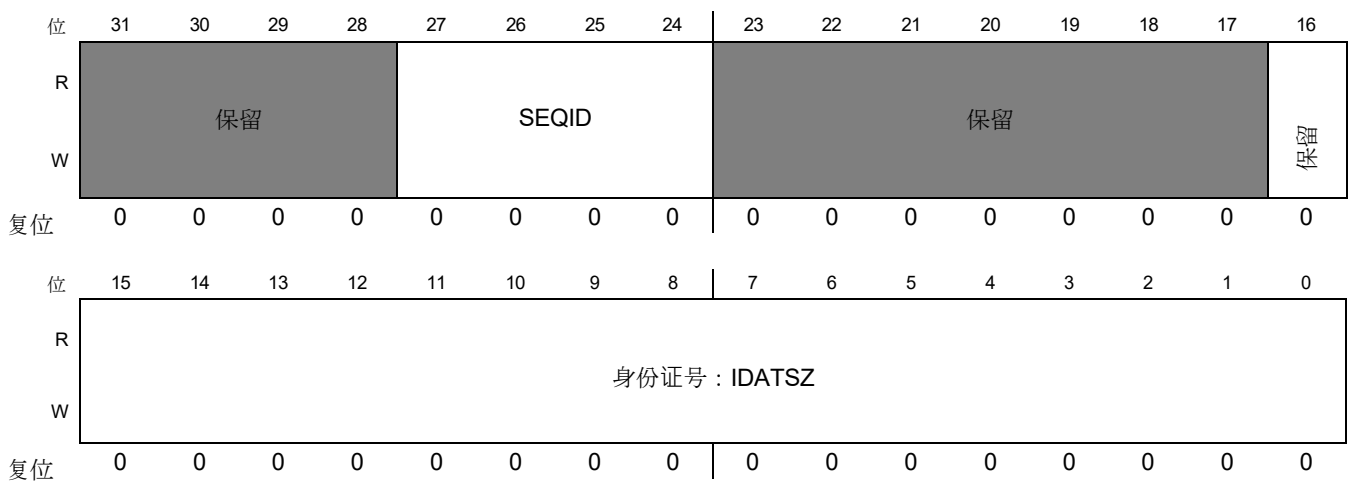
## 34.4.2.2 IP配置寄存器（QuadSPI_IPCR）。

IP配置寄存器提供了一个IP发起的命令所需的所有配置。一个IP命令可以通过在这个寄存器的SEQID字段中写入而被触发。如果SEQID字段被成功写入，一个新的命令将按照SEQID字段所指向的顺序开始向外部串行闪存发送。关于命令触发和命令执行的细节，请参考正常模式。

*写。*

- *qspi_sr[ip_acc]=0*

地址。0h基数+8h偏移量=8h




**QuadSPI_IPCR字段描述**

| 场地 | 描述 |
|---|---|
| 31-28<br>保留 | 这个字段是保留的。 |
| 27-24<br>SEQID | 指向查找表中的一个序列。这个字段定义了LUT索引的位[6:2]。位[1:0]总是被假定为0。 更多细节请参考查找表。对这个字段的写入会在串行闪存接口上触发一个事务。 |
| 23-17<br>保留 | 这个字段是保留的。 |
| 16<br>保留 | 这个字段是保留的。 |
| 身份证号： | IP数据传输大小。定义IP命令的数据传输大小，单位是字节。 |

QuadSPI_MCR字段描述（续）

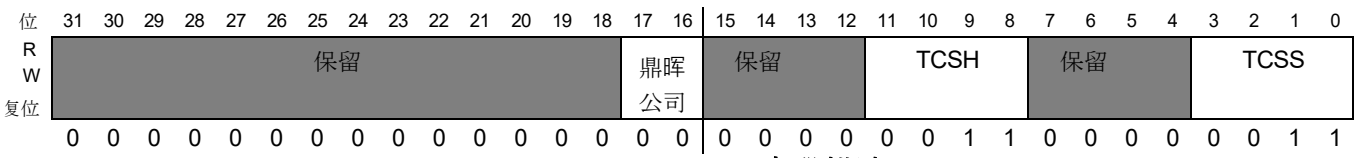
## 34.4.2.3 闪存配置寄存器（QuadSPI_FLSHCR）。

闪存配置寄存器包含闪存设备的特定时序，QuadSPI控制器必须满足这些时序才能使设备正常工作。

*写。*

- *qspi_sr[ahb_acc] = 0*
- *qspi_sr[ip_acc] = 0*

地址。0h基数+Ch偏移量=Ch




QuadSPI_FLSHCR字段描述

| 场地 | 描述 |
|---|---|
| 31-18<br>保留 | 这个字段是保留的。 |
| 17-16<br>鼎晖公司 | 串行闪存数据的保持时间。<br>这有助于满足闪存的数据保持时间要求。这只在DDR模式下有效。<br><br>**注意：**在SDR模式下，该字段应设置为0x00（QuadSPI_MCR[DDR_EN]=0）。详情请参考Flash的数据输入保持要求。<br><br>有效的TDH编程选项如下所示。其他组合是无效的。<br><br>00数据与QuadSPI的内部参考时钟的位置对齐 01数据 与2个串行闪存的半时钟对齐 |
| 15-12<br>保留 | 这个字段是保留的。 |
| 11-8<br>TCSH | 以串行闪存时钟周期计算的串行闪存CS保持时间。<br><br>**注意：**芯片选择和时钟之间的实际延迟被定义为。<br>• TCSH = 1 SCK clk 如果N= 0/1，否则，N SCK clk 如果N>1，其中N是TCSH的设置。 |
| 7-4<br>保留 | 这个字段是保留的。<br>保留。 |
| TCSS | 串行闪存的CS设置时间，以串行闪存时钟周期计算。<br><br>**注意事项** 1.芯片选择和时钟之间的实际延迟定义为：。<br>• 如果N=0/1，TCSS=0.5 SCK clk，否则，如果N>1，N+0.5 SCK clk，其中N是TCSS的设置。<br>2.对TCSS寄存器位的任何更新只在更新后的第二个交易中在闪存接口上可见。 |

### 34.4.2.4 缓冲器0配置寄存器（QuadSPI_BUF0CR）。

该寄存器为对缓冲区0的任何访问提供配置。当传入的AHB请求的主端口号与BUF0CR的MSTRID字段相匹配时，访问被路由到缓冲区0。任何缓冲区的 "缺失 "都会导致按照SEQID字段所指向的序列触发一个串行闪存交易。缓冲区0也可以通过设置该寄存器的HP_EN字段被配置为高优先级缓冲区。

*写。*

- *qspi_sr[ahb_acc] = 0*

地址。0h基数+10h偏移量=10h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R<br>W | HP_EN | 保留 | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 位 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R<br>W | 呼叫中心 | | | | | | | | 保留 | | | | 医学博士 | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | | | |

* 注意事项。

- MSTRID字段。重置值见模块配置。

**QuadSPI_BUF0CR字段描述**

| 场地 | 描述 |
|---|---|
| 31<br>HP_EN | 高优先级启用。当设置时，与该缓冲区相关的主站被分配了一个高于其他主站的优先级。高优先级主控器的访问将暂停另一个AHB主控器正在进行的预取，将以高优先级提供服务。详情请参考灵活的AHB缓冲区。 |
| 30-16<br>保留 | 这个字段是保留的。 |
| 15-8<br>呼叫中心 | AHB数据传输大小<br><br>定义AHB触发的对串行闪存的访问的数据传输大小（8字节）。例如，0x2的值将设置传输大小为16字节。当ADATSZ=0时，SEQID字段所指向的序列的数据大小会覆盖这个值。软件应确保该传输大小不大于该缓冲区的大小。 |
| 7-4<br>保留 | 这个字段是保留的。 |
| 医学博士 | 主站ID。与BUFFER0相关的AHB主站的ID。任何带有该主控端口号的AHB访问都被路由到该缓冲区。必须确保与所有缓冲区相关的主控ID必须是不同的。 |

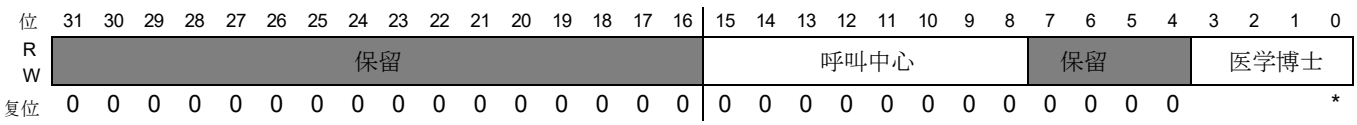
### 34.4.2.5 缓冲器1配置寄存器（QuadSPI_BUF1CR）。

该寄存器提供对缓冲区1的任何访问的配置。当传入的AHB请求的主端口号与BUF1CR的MSTRID字段相匹配时，访问被路由到缓冲区1。任何缓冲区的 "错过 "都会导致缓冲区被刷新，并按照SEQID字段所指向的序列触发一个串行闪存交易。

*写。*

- *qspi_sr[ahb_acc] = 0*

地址。0h基数+14h偏移量=14h




* 注意事项。

- MSTRID字段。重置值见模块配置。

**QuadSPI_BUF1CR字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。 |
| 15-8<br>呼叫中心<br>(ADATSZ) | AHB数据传输大小<br>定义AHB触发的对串行闪存的访问的数据传输大小（8字节）。例如，0x2的值将设置传输大小为16字节。当ADATSZ=0时，SEQID字段所提到的序列的数据大小将覆盖这个值。软件应确保该传输大小不大于该缓冲区的大小。 |
| 7-4<br>保留 | 这个字段是保留的。 |
| 医学博士（<br>MSTRID | 主站ID。与BUFFER1相关的AHB主站的ID。任何带有该主控端口号的AHB访问都被路由到该缓冲区。必须确保与所有缓冲区相关的主站ID必须是不同的。 |

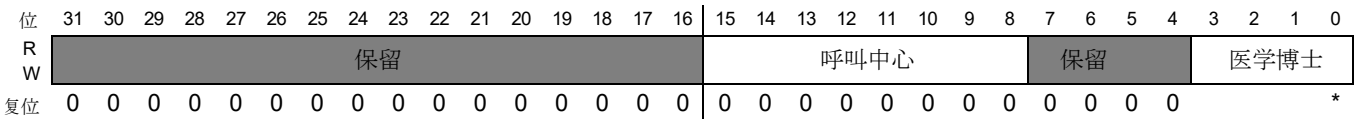
### 34.4.2.6 缓冲器2配置寄存器（QuadSPI_BUF2CR）。

该寄存器提供对缓冲区2的任何访问的配置。当传入的AHB请求的主端口号与BUF2CR的MSTRID字段相匹配时，访问将被路由到缓冲区2。任何缓冲区的 "错过 "都会导致缓冲区被刷新，并按照SEQID字段所指向的序列触发一个串行闪存交易。

*写。*

- *qspi_sr[ahb_acc] = 0*

地址。0h基数+18h偏移量=18h




* 注意事项。

- MSTRID字段。参见模块配置中的设备特定复位值。

**QuadSPI_BUF2CR字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>保留。 |
| 15-8<br>呼叫中心<br>(ADATSZ) | AHB数据传输大小<br><br>定义AHB触发的对串行闪存的访问的数据传输大小（8字节）。例如，0x2的值将设置传输大小为16字节。<br>当ADATSZ=0时，SEQID字段所提到的序列的数据大小将覆盖这个值。软件应确保该传输大小不大于该缓冲区的大小。 |
| 7-4<br>保留 | 这个字段是保留的。<br>保留。 |
| 医学博士 | 主站ID。与BUFFER2相关的AHB主站的ID。任何带有该主控端口号的AHB访问都被路由到该缓冲区。必须确保与所有缓冲区相关的主站ID必须是不同的。 |

## 34.4.2.7 缓冲器3配置寄存器（QuadSPI_BUF3CR）。

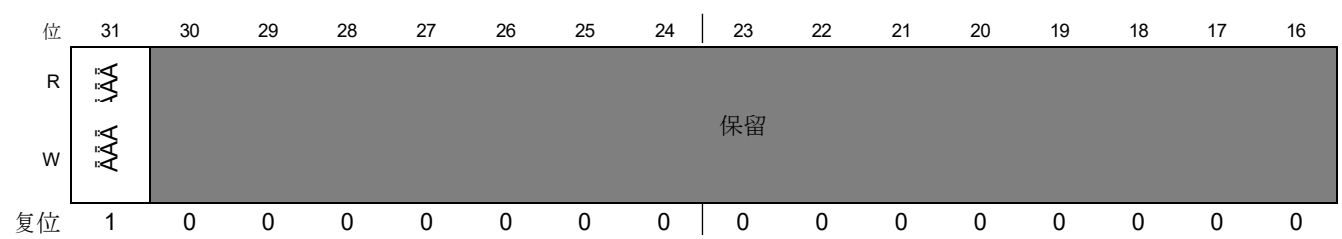
该寄存器提供对缓冲区3的任何访问的配置。当传入的AHB请求的主端口号与BUF3CR的MSTRID字段相匹配时，访问被路由到缓冲区3。任何缓冲区的 "缺失 "都会导致缓冲区被刷新，并根据SEQID字段所指向的序列触发串行闪存交易。

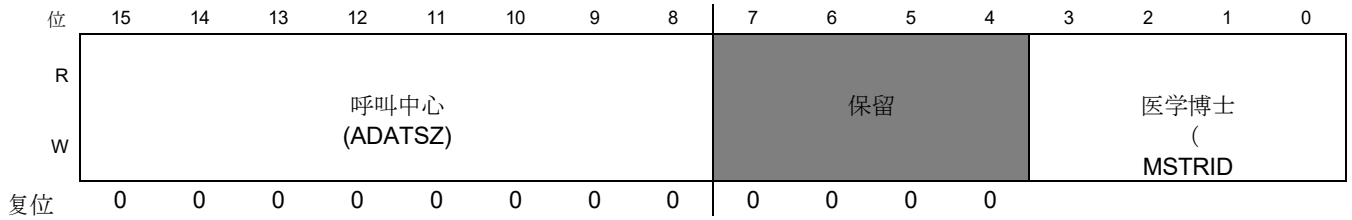
如果没有设置ALLMST字段，任何这样的交易（主端口号与MSTRID字段不匹配）将被返回一个ERROR响应。

*写。*

- *qspi_sr[ahb_acc] = 0*

地址。0h基数+1Ch偏移量=1Ch






* 注意事项。

- MSTRID字段。参见模块配置中的设备特定复位值。

**QuadSPI_BUF3CR字段描述**

| 场地 | 描述 |
|---|---|
| 31<br>辽宁省 | 全主控启用。当设置时，缓冲区3作为一个全主控缓冲区。任何主控端口号与缓冲区0或缓冲区1或缓冲区2的主控ID不匹配的AHB访问都被路由到缓冲区3。当设置时，该寄存器的MSTRID字段被忽略。 |
| 30-16<br>保留 | 这个字段是保留的。<br>保留。 |
| 15-8<br>呼叫中心 | AHB数据传输大小<br><br>定义AHB触发的对串行闪存访问的数据传输大小，单位为8字节。当ADATSZ=0时，SEQID字段所指向的序列的数据大小会覆盖这个值。软件应确保该传输大小不大于该缓冲区的大小。 |
| 7-4<br>保留 | 这个字段是保留的。 |
| 医学博士 | 主站ID。与BUFFER3相关的AHB主站的ID。任何带有该主端口号的AHB访问都会被路由到该缓冲区。必须确保与所有缓冲区相关的主站ID必须是不同的。 |

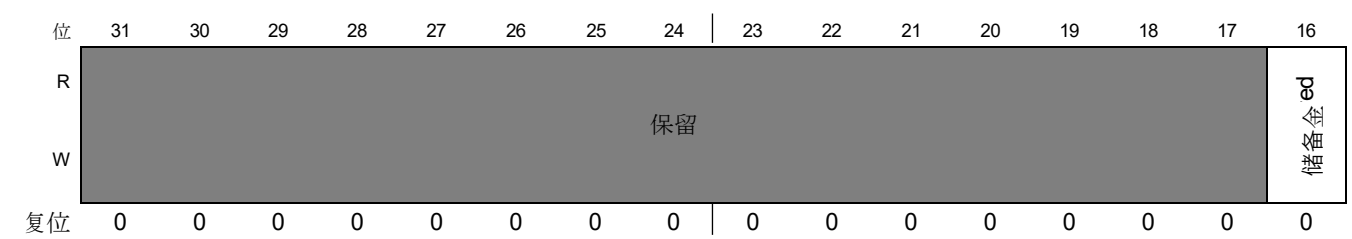
## 34.4.2.8 缓冲器通用配置寄存器（QuadSPI_BFGENCR）。

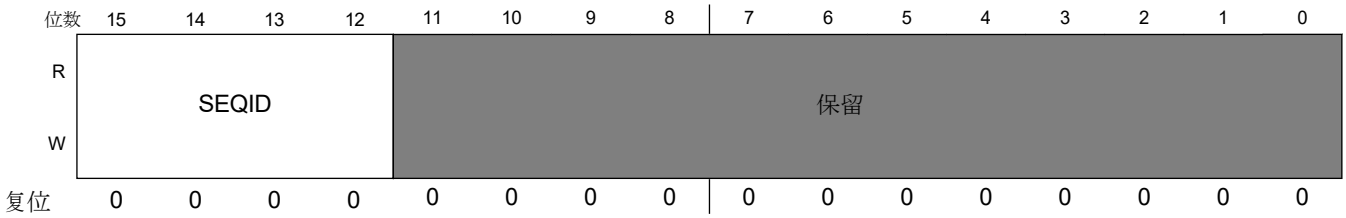
这个寄存器提供了对任何缓冲区访问的通用配置。任何缓冲区的 "错过
"都会导致缓冲区被刷新，并根据SEQID字段所指向的序列触发一个串行闪存交易
。

*写。*

- *qspi_sr[ahb_acc] = 0*

地址。0h基数+20h偏移量=20h






**QuadSPI_BFGENCR字段描述**

| 场地 | 描述 |
|---|---|
| 31-17<br>保留 | 这个字段是保留的。 |
| 16<br>保留 | 这个字段是保留的。 |
| 15-12<br>SEQID | 指向查找表中的一个序列。SEQID定义了LUT索引的位[6:2]。位[1:0]总是被假定为0。 请参考查找表。<br><br>**注意：**如果新的序列和上一个序列之间的序列指针不同，那么用户应该重新设置。更多信息请参见QSPI_SPTRCLR。 |
| 保留 | 这个字段是保留的。 |

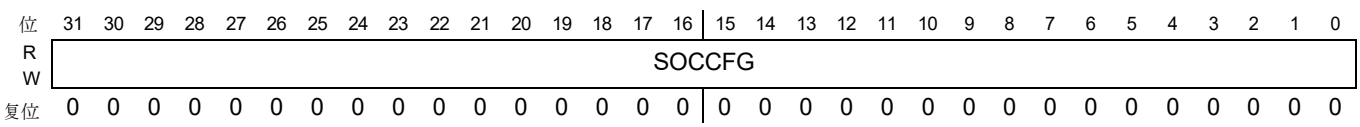
## 34.4.2.9 SOC配置寄存器（QuadSPI_SOCCR）。

该寄存器在芯片级编程，用于QuadSPI延迟链配置。详情请参考特定芯片的QuadSPI信息。

*写。*

- *qspi_sr[ahb_acc] = 0*

地址。0h基数+24h偏移量=24h




**QuadSPI_SOCCR字段描述**

| 场地 | 描述 |
|---|---|
| SOCCFG | SOC配置<br><br>详情请参考特定芯片的QuadSPI信息。 |

### 34.4.2.10 缓冲区0顶部索引寄存器（QuadSPI_BUF0IND）。

这个寄存器指定了缓冲区0的顶部索引，它定义了缓冲区的大小。请注意，这个寄存器的3个LSB被设置为0。这确保了缓冲区是64位对齐的，因为每个缓冲区条目都是64位长。

寄存器的值应被设置为所需的字节数。例如，将BUF0IND[31:3]设置为0可以得到0个字节，1可以得到8个字节等。

缓冲区0的大小是BUF0IND和0之间的差。

软件应确保BUF0IND值不大于缓冲区的总体大小。硬件不提供任何防止非法编程的保护。

*写。*

- *qspi_sr[ahb_acc] = 0*

地址。0h基数+30h偏移量=30h

| 位 | 31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 | 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0 |
|---|---|---|
| R<br>W | TPINDX0 | 保留 |
| 复位 | 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 | 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 |

**QuadSPI_BUF0IND字段描述**

| 场地 | 描述 |
|---|---|
| 31-3<br>TPINDX0 | 缓冲区0的顶部索引。 |
| 保留 | 这个字段是保留的。<br>保留。 |

### 34.4.2.11 缓冲区1顶部索引寄存器（QuadSPI_BUF1IND）。

这个寄存器指定了缓冲区1的顶部索引，它定义了缓冲区的大小。请注意，这个寄存器的3个LSB被设置为0。这确保了缓冲区是64位对齐的，因为每个缓冲区条目都是64位长。

缓冲区1的大小是BUF1IND和BUF0IND之差。寄存器的值应以字节为单位输入。例如，如果BUF0IND = 0x100，那么设置BUF1IND = 0x130将把缓冲区1的大小设置为0x30字节。

软件有责任确保BUF1IND值不大于缓冲区的总体大小。硬件不提供任何防止非法编程的保护。

*写。*

- *qspi_sr[ahb_acc] = 0*

地址。0h基数+34h偏移量=34h




**QuadSPI_BUF1IND字段描述**

| 场地 | 描述 |
|---|---|
| 31-3<br>辽宁沈阳 | 缓冲区1的顶部索引。 |
| 保留 | 这个字段是保留的。 |

## 34.4.2.12 缓冲区2顶部索引寄存器（QuadSPI_BUF2IND）。

这个寄存器指定了缓冲区2的顶部索引，它定义了缓冲区的大小。请注意，这个寄存器的3个LSB被设置为0。这确保了缓冲区是64位对齐的，因为每个缓冲区条目都是64位长。

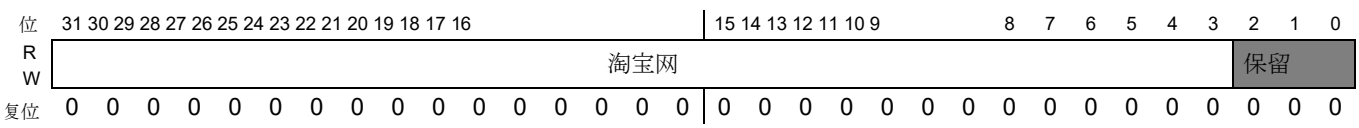
缓冲区2的大小是BUF2IND和BUF1IND之差。寄存器的值应以字节为单位输入。例如，如果BUF1IND = 0x130，那么设置BUF2IND = 0x180将把缓冲区2的大小设置为0x50字节。

软件有责任确保BUF2IND值不大于缓冲区的总体大小。硬件不提供任何防止非法编程的保护。

*写。*

- *qspi_sr[ahb_acc] = 0*

地址。0h基数+38h偏移量=38h




**QuadSPI_BUF2IND字段描述**

| 场地 | 描述 |
|---|---|
| 31-3<br>淘宝网 | 缓冲区的顶部索引 2. |
| 保留 | 这个字段是保留的。 |

### 34.4.2.13 串行闪存地址寄存器（QuadSPI_SFAR）。

模块会自动将内存地图上的这个地址转换为闪存本身的地址。当工作在24位模式时，只有第23-
0位被发送到闪存中。在32位模式下，当QSPI_SFACR[CAS]设置为0时，位27-
0被使用，位31-28被驱动为0。
例如，如果QSPI_SFACR[CAS]为3，那么在闪存工作在24位模式下，位26-
3被发送到闪存作为其页地址。闪存器请求作为页和列地址的地址位总数不能超过32位。关于访问模式和QSPI_SFAR内容之间的映射，请参考表34-
8，关于命令触发和命令执行的细节请参考正常模式。软件应该确保QSPI_SFAR寄存器中提供的串行闪存地址位于表34-8中定义的有效闪存地址范围内。

*写。*

- *qspi_sr[ip_acc] = 0*

地址。0h基数+100h偏移量=100h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R<br>W | 证券投资基金（SFADR | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**QuadSPI_SFAR字段描述**

| 场地 | 描述 |
|---|---|
| 辽宁沈阳 | 串行闪存地址。该寄存器的内容被用作以下所有IP命令的字节地址。 |

### 34.4.2.14 串行闪存地址配置寄存器（QuadSPI_SFACR）。

这个寄存器包含了串行闪存的具体地址要求，必须根据所连接的闪存进行配置，以使控制器正常工作。模块自动将内存地图上的地址QSPI_SFAR或AHB总线上的输入地址转换为闪存本身的列地址。假设一个闪存需要3位作为它的列地址，那么只有QSPI_SFAR/AHB地址的低3位作为它的列地址被发送到闪存上。软件应该确保QSPI_SFAR寄存器中提供的串行闪存地址或传入的AHB地址位于有效的闪存地址范围内。

*写。*

- *qspi_sr[ip_acc] = 0*
- *qspi_sr[ahb_acc] = 0*

地址。0h基数+104h偏移量=104h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R<br>W | 保留 | | | | | | | | | | | | | | | WA |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 位 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R<br>W | 保留 | | | | | | | | | | | | 中科院 | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**QuadSPI_SFACR字段描述**

| 场地 | 描述 |
|---|---|
| 31-17<br>保留 | 这个字段是保留的。 |
| 16<br>WA | 可寻址字数<br><br>定义串行闪存是字节寻址的闪存还是字寻址的闪存。根据这个位的配置，地址被重新映射到闪存接口上。详情请参考地址方案。<br><br>0 字节可寻址的串行闪存模式。<br>1 字（2字节）可寻址的串行闪存模式。 |
| 15-4<br>保留 | 这个字段是保留的。 |
| 中科院 | 专栏地址空间<br><br>定义了列地址的宽度。如果列地址是QSPI_SFAR/AHB地址的[2:0]，那么CAS必须是3。如果在任何串行闪存中没有列地址的分隔，那么这个位必须被编程为0。 |

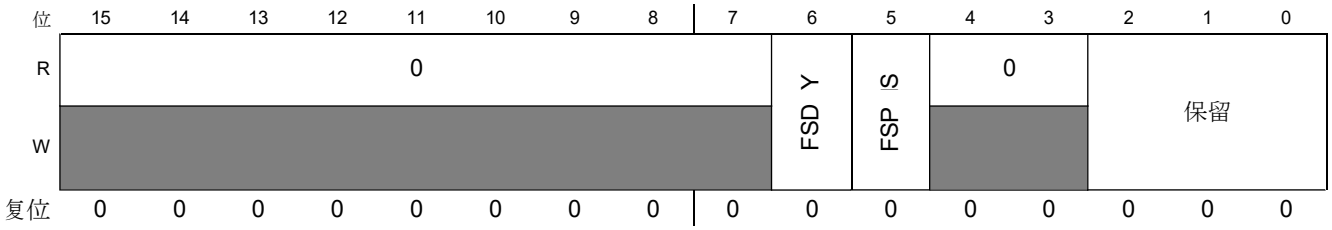
## 34.4.2.15 采样寄存器（QuadSPI_SMPR）。

采样寄存器允许配置如何在QuadSPI模块中对来自外部串行闪存设备的传入数据进行采样。

*写。禁用模式*

地址。0h base + 108h offset = 108h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | 保留 | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |



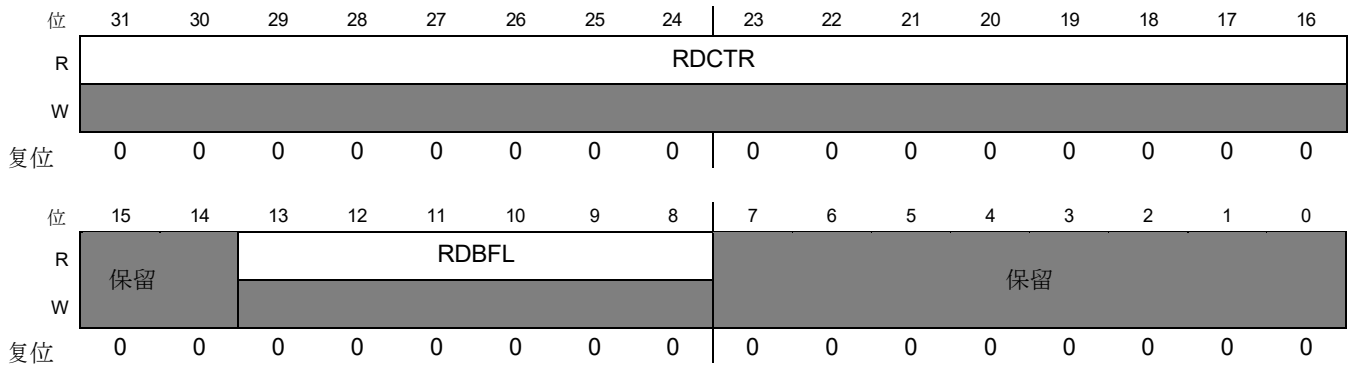

## QuadSPI_SMPR字段描述

| 场地 | 描述 |
|---|---|
| 31-19<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 18-16<br>保留 | 这个字段是保留的。 |
| 15-7<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 6<br>FSDLY | SDR指令的全速延时选择。选择相对于全速指令有效的采样点参考边缘的延迟。<br><br>0 一个时钟周期的延迟<br>1 两个时钟周期的延迟。 |
| 5<br>FSPHS | SDR指令的全速相位选择。<br><br>选择对全速命令有效的采样时钟的边缘。<br><br>0 选择在非反相时钟下采样<br>1 选择倒置时钟下的采样。 |
| 4-3<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 保留 | 这个字段是保留的。 |

## 34.4.2.16 RX缓冲器状态寄存器（QuadSPI_RBSR）。

该寄存器包含与接收数据缓冲区有关的信息。

地址。0h基数+10Ch偏移量=10Ch




**QuadSPI_RBSR字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>RDCTR | 读取计数器。表示有多少个4字节的条目被从RX缓冲区移除。例如，一个0x2的值表示有8个字节被移除。<br>在RX Buffer POP事件中，它被数字（QSPI_RBCT[WMRK]+1）递增。RX 缓冲区可以通过 DMA 或弹出标志 QSPI_FR[RBDF]来弹出。QSPI_RSER[RBDDE]定义了必须要做的弹出。更多细节请参考AHB RX数据缓冲区（QSPI_ARDB0至QSPI_ARDB31）和Flash阅读中的 "从QuadSPI模块内部缓冲区传输数据 "部分。 |
| 15-14<br>保留 | 这个字段是保留的。 |
| 13-8<br>RDBFL | RX Buffer Fill Level.表示RX缓冲区中还有多少个4字节的条目可用。例如，一个0x2的值表示有8个字节可用。 |
| 保留 | 这个字段是保留的。 |

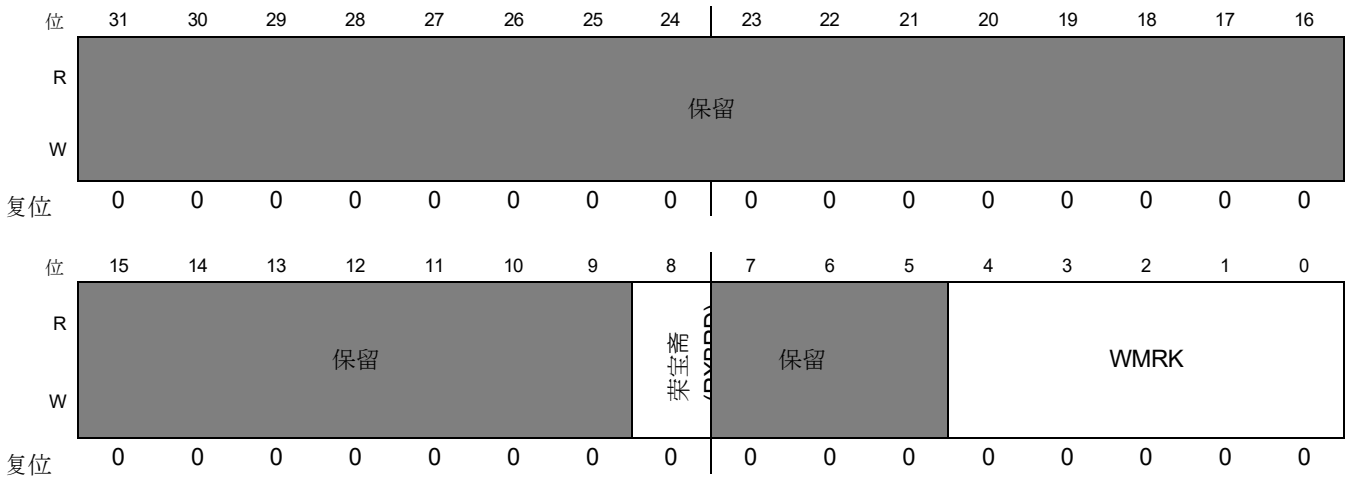
## 34.4.2.17 RX缓冲器控制寄存器（QuadSPI_RBCT）。

该寄存器包含与接收数据缓冲区有关的控制数据。

*写。*

- *qspi_sr[ip_acc] = 0*

地址。0h base + 110h offset = 110h



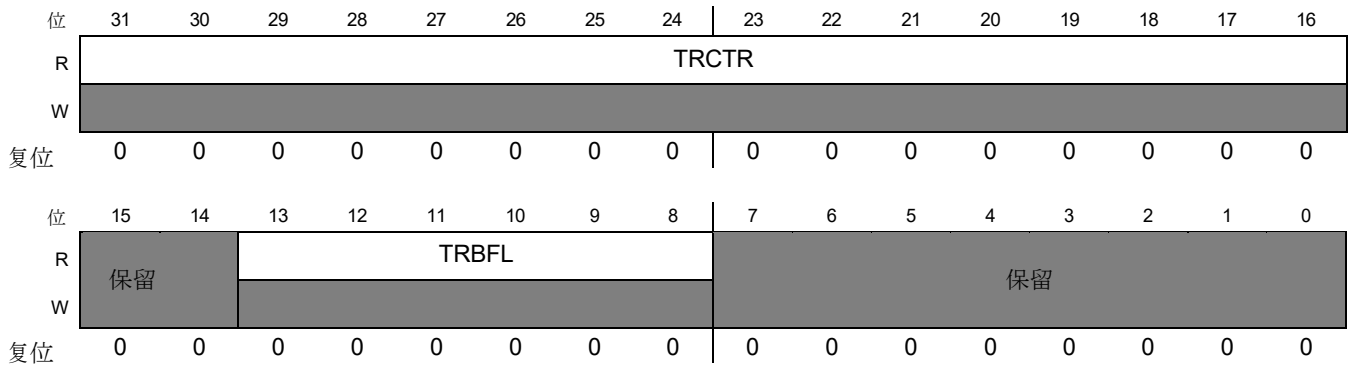

## QuadSPI_RBCT字段描述

| 场地 | 描述 |
|---|---|
| 31-9<br>保留 | 这个字段是保留的。 |
| 8<br>RXBRD | RX Buffer Readout.该字段指定了RX缓冲区读出的访问方案。<br>0 使用AHB总线寄存器QSPI_ARDB0至QSPI_ARDB31读取RX缓冲区内容。详情请参考《AHB命令对串行闪存的独占访问》。<br>1 使用IP总线寄存器QSPI_RBDR0至QSPI_RBDR31读取RX缓冲区内容。 |
| 7-5<br>保留 | 这个字段是保留的。 |
| WMRK | RX缓冲器水印。这个字段决定了何时触发RX缓冲器的读出动作。当RX缓冲区的有效条目数等于或大于(WMRK+1)给出的数字时，QSPI_SR[RXWE]标志被断言。该值应被输入为4字节条目数减去1。例如，0x0的值将设置水印为4字节，1为8字节，2为12字节，以此类推。<br>详情请参考《DMA使用》。 |

## 34.4.2.18 TX缓冲器状态寄存器（QuadSPI_TBSR）。

该寄存器包含与发送数据缓冲区有关的信息。

地址。0h基数+150h偏移量=150h




**QuadSPI_TBSR字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>TRCTR | 传输计数器。这个字段表示有多少个4字节的条目被主机访问写入了TX缓冲区。当1被写入QSPI_MCR[CLR_TXF]时，它被重置为0。当另一个字被推入 TX 缓冲区时，每次对 QSPI_TBDR 寄存器的写操作都会使它增量。当它没有被清除时，TRCTR 字段就会环绕到 0。 详情请参考 TX 缓冲区数据寄存器（QuadSPI_TBDR）。 |
| 15-14<br>保留 | 这个字段是保留的。 |
| 13-8<br>TRBFL | TX Buffer Fill Level.TRBFL字段包含TX缓冲区中可用的条目数，每个条目4字节，供QuadSPI模块传输到串行闪存设备。 |
| 保留 | 这个字段是保留的。 |

## 34.4.2.19 TX缓冲器数据寄存器（QuadSPI_TBDR）。

QSPI_TBDR寄存器提供了对深度为128字节的循环TX缓冲器的访问。这个缓冲区提供了写入它的数据，作为对串行闪存设备的页面编程命令的写入数据。字节排序方案请参考表34-16。在闪存上进行数据大小小于32位的写操作，将导致从TX缓冲区中删除四个数据条目。有效位将被使用，其余的位将被丢弃。

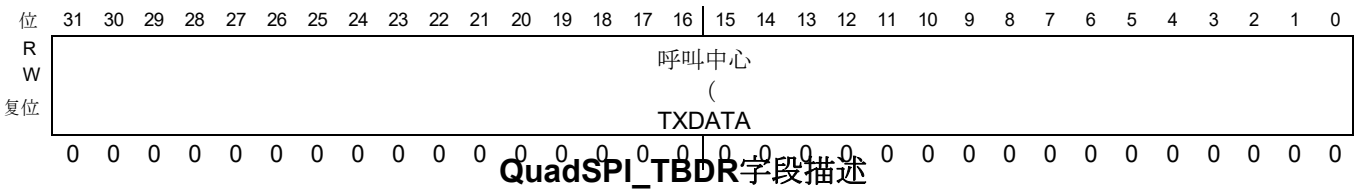
*写。*

- *qspi_sr[txfull] = 0*

需要32位写访问

寄存器 0h base + 110h offset = 110h

地址。0h base + 154h offset = 154h



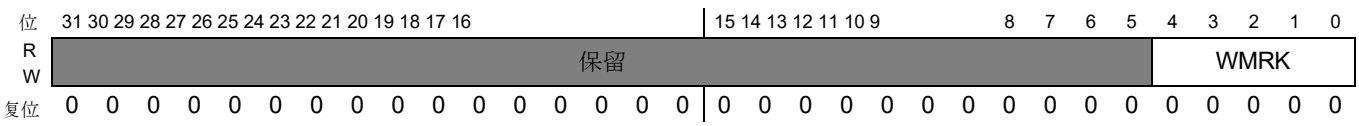

**QuadSPI_TBDR字段描述**

| 场地 | 描述 |
|---|---|
| 呼叫中心（ TXDATA | TX数据<br>写入时，数据被写入TX缓冲区的下一个可用条目，QPSI_TBSR[TRBFL]字段相应地被更新。<br>在读访问中，最后写入寄存器的数据被返回。 |

## 34.4.2.20 Tx缓冲器控制寄存器（QuadSPI_TBCT）。

该寄存器包含发送数据缓冲区的控制信息。

地址。0h base + 158h offset = 158h




**QuadSPI_TBCT字段描述**

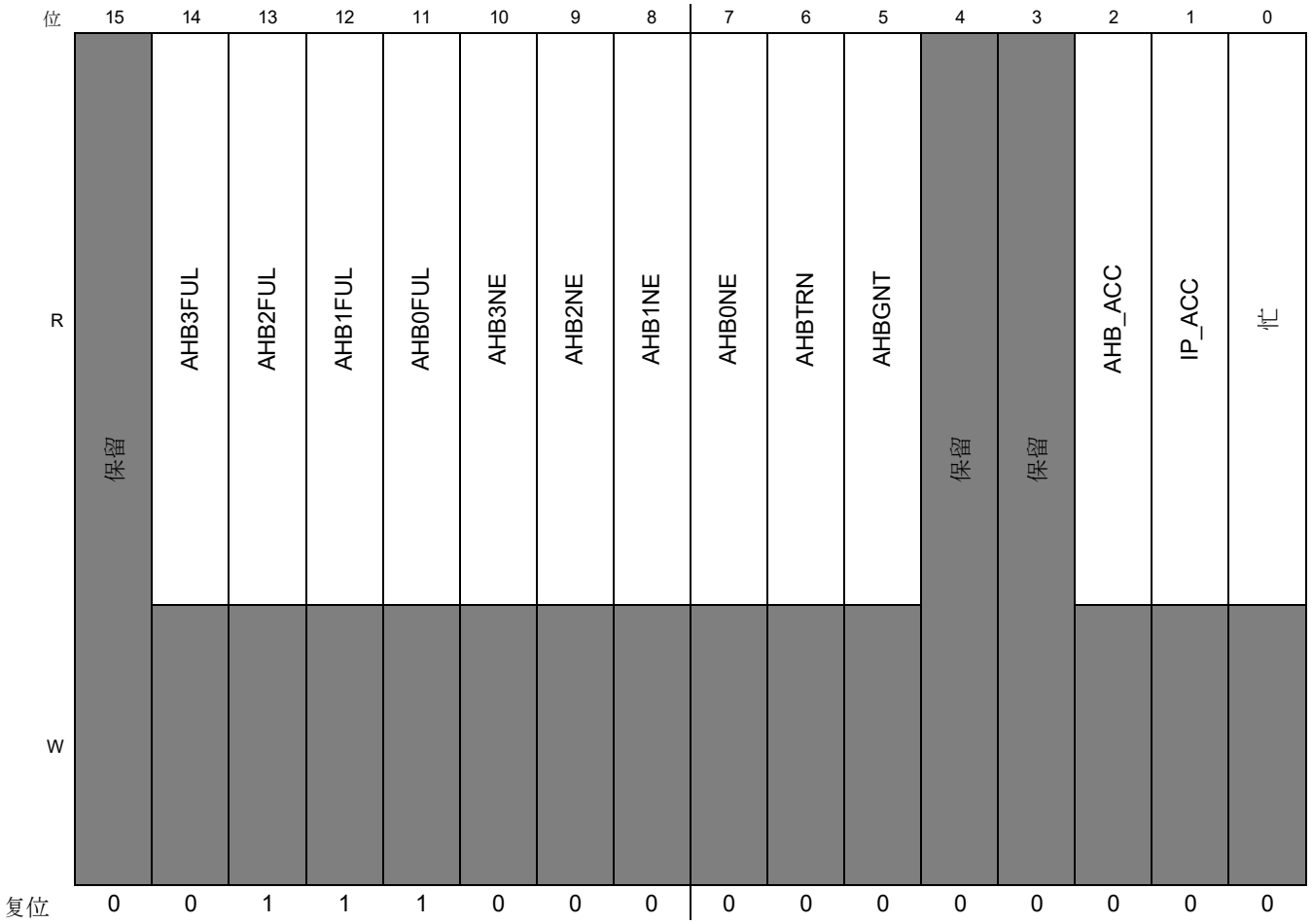
| 场地 | 描述 |
|---|---|
| 31-5<br>保留 | 这个字段是保留的。 |
| WMRK | 确定TX Buffer的水印。当TX缓冲区的可用空间数大于(WMRK+1)给出的数字时，QSPI_SR[TXWA]被断言。这些值应该是输入4Bytes的条目数减去1。例如，0x0的值将设置水印为4字节，1为8字节，2为12字节，以此类推。详情请参考DMA用法。 |

## 34.4.2.21 状态寄存器（QuadSPI_SR）。

QSPI_SR寄存器提供关于SFM命令执行和仲裁、RX缓冲器、TX缓冲器和AHB缓冲器的所有可用状态信息。

地址。0h基数+15Ch偏移量=15Ch

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 保留 | | | 保留 | 辽宁省 | TXDMA | TXWA | TXEDA | 呼叫中心 | 保留 | | | 荣盛 | 保留 | | RXWE |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |




## QuadSPI_SR字段描述

| 场地 | 描述 |
|---|---|
| 31-29<br>保留 | 这个字段是保留的。 |
| 28<br>保留 | 这个字段是保留的。 |
| 27<br>辽宁省 | TX缓冲区满。当没有更多的数据可以被存储时，就会发出警报。 |
| 26<br>TXDMA | TXDMA<br><br>当通过DMA填充TXFIFO是有效的，即DMA被请求或正在运行时，被断言。 |
| 25<br>TXWA | TX缓冲区水印可用<br><br>当TX缓冲区的可用空间数大于或等于QSPI_TBCT[WMRK]给出的值时，发出警报。 |
| 24<br>TXEDA | Tx 缓冲区有足够的数据可用<br><br>当 TX 缓冲区包含足够的数据来进行任何弹出操作时，就会被引爆。对于任何弹出操作，TX FIFO 中必须至少有 128 位数据可用；否则，QSPI_FR[TBUF]将被设置。 |
| 23<br>呼叫中心 | RX Buffer DMA.当通过DMA读出的RX Buffer处于激活状态时，即DMA被请求或正在运行时，就会被引爆。 |

表继续在下一页...

## QuadSPI_SR字段描述（续）

| 场地 | 描述 |
|---|---|
| 22-20<br>保留 | 这个字段是保留的。 |
| 19<br>荣盛 | RX 缓冲区满。当 RX 缓冲区满时，即 QSPI_RBSR[RDBFL]字段等于 32 时，发出信号。 |
| 18-17<br>保留 | 这个字段是保留的。 |
| 16<br>RXWE | RX Buffer Watermark Exceeded.当RX缓冲区中的有效条目数超过QSPI_RBCT[WMRK]字段中给出的数字时，发出警报。 |
| 15<br>保留 | 这个字段是保留的。 |
| 14<br>AHB3FUL | AHB 3 缓冲区满。当AHB 3 缓冲区满时，发出信号。 |
| 13<br>AHB2FUL | AHB 2 缓冲区满。当AHB 2 缓冲区满时，发出信号。 |
| 12<br>AHB1FUL | AHB 1 缓冲区满。当AHB 1 缓冲区满时，被断言。 |
| 11<br>AHB0FUL | AHB 0 缓冲区满。当AHB 0 缓冲区满时，发出信号。 |
| 10<br>AHB3NE | AHB 3 Buffer Not Empty.当AHB 3缓冲区包含数据时，发出信号。 |
| 9<br>AHB2NE | AHB 2 Buffer Not Empty.当AHB 2 缓冲区包含数据时被引爆。 |
| 8<br>AHB1NE | AHB 1 Buffer Not Empty.当AHB 1缓冲区包含数据时，发出信号。 |
| 7<br>AHB0NE | AHB 0 Buffer Not Empty.当AHB 0缓冲区包含数据时，被断言。 |
| 6<br>AHBTRN | AHB访问交易未决。当AHB接口上有一个挂起的请求时，被断言。详情请参考AMBA规范。 |
| 5<br>AHBGNT | 授予AHB命令优先权。当另一个模块被授予AHB命令对IP命令的优先权时被断言。详情请参考命令仲裁。 |
| 4<br>保留 | 这个字段是保留的。 |
| 3<br>保留 | 这个字段是保留的。 |
| 2<br>AHB_ACC | AHB访问。当当前执行的交易是由AHB总线发起时，发出信号。 |
| 1<br>IP_ACC | IP访问。当当前执行的交易是由IP总线发起时，被断言。 |
| 0<br>忙 | 模块忙。当模块目前正忙于处理对外部闪存设备的交易时，发出该信号。 |

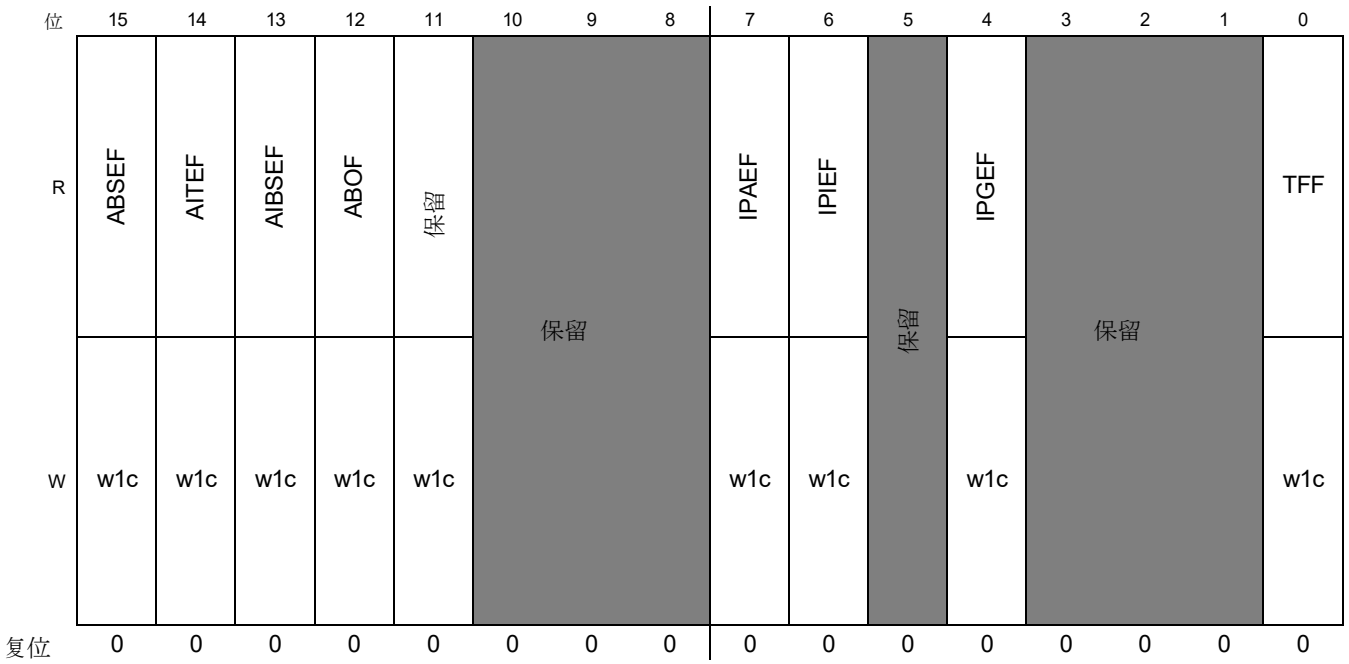
## 34.4.2.22 标志寄存器 (QuadSPI_FR )

QSPI_FR寄存器提供了所有关于SFM命令执行和仲裁的可用标志，可以作为产生中断服务请求的来源。请注意，这个寄存器中的错误标志并不直接与串行闪存设备本身的事务执行有关，而只是与QuadSPI模块中的行为和条件有关。

*写。启用模式*

地址。0h基数+160h偏移量=160h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 保留 | 保留 | | 保留 | TBFF | TBUF | 保留 | | 伊利兰 | 保留 | | | | | RBOF | RBDF |
| W | | | | | w1c | w1c | | | w1c | | | | | | w1c | w1c |
| 复位 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |




## QuadSPI_FR字段描述

| 场地 | 描述 |
|---|---|
| 31<br>保留 | 这个字段是保留的。 |
| 30-29<br>保留 | 这个字段是保留的。 |
| 28<br>保留 | 这个字段是保留的。 |
| 27<br>TBFF | TX缓冲器填充标志。在写到TX缓冲区之前，这个位应该被清空。然后必须读回这个位。如果该位被设置，TX缓冲区可以接受更多的数据。如果该位保持清零，则TX缓冲区已满。<br>详情请参考Tx Buffer Operation。 |
| 26<br>TBUF | TX Buffer Underrun Flag.当模块试图提取数据时，尽管TX Buffer是空的或缓冲区包含的数据少于128位，但仍会被设置。应用程序必须确保在交易过程中，除了最后的数据获取，缓冲区永远不会变空。导致TX Buffer underrun的IP命令被继续执行（在有效的TX underrun的情况下，发送到串行闪存设备的数据都是F。应用程序必须通过向QSPI_MCR[CLR_TXF]位写1来清除TX缓冲区，以响应这一事件。 |
| 25-24<br>保留 | 这个字段是保留的。 |
| 23<br>伊利兰 | 非法指令错误标志。当控制器在任何序列中遇到非法指令时，将其置入。有关合法指令的列表，请参考表34-14。 |
| 22-18<br>保留 | 这个字段是保留的。 |
| 17<br>RBOF | RX缓冲器溢出标志。当从串行闪存设备中读取的数据不能全部推入RX缓冲区时置位。<br>导致这种情况的IP命令一直持续到从串行闪存设备中读出符合QSPI_IPCR[IDATSZ]字段的字节数。<br>RX缓冲区的内容没有改变。 |

*表继续在下一页...*

## QuadSPI_FR字段描述（续）

| 场地 | 描述 |
|---|---|
| 16<br>RBDF | RX Buffer Drain Flag。如果QuadSPI_SR[RXWE]状态位被宣称，将被设置。向该位写1<br><br>会触发以下动作之一。<br><br>• 如果RX缓冲区有多达QuadSPI_RBCT[WMRK]的有效条目，则该标志被清除。<br>• 如果RX缓冲区的有效条目超过QuadSPI_RBCT[WMRK]，并且QuadSPI_RSER[RBDDE]位没有被设置（标志驱动模式），则触发RX缓冲区POP事件。<br><br>在RX Buffer POP事件结束后，如果RX Buffer包含超过QuadSPI_RBCT[WMRK]的有效条目，则该标志保持设置。<br><br>在RX Buffer POP事件结束后，如果RX Buffer包含的有效条目少于或等于QuadSPI_RBCT[WMRK]，则该标志被清零。<br><br>详情请参考正常模式中断和DMA请求中的 "接收缓冲区耗尽中断或DMA请求 "部分。 |
| 15<br>ABSEF | AHB 序列错误标志。当AHB命令的执行是以QSPI_BUFxCR寄存器所指向的序列中的WRITE或WRITE_DDR 命令开始时，置位。(QSPI_BUFxCR 意味着QSPI_BUF0CR/QSPI_BUF1CR/QSPI_BUF2CR/QSPI_BUF3CR中的任意一个）。<br><br>在QuadSPI模块执行WRITE/WRITE_DDR命令之前，与串行闪存设备的通信被终止。<br><br>触发该命令的AHB总线请求以ERROR响应回答。 |
| 14<br>AITEF | AHB非法交易错误标志。在非法交易的情况下，只要QSPI到AHB总线没有产生响应，并且看门狗定时器过期，就会被设置。 |
| 13<br>AIBSEF | AHB非法突发大小错误标志。当AHB事务的总突发大小（大小x节拍）大于预取数据大小时，将被设置。预取数据大小由QSPI_BUFxCR[ADATSZ]定义，或者在ADATSZ=0的情况下，由SEQID字段指向的序列中提到的数据大小。 关于HBURST功能的更多细节，请参考HBURST支持。 |
| 12<br>ABOF | AHB缓冲区溢出标志。当AHB访问的大小超过AHB缓冲区的大小时置位。只有当QSPI_BUFxCR[ADATSZ]字段被错误编程时，才会出现这种情况。<br><br>导致这种情况的AHB命令一直持续到根据QSPI_BUFxCR[ADATSZ]字段的条目数从串行闪存设备中读取为止。<br><br>AHB Buffer的内容没有改变。 |
| 11<br>保留 | 这个字段是保留的。 |
| 10-8<br>保留 | 这个字段是保留的。 |
| 7<br>IPAEF | AHB访问时IP命令触发错误标志。当发生以下情况时置位。<br>• 对QSPI_IPCR[SEQID]字段进行写访问，QSPI_SR[AHB_ACC]位被设置。任何导致IPAEF标志断言的命令都会被忽略。 |

| | |
|---|---|
| 6<br>IPIEF | IP命令触发器无法执行 错误标志。当QSPI_SR[IP_ACC]位被设置时（即当前正在执行一个IP触发的命令），并发生以下任何一种情况时置位。<br>• 对QSPI_IPCR寄存器的写访问。任何导致IPIEF标志断言的命令都会被忽略。<br>• 对QSPI_SFAR寄存器的写访问。<br>• 对QSPI_RBCT寄存器的写访问。 |
| 5<br>保留 | 这个字段是保留的。 |
| 4<br>IPGEF | 在AHB授予错误标志期间IP命令触发。当发生以下情况时，设置。<br>• 对QSPI_IPCR[SEQID]字段进行写访问，QSPI_SR[AHBGNT]位被设置。任何导致IPGEF标志断言的命令都会被忽略。 |

*表继续在下一页...*

**QuadSPI_FR字段描述（续）**

| 场地 | 描述 |
|---|---|
| 3-1<br>保留 | 这个字段是保留的。 |
| 0<br>TFF | IP命令交易完成标志。当QuadSPI模块完成一个正在运行的IP命令时置位。如果发生错误，相关的错误标志最迟在TFF标志被断言的同一时钟周期内有效。 |

## 34.4.2.23 中断和DMA请求选择和启用寄存器（QuadSPI_RSER）。

QuadSPI_RSER寄存器为QuadSPI模块中的中断提供使能和选择器。

**注意事项**

QuadSPI_FR寄存器的每个标志被启用为中断的来源，当该标志被设置时，可以防止QuadSPI模块进入停止模式或模块禁用模式。

*写。任何时候*

地址。0h base + 164h offset = 164h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R<br>W | 保留 | 保留 | | 保留 | TBFIE | TBUIE | TBFDE | 保留 | 伊利诺伊州 | 保留 | RBDDE | 保留 | | | RBOIE | RBDIE |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 位 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R<br>W | ABSEIE | AITIE | 缩写：AIBUF | ÂÂÂ | 保留 | 保留 | | | IPAEIE | IPIEIE | 保留 | IPGEIE | 保留 | | | TFIE |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

## QuadSPI_RSER字段描述

| 场地 | 描述 |
|---|---|
| 31<br>保留 | 保留<br><br>这个字段是保留的。 |
| 30-29<br>保留 | 这个字段是保留的。 |
| 28<br>保留 | 这个字段是保留的。 |
| 27<br>TBFIE | TX缓冲区填充中断启用<br><br>0 将不产生TBFF中断<br>1 将产生TBFF中断 |
| 26<br>TBUIE | TX缓冲区欠运行中断启用<br><br>0 将不产生TBUF中断<br>1 将产生TBUF中断 |
| 25<br>TBFDE | TX缓冲区填充DMA启用<br><br>启用生成 DMA 请求以填充 TX 缓冲区。当设置该选项时，只要QSPI_SR[TXWA]状态位被设置，就会产生DMA请求。<br><br>0 将不产生DMA请求<br>1 将产生DMA请求 |
| 24<br>保留 | 这个字段是保留的。 |
| 23<br>伊利诺伊州 | 非法指令错误中断启用。由 QSPI_FR 中的 ILLINE 标志触发。<br><br>0 将不会产生ILLINE中断<br>1 将产生ILLINE中断 |
| 22<br>保留 | 这个字段是保留的。 |
| 21<br>RBDDE | RX Buffer Drain DMA Enable:启用RX Buffer Drain的DMA请求的生成。当该位被设置时，只要QSPI_SR[RXWE]状态位被设置，就会产生DMA请求。<br><br>0 将不产生DMA请求<br>1 将产生DMA请求 |
| 20-18<br>保留 | 这个字段是保留的。 |
| 17<br>RBOIE | RX缓冲区溢出中断启用<br><br>0 将不产生RBOF中断<br>1 将产生RBOF中断 |

| | |
|---|---|
| 16<br>RBDIE | RX Buffer Drain中断启用。启用 RX 缓冲区排水的 IRQ 请求的生成。当该位被设置时，只要QuadSPI_SR[RBDF]标志被设置，中断就会被断言。<br><br>0 将不产生RBDF中断<br>1 将产生RBDF中断 |
| 15<br>ABSEIE | AHB 序列错误中断启用。由QSPI_FR的ABSEF标志触发 |

*表继续在下一页...*

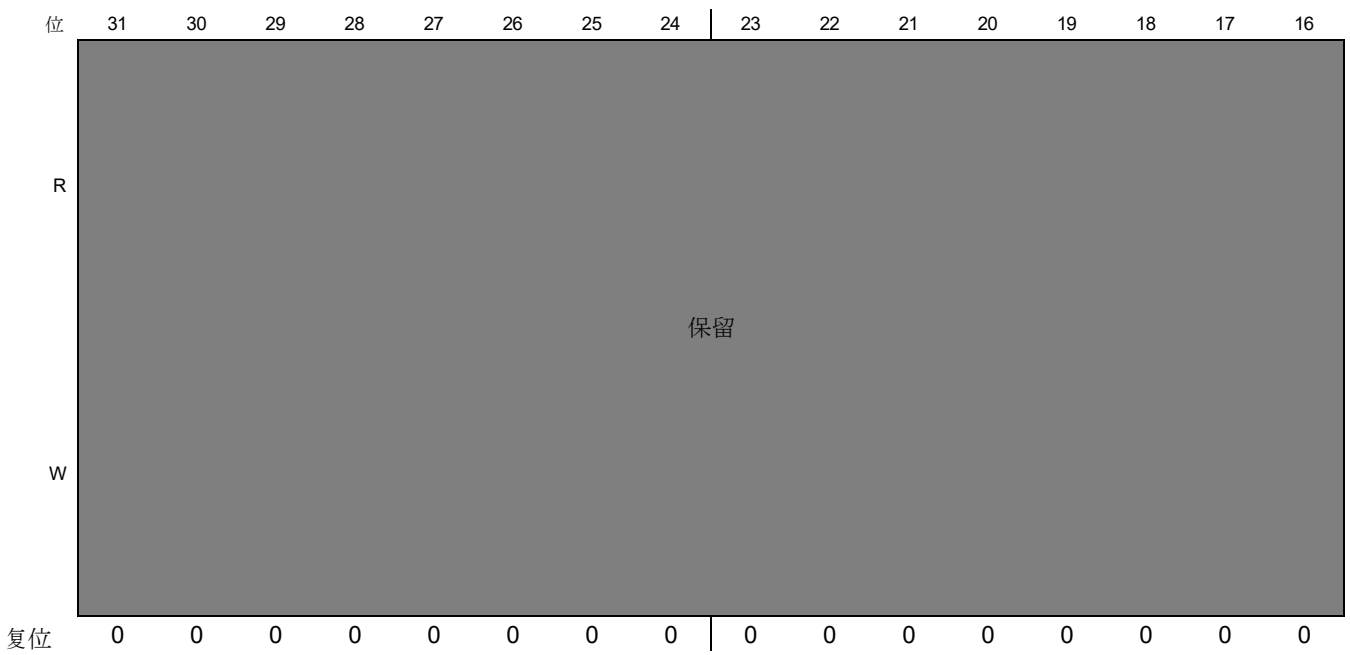
## QuadSPI_RSER字段描述（续）

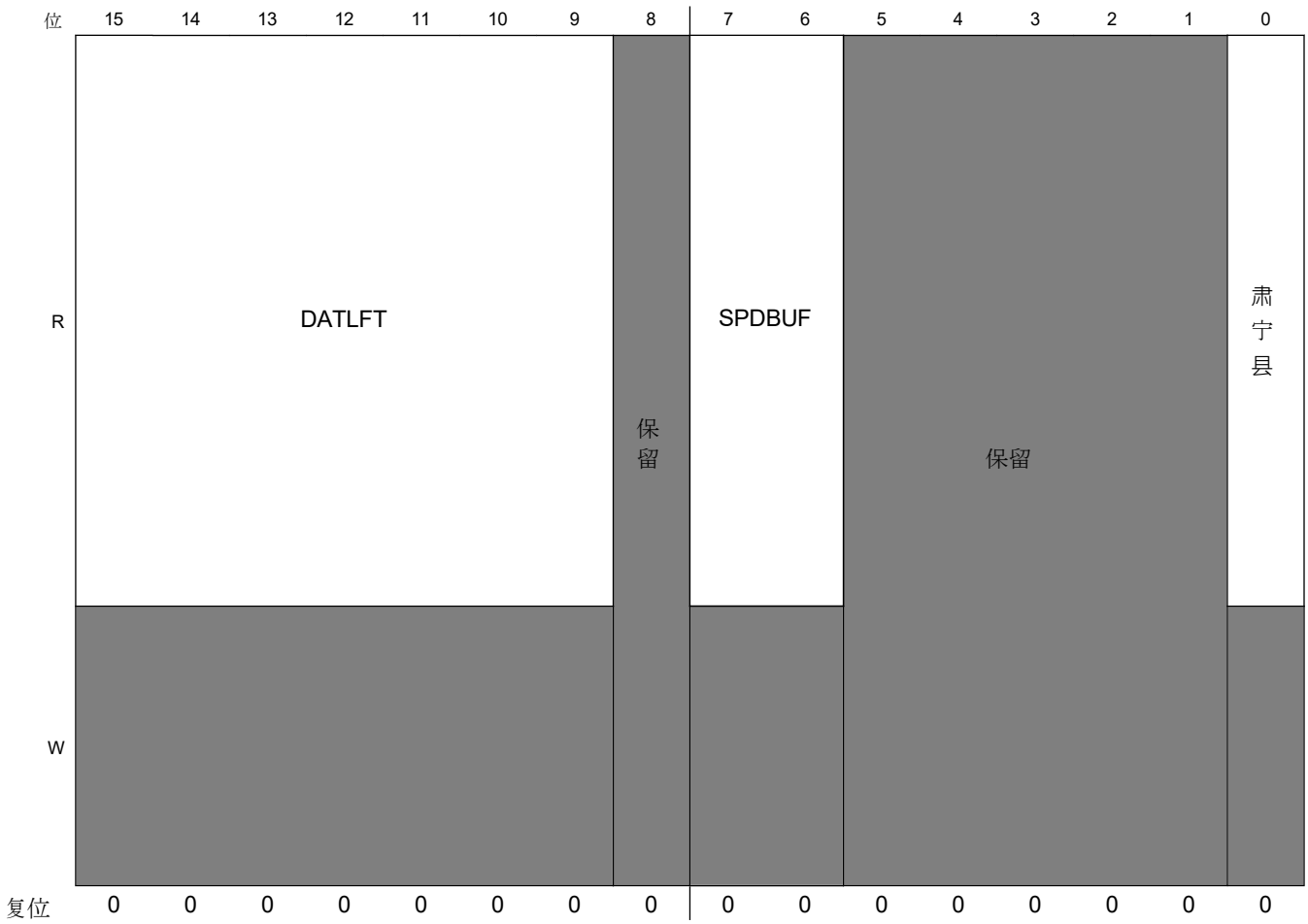
| 场地 | 描述 |
|---|---|
| | 0 将不产生ABSEF中断<br>1 将产生ABSEF中断 |
| 14<br>AITIE | AHB非法交易中断启用。<br>0 将不产生AITEF中断<br>1 将产生AITEF中断 |
| 13<br>缩写：AIBSIE | AHB非法突发尺寸中断启用<br>0 将不产生AIBSEF中断<br>1 将产生AIBSEF中断 |
| 12<br>ÃÃÃ | AHB缓冲区溢出中断启用<br>0 将不产生ABOF中断<br>1 ABOF中断将被产生 |
| 11<br>保留 | 这个字段是保留的。 |
| 10-8<br>保留 | 这个字段是保留的。 |
| 7<br>IPAEIE | 在AHB访问错误中断启用期间IP命令触发<br>0 将不会产生IPAEF中断<br>1 将产生IPAEF中断 |
| 6<br>IPIEIE | IP访问错误中断启用期间的IP命令触发器<br>0 将不产生IPIEF中断<br>1 将产生IPIEF中断 |
| 5<br>保留 | 这个字段是保留的。 |
| 4<br>IPGEIE | 在AHB授予错误中断启用期间IP命令触发<br>0 将不产生IPGEF中断<br>1 将产生IPGEF中断 |
| 3-1<br>保留 | 这个字段是保留的。<br>保留。 |
| 0<br>TFIE | 交易完成的中断启用<br>0 将不产生TFF中断<br>1 将产生TFF中断 |

## 34.4.2.24 序列暂停状态寄存器（QuadSPI_SPNDST）。

序列暂停状态寄存器提供任何暂停序列的具体信息。当一个高优先级的AHB主站在AHB序列完成所要求的数据传输之前进行访问时，一个AHB序列可能被暂停。

地址。0h base + 168h offset = 168h






## QuadSPI_SPNDST字段描述

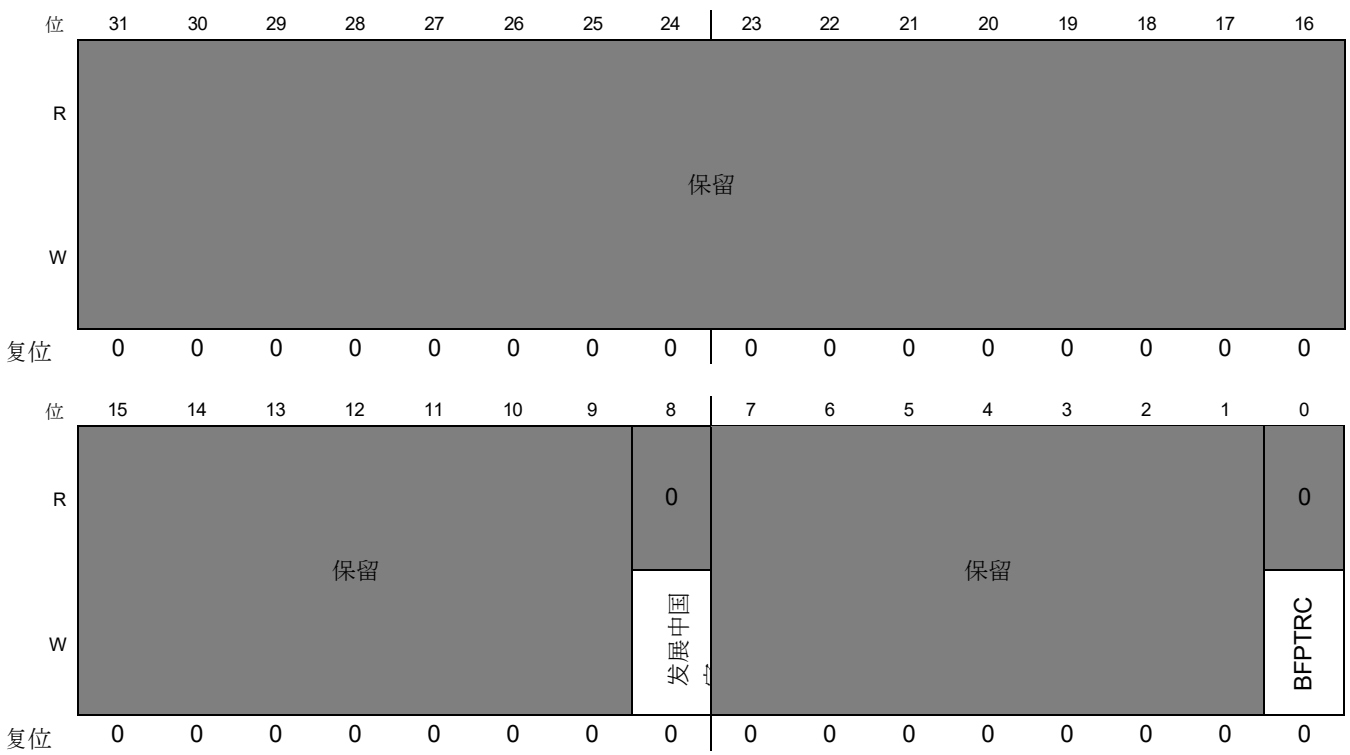
| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。 |
| 15-9<br>DATLFT | 剩余数据。提供关于暂停序列中剩余的数据量的信息。仅当SUSPND设置为1'b1时有效。以64比特或8字节为单位的值 |
| 8<br>保留 | 这个字段是保留的。 |
| 7-6<br>SPDBUF | 暂停的缓冲区。提供暂停的缓冲区编号。仅当SUSPND被设置为1'b1时有效。 |
| 5-1<br>保留 | 这个字段是保留的。 |
| 0<br>肃宁县 | 当设置时，它标志着一个序列处于暂停状态 |

## 34.4.2.25 序列指针清除寄存器（QuadSPI_SPTRCLR）。

序列指针清除寄存器提供了重置IP和缓冲器序列指针的位。序列指针包含了LUT条目中哪条指令将被接下来执行的索引。例如，如果LUT条目在JMP_ON_CS值为2时结束，索引将被存储为2。

每当序列ID改变时，软件应通过更新QSPI_IPCR或QSPI_BFGENCR中的SEQID字段来重置序列指针。

地址。0h基数+16Ch偏移量=16Ch




**QuadSPI_SPTRCLR字段描述**

| 场地 | 描述 |
|---|---|
| 31-9<br>保留 | 这个字段是保留的。 |
| 8<br>发展中国家 | IP 指针清除。<br>1: 清除QuadSPI_IPCR中定义的IP访问的序列指针，这是一个自我清除的字段。 |
| 7-1<br>保留 | 这个字段是保留的。<br>保留。 |
| 0<br>BFPTRC | 缓冲区指针清除。<br>1：按照QuadSPI_BFGENCR中的定义，清除AHB访问的序列指针。这是一个自我清除的<br>字段。 |

## 34.4.2.26 串行闪存A1顶部地址（QuadSPI_SFA1AD）。

QSPI_SFA1AD寄存器提供了串行闪存A1的地址映射。QSPI_SFA1AD[TPADA1]和QSPI_AMBA_BASE之间的差异定义了串行闪存A1的内存映射大小。

*写。*

- *qspi_sr[ip_acc] = 0*
- *qspi_sr[ahb_acc] = 0*

地址。0h基数+180h偏移量=180h

| 位 | 31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 | 15 14 13 12 11 10 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| R<br>W | TPADA1 | | 保留 | | | | | | | | |
| 复位 | * | * | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

* 注意事项。

- TPADA1字段。关于设备特定的复位值，请参见模块配置。

**QuadSPI_SFA1AD字段描述**

| 场地 | 描述 |
|---|---|
| 31-10<br>TPADA1 | 串行闪存A1的最高地址。实际上，TPADxx是下一个存储器的第一个位置。 |
| 保留 | 这个字段是保留的。 |

## 34.4.2.27 串行闪存A2顶部地址（QuadSPI_SFA2AD）。

QSPI_SFA2AD寄存器提供了串行闪存A2的地址映射。QSPI_SFA2AD[TPADA2]和QSPI_SFA1AD[TPADA1]之间的区别定义了串行闪存A2的内存映射大小。

*写。*

- *qspi_sr[ip_acc] = 0*
- *qspi_sr[ahb_acc] = 0*

地址。0h base + 184h offset = 184h

| 位 | 31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 | 15 14 13 12 11 10 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| R<br>W | TPADA2 | | 保留 | | | | | | | | |
| 复位 | * | * | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

* 注意事项。

- TPADA2字段。关于设备特定的复位值，请参见模块配置。

**QuadSPI_SFA2AD字段描述**

| 场地 | 描述 |
|---|---|
| 31-10<br>TPADA2 | 串行闪存A2的顶部地址。实际上，TPxxAD是下一个存储器的第一个位置。 |
| 保留 | 这个字段是保留的。 |

## 34.4.2.28 串行闪存B1顶部地址（QuadSPI_SFB1AD）。

QSPI_SFB1AD寄存器提供了串行闪存B1的地址映射。QSPI_SFB1AD[TPADB1]和QSPI_SFA2AD[TPADA2]之间的区别定义了串行闪存B1的内存映射的大小。

*写。*

- *qspi_sr[ip_acc] = 0*
- *qspi_sr[ahb_acc] = 0*

地址。0h base + 188h offset = 188h

| 位 | 31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 | 15 14 13 12 11 10 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| R<br>W | TPADB1 | | 保留 | | | | | | | | |
| 复位 | * | * | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

* 注意事项。

- TPADB1字段。关于设备特定的复位值，请参见模块配置。

**QuadSPI_SFB1AD字段描述**

| 场地 | 描述 |
|---|---|
| 31-10<br>TPADB1 | 串行闪存B1的顶部地址。实际上，TPxxAD是下一个存储器的第一个位置。 |
| 保留 | 这个字段是保留的。 |

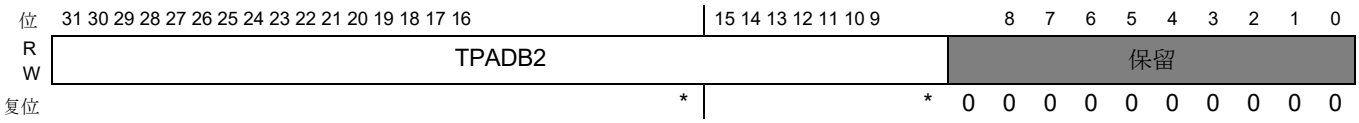
## 34.4.2.29 串行闪存B2顶部地址(QuadSPI_SFB2AD)

QSPI_SFB2AD寄存器提供了串行闪存B2的地址映射。QSPI_SFB2AD[TPADB2]和QSPI_SFB1AD[TPADB1]之间的区别定义了串行闪存B2的内存映射的大小。

*写。*

- *qspi_sr[ip_acc] = 0*
- *qspi_sr[ahb_acc] = 0*

地址。0h基数+18Ch偏移量=18Ch




* 注意事项。

- TPADB2字段。关于设备特定的复位值，请参见模块配置。

**QuadSPI_SFB2AD字段描述**

| 场地 | 描述 |
|---|---|
| 31-10<br>TPADB2 | 串行闪存B2的最高地址。实际上，TPxxAD是下一个存储器的第一个位置。 |
| 保留 | 这个字段是保留的。 |

## 34.4.2.30 RX缓冲器数据寄存器（QuadSPI_RBDRn）。

QuadSPI_RBDR 寄存器提供了对 RX
缓冲器中各个项目的访问。字节排序方案请参考表34-16。

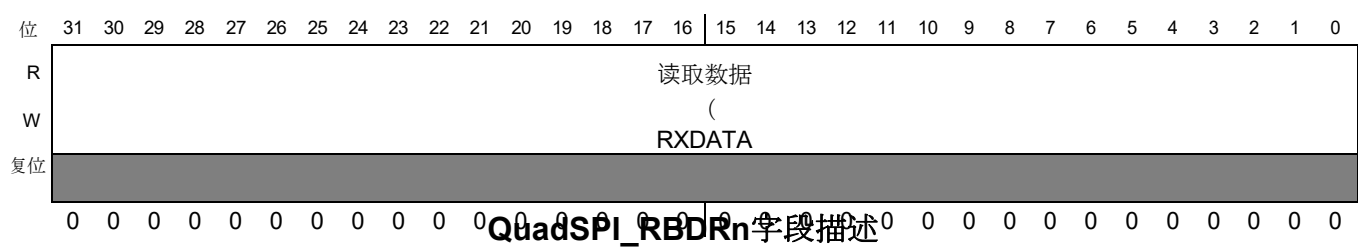
QuadSPI_RBDR0对应的是RX缓冲区内读指针的实际位置。可用的有效条目数取决于实现的RX缓冲区条目数和RX缓冲区中可用的有效缓冲区条目数。

例1，RX缓冲区完全充满了32个字。在这种情况下，有效读访问的地址范围从QuadSPI_RBDR0延伸到QuadSPI_RBDR31。

例2，RX缓冲区充满了5个有效字。在这种情况下，对QuadSPI_RBDR4的访问提供了最后一个有效条目。

任何超出有效的RX缓冲区条目范围的访问都会产生未定义的结果。

地址。0h基数+200h偏移量+（4d×i），其中i=0d至31d




**QuadSPI_RBDRn字段描述**

| 场地 | 描述 |
|---|---|
| 读取数据（<br>RXDATA | RX数据。RXDATA字段包含与相关的RX缓冲区条目相关的数据。数据格式和字节排序在《串行闪存读取数据的字节排序》中给出。 |

## 34.4.2.31 LUT键寄存器（QuadSPI_LUTKEY）。

LUT Key寄存器包含锁定和解锁查找表的钥匙。详情请参考查找表。

*写。任何时候*

地址。0h基数+300h偏移量=300h




**QuadSPI_LUTKEY字段描述**

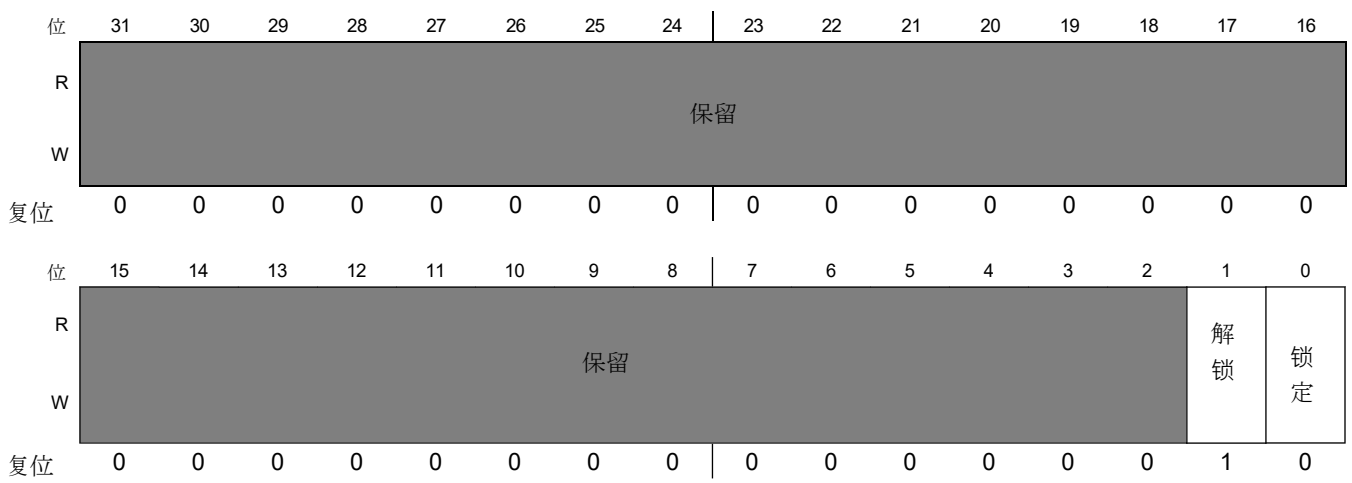
| 场地 | 描述 |
|---|---|
| 关键 | 锁定或解锁LUT的钥匙。KEY是0x5AF05AF0。读取值总是0x5AF05AF0 |

## 34.4.2.32 LUT锁定配置寄存器（QuadSPI_LCKCR）。

LUT锁配置寄存器与QSPI_LUTKEY寄存器一起用于锁定或解锁LUT。这个寄存器必须在QSPI_LUTKEY寄存器之后立即写入，才能使锁定或解锁操作成功。详情请参考查找表。不允许将LOCK和UNLOCK位同时设置为 "00 "或 "11"。

*写。刚写完LUT键寄存器（QSPI_LUTKEY）。*

地址。0h基数+304h偏移量=304h



QuadSPI_LCKCR字段描述

| 场地 | 描述 |
|---|---|
| 31-2<br>保留 | 这个字段是保留的。 |
| 1<br>解锁 | 当满足以下两个条件时，解锁LUT。<br>1. 该寄存器正好写在LUT键寄存器（QuadSPI_LUTKEY）之后。<br>2. LUT键寄存器被写入0x5AF05AF0键 |
| 0<br>锁定 | 当满足以下条件时，锁定LUT。<br>1. 该寄存器正好写在LUT键寄存器（QuadSPI_LUTKEY）之后。<br>2. LUT键寄存器被写入0x5AF05AF0键 |

## 34.4.2.33 查找表寄存器（QuadSPI_LUTn）。

指令-操作数对的序列可以根据板上连接的设备预先填充到LUT中。每个指令-操作数对都是16比特（2字节）。程序序列引擎在LUT中预置的每个序列都以其索引来表示。LUT寄存器是一个指令序列的查找表。可编程序列引擎执行这些序列中的指令，以产生一个有效的串行闪存交易。一共有64个LUT寄存器。这64个寄存器
，分为4个寄存器组，组成一个有效的序列。因此，QSPI_LUT[0]、QSPI_LUT[4]、QSPI_LUT[8].....QSPI_LUT[60]是一个有效序列的起始寄存器序。这4组寄存器中的每一组最多可以有8条指令。下面显示的寄存器的复位值只适用于LUT2到LUT63。一次最多可以定义16条序列。查找表详细描述了LUT寄存器的情况。

### 注意事项

LUT0和LUT1的复位值分别为0818_0403h和2400_1C08h。

*写。一旦LUT被解锁*

地址。0h基数+310h偏移量+（4d×i），其中i=0d至63d

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R<br>W | INSTR1 | | | | | | PAD1 | | OPRND1 | | | | | | | |
| 复位 | 0* | 0* | 0* | 0* | 0* | 0* | 0* | 0* | 0* | 0* | 0* | 0* | 0* | 0* | 0* | 0* |

| 位 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R<br>W | INSTR0 | | | | | | PAD0 | | OPRND0 | | | | | | | |
| 复位 | 0* | 0* | 0* | 0* | 0* | 0* | 0* | 0* | 0* | 0* | 0* | 0* | 0* | 0* | 0* | 0* |

* 注意事项。

- LUT0和LUT1的复位值分别为0818_0403h和2400_1C08h。

**QuadSPI_LUTn字段描述**

| 场地 | 描述 |
|---|---|
| 31-26<br>INSTR1 | 指示1 |
| 25-24<br>PAD1 | INSTR1的焊盘信息。<br>001 垫子<br>012 垫子<br>104个垫子<br>118个垫子 |
| 23-16<br>OPRND1 | INSTR1的操作数。 |
| 15-10<br>INSTR0 | 指示0 |
| 9-8<br>PAD0 | INSTR0的焊盘信息。<br>001 垫子<br>012 垫子<br>104个垫子<br>118个垫子 |
| OPRND0 | INSTR0的操作数。 |

## 34.4.3 串行闪存地址的分配

串行闪存地址分配可以通过写进设备A的串行闪存A1顶层地址(QuadSPI_SFA1AD)和串行闪存A2顶层地址(QuadSPI_SFA2AD)以及设备B的串行闪存B1顶层地址(QuadSPI_SFB1AD)和串行闪存B2顶层地址(QuadSPI_SFB2AD)来修改。注意，这个地址分配对IP和AHB命令都有效。

**表34-8.串行闪存地址的分配**

| 参数 | 职能 | 访问模式 |
|---|---|---|
| QSPI_AMBA_BASE<br>((31:10) - 22位) | QuadSPI AHB基础地址 | |
| top_addr_mema1(tpada1) | 外置闪光灯A1的顶部地址（共享IOFA的两个独立闪光灯中的第一个）。 | 任何对TOP_ADDR_MEMA1和QSPI_AMBA_BASE之间的地址空间的访问将被路由到**串行闪存A1。** |
| top_addr_mema2(tpada2) | 外置闪光灯A2（共享IOFA的两个独立闪光灯中的第二个）的最高地址。 | 任何对TOP_ADDR_MEMA2和TOP_ADDR_MEMA1之间的地址空间的访问都将被路由到**串行闪存A2。** |
| top_addr_memb1(tpadb1) | 外置闪光灯B1的顶部地址（共享IOFB的两个独立闪光灯中的第一个 | 任何对TOP_ADDR_MEMB1和TOP_ADDR_MEMA2之间的地址空间的访问将被路由到**串行闪存B1。** |

| | | |
|---|---|---|
| | ）。 | |

*表继续在下一页...*

表34-8.串行闪存地址分配（续）

| 参数 | 职能 | 访问模式 |
| --- | --- | --- |
| top_addr_memb2(tpadb2) | 外部闪光灯B2的顶部地址（共享IOFB的两个独立闪光灯中的第二个）。 | 任何对TOP_ADDR_MEMB2和TOP_ADDR_MEMB1之间的地址空间的访问将被路由到**串行闪存A2。** |

## 34.5 闪存映射的AMBA总线

QSPI_AMBA_BASE
定义了由系统内存地图定义的串行闪存设备的起始地址。请注意，这可能是系统中串行闪存的物理地址的重映射。请参考系统地址图。请注意，这可能是系统中串行闪存物理地址的重映射。
请参考系统地址图。

表34-9.QuadSPI AMBA总线内存图

| 地址 | 寄存器名称 |
| --- | --- |
| 内存映射的串行闪存数据 - 闪存A上的单独闪存模式 | |
| QSPI_AMBA_BASE到（TOP_ADDR_MEMA2 - 0x01）。 | 内存映射的串行闪存数据--闪存A上的单独闪存模式 详细内容请参考内存映射的串行闪存数据--闪存A上的单独闪存模式，关于字节排序的信息请参考表34-16和表34-17。 |
| 内存映射的串行闪存数据 - 闪存B上的单独闪存模式 | |
| TOP_ADDR_MEMA2到（TOP_ADDR_MEMB2 - 0x01）。 | 内存映射的串行闪存数据 - 闪存B上的单独闪存模式 详细内容请参考内存映射的串行闪存数据 - 闪存B上的单独闪存模式，关于字节排序的信息请参考表34-16和表34-17。 |
| AHB RX数据缓冲器（QSPI_ARDB0至QSPI_ARDB31）。 | |
| QSPI_ARDB_ASE 到... (32 * 4 Byte)<br>QSPI_ARDB_ASE + 0x0000_01FF | AHB RX数据缓冲区（QSPI_ARDB0至QSPI_ARDB31） 参考表34-16和关于字节排序的信息。 |

### 注意事项

任何对非实现地址的读取访问都将提供未定义的结果。

如果是单片闪存设备，TOP_ADDR_MEMA2和TOP_ADDR_MEMB2应该分别被初始化/编程为TOP_ADDR_MEMA1和TOP_ADDR_MEMB1（实际上，将这些设备的大小设置为0）。这将确保完整的内存图只分配给一个闪存设备。

在 "单独闪存模式
"中，可用于闪存地址的3/4个地址字节（如序列中指令/操作数的编程）由SFADR或SFADR决定，如上表所示。

### 34.5.1 AHB总线访问的考虑因素

必须注意的是，QuadSPI模块中实现AHB总线访问的所有逻辑都是为了读取外部串行闪存设备的内容。因此，下列限制适用于QuadSPI模块对AHB总线的访问。

- 目前，QuadSPI不支持AHB写，所以根据AMBA AHB规范，任何写访问都以ERROR条件回答。不发生写操作。
- 根据AMBA_AHB规范，任何导致QSPI_FR[ABSEF]标志宣称的AHB命令都以ERROR条件回答。由此产生的AHB命令被忽略。
- 完全支持的AHB总线访问类型是NONSEQ和BUSY。
- AHB访问类型SEQ的处理方式与NONSEQ相同。更多细节请参考AMBA AHB规范。

### 34.5.2 内存映射的串行闪存数据 - 闪存A上的单独闪存模式

从地址QSPI_AMBA_BASE开始，第一个外部串行闪存设备的内容被映射到包含QuadSPI模块的设备的地址空间。串行闪存地址字节地址0x0与总线地址QSPI_AMBA_BASE相对应，顺序递增。地址映射情况请参考下表。表34-16中给出了32位访问的字节排序，表34-17中给出了64位读取访问的字节排序。

**表34-10.内存映射的单独闪存模式--闪存A地址方案**

| 内存映射地址32位访问 | 内存映射地址64位访问 | 串行闪存字节地址 | 闪存设备 |
|---|---|---|---|
| QSPI_AMBA_BASE + 0x00 | QSPI_AMBA_BASE + 0x00 | 0x00_0000至0x00_0003 | A1 |
| QSPI_AMBA_BASE + 0x04 | | 0x00_0004至0x00_0007 | |
| ... | ... | ... | |

*表继续在下一页...*

表34-10.内存映射的单独闪存模式--闪存A地址方案（续）

| 内存映射地址32位访问 | 内存映射地址64位访问 | 串行闪存字节地址 | 闪存设备 |
|---|---|---|---|
| TOP_ADDR_MEMA1 - 0x08 | TOP_ADDR_MEMA1 - 0x08 | (TOP_ADDR_MEMA1- 0x08)到 (TOP_ADDR_MEMA1 - 0x04 -0x01) | |
| TOP_ADDR_MEMA1 - 0x04 | | (TOP_ADDR_MEMA1 - 0x04) 到 (TOP_ADDR_MEMA1 - 0x01) | |
| TOP_ADDR_MEMA1 + 0x00 | TOP_ADDR_MEMA1 + 0x00_0000 | 0x00_0000至0x00_0003 | A2 |
| TOP_ADDR_MEMA1 + 0x04 | | 0x00_0004至0x00_0007 | |
| ..... | ... | ... | |
| TOP_ADDR_MEMA2 - 0x08 | TOP_ADDR_MEMA2 - 0x08 | (TOP_ADDR_MEMA2 - 0x08) 到 (TOP_ADDR_MEMA2 - 0x04 - 0x01) | |
| TOP_ADDR_MEMA2 - 0x04 | | (TOP_ADDR_MEMA2 - 0x04) 到 (TOP_ADDR_MEMA2 - 0x01) | |

可用的地址范围取决于外部串行闪存设备的大小。任何超出外部串行闪存大小的访问都会产生未定义的结果。

关于读取过程的细节，请参考Flash Read。

## 34.5.3 内存映射的串行闪存数据 - 闪存B上的单独闪存模式

从地址TOP_ADDR_MEMA2开始，第一个外部串行闪存设备的内容被映射到包含QuadSPI模块的设备地址空间。串行闪存地址字节地址0x0与总线地址TOP_ADDR_MEMA2相对应，顺序递增。地址映射情况请参考下表。表34-16中给出了32位访问的字节排序，表34-17中给出了64位读取访问的字节排序。

表34-11.内存映射的单独闪存模式--闪存B地址方案

| 内存映射地址32位访问 | 内存映射地址64位访问 | 串行闪存字节地址 | 闪存设备 |
|---|---|---|---|
| TOP_ADDR_MEMA2 + 0x00 | TOP_ADDR_MEMA2 + 0x00 | 0x00_0000至0x00_0003 | B1 |
| TOP_ADDR_MEMA2 + 0x04 | | 0x00_0004至0x00_0007 | |
| ... | ... | ... | |
| TOP_ADDR_MEMB1 - 0x08 | TOP_ADDR_MEMB1 - 0x08 | (TOP_ADDR_MEMB1- TOP_ADDR_MEMA2 - 0x08) 到 (TOP_ADDR_MEMB1 - TOP_ADDR_MEMA2 - 0x04 -0x01) | |

*表继续在下一页...*

表34-11.内存映射的单独闪存模式--闪存B地址方案（续）

| 内存映射地址32位访问 | 内存映射地址64位访问 | 串行闪存字节地址 | 闪存设备 |
|---|---|---|---|
| TOP_ADDR_MEMB1 - 0x04 | | (TOP_ADDR_MEMB1- TOP_ADDR_MEMA2 - 0x04) 到 (TOP_ADDR_MEMB1- TOP_ADDR_MEMA2- 0x01) | |
| TOP_ADDR_MEMB1 + 0x00 | TOP_ADDR_MEMB1 + 0x00_0000 | 0x00_0000至0x00_0003 | B2 |
| TOP_ADDR_MEMB1 + 0x04 | | 0x00_0004至0x00_0007 | |
| ..... | ... | ... | |
| TOP_ADDR_MEMB2 - 0x08 | TOP_ADDR_MEMA2 - 0x08 | (TOP_ADDR_MEMB2 - TOP_ADDR_MEMB1- 0x08) 到 (TOP_ADDR_MEMB2 - TOP_ADDR_MEMB1 - 0x04 - 0x01) | |
| TOP_ADDR_MEMB2 - 0x04 | | (TOP_ADDR_MEMB2- TOP_ADDR_MEMB1 - 0x04) 到 (TOP_ADDR_MEMB2 - TOP_ADDR_MEMB1 - 0x01) | |

可用的地址范围取决于外部串行闪存设备的大小。任何超出外部串行闪存大小的访问都会产生未定义的结果。

关于读取过程的细节，请参考Flash Read。

## 34.5.4 AHB RX数据缓冲器（QSPI_ARDB0至QSPI_ARDB31）。

### 注意事项

关于QSPI AHB RX数据缓冲器的基本地址，请参见本文件中的系统内存图。

### 记忆地图

| 地址偏移（十六进制） | 注册名称 | 宽度（单位：比特） | 访问 | 重置值 | 节/页 |
|---|---|---|---|---|---|
| 0 | AHB RX数据缓冲寄存器（ARDB0）。 | 32 | R/W | 0000_0000h | 34.5.4.1/897 |
| 4 | AHB RX数据缓冲寄存器（ARDB1）。 | 32 | R/W | 0000_0000h | 34.5.4.1/897 |
| 8 | AHB RX数据缓冲器寄存器（ARDB2）。 | 32 | R/W | 0000_0000h | 34.5.4.1/897 |

| | | | | | |
|---|---|---|---|---|---|
| C | AHB RX数据缓冲寄存器（ARDB3）。 | 32 | R/W | 0000_0000h | 34.5.4.1/897 |
| 10 | AHB RX数据缓冲器寄存器（ARDB4）。 | 32 | R/W | 0000_0000h | 34.5.4.1/897 |
| 14 | AHB RX数据缓冲寄存器（ARDB5）。 | 32 | R/W | 0000_0000h | 34.5.4.1/897 |
| 18 | AHB RX数据缓冲寄存器（ARDB6）。 | 32 | R/W | 0000_0000h | 34.5.4.1/897 |

*表继续在下一页...*

内存图（续）

| 地址偏移（十六进制） | 注册名称 | 宽度（单位：比特） | 访问 | 重置值 | 节/页 |
|---|---|---|---|---|---|
| 1C | AHB RX数据缓冲寄存器（ARDB7）。 | 32 | R/W | 0000_0000h | 34.5.4.1/897 |
| 20 | AHB RX数据缓冲寄存器（ARDB8）。 | 32 | R/W | 0000_0000h | 34.5.4.1/897 |
| 24 | AHB RX数据缓冲寄存器（ARDB9）。 | 32 | R/W | 0000_0000h | 34.5.4.1/897 |
| 28 | AHB RX数据缓冲寄存器（ARDB10）。 | 32 | R/W | 0000_0000h | 34.5.4.1/897 |
| 2C | AHB RX数据缓冲寄存器（ARDB11）。 | 32 | R/W | 0000_0000h | 34.5.4.1/897 |
| 30 | AHB RX数据缓冲寄存器（ARDB12）。 | 32 | R/W | 0000_0000h | 34.5.4.1/897 |
| 34 | AHB RX数据缓冲寄存器（ARDB13）。 | 32 | R/W | 0000_0000h | 34.5.4.1/897 |
| 38 | AHB RX数据缓冲寄存器（ARDB14）。 | 32 | R/W | 0000_0000h | 34.5.4.1/897 |
| 3C | AHB RX数据缓冲寄存器（ARDB15）。 | 32 | R/W | 0000_0000h | 34.5.4.1/897 |
| 40 | AHB RX数据缓冲寄存器（ARDB16）。 | 32 | R/W | 0000_0000h | 34.5.4.1/897 |
| 44 | AHB RX数据缓冲器寄存器（ARDB17）。 | 32 | R/W | 0000_0000h | 34.5.4.1/897 |
| 48 | AHB RX数据缓冲寄存器（ARDB18）。 | 32 | R/W | 0000_0000h | 34.5.4.1/897 |
| 4C | AHB RX数据缓冲寄存器（ARDB19）。 | 32 | R/W | 0000_0000h | 34.5.4.1/897 |
| 50 | AHB RX数据缓冲寄存器（ARDB20）。 | 32 | R/W | 0000_0000h | 34.5.4.1/897 |
| 54 | AHB RX数据缓冲寄存器（ARDB21）。 | 32 | R/W | 0000_0000h | 34.5.4.1/897 |
| 58 | AHB RX数据缓冲寄存器（ARDB22）。 | 32 | R/W | 0000_0000h | 34.5.4.1/897 |
| 5C | AHB RX数据缓冲器寄存器（ARDB23）。 | 32 | R/W | 0000_0000h | 34.5.4.1/897 |
| 60 | AHB RX数据缓冲寄存器（ARDB24）。 | 32 | R/W | 0000_0000h | 34.5.4.1/897 |
| 64 | AHB RX数据缓冲器寄存器（ARDB25）。 | 32 | R/W | 0000_0000h | 34.5.4.1/897 |
| 68 | AHB RX数据缓冲寄存器（ARDB26）。 | 32 | R/W | 0000_0000h | 34.5.4.1/897 |
| 6C | AHB RX数据缓冲寄存器（ARDB27）。 | 32 | R/W | 0000_0000h | 34.5.4.1/897 |
| 70 | AHB RX数据缓冲器寄存器（ARDB28）。 | 32 | R/W | 0000_0000h | 34.5.4.1/897 |
| 74 | AHB RX数据缓冲寄存器（ARDB29）。 | 32 | R/W | 0000_0000h | 34.5.4.1/897 |
| 78 | AHB RX数据缓冲寄存器（ARDB30）。 | 32 | R/W | 0000_0000h | 34.5.4.1/897 |
| 7C | AHB RX数据缓冲寄存器（ARDB31）。 | 32 | R/W | 0000_0000h | 34.5.4.1/897 |

### 34.5.4.1 AHB RX数据缓冲寄存器（ARDBn）。

AHB的RX数据缓冲区寄存器0到31可以用来从连续的地址读取RX缓冲区的缓冲内容。QSPI_ARDB0对应于RX缓冲器的条目，对应于读取指针的当前值，顺序递增。

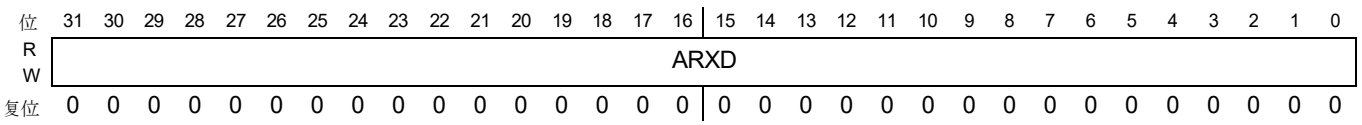
读取指针的增量取决于访问方案（DMA或标志驱动）。关于RX缓冲区内容的连续

访问描述，请参考闪存读取部分的"QuadSPI模块内部缓冲区的数据传输"，RX缓冲区，通过寄存器接口读取数据和AHB读取。字节排序方案也请参考串行闪存读取数据的字节排序。

在QSPI_ARDBn范围内可访问的有效地址范围取决于实现的RX缓冲区条目的数量和RX缓冲区中可用的有效缓冲区条目的数量。

- 例1，RX缓冲区完全被32个字填满。在这种情况下，有效读访问的地址范围从QSPI_ARDB0延伸到QSPI_ARDB31。
- 例2，RX缓冲区充满了5个有效字，RX缓冲区的填充水平QSPI_RBSR[RDBFL]为5。 在这种情况下，对QSPI_ARDB4的访问提供了最后一个有效条目。

地址。0h基数+0h偏移量+（4d×i），其中i=0d至31d




**ARDBn字段描述**

| 场地 | 描述 |
|---|---|
| ARXD | ARDB提供的RX缓冲区数据。<br><br>字节顺序(endianness)与RX缓冲器数据寄存器相同。 |

## 34.6 中断信号

QuadSPI模块的中断请求线根据下表中的内容映射到内部标志。

**表34-12.中断请求线的分配**

| IRQ/DMA线 | QSPI_FR 标志 | 中断描述 |
|---|---|---|
| 呼叫中心： 呼叫中心：<br>呼叫中心： 呼叫中心：<br>呼叫中心： 呼叫中心：<br>呼叫中心： 呼叫中心： | TBFF | TX缓冲区填充 |
| ipi_int_tcf | TFF | 周边命令交易完成 |
| 呼叫中心 | RBDF | RX缓冲器漏电 |
| ipi_int_overrun | | 缓冲区溢出/未运行错误<br><br>逻辑上的OR从。 |
| | RBOF | RX缓冲区超限 |
| | TBUF | TX缓冲区运行不足 |
| | ABOF | AHB缓冲区溢出 |
| ipi_int_cerr | | 串行闪存命令错误<br><br>逻辑上的OR从。 |
| | IPAEF | AHB繁忙时的外设访问 错误 |

| | |
|---|---|
| IPIEF | 周边命令不能被触发的错误 |
| IPGEF | 外围访问而AHB授予错误 |

*表继续在下一页...*

表34-12.中断请求线的分配（续）

| IRQ/DMA线 | QSPI_FR 标志 | 中断描述 |
|---|---|---|
| | IUEF | 周边命令使用错误 |
| ÃÃÃÃ | tbff, tff, illine, rbdf, rbof, tbuf, absef, abof, ipaef, ipief, ipgef, iuef, aitef,aibsef | 所有提到的QSPI_FR标志的逻辑OR |

## 34.7 功能描述

本节提供QuadSPI模块的功能信息。

### 34.7.1 串行闪存访问方案

根据连接到QuadSPI模块的串行闪存设备，可以采用以下访问方案。

表34-13.串行闪存数据访问的访问方案

| 访问计划 | 端口A的一个闪存设备 | 端口B上的一个闪存设备 | 在端口A和端口B上连接两个相同的闪存设备 |
|---|---|---|---|
| 个人闪光模式。访问闪光灯A | 是 | 不适用 | 是 |
| 个人闪光模式。访问闪光灯B | 不适用 | 是 | 是 |

**注意事项**

如果连接了两个不同的闪光设备，它们只能在单独闪光模式下操作。

在单独闪存模式下，所有支持的命令都可用。

除非明确指出，以下所有描述都与单独闪光模式有关。

### 34.7.2 正常模式

这种模式用于允许与外部串行闪存设备进行通信。与标准的SPI协议相比，这种通

信方式最多使用4条双向数据线，以高数据速率运行。与外部的通信

串行闪存设备包括一个指令代码和可选的地址、模式、假数和数据传输。下面描述的灵活的可编程核心引擎对各种闪存供应商提供的串行闪存设备的各种指令/协议差异具有免疫力。

### 34.7.2.1 可编程序列引擎

QuadSPI模块的核心是一个可编程的序列引擎，在 "指令-操作数 "对上工作。核心控制器按顺序执行每条编程的指令。指令和相应操作数的完整列表见下表。

**表34-14.指令集**

| 指示 | 指令编码 | 插销 | 操作数 | 对串行闪存采取行动 |
|---|---|---|---|---|
| 医学博士 | 6'd1 | N=2'd{0,1,2,3}。<br>2'd0 - 一个垫子<br>2'd1 - 两个垫子<br>2'd2 - 四个垫子<br>2'd3- 八个垫子 | 8位指令值 | 在指定的焊盘数量上提供带操作数的串行闪存 |
| 地址 | 6'd2 | | 要发送的地址位数（例如，8'd24 => 24需要的地址位） | 在指定的焊盘数量上，根据操作数向串行闪存提供地址周期。如果QSPI_SFACR[CAS]被设置为0，实际提供的地址将从AHB发起的交易的传入地址和IPS发起的交易的SFAR的值中得出，否则实际地址将考虑CAS。 |
| DUMMY | 6'd3 | | 虚拟时钟周期的数量（应<=64周期） | 按照操作数向串行闪存提供假周期。PAD信息定义了输入模式下的焊盘数量。(例如，一个焊盘意味着焊盘1不被驱动，其余的都被驱动) |
| 模式 | 6'd4 | | 8位模式值 | 在指定的焊盘数量上为串行闪存提供8位操作数。 |
| MODE2 | 6'd5 | N=2'd{0,1}。 | 2位模式值 | 在指定的焊盘数1上为串行闪存提供2位操作数。 |
| MODE4 | 6'd6 | N=2'd{0,1,2}。 | 4位模式值 | 在指定的焊盘数2上为串行闪存提供4位操作数。 |
| 阅读 | 6'd7 | N=2'd{0,1,2,3}。<br>2'd0 - 一个垫子<br>2'd1 - 两个垫子<br>2'd2 - 四个垫子<br>2'd3- 八个垫子 | 读取数据大小，以字节为单位（对于AHB交易，用户的应用程序应确保数据大小是8字节的倍数）。 | 在指定的垫子数量上从闪存中读取数据。对于AHB发起的交易，可以通过写入QSPI_BUFxCR寄存器的ADATSZ字段，对于IP发起的交易，可以通过写入IP配置寄存器（QuadSPI_IPCR）的IDATSZ字段来覆盖数据大小。 |

表继续在下一页...

## 表34-14.指令集（续）

| 指示 | 指令编码 | 插销 | 操作数 | 对串行闪存采取行动 |
|---|---|---|---|---|
| 写作 | 6'd8 | | 写入数据大小，以字节为单位 | 在分隔的焊盘数量上写入数据。数据大小可以通过写入IP配置寄存器（QuadSPI_IPCR）的IDATSZ字段来覆盖。 |
| 呼叫中心：JMP_ON_CS | 6'd9 | 这是对的。 | 指令号 | 每当芯片选择（CS）被解除时，跳到操作数所指向的指令。这条指令允许程序员指定当一个新的读事务在CS去势后启动时控制器的行为。 |
| ADDR_DDR | 6'd10 | N=2'd{0,1,2,3}。<br>2'd0 - 一个垫子<br>2'd1 - 两个垫子<br>2'd2 - 四个垫子<br>2'd3- 八个垫子 | 要发送的地址位数（例如，8'd24 => 24需要的地址位） | 在串行闪存时钟的每个时钟沿上，根据指定的焊盘数量向串行闪存提供地址周期。如果QSPI_SFACR[CAS]被设置为0，那么在AHB启动的交易中，实际提供的地址将来自传入的地址，而在IPS启动的交易中，QSPI_SFAR的值将会被考虑。 |
| MODE_DDR | 6'd11 | | 8位模式值 | 在串行闪存的每个时钟边缘，在指定的焊盘数量上为串行闪存提供8位操作数。 |
| MODE2_DDR | 6'd12 | N=2'd{0} | 2位模式值 | 在串行闪存的每个时钟边缘为指定的焊盘数量提供串行闪存的2位操作数3。 |
| MODE4_DDR | 6'd13 | N=2'd{0,1}。 | 4位模式值 | 在串行flash4的每个时钟沿上为指定的焊盘数量提供串行flash的4位操作数。 |
| READ_DDR | 6'd14 | N=2'd{0,1,2,3}。<br>2'd0 - 一个垫子<br>2'd1 - 两个垫子<br>2'd2 - 四个垫子<br>2'd3- 八个垫子 | 读取数据大小，以字节为单位（对于AHB交易，用户的应用程序应确保数据大小是8字节的倍数）。 | 在串行闪存的每个时钟沿指定的焊盘数量上从闪存读取数据。数据大小可以通过写给AHB发起的交易的QSPI_BUFxCR寄存器的ADATSZ字段和IP发起的交易的IP配置寄存器（QuadSPI_IPCR）的IDATSZ字段而被覆盖。 |
| WRITE_DDR | 6'd15 | | 写入数据大小，以字节为单位 | 在串行闪存的每个时钟沿指定的焊盘数量上写入数据。数据大小可以通过写入IP配置寄存器（QuadSPI_IPCR）的IDATSZ字段进行覆盖。 |
| 呼叫中心：CMD_DDR | 6'd17 | N=2'd{0,1,2,3}。<br>2'd0 - 一个垫子 | 8位指令值 | 向串行闪存提供操作数，并在串行闪存的每个时钟边缘指定垫子的数量 |

| | | | | |
|---|---|---|---|---|
| CADDR | 6'd18 | 2'd1 - 两个垫子<br>2'd2 - 四个垫子<br>2'd3- 八个垫子 | 要发送的列地址位数（8'd8表示要发送8位的列地址）。 | 在指定的焊盘数量上，根据操作数向串行闪存提供列地址周期。提供给闪存的实际地址将取决于QSPI_SFACR[CAS]的值。例如，如果QSPI_SFACR[CAS]是3，那么在AHB情况下，提供给闪存的地址将是传入地址的[2:0]，在IP情况下是QSPI_SFAR的值。如果QSPI_SFACR[CAS]小于闪存的焊盘数，则闪存的地址将被添加为零。 |
| 编码：CADDR_DDR | 6'd19 | | 专栏地址位数 | 在串行闪存的每个时钟沿上，根据操作数向串行闪存提供列地址周期，并指定焊盘数量。要提供给的实际地址是 |

*表继续在下一页...*

表34-14.指令集（续）

| 指示 | 指令编码 | 插销 | 操作数 | 对串行闪存采取行动 |
|---|---|---|---|---|
| | | | 要发送（8'd8表示要发送8位的列地址）。 | 闪存的地址将取决于QSPI_SFACR[CAS]的值。例如，如果CAS是3，那么在AHB的情况下，闪存的地址将是输入地址的[2:0]，在IP的情况下是QSPI_SFAR的值。如果CAS小于闪存的焊盘数，则该值将被附加为零。 |
| 停止 | 8'd0 | 这是对的。 | 这是对的。 | 停止执行；解除对CS的断言 |

1. 对于一条焊盘指令，MODE2将在闪存接口上花费2个串行闪存时钟周期。
2. 对于一条焊盘指令，MODE4将在闪存接口上占用4个串行闪存时钟周期。对于一个4焊盘指令，MODE4将在闪存接口上占用一个串行闪存时钟周期。
3. 对于一条焊盘指令，MODE2_DDR将在闪存接口上占用1个串行闪存时钟周期。
4. 对于一条焊盘指令，MODE4_DDR将在闪存接口上花费2个串行闪存时钟周期。对于一个4焊盘指令，MODE4_DDR将在串行闪存接口上花费半个周期。

可编程序列引擎允许用户根据板上连接的串行闪存来配置QuadSPI模块。这种灵活的结构很容易适应不同供应商的新命令/协议变化。

### 34.7.2.2 灵活的AHB缓冲区

为了减少AHB主站的读取延迟，从串行闪存中读取的数据被缓冲在灵活的AHB缓冲区中。有四个这样的灵活缓冲区。每个缓冲区的大小都是可配置的，最小大小为0字节，最大大小为整个缓冲区的实例化大小。缓冲区1的大小是从QSPI_BUF0IND到QSPI_BUF1IND，缓冲区2是从QSPI_BUF1IND到QSPI_BUF2IND，缓冲区3是从QSPI_BUF2IND到完整缓冲区的大小，这在芯片特定的QuadSPI信息中给出。

每个灵活的AHB缓冲区都与以下内容有关

1. 一个AHB主站。通过设置QSPI_BUF3CR[ALLMST]位，可以选择将缓冲区3配置为 "所有主控
   "缓冲区。当缓冲区3被配置为这种方式时，任何来自与任何其他缓冲区无关的主站的访问都会被路由到缓冲区3。
2. 一个数据量字段，代表每次 "错过 "访问时要从闪存中获取的数据量。

每一个传入的请求的主端口号被检查，数据被返回/取到相应的相关缓冲区。每一次对缓冲区的 "错过

"访问都会导致控制器清空缓冲区并获取QSPI_BUFxCR[ADATSZ]数量的数据。

因此，配置一个大于ADATSZ的缓冲区大小是没有好处的，因为大于ADATSZ的位置永远不会被使用。对于任何AHB访问，QSPI_BFGENCR[SEQID]字段所指向的序列被用于启动闪存交易。一旦从串行闪存中读出所要求的数据量，数据就会返回给主控器。然而，控制器继续预取其余的数据，以期待下一个连续请求。图34-4显示了灵活的AHB缓冲区。

QSPI_BFGENCR[SEQID]字段指向LUT的一个索引。详情请参考查找表。

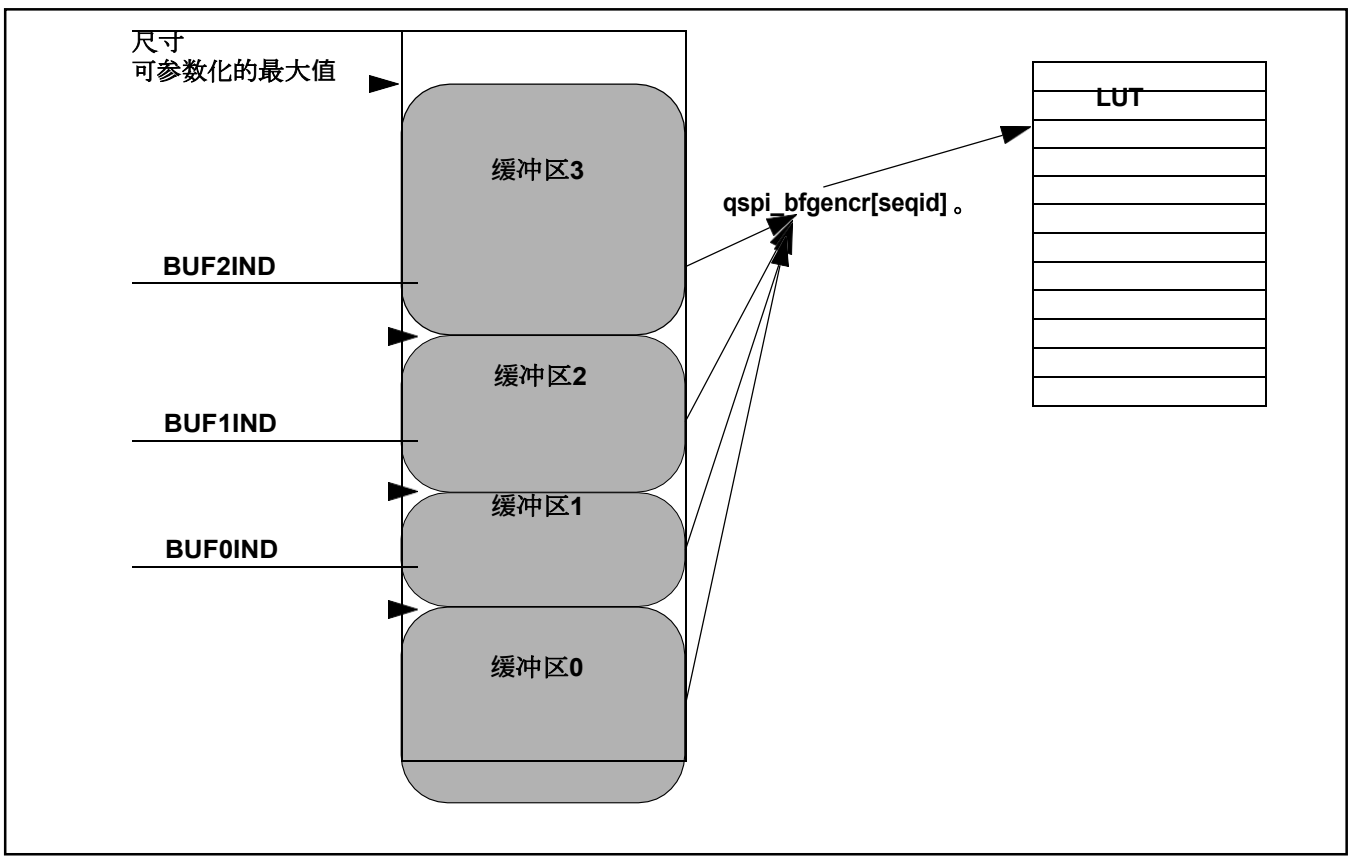



**图34-4.灵活的AHB缓冲区**

缓冲区0可以通过设置QSPU_BUF0CR[HP_EN]位，选择性地被配置为与高优先级主站相关联。高优先级主站的访问会暂停任何正在进行的对其他缓冲区的预取。正在进行的预取被暂停，高优先级的主站首先被服务。一旦高优先级主站访问完成，暂停的事务就会恢复（在任何其他AHB访问被受理之前）。暂停的缓冲区的状态可以从序列暂停状态寄存器（QuadSPI_SPNDST）读取。

#### 34.7.2.3 暂停-中止机制

任何低优先级的AHB访问都可以被高优先级的AHB主请求暂停。正在进行的交易在64位边界被暂停。暂停的事务在高优先级主控器提供服务和包括数据预取的高优先级事务完成后重新启动。当一个交易处于暂停状态时，如果同一个暂停的主控器对一个与暂停交易的位置不同的位置进行交易，该交易可能被中止。

如果来自同一个主站的请求到达一个与正在进行的交易地点不同的地点，任何正在进行的交易都会被中止。终止可能发生在任何时间点。

#### 34.7.2.4 HBURST支持

QuadSPI控制器在AHB接口上支持HBURST和HSIZE。HBURST表示传输是否构成一个突发的一部分。支持4个、8个和16个拍子的突发，突发可以是递增的，也可以是包裹的。HSIZE表示传输的大小。支持8、16、32和64位数据大小。在WRAP访问的情况下，如果任何进入的非连续AHB访问没有缓冲区被击中，QuadSPI将生成对串行闪存的对齐访问。如果有缓冲区被击中，haddr线中传入的地址将被锁定。如果总突发大小超过了数据预取大小，就会产生错误响应，并设置QSPI_FR[AIBSEF]。数据预取大小可以由QSPI_BUFxCR[ADATSZ]定义，或者当ADATSZ被编程为0时，SEQID字段所指向的序列中提到的数据大小。下图显示了几个例子。

HADDR = 0x38
HBUST = WRAP4
HSIZE = 64位
Flash xsaction start = 0x20
进入的AHB访问= 0x38, 0x20, 0x28, 0x30

HADDR = 0x38
HBUST = INCR4
HSIZE = 64位
Flash xsaction start = 0x38
进入的AHB访问= 0x38, 0x40, 0x48, 0x50

HADDR = 0x50
HBUST = WRAP8
HSIZE = 64位
Flash xsaction start = 0x40
传入AHB访问= 0x50, 0x58, 0x60, 0x68, 0x70, 0x78, 0x40, 0x48

HADDR = 0xD0
HBUST = WRAP16
HSIZE = 64位
Flash xsaction start = 0x80
传入AHB访问= 0xD0, 0xD8, 0xE0, ...0xF8, 0x80, 0x88, ...0xC8

HADDR = 0xD4
HBUST = WRAP8
HSIZE = 32bits
Flash xsaction start = 0xC0
传入AHB访问= 0xD4, 0xD8, 0xDC, 0xC0, 0xC4, 0xC8,0xCC, 0xD0

HADDR = 0x54
HBUST = INCR8
HSIZE = 32bits
Flash xsaction start = 0x54
传入AHB访问= 0x54, 0x58, 0x5C, 0x60, 0x64, 0x68,0x6C, 0x70

**图34-5.QuadSPI HBURST支持**

**注意事项**

软件必须注意，预取大小的设置绝不应低于任何外部接口开始处理所需的最小数据。

### 34.7.2.5 查询表

查找表或LUT由一些预先编程的序列组成。每个序列基本上是一个指令-操作数对的序列，当按顺序执行时产生一个有效的串行闪存交易。每个序列最多可以有8个指令-操作数对。LUT最多可以容纳16个序列。下图显示了LUT中序列的基本结构。

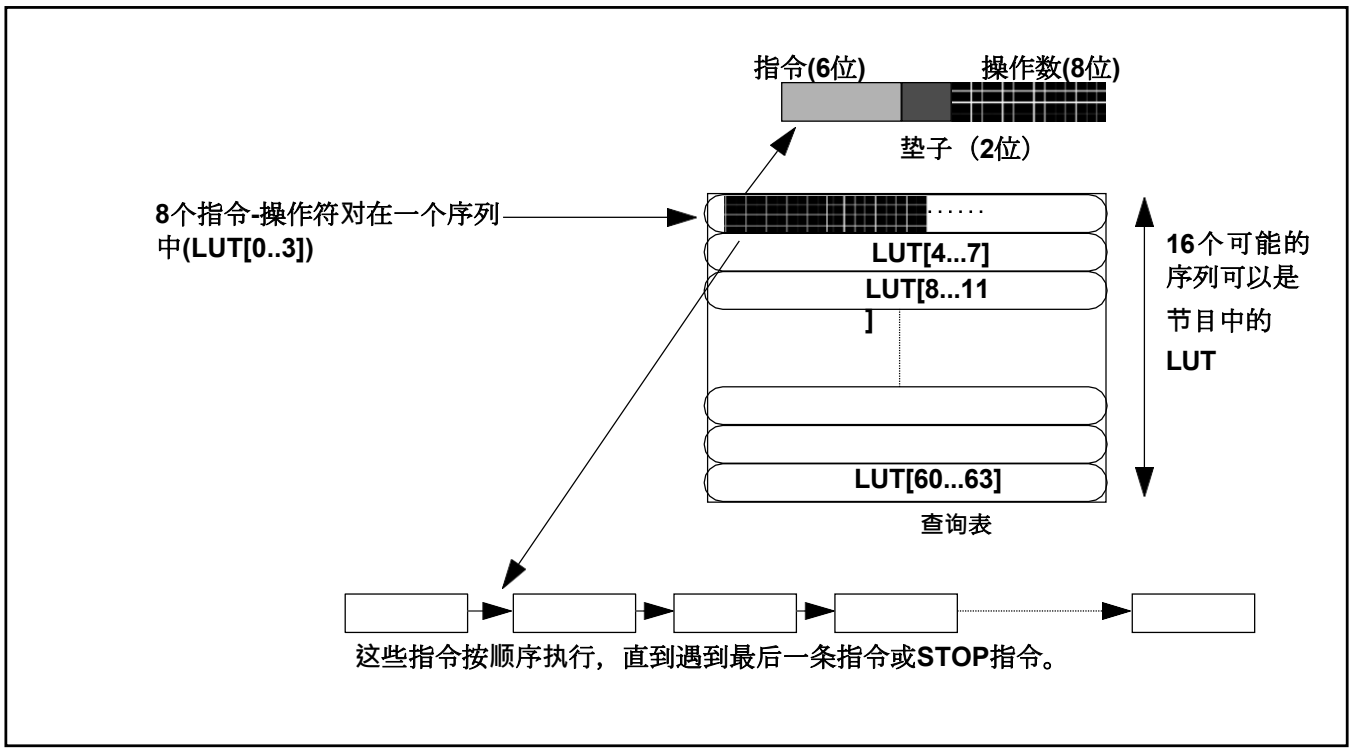



**图34-6.LUT和序列结构**

复位时，查找表（LUT[0...3]）的索引0被编程，其基本读取顺序见表34-15。复位后，整个LUT可以根据板上连接的设备重新编程。为了在代码运行期间保护其内容，LUT可以被锁定，之后对LUT的写入将不成功，直到它再次被解锁。锁定或解锁LUT的钥匙是**0x5AF05AF0**。锁定和解锁LUT的过程如下。

### 锁定LUT

1. 将密钥（**0x5AF05AF0**）写入LUT密钥寄存器（QuadSPI_LUTKEY）中。
2. 将0b01写入LUT锁定配置寄存器（QuadSPI_LCKCR）。请注意，这个IPS交易应该紧随上述IPS交易之后（中间不能有其他IPS交易）。成功写入该寄存器后，将锁定LUT。

### 解锁LUT

1. 将密钥（**0x5AF05AF0**）写入LUT密钥寄存器（QuadSPI_LUTKEY）中。
2. 向LUT锁定配置寄存器（QuadSPI_LCKCR）写入0b10。请注意，这个IPS交易应该紧随上述IPS交易之后（中间不能有其他IPS交易）。成功写入该寄存器后，LUT将被解锁。

LUT的锁定状态可以从QSPI_LCKCR[UNLOCK]和QSPI_LCKCR[LOCK]位读取。

一些实例序列在实例序列中定义。LUT索引0处的复位序列在下表中给出。

**表34-15.复位顺序**

| 指示 | 垫子 | 操作数 | 评论 |
|---|---|---|---|
| 医学博士 | 0x00 | 0x03 | 在一个焊盘上读取数据字节的命令 |
| 地址 | 0x00 | 0x18 | 在一个焊盘上发送24个Addr位 |
| 阅读 | 0x00 | 0x08 | 读取64位 |
| 呼叫中心：JMP_ON_CS | 0x00 | 0x00 | 跳到指令0（CMD）。 |

### 34.7.2.6 发布串行闪存（SFM）命令

对外部设备的每次访问都遵循相同的顺序。

1. 用户必须用所使用的闪存设备所需的串行闪存命令序列来预先填充LUT。
2. QuadSPI模块开始逐一执行该序列中的指令。交易开始，状态位QSPI_SR[BUSY]被设置。
3. 与外部串行闪存设备的通信被启动，交易被执行。
4. 当交易完成后（与外部串行闪存设备的所有发送和接收操作都已完成），状态位QSPI_SR[BUSY]被重置。在IP命令的情况下，QSPI_FR[TFF]标志被断电。

更多的细节在下面的Flash编程和Flash读取中给出。

你可以通过以下方式之一在QuadSPI模块中触发对SFM命令的处理。

- **使用IP命令**

  对于IP命令，所需的组件需要写入以下寄存器中。

  - 将指令使用的串行闪存地址写入QSPI_SFAR中，请参考串行闪存地址寄存器（QSPI_SFAR）。对于不涉及具体地址的IP指令，需要将相关闪存的基本地址写入QSPI_SFAR。

例如，对于一个不需要地址的指令（即写使能指令），SFAR应该被编程为该指令所要发送的存储器的基址。

- 在IP配置寄存器（QSPI_IPCR）中写入序列ID和数据大小的详细信息。
- 注意，对QSPI_IPCR[SEQID]字段的写入必须是序列的最后一步。可以将QSPI_IPCR的所有字段合并为一个单一的写入。详情请参考IP配置寄存器（QSPI_IPCR）。

请注意，在某些情况下，在写入QSPI_IPCR[SEQID]字段后，没有IP命令被执行，写入操作本身被忽略。它们将在命令仲裁中描述。

- **使用AHB命令**

  任何AHB内存映射的访问都被路由到其中一个缓冲区，这取决于请求的主端口号。如果访问是
  "错过"，就会启动一个新的串行闪存事务。该事务是基于BFGENCR[SEQID]字段所指向的序列，如《灵活的AHB缓冲区》所述。

  当对内存映射的串行闪存进行访问时，AHB访问被称为内存映射，如内存映射的串行闪存数据--闪存A上的独立闪存模式和内存映射的串行闪存数据--闪存B上的独立闪存模式中所述。

  同样，可能的错误条件在命令仲裁中有所描述。

### 34.7.2.7 闪存编程

在所有情况下，要写入的内存扇区都需要先被擦除。然后以下列方式启动编程程序本身。

1. 检查QuadSPI_SR[BUSY]是否去势或为"0"。检查TX缓冲器是否为空。如果需要丢弃TX缓冲区中的数据（QSPI_SR[TXEDA]）位被设置，那么必须通过向QSPI_MCR[CLR_TXF]位写'1'来清除TX缓冲区。
2. 在QSPI_SFAR寄存器中编程与命令有关的地址。QSPI_SFACR[CAS]在需要的情况下被编程为所需的值，否则为0。如果串行闪存是可字寻址的闪存，QSPI_SFACR[WA]被编程为1，如果串行闪存是可字节寻址的，则为0。

3. 通过寄存器TX缓冲区数据寄存器（QSPI_TBDR）向循环缓冲区提供程序命令的初始数据。至少有4个字的数据必须写入TX缓冲器，最多不超过32个。
4. 对QSPI_IPCR寄存器进行编程以触发该命令。QSPI_IPCR[SEQID]应该指向LUT的一个索引，该索引已经预先编程了闪存程序序列。IDATSZ字段应该被设置为表示写的大小。
5. 根据所需的数据量，必须重复步骤3，直到所有需要的数据都被写入QSPI_TBDR寄存器。QSPI_SR[TXFULL]可以用来检查缓冲器是否准备好接收更多的数据。在任何时候，都可以读取QSPI_TBSR[TRCTR]字段来检查有多少字被实际写入TX缓冲区。

在写入QSPI_IPCR[SEQID]字段后（参考步骤4），QuadSPI模块将开始执行编程的序列。软件有责任确保按照连接到模块的闪存将正确的序列编程到LUT中。数据从TX缓冲器中获取。它由**32个**32位的条目组成，被组织成一个循环的FIFO，其读指针在每次取数后增加4。当所有的数据都传输完毕后，QuadSPI模块将从 "忙 "返回到
"闲"。但是，对于外部设备来说，情况并非如此，因为内部编程仍在进行。这就需要用户监控来自串行闪存设备的相关状态信息，并确保编程工作正常完成。

### 34.7.2.8 闪读

主机对存储在外部串行闪存设备中的数据的访问分两步进行。首先，数据必须被读入内部缓冲区，在第二步，这些内部缓冲区可以被主机读取。

1. **将串行闪存数据读入QuadSPI模块内部缓冲区**

   对外部串行闪存设备的读访问可以通过两种不同的方式触发。

   - **IP命令读取**。对于**向RX缓冲区读取闪存数据**，用户必须在QuadSPI_IPCR[SEQID]寄存器中提供正确的序列ID。该序列ID指向LUT中的一个序列。软件有责任确保根据板上连接的串行闪存设备，在LUT中编入正确的读取序列。用户应该

对串行闪存地址寄存器（QSPI_SFAR）、QSPI_SFACR[CAS]和IP配置寄存器（QSPI_IPCR）进行编程。所有外部串行闪存支持的可用读取命令都可以实现。

在触发IP命令之前，可以通过在QuadSPI_MCR[CLR_RXF]字段中写1来清除RX缓冲器指针。

从这些输入中，当QSPI_IPCR[SEQID]字段被写入时，完整的交易被建立起来。与读访问有关的事务开始了，要求的字节数从外部串行闪存设备中被取到RX缓冲器中。由于读访问是由IP命令触发的，IP_ACC状态位和BUSY位都被设置（两者都位于状态寄存器（QSPI_SR）中）。从QSPI_RBSR[RDBFL]可以获得当前在Rx Buffer中的条目数。

当读完指定的字节数后，与外部串行闪存的通信就停止了（交易成功完成）。

- **AHB命令读取**。为了**将闪存数据读入AHB缓冲区**，用户必须设置一个主站对外部串行闪存设备所映射的系统内存地图中的地址范围进行读取访问。用户应该将QSPI_SFACR[CAS]（如果需要的话）编程为所需的值，否则为0。用户还应该对发起请求的AHB主站对应的缓冲器进行编程，这取决于QSPI_RBCT[RXBRD]的配置。用户应该在缓冲区通用配置寄存器（QSPI_BFGENCR）中提供正确的序列ID。软件有责任确保根据板上连接的串行闪存设备，在LUT中编入正确的读序列。闪存设备的选择和访问模式由与QuadSPI模块相关的AHB地址空间中的访问地址决定（参考内存映射的串行闪存数据--闪存A的单独闪存模式，内存映射的串行闪存数据--闪存B的单独闪存模式

  在每次AHB读访问内存映射区域时，AHB缓冲区中的有效数据都会与实际读请求的地址进行检查。当AHB缓冲区的内容不能满足AHB读取请求时，缓冲区被刷新，序列ID所指向的序列被控制器执行。然后，在QSPI_BUFxCR[ADATSZ]字段中定义的缓冲区条目的请求数量被从外部串行闪存设备取到内部AHB缓冲区。由于读访问是通过AHB总线触发的，QSPI_SR[AHB_ACC]状态位被设置，依次驱动

QSPI_SR[BUSY]位，直到交易完成。当指定的条目数被填满后，与外部串行闪存的通信就会停止。

## 2. 从QuadSPI模块内部缓冲区传输数据

QuadSPI模块从外部串行闪存设备中读出的数据存储在内部缓冲器中。从缓冲器中访问数据的方法因数据被加载到哪个缓冲器而不同。关于QuadSPI模块中两个可用的缓冲区，即RX缓冲区和AHB缓冲区的详情，请参考方框图。

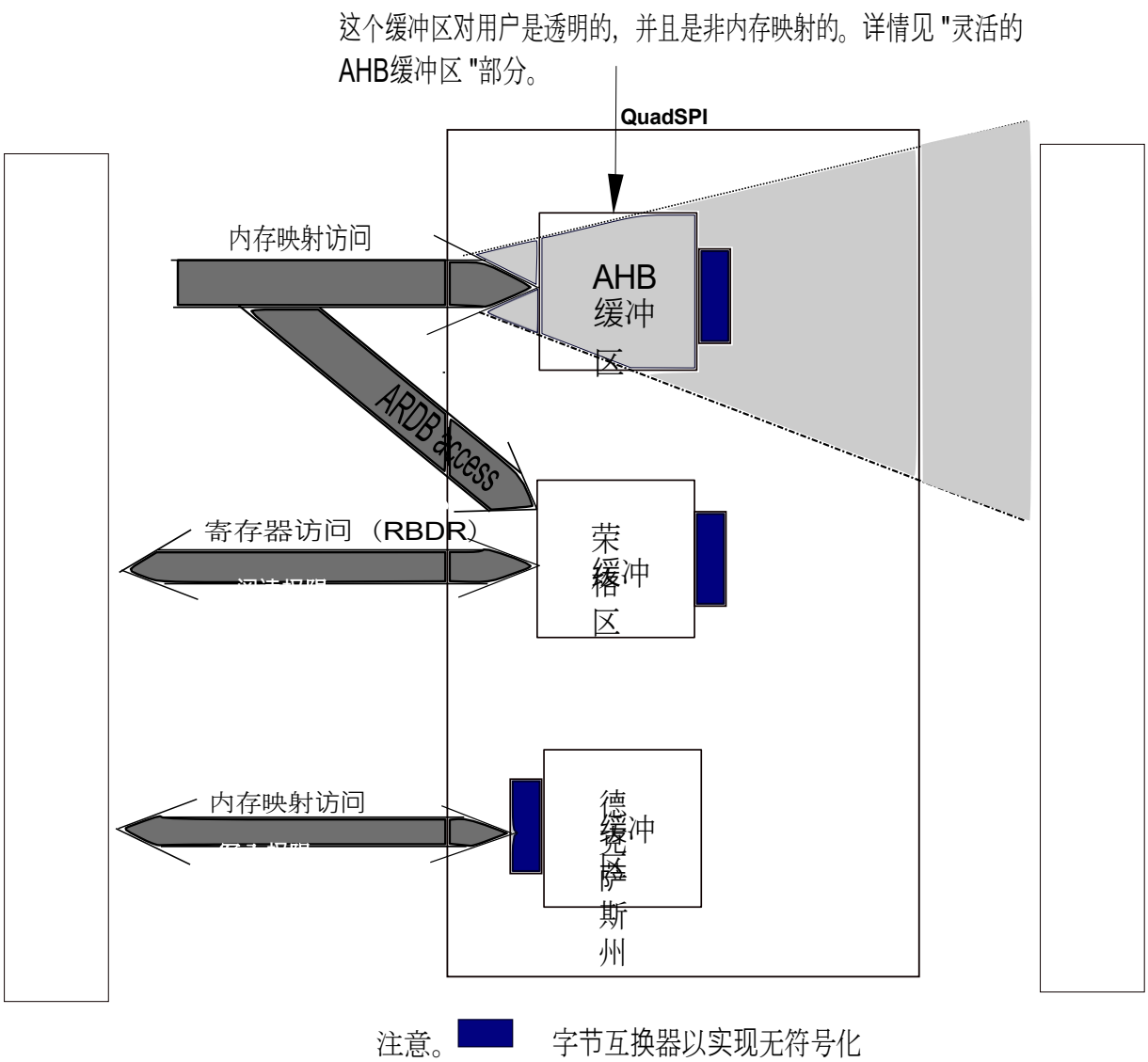



**图34-7.QuadSPI内存图**

- RX缓冲器是作为深度为4字节的32个条目的FIFO来实现的。它的内容可以在两个不同的地址区访问，这两个地址区指的是相同的数据和相同的物理存储器。

在IPS地址空间中与QSPI_RBDR0至QSPI_RBDR31相关的区域中

在AHB地址空间中与QSPI_ARDB0至QSPI_ARDB31相关的区域。通过一个单一的64位AHB读操作访问两个连续的条目。

RX缓冲区的操作可以总结为以下几点。QSPI_RBCT[WMRK]字段决定了RXWE位在哪个填充电平上被断言，以及在每次Buffer POP操作中从RX Buffer中删除多少个条目。因此，QSPI_SR[RXWE]位表示RX缓冲区内有配置好的数据条目数，QSPI_RBSR[RDBFL]字段表示总共有多少有效条目。请注意，第一个条目（QSPI_RBDR0或QSPI_ARDB0）总是对应于RX缓冲区的第一个有效条目。软件需要自己管理有效数据字节的数量。

更多的细节可以在RX缓冲区数据寄存器（QuadSPI_RBDRn）和AHB RX数据缓冲区（QSPI_ARDB0至QSPI_ARDB31）中找到。

- **基于标志的RX缓冲区数据读取**是通过轮询QSPI_SR[RXWE]位进行的。当它被断言时，可以通过IPS地址空间（QSPI_RBDRn）或AHB地址空间（QSPI_ARDBn）读取有效条目。缓冲区POP操作必须由应用程序通过向QSPI_FR[RBDF]位写1来触发 -
  这将自动更新FIFO，使其指向RBCT[WMRK]定义的下一个条目。例如，如果WMRK被设置为3，那么缓冲区将丢弃16字节的数据。

- **DMA控制的RX缓冲区的数据读取**是通过DMA模块完成的。应用程序必须确保相关设备的DMA控制器被适当编程，就像DMA使用中描述的那样。

  DMA控制的读出是由QSPI_SR[RXWE]位的断言完全自动触发的。相关的缓冲区POP操作也完全在QuadSPI模块内部处理。和上面的情况一样，访问RX缓冲区的内容无论是在QSPI_RBDRn还是在QSPI_ARDBn相关的地址上都是等同的。

- **通过内存映射访问读取AHB Buffer数据。**这种访问是通过在表34-9中给出的范围内读取分配给外部串行闪存设备的一个地址来完成的，*条件是要求的数据已经存在于AHB Buffer中，或者它目前正在被正在进行的指令从串行闪存设备中读取*。如果不是这种情况，就会触发上述的内存映射AHB指令读取。如果要求的数据在AHB Buffer中已经存在，它们将直接提供给主机。

当AHB访问闪存映射的地址时，数据将被取走并返回到AHB接口。在数据被取走之前，AHB接口将处于停滞状态。一旦QuadSPI模块从请求的地址读取了数据，AHB的读取访问就会得到满足。因此，有可能以任意的速度从AHB缓冲区进行连续的读取，而不需要监控任何关于数据可用性的信息。
然而，这种访问方案使AHB总线在从串行闪存设备中读取数据所需的时间内停滞。一个更好的方法是（当知道访问是连续的时候）启用预取（通过对ADATSZ字段编程），这样在下一个连续的AHB访问到来之前，数据已经被取到了缓冲区。

只要主机将其访问限制在已经在缓冲器中的数据和当前从串行闪存中获取的数据，就有可能在串行闪存读入AHB缓冲器的同时运行主机从AHB缓冲器的读取。

### 34.7.2.9 串行闪存读取数据的字节排序

在本段中给出了串行闪存数据的字节排序。基本的方案是，从串行闪存设备中读出的第一个字节--由QSPI_SFAR[SFADR]字段寻址--
对应于IP命令读取的QSPI_RBDR0[ ]
寄存器的位位置。与此相反，对于AHB命令的读取，字节总是按照AHB总线的字节排序来定位。

- **单独闪光模式下的字节排序**

  下表给出了串行闪存设备的面向字节的数据空间如何被映射到RX缓冲器或AHB缓冲器的一个单一的32位条目中的字节排序方案。该表在以下情况下有效。

  - 单独闪光模式下的闪光A或闪光B
  - 所有访问大小为32位的AHB数据读取命令

**表34-16.单独闪存模式下的字节排序**

**注意事项**

对于IP命令，读取的大小可以用字节数给出。如果这个数字不是4的倍数，那么最后一个缓冲区条目就不会完全被未定义值的缺失的高字节填满。

对于AHB命令，读，从一个不与32位边界对齐的地址开始，根据AMBA的AHB规范，在适当的位置给出要求的字节。

- **64位读访问的缓冲区条目排序**

  对于通过AHB接口的读访问，可以进行64位访问。每个64位访问同时读取2个32位条目。这些32位条目在64位字中的顺序在下表中给出。

**表34-17.64位读访问缓冲区条目排序**

## 34.7.2.10 正常模式中断和DMA请求

QuadSPI模块有不同的标志，只能产生中断请求，还有一个标志既能产生中断，又能产生DMA请求。下表列出了这八种情况。注意，表中提到的标志与标志寄存器（QSPI_FR）有关。

**表34-18.中断和DMA请求条件**

| 状况 | 标志(QSPI_FR) | DMA |
|---|---|---|
| TX缓冲区填充 | TBFF | - |
| TX缓冲区运行不足 | TBUF | - |
| 非法指令错误 | 伊利兰 | - |
| RX缓冲器漏电 | RBDF | X |
| RX缓冲区溢出 | RBOF | - |
| AHB缓冲区溢出 | ABOF | - |
| AHB 序列错误 | ABSEF | - |
| AHB非法交易错误 | AITEF | - |
| AHB非法突发大小错误 | AIBSEF | - |
| 在AHB访问错误时触发IP命令 | IPAEF | - |

*表继续在下一页...*

表34-18.中断和DMA请求条件（续）

| 状况 | 标志(QSPI_FR) | DMA |
|---|---|---|
| IP命令触发器无法执行错误 | IPIEF | - |
| AHB授权期间的IP访问错误 | IPGEF | - |
| 与IP命令相关的交易完成了 | TFF | - |

每个条件在标志寄存器（QSPI_FR）中有一个标志位，在DMA请求选择和启用寄存器（QSPI_RSER）中有一个请求启用位。RX Buffer Drain Flag (RBDF)有单独的使能位用于产生IRQ和DMA请求。请注意，并非所有的标志都有单独的IRQ线。检查设备的中断向量表以了解更多细节。

- 发送缓冲区填充中断请求。

  传输缓冲区填充IRQ表示TX缓冲区可以接受新数据。如果QSPI_FR[TBFF]标志被断言，并且相应的使能位（QSIP_RSER[TBFIE]）被设置，它就被断言。关于QSPI_FR[TBFF]标志断言的细节，请参考TX缓冲器操作。

  除了IRQ之外，还可以通过DMA来处理TX Buffer的填充。如果QSPI_RSER[TBFDE]位被设置，当TX Buffer中的可用空间数量超过QSPI_TBCT[WMRK]的有效条目且QSPI_SR[TXWA]被设置时，将触发一个DMA请求。应用程序必须为 DMA 传输适当地设置环境（例如，DMA 控制器）。

- 接收缓冲器耗尽中断或DMA请求。

  来自QSPI_FR[RBDF]标志的接收缓冲器泄放IRQ表明QuadSPI模块的RX缓冲器有来自串行闪存设备的数据可供主机读取。只要QSPI_RBSR[RXWE]位被设置，它就一直被设置。
  QSPI_RSER[RBDIE]位启用相关的IRQ。

  除了IRQ之外，还可以通过DMA来处理RX Buffer的耗尽。如果QSPI_RSER[RBDDE]位被设置，当RX Buffer包含超过QSPI_RBCT[WMRK]的有效条目时，将触发一个DMA请求。

- 缓冲区溢出/超限中断请求。

  缓冲区溢出/超限IRQ是下列标志的组合（都位于QSPI_FR寄存器中，相关的使能位在QSPI_RSER寄存器中）。

- TBUF - TX缓冲区欠运行，由TBUIE启用
- RBOF - RX缓冲区溢出，由RBOIE启用
- ABOF - AHB缓冲区溢出，由ABOIE启用

  传输缓冲区未用指示TX缓冲区发生未用情况。当Tx Buffer为空且QSPI_RSER[TBUIE]位被设置时，写指令被触发，就会产生这种情况。

  接收缓冲区溢出表明RX缓冲区发生了溢出情况。当RX缓冲区满了，一个额外的读传输试图写入RX缓冲区，并且QSPI_RSER[RBOIE]位被设置时，它就会产生。

  AHB缓冲区溢出表明AHB缓冲区发生了溢出情况。当AHB缓冲区满了，一个额外的读传输试图写入AHB缓冲区，并且QSPI_RSER[ABOIE]位被设置时，它就会产生。

  产生单个溢出条件的传输的数据被忽略了。

- 串行闪存命令错误中断请求

  如果QSPI_FR中的IPAEF、IPIEF、IPGEF标志被设置，并且QSPI_RSER中的相关中断使能位也被设置，那么就会请求一个中断。

- 交易完成的中断请求

  IP命令交易完成IRQ表示当前IP命令的完成。它由QSPI_FR[TFF]标志触发，并由QSPI_RSER[TFIE]位屏蔽。

### 34.7.2.11 TX缓冲器操作

TX缓冲器提供用于页面编程的数据。为了正确操作，在开始执行页面编程命令之前，需要在TX Buffer中提供至少四个条目。应用程序必须确保所需的数据字节数足够快地写入TX缓冲区，只要执行命令时不出现TX缓冲区溢出或欠载。

只要TX缓冲区没有满，可以接受更多的数据，QuadSPI模块就会设置QSPI_FR[TBFF]标志。

当QuadSPI模块试图从一个空的TX缓冲区拉出数据时，TX缓冲区欠载由QSPI_FR[TBUF]标志发出信号。当TX缓冲区包含的数据少于128位，QuadSPI模块试图从其中拉出数据时，TX缓冲区欠载标志也被提出来。在欠载条件下，当QuadSPI模块试图从空的TX缓冲区中取出数据时，传输的数据都是F，也就是说，一旦欠载标志在此条件下被设置，它将返回F，直到所需的字节数没有被发送。这样做是为了确保软件不需要在欠载后擦除整个扇区，只要从故障点重新编程就可以了。当这个序列命令完成后，QSPI_FR[TBFF]标志被引爆，表明Tx缓冲区可以再次被写入。

TX缓冲区的溢出没有明确的信号，但是TX缓冲区的填充水平可以通过QSPI_TBSR[TRBFL]字段监控。

有关TX缓冲区相关寄存器的详细信息，请参考TX缓冲区状态寄存器（QuadSPI_TBSR）和标志寄存器（QuadSPI_FR）。

### 34.7.2.12 地址计划

早期的串行闪存只支持24位地址空间，因此限制了串行闪存的最大内存大小为16 MB。除了传统的24位地址模式外，新的内存规范还支持两种32位寻址模式。它还支持地址编程的分离，根据要求将闪存分为行地址和列地址。

- **扩展地址模式**

  在这种模式下，传统的24位指令被转换为接受32位地址指令。闪存需要被配置为32位地址模式。另外，在QuadSPI中对32位模式的LUT序列进行编程时，ADDR和ADDR_DDR命令应以8'd32作为操作数，QSPI_SFACR[CAS]编程为0。
  如果闪存需要某些位的地址作为其列地址，那么必须始终考虑闪存所需的总位数不应超过32，因为QuadSPI支持的最大地址是32位。
  每个内存供应商都有不同的方式来启用这种模式（参考内存供应商的内存规范）。例如，向Macronix闪存发送的B7h命令将使其进入32位地址模式。

- **扩展地址寄存器**

在这种模式下，32位地址的前8位是由存储器本身的扩展地址寄存器提供的。存储器提供了一个特定的寄存器，根据要访问的地址进行更新。这有效地将传统的24位地址命令转换为32位地址命令。大于16MB的存储器，由16MB的库组成。写在扩展地址寄存器中的8位有效地启用了一个库。例如，在Spansion存储器中，当扩展地址寄存器在命令17h的帮助下被更新为0x01的值时，它将打开存储器的Bank1。随后的24位地址命令将指向Bank1。扩展地址寄存器需要用相应的值进行更新，以便访问其他银行。这就有效地转换了传统的24位地址命令变成32位地址命令。

- **将地址分离成行和列地址**

  这种模式是为需要将地址分隔成行和列的闪存而设立的。QSPI_SFACR[CAS]中的值定义了闪存所需的列地址的宽度。如果QSPI_SFACR[CAS]被设置为0，那么实际的地址将从AHB启动的交易的传入地址和IPS启动的交易的SFAR值中得出，否则实际地址将考虑CAS。如果QSPI_SFACR[CAS]为3，那么在闪存工作在24位模式的情况下，编程的地址的第26-3位将作为页面地址发送到闪存，第2-0位将作为列地址发送。如果闪存对列地址的要求小于需要发送地址的焊盘数，QuadSPI会将剩余的位附加为0。用户必须对CADDR和CADDR_DDR命令中的操作数进行相应的编程。必须确保闪存请求的地址位总数不超过32位，作为其页和列地址。

- **闪存的字寻址模式**

  这种模式适用于具有字寻址存储器的闪存，即闪存的每个地址包含一个字（两个字节）的数据。QSPI_SFACR[WA]被设置为1以进入该模式。QuadSPI在内部对AHB总线上的传入地址或QSPI_SFAR中的地址进行划分，以将其映射到有效的闪存位置。例如，如果传入的地址是0x2004，控制器会重新映射这个地址，以访问闪存位置0x1002。如果不是在这种模式下，传入的地址0x2004将被映射到闪存位置0x2004。

### 34.7.3 支持HyperRAM

QuadSPI支持HyperRAM存储器，凭借这一协议，QuadSPI支持以下功能。

- 双向数据选通/读写数据选通（RWDS）。
  - 当QuadSPI被配置为使用HyperRAM模式时，RWDS焊盘应被拉低。
- 可变的刷新延时
  - 如果QuadSPI_MCR[VAR_LAT_EN]字段被设置，基于命令/地址阶段来自HyperRAM的RWDS的状态，QuadSPI包括额外的初始访问延迟。如果RWDS为高，QuadSPI将包括在DUMMY阶段提到的两倍+1的延迟。如果RWDS为低，则只包括DUMMY阶段中提到的延迟。
  - 在QuadSPI配置寄存器读/写过程中，DUMMY不是LUT编程的一部分，因此RWDS的状态不会被QuadSPI考虑。
  - 如果没有设置QuadSPI_MCR[VAR_LAT_EN]，则DUMMY阶段提到的固定延时将被包括在内。
- 读取数据选通，从HyperRAM锁存数据
- 写入时的数据屏蔽
  - 如果QuadSPI_MCR[DQS_OUT_EN]启用。
  - RWDS在写数据阶段由QuadSPI驱动为低电平。
  - 不支持数据屏蔽，因为写数据的大小应该是16位对齐的。

## 34.8 初始化/应用信息

本节提供QuadSPI模块的初始化和应用信息。

### 34.8.1 上电和复位

请注意，连接到QuadSPI模块的串行闪存设备在上电或复位时可能需要其输入的特殊电压特性。确保这一点是应用程序的责任。

### 34.8.2 可用的状态/旗帜信息

本段概述了不同的状态和标志信息，以及它们在不同使用情况下的相互依赖关系。相关的寄存器是QSPI_SR和QSPI_FR。请参考相关描述，如何适当地设置QuadSPI模块。

#### 34.8.2.1 IP命令

关于本段没有明确涉及的其他细节，请参考IP配置寄存器（QuadSPI_IPCR）。

- **IP命令--正常操作**

  写入QSPI_IPCR[SEQID]字段会触发一个新的IP命令的执行。考虑到这是一个合法的命令，QSPI_SR[IPACC]和QSPI_SR[BUSY]位在执行开始后立即被断电。

  当串行闪存设备上的指令完成后，这些位将被去掉，QSPI_FR[TFF]标志被设置。

- **IP命令--错误情况**

  请参考下面的表34-19。

#### 34.8.2.2 AHB命令

关于本段没有明确涉及的其他细节，请参考第1节"将串行闪存数据读入QuadSPI模块"。

- **AHB命令 - 正常操作**

  对不包含在AHB缓冲器中的串行闪存地址的内存映射读访问，触发了AHB命令的执行。考虑到这是一个合法的命令，QSPI_SR[AHBACC]和QSPI_SR[BUSY]位在执行开始后立即被同时引爆。当串行闪存设备上的指令执行完毕后，这些位就不再被引出。

- **IP命令--错误情况**

  请参考下面的表34-19。

#### 34.8.2.3 错误标志的概述

下表概述了QSPI_FR寄存器中不同的错误标志以及与错误有关的其他细节。

## 表34-19.QSPI_FR 错误标志的概述

| 错误类别 | QSPI_FR中的错误标志 | 在串行闪存设备上执行命令TFF行为（仅在IP命令的情况下）。 | 描述 |
|---|---|---|---|
| **AHB错误标志** | ABSEF | 闪存交易被中止 | AHB序列包含<br>• WRITE指令<br>• WRITE_DDR指令 |
| **AHB错误标志** | ABOF | 闪存交易继续进行，直到它完成 | 当模块试图将数据推入AHB缓冲区而超过AHB缓冲区的大小时，设置。只有在对QSPI_BUFxCR[ADATSZ]编程错误时才会发生。 |
| **AHB错误标志** | AIBSEF | 闪存交易被中止 | AHB事务的总突发量大于预取数据量 |
| **AHB错误标志** | AITEF | 闪存交易被中止 | 在非法交易和看门狗定时器过期的情况下，没有从QSPI到AHB总线产生响应。 |
| **杂项错误标志** | 伊利兰 | 闪存交易被中止 | 非法指令错误标志 - 当控制器在任何一个序列中遇到非法指令时设置。 |
| **命令仲裁错误** | IPIEF | 未与该命令一起断言的TFF | IP命令错误 - 当IP访问目前正在进行（IP_ACC设置）和<br>• 写入QSPI_IPCR寄存器的尝试。<br>• 写入QSPI_SFAR寄存器的尝试。<br>• 写入QSPI_RBCT寄存器的尝试。 |
| **命令仲裁错误** | IPAEF | | • AHB命令已经运行，另一个IP命令无法执行。<br>• AHB命令已经在运行，试图写到QSPI_IPCR[SEQID]字段。 |
| **命令仲裁错误** | IPGEF | | • 授予AHB命令对串行闪存的独占访问权，对QSPI_IPCR[SEQID]字段进行写入尝试。 |
| **缓冲区相关错误** | RBOF | TFF在完成时被宣称 | • RX缓冲区超限 |
| **缓冲区相关错误** | TBUF | | • TX缓冲区运行不足 |

注意，只有与缓冲区有关的错误与外部串行闪存上的事务有关。所有其他的错误都不会触发一个实际的事务。

#### 34.8.2.4 IP总线和AHB访问命令的碰撞

有两个与此相关的标志，QSPI_FR[IPAEF]和QSPI_FR[IPIEF]。请参考闪存读取部分的 "将串行闪存数据读入QuadSPI模块
"子章节，了解这些标志的描述和命令仲裁，了解可能的命令冲突的细节。

### 34.8.3 为AHB命令独占访问串行闪存

有可能几个主站需要分别访问连接到QuadSPI模块的串行闪存设备，一个主站通过触发IP命令并读取RX缓冲区（通过RBDRn寄存器），其他主站通过触发AHB命令（通过ARDBn寄存器）。这两组缓冲区（RBDR和ARDB缓冲区）指向同一个物理缓冲区。参考图34-7
为了避免命令冲突导致过度延迟，QuadSPI模块在触发AHB命令的主站和QuadSPI模块之间实现了一个请求-
握手机制，允许这个特定的主站为AHB命令请求对串行闪存设备的独占访问。如果这种独占访问被授予，IP命令的执行就会被阻止。这就解决了命令冲突和AHB接口可能被封锁的过多时间。

如果设备中使用了这种能力，就会有与这种机制相关的额外的状态和标志信息。QSPI_SR[AHBGNT]位反映了模块内部的状态，即上述的独占访问被授予，任何试图触发IP命令的行为都被拒绝，并导致QSPI_FR[IPGEF]标志的断言。
详情请参考相关位和标志的描述。

适当地使用这一机制来设置主站是应用程序的责任，如果使用不正确，根本无法触发IP命令。

可以区分两种不同的情况。

#### 34.8.3.1 通过QSPI_ARDB寄存器读取RX缓冲器

在这种情况下，所有的主站都共享AHB总线用于RX Buffer和AHB Buffer的读取。在这种情况下，触发AHB命令的主控器对AHB接口的访问必须推迟到任何悬而未决的IP命令完成，并且RX缓冲区的读出也已完成。QSPI ARDB缓冲器访问Rx缓冲器，即

返回Rx Buffer的数据，不触及AHB Buffer的数据。这是保守的使用情况，对应于QSPI_RBCT[RXBRD]位的复位值0。

在这种情况下，QSPI_SR[AHBGNT]位不早于任何正在运行的IP命令已经完成（QSPI_SR[IP_ACC]为0），RX缓冲区已经完全读出（QSPI_RBSR[RDBFL]等于0）或没有DMA读出等待（QSPI_SR[RXDMA]等于0，Rx缓冲区通过AHB（QSPI_RBCT[RXBRD]）读出等于1）。

#### 34.8.3.2 通过QSPI_RBDR寄存器读取RX缓冲器

这是首选的使用情况，因为对AHB缓冲区（内存映射的闪存）的访问不会干扰任何IPS对RBDR缓冲区的读取。由于RX缓冲区是通过基于IP总线的寄存器QSPI_RBDR0到QSPI_RBDR31读取的，因此不可能出现由AHB命令触发的悬而未决的AHB总线访问使AHB总线停滞并阻止RX缓冲区的读出。

对于这种情况，建议将QSPI_RBCT[RXBRD]位编程为1。QSPI_SR[AHBGNT]位在任何正在运行的IP命令完成（QSPI_SR[IP_ACC]为0）、RX缓冲区完全读出（QSPI_RBSR[RDBFL]等于0）或没有DMA读出等待（QSPI_SR[RXDMA]等于0）后立即被断言，允许触发AHB命令的主站尽快触发AHB命令，而无需等待RX缓冲区读出完成。

### 34.8.4 命令仲裁

在命令重叠的情况下，假设不使用《AHB命令对串行闪存的独占访问》中描述的优先级机制，下面几段将描述仲裁方案。

- 在一个IP命令的执行过程中，运行中的IP命令不能通过发出另一个IP命令或AHB命令来终止。当主机试图写入QSPI_IPCR寄存器时，QSPI_FR[IPIEF]标志被引爆。当主机触发AHB命令时（详见闪存读取部分的
  "将串行闪存数据读入QuadSPI模块
  "子章节），该命令被停顿，直到当前运行的IP命令结束。
- 在AHB命令的执行过程中，运行中的AHB命令不能通过发出一个IP命令来终止。命令被忽略，QSPI_FR[IPAEF]标志被宣称。关于这些标志的描述，请参考标志寄存器（QuadSPI_FR）。

当另一个AHB命令被触发时，将考虑内存映射访问的地址。如果所请求的地址是当前从串行闪存设备中读出的，则继续运行命令。如果不是这样，当前运行的命令将被终止，并执行与请求地址相关的另一个AHB命令。更多细节请参考闪存读取部分的 "将串行闪存数据读入QuadSPI模块 "子章节。

在命令重合的情况下，IP命令被触发，AHB命令被停止，直到IP命令完成（QSPI_SR[IP_ACC]被解除）。

在命令碰撞的情况下，被忽略的IP命令不会导致QSPI_FR[TFF]标志的断言。

### 34.8.5 闪存设备选择

无论SFM命令(IP或AHB)，访问模式都是通过指定以下SFM命令的32位地址值来选择。

对于IP命令来说，访问模式是通过编程到QSPI_SFAR寄存器的地址来选择的。详情请参考串行闪存地址寄存器（QuadSPI_SFAR）。

对于AHB命令，访问模式由被访问的内存映射地址决定，详情请参考闪存映射的AMBA总线。

### 34.8.6 DMA使用情况

关于DMA模块的完整描述，请参考相关的DMA控制器章节。在本段中，只给出与QuadSPI模块有关的DMA使用的具体细节。

#### 34.8.6.1 正常模式下的DMA使用情况

#### 34.8.6.1.1 带宽考虑

仔细考虑读/写数据过程中涉及的整个链条（串行闪存->AHB总线/IP总线->DMA控制器）的吞吐率对于正确的操作至关重要。这种分析不仅要考虑到串行闪存提供的数据速率，还要考虑到AHB总线的数据速率和DMA控制器从/向RX/TX缓冲器读/写数据的性能。

两个数字必须匹配才能正常工作，也就是说，串行闪存设备提供的数据速率不能超过RX缓冲器的平均读出数据速率。
否则，这种状态持续的时间越长，就会导致RX缓冲器溢出。同样，串行闪存设备所消耗的数据也不能超过TX缓冲区的平均填充率。如果这种情况持续下去，最终会导致欠费。

**AHB总线侧（数据读取）。**

下表给出了一些典型使用情况的例子。

案例1：DMA需要从SRAM中读取4个字节并提供给QuadSPI。它总共花费了4个BUS时钟周期。然后DMA握手又增加了6个BUS时钟周期。总共[6 + 4 * (32/4) = 38]BUS时钟周期。

**表34-20.访问持续时间示例 - 总线时钟侧**

| qspi_tbct[wmrk]。 | 每个DMA环路的字节数 | DMA小环路的总线时钟周期的数量 | 60Mhz总线时钟频率的DMA小环路的时间长度 |
|---|---|---|---|
| 3 | 16 | 6+(16/4)*4 = 22 | ~366ns |
| 7 | 32 | 6+(32/4)*4 = 38 | ~633ns |
| 11 | 48 | 6+(48/4)*4 = 54 | ~900ns |
| 15 | 64 | 6+(64/4)*4 = 70 | ~1166ns |

情况2。DMA需要从SRAM读取32个字节并提供给QuadSPI
DMA握手需要额外的6个BUS周期，从SRAM读取32字节的DMA需要花费(8+3)个CORE时钟周期。DMA写32字节到QSPI需要2 * (32/4) = 16个BUS周期，另外CPU访问QuadSPI需要2个BUS时钟周期。总共6 + (8+3)/2 + 2 * (32/4) +2 = 30个BUS时钟周期。

**表34-21.访问持续时间示例 - 总线时钟侧**

| qspi_tbct[wmrk]。 | 每个DMA循环的字节数 > | DMA小环路的总线时钟周期的数量 | 80Mhz总线时钟频率的DMA小环路的时间长度 |
|---|---|---|---|

| | | | |
|---|---|---|---|
| 3 | 16 | 6 + (4+3) /2 + (16/4)*2 + 2 = 20 | ~333ns |
| 7 | 32 | 6 + (8+3) /2 + (32/4)*2 + 2 = 30 | ~500ns |

*表继续在下一页...*

表34-21.访问持续时间示例 - 总线时钟侧（续）

| qspi_tbct[wmrk]。 | 每个DMA循环的字节数 > | DMA小环路的总线时钟周期的数量 | 80Mhz总线时钟频率的DMA小环路的时间长度 |
|---|---|---|---|
| 11 | 48 | 6 + (4+3)/2*3 + (48/4)*2 + 2 = 44 | ~733ns |
| 15 | 64 | 6 + (8+3) + (64/4)*2 + 2 = 51 | ~810ns |

**注意事项**

表中的数字代表理想情况，实际性能将取决于系统的集成方式。

## 串行闪存设备端（数据读取）。

串行闪存周期的数量可以通过以下方式确定。

- 读取4个字节所需的串行闪存时钟周期数，对应于一个RX缓冲区条目（不考虑指令和地址的设置）。，
  在单独闪存模式下，八进制DDR模式（超闪存/超RAM）需要2个周期，在单独闪存模式下，四进制模式（SDR）指令需要8个周期等。
- 因时钟域交叉而产生的开销：1个周期。

下表列出了对应于QSPI_RBCT[WMRK]字段的不同设置，从串行闪存中读取数据所需的时钟周期数量。

表34-22.访问持续时间示例 - 串行闪存端

| QSPI_RBCT[WMRK]设置 | 每个DMA环路[1]的字节数 | 80MHz SCKFx的Num SCKFx | | 80MHz SCKFx的闪存数据读出时间（~12.5ns周期）。 | |
|---|---|---|---|---|---|
| | | IFM [2] Quad | IFM Quad DDR | IFM Quad | IFM Quad DDR |
| 0 | 4 | 8 | 4 | ~100ns | ~50ns |
| 1 | 8 | 16 | 8 | ~200ns | ~100ns |
| 3 | 16 | 32 | 16 | ~400ns | ~200ns |
| 7 | 32 | 64 | 32 | ~800ns | ~400ns |
| 11 | 48 | 96 | 48 | ~1200ns | ~600ns |

1. DMA循环意味着一个小循环的完成，相当于一个大循环的迭代。
2. 个人闪光模式。

**注意事项**

表中的数字代表理想情况，实际性能将取决于系统的集成

方式。

一个补充的例子是当水印被设置得太高时。在这种情况下，DMA读出RX缓冲区条目的时间应该小于控制器推入缓冲区中剩余条目的时间。

### IPS总线侧（数据写入）。

每个DMA小环路完成的总的总线周期数是由以下部分加起来的。

- 每个小循环的开销，由DMA控制器给出。假设10个周期
- 由于时钟域交叉而产生的开销。假设2个周期
- 16个字节（128位写大小）所需的总线时钟周期数。假设4个周期(DMA控制器的读/写序列)

请注意，小循环的大小是由QSPI_TBCT[WMRK]字段的大小决定的，因此上面给出的开销在每个32位的（QSPI_TBCT[WMRK]+1）写访问中分配。

下表给出了一些典型使用案例的例子。

**表34-23.访问持续时间示例 - 总线时钟侧**

| qspi_tbct[wmrk]。 | 每个DMA环路[1]的字节数 | DMA小环路的总线时钟周期的数量 | 80Mhz总线时钟频率下的DMA小循环的时间长度 |
|---|---|---|---|
| 3 | 16 | 12+4 = 16 | ~200ns |
| 7 | 32 | 12+8 = 20 | ~250ns |
| 11 | 48 | 12+12 = 24 | ~300ns |
| 15 | 64 | 12+16 = 28 | ~350ns |
| 19 | 80 | 12+20 = 32 | ~400ns |

1.DMA环路是指一个小环路完成，相当于1。

**注意事项**

表中的数字代表理想情况，实际性能将取决于系统的集成方式。

### 串行闪存设备端（数据写入）。

串行闪存周期的数量可以通过以下方式确定。

- 写入16个字节所需的串行闪存时钟周期数，对应于四个TX Buffer条目（不考虑指令和地址的设置）。在单独闪存模式下，八进制DDR模式（超闪存/超RAM）指令需要8个周期，在单独闪存模式下，四进制SDR写入需要32个周期。
- 因时钟域交叉而产生的开销：1个周期。

下表列出了对应于QSPI_TBCT[WMRK]字段的不同设置，从串行闪存中读取数据所需的时钟周期数量。

**表34-24.访问持续时间示例 - 串行闪存端**

| qspi_tbct[wmrk]。设置 | 每个DMA环路[1]的字节数 | Num SCKFx | | 在闪存接口消耗数据的时间长度 100MHz SCKFx（10ns周期）[2] | | 是时候让fifo变空了[3] | |
|---|---|---|---|---|---|---|---|
| | | IFM [4] Quad | IFM Octal DDR | IFM Quad | IFM Octal DDR | IFM Quad | IFM Octal DDR |
| 3 | 16 | 32 | 8 | 320ns | 80ns | 2240ns | 560ns |
| 7 | 32 | 64 | 16 | 640ns | 160ns | 1920ns | 480ns |
| 15 | 64 | 128 | 32 | 1280ns | 320ns | 1280ns | 320ns |
| 23 | 96 | 192 | 48 | 1920ns | 480ns | 640ns | 160ns |

1. DMA循环意味着一个小循环的完成，相当于一个大循环的迭代。
2. 并非所有的闪存设备都支持100Mhz的写入频率。请参考闪存数据手册，了解支持的实际页面编程频率。
3. 这些时序的假设是，当交易开始时，TX Fifo已经满了。
4. 个人闪光模式。

**注意事项**

上面提到的表格只是例子，必须与系统中的DMA相关联。

从上面两个表格中给出的TX fifo的例子可以看出，根据总线时钟和串行闪存时钟频率之间的关系，有可能出现串行闪存消耗数据的速度快于IPS总线在TX缓冲区写入数据的设置。在这些情况下，会出现TX缓冲区欠载的情况。为了避免TX缓冲器欠载，数据交易的大小应该足够大。

## 34.9 字节排序 - Endianness

QuadSPI提供了对基于QSPI_MCR[END_CFG]配置的闪存读/写数据交换的支持。默认情况下，通过RX缓冲区读取数据时，在AHB总线上总是以64位格式返回，在IPS接口上以32位格式返回，通过TX缓冲区写入时以32位格式写入。

下表（QSPI_MCR[END_CFG]）显示了完整的位排序。BE表示大端，意味着相关数据向量的高阶位与低阶地址位置有关。LE表示小端，意味着相关数据向量的低阶位与低阶地址位置相关联，参见图34-7。

**表34-25.qspi_mcr[end_cfg]。**

| | |
|---|---|
| 00 | 64位BE |
| 01 | 32位LE |
| 10 | 32位BE |
| 11 | 64位LE |

下表（AHB中的字节排序配置）和（IPS中的字节排序配置）分别显示了这种配置在QSPI AHB和IPS接口中的实现方式。表中的B表示字节，索引[1-8]指的是字节的位置，即1指的是比特[7:0]，8指的是比特[63:56]，以此类推。

**表34-26.AHB中的字节排序配置**

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 64位BE | B1 | B2 | B3 | B4 | B5 | B6 | B7 | B8 |
| 64位LE | B8 | B7 | B6 | B5 | B4 | B3 | B2 | B1 |
| 32位BE | B5 | B6 | B7 | B8 | B1 | B2 | B3 | B4 |
| 32位LE | B4 | B3 | B2 | B1 | B8 | B7 | B6 | B5 |

**表34-27.IPS中的字节排序配置**

| | | | | |
|---|---|---|---|---|
| 32BE | B1 | B2 | B3 | B4 |
| 32LE | B4 | B3 | B2 | B1 |

下面的例子显示了AHB缓冲区的64位BE配置和TX/RX缓冲区的32位BE配置中的字节排序。

### 34.9.1 闪存数据编程

CPU向QSPI_TBDR寄存器写指令，如

- 写入 QSPI_TBDR -> 0x01_02_03_04
- 写QSPI_TBDR ->

0x05_06_07_08，结果是TX缓冲区的内容如

下。

表34-28.QuadSPI TX 缓冲器的例子

| TX缓冲区条目 | 内容 |
|---|---|
| 0 | 32'h01_02_03_04 |
| 1 | 32'h05_06_07_08 |

将TX缓冲器编程到外部串行闪存设备中的结果是将以下字节顺序发送到串行闪存中。

- 01…02…03…04…05…06…07…08

## 34.9.2 将闪存数据读入RX缓冲区

从同一地址读取内容提供了以下的字节序列，与写入情况相同。

- 01…02…03…04…05…06…07…08

这导致RX缓冲区充满了。

表34-29.结果的RX缓冲区内容

| RX缓冲区条目 | 内容 |
|---|---|
| 0 | 32'h01_02_03_04 |
| 1 | 32'h05_06_07_08 |

### 34.9.2.1 通过QSPI_RBDRn读出RX缓冲器

RX缓冲区的内容在CPU通过外设总线接口进行读取时按以下顺序出现。

- 读取QSPI_RBDR0 <- 0x01_02_03_04
- 读取QSPI_RBDR1 <- 0x05_06_07_08

### 34.9.2.2 通过ARDBn读出RX缓冲器

RX缓冲区的内容出现在QuadSPI模块边界的AMBA AHB接口上的读取访问。

- (1a):32位访问。读取QSPI_ARDB0 <- 0x01_02_03_04
- (2a):32位访问。读取QSPI_ARDB1 <- 0x05_06_07_08
- (1b/2b)。64位访问。读取QSPI_ARDB0 <- 0x01_02_03_04_05_06_07_08

### 34.9.3 将闪存数据读入AHB缓冲区

从被写入的相同地址读取内容，提供以下字节序列，与写入情况相同。

- 01…02…03…04…05…06…07…08

这导致AHB缓冲区充满了。

**表34-30.结果AHB缓冲区内容**

| AHB缓冲区条目 | 内容 |
|---|---|
| 0 | 64'h01_02_03_04_05_06_07_08 |

#### 34.9.3.1 通过内存映射读取AHB缓冲区的读出量

AHB缓冲区的内容出现在QuadSPI模块边界的AMBA AHB接口上的读取访问。

- (1a):32位读访问：<- 0x01_02_03_04
- (2a):32位读访问：<- 0x05_06_07_08
- (1/2):64位读访问：<- 0x01_02_03_04_05_06_07_08

## 34.10 在单、双模式下驱动闪光控制信号

在单模式和双模式下，可以连接到QuadSPI模块的串行闪存设备希望在输入端获得额外的控制信号，这些信号在四模式下连接到IOFA[3]、IOFA[2]、IOFB[3]和IOFB[2]。为了便于连接，闪存A的输出IOFA[3:2]和闪存B的输出IOFB[3:2]被驱动到配置位QSPI_MCR[ISD3FA]、QSPI_MCR[ISD2FA]、QSPI_MCR[ISD3FB]和QSPI_MCR[ISD2FB]所给的逻辑状态。

这些输出一直被驱动到QSPI_MCR寄存器中设定的逻辑电平，除了在执行串行闪存的四字节命令时。关于非活动电平的详细情况，请参考相关串行闪存设备的说明书。

## 34.11 串行闪存设备

有几家不同的供应商生产了带有QuadSPI接口的闪存设备。目前，QuadSPI指令集还没有固定的标准。目前，大多数常见的指令对所有供应商都有相同的指令代码，但有些指令对特定的供应商来说是独一无二的。下面提供了一些序列的例子。

### 34.11.1 序列示例

本节提供QuadSPI模块的实例序列。

**表34-31.退出4 x I/O读取增强性能模式（XIP）（Macronix）和读取状态**

| 训练 | PAD | 运作方式 | 评论 |
|---|---|---|---|
| 医学博士 | 0x0 | 0xEB | 4xIO 读取命令 |
| 地址 | 0x2 | 0x18 | 24位地址将在4个垫子上发送 |
| 模式 | 0x2 | 0x00 | 2个模式循环（退出XIP）。 |
| DUMMY | 0x0 | 0x04 | 4个虚拟周期 |
| 阅读 | 0x2 | 0x08 | 读取64位 |
| 医学博士 | 0x0 | 0x05 | 读取状态寄存器 |
| 阅读 | 0x0 | 0x01 | 状态寄存器数据 |
| 停止 | 0x0 | 0x00 | STOP，指令结束 |

#### 34.11.1.1 读取命令（Spansion Hyperflash/HyperRAM）。

本节提供了QuadSPI模块的读取命令序列（Spansion Hyperflash/HyperRAM）。

**表34-32.读取命令（Spansion Hyperflash/HyperRAM）。**

| 训练 | PAD | 运作方式 | 评论 |
|---|---|---|---|
| 呼叫中心：CMD_DDR | 0x3 | 0xA0 | 连续突发型的读命令 |
| ADDR_DDR | 0x3 | 0x18 | 24位行地址 |

在单、双模式下驱动闪光控制信号

*表继续在下一页...*

表34-32.读取命令（Spansion Hyperflash/HyperRAM）（续）

| | | | |
|---|---|---|---|
| 编码：CADDR_DDR | 0x3 | 0x10 | 16位列地址，低3位有效，其余为0 |
| DUMMY | 0x3 | 0x0F | 15个虚拟周期 |
| READ_DDR | 0x3 | 0x4 | 在8个垫子上读取32位数据 |
| 停止 | 0x3 | 0x00 | STOP，指令结束 |

Hyperflash/HyperRAM是一个可字寻址的闪存，即每个地址访问一个字宽（2字节）的数据值，软件应该确保当Hyperflash/HyperRAM连接到控制器时，QSPI_SFACR
[WA]位必须被设置。如果这个位被设置，控制器就会将一个字节的可寻址访问重新映射为一个字的可寻址访问。

### 34.11.1.2 读取状态寄存器（Spansion Hyperflash/HyperRAM）。

本节提供QuadSPI模块的读取状态寄存器。

表34-33.读取状态寄存器（Spansion Hyperflash/HyperRAM）。

| 训练 | 训练序列 | PAD | 运作方式 | 评论 |
|---|---|---|---|---|
| 呼叫中心：CMD_DDR | 请阅读 "预审员"。 | 0x3 | 0x00 | 写入命令，有包裹的突发类型。 |
| ADDR_DDR | | 0x3 | 0x18 | 24位行地址(0000AAh) |
| 编码：CADDR_DDR | | 0x3 | 0x10 | 16位列地址，低3位有效，其余为0(0005h)，作为命令处理。 |
| 呼叫中心：CMD_DDR | | 0x3 | 0x00 | 带有包裹式脉冲类型的写命令 |
| 呼叫中心：CMD_DDR | | 0x3 | 0x70 | 写入数据，作为预命令发送到闪存中 |
| 呼叫中心：CMD_DDR | 命令阶段（第四/最后一个芯片选择阶段）。 | 0x3 | 0xA0 | 连续突发类型的读命令 |
| ADDR_DDR | | 0x3 | 0x18 | 24位行地址 |
| 编码：CADDR_DDR | | 0x3 | 0x10 | 16位列地址，低3位有效，其余为0 |
| DUMMY | | 0x3 | 0x0F | 15个虚拟周期 |
| READ_DDR | | 0x3 | 0x4 | 在8个垫子上读取32位数据 |
| 停止 | | 0x3 | 0x00 | STOP，指令结束 |

### 34.11.1.3 字程序（Spansion Hyperflash/HyperRAM）。

本节提供QuadSPI模块的字程序（Spansion Hyperflash/HyperRAM）。

**表34-34.字程序（Spansion Hyperflash/HyperRAM）。**

| 训练 | 训练序列 | PAD | 运作方式 | 评论 |
|---|---|---|---|---|
| 呼叫中心：CMD_DDR | 解锁顺序1（第一个芯片选择阶段） | 0x3 | 0x00 | 带有包裹式脉冲类型的写命令 |
| 呼叫中心：CMD_DDR | | 0x3 | 0x00 | 8位地址00h被视为命令 |
| 呼叫中心：CMD_DDR | | 0x3 | 0x00 | 8位地址00h被视为命令 |
| 呼叫中心：CMD_DDR | | 0x3 | 0xAA | 8位地址AAh被视为命令 |
| 编码：CADDR_DDR | | 0x3 | 0x10 | 16位列地址，低3位有效，其余为0(0005h)，作为命令处理。 |
| 呼叫中心：CMD_DDR | | 0x3 | 0x00 | 写入命令，有包裹的突发类型。 |
| 呼叫中心：CMD_DDR | | 0x3 | 0xAA | 写入数据，作为预命令发送到闪存。 |
| 呼叫中心：CMD_DDR | 解锁序列2（第二个芯片选择阶段）。 | 0x3 | 0x00 | 带有包裹式脉冲类型的写命令 |
| 呼叫中心：CMD_DDR | | 0x3 | 0x00 | 8位地址00h被视为命令 |
| 呼叫中心：CMD_DDR | | 0x3 | 0x00 | 8位地址00h被视为命令 |
| 呼叫中心：CMD_DDR | | 0x3 | 0x55 | 8位地址55h被视为命令 |
| 编码：CADDR_DDR | | 0x3 | 0x10 | 16位列地址，低3位有效，其余为0(0002h)，作为命令处理。 |
| 呼叫中心：CMD_DDR | | 0x3 | 0x00 | 带有包裹式脉冲类型的写命令 |
| 呼叫中心：CMD_DDR | | 0x3 | 0x55 | 写入数据，作为预命令发送到闪存中 |
| 呼叫中心：CMD_DDR | 程序设置阶段（第三个芯片选择阶段） | 0x3 | 0x00 | 带有包裹式脉冲类型的写命令 |
| 呼叫中心：CMD_DDR | | 0x3 | 0x00 | 8位地址00h被视为命令 |
| 呼叫中心：CMD_DDR | | 0x3 | 0x00 | 8位地址00h被视为命令 |

| | | | |
|---|---|---|---|
| 呼叫中心：CMD_DDR | 0x3 | 0xAA | 8位地址AAh被视为命令 |

表34-32.读取命令（Spansion Hyperflash/HyperRAM）（续）

*表继续在下一页...*

表34-34.字程序（Spansion Hyperflash/HyperRAM）（续）

| | | | | |
|---|---|---|---|---|
| 编码：CADDR_DDR | | 0x3 | 0x10 | 16位列地址，低3位有效，其余为0(0005h)，作为命令处理。 |
| 呼叫中心：CMD_DDR | | 0x3 | 0x00 | 带有包裹式脉冲类型的写命令 |
| 呼叫中心：CMD_DDR | | 0x3 | 0xA0 | 写入数据，作为预命令发送到闪存中 |
| 呼叫中心：CMD_DDR | 命令阶段（第四/最后一个芯片选择阶段）。 | 0x3 | 0x00 | 带有包裹式脉冲类型的写命令 |
| ADDR_DDR | | 0x3 | 0x18 | 24位行地址 |
| 编码：CADDR_DDR | | 0x3 | 0x10 | 16位列地址，低3位有效，其余为0 |
| WRITE_DDR | | 0x3 | 0x2 | 2个字节的数据写在8个垫子上（D1D2）。 |
| 停止 | | 0x3 | 0x00 | STOP，指令结束 |

### 34.11.1.4 快速读取序列（Macronix/Numonyx/Spansion/Winbond）。

下表显示了旺宏/Numonyx/Spansion/Winbond闪光灯的快速读取顺序。

表34-35.快速读取序列

| 指示 | 垫子 | 操作数 | 评论 |
|---|---|---|---|
| 医学博士 | 0x0 | 0x0B | 快速读取命令=0x0B |
| 地址 | 0x0 | 0x18 | 在一个焊盘上发送24个Addr位 |
| DUMMY | 0x0 | 0x08 | 8个虚拟周期 |
| 阅读 | 0x0 | 0x04 | 在一个焊盘上读取32位 |
| 呼叫中心：JMP_ON_CS | 0x0 | 0x00 | 跳到指令0（CMD）。 |

### 34.11.1.5 快速双I/O DT读取序列（旺宏）。

下表显示了Macronix闪存的快速双I/O DT读取顺序。

表34-36.快速双I/O DT读取顺序

| 指示 | 垫子 | 操作数 | 评论 |
|---|---|---|---|
| 医学博士 | 0x0 | 0xBD | 快速双路I/O DT读取命令=0xBD |

*表继续在下一页...*

表34-36.快速双I/O DT读取序列（续）

| 指示 | 垫子 | 操作数 | 评论 |
|---|---|---|---|
| ADDR_DDR | 0x1 | 0x18 | 在DDR模式下，24个Addr位将被发送到2个焊盘上 |
| MODE4_DDR | 0x1 | 0x00 | P2=P0或P3=P1是必要的。详情请参考旺宏的数据手册。模式为一个时钟周期。 |
| DUMMY | 0x0 | 0x06 | 6个虚拟周期 |
| READ_DDR | 0x1 | 0x04 | 在DDR模式下，在2个垫子上读取32个比特 |
| 呼叫中心：JMP_ON_CS | 0x0 | 0x00 | 跳到指令0（CMD）。 |

### 34.11.1.6 快速读取四路输出（温邦德）。

下表显示了Winbond存储器的快读四路输出顺序

表34-37。快速读取四路输出序列

| 指示 | 垫子 | 操作数 | 评论 |
|---|---|---|---|
| 医学博士 | 0x0 | 0x6B | 快速读取四路输出命令 = 0x6B |
| 地址 | 0x0 | 0x18 | 24个Addr位将在1个焊盘上发送 |
| DUMMY | 0x0 | 0x08 | 8个虚拟周期 |
| 阅读 | 0x2 | 0x04 | 在4个垫子上读取32位 |
| 呼叫中心：JMP_ON_CS | 0x0 | 0x00 | 跳到指令0（CMD）。 |

### 34.11.1.7 4 x I/O 读取增强性能模式（XIP）（Macronix）。

下表显示了Macronix闪存的4 x
I/O读取增强性能模式。增强性能模式也被称为XIP模式。

表34-38.快速读取四路输出序列

| 指示 | 垫子 | 操作数 | 评论 |
|---|---|---|---|
| 医学博士 | 0x0 | 0xEB | 4xI/O 读取命令 = 0xEB |
| 地址 | 0x2 | 0x18 | 24个Addr位将在4个焊盘上发送 |
| 模式 | 0x2 | 0xA5 | 2个模式循环 |
| DUMMY | 0x0 | 0x04 | 4个虚拟周期 |
| 阅读 | 0x2 | 0x04 | 在4个垫子上读取32位 |

| | | | |
|---|---|---|---|
| 呼叫中心：JMP_ON_CS | 0x0 | 0x01 | 跳到指令1（ADDR）。 |

当处于XIP模式时，软件应该确保所有连接到控制器的闪光灯都处于XIP模式。作为初始化控制器的一部分，所有的闪光灯可以通过进行假读来启用XIP。

### 34.11.1.8 双指令页程序（Numonyx）。

下表显示了Numonyx闪光灯的双指令页程序顺序。

**表34-39.双指令页程序顺序**

| 指示 | 垫子 | 操作数 | 评论 |
|---|---|---|---|
| 医学博士 | 0x1 | 0x02 | 双指令页程序=0x02在2个垫子上 |
| 地址 | 0x1 | 0x18 | 24个Addr位将在2个焊盘上发送 |
| 写作 | 0x1 | 0x20 | 在2个垫子上写32个字节 |
| 停止 | 0x0 | 0x00 | STOP，指令结束 |

### 34.11.1.9 扇区擦除 (旺宏/Spansion/Numonyx)

下表显示Macronix/Spansion/Numonyx闪存的扇区擦除顺序

**表34-40.扇区擦除顺序**

| 指示 | 垫子 | 操作数 | 评论 |
|---|---|---|---|
| 医学博士 | 0x0 | 0xD8 | 扇区擦除命令=0xD8 |
| 地址 | 0x0 | 0x18 | 24个Addr位将在1个焊盘上发送 |
| 停止 | 0x0 | 0x00 | STOP，指令结束 |

### 34.11.1.10 读取状态寄存器（ Macronix/Spansion/Numonyx/Winbond）。

下表显示了Macronix/Spansion/Numonyx/Winbond闪光灯的读取状态寄存器顺序。

**表34-41.读取状态寄存器的顺序**

| 指示 | 垫子 | 操作数 | 评论 |
|---|---|---|---|
| 医学博士 | 0x0 | 0x05 | 读取状态寄存器命令=0x05 |
| 阅读 | 0x0 | 0x01 | 读取状态寄存器数据 |
| 停止 | 0x0 | 0x00 | STOP，指令结束 |

# 34.12 串行闪存输入数据的采样

## 34.12.1 基本描述

QuadSPI用于从串行闪存设备中读取数据。根据实际执行情况，QuadSPI模块中的内部时钟和外部串行闪存设备之间有一个延迟。请参考下图，了解这一方案的概况。

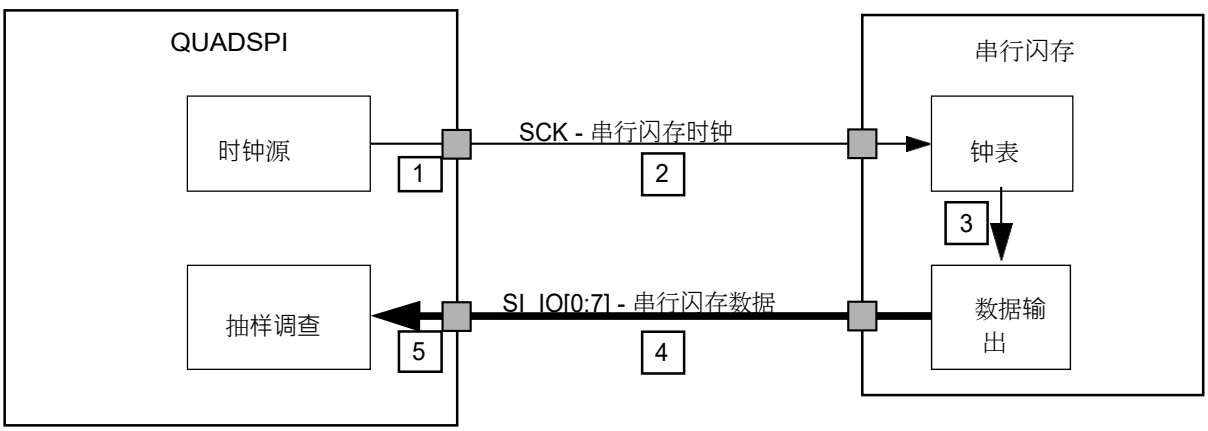



**图34-8.串行闪存取样时钟概述**

内部参考时钟的上升沿被作为串行闪存的数据输出的时间参考。经过$tDel_{,total}$时间后，数据到达QuadSPI模块的内部采样阶段。根据串行闪存采样时钟概述图，延迟链的以下部分对$tDel_{,total}$有贡献。

1. 包含QuadSPI模块的设备的串行闪存时钟输出的延迟
2. 从包含QuadSPI模块的设备到外部串行闪存设备的应用/PCB的导线延迟
3. 外部串行闪存设备的时钟到数据输出延迟，包括输入和输出延迟
4. 应用程序/PCB从外部串行闪存设备到包含QuadSPI模块的设备的电线延迟
5. 与输入数据相对应的设备延迟

**注意事项**

总延迟量$tDel_{,total}$是根据实际实现的特点来确定的。另外，相对于QuadSPI内部参考时钟，串行闪存设备时钟（SCK）是倒置的。

## 34.12.2 支持的读取模式

这里列出的一些模式可能在该芯片上无法使用。关于该芯片支持的读取模式，请参见特定芯片的QuadSPI信息。

### 34.12.2.1 SDR模式

大多数闪存在单数据速率（SDR）模式下工作。在SDR模式下，数据只在时钟信号的一个边缘传输。SDR串行闪存在串行闪存时钟的上升沿采样输入数据，并在串行闪存时钟的下降沿驱动输出数据。

#### 34.12.2.1.1 内部采样

**注意事项**

这种模式在该芯片上可能不适用。关于该芯片支持的读取模式，请参见芯片的具体QuadSPI信息。

QuadSPI在SDR模式下使用内部参考时钟的不同边沿对输入数据进行采样。

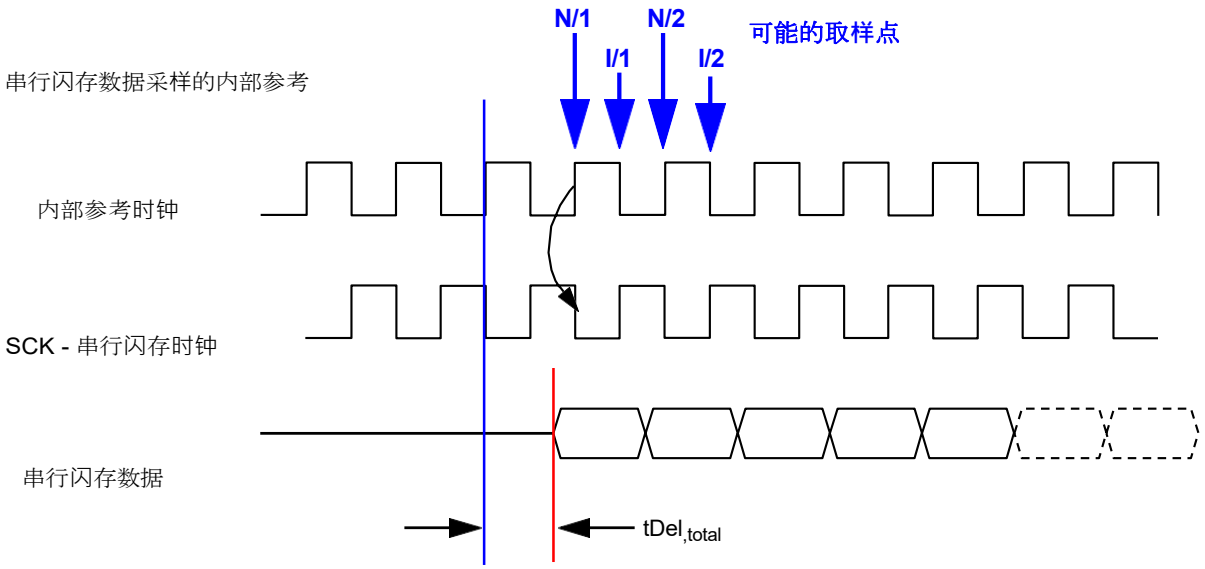



**图34-9.SDR模式下的内部采样**

对传入数据进行采样的可能的时间点在上面被表示为N/1、I/1、N/2和I/2。与内部采样有关的采样点是在QSPI_SMPR寄存器中配置的。详情请参考采样寄存器（QuadSPI_SMPR）。下表概述了以常规（全速）速度运行的命令的可用配置。

**表34-42.采样配置**

| 采样点 | 描述 | 延迟 [FSDLY] | 阶段 [FSPHS] | 全速设置的 QSPI_SMPR1 |
|---|---|---|---|---|
| N/1 | 用非反相时钟采样，1个采样延迟 | 0 | 0 | 0x0000000x |
| I/1 | 带倒置时钟的采样，1个采样延迟 | 0 | 1 | 0x0000002x |
| N/2 | 用非反相时钟采样，2个样本延迟 | 1 | 0 | 0x0000004x |
| I/2 | 带倒置时钟的采样，2个样本延迟 | 1 | 1 | 0x0000006x |

1.这里不考虑'x'。

根据实际的延迟和串行闪存的时钟频率，可以选择适当的采样点。在选择适当的设置时，应考虑以下几点。

- 理论上应该有两种设置可以捕获正确的数据，因为串行闪存输出在1个时钟周期内有效，不考虑上升和下降时间以及时间的不确定性。

- 根据时间上的不确定性，在实际应用中，可能会发现只剩下一个可能的采样位置。这需要根据实际的实施情况仔细考虑。
- 延迟$tDel_{,total}$是一个绝对大小，用于转移串行闪存数据在QuadSPI输入端获得有效的时间点。
- 当串行闪存时钟的频率降低时，边缘之间的距离会增加。因此，对于频率上的巨大差异，所需的设置可能会改变。

#### 34.12.2.1.2 DQS抽样方法

**注意事项**

这种模式在该芯片上可能不适用。关于该芯片支持的读取模式，请参见芯片的具体QuadSPI信息。

可以使用DQS采样方法支持SDR模式下的数据采样。更多细节请参考数据选通（DQS）采样方法。

### 34.12.2.2 DDR模式

由于对提高吞吐量的要求越来越高，因此引入了双数据速率（DDR）模式。在DDR模式下，数据在串行闪存时钟的上升沿和下降沿上传输。DDR串行闪存在串行闪存时钟的上升沿和下降沿上采样和驱动数据。

#### 34.12.2.2.1 DQS抽样方法

**注意事项**

这种模式在该芯片上可能不适用。关于该芯片支持的读取模式，请参见芯片的具体QuadSPI信息。

使用DQS方法可以支持DDR模式下的数据采样。更多细节请参考数据选通（DQS）采样方法。

## 34.12.3 数据选通（DQS）采样方法

### 34.12.3.1 基本描述

在DQS模式下，数据选通信号（DQS/RWDS）被用来对读取的数据进行采样。在这里，DQS和闪存发送的数据都在同一方向移动，所以在较高的频率下相对更容易实现。

当使用DQS进行SDR读取时，QuadSPI内部在选通信号的上升沿对输入的数据进行采样。

下图显示了在SDR模式下使用DQS采样读取数据的情况。

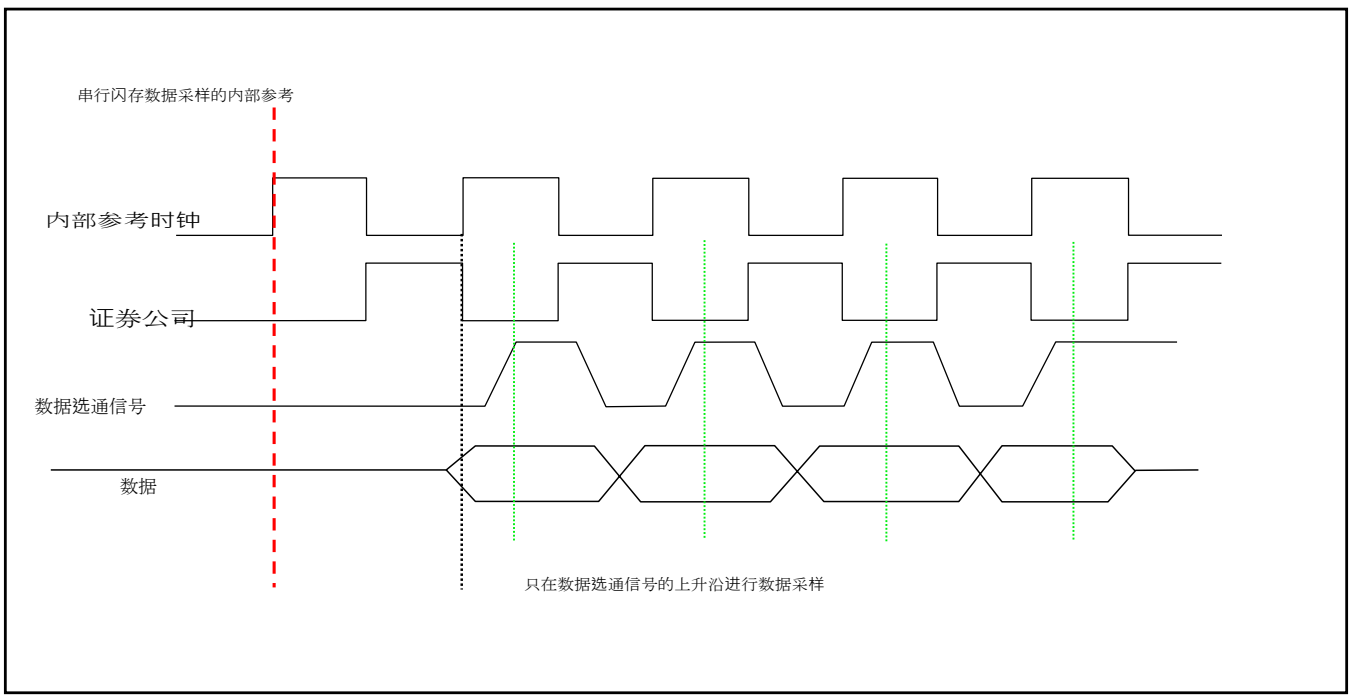



**图34-10.SDR模式下的数据选通功能**

当使用DQS进行DDR读取时，QuadSPI在内部对选通信号的两个边沿的传入数据进行采样。更多细节请参考下图。

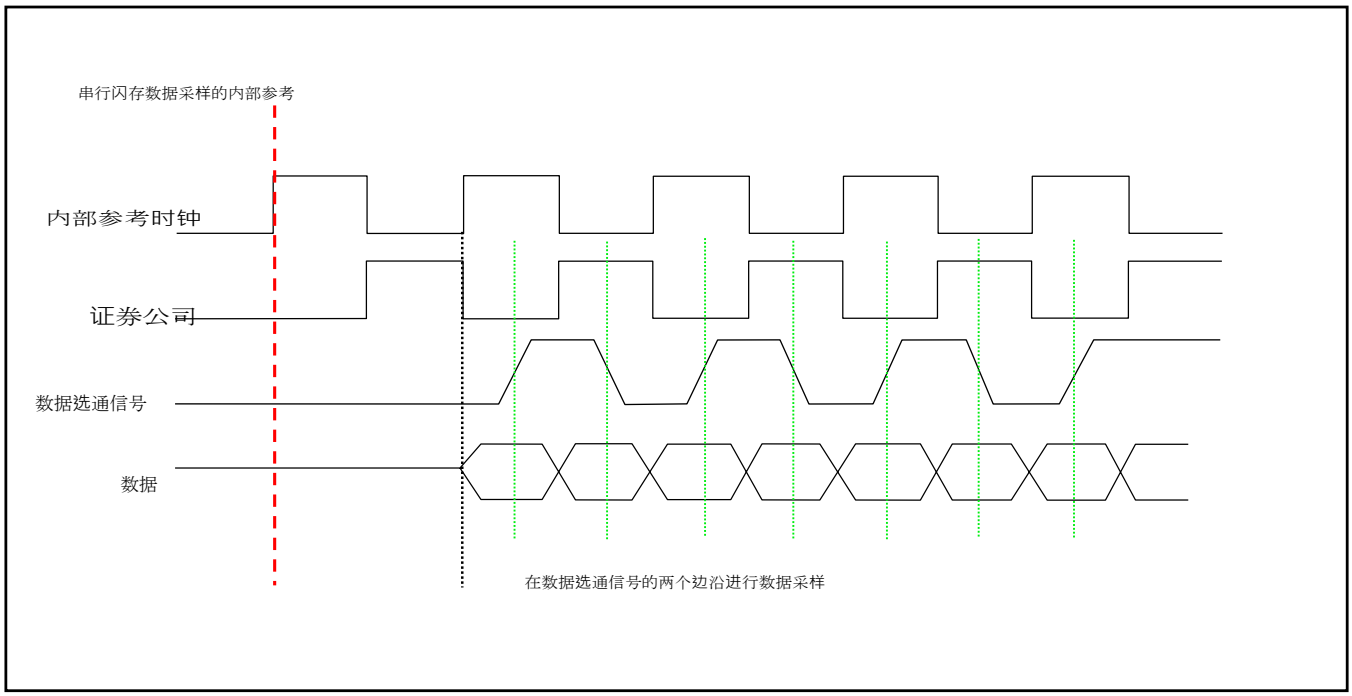



**图34-11.DDR模式下的数据选通功能**

### 34.12.3.2 内部产生的DQS

**注意事项**

这种模式在该芯片上可能不适用。关于该芯片支持的读取模式，请参见芯片的具体QuadSPI信息。

在这种模式下，内部参考时钟作为选通器被送入QuadSPI，用于读取数据采样。数据选通信号的产生方式必须使数据被QuadSPI模块正确采样。这个内部产生的数据选通信号可以被QuadSPI用来捕获数据，在。

- SDR模式
- DDR模式

关于内部产生的DQS的其他细节，请参考QuadSPI芯片的具体信息。

### 34.12.3.3 外部DQS

**注意事项**

这种模式在该芯片上可能不适用。关于该芯片支持的读取模式，请参见芯片的具体QuadSPI信息。

在支持DQS的串行闪存中，数据选通信号是闪存设备的一个输出，指示数据何时从闪存传输到主机控制器。然后，该数据被控制器捕获。

- 在SDR模式下，DQS信号只有一个边沿（无论是上升还是下降边沿）。
- 在DDR模式下DQS信号的两个上升/下降沿

一些串行闪存（例如，HyperFlash）提供了数据选通输出（RWDS），在正在进行的读取交易之间包含了延迟周期。考虑到当闪存在正在进行的交易之间没有提供信号时，数据不会被采样，而在交易结束时，需要获取的数据数量保持不变，所以数据会在这个信号的两个边沿上被取样。更多细节请参考下图。

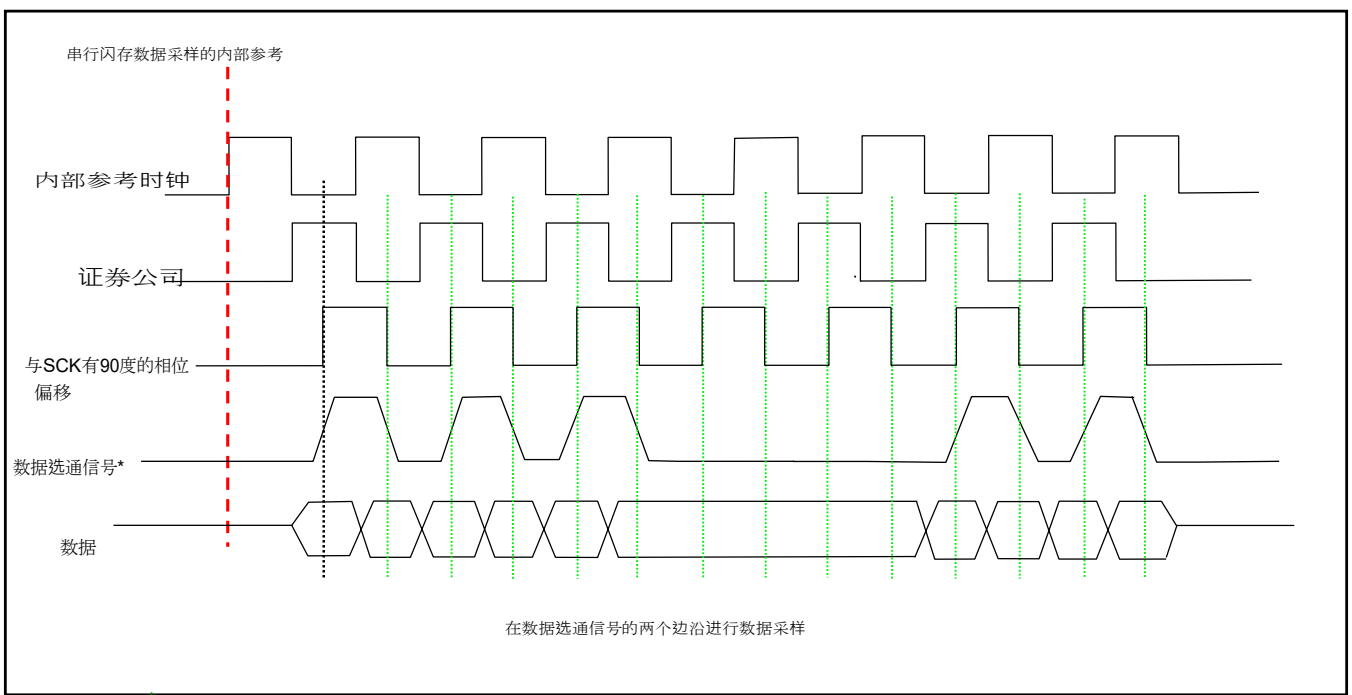



**图34-12.包括延时的数据选通功能**

## 34.13 闪存的数据输入保持要求

在DDR模式下，数据只在半个时钟周期内有效。这就很难满足闪存的数据输入保持时间要求。QuadSPI在内部延迟了发送到闪存的数据，这样就很容易满足保持时间的要求。QSPI_FLSHCR[TDH]就是用于此目的。如果QSPI_FLSHCR[TDH]配置为0，则发送到闪存的数据将与QuadSPI的内部参考时钟对齐，如果配置为1，则数据将与2倍的内部参考半时钟对齐。

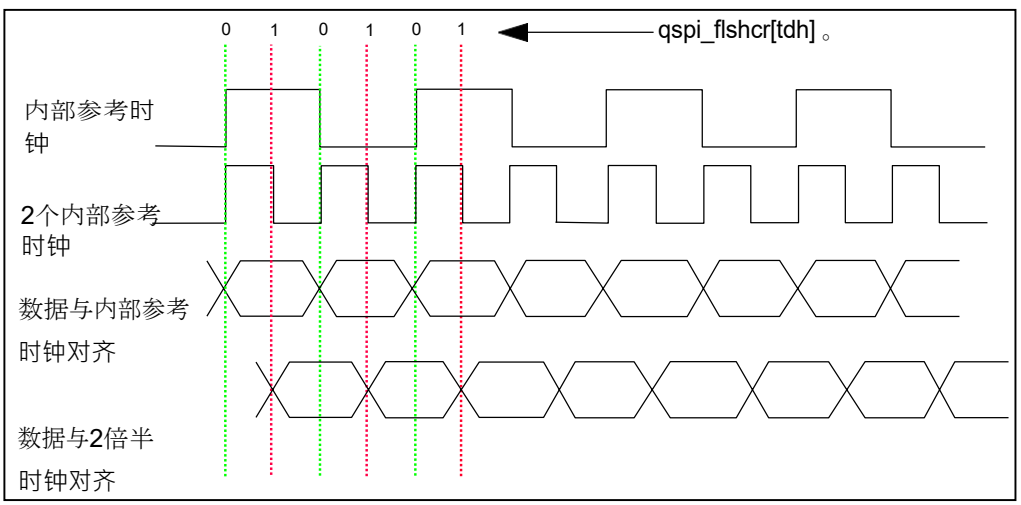



**图34-13.数据保持**

# 第35章
# 电源管理

## 35.1 简介

本章介绍了各种芯片电源模式以及各个模块在这些模式下的功能。

## 35.2 电源模式描述

电源管理控制器（PMC）提供多种电源选项，允许用户根据所需的功能水平优化功耗。

根据用户应用的停止要求，有多种停止模式可供选择，提供状态保留、部分断电或某些逻辑和/或存储器的完全断电。在所有的电源模式下，I/O状态都被保持。下表对各种可用的电源模式进行了比较。

对于运行和极低功率运行（VLPR）模式，有相应的停止模式。停止模式（VLPS、STOP1、STOP2）类似于ARM睡眠深度模式。当不需要最大的总线频率来支持应用需求时，VLPR运行模式可以减少运行时的功率。

该芯片不能从高速运行（HSRUN）模式直接进入停止模式。要从HSRUN进入停止模式，芯片必须首先过渡到正常运行模式。
之后，它可以进入停止模式。

两种主要的操作模式是运行、停止。等待中断（WFI）指令可以调用芯片的停止模式。可用的功率模式允许应用程序只消耗执行所需的功率。

## 注意事项

关于WFI/WFE指令的更多细节，请参见ARM文档。

**表35-1.芯片功率模式**

| 芯片模式 | 描述 | 核心模式 | 正常恢复方法 |
|---|---|---|---|
| 正常运行 | 复位后的默认模式；片上电压调节器处于开启状态。 | 运行 | - |
| 高速运行(HSRUN)[1] | 允许芯片的最大性能。在这种模式下，与正常运行模式相比，芯片可以以更高的频率运行，但功能受到限制。更多细节请参见模块在可用的低功耗模式下的操作。更多细节请参见内部时钟要求。 | 运行 | - |
| 正常停止（通过WFI指令） | 将芯片置于静态状态。在这种电源模式下，所有的寄存器都被保留，LVD保护也被维持。<br>• NVIC被禁用。<br>• AWIC用于从中断中唤醒。<br>• 一些外设的时钟被停止。<br>更多细节请参见模块在可用的低功率模式下的操作。 | 深度睡眠 | 中断 |
| 极低功率运行（VLPR） | 片上电压调节器处于低功率模式，只提供足够的功率让芯片在降低的频率下运行。<br>• 降低频率的闪存访问模式(1MHz)<br>• 关闭LVD<br>• SIRC为核心、总线和外设时钟提供了一个低功耗的4MHz源 | 运行 | - |
| 极低功率停止（VLPS，通过WFI指令）。 | 将芯片置于静态状态，关闭低电压检测（LVD）操作。这是一种最低功率的模式，在这种模式下，引脚中断可以发挥作用。<br>• 一些外设时钟被停止。参见模块在可用的低功率模式下的操作。<br>• 可以使用LPTMR和CMP。<br>• NVIC被禁用。<br>• AWIC用于从中断中唤醒。<br>• 片上电压调节器处于低功率模式，只提供在降低频率下运行芯片所需的功率。<br>• 所有的SRAM都在工作（内容被保留，I/O状态被保持）。 | 深度睡眠 | 中断 |

1.在S32K11x系列设备中，HSRUN模式不可用。

## 注意事项

在禁用一个模块之前，其中断和DMA请求应被禁用。

## 35.3 进入和退出电源模式

WFI指令调用了芯片的停止模式。处理器通过一个中断退出低功耗模式。关于中断操作的描述以及哪些外设可以引起中断，请参见嵌套的矢量中断控制器（NVIC）配置。

**注意事项**

等待事件（WFE）指令可能有进入低功耗模式的副作用，但这不是该指令的目的。

## 35.4 时钟模式

下面的章节描述了该芯片支持的时钟模式。

### 35.4.1 时钟门控

为了节省电力，大多数模块的时钟可以通过配置PCC模块中外围控制寄存器的CGC字段来关闭。这些字段在任何复位后被清空，从而禁用相应模块的协议时钟。在初始化一个模块之前，需要设置PCC外围控制寄存器中的相应字段，以启用模块时钟。请确保在关闭时钟之前禁用模块（详见时钟分配和PCC）。

### 35.4.2 停止模式选项

该芯片有两种停止模式。

- STOP1
- 暂停2

更多细节请参见停止控制寄存器（SMC_ STOPCTRL）中的STOPO字段。当配置为STOP1时，核心时钟、系统时钟和总线时钟都被门控。

当配置为STOP2时，只有核心时钟和系统时钟被门控，但总线时钟保持活动。由

系统时钟驱动的总线主站和总线从站进入停止模式，但由总线时钟驱动的总线从站仍处于运行模式。SCG的时钟发生器和PMC的片上调节器也保持在运行模式。以下情况可以启动从STOP退出。

- 复位
- 来自总线主站的异步中断（对STOP1有效）。
- 来自总线从机的同步中断，由总线时钟驱动（对STOP2有效）。

如果配置好，一个DMA请求（使用异步DMA唤醒）也可以用来在芯片过渡到STOP2之前的DMA传输期间退出停止。

### 35.4.3 DMA唤醒

DMA可以被配置为在设备处于停止模式后，通过DMA请求唤醒设备。唤醒是按DMA通道配置的，在下列低功耗模式下支持。

- 计算操作
- 停止
- VLPS

当在停止或VLPS中检测到DMA唤醒时，芯片会启动一个正常的退出低功率模式。这可以包括

- 恢复内部调节器和内部电源开关
- 启用SCG中的时钟发生器
- 启用系统、总线时钟和闪存时钟（但不是核心时钟）。
- 向总线主站和总线从站否定停止模式信号

唯一的区别是，CPU将保持在低功耗模式下，CPU时钟被禁用。

在计算操作期间，DMA唤醒将启动一个正常的退出。这包括启用时钟和否定对总线主站和总线从站的停止模式信号。在计算操作期间，核心时钟始终保持启用状态。

由于DMA唤醒将启用时钟并否定所有总线主站和从站的停止模式信号，软件必须确保不参与DMA唤醒和传输的总线主站和从站保持在已知状态。这可以通过在进入低功耗模式前禁用模块或在选定的模块中设置休眠使能位来实现。

在启动唤醒的DMA请求被否定后，DMA完成了当前的传输，芯片又过渡到原来的低功耗模式。这包括请求所有非CPU总线主站进入停止模式，然后请求总线从站进入停止模式。在Stop和VLPS模式下，SCG和PMC也会随之进入它们相应的模式。

**注意事项**

如果请求的DMA传输不能导致DMA请求被否定，设备将保持在较高的功率状态，直到完全退出低功率模式。

如果DMA请求在进入停止模式的序列中断言，启用的DMA唤醒会导致进入低功耗模式的中止（如果请求在DMA唤醒期间断言，则重新进入），并且在VLPS模式进入发生时，会导致SMC断言其停止中止标志。在DMA唤醒完成后，开始进入低功率模式。

在DMA唤醒过程中发生的中断会导致立即退出低功率模式（这对计算操作来说是可选的），而不影响DMA传输。

DMA唤醒可以由同步或异步DMA请求产生。不是所有的外设都能在停止模式下产生异步DMA请求。一般来说，如果一个外设可以产生同步DMA请求，并且也支持停止模式下的异步中断，那么它就可以产生一个异步DMA请求。

**注意事项**

DMA Enable Asynchronous Request in Stop Register（DMA_EARS）用于配置每个通道的异步DMA请求。

## 35.4.4 计算操作（CPO）

计算操作（CPO）是一种执行或仅计算的操作模式，它使CPU保持启用状态，可以完全访问SRAM和闪存读取端口，但将所有其他总线主站和总线从站置于其停止模式。CPO可以在运行、HSRUN或VLPR模式下启用。

**注意事项**

在未退出CPO前，不要进入任何停止模式。

因为CPO重用了停止模式的逻辑（包括总线主站在总线从站之前被禁用的分阶段进入），任何能在停止模式下保持功能的总线主站或总线从站在CPO中也保持功能。这包括异步中断和DMA请求的产生。当在运行模式下启用CPO时，总线主站和从站的模块功能等同于停止模式。当在VLPR模式下启用CPO时，总线主站和从站的模块功能等同于VLPS模式。CPO不影响SCG、PMC、SRAM和闪存读取端口，尽管闪存的寄存器接口被禁用。

在CPO期间，AIPS外设空间被禁用，尝试访问会产生总线错误。在CPO期间，私有外设总线（PPB）--包括MCM、系统控制空间（SCS）（用于NVIC）和SysTick--仍然可以访问。尽管支持对GPIO寄存器的访问，但由于端口控制和中断模块的时钟被禁用，GPIO端口的数据输入寄存器不会返回有效数据。通过写入GPIO端口数据输出寄存器，可以控制那些被配置为输出引脚的GPIO端口。

CPO由MCM的CPO寄存器控制，它只能由CPU访问。设置或清除MCM的CPOREQ字段可以启动进入或退出CPO。CPO也可以被配置为在检测到中断时自动退出，这对于服务大多数中断是必需的。只有核心系统中断（NMI和SysTick是例外）和任何边缘敏感中断可以在不退出CPO的情况下得到服务。

当进入CPO时，CPOACK状态字段表明输入已完成。当在运行模式下退出CPO时，CPOACK立即被否定。当在VLPR模式下退出CPO时，退出是延迟的。这允许PMC管理功耗的变化。这种延迟意味着CPOACK被轮询以确定何时可以访问AIPS外围空间而不产生总线错误。

在CPO期间也支持DMA唤醒，并使CPOACK状态清零，在DMA唤醒期间AIPS外设空间可以被访问。在DMA唤醒完成后，芯片又过渡到CPO。

### 35.4.5 周边打盹

一些外设支持外设休眠模式。在这种模式下，一个寄存器字段可以被用来在低功耗模式的持续时间内禁用外设。

外围休眠被定义为包括以下所有的操作模式。

- CPU处于停止模式，包括进入序列和DMA唤醒的时间。
- CPU处于计算操作状态，包括进入序列和DMA唤醒的时间。

它还可以用来在进入任何停止模式（或CPO）时立即禁用选定的总线从机，而不是等待总线主站确认作为停止进入序列一部分的进入。最后，它可以用来禁用选定的总线主站或从站，它们在DMA唤醒期间应保持不活动。

## 35.5 电源模式转换

下图显示了电源模式的转换。任何复位都会使芯片回到正常运行状态。在运行、停止模式下，主动电源调节被启用。VLPR模式提供了一个比正常模式更低的功率工作模式。VLPR在频率上是有限的。

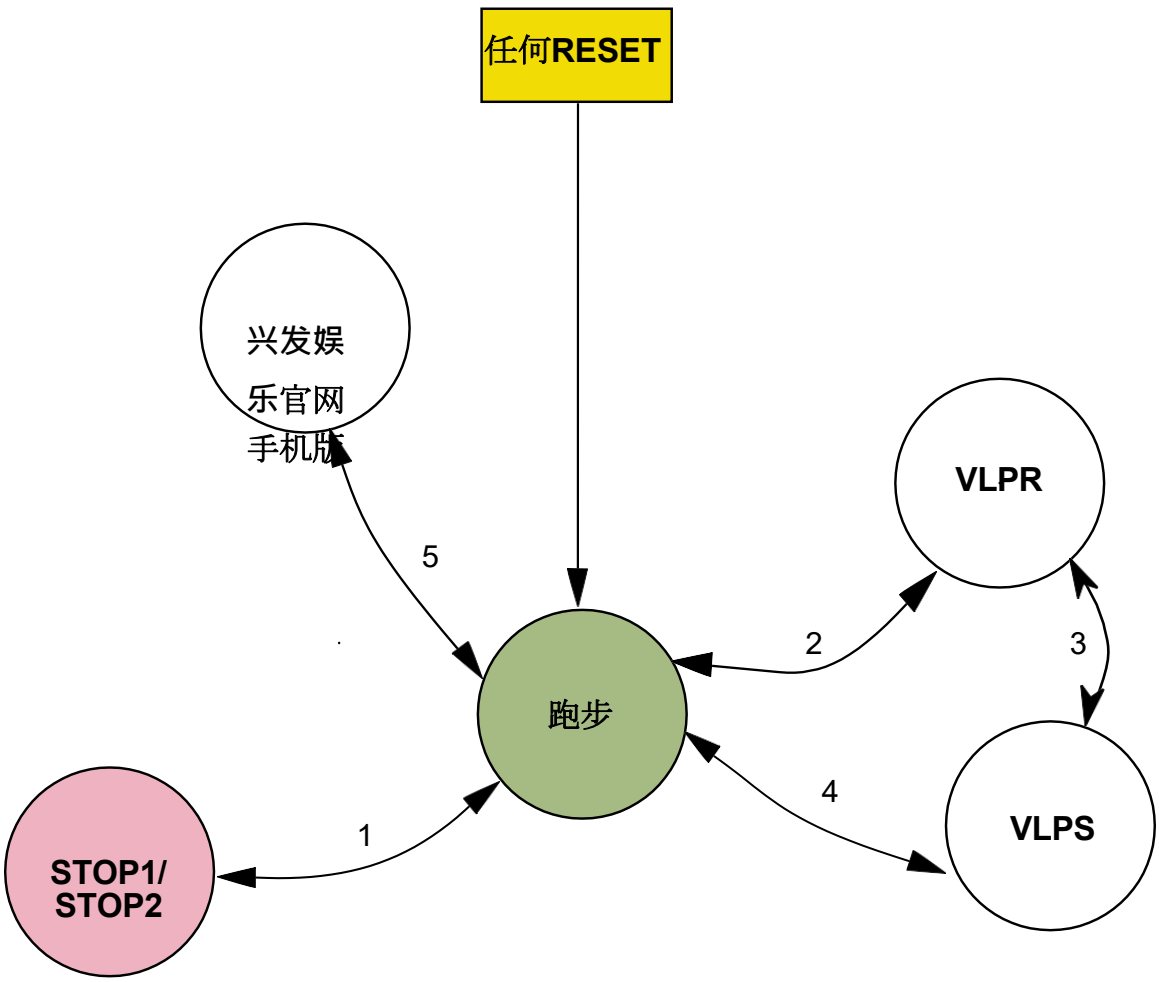



**图35-1.电源模式状态转换图**

### 注意事项

更详细的模式转换条件见SMC章节的图36-1。

## 35.6 电源模式的关机排序

当进入停止或其他低功耗模式时，时钟被有序地关闭，以安全地将芯片置于目标低功耗状态。所有的低功率进入序列都是由内核执行WFI指令启动的。ARM内核的输出，SLEEPDEEP和SLEEPING，触发进入各种低功率模式。

- 低功率模式等同于。睡眠和深度睡眠

当进入低功耗模式时，芯片执行以下顺序。

**表35-2.低功率进入顺序**

| 步骤 | 行动 |
|---|---|
| 1 | 该芯片立即关闭了ARM Cortex-M4内核的核心和系统时钟。 |
| 2 | 芯片轮询来自非核心交叉条主控器（DMA）、支持外设（SPI、PIT）和闪存控制器的停止-确认指示，以了解系统时钟、总线时钟和/或闪存时钟需要保持启用状态以完成先前启动的操作，有效地阻止进入目标低功耗模式。当检测到所有的确认时，系统时钟、总线时钟和闪存时钟将同时关闭。 |
| 3 | SCG和模式控制器关闭时钟源和/或由内部稳压器驱动的内部电源，这是针对目标低功耗模式的定义。 |

调试器模块支持从停止、VLPS到停止状态的转换，当调试器被启用时。这种转换是通过设置MDM-
AP控制寄存器中的调试请求字段启动的。作为这种转换的一部分，系统时钟被重新建立，相当于正常运行/VLPR模式的时钟配置。

## 35.7 闪存编程的电源模式限制

在芯片工作时，不应该对闪存进行编程或擦除。

- VLPR模式
- HSRUN模式

当芯片处于这些模式时，任何类型的FTFC命令，包括CSE命令（用于CSEc部件），都不可用。

## 35.8 模块在可用的低功率模式下运行

表35-4说明了每个可用模式中的模块功能。

*表继续在下一页...*

调试模块将单独讨论，见低功耗模式下的调试。

### 注意事项

S32K1xx器件不支持WAIT、VLPW和低泄漏模式（电源门控模式）。

表35-3定义了一些经常使用的术语和表35-4中的缩写。

**表35-3.术语和缩略语**

| 术语 | 定义 |
|---|---|
| 异步操作 | 完全可以使用备用时钟源（如果所选的时钟源仍然启用）。 |
| FF | 全功能。在VLPR中，系统频率受到限制，但如果一个模块没有功能限制，它仍然被列为FF。 |
| 低功率 | 存储器通电后，在低功率状态下保留内容。 |
| 关闭 | 模块和时钟源被禁用。 |
| 权力机构 | 内存通电后可以保留内容。 |
| 唤醒 | 模块可以作为芯片的唤醒源。 |

**表35-4.模块在可用的低功率模式下的操作**

| 模块 | 辽宁省 | 停止(停止1/停止2) | 谷神星（VLPS1 | 跑步 | 兴发娱乐官网手机版 |
|---|---|---|---|---|---|
| 核心模块 | | | | | |
| NVIC | FF | 关闭 | 关闭 | FF | FF |
| 系统模块 | | | | | |
| 系统模式控制器/电源管理控制器 | FF | FF | FF | FF | FF |
| 调节器 | 低功率模式（LPM） | 全性能模式（FPM） | 低功率模式（LPM） | 全性能模式（FPM） | 全性能模式（FPM） |
| LVD/LVR | LVD 禁用/LVR 激活 | 都是活跃的 | LVD禁用/LVR激活 | 都是活跃的 | 都是活跃的 |
| 断电检测 | FF | FF | FF | FF | FF |
| DMA | FF | 异步操作 | 异步操作 | FF | FF |
| 看门狗 | FF (只有SIRC和LPO时钟为来源) | 异步操作（STOP位需要被设置） STOP1，FF在STOP2中 | 异步操作（STOP位需要被设置）。异步唤醒 | FF | FF |

| | | | | | |
|---|---|---|---|---|---|
| EWM | FF（只有SIRC和LPO时钟为源）。 | 静态在STOP1，FF在STOP2 | 关闭 | FF | FF |
| 钟表 | | | | | |
| 128 kHz LPO | FF | FF | FF | FF | FF |

*表继续在下一页...*

表35-4.模块在可用的低功率模式下的操作（续）

| 模块 | 辽宁省 | 停止(停止1/停止2) | 谷神星（VLPS1 | 跑步 | 兴发娱乐官网手机版 |
|---|---|---|---|---|---|
| PLL | 关闭 | FF | 关闭 | FF | FF |
| SIRC | FF | FF | FF | FF | FF |
| FIRC | 关闭 | FF | 关闭 | FF | FF |
| 系统振荡器（OSC） | 关闭 | FF | 关闭 | FF | FF |
| 渣打集团 | 只有SIRC可用 | 所有的时钟源都可以使用 | 只有SIRC可用 | 所有的时钟源都可以使用 | 所有的时钟源都可以使用 |
| 核心时钟 | 最大4兆赫（SIRC作为一个源）。 | 关闭 | 关闭 | 最大80兆赫（PLL作为一个源）。 | 最大112兆赫（PLL作为一个源）。 |
| 闪存时钟 | 最大1兆赫（SIRC作为一个源）。 | 在STOP1中关闭，在STOP2中可用 | 关闭 | 最大26.67 MHz (PLL为源) | 最大28兆赫（PLL作为一个源）。 |
| 系统时钟 | 最大4兆赫（SIRC作为一个源）。 | 关闭 | 关闭 | 最大80兆赫（PLL作为一个源）。 | 最大112兆赫（PLL作为一个源）。 |
| 总线时钟 | 最大4兆赫（SIRC为源）。 | 在STOP1中关闭，在STOP2中可用 | 关闭 | 最大48兆赫（FIRC作为一个源）/最大40兆赫（PLL作为一个源） | 最大56兆赫（PLL作为一个源）。 |
| 内存和内存接口 | | | | | |
| 闪存 | 只能进行读操作（最大1MHz）。无FTFC任何类型的命令，包括CSE命令（用于CSEc部分），都可以使用。 | 低功耗（无读取和编程/擦除）。 | 低功耗（无读取和编程/擦除）。 | FF | 只能进行读操作。没有FTFC任何类型的命令，包括CSE命令（用于CSEc部分），都可以使用。 |
| 系统RAM（SRAM_U和SRAM_L） | 低功耗（读和写操作）。 | 低功耗（无读写）。 | 低功耗（无读写）。 | FF | FF |
| 缓存 | 低功耗（读和写操作）。 | 低功耗（无读写）。 | 低功耗（无读写）。 | FF | FF |
| 作为EEPROM仿真的FlexMemory | 低功耗（读和写操作）。只能进行读操作。没有FTFC命令。 | 低功耗（无读写）。 | 低功耗（无读写）。 | FF | 只能进行读操作。没有FTFC指挥部。 |

| | | | | | |
|---|---|---|---|---|---|
| FlexMemory（FlexRAM作为传统的SRAM）。 | 低功耗（读和写操作） | 低功耗（无读写）。 | 低功耗（无读写）。 | FF | FF |
| 通信接口 | | | | | |

*表继续在下一页...*

*表继续在下一页...*

表35-4.模块在可用的低功率模式下的操作（续）

表33-4.模块在可用的低功率模式下的操作（续）

| 模块 | 辽宁省 | 停止(停止1/停止2) | 谷神星（VLPS1 | 跑步 | 兴发娱乐官网手机版 |
|---|---|---|---|---|---|
| 朴槿惠 | FF（仅以SIRC为源）。 | STOP1中的异步操作，STOP2中的FF | 关闭 | FF | FF |
| LPSPI | FF（仅以SIRC为源）。 | STOP1中的异步操作，STOP2中的FF | 关闭 | FF | FF |
| LPI2C | FF（仅以SIRC为源）。 | STOP1中的异步操作，STOP2中的FF | 关闭 | FF | FF |
| 弹性IO | FF（仅以SIRC为源）。 | STOP1中的异步操作，STOP2中的FF | 关闭 | FF | FF |
| CAN | FF（仅以SIRC为源）。 | PNET在CAN0的STOP1和STOP2中都支持。 | 关闭 | FF | FF |
| 安全问题 | | | | | |
| 儿童权利委员会 | FF | 在STOP1中关闭 在STOP2中可用 | 关闭 | FF | FF |
| 计时器 | | | | | |
| 基金会 | FF（仅以SIRC为来源） | 关闭 | 关闭 | FF | FF |
| LPIT | FF（仅以SIRC为来源） | STOP1中的异步操作，STOP2中的FF | 关闭 | FF | FF |
| PDB | FF（仅以SIRC为源）。 | 关闭 | 关闭 | FF | FF |
| LPTMR | FF（仅以SIRC为来源） | STOP1中的异步操作，STOP2中的FF | 异步操作 | FF | FF |
| RTC | FF（仅以SIRC为来源） | STOP1中的异步操作，STOP2中的FF | 关闭 | FF | FF |
| 模拟 | | | | | |

| | | | | | |
|---|---|---|---|---|---|
| 12位ADC | 关闭 | 在STOP1中关闭，在STOP2中FF | 关闭 | FF | FF |
| CMP2 | 仅比较LS | HS或LS比较<br>FF在STOP2 | 仅比较LS | FF | FF |

*表继续在下一页...*

表35-4.模块在可用的低功率模式下的操作（续）

| 模块 | 辽宁省 | 停止(停止1/停止2) | 谷神星（VLPS1 | 跑步 | 兴发娱乐官网手机版 |
|---|---|---|---|---|---|
| 人机界面 | | | | | |
| GPIO/端口 | FF | STOP1中的异步操作，STOP2中的FF | 关闭[3] | FF | FF |

1. 当使用VLP*模式时，PMC_REGSC[BIASEN]必须被设置为1。
2. 在停止或VLPS模式下，CMP支持高速或低速外部引脚对引脚或外部引脚对DAC的比较。在停止或VLPS模式下，窗口、采样和过滤的操作模式不可用。
3. 器件可以支持通过GPIO从VLPS进行ASYNC唤醒。详见表AWIC停止和VLPS唤醒来源

## 35.9 QuadSPI、以太网和SAI操作

在QuadSPI、以太网和SAI的运行应用中，应用应确保电压在3.3 V ±10%的范围内。如果外部电源电压不在3.3V±10%的范围内，则可能导致错误的通信。如果S32K148被用于不使用QuadSPI、以太网和SAI的应用，那么片上电压监控器将确保按照S32K1xx数据表中的规格操作（见表LVR、LVD和POR操作要求）。

# 第三十六章
# 系统模式控制器（SMC）

## 36.1 简介

系统模式控制器（SMC）负责对系统进入和退出所有低功率的停止和运行模式进行排序。

具体来说，它监测事件以触发电源模式之间的转换，同时控制系统的电源、时钟和存储器以实现该模式的功耗和功能。

本章介绍了所有可用的低功耗模式，进入/退出每种模式的顺序，以及在每种模式下的可用功能。

SMC甚至能够在最深的低功率模式下运作。

参见AN4503。关于使用SMC的进一步细节，请参见AN4503：Kinetis MCU的电源管理。

## 36.2 操作模式

ARM CPU有三种主要的操作模式。

- 运行
- 睡眠
- 深度睡眠

WFI或WFE指令用于调用睡眠和深睡眠模式。运行和停止是用于主要工作模式的通用术语。下表显示了ARM CPU模式和MCU电源模式之间的转换。RUN被映射到RUN/VLPR，深度睡眠被映射到STOP/VLPS。

| ARM CPU模式 | 单片机模式 |
|---|---|

| | |
|---|---|
| 深度睡眠 | 停止/VLPS |



表35-4.模块在可用的低功率模式下的操作（续）

此外，MCU还以多种方式增强了停止和运行模式。电源管理控制器（PMC）包含一个运行和一个停止模式调节器。运行调节用于正常运行和停止模式。停止模式调节用于所有极低功率和低泄漏模式。在停止模式调节期间，总线频率在非常低的功率模式下被限制。

SMC为用户提供了多种电源选择。当不需要最大的总线频率来处理应用需求时，极低功率运行（VLPR）模式可以大大降低运行时间的功率。从正常运行模式，运行模式（RUNM）字段可以被修改，以便在有限的频率足以满足应用的情况下将MCU变为VLPR模式。从VLPR模式，可以进入相应的停止（VLPS）模式。

根据用户应用的需要，有多种停止模式可供选择，允许某些逻辑和/或存储器的状态保持、部分断电或完全断电。在所有的操作模式下，I/O状态都被保留。几个寄存器用于配置器件的各种操作模式。

下表描述了设备可用的电源模式。

**表36-1.电源模式**

| 模式 | 描述 |
|---|---|
| 跑步 | 单片机可以全速运行，内部电源是完全调节的，也就是处于运行调节状态。这种模式也被称为正常运行模式。 |
| 兴发娱乐官网手机版 | 与RUN模式相比，MCU可以在更快的频率下运行，而且内部电源是完全调节的。有关最大允许频率的细节，请参见电源管理一章。 |
| 停止 | 核心时钟被门控关闭。有两种不同的停止模式 - STOP1和STOP2。在STOP1中，系统时钟和总线时钟都被门控。在STOP2中，总线时钟继续运行，而系统时钟被门控。 |
| VLPR | 在这种模式下，核心、系统、总线和闪存时钟的最高频率受到限制。有关最大允许频率的细节，请参见电源管理一章。 |
| VLPS | 核心时钟被关闭。在所有来自支持外设的停止确认信号有效后，其他主站的系统时钟和总线时钟被关断。 |

**注意事项**

如果系统时钟在模式转换过程中被改变，建议让所有外设的通信停顿。

## 36.3 内存图和寄存器描述

与系统模式控制器有关的寄存器的信息可以在这里找到。

不同的SMC寄存器在不同的复位类型下复位。每个寄存器的描述都提供了细节。关于本芯片的复位类型的更多信息，请参考复位部分的细节。

**注意事项**

SMC寄存器只能在监督员模式下被写入。在用户模式下的写访问被阻止，并将导致总线错误。

**注意事项**

在执行WFI指令之前，必须将最后写入的寄存器读回。这可以确保在MCU进入低功耗模式之前，所有与设置正在进入的低功耗模式有关的寄存器写入已经完成。
如果不这样做，可能会导致不能正确进入低功率模式。

**SMC内存图**

| 地址偏移（十六进制） | 注册名称 | 宽度（单位：比特） | 访问 | 重置值 | 节/页 |
|---|---|---|---|---|---|
| 0 | SMC版本ID寄存器（SMC_VERID）。 | 32 | R | 0100_0000h | 36.3.1/961 |
| 4 | SMC参数寄存器（SMC_PARAM）。 | 32 | R | 见本节 | 36.3.2/962 |
| 8 | 电源模式保护寄存器（SMC_PMPROT）。 | 32 | R/W | 0000_0000h | 36.3.3/963 |
| C | 电源模式控制寄存器（SMC_PMCTRL）。 | 32 | R/W | 0000_0000h | 36.3.4/965 |
| 10 | 停止控制寄存器（SMC_STOPCTRL）。 | 32 | R/W | 0000_0003h | 36.3.5/967 |
| 14 | 电源模式状态寄存器（SMC_PMSTAT）。 | 32 | R | 0000_0001h | 36.3.6/968 |

### 36.3.1 SMC版本ID寄存器（SMC_VERID）。

地址。0h base + 0h offset = 0h

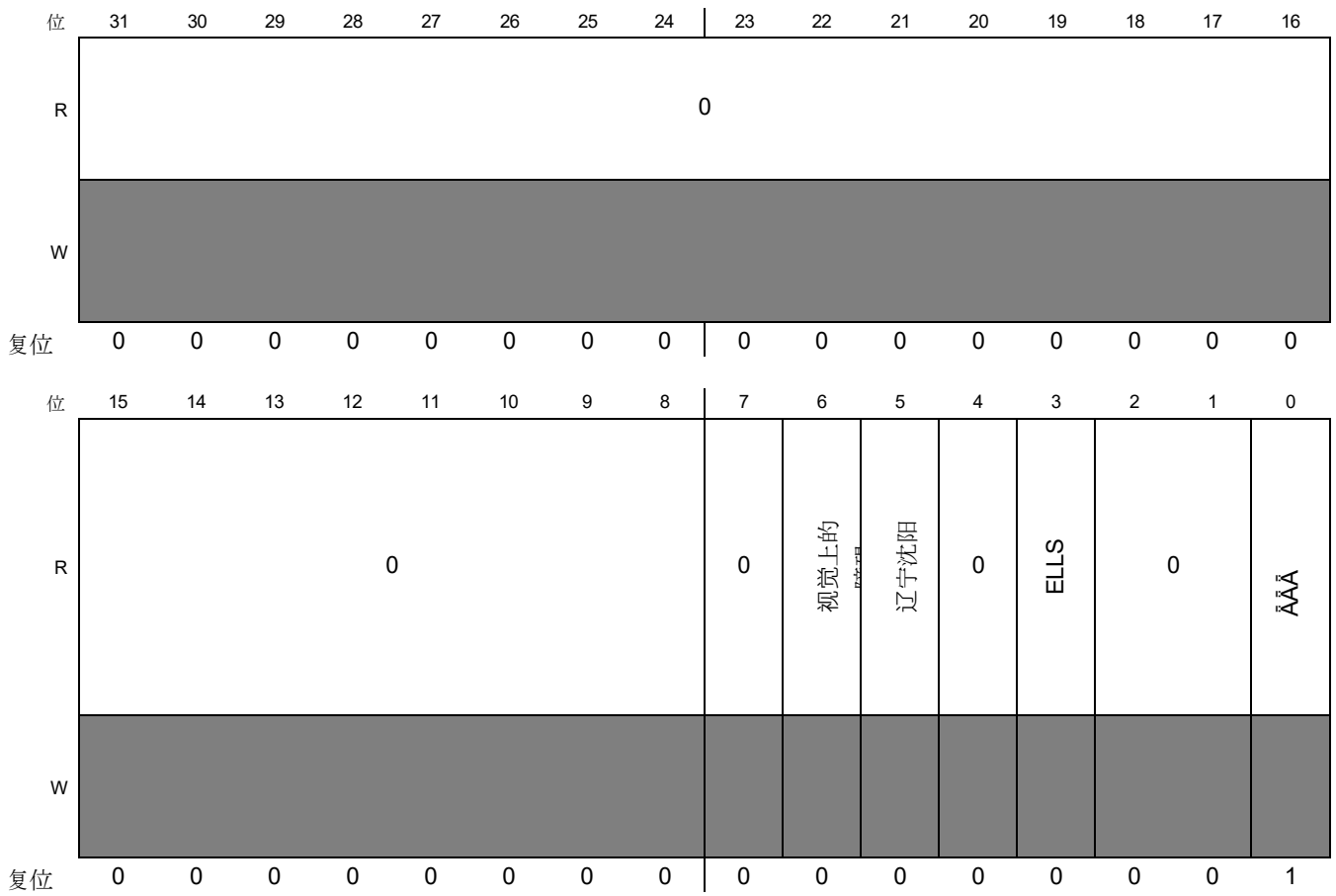
| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 主要的 | | | | | | | | 小学 | | | | | | | | 特写 | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**SMC_VERID字段描述**

| 场地 | 描述 |
|---|---|
| 31-24<br>主要的 | 主要版本号<br><br>这个只读字段返回模块规范的主要版本号。 |
| 23-16<br>小学 | 次要版本号<br><br>这个只读字段返回模块规范的次要版本号。 |
| 特写 | 特点规格号<br><br>这个只读字段返回特征集编号。0x0000　　已实<br><br>施的标准 功能 |

## 36.3.2 SMC参数寄存器（SMC_PARAM）。

地址。0h基数+4h偏移量=4h



**SMC_PARAM字段描述**

| 场地 | 描述 |
|---|---|
| 31-8<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 7<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 6<br>视觉上的障碍<br>(EVLLS0) | VLLS0功能的存在<br>这个静态位说明该功能在设备上是否可用。<br>0该功能不可用。<br>1该功能是可用的。 |
| 5<br>辽宁沈阳 | 存在LLS2功能<br>这个静态位说明该功能在设备上是否可用。<br>0该功能不可用。<br>1该功能是可用的。 |
| 4<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 3<br>ELLS | 存在LLS功能<br>这个静态位说明该功能在设备上是否可用。<br>0该功能不可用。<br>1该功能是可用的。 |
| 2-1<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 0<br>ÅÅÅ | 存在HSRUN功能<br>这个静态位说明该功能在设备上是否可用。<br>0该功能不可用。<br>1该功能是可用的。 |

## 36.3.3 电源模式保护寄存器（SMC_PMPROT）。

该寄存器为进入任何低功率运行或停止模式提供保护。通过配置电源模式控制寄存器（PMCTRL）来实现低功耗运行或停止模式的启用。

在任何系统复位后，PMPROT寄存器只能被写入一次。

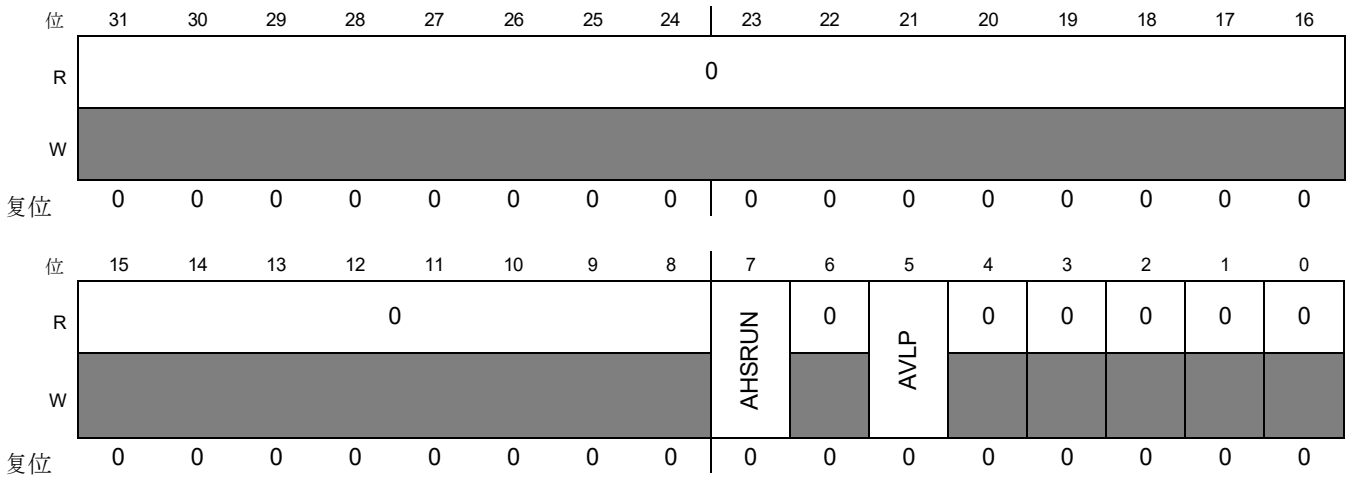
如果MCU被配置为不允许或保留的电源模式，MCU将保持在其当前的电源模式。例如，如果MCU处于正常的RUN模式且AVLP为0，使用PMCTRL[RUNM]进入

VLPR模式的尝试被阻止，PMCTRL[RUNM]保持00b，表明MCU仍处于正常运行模式。

## 注意事项

这个寄存器在芯片复位和触发芯片复位的复位类型中被复位。它不会受到不触发芯片复位的复位类型的影响。更多信息请参见复位部分的细节。

地址。0h基数+8h偏移量=8h




### SMC_PMPROT字段描述

| 场地 | 描述 |
|---|---|
| 31-8<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 7<br>AHSRUN | 允许高速运行模式<br><br>只要在PMCTRL中设置了适当的控制位，这个只写一次的字段允许MCU进入高速运行模式（HSRUN）。<br><br>0 不允许使用HSRUN<br>1 允许HSRUN |
| 6<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 5<br>AVLP | 允许非常低的功率模式<br><br>只要在PMCTRL中设置了适当的控制位，这个只写一次的字段允许MCU进入任何极低功率模式（VLPR和VLPS）。<br><br>0 不允许使用VLPR和VLPS。<br>1 允许使用VLPR和VLPS。 |
| 4<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 3<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |

| | |
|---|---|
| 2<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 1<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |

*表继续在下一页...*

SMC_PMPROT字段描述（续）

| 场地 | 描述 |
|---|---|
| 0<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |

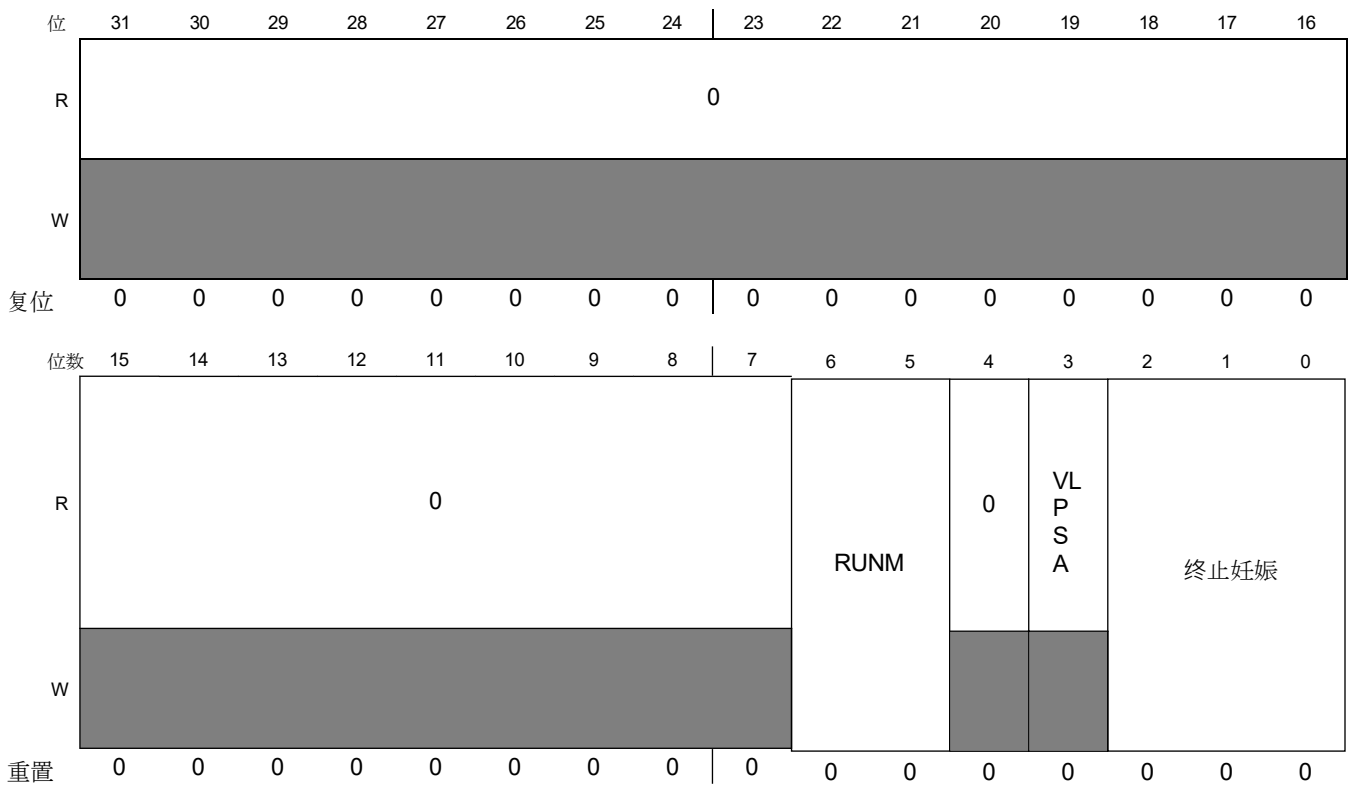
## 36.3.4 电源模式控制寄存器（SMC_PMCTRL）。

PMCTRL寄存器控制进入低功耗的运行和停止模式，前提是通过保护（PMPROT）寄存器的适当设置允许选择的电源模式。

**注意事项**

这个寄存器在芯片POR和触发芯片POR的复位类型中被复位。它不受不触发芯片POR的复位类型的影响。更多信息请参见复位部分的细节。

地址。0h基数+Ch偏移量=Ch



## SMC_PMCTRL字段描述

| 场地 | 描述 |
|---|---|
| 31-7<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 6-5<br>RUNM | 运行模式控制<br>当写入时，导致进入选定的运行模式。如果没有使用PMPROT寄存器启用保护级别，对该字段的写入将被阻止。<br>**注意：**只有当PMSTAT=RUN时，RUNM才能被设置为VLPR。在被写入VLPR后，RUNM不应该被写回RUN，直到PMSTAT=VLPR。<br>**注意：**只有当PMSTAT=RUN时，RUNM才能被设置为HSRUN。在被编程为HSRUN后，RUNM不应该被编程为RUN，直到PMSTAT=HSRUN。另外，当RUNM=HSRUN或PMSTAT=HSRUN时，不应试图进入停止模式。<br>当处于任何模式时，任何复位都会清除RUNM，并使系统在MCU退出其复位流程后退出到正常的RUN模式。<br>00正常运行模式（RUN） 01<br>保留<br>10 极低功率运行模式（VLPR）<br>11 高速运行模式（HSRUN） |
| 4<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 3<br>VLPSA | 非常低的功率停止中止了<br>当设置时，这个只读的状态位表示在前一个停止模式进入序列中发生了中断，阻止系统进入该模式。这个字段在任何停止模式进入序列开始时被重置或被硬件清除，如果序列被中止则被设置。<br>**注意：**如果STOP1/STOP2被中止，这个位不会被设置。<br>0 上一次进入停止模式是成功的。<br>1 前一个停止模式的输入被中止。 |
| 终止妊娠 | 停止模式控制<br>当写入时，控制在SLEEPDEEP=1时进入Sleep-Now或Sleep-On-Exit模式时，进入选定的停止模式。如果没有使用PMPROT寄存器启用保护级别，对该字段的写入将被阻止。在任何系统复位后，这个字段在任何成功写入PMPROT寄存器的情况下被硬件清除。<br>**注意：**当设置为STOP时，必须使用STOPCTRL寄存器中的STOPO位来选择停止的变体--STOP1/STOP2。<br>000正常停止 (STOP) 001<br>保留<br>010超低功率停止（VLPS） 011<br>保留<br>101 保留<br>110 被拒绝 |

| | |
|---|---|
| | 111　保留 |

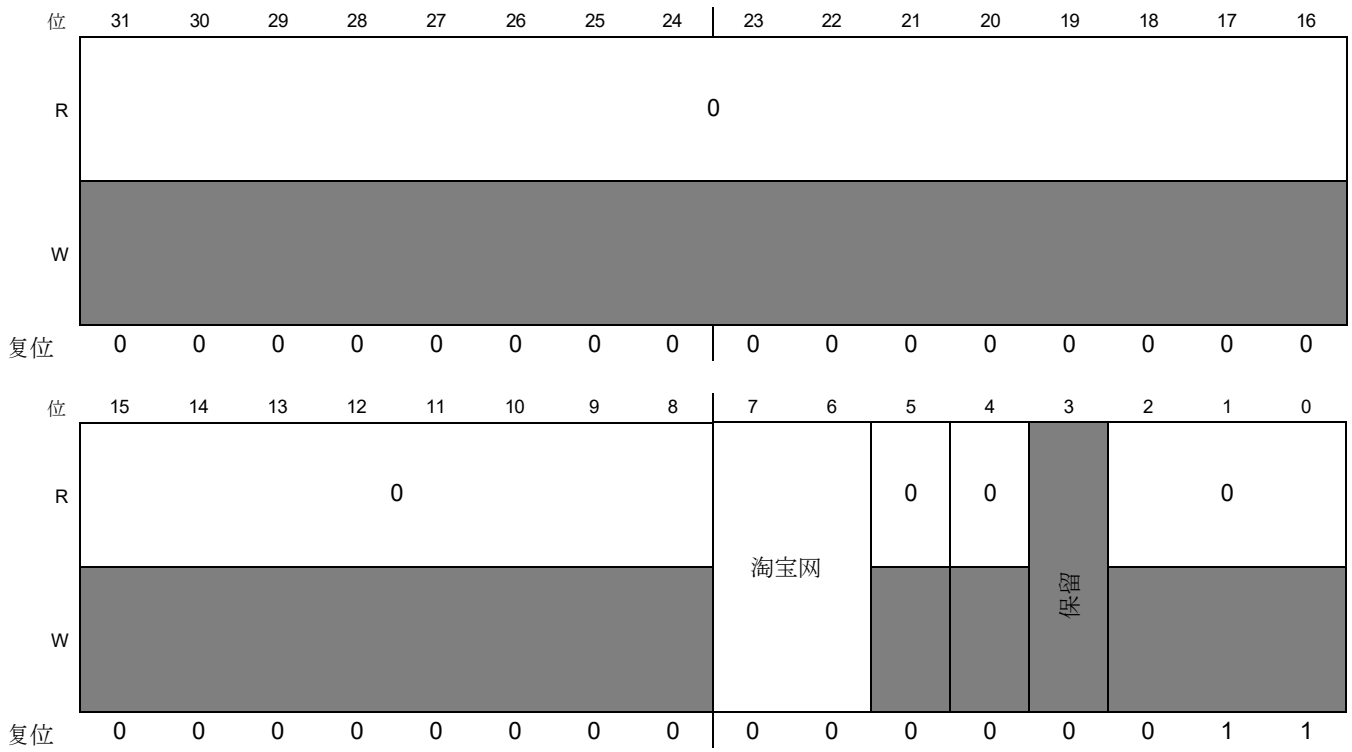
## 36.3.5 停止控制寄存器 (SMC_ STOPCTRL )

STOPCTRL寄存器提供了各种控制位，允许用户在STOPM字段选择的停止模式下对功耗进行微调。

**注意事项**

这个寄存器在芯片POR和触发芯片POR的复位类型中被复位。它不受不触发芯片POR的复位类型的影响。更多信息请参见复位部分的细节。

地址。0h基数+10h偏移量=10h




**SMC_STOPCTRL字段描述**

| 场地 | 描述 |
|---|---|
| 31-8<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 7-6<br>淘宝网 | 停止选项<br>当STOPM=STOP时，这些位控制停止操作。当从RUN模式进入Stop模式时，PMC、SCG和Flash保持完全供电，允许器件几乎瞬间唤醒，但代价是更高的功耗。在STOP2中，只有系统时钟被门控，允许在总线时钟上运行的外围设备保持完全的功能。在STOP1中，系统时钟和总线时钟都被门控。<br>00 保留 |

表继续在下一页...

**SMC_STOPCTRL字段描述（续）**

| 场地 | 描述 |
|---|---|
| | 01STOP1 - 在禁用系统和总线时钟的情况下停止工作<br>10 STOP2 - 在禁用系统时钟和启用总线时钟的情况下停止。<br>11 保留 |
| 5<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 4<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 3<br>保留 | 这个字段是保留的。<br>这个位是为将来的扩展而保留的。软件应将此位写入0以保持兼容性。 |
| 保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |

## 36.3.6 电源模式状态寄存器（SMC_PMSTAT）。

PMSTAT是一个只读的单热寄存器，表示系统当前的电源模式。

**注意事项**

这个寄存器在芯片POR和触发芯片POR的复位类型中被复位。它不受不触发芯片POR的复位类型的影响。更多信息请参见复位部分的细节。

当处于任何模式时，任何复位都会使系统在MCU退出其复位流程后退出到正常的RUN模式。PMSTAT字段就会自动更新，显示RUN为当前的电源模式。

地址。0h基数+14h偏移量=14h

| 位 | 31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 | 15 14 13 12 11 10 9 8 | 7 6 5 4 3 2 1 0 |
|---|---|---|---|
| R | 0 | | PMSTAT |
| W | | | |
| 复位 | 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 | 0 0 0 0 0 0 0 0 | 0 0 0 0 0 0 0 1 |

**SMC_PMSTAT字段描述**

| 场地 | 描述 |
|---|---|
| 31-8<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |

| | |
|---|---|
| PMSTAT | 电源模式状态<br><br>**注意：**当调试被启用时，PMSTAT将不会更新为STOP或VLPS。 |

*表继续在下一页...*

SMC_PMSTAT字段描述（续）

| 场地 | 描述 |
| --- | --- |
| | **注意：**当启用STOP模式时，PMSTAT将不会更新为STOP<br><br>0000_0001 当前的电源模式是RUN。<br>0000_0010 保留。<br>0000_0100 当前的电源模式是VLPR。<br>0000_1000 保留。<br>0001_0000 当前的电源模式是VLPS。<br>0010_0000 保留<br>0100_0000 保留<br>1000_0000 当前的电源模式是HSRUN |

## 36.4 职能描述

### 36.4.1 电源模式转换

下图显示了芯片上可用的电源模式状态转换。任何复位都会使MCU回到正常的RUN状态。

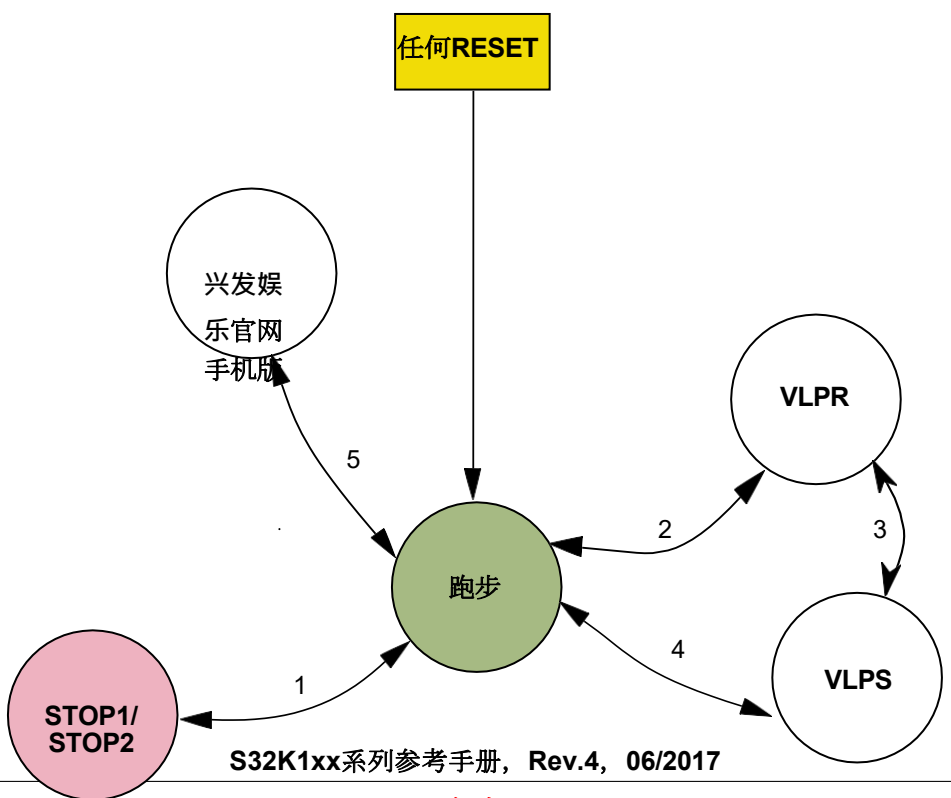


图36-1.电源模式状态图

下表定义了上图所示各种状态转换的触发器。

**表36-2.电源模式转换触发器**

| 过渡期 # | 来自 | 拟 | 触发条件 |
|---|---|---|---|
| 1 | 跑步 | 停止 | pmctrl[runm]=00, pmctrl[stopp]=000 [1]<br>在设置SLEEPDEEP的情况下进入现在睡眠或退出睡眠模式，该模式在ARM内核的系统控制寄存器中控制。 |
| | 停止 | 跑步 | 中断或复位 |
| 2 | 跑步 | VLPR | 在这种模式下，核心、系统、总线和闪存的时钟频率以及SCG时钟模式都受到限制。关于支持的最大允许频率和SCG模式，请参见电源管理章节。<br>设置PMPROT[AVLP]=1，PMCTRL[RUNM]=10。 |
| | VLPR | 跑步 | 设置PMCTRL[RUNM]=00或<br>重置。 |
| 3 | VLPR | VLPS | PMCTRL[STOPM]=000或010。<br>在设置SLEEPDEEP的情况下进入现在睡眠或退出睡眠模式，该模式在ARM内核的系统控制寄存器中控制。 |
| | VLPS | VLPR | 中断<br>**注意：**如果VLPS是直接从RUN进入的（过渡#4），硬件强制退出到RUN，不允许过渡到VLPR。 |
| 4 | 跑步 | VLPS | PMPROT[AVLP]=1, PMCTRL[STOPM]=010,<br>在设置SLEEPDEEP的情况下进入现在睡眠或退出睡眠模式，该模式在ARM内核的系统控制寄存器中控制。 |
| | VLPS | 跑步 | 中断和VLPS模式是直接从RUN或VLPS进入的。<br>复位 |
| 5 | 跑步 | 兴发娱乐官网手机版 | 设置PMPROT[AHSRUN]=1，PMCTRL[RUNM]=11。 |
| | 兴发娱乐官网手机版 | 跑步 | 设置PMCTRL[RUNM]=00或<br>复位 |

1.STOPO寄存器必须被配置为选择停止模式的变量。

## 36.4.2 电源模式进入/退出顺序

当进入或退出低功耗模式时，系统必须符合一个有序的顺序，以安全地管理过渡。

SMC管理着系统进入和退出所有的电源模式。该图说明了SMC与芯片中其他系统组

件的连接，这些连接是系统在所有电源模式中排序所必需的。

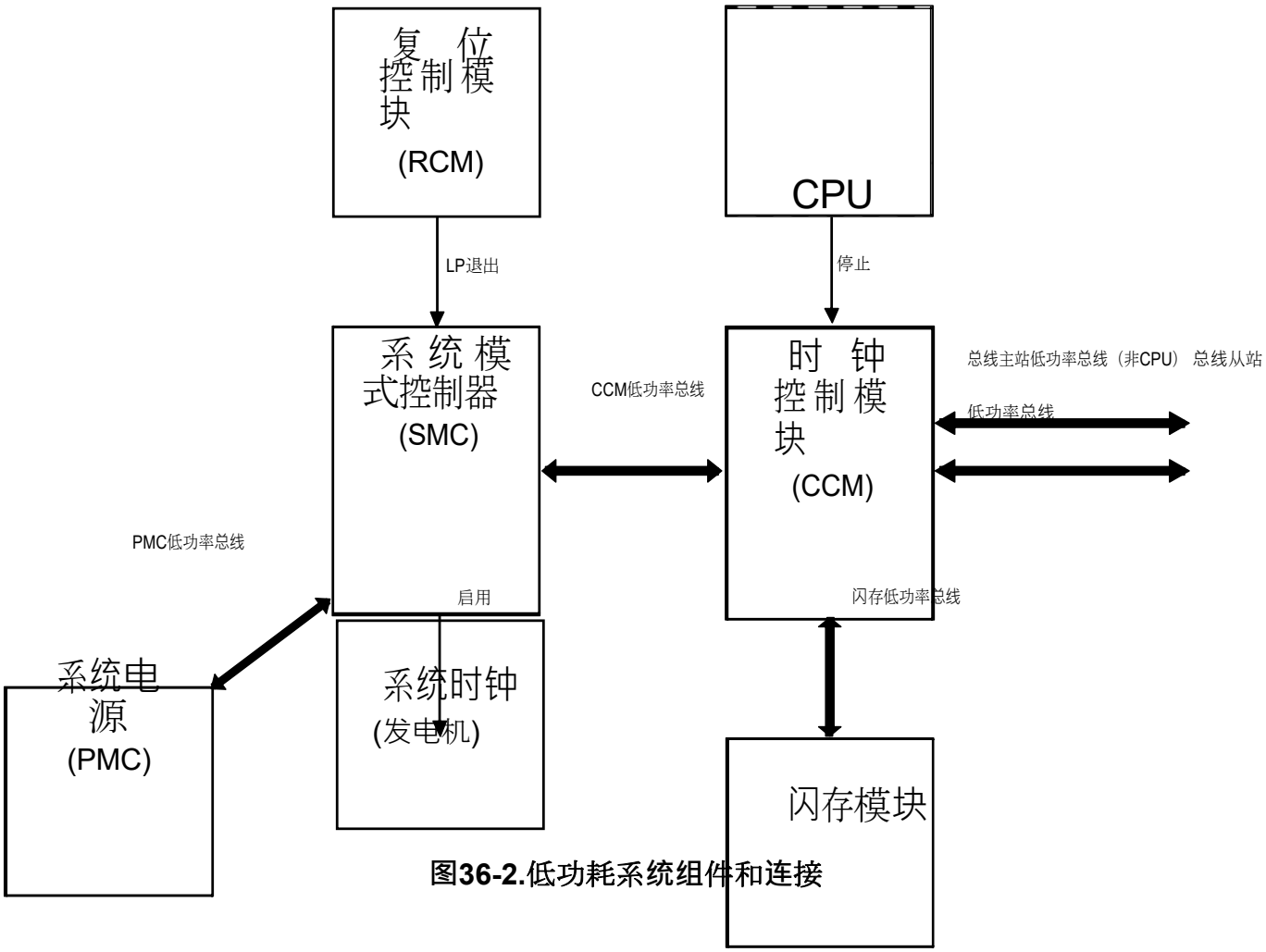



**图36-2.低功耗系统组件和连接**

### 36.4.2.1 VLPS/Stop模式进入顺序

进入低功耗停止模式（停止，VLPS）是由CPU执行WFI指令启动的。该指令执行后，会出现以下序列。

1. CPU的时钟立即被关上。
2. 向所有非CPU总线主站发出进入停止模式的请求。
3. 在所有主站确认准备进入 "停止 "模式后，向所有总线从站发出进入 "停止 "模式的请求。
4. 在所有的从机都确认他们准备好进入停止模式后，系统和总线时钟会根据目标模式--VLPS/STOP1/STOP2--
   进行门控关闭。请注意，在STOP2模式下，总线时钟将不会被门控。
5. 此外，对于VLPS模式，时钟发生器在SCG中被禁用，除非被配置为启用。参见SCG的编程选项。

6. PMC中的片上稳压器和内部电源开关被配置为满足目标低功耗模式的功耗目标。这一步只对VLPS有效，对STOP无效，因为在STOP模式下，PMC处于运行调节状态。

#### 36.4.2.2 VLPS/Stop模式退出顺序

从低功耗停止模式的退出是由一个复位或一个中断事件启动的。然后执行以下程序，将系统恢复到运行模式（RUN或VLPR）。

1. PMC的片上调节器、时钟发生器和内部电源开关被恢复。这一步只对VLPS有效，对STOP无效，因为在STOP模式下，PMC处于运行调节状态。
2. 系统和总线时钟被启用到所有主站和从站。
3. CPU时钟被启用，CPU开始为启动退出低功耗停止模式的复位或中断服务。

#### 36.4.2.3 终止了极低功率停止模式的进入

如果在停止进入序列中发生中断，SMC可以提前中止过渡，返回到RUN模式，而不完全进入停止模式。只有当中断发生在PMC开始过渡到停止模式调节之前，才有可能中止进入。在这一点上，中断被忽略，直到PMC完成向停止模式调节的过渡。当一个中止的极低功率停止模式进入序列发生时，SMC_PMCTRL[VLPSA]被设置为1。

#### 36.4.2.4 从停止模式过渡到调试模式

当调试器被启用时，调试器模块支持从STOP和VLPS过渡到Halted状态。作为这种转换的一部分，系统时钟被重新建立，相当于正常的RUN和VLPR模式的时钟配置。

### 36.4.3 运行模式

该设备所支持的运行模式可以在这里找到。

- 运行 (RUN)
- 极低功率运行(VLPR)
- 高速运行（HSRUN）。

#### 36.4.3.1 运行模式

这是设备的正常工作模式。

这种模式在任何复位后都被选择。当ARM处理器退出复位时，它设置了堆栈、程序计数器（PC）和链接寄存器（LR）。

- 处理器从矢量表的偏移量0x000读取起始SP（SP_main）。
- 处理器从矢量表偏移量0x004读取起始PC
- LR被设置为0xFFFF_FFFF。

为了减少这种模式下的功耗，请禁用未使用的模块的时钟。

#### 36.4.3.2 极低功率运行（VLPR）模式

在VLPR模式下，片上电压调节器被置于停止模式调节状态。在这种状态下，稳压器被设计为在降低的频率上向MCU提供足够的电流。为了在这种模式下进一步降低功率，使用PCC的寄存器中相应的时钟门控位来禁用未使用的模块的时钟。

在进入该模式之前，必须满足以下条件。

- 必须禁用SCG中的所有时钟监视器。
- 系统、总线、闪存和内核的最大频率受到限制。关于支持哪些频率，请参见电源管理的详细信息。
- 模式保护必须设置为允许VLP模式，也就是说，PMPROT[AVLP]为1。
- PMCTRL[RUNM]必须设置为10b才能进入VLPR。
- 不允许进行闪存编程/擦除。

**注意事项**

在VLPR模式下，不要增加时钟频率，因为调节器的反应很慢，不能管理快速的负载转换。此外，不要修改SCG模块或任何时钟分频器寄存器中的时钟源。PCC中的模块时钟使能可以被设置，但不能被清除。

要重新进入正常运行模式，清除PMCTRL[RUNM]。PMSTAT是一个只读的状态寄存器，可以用来确定系统何时完成退出RUN模式。当PMSTAT=RUN时，系统处于运行调节状态，MCU可以在任何时钟模式下全速运行。如果需要更高的执行频率，在从VLPR模式返回时，轮询PMSTAT直到它被设置为RUN。

任何复位都会导致退出VLPR，并在MCU退出其复位流程后将器件返回到RUN模式。

### 36.4.3.3 高速运行（HSRUN）模式

在HSRUN模式下，片上电压调节器仍处于运行调节状态，但电压输出略有升高。在这种状态下，与正常RUN模式相比，MCU能够以更快的频率运行。关于最大允许的频率，请参见电源管理一章。

在这种模式下，必须遵守以下限制。

- 系统、总线、闪存或核心时钟的频率的最大允许变化被限制在3倍（频率的三倍）。
- 不支持从HSRUN进入停止模式。
- 禁止对时钟门控位进行修改。
- 不允许进行闪存编程/擦除。

要进入HSRUN模式（用112MHz SPLL作为时钟源）。

1. 禁用SPLL。通过向SCG_RCCR[SCS]和SCG_HCCR[SCS]写入0011，配置FIRC作为RUN模式和HSRUN模式的时钟源。
2. 配置PMPROT[AHSRUN]以允许HSRUN。
3. 写11到SMC_PMCTRL[RUNM]，进入HSRUN。(现在系统将进入HSRUN模式，FIRC被配置为系统时钟源）。
4. 为112MHz重新配置PLL，并启用它。
5. 通过将SCG_HCCR[SCS]配置为0110，切换到PLL作为时钟源。

进入HSRUN模式（用80MHz SPLL作为时钟源）。

1. 通过向SCG_RCCR[SCS]和SCG_HCCR[SCS]写入0110，配置80MHz的SPLL作为RUN模式和HSRUN模式的时钟源。
2. 配置PMPROT[AHSRUN]以允许HSRUN。
3. 写11到SMC_PMCTRL[RUNM]，进入HSRUN。

在增加时钟频率之前，应该轮询PMSTAT寄存器以确定系统何时完成进入HSRUN模式。要重新进入正常的RUN模式，请清除PMCTRL[RUNM]。任何复位也会清除PMCTRL[RUNM]，并使系统在MCU退出复位流程后退出到正常RUN模式。

## 36.4.4 停止模式

通过设置PMPROT和PMCTRL中的相应字段来选择各种停止模式。所选择的停止模式是在现在睡眠或退出睡眠时进入的，在ARM内核的系统控制寄存器中设置SLEEPDEEP位。

可用的停止模式是。

- 正常停止 (STOP)
- 极低功率停止 (VLPS)

### 36.4.4.1 停机模式

STOP模式是通过在ARM内核的系统控制寄存器中设置SLEEPDEEP位的现在睡眠或退出睡眠来进入的。

SCG模块可以被配置为让参考时钟运行。

一个能够向设备提供异步中断的模块会使设备脱离STOP模式，并使设备返回到正常的RUN模式。关于STOP模式下的外设、I/O和内存操作，请参考设备的电源管理章节。当一个中断请求发生时，CPU退出STOP模式并恢复处理，从导致中断服务例程的堆叠操作开始。

系统复位将导致退出STOP模式，通过MCU复位使设备回到正常的RUN模式。

### 36.4.4.2 极低功率停止（VLPS）模式

这里列出了可以进入VLPS模式的两种方式。

- 当MCU处于VLPR模式且PMCTRL[STOPM]=010或000时，在ARM内核的系统控制寄存器中设置SLEEPDEEP位，通过现在睡眠或退出时睡眠进入停止状态。
- 当MCU处于正常的RUN模式且PMCTRL[STOPM]=010时，在ARM内核的系统控制寄存器中设置SLEEPDEEP位，通过现在睡眠或退出时睡眠进入停止。当从RUN模式直接进入VLPS时，退出到VLPR被硬件禁止，系统将总是退出到RUN。

在VLPS中，片上电压调节器与VLPR中一样保持在停止调节状态。

一个能够向设备提供异步中断的模块将设备从VLPS中取出，并将设备返回到VLP

R模式。

系统复位也会导致VLPS退出，使设备回到正常的RUN模式。

### 36.4.5 低功耗模式下的调试

当MCU是安全的，设备会禁用/限制调试器的操作。当MCU不安全时，ARM调试器可以发出两个上电请求信号。

- 系统上电，通过调试端口控制/状态寄存器中的SYSPWR。
- 调试上电，通过调试端口控制/状态寄存器中的CDBGPWRUPREQ。

当处于RUN或VLPR状态时，模式控制器为每个信号驱动一个相应的确认，即CDBGPWRUPACK和CSYSPWRUPACK。当这两个请求被断言时，模式控制器通过进入模拟停止状态来处理进入STOP和VLPS的尝试。在这个模拟的停止状态中。

- 调节器处于运行调节状态。
- SCG生成的时钟源被启用。
- 所有的系统时钟，除了核心时钟，都被禁用。
- 调试模块可以访问核心寄存器，并且
- 对片上外设的访问被阻止了。

# 第37章
# 电源管理控制器（PMC）

## 37.1 芯片特定的PMC信息

### 37.1.1 支持的模式

该设备不支持等待和VLPW模式。关于可用的电源模式的详细信息，请参见模块在可用的低功率模式下的操作。

## 37.2 简介

PMC包含内部电压调节器、上电复位（POR）和低电压检测（LVD）系统。

## 37.3 特点

PMC的功能包括。

- 内部电压调节器提供各种电源模式
- 有源POR提供断电检测
- 低电压复位（LVR）
- 低电压检测支持两个低电压跳闸点和中断
- 低功率振荡器（LPO），典型频率为128kHz

## 37.4 操作模式

### 37.4.1 全性能模式 (FPM)

对于以下芯片电源模式，内部电压调节器处于全性能模式。HSRUN，RUN，STOP1，STOP2。

### 37.4.2 低功率模式（LPM）

对于以下芯片电源模式，内部电压调节器处于低功率模式。VLPR和VLPS。

## 37.5 低电压检测（LVD）系统

**注意事项**

低电压检测系统（低电压检测标志、低电压警告标志和低电压检测复位生成）在低功率模式下被禁用。

该器件包括一个防范低电压条件的系统。这可以在电源电压变化期间保护存储器内容并控制MCU系统状态。该系统由一个上电复位（POR）电路和一个有两个跳闸点的LVD电路组成。LVD在进入低功率模式时被禁用。

有两个标志可用于指示低电压检测系统的状态。

- 低电压检测标志（LVDF）以电平敏感的方式工作。当电源电压低于跳闸点（$V_{LVD}$）时，LVDF位被设置。通过向LVDACK位写1，LVDF位被清除，但只有当内部电源恢复到跳闸点以上时，LVDF位才被设置。这个标志在复位时被清除。该标志只有在器件从复位中出来后才有效，这时该标志将根据电压水平进行相应设置。如果电源电平高于LVD阈值，则该标志保持清零，否则该标志被设置。
- 低电压警告标志（LVWF）以电平敏感的方式运行。当电源电压低于选定的监控器跳闸点（$V_{LVW}$）时，LVWF位被设置。通过向LVWACK位写1，LVWF位被清除，但只有当内部电源恢复到跳闸点以上时，LVWF位才保持设置。这个标志在复位时被清除。该标志只有在器件从复位中出来后才有效，此时该标志将根据电压水平进行相应的设置。如果电源电平高于LVW阈值，则该标志保持清零，否则该标志被设置。

### 37.5.1 低电压复位（LVR）操作

如果电源电压低于复位跳闸点（$V_{LVR}$），将产生一个系统复位。

如果PMC_LVDSC1[LVDRE]被设置，并且电源电压低于$V_{LVD}$，将产生一个系统复位。

PMC_LVDSC1[LVDF]将被系统复位清除，所以恢复后PMC_LVDSC1[LVDF]将读为0。PMC_LVDSC1[LVDF]的使用仅用于LVD中断选择（例如，PMC_LVDSC1[LVDIE]=1，PMC_LVDSC1[LVDRE]=0）。

### 37.5.2 LVD中断操作

通过将LVD电路配置为中断操作（LVDIE设置），PMC_LVDSC1[LVDF]被设置，并且在检测到低电压条件时发生LVD中断请求。当电源恢复到跳闸点以上时，通过向PMC_LVDSC1[LVDACK]位写1，LVDF位被清除。

### 37.5.3 低电压警告（LVW）中断操作

LVD系统包含一个低电压警告标志（LVWF），用于指示电源电压正在接近但高于LVD电压。LVW也有一个中断，通过设置PMC_LVDSC2[LVWIE]位启用。如果启用，当LVWF被设置时，就会发生一个LVW中断请求。当电源恢复到跳闸点以上时，通过向PMC_LVDSC2[LVWACK]位写1清除LVWF。

## 37.6 内存地图和寄存器的定义

本节提供了PMC模块的所有寄存器的详细信息。

### 37.6.1 PMC寄存器描述

**注意事项**

PMC寄存器的不同部分只能由特定的复位类型来复位。每个寄存器的描述都提供了细节。

**注意事项**

PMC的寄存器只能在监管者模式下被写入。在用户模式下的写访问被阻止，并会导致总线错误。

### 37.6.1.1 PMC内存图

PMC基础地址。4007_D000h

| 偏移 | 注册 | 宽度 (单位：比特) | 访问 | 重置值 |
|---|---|---|---|---|
| 0h | 低电压检测状态和控制1寄存器（LVDSC1） | 8 | RW | 见描述。 |
| 1h | 低电压检测状态和控制2寄存器（LVDSC2） | 8 | RW | 00h |
| 2h | 稳压器状态和控制寄存器（REGSC）。 | 8 | RW | 见描述。 |
| 4h | 低功率振荡器微调寄存器（LPOTRIM）。 | 8 | RW | 见描述。 |

### 37.6.1.2 低电压检测状态和控制1寄存器（LVDSC1）

#### 37.6.1.2.1 偏移

| 注册 | 偏移 |
|---|---|
| LVDSC1 | 0h |

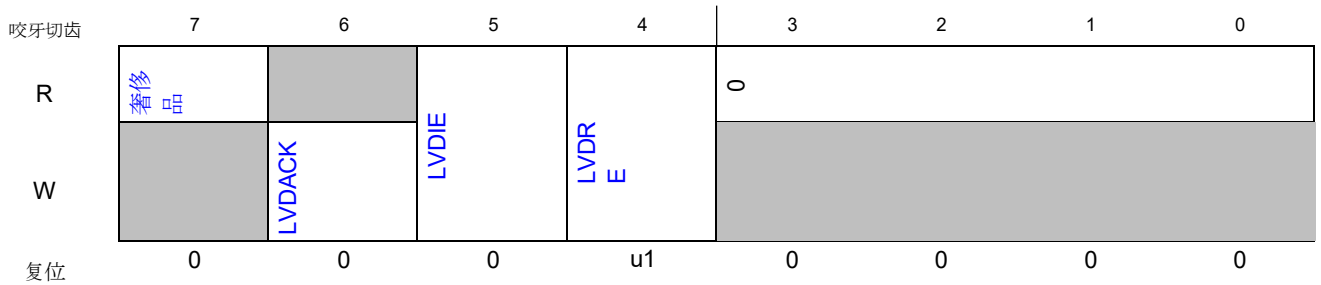
#### 37.6.1.2.2 职能

该寄存器包含状态和控制位，以支持低电压检测功能。

**注意事项**

当内部电压调节器处于低功耗模式时，无论PMC_LVDSC1设置如何，LVD系统都被禁用。

### 37.6.1.2.3 图示




1.POR后LVDRE=0b。不受复位的影响。

### 37.6.1.2.4 领域

| 场地 | 职能 |
|---|---|
| 7<br>奢侈品 | 低电压检测标志<br>该位的只读状态位表示低电压检测事件。阈值电压为$V_{LVD}$。<br>0b - 未检测到低电压事件 1b - 检测到<br>低电压事件 |
| 6<br>LVDACK | 低电压检测确认<br>这个只写的位用于确认低电压检测错误。写1来清除LVDF。读取时总是返回0。 |
| 5<br>LVDIE | 低电压检测中断启用<br>该位启用LVDF的硬件中断请求。<br>0b - 禁用硬件中断（使用轮询）。<br>1b - 当LVDF=1时，请求一个硬件中断 |
| 4<br>LVDRE | 低电压检测 复位启用<br>该位使低电压检测事件产生一个系统复位。<br>0b - 在低电压检测事件中没有系统复位。<br>1b - 如果电源电压低于$V_{LVD}$，将产生一个系统复位。 |
| 3-0<br>- | 保留 |

## 37.6.1.3 低电压检测状态和控制2寄存器（LVDSC2）

### 37.6.1.3.1 偏移

| 注册 | 偏移 |
| --- | --- |
| LVDSC2 | 1h |

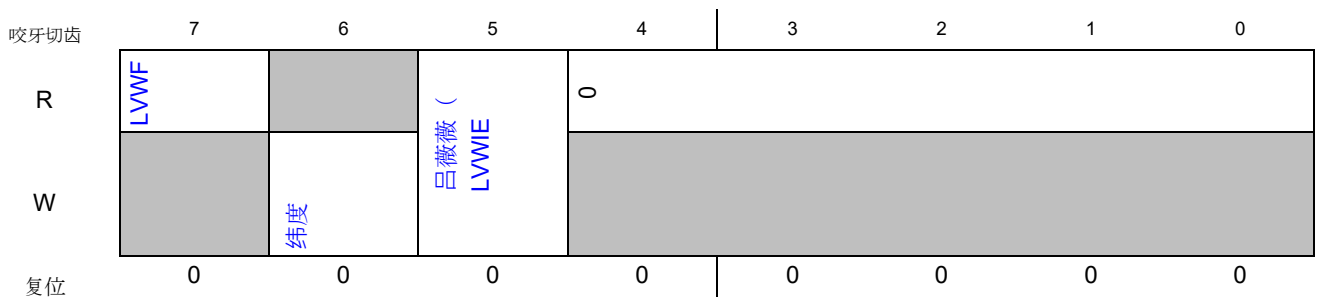
### 37.6.1.3.2 职能

该寄存器包含状态和控制位，支持低电压警告（LVW）功能。

**注意事项**

当内部电压调节器处于低功耗模式时，无论PMC_LVDSC2设置如何，LVD系统都被禁用。

### 37.6.1.3.3 图示




### 37.6.1.3.4 领域

| 场地 | 职能 |
|---|---|
| 7<br>吕梁日报 | 低电压警告标志<br>该位只读状态位表示低电压检测事件。阈值电压为$V_{LVW}$。<br>0b - 未检测到低电压警告事件 1b - 检测到低电压警告事件 |
| 6<br>纬度 | 低电压警告确认<br>这个只写的位用于确认低电压警告错误。写1来清除LVWF。读取时总是返回0。 |
| 5<br>吕薇薇（LVWIE | 低电压警告中断启用<br>该位启用LVWF的硬件中断请求。<br>0b - 禁用硬件中断（使用轮询）。<br>1b - 当LVWF=1时请求一个硬件中断 |
| 4-0<br>- | 保留 |

### 37.6.1.4 稳压器状态和控制寄存器（REGSC）。

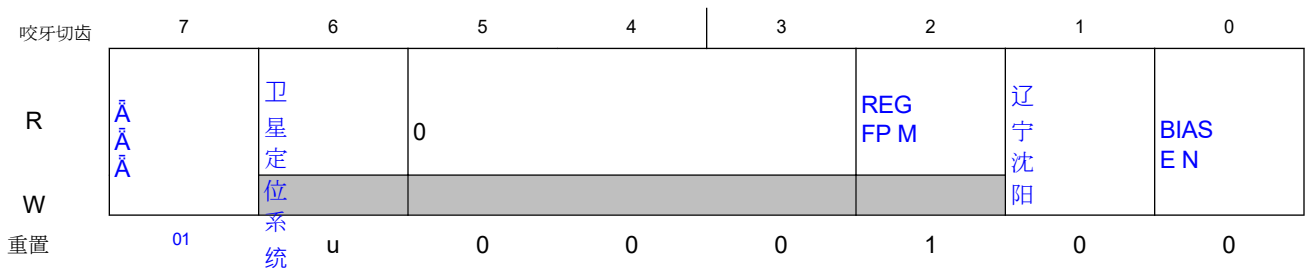
### 37.6.1.4.1 偏移

| 注册 | 偏移 |
|---|---|
| REGSC | 2h |

### 37.6.1.4.2 职能

该寄存器包含调节器和LPO的一般控制和状态位。

### 37.6.1.4.3 图示




1.上电复位时被清除，不受其他复位的影响。

### 37.6.1.4.4 领域

| 场地 | 职能 |
|---|---|
| 7<br>ÃÃÃ | LPO 禁用位<br>该位启用或禁用低功率振荡器。<br>**注意：**在禁用LPO后，需要2个LPO时钟周期的时间才允许再次启用它。违反这2个周期的等待时间可能导致LPO的故障。<br>0b - 低功率振荡器启用 1b - 低功率振荡器禁用 |
| 6<br>卫星定位系统(LPOSTAT) | LPO状态位<br>该位显示 LPO 时钟的状态是处于时钟周期的高相位（逻辑 1）或低相位（逻辑 0）。软件可以通过轮询这个状态位来测量实际的LPO时钟频率，最终使用LPOTRIM[4:0]寄存器来改变LPO频率。<br>0b - 低相位的低功率振荡器 1b - 高相位的低功率振荡器 |
| 5-3<br>- | 保留 |
| 2<br>REGFPM | 处于全性能模式的调节器状态位<br>这个只读位提供内部电压调节器的当前状态。 |

| | |
|---|---|
| | 0b - 稳压器处于低功耗模式或过渡到/过渡到 1b - 稳压器处于全性能模式 |
| 1<br>辽宁省 | 时钟偏置禁用位 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| | 该位禁用某些时钟模块的偏置电流和参考电压，以进一步降低VLPS模式下的功耗。<br>**注意：在使用该位时，必须确保在VLPS模式下禁用各个时钟模块。否则，将发生时钟模块的严重故障。**<br>0b - 没有影响<br>1b - 在VLPS模式下，下列时钟模块的偏置电流和参考电压被禁用。SIRC、FIRC、PLL。(如果设备上有的话) |
| 0<br>白森 | 偏压启用位<br>该位在低功耗模式下为核心逻辑启用源和井偏压。在全性能模式下，该位没有影响。这对于在低功耗模式下进一步降低MCU的功耗是非常有用的。当使用VLP*模式时，该位必须被设置为1。<br>0b - 禁用偏压，核心逻辑可以全性能运行<br>1b - 启用偏置，核心逻辑较慢，允许的系统时钟速度有限制（详见数据表）。 |

## 37.6.1.5 低功率振荡器微调寄存器（LPOTRIM）。

### 37.6.1.5.1 偏移

| 注册 | 偏移 |
|---|---|
| LPOTRIM | 4h |

### 37.6.1.5.2 职能

该寄存器包含低功率振荡器的周期微调位。

**表37-1.LPOTRIM[4:0]的修剪效果**

| LPOTRIM[4:0] | 十进制 | LPO时钟的周期 |
|---|---|---|
| 10000 | -16 | 最低的 |
| 10001 | -15 | 增加 |
| ... | ... | |
| 11110 | -2 | |
| 11111 | -1 | |
| 00000 | 0 | 典型的128kHz |
| 00001 | +1 | 增加 |
| ... | ... | |
| 01110 | +14 | |
| 01111 | +15 | 最高 |

**注意事项**

LPO的修剪位代表有符号的值。从
-
16，LPO时钟的周期将单调地增加（例如，频率单调地减少）。

### 37.6.1.5.3 图示

| 咬牙切齿 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| R | 0 | | | LPOTRIM | | | | |
| W | | | | | | | | |
| 复位 | 0 | 0 | 0 | u1 | u | u | u | u |

1.在任何重置后自动从闪存IFR加载。

### 37.6.1.5.4 领域

| 场地 | 职能 |
|---|---|
| 7-5<br>- | 保留 |
| 4-0<br>LPOTRIM | LPO修饰位<br>这些位用于修整低功率振荡器的频率。修剪效果见上表。 |

# 第三十八章
# ADC配置

## 38.1 实例化信息

S32K14x包含两个12位ADC模块，ADC0和ADC1。S32K11x

包含一个12位ADC模块，即ADC0。

该设备不支持Wait/VLPW模式。关于可用的电源模式的详情，请参见模块在可用的低功率模式下的操作。

**注意事项**

在VLP模式下，带隙电压将无法使用。

### 38.1.1 ADC通道的数量

每个ADC最多支持32个外部模拟输入通道，但器件上存在的确切ADC通道数随封装不同而不同，如下表所示。

有关特定封装上可用的特定ADC通道的详细信息，请参见简介。

**表38-1.每个封装的ADC外部通道**

| 芯片 | 包装 | ADC0 | ADC1 |
|---|---|---|---|
| S32K116 | 32 QFN | 8 | 这是对的。 |
| | 48 LQFP | 15 | 这是对的。 |
| S32K118 | 48 LQFP | 14 | 这是对的。 |
| | 64 LQFP | 16 | 这是对的。 |
| S32K142 | 64 LQFP | 16 | 11 |
| | 100 LQFP | 16 | 16 |
| S32K144 | 64 LQFP | 16 | 11 |
| | 100 LQFP | 16 | 16 |
| | 100 MAPBGA | 16 | 16 |

表继续在下一页...

表38-1.每个封装的ADC外部通道（续）

| 芯片 | 包装 | ADC0 | ADC1 |
|---|---|---|---|
| S32K146 | 100 LQFP | 16 | 16 |
| | 144 LQFP | 24 | 24 |
| | 100 MAPBGA | 16 | 16 |
| S32K148 | 144 LQFP | 32 | 32 |
| | 100 MAPBGA | 16 | 16 |
| | 176 LQFP | 32 | 32 |

一个芯片的ADCx_SC1n[ADCH]位域配置是按照该芯片跨封装的最大通道配置，如下表所述。

表38-2.ADCx_SC1n[ADCH]字段配置

| 芯片 | ADCx_SC1n[ADCH]范围 | 禁用模块的ADCx_SC1n[ADCH]配置 |
|---|---|---|
| S32K116 | 00000 B - 11111 B | 11111 b |
| S32K118 | 00000 B - 11111 B | 11111 b |
| S32K142 | 00000 B - 11111 B | 11111 b |
| S32K144 | 00000 B - 11111 B | 11111 b |
| S32K146 | 000000 B - 111111 B | 111111 b |
| S32K148 | 000000 B - 111111 B | 111111 b |

在一个特定的包中，不可用的频道被保留给相应的包。

**注意事项**

有关特定封装的通道细节，请参见《参考手册》所附的IO信号描述输入复用表。

## 38.1.2 ADC连接/通道分配

有关ADC通道连接的细节，请参见《参考手册》所附的IO信号描述输入复用表。

注意，ADCx ADy和ADCx_SEy都是指ADCx的通道y。

表38-3.内部通道的可用性

| ADC内部通道 | ADC0 | ADC1 |
|---|---|---|
| 频道0 | 用于电源监控（详见ADC内部电源监控）。 | 保留 |

*表继续在下一页...*

表38-3.内部通道的可用性（续）

| ADC内部通道 | ADC0 | ADC1 |
|---|---|---|
| 频道1 | 保留 | 保留 |
| 频道2 | 保留 | 保留 |
| 第三频道 | 保留 | 保留 |

## 38.2 登记实施

在S32K14x的每个变体中，并非所有ADC内存图中显示的寄存器都能实现。关于寄存器的实现，见下表。

表38-4.S32K14x寄存器的实现

| 寄存器/偏移量 | S32K142 | S32K144 | S32K148 |
|---|---|---|---|
| ADC_SC1A-ADC_SC1FF/108-184 | 未实施 | 未实施 | 已实施 |
| ADC_RA-RFF/188-204 | 未实施 | 未实施 | 已实施 |

## 38.3 ADC上的DMA支持

应用可能需要对ADC进行连续采样（4K样本/秒），这可能会对CPU产生相当大的负荷。尽管使用可编程延迟块（PDB）来触发ADC可能会减少一些CPU负载，但当ADC以非常高的速率采样或在绕过PDB的情况下，ADC支持DMA请求功能，以获得更高的性能。ADC可以在转换完成后触发DMA（通过DMA请求）。

对于大多数情况，DMA请求可以从ADC转换完成后直接触发。该器件还支持另一种触发DMA的方式：通过TRGMUX。TRGMUX将根据不同的应用需求，利用软件为用户提供更灵活的DMA触发方案，例如，DMA可以在多个ADC转换完成后被触发，而不是在每个ADC转换完成后被触发。

## 38.4 ADC硬件交错通道

在这个设备上，有几个特殊的ADC通道，支持多个ADC之间的硬件交错。以ADC0_SE4和ADC1_SE14通道为例，这两个通道可以独立工作，但它们也可以如下图所示进行硬件交错。在硬件交错模式下，引脚PTB0上的信号可以被ADC0和ADC1同时采样。交错模式由SIM_CHIPCTL[ADC_INTERLEAVE_EN]位启用。

该设备上的硬件交错实现如下。

- ADC0_SE4和ADC1_SE14通道在PTB0引脚上交错排列
- ADC0_SE5和ADC1_SE15通道在PTB1引脚上交错排列
- ADC1_SE8和ADC0_SE8通道在PTB13引脚上交错排列
- ADC1_SE9和ADC0_SE9通道在PTB14引脚上交错排列

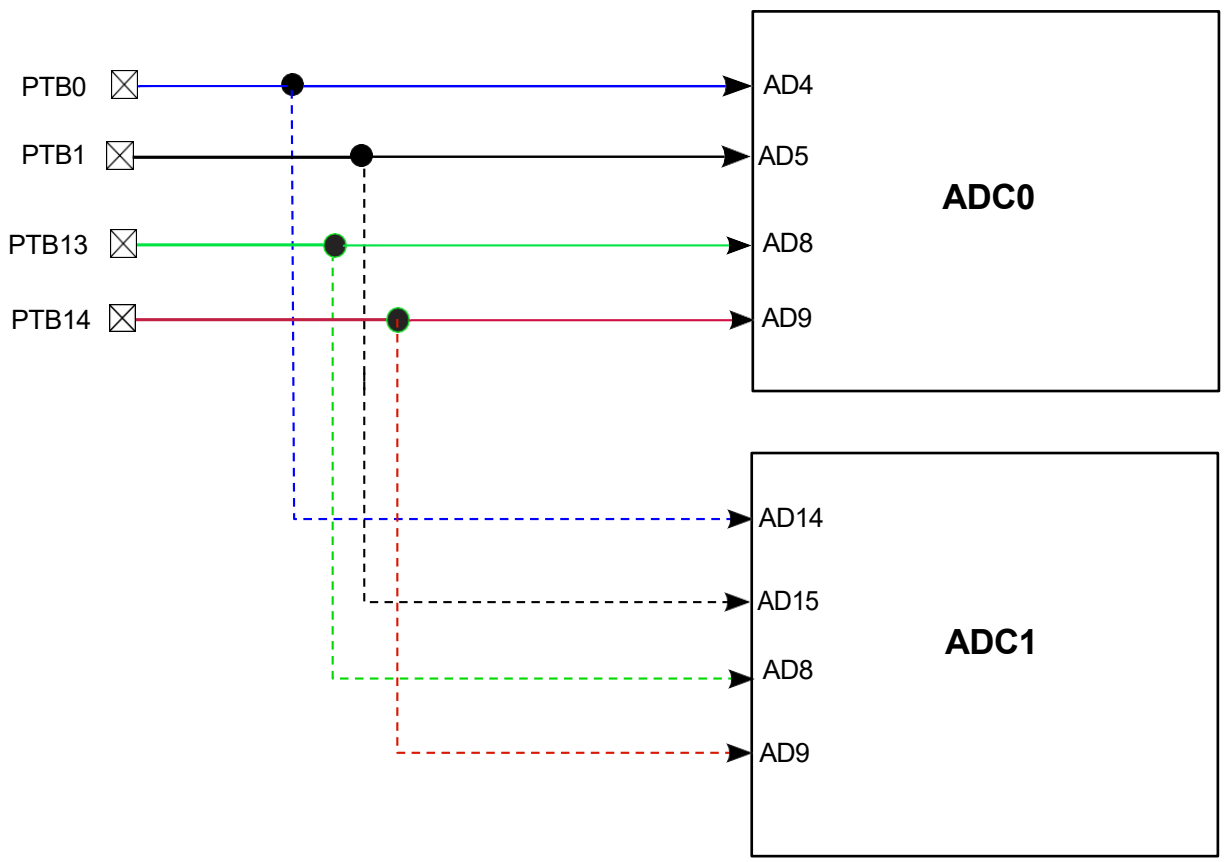



**图38-1.ADC0和ADC1的硬件交错通道集成**

## 38.5 ADC内部电源监控

内部电源在ADC0内部通道0上进行监测（通过选择ADC0_SC1n[ADCH]为010101b进行配置）。请参考SIM_CHIPCTL[ADC_SUPPLY]和SIM_CHIPCTL[ADC_SUPPLYEN]位。

## 38.6 ADC参考选项

该ADC支持以下参考文献。

- VREFH/VREFL - 作为主要参考选项连接
- VALTH/VREFL - 作为一个备用参考选项连接起来

ADCx_SC2[REFSEL]位选择ADC的电压参考源。更多细节请参考ADC章节的REFSEL描述。

在该器件上，VREFH和VREFL是专用引脚，VALTH与VDDA相同。在没有专用引脚的封装中，VREFH在封装中与VDDA相连，VREFL与VSSA相连。如果外部可用，正参考应连接到与VDDA相同的电位，或可由外部源驱动到最小参考电压高和VDDA电位之间的电平。VREFH决不能超过VDDA。VREFL应连接到与VSSA相同的电压电位。

## 38.7 ADC触发源

- 触发器通过PDB或TRGMUX连接，提供灵活的触发方案
- 2×PDB为2×ADC产生触发器和预触发器（ADC和PDB成对工作，为PDB0；ADC0和PDB1；ADC1），每个PDB通道将有多达8个预触发器用于ADC通道控制，这提供了一个自动触发方案，因此不需要CPU参与。
- 每个ADC至少有一个外部引脚（通过TRGMUX支持）。
  - 软件必须确定相对优先级
  - 在一次正在进行的转换完成后开始转换

- CMP输出、LPIT、RTC和LPTMR能够通过TRGMUX触发每个ADC。关于能够触发ADC的模块，见TRGMUX部分的TRGMUX模块互连性。
- LPIT最多支持4个预触发器，这些预触发器仅限于在每个ADC的ADHWTS0-
  ADHWTS3上使用。对于其余的外设，需要用软件配置来提供预触发器。参见SIM_ADCOPT[ADCxSWPRETRG]以配置软件预触发。

下面的规范和图示只是给出一个例子，以帮助理解ADC的触发方案。一般来说，ADC支持两种硬件触发方案。

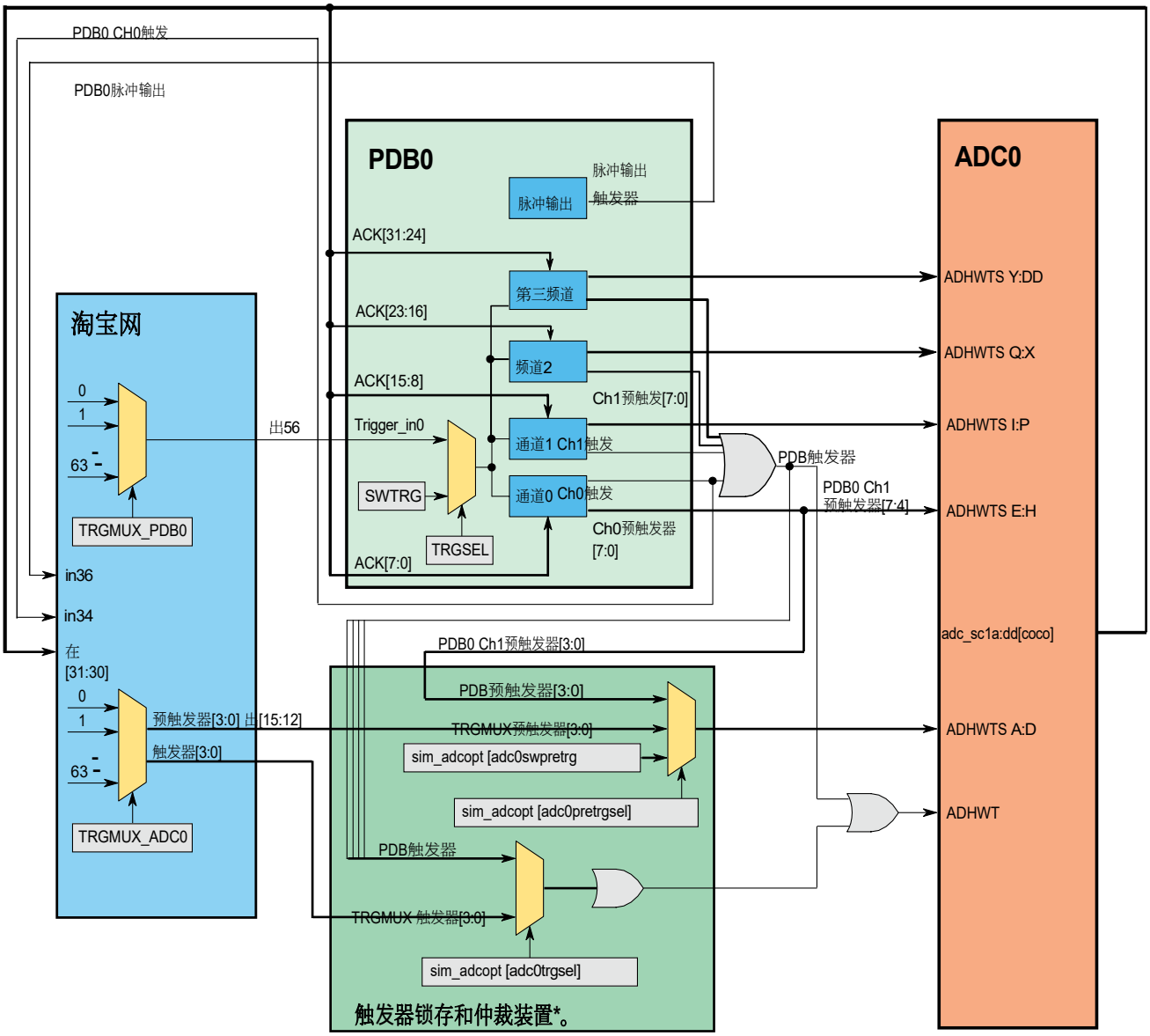
- 默认的硬件触发方案使用PDB来触发ADC（建议）。
- 另一个可选的硬件触发方案是使用TRGMUX。
- SIM_ADCOPT[ADCxTRGSEL]字段用于控制ADC的触发源/方案。
  - 当ADCxTRGSEL=0时，ADC预触发直接来自PDB。
  - 当ADCxTRGSEL=1时，ADC预触发器来自TRGMUX，如同LPIT的情况。

**注意事项**

对于TRGMUX触发，最多只能支持4个预触发器。

**注意事项**

对于SIM_ADCOPT[ADC0TRGSEL]或SIM_ADCOPT[ADC1TRGSEL]，每个PDB支持多达四个ADC通道，每个通道支持8个预触发器。如果有必要，ADC通道的触发器被OR'ed在一起，以支持多达32个预触发器。




* 该图块描述的是简化的原理图，没有显示详细的锁存。详见ADC章节的 "触发器锁存和仲裁 "一节。

**图38-2.ADC0_PDB0 ADC触发方案示例**

## 注意事项

在使用TRGMUX时，只有LPIT支持预触发器。对于其他外设，需要使用SIM_ADCOPT[ADCxSWPRETRG]配置软件预触发器。

## 38.7.1 PDB的触发方案

PDB触发方案是ADC的默认和建议的触发方法。一个ADC和一个PDB作为一对工作，在这个设备上的实现是。PDB0-ADC0，PDB1-ADC1。PDB0和PDB1的触发源可以分别通过TRGMUX_PDB0和TRGMUX_PDB1进行配置。这里我们以PDB0- ADC0为例来指定触发方案。

- 设置SIM_ADCOPT[ADCxTRGSEL]=0，PDB0通道0被选为ADC触发源。
- PDB0预触发器将直接连接到ADC0的ADHWTS端口来控制通道。
- ADC0的COCO信号直接反馈到PDB0，以停用PDB锁定状态。

下图说明了使用PDB进行ADC触发的典型情况：情况1。

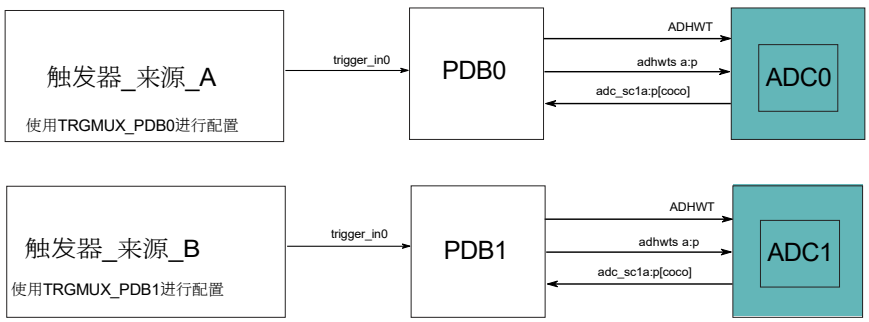



**图38-3.用独立源触发的ADC**

案例2:

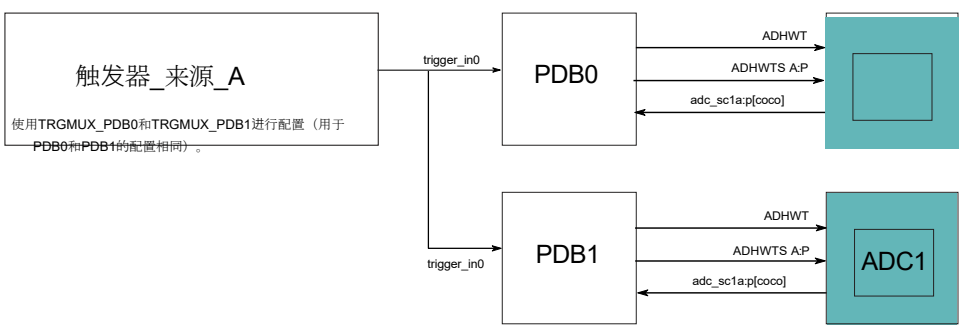



**图38-4.同源的ADC触发**

### 38.7.2 TRGMUX触发方案

TRGMUX支持许多触发源。这里我们以LPIT为典型例子，但触发源也可以是上面提到的其他的。LPIT最多支持4个通道，每个通道有一个触发器和一个预触发器。

- 设置SIM_ADCOPT[ADCxTRGSEL]=1。TRGMUX输出被选为ADC触发源。
- 配置TRGMUX以选择LPIT触发器作为ADC触发器和预触发源。
- TRGMUX只支持每个ADC最多4个预触发器（预触发0-预触发3；其他预触发器不能与TRGMUX一起使用）。
- 在这种情况下，ADC COCO是不需要的。软件必须适应每个ADC转换之间的中断时间。
- 使用TRGMUX可以使一个LPIT被用来同时触发两个ADC。这是使用TRGMUX触发而不是PDB触发的好处之一。

**注意事项**

对于PDB和LPIT以外的触发源，需要用软件来提供ADC预触发。

## 38.8 触发器选择

可以选择触发器使能和触发器的任何组合；它们是相互独立的。但在选择之后，不能即时改变。改变触发源和使能源有明确的步骤。改变触发源可以通过以下两种方式中的任何一种随时进行。

方式1

1. 停止当前的触发生成单元。
2. 等待一个时间段，相当于ADC工作时钟的2.5个周期和ADC主机接口时钟的1.5个周期的总和，以便有时间锁住最后一次触发，如果有的话
3. 轮询ADC_SC2[TRGSTLAT]的状态为0，在完成了所有的转换查询后，它将归为0。
4. 改变所需来源的选择，然后
5. 启动新的触发生成单元

如果需要立即切换

1. 停止当前的触发器生成单元

2. 通过设置ADC_CFG1[CLRLTRG]来刷新触发器处理块的所有排队的触发器，这将刷新所有排队的触发器，如果有的话，除了ADC正在处理的那一个。
3. 等待一个时间段，相当于ADC工作时钟的2.5个周期的总和
4. 等待ADC_SC2[TRGSTLAT]的状态变为全0
5. 改变所需来源的选择，然后
6. 启动新的触发器生成单元

**注意事项**

如果不遵守上述限制，那么这个过程所遗漏的触发器可能不会被报告。

## 38.9 触发器锁存和仲裁

四个较低的触发器有被锁定的能力。无论选择哪种信号源组合，来自该信号源的触发请求都会被锁存和处理。潜伏的触发请求每次都会提交给ADC。只有当ADC完成对当前触发请求的处理后，才会给出下一个请求。因此，在任何时候都只有一个请求被处理，所有其他请求都被锁住。如果一个触发器请求再次出现在0-3个使能/触发器中的任何一个，而它正在被ADC处理或已经在触发器处理程序中被锁定，那么新的请求将被忽略，并且在寄存器ADC_SC2[TRGSTERR]中为该触发器指出一个错误。

来自选定源的触发请求之间的关系可以随机为（a）单次，或（b）同时。然而，它们将在轮流的基础上得到服务。例如，在第P个请求被处理后[0 < P <(N-1)]，搜索将以（P+1），然后是（P+2），直到找到下一个被锁定的请求。在第(N-1)次触发后，这种搜索会翻转到0，并继续进行，直到达到P。

为了使用多路触发器与ADC一起工作，你需要在不同情况下遵循以下编程顺序。

| | |
|---|---|
| 情况1：初始化/开始转换。 | a.初始化/重新编程（在中间启动的情况下）SAR ADC，以实现所需的操作（详见ADC部分）。 |
| | b.选择触发器处理块的触发源（其配置在芯片的一个单独模块中，请参考SIM章节中的SIM_ADCOPT寄存器）。 |
| | c.配置并启动触发器生成模块（PDB或TRGMUX）。 |
| 案例2：改变触发源/复用器控制。 | a.停止触发器生成模块（PDB或TRGMUX）。 |
| | b.做下列选项之一。 |

*表继续在下一页...*

| | |
|---|---|
| | 选择1。<br>1. 等待的时间相当于ADC工作时钟的2.5个周期和ADC主机接口时钟的1.5个周期的总和，以便有时间锁住最后一次触发，如果有的话。<br>2. 等待潜伏的触发器被处理，触发器处理程序空闲（轮询ADC_SC2[TRGSTLAT]的状态，所有0）。<br><br>选项2。<br>1. 在触发器处理程序块中刷新潜伏的触发器请求（向ADC_CFG1[CLRLTRG]写一个1b）。<br>2. 等待一个时间段，相当于ADC工作时钟的2.5个周期的总和<br>3. 等待触发器处理程序的空闲状态（轮询ADC_SC2[TRGSTLAT]的状态，全部为0）。 |
| | c.重新选择触发器处理块的触发源（其配置在芯片上的一个单独模块中，见芯片的具体章节）。 |
| | d.配置并启动触发器生成模块（PDB或TRGMUX）。 |
| 案例3：停止转换 | a.停止触发器生成模块（PDB或TRGMUX）。 |
| | b.等待的时间相当于ADC工作时钟的2.5个周期加上ADC主机接口时钟的1.5个周期的总和，以便有时间锁住最后一个触发器（如果有）。 |
| | c.等待潜伏的触发器被处理并且触发器处理程序空闲（轮询ADC_SC2[TRGSTLAT]的状态为所有0）。 |
| | d.停止SAR ADC，并在必要的处理后清除中断（详见ADC部分）。 |
| 案例4：在连续转换的情况下改变触发源/多路复用器控制 | a.停止触发器生成模块（PDB或TRGMUX）。 |
| | b.读取ADC的任何配置寄存器。 |
| | c.将上面的读值写在同一个寄存器中。 |
| | d.这种写法将中止正在进行的连续转换。 |
| | e.重新选择触发器处理程序块的触发源（其配置在芯片的一个单独模块中，请参考SIM章节中的SIM_ADCOPT寄存器）。 |
| | f.配置并启动触发器生成模块（PDB或TRIGMUX）。 |
| 情况5：清除错误状态（仅适用于TRGMUX情况）。 | a.读取错误状态（ADC_SC2[TRGSTERR]）。 |
| | b.如果上述任何一个寄存器位为1，则停止触发生成模块（TRGMUX）。 |
| | c.等待的时间相当于ADC工作时钟的2.5个周期加上ADC主机接口时钟的1.5个周期的总和，以便有时间锁定最后一个触发器（如果有）。 |
| | d.等待触发器处理程序的空闲状态（轮询 |

| |
|---|
| ADC_SC2[TRGSTLAT]的状态为所有0）。 |
| e.通过写1来清除上述寄存器的位。 |
| f.等待潜伏的触发器被处理，并且触发器处理程序处于空闲状态（轮询ADC_SC2[TRGSTLAT]的状态为所有0）。 |

*表继续在下一页...*

| | |
|---|---|
| | g.重新选择触发器处理程序块的触发源（其配置在芯片的一个单独模块中，请参考SIM章节中的SIM_ADCOPT寄存器）。 |
| | h.配置并启动触发器生成模块（PDB或TRGMUX）。 |

**注意事项**

在PDB通过触发处理程序进行触发时，不应通过向ADC_CFG1[CLRLTRG]写1来清除潜伏状态，因为PDB在闭环中与来自ADC的COCO标志一起工作，每次提供一个触发。在触发器处理程序内处理时（在启动到ADC之前），清除锁存状态将停止其未来的转换，并且将没有COCO标志。在这种情况下，PDB可能进入一个不确定的状态。

## 38.10 ADC的低功耗模式

ADC将在STOP2模式下可用。

## 38.11 ADC触发器概念 - 使用案例

FlexTimer模块（FTM）支持计数器初始触发和通道匹配触发，这可以作为PDB的触发输入。然后，PDB可以用来触发其他模块，如ADC。PDB模块中的每个ADC通道最多支持8个预触发器，它们可以作为ADC硬件通道选择，在实际触发之前对ADC块进行预处理。在预触发之后，ADC触发器启动了ADC转换。下图所示的波形说明了PDB到ADC的预触发和触发输出。当PDB预触发和触发输出启动ADC转换时，一个与相应预触发器相关的内部锁被激活。当接收到ADC的COCO信号时，这个锁就变得不活跃了。

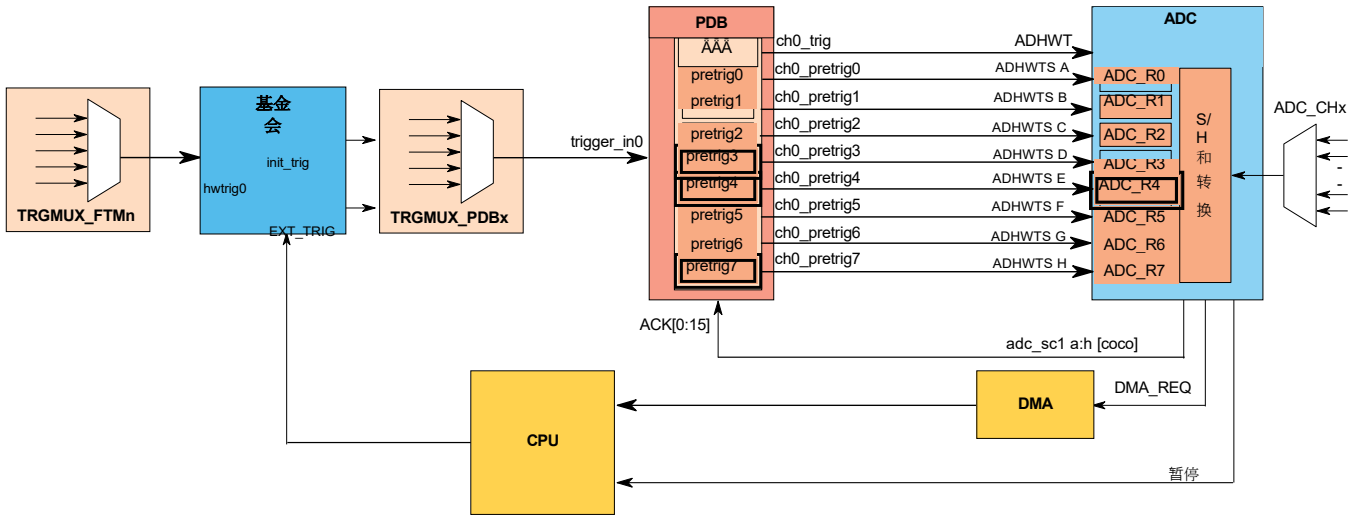



**图38-5.PWM负载诊断 - ADC触发概念（框图）**

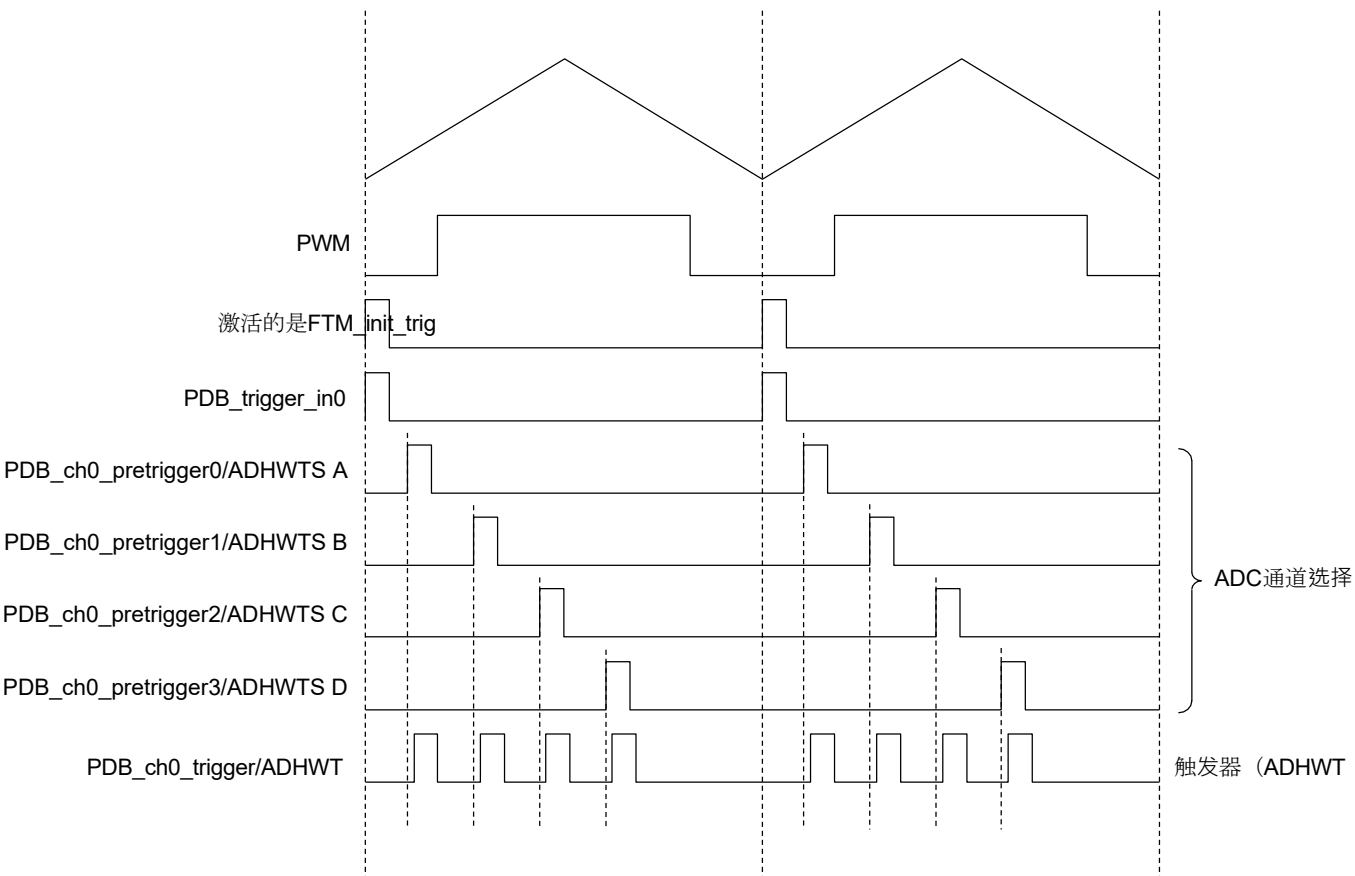



**图38-6.例子。PWM 负载诊断 - ADC 触发概念 1 (计时)**

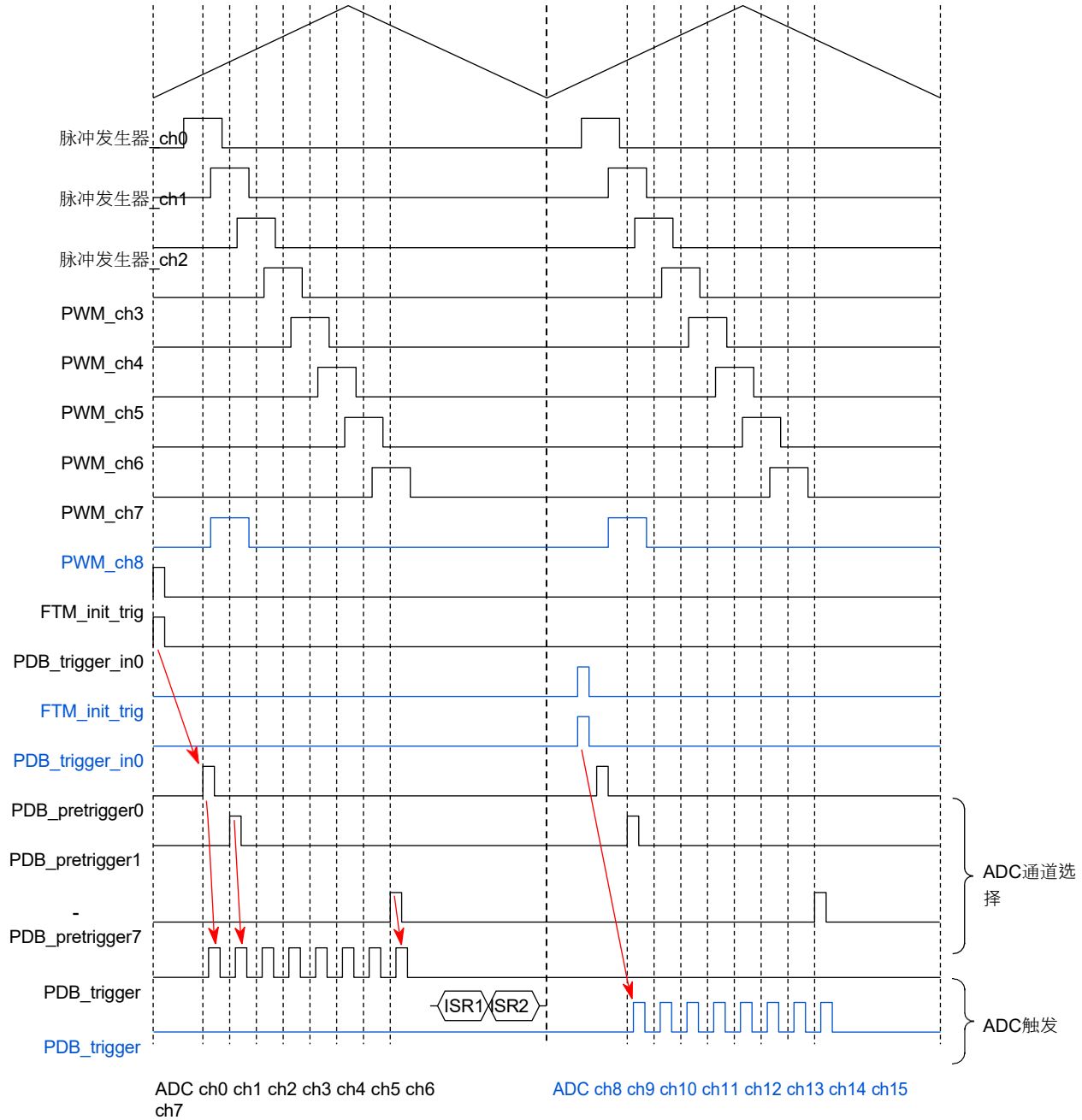



**图38-7.例子。PWM 负载诊断 - ADC 触发概念 2 (计时)**

## 38.12 ADC校准方案

在POR之后，在闪存可用之后，必须通过使用ADC_SC3控制寄存器中的CAL位来执行ADC校准。

在ADC被配置和校准后，只要电源保持不变，就不需要进一步校准。因此，在随后从电源模式STOP2退出时，不需要重新校准。由于在所有的器件电源模式（STOP2
RUN和HSRUN）下，电源是连续的，所有的ADC规格都按照数据表的要求保持。

# 第39章
# 模拟数字转换器（ADC）

## 39.1 芯片特定的ADC信息

参见ADC配置章节。

## 39.2 简介

12位模数转换器（ADC）是一个逐次逼近的ADC，设计用于在集成微控制器的片上系统中运行。

**注意事项**

关于芯片的具体操作模式，请参见设备的电源管理信息。

### 39.2.1 特点

以下是ADC模块的特点。

- 具有高达12位分辨率的线性逐次逼近算法
- 多达32个单端外部模拟输入
- 单端12位、10位和8位输出模式
- 以右对齐的无符号格式输出，用于单端
- 单一或连续转换模式
- 单次转换后自动返回到空闲状态
- 可配置的采样时间和转换速度/功率
- 转换完成/硬件平均完成标志和中断
- 可从最多四个来源选择输入时钟
- 在低功率模式下运行，噪音更低
- 可选择的硬件转换触发器与硬件通道选择

- 对于小于、大于或等于、在范围内或超出范围的可编程值，自动与中断进行比较
- 硬件平均功能
- 可选择电压基准：外部或备用
- 自我校准模式

## 39.2.2 框图

下图是ADC模块框图。

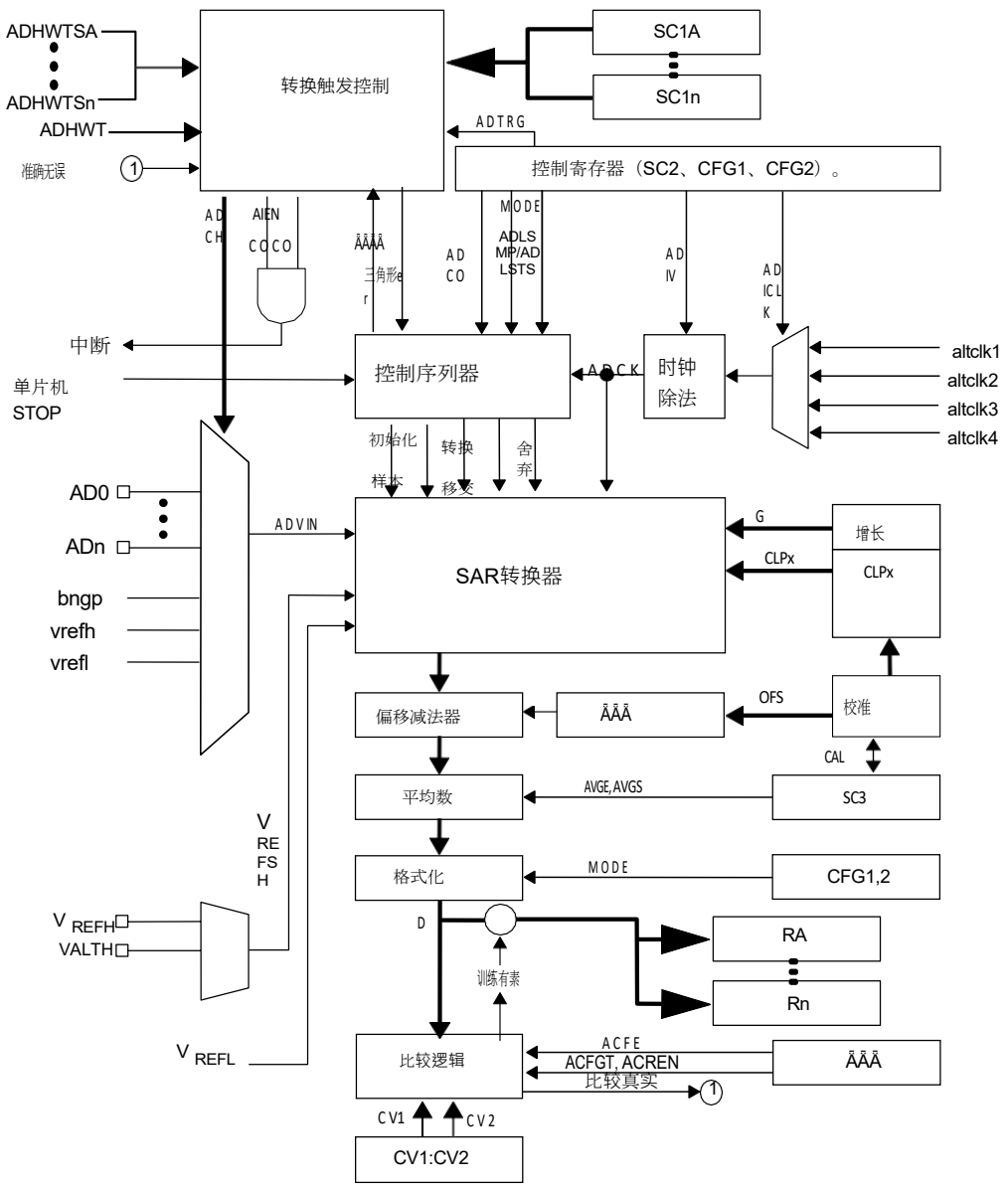



**图39-1.ADC框图**

## 39.3 ADC信号描述

每个ADC模块最多支持32个单端输入。

ADC还需要四个电源/基准/地线连接。

**注意事项**

关于该设备支持的通道数，请参见芯片特定的ADC信息。

ADC不产生任何输出信号。

**表39-1.ADC输入信号描述**

| 信号 | 描述 |
|---|---|
| ADn | 单端模拟通道输入 |
| 谷雨 | 电压基准选择高 |
| ĀĀĀ | 电压基准选择低 |
| VDD<br>A | 模拟电源 |
| VSS<br>A | 模拟地 |

### 39.3.1 模拟电源 ($V_{DDA}$)

ADC模拟部分使用$V_{DDA}$作为其电源连接。在某些封装中，$V_{DDA}$内部连接到$V_{DD}$。如果外部可用，则将$V_{DDA}$引脚连接到与$V_{DD}$相同的电压电位。可能需要进行外部滤波，以确保干净的$V_{DDA}$，从而获得良好的效果。

### 39.3.2 模拟地 ($V_{SSA}$)

ADC模拟部分使用$V_{SSA}$作为其接地连接。声明在某些封装中，$V_{SSA}$内部连接到$V_{SS}$。声明如果外部可用，将$V_{SSA}$引脚连接到与$V_{SS}$相同的电压电位。

### 39.3.3 电压基准选择

$V_{REFSH}$ 和$V_{REFSL}$ 是ADC模块的高和低参考电压。

通过将$V_{REFSH}$ 配置为$V_{REFH}$ 或$V_{ALTH}$， 可以将ADC配置为接受$V_{REFSH}$ 和$V_{REFSL}$ 的电压参考对中的一个。每个参考对包含一个正向参考，必须在最小参考电压高和$V_{DDA}$ 之间，以及一个接地参考，必须与$V_{SSA}$ 处于同一电位。这两个参考对是外部（$V_{REFH}$ 和$V_{REFL}$ ）交替（$V_{ALTLH}$ 和$V_{REFL}$ ）。这些电压基准是通过配置SC2[REFSEL]来选择的。备用的电压基准，$V_{ALTH}$ 可以选择额外的外部引脚或内部源，这取决于MCU的配置。请参见该MCU特有的电压基准的芯片配置信息。

在某些封装中，$V_{REFH}$ 在内部连接到$V_{DDA}$， $V_{REFL}$ 连接到$V_{SSA}$ 。如果外部可用，正参考可以连接到与$V_{DDA}$ 相同的电位，或者由外部源驱动到最小$V_{REFH}$ 和$V_{DDA}$ 电位之间的水平。$V_{REFH}$ 决不能超过$V_{DDA}$ 。将接地基准连接到与$V_{SSA}$ 相同的电压电位。

### 39.3.4 模拟通道输入（ADx）

ADC模块最多支持32个模拟输入。通过SC1[ADCH]通道选择字段，选择一个模拟输入进行转换。

## 39.4 ADC寄存器描述

### 39.4.1 ADC内存图

ADC基础地址。4003_B000h

ADC_40027000基础地址。4002_7000h

| 偏移 | 注册 | 宽度<br>(单位：比特) | 访问 | 重置值 |
|---|---|---|---|---|
| 0h - 3Ch (别名) | ADC状态和控制寄存器1（SC1A - SC1P）。 | 32 | RW | 0000_003Fh |
| 40h | ADC配置寄存器1（CFG1）。 | 32 | RW | 0000_0000h |
| 44h | ADC配置寄存器2（CFG2） | 32 | RW | 0000_000Ch |
| 48h - 84h (别名) | ADC数据结果寄存器（RA - RP）。 | 32 | RO | 0000_0000h |

**ADC寄存器描述**

| | | | | |
|---|---|---|---|---|
| 88h - 8Ch | 比较值寄存器(CV1 - CV2) | 32 | RW | 0000_0000h |
| 90h | 状态和控制寄存器2（SC2） | 32 | RW | 0000_0000h |

*表格在下一页继续...*

| 偏移 | 注册 | 宽度 (单位：比特) | 访问 | 重置值 |
|---|---|---|---|---|
| 94h | 状态和控制寄存器3（SC3） | 32 | RW | 0000_0000h |
| 98h | 基准偏移寄存器（BASE_OFS）。 | 32 | RW | 0000_0040h |
| 9Ch | ADC偏移校正寄存器（OFS） | 32 | RW | 见描述。 |
| A0h | 用户偏移校正寄存器（USR_OFS）。 | 32 | RW | 0000_0000h |
| A4h | ADC X偏移校正寄存器（XOFS） | 32 | RW | 0000_0030h |
| A8h | ADC Y偏移校正寄存器（YOFS）。 | 32 | RW | 0000_0037h |
| ACh | ADC增益寄存器（G） | 32 | RW | 见描述。 |
| B0h | ADC用户增益寄存器（UG） | 32 | RW | 0000_0004h |
| B4h | ADC通用校准值寄存器S（CLPS）。 | 32 | RW | 见描述。 |
| B8h | ADC Plus-Side通用校准值寄存器3（CLP3）。 | 32 | RW | 见描述。 |
| BCh | ADC Plus-Side通用校准值寄存器2（CLP2）。 | 32 | RW | 见描述。 |
| C0h | ADC Plus-Side通用校准值寄存器1（CLP1）。 | 32 | RW | 见描述。 |
| C4h | ADC Plus-Side通用校准值寄存器0（CLP0）。 | 32 | RW | 见描述。 |
| C8h | ADC Plus-Side通用校准值寄存器X（CLPX）。 | 32 | RW | 见描述。 |
| CCh | ADC Plus-Side通用校准值寄存器9（CLP9）。 | 32 | RW | 见描述。 |
| D0h | ADC通用校准偏移值寄存器S（CLPS_OFS）。 | 32 | RW | 0000_0000h |
| D4h | ADC Plus-Side通用校准偏移值寄存器3（CLP3_ OFS）。 | 32 | RW | 0000_0000h |
| D8h | ADC Plus-Side通用校准偏移值寄存器2（CLP2_ OFS）。 | 32 | RW | 0000_0000h |
| DCh | ADC Plus-Side通用校准偏移值寄存器1（CLP1_ OFS）。 | 32 | RW | 0000_0000h |
| E0h | ADC Plus-Side通用校准偏移值寄存器0（CLP0_ OFS）。 | 32 | RW | 0000_0000h |
| E4h | ADC Plus-Side通用校准偏移值寄存器X（CLPX_ OFS）。 | 32 | RW | 0000_0440h |
| E8h | ADC Plus-Side通用校准偏移值寄存器9（CLP9_ OFS）。 | 32 | RW | 0000_0240h |
| 108h - 144h (别名) | ADC状态和控制寄存器1（aSC1A - aSC1P）。 | 32 | RW | 0000_003Fh |
| 148h | ADC状态和控制寄存器1（SC1Q）。 | 32 | RW | 0000_003Fh |
| 14Ch | ADC状态和控制寄存器1（SC1R）。 | 32 | RW | 0000_003Fh |
| 150h | ADC状态和控制寄存器1（SC1S）。 | 32 | RW | 0000_003Fh |
| 154h | ADC状态和控制寄存器1（SC1T）。 | 32 | RW | 0000_003Fh |

| | | | | |
|---|---|---|---|---|
| 158h | ADC状态和控制寄存器1（SC1U）。 | 32 | RW | 0000_003Fh |

*表继续在下一页...*

| 偏移 | 注册 | 宽度<br>(单位：比特) | 访问 | 重置值 |
|---|---|---|---|---|
| 15Ch | ADC状态和控制寄存器1（SC1V）。 | 32 | RW | 0000_003Fh |
| 160h | ADC状态和控制寄存器1（SC1W）。 | 32 | RW | 0000_003Fh |
| 164h | ADC状态和控制寄存器1（SC1X）。 | 32 | RW | 0000_003Fh |
| 168h | ADC状态和控制寄存器1（SC1Y）。 | 32 | RW | 0000_003Fh |
| 16Ch | ADC状态和控制寄存器1（SC1Z）。 | 32 | RW | 0000_003Fh |
| 170h | ADC状态和控制寄存器1（SC1AA）。 | 32 | RW | 0000_003Fh |
| 174h | ADC状态和控制寄存器1（SC1AB）。 | 32 | RW | 0000_003Fh |
| 178h | ADC状态和控制寄存器1（SC1AC）。 | 32 | RW | 0000_003Fh |
| 17Ch | ADC状态和控制寄存器1（SC1AD）。 | 32 | RW | 0000_003Fh |
| 180h | ADC状态和控制寄存器1（SC1AE）。 | 32 | RW | 0000_003Fh |
| 184h | ADC状态和控制寄存器1（SC1AF）。 | 32 | RW | 0000_003Fh |
| 188h - 1C4h (别名) | ADC数据结果寄存器(ARA - aRP) | 32 | RO | 0000_0000h |
| 1C8h | ADC数据结果寄存器（RQ） | 32 | RO | 0000_0000h |
| 1CCh | ADC数据结果寄存器（RR） | 32 | RO | 0000_0000h |
| 1D0h | ADC数据结果寄存器（RS） | 32 | RO | 0000_0000h |
| 1D4h | ADC数据结果寄存器（RT） | 32 | RO | 0000_0000h |
| 1D8h | ADC数据结果寄存器（RU） | 32 | RO | 0000_0000h |
| 1DCh | ADC数据结果寄存器（RV） | 32 | RO | 0000_0000h |
| 1E0h | ADC数据结果寄存器（RW） | 32 | RO | 0000_0000h |
| 1E4h | ADC数据结果寄存器（RX） | 32 | RO | 0000_0000h |
| 1E8h | ADC数据结果寄存器（RY） | 32 | RO | 0000_0000h |
| 1ECh | ADC数据结果寄存器（RZ） | 32 | RO | 0000_0000h |
| 1F0h | ADC数据结果寄存器（RAA） | 32 | RO | 0000_0000h |
| 1F4h | ADC数据结果寄存器（RAB） | 32 | RO | 0000_0000h |
| 1F8h | ADC数据结果寄存器（RAC） | 32 | RO | 0000_0000h |
| 1FCh | ADC数据结果寄存器（RAD） | 32 | RO | 0000_0000h |
| 200h | ADC数据结果寄存器（RAE） | 32 | RO | 0000_0000h |
| 204h | ADC数据结果寄存器（RAF） | 32 | RO | 0000_0000h |

## 39.4.2 ADC状态和控制寄存器1（SC1A - aSC1P）。

### 39.4.2.1 偏移

对于a=A到P（0到15）。

| 注册 | 偏移 |
| --- | --- |
| SC1a | 0h + (a × 4h) |
| aSC1a（SC1a的别名）。 | 108h + (a × 4h) |

### 39.4.2.2 职能

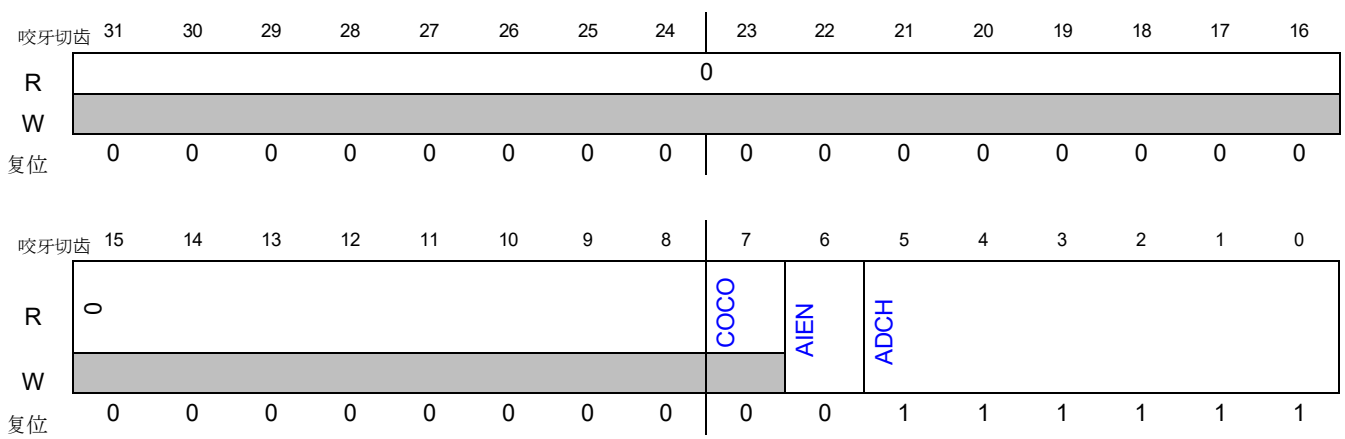
SC1A同时用于软件和硬件触发的工作模式。

在任何一个时间点，只有一个SC1n寄存器在主动控制ADC的顺序转换。允许在SC1n主动控制转换时更新SC1A，反之亦然，因为任何SC1n寄存器都是针对该MCU的。

当SC1A主动控制一个转换时，向SC1A的ADCH字段写111111，会中止当前的转换。在软件触发模式下（当SC2[ADTRG]=0时），对SC1A的写入会启动一个新的转换。这对SC1A[ADCH]的所有值都有效，除了111111（模块禁用）。

当特定的SC1n寄存器正在积极控制转换时，写入任何SC1n寄存器都会中止当前的转换。SC1B-
SC1n寄存器都不用于软件触发操作，因此写到SC1BSC1n寄存器不会启动新的转换。

### 39.4.2.3 图示



## 39.4.2.4 领域

| 场地 | 职能 |
|---|---|
| 31-8<br>- | 保留 |
| 7<br>COCO | 转换完成标志<br>这是一个只读字段，在每次完成转换时，当出现以下一个或多个情况时，就会设置这个字段是真的。<br>• 比较功能被禁用<br>• SC2[ACFE]=0，硬件平均功能被禁用。<br>• SC3[AVGE]=0<br>如果比较结果为真，那么在转换完成后，如果一个或多个以下是真实的。<br>• 启用了比较功能<br>• SC2[ACFE]=1<br>在完成选定的转换数量（由AVGS决定）后，如果有一个或多个转换，则设置COCO。下列各项中，有一项是真的。<br>• 启用了硬件平均功能<br>• SC3[AVGE]=1<br>SC1A中的COCO也在校准序列完成时被设置。<br>当以下情况之一为真时，COCO被清除。<br>• 各个SC1n寄存器被写入<br>• 各个Rn寄存器被读取<br>0b - 转换未完成。<br>1b - 转换完成。 |
| 6<br>AIEN | 中断启用<br>启用转换完成中断。当COCO被设置，而相应的AIEN为高电平时，一个中断被断言。<br>0b - 转换完成中断被禁用。1b - 转换完成中断被启用。 |
| 5-0<br>ADCH | 输入通道选择<br>选择其中一个输入通道。<br>**注意：**位域设置描述中的一些输入通道选项可能对你的芯片不可用。关于你的设备的实际ADC通道分配，请参见芯片的具体信息。<br>当通道位全部设置时（即ADCH设置为全部1），逐次逼近转换器子系统被关闭。这一特性允许明确地禁用ADC，并将输入通道与所有来源隔离。以这种方式终止连续转换可以防止执行额外的单一转换。在不启用连续转换时，没有必要将ADCH设置为所有1来使ADC处于低功耗状态，因为当转换完成时，模块会自动进入低功耗状态。<br>000000b - 外部通道0被选择为输入。000001b - 外部通道1被选为输入。000010b - 外部通道2被选为输入。000011b - 外部通道3被选为输入。000100b - 外部通道4被选择为输入。000101b - 外部通道5被选择为输入。000110b - 外部通道6被选 |

| | |
|---|---|
| | 择为输入。 |

| 场地 | 职能 |
| --- | --- |
| | 000111b - 外部通道7被选择为输入。001000b - 外部通道8被选为输入。001001b - 外部通道9被选为输入。001010b - 外部通道10被选择为输入。001011b - 外部通道11被选择为输入。001100b - 外部通道12被选择为输入。001101b - 外部通道13被选择为输入。001110b - 外部通道14被选择为输入。001111b - 外部通道15被选择为输入。010000b - 保留<br>010001b - 保留<br>010010b - 保留<br>010011b - 保留<br>010100b - 保留<br>010101b - 内部通道0被选为输入。010110b - 内部通道1被选为输入。010111b - 内部通道2被选择为输入。011000b - 保留<br>011001b - Reserved<br>011010b - Reserved<br>011011b - Band Gap<br>011100b - 内部通道3被选为输入。<br>011101b - $V_{REFSH}$ 被选择为输入。选择的电压基准由SC2[REFSEL]决定。011110b - $V_{REFSL}$ 被选为输入。选择的电压基准由SC2[REFSEL]决定。011111b - 保留<br>100000b - 外部通道16被选为输入。100001b - 外部通道17被选为输入。100010b - 外部通道18被选为输入。100011b - 外部通道19被选为输入。100100b - 外部通道20被选择为输入。100101b - 外部通道21被选择为输入。100110b - 外部通道22被选择为输入。100111b - 外部通道23被选择为输入。101000b - 外部通道24被选择为输入。101001b - 外部通道25被选为输入。101010b - 外部通道26被选为输入。101011b - 外部通道27被选为输入。101100b - 外部通道28被选为输入。101101b - 外部通道29被选为输入。101110b - 外部通道30被选为输入。101111b - 外部通道31被选择为输入。<br>110000b - 保留<br>110001b - Reserved<br>110010b - Reserved<br>110011b - Reserved<br>110100b - Reserved<br>110101b - Reserved<br>110110b - Reserved<br>110111b - Reserved<br>111000b - Reserved<br>111001b - Reserved<br>111010b - Reserved<br>111011b - Reserved<br>111100b - Reserved<br>111101b - Reserved<br>111110b - Reserved |

| | |
|---|---|
| | 111111b - 模块已禁用 |

## 39.4.3 ADC配置寄存器1（CFG1）。

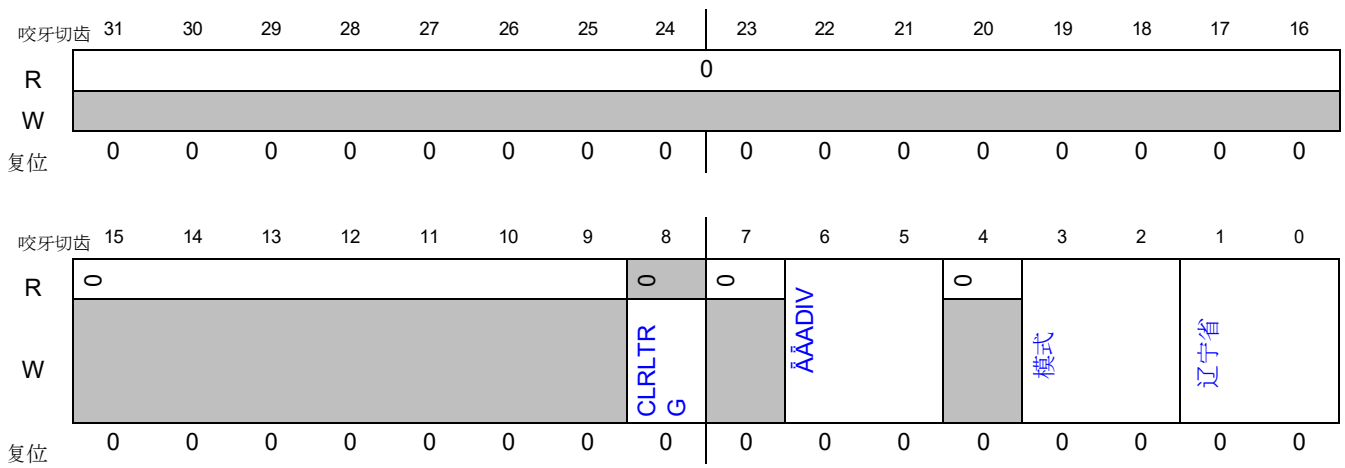
### 39.4.3.1 偏移

| 注册 | 偏移 |
|---|---|
| CFG1 | 40h |

### 39.4.3.2 职能

配置寄存器1（CFG1）选择工作模式、时钟源、时钟分频。

### 39.4.3.3 图示




### 39.4.3.4 领域

| 场地 | 职能 |
|---|---|
| 31-9<br>- | 保留 |
| 8<br>辽宁省 | 清除触发器处理程序块中的锁存器触发器<br>向这个字段写1，可以清除触发器处理程序中所有被锁定的触发器，除了正在处理的一个。写入0则没有任 |

| | |
|---|---|
| | 何作用。这是一个只写一个的位，可以立即自我清除。 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | **注意：**这个字段并不是在每个实例中都支持。<br><br>**支持的**：ADC_CFG1<br>**不支持**：ADC_CFG1 |
| 7<br>- | 保留 |
| 6-5<br>ÅÅADIV | 时钟分频选择<br>选择ADC用于产生内部时钟ADCK的除法比率。<br>00b - 分割率为1，时钟频率为输入时钟。01b - 分割率为2，时钟速率为（输入时钟）/2。10b - 分割率为4，时钟速率为（输入时钟）/4。11b - 分割率为8，时钟速率为（输入时钟）/8。 |
| 4<br>- | 保留 |
| 3-2<br>模式 | 转换模式选择<br>选择ADC的分辨率。00b - 8位转换。01b - 12位转换。10b - 10位转换。11b - 保留 |
| 1-0<br>辽宁省 | 输入时钟选择<br>选择输入时钟源以产生内部时钟ADCK。有关支持哪些备用时钟的细节，请参见你的设备的时钟分配/时钟章节。<br>00b - 备用时钟1（ALTCLK1）<br>01b - 备用时钟2（ALTCLK2）<br>10b - 备用时钟3（ALTCLK3）<br>11b - 备用时钟4（ALTCLK4） |

## 39.4.4 ADC配置寄存器2（CFG2）

### 39.4.4.1 偏移

| 注册 | 偏移 |
| --- | --- |
| CFG2 | 44h |

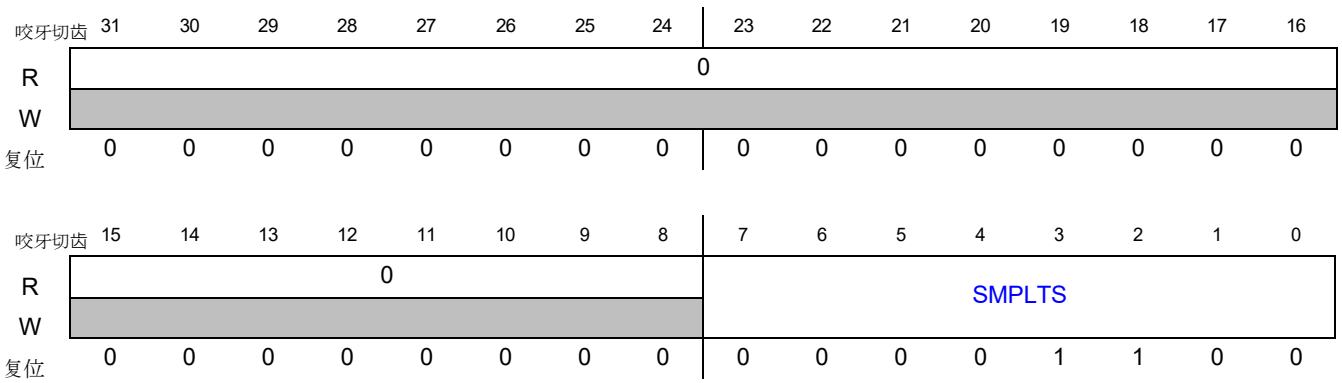
#### 39.4.4.2 职能

配置寄存器2（CFG2）在长采样模式下选择长采样时间的长短。

**注意事项**

该寄存器不支持写0。

### 39.4.4.3 图示




### 39.4.4.4 领域

| 场地 | 职能 |
|---|---|
| 31-8<br>- | 保留 |
| 7-0<br>SMPLTS | 采样时间选择<br>选择2至256个ADCK时钟周期的采样时间。写到这个寄存器字段的值是所需的采样时间减去1。不支持1的采样时间。允许对较高阻抗的输入进行精确采样，或对较低阻抗的输入最大限度地提高转换速度。如果不需要高转换率，在启用连续转换时，较长的采样时间也可用于降低整体功耗。 |

## 39.4.5 ADC数据结果寄存器（RA - aRP）。

### 39.4.5.1 偏移

对于a=A到P（0到15）。

| 注册 | 偏移 |
|---|---|
| 拉 | 48h + (a × 4h) |

| | |
|---|---|
| 阿拉(Ra)的别名 | 188h + (a × 4h) |

### 39.4.5.2 职能

数据结果寄存器（Rn）包含由相应的状态和通道控制寄存器（SC1A:SC1n）选择的通道的ADC转换结果。对于每个状态和通道控制寄存器，都有一个相应的数据结果寄存器。

Rn中未使用的位被清空。

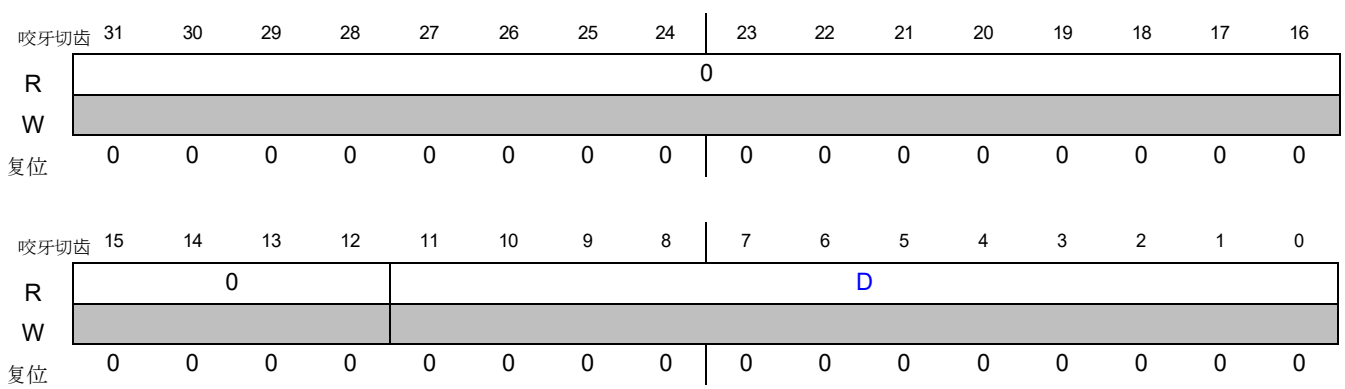
下表描述了数据结果寄存器在不同操作模式下的行为。

**表39-2.数据结果寄存器描述**

| 转换模式 | D11 | D10 | D9 | D8 | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | 格式 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 12位单端 | D | | | | | | | | | | | | 无符号右对齐 |
| 10位单端 | 0 | | D | | | | | | | | | | |
| 8位单端 | 0 | | | | D | | | | | | | | |

D：数据。数据结果寄存器是只读的；对这些寄存器的写入会产生一个传输错误。

### 39.4.5.3 图示




### 39.4.5.4 领域

| 场地 | 职能 |
|---|---|
| 31-12<br>- | 保留 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| 11-0<br>D | 数据结果 |

## 39.4.6 比较值寄存器(CV1 - CV2)

### 39.4.6.1 偏移

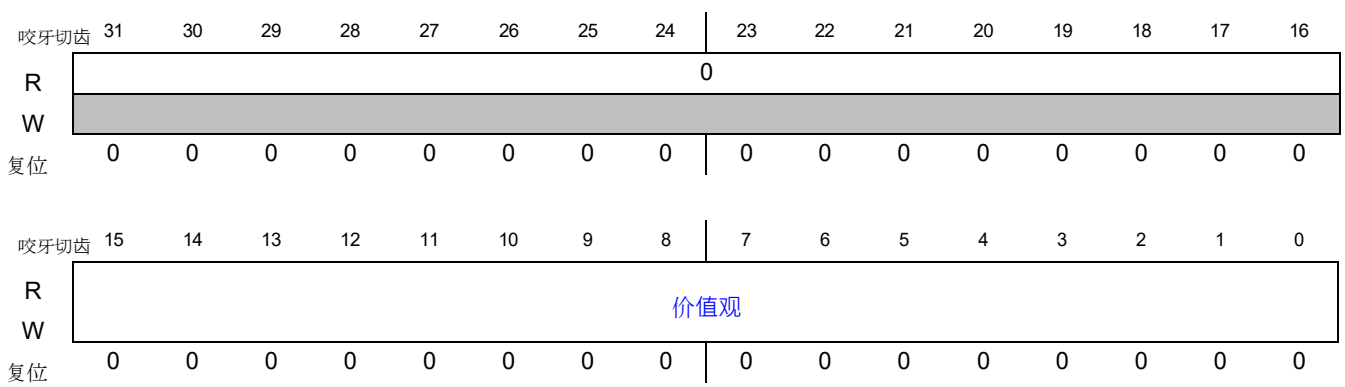
对于a=1到2。

| 注册 | 偏移 |
| --- | --- |
| ÅÅÅ | 84h + (a × 4h) |

### 39.4.6.2 职能

比较值寄存器（CV1和CV2）包含一个比较值，当比较功能启用时，即SC2[ACFE]=1，用于比较转换结果。该寄存器的格式与Rn寄存器的格式相同。因此，比较功能只使用与ADC工作模式有关的CVn字段。只有当比较范围功能被启用时，即SC2[ACREN]=1，才会使用CV2。

### 39.4.6.3 图示



### 39.4.6.4 领域

| 场地 | 职能 |
|---|---|
| 31-16<br>- | 保留 |
| 15-0<br>价值观 | 比较价值。 |

## 39.4.7 状态和控制寄存器2（SC2）

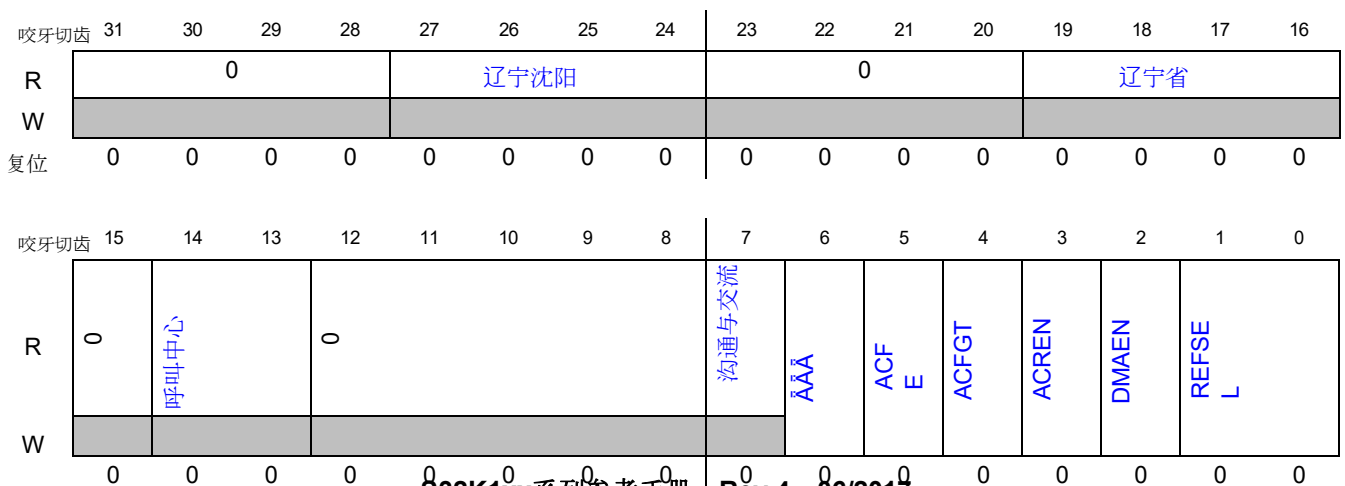
### 39.4.7.1 偏移

| 注册 | 偏移 |
|---|---|
| ÃÃÃ | 90h |

### 39.4.7.2 职能

状态和控制寄存器2（SC2）包含ADC模块的转换激活、硬件/软件触发选择、比较功能和电压基准选择。

### 39.4.7.3 图示



## 39.4.7.4 领域

<table>
<tr><th>场地</th><th>职能</th></tr>
<tr><td>31-28<br>-</td><td>保留</td></tr>
<tr><td>27-24<br>辽宁沈阳</td><td>多重触发请求中的错误<br>这些错误信号中的每一个都表示错过了来自某个源的复用硬件触发请求，在这种情况下，该请求已经被锁定或正在被服务。这个字段中的每个位都可以通过写1来清除，每个位都对应于一个单独的触发请求。<br>• 第24位对应的是触发请求0<br>• 第25位对应的是触发请求1<br>• 第26位对应的是触发请求2<br>• 第27位对应的是触发请求3<br>该字段中每个位的读值被单独解释如下。<br>注意：这个字段并不是在每个实例中都支持。<table><tr><th>支持的</th><th>不支持</th></tr><tr><td>ADC_SC2</td><td>ADC_SC2</td></tr></table>0000b - 没有发生错误 0001b - 发生了一个错误</td></tr>
<tr><td>23-20<br>-</td><td>保留</td></tr>
<tr><td>19-16<br>辽宁省</td><td>触发状态<br>这些状态位中的每一个都表示来自一个源的多路硬件触发请求已经被锁定。这个字段中的每个位都对应于一个单独的触发请求。<br>• 第16位对应的是触发请求0<br>• 第17位对应的是触发请求1<br>• 第18位对应的是触发请求2<br>• 第19位对应的是触发请求3<br>该字段中每个位的读值被单独解释如下。<br>注意：这个字段并不是在每个实例中都支持。<table><tr><th>支持的</th><th>不支持</th></tr><tr><td>ADC_SC2</td><td>ADC_SC2</td></tr></table>0000b - 没有触发请求被锁定 0001b - 一个触发请求被锁定</td></tr>
<tr><td>15<br>-</td><td>保留</td></tr>
</table>

| | |
|---|---|
| 14-13<br>ĂĂĂĂ | 触发程序编号<br><br>表示正在被服务的触发器编号。这必须与相应的触发器锁存状态的1位值相匹配。<br>• TRGPRNUM=00只有在TRGSTLAT[16]=1时有效。 |

*表继续在下一页...*

<table>
<tr><th>场地</th><th>职能</th></tr>
<tr><td></td><td>• TRGPRNUM=01只有在TRGSTLAT[17]=1时有效。<br>• TRGPRNUM=10只有在TRGSTLAT[18]=1时有效。<br>• TRGPRNUM=11只有在TRGSTLAT[19]=1时有效。<br><br>注意：这个字段并不是在每个实例中都支持。<br><table><tr><th>支持的</th><th>不支持</th></tr><tr><td>ADC_SC2</td><td>ADC_SC2</td></tr></table></td></tr>
<tr><td>12-8<br>-</td><td>保留</td></tr>
<tr><td>7<br>沟通与交流</td><td>转换活跃<br><br>表示转换或硬件平均化正在进行中。ADACT在转换开始时被设置，在转换完成或中止时被清除。<br>0b - 转换没有进行。1b - 转换正在进行中。</td></tr>
<tr><td>6<br>ĀĀĀ</td><td>转换触发器选择<br><br>选择用于启动转换的触发器的类型。可以选择两种类型的触发器。<br>• 软件触发。当选择软件触发时，在对SC1A进行写操作后开始转换。<br>• 硬件触发。当选择硬件触发时，在ADHWTSn输入的一个脉冲后，ADHWT输入的断言将启动转换。<br><br>0b - 选择软件触发。1b - 选择了硬件触发。</td></tr>
<tr><td>5<br>ACFE</td><td>启用比较功能<br><br>启用比较功能。<br>0b - 关闭比较功能。1b - 启用比较功能。</td></tr>
<tr><td>4<br>ACFGT</td><td>大于的比较功能启用<br><br>配置比较功能，根据ACREN的值检查相对于CV1和CV2的转换结果。ACFE必须被设置为ACFGT才能产生任何作用。更多细节请参见表39-4 "比较模式"。</td></tr>
<tr><td>3<br>ACREN</td><td>比较功能范围启用<br><br>配置比较功能，以检查被监测的输入的转换结果是否在由ACFGT的值决定的CV1和CV2形成的范围之间或之外。必须设置ACFE才能使ACFGT产生任何作用。更多细节见表39-4 "比较模式"。</td></tr>
<tr><td>2<br>DMAEN</td><td>启用DMA<br>0b - DMA被禁用。<br>1b - 启用DMA，并将在ADC转换完成的事件中发出ADC DMA请求，这在任何SC1n[COCO]标志被断言时表示。</td></tr>
</table>

| | |
|---|---|
| 1-0<br>REFSEL | 电压基准选择<br><br>选择用于转换的电压参考源。<br>00b - 默认的电压基准引脚对，即外部引脚$V_{REFH}$ 和 $V_{REFL}$<br>01b - 替代参考电压，即$V_{ALTH}$ 。该电压可能是额外的外部引脚或内部来源，取决于MCU配置。有关该MCU的具体细节，请参见芯片配置信息。<br>10b - 保留 11b - 保留 |

## 39.4.8 状态和控制寄存器3（SC3）

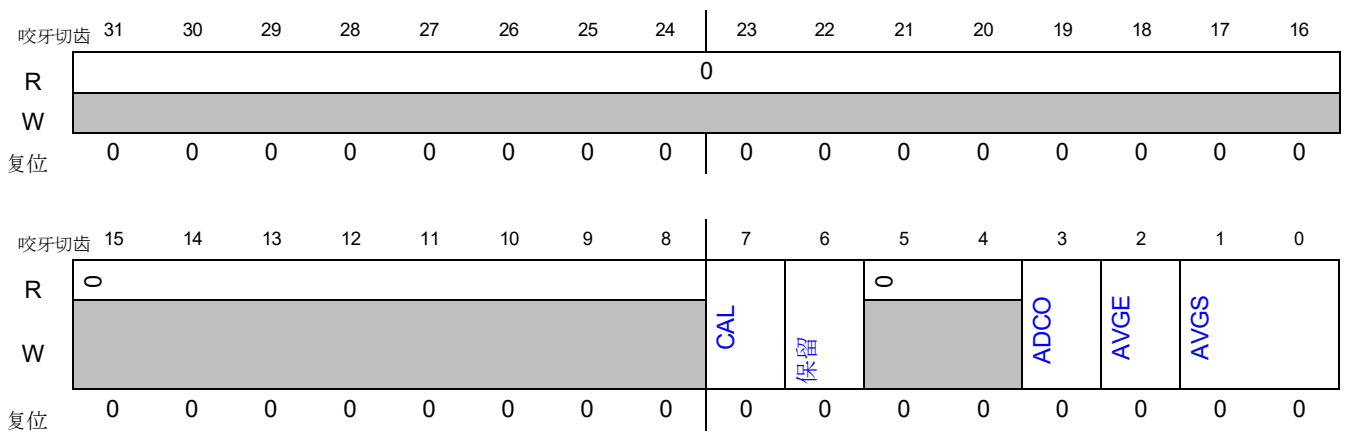
### 39.4.8.1 偏移

| 注册 | 偏移 |
|---|---|
| SC3 | 94h |

### 39.4.8.2 职能

状态和控制寄存器3（SC3）控制ADC模块的校准、连续转换和硬件平均功能。

### 39.4.8.3 图示




### 39.4.8.4 领域

| 场地 | 职能 |
|---|---|
| 31-8<br>- | 保留 |
| 7<br>CAL | 校准<br>当CAL=1时，ADC开始进行校准序列。该字段在校准过程中保持设置，并在校准序列完成后被清空。在它开始后，校准 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | 例程不能被写入ADC寄存器的操作打断，否则结果将无效。设置CAL将中止任何当前的转换。<br>**注意：**对于校准，必须使用平均数和平均数32。<br>**注意：**如果一个设备上有几个ADC，它们应按顺序进行校准。不允许对ADC进行平行校准，因为它们会相互干扰。 |
| 6<br>- | 保留 |
| 5-4<br>- | 保留 |
| 3<br>ADCO | 启用连续转换功能<br>启用连续转换。<br>0b - 启动转换后将进行一次转换（或一组转换，如果设置了AVGE）。<br>1b - 开始转换后，将进行连续转换（或连续转换组，如果设置了AVGE）。 |
| 2<br>AVGE | 硬件平均启用<br>启用ADC的硬件平均功能。<br>0b - 禁用硬件平均功能。1b - 启用硬件平均功能。 |
| 1-0<br>AVGS | 硬件平均选择<br>决定多少次ADC转换将被平均化以创建ADC平均结果。<br>00b - 4个样本的平均数。01b - 8个样本的平均数。10b - 16个样本平均。11b - 32个样本的平均值。 |

## 39.4.9 基准偏移寄存器（BASE_OFS）。

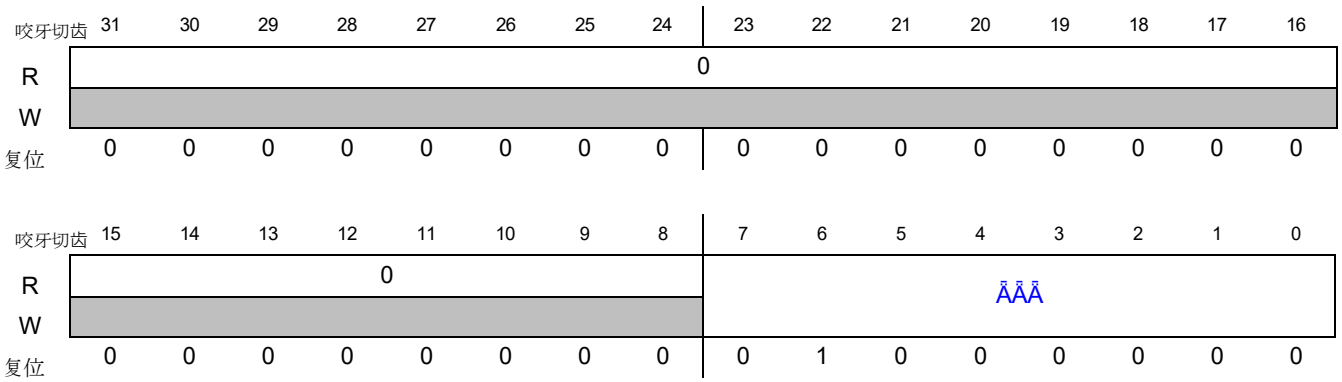
### 39.4.9.1 偏移

| 注册 | 偏移 |
| --- | --- |
| 基础设施 | 98h |

### 39.4.9.2 职能

基准偏移寄存器（BASE_OFS）包含校准算法用来确定偏移校准值（OFS）的偏移值。

### 39.4.9.3 图示




### 39.4.9.4 领域

| 场地 | 职能 |
|---|---|
| 31-8<br>- | 保留 |
| 7-0<br>ĀĀĀ | 基准偏移错误校正值 |

## 39.4.10 ADC偏移校正寄存器（OFS）

### 39.4.10.1 偏移

| 注册 | 偏移 |
|---|---|
| OFS | 9Ch |

### 39.4.10.2 职能

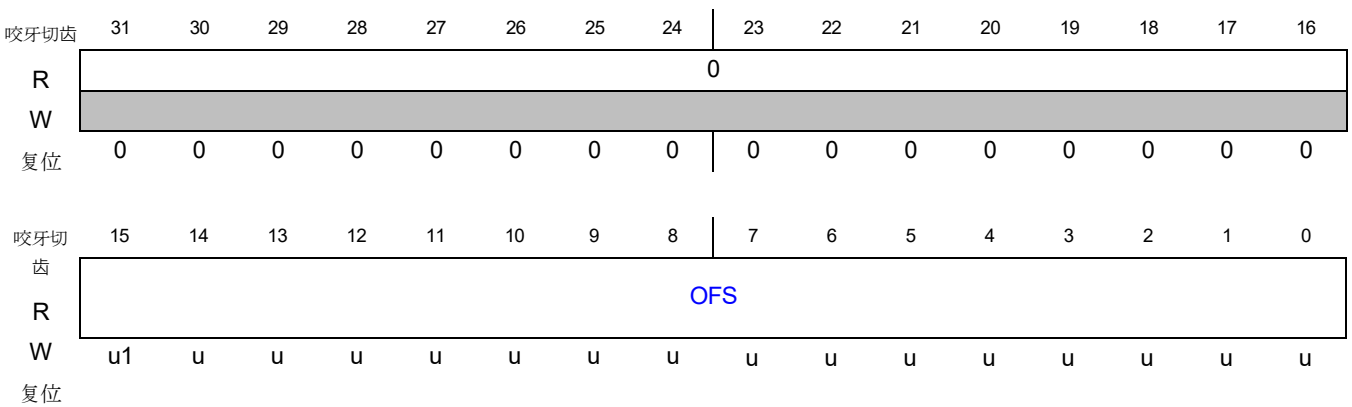
ADC偏移校正寄存器（OFS）包含校准生成的偏移误差校正值（OFS）。BA_OFS中的值在校准算法中被用来计算被存储在OFS寄存器中的偏移校正值。寄存器中的值

OFS从转换中减去，结果被转移到结果寄存器Rn中。如果结果大于最大或小于最小的结果值，它将被强制到当前操作模式的适当限制。

**注意事项**

如果偏移量寄存器被设置为负值，并且低于或等于0xFFF8，ADC将不会产生代码0；如果偏移量寄存器被设置为负值，并且低于或等于0xFFF0，ADC将不会产生代码1。

### 39.4.10.3 图示




1.复位值从IFR中加载。

### 39.4.10.4 领域

| 场地 | 职能 |
|---|---|
| 31-16<br>- | 保留 |
| 15-0<br>OFS | 偏移错误校正值 |

## 39.4.11 用户偏移校正寄存器（USR_OFS）。

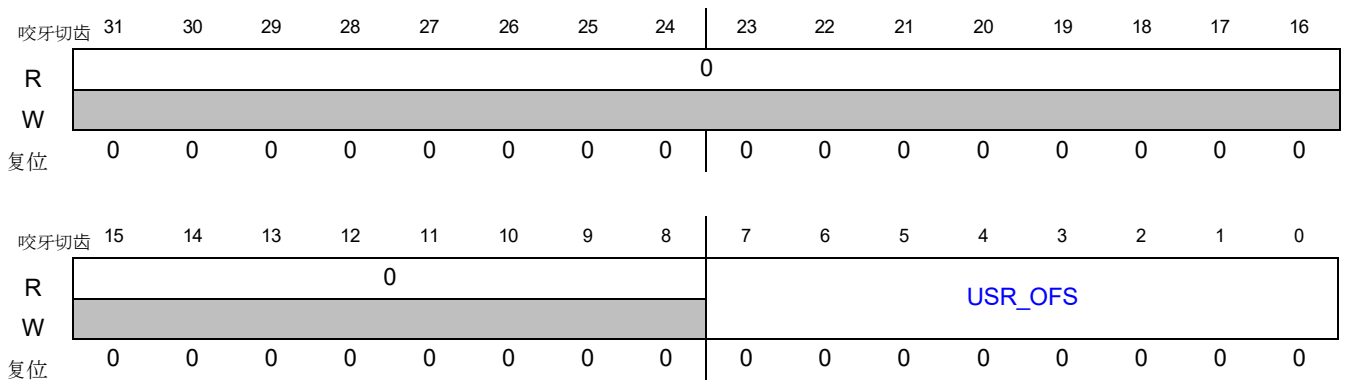
## 39.4. 偏移
1024.1

| 注册 | 偏移 |
|---|---|
| USR_OFS | A0h |

### 39.4.11.2 职能

ADC
USER偏移校正寄存器（USR_OFS）包含用户定义的偏移误差校正值，用于转换结果误差校正算法中。

### 39.4.11.3 图示




### 39.4.11.4 领域

| 场地 | 职能 |
|---|---|
| 31-8<br>- | 保留 |
| 7-0<br>USR_OFS | USER 偏移量错误校正值 |

## 39.4.12 ADC X偏移校正寄存器（XOFS）

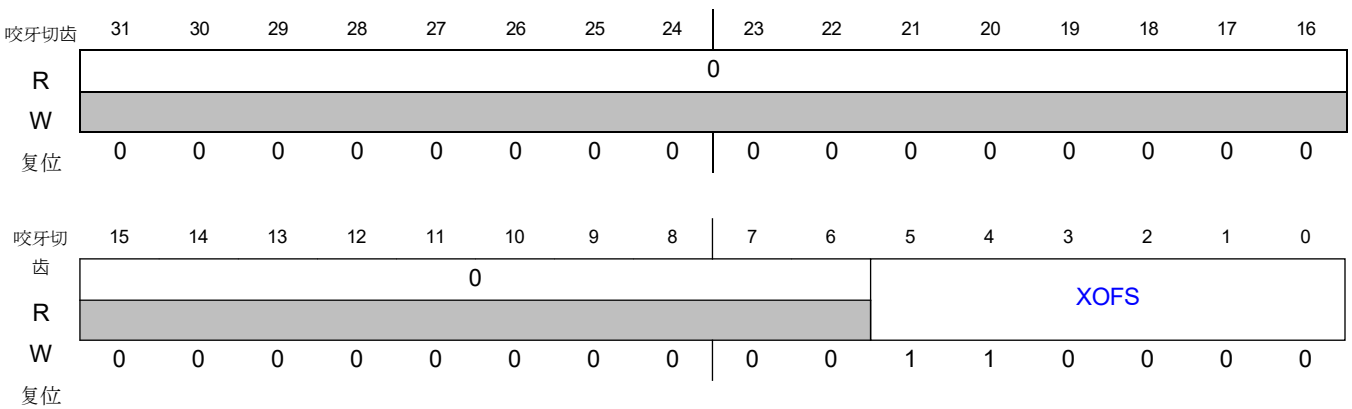
## 39.4. 偏移
1025.1

| 注册 | 偏移 |
|---|---|
| XOFS | A4h |

### 39.4.12.2 职能

ADC的X偏移校正寄存器（XOFS）包含转换结果错误校正算法中使用的X偏移。

### 39.4.12.3 图示




### 39.4.12.4 领域

| 场地 | 职能 |
|---|---|
| 31-6<br>- | 保留 |
| 5-0<br>XOFS | X偏移误差修正值 |

## 39.4.13 ADC Y偏移校正寄存器（YOFS）。

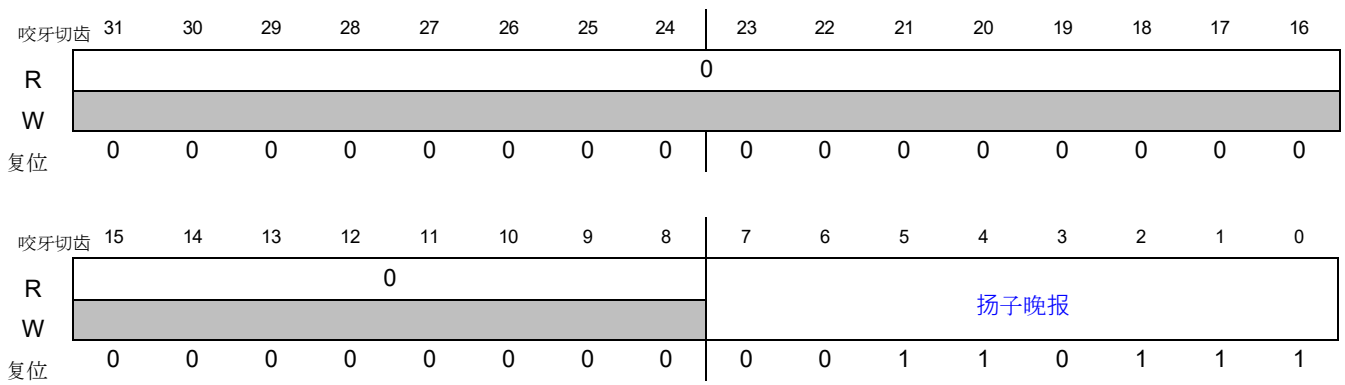
## 39.4. 偏移
1026.1

| 注册 | 偏移 |
|---|---|
| 扬子晚报 | A8h |

### 39.4.13.2 职能

ADC的Y偏移校正寄存器（YOFS）包含转换结果错误校正算法中使用的Y偏移。

### 39.4.13.3 图示




### 39.4.13.4 领域

| 场地 | 职能 |
|---|---|
| 31-8<br>- | 保留 |
| 7-0<br>扬子晚报 | Y偏移误差校正值 |

## 39.4.14 ADC增益寄存器（G）

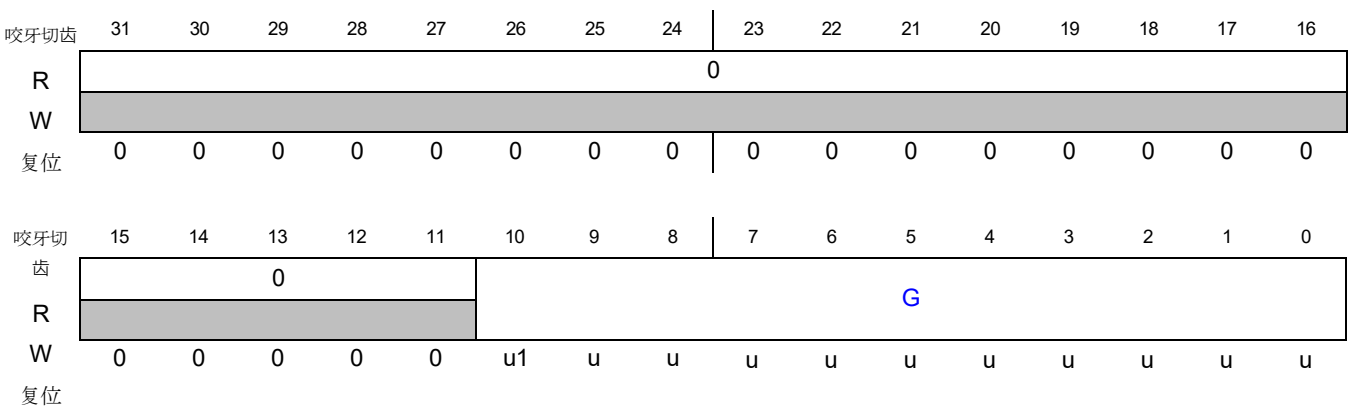
## 39.4. 偏移 1027.1

| 注册 | 偏移 |
|---|---|
| G | ACh |

### 39.4.14.2 职能

增益寄存器（G）包含整体转换的增益误差修正。G是一个二进制格式的11位实数，是增益调整系数。这个寄存器的值是由校准算法决定和上传的。

### 39.4.14.3 图示




1.复位值从IFR中加载。

### 39.4.14.4 领域

| 场地 | 职能 |
|---|---|
| 31-11<br>- | 保留 |
| 10-0<br>G | G<br>整体转换的增益误差调整系数 |

# 39.4.15 ADC用户增益寄存器（UG）

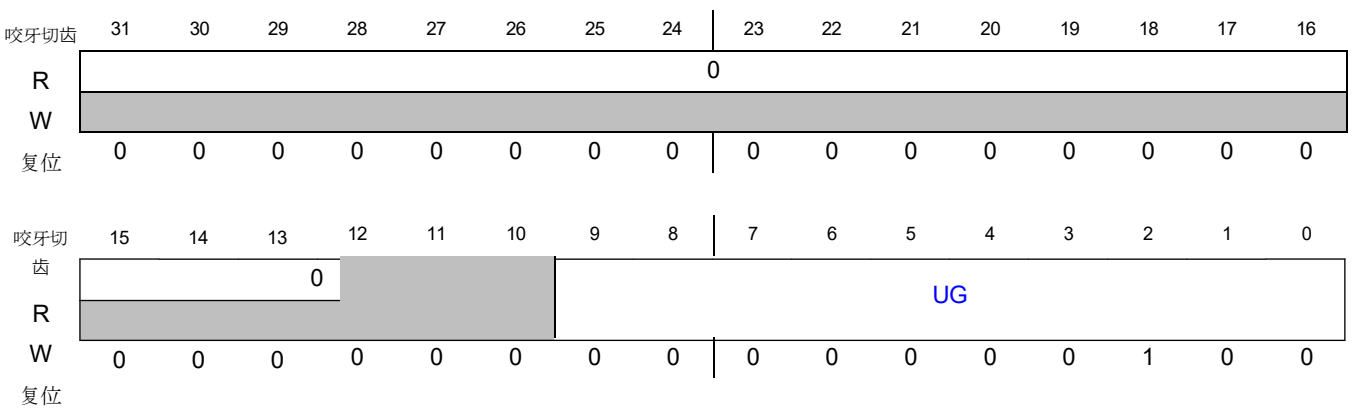
## 39.4.15.1 偏移

| 注册 | 偏移 |
|---|---|
| UG | B0h |

## 39.4.15.2 职能

用户增益寄存器（UG）包含用户增益误差修正。它允许你调整最终的校准增益值。

## 39.4.15.3 图示




## 39.4.15.4 领域

| 场地 | 职能 |
|---|---|
| 31-10<br>- | 保留 |
| 9-0<br>UG | UG<br>用户增益误差修正值 |

# 39.4.16 ADC通用校准值寄存器S（CLPS）。

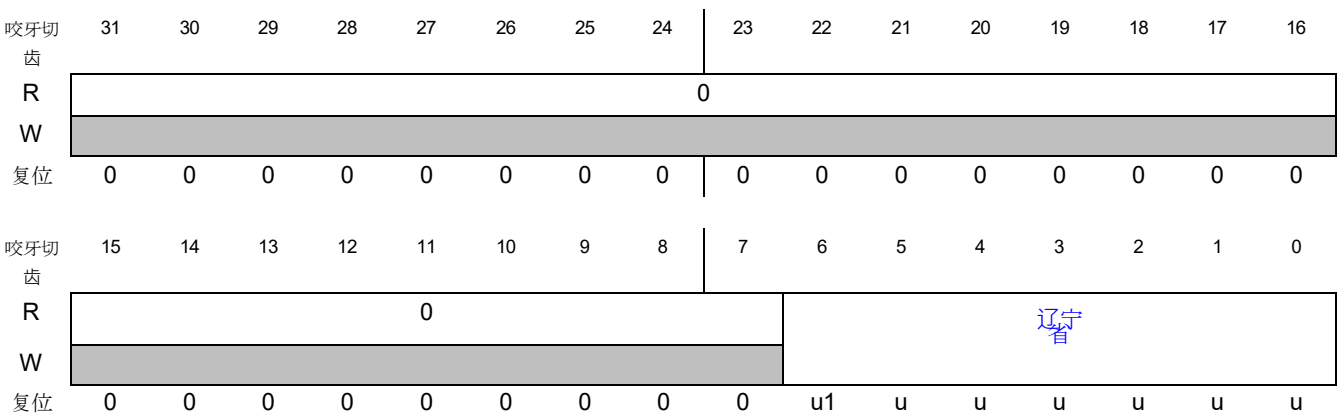
## 39.4.16.1 偏移

| 注册 | 偏移 |
|---|---|
| 辽宁省 | B4h |

## 39.4.16.2 职能

一般校准值寄存器（CLPx）包含由校准功能产生的校准信息。这些寄存器包含了7个宽度不同的二补格式的有符号校准值。CLPx在自校准序列完成后自动设置，即CAL被清零。如果这些寄存器在校准后被用户写入，则可能不符合线性误差规范。

## 39.4.16.3 图示




1.复位值从IFR中加载。

## 39.4.16.4 领域

| 场地 | 职能 |
|---|---|
| 31-7<br>- | 保留 |

| | |
|---|---|
| 6-0<br>辽宁省 | 辽宁省<br>校准值 |

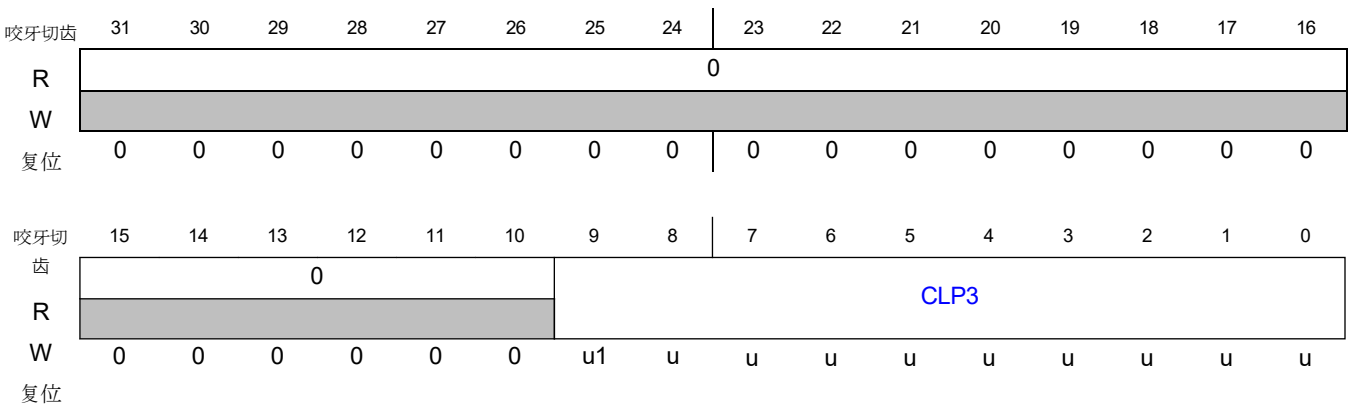
## 39.4.16 ADC通用校准值寄存器S（CLPS）。

## 39.4.17 ADC Plus-Side通用校准值寄存器3（CLP3）。

### 39.4.17.1 偏移

| 注册 | 偏移 |
|---|---|
| CLP3 | B8h |

### 39.4.17.2 图示




1.复位值从IFR中加载。

### 39.4.17.3 领域

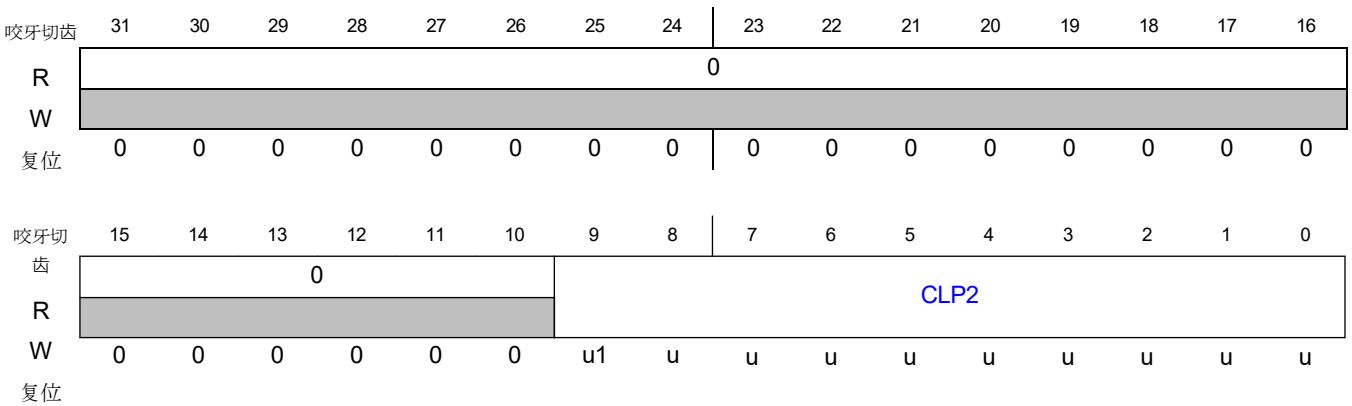
| 场地 | 职能 |
|---|---|
| 31-10<br>- | 保留 |
| 9-0<br>CLP3 | CLP3<br>校准值 |

## 39.4.18 ADC Plus-Side通用校准值寄存器2（CLP2）。

### 39.4.18.1 偏移

| 注册 | 偏移 |
|---|---|
| CLP2 | BCh |

### 39.4.18.2 图示




1.复位值从IFR中加载。

### 39.4.18.3 领域

| 场地 | 职能 |
|---|---|
| 31-10<br>- | 保留 |
| 9-0<br>CLP2 | CLP2<br>校准值 |

## 39.4.19 ADC Plus-Side通用校准值寄存器1（CLP1）。

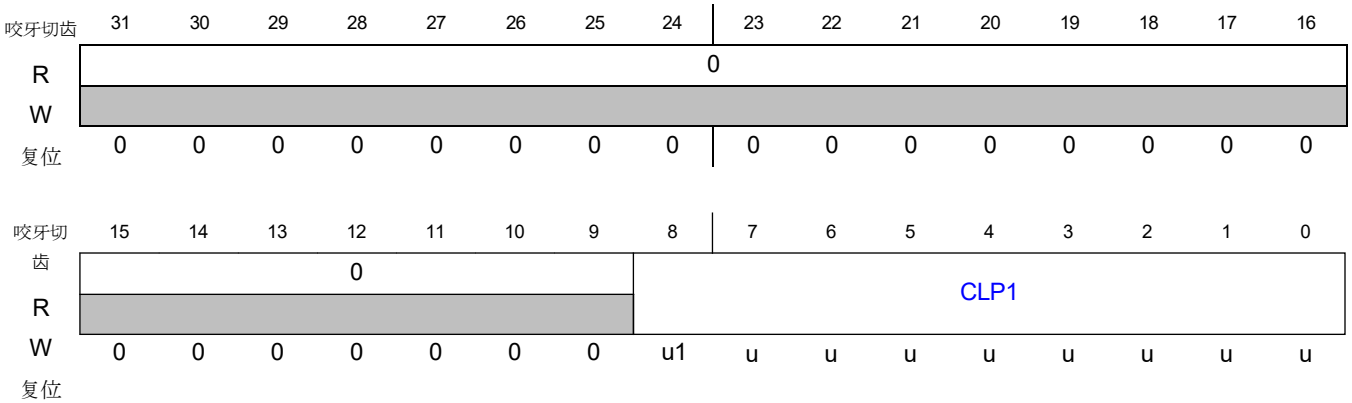
### 39.4.19.1 偏移

| 注册 | 偏移 |
|---|---|

| | |
|---|---|
| ADC寄存器描述 | C0h |

### 39.4.18.1 偏移

### 39.4.19.2 图示




1.复位值从IFR中加载。

### 39.4.19.3 领域

| 场地 | 职能 |
|---|---|
| 31-9<br>- | 保留 |
| 8-0<br>CLP1 | CLP1<br>校准值 |

## 39.4.20 ADC Plus-Side通用校准值寄存器0（CLP0）。

### 39.4.20.1 偏移

| 注册 | 偏移 |
|---|---|
| CLP0 | C4h |

### 39.4.1033.2 图示

| 咬牙切齿 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 芯片 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | CLP0 | | | | | | | |
| W | | | | | | | | | | | | | | | | |
| 重置 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | u1 | u | u | u | u | u | u | u |

1.复位值从IFR中加载。

## 39.4.20.3 领域

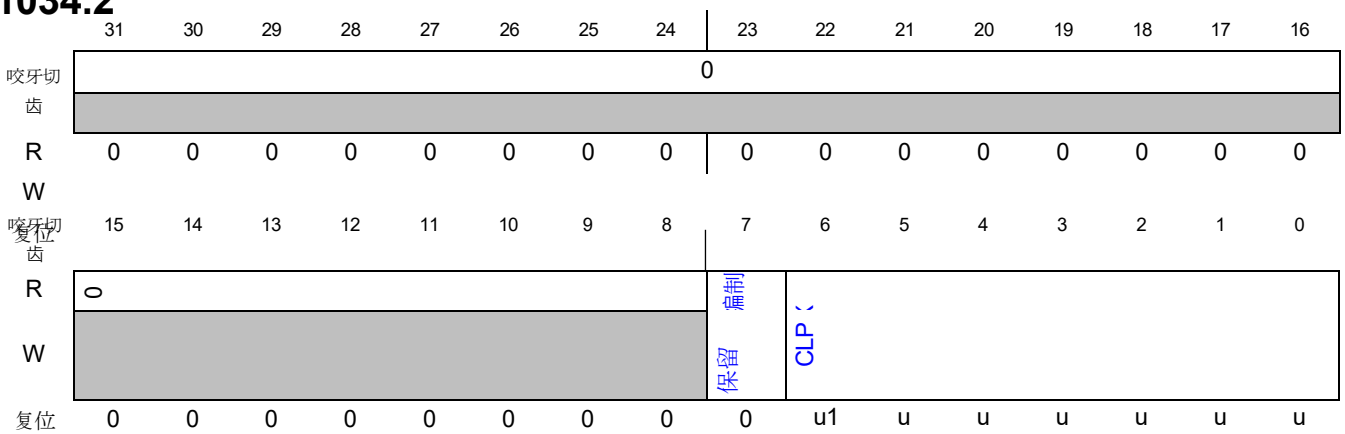
| 场地 | 职能 |
|---|---|
| 31-8<br>- | 保留 |
| 7-0<br>CLP0 | CLP0<br>校准值 |

# 39.4.21 ADC Plus-Side通用校准值寄存器X（CLPX）。

## 39.4.21.1 偏移

| 注册 | 偏移 |
|---|---|
| CLPX | C8h |

## 39.4.1034.2 图示




1.复位值从IFR中加载。

### 39.4.21.3 领域

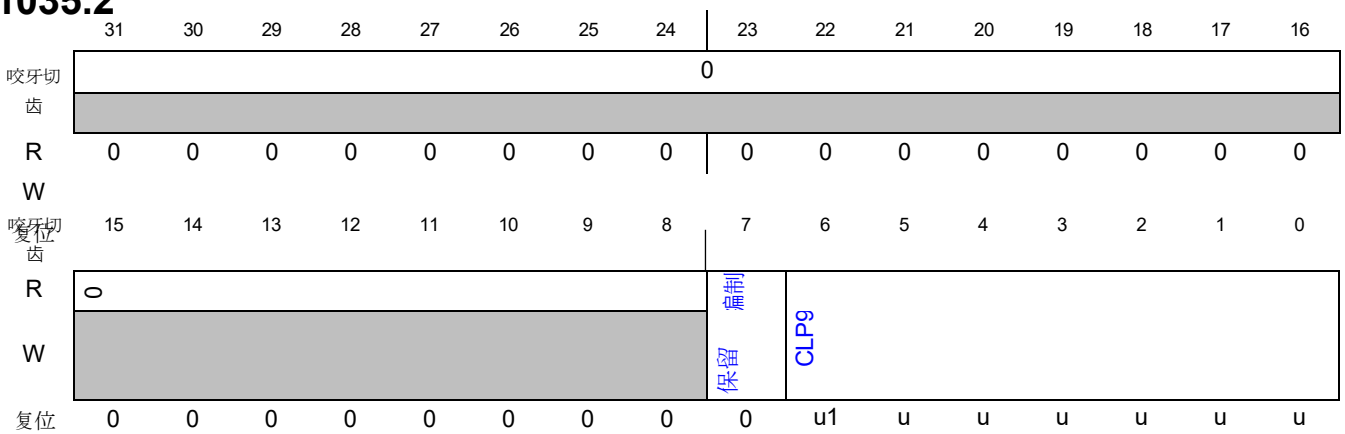
| 场地 | 职能 |
|---|---|
| 31-8<br>- | 保留 |
| 7<br>- | 保留 |
| 6-0<br>CLPX | CLPX<br>校准值 |

## 39.4.22 ADC Plus-Side通用校准值寄存器9（CLP9）。

### 39.4.22.1 偏移

| 注册 | 偏移 |
|---|---|
| CLP9 | CCh |

## 39.4.1035.2 图示




1.复位值从IFR中加载。

### 39.4.22.3 领域

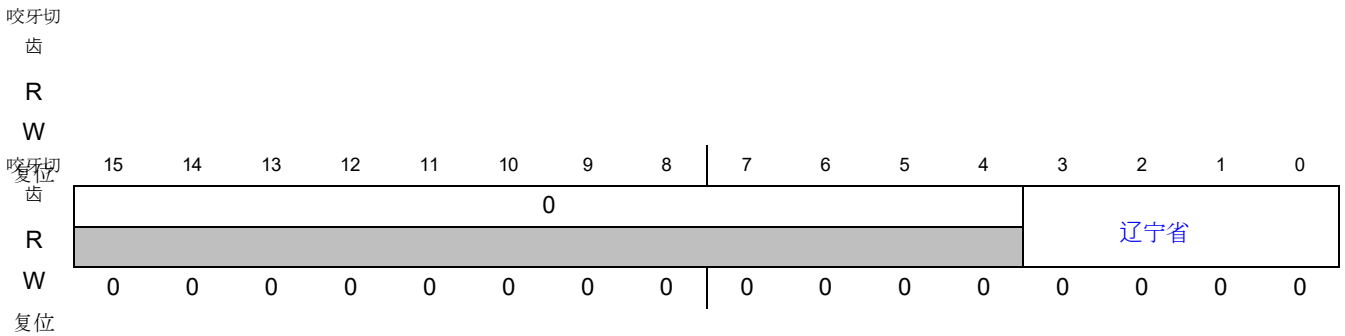
| 场地 | 职能 |
|---|---|
| 31-8<br>- | 保留 |
| 7<br>- | 保留 |
| 6-0<br>CLP9 | CLP9<br>校准值 |

## 39.4.23 ADC通用校准偏移值寄存器S（CLPS_ OFS）。

### 39.4.23.1 偏移

| 注册 | 偏移 |
|---|---|
| 辽宁省 | D0h |

## 39.4. 1036.2 图示




## 39.4.23.3 领域

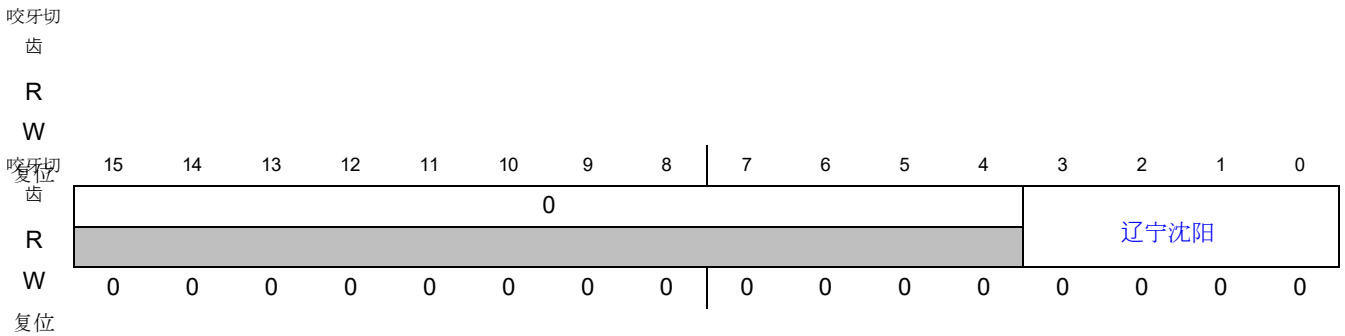
| 场地 | 职能 |
|---|---|
| 31-4<br>- | 保留 |
| 3-0<br>辽宁省 | CLPS偏移<br>电容器偏移校正值 |

# 39.4.24 ADC Plus-Side通用校准偏移值寄存器3（CLP3_OFS）。

## 39.4.24.1 偏移

| 注册 | 偏移 |
|---|---|
| 辽宁沈阳 | D4h |

## 39.4.1037.2 图示




## 39.4.24.3 领域

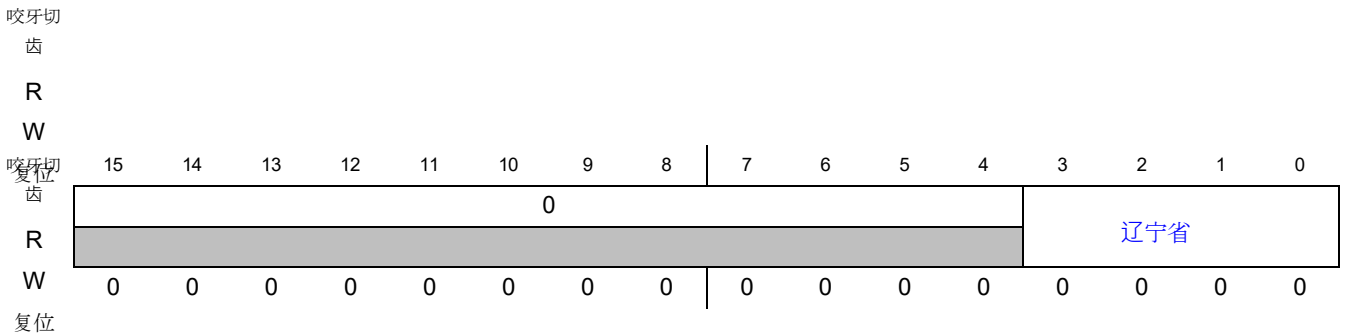
| 场地 | 职能 |
|---|---|
| 31-4<br>- | 保留 |
| 3-0<br>辽宁沈阳 | CLP3偏移<br>电容器偏移校正值 |

# 39.4.25 ADC Plus-Side通用校准偏移值寄存器2（CLP2_OFS）。

## 39.4.25.1 偏移

| 注册 | 偏移 |
|---|---|
| 辽宁省 | D8h |

## 39.4.1038.2 图示




## 39.4.25.3 领域

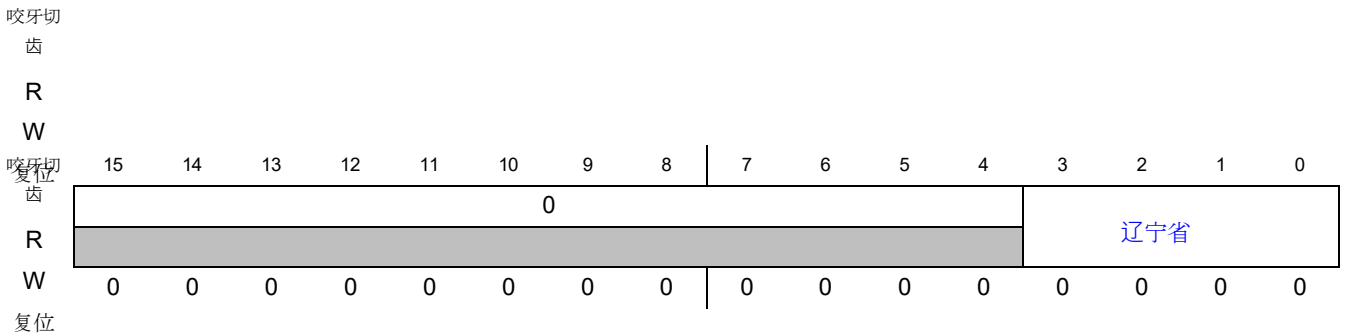
| 场地 | 职能 |
|---|---|
| 31-4<br>- | 保留 |
| 3-0<br>辽宁省 | CLP2偏移<br>电容器偏移校正值 |

# 39.4.26 ADC Plus-Side通用校准偏移值寄存器1（CLP1_OFS）。

## 39.4.26.1 偏移

| 注册 | 偏移 |
|---|---|
| 辽宁省 | DCh |

## 39.4.1039.2 图示




### 39.4.26.3 领域

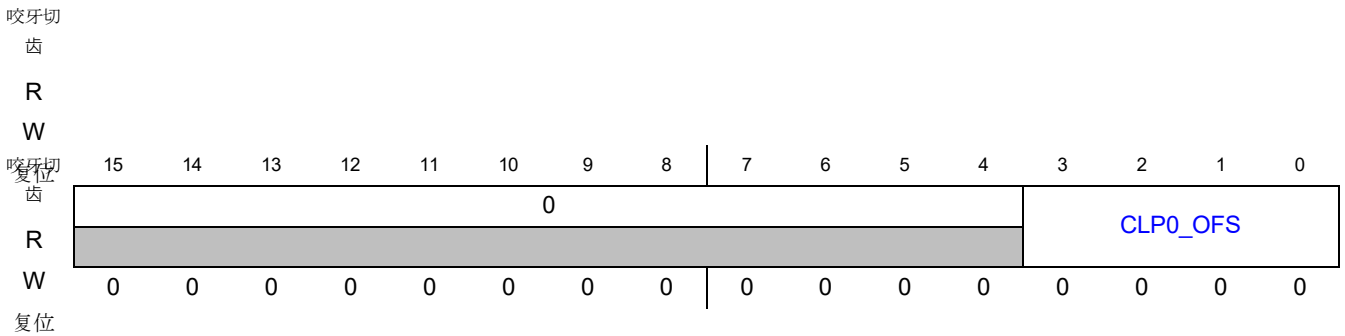
| 场地 | 职能 |
|---|---|
| 31-4<br>- | 保留 |
| 3-0<br>辽宁省 | CLP1偏移<br>电容器偏移校正值 |

## 39.4.27 ADC Plus-Side通用校准偏移值寄存器0（CLP0_OFS）。

### 39.4.27.1 偏移

| 注册 | 偏移 |
|---|---|
| 辽宁省 | E0h |

## 39.4.1040.2 图示




## 39.4.27.3 领域

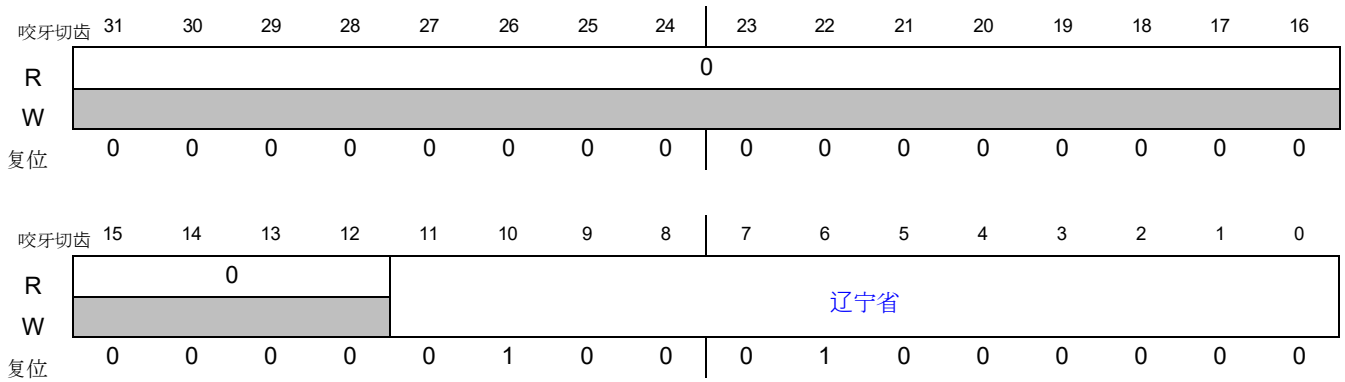
| 场地 | 职能 |
|---|---|
| 31-4<br>- | 保留 |
| 3-0<br>CLP0_OFS | CLP0 偏移<br>电容器偏移校正值 |

# 39.4.28 ADC Plus-Side通用校准偏移值寄存器X（CLPX_OFS）。

## 39.4.28.1 偏移

| 注册 | 偏移 |
|---|---|
| 辽宁省 | E4h |

### 39.4.28.2 图示




### 39.4.28.3 领域

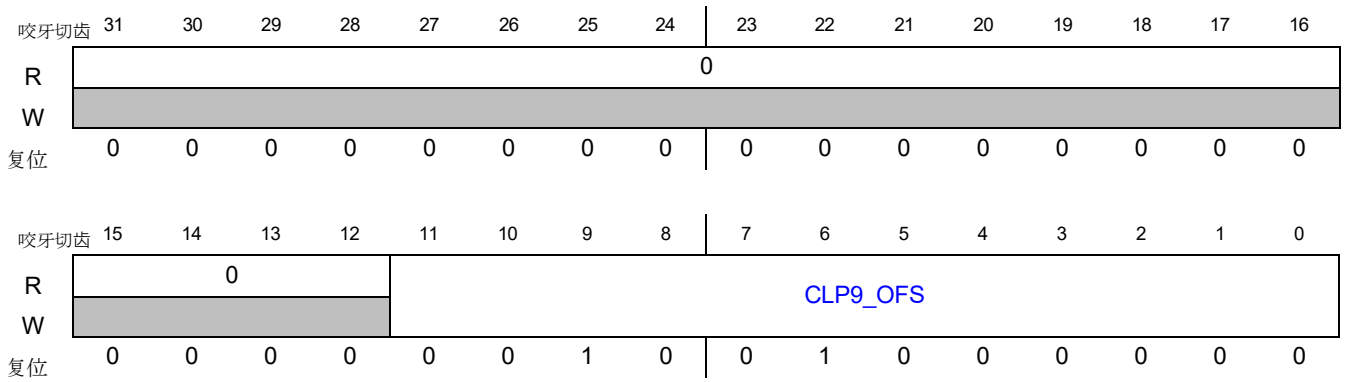
| 场地 | 职能 |
|---|---|
| 31-12<br>- | 保留 |
| 11-0<br>辽宁省 | CLPX 偏移量<br>电容器偏移校正值 |

## 39.4.29 ADC Plus-Side通用校准偏移值寄存器9（CLP9_OFS）。

### 39.4.29.1 偏移

| 注册 | 偏移 |
|---|---|
| CLP9_OFS | E8h |

### 39.4.29.2 图示




### 39.4.29.3 领域

| 场地 | 职能 |
|---|---|
| 31-12<br>- | 保留 |
| 11-0<br>CLP9_OFS | CLP9 偏移量<br>电容器偏移校正值 |

## 39.4.30 ADC状态和控制寄存器1（SC1Q - SC1AF）。

### 39.4.30.1 偏移

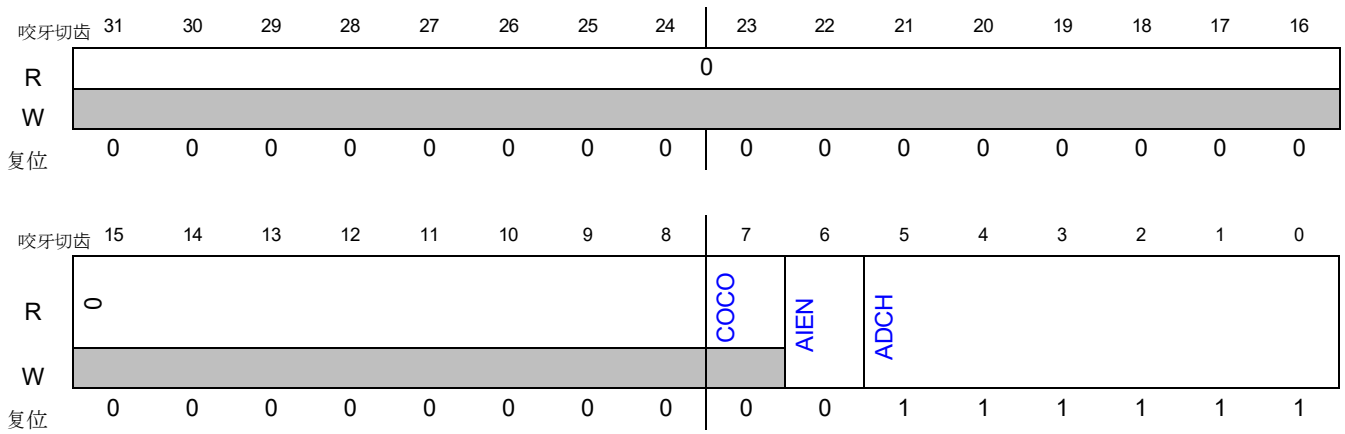
| 注册 | 偏移 |
|---|---|
| SC1Q | 148h |
| SC1R | 14Ch |
| SC1S | 150h |
| SC1T | 154h |
| SC1U | 158h |
| SC1V | 15Ch |
| SC1W | 160h |
| SC1X | 164h |
| SC1Y | 168h |

| | |
|---|---|
| SC1Z | 16Ch |

*表继续在下一页...*

| 注册 | 偏移 |
|---|---|
| SC1AA | 170h |
| SC1AB | 174h |
| SC1AC | 178h |
| SC1AD | 17Ch |
| SC1AE | 180h |
| SC1AF | 184h |

## 39.4.30.2 图示




## 39.4.30.3 领域

| 场地 | 职能 |
|---|---|
| 31-8<br>- | 保留 |
| 7<br>COCO | 转换完成标志<br>0b - 转换未完成。1b - 转换已完成<br>。 |
| 6<br>AIEN | 中断启用<br>0b - 转换完成中断被禁用。1b - 转换完成中断被<br>启用。 |

| | |
|---|---|
| 5-0<br>ADCH | 输入通道选择<br>000000b - 外部通道0被选择为输入。000001b - 外部通道1被选为输入。000010b - 外部通道2被选为输入。000011b - 外部通道3被选为输入。000100b - 外部通道4被选择为输入。000101b - 外部通道5被选择为输入。000110b - 外部通道6被选择为输入。000111b - 外部通道7被选择为输入。 |

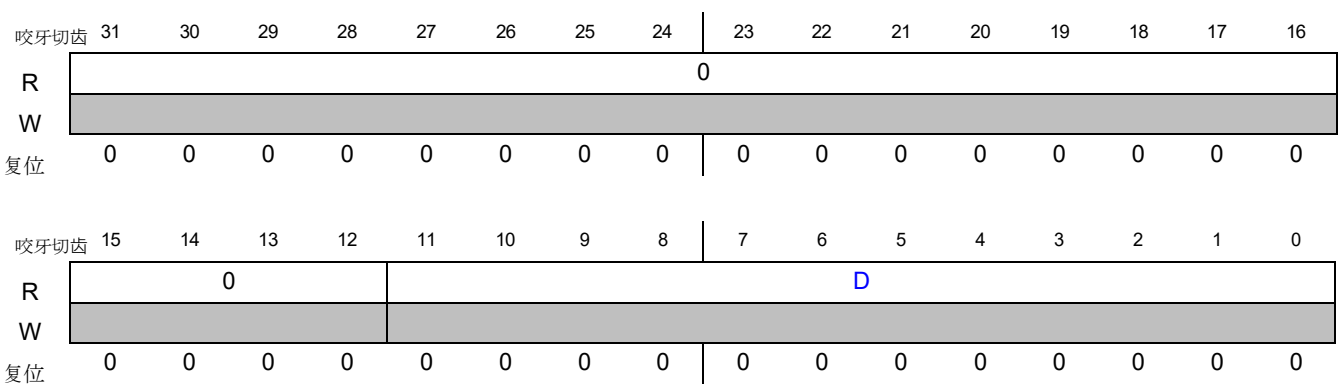
| 场地 | 职能 |
| --- | --- |
| | 001000b - 外部通道8被选择为输入。001001b - 外部通道9被选为输入。001010b - 外部通道10被选为输入。001011b - 外部通道11被选择为输入。001100b - 外部通道12被选择为输入。001101b - 外部通道13被选择为输入。001110b - 外部通道14被选择为输入。001111b - 外部通道15被选择为输入。010000b - 保留<br>010001b - 保留<br>010010b - 保留<br>010011b - 保留<br>010100b - 保留<br>010101b - 内部通道0被选为输入。010110b - 内部通道1被选为输入。010111b - 内部通道2被选择为输入。011000b - 保留<br>011001b - Reserved<br>011010b - Reserved<br>011011b - Band Gap<br>011100b - 内部通道3被选择为输入。<br>011101b - $V_{REFSH}$ 被选为输入。选择的电压基准由SC2[REFSEL]决定。011110b - $V_{REFSL}$ 被选为输入。选择的电压基准由SC2[REFSEL]决定。011111b - 保留<br>100000b - 外部通道16被选为输入。100001b - 外部通道17被选为输入。100010b - 外部通道18被选为输入。100011b - 外部通道19被选为输入。<br>100100b - 外部通道20被选择为输入。100101b - 外部通道21被选择为输入。100110b - 外部通道22被选择为输入。100111b - 外部通道23被选择为输入。101000b - 外部通道24被选择为输入。101001b - 外部通道25被选为输入。101010b - 外部通道26被选为输入。101011b - 外部通道27被选为输入。<br>101100b - 外部通道28被选为输入。101101b - 外部通道29被选为输入。101110b - 外部通道30被选为输入。101111b - 外部通道31被选择为输入。<br>110000b - 保留<br>110001b - Reserved<br>110010b - Reserved<br>110011b - Reserved<br>110100b - Reserved<br>110101b - Reserved<br>110110b - Reserved<br>110111b - Reserved<br>111000b - Reserved<br>111001b - Reserved<br>111010b - Reserved<br>111011b - Reserved<br>111100b - Reserved<br>111101b - Reserved<br>111110b - Reserved<br>111111b - 模块已禁用 |

# 39.4.31 ADC数据结果寄存器（RQ - RAF）。

## 39.4.31.1 偏移

| 注册 | 偏移 |
|---|---|
| RQ | 1C8h |
| RR | 1CCh |
| ÅÅÅ | 1D0h |
| 报道 | 1D4h |
| RU | 1D8h |
| 农村妇女 | 1DCh |
| RW | 1E0h |
| 荣格 | 1E4h |
| RY | 1E8h |
| RZ | 1ECh |
| RAA | 1F0h |
| RAB | 1F4h |
| RAC | 1F8h |
| RAD | 1FCh |
| RAE | 200h |
| 皇家空军 | 204h |

## 39.4.31.2 图示



### 39.4.31.3 领域

| 场地 | 职能 |
|---|---|
| 31-12<br>- | 保留 |
| 11-0<br>D | 数据结果 |

## 39.5 职能描述

在复位过程中，或当SC1n[ADCH]全部为高电平时，ADC模块被禁用；详情见电源管理信息。当一个转换完成而另一个转换尚未开始时，模块处于空闲状态。当它处于空闲状态时，模块处于最低功率状态。ADC可以在任何一个软件可选通道上进行模数转换。所有模式都通过逐次逼近算法进行转换。

为了满足精度规格，必须使用片上校准功能对ADC模块进行校准。

关于如何进行校准的细节，请参见校准功能。

当转换完成后，其结果被放置在Rn数据寄存器中。然后，各自的SC1n[COCO]被设置，如果各自的转换完成中断被启用，或者当SC1n[AIEN]=1时，则产生中断。

ADC模块具有将转换结果与CV1和CV2寄存器的内容进行自动比较的能力。比较功能通过设置SC2[ACFE]来启用，并在任何转换模式和配置中运行。

ADC模块具有对多次转换结果进行自动平均的能力。通过设置SC3[AVGE]启用硬件平均功能，并在任何转换模式和配置中运行。

**注意事项**

关于芯片的具体操作模式，请参见该芯片的电源管理信息。

### 39.5.1 时钟选择和分频控制

可以选择四个时钟源中的一个作为ADC模块的时钟源。这个时钟源再除以一个可配置的值，以产生模块的输入时钟ADCK。该时钟是通过配置CFG1[ADICLK]来选择的。ALTCLKx，为该MCU定义。参见芯片配置信息。当MCU处于正常停止模式时，使用ALTCLKx作为输入时钟源可以进行转换。
ALTCLK1是复位后的默认选择。

无论选择哪种时钟，其频率必须落在ADCK的指定频率范围内。如果可用的时钟太慢，ADC的性能可能不符合规范。如果可用的时钟过快，则必须将时钟划分到适当的频率。这个分频器由CFG1[ADIV]指定，可以被1、2、4或8分。ADC总线时钟频率必须大于或等于ADC
ALT时钟频率。有关ADC规格，请参考器件*数据表*。

### 39.5.2 电压基准选择

ADC可以被配置为接受两个电压参考对中的一个作为转换用的参考电压（$V_{REFSH}$ 和 $V_{REFSL}$）。

每一对都包含一个必须在最小参考电压高电平和$V_{DDA}$
之间的正向参考，以及一个必须与$V_{SSA}$
处于同一电位的接地参考。这两对参考是外部的（$V_{REFH}$ 和$V_{REFL}$
）和交替的（$V_{ALTH}$
）。这些电压基准可以通过SC2[REFSEL]来选择。备用的$V_{ALTH}$
电压基准可以选择额外的外部引脚或内部源，这取决于MCU配置。关于该MCU特有的电压基准，请参见芯片配置信息。

### 39.5.3 硬件触发器和通道选择

ADC模块有一个可选择的异步硬件转换触发器ADHWT，当SC2[ADTRG]被设置且硬件触发器选择事件ADHWTSn发生时，该触发器被启用。并非所有的MCU都有这个来源。有关ADHWT源和该MCU特有的ADHWTSn配置的信息，请参见芯片特定的ADC信息。

当ADHWT源可用且硬件触发器启用时，即SC2[ADTRG]=1，在硬件触发器选择事件ADHWTSn发生后，ADHWT的上升沿启动转换。如果触发器的上升沿发生时，

转换正在进行，则上升沿被忽略。在连续转换n配置中，只有启动连续转换的初始上升沿

观察，直到转换被中止，ADC继续在启动转换的同一SCn寄存器上进行转换。硬件触发功能与任何一种转换模式和配置一起操作。

硬件触发器选择事件ADHWTSn必须在收到ADHWT信号之前设置。如果不满足这些条件，转换器可能会忽略触发或使用不正确的配置。如果硬件触发器选择事件在转换过程中被断言，它必须保持断言，直到当前转换结束，并保持设置，直到收到ADHWT信号来触发一个新的转换。为转换选择的通道和状态字段取决于活动的触发选择信号。

- ADHWTSA主动选择SC1A。
- ADHWTSn主动选择SC1n。

**注意事项**

同时断言一个以上的硬件触发选择信号（ADHWTSn）将导致不可预测的结果。为了避免这种情况，在下一次转换之前，只选择一个硬件触发选择信号（ADHWTSn）。

当转换完成后，结果被放置在与收到的ADHWTSn相关的Rn寄存器中。比如说。

- ADHWTSA有效选择RA寄存器
- ADHWTSn有源选择Rn寄存器

然后，与收到的ADHWTSn相关的转换完成标志，即SC1n[COCO]被设置，如果相应的转换完成中断被启用，即SC1[AIEN]=1，则产生中断。

### 39.5.4 转换控制

转换模式是通过配置CFG1[MODE]来选择的。转换可以由

软件或硬件触发启动。此外，ADC模块可以被配置为。

- 低功耗操作
- 采样时间长
- 连续转换
- 硬件平均水平
- 自动将转换结果与软件确定的比较值进行比较

### 39.5.4.1 启动转换

开始转换。

- 在对SC1A进行写操作后，如果选择了软件触发操作，即当SC2[ADTRG]=0时。
- 在硬件触发或ADHWT事件之后，如果选择了硬件触发操作，即SC2[ADTRG]=1，并且发生了硬件触发选择事件ADHWTSn。选择的通道和状态字段取决于活动的触发选择信号。
  - ADHWTSA主动选择SC1A。
  - ADHWTSn主动选择SC1n。
  - 如果两者都没有激活，则选择关闭条件。

**注意事项**

在转换完成前选择一个以上的ADHWTSn将导致不可预知的结果。为了避免这种情况，在转换完成之前只选择一个ADHWTSn。

- 在连续转换启用时，即SC3[ADCO]=1时，将结果传输到数据寄存器之后。

如果启用了连续转换，则在当前转换完成后会自动启动新的转换。在软件触发的操作中，即当SC2[ADTRG]=0时，连续转换在SC1A被写入后开始，并持续到终止。在硬件触发操作中，即当SC2[ADTRG]=1且发生了一个ADHWTSn事件时，连续转换在硬件触发事件后开始，并持续到终止。

如果启用了硬件平均，在当前转换完成后，会自动启动新的转换，直到完成正确的转换次数。在软件触发的操作中，转换在SC1A被写入后开始。在硬件触发的操作中，转换在硬件触发后开始。如果连续转换也被启用，那么在选定的转换次数的最后一次之后，将启动一组新的转换来进行平均化。

### 39.5.4.2 完成转换

当转换的结果被传送到数据结果寄存器Rn中时，转换就完成了，如下表所示。

表39-3.转换完成的指示

| 比较功能 | 硬件平均化 | 转换状态 | SC1n[COCO]是否设置为1，转换结果是否转入数据结果寄存器？ |
|---|---|---|---|
| 失能 | 失能 | 未完成 | 没有 |
| 失能 | 失能 | 已完成 | 是 |
| 失能 | 已启用 | 未完成 | 没有 |
| 失能 | 已启用 | 已完成 | 是的，如果所选数量的转换中的最后一个已经完成 |
| 已启用 | 失能 | 未完成 | 没有 |
| 已启用 | 失能 | 已完成 | 是的，如果比较条件为真 |
| 已启用 | 已启用 | 未完成 | 没有 |
| 已启用 | 已启用 | 已完成 | 是的，如果[（所选转换数量的最后一个完成）和（比较条件为真）]。 |

如果各SC1n[AIEN]在各SC1n[COCO]被设置时为高电平，则产生一个中断。

### 39.5.4.3 中止转换

任何正在进行的转换在以下情况下被中止。

- 当SC1A正在主动控制一个转换时，对它的写入会中止当前的转换。在软件触发模式下，当SC2[ADTRG]=0时，如果SC1A[ADCH]等于一个非全1的值，对SC1A的写入将启动一个新的转换。当任何一个SC1B-SC1n寄存器正在控制一个转换时，对该寄存器的写入将中止当前的转换。SC1B-SC1n寄存器不用于软件触发操作，因此对SC1B-SC1n寄存器的写操作不会启动新的转换。
- 发生对除SC1A-SC1n寄存器以外的任何ADC寄存器的写入。这表明发生了操作模式的改变，因此当前的转换无效。
- MCU被重置。

当一个转换被中止时，数据寄存器Rn的内容不会被改变。数据寄存器继续为上一次成功转换完成后传输的数值。如果转换被复位中止，RA和Rn返回到它们的复位状态。

### 39.5.4.4 电源控制

在启动转换之前，ADC模块一直处于空闲状态。闲置状态意味着ADC转换程序被保持在复位状态。

### 39.5.4.5 采样时间和总转换时间

总的转换时间取决于。

- 由CFG2[SMPLTS]确定的采样时间
- 单片机总线频率
- 转换模式，由CFG1[MODE]决定。
- 转换时钟的频率，即$f_{ADCK}$。

在模块变得活跃之后，开始对输入进行采样。

1. CFG2[SMPLTS]根据所选择的转换模式，在采样时间之间进行选择。
2. 采样完成后，转换器与输入通道隔离，应用逐次逼近算法来确定模拟信号的数字值。
3. 转换算法完成后，转换的结果被转移到Rn。

最大的总转换时间由选择的时钟源和选择的分频比决定。时钟源可由CFG1[ADICLK]选择，除法率由CFG1[ADIV]指定。计算总转换时间时，采用以下公式。

ADC总转换时间=采样阶段时间（由SMPLTS+1设置）+保持阶段（1个ADC周期）+比较阶段时间（8位模式=20个ADC周期，10位模式=24个ADC周期，12位模式=28个ADC周期）+单或第一连续时间加法器（5个ADC周期+5个总线时钟周期）

### 39.5.4.6 硬件平均功能

通过设置SC3[AVGE]=1可以启用硬件平均功能，对多个转换进行硬件平均。转换的数量由AVGS[1:0]位决定，可以选择4、8、16或32个转换进行平均。当硬件平均功能正在进行时，SC2[ADACT]将被设置。

在对选定的输入进行采样和转换后，结果被放在一个累加器中，在完成选定的转换次数后，从中计算出一个平均值。当选择硬件平均时，单个转换的完成将不会设置SC1n[COCO]。

如果比较功能被禁用或评估为真，在选定的转换次数完成后，平均转换结果被转移到数据结果寄存器Rn，SC1n[COCO]被设置。如果相应的ADC中断被启用，即SC1n[AIEN]=1，则在设置SC1n[COCO]时产生一个ADC中断。

**注意事项**

当MCU处于等待或正常停止模式时，硬件平均功能可以在一个通道上执行转换。如果SC1n[AIEN]被设置，当硬件平均完成时，ADC中断会唤醒MCU。

### 39.5.5 自动比较功能

比较功能可以被配置为检查结果是否小于或大于或等于一个单一的比较值，或者，如果结果落在由两个比较值决定的范围之内或之外。

比较模式由SC2[ACFGT]、SC2[ACREN]以及比较值寄存器（CV1和CV2）中的值决定。在对输入进行采样和转换后，CV1和CV2中的比较值会被使用，如下表所述。有六种比较模式，如下表所示。

**表39-4.比较模式**

| SC2[ACFGT] | SC2[ACREN] | CV1相对于CV2 | 职能 | 比较模式描述 |
|---|---|---|---|---|
| 0 | 0 | - | 小于阈值 | 如果结果小于CV1寄存器，则比较为真。 |
| 1 | 0 | - | 大于或等于阈值 | 如果结果大于**OR**，则比较为真<br>等于CV1寄存器。 |
| 0 | 1 | 小于或等于 | 范围之外，不包括 | 如果结果小于CV1，则比较为真。<br>**或**结果大于CV2。 |
| 0 | 1 | 大于 | 范围内，不包括在内 | 如果结果小于CV1，则比较为真。<br>**且**结果大于CV2。 |
| 1 | 1 | 小于或等于 | 内部范围，包括 | 如果结果大于或等于CV1，**并且**结果小于或等于CV2，则比较真。 |

*表继续在下一页...*

表39-4.比较模式（续）

| SC2[ACFGT] | SC2[ACREN] | CV1相对于CV2 | 职能 | 比较模式描述 |
|---|---|---|---|---|
| 1 | 1 | 大于 | 范围之外，包括 | 如果结果大于或等于CV1或者结果小于或等于CV2，则比较为真。 |

在SC2[ACREN]=1的情况下，如果CV1的值小于或等于CV2的值，那么设置SC2[ACFGT]将选择一个触发-如果-内部比较-范围包括-端点的功能。清除SC2[ACFGT]将选择一个触发-如果-外部比较-范围，不包括-端点的功能。

如果CV1大于CV2，设置SC2[ACFGT]将选择一个触发-如果-外侧比较-范围，包括端点的功能。清除SC2[ACFGT]将选择一个触发-如果-内部-比较-范围，不包括端点的功能。

如果选择的条件评估为真，SC1n[COCO]被设置。

在比较功能启用的情况下完成转换时，如果比较条件不成立，SC1n[COCO]不被设置，转换结果数据不会被传送到结果寄存器Rn。如果硬件平均化功能被启用，比较功能将平均化结果与比较值进行比较。同样的比较功能定义也适用。当SC1n[COCO]被设置且相应的ADC中断被启用时，即SC1n[AIEN]=1，会产生ADC中断。

**注意事项**

当MCU处于等待或正常停止模式时，比较功能可以监测一个通道的电压。当比较条件得到满足时，ADC中断会唤醒MCU。

## 39.5.6 校准功能

该ADC配备了一个校准机制，以提供数据表中规定的高精确度。

**注意事项**

在上电或复位后，必须对ADC进行校准。不这样做会导致ADC转换结果低于规定的精度。

为了正确校准ADC，必须做以下工作。

- 启动时，等待参考电压（VREFH）稳定下来。
- 每次系统复位后，ADC都必须重新校准。

- 一次只校准一个ADC实例。因此，在校准实例ADC0时，实例ADC1、ADC2等都需要空闲。
- 将ADCK（ADC时钟）设置为最大指定频率的一半。
- 在开始校准之前，校准寄存器（CLPS、CLP3、CLP2、CLP1、CLP0、CLPX和CLP9）必须通过向每个寄存器写入0000_0000h而被清除。
- 通过写入SC3[CAL]=1、SC3[AVGE]=1和SC3[AVGS]开始ADC校准。
  = 11b.
- 等待校准完成。这将由转换完成标志（SC1n[COCO] = 1）表示。
- 现在你可以在你的应用中以高精确度运行ADC转换。请确保重新配置ADCK的时钟速度，并将AVGE和AVGS重新配置为你所需要的设置。(最大的时钟速度和不使用硬件平均化是可能的)。

总的校准转换时间是：12×（#平均数×[采样时间（采样+1）+1个保持周期+34个比较阶段周期]）+第一次转换同步（~5个ADC周期+5个模块时钟）。

为了使你的应用板（PCB）上的ADC具有较高的精度（如数据表所规定），应满足以下要求。

- VREFH和VREFL之间的旁路帽。建议的电容尺寸。1 nF, 100 nF, 10 μF.
- 在PCB上放置电容，尽量靠近器件引脚VREFH和VREFL。
- VDDA和VSSA之间的旁路帽。建议的电容尺寸。1 nF, 100 nF, 10 μF。
- 在PCB上放置电容，尽量靠近器件引脚VDDA和VSSA。
- PCB上VDDA、VSSA、VREFH和VREFL的路由。
  - 旁路帽和MCU引脚之间的低阻抗。
  - 保持路由远离嘈杂的信号路线，如开关I/O。

### 39.5.7 用户定义的偏移功能

OFS是一个两两相补、左对齐的寄存器，包含校准产生的偏移误差修正值。

OFS中的值被减去转换，结果被转移到结果寄存器Rn中。如果结果大于最大或小于最小的结果值，它将被强制到当前操作模式的适当限制。

OFS的格式化与数据结果寄存器Rn不同，以保持校准值的分辨率，无论选择何种转换模式。在低分辨率模式下，低阶位被忽略。例如，在8位单端模式下，OFS[14:7]从D[7:0]中减去；OFS[15]表示符号（负数有效地加入到结果中），OFS[6:0]被忽略。

在自校准序列完成后，即SC3[CAL]被清除后，OFS会根据校准要求自动设置。如果需要，你可以写到OFS来覆盖校准结果。如果你写入的OFS值与校准值不同，则可能不符合ADC误差规范。在用用户指定的值覆盖之前，你应该将校准功能产生的值存储在内存中。

**注意事项**

你可以设置的偏移值有一个有效的限制。如果偏移量太高，转换的结果就会在限制值处截止。

你可以使用偏移量校准功能来消除应用偏移量或直流偏置值。USR_OFS可以用一个数字以二的互补格式写入，这个偏移量将从结果或硬件平均值中减去。要增加一个偏移量，以二的补数格式存储负偏移量，其效果将是一个加法。导致超范围值的偏移校正将被强制为最小值或最大值。单端转换的最小值是0000h。

### 39.5.8 单片机正常停止模式操作

停止模式是一种低功耗的待机模式，在此期间，MCU上的大多数或所有时钟源都被禁用。

#### 39.5.8.1 正常停止模式，启用备用时钟源

如果为转换时钟选择的备用时钟源被启用，ADC在正常停止模式下继续工作。有关该器件的配置信息，请参见特定芯片的ADC信息。

如果MCU进入正常停止模式时，转换正在进行中，它将继续进行，直到完成。当MCU处于正常停止模式时，可以通过硬件触发器或启用连续转换功能来启动转换。

如果比较和硬件平均功能被禁用，如果相应的ADC中断被启用，即SC1n[AIEN]=1时，转换完成事件会设置SC1n[COCO]并产生ADC中断，将MCU从正常停止模式唤醒。结果寄存器Rn将包含在正常停止模式下发生的第一次完成转换的数据。如果硬件平均功能被启用，SC1n[COCO]将被设置，并在选定的转换完成后产生一个中断。如果比较功能被启用，SC1n[COCO]将被设置，如果被启用，只有在满足比较条件时才会产生一个中断。如果选择了单次转换，且比较条件不成立，ADC将回到空闲状态，不能从正常停止模式唤醒MCU，除非新的转换是由另一个硬件触发器启动。

# 第40章 比较器（CMP）

## 40.1 芯片特有的CMP信息

### 40.1.1 实例化信息

该芯片有一个模拟比较器（CMP0）。对于这些器件上的DAC外部应用，不支持DAC输出。

- CMP0有自己独立的8位DAC。
- CMP0支持来自外部引脚的最多8个模拟输入。
- CMP0能够转换来自带隙的内部参考（1V参考电压）。
- CMP0支持轮流采样方案。总之，这允许CMP在STOP1和VLPS模式下独立工作，同时被周期性地触发，对多达8个输入进行采样。只有当一个输入的状态发生变化时，才会产生一个完整的唤醒。

在分段触发模式下，RTC_CLK是指LPTMR预分频器时钟，STOP是指STOP1。

关于可用的电源模式的详情，请参见模块在可用的低功率模式下的操作。

### 40.1.2 CMP输入连接

下表显示了与CMP的输入连接。

**表40-1.CMP输入连接**

| CMP输入 | CMP0 |
|---|---|
| IN0 | CMP0_IN0 |

*表继续在下一页...*

表40-1.CMP输入连接（续）

| CMP输入 | CMP0 |
|---|---|
| IN1 | CMP0_IN1 |
| IN2 | CMP0_IN2 |
| IN3 | CMP0_IN3 |
| IN4 | CMP0_IN4 |
| IN5 | CMP0_IN5 |
| IN6 | CMP0_IN6 |
| IN7 | CMP0_IN7 |

### 40.1.3 CMP的外部参考资料

CMP可以通过紧密集成的8位DAC子块获得外部参考。8位DAC子块支持选择两个参考。对于这个设备，参考值的连接方式如下。

- VDDA
- PMC带隙缓冲器输出（1V参考电压）[1]

### 40.1.4 外部窗口/样品输入

PDB和LPIT可以用来产生脉冲输出，通过TRGMUX可以作为CMP块的采样窗口。

### 40.1.5 CMP触发模式

当芯片处于STOP1或VLPS模式时，CMP和8位DAC子块支持触发模式操作。当触发模式被启用时，触发源将为CMP提供一个低功耗的时钟和触发器。触发事件将启动一个比较序列，在执行CMP操作和捕获输出之前，必须首先启用CMP和DAC。在该器件中，这两个阶段的排序控制由LPTMR提供。当LPTMR被启用时，LPTMR的触发输出总是被启用。LPTMR触发输出被用作触发事件，LPTM预分频器时钟在触发模式下作为CMP的低功率时钟。第一个信号用于启用CMP和DAC，并在TCF标志被设置的同时被断言。第二个信号的延迟是触发CMP的，以捕获TCF的结果。

1. 在VLP模式下，1V参考电压将无法使用

比较操作取决于LPTMR配置。在启用预分频器的时间计数器模式下，延迟为1/2预分频器输出周期。在预分频器旁路的时间计数器模式下，延迟为1/2预分频器源时钟周期。

来自LPTMR的第一个信号和来自LPTMR的第二个信号之间的延迟必须大于器件数据手册中定义的模拟比较器初始化延迟。

**注意事项**

当在循环模式下使用时，CMP0在比较器电路的每个周期输出一个在CMP0_RRT引脚上产生的信号。关于CMP0_RRT功能的引脚，见本参考手册所附的S32K14x_IO_Signal_Description_Input_Multiplexing.xlsx。

### 40.1.6 编程建议

以下是CMP循环操作应遵循的顺序。

1. 编程LPTMR预分频器时钟（循环时钟）。
2. 写1来清除寄存器的位域。CMP_C2[23:16]
3. 启用轮回模式 CMP_C2[RRE]

## 40.2 简介

比较器（CMP）模块提供了一个用于比较两个模拟输入电压的电路。比较器电路被设计成在电源电压的整个范围内工作，称为轨至轨操作。

模拟混合器（ANMUX）提供了一个从八个通道中选择模拟输入信号的电路。一个信号由8位数模转换器（DAC）提供。
该多路复用电路被设计为在整个电源电压范围内工作。

DAC是一个256抽头的电阻梯形网络，为需要电压基准的应用提供一个可选择的电压基准。256抽头电阻梯形网络将电源基准$V_{in}$
分成256个电压等级。一个8位数字信号输入可选择输出电压电平，该电平从$V_{in}$到$V_{in}$ /256不等。$V_{in}$ 可以从两个电压源中选择，即$V_{in1}$ 和$V_{in2}$
。来自比较器的DAC只能作为片上的内部信号，不能从外部引脚获得。

## 40.3 特点

以下各小节列出了CMP、DAC和ANMUX的特点。

### 40.3.1 CMP的特点

CMP有以下特点。

- 在整个供应范围内运行
- 输入的范围可以从轨道到轨道
- 可编程磁滞控制
- 可选择在比较器输出的上升沿、下降沿或两个上升沿或下降沿中断
- 可选择比较器输出的反转
- 有能力产生广泛的产出，如：。
  - 采样的
  - 窗口化，是某些PWM零交叉检测应用的理想选择
  - 数字过滤。
    - 可以绕过过滤器
    - 可以通过外部SAMPLE信号或按比例的总线时钟进行时钟控制
- 在输出滤波器用于内部功能的同时，可以使用外部磁滞现象
- 两个软件可选择的性能级别。
  - 以更高的功率为代价缩短传播延迟
  - 低功率，有较长的传播延迟
- 支持DMA传输
  - 可以选择一个比较事件来触发一个DMA传输
- 在该MCU上可用的所有电源模式下都能发挥作用

- 窗口和过滤功能在STOP模式下不可用
- 比较器可以被其他外设触发，只在一小部分时间内工作

### 40.3.2 8位DAC的主要特点

该DAC具有以下特点。

- 8比特分辨率
- 可选择的电源参考源
- 在不使用时，采用 "断电 "模式以节省电力
- 可选择将输出路由至内部比较器输入

### 40.3.3 ANMUX的主要特点

ANMUX具有以下特点。

- 两个8对1通道MUXes
- 在整个供应范围内运行

## 40.4 CMP、DAC和ANMUX图示

下图是高速比较器、DAC和ANMUX模块的框图。

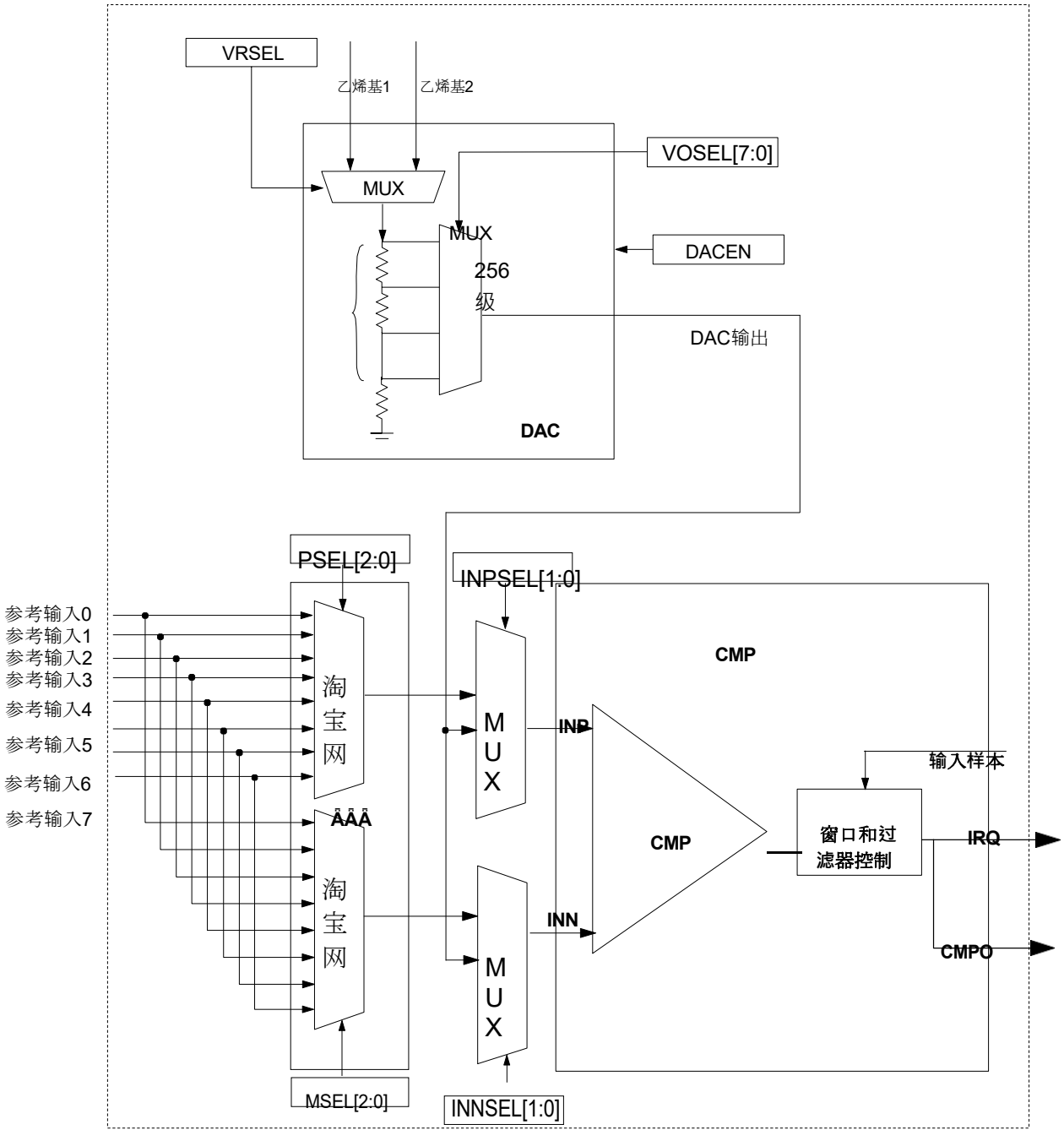



**图40-1.CMP高层图**

## 40.5 CMP框图

下图是CMP模块的框图。

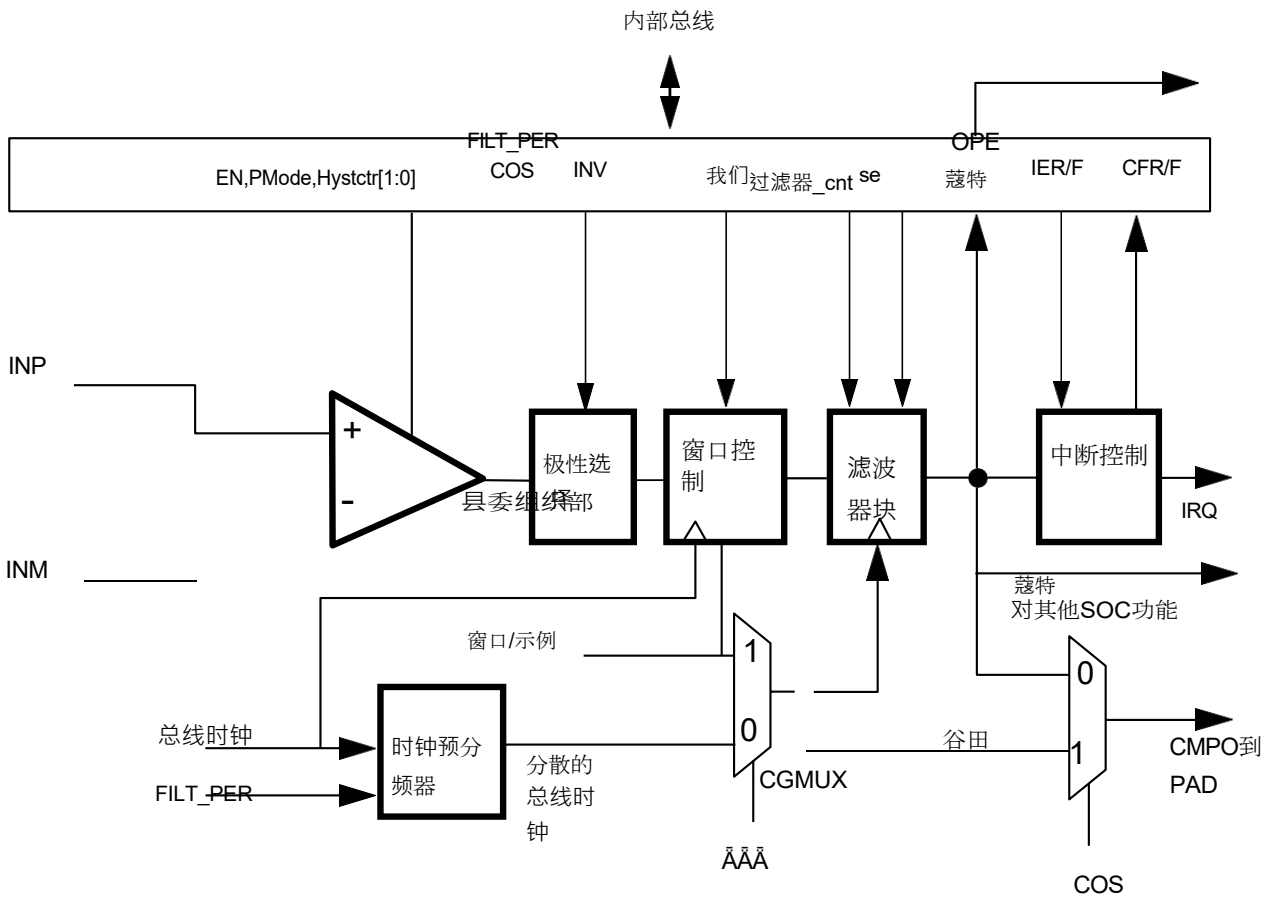



**图40-2.比较器模块框图**

在CMP框图中。

- 当C0[WE]=0时，窗口控制块被绕过。
- 如果C0[WE]=1，当WINDOW=1时，比较器的输出在每个总线时钟上被采样以产生COUTA。当WINDOW=0时，采样不发生。
- 在不使用时，过滤块被绕过。

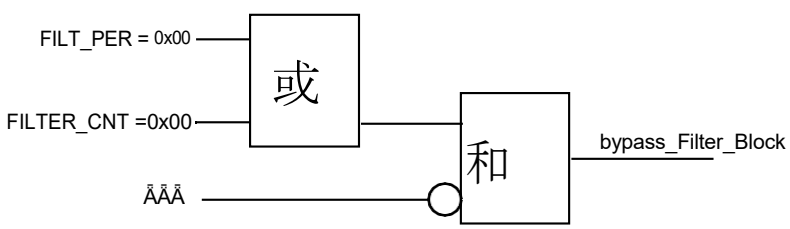



**图40-3.滤波器块的旁路逻辑**

- 如果滤波器被绕过，并且C0[FILTER_CNT]被设置为0x01，那么滤波器块就会作为一个简单的采样器。
- 当滤波器被绕过且C0[FILTER_CNT]被设置为大于0x01时，滤波器块基于

多个样本进行过滤。

- 如果C0[SE]=1，外部SAMPLE输入被用作采样时钟。
- 如果C0[SE]=0，则使用被划分的总线时钟作为采样时钟。

- 如果启用，滤波器块将在COUT上产生多达一个总线时钟的额外延迟惩罚，这是因为跨越时钟域边界的COUT必须与总线时钟重新同步。
- C0[WE]和C0[SE]是相互排斥的。
- 如果启用，滤波器时钟和采样周期必须至少比系统时钟慢4个定时器到比较器。

## 40.6 CMP引脚描述

本节提供了比较器的引脚描述。外部输入IN[7:0]事先由CMP_C1[PSEL]和CMP_C1[MSEL]复用，复用后的输出将与8位DAC的输入和其他两个内部保留的测试信号进入第二级复用，由CMP_C1[INPSEL]和CMP_C1[INNSEL]决定，第二级复用的输出将最后分别进入比较器的正负端口。

**表40-2.CMP信号描述**

| 信号 | 描述 | 输入/输出 |
|---|---|---|
| IN[7:0] | 模拟电压输入 | I |

### 40.6.1 外部引脚

CMP有两个模拟输入。INP和INM。这些引脚中的每一个都可以接受一个在MCU的整个工作范围内变化的输入电压。如果模块没有被启用，每个引脚可以作为数字输入或输出使用。请查阅具体的MCU文件，以确定与这些模拟输入共享哪些功能。

用户可以选择过滤或未过滤的比较器输出，以便在外部I/O垫上使用。

## 40.7 CMP的功能模式

CMP模块有三个主要的子块。

- 比较器本身
- 窗口功能
- 过滤功能

滤波器，C0[FILTER_CNT]，可以由内部或外部的时钟源进行计时。滤波器是可编程的，在输出的变化被记录之前，必须有多少个样本同意。在最简单的情况下，只有一个样本必须同意。在这种情况下，滤波器作为一个简单的采样器。

外部采样输入通过C0[SE]被启用。当设置时，比较器的输出只在采样输入的上升沿进行采样。

通过设置C0[WE]启用"窗口模式"。当设置时，比较器输出仅在WINDOW=1时被采样。这个功能可以用来在输入电压无效的时间段内忽略比较器的输出。这在为某些PWM应用实现零交叉检测时特别有用。

比较器的滤波和采样功能可按下表所示进行组合。下面将讨论各个模式。

**表40-3.比较器采样/过滤控制**

| 模式 # | C0[EN] | C0[WE] | C0[SE] | c0[filter_cn t] | C0[FPR] | 运作 |
|---|---|---|---|---|---|---|
| 1 | 0 | X | X | X | X | **失能**<br>参见禁用模式（#1）。 |
| 2A | 1 | 0 | 0 | 0x00 | X | **连续模式**<br>参见连续模式（#s 2A & 2B）。 |
| 2B | 1 | 0 | 0 | X | 0x00 | |
| 3A | 1 | 0 | 1 | 0x01 | X | **采样，非滤波模式**<br>参见采样、非过滤模式（#s 3A & 3B）。 |
| 3B | 1 | 0 | 0 | 0x01 | > 0x00 | |
| 4A | 1 | 0 | 1 | > 0x01 | X | **采样，过滤模式**<br>参见采样、过滤模式（#s 4A & 4B）。 |
| 4B | 1 | 0 | 0 | > 0x01 | > 0x04 | |
| 5A | 1 | 1 | 0 | 0x00 | X | **窗口模式** |
| 5B | 1 | 1 | 0 | X | 0x00 | |

表继续在下一页...

表40-3.比较器样品/过滤器控制（续）

| 模式 # | C0[EN] | C0[WE] | C0[SE] | c0[filter_cn t] | C0[FPR] | 运作 |
|---|---|---|---|---|---|---|
| | | | | | | 当SAMPLE=1时，比较器输出在每个上升的总线时钟边沿被采样，以产生COUTA。<br>见窗口模式（#s 5A & 5B）。 |
| 6 | 1 | 1 | 0 | 0x01 | 0x01-0xFF | **窗口/取样模式**<br>当SAMPLE=1时，比较器输出在每个上升的总线时钟边沿被采样，以产生COUTA，然后在由C0[FPR]决定的时间间隔内重新采样，以产生COUT。<br>见窗口/采样模式（#6）。 |
| 7 | 1 | 1 | 0 | > 0x01 | 0x01-0xFF | **窗口/过滤模式**<br>当SAMPLE=1时，比较器输出在每个上升的总线时钟边沿被采样，以产生COUTA，然后被重新采样和过滤以产生COUT。<br>见窗口/过滤模式（#7）。 |
| C0[EN]、C0[WE]、C0[SE]、C0[FILTER_CNT]和C0[FPR]的所有其他组合均为非法。 | | | | | | |

对于比较器用于驱动故障输入的情况，例如，用于电机控制模块，如FTM，它必须被配置为在连续模式下工作，以便外部故障可以立即通过比较器到目标故障电路。

**注意事项**

滤波和采样设置必须在设置C0[SE]=0、C0[FPR]=0和C0[FILTER_CNT]=0x00之后才能改变。
这将使过滤器复位到一个已知的状态。

## 40.7.1 禁用模式 (# 1)

在Disabled模式下，模拟比较器没有功能，不消耗功率。在这种模式下，CMPO为0。

## 40.7.2 连续模式（#2A & 2B）

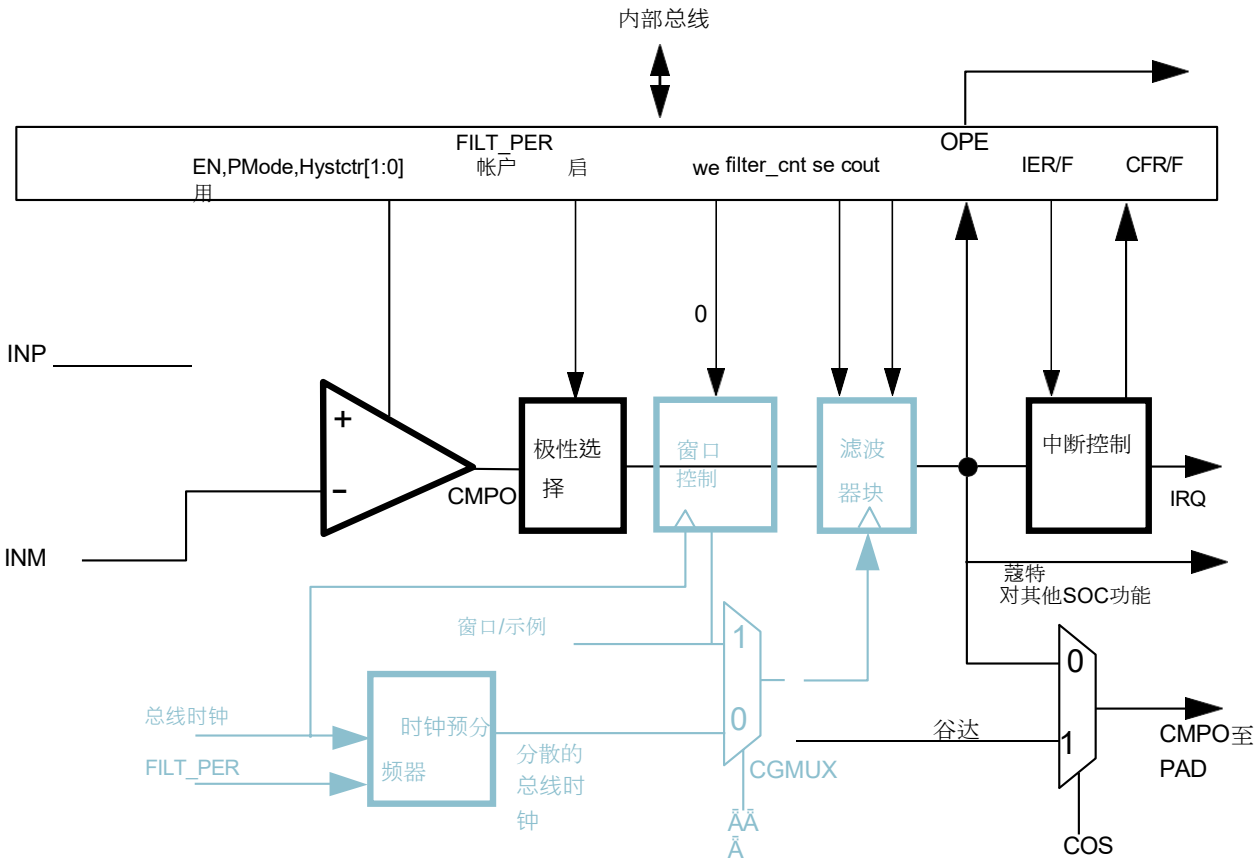



**图40-4.连续模式下的比较器操作**

### 注意事项

关于采样/窗口输入的来源，见芯片配置部分。

模拟比较器块被供电并激活。CMPO可以选择反相，但不受外部采样或过滤的影响。窗口控制和滤波块都是完全旁路的（如图中的灰色部分）。C0[COUT]被连续更新。从比较器输入引脚到输出引脚的路径是以组合式非时钟模式运行的。COUT和COUTA是相同的。

对于导致禁用过滤块的控制配置，见图40-3。

## 40.7.3 采样，非过滤模式（#s 3A & 3B）。

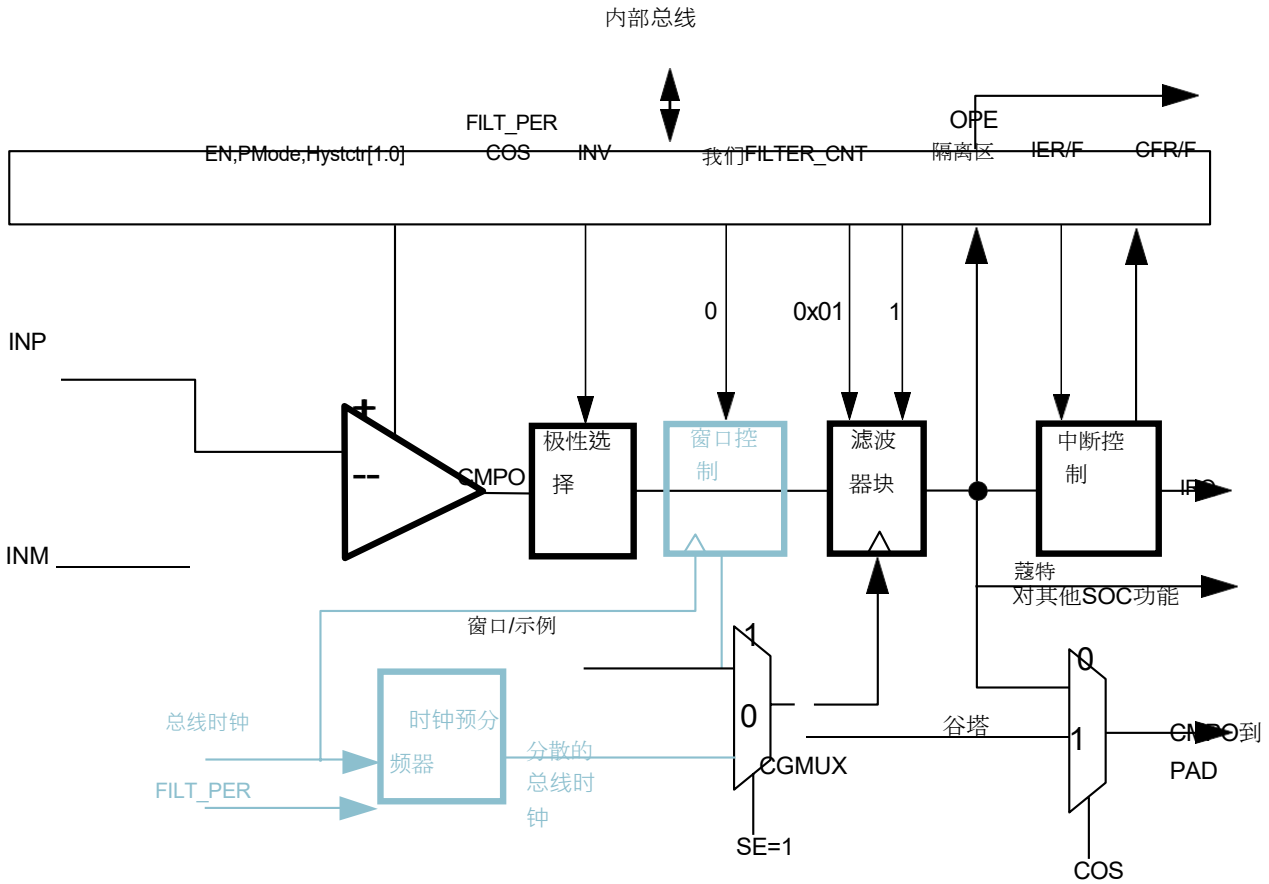



**图40-5.采样，非过滤（#3A）：采样点由外部驱动**

在采样、非滤波模式下，模拟比较器块被供电和激活。从模拟输入到COUTA的路径是组合式无时钟的。窗口控制是完全旁路的。每当检测到滤波器块时钟输入的上升沿时，COUTA就被采样。

采样、非滤波(#3A)和采样、非滤波(#3B)之间操作的唯一区别是滤波块的时钟如何产生。在#3A中，到滤波器块的时钟是外部派生的，而在#3B中，到滤波器块的时钟是内部派生的。

在这种模式下，比较器滤波器除了对比较器输出进行采样/保持外，没有其他功能（#3B）。

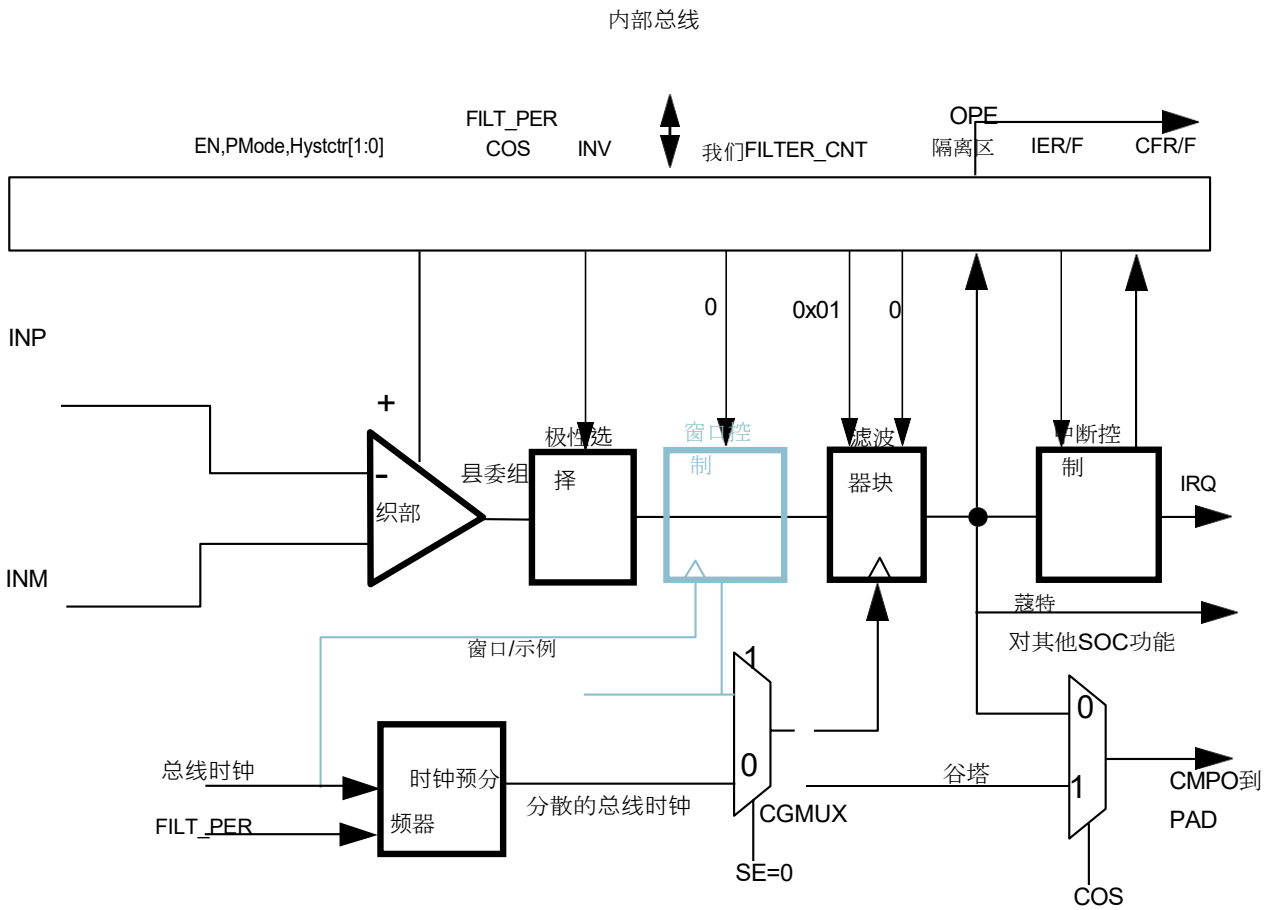



**图40-6.采样，非过滤（# 3B）：采样间隔内部得出**

## 40.7.4 采样、过滤模式（#s 4A & 4B）。

在采样、滤波模式下，模拟比较器块被供电并激活。从模拟输入到COUTA的路径是组合式无时钟的。窗口控制是完全旁路的。每当检测到滤波器块时钟输入的上升沿时，COUTA就被采样。

采样，非滤波（#3A）和采样，滤波（#4A）之间操作的唯一区别是，现在，C0[FILTER_CNT]>1，激活了滤波操作。

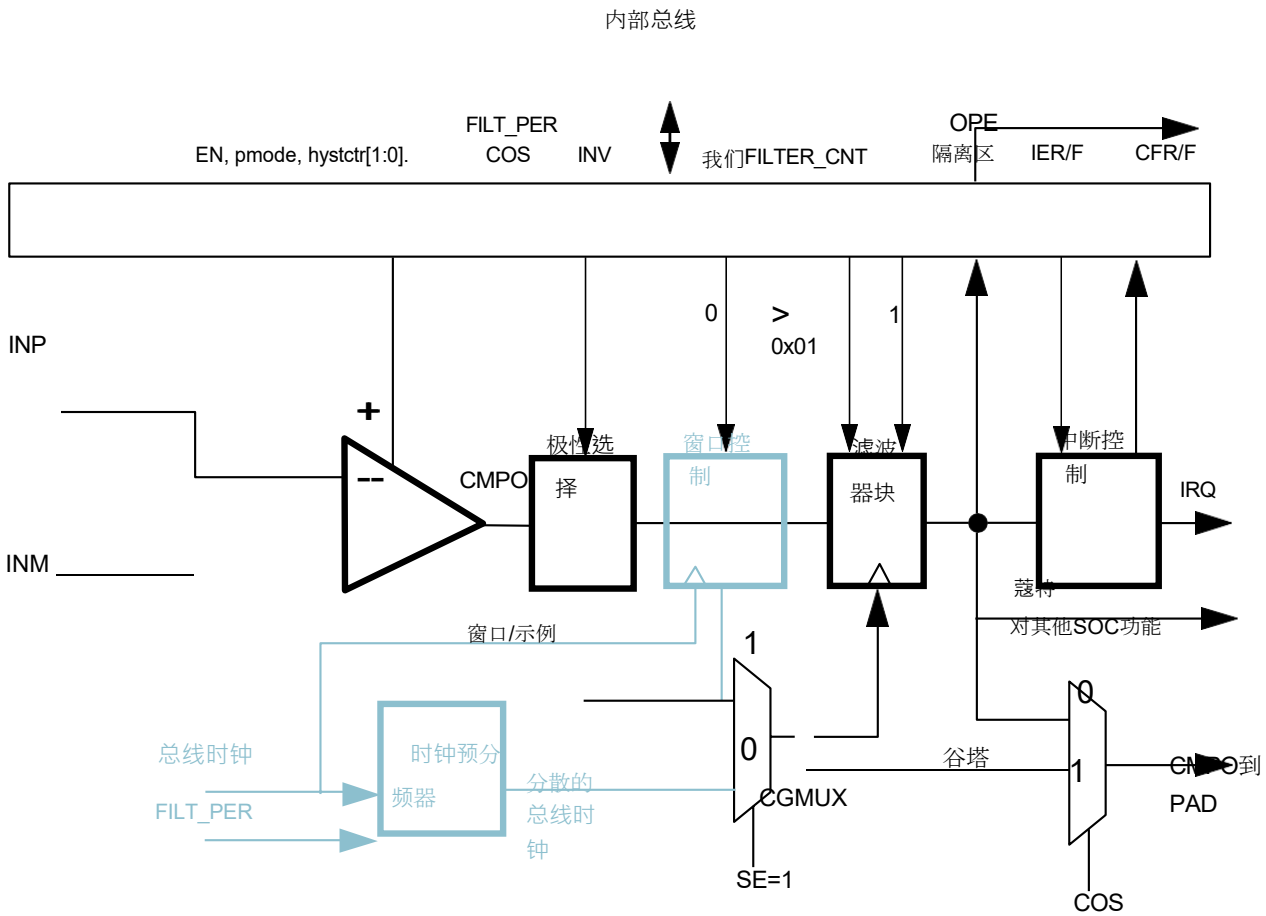



**图40-7.采样，过滤（# 4A）：采样点由外部驱动**

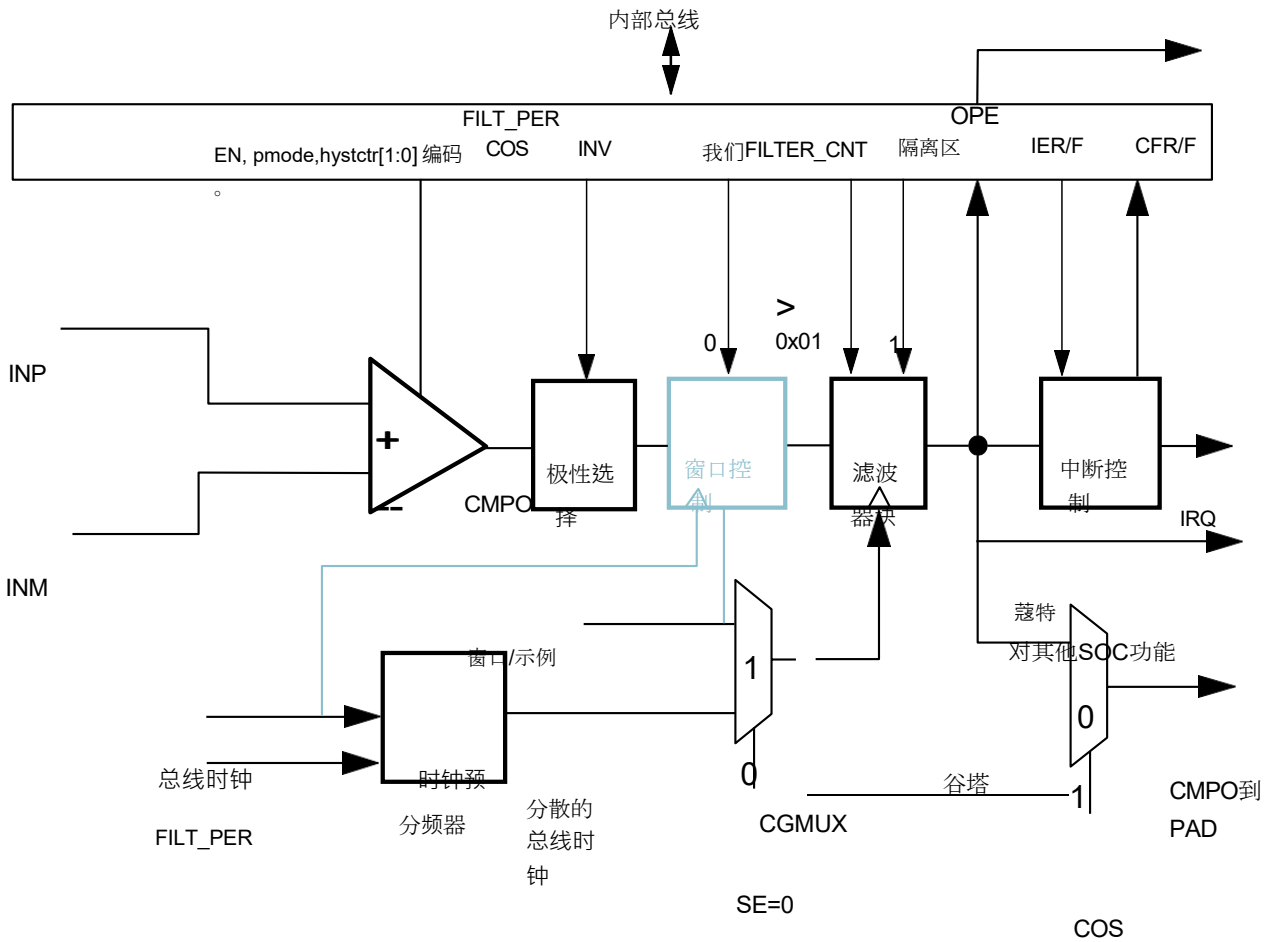



**图40-8.采样，过滤（# 4B）：采样点内部得出**

采样、非滤波（#3B）和采样、滤波（#4B）之间操作的唯一区别是，现在C0[FILTER_CNT]>1，激活了滤波操作。

## 40.7.5 窗口模式（#s 5A & 5B）

下图说明了窗口模式下的比较器操作，忽略了模拟比较器、极性选择和窗口控制块的延迟。它还假设极性选择被设置为非反转状态。

**注意事项**

只有当WINDOW信号为高电平时，模拟比较器的输出才会传给COUTA。

在实际操作中，COUTA可能会滞后模拟输入一个总线时钟周期，加上通过比较器和极性选择逻辑的组合路径延迟。

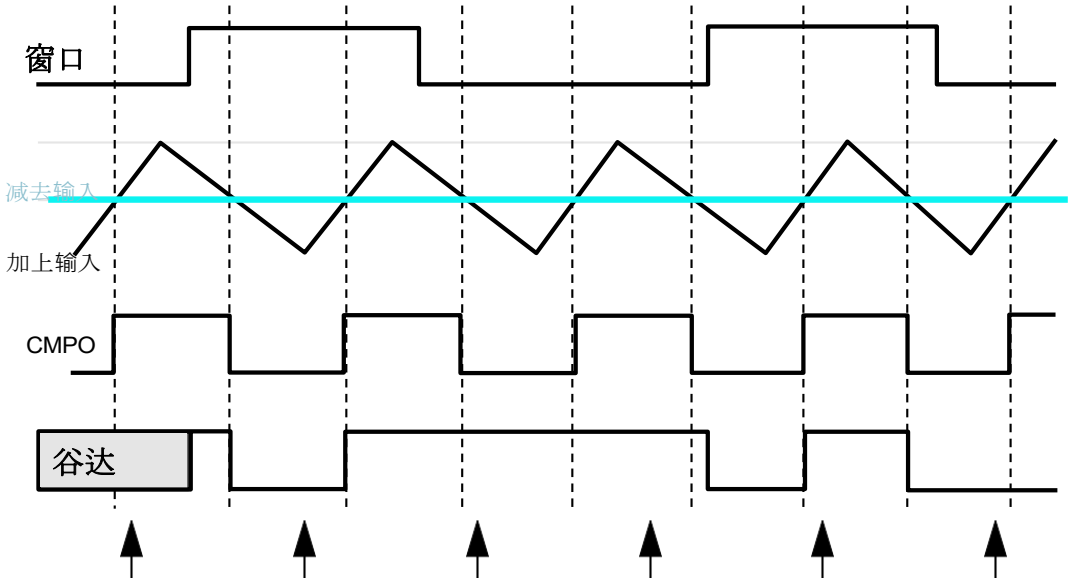



图40-9.窗口模式的时序图

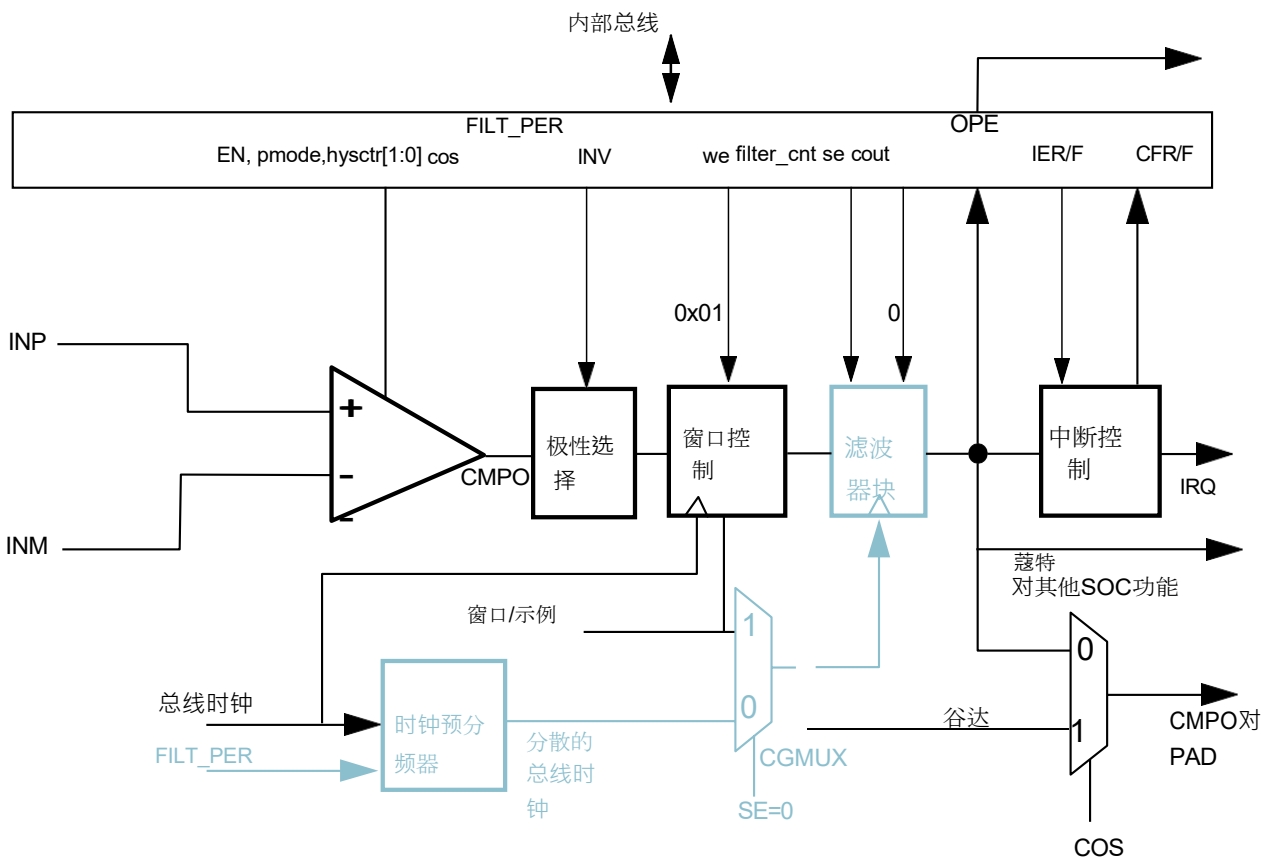



图40-10.窗口模式

对于导致禁用过滤块的控制配置，见图40-3。

当任何窗口模式处于活动状态时，只要WINDOW =

1，COUTA就会被总线时钟所控制。当WINDOW = 0时，最后的锁定结果被保持。

**注意事项**

采样输入必须在≥2.5个CMP总线时钟周期内保持高电平，以确保不会错过任何采样事件。

## 40.7.6 窗口/取样模式（#6）

下图使用了图40-9所示的相同的输入刺激，并增加了对COUTA的重新取样，以产生COUT。采样是在图中箭头所指的时间点进行的。同样，为了清楚起见，道具延迟和潜伏期被忽略了。

这个例子只是为了演示比较器在窗口化/重采样模式下的操作，并不反映任何具体的应用。取决于采样率和窗口位置，COUT可能看不到模拟比较器检测到的零交叉事件。采样周期和/或窗口位置必须为一个特定的应用仔细考虑。

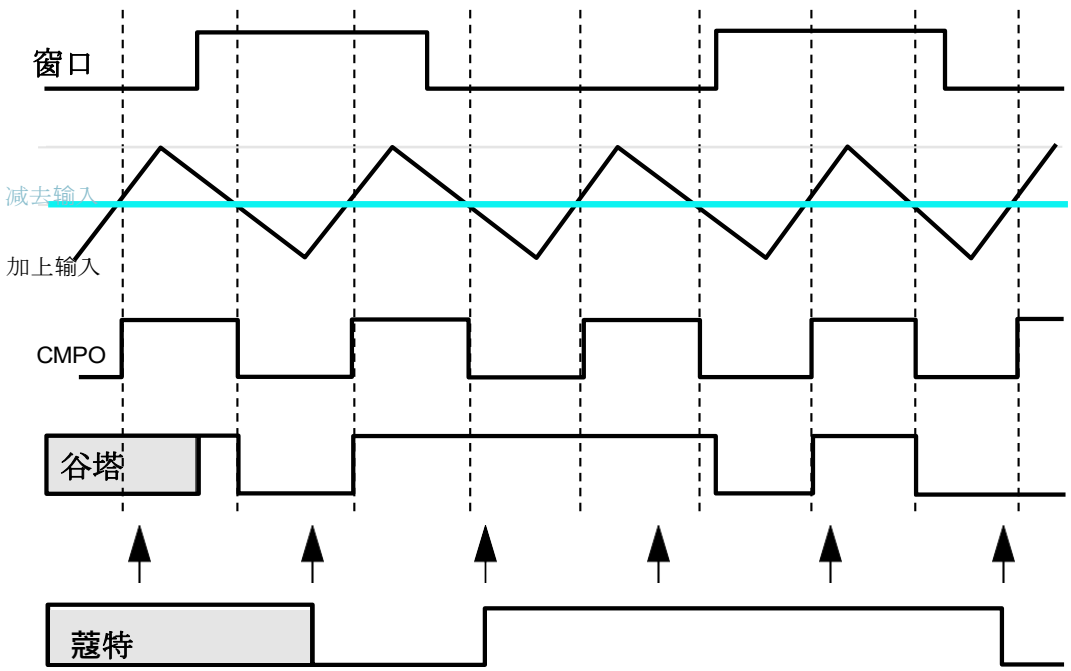



**图40-11.窗口/重采样模式操作**

这种操作模式会产生一串未经过滤的比较器样本，样本之间的间隔由FPR[FILT_PER]和总线时钟速率决定。这种模式的配置与下一节所示的窗口/过滤模式的配置几乎相同。唯一的区别是C0[FILTER_CNT]的值必须是1。

**注意事项**

采样输入必须在≥2.5个CMP总线时钟周期内保持高电平，以确保不会错过任何采样事件。

## 40.7.7 窗口/过滤模式 (#7)

这是比较器块最复杂的操作模式，因为它同时使用了窗口化和过滤功能。它的延迟也是所有模式中最高的。这可以近似为：窗口功能中最多1个总线时钟同步+[(C0[FILTER_CNT] × C0[FPR])+1] × 滤波器功能的总线时钟。

当任何窗口模式处于活动状态时，只要WINDOW = 1，COUTA就会被总线时钟所控制。当WINDOW = 0时，最后的锁定值被保留。

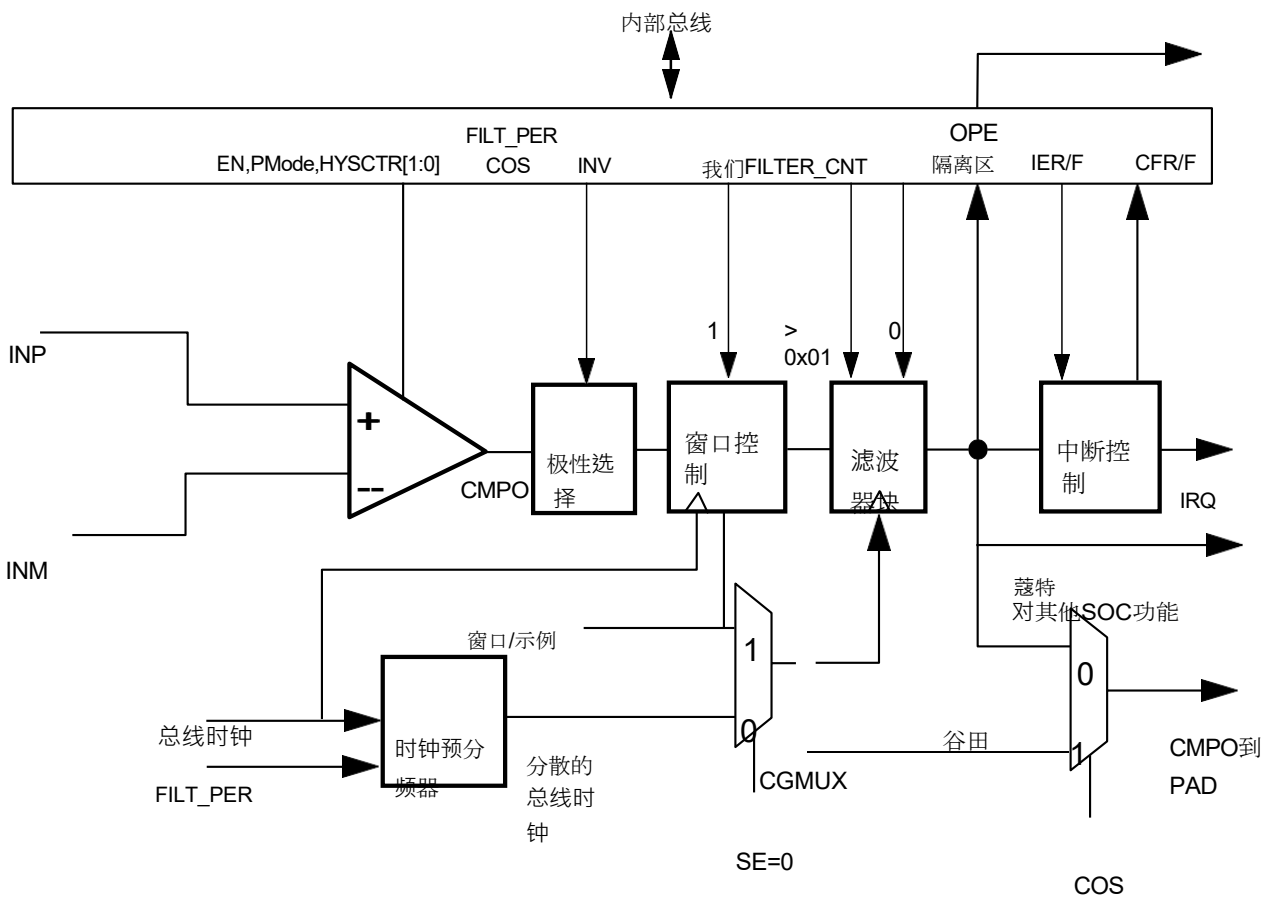



**图40-12.窗口/过滤模式**

下图显示了这种模式的操作时间，考虑到不确定性是由滤波器块的内部同步引入的。

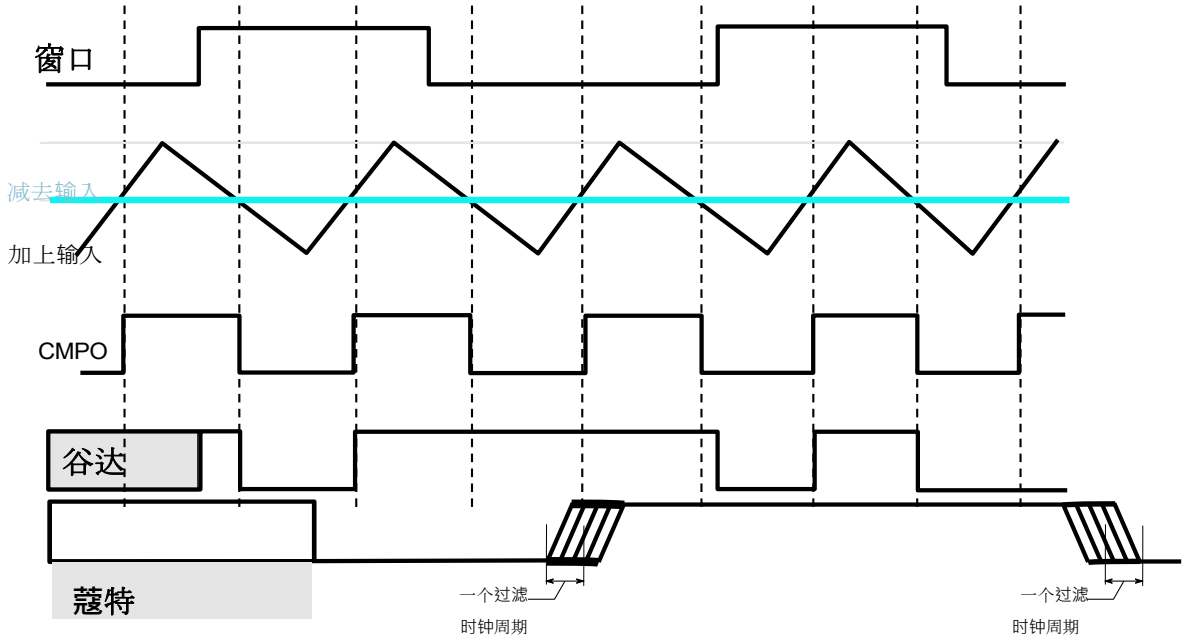



图40-13.窗口/过滤模式的操作

## 40.8 内存图/寄存器的定义

CMP内存图

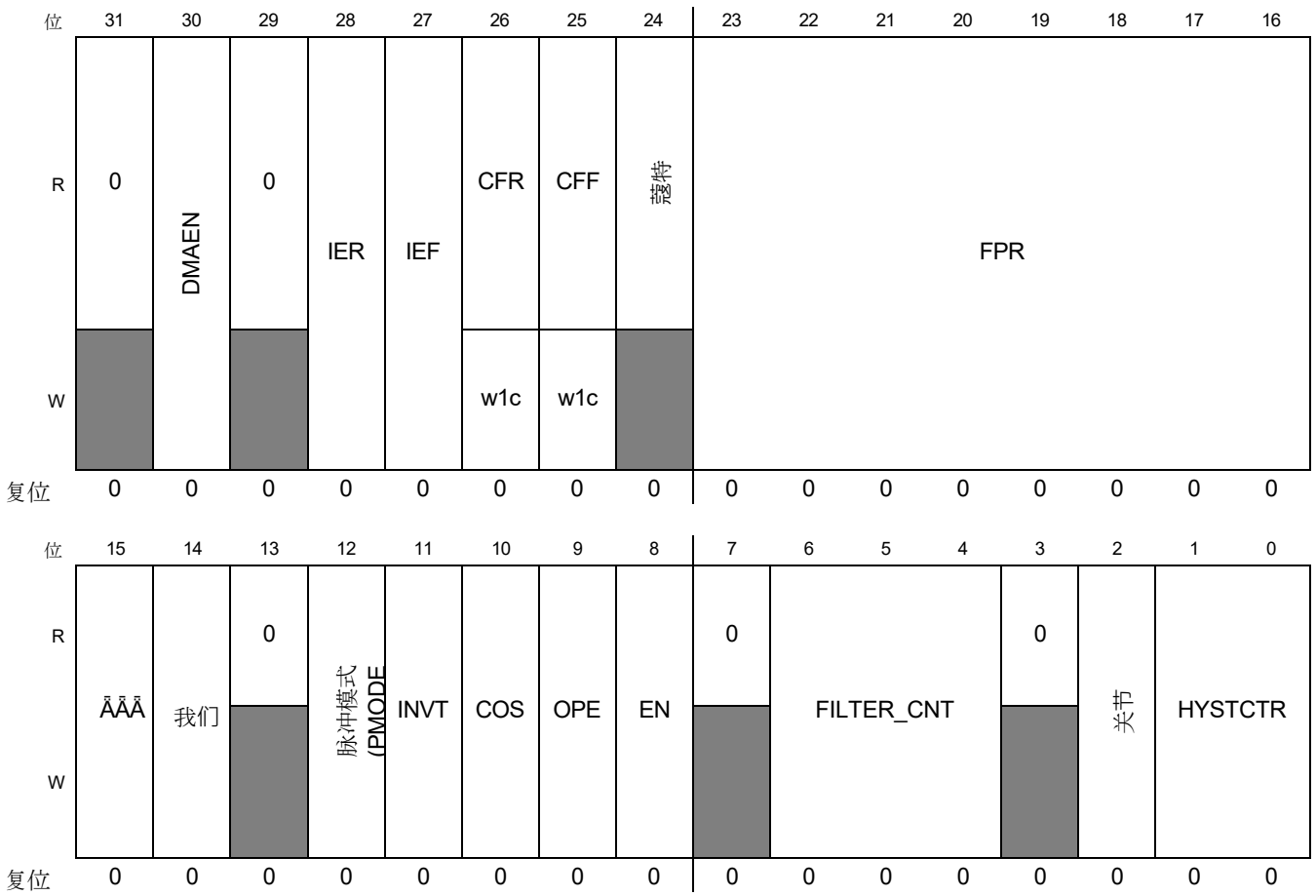
| 地址偏移（十六进制） | 注册名称 | 宽度（单位：比特） | 访问 | 重置值 | 节/页 |
|---|---|---|---|---|---|
| 0 | CMP控制寄存器0（CMP_C0）。 | 32 | R/W | 0000_0000h | 40.8.1/1075 |
| 4 | CMP控制寄存器1（CMP_C1）。 | 32 | R/W | 0000_0000h | 40.8.2/1079 |
| 8 | CMP控制寄存器2（CMP_C2）。 | 32 | R/W | 0000_0000h | 40.8.3/1082 |

### 40.8.1 CMP控制寄存器0（CMP_C0）。

访问。

- 监督员读/写
- 用户读/写

地址。0h base + 0h offset = 0h




## CMP_C0字段描述

| 场地 | 描述 |
|---|---|
| 31<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 30<br>DMAEN | 启用DMA<br>启用从CMP模块触发的DMA传输。当这个字段被设置时，一个DMA请求被<br>当CFR或CFF被设置时，就会被断言。<br>0DMA被禁用。<br>1DMA被启用。 |
| 29<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 28<br>IER | 比较器中断使能上升<br>启用来自CMP的CFR中断。当这个字段被设置时，当CFR被设置时，一个中断将被断言。<br>设置。<br>0中断被禁用。<br>1中断被启用。 |
| 27<br>IEF | 比较器中断使能下降 |

表继续在下一页...



**CMP_C0字段描述（续）**

| 场地 | 描述 |
| --- | --- |
| | 启用CMP的CFF中断。当这个字段被设置时，当CFF被设置时，一个中断将被断言。<br><br>0 中断被禁用。<br>1 中断被启用。 |
| 26<br>CFR | 模拟比较器标志上升<br><br>检测COUT的上升沿，当设置时，在正常操作期间。CFR通过写1被清除。在停止模式下，CFR对电平敏感<br><br>0 在COUT上没有检测到一个上升沿。<br>1 COUT的上升沿已经发生。 |
| 25<br>CFF | 模拟比较器标志下降<br><br>检测COUT的下降沿，当设置时，在正常操作期间。CFF通过写1被清除。在停止模式下，CFF对电平有敏感度。<br><br>0 在COUT上没有检测到一个下降沿。<br>1 COUT的下降沿已经发生。 |
| 24<br>蔻特 | 模拟比较器输出<br><br>读取时，返回模拟比较器输出的当前值。当模拟比较器模块被禁用时，即C0[EN]=0时，该字段被重置为0，并将读作C0[INVT]。 对该字段的写入被忽略。 |
| 23-16<br>FPR | 滤波器的采样周期<br><br>当C0[SE]=0时，指定比较器输出滤波器的采样周期，以总线时钟周期为单位。 将FPR设置为0x0，可禁用该滤波器。滤波器编程和延迟细节在CMP功能描述中提供。当C0[SE]=1时，这个字段没有影响。 在这种情况下，外部SAMPLE信号被用来确定采样周期。 |
| 15<br>ÅÅÅ | 启用样本<br><br>在任何时候，SE或WE都可以被设置。如果对这个寄存器的写入试图同时设置，那么SE被设置，WE被清零。然而，要避免向两个位都写1，因为这种 "11 "的情况是保留的。<br><br>0 没有选择采样模式。<br>1 采样模式被选中。 |
| 14<br>我们 | 启用窗口<br><br>在任何时候，SE或WE都可以被设置。如果对这个寄存器的写入试图同时设置，那么SE被设置，WE被清零。然而，要避免向两个位都写1，因为这种 "11 "的情况是保留的。<br><br>0 没有选择窗口模式。<br>1 窗口模式被选中。 |
| 13<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |

**CMP_C0字段描述（续）**

| | |
|---|---|
| 12<br>脉冲模式<br>(PMODE) | 电源模式选择<br><br>0 低速（LS）比较模式被选中。<br>1 选择高速（HS）比较模式，在VLPx模式下，或停止模式切换到低速（LS）模式。 |

*表继续在下一页...*

## CMP_C0字段描述（续）

| 场地 | 描述 |
|---|---|
| 11<br>INVT | 比较器反转<br><br>该位允许选择模拟比较器功能的极性。当C0[OPE]=0时，它也被驱动到COUT输出（在器件引脚和作为C0[COUT]）。<br><br>0 不反转比较器的输出。<br>1 反转比较器的输出。 |
| 10<br>COS | 比较器输出选择<br><br>0 将CMPO设置为等于COUT（过滤的比较器输出）。<br>1 将CMPO设置为等于COUTA（未经过滤的比较器输出）。 |
| 9<br>OPE | 比较器输出引脚启用<br><br>OPE位启用了从比较器输出到选定引脚的路径。<br><br>0 当OPE为0时，比较器的输出（在取决于软件配置的窗口/滤波器设置后）不可用于封装的引脚。<br>1 当OPE为1时，如果软件将比较器配置为拥有一个封装的引脚，则比较器可在一个封装的引脚中使用。 |
| 8<br>EN | 比较器模块启用<br><br>EN位启用模拟比较器模块。当该模块未被启用时，模拟部分保持在关闭状态，不消耗任何功率。<br><br>0 模拟比较器被禁用。<br>1 模拟比较器被启用。 |
| 7<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 6-4<br>FILTER_CNT | 筛选样本数<br><br>这个字段规定了在比较器输出滤波器接受新的输出状态之前，必须同意的连续样本的数量。有关滤波器编程和延迟的信息，请参见功能描述。<br><br>00 滤波器被禁用。如果SE=1，那么COUT是一个逻辑零（这不是一个合法的状态，不推荐）。如果SE = 0，COUT = COUTA。<br>01 1个连续的样本必须一致（比较器的输出是简单的采样）。0102个连续样本必须一致。<br>0113个连续样本必须一致。<br>100 4个连续的样本必须一致。<br>101 5个连续的样品必须一致。<br>110 6个连续的样本必须一致。<br>111 7个连续的样本必须一致。 |

CMP_C0字段描述（续）

| | |
|---|---|
| 3<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 2<br>关节 | 比较器硬块偏移控制。参见芯片数据表以获得每个级别的实际偏移值 |

*表继续在下一页...*

| 场地 | 描述 |
|---|---|
| | **注意。** - 如果OFFSET=1，那么在INP正方向跨越INN（或INN负方向跨越INP）的情况下，将没有滞后现象。对于INP在下降方向上跨越INN的情况，仍然存在一个半滞后值。<br>• 如果OFFSET=0，那么由HYSTCTR选择的磁滞对两个方向都有效。<br><br>0 比较器硬块输出内部有0级偏移。<br>1 比较器硬块输出内部有1级偏移。 |
| HYSTCTR | 比较器硬块滞后控制。参见芯片数据表以获得每个级别的实际滞后值<br><br>00硬块输出内部有0级磁滞。01硬块输出内部有1级迟滞。<br>10 硬块输出内部有2级迟滞。<br>11 硬块输出内部有3级迟滞。 |

## 40.8.2 CMP控制寄存器1（CMP_C1）。

访问。

- 监督员读/写
- 用户读/写

地址。0h基数+4h偏移量=4h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | 0 | INPSEL | | 0 | ÃÃÃ | | CHN7 | CHN6 | CHN5 | 辽宁沈阳 | CHN3 | 辽宁省 | CHN1 | CHN0 |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 位 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | DACEN | VRSEL | PSEL | | | MSEL | | | VOSEL | | | | | | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**CMP_C1字段描述**

| 场地 | 描述 |
|---|---|
| 31-30<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 29<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |

表继续在下一页...



**CMP_C0字段描述（续）**

## CMP_C1字段描述（续）

| 场地 | 描述 |
|---|---|
| 28-27<br>INPSEL | 选择比较器的正端口的输入<br><br>决定哪个输入被选为比较器的加号输入。<br><br>注意：这些选择是用来选择比较器的最终正向输入。<br>注意：对于轮流工作模式，CMPx_C1寄存器中的MSEL和PSEL位域必须是有不同的价值。<br><br>00IN0 ，来自8位DAC的输出<br>01IN1 ，来自模拟8-1混频器<br>10 保留<br>11 保留 |
| 26<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 25-24<br>ÃÃÃ | 选择比较器负口的输入端<br><br>决定哪个输入被选为比较器的加号输入。<br><br>注意：这些选择是用来选择比较器的最终负输入。<br>注意：对于轮流工作模式，CMPx_C1寄存器中的MSEL和PSEL位域必须是有不同的价值。<br><br>00IN0 ，来自8位DAC的输出<br>01IN1 ，来自模拟8-1混频器<br>10 保留<br>11 保留 |
| 23<br>CHN7 | 通道7输入启用<br><br>循环检查器的输入启用通道7。如果CHN7被设置，那么在循环模式下，非固定复用端口的相应通道被启用以检查其电压值。如果选择同一通道作为参考电压，则该位没有影响。 |
| 22<br>CHN6 | 通道6输入启用<br><br>循环检查器的输入启用通道6。如果CHN6被设置，那么在循环模式下，非固定复用端口的相应通道被启用以检查其电压值。如果选择同一通道作为参考电压，则该位没有影响。 |
| 21<br>CHN5 | 通道5输入启用<br><br>循环检查器的输入通道5启用。如果CHN5被设置，那么在循环模式下，非固定复用端口的相应通道被启用以检查其电压值。如果选择同一通道作为参考电压，则该位没有影响。 |
| 20<br>辽宁沈阳 | 通道4输入启用<br><br>循环检查器的输入通道4启用。如果CHN4被设置，那么非固定多路复用端口的相应通道被启用，在轮流模式下检查其电压值。如果选择同一通道作为参考电压，则该位没有影响。 |
| 19<br>CHN3 | 通道3输入启用<br><br>循环检查器的输入通道3启用。如果CHN3被设置，那么在循环模式下，非固定复用端口的相应通道被启用以检查其电压值。如果选择同一通道作为参考电压，则该位没有影响。 |

*表继续在下一页...*

## CMP_C1寄存器描述（续）

| 场地 | 描述 |
|---|---|
| 18<br>辽宁省 | 通道2输入启用<br><br>循环检查器的输入启用通道2。如果CHN2被设置，那么在循环模式下，非固定复用端口的相应通道被启用以检查其电压值。如果选择同一通道作为参考电压，则该位没有影响。 |
| 17<br>CHN1 | 通道1输入启用<br><br>循环检查器的输入启用通道1。如果CHN1被设置，那么在循环模式下，非固定复用端口的相应通道被启用以检查其电压值。如果选择同一通道作为参考电压，则该位没有影响。 |
| 16<br>CHN0 | 通道0输入启用<br><br>循环检查器输入的通道0启用。如果CHN0被设置，那么在循环模式下，非固定复用端口的相应通道被启用以检查其电压值。如果选择同一通道作为参考电压，则该位没有影响。 |
| 15<br>DACEN | 启用DAC<br><br>该位用于启用DAC。当DAC被禁用时，它被断电以节省电力。<br><br>0DAC被禁用。<br>1DAC被启用。 |
| 14<br>VRSEL | 电源电压参考源选择<br><br>0 Vin1被选为电阻梯形网络供电参考Vin。<br><br>1 Vin2被选为电阻梯形网络供电参考Vin。 |
| 13-11<br>PSEL | 加上输入MUX控制<br><br>决定哪个输入被选为加法多路复用。<br>注意：这些位用于选择加法多路复用的外部8个输入，实际输入到<br>比较器的正端口在这个多路复用输出和其他输入之间进行选择，最后见定义。<br>在INPSEL。<br>注意：对于轮流工作模式，CMPx_C1寄存器中的MSEL和PSEL位域必须是<br>有不同的价值。<br><br>000 IN0<br>001 IN1<br>010 IN2<br>011 IN3<br>100 IN4<br>101 IN5<br>110 IN6<br>111 IN7 |
| 10-8<br>MSEL | 负数输入MUX控制<br><br>决定哪一个输入被选择为减法多路复用。<br><br>注意：这些位用于选择负数复用的外部8个输入，比较器负口的实际输入是在这个复用输出和其他输入之间最后选择的，见INNSEL中的定义。<br><br>注意：对于轮流工作模式，CMPx_C1寄存器中的MSEL和PSEL位域必须有不同的值。 |

*表继续在下一页...*

**CMP_C1字段描述（续）**

| 场地 | 描述 |
|---|---|
| | 00 IN0<br>01 IN1<br>10 IN2<br>11 IN3<br>100 IN4<br>101 IN5<br>110 IN6<br>111 IN7 |
| VOSEL | DAC输出电压选择<br><br>这个位从256个不同的电平中选择一个输出电压。DACO = (Vin/256) * (VOSEL[7:0] + 1)，所以DACO的范围是Vin/256到Vin。 |

## 40.8.3 CMP控制寄存器2（CMP_C2）。

访问。

- 监督员读/写
- 用户读/写

地址。0h基数+8h偏移量=8h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | RRE | RRIE | FXMP | 0 | FXMXCH | | | 0 | CH7F | CH6F | CH5F | CH4F | CH3F | CH2F | CH1F | CH0F |
| W | | | | | | | | | w1c | w1c | w1c | w1c | w1c | w1c | w1c | w1c |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 位 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | ÅÅÅ | | 适应性模型（ | | | | | | ACOn | | | | | | | |
| W | | | INITMOD | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

## CMP_C2字段描述

| 场地 | 描述 |
|---|---|
| 31<br>RRE | 循环启用<br><br>该位启用轮流操作。<br><br>0 循环操作被禁用。<br>1 启用了轮流操作。 |
| 30<br>RRIE | 启用Round-Robin中断<br><br>当某一通道的比较结果发生变化时，该位启用中断/唤醒。<br><br>0 循环中断被禁用。<br>1 当比较结果与上一个样本发生变化时，轮回中断被启用。 |
| 29<br>FXMP | 固定MUX端口<br><br>该位用于固定轮流模式的模拟多路复用端口。<br><br>0 加号端口是固定的。在每一轮中，只有减号端口的输入被扫描。<br>1 减号端口是固定的。在每一轮中，只有加号端口的输入被扫描。 |
| 28<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 27-25<br>FXMXCH | 固定频道选择<br><br>该字段表示在给定的轮流模式下，多路复用端口中的哪个通道是固定的。000 通道0被<br><br>选择为固定多路复用端口的固定参考输入。<br><br>001 通道1被选为固定多路复用端口的固定参考输入。010 通道2被选为固定多<br><br>路复用端口的固定参考输入。011 通道3被选为固定多路复用端口的固定参考输<br><br>入。<br>100 通道4被选为固定多路复用端口的固定参考输入。<br>101 通道5被选为固定多路复用端口的固定参考输入。<br>110 通道6被选为固定多路复用端口的固定参考输入。<br>111 通道7被选为固定多路复用端口的固定参考输入。 |
| 24<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 23<br>CH7F | 通道7输入改变标志。如果通道7的输入与固定多路复用端口的最后一次比较发生了变化，该位被设置。 |
| 22<br>CH6F | 通道6输入改变标志。如果通道6的输入从上次与固定多路复用端口的比较中发生了变化，则该位被设置。 |
| 21<br>CH5F | 通道5输入改变标志。如果通道5的输入从上次与固定多路复用端口的比较中发生了变化，则该位被设置。 |
| 20<br>CH4F | 通道4输入改变标志。如果通道4的输入与固定多路复用端口的最后一次比较发生了变化，则该位被设置。 |
| 19<br>CH3F | 通道3输入改变标志。如果通道3的输入从上次与固定多路复用端口的比较中发生了变化，则该位被设置。 |

**CMP_C1字段描述（续）**

| | |
|---|---|
| 18<br>CH2F | 通道2输入改变标志。如果通道2的输入从上次与固定多路复用端口的比较中发生了变化，则该位被设置。 |
| 17<br>CH1F | 通道1输入改变标志。如果通道1的输入从最后一次与固定多路复用端口的比较中发生变化，则该位被设置。 |

*表继续在下一页...*

CMP_C2字段描述（续）

| 场地 | 描述 |
| --- | --- |
| 16<br>CH0F | 通道0输入改变标志。如果通道0的输入从上次与固定多路复用端口的比较中发生了变化，则该位被设置。 |
| 15-14<br>ÃÃÃ | 样本时钟的数量<br><br>对于一个给定的通道，这个字段指定了采样发生在多少个轮回时钟周期之后。<br><br>00当活动通道在一个循环时钟中被扫描后，比较结果就被采样。01　　采样发生在循环时钟的下一个循环周期之后的一个循环时钟周期。<br>10　采样发生在轮转时钟的下一个周期之后的2个轮转时钟周期。<br>11　采样发生在轮转时钟的下一个周期之后的3个轮转时钟周期。 |
| 13-8<br>适应性模型（<br>INITMOD | 比较器和DAC初始化延迟模数。<br><br>这些值指定了用于确定数据手册所规定的比较器和DAC初始化延迟的循环时钟周期。例如，初始化延迟是80us，循环时钟是100kHz，那么INITMOD应该被设置为80us/10us=8。<br><br>000000模数被设置为64（与111111相同）。<br>其他值 苯胺化延迟被设置为INITMOD *轮回时钟周期 |
| ACOn | 通道*n*的输入比较结果。这个字段存储了输入通道*n*与固定mux端口的最新比较结果。读取此位会返回最新的比较结果。写入这个字段定义了通道*n*的预设状态。 |

## 40.9 CMP功能描述

CMP模块可用于比较应用于INP和INM的两个模拟输入电压。当非反相输入大于反相输入时，CMPO为高电平，当非反相输入小于反相输入时，为低电平。这个信号可以通过设置C0[INVT]=1来选择性地反相。

C0[IER]和C0[IEF]用于选择导致CMP模块向处理器发出中断指令的条件。C0[CFF]在下降沿被设置，而C0[CFR]在比较器输出的上升沿被设置。可以通过C0[COUT]直接读取可选择的过滤的CMPO。

### 40.9.1 初始化

一个典型的启动顺序如下。

稳定COUT所需的时间是比较器的通电延迟加上从选定的模拟源通过模拟比较器、开窗功能和滤波器的最大传播延迟。关于比较器的上电延迟，请参见数据手册。开窗功能有一个最大的总线时钟周期延迟。滤波器的延迟在低通滤波器部分指定。

在操作过程中，必须始终考虑所选数据路径的传播延迟。COUT和C0[CFR]/C0[CFF]可能需要许多总线时钟周期来反映输入变化或数据路径中涉及的一个组件的配置变化。

当为滤波模式编程时，COUT最初等于0，直到有足够的时钟周期来填充滤波器的所有阶段。即使COUTA为逻辑1，也会发生这种情况。

### 40.9.2 低通滤波器

低通滤波器在未经过滤的、非同步的、可选择反转的比较器输出COUTA上运行，并产生过滤的、同步的输出COUT。COUTA和COUT都可以被配置为模块输出，在系统中用于不同的目的。

同步和边缘检测总是被用来确定状态寄存器的位值。它们也适用于所有采样和窗口模式的COUT。滤波可以使用由FPR[FILT_PER]定义的内部时间基，或者使用外部SAMPLE输入来确定采样时间。

对数字滤波的需求和滤波量取决于用户要求。在没有外部磁滞电路的情况下，滤波会变得更加有用。如果没有外部滞后，当选择的INM和INP输入电压相差小于差分比较器的偏移电压时，就会在COUTA产生高频振荡。

#### 40.9.2.1 启用过滤模式

过滤模式可以通过以下方式启用。

- 设置C0[FILTER_CNT] > 0x01和
- 将C0[FPR]设置为非零值或设置C0[SE]=1

如果使用划分的总线时钟来驱动滤波器，它每隔C0[FPR]总线时钟周期对COUTA进行采样。

滤波器的输出在第一次启动时为逻辑0，随后当所有连续的C0[FILTER_CNT]样本同意输出值改变时，输出就会改变。换句话说，C0[COUT]在某个初始阶段是0，即使COUTA处于逻辑1。

将C0[SE]、C0[FPR]和C0[FILTER_CNT]全部设置为0，可以禁用滤波器，消除与滤波过程有关的开关电流。

**注意事项**

在对过滤器参数做任何改变之前，一定要切换到这个设置。这将使过滤器复位到一个已知的状态。
在没有这个中间步骤的情况下，即时切换C0[FILTER_CNT]可能会导致意外的行为。

如果C0[SE]=1，滤波器在采样输入的每个正向转换时对COUTA采样。当所有连续的C0[FILTER_CNT]样本同意输出值改变时，滤波器的输出状态就会改变。

### 40.9.2.2 延迟问题

C0[FPR]或SAMPLE周期的值必须被设定为使采样周期刚好长于预期噪声的周期。这样，一个噪声尖峰将只破坏一个样本。C0[FILTER_CNT]的值必须被选择，以减少噪音样本导致不正确过渡被识别的概率。不正确转换的概率被定义为不正确样本的概率提高到C0[FILTER_CNT]的幂。

C0[FPR]或SAMPLE周期和C0[FILTER_CNT]的值也必须与识别实际比较器输出转换的最小延迟的愿望进行权衡。在额定延迟内检测到实际输出变化的概率是正确采样的概率提高到C0[FILTER_CNT]的功率。

下表总结了在*没有噪声的情况下*各种工作模式的最大延迟值。每次实际输出转换被噪声掩盖时，过滤延时都会重新开始。

**表40-4.比较器采样/过滤的最大延迟**

| 模式 # | C0[EN] | C0[WE] | C0[SE] | c0[过滤器_cnt] | Co[FPR] | 运作 | 最大延时1 |
|---|---|---|---|---|---|---|---|
| 1 | 0 | X | X | X | X | 失能 | 不适用 |
| 2A | 1 | 0 | 0 | 0x00 | X | 连续模式 | TPD |
| 2B | 1 | 0 | 0 | X | 0x00 | | |
| 3A | 1 | 0 | 1 | 0x01 | X | 采样，非滤波模式 | TPD + TSAMPLE + Tper |

*表继续在下一页...*

表40-4.比较器采样/过滤的最大延迟（续）

| 模式 # | C0[EN] | C0[WE] | C0[SE] | c0[过滤器_cnt] | Co[FPR] | 运作 | 最大延时1 |
|---|---|---|---|---|---|---|---|
| 3B | 1 | 0 | 0 | 0x01 | > 0x00 | | $t_{PD}$ + (c0[fpr] * $t_{per}$ ) + $t_{per}$ |
| 4A | 1 | 0 | 1 | > 0x01 | X | 采样，过滤模式 | $T_{PD}$ + (C0[FILTER_CNT] * $_{TSAMPLE}$) + Tper |
| 4B | 1 | 0 | 0 | > 0x01 | > 0x00 | | $t_{PD}$ + (c0[filter_cnt] * C0[FPR] x $T_{per}$ ) + $T_{per}$ |
| 5A | 1 | 1 | 0 | 0x00 | X | 窗口模式 | $_{TPD}$ + Tper |
| 5B | 1 | 1 | 0 | X | 0x00 | | $_{TPD}$ + Tper |
| 6 | 1 | 1 | 0 | 0x01 | 0x01 - 0xFF | 窗口化/重采样模式 | $t_{PD}$ + (c0[fpr] * $t_{per}$ ) + 2Tper |
| 7 | 1 | 1 | 0 | > 0x01 | 0x01 - 0xFF | 窗口化/过滤化模式 | $t_{PD}$ + (c0[filter_cnt] * C0[FPR] x $T_{per}$ ) + 2$T_{per}$ |

1.$T_{PD}$ 代表模拟组件的内在延迟加上极性选择逻辑。$_{TSAMPLE}$是外部采样时钟的时钟周期。$T_{per}$ 是总线时钟的周期。

## 40.10 中断

CMP模块能够在比较器输出的上升沿或下降沿或两者同时产生一个中断。假设CMP的DMA使能位没有被设置，下表给出了中断请求被断言和反断言的条件。

表40-5.CMP中断代数

| 当 | 那么 |
|---|---|
| C0[IER]和C0[CFR]被设置为 | 中断请求被断言 |
| C0[IEF]和C0[CFF]被设置为 | 中断请求被断言 |
| C0[IER]和C0[CFR]在上升沿中断中被清零 | 中断请求被取消 |
| C0[IEF]和C0[CFF]在下降沿中断中被清零 | 中断请求被取消 |

## 40.11 支持DMA

通常情况下，如果COUT上有变化，CMP会产生一个CPU中断。当通过设置C0[DMAEN]启用DMA支持，并通过设置C0[IER]、C0[IEF]或两者启用中断时，COUT上的相应变化会迫使DMA传输请求而不是CPU中断。当DMA完成了传输后，它

发送一个传输完成指示信号，取消DMA传输请求并清除标志，以允许比较器输出的后续变化发生并强制执行另一个DMA请求。

## 40.12 DAC功能描述

本节提供了DAC的功能描述。

### 40.12.1 数字-模拟转换器框图

下图是DAC模块的框图。它包含一个256抽头的电阻梯形网络和一个256对1的复用器，它从DACO输出的256个不同电平中选择一个输出电压。它是通过控制寄存器1（CMP_C1）控制的。它的电源参考源可以从两个来源$V_{in1}$和$V_{in2}$中选择。该模块在不使用时可以断电或禁用。当处于禁用模式时，DACO被连接到模拟地。

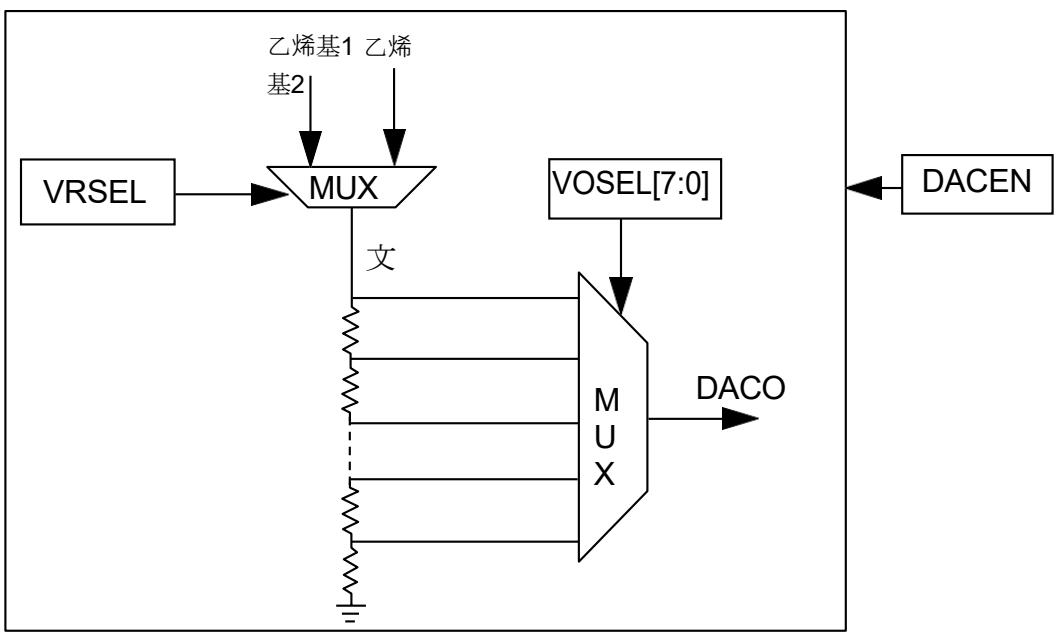



**图40-14.8位DAC框图**

### 40.12.2 电压参考源选择

- $V_{in1}$， 必须用于连接到初级电压源，作为64个分接电阻梯形的电源参考。
- $V_{in2}$， 必须用于连接备用电压源，如果没有备用电压源，则用于连接主源。

## 40.13 DAC复位

该模块有一个单一的复位输入，对应于整个芯片的外设复位。

## 40.14 DAC时钟

该模块有一个单一的时钟输入，即总线时钟。

## 40.15 DAC中断

该模块没有中断。

## 40.16 触发模式

CMP和8位DAC被设计为支持触发模式操作，当MCU进入STOP模式时，CMP_x_C2[RRE]和CMP_C0[EN]被设置为启用。

启用该模式后，包括操作时钟和触发器启动信号的触发事件将启动一个比较序列，在执行CMP操作和捕获输出之前，必须首先启用CMP和DAC。软件通过CMP_x_C2[FXMP]和CMP_x_C2[FXMXCH]来选择一个固定的通道，用于正片复用或负片复用。这是一个强制性的要求，循环周期的设置必须长于所有活动通道完成由CMP_C2[NSAM]指定的比较周期的时间。

然后，由CMP_x_C1[CHNn]选择的有效通道被路由到非固定通道复用器，并以轮流的方式与参考输入进行比较。为了满足比较器的稳定时间，在CMP_x_C2[NSAM]定义的可配置的操作时钟数之后，比较结果被采样，用于选定的

通道。通过写入
CMP_x_C2[ACOn]字段为每个通道配置一个软件预编程状态。在所有活动通道被采样后，如果比较结果与预编程状态发生变化，CMP_x_C2[CHnF]中的相应标志被设置。如果CMP_x_C2[RRIE]被设置，一个异步复位被断言，使MCU脱离STOP模式。

注意，这些标志不支持生成DMA传输事件。

当MCU处于STOP模式时，这种模式是有效的，所以没有任何窗口/过滤器功能可用。该模式的一个基本假设是，所选输入的变化速度比操作时钟慢得多。建议将比较器也配置在低功率比较模式下。在对CMP_C2[INITMOD]寄存器进行编程时，需要确保INITMOD*round
robin时钟周期必须长于初始化延迟，这一点可以参考芯片的数据手册。

下图显示了这种模式的基本流程。在图中，CMP_x_C1[CHN1]、CMP_x_C1[CHN3]和CMP_x_C1[CHN7]被设置，因此通道#1、#3和#7被选中进行轮循。CMP_x_C2[NSAM]被设置为2'b01，所以一个时钟后，所选通道的比较结果被采样。当通道#7被比较时，其结果被采样，轮回结束。如果通道#1、#3或#7的任何一个比较结果从它们的编程值（写入CMP_x_C2[ACO1]、CMP_x_C2[ACO3]和CMP_x_C2[ACO7]）中改变，就会产生一个中断，将MCU从STOP模式中唤醒。然后，软件可以轮询CMP_x_C2[CHnF]，以查看在STOP模式期间哪个通道的输入值发生了变化。

**注意事项**

在循环模式下，应确保RTC_CLK周期大于CMPx_C0[PMODE]值对应的比较时间。还要求不要通过INPSEL和INNSEL选择内部保留的通道进行轮循。

**注意事项**

在循环模式下，建议始终将DAC输出配置为固定端口参考。

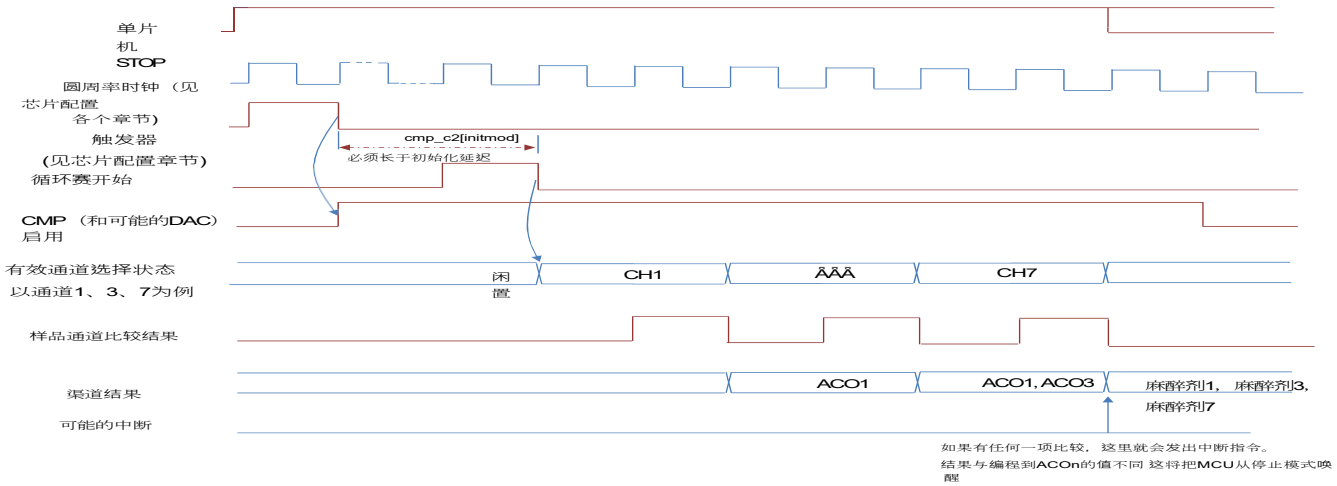

单片
机
STOP
圆周率时钟（见
芯片配置
各个章节）
触发器
（见芯片配置章节）
循环赛开始
CMP（和可能的DAC）
启用
有效通道选择状态
以通道1、3、7为例
样品通道比较结果
渠道结果
可能的中断
cmp_c2[initmod]
必须长于初始化延迟
闲
置
CH1
ÃÃÃ
CH7
ACO1
ACO1, ACO3
麻醉剂1，麻醉剂3，
麻醉剂7
如果有任何一项比较，这里就会发出中断指令。
结果与编程到ACOn的值不同 这将把MCU从停止模式唤
醒

**图40-15.触发模式**

# 第四十一章
# 可编程延迟块（PDB）

## 41.1 特定芯片的PDB信息

### 41.1.1 实例化信息

#### 41.1.1.1 PDB输出触发连接

S32K14x包含两个PDB模块。S32K11x包含一个PDB模块。

**表41-1.PDB输出触发器连接**

| 芯片 | PDB上的槽位数 | 每个插槽的触发器数量 | 脉冲输出的数量 | 预触发槽的数量 |
|---|---|---|---|---|
| S32K116 | 2 | 1 | 1 | 8 |
| S32K118 | 2 | 1 | 1 | 8 |
| S32K142 | 2 | 1 | 1 | 8 |
| S32K144 | 2 | 1 | 1 | 8 |
| S32K146 | 3 | 1 | 1 | 8 |
| S32K148 | 4 | 1 | 1 | 8 |

PDB有插槽，也被称为PDB通道，用于与芯片连接。每个槽由脉冲输出、触发器和预触发器组成，如上表所述。见图41-6和图38-2

#### 41.1.1.2 PDB输入触发器的连接

在这个设备上，PDB的触发源选择是通过TRGMUX模块实现的。对于每个PDB单元，只有一个来自TRGMUX的触发输入，但它支持不同的触发源。PDB内部的触发器复用器不被使用

任何更多。

触发模式

表41-2.PDB输入触发器连接

| PDB触发器 | PDB输入 |
| --- | --- |
| 0000 | PDBn_trigger_in0 |

## 41.1.2 PDB触发器与ADC和TRGMUX的相互联系

除了特定的PDB-
ADC触发方案（参考ADC触发源部分），PDB通道触发器也可以作为TRGMUX模块的触发源，可以用来触发ADC以外的其他模块。

以下是PDB模块的互连性。

表41-3.PDB触发器与ADC和TRGMUX的互连情况

| PDBx 触发器输出 | 互联性 |
| --- | --- |
| 通道0的触发器 | 触发器同时连接到ADC和TRGMUX |
| 通道1触发 | 触发器连接到ADC |
| 通道2触发 | 触发器连接到ADC |
| 通道3触发 | 触发器连接到ADC |
| 脉冲输出 | 脉冲输出连接到TRGMUX |

## 41.1.3 背对背的确认连接

背对背操作使ADC转换完成后，触发下一个PDB通道的预触发和触发输出。

在该MCU中，基于SIM_CHIPCTL[PDB_BB_SEL]位的设置，实现了以下PDB背对背操作的确认连接。每个PDB的所有通道上都支持PDB背对背操作。只有每个PDB的通道0支持PDB背靠背形成一个环。

当SIM_CHIPCTL[PDB_BB_SEL]=0时。

PDB背对背操作确认连接是在每个PDB单元内部以环形方式实现的。

PDB0 CH0、PDB1 CH1和PDB0-PDB1形成环状的背对背链图如下所示。

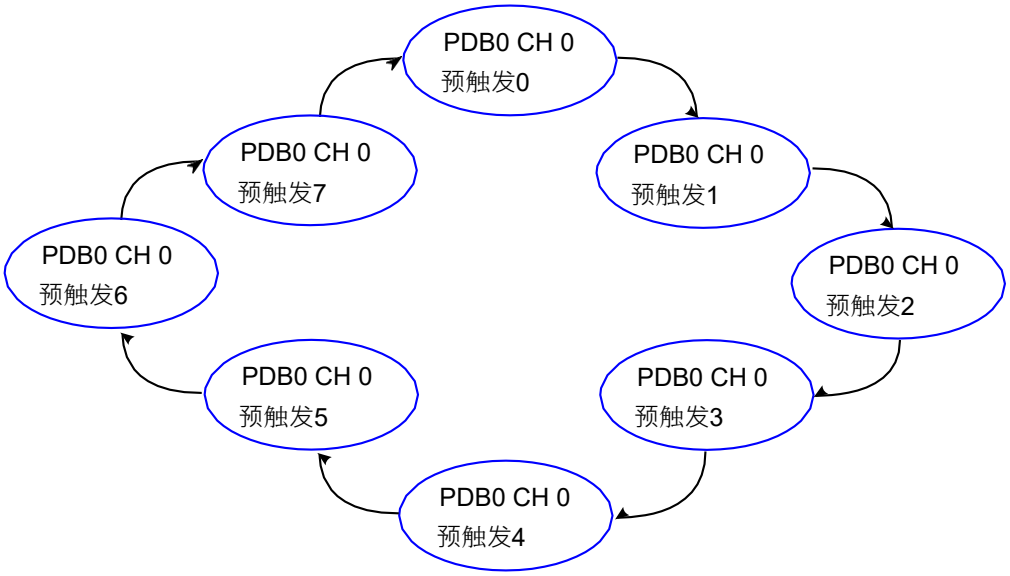



**图41-1.例子。PDB0 CH0背对背链**

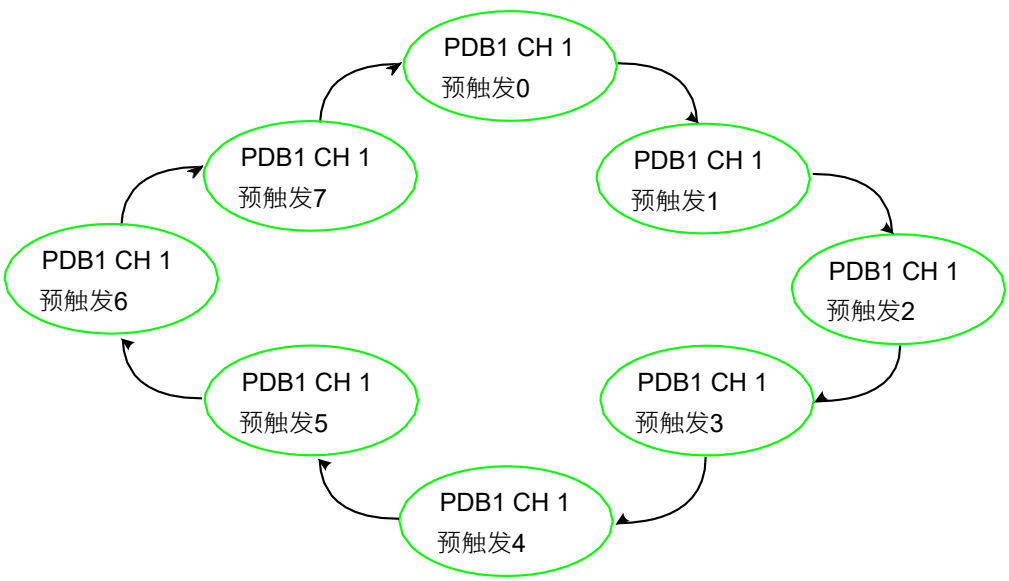



**图41-2.例子。PDB1 CH1背对背链**

当SIM_CHIPCTL[PDB_BB_SEL]=1。

PDB背对背操作确认连接是以所有PDB单元作为一个环来实现的。链状图如下。

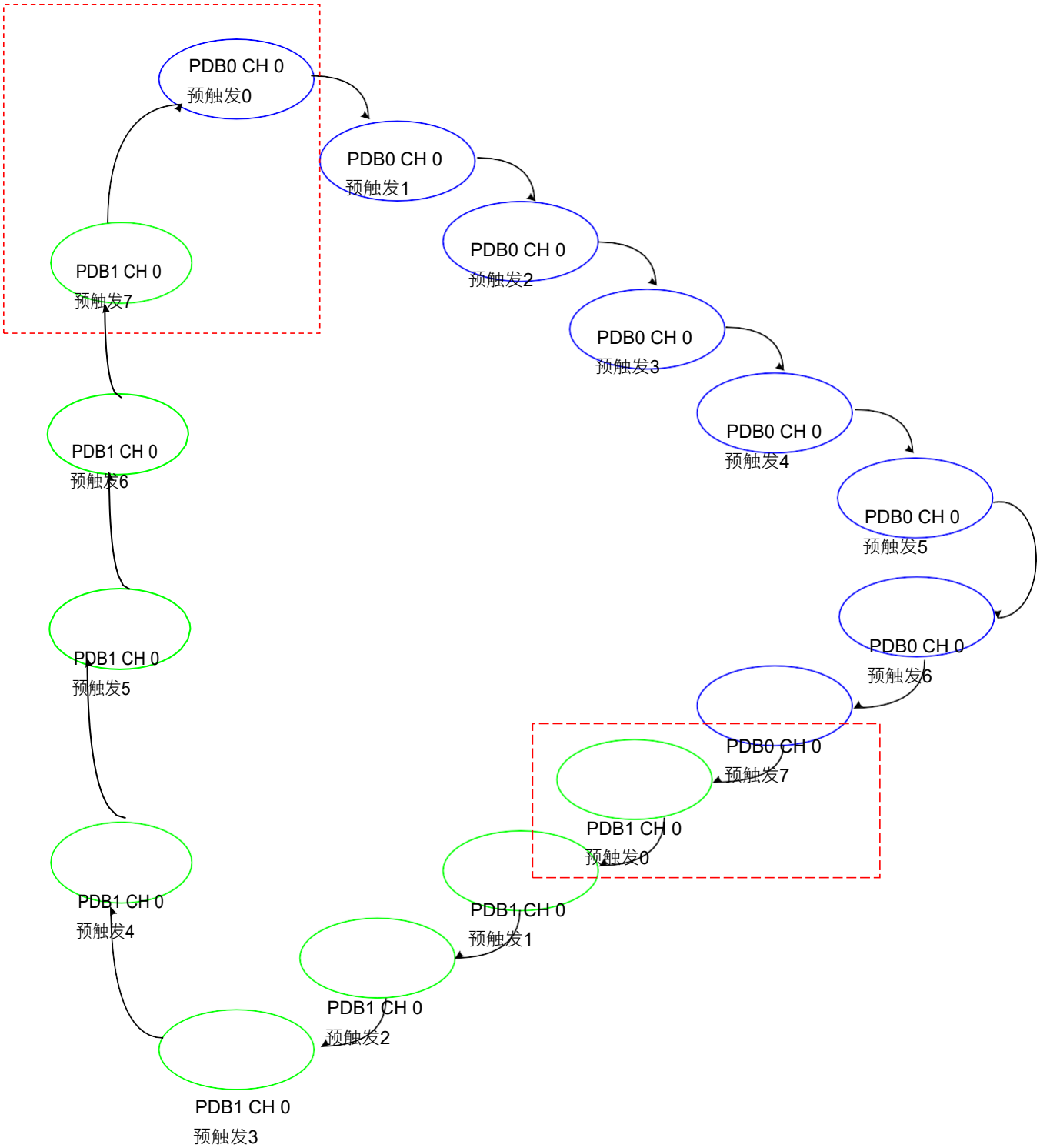



**图41-3.PDB背对背链形成PDB0-PDB1环**

应用代码可以设置PDBx_CHnC1[BB]位，将PDB预触发器配置为单链或多链。

链的开始的预触发应该由PDB提供（例如，通过设置CHnDLY1和CHnC1[EN]位），清除PDBx_CHnC1[BB]位。

**表41-4.PDB-ADC背对背连接（2的第1部分）**

| PDB实例 | PDB通道 | 触发前 # | Ack连接来自 | |
|---|---|---|---|---|
| | | | sim_chipctl[pdb_bb_sel]=0 | sim_chipctl[pdb_bb_sel]=1 |
| PDB0 | ÃÃÃ | 0 | ADC0 SC1H_COCO | ADC1 SC1H_COCO |
| | | 1 | ADC0 SC1A_COCO | ADC0 SC1A_COCO |
| | | 2 | ADC0 SC1B_COCO | ADC0 SC1B_COCO |
| | | 3 | ADC0 SC1C_COCO | ADC0 SC1C_COCO |
| | | 4 | ADC0 SC1D_COCO | ADC0 SC1D_COCO |
| | | 5 | ADC0 SC1E_COCO | ADC0 SC1E_COCO |
| | | 6 | ADC0 SC1F_COCO | ADC0 SC1F_COCO |
| | | 7 | ADC0 SC1G_COCO | ADC0 SC1G_COCO |
| PDB0 | CH1 | 0 | ADC0 SC1P_COCO | 1 |
| | | 1 | ADC0 SC1I_COCO | |
| | | 2 | ADC0 SC1J_COCO | |
| | | 3 | ADC0 SC1K_COCO | |
| | | 4 | ADC0 SC1L_COCO | |
| | | 5 | ADC0 SC1M_COCO | |
| | | 6 | ADC0 SC1N_COCO | |
| | | 7 | ADC0 SC1O_COCO | |
| PDB0 | ÃÃÃ | 0 | ADC0 SC1X_COCO | |
| | | 1 | ADC0 SC1Q_COCO | |
| | | 2 | ADC0 SC1R_COCO | |
| | | 3 | ADC0 SC1S_COCO | |
| | | 4 | ADC0 SC1T_COCO | |
| | | 5 | ADC0 SC1U_COCO | |
| | | 6 | ADC0 SC1V_COCO | |
| | | 7 | ADC0 SC1W_COCO | |
| PDB0 | ÃÃÃ | 0 | adc0 sc1ff_coco | |
| | | 1 | ADC0 SC1Y_COCO | |
| | | 2 | ADC0 SC1Z_COCO | |
| | | 3 | adc0 sc1aa_coco | |
| | | 4 | adc0 sc1bb_coco | |
| | | 5 | adc0 sc1cc_coco | |
| | | 6 | adc0 sc1dd_coco | |
| | | 7 | adc0 sc1ee_coco | |

1.仅在通道0上支持PDB背对背操作形成一个环形。

## 表41-5.PDB-ADC背对背连接（第2部分）。

| PDB 实例 | PDB通道 | 触发前 # | Ack连接来自 | |
|---|---|---|---|---|
| | | | sim_chipctl[pdb_bb_sel]=0 | sim_chipctl[pdb_bb_sel]=1 |
| PDB1 | ĀĀĀ | 0 | ADC1 SC1H_COCO | ADC0 SC1H_COCO |
| | | 1 | ADC1 SC1A_COCO | ADC1 SC1A_COCO |
| | | 2 | ADC1 SC1B_COCO | ADC1 SC1B_COCO |
| | | 3 | ADC1 SC1C_COCO | ADC1 SC1C_COCO |
| | | 4 | ADC1 SC1D_COCO | ADC1 SC1D_COCO |
| | | 5 | ADC1 SC1E_COCO | ADC1 SC1E_COCO |
| | | 6 | ADC1 SC1F_COCO | ADC1 SC1F_COCO |
| | | 7 | ADC1 SC1G_COCO | ADC1 SC1G_COCO |
| PDB1 | CH1 | 0 | ADC1 SC1P_COCO | 1 |
| | | 1 | ADC1 SC1I_COCO | |
| | | 2 | ADC1 SC1J_COCO | |
| | | 3 | ADC1 SC1K_COCO | |
| | | 4 | ADC1 SC1L_COCO | |
| | | 5 | ADC1 SC1M_COCO | |
| | | 6 | ADC1 SC1N_COCO | |
| | | 7 | ADC1 SC1O_COCO | |
| PDB1 | ĀĀĀ | 0 | ADC1 SC1X_COCO | |
| | | 1 | ADC1 SC1Q_COCO | |
| | | 2 | ADC1 SC1R_COCO | |
| | | 3 | ADC1 SC1S_COCO | |
| | | 4 | ADC1 SC1T_COCO | |
| | | 5 | ADC1 SC1U_COCO | |
| | | 6 | ADC1 SC1V_COCO | |
| | | 7 | ADC1 SC1W_COCO | |
| PDB1 | ĀĀĀ | 0 | adc1 sc1ff_coco | |
| | | 1 | ADC1 SC1Y_COCO | |
| | | 2 | ADC1 SC1Z_COCO | |
| | | 3 | adc1 sc1aa_coco | |
| | | 4 | adc1 sc1bb_coco | |
| | | 5 | adc1 sc1cc_coco | |
| | | 6 | adc1 sc1dd_coco | |
| | | 7 | adc1 sc1ee_coco | |

1.仅在通道0上支持PDB背对背操作形成一个环形。

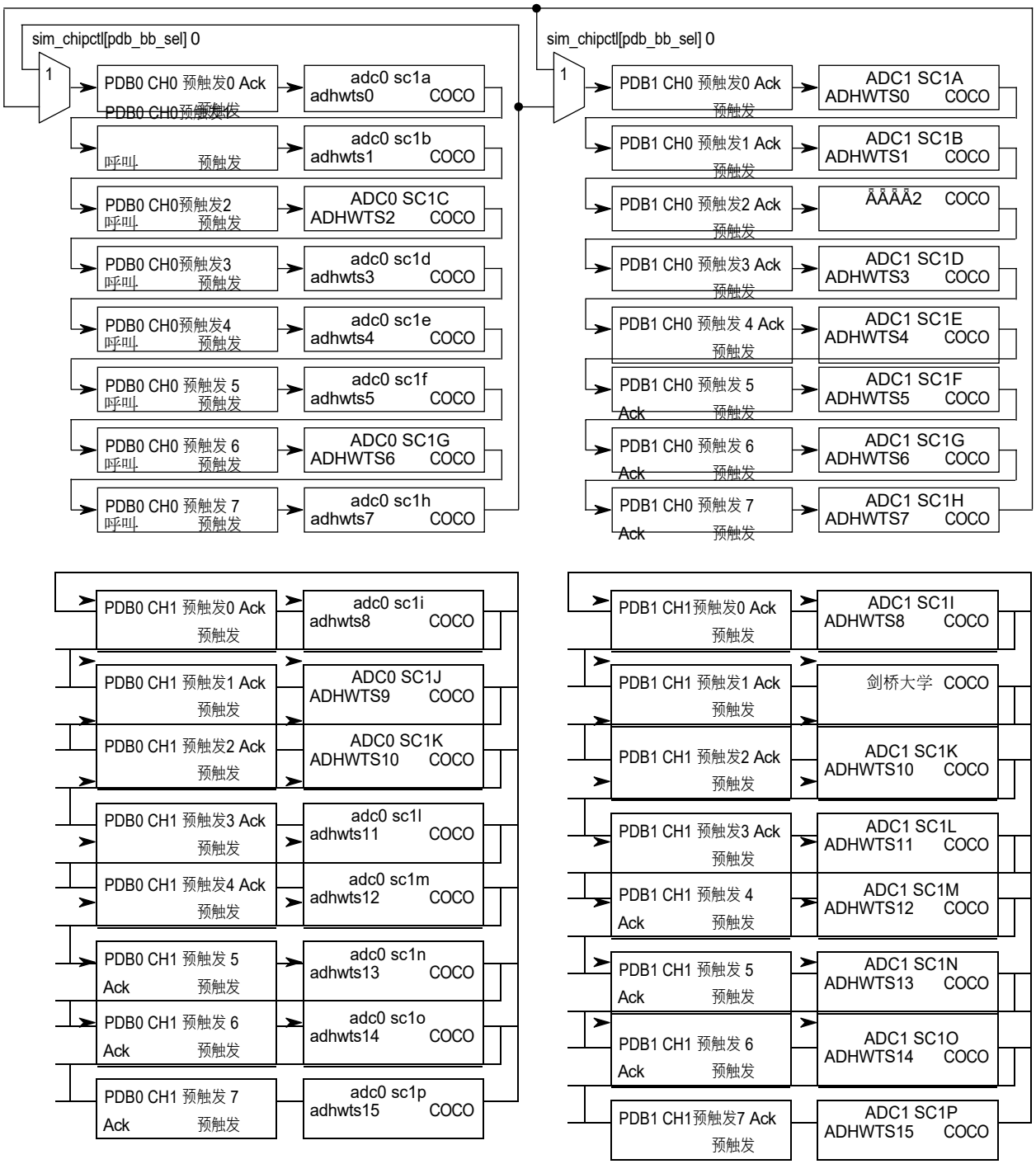



**图41-4.PDB-ADC背对背连接（第1部分，共2部分）**

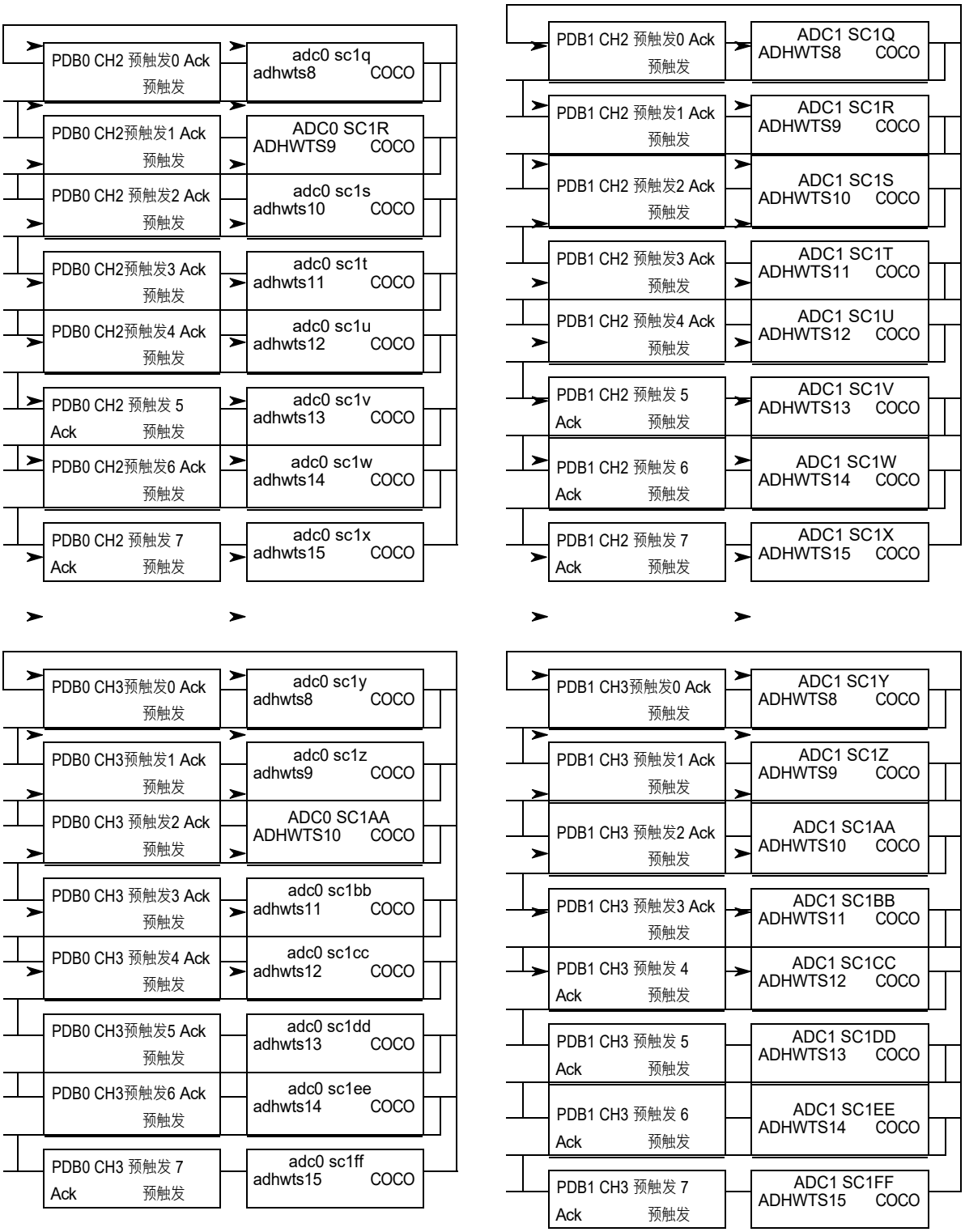



**图41-5.PDB-ADC背对背连接（第2部分）。**

### 41.1.4 脉冲输出使能寄存器的实现

下表显示了模块和芯片级的脉冲输出使能寄存器的比较。在这个设备中，每个PDB单元将有一个脉冲输出通道。脉冲输出连接到TRGMUX，然后它可以灵活地作为任何CMP模块的采样窗口工作。

**表41-6.PDB脉冲输出使能寄存器**

| 注册 | 模块实施 | 芯片实施 |
|---|---|---|
| POnEN | 7:0 - POEN<br>31:8 - 保留 | 0 - POEN[0]用于CMP<br>31:1 - 保留 |

## 41.2 简介

可编程延迟块（PDB）为ADC的硬件触发输入提供来自内部或外部触发器的可控延迟，或者一个可编程的间隔时间。PDB可以选择提供脉冲输出（Pulse-Out），作为CMP块的采样窗口。

### 41.2.1 特点

- 多达2个触发输入源和一个软件触发源
- 多达8个可配置的PDB通道用于ADC硬件触发
  - 一个PDB通道与一个ADC相关联
  - 每个PDB通道有一个用于ADC硬件触发的触发输出和多达8个用于ADC触发选择的预触发输出
  - 触发输出可以独立启用或禁用
  - 每个预触发输出有一个16位延迟寄存器
  - 预触发输出的延迟寄存器的可选旁路
  - 在单拍或连拍模式下操作

- 可选的背对背模式操作，使ADC转换完成后触发下一个PDB通道

  - 一个可编程的延迟中断
  - 一个序列错误中断
  - 每个预触发器有一个通道标志和一个序列错误标志
  - 支持DMA
- 多达8个脉冲输出（脉冲输出）。
  - 脉冲输出可以独立启用或禁用
  - 可编程的脉冲宽度

**注意事项**

PDB输入和输出触发器的数量是特定于芯片的。详见特定芯片的PDB信息。

## 41.2.2 实施

在本节中，以下字母是指输出触发器的数量。

- N-PDB通道的总可用数量。
- n-PDB通道号，从0到*N-1*有效。
- M-每个PDB通道可用的预触发器总数。
- m-预触发号码，从0到*M-1*有效。
- Y-脉冲输出的总数。
- y-脉冲输出数，有效值从0到*Y-1*。

**注意事项**

模块输出到核心的触发器的数量是由芯片决定的。关于模块到核心的输出触发器的实现，见芯片配置信息。

## 41.2.3 背对背的确认连接

PDB背对背操作确认连接是特定的芯片。关于实施，请参阅芯片配置信息。

## 41.2.4 框图

这张图说明了PDB的主要组成部分。

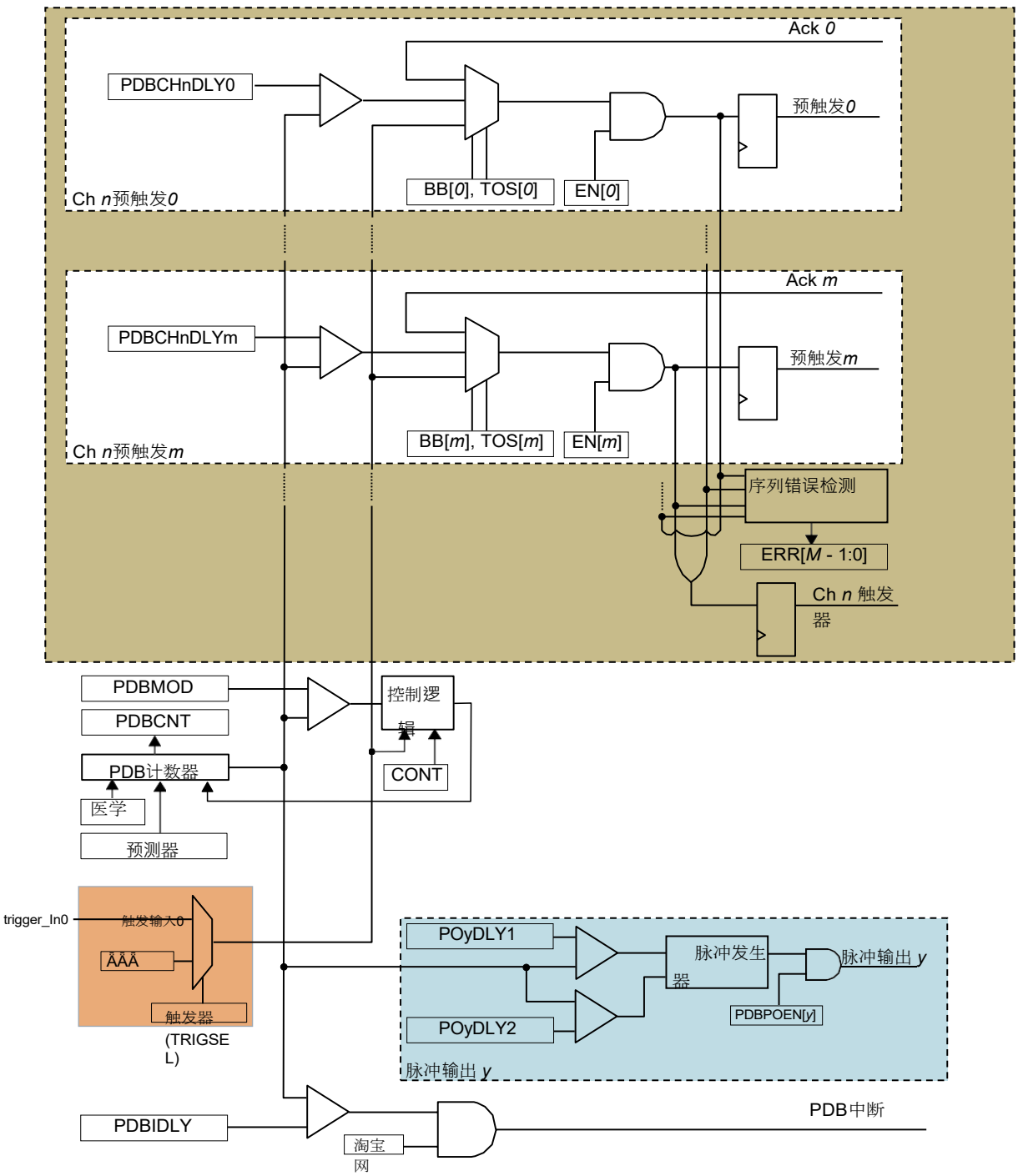



**图41-6.PDB方框图**

在此图中，只显示了一个PDB通道*n*，和一个脉冲输出*y*。启用PDB的控制逻辑和序列错误中断逻辑没有显示。

### 41.2.5 操作模式

PDB ADC触发器在以下模式下工作。

- Disabled-
  计数器关闭，如果PDB不在Bypass模式的背对背操作中，所有预触发和触发输出均为低电平。
- 当处理器处于Debug模式时，Debug-计数器会暂停。
- 在每个PDB通道中，每一个触发输入事件都有一个启用的预触发器发出信号，并在收到所选触发输入源的正沿或软件触发器被选中且SC[SWTRIG]被写为1时重新开始计数。每当任何一个预触发器被断言时，触发器输出就会被断言。
- 启用 连续-
  计数器被启用并在计数为零时重新启动。当计数达到模数寄存器中指定的值时，计数器再次翻转为零，并重新开始计数。这就使一个预触发器/触发器输出的连续流成为单一触发输入事件的结果。
- Enabled Bypassed-
  预触发和触发输出在选定的触发输入源或软件触发的正边被选定后立即断电，SC[SWTRIG]被写为1，即延迟寄存器被旁路了。可以绕过任何一个或多个延时寄存器；因此，这种模式可以与单发或连续模式一起使用。

## 41.3 内存图和寄存器定义

**PDB内存图**

| 地址偏移（十六进制） | 注册名称 | 宽度（单位：比特） | 访问 | 重置值 | 节/页 |
|---|---|---|---|---|---|
| 0 | 状态和控制寄存器（PDB_SC）。 | 32 | R/W | 0000_0000h | 41.3.1/1106 |
| 4 | 模数寄存器（PDB_MOD）。 | 32 | R/W | 0000_FFFFh | 41.3.2/1109 |
| 8 | 计数器寄存器（PDB_CNT）。 | 32 | R | 0000_0000h | 41.3.3/1110 |
| C | 中断延时寄存器（PDB_IDLY）。 | 32 | R/W | 0000_FFFFh | 41.3.4/1110 |
| 10 | 通道n控制寄存器1（PDB_CH0C1）。 | 32 | R/W | 0000_0000h | 41.3.5/1111 |

| | | | | | |
|---|---|---|---|---|---|
| 14 | 通道n状态寄存器（PDB_CH0S）。 | 32 | R/W | 0000_0000h | 41.3.6/1112 |
| 18 | 通道n延迟0寄存器（PDB_CH0DLY0）。 | 32 | R/W | 0000_0000h | 41.3.7/1112 |
| 1C | 通道n延迟1寄存器（PDB_CH0DLY1）。 | 32 | R/W | 0000_0000h | 41.3.8/1113 |

*表继续在下一页...*

*表继续在下一页...*

## PDB内存图（续）

| 地址偏移（十六进制） | 注册名称 | 宽度（单位：比特） | 访问 | 重置值 | 节/页 |
|---|---|---|---|---|---|
| 20 | 通道n延迟2寄存器（PDB_CH0DLY2）。 | 32 | R/W | 0000_0000h | 41.3.9/1114 |
| 24 | 通道n延迟3寄存器（PDB_CH0DLY3）。 | 32 | R/W | 0000_0000h | 41.3.10/1114 |
| 28 | 通道n延迟4寄存器（PDB_CH0DLY4）。 | 32 | R/W | 0000_0000h | 41.3.11/1115 |
| 2C | 通道n延迟5寄存器（PDB_CH0DLY5）。 | 32 | R/W | 0000_0000h | 41.3.12/1116 |
| 30 | 通道n延迟6寄存器（PDB_CH0DLY6）。 | 32 | R/W | 0000_0000h | 41.3.13/1116 |
| 34 | 通道n延迟7寄存器（PDB_CH0DLY7）。 | 32 | R/W | 0000_0000h | 41.3.14/1117 |
| 38 | 通道n控制寄存器1（PDB_CH1C1）。 | 32 | R/W | 0000_0000h | 41.3.5/1111 |
| 3C | 通道n状态寄存器（PDB_CH1S）。 | 32 | R/W | 0000_0000h | 41.3.6/1112 |
| 40 | 通道n延迟0寄存器（PDB_CH1DLY0）。 | 32 | R/W | 0000_0000h | 41.3.7/1112 |
| 44 | 通道n延迟1寄存器（PDB_CH1DLY1）。 | 32 | R/W | 0000_0000h | 41.3.8/1113 |
| 48 | 通道n延迟2寄存器（PDB_CH1DLY2）。 | 32 | R/W | 0000_0000h | 41.3.9/1114 |
| 4C | 通道n延迟3寄存器（PDB_CH1DLY3）。 | 32 | R/W | 0000_0000h | 41.3.10/1114 |
| 50 | 通道n延迟4寄存器（PDB_CH1DLY4）。 | 32 | R/W | 0000_0000h | 41.3.11/1115 |
| 54 | 通道n延迟5寄存器（PDB_CH1DLY5）。 | 32 | R/W | 0000_0000h | 41.3.12/1116 |
| 58 | 通道n延迟6寄存器（PDB_CH1DLY6）。 | 32 | R/W | 0000_0000h | 41.3.13/1116 |
| 5C | 通道n延迟7寄存器（PDB_CH1DLY7）。 | 32 | R/W | 0000_0000h | 41.3.14/1117 |
| 60 | 通道n控制寄存器1（PDB_CH2C1）。 | 32 | R/W | 0000_0000h | 41.3.5/1111 |
| 64 | 通道n状态寄存器（PDB_CH2S）。 | 32 | R/W | 0000_0000h | 41.3.6/1112 |
| 68 | 通道n延迟0寄存器（PDB_CH2DLY0）。 | 32 | R/W | 0000_0000h | 41.3.7/1112 |
| 6C | 通道n延迟1寄存器（PDB_CH2DLY1）。 | 32 | R/W | 0000_0000h | 41.3.8/1113 |
| 70 | 通道n延迟2寄存器（PDB_CH2DLY2）。 | 32 | R/W | 0000_0000h | 41.3.9/1114 |
| 74 | 通道n延迟3寄存器（PDB_CH2DLY3）。 | 32 | R/W | 0000_0000h | 41.3.10/1114 |
| 78 | 通道n延迟4寄存器（PDB_CH2DLY4）。 | 32 | R/W | 0000_0000h | 41.3.11/1115 |
| 7C | 通道n延迟5寄存器（PDB_CH2DLY5）。 | 32 | R/W | 0000_0000h | 41.3.12/1116 |
| 80 | 通道n延迟6寄存器（PDB_CH2DLY6）。 | 32 | R/W | 0000_0000h | 41.3.13/1116 |

| | | | | | |
|---|---|---|---|---|---|
| 84 | 通道n延迟7寄存器（PDB_CH2DLY7）。 | 32 | R/W | 0000_0000h | 41.3.14/1117 |
| 88 | 通道n控制寄存器1（PDB_CH3C1）。 | 32 | R/W | 0000_0000h | 41.3.5/1111 |
| 8C | 通道n状态寄存器（PDB_CH3S）。 | 32 | R/W | 0000_0000h | 41.3.6/1112 |

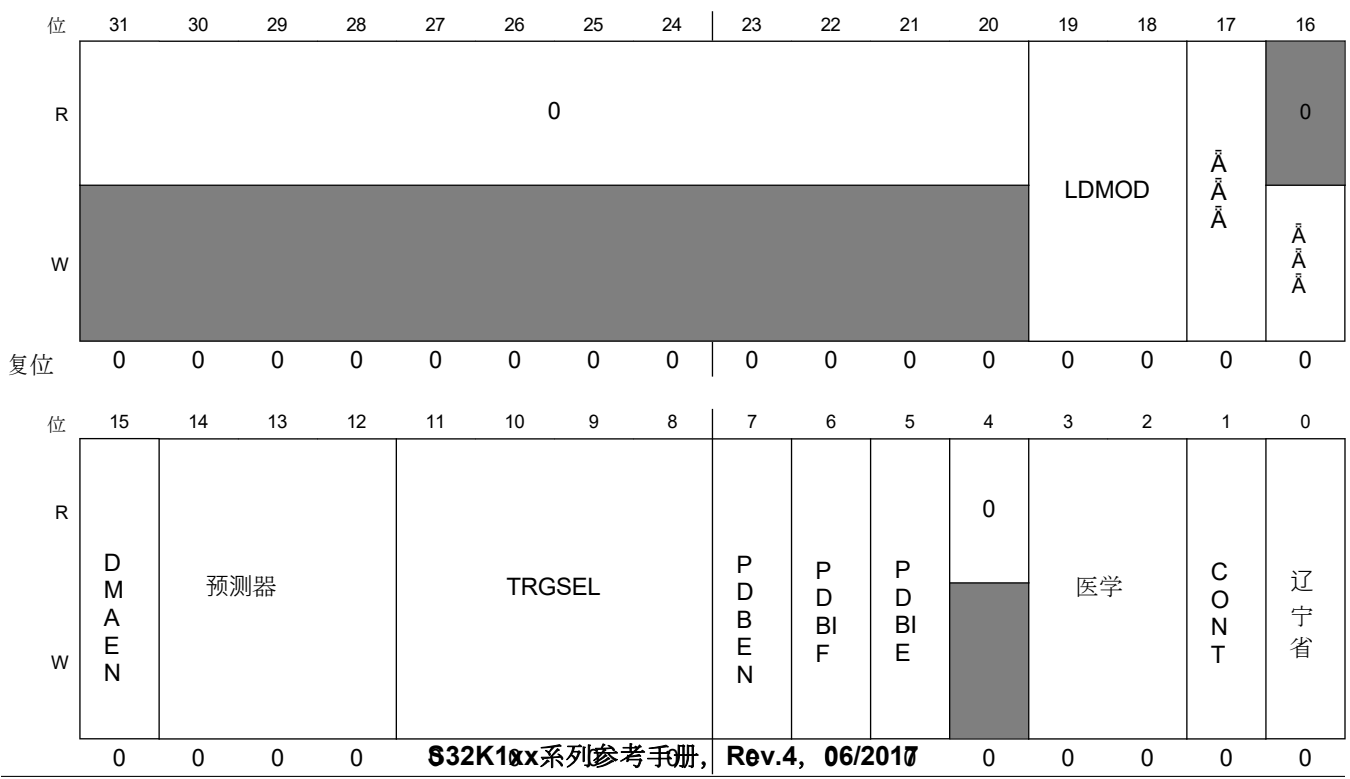
*表继续在下一页...*

PDB内存图（续）

| 地址偏移（十六进制） | 注册名称 | 宽度（单位：比特） | 访问 | 重置值 | 节/页 |
|---|---|---|---|---|---|
| 90 | 通道n延迟0寄存器（PDB_CH3DLY0）。 | 32 | R/W | 0000_0000h | 41.3.7/1112 |
| 94 | 通道n延迟1寄存器（PDB_CH3DLY1）。 | 32 | R/W | 0000_0000h | 41.3.8/1113 |
| 98 | 通道n延迟2寄存器（PDB_CH3DLY2）。 | 32 | R/W | 0000_0000h | 41.3.9/1114 |
| 9C | 通道n延迟3寄存器（PDB_CH3DLY3）。 | 32 | R/W | 0000_0000h | 41.3.10/1114 |
| A0 | 通道n延迟4寄存器（PDB_CH3DLY4）。 | 32 | R/W | 0000_0000h | 41.3.11/1115 |
| A4 | 通道n延迟5寄存器（PDB_CH3DLY5）。 | 32 | R/W | 0000_0000h | 41.3.12/1116 |
| A8 | 通道n延迟6寄存器（PDB_CH3DLY6）。 | 32 | R/W | 0000_0000h | 41.3.13/1116 |
| 交流 | 通道n延迟7寄存器（PDB_CH3DLY7）。 | 32 | R/W | 0000_0000h | 41.3.14/1117 |
| 190 | 脉冲输出n启用寄存器（PDB_POEN）。 | 32 | R/W | 0000_0000h | 41.3.15/1117 |
| 194 | 脉冲输出n延时寄存器（PDB_PO0DLY） | 32 | R/W | 0000_0000h | 41.3.16/1118 |

## 41.3.1 状态和控制寄存器（PDB_SC）。

地址。0h base + 0h offset = 0h



复位

*表继续在下一页...*

## PDB_SC字段描述

| 场地 | 描述 |
|---|---|
| 31-20<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 19-18<br>LDMOD | 负载模式选择<br><br>选择在LDOK写入1后，加载MOD、IDLY、CHnDLYm、INTx和POyDLY寄存器的模式。<br><br>00 在LDOK被写入1后，内部寄存器立即被加载其缓冲器中的值。<br>01 当PDB计数器(CNT)=MOD+1 CNT延迟过后，内部寄存器被加载其缓冲器的值，在1被写入LDOK后。<br>10 当检测到一个触发输入事件时，内部寄存器被加载其缓冲器的值，在1被写入LDOK后。<br>11 当PDB计数器(CNT)=MOD+1 CNT延迟过后，或者检测到一个触发输入事件后，内部寄存器被加载其缓冲器的值，1被写入LDOK。 |
| 17<br>ÃÃÃ | PDB 序列错误中断启用<br><br>启用PDB序列错误中断。当PDBEIE被设置时，任何一个PDB通道的序列错误标志都会产生一个PDB序列错误中断。<br><br>0 PDB序列错误中断失效。<br>1 启用PDB序列错误中断。 |
| 16<br>ÃÃÃ | 软件触发器<br><br>当PDB被启用并且软件触发器被选为触发输入源时，向SWTRIG写1会重置并重新启动计数器。向SWTRIG写0没有作用。读取SWTRIG可以得到0。 |
| 15<br>DMAEN | 启用DMA<br><br>当DMA被启用时，PDBIF标志会产生一个DMA请求而不是一个中断。<br><br>0 DMA禁用。<br>1 启用DMA。 |
| 14-12<br>预测器 | 预分频器分频器选择<br><br>计数使用外围时钟除以一个因子（由MULT字段选择）和一个整数因子（由PRESCALAR字段设置）的乘积，或者换句话说，（外围时钟）/（MULT x PRESCALAR）。<br><br>000Counting 使用外围时钟除以MULT（乘法系数）。001计数使用外围时钟除以2 x MULT（乘法系数）。010计数使用外设时钟除以4 x MULT（乘法系数）。011C 计数使用外围时钟除以8 x MULT（乘法系数）。<br>100 计数使用外设时钟除以16 x MULT（乘法系数）。<br>101 计数使用外设时钟除以32 x MULT（乘法系数）。<br>110 计数使用外设时钟除以64 x MULT（乘法系数）。 |

| | |
|---|---|
| | 111　计数使用外设时钟除以128 x MULT（乘法系数）。 |
| 11-8<br>TRGSEL | 触发输入源选择<br><br>选择PDB的触发输入源。触发输入源可以是内部或外部（EXTRG引脚），或软件触发。关于实际的PDB输入触发连接，请参考芯片配置细节。<br><br>0000Trigger-In 0被选中。 |

*表继续在下一页...*

## PDB_SC字段描述（续）

| 场地 | 描述 |
| --- | --- |
| | 0001Trigger-In 1被选中。<br>0010Trigger-In 2被选中。<br>0011Trigger-In 3被选中。<br>0100Trigger-In 4被选中。<br>0101Trigger-In 5被选中。<br>0110Trigger-In 6被选中。<br>0111Trigger-In 7被选中。<br>1000Trigger-In 8被选中。<br>1001Trigger-In 9被选中。<br>1010Trigger-In 10被选中。<br>1011Trigger-In 11被选中。<br>1100Trigger-In 12被选中。<br>1101Trigger-In 13被选中。<br>1110Trigger-In 14被选中。1111<br>软件触发器被选中。 |
| 7<br>PDBEN | 启用PDB<br><br>0 PDB禁用。计数器关闭。<br>1 启用PDB。 |
| 6<br>PDBIF | PDB中断标志<br><br>当计数器的值等于IDLY寄存器+1时，PDBIF被设置。写0来清除PDBIF。 |
| 5<br>PDBIE | PDB中断启用<br><br>启用PDB中断。当PDBIE被设置且DMAEN被清零时，PDBIF产生一个PDB中断。<br><br>0 禁用PDB中断。<br>1 启用PDB中断。 |
| 4<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 3-2<br>医学 | 预分频器的乘法系数选择<br><br>选择计数器时钟的预分频器分频器的乘法系数。00乘法 系数为1。<br>01乘法系数为10。<br>10 乘法系数为20。<br>11 乘法系数是40。 |

| | |
|---|---|
| 1<br>CONT | 启用连续模式<br><br>启用连续模式下的PDB操作。<br><br>0 在单发模式下的PDB操作<br>1 连续模式下的PDB操作 |
| 0<br>辽宁省 | 加载OK<br><br>向LDOK位写1，可以用先前写入其内部缓冲器（并存储在那里）的值更新MOD、IDLY、CHnDLYm和POyDLY寄存器。MOD、IDLY、CHnDLYm和POyDLY寄存器的新值将根据LDMOD字段（加载模式选择）的设置而生效。 |

*表继续在下一页...*

*表继续在下一页...*

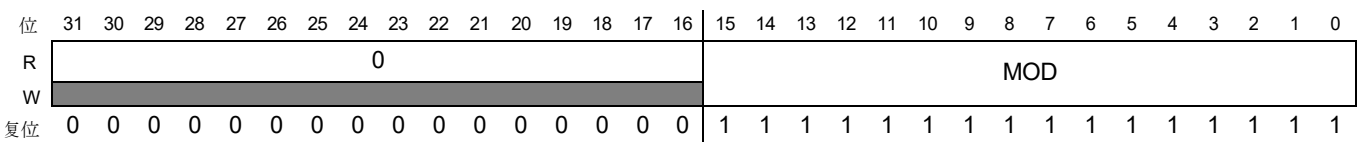
**PDB_SC字段描述（续）**

| 场地 | 描述 |
|---|---|
| | 在1被写入LDOK字段之前，这些寄存器的内部缓冲区的值是无效的，在内部缓冲区的现有值被加载到相应的寄存器之前，新的值不能被写入内部缓冲区。<br>• 只有当PDBEN被设置时才能写入LDOK，或者当PDBEN被写为1时，LDOK可以同时被写入。<br>• 当内部缓冲区的值被加载到寄存器中或者PDBEN位（PDB Enable）被清除时，LDOK自动被清除。<br>• 向LDOK写0没有效果。 |

## 41.3.2 模数寄存器（PDB_MOD）。

**注意**：该寄存器是内部缓冲区，任何写入该寄存器的值都被写入其内部缓冲区，换句话说，内部设备总线不直接写入该寄存器。只有在SC[LDOK]位被写入 "1 "后，该寄存器内部缓冲区的值才会被加载到该寄存器。

地址。0h基数+4h偏移量=4h




**PDB_MOD字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| MOD | PDB模数<br><br>指定计数器的周期。当计数器达到这个值时，它将被重置为零。如果PDB处于连续模式，计数将重新开始。<br>读取该字段会返回对PDB当前周期有效的内部寄存器的值。 |

## 41.3.3 计数器寄存器（PDB_CNT）。

**注意事项**

对这个只读寄存器的写入将产生一个传输错误（也可能是一个硬故障）。

地址。0h基数+8h偏移量=8h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | | CNT | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**PDB_CNT字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| CNT | PDB计数器<br><br>包含计数器的当前值。 |

## 41.3.4 中断延时寄存器（PDB_IDLY）。

**注意**：这个寄存器是内部缓冲区，任何写入寄存器的值都会被写入其内部缓冲区，换句话说，内部设备总线不会直接写入这个寄存器。只有在SC[LDOK]位被写入"1 "后，该寄存器内部缓冲区的值才会被加载到该寄存器。

地址。0h基数+Ch偏移量=Ch

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | | 身份证 | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

**PDB_IDLY字段描述**

| 场地 | 描述 |
|---|---|

| | |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 身份证 | PDB中断延时 |

*表继续在下一页...*

**PDB_IDLY字段描述（续）**

| 场地 | 描述 |
|---|---|
| | 指定用于安排PDB中断的延迟值。它可以用来在PDB周期的某个点上安排一个独立的中断。如果启用，当计数器等于IDLY时，将产生一个PDB中断。读取这个字段返回内部寄存器的值，该值对PDB的当前周期有效。 |

## 41.3.5 通道n控制寄存器1（PDB_CHnC1）。

每个PDB通道都有一个控制寄存器，CHnC1。该寄存器中的位控制每个PDB通道操作的功能。

地址。0h基数+10h偏移量+（40d×i），其中i=0d至3d

| 位 | 31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 | 15 14 13 12 11 10 9 | 8 | 7 6 5 4 3 2 1 0 |
|---|---|---|---|---|
| R | 0 | ĂĂĂ | TOS | EN |
| W | | | | |
| 复位 | 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 | 0 0 0 0 0 0 0 | 0 | 0 0 0 0 0 0 0 0 |

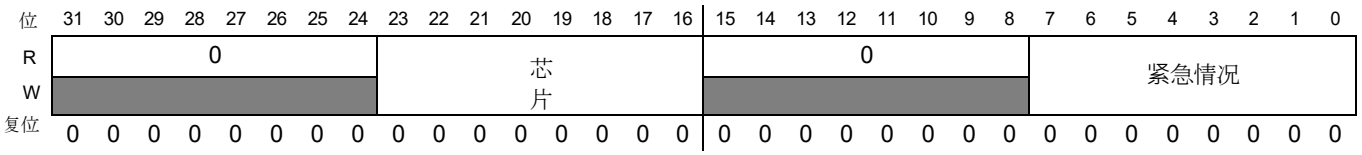
**PDB_CHnC1字段描述**

| 场地 | 描述 |
|---|---|
| 31-24<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 23-16<br>ĂĂĂ | PDB通道预触发背对背操作启用<br><br>这些位使PDB ADC预触发操作为背对背模式。在这个MCU中只实现了低M预触发位。背对背操作使ADC转换完成后触发下一个PDB通道预触发和触发输出，这样ADC转换可以在下一组配置和结果寄存器上被触发。应用代码必须只在背对背连接链的领先位置启用PDB预触发器的背对背操作。<br><br>0 PDB通道相应的预触发背对背操作被禁用。<br>1 PDB通道对应的预触发背对背操作启用。 |
| 15-8<br>TOS | PDB通道预触发输出选择<br><br>这些位选择PDB ADC预触发输出。在这个MCU中只实现了低M预触发位。<br><br>0 PDB通道的相应预触发器处于旁路模式。预触发器在检测到所选触发输入源的上升沿或选择软件触发并将SWTRIG写为1后的一个外设时钟周期内发出信号。<br>1 PDB通道对应的预触发器在所选触发输入源检测到上升沿或选择软件触发且SETRIG被写为1后，当计数器达到通道延迟寄存器加上一个外设时钟周期时发出信号。 |

| | |
|---|---|
| EN | PDB通道预触发启用<br><br>这些位启用PDB ADC预触发输出。在这个MCU中只实现了低M预触发位。<br><br>0 PDB通道的相应预触发器被禁用。<br>1 启用PDB通道的相应预触发器。 |

## 41.3.6 通道n状态寄存器（PDB_CHnS）。

地址。0h基数+14h偏移量+（40d×i），其中i=0d至3d



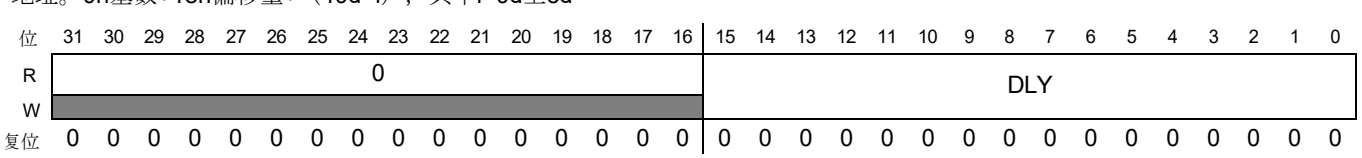

**PDB_CHnS字段描述**

| 场地 | 描述 |
|---|---|
| 31-24<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 23-16<br>芯片 | PDB通道标志<br><br>当PDB计数器（PDB_CNT）与CHnDLYm+1的值相匹配时，CF[*m*]字段被设置。写0来清除CF。 |
| 15-8<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 紧急情况 | PDB通道序列错误标志<br><br>在这个MCU中只实现了较低的M位。<br><br>0 在PDB通道相应的预触发器上没有检测到序列错误。<br>1 在PDB通道的相应预触发器上检测到序列错误。当由PDB通道*n*的一个预触发器触发的一次转换正在进行时，PDB通道对应的预触发器m的新触发不能被ADCn接受，ERR[m]被设置。写0来清除序列错误标志。 |

## 41.3.7 通道n延迟0寄存器（PDB_CHnDLY0）。

**注意**：这个寄存器是内部缓冲区，任何写入寄存器的值都会被写入其内部缓冲区，换句话说，内部设备总线不会直接写入这个寄存器。只有在SC[LDOK]位被写入"1 "后，该寄存器内部缓冲区的值才会被加载到该寄存器。

地址。0h基数+18h偏移量+（40d×i），其中i=0d至3d



**PDB_CHnDLY0字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| DLY | PDB通道延时<br><br>这些位指定了该通道相应的预触发器的延迟值。当计数器等于DLY时，预触发器就会启动。读取这些位会返回对当前PDB周期有效的内部寄存器的值。 |

## 41.3.8 通道n延迟1寄存器（PDB_CHnDLY1）。

**注意**：这个寄存器是内部缓冲区，任何写入寄存器的值都会被写入其内部缓冲区，换句话说，内部设备总线不会直接写入这个寄存器。只有在SC[LDOK]位被写入"1 "后，该寄存器内部缓冲区的值才会被加载到该寄存器。

地址。0h基数+1Ch偏移量+（40d×i），其中i=0d至3d

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | | DLY | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**PDB_CHnDLY1字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| DLY | PDB通道延时<br><br>这些位指定了该通道相应的预触发器的延迟值。当计数器等于DLY时，预触发器就会启动。读取这些位会返回对当前PDB周期有效的内部寄存器的值。 |

## 41.3.9 通道n延迟2寄存器（PDB_CHnDLY2）。

**注意**：这个寄存器是内部缓冲区，任何写入寄存器的值都会被写入其内部缓冲区，换句话说，内部设备总线不会直接写入这个寄存器。只有在SC[LDOK]位被写入"1 "后，该寄存器内部缓冲区的值才会被加载到该寄存器。

地址。0h基数+20h偏移量+（40d×i），其中i=0d至3d

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | | DLY | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**PDB_CHnDLY2字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| DLY | PDB通道延时<br><br>这些位指定了该通道相应的预触发器的延迟值。当计数器等于DLY时，预触发器就会启动。读取这些位会返回对当前PDB周期有效的内部寄存器的值。 |

## 41.3.10 通道n延迟3寄存器（PDB_CHnDLY3）。

**注意**：这个寄存器是内部缓冲区，任何写入寄存器的值都会被写入其内部缓冲区，换句话说，内部设备总线不会直接写入这个寄存器。只有在SC[LDOK]位被写入"1 "后，该寄存器内部缓冲区的值才会被加载到该寄存器。

地址。0h基数+24h偏移量+（40d×i），其中i=0d至3d

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | | DLY | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**PDB_CHnDLY3字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| DLY | PDB通道延时<br><br>这些位指定了该通道相应的预触发器的延迟值。当计数器等于DLY时，预触发器就会启动。读取这些位会返回对当前PDB周期有效的内部寄存器的值。 |

## 41.3.11 通道n延迟4寄存器（PDB_CHnDLY4）。

**注意**：这个寄存器是内部缓冲区，任何写入寄存器的值都会被写入其内部缓冲区，换句话说，内部设备总线不会直接写入这个寄存器。只有在SC[LDOK]位被写入"1 "后，该寄存器内部缓冲区的值才会被加载到该寄存器。

地址。0h基数+28h偏移量+（40d×i），其中i=0d至3d

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | | DLY | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**PDB_CHnDLY4字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| DLY | PDB通道延时<br><br>这些位指定了该通道相应的预触发器的延迟值。当计数器等于DLY时，预触发器就会启动。读取这些位会返回对当前PDB周期有效的内部寄存器的值。 |

## 41.3.12 通道n延迟5寄存器（PDB_CHnDLY5）。

**注意**：这个寄存器是内部缓冲区，任何写入寄存器的值都会被写入其内部缓冲区，换句话说，内部设备总线不会直接写入这个寄存器。只有在SC[LDOK]位被写入"1 "后，该寄存器内部缓冲区的值才会被加载到该寄存器。

地址。0h基数+2Ch偏移量+（40d×i），其中i=0d至3d

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | | DLY | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**PDB_CHnDLY5字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| DLY | PDB通道延时<br><br>这些位指定了该通道相应的预触发器的延迟值。当计数器等于DLY时，预触发器就会启动。读取这些位会返回对当前PDB周期有效的内部寄存器的值。 |

## 41.3.13 通道n延迟6寄存器（PDB_CHnDLY6）。

**注意**：这个寄存器是内部缓冲区，任何写入寄存器的值都会被写入其内部缓冲区，换句话说，内部设备总线不会直接写入这个寄存器。只有在SC[LDOK]位被写入"1 "后，该寄存器内部缓冲区的值才会被加载到该寄存器。

地址。0h基数+30h偏移量+（40d×i），其中i=0d至3d

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | | DLY | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**PDB_CHnDLY6字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| DLY | PDB通道延时<br><br>这些位指定了该通道相应的预触发器的延迟值。当计数器等于DLY时，预触发器就会启动。读取这些位会返回对当前PDB周期有效的内部寄存器的值。 |

## 41.3.14 通道n延迟7寄存器（PDB_CHnDLY7）。

**注意**：这个寄存器是内部缓冲区，任何写入寄存器的值都会被写入其内部缓冲区，换句话说，内部设备总线不会直接写入这个寄存器。只有在SC[LDOK]位被写入"1 "后，该寄存器内部缓冲区的值才会被加载到该寄存器。

地址。0h基数+34h偏移量+（40d×i），其中i=0d至3d

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | | DLY | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**PDB_CHnDLY7字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| DLY | PDB通道延时<br><br>这些位指定了该通道相应的预触发器的延迟值。当计数器等于DLY时，预触发器就会启动。读取这些位会返回对当前PDB周期有效的内部寄存器的值。 |

## 41.3.15 脉冲输出n启用寄存器（PDB_POEN）。

地址。0h基数+190h偏移量=190h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | | | | | | | | | | 坡恩 | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**PDB_POEN字段描述**

| 场地 | 描述 |
|---|---|
| 31-8<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 坡恩 | PDB 脉冲输出启用<br><br>启用脉冲输出。在这个MCU中只实现了低8位。<br><br>0 禁用PDB脉冲输出<br>1 启用PDB脉冲输出 |

## 41.3.16 脉冲输出n延时寄存器（PDB_POnDLY）

**注意**：这个寄存器是内部缓冲区，任何写入寄存器的值都会被写入其内部缓冲区，换句话说，内部设备总线不会直接写入这个寄存器。只有在SC[LDOK]位被写入"1 "后，该寄存器内部缓冲区的值才会被加载到该寄存器。

地址。0h基数+194h偏移量+（4d×i），其中i=0d至0d

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R<br>W | DLY1 | | | | | | | | | | | | | | | | DLY2 | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**PDB_POnDLY字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>DLY1 | PDB 脉冲输出延时1<br><br>这些位指定PDB脉冲输出的延迟1值。当PDB计数器等于DLY1时，Pulse-Out为高电平。读取这些位会返回对当前PDB周期有效的内部寄存器的值。 |
| DLY2 | PDB 脉冲输出延迟2<br><br>这些位指定PDB脉冲输出的延迟2值。当PDB计数器等于DLY2时，Pulse-Out为低电平。读取这些位会返回对当前PDB周期有效的内部寄存器的值。 |

# 41.4 职能描述

### 41.4.1 PDB预触发和触发输出

PDB包含一个计数器，其输出与几个不同的数字值进行比较。如果PDB被启用，那么一个触发输入事件将重置计数器并使其开始计数。触发输入事件被定义为在选定的触发输入源上检测到一个上升沿，或者如果选择了软件触发器，并且SC[SWTRIG]被写入了
1.对于每个通道，延迟*m*决定了从断言触发输入事件到预触发*m*输出信号的变化开始的时间。该时间定义为：。

- 触发输入事件到预触发*m*=（预分频器×乘法系数×延迟*m*）+2个外围时钟周期
- 增加1个额外的外设时钟周期来确定通道触发器输出变化的时间。

每个通道与1个ADC块相关。PDB通道n预触发输出0到*M*；每个预触发输出都与ADC硬件触发选择和硬件触发输入相连。预触发器用于在实际触发发生之前对ADC块进行预处理。当ADC收到触发器的上升沿时，ADC将根据预触发器确定的预设条件开始转换。ADC包含*M*组配置和结果寄存器，允许它在*M*个不同的模拟源之间交替转换（就像一个乒乓球游戏）。预触发输出用于指定下一个被采样的信号。当预触发器*m*被断言时，ADC的转换被触发了配置和结果寄存器的*m*组。

下图所示的波形显示了PDB通道*n*的预触发和触发输出。延迟可以使用CHnDLYm寄存器独立设置，预触发可以在CHnC1[EN[*m*]]中启用或禁用。

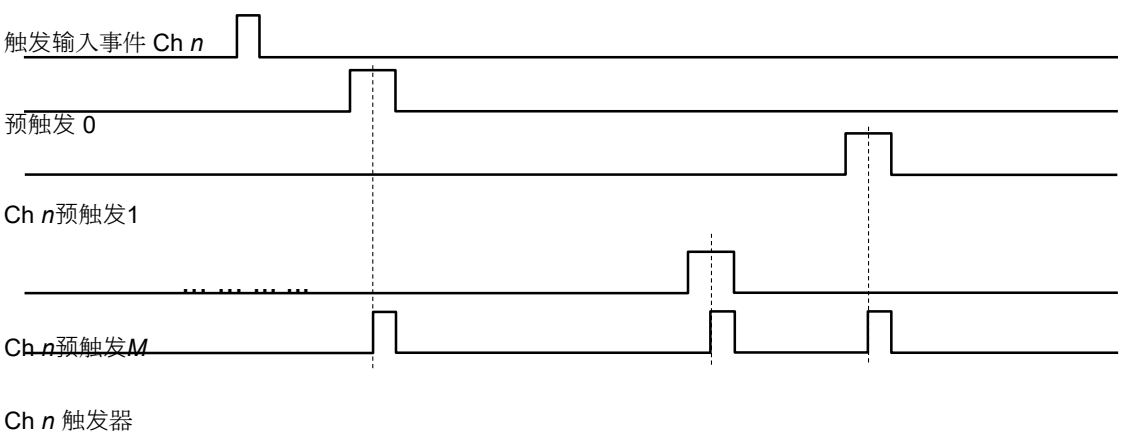



**图41-7.预触发和触发输出**

如果CHnC1[TOS[*m*]]被清零，CHnDLYm寄存器中的延迟可以被选择性地绕过。在这种情况下，当触发输入事件发生时，预触发器*m*在2个外设时钟周期后被断言
。

PDB可以被配置为背对背操作。背对背操作使ADC转换的完成能够触发下一个PDB通道的预触发和触发输出，这样就可以在下一组配置和结果寄存器上触发ADC转换。当通过设置CHnC1[BB[*m*]]启用背对背操作时，则延迟*m*被忽略，预触发器*m*在收到确认*m*后的2个外设周期被断言。该MCU中的确认连接在背对背确认连接中描述。

当来自PDB通道*n*的预触发器被断言时，预触发器的相关锁变为有效。相关的锁被相应的ADCnSC1[COCO]的上升沿释放；ADCnSC1[COCO]应在转换结果被读取后被清除，以便以后可以产生ADCnSC1[COCO]的下一个上升沿来清除该锁。当以下情况发生时，锁会变得不活跃。

- 的上升沿发生。
- 或相应的PDB预触发器被禁用。
- 或PDB被禁用

当通道*n*中任何一个预触发器的锁处于活动状态时，通道*n*的触发器输出被抑制。如果一个新的预触发器*m*在PDB通道*n*中的锁处于活动状态时断电，那么一个寄存器标志位CHnS[ERR[*m*]]（与预触发器*m*相关）被设置。如果SC[PDBEIE]被设置，那么就会产生序列错误中断。序列错误的发生通常是因为延迟*m*设置得太短，在先前触发的ADC转换完成之前，预触发器*m*就已经发出了信号。

当PDB计数器达到数值（CNT+1）时，SC[PDBIF]标志被设置。如果SC[PDBIE]被设置且SC[DMAEN]被清零，则可以产生一个PDB中断。如果SC[DMAEN]被设置，那么当SC[PDBIF]标志被设置时，PDB请求一个DMA传输。

MOD寄存器中的模数值用于在计数结束后将计数器重置为零。如果SC[CONT]被设置，那么计数器就会恢复新的计数；否则，计数器的运行将停止，直到下一个触发输入事件发生。

## 41.4.2 PDB触发输入源选择

PDB有多达3个触发输入源：软件触发、内部触发、外部触发（通过一个引脚）。它们被连接到片上或片下的事件源。PDB可以通过SC[SWTRIG]被软件触发。

关于在该MCU中实现的触发输入源，请参见芯片配置信息。

### 41.4.3 脉冲输出的

PDB可以产生可配置宽度的脉冲输出。

- 当PDB计数器达到POyDLY[DLY1]中设置的值时，则脉冲输出为高电平。
- 当PDB计数器达到POyDLY[DLY2]时，就会变成低电平。POy

DLY[DLY2]的设置可以大于或小于POyDLY[DLY1]。

ADC预触发/触发输出和脉冲输出生成具有相同的时基。
因为它们都共享PDB计数器。在该MCU中实现的脉冲输出连接在该器件的芯片配置细节中有所描述。




**图41-8.脉冲输出是如何产生的**

## 41.4.4 更新延迟寄存器

以下寄存器控制PDB操作的时间；在一些应用中，它们可能需要同时生效。

- PDB模数寄存器（MOD）。
- PDB中断延迟寄存器（IDLY）。

- PDB通道*n*延迟*m*寄存器(CHnDLYm)
- PDB 脉冲输出延时寄存器（POyDLY）。

它们的内部寄存器是有缓冲区的，任何写入它们的值都会先写入它们的缓冲区。导致其内部寄存器用缓冲区的值进行更新的情况在下表中进行了总结。

**表41-7.当延迟寄存器被更新时**

| SC[LDMOD] | 对延迟寄存器的更新 |
|---|---|
| 00 | 在向SC[LDOK]写入1后，内部寄存器立即被加载其缓冲区的值。 |
| 01 | PDB计数器达到PDB_MOD[MOD]+1的值后，1被写入SC[LDOK]。 |
| 10 | 在向SC[LDOK]写入1后，检测到一个触发输入事件。 |
| 11 | 要么PDB计数器达到PDB_MOD[MOD]+1的值，要么检测到一个触发输入事件，之后1被写入SC[LDOK]。 |

在1被写入SC[LDOK]后，缓冲区不能被写入，直到缓冲区的值被加载到其内部寄存器。当内部寄存器被加载时，SC[LDOK]被自我清除，所以应用程序代码可以读取它来确定内部寄存器的更新。

下图显示了SC[LDMOD]为00和x1时内部寄存器被更新的情况。

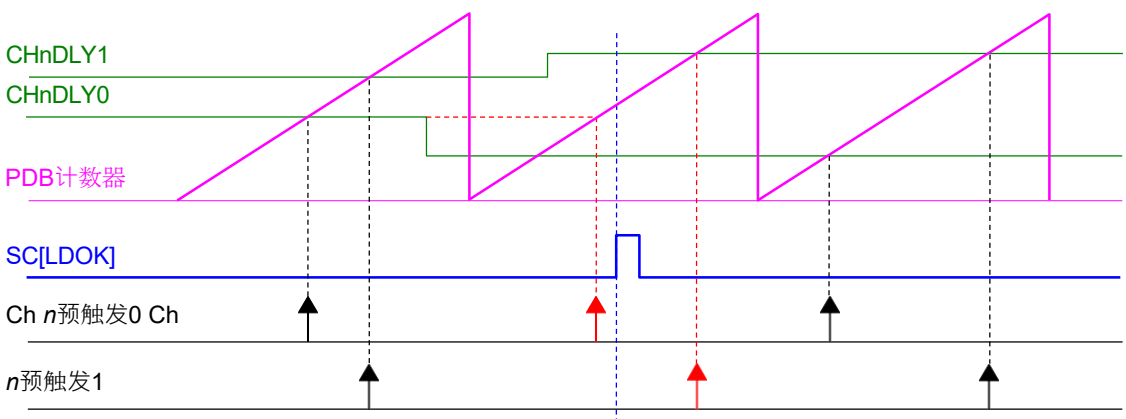



**图41-9.SC[LDMOD] = 00时的寄存器更新**

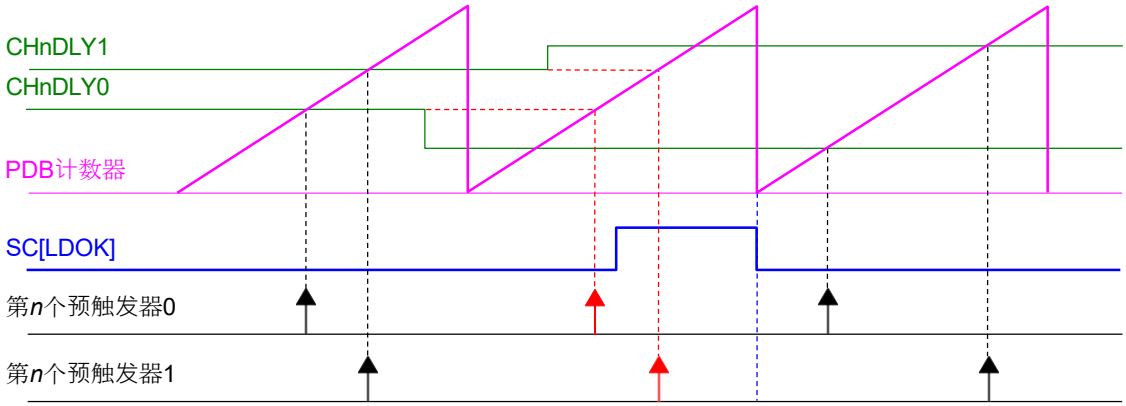



**图41-10.SC[LDMOD] = x1时的寄存器更新**

### 41.4.5 中断

PDB可以产生两个中断。PDB中断和PDB序列错误中断。下表总结了这些中断。

**表41-8.PDB中断概要**

| 中断 | 旗帜 | 启用位 |
|---|---|---|
| PDB中断 | SC[PDBIF] | SC[PDBIE] = 1和<br>SC[DMAEN] = 0 |
| PDB序列错误中断 | CHnS[ERRm] | SC[PDBEIE] = 1 |

### 41.4.6 DMA

如果SC[DMAEN]被设置，当SC[PDBIF]被设置时，PDB可以产生一个DMA传输请求。当DMA被启用时，PDB的中断不会被发出。

## 41.5 申请信息

### 41.5.1 使用预分频器和乘法系数对时间分辨率的影响

使用预分频器和大于1的乘法系数限制了外设时钟周期方面的计数/延迟精度（以预分频器X乘法系数的模数计算）。如果乘法系数设置为1，预分频器设置为2，那么可以检测的总外设时钟的唯一值是偶数值；如果预分频器设置为4，那么可以解码为检测的总外设时钟的唯一值是mod（4），以此类推。如果应用程序需要一个非常长的延迟值并使用预分频器设置为128，那么分辨率将被限制在128个外设时钟周期。

因此，对于一个特定的应用，要使用尽可能低的预分频器和乘法系数。

# 第四十二章
# FlexTimer模块（FTM）

## 42.1 特定芯片的FTM信息

### 42.1.1 实例化信息

下表列出了产品系列中每个芯片的FTM实例，并总结了不同产品和实例的特点。

**表42-1.FTM实例和特征**

| 薯片 | 实例 | 渠道 | 故障输入 | 基金会<br>调制 | 正交解码器 | 支持霍尔传<br>感器 | 抖动 |
|---|---|---|---|---|---|---|---|
| S32K116 | FTM0 | 8 | 4 | 是 | 没有 | 没有 | 没有 |
| | FTM1 | 8 | 4 | 没有 | 是 | 是 | 是 |
| S32K118 | FTM0 | 8 | 4 | 是 | 没有 | 没有 | 没有 |
| | FTM1 | 8 | 4 | 没有 | 是 | 是 | 是 |
| S32K142 | FTM0 | 8 | 4 | 是 | 没有 | 没有 | 没有 |
| | FTM1 | 8 | 4 | 没有 | 是 | 是 | 是 |
| | FTM2 | 8 | 4 | 没有 | 是 | 是 | 是 |
| | FTM3 | 8 | 4 | 是 | 没有 | 没有 | 没有 |
| S32K144 | FTM0 | 8 | 4 | 是 | 没有 | 没有 | 没有 |
| | FTM1 | 8 | 4 | 没有 | 是 | 是 | 没有 |
| | FTM2 | 8 | 4 | 没有 | 是 | 是 | 没有 |
| | FTM3 | 8 | 4 | 是 | 没有 | 没有 | 没有 |
| S32K146 | FTM0 | 8 | 4 | 是 | 没有 | 没有 | 没有 |
| | FTM1 | 8 | 4 | 没有 | 是 | 是 | 是 |
| | FTM2 | 8 | 4 | 没有 | 是 | 是 | 是 |
| | FTM3 | 8 | 4 | 是 | 没有 | 没有 | 没有 |
| | FTM4 | 8 | 2 | 没有 | 没有 | 没有 | 没有 |
| | FTM5 | 8 | 2 | 没有 | 没有 | 没有 | 没有 |
| S32K148 | FTM0 | 8 | 4 | 是 | 没有 | 没有 | 没有 |
| | FTM1 | 8 | 4 | 没有 | 是 | 是 | 是 |

*表继续在下一页...*

表42-1.FTM实例和特征（续）

| 薯片 | 实例 | 渠道 | 故障输入 | 基金会<br>调制 | 正交解码器 | 支持霍尔传<br>感器 | 抖动 |
|---|---|---|---|---|---|---|---|
| | FTM2 | 8 | 4 | 没有 | 是 | 是 | 是 |
| | FTM3 | 8 | 4 | 是 | 没有 | 没有 | 没有 |
| | FTM4 | 8 | 2 | 没有 | 没有 | 没有 | 没有 |
| | FTM5 | 8 | 2 | 没有 | 没有 | 没有 | 没有 |
| | FTM6 | 8 | 2 | 没有 | 没有 | 没有 | 没有 |
| | FTM7 | 8 | 2 | 没有 | 没有 | 没有 | 没有 |

**注意事项**

并非所有显示在FTM内存图中的寄存器都在S32K14x的每个变体/实例中实现。请参阅上表，了解可用的功能，从而了解寄存器的实现。

对于该系列的所有产品，所有FTM实例都具有这些功能。

- 全球时基
- 输入通道的过滤器
- 故障通道上的过滤器
- 半周期重载
- 分开通道对的死机时间

FTM全局负载启用是通过SIM_FTMOPT1[FTMGLDOK]实现的。请参考全局负载和SIM_FTMOPT1[FTMGLDOK]。

该设备不支持等待模式。关于可用的电源模式的详细信息，请参见模块在可用的低功率模式下的操作。

## 42.1.2 FTM中断

FlexTimer有多个中断源。请参考本参考手册所附的DMA_Interrupt_mapping.xlsm。当FTM中断发生时，读取FTM状态寄存器（FMS、SC和STATUS）以确定确切的中断源。

## 42.1.3 FTM 故障检测输入

通过SIM_FTMOPT0寄存器，可以选择FTM模块的以下故障检测输入选项。默认情

况下选择外部引脚选项。

- FTM0 FAULT0 = FTM0_FLT0引脚或TRGMUX输出
- FTM0 FAULT1 = FTM0_FLT1引脚或TRGMUX输出
- FTM0 FAULT2 = FTM0_FLT2引脚或TRGMUX输出
- FTM0 FAULT3 = FTM0_FLT3 引脚

- FTM1 FAULT0 = FTM1_FLT0引脚或TRGMUX输出
- FTM1 FAULT1 = FTM1_FLT1引脚或TRGMUX输出
- FTM1 FAULT2 = FTM1_FLT2引脚或TRGMUX输出
- FTM1 FAULT3 = FTM1_FLT3 引脚

- FTM2 FAULT0 = FTM2_FLT0引脚或TRGMUX输出
- FTM2 FAULT1 = FTM2_FLT1引脚或TRGMUX输出
- FTM2 FAULT2 = FTM2_FLT2引脚或TRGMUX输出
- FTM2 FAULT3 = FTM2_FLT3 引脚

- FTM3 FAULT0 = FTM3_FLT0引脚或TRGMUX输出
- FTM3 FAULT1 = FTM3_FLT1引脚或TRGMUX输出
- FTM3 FAULT2 = FTM3_FLT2引脚或TRGMUX输出
- FTM3 FAULT3 = FTM3_FLT3 引脚
- FTM4 FAULT0 = FTM4_FLT0引脚

- FTM4 FAULT1 = FTM4_FLT1 引脚

- FTM5 FAULT0 = FTM5_FLT0 引脚

- FTM5 FAULT1 = FTM5_FLT1 引脚

- FTM6 FAULT0 = FTM6_FLT0 引脚

- FTM6 FAULT1 = FTM6_FLT1 引脚

- FTM7 FAULT0 = FTM7_FLT0 引脚

- FTM7 FAULT1 = FTM7_FLT1 引脚

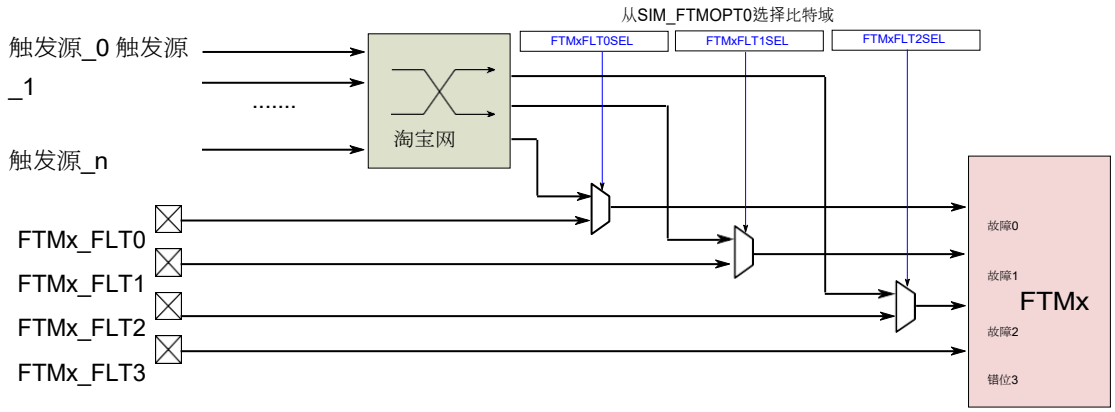



**图42-1.故障检测输入**

## 42.1.4 FTM硬件触发器和同步化

FlexTimer支持外部硬件触发输入，可用于多个FlexTimer之间的定时器动态同步或计数器复位。FlexTimer的硬件触发器的实现方式如下。

FTM0。

- FTM0硬件触发器0=TRGMUX触发输出
- FTM0硬件触发1 = SIM_FTMOPT1[FTM0SYNCBIT]
- FTM0硬件触发器2=FTM0_FLT0引脚FTM

1。

- FTM1硬件触发 0 = TRGMUX触发输出
- FTM1硬件触发1 = SIM_FTMOPT1[FTM1SYNCBIT]
- FTM1硬件触发器2=FTM1_FLT0引脚FTM

2。

- FTM2硬件触发 0 = TRGMUX触发输出
- FTM2硬件触发1 = SIM_FTMOPT1[FTM2SYNCBIT]
- FTM2硬件触发器2=FTM2_FLT0引脚FTM

3。

- FTM3硬件触发器 0 = TRGMUX触发输出
- FTM3硬件触发1 = SIM_FTMOPT1[FTM3SYNCBIT]
- FTM3硬件触发器2 = FTM3_FLT0引脚

- FTM4硬件触发器 0 = TRGMUX触发输出
- FTM4硬件触发1 = SIM_FTMOPT1[FTM4SYNCBIT]
- FTM4硬件触发器2 = FTM4_FLT0引脚

FTM5

- FTM5硬件触发器 0 = TRGMUX触发输出
- FTM5硬件触发1 = SIM_FTMOPT1[FTM5SYNCBIT]
- FTM5硬件触发器2 = FTM5_FLT0引脚

FTM6

- FTM6硬件触发器 0 = TRGMUX触发输出
- FTM6硬件触发1 = SIM_FTMOPT1[FTM6SYNCBIT]
- FTM6硬件触发器2 = FTM6_FLT0引脚

FTM7

- FTM7硬件触发器 0 = TRGMUX触发输出
- FTM7硬件触发1 = SIM_FTMOPT1[FTM7SYNCBIT]
- FTM7硬件触发器2 = FTM7_FLT0引脚

硬件触发源可以通过TRGMUX来自许多其他模块，如LPIT、低功耗定时器、CMP等。它还支持FlexTimer的自触发输出，例如：计数器初始化触发（init_trig）和通道匹配触发（ext_trig），通过灵活的TRGMUX模块。

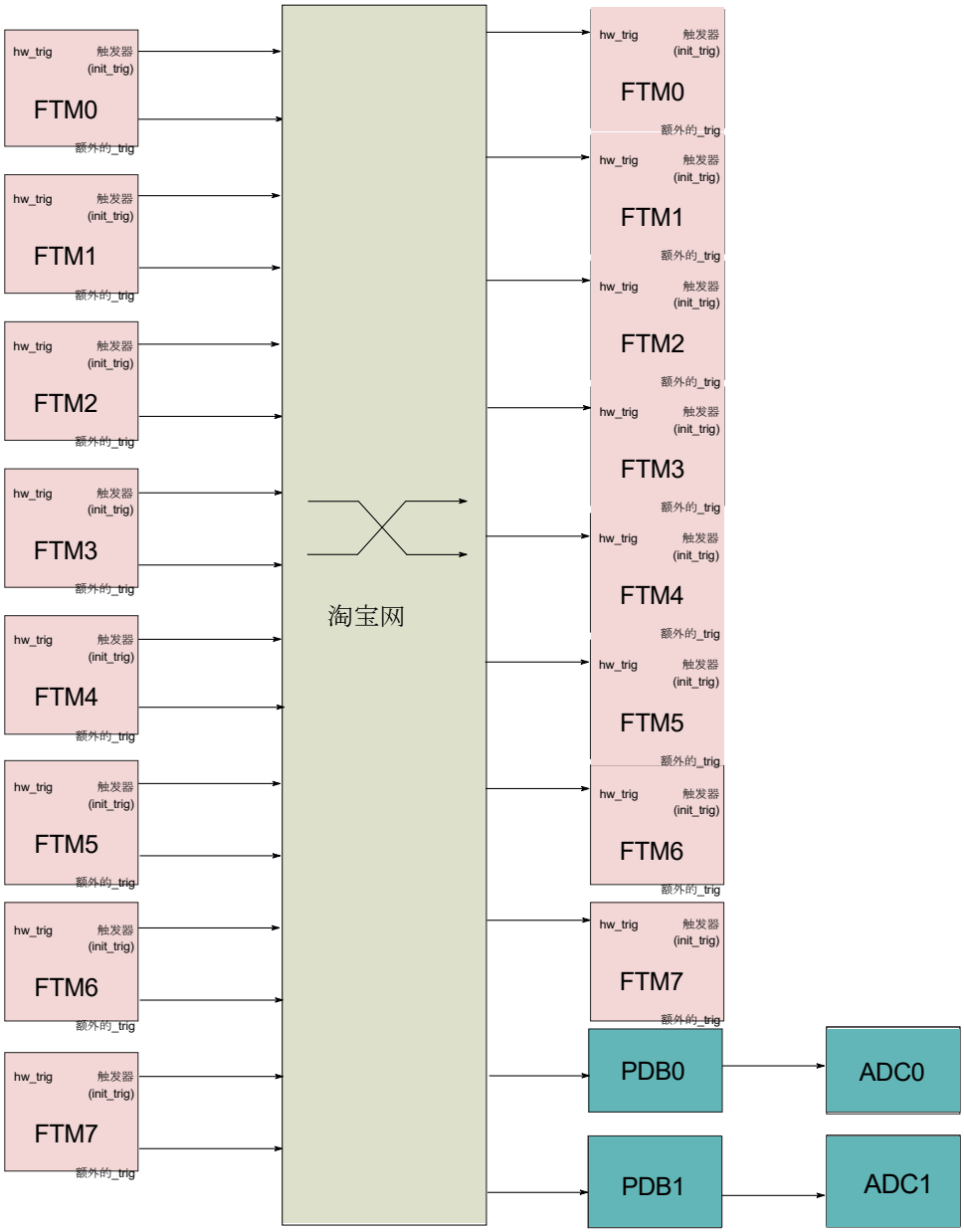



**图42-2.硬件触发器**

FlexTimer的触发输出通常也被其他模块用作触发源，例如，上图是触发PDB和ADC的情况。详见实例化信息和TRGMUX连接性模块互连性。

## 42.1.5 FTM输入捕捉选项

以下是通过SIM_FTMOPT1选择的通道0输入捕获源选项。默认情况下选择外部引脚选项。

- FTM1通道0输入捕捉=FTM1_CH0引脚或CMP0输出
- FTM2通道0输入捕捉=FTM2_CH0引脚或CMP0输出
- FTM2通道1输入捕捉=FTM2_CH1引脚或FTM2_CH0、FTM2_CH1和FTM1_CH1的独占OR。见FTM霍尔传感器支持。

## 42.1.6 支持FTM霍尔传感器

对于3相电机控制传感器的应用，使用霍尔传感器，一般是在转子周围间隔120度放置3个传感器，用于检测位置和速度。3个传感器中的每一个都提供一个脉冲，应用到一个输入捕捉引脚，然后可以进行分析并推断出速度和位置。为了简化CPU对每个霍尔传感器输入的计算，如果所有3个输入都被 "专一OR "到一个定时器通道，并且自由运行的计数器在每个边缘都被刷新，那么这可以简化速度计算。

通过SIM模块和SIM_FTMOPT1寄存器，FTM2CH1SEL位提供了正常的FTM2_CH1输入或FTM2_CH0、FTM2_CH1和FTM1_CH1引脚的XOR的选择，将应用于FTM2_CH1。

**注意事项**

如果用户利用FTM1_CH1作为FTM2_CH1的输入，FTM1_CH0仍可用于其他功能。

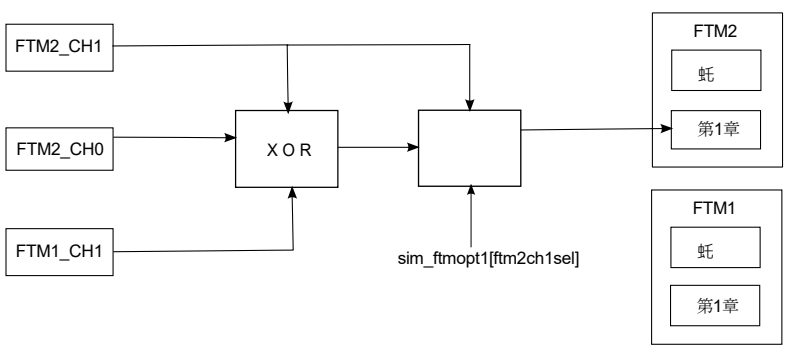



**图42-3.FTM霍尔传感器配置**

## 42.1.7 FTM调制的实现

FTM0和FTM3支持一种调制功能，当通道信号被断定为PWM或输出比较模式时，输出通道调制另一个定时器输出。FTM0的8个通道和FTM3的8个通道中的任何一

个都可以被配置为支持这种调制功能。

SIM_FTMOPT1寄存器有控制位（FTMxCHySEL），允许用户在相应的FTM定时器通道上选择正常的PWM/输出比较模式或用FTM1_CH1调制。下图显示了FTM0的实现。
FTM3也有类似的实现，由SIM_FTMOPT1[FTM3CHySEL]控制其8个通道中的每一个，可通过FTM2_CH1进行调制。更多信息见SIM章节。

当 FTM1_CH1 用于调制 FTM0 通道时，那么用户必须配置 FTM1_CH1 来提供一个频率高于调制 FTM0 通道输出的信号。同时它也限制了 FTM1_CH0 功能的使用，因为 FTM1_CH1 将被编程为提供一个 50% 占空比的 PWM 信号，并限制自由运行计数器的起始值和模数。当 FTM2_CH1 用于调制 FTM3 通道时，FTM2 也有类似的限制。

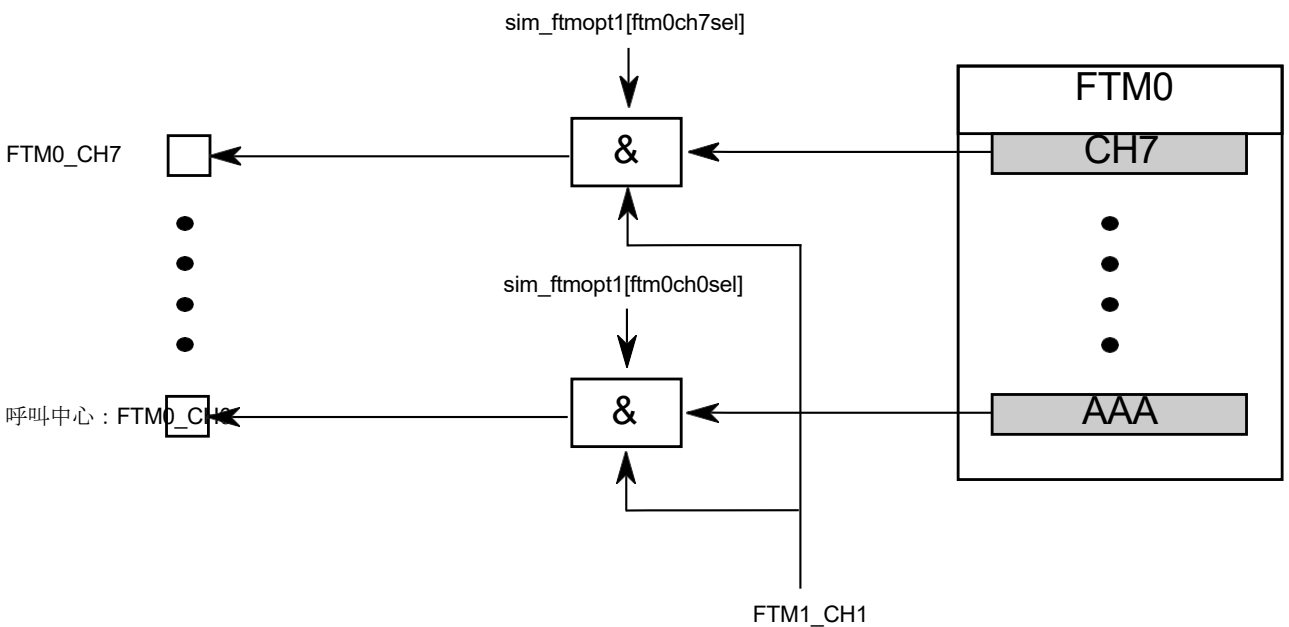



**图42-4.FTM输出调制**

## 42.1.8 FTM全球时基

该芯片提供了可选的FTM全球时基功能，见全球时基（GTB）。

FTM 通过 GTB 功能支持全局定时器基础。任何一个 FTM 模块都可以作为 GTB_EN 源。全局定时器基数只允许 FTM 计数器同步开始工作，它不自动提供 FTM 计数器的连续同步，这意味着 FTM 计数器在 FTM 误操作期间可能失去同步。

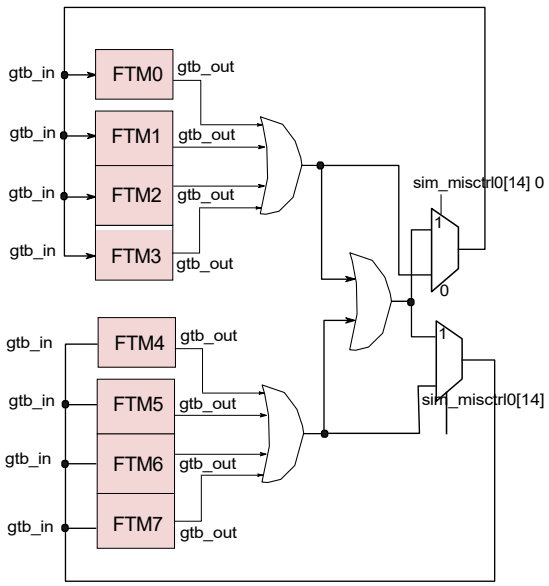



**图42-5.FTM全球时基系统互连**

## 42.1.9 FTM BDM和调试停顿模式

在FTM章节中，提到芯片处于 "BDM "与芯片处于 "调试停止模式 "是一样的。

### 42.1.9.1 FTM输出通道状态保持

该器件支持在无时钟条件下（CLKS=2'b00）的FTM通道的可选状态保持。参见SIM_MISCTRL0[FTMx_OBE_CTRL]以启用该功能。

## 42.2 简介

FlexTimer模块（FTM）是一个二至八通道的定时器，支持输入捕捉、输出比较和生成PWM信号，以控制电动马达和电源管理应用。FTM的时间基准是一个16位的计数器，可以作为无符号或有符号计数器使用。

### 42.2.1 FlexTimer理念

FlexTimer建立在一个简单的定时器上，即HCS08定时器PWM模块--TPM，在8位微控制器上使用多年。FlexTimer扩展了功能，以满足电机控制、数字照明解决方案和电源转换的需求，同时提供低成本和向后兼容TPM模块。

做出了几个关键性的改进。

- 签收的柜台
- 死亡时间插入硬件
- 故障控制输入
- 增强的触发功能
- 初始化和极性控制

所有与TPM通用的功能都有完全向后兼容的寄存器分配。FlexTimer也可以使用同一核心平台上的代码，而不需要改变，以执行相同的功能。

电机控制和功率转换功能是通过一组专门的寄存器增加的，默认情况下关闭所有新功能。新功能，如硬件死区插入、极性、故障控制、输出强制和屏蔽，大大减少了执行软件的负载，通常每个都由一组寄存器控制。

FlexTimer输入触发器可以来自比较器、ADC或其他子模块，以自动启动定时器功能。在子模块的集成过程中，这些触发器可以以各种方式连接，所以请注意用于FlexTimer配置的选项。

假设每个FlexTimer的初始化、输入时钟、初始和最终计数值都相同，那么多个FlexTimer可以同步提供一个更大的定时器，其计数器联合递增。

所有主要的用户访问寄存器都是缓冲的，以减轻执行软件的负担。有许多触发器选项可以决定哪些寄存器被用户定义的数据所更新。

### 42.2.2 特点

FTM的功能包括。

- 可选择FTM源时钟

  - 源时钟可以是FTM输入时钟，固定频率时钟，或一个外部时钟
  - 固定频率时钟是一个额外的时钟输入，允许选择FTM输入时钟以外的片上时钟源。
  - 选择外部时钟将FTM时钟连接到一个芯片级输入引脚，因此可以使FTM计数器与一个非芯片时钟源同步。
- 预分频器除以1、2、4、8、16、32、64或128
- 16位计数器
  - 它可以是一个自由运行的计数器，也可以是一个有初值和终值的计数器
  - 计数可以是向上或向下的
- 每个通道都可以配置为输入捕获、输出比较或边缘对齐的PWM模式
- 在输入捕捉模式下。
  - 捕获可以发生在上升沿、下降沿或两个沿上。
  - 可以为某些通道选择一个输入滤波器。所有滤波器都有一个唯一的预分频器
- 在输出比较模式下，输出信号可以被设置、清除或切换到匹配状态
- 所有通道都可以配置为中心对齐的PWM模式
- 每一对通道可以组合起来产生一个PWM信号，并对PWM信号的两个边沿进行独立控制
- FTM通道可以作为具有相同输出的一对，具有互补输出的一对，或具有独立输出的独立通道运行。
- 每个互补对都有死机时间的插入
- 匹配触发器的生成
- 对PWM输出的软件控制
- 多达4个故障输入，用于全局故障控制
- 每个通道的极性是可配置的
- 每个通道产生一个中断
- 当计数器溢出时产生一个中断

- 当检测到故障条件时，产生一个中断
- 当一个寄存器重载点发生时，产生一个中断
- 写缓冲的FTM寄存器的同步加载
- 半周期和全周期寄存器重载能力
- 关键寄存器的写保护
- 向后兼容TPM
- 测试输入捕捉模式
- 直接访问输入引脚状态
- 脉冲和周期宽度测量的双边缘捕获
- 带输入滤波器的正交解码器、相对位置计数、位置计数中断或在外部事件中捕获位置计数
- FTM通道可以选择在通道输出上产生一个触发脉冲，而不是一个PWM。
- 抖动能力，模拟PWM周期或PWM占空比的精细边缘控制

### 42.2.3 操作模式

当芯片处于活动的调试模式时，FTM
暂时中止所有计数，直到芯片恢复到正常的用户操作模式。在停止模式期间，所有的FTM输入时钟被停止，因此FTM被有效地禁用，直到时钟恢复。在等待模式期间，FTM继续正常工作。如果FTM不需要产生实时参考或提供从Wait模式唤醒芯片所需的中断源，那么在进入Wait模式前禁用FTM功能就可以节省电源。

### 42.2.4 框图

FTM每个通道使用一个输入/输出（I/O）引脚，CHn（FTM通道（n）），其中n是通道编号（0-7）。

**注意事项**

一个芯片上的FTM模块的每个实例所支持的通道数量可能

不同。请参见芯片特定的FTM

信息来查看每个模块实例支持多少个通道。例如，如果一个模块实例只支持6个通道，那么对通道号6和7的引用就不适用于该实例。

下图显示了FTM的结构。FTM的中心部件是16位计数器，其初始值和最终值都是可编程的，其计数可以是向上或向下的。

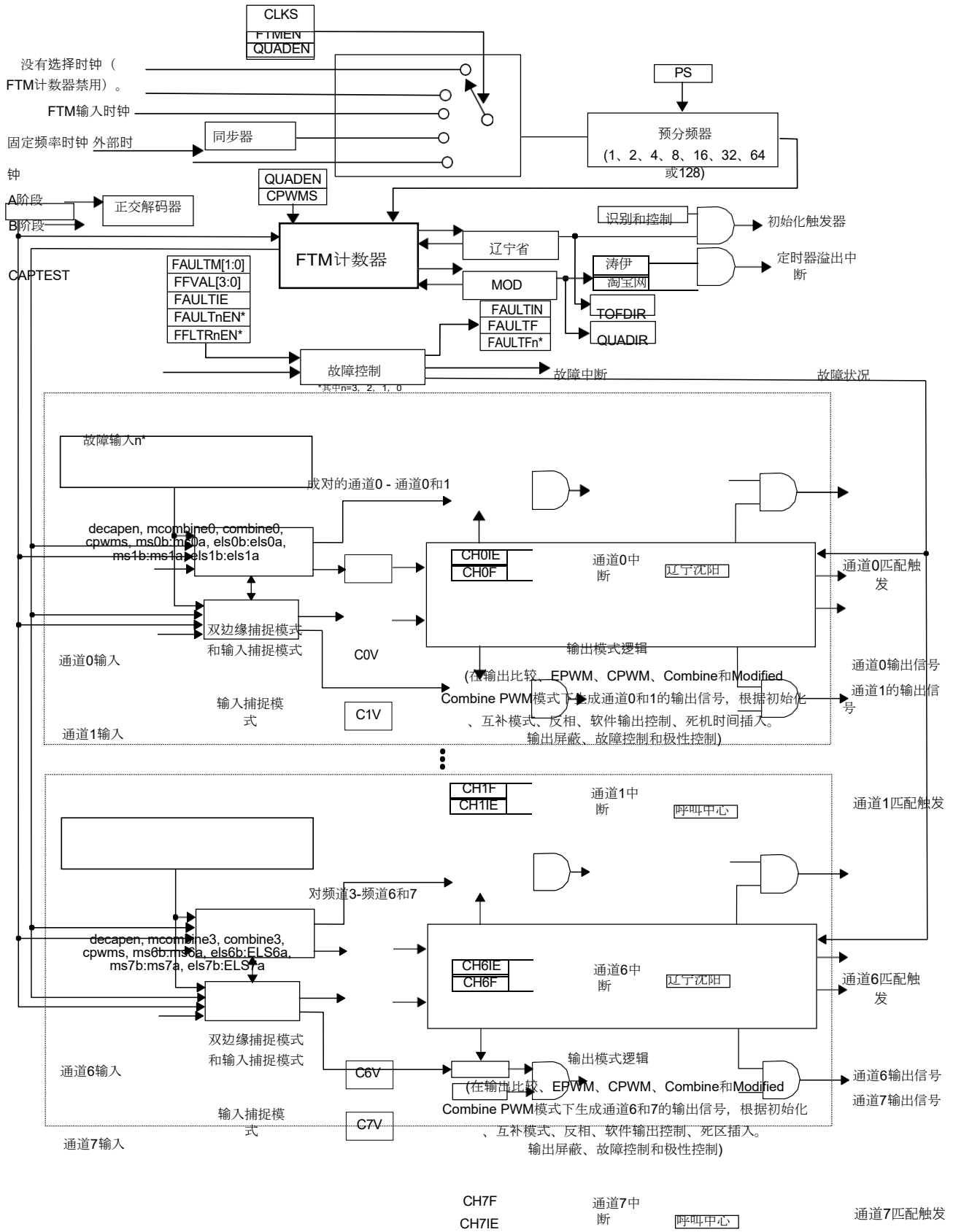

CLKS
FTMEN
QUADEN
没有选择时钟（
FTM计数器禁用）。
FTM输入时钟
固定频率时钟 外部时
钟
同步器
PS
预分频器
(1、2、4、8、16、32、64
或128)
QUADEN
CPWMS
A阶段
B阶段
正交解码器
识别和控制
初始化触发器
FTM计数器
辽宁省
MOD
涛伊
淘宝网
定时器溢出中
断
CAPTEST
FAULTM[1:0]
FFVAL[3:0]
FAULTIE
FAULTnEN*
FFLTRnEN*
FAULTIN
FAULTF
FAULTFn*
TOFDIR
QUADIR
故障控制
*其中n=3、2、1、0
故障中断
故障状况
故障输入n*
成对的通道0 - 通道0和1
decapen, mcombine0, combine0,
cpwms, ms0b:ms0a, els0b:els0a,
ms1b:ms1a, els1b:els1a
CH0IE
CH0F
通道0中
断
辽宁沈阳
通道0匹配触
发
双边缘捕捉模式
和输入捕捉模式
通道0输入
C0V
输出模式逻辑
(在输出比较、EPWM、CPWM、Combine和Modified
Combine PWM模式下生成通道0和1的输出信号，根据初始化
、互补模式、反相、软件输出控制、死机时间插入。
输出屏蔽、故障控制和极性控制)
通道0输出信号
通道1的输出信
号
输入捕捉模
式
C1V
通道1输入
CH1F
CH1IE
通道1中
断
呼叫中心
通道1匹配触发
对频道3-频道6和7
decapen, mcombine3, combine3,
cpwms, ms6b:ms6a, els6b:ELS6a,
ms7b:ms7a, els7b:ELS7a
CH6IE
CH6F
通道6中
断
辽宁沈阳
通道6匹配触
发
双边缘捕捉模式
和输入捕捉模式
通道6输入
C6V
输出模式逻辑
(在输出比较、EPWM、CPWM、Combine和Modified
Combine PWM模式下生成通道6和7的输出信号，根据初始化
、互补模式、反相、软件输出控制、死区插入。
输出屏蔽、故障控制和极性控制)
通道6输出信号
通道7输出信号
输入捕捉模
式
C7V
通道7输入
CH7F
CH7IE
通道7中
断
呼叫中心
通道7匹配触发

图42-6.FTM方框图

## 42.3 FTM信号描述

表42-2显示了FTM的用户可访问的信号。

**表42-2.FTM信号描述**

| 信号 | 描述 | 输入/输出 | 职能 |
|---|---|---|---|
| 呼叫中心 | 外部时钟。可以选择FTM外部时钟来驱动FTM计数器。 | I | 如果由 SC 寄存器中的 CLKS[1:0]位选择，外部时钟输入信号被用作 FTM 计数器时钟。这个时钟信号不能超过 FTM 输入时钟频率的 1/4。当选择外部时钟时，FTM 计数器预分频器的选择和设置也会被使用。 |
| ĂĂĂ | FTM通道（n），其中n可以是7-0 | 输入/输出 | 每个FTM通道可以被配置为输入或输出操作。与每个通道相关的方向，输入或输出，是根据为该通道分配的模式来选择的。 |
| 违约金j | 故障输入（j），其中j可以是3-0 | I | 故障输入信号用于控制CHn通道的输出状态。如果检测到故障，FAULTj 信号被断言，通道输出被置于安全状态。故障逻辑的行为由 MODE 寄存器中的 FAULTM[1:0] 控制位和 COMBINE 寄存器中的 FAULTEN 位定义。请注意，每个 FAULTj 输入可能有选择地影响所有通道，因为 FAULTM[1:0] 和 FAULTEN 控制位是为每对通道定义的。因为有几个 FAULTj输入，FTM模块最多有4个，每个输入都是由 FLTCTRL寄存器中的FAULTjEN位激活。 |
| PHA | 正交解码器A相输入。与正交解码器相位A有关的输入引脚。 | I | 正交解码器 A 相输入信号在选择正交解码器模式时被使用。A相输入信号是控制正交解码器模式下FTM计数器增量或减量的信号之一。 |
| PHB | 正交解码器B相输入。与正交解码器相位B有关的输入引脚。 | I | 正交解码器 B 相输入信号在选择正交解码器模式时被使用。B相输入信号是控制正交解码器模式下FTM计数器增量或减量的信号之一。 |

## 42.4 内存图和寄存器定义

### 42.4.1 记忆地图

本节介绍了FTM寄存器的高层次概要，以及它们是如何被映射的。

FTM中不可用的功能的寄存器和位仍保留在内存图和复位值中，但它们没有活动

功能。

**注意事项**

一个芯片上的FTM模块的每个实例所支持的通道数可能不同。请参阅特定芯片的FTM信息，以了解每个模块实例支持多少个通道。例如，如果一个模块实例只支持6个通道，对通道号6和7的引用不适用于该实例。

**注意事项**

当FTMEN=0时，不要从CNTIN寄存器中写入该区域。

## 42.4.2 登记簿描述

对保留地址的访问会导致传输错误。不存在的通道的寄存器被认为是保留的。双缓冲寄存器的写入必须使用32位操作。

## 42.4.3 FTM寄存器描述

### 42.4.3.1 FTM 内存地图

FTM0 基准地址。4003_8000h

FTM1 基准地址。4003_9000h

FTM2 基准地址。4003_A000h

FTM3 基准地址。4002_6000h

FTM4 基准地址。4006_E000h

FTM5 基准地址。4006_F000h

FTM6 基准地址。4007_0000h

FTM7 基准地址。4007_1000h

| 偏移 | 注册 | 宽度<br>(单位：比特) | 访问 | 重置值 |
|---|---|---|---|---|
| 0h | 状态和控制 (SC) | 32 | RW | 0000_0000h |
| 4h | 计数器 (CNT) | 32 | RW | 0000_0000h |
| 8h | 模数(MOD) | 32 | RW | 0000_0000h |
| 丘吉尔 | 通道（n）状态和控制（C0SC）。 | 32 | RW | 0000_0000h |
| 10h | 通道(n)值(C0V) | 32 | RW | 0000_0000h |
| 14h | 通道（n）状态和控制（C1SC）。 | 32 | RW | 0000_0000h |
| 18h | 通道（n）值（C1V） | 32 | RW | 0000_0000h |
| 1Ch | 通道（n）状态和控制（C2SC）。 | 32 | RW | 0000_0000h |
| 20h | 通道(n)值(C2V) | 32 | RW | 0000_0000h |
| 24h | 通道（n）状态和控制（C3SC）。 | 32 | RW | 0000_0000h |
| 28h | 通道（n）值（C3V） | 32 | RW | 0000_0000h |
| 2Ch | 通道（n）状态和控制（C4SC）。 | 32 | RW | 0000_0000h |
| 30h | 通道(n)值(C4V) | 32 | RW | 0000_0000h |
| 34h | 通道（n）状态和控制（C5SC）。 | 32 | RW | 0000_0000h |
| 38h | 通道（n）值（C5V） | 32 | RW | 0000_0000h |
| 3Ch | 通道（n）状态和控制（C6SC）。 | 32 | RW | 0000_0000h |
| 40h | 通道(n)值(C6V) | 32 | RW | 0000_0000h |
| 44h | 通道（n）状态和控制（C7SC）。 | 32 | RW | 0000_0000h |
| 48h | 通道（n）值（C7V） | 32 | RW | 0000_0000h |
| 4Ch | 计数器初始值（CNTIN） | 32 | RW | 0000_0000h |
| 50h | 捕获和比较状态（STATUS） | 32 | ROWZ | 0000_0000h |
| 54h | 功能模式选择（MODE） | 32 | RW | 0000_0004h |
| 58h | 同步（SYNC）。 | 32 | RW | 0000_0000h |
| 5Ch | 通道输出的初始状态(OUTINIT) | 32 | RW | 0000_0000h |
| 60h | 输出屏蔽（OUTMASK） | 32 | RW | 0000_0000h |
| 64h | 链接通道的功能 (COMBINE) | 32 | RW | 0000_0000h |
| 68h | 死亡时间配置（DEADTIME） | 32 | RW | 0000_0000h |
| 6Ch | FTM 外部触发器 (EXTTRIG) | 32 | RW | 0000_0000h |
| 70h | 通道的极性（POL）。 | 32 | RW | 0000_0000h |
| 74h | 故障模式状态（FMS） | 32 | RW | 0000_0000h |
| 78h | 输入捕获过滤器控制（FILTER） | 32 | RW | 0000_0000h |
| 7Ch | 故障控制（FLTCTRL） | 32 | RW | 0000_0000h |

*表继续在下一页...*

| | | | | |
|---|---|---|---|---|
| 80h | 正交解码器控制和状态（QDCTRL） | 32 | RW | 0000_0000h |
| 84h | 配置 (CONF) | 32 | RW | 0000_0000h |
| 88h | FTM 故障输入极性 (FLTPOL) | 32 | RW | 0000_0000h |
| 8Ch | 同步配置（SYNCONF） | 32 | RW | 0000_0000h |
| 90h | FTM反转控制 (INVCTRL) | 32 | RW | 0000_0000h |
| 94h | FTM 软件输出控制 (SWOCTRL) | 32 | RW | 0000_0000h |
| 98h | FTM PWM负载（PWMLOAD）。 | 32 | RW | 0000_0000h |
| 9Ch | 半周期寄存器（HCR） | 32 | RW | 0000_0000h |

| 偏移 | 注册 | 宽度<br>(单位：比特) | 访问 | 重置值 |
|---|---|---|---|---|
| A0h | 第0对死机时间配置（PAIR0DEADTIME）。 | 32 | RW | 0000_0000h |
| A8h | 第1对死机时间配置（PAIR1DEADTIME） | 32 | RW | 0000_0000h |
| B0h | 第2对死机时间配置（PAIR2DEADTIME） | 32 | RW | 0000_0000h |
| B8h | 第3对死机时间配置（PAIR3DEADTIME） | 32 | RW | 0000_0000h |
| 200h | 模数值的镜像 (MOD_MIRROR) | 32 | RW | 0000_0000h |
| 204h - 220h | 通道的镜像（n）匹配值（C0V_MIRROR - C7V_MIRROR）。 | 32 | RW | 0000_0000h |

## 42.4.3.2 状态和控制 (SC)

### 42.4.3.2.1 偏移

| 注册 | 偏移 |
|---|---|
| SC | 0h |

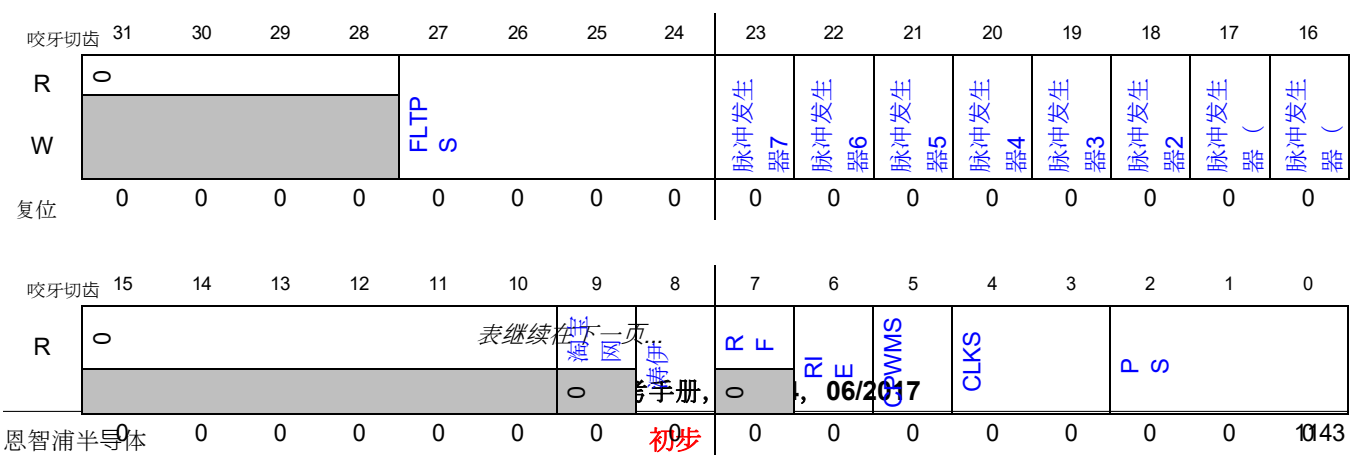
### 42.4.3.2.2 职能

SC包含溢出状态标志和控制位，用于配置中断使能、FTM配置、时钟源、滤波器预分频和预分频因子。

该寄存器还包含输出使能控制位和重载机会标志控制。

这些控制与本模块内的所有通道有关。

### 42.4.3.2.3 图示




表继续在下一页...

W

复位

### 42.4.3.2.4 领域

| 场地 | 职能 |
|---|---|
| 31-28<br>- | 保留 |
| 27-24<br>FLTPS | 滤波器预分频器<br>位FLTPS选择FTM滤波器中使用的时钟预分频器。<br>• 通道输入滤波器<br>• 故障输入滤波器<br>写入位FLTPS有即时效果。<br>0000b - 除以1 0001b - 除以2 0010b - 除以3 0011b - 除以4 0100b - 除以5 0101b - 除以6 0110b - 除以7 0111b - 除以8 1000b - 除以9 1001b - 除以10 1010b - 除以11 1011b - 除以12 1100b - 除以13 1101b - 除以14 1110b - 除以15 111b - 除以16 |
| 23<br>脉冲发生器（PWMEN7 | 通道7 PWM启用位<br>该位启用PWM通道的输出。当使用输入模式时，该位应被设置为0（输出禁用）。<br>0b - 通道输出端口被禁用 1b - 通道输出端口被启用 |
| 22<br>脉冲发生器6 | 通道6 PWM启用位<br>该位启用PWM通道的输出。当使用输入模式时，该位应被设置为0（输出禁用）。<br>0b - 通道输出端口被禁用 1b - 通道输出端口被启用 |
| 21<br>脉冲发生器5 | 通道5 PWM启用位<br>该位启用PWM通道的输出。当使用输入模式时，该位应被设置为0（输出禁用）。<br>0b - 通道输出端口被禁用 1b - 通道输出端口被启用 |

*表继续在下一页...*

| | |
|---|---|
| 20<br><br>脉冲发生器4 | 通道4 PWM启用位<br><br>该位启用PWM通道的输出。当使用输入模式时，该位应被设置为0（输出禁用）。<br><br>0b - 通道输出端口被禁用 1b - 通道输出端口被启用 |
| 19 | 通道3 PWM启用位 |

| 场地 | 职能 |
| --- | --- |
| 脉冲发生器3 | 该位启用PWM通道的输出。当使用输入模式时，该位应被设置为0（输出禁用）。<br>0b - 通道输出端口被禁用 1b - 通道输出端口被启用 |
| 18<br>脉冲发生器2 | 通道2 PWM启用位<br>该位启用PWM通道的输出。当使用输入模式时，该位应被设置为0（输出禁用）。<br>0b - 通道输出端口被禁用 1b - 通道输出端口被启用 |
| 17<br>脉冲发生器（PWMEN1 | 通道1 PWM启用位<br>该位启用PWM通道的输出。当使用输入模式时，该位应被设置为0（输出禁用）。<br>0b - 通道输出端口被禁用 1b - 通道输出端口被启用 |
| 16<br>脉冲发生器（PWMEN0 | 通道0 PWM启用位<br>该位启用PWM通道的输出。当使用输入模式时，该位应被设置为0（输出禁用）。<br>0b - 通道输出端口被禁用 1b - 通道输出端口被启用 |
| 15-10<br>- | 保留 |
| 9<br>淘宝网 | 定时器溢出标志<br>当FTM计数器超过MOD寄存器中的值时，由硬件设置。当TOF被设置时，通过读取SC寄存器，然后向TOF位写0，可以清除TOF位。写1到TOF位没有效果。<br>如果在读和写操作之间发生了另一个FTM溢出，则写操作没有效果；因此，TOF保持设置，表明发生了溢出。在这种情况下，一个TOF中断请求不会因为前一个TOF的清零顺序而丢失。<br>0b - FTM计数器没有溢出。1b - FTM计数器已经溢出。 |
| 8<br>涛伊 | 定时器溢出中断启用<br>启用FTM溢出中断。<br>0b - 禁用TOF中断。使用软件轮询。<br>1b - 启用TOF中断。当TOF等于1时，会产生一个中断。 |
| 7<br>射频 | 重新加载标志<br>RF位在每个选定的重载点被设置。见重载点。<br>当RF被设置时，通过读取SC寄存器，然后向RF位写0，可以清除RF位。向RF写1没有效果。如果在读和写操作之间发生了另一个选定的重载点，那么写操作就没有效果；因此，RF仍然被设置。<br>0b - 没有发生选定的重新装载点。1b - 发生了一个选定的重新装载点。 |

*表继续在下一页...*

| | |
|---|---|
| 6<br>RIE | 重载点中断启用 启用重载点中断。<br><br>0b - 重新加载点中断被禁用。 1b - 重新加载点中断被启用。 |
| 5 | 中心对齐的PWM选择 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| CPWMS | 选择CPWM模式。该模式将FTM配置为上-下计数模式运行。这个字段是受写保护的。只有当<br>MODE[WPDIS]=1时才能写入。<br>0b - FTM计数器工作在向上计数模式。<br>1b - FTM计数器工作在上-下计数模式。 |
| 4-3<br>CLKS | 时钟源选择<br>选择三个FTM计数器时钟源中的一个。<br>这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。<br>00b - 没有选择时钟。这实际上是禁用了FTM计数器。01b - FTM输<br>入时钟<br>10b - 固定频率的时钟 11b -<br>外部时钟 |
| 2-0<br>PS | 比例尺前的因素选择<br>为CLKS选择的时钟源选择8个分频因子中的一个。新的预分频器因子在新值更新到寄存器位后，会影响下一个FTM输入时钟周期的时钟源。<br>这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。<br>000b - 除以1 001b -<br>除以2 010b - 除以4<br>011b - 除以8 100b -<br>除以16 101b - 除以32<br>110b - 除以64 111b -<br>除以128 |

## 42.4.3.3 计数器 (CNT)

### 42.4.3.3.1 偏移

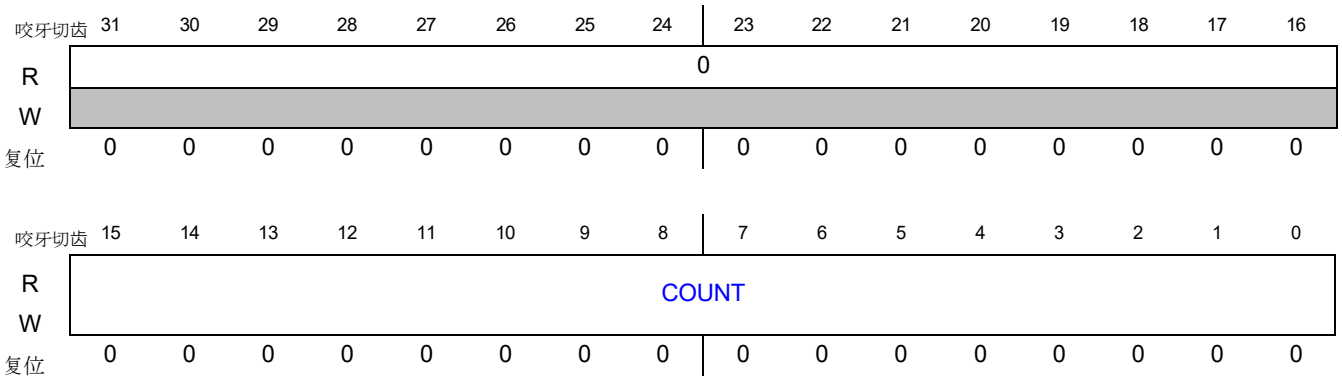
| 注册 | 偏移 |
| --- | --- |
| CNT | 4h |

### 42.4.3.3.2 职能

CNT寄存器包含FTM计数器的值。

复位会清除CNT寄存器。向COUNT写任何值，都会以其初始值CNTIN更新计数器

## 42.4.3.1148.3 图示




#### 42.4.3.3.4 领域

| 场地 | 职能 |
|---|---|
| 31-16<br>- | 保留 |
| 15-0<br>COUNT | 计数器值 |

### 42.4.3.4 模数(MOD)

#### 42.4.3.4.1 偏移

| 注册 | 偏移 |
|---|---|
| MOD | 8h |

#### 42.4.3.4.2 职能

Modulo 寄存器包含 FTM
计数器的模数值。在FTM计数器达到模数值后，溢出标志（TOF）在下一个时钟周期被设置，FTM计数器的下一个值取决于所选择的计数方法；见计数器。

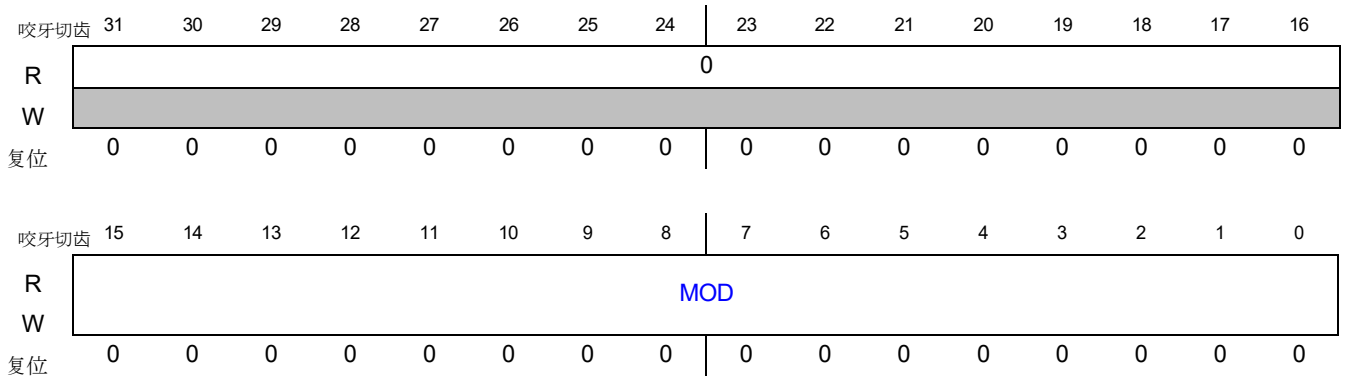
对MOD寄存器的写入是在其写缓冲器上进行的。MOD寄存器根据写缓冲区更新的寄存器的值来更新。如果FTMEN
= 0，对SC寄存器的写入会手动重置这个写入一致性机制。

42.4.3.4.2 图示

在写入MOD寄存器之前，通过写入CNT来初始化FTM计数器，以避免对第一次计数器溢出的时间产生混淆。

1150 3

### 42.4.3.4.3 图示




### 42.4.3.4.4 领域

| 场地 | 职能 |
|---|---|
| 31-16<br>- | 保留 |
| 15-0<br>MOD | MOD<br>模数值 |

## 42.4.3.5 通道(n)状态和控制(C0SC - C7SC)

### 42.4.3.5.1 偏移

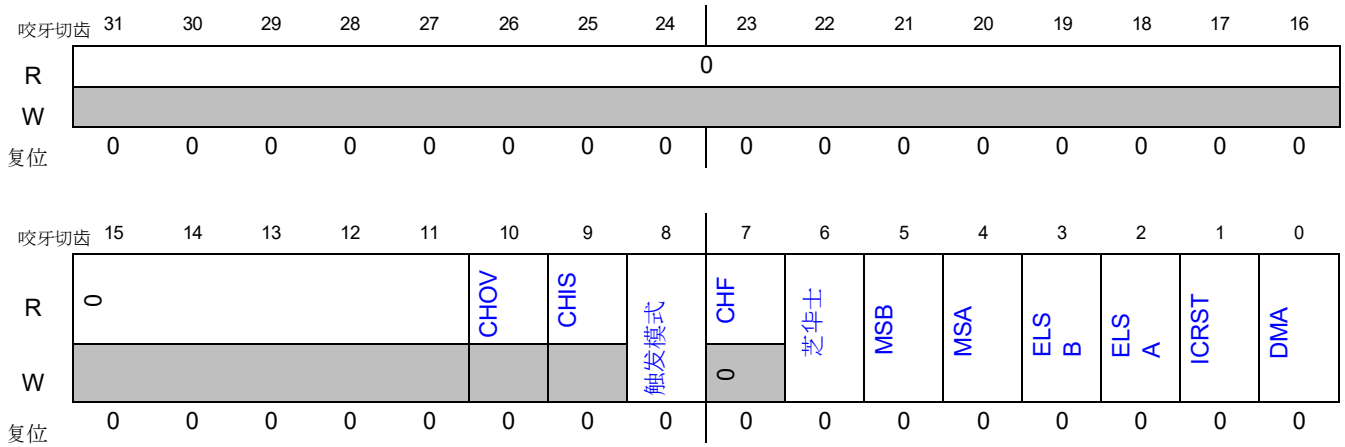
对于a=0到7。

| 注册 | 偏移 |
|---|---|
| 剑桥大学 | Ch + (a × 8h) |

### 42.4.3.5.2 职能

CnSC包含通道（n）状态位和控制位，选择通道（n）模式及其功能。

### 42.4.3.5.3 图示




### 42.4.3.5.4 领域

| 场地 | 职能 |
|---|---|
| 31-11<br>- | 保留 |
| 10<br>CHOV | 通道(n) 输出值<br>CHOV位有通道（n）输出的最终值。<br>**注意：**当通道(n)不处于输出模式时，CHOV位应被忽略。<br>0b - 通道（n）的输出为零。1b - 通道（n）的输出为1。 |
| 9<br>CHIS | 通道(n) 输入状态<br>CHIS 位有双采样或滤波后的通道（n）输入值（如果通道（n）滤波器被启用），它们都在 FTM 内。<br>**注意：**当通道（n）不处于输入模式时，CHIS位应被忽略。<br>**注意：**当一对通道处于双边缘模式时，通道（n+1）的CHIS位是通道（n+1）的输入值，而不是通道（n）的输入值（这个信号是双边缘模式使用的输入信号）。<br>0b - 通道（n）的输入为零。1b - 通道（n）的输入是1。 |
| 8<br>触发模式 | 触发模式控制<br>该位控制 FTM 通道输出的触发生成。只有当 FTM 通道被配置为 EPWM 或 CPWM 模式时，才允许这种模式。如果通道中发生匹配，将在通道输出中产生一个FTM时钟周期宽度的触发脉冲。参见通道触发输出。<br>这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。<br>0b - 通道输出将产生正常的PWM输出，不产生脉冲。 |

| | |
|---|---|
| | 1b - 如果通道中发生匹配，将在通道输出上发生触发生成。触发脉冲宽度为一个FTM时钟周期。 |
| 7<br>CHF | 通道（n）标志<br>当通道（n）上发生事件时由硬件设置。当 CHF 被设置时，通过读取 CnSC 寄存器，然后向 CHF 位写 0 来清除 CHF。向CHF写1没有效果。 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | 如果在读和写操作之间发生了另一个事件，写操作就没有效果；因此，CHF仍然被设置，表明发生了一个事件。在这种情况下，一个CHF中断请求不会因为前一个CHF的清零顺序而丢失。<br>0b - 没有发生通道（n）事件。1b - 发生了一个通道（n）事件。 |
| 6<br>芝华士 | 通道（n）中断启用<br>启用通道（n）中断。<br>0b - 禁用通道（n）中断。使用软件轮询。1b - 启用通道(n)中断。 |
| 5<br>MSB | 通道（n）模式选择<br>用于通道（n）模式的选择上。见通道模式。<br>这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。 |
| 4<br>MSA | 通道（n）模式选择<br>用于通道（n）模式的选择上。见通道模式。<br>这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。 |
| 3<br>ELSB | 通道（n）边缘或电平选择<br>用于通道（n）模式的选择上。见通道模式。<br>这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。 |
| 2<br>ELSA | 通道（n）边缘或电平选择<br>用于通道（n）模式的选择上。见通道模式。<br>这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。 |
| 1<br>ICRST | FTM计数器被选定的输入捕捉事件重置。<br>FTM计数器复位是由输入捕捉模式下通道（n）的选定事件驱动的。这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。<br>0b - 当检测到选定的通道(n)输入事件时，FTM计数器不被重置。1b - 当检测到所选通道(n)的输入事件时，FTM计数器被复位。 |
| 0<br>DMA | 启用DMA<br>启用通道的DMA传输。<br>0b - 禁用DMA传输。1b - 启用DMA传输。 |

## 42.4.3.6 通道（n）值（C0V - C7V）。

#### 42.4.3.6.1 偏移

对于a=0到7。

| 注册 | 偏移 |
| --- | --- |
| 钙钛矿 | 10h + (a × 8h) |

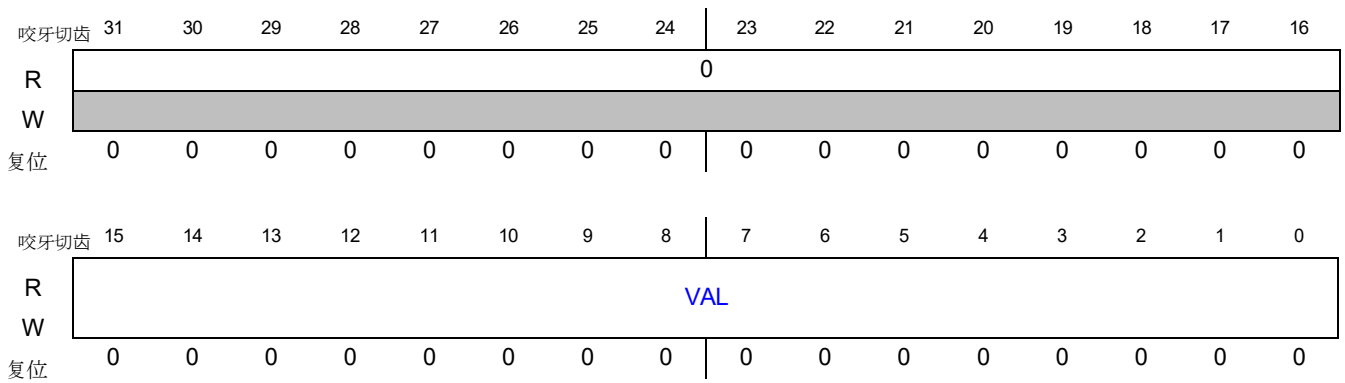
#### 42.4.3.6.2 职能

这些寄存器包含了输入模式下捕获的FTM计数器值或输出模式下的匹配值。

在输入捕捉、捕捉测试和双边缘捕捉模式下，对CnV寄存器的任何写入都被忽略。

在输出模式下，对CnV寄存器的写入是在其写缓冲器上完成的。根据从写缓冲区更新的寄存器，CnV寄存器以其写缓冲区的值进行更新。
如果 FTMEN = 0，对 CnSC 寄存器的写入将手动重置该写入一致性机制。

#### 42.4.3.6.3 图示




#### 42.4.3.6.4 领域

| 场地 | 职能 |
|---|---|
| 31-16<br>- | 保留 |
| 15-0<br>VAL | 渠道价值<br>捕获的输入模式的FTM计数器值或输出模式的匹配值 |

### 42.4.3.7 计数器初始值（CNTIN）

#### 42.4.3.7.1 偏移

| 注册 | 偏移 |
| --- | --- |
| 辽宁省 | 4Ch |

#### 42.4.3.7.2 职能

计数器初始值寄存器包含FTM计数器的初始值。

写到CNTIN寄存器中的值被锁入一个缓冲区。CNTIN寄存器根据写缓冲区的值更新，从写缓冲区更新的寄存器。

当最初选择FTM时钟时，通过向CLKS位写入非零值，FTM计数器以0x0000的值开始。为了避免这种行为，在第一次写入选择FTM时钟之前，将新值写入CNTIN寄存器，然后通过向CNT寄存器写入任何值来初始化FTM计数器。

#### 42.4.3.7.3 图示




#### 42.4.3.7.4 领域

| 场地 | 职能 |
|---|---|
| 31-16<br>- | 保留 |
| 15-0<br>归档 | 归档<br>FTM计数器的初始值 |

### 42.4.3.8 捕获和比较状态（STATUS）

#### 42.4.3.8.1 偏移

| 注册 | 偏移 |
|---|---|
| 状态 | 50h |

#### 42.4.3.8.2 职能

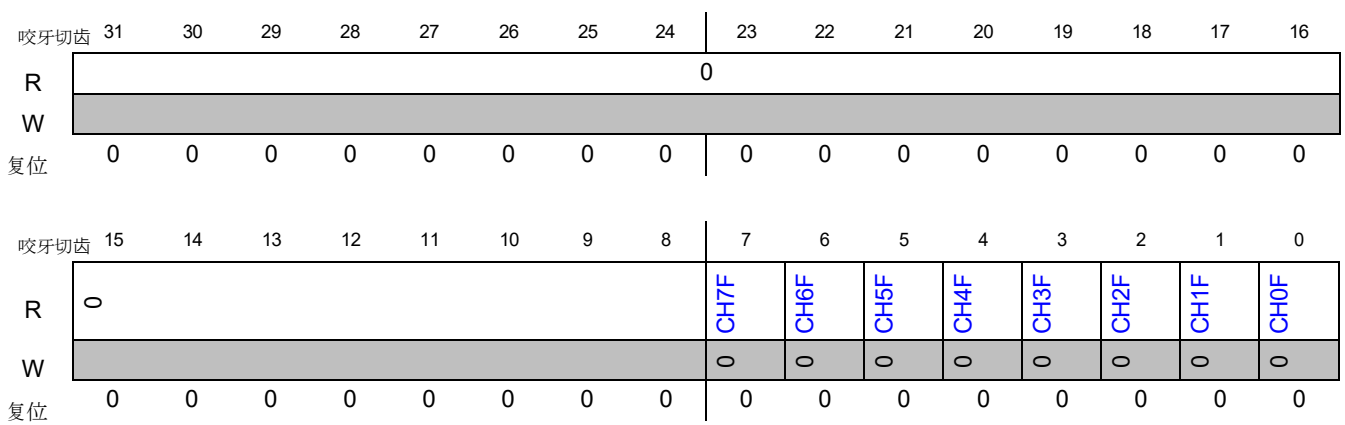
STATUS 寄存器包含每个 FTM 通道的 CnSC 中状态标志 CHF
位的副本，以方便软件使用。

STATUS中的每个CHF位都是CnSC中CHF位的镜像。所有 CHF 位都可以通过读取 STATUS
来检查。所有的CHF位可以通过读取STATUS，然后写0x00到STATUS来清除。

当通道上发生事件时，硬件设置各个通道的标志。当CHF被设置时，通过读取STATUS，然后向CHF位写0，就可以清除CHF。
给CHF写1没有效果。

如果在读和写操作之间发生了另一个事件，写操作就没有效果；因此，CHF仍然被设置，表明发生了一个事件。在这种情况下，一个CHF中断请求不会因为前一个CHF的清零序列而丢失。

#### 42.4.3.8.3 图示




#### 42.4.3.8.4 领域

| 场地 | 职能 |
|---|---|

| | |
|---|---|
| 31-8<br>- | 保留 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| 7<br>CH7F | 第7频道旗帜<br>见寄存器描述。<br>0b - 没有发生通道事件。1b - 发生了一个通道事件。 |
| 6<br>CH6F | 第6频道旗帜<br>见寄存器描述。<br>0b - 没有发生通道事件。1b - 发生了一个通道事件。 |
| 5<br>CH5F | 第5频道旗帜<br>见寄存器描述。<br>0b - 没有发生通道事件。1b - 发生了一个通道事件。 |
| 4<br>CH4F | 第4频道旗帜<br>见寄存器描述。<br>0b - 没有发生通道事件。1b - 发生了一个通道事件。 |
| 3<br>CH3F | 第3频道旗帜<br>见寄存器描述。<br>0b - 没有发生通道事件。1b - 发生了一个通道事件。 |
| 2<br>CH2F | 频道2标志<br>见寄存器描述。<br>0b - 没有发生通道事件。1b - 发生了一个通道事件。 |
| 1<br>CH1F | 通道1标志<br>见寄存器描述。<br>0b - 没有发生通道事件。1b - 发生了一个通道事件。 |
| 0<br>CH0F | 通道0标志<br>见寄存器描述。<br>0b - 没有发生通道事件。1b - 发生了一个通道事件。 |

## 42.4.3.9 功能模式选择（MODE）

## 42.4.3.9.1 偏移

| 注册 | 偏移 |
| --- | --- |
| 模式 | 54h |

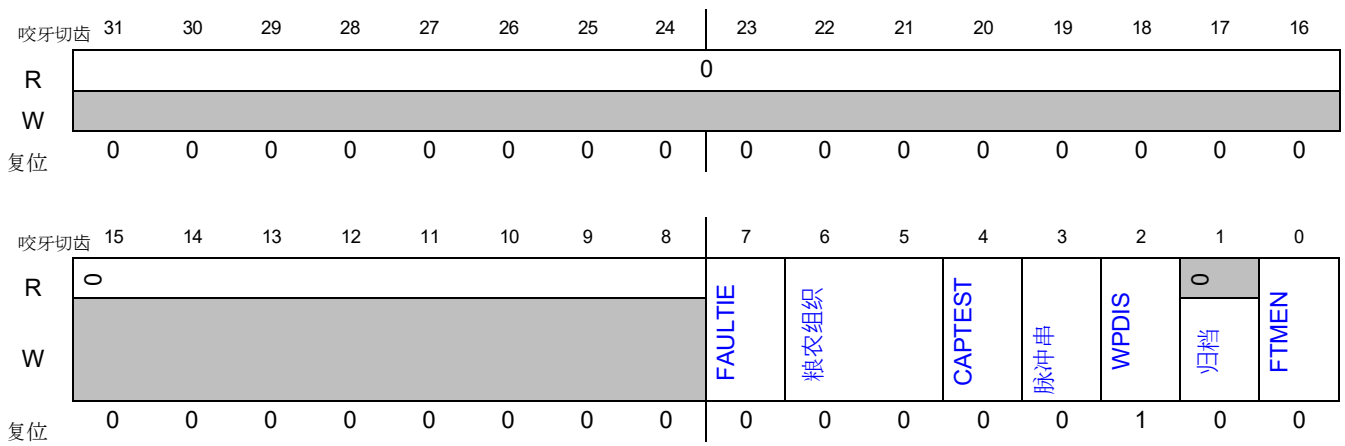
### 42.4.3.9.2 职能

该寄存器包含FTM特定功能的全局使能位和用于配置的控制位。

- 故障控制模式和中断
- 捕获测试模式
- PWM同步
- 写保护
- 通道输出初始化

这些控制与本模块内的所有通道有关。

### 42.4.3.9.3 图示




### 42.4.3.9.4 领域

| 场地 | 职能 |
|---|---|
| 31-8<br>- | 保留 |
| 7<br>FAULTIE | 故障中断启用<br>当FTM检测到故障并启用FTM故障控制时，启用产生一个中断。<br>0b - 故障控制中断被禁用。 1b - 故障控制中断被启用。 |
| 6-5<br>粮农组织 | 故障控制模式<br>定义了FTM故障控制模式。<br>这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。<br>00b - 所有通道的故障控制被禁用。<br>01b - 仅对偶数通道（通道0、2、4和6）启用故障控制，选择的模式是手动清除故障。 |

表继续在下一页...

| 场地 | 职能 |
| --- | --- |
| | 10b - 所有通道的故障控制被启用，选择的模式是手动清除故障。11b - 所有通道的故障控制被启用，选择的模式是自动故障清除。 |
| 4<br>CAPTEST | 捕获测试模式启用 捕获测试模式启<br>用。<br>这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。<br>0b - 捕获测试模式被禁用。1b - 捕<br>获测试模式被启用。 |
| 3<br>脉冲宽度控制（<br>PWMSYNC | PWM同步模式<br>选择哪些触发器可以被MOD、CnV、OUTMASK和FTM计数器同步使用。参见PWM同步。当SYNCMODE为0时，PWMSYNC位对同步进行配置。<br>0b - 无限制。软件和硬件触发器可由MOD、CnV、OUTMASK和FTM计数器同步使用。<br>1b - 软件触发只能由MOD和CnV同步使用，硬件触发只能由OUTMASK和FTM计数器同步使用。 |
| 2<br>WPDIS | 禁用写保护<br>当写保护被启用时（WPDIS = 0），写保护位不能被写入。当写保护被禁用时（WPDIS=1），可以写入受写保护的位。WPDIS位是对WPEN位的否定。当WPEN被写入1时，WPDIS被清零。当WPEN位被读为1，然后1被写入WPDIS时，WPDIS被设置。向WPDIS写0没有效果。<br>0b - 写保护被启用。1b - 写保护<br>被禁用。 |
| 1<br>归档 | 初始化通道的输出<br>当1被写入INIT位时，通道的输出根据OUTINIT寄存器中相应位的状态被初始化。将0写入INIT位没有任何作用。<br>INIT位总是读为0。 |
| 0<br>FTMEN | 启用FTM<br>这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。<br>0b - TPM兼容性。自由运行的计数器和同步与TPM兼容。1b - 自由运行计数器和同步与<br>TPM行为不同。 |

## 42.4.3.10 同步（SYNC）。

### 42.4.3.10.1 偏移

| 注册 | 偏移 |
| --- | --- |
| 同步性 | 58h |

### 42.4.3.10.2 职能

该寄存器用于配置PWM同步。

一个同步事件可以用MOD、CnV和OUTMASK寄存器的写缓冲区的值和FTM计数器的初始化来执行对它们的同步更新。

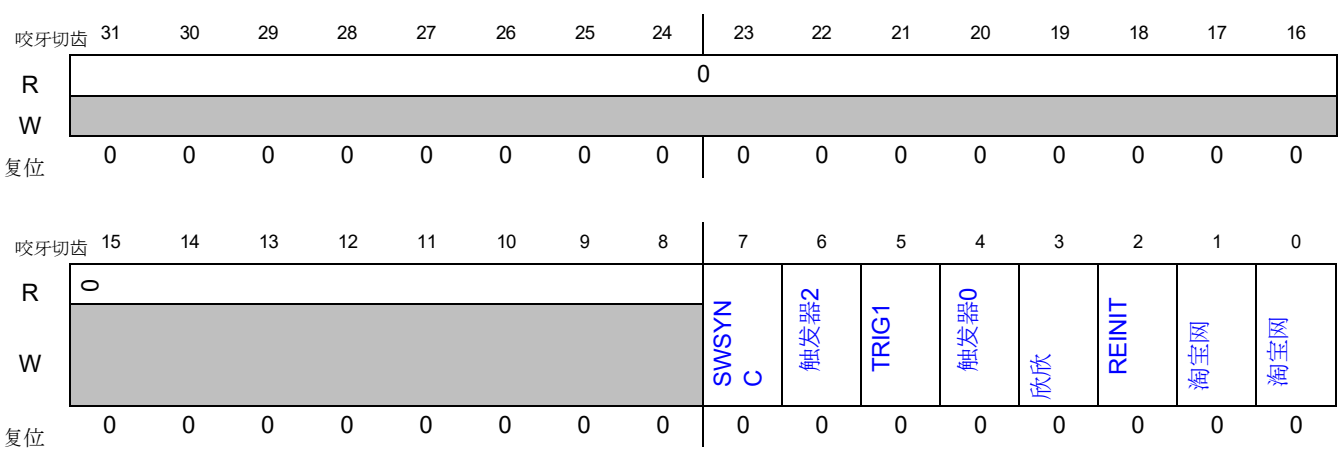
**注意事项**

当SYNCMODE=0时，软件触发器SWSYNC位和硬件触发器TRIG0、TRIG1、TRIG2位一起使用有可能发生冲突。只使用硬件或软件触发器，但不能同时使用，否则很可能发生不可预测的行为。

选择加载点、CNTMAX 和 CNTMIN
位是为了在所有启用的通道上同时提供 MOD、CNTIN 和 CnV 寄存器的更新。将加载点选择与 SYNCMODE = 0 和硬件触发器选择 TRIG0、TRIG1 或 TRIG2
位一起使用，可能会导致不可预测的行为。

同步事件的选择也取决于PWMSYNC（MODE寄存器）和SYNCMODE（SYNCONF
寄存器）位。见PWM同步。

### 42.4.3.10.3 图示




### 42.4.3.10.4 领域

| 场地 | 职能 |
| --- | --- |
| 31-8 | 保留 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| - | |
| 7<br>呼叫中心 | PWM同步软件触发<br>选择软件触发器作为PWM同步触发器。当SWSYNC位被写入1时，软件触发发生。<br>0b - 未选择软件触发。1b - 选择软件触发。 |
| 6<br>触发器2 | PWM同步硬件触发器2<br>启用硬件触发器2以实现PWM同步。当检测到触发器2输入信号的上升沿时，硬件触发器2发生。<br>0b - 触发器被禁用。<br>1b - 触发器被启用。 |
| 5<br>TRIG1 | PWM同步硬件触发器1<br>启用硬件触发器1以实现PWM同步。当检测到触发器1输入信号的上升沿时，硬件触发器1发生。<br>0b - 触发器被禁用。<br>1b - 触发器被启用。 |
| 4<br>触发器0 | PWM同步硬件触发器0<br>启用硬件触发器0以实现PWM同步。当检测到触发器0输入信号的上升沿时，硬件触发器0发生。<br>0b - 触发器被禁用。<br>1b - 触发器被启用。 |
| 3<br>欣欣 | 输出掩码同步化<br>选择OUTMASK寄存器何时用其缓冲区的值更新。<br>0b - OUTMASK 寄存器在 FTM 输入时钟的所有上升沿中以其缓冲器的值进行更新。<br>1b - OUTMASK寄存器仅通过PWM同步更新其缓冲区的值。 |
| 2<br>REINIT | 通过同步化重新初始化FTM计数器<br>决定当检测到同步的选定触发器时，是否重新初始化FTM计数器（FTM计数器同步）。当SYNCMODE为0时，REINIT位配置同步。<br>0b - FTM计数器继续正常计数。<br>1b - 当检测到选定的触发器时，FTM计数器以其初始值更新。 |
| 1<br>淘宝网 | 启用最大装载点<br>选择对PWM同步的最大加载点（同步点）。如果CNTMAX为1，选择的加载点是FTM计数器达到最大值时（MOD寄存器）。<br>0b - 最大装载点被禁用。1b - 最大装载点被启用。 |
| 0<br>ÃÃÃ | 启用最小装载点<br>选择到PWM同步的最小加载点（同步点）。如果CNTMIN为1，选择的加载点是当FTM计数器达到其最小值时（CNTIN寄存器）。<br>0b - 最小装载点被禁用。1b - 最小装载点被启用。 |

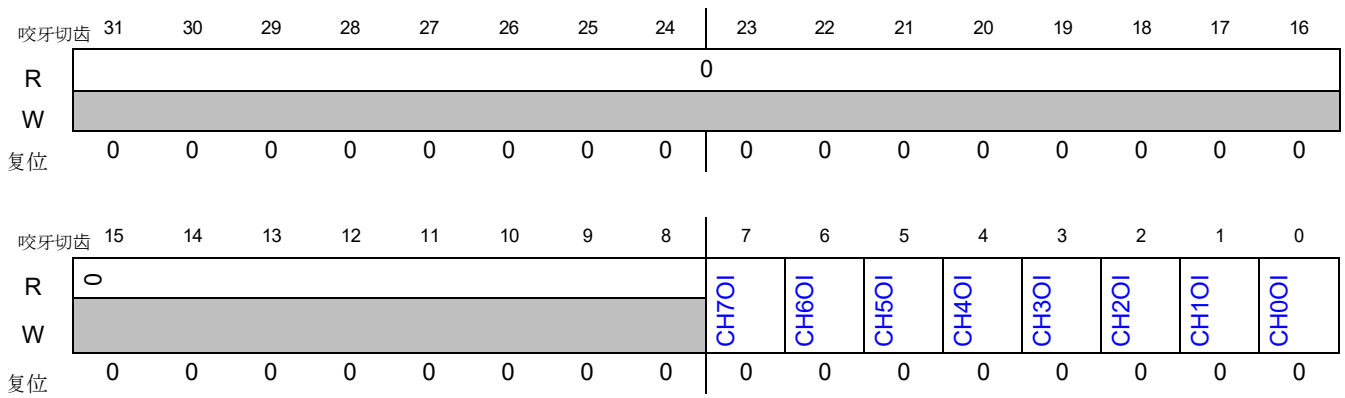
### 42.4.3.11 通道输出的初始状态(OUTINIT)

表继续在下一页...

## 42.4.3.11.1 偏移

| 注册 | 偏移 |
|---|---|
| 淘汰 | 5Ch |

## 42.4.3.11.2 图示




## 42.4.3.11.3 领域

| 场地 | 职能 |
|---|---|
| 31-8<br>- | 保留 |
| 7<br>CH7OI | 通道7输出初始化值<br>选择初始化发生时强制进入通道输出的值。<br>0b - 初始化值为0。 1b - 初始化值为1。 |
| 6<br>CH6OI | 通道6输出初始化值<br>选择初始化发生时强制进入通道输出的值。<br>0b - 初始化值为0。 1b - 初始化值为1。 |
| 5<br>CH5OI | 通道5输出初始化值<br>选择初始化发生时强制进入通道输出的值。<br>0b - 初始化值为0。 1b - 初始化值为1。 |

*表继续在下一页...*

| | |
|---|---|
| 4<br>CH4OI | 通道4输出初始化值<br>选择初始化发生时强制进入通道输出的值。<br>0b - 初始化值为0。 1b - 初始化值为1。 |
| 3<br>CH3OI | 通道3输出初始化值<br>选择初始化发生时强制进入通道输出的值。<br>0b - 初始化值为0。 1b - 初始化值为1。 |

| 场地 | 职能 |
| --- | --- |
| 2<br>CH2OI | 通道2输出初始化值<br>选择初始化发生时强制进入通道输出的值。<br>0b - 初始化值为0。 1b - 初始化值为1。 |
| 1<br>CH1OI | 通道1输出初始化值<br>选择初始化发生时强制进入通道输出的值。<br>0b - 初始化值为0。 1b - 初始化值为1。 |
| 0<br>CH0OI | 通道0输出初始化值<br>选择初始化发生时强制进入通道输出的值。<br>0b - 初始化值为0。 1b - 初始化值为1。 |

## 42.4.3.12 输出屏蔽（OUTMASK）

### 42.4.3.12.1 偏移

| 注册 | 偏移 |
| --- | --- |
| 呼叫中心 | 60h |

### 42.4.3.12.2 职能

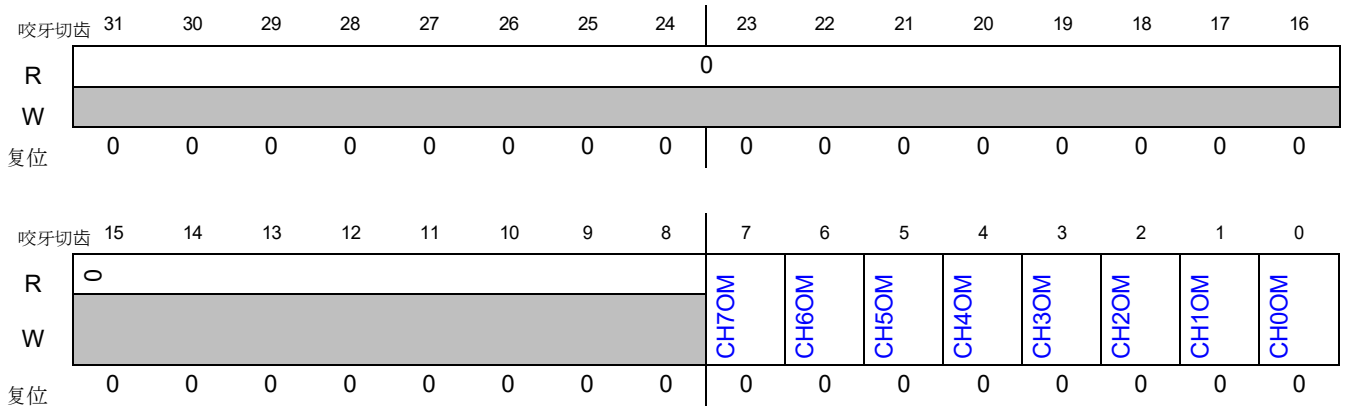
该寄存器为每个 FTM
通道提供一个掩码。一个通道的掩码决定了它的输出是否有反应，也就是说，当匹配发生时，它是否被掩码。这个功能用于BLDC控制，其中PWM信号在特定时间呈现给电动马达，以提供电子换向。

任何对OUTMASK寄存器的写入，都会将其值存储在其写入缓冲区。寄存器根据PWM的同步性，用其写缓冲区的值来更新。

输出屏蔽位不得为触发模式设置。

*表继续在下一页...*

## 42.4.3.12.3 图示




## 42.4.3.12.4 领域

| 场地 | 职能 |
|---|---|
| 31-8<br>- | 保留 |
| 7<br>CH7OM | 通道7输出屏蔽<br>定义通道输出是否被掩蔽或未被掩蔽。<br>0b - 通道输出没有被屏蔽。它继续正常工作。1b - 通道输出被屏蔽。<br>它被强制到不活动状态。 |
| 6<br>CH6OM | 通道6输出屏蔽<br>定义通道输出是否被掩蔽或未被掩蔽。<br>0b - 通道输出没有被屏蔽。它继续正常工作。1b - 通道输出被屏蔽。<br>它被强制到不活动状态。 |
| 5<br>CH5OM | 通道5输出屏蔽<br>定义通道输出是否被掩蔽或未被掩蔽。<br>0b - 通道输出没有被屏蔽。它继续正常工作。1b - 通道输出被屏蔽。<br>它被强制到不活动状态。 |
| 4<br>CH4OM | 通道4输出屏蔽<br>定义通道输出是否被掩蔽或未被掩蔽。<br>0b - 通道输出没有被屏蔽。它继续正常工作。1b - 通道输出被屏蔽。<br>它被强制到其非活动状态。 |

| | |
|---|---|
| 3<br>CH3OM | 通道3输出屏蔽<br>定义通道输出是否被掩蔽或未被掩蔽。<br>0b - 通道输出没有被屏蔽。它继续正常工作。1b - 通道输出被屏蔽。它被强制到不活动状态。 |
| 2<br>CH2OM | 通道2输出屏蔽<br>定义通道输出是否被掩蔽或未被掩蔽。<br>0b - 通道输出没有被屏蔽。它继续正常工作。1b - 通道输出被屏蔽。它被强制到不活动状态。 |
| 1<br>CH1OM | 通道1输出屏蔽<br>定义通道输出是否被掩蔽或未被掩蔽。<br>0b - 通道输出没有被屏蔽。它继续正常运行。 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| | 1b - 通道输出被屏蔽。它被强制到其非活动状态。 |
| 0<br>CH0OM | 通道0输出屏蔽<br>定义通道输出是否被掩蔽或未被掩蔽。<br>0b - 通道输出没有被屏蔽。它继续正常工作。1b - 通道输出被屏蔽。<br>它被强制到不活动状态。 |

## 42.4.3.13 链接通道的功能 (COMBINE)

### 42.4.3.13.1 偏移

| 注册 | 偏移 |
|---|---|
| 融合 | 64h |

### 42.4.3.13.2 职能

该寄存器包含每对通道的配置位。

### 42.4.3.13.3 图示

| 咬牙切齿 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R<br>W | MCOMBINE3 | FAULTEN3 | 卫星导航3 | DTEN3 | DECAP3 | デコンページ 3 | COMP3 | 搭配3 | 牡丹亭2 | 傅尔登2 | 卫星导航2 | DTEN2 | DECAP2 | デコンページ 2 | 竞争者2 | 协同工作2 |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 咬牙切齿 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R<br>W | MCOMBINE1 | FAULTEN1 | 卫星导航1 | DTEN1 | DECAP1 | デコンページ 1 | COMP1 | 梳妆台1 | MCOMBINE0 | FAULTEN0 | 卫星导航系统 (SYNCEN0) | DTEN0 | DECAP0 | ディアパン0 | 竞争者0 | COMBINE0 |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

## 42.4.3.13.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>MCOMBINE3 | 修改后的组合模式 对于n = 6<br>用于选择通道（n）和（n+1）的修正组合模式。参见通道模式。这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。 |
| 30<br>FAULTEN3 | 故障控制启用 对于n = 6<br>启用通道（n）和（n+1）的故障控制。<br>这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。<br>0b - 这对通道的故障控制被禁用。1b - 这一对通道的故障控制被启用。 |
| 29<br>卫星导航3 | 同步启用 对于n = 6<br>启用寄存器C（n）V和C（n+1）V的PWM同步。<br>0b - 这对通道的PWM同步被禁用。1b - 这一对通道的PWM同步被启用。 |
| 28<br>DTEN3 | 死锁时间启用 对于n = 6<br>启用通道(n)和(n+1)中的死机时间插入。<br>这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。<br>0b - 这对通道的死区插入被禁用。1b - 这一对通道的死区插入被启用。 |
| 27<br>DECAP3 | 双边缘捕获模式对n=6进行捕获<br>根据通道(n)输入事件和双边缘捕获位的配置，启用FTM计数器值的捕获。<br>该字段仅在DECAPEN=1时适用。<br>如果选择了双边沿捕获--单发模式，当捕获通道(n+1)事件时，DECAP位被硬件自动清除。<br>0b - 双边沿捕获不活动。1b - 双边沿捕获是激活的。 |
| 26<br>デコンページ3 | 双边缘捕获模式启用 对于n = 6<br>启用通道(n)和(n+1)中的双边缘捕获模式。参见通道模式。这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。 |
| 25<br>COMP3 | n=6时的通道补数(n)<br>在互补模式下，通道（n+1）的输出是通道（n）输出的倒数。这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。<br>0b - 通道（n+1）的输出与通道（n）的输出相同。<br>1b - 通道（n+1）的输出是通道（n）输出的补充。 |

*表继续在下一页...*

| | |
|---|---|
| 24<br>搭配3 | 合并通道 对于n = 6<br>用于选择通道（n）和（n+1）的组合模式。参见通道模式。这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。 |
| 23<br>牡丹亭2 | 修改后的组合模式 对于n = 4<br>用于选择通道（n）和（n+1）的修正组合模式。参见通道模式。这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。 |
| 22 | 故障控制启用 对于n = 4 |

| 场地 | 职能 |
|---|---|
| 傅尔登2 | 启用通道（n）和（n+1）的故障控制。<br>这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。<br>0b - 这对通道的故障控制被禁用。1b - 这一对通道的故障控制被启用。 |
| 21<br>卫星导航2 | 同步启用 对于n = 4<br>启用寄存器C（n）V和C（n+1）V的PWM同步。<br>0b - 这对通道的PWM同步被禁用。1b - 这一对通道的PWM同步被启用。 |
| 20<br>DTEN2 | 死锁时间启用 对于n = 4<br>启用通道(n)和(n+1)中的死机时间插入。<br>这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。<br>0b - 这对通道的死区插入被禁用。1b - 这一对通道的死区插入被启用。 |
| 19<br>DECAP2 | 双边缘捕获模式对n = 4进行捕获<br>根据通道(n)输入事件和双边缘捕获位的配置，启用FTM计数器值的捕获。<br>该字段仅在DECAPEN=1时适用。<br>如果选择了双边沿捕获--单发模式，当捕获通道(n+1)事件时，DECAP位被硬件自动清除。<br>0b - 双边沿捕获不活动。1b - 双边沿捕获是激活的。 |
| 18<br>デコンページ2 | 双边缘捕获模式启用 对于n = 4<br>启用通道(n)和(n+1)中的双边缘捕获模式。参见通道模式。这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。 |
| 17<br>竞争者2 | 通道的补足(n) 对于n = 4<br>在互补模式下，通道（n+1）的输出是通道（n）输出的倒数。这个字段是受写保护的。只有当MODE[WPDIS]=1时才可以写入。<br>0b - 通道（n+1）的输出与通道（n）的输出相同。<br>1b - 通道（n+1）的输出是通道（n）输出的补充。 |
| 16<br>搭配2 | 合并通道 对于n = 4<br>用于选择通道（n）和（n+1）的组合模式。参见通道模式。这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。 |
| 15<br>MCOMBINE1 | 修改后的组合模式 对于n = 2<br>用于选择通道（n）和（n+1）的修正组合模式。参见通道模式。这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。 |

*表继续在下一页...*

| | |
|---|---|
| 14<br>FAULTEN1 | 故障控制启用 对于n = 2<br>启用通道（n）和（n+1）的故障控制。<br>这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。<br>0b - 这对通道的故障控制被禁用。1b - 这一对通道的故障控制被启用。 |
| 13<br>卫星导航1 | 同步启用 对于n = 2<br>启用寄存器C（n）V和C（n+1）V的PWM同步。 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | 0b - 这对通道的PWM同步被禁用。1b - 这一对通道的PWM同步被启用。 |
| 12<br>DTEN1 | 死锁时间启用 对于n = 2<br>启用通道(n)和(n+1)中的死机时间插入。<br>这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。<br>0b - 这对通道的死区插入被禁用。1b - 这一对通道的死区插入被启用。 |
| 11<br>DECAP1 | 双边缘捕获模式对n = 2进行捕获<br>根据通道(n)输入事件和双边缘捕获位的配置，启用FTM计数器值的捕获。<br>该字段仅在DECAPEN=1时适用。<br>如果选择了双边缘捕获-单次模式，当捕获通道(n+1)事件时，DECAP位被硬件自动清除。<br>0b - 双边沿捕获不活动。1b - 双边沿捕获是激活的。 |
| 10<br>デコンページ1 | 双边缘捕获模式启用 对于n = 2<br>启用通道(n)和(n+1)中的双边缘捕获模式。参见通道模式。这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。 |
| 9<br>COMP1 | 通道的补足(n) 对于n = 2<br>在互补模式下，通道（n+1）的输出是通道（n）输出的倒数。这个字段是受写保护的。只<br>有当MODE[WPDIS]=1时才能写入。<br>0b - 通道（n+1）的输出与通道（n）的输出相同。<br>1b - 通道（n+1）的输出是通道（n）输出的补充。 |
| 8<br>梳妆台1 | 合并通道 对于n = 2<br>用于选择通道（n）和（n+1）的组合模式。参见通道模式。这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。 |
| 7<br>MCOMBINE0 | 修改后的组合模式 对于n = 0<br>用于选择通道（n）和（n+1）的修正组合模式。参见通道模式。这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。 |
| 6<br>FAULTEN0 | 故障控制启用 对于n = 0<br>启用通道（n）和（n+1）的故障控制。<br>这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。<br>0b - 这对通道的故障控制被禁用。1b - 这一对通道的故障控制被启用。 |

*表继续在下一页...*

| | |
|---|---|
| 5<br>卫星导航系统<br>(SYNCEN0) | 同步启用 对于n = 0<br>启用寄存器C（n）V和C（n+1）V的PWM同步。<br>0b - 这对通道的PWM同步被禁用。 1b - 这一对通道的PWM同步被启用。 |
| 4<br>DTEN0 | 死锁时间启用 对于n = 0<br>启用通道(n)和(n+1)中的死机时间插入。<br>这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。<br>0b - 这对通道中的死机插入被禁用。 |

| 场地 | 职能 |
| --- | --- |
| | 1b - 启用这对通道的死机时间插入。 |
| 3<br>DECAP0 | 双边缘捕获模式对n=0进行捕获<br>根据通道(n)输入事件和双边缘捕获位的配置，启用FTM计数器值的捕获。<br>该字段仅在DECAPEN=1时适用。<br>如果选择了双边沿捕获--单发模式，当捕获通道(n+1)事件时，DECAP位被硬件自动清除。<br>0b - 双边沿捕获不活动。1b - 双边沿捕获<br>是激活的。 |
| 2<br>ディアパン0 | 双边缘捕获模式启用 对于n = 0<br>启用通道(n)和(n+1)中的双边缘捕获模式。参见通道模式。这个字段是受写保护的。只有当<br>MODE[WPDIS]=1时才能写入。 |
| 1<br>竞争者0 | 通道的补足(n) 对于n = 0<br>在互补模式下，通道（n+1）的输出是通道（n）输出的倒数。这个字段是受写保护的。只<br>有当MODE[WPDIS]=1时才能写入。<br>0b - 通道（n+1）的输出与通道（n）的输出相同。<br>1b - 通道（n+1）的输出是通道（n）输出的补充。 |
| 0<br>COMBINE0 | 合并通道 对于n = 0<br>用于选择通道（n）和（n+1）的组合模式。参见通道模式。这个字段是受写保护的。只有当<br>MODE[WPDIS]=1时才能写入。 |

## 42.4.3.14 死亡时间配置（DEADTIME）

### 42.4.3.14.1 偏移

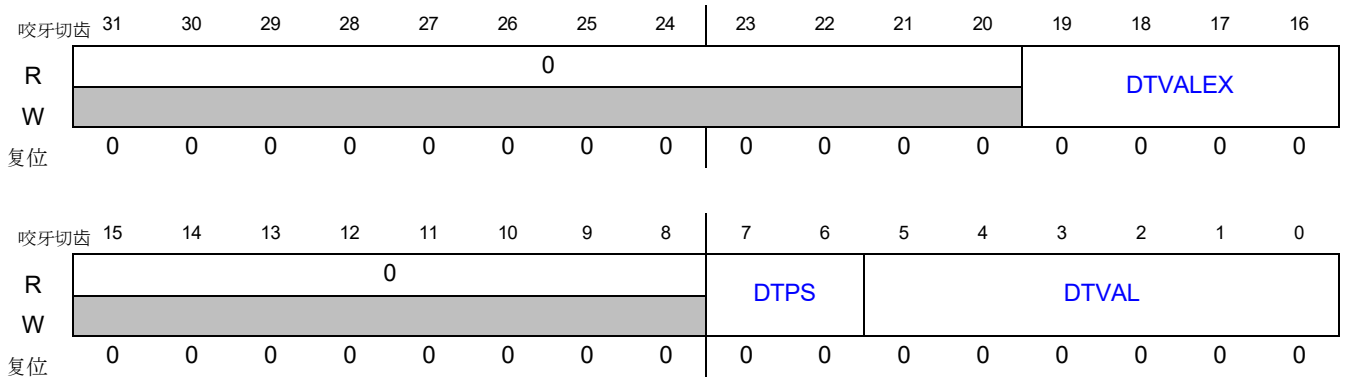
| 注册 | 偏移 |
| --- | --- |
| 死亡时间 | 68h |

### 42.4.3.14.2 职能

该寄存器选择死区预分频器和值。

*表继续在下一页...*

### 42.4.3.14.3 图示




### 42.4.3.14.4 领域

| 场地 | 职能 |
|---|---|
| 31-20<br>- | 保留 |
| 19-16<br>DTVALEX | 扩展死区值<br>这个字段是DTVAL字段的一个比特扩展。它定义了死锁时间值的4个最重要的位。使用{DTVALEX, DTVAL}的连接，最大死区值被扩展到1023。<br>死亡时间插入值=（DTPS × {DTVALEX, DTVAL}）。<br>这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。<br>**注意：**如果需要与以前的软件版本完全兼容，请将DTVALEX位写入0。 |
| 15-8<br>- | 保留 |
| 7-6<br>DTPS | 死锁时间预分频器值<br>选择FTM输入时钟的分割系数。这个预标定的时钟被死区计数器所使用。这个字段是受写保护的。只有当MODE[WPDIS]=1时才可以写入。<br>0xb - 将FTM输入时钟除以1。 10b - 将FTM输入时钟除以4。 11b - 将FTM输入时钟除以16。 |
| 5-0<br>DTVAL | 死亡时间值<br>选择死机时间值。<br>这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。 |

### 42.4.3.15 FTM 外部触发器 (EXTTRIG)

表继续在下一页...

### 42.4.3.15.1 偏移

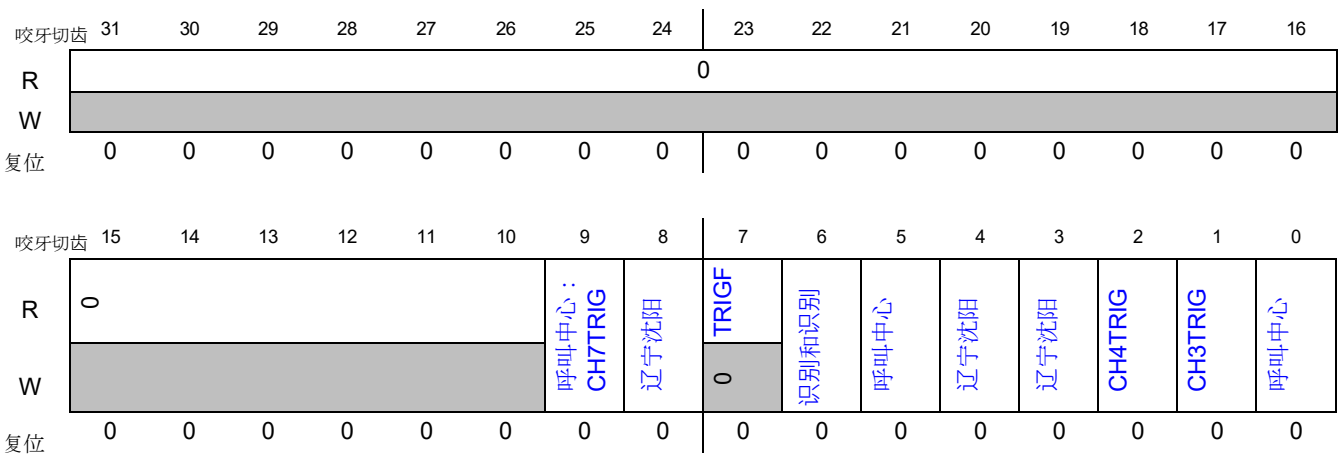
| 注册 | 偏移 |
|---|---|
| 额外的 | 6Ch |

### 42.4.3.15.2 职能

这个登记簿。

- 表明外部触发器何时产生
- 当FTM计数器等于其初始值时，允许生成一个触发器
- 选择哪些通道被用于生成外部触发器

### 42.4.3.15.3 图示




### 42.4.3.15.4 领域

| 场地 | 职能 |
|---|---|
| 31-10<br>- | 保留 |
| 9<br>辽宁沈阳 | 通道7外部触发器启用<br>当FTM计数器=C7V时，启用外部触发器的生成。<br>0b - 该外部触发器的生成被禁用。1b - 该外部触发器的产生被激活。 |

| | |
|---|---|
| 8<br>辽宁沈阳 | 通道6外部触发器启用<br>当FTM计数器=C6V时，启用外部触发器的生成。<br>0b - 该外部触发器的生成被禁用。1b - 该外部触发器的产生被激活。 |
| 7 | 通道触发标志 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| TRIGF | 当一个通道触发器产生时，由硬件设置。当TRIGF被设置时，通过读取EXTTRIG，然后向TRIGF写0来清除TRIGF。向TRIGF写1没有效果。<br>如果在清零序列完成之前产生了另一个通道的触发器，那么该序列就会被重置，所以在清零序列完成之后，TRIGF仍然被设置为先前的TRIGF。<br>0b - 没有产生通道触发。1b - 产生了一个通道触发器。 |
| 6<br>识别和识别 | 初始化触发器启用<br>当FTM计数器等于CNTIN寄存器时，启用触发器的生成。<br>0b - 初始化触发器的生成被禁用。1b - 启用初始化触发的生成。 |
| 5<br>呼叫中心 | 通道1外部触发器启用<br>当FTM计数器=C1V时，启用外部触发器的生成。<br>0b - 该外部触发器的生成被禁用。1b - 该外部触发器的生成被激活。 |
| 4<br>辽宁沈阳 | 通道0外部触发器启用<br>当FTM计数器=C0V时，启用外部触发器的生成。<br>0b - 该外部触发器的生成被禁用。1b - 该外部触发器的生成被激活。 |
| 3<br>辽宁沈阳 | 通道5外部触发器启用<br>当FTM计数器=C5V时，启用外部触发器的生成。<br>0b - 该外部触发器的生成被禁用。1b - 该外部触发器的生成被激活。 |
| 2<br>CH4TRIG | 通道4外部触发器启用<br>当FTM计数器=C4V时，启用外部触发器的生成。<br>0b - 该外部触发器的生成被禁用。1b - 该外部触发器的生成被激活。 |
| 1<br>CH3TRIG | 通道3外部触发器启用<br>当FTM计数器=C3V时，启用外部触发器的生成。<br>0b - 该外部触发器的生成被禁用。1b - 该外部触发器的生成被激活。 |
| 0<br>呼叫中心 | 通道2外部触发器启用<br>当FTM计数器=C2V时，启用外部触发器的生成。<br>0b - 该外部触发器的生成被禁用。1b - 该外部触发器的产生被启用。 |

## 42.4.3.16 通道的极性（POL）。

### 42.4.3.16.1 偏移

| 注册 | 偏移 |
| --- | --- |
| 铂金 | 70h |

*表继续在下一页...*

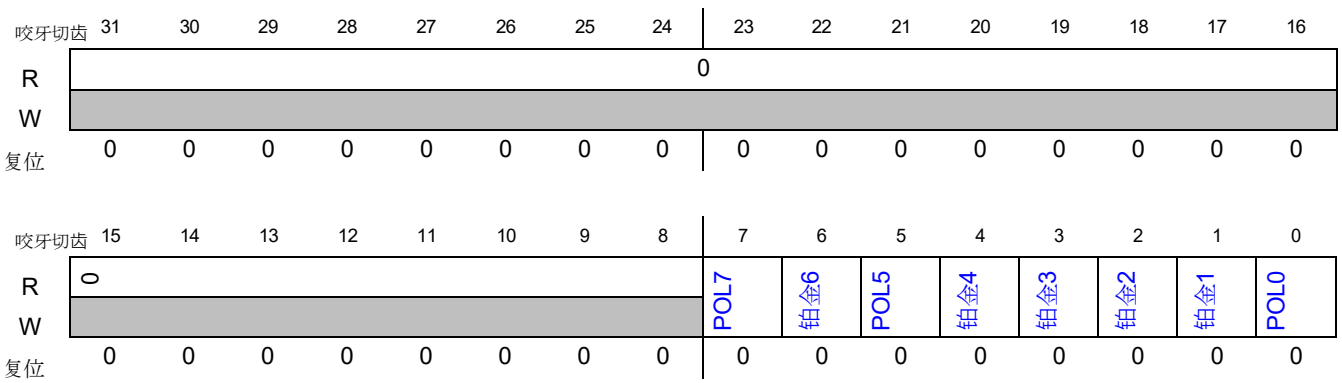
### 42.4.3.16.2 职能

该寄存器定义了FTM通道的输出极性。

**注意事项**

通道安全值是其POL位的值。当故障控制被启用且检测到故障情况时，通道安全值在通道输出上被驱动。

### 42.4.3.16.3 图示




### 42.4.3.16.4 领域

| 场地 | 职能 |
|---|---|
| 31-8<br>- | 保留 |
| 7<br>POL7 | 通道7的极性<br>定义了通道输出的极性。<br>这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。<br>0b - 通道的极性是高电平有效。1b - 通道的极性为低电平。 |
| 6<br>铂金6 | 通道6的极性<br>定义了通道输出的极性。<br>这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。<br>0b - 通道的极性是高电平有效。1b - 通道的极性为低电平。 |

| | |
|---|---|
| 5<br>POL5 | 通道5的极性<br>定义了通道输出的极性。<br>这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。<br>0b - 通道的极性是高电平有效。1b - 通道的极性为低电平。 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| 4<br>铂金4 | 频道4极性<br>定义了通道输出的极性。<br>这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。<br>0b - 通道的极性是高电平有效。1b - 通道的极性为低电平。 |
| 3<br>铂金3 | 通道3的极性<br>定义了通道输出的极性。<br>这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。<br>0b - 通道的极性是高电平有效。1b - 通道的极性为低电平。 |
| 2<br>铂金2 | 通道2的极性<br>定义了通道输出的极性。<br>这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。<br>0b - 通道的极性是高电平有效。1b - 通道的极性为低电平。 |
| 1<br>铂金1 | 通道1的极性<br>定义了通道输出的极性。<br>这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。<br>0b - 通道的极性是高电平有效。1b - 通道的极性为低电平。 |
| 0<br>POL0 | 通道0的极性<br>定义了通道输出的极性。<br>这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。<br>0b - 通道的极性是高电平有效。1b - 通道的极性为低电平。 |

## 42.4.3.17 故障模式状态（FMS）

### 42.4.3.17.1 偏移

| 注册 | 偏移 |
| --- | --- |
| FMS | 74h |

#### 42.4.3.17.2 职能

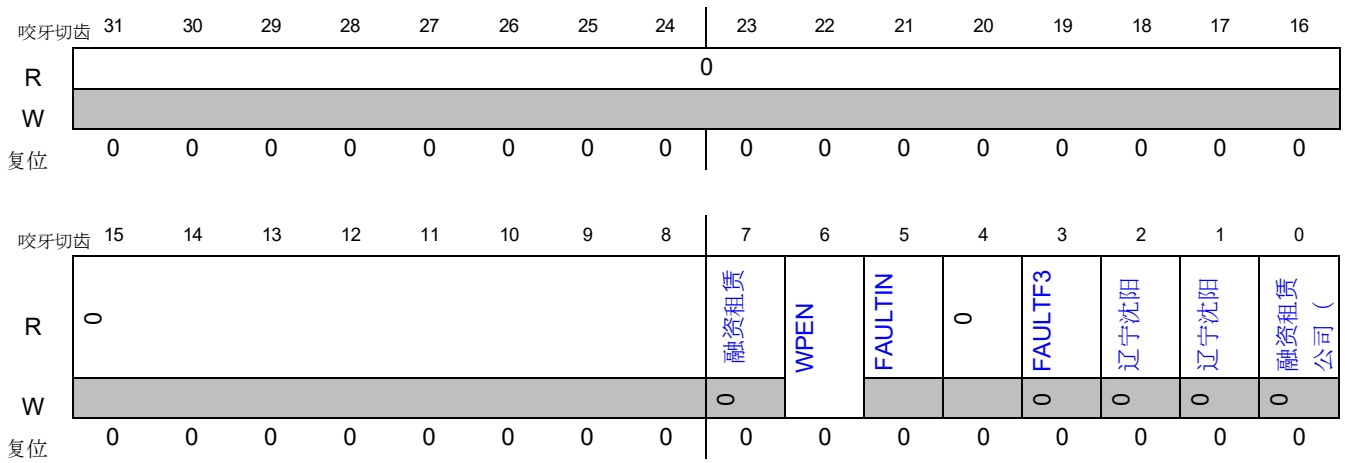
该寄存器包含。

- 写保护启用位

表继续在下一页...

- 故障检测标志
- 启用的故障输入的逻辑OR

## 42.4.3.17.3 图示




## 42.4.3.17.4 领域

| 场地 | 职能 |
|---|---|
| 31-8<br>- | 保留 |
| 7<br>融资租赁 | 故障检测标志<br>代表每个启用的故障输入的FAULTF位的逻辑OR。清除FAULTF的方法是：在FAULTF被设置时读取FMS寄存器，然后在启用的故障输入端没有现有的故障条件时向FAULTF写0。向 FAULTF 写 1 没有效果。<br>如果在清零序列完成之前，在一个启用的故障输入中检测到另一个故障条件，则该序列被重置，因此在清零序列完成后，FAULTF仍被设置为先前的故障条件。当每个启用的故障输入的FAULTF位被清除时，FAULTF也被清除。<br>0b - 没有检测到故障条件。 1b - 检测到一个故障条件。 |
| 6<br>WPEN | 写保护启用<br>WPEN位是对WPDIS位的否定。当WPEN被写入1时，WPEN被设置。当WPEN位被读为1，然后将1写入WPDIS时，WPEN被清除。向WPEN写0没有效果。<br>0b - 禁用写保护。受写保护的位可以被写入。 1b - 启用写保护。受写保护的位不能被写入。 |

| | |
|---|---|
| 5<br>FAULTIN | 故障输入<br>代表在启用故障控制时，启用的故障输入在其滤波器（如果其滤波器被启用）之后的逻辑OR。<br>0b - 启用故障输入的逻辑OR为0。 1b - 启用故障输入的逻辑OR为1。 |
| 4<br>- | 保留 |

*表继续在下一页...*

<table>
<tr><th>场地</th><th>职能</th></tr>
<tr><td>3<br>FAULTF3</td><td>故障检测标志3<br>当故障控制被启用时，由硬件设置，相应的故障输入被启用，并且在故障输入处检测到一个故障条件。<br>清除FAULTF3的方法是：在FAULTF3被设置时读取FMS寄存器，然后在相应的故障输入端没有现存的故障条件时，向FAULTF3写0。向 FAULTF3 写 1 没有效果。当FAULTF位被清除时，FAULTF3位也被清除。<br>如果在清零序列完成之前，在相应的故障输入端检测到另一个故障条件，那么该序列将被重置，因此在清零序列完成之后，FAULTF3仍然被设置为先前的故障条件。<br>注意：这个字段并不是在每个实例中都支持。<br><table><tr><th>支持的</th><th>不支持</th></tr><tr><td>FTM0_FMS<br>FTM1_FMS<br>FTM2_FMS<br>FTM3_FMS</td><td>FTM4_FMS<br>FTM5_FMS<br>FTM6_FMS<br>FTM7_FMS</td></tr></table><br>0b - 在故障输入端没有检测到故障条件。1b - 在故障输入端检测到一个故障条件。</td></tr>
<tr><td>2<br>辽宁沈阳</td><td>故障检测标志2<br>当故障控制被启用时，由硬件设置，相应的故障输入被启用，并且在故障输入处检测到一个故障条件。<br>清除FAULTF2的方法是：当FAULTF2被设置时，读取FMS寄存器，然后在相应的故障输入端没有现有的故障条件时，向FAULTF2写0。向 FAULTF2 写 1 没有效果。当FAULTF位被清除时，FAULTF2位也被清除。<br>如果在清零序列完成之前，在相应的故障输入端检测到另一个故障条件，则该序列被重置，因此在清零序列完成后，FAULTF2仍被设置为先前的故障条件。<br>注意：这个字段并不是在每个实例中都支持。<br><table><tr><th>支持的</th><th>不支持</th></tr><tr><td>FTM0_FMS<br>FTM1_FMS<br>FTM2_FMS<br>FTM3_FMS</td><td>FTM4_FMS<br>FTM5_FMS<br>FTM6_FMS<br>FTM7_FMS</td></tr></table><br>0b - 在故障输入端没有检测到故障条件。1b - 在故障输入端检测到一个故障条件。</td></tr>
<tr><td>1<br>辽宁沈阳</td><td>故障检测标志1<br>当故障控制被启用时，由硬件设置，相应的故障输入被启用，并且在故障输入处检测到一个故障条件。<br>清除FAULTF1的方法是：在FAULTF1被设置时读取FMS寄存器，然后在相应的故障输入端不存在故障条件的情况下向FAULTF1写0。向 FAULTF1 写 1 没有效果。当FAULTF位被清除时，FAULTF1位也被清除。<br>如果在清零序列完成之前，在相应的故障输入端检测到另一个故障条件，则该序列被重置，因此在清零序列完成后，先前的故障条件的FAULTF1仍然被设置。</td></tr>
</table>

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | 0b - 在故障输入端没有检测到故障条件。1b - 在故障<br>输入端检测到一个故障条件。 |
| 0<br>融资租赁公司（<br>FAULTF0 | 故障检测标志0<br>当故障控制被启用时，由硬件设置，相应的故障输入被启用，并且在故障输入处检测到一个故障条件。<br>清除FAULTF0的方法是：在FAULTF0被设置时，读取FMS寄存器，然后在相应的故障输入端没有现有的故障条件时，向FAULTF0写0。向 FAULTF0 写 1 没有效果。当FAULTF位被清除时，FAULTF0位也被清除。<br>如果在清零序列完成之前，在相应的故障输入端检测到另一个故障条件，则该序列被重置，因此在清零序列完成之后，FAULTF0仍然被设置为先前的故障条件。<br>0b - 在故障输入端没有检测到故障条件。1b - 在故障<br>输入端检测到一个故障条件。 |

## 42.4.3.18 输入捕获过滤器控制（FILTER）

### 42.4.3.18.1 偏移

| 注册 | 偏移 |
| --- | --- |
| 筛选器 | 78h |

### 42.4.3.18.2 职能

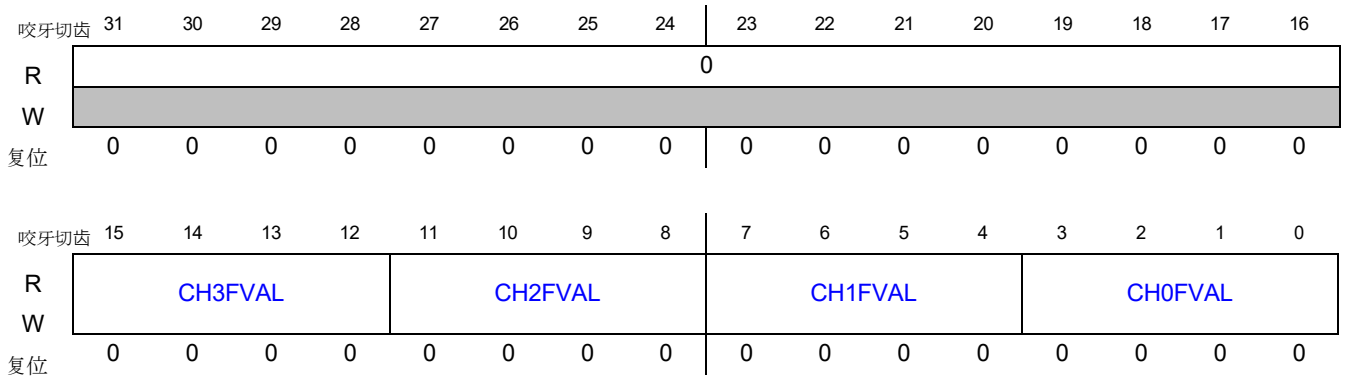
这个寄存器为通道的输入选择滤波值。通道4、5、6和7没

有输入滤波器。

**注意事项**

对 FILTER 寄存器的写入是立即生效的，必须在通道
0、1、2、3
不在输入模式时才可进行。如果不这样做，可能会导致有效信号的丢失。

### 42.4.3.18.3 图示




### 42.4.3.18.4 领域

| 场地 | 职能 |
|---|---|
| 31-16<br>- | 保留 |
| 15-12<br>CH3FVAL | 通道3输入滤波器<br>选择通道输入的滤波值。当该值为零时，滤波器被禁用。 |
| 11-8<br>CH2FVAL | 通道2输入滤波器<br>选择通道输入的滤波值。当该值为零时，滤波器被禁用。 |
| 7-4<br>CH1FVAL | 通道1输入滤波器<br>选择通道输入的滤波值。当该值为零时，滤波器被禁用。 |
| 3-0<br>CH0FVAL | 通道0输入滤波器<br>选择通道输入的滤波值。当该值为零时，滤波器被禁用。 |

## 42.4.3.19 故障控制（FLTCTRL）

### 42.4.3.19.1 偏移

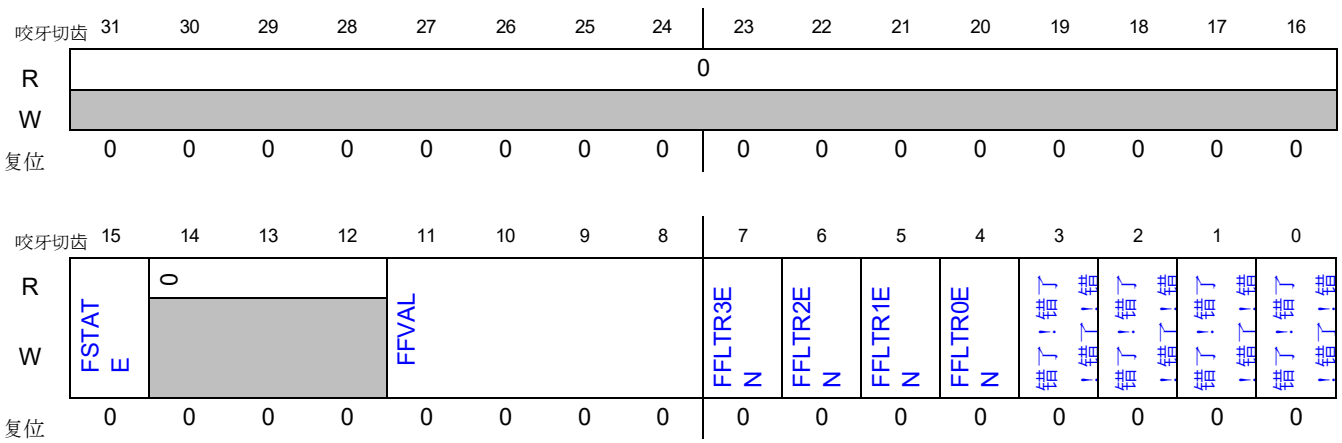
| 注册 | 偏移 |
|---|---|
| 呼叫中心 | 7Ch |

### 42.4.3.19.2 职能

该寄存器包含。

- 故障事件发生时通道输出的状态
- 每个故障输入的启用
- 每一个故障输入的过滤器启用
- 启用的故障输入的滤波值，并带有滤波器

### 42.4.3.19.3 图示




### 42.4.3.19.4 领域

| 场地 | 职能 |
|---|---|
| 31-16<br>- | 保留 |
| 15<br>基金会 | 故障输出状态<br>该配置允许在故障事件发生时将FTM输出变成三态。这个字段是受写保护的。只有当<br>MODE[WPDIS]=1时才可以写入。<br>0b - 当故障事件正在进行时，FTM 输出将被置于安全值中（由 POL 位定义）。1b - 当故障事件正在发生时，FTM输出将被三振。 |
| 14-12<br>- | 保留 |

| | |
|---|---|
| 11-8<br>FFVAL | 故障输入过滤器<br>选择故障输入的滤波值。<br>当数值为零时，故障过滤器被禁用。<br>**注意：**对这个字段的写入是立即生效的，必须在故障控制或所有故障输入被禁用的情况下才能进行。如果不这样做，可能会导致故障检测丢失。 |
| 7<br>FFLTR3EN | 故障输入 3 滤波器启用<br>启用故障输入的过滤器。 |

*表继续在下一页...*

<table>
<tr><th>场地</th><th>职能</th></tr>
<tr><td></td><td>这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。<br>注意：这个字段并不是在每个实例中都支持。<br><table><tr><th>支持的</th><th>不支持</th></tr><tr><td>ftm0_fltctrl<br>ftm1_fltctrl<br>ftm2_fltctrl<br>ftm3_fltctrl</td><td>ftm4_fltctrl<br>ftm5_fltctrl<br>ftm6_fltctrl<br>ftm7_fltctrl</td></tr></table><br>0b - 故障输入滤波器被禁用。 1b<br>- 故障输入滤波器被启用。</td></tr>
<tr><td>6<br>FFLTR2EN</td><td>故障输入2 滤波器启用<br>启用故障输入的过滤器。<br>这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。<br>注意：这个字段并不是在每个实例中都支持。<br><table><tr><th>支持的</th><th>不支持</th></tr><tr><td>ftm0_fltctrl<br>ftm1_fltctrl<br>ftm2_fltctrl<br>ftm3_fltctrl</td><td>ftm4_fltctrl<br>ftm5_fltctrl<br>ftm6_fltctrl<br>ftm7_fltctrl</td></tr></table><br>0b - 故障输入滤波器被禁用。 1b<br>- 故障输入滤波器被启用。</td></tr>
<tr><td>5<br>FFLTR1EN</td><td>故障输入1 滤波器启用<br>启用故障输入的过滤器。<br>这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。<br>0b - 故障输入滤波器被禁用。 1b<br>- 故障输入滤波器被启用。</td></tr>
<tr><td>4<br>FFLTR0EN</td><td>故障输入 0 滤波器启用<br>启用故障输入的过滤器。<br>这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。<br>0b - 故障输入滤波器被禁用。 1b<br>- 故障输入滤波器被启用。</td></tr>
<tr><td>3<br>错了！错了！错了！错了！错了！错了！错了！错了！错了</td><td>故障输入3启用 启用故障<br>输入。<br>这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。<br>注意：这个字段并不是在每个实例中都支持。<br><table><tr><th>支持的</th><th>不支持</th></tr></table></td></tr>
</table>

| | |
|---|---|
| ftm0_fltctrl<br>ftm1_fltctrl<br>ftm2_fltctrl<br>ftm3_fltctrl | ftm4_fltctrl<br>ftm5_fltctrl<br>ftm6_fltctrl<br>ftm7_fltctrl |

*表继续在下一页...*

<table>
<tr><th>场地</th><th>职能</th></tr>
<tr><td></td><td>0b - 故障输入被禁用。1b -<br>故障输入被启用。</td></tr>
<tr><td>2<br>错了！错了！错了！错了！错了！错了！错了！错了！错了</td><td>故障输入2启用 启用故障<br>输入。<br>这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。<br>注意：这个字段并不是在每个实例中都支持。<table><tr><th>支持的</th><th>不支持</th></tr><tr><td>ftm0_fltctrl<br>ftm1_fltctrl<br>ftm2_fltctrl<br>ftm3_fltctrl</td><td>ftm4_fltctrl<br>ftm5_fltctrl<br>ftm6_fltctrl<br>ftm7_fltctrl</td></tr></table>0b - 故障输入被禁用。1b -<br>故障输入被启用。</td></tr>
<tr><td>1<br>错了！错了！错了！错了！错了！错了！错了！错了！错了</td><td>故障输入1启用 启用故障<br>输入。<br>这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。<br>0b - 故障输入被禁用。1b -<br>故障输入被启用。</td></tr>
<tr><td>0<br>错了！错了！错了！错了！错了！错了！错了！错了！错了</td><td>故障输入0 启用 启用故<br>障输入。<br>这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。<br>0b - 故障输入被禁用。1b -<br>故障输入被启用。</td></tr>
</table>

## 42.4.3.20 正交解码器控制和状态（QDCTRL）

### 42.4.3.20.1 偏移

| 注册 | 偏移 |
|---|---|
| 呼叫中心 | 80h |

### 42.4.3.20.2 职能

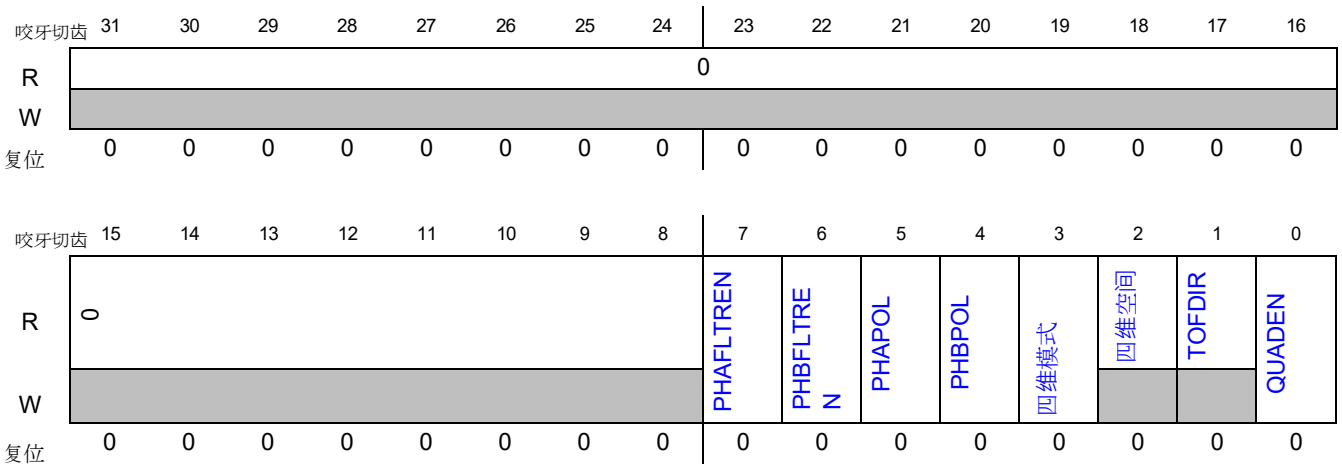
该寄存器具有正交解码器模式的控制和状态位。

**注意事项**

这个寄存器不是每个模块实例都使用的。

- 适用于FTM1, FTM2, FTM4。
- 不适用于FTM0, FTM3, FTM5, FTM6, FTM7。

## 42.4.3.20.3 图示




## 42.4.3.20.4 领域

| 场地 | 职能 |
|---|---|
| 31-8<br>- | 保留 |
| 7<br>PHAFLTREN | A相输入滤波器启用<br>启用正交解码器A相输入的滤波器。A相输入的滤波器值由FILTER的CH0FVAL字段定义。当CH0FVAL为零时，A相滤波器也被禁用。<br>**注意**：这个字段并不是在每个实例中都支持。<br>支持的：ftm1_qdctrl ftm2_qdctrl<br>不支持：呼叫中心：FTM4_QDCTRL<br>0b - A相输入滤波器被禁用。1b - A相输入滤波器被启用。 |
| 6<br>PHBFLTREN | B相输入滤波器启用<br>启用正交解码器B相输入的滤波器。B相输入的滤波器值由FILTER的CH1FVAL字段定义。当CH1FVAL为零时，B相滤波器也被禁用。<br>**注意**：这个字段并不是在每个实例中都支持。<br>支持的<br>不支持 |

| | |
|---|---|
| ftm1_qdctrl ftm2_qdctrl | 呼叫中心：FTM4_QDCTRL |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | 0b - B相输入滤波器被禁用。1b - B相输入滤波器被启用。 |
| 5<br>PHAPOL | A相输入极性<br>选择正交解码器A相输入的极性。<br>0b - 正常极性。在识别该信号的上升沿和下降沿之前，A相输入信号不反转。<br>1b - 倒置极性。在识别该信号的上升沿和下降沿之前，A相输入信号被反转。 |
| 4<br>PHBPOL | B相输入极性<br>选择正交解码器B相输入的极性。<br>0b - 正常极性。在识别该信号的上升沿和下降沿之前，B相输入信号不反转。<br>1b - 倒置极性。在识别该信号的上升沿和下降沿之前，B相输入信号被反转。 |
| 3<br>四维模式 | 正交解码器模式<br>选择正交解码器模式中使用的编码模式。<br>0b - A相和B相编码模式。1b - 计数和方向编码模式。 |
| 2<br>四维空间 | 正交解码器模式下的FTM计数器方向<br>表示计数方向。<br>0b - 计数方向为递减（FTM计数器递减）。1b - 计数方向是增加（FTM计数器增加）。 |
| 1<br>TOFDIR | 正交解码器模式下的定时器溢出方向<br>表明TOF位是在计数的顶部还是底部被设置。<br>0b - TOF位在计数的底部被设置。有一个FTM计数器递减，FTM计数器从它的最小值（CNTIN寄存器）变为它的最大值（MOD寄存器）。<br>1b - TOF位在计数的顶部被设置。有一个FTM计数器增量，FTM计数器从它的最大值（MOD寄存器）变为它的最小值（CNTIN寄存器）。 |
| 0<br>QUADEN | 正交解码器模式启用<br>启用正交解码器模式。在这种模式下，A相和B相的输入信号控制FTM计数器的方向。正交解码器模式优先于其他模式。<br>这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。<br>0b - 正交解码器模式被禁用。1b - 正交解码器模式被启用。 |

## 42.4.3.21 配置 (CONF)

### 42.4.3.21.1 偏移

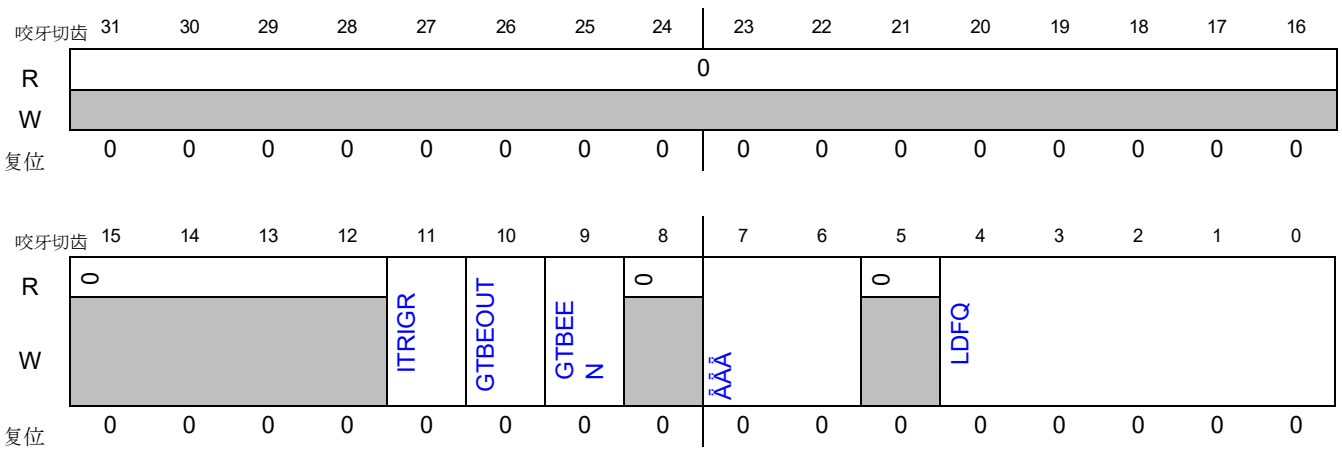
| 注册 | 偏移 |
| --- | --- |
| 委员长 | 84h |

### 42.4.3.21.2 职能

这个寄存器可以选择重新加载机会的频率，调试模式下的FTM行为，使用外部全局时基，以及全局时基信号的产生。

这个寄存器还控制在达到重载点时是否要产生初始化触发。

### 42.4.3.21.3 图示




### 42.4.3.21.4 领域

| 场地 | 职能 |
|---|---|
| 31-12<br>- | 保留 |
| 11<br>ITRIGR | 重新加载点的初始化触发<br>该位控制当考虑到FTM_CONF[LDFQ]设置，达到PWMLOAD寄存器配置的重载点时，是否产生初始化触发。<br>0b - 初始化触发器在计数器包络事件中产生。<br>1b - 达到重载点时产生初始化触发。 |
| 10<br>GTBEOUT | 全局时基输出<br>启用全局时基信号生成到其他FTMs。<br>0b - 全局时基信号生成被禁用。1b - 启用全局时基信号的生成。 |
| 9<br>ÃÃÃ | 全局时基启用<br>配置FTM使用一个由另一个FTM产生的外部全局时基信号。<br>0b - 禁止使用外部全局时基。1b - 启用外部全局时基的使用。 |

| | |
|---|---|
| 8<br>-<br> | 保留 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| 7-6<br>ÃÃÃ | 调试模式<br>选择调试模式下的FTM行为。见调试模式。 |
| 5<br>- | 保留 |
| 4-0<br>LDFQ | 重新加载机会的频率<br>LDFQ[4:0]位定义了启用的重载机会应发生的次数，直到启用的重载机会成为重载点。见重载点<br>LDFQ = 0：所有重装机会都是重装点。<br>LDFQ=1：每2次重装机会就有一个重装点。LDFQ = 2: 每3次重装机会就有一个重装点。LDFQ=3：每4次重载机会就有一个重载点。这种模式一直持续到最多32个。 |

## 42.4.3.22 FTM 故障输入极性 (FLTPOL)

### 42.4.3.22.1 偏移

| 注册 | 偏移 |
|---|---|
| FLTPOL | 88h |

### 42.4.3.22.2 职能

该寄存器定义了故障输入的极性。

### 42.4.3.22.3 图示

| 咬牙切齿 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 咬牙切齿 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | FLT3 | FLT2 | FLT1 | FLT0 |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

## 42.4.3.22.4 领域

| 场地 | 职能 |
|---|---|
| 31-4<br>- | 保留 |
| 3<br>FLT3POL | 故障输入3极性<br>定义了故障输入的极性。<br>这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。<br>**注意：**这个字段并不是在每个实例中都支持。<br>**支持的**: ftm0_fltpol, ftm1_fltpol, ftm2_fltpol, ftm3_fltpol<br>**不支持**: ftm4_fltpol, ftm5_fltpol, ftm6_fltpol, ftm7_fltpol<br>0b - 故障输入极性为高电平有效。在故障输入端出现1表示有故障。1b - 故障输入极性为低电平。故障输入端为0表示有故障。 |
| 2<br>FLT2POL | 故障输入2极性<br>定义了故障输入的极性。<br>这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。<br>**注意：**这个字段并不是在每个实例中都支持。<br>**支持的**: ftm0_fltpol, ftm1_fltpol, ftm2_fltpol, ftm3_fltpol<br>**不支持**: ftm4_fltpol, ftm5_fltpol, ftm6_fltpol, ftm7_fltpol<br>0b - 故障输入极性为高电平有效。在故障输入端出现1表示有故障。1b - 故障输入极性为低电平。故障输入端为0表示有故障。 |
| 1<br>FLT1POL | 故障输入1极性<br>定义了故障输入的极性。<br>这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。<br>0b - 故障输入极性为高电平有效。在故障输入端出现1表示有故障。1b - 故障输入极性为低电平。故障输入端为0表示有故障。 |
| 0<br>FLT0POL | 故障输入0极性<br>定义了故障输入的极性。<br>这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。<br>0b - 故障输入极性为高电平有效。在故障输入端出现1表示有故障。1b - 故障输入 |

| | |
|---|---|
| | 极性为低电平。故障输入端为0表示有故障。 |

# 42.4.3.23 同步配置（SYNCONF）

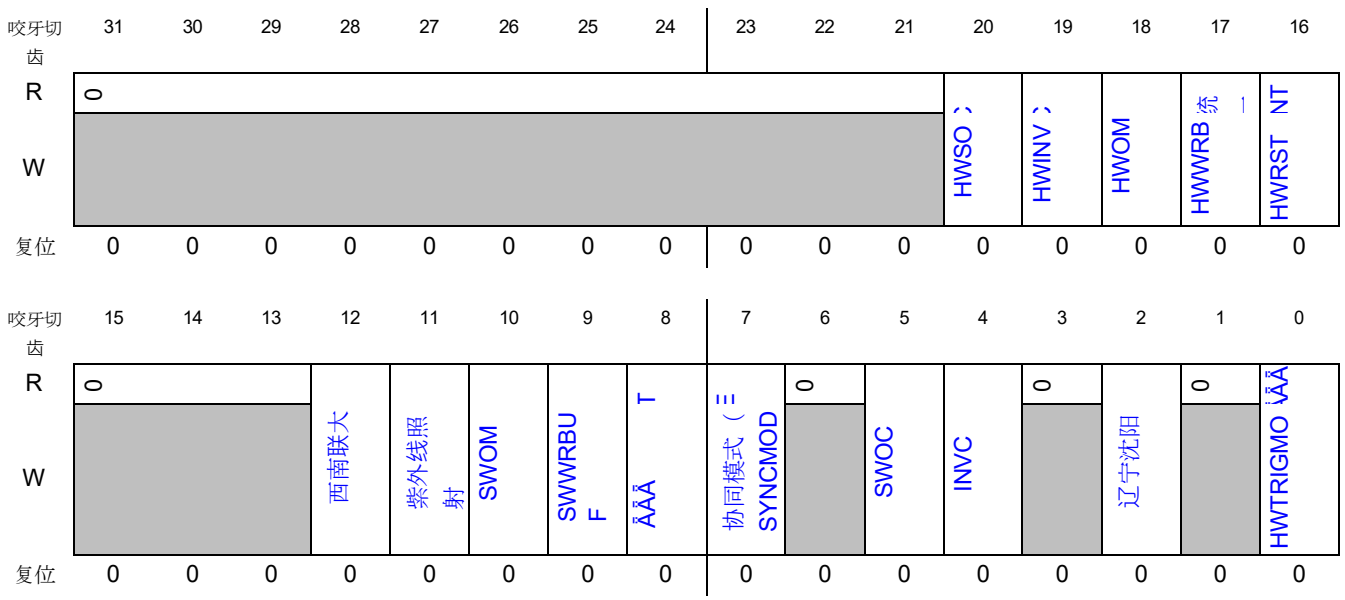
## 42.4.3.23.1 偏移

| 注册 | 偏移 |
|---|---|
| 卫星定位系统(SYNCONF) | 8Ch |

## 42.4.3.23.2 职能

该寄存器选择PWM同步配置，SWOCTRL，INVCTRL和CNTIN寄存器同步，如果FTM清除TRIGj位，其中j=0，1，2，当检测到硬件触发器j。

## 42.4.3.23.3 图示




## 42.4.3.23.4 领域

| 场地 | 职能 |
|---|---|
| 31-21<br>- | 保留 |
| 20<br>HWSOC | 软件输出控制同步是由硬件触发器激活的<br>0b - 硬件触发器不激活SWOCTRL寄存器的同步。1b - 硬件触发器激活了SWOCTRL寄存器的同步。 |

| | |
|---|---|
| 19<br>HWINVC | 反向控制同步是由硬件触发器激活的<br>0b - 一个硬件触发器不激活INVCTRL寄存器的同步。1b - 一个硬件触发器激活INVCTRL寄存器的同步。 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| 18<br>HWOM | 输出掩码同步由一个硬件触发器激活<br>0b - 一个硬件触发器不激活OUTMASK寄存器的同步。1b - 硬件触发器激活了OUTMASK寄存器的同步。 |
| 17<br>HWWRBUF | MOD、HCR、CNTIN和CV寄存器的同步由一个硬件触发器激活<br>0b - 硬件触发器不激活MOD、HCR、CNTIN和CV寄存器的同步。1b - 硬件触发器激活了MOD、HCR、CNTIN和CV寄存器的同步。 |
| 16<br>HWRSTCNT | FTM计数器的同步是由硬件触发器激活的<br>0b - 硬件触发器不激活FTM计数器的同步。1b - 硬件触发器激活了FTM计数器的同步。 |
| 15-13<br>- | 保留 |
| 12<br>西南联大 | 软件输出控制同步是由软件触发器激活的<br>0b - 软件触发器不激活SWOCTRL寄存器的同步。1b - 软件触发器激活了SWOCTRL寄存器的同步。 |
| 11<br>紫外线照射 | 反向控制同步是由软件触发激活的<br>0b - 软件触发器不激活INVCTRL寄存器的同步。1b - 软件触发器激活了INVCTRL寄存器的同步。 |
| 10<br>SWOM | 输出掩码同步是由软件触发激活的<br>0b - 软件触发器不激活OUTMASK寄存器的同步。1b - 软件触发器激活了OUTMASK寄存器的同步。 |
| 9<br>ÅÃÃ | MOD、HCR、CNTIN和CV寄存器的同步是由软件触发的。<br>0b - 软件触发器不激活MOD、HCR、CNTIN和CV寄存器的同步。1b - 软件触发器激活了MOD、HCR、CNTIN和CV寄存器的同步。 |
| 8<br>卫星导航系统(SWRSTCNT) | FTM计数器的同步是由软件触发的<br>0b - 软件触发器不激活FTM计数器的同步。1b - 软件触发器激活了FTM计数器的同步。 |
| 7<br>同步模式 | 同步模式<br>选择PWM同步模式。<br>0b - 选择传统的PWM同步。1b - 选择增强型PWM同步。 |
| 6<br>- | 保留 |
| 5<br>SWOC | SWOCTRL 寄存器同步化<br>0b - SWOCTRL 寄存器在 FTM 输入时钟的所有上升沿更新其缓冲器的值。1b - SWOCTRL寄存器通过PWM同步更新其缓冲区的值。 |
| 4<br>INVC | INVCTRL 寄存器同步化<br>0b - INVCTRL 寄存器在 FTM 输入时钟的所有上升沿更新其缓冲值。1b - INVCTRL寄存器通过PWM同步更新其缓冲区的值。 |
| 3<br>- | 保留 |

| | |
|---|---|
| 2<br>辽宁沈阳 | CNTIN 寄存器同步化<br>0b - CNTIN寄存器在FTM输入时钟的所有上升沿更新其缓冲区的值。1b - CNTIN寄存器通过PWM同步更新其缓冲区的值。 |
| 1<br>- | 保留 |
| 0<br>HWTRIGMODE | 硬件触发模式<br>0b - 当检测到硬件触发器j时，FTM清除TRIGj位，其中j=0，1，2。<br>1b - 当检测到硬件触发器j时，FTM不清除TRIGj位，其中j=0，1，2。 |

## 42.4.3.24 FTM反转控制 (INVCTRL)

### 42.4.3.24.1 偏移

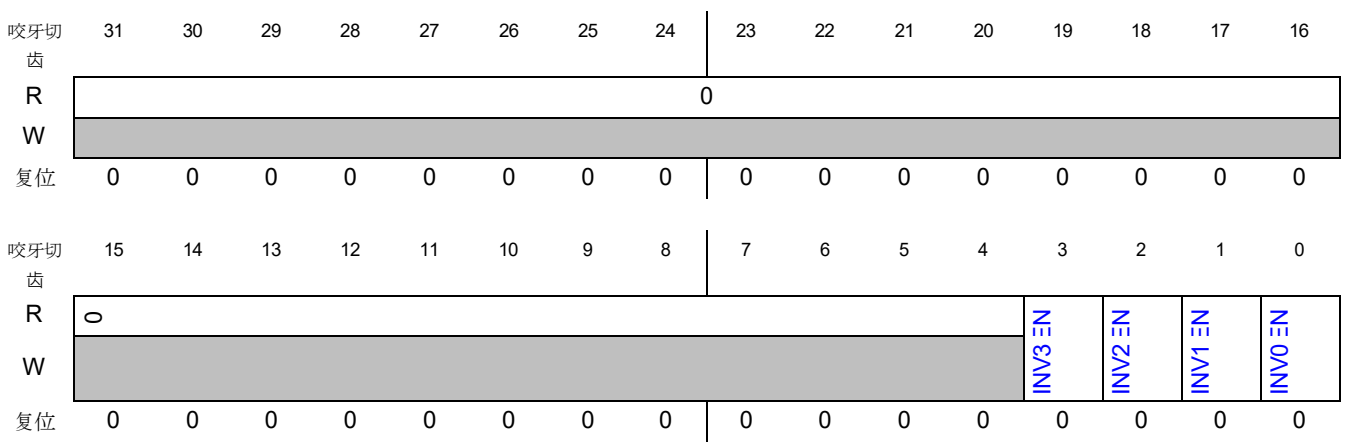
| 注册 | 偏移 |
|---|---|
| 识别系统 | 90h |

### 42.4.3.24.2 职能

该寄存器控制通道(n)输出何时变成通道(n+1)输出，通道(n+1)输出何时变成通道(n)输出。每个INVmEN位使相应的一对通道m的反相操作。

该寄存器有一个写缓冲器。INVmEN位由INVCTRL寄存器同步更新。

### 42.4.3.24.3 图示




### 42.4.3.24.4 领域

| 场地 | 职能 |
|---|---|
| 31-4<br>- | 保留 |
| 3<br>INV3EN | 成对通道 3 颠倒启用 0b - 颠倒功能被禁用。1b - 颠倒功能被启用。 |

| | |
|---|---|
| 2 | 成对通道2反转启用 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| INV2EN | 0b - 颠倒功能被禁用。1b - 启用反转功能。 |
| 1<br>INV1EN | Pair Channels 1 Inverting Enable<br>0b - Inverting is disabled.1b - 颠倒功能被启用。 |
| 0<br>INV0EN | Pair Channels 0 Inverting Enable<br>0b - Inverting is disabled.1b - 颠倒功能被启用。 |

## 42.4.3.25 FTM 软件输出控制 (SWOCTRL)

### 42.4.3.25.1 偏移

| 注册 | 偏移 |
| --- | --- |
| 卫星定位系统(SWOCTRL) | 94h |

### 42.4.3.25.2 职能

该寄存器使软件能够控制通道（n）的输出，并定义强制到通道（n）输出的值。

- CH(n)OC位使软件能够控制相应的通道（n）输出。
- CH(n)OCV位选择在相应的通道(n)输出端强制执行的值。

这个寄存器有一个写缓冲器。这些字段是由SWOCTRL寄存器同步更新的。

### 42.4.3.25.3 图示

| 咬牙切齿 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R<br>W | CH7OCV | CH6OCV | CH5OCV | CH4OCV | CH3OCV | CH2OCV | CH1OCV | CH0OCV | CH7OC | CH6OC | CH5OC | CH4OC | CH3OC | CH2OC | CH1OC | CH0OC |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

### 42.4.3.25.4 领域

| 场地 | 职能 |
|---|---|
| 31-16<br>- | 保留 |
| 15<br>CH7OCV | 通道7软件输出控制值<br>0b - 软件输出控制强制0到通道输出。1b - 软件输出控制强迫通道输出为1。 |
| 14<br>CH6OCV | 通道6软件输出控制值<br>0b - 软件输出控制强制0到通道输出。1b - 软件输出控制强迫通道输出为1。 |
| 13<br>CH5OCV | 通道5软件输出控制值<br>0b - 软件输出控制强制0到通道输出。1b - 软件输出控制强迫通道输出为1。 |
| 12<br>CH4OCV | 通道4软件输出控制值<br>0b - 软件输出控制强制0到通道输出。1b - 软件输出控制强迫通道输出为1。 |
| 11<br>CH3OCV | 通道3软件输出控制值<br>0b - 软件输出控制强制0到通道输出。1b - 软件输出控制强迫通道输出为1。 |
| 10<br>CH2OCV | 通道2软件输出控制值<br>0b - 软件输出控制强制0到通道输出。1b - 软件输出控制强迫通道输出为1。 |
| 9<br>CH1OCV | 通道1软件输出控制值<br>0b - 软件输出控制强制0到通道输出。1b - 软件输出控制强迫通道输出为1。 |

| | |
|---|---|
| 8<br>CH0OCV | 通道0的软件输出控制值<br>0b - 软件输出控制强制0到通道输出。1b - 软件输出控制强迫通道输出为1。 |
| 7<br>CH7OC | 通道7软件输出控制启用<br>0b - 通道输出不受软件输出控制的影响。1b - 通道的输出受软件输出控制的影响。 |
| 6 | 通道6软件输出控制启用 |

*表继续在下一页...*

| 42.4.3.25.3 场地 | 图示 | 职能 |
| --- | --- | --- |
| CH6OC<br>咬牙切齿 | | 0b - 通道输出不受软件输出控制的影响。1b - 通道的输出受软件输出控制的影响。 |
| R W 复位 | 5<br>CH5OC | 通道5软件输出控制启用<br>0b - 通道输出不受软件输出控制的影响。1b - 通道的输出受软件输出控制的影响。 |
| 咬牙切齿 | 4<br>CH4OC | 通道4软件输出控制启用<br>0b - 通道输出不受软件输出控制的影响。1b - 通道的输出受软件输出控制的影响。 |
| W R 复位 | 3<br>CH3OC | 通道3软件输出控制启用<br>0b - 通道输出不受软件输出控制的影响。1b - 通道的输出受软件输出控制的影响。 |
| | 2<br>CH2OC | 通道2软件输出控制启用<br>0b - 通道输出不受软件输出控制的影响。1b - 通道的输出受软件输出控制的影响。 |
| | 1<br>CH1OC | 通道1软件输出控制启用<br>0b - 通道输出不受软件输出控制的影响。1b - 通道的输出受软件输出控制的影响。 |
| | 0<br>CH0OC | 通道0软件输出控制启用<br>0b - 通道输出不受软件输出控制的影响。1b - 通道的输出受软件输出控制的影响。 |

## 42.4.3.26 FTM PWM负载（PWMLOAD）。

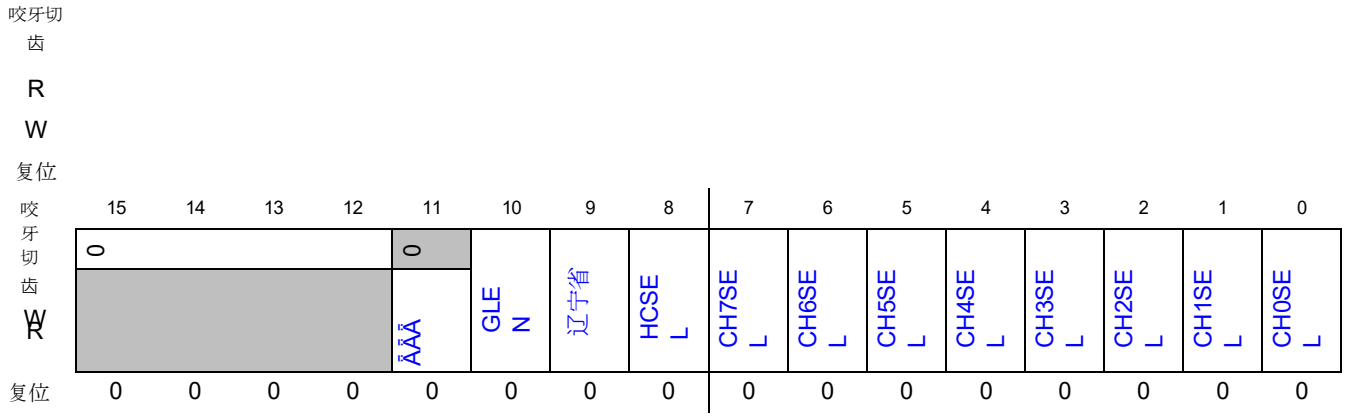
### 42.4.3.26.1 偏移

| 注册 | 偏移 |
| --- | --- |
| PWMLOAD | 98h |

### 42.4.3.26.2 职能

当FTM计数器从MOD寄存器的值变为下一个值或发生通道(j)匹配时，启用MOD、HCR、CNTIN、C(n)V和C(n+1)V寄存器的写缓冲区的值重新加载。当FTM计数器= C(j)V时，通道(j)发生匹配。当 FTM 计数器 = HCR
寄存器在半周期匹配时，也可以发生重载。这个寄存器也控制着本地和全局的加载机制。

## 42.4.3.26.3 图示




## 42.4.3.26.4 领域

| 场地 | 职能 |
|---|---|
| 31-12<br>- | 保留 |
| 11<br>辽宁省 | 全局加载OK<br>该位控制全局负载机制。它在 FTM 模块的全局负载输出端产生一个脉冲，其宽度为一个 FTM 时钟周期，用于设置 FTM 和其他模块（包括其他 FTM）的 LDOK 位。该位是自清的，读值总是为零。<br>全局加载机制取决于SoC的具体信息。请参考FTM SoC的具体信息以了解更多细节。<br>0b - 没有行动。<br>1b - LDOK位被设置。 |
| 10<br>GLEN | 全局负载启用<br>该位启用由GLDOK实现的全局负载机制。如果GLEN位被设置，那么在FTM全局负载输入上的一个外部事件将设置LDOK位。LDOK位的清除是由CPU向该位写 "0 "来完成的。<br>0b - 禁用全局加载Ok。<br>1b - 启用全局负载确定。模块全局负载输入的脉冲事件设置LDOK位。 |
| 9<br>辽宁省 | 启用负载<br>启用MOD、CNTIN、HCR和CV寄存器加载其缓冲区的值。如果GLEN位被启用，LDOK位也可<br>以被全局加载机制设置。<br>0b - 禁止加载更新的值。1b - 启用加载更新值。 |

| | |
|---|---|
| 8<br>HCSEL | 半周期选择<br>该位启用半周期匹配作为重载的机会。半周期的定义是FTM计数器与HCR寄存器匹配的时间。<br>0b - 半周期重载被禁用，它不被视为重载机会。1b - 启用半周期重载，它被视为重载机会。 |
| 7 | 频道7选择<br>0b - 通道匹配不包括在重载机会中。 |

*表继续在下一页...*

| 42.4.3.26.3 场地 | 图示 | 职能 |
|---|---|---|
| | CH7SEL | 1b - 频道匹配被列为重载机会。 |
| 咬牙切齿 | 6<br>CH6SEL<br>R<br>W | 频道6选择<br>0b - 频道匹配不包括在重载机会中。1b - 频道匹配被包括在重载机会中。 |
| 复位 | 5<br>咬牙切齿<br>CH5SEL | 频道5选择<br>0b - 频道匹配不包括在重载机会中。1b - 频道匹配被包括在重载机会中。 |
| | 4<br>W<br>R<br>CH4SEL<br>复位 | 第4频道选择<br>0b - 频道匹配不包括在重载机会中。1b - 频道匹配被包括在重载机会中。 |
| | 3<br>CH3SEL | 通道3选择<br>0b - 频道匹配不包括在重载机会中。1b - 频道匹配被包括在重载机会中。 |
| | 2<br>CH2SEL | 通道2选择<br>0b - 频道匹配不包括在重载机会中。1b - 频道匹配被包括在重载机会中。 |
| | 1<br>辽宁沈阳 | 通道1选择<br>0b - 频道匹配不包括在重载机会中。1b - 频道匹配被包括在重载机会中。 |
| | 0<br>CH0SEL | 通道0选择<br>0b - 频道匹配不包括在重载机会中。1b - 频道匹配被包括在重载机会中。 |

## 42.4.3.27 半周期寄存器（HCR）

### 42.4.3.27.1 偏移

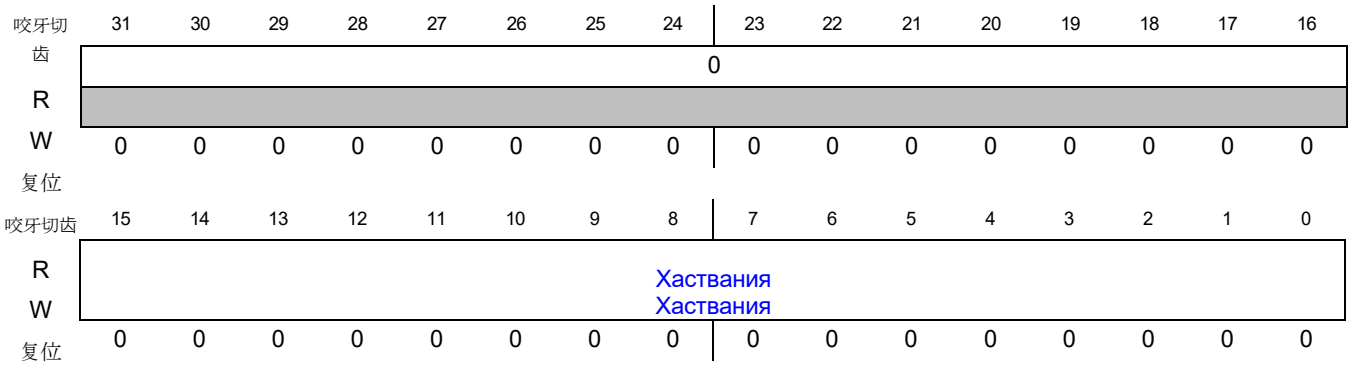
| 注册 | 偏移 |
|---|---|
| HCR | 9Ch |

### 42.4.3.27.2 职能

半周期寄存器包含 FTM 半周期重载功能的匹配值。在 FTM 计数器达到这个值后，如果 FTM_PWMLOAD[HCSEL]被启用，就会产生一个重载机会。

写入HCR寄存器的值被锁入一个缓冲区。根据从写缓冲区更新的寄存器，HCR寄存器的值会被更新。

### 42.4.3.27.3 图示




### 42.4.3.27.4 领域

| 场地 | 职能 |
|---|---|
| 31-16<br>- | 保留 |
| 15-0<br>Хаствания<br>Хаствания | 半周期值 |

## 42.4.3.28 第0对死机时间配置（PAIR0DEADTIME）。

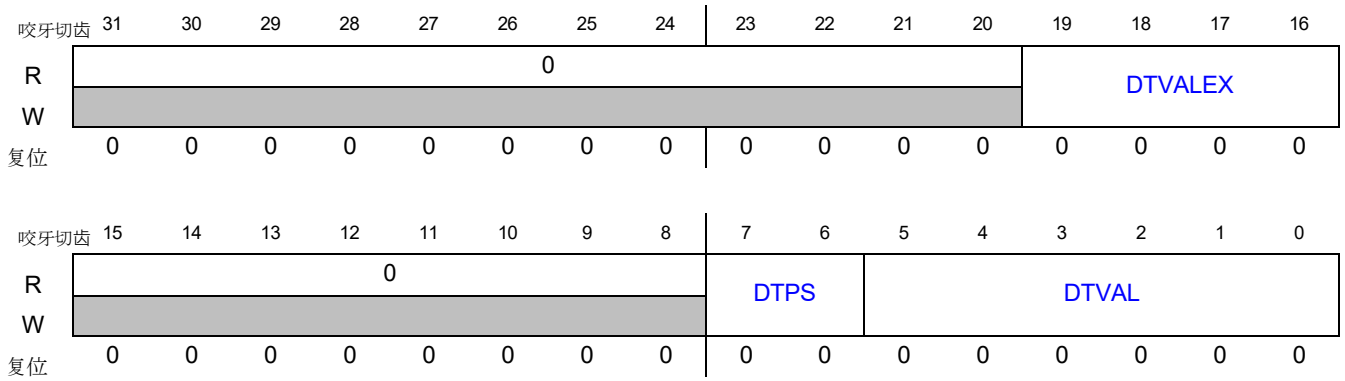
### 42.4.3.28.1 偏移

| 注册 | 偏移 |
|---|---|
| 伴侣0DEADTIME | A0h |

### 42.4.3.28.2 职能

该寄存器选择死区预分频器和对0的值。

## 42.4.3.28.3 图示




## 42.4.3.28.4 领域

| 场地 | 职能 |
|---|---|
| 31-20<br>- | 保留 |
| 19-16<br>DTVALEX | 扩展死区值<br>这个字段是DTVAL字段的一个比特扩展。它定义了死锁时间值的4个最重要的位。使用{DTVALEX, DTVAL}的连接，最大死区值被扩展到1023。<br>死亡时间插入值=（DTPS × {DTVALEX, DTVAL}）。<br>这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。 |
| 15-8<br>- | 保留 |
| 7-6<br>DTPS | 死锁时间预分频器值<br>选择FTM输入时钟的分割系数。这个预标定的时钟被死区计数器所使用。这个字段是受写保护的。只有当MODE[WPDIS]=1时才可以写入。<br>0xb - 将FTM输入时钟除以1。 10b - 将FTM输入时钟除以4。 11b - 将FTM输入时钟除以16。 |
| 5-0<br>DTVAL | 死亡时间值<br>选择死机时间值。<br>这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。 |

## 42.4.3.23 第1对死机时间配置（PAIR1DEADTIME）

图示

咬牙切齿

R

W

复位

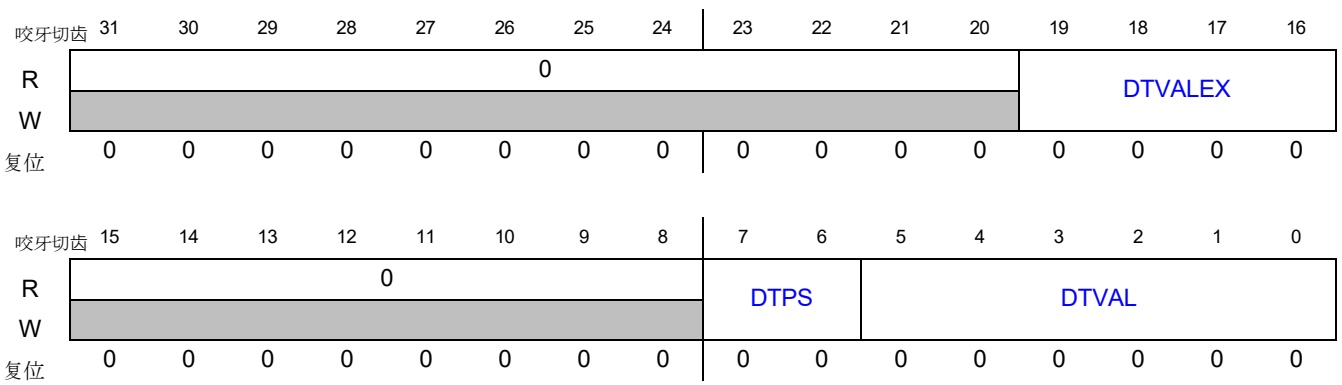
### 42.4.3.29.1 偏移

| 注册 | 偏移 |
|---|---|
| 伴侣1DEADTIME | A8h |

### 42.4.3.29.2 职能

该寄存器选择死区预分频器和对1的值。

### 42.4.3.29.3 图示




### 42.4.3.29.4 领域

| 场地 | 职能 |
|---|---|
| 31-20<br>- | 保留 |
| 19-16<br>DTVALEX | 扩展死区值<br>这个字段是DTVAL字段的一个比特扩展。它定义了死锁时间值的4个最重要的位。使用{DTVALEX, DTVAL}的连接，最大死区值被扩展到1023。<br>死亡时间插入值=（DTPS × {DTVALEX, DTVAL}）。<br>这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。 |
| 15-8<br>- | 保留 |

| | |
|---|---|
| 7-6<br>DTPS | 死锁时间预分频器值<br>选择FTM输入时钟的分割系数。这个预标定的时钟被死区计数器所使用。这个字段是受写保护的。只有当MODE[WPDIS]=1时才可以写入。<br>0xb - 将FTM输入时钟除以1。 10b - 将FTM输入时钟除以4。11b - 将FTM输入时钟除以16。 |
| 5-0 | 死亡时间值 |

| 场地 | 职能 |
|---|---|
| DTVAL | 选择死机时间值。<br>这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。 |

## 42.4.3.30 第2对死机时间配置（PAIR2DEADTIME）

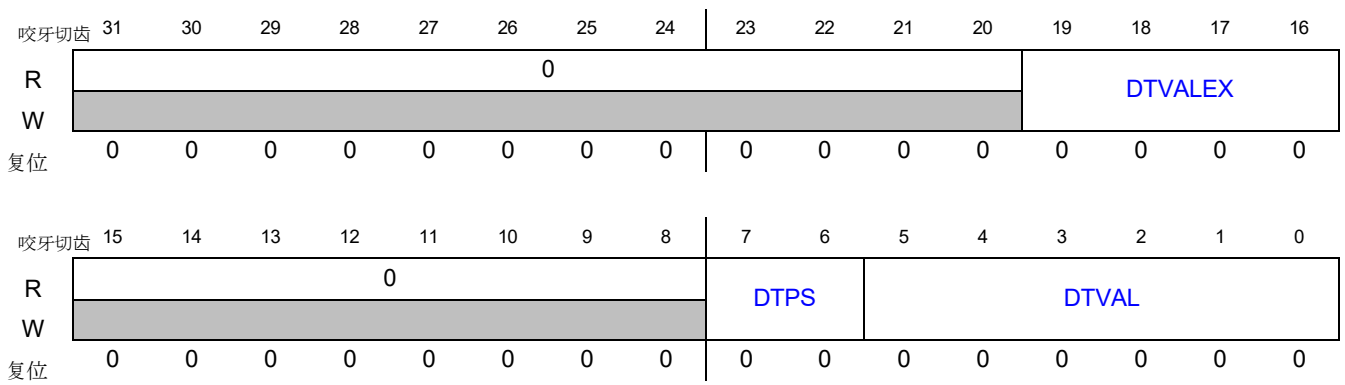
### 42.4.3.30.1 偏移

| 注册 | 偏移 |
|---|---|
| 伙伴2DEADTIME | B0h |

### 42.4.3.30.2 职能

该寄存器选择死区预分频器和对2的值。

### 42.4.3.30.3 图示




### 42.4.3.30.4 领域

| 场地 | 职能 |
|---|---|
| 31-20<br>- | 保留 |

| | |
|---|---|
| 19-16<br>DTVALEX | 扩展死区值<br>这个字段是DTVAL字段的一个比特扩展。它定义了死锁时间值的4个最重要的位。使用{DTVALEX, DTVAL}的连接，最大死区值被扩展到1023。<br>死亡时间插入值=（DTPS × {DTVALEX, DTVAL}）。<br>这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| 15-8<br>- | 保留 |
| 7-6<br>DTPS | 死锁时间预分频器值<br>选择FTM输入时钟的分割系数。这个预标定的时钟被死区计数器所使用。这个字段是受写保护的。只有当MODE[WPDIS]=1时才可以写入。<br>0xb - 将FTM输入时钟除以1。 10b - 将FTM输入时钟除以4。11b - 将FTM输入时钟除以16。 |
| 5-0<br>DTVAL | 死亡时间值<br>选择死机时间值。<br>这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。 |

## 42.4.3.31 第3对死机时间配置（PAIR3DEADTIME）

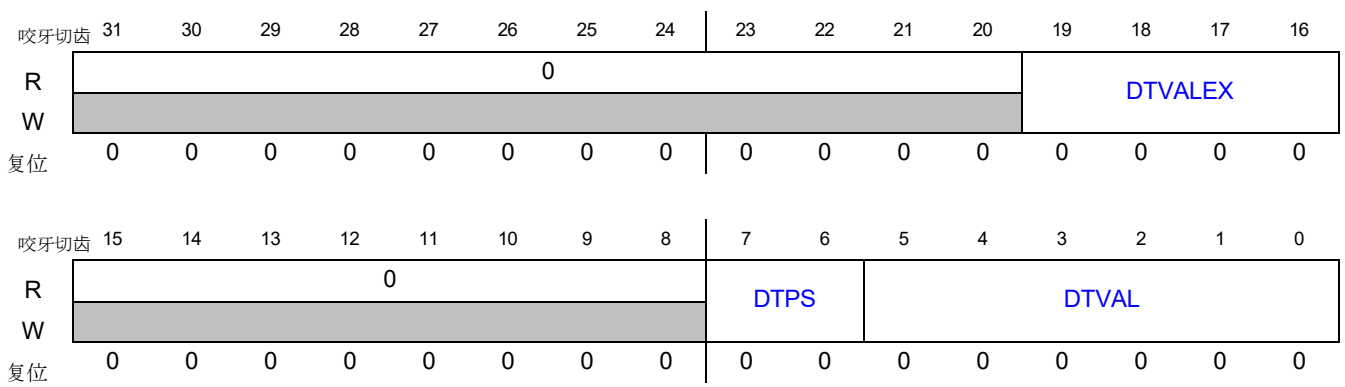
### 42.4.3.31.1 偏移

| 注册 | 偏移 |
|---|---|
| 伴侣3EADTIME | B8h |

### 42.4.3.31.2 职能

该寄存器选择死区预分频器和对3的值。

### 42.4.3.31.3 图示



#### 42.4.3.31.4 领域

| 场地 | 职能 |
|---|---|
| 31-20<br>- | 保留 |
| 19-16<br>DTVALEX | 扩展死区值<br>这个字段是DTVAL字段的一个比特扩展。它定义了死锁时间值的4个最重要的位。使用{DTVALEX, DTVAL}的连接，最大死区值被扩展到1023。<br>死亡时间插入值=（DTPS × {DTVALEX, DTVAL}）。<br>这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。 |
| 15-8<br>- | 保留 |
| 7-6<br>DTPS | 死锁时间预分频器值<br>选择FTM输入时钟的分割系数。这个预标定的时钟被死区计数器所使用。这个字段是受写保护的。只有当MODE[WPDIS]=1时才可以写入。<br>0xb - 将FTM输入时钟除以1。 10b - 将FTM输入时钟除以4。 11b - 将FTM输入时钟除以16。 |
| 5-0<br>DTVAL | 死亡时间值<br>选择死机时间值。<br>这个字段是受写保护的。只有当MODE[WPDIS]=1时才能写入。 |

## 42.4.3.32 模数值的镜像 (MOD_MIRROR)

### 42.4.3.32.1 偏移

| 注册 | 偏移 |
|---|---|
| MOD_MIRROR | 200h |

### 42.4.3.32.2 职能

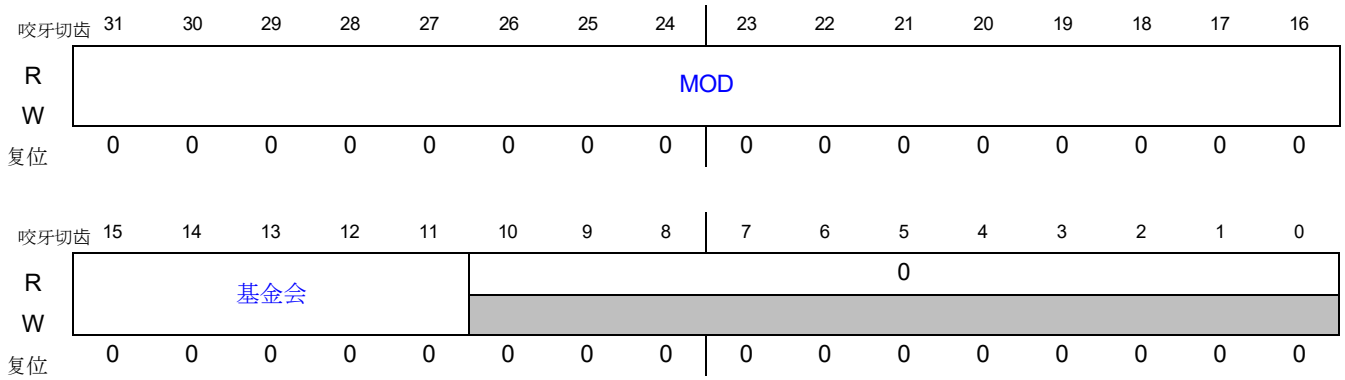
该寄存器包含FTM计数器的整数和小数模值。

## 注意事项

这个寄存器不是每个模块实例都使用的。

- 适用于FTM1、FTM2。
- 不适用于FTM0, FTM3, FTM4, FTM5, FTM6, FTM7。

### 42.4.3.32.3 图示




### 42.4.3.32.4 领域

| 场地 | 职能 |
|---|---|
| 31-16<br>MOD | 模数整数值的镜像<br>参见寄存器MOD的字段MOD。 |
| 15-11<br>基金会 | 模数化的分数值<br>该模数小数值用于PWM周期抖动。这个值在每个PWM周期结束时被加到一个内部累加器中。<br>对字段FRACMOD的写入是在其写缓冲器上进行的。根据从写缓冲区更新的寄存器，FRACMOD用其写缓冲区的值进行更新。如果FTMEN=0，对SC寄存器的写入会手动重置这个写入一致性机制。 |
| 10-0<br>- | 保留 |

## 42.4.3.33 通道的镜像（n）匹配值（C0V_MIRROR - C7V_ MIRROR）。

### 42.4.3.33.1 偏移

对于a=0到7。

| 注册 | 偏移 |
|---|---|
| 镜像 | 204h + (a × 4h) |

### 42.4.3.33.2 职能

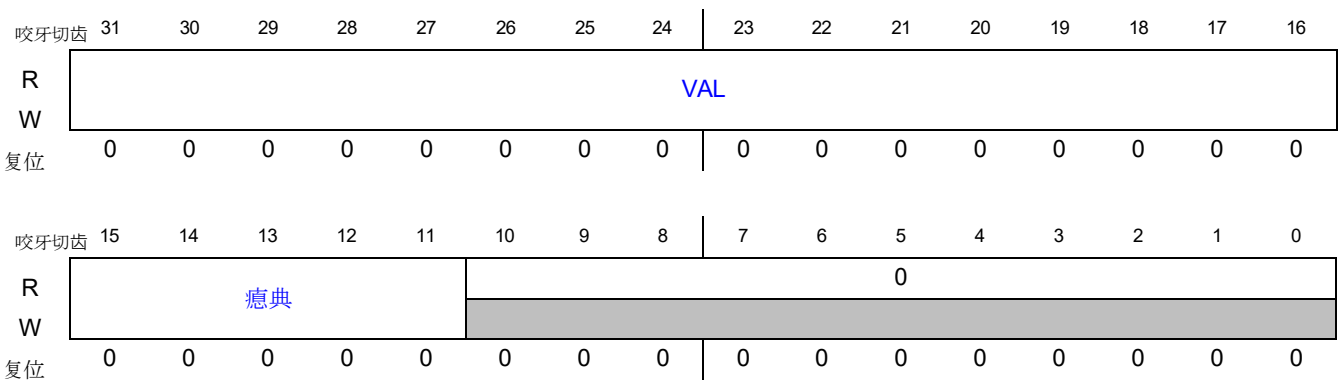
该寄存器包含通道（n）匹配的整数和小数值。

**注意事项**

这个寄存器不是每个模块实例都使用的。

- 适用于FTM1、FTM2。
- 不适用于FTM0, FTM3, FTM4, FTM5, FTM6, FTM7。

### 42.4.3.33.3 图示




### 42.4.3.33.4 领域

| 场地 | 职能 |
|---|---|
| 31-16<br>VAL | 通道的镜像（n）匹配整数值<br>参见寄存器CnV的字段VAL。 |
| 15-11<br>癒典 | 通道（n） 匹配分数值<br>通道（n）的匹配小数值用于PWM边缘抖动。这个值在每个PWM周期结束时被添加到通道（n）的内部累积器中。<br>对字段FRACVAL的写入是在其写缓冲器上进行的。根据从写缓冲区更新的寄存器，FRACVAL与它的写缓冲区值一起更新。如果FTMEN=0，对CnSC寄存器的写入会手动重置这个写入一致性机制。 |
| 10-0<br>- | 保留 |

## 42.5 功能描述

### 42.5.1 时钟源

FTM只有一个时钟域：FTM输入时钟。

#### 42.5.1.1 计数器时钟源

CLKS[1:0]位为 FTM 计数器选择三个可能的时钟源之一，或者禁用 FTM 计数器。在任何芯片复位后，CLKS[1:0] = 0:0，因此没有选择时钟源。

CLKS[1:0]位可以在任何时候读或写。通过向CLKS[1:0]位写0:0来禁用FTM计数器，并不影响FTM计数器的值或其他寄存器。

固定频率时钟是FTM计数器的替代时钟源，允许选择FTM输入时钟以外的时钟或外部时钟。这个时钟输入是由芯片集成定义的。更多信息请参考芯片的具体文件。由于FTM硬件实现的限制，固定频率时钟的频率不能超过FTM输入时钟频率的1/2。

外部时钟通过FTM输入时钟的同步器，以确保计数器的转换与FTM输入时钟的转换正确对齐。因此，为了满足奈奎斯特标准，同时考虑到抖动，外部时钟源的频率必须不超过FTM输入时钟频率的1/4。

### 42.5.2 预分频器

选择的计数器时钟源通过一个预分频器，该预分频器是一个7位的计数器。预分频器的值由PS[2:0]位选择。下图是一个预分频器计数器和FTM计数器的例子。

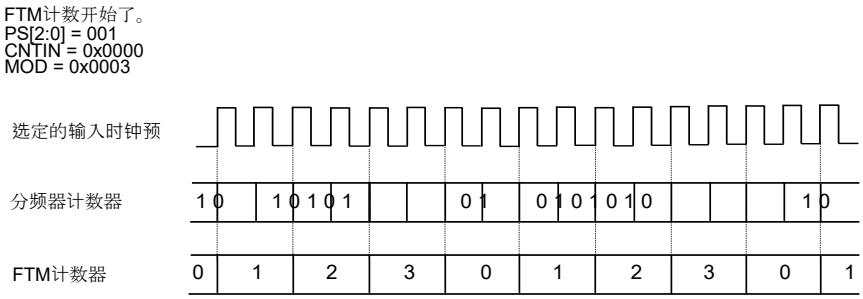



**图42-7.预分频器计数器的例子**

## 42.5.3 计数器

FTM有一个16位的计数器，被各通道用于输入或输出模式。FTM计数器的时钟是选定的时钟除以预分频器。

FTM计数器有这些操作模式。

- 上升计数
- 自上而下计数
- 正交解码器模式

### 42.5.3.1 上升计数

当选择向上计数时。

- QUADEN = 0, 和
- CPWMS = 0

CNTIN定义了计数的起始值，MOD定义了计数的最终值，见下图。CNTIN的值被加载到FTM计数器中，计数器不断增加，直到达到MOD的值，这时计数器被重新加载CNTIN的值。

使用向上计数时的 FTM 周期是（MOD - CNTIN + 0x0001）×FTM 计数器时钟的周期。

当FTM计数器从MOD变为CNTIN时，TOF位被设置。

当FTM计数器从MOD变为CNTIN时，在TOF位设置的同时发生一个计数器事件。更多细节请参见计数器事件。

FTM的计数已经开始。
CNTIN = 0xFFFC (在二进制补码中等于-4)
MOD = 0x0004

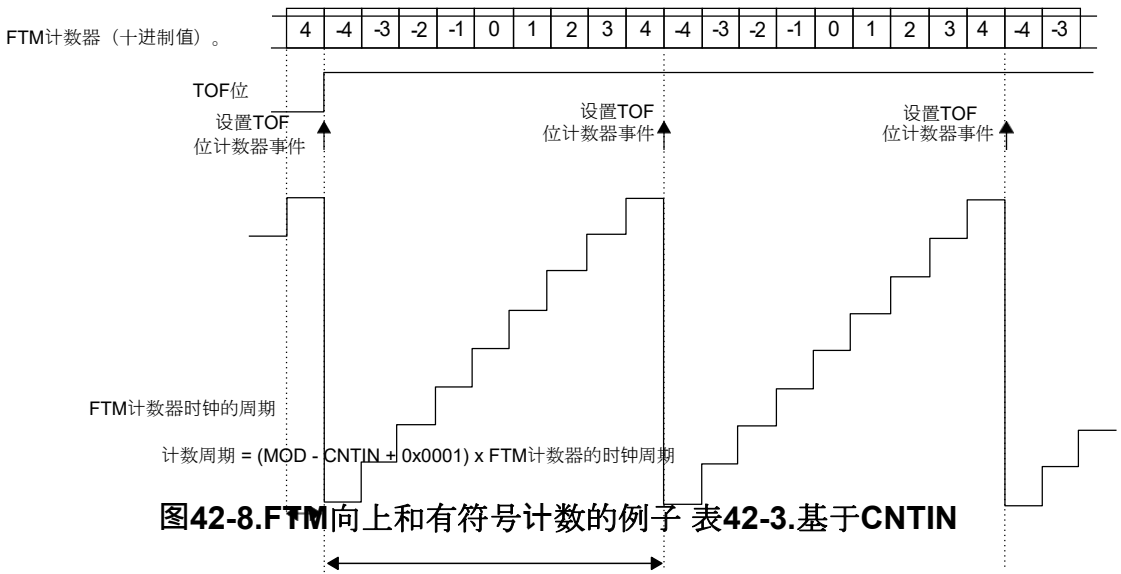



**图42-8.FTM向上和有符号计数的例子 表42-3.基于CNTIN值的FTM计数**

| 当 | 那么 |
|---|---|
| CNTIN = 0x0000 | FTM计数相当于TPM向上计数，即向上和无符号计数。见下图。 |
| CNTIN[15] = 1 | FTM计数器的初始值是一个二补的负数，所以FTM的计数是向上的，并有符号。 |
| CNTIN[15]=0且CNTIN≠0x0000 | FTM计数器的初始值是一个正数，所以FTM的计数是向上的，而且是无符号的。 |

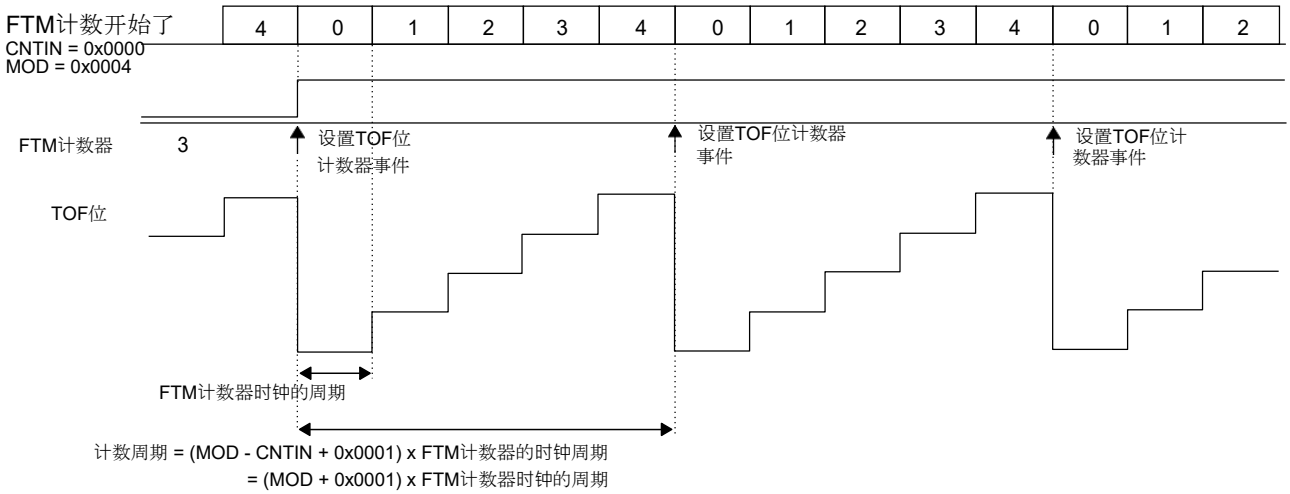



**图42-9.CNTIN=0x0000时FTM向上计数的例子**

## 注意事项

- 只有当CNTIN寄存器的值小于MOD寄存器的值时，FTM操作才有效，无论是在无符号计数还是有符号计数。软件有责任确保CNTIN和MOD寄存器的值满足这一要求。任何不满足这个条件的CNTIN和MOD的值都会导致不可预测的行为。
- MOD = CNTIN是一个多余的条件。在这种情况下，FTM计数器总是等于MOD，TOF位在FTM计数器时钟的每个上升沿被设置。
- 当MOD=0x0000，CNTIN=0x0000时，例如在复位后，FTMEN=1，FTM计数器保持停在0x0000，直到一个非零值被写入MOD或CNTIN寄存器中。
- 不建议将CNTIN设置为大于MOD的值，因为这种不寻常的设置可能使FTM的操作难以理解。但是，对这种配置没有限制，下图是一个例子。

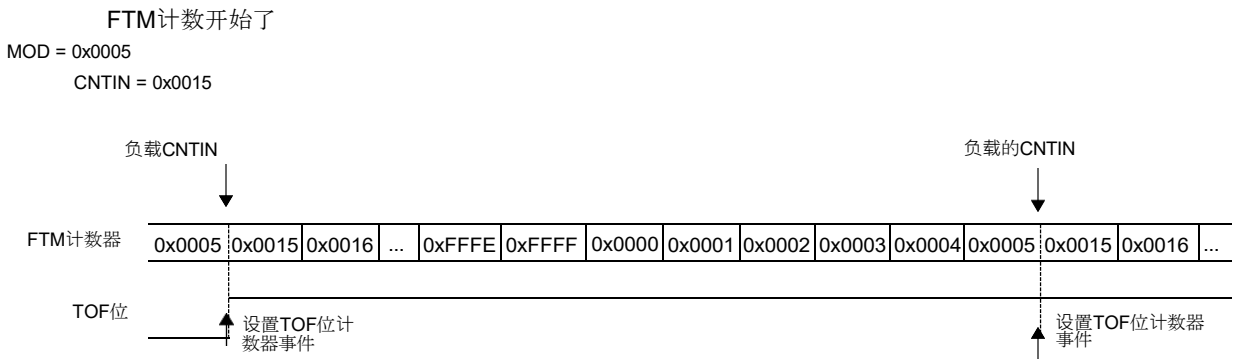



**图42-10.当CNTIN的值大于MOD的值时向上计数的例子**

### 42.5.3.2 自上而下计数

在以下情况下，选择向上-向下计数。

- QUADEN = 0, 和
- CPWMS = 1

CNTIN定义了计数的起始值，MOD定义了计数的最终值。CNTIN的值被加载到FTM计数器中，计数器不断增加，直到达到MOD的值，这时计数器被递减，直到回到CNTIN的值，并重新开始上下计数。

使用上-下计数时的FTM周期为2×（MOD-CNTIN）×FTM计数器时钟的周期。

当FTM计数器从MOD变为（MOD-1）时，TOF位被设置。

如果(CNTIN=0x0000)，FTM计数相当于TPM的上下计数，即上下和无符号计数。见下图。

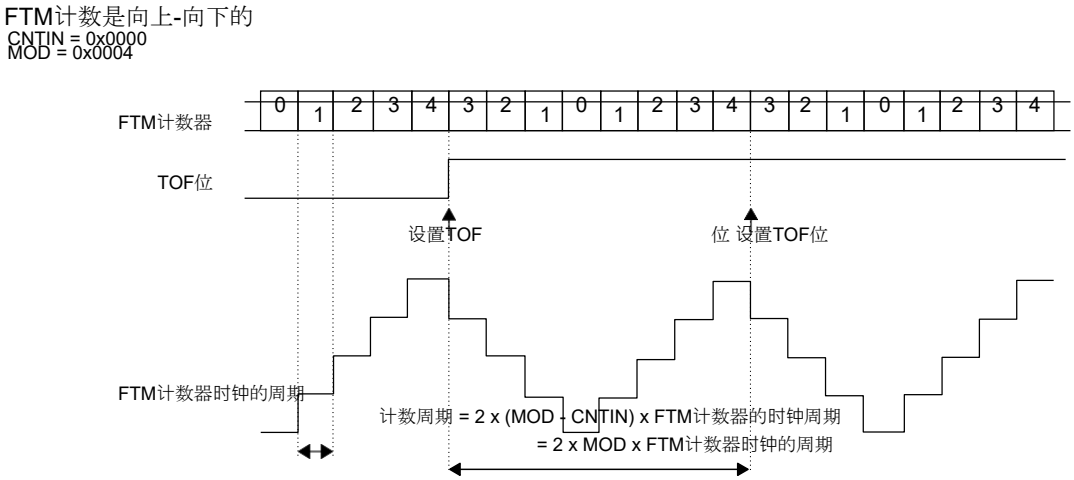



**图42-11.当CNTIN=0x0000时，向上-向下计数的例子**

## 注意事项

当CNTIN在向上-
向下计数中与零不同时，会产生一个有效的CPWM信号。

- 如果CnV>CNTIN，或
- 如果CnV = 0或CnV[15] =
  1。在这种情况下，会产生0%的CPWM。

下图显示了在向上-
向下计数模式时可能发生的计数器事件。更多细节请参见计数器事件。

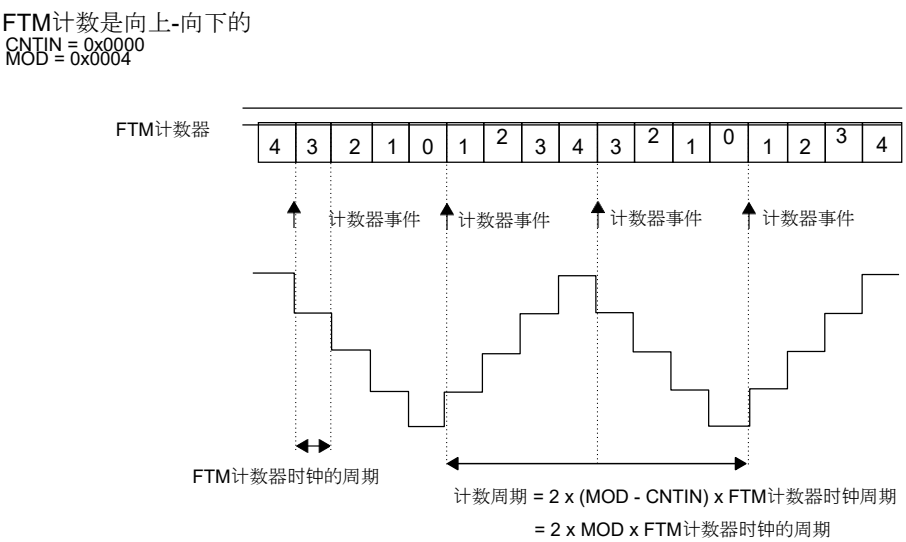



**图42-12.当CNTIN=0x0000时，在上-下计数模式下的计数器事件示例**

### 42.5.3.3 自由运行的计数器

如果（FTMEN=0）和（MOD=0x0000或MOD=0xFFFF），FTM计数器是一个自由运行的计数器。在这种情况下，FTM计数器从0x0000到0xFFFF自由运行，当FTM计数器从0xFFFF变为0x0000时，TOF位被设置。见下图。

当FTM计数器从0xFFFF变为0x0000时，在TOF位设置的同时发生一个计数器事件。更多细节请参见计数器事件。

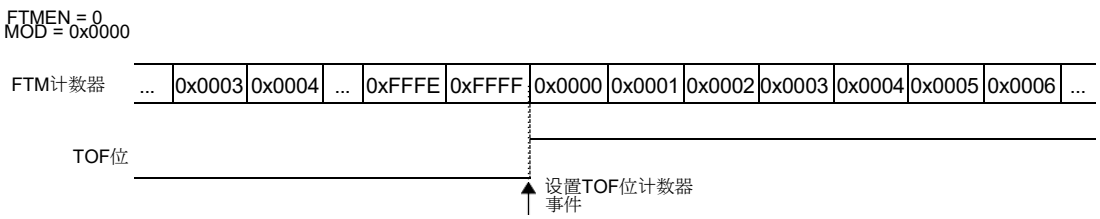



**图 42-13.FTM计数器自由运行时的例子**

FTM计数器也是一个自由运行的计数器，当。

- FTMEN = 1
- QUADEN = 0
- CPWMS = 0
- CNTIN = 0x0000, 和
- MOD = 0xFFFF

### 42.5.3.4 计数器复位

以下任何一种情况都会将 FTM 计数器重置为 CNTIN
寄存器中的值，并将通道输出重置为其初始值，但输出比较模式下的通道除外。

- 对CNT的任何写入。
- FTM计数器同步。
- 一个处于输入捕捉模式的通道，ICRST=1（输入捕捉模式下的FTM计数器复位）。

注意，重置计数器也会产生一个计数器事件。更多细节请参见计数器事件。

### 42.5.3.5 计数器事件

计数器事件可以作为FTM寄存器同步机制的重新加载机会。更多细节见重载点。根据计数器模式的不同，有一些可能的计数器事件。请看下表以了解更多细节。

**表42-4.FTM计数器事件**

| 当 | 那么 |
|---|---|
| FTM计数器处于上升计数模式或自由运行状态 | • 当 FTM 计数器从 MOD 变成 CNTIN（计数包）时，在 TOF 位被设置的同时发生一个计数事件。上图显示了计数器事件的产生。<br>• 当处于自由运行状态时，当 FTM 计数器从 0xFFFF 变为 0x0000 时，会有一个计数器事件。图中自由运行计数器显示了计数器事件的产生。 |
| FTM计数器处于向上-向下计数模式 | • 在上-下计数模式下，有两种可能的计数事件，即FTM计数器从向下计数转为向上计数以及计数器从向上计数转为向下计数。用户可以选择使用哪一个点来生成计数器事件。图中的 "上下计数 "显示了可能的计数器事件。 |
| FTM计数器被复位（见计数器复位）或在CLKS字段写入一个不同于零的值 | • 在上计模式下，所有的计数器复位事件或在CLKS中写入一个不同于零的值都会产生一个计数器事件。<br>• 在上下计数模式下，只有在配置了 FTM 计数器从向下计数转为向上计数时的最小负载点时，计数器复位事件才会产生一个计数器事件。在上-下计数模式下，在CLKS中写入一个不同于零的值总是会产生一个计数器事件。 |

## 42.5.4 频道模式

下表显示了通道模式的选择。

**表42-5.通道模式选择**

| デコンページ | MCOMBINE | 融合 | CPWMS | MSB:MSA | ELSB:ELSA | 模式 | 配置 |
|---|---|---|---|---|---|---|---|
| X | X | X | X | XX | 00 | 不用于FTM的引脚--将通道引脚转换为通用I/O或其他外设控制 | |
| 0 | 0 | 0 | 0 | 00 | 01 | 输入捕捉 | 仅在上升沿捕获 |

*表继续在下一页...*

表42-5.通道模式选择（续）

| デコンページ | MCOMBINE | 融合 | CPWMS | MSB:MSA | ELSB:ELSA | 模式 | 配置 |
|---|---|---|---|---|---|---|---|
| | | | | | 10 | | 仅在下降沿捕获 |
| | | | | | 11 | | 在上升或下降沿捕获 |
| | | | | 01 | 01 | 输出比较 | 匹配时切换输出 |
| | | | | | 10 | | 匹配时清除输出 |
| | | | | | 11 | | 匹配时设置输出 |
| | | | | 1X | 10 | 边缘对齐的PWM | 高真实脉冲（匹配时清除输出）。 |
| | | | | | X1 | | 低真脉冲（设置为匹配时输出） |
| | | | 1 | XX | 10 | 中心对齐的PWM | 高真实脉冲（匹配时输出清晰）。 |
| | | | | | X1 | | 低真脉冲（在匹配时设置输出）。 |
| | | 1 | 0 | XX | 10 | 结合PWM | 高真脉冲（在通道（n）匹配时设置，在通道（n+1）匹配时清除） |
| | | | | | X1 | | 低真脉冲（通道（n）匹配时清除，通道（n+1）匹配时设置）。 |
| | 1 | 0 | X | XX | XX | 保留给未来使用 | |
| | 1 | 1 | 0 | XX | 10 | 修改后的组合式PWM | 高真个脉冲（设置在通道（n）匹配上，以及 |

*表继续在下一页...*

表42-5.通道模式选择（续）

| デコンページ | MCOMBINE | 融合 | CPWMS | MSB:MSA | ELSB:ELSA | 模式 | 配置 |
|---|---|---|---|---|---|---|---|
| | | | | | | | 清除通道（n+1）上的匹配）。 |
| | | | | | X1 | | 低真脉冲（通道（n）匹配时清除，通道（n+1）匹配时设置）。 |
| 1 | 0 | 0 | 0 | X0 | 见表42-6。 | 双重边缘捕捉 | 单次拍摄模式 |
| | | | | X1 | | | 连续拍摄模式 |

表42-6.双边缘捕获模式 - 边缘极性选择

| ELSB | ELSA | 通道端口启用 | 检测到的边缘 |
|---|---|---|---|
| 0 | 0 | 失能 | 没有边缘 |
| 0 | 1 | 已启用 | 上升沿 |
| 1 | 0 | 已启用 | 跌落边缘 |
| 1 | 1 | 已启用 | 上升沿和下降沿 |

## 42.5.5 输入捕捉模式

当选择输入捕捉模式时。

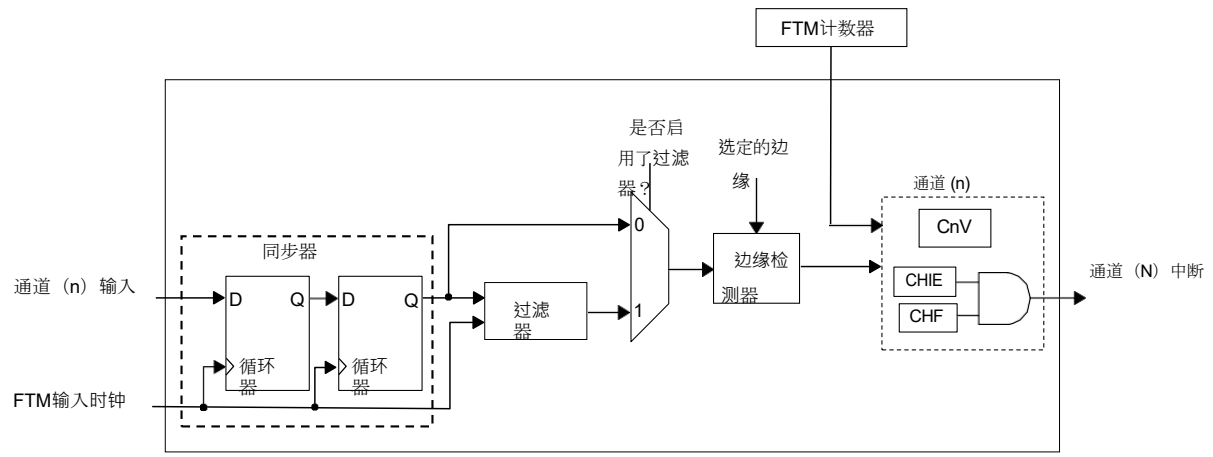
- DECAPEN = 0
- MCOMBINE = 0
- COMBINE = 0
- CPWMS = 0
- MSB:MSA=0:0，并且
- elsb:Elsa ≠ 0:0

当通道输入端出现选定的边沿时，FTM计数器的当前值被捕获到CnV寄存器中，同时CHF位被设置，如果CHIE=1使能，则产生通道中断。见下图。

当一个通道被配置为输入捕捉时，FTMxCHn引脚是一个边沿敏感的输入。ELSB:ELSA位决定哪个边沿，下降或上升，触发输入捕获事件。

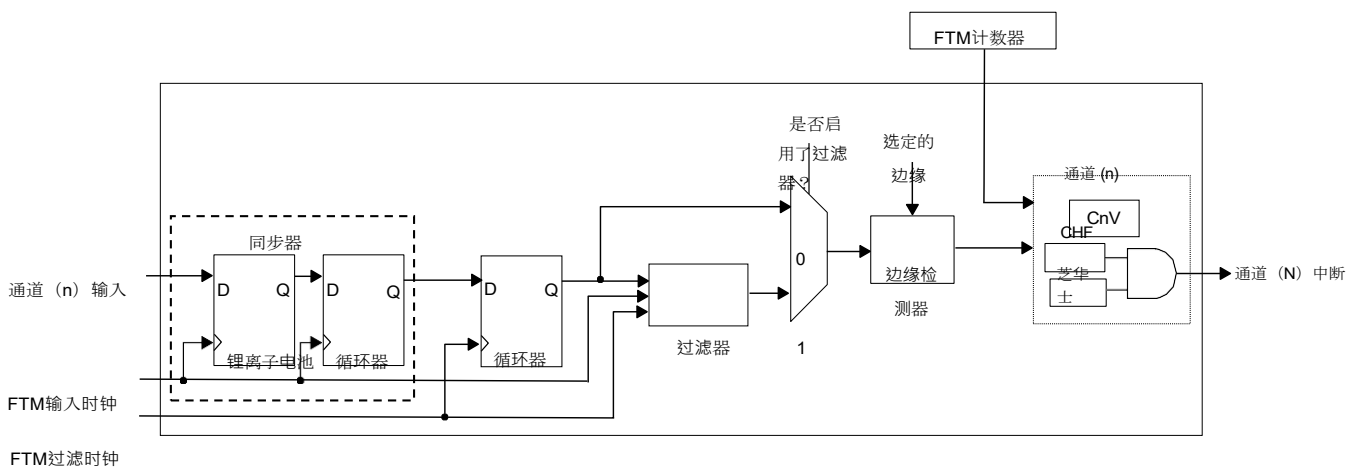
在输入捕捉模式下，对CnV寄存器的写入被忽略。

在调试模式下，输入捕捉功能按配置工作。当一个选定的边沿事件发生时，因调试而被冻结的FTM计数器值被捕获到CnV寄存器，CHF位被设置。




注意：该滤波器仅适用于通道0、1、2和3的输入。

**图42-14.当FLTPS[3:0]=0时输入捕捉模式示意图**




注意：该滤波器仅适用于通道0、1、2和3的输入。

**图42-15.当FLTPS[3:0]≠0时，输入捕捉模式的示意图**

### 42.5.5.1 输入捕捉模式的过滤器

该过滤器只适用于通道0、1、2和3。

如果 FLTPS[3:0] = 0，则由 FTM 输入时钟同步后的通道输入（图 42-

14为滤波器输入。

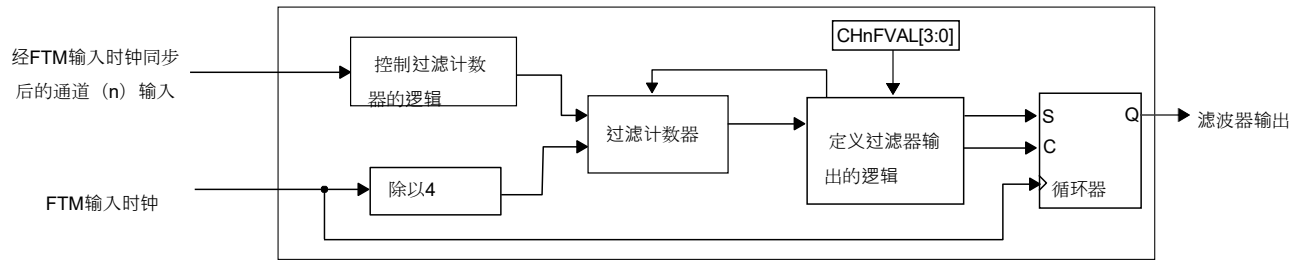



**图42-16.当FLTPS[3:0] = 0时，通道输入滤波器**

如果 FLTPS[3:0] ≠ 0，在被 FTM 输入时钟同步并被 FTM 滤波时钟采样后的通道输入（图 42-15）为滤波输入。

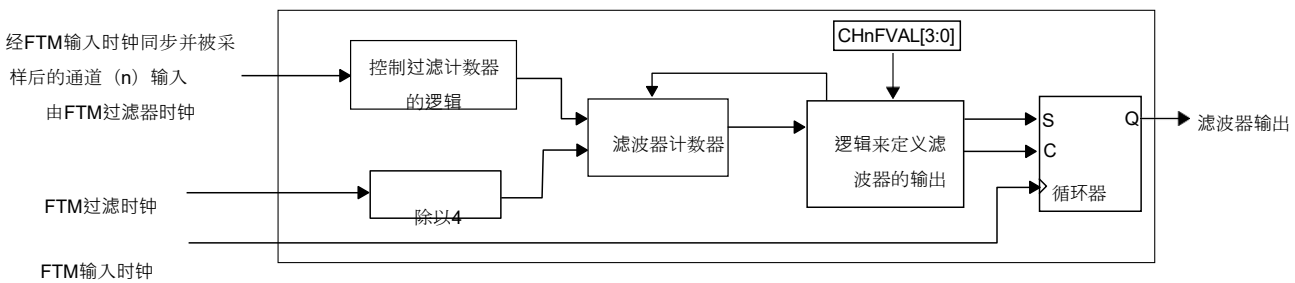



**图42-17.当FLTPS[3:0]≠0时，通道输入滤波器**

## 注意事项

正确检测通道输入的最大频率是FTM滤波器时钟除以4，这是满足信号采样的奈奎斯特标准的要求。

当通道输入的状态发生变化时，计数器被重置并开始向上计数。只要通道输入的新状态是稳定的，计数器继续递增。当计数器等于CHnFVAL[3:0]时，新的通道输入信号值被验证。然后它被作为一个脉冲传送给边缘检测器。

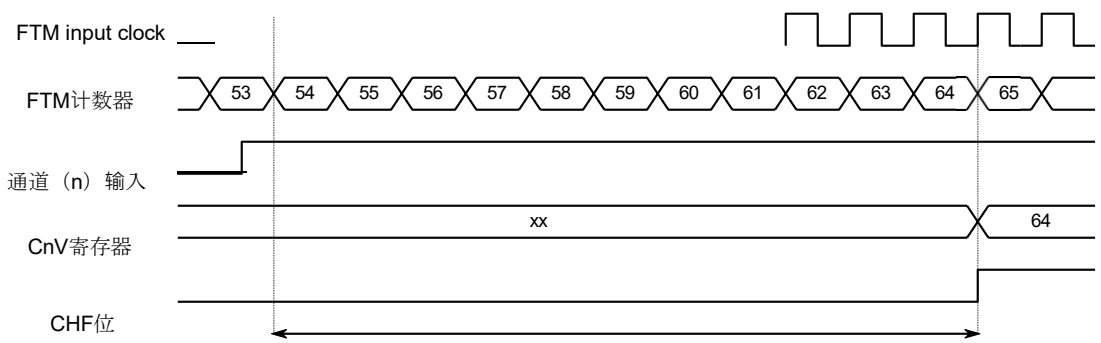
如果在验证之前，通道输入信号上出现相反的边沿，则计数器被重置。在下一个输入转换时，计数器又开始计数。如果一个脉冲的采样值小于（CHnFVAL[3:0]×4）FTM 滤波器时钟的连续上升沿，则被视为突发事件，不传递给边缘检测器。

下表显示了FTM通道输入滤波器根据其配置而增加的延迟。

**表42-7.FTM通道输入滤波延时**

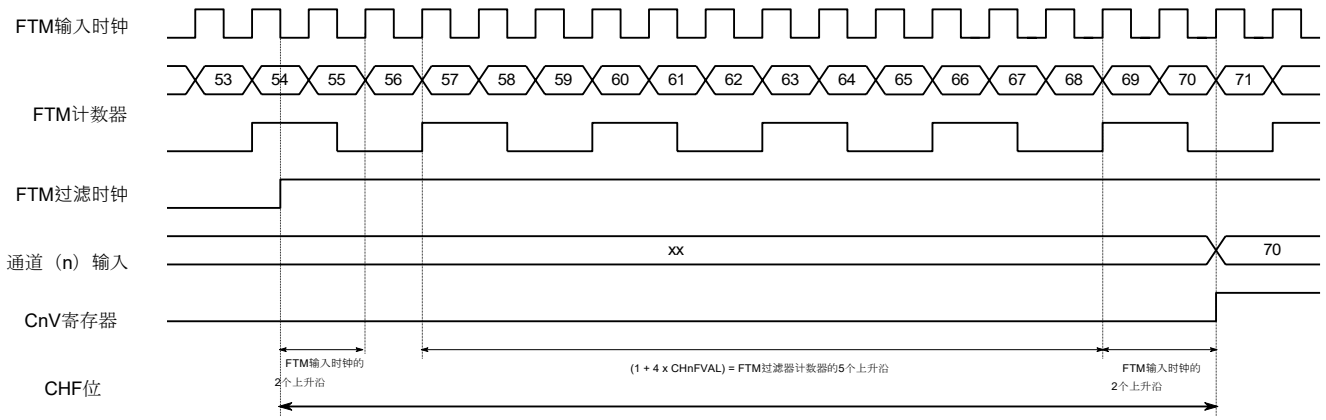
| FTM通道输入滤波器 | FLTPS[3:0]位 | 在通道输入上选择的边沿和设置CHF位之间的上升沿的数量 |
|---|---|---|
| • 通道没有输入滤波器，或<br>• 通道输入滤波器被禁用（CHnFVAL[3:0] = 0）。 | FLTPS[3:0] = 0 | • FTM输入时钟的3个上升沿 |
| | FLTPS[3:0] ≠ 0 | • FTM输入时钟的3个上升沿，加上<br>• 1个FTM滤波器时钟的上升沿 |
| • 通道有输入滤波器，而<br>• 通道输入滤波器被启用（CHnFVAL[3:0]≠0） | FLTPS[3:0] = 0 | • (4 + 4 × CHnFVAL[3:0]) FTM输入时钟的上升沿 |
| | FLTPS[3:0] ≠ 0 | • FTM输入时钟的4个上升沿，加上<br>• (1 + 4 × CHnFVAL[3:0]) FTM滤波器时钟的上升沿 |

下面的数字说明了通道输入滤波器的两个例子。




请注意。
PS[2:0] = 3'b000
通道(n)处于输入捕捉模式，仅在上升沿进行捕捉 CHnFVAL[3:0] = 4'h2 (
通道(n)输入滤波器被启用)

**图42-18.当FLTPS[3:0]=0时，通道输入滤波器的例子**




FTM输入时钟的4个上升沿+（1+4 x CHnFVAL）FTM过滤时钟的上升沿=FTM输入时钟的4个上升沿+FTM过滤时钟的5个上升沿

请注意。
PS[2:0] = 3'b000
通道(n)处于输入捕获模式，仅在上升沿捕获 CHnFVAL[3:0] = 4'h1 (
通道(n)输入滤波器被启用) FLTPS[3:0] = 4'h2 (除以 3)

**图42-19.当FLTPS[3:0]≠0时，通道输入滤波器的例子**

### 42.5.5.2 输入捕捉模式下的FTM计数器复位

如果通道（n）处于输入捕获模式，并且 CnSC[ICRST = 1]，那么当通道（n）输入信号中发生选定的输入捕获事件时，FTM 计数器的当前值被捕获到 CnV 寄存器中，CHF 位被设置，通道（n）中断被产生（如果 CHIE = 1），FTM 计数器被复位到 CNTIN 寄存器的值。

这允许FTM测量应用于通道（n）输入（FTM输入时钟的数量）的周期/脉冲，而不必在事件发生后在软件中实现减法计算。

下图显示了在输入捕捉模式下，在ICRST=1的通道中检测到选定的输入捕捉事件时，FTM计数器复位。

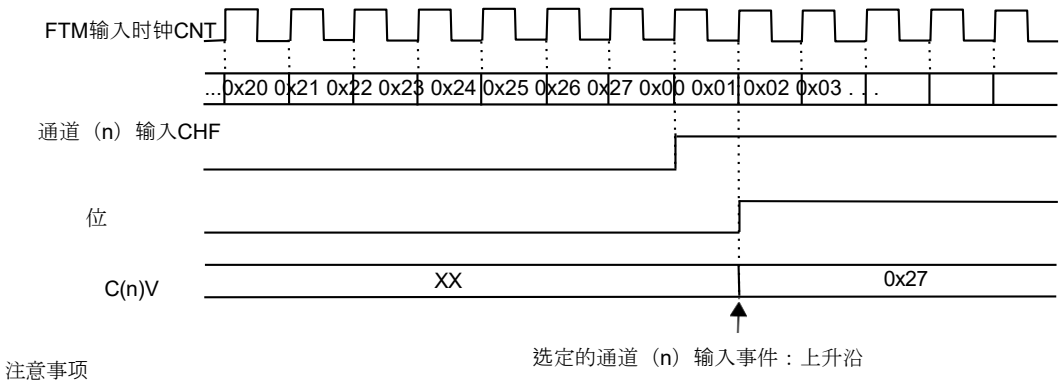



**图42-20.ICRST=1时的输入捕捉模式示例**

### 注意事项

- 预计只有当通道处于输入捕获模式时，ICRST位才会被设置。
- 如果FTM计数器被重置，因为通道处于输入捕获模式，ICRST=1，那么预分频器计数器（Prescaler）也被重置。

## 42.5.6 输出比较模式

当选择输出比较模式时。

- DECAPEN = 0
- MCOMBINE = 0
- COMBINE = 0
- CPWMS = 0, 和
- MSB:MSA = 0:1

在输出比较模式下，FTM可以产生具有可编程位置、极性、持续时间和频率的定时脉冲。当计数器与输出比较通道的CnV寄存器中的值相匹配时，通道（n）的输出可以被设置、清除或切换。

当一个通道最初被配置为Toggle模式时，通道输出的前一个值被保持，直到第一个输出比较事件发生。

如果CHIE = 1，CHF位被设置，通道(n)中断被产生。
(n）匹配（FTM计数器=CnV）。

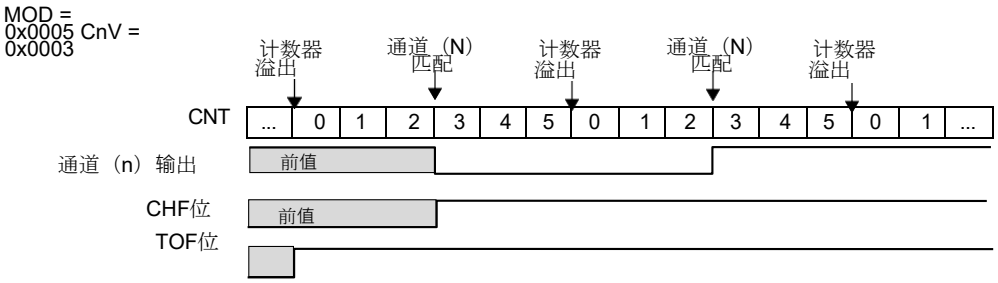



**图42-21.匹配切换通道输出时输出比较模式的例子**

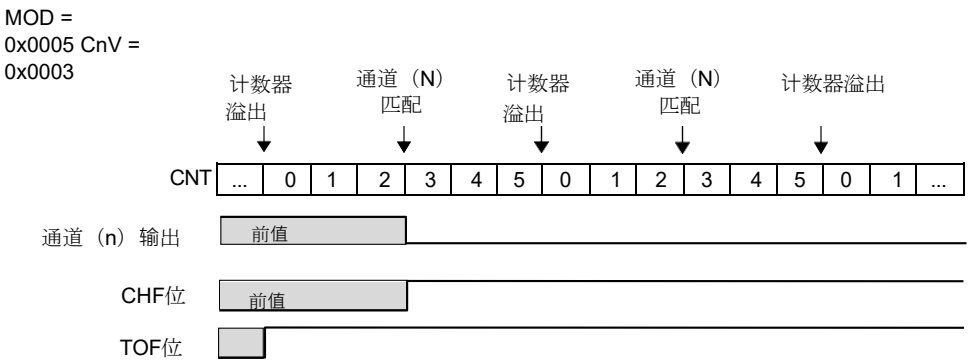



**图42-22.匹配清除通道输出时输出比较模式的例子**

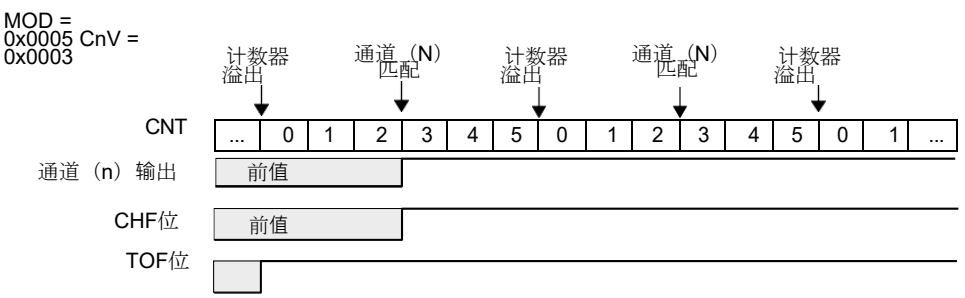



**图42-23.匹配设置通道输出时的输出比较模式示例**

如果(ELSB:ELSA=0:0)当计数器达到CnV寄存器中的值时，CHF位被置位，如果CHIE=1，则产生通道(n)中断，然而通道(n)的输出不被修改，由FTM控制。

## 42.5.7 边缘对齐的PWM（EPWM）模式

边缘对齐模式在以下情况下被选中。

- QUADEN = 0

- DECAPEN = 0
- MCOMBINE = 0
- COMBINE = 0
- CPWMS = 0, 和
- MSB = 1

EPWM的周期由（MOD - CNTIN + 0x0001）决定，脉冲宽度（占空比）由（CnV - CNTIN）决定。

如果CHIE = 1，CHF位被设置，通道(n)中断被产生。
(n）匹配（FTM计数器=CnV），也就是在脉冲宽度的末端。

这种类型的PWM信号被称为边沿对齐，因为所有PWM信号的前缘都与周期的起点对齐，这对FTM内的所有通道都是一样的。

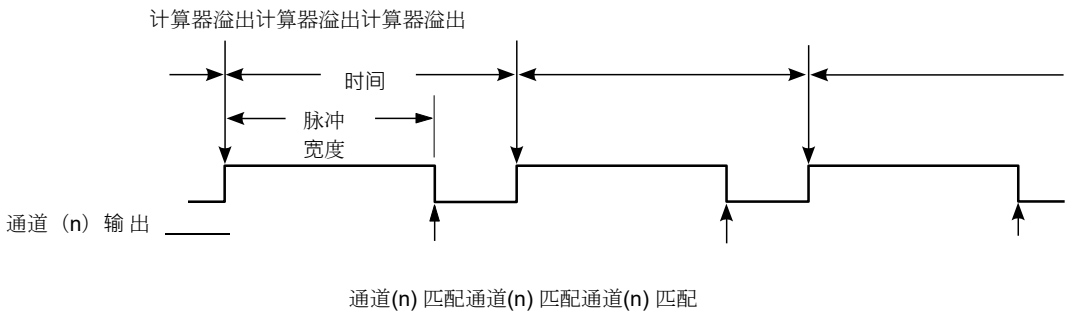



**图42-24.ELSB:ELSA=1:0时的EPWM周期和脉冲宽度**

如果(ELSB:ELSA=0:0)当计数器达到CnV寄存器中的值时，CHF位被置位，如果CHIE=1，则产生通道(n)中断，然而通道(n)输出不受FTM控制。

如果(ELSB:ELSA=1:0)，那么当CNTIN寄存器的值被加载到FTM计数器时，通道(n)输出在计数器溢出时被强制为高电平，而在通道(n)匹配时(FTM计数器=CnV)则被强制为低电平。见下图。

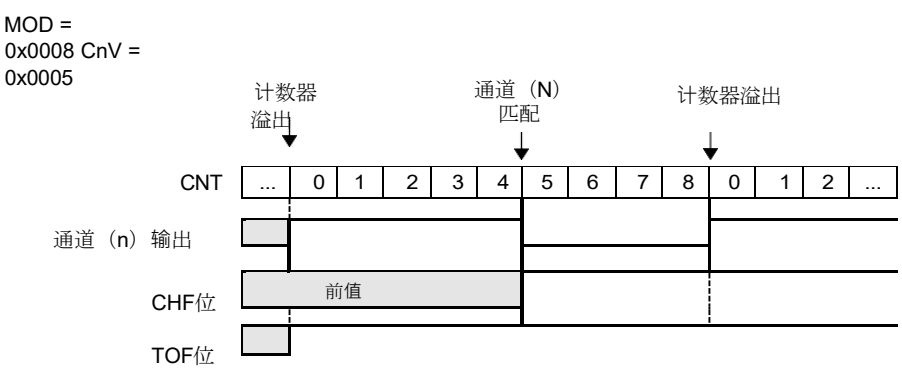



**图42-25.ELSB:ELSA=1:0时的EPWM信号**

如果(ELSB:ELSA = X:1)，那么当CNTIN寄存器的值被载入FTM计数器时，通道(n)输出在计数器溢出时被强制为低电平，而在通道(n)匹配时(FTM计数器=CnV)则被强制为高电平。见下图。

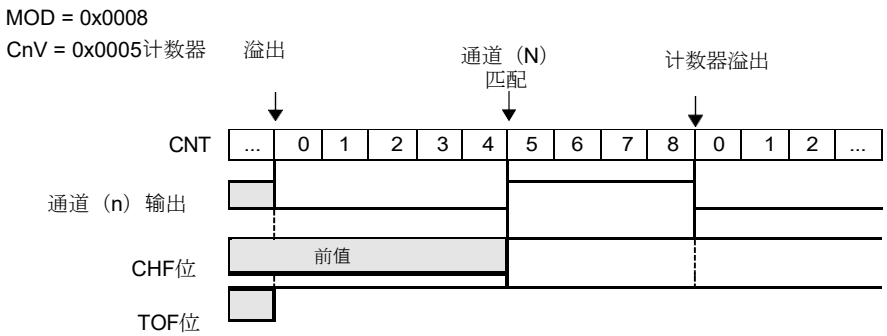



**图42-26.ELSB:ELSA=X:1的EPWM信号**

如果(CnV = 0x0000)，那么通道(n)的输出是一个0%占空比的EPWM信号，即使有通道(n)的匹配，CHF位也不会被设置。

如果(CnV>MOD)，那么通道(n)的输出是一个100%占空比的EPWM信号，CHF位没有被设置。因此，MOD必须小于0xFFFF，才能得到一个100%占空比的EPWM信号。

**注意事项**

当CNTIN与零不同时，可以产生以下EPWM信号。

- 如果CnV=CNTIN，0%EPWM信号。
- 如果CNTIN < CnV <= MOD，EPWM信号在0%和100%之间。
- 当CNTIN>CnV或CnV>MOD时，100%EPWM信号。

## 42.5.8 中心对齐的PWM（CPWM）模式

当选择中心对齐模式时。

- QUADEN = 0
- DECAPEN = 0
- MCOMBINE = 0
- COMBINE = 0，和

- CPWMS = 1

CPWM的脉冲宽度（占空比）由2×（CnV-CNTIN）决定，周期由2×（MOD-CNTIN）决定。见下图。MOD必须保持在0x0001到0x7FFF的范围内，因为超出这个范围的值会产生不明确的结果。

在CPWM模式下，FTM计数器向上计数直到达到MOD，然后向下计数直到达到CNTIN。

当 FTM 计数下降（脉宽开始时）和 FTM 计数上升（脉宽结束时）时，CHF 位被设置，通道（n）中断被产生（如果 CHIE = 1），在通道（n）匹配（FTM 计数 = CnV）。

这种类型的PWM信号被称为中心对齐，因为所有通道的脉冲宽度中心都与CNTIN的值对齐。

其他通道模式与上-下计数器（CPWMS = 1）不兼容。因此，当(CPWMS = 1)时，所有FTM通道必须在CPWM模式下使用。

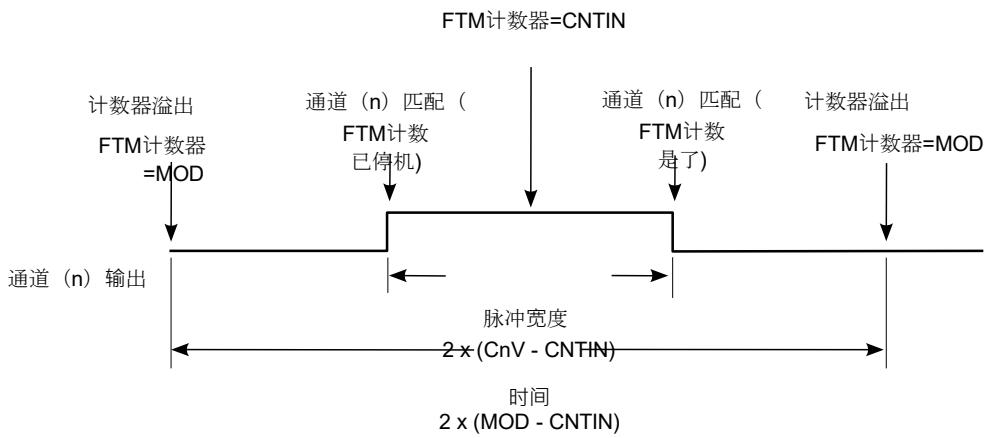



**图42-27.ELSB:ELSA=1:0时的CPWM周期和脉冲宽度**

如果(ELSB:ELSA=0:0)当FTM计数器达到CnV寄存器中的值时，CHF位被设置，通道(n)中断被产生(如果CHIE=1)，然而通道(n)的输出不受FTM控制。

如果(ELSB:ELSA = 1:0)，那么在通道(n)匹配处(FTM 计数器 = CnV)，在倒数时，通道(n)输出被强制为高电平，在倒数时，通道(n)匹配处被强制为低电平。见下图。

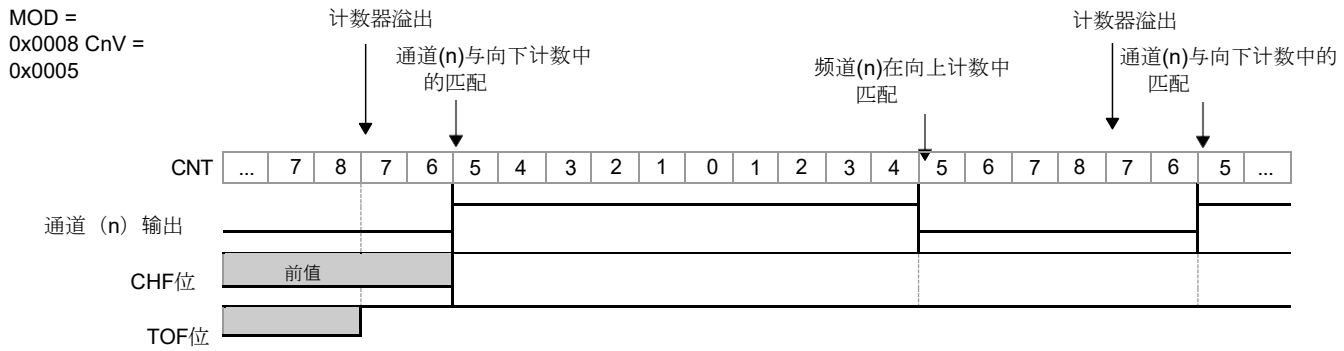



**图42-28.ELSB:ELSA=1:0的CPWM信号**

如果(ELSB:ELSA = X:1)，那么在通道(n)匹配处(FTM 计数器 = CnV)，在倒数时，通道(n)输出被强制为低电平，在倒数时，通道(n)匹配处被强制为高电平。见下图。

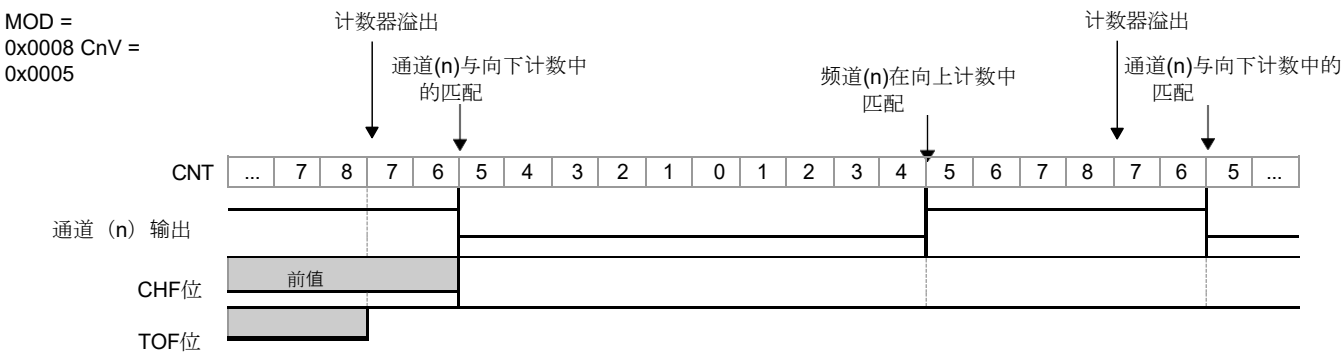



**图42-29.ELSB:ELSA=X:1的CPWM信号**

如果(CnV=0x0000)或CnV为负值，即(CnV[15]=1)，则通道(n)输出为0%占空比的CPWM信号，即使有通道(n)匹配，CHF位也不会被设置。

如果 CnV 是正值，即（CnV[15] = 0），（CnV ≥ MOD），且（MOD ≠ 0x0000），那么通道（n）的输出是 100% 占空比的 CPWM 信号，即使有通道（n）的匹配，CHF 位也不会被设置。这意味着MOD设置的可用周期范围是0x0001到0x7FFE，如果你不需要产生100%占空比的CPWM信号，则0x7FFF。这不是一个重要的限制，因为产生的周期要比正常应用所需的长得多。

当FTM计数器是一个自由运行的计数器时，不得使用CPWM模式。

## 42.5.9 合并模式

合并模式在以下情况下被选中。

QUADEN = 0

- DECAPEN = 0
- MCOMBINE = 0
- COMBINE = 1, 和
- CPWMS = 0

在组合模式下，一个偶数通道（n）和相邻的奇数通道（n+1）被组合起来，在通道（n）输出中产生一个PWM信号。

在组合模式下，PWM周期由（MOD - CNTIN + 0x0001）决定，PWM脉冲宽度（占空比）由（|C（n+1）V - C（n）V|）决定。

如果通道(n)CHIE=1，在通道(n)匹配时(FTM计数器=C(n)V)，则通道(n)CHF位被设置，其中断被产生。如果通道(n+1)CHIE=1，通道(n+1)匹配时(FTM计数器=C(n+1)V)，通道(n+1)CHF位被设置，其中断被生成。

如果通道(n)ELSB:ELSA=1:0，那么通道(n)的输出在周期开始时(FTM计数器=CNTIN)和通道(n+1)匹配时(FTM计数器=C(n+1)V)被强制为低。在通道(n)匹配时(FTM计数器=C(n)V)，它被强制为高电平。见下图。

如果通道(n)ELSB:ELSA=X:1，那么通道(n)的输出在周期开始时(FTM计数器=CNTIN)和通道(n+1)匹配时(FTM计数器=C(n+1)V)被强制为高。在通道(n)匹配时(FTM计数器=C(n)V)，它被强制为低。见下图。

在组合模式下，通道(n+1)的 ELSB:ELSA 位不用于生成通道(n)和(n+1)输出。但是，如果通道(n)ELSB:ELSA=0:0，则通道(n)输出不受FTM控制，如果通道(n+1)ELSB:ELSA=0:0，则通道(n+1)输出不受FTM控制。

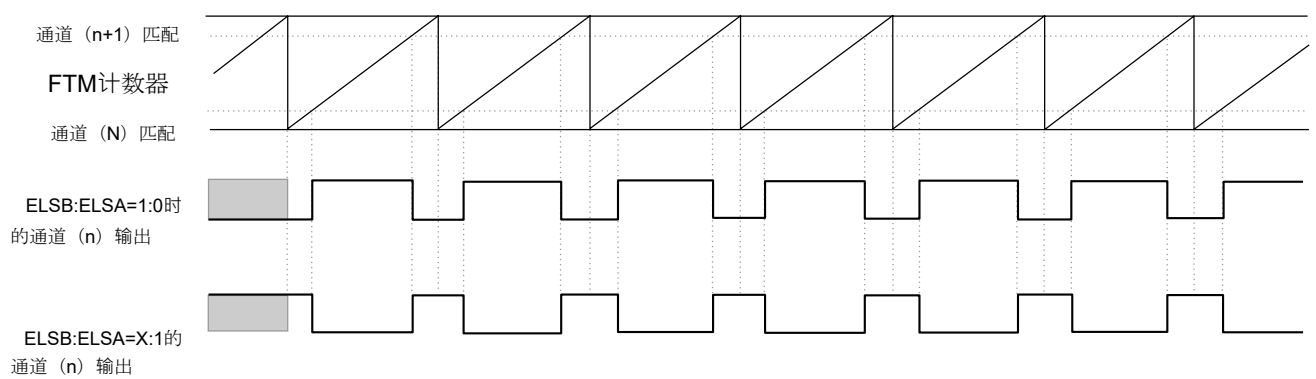



**图42-30.合并模式**

下面的数字说明了使用组合模式产生的PWM信号。

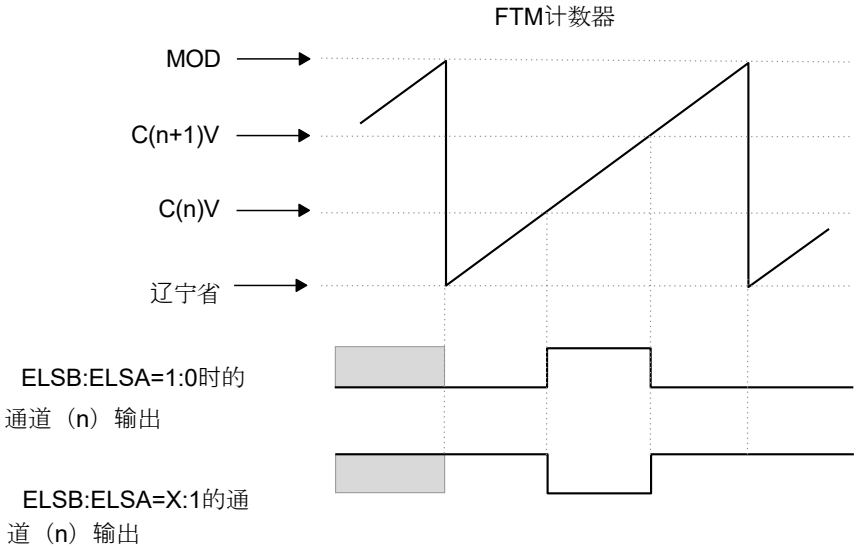



**图42-31.如果（CNTIN < C(n)V < MOD）和（CNTIN < C(n+1)V < MOD）以及（C(n)V < C(n+1)V），通道（n）输出**

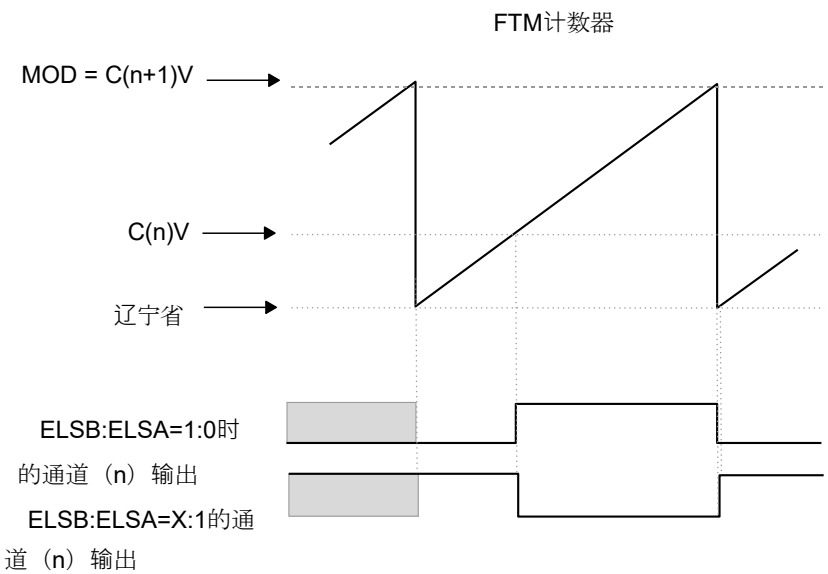
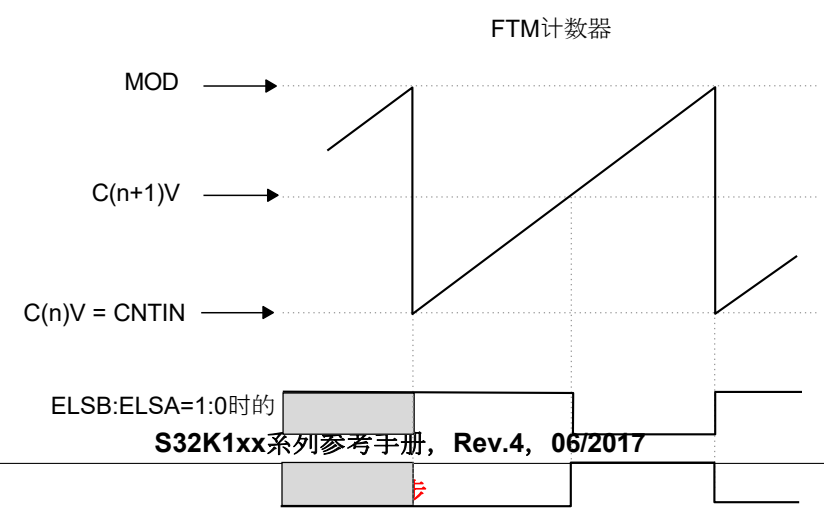



**图42-32.如果（CNTIN < C(n)V < MOD）和（C(n+1)V = MOD），通道（n）输出**



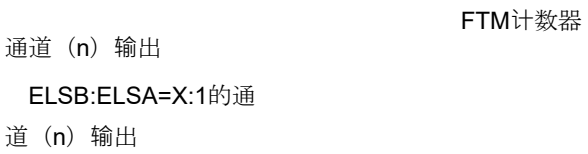



**图42-33.如果（C(n)V = CNTIN）和（CNTIN < C(n+1)V < MOD），通道（n）输出**

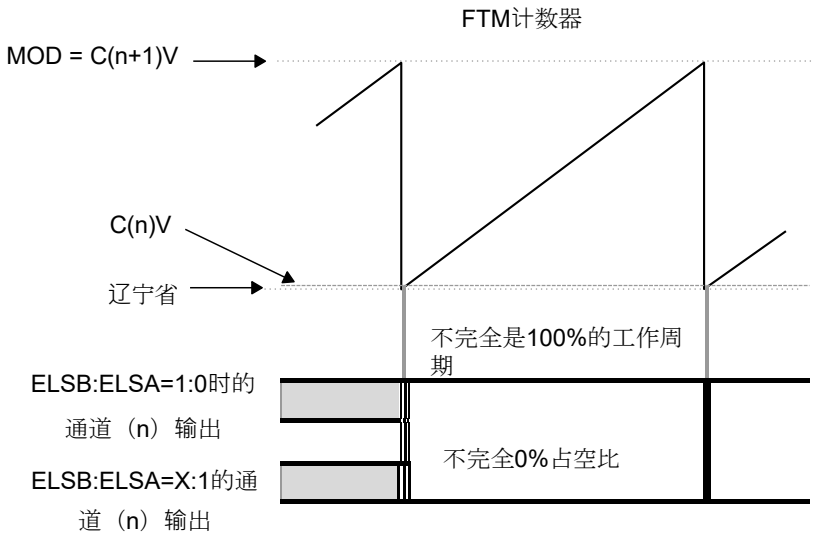



**图42-34.如果（CNTIN < C(n)V < MOD）和（C(n)V几乎等于CNTIN）和（C(n+1)V = MOD），通道（n）输出**

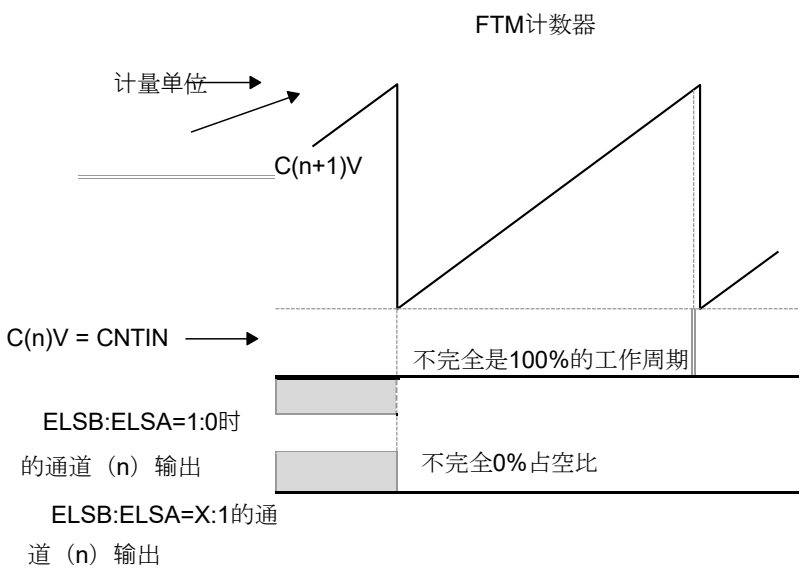



**图42-35.如果（C(n)V = CNTIN）和（CNTIN < C(n+1)V < MOD）以及（C(n+1)V几乎等于MOD），通道（n）输出**

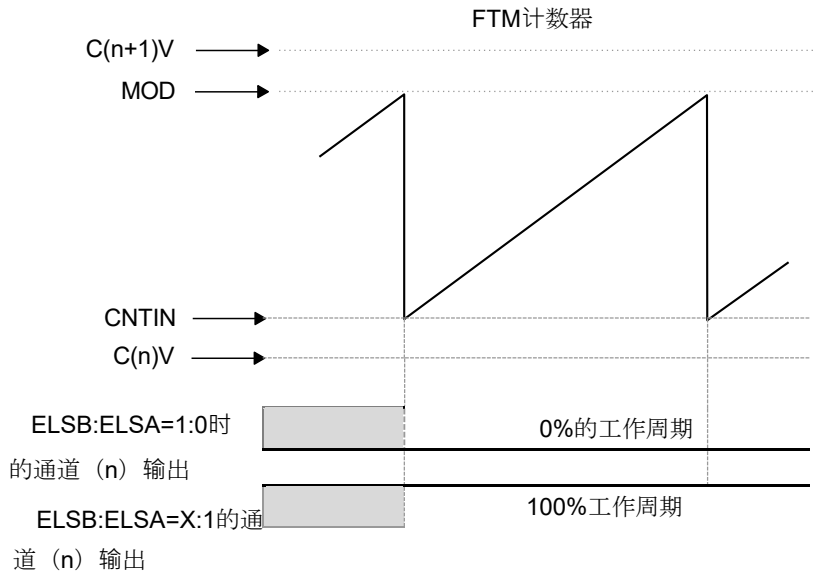



**图42-36.如果C(n)V和C(n+1)V不在CNTIN和MOD之间，通道（n）输出**

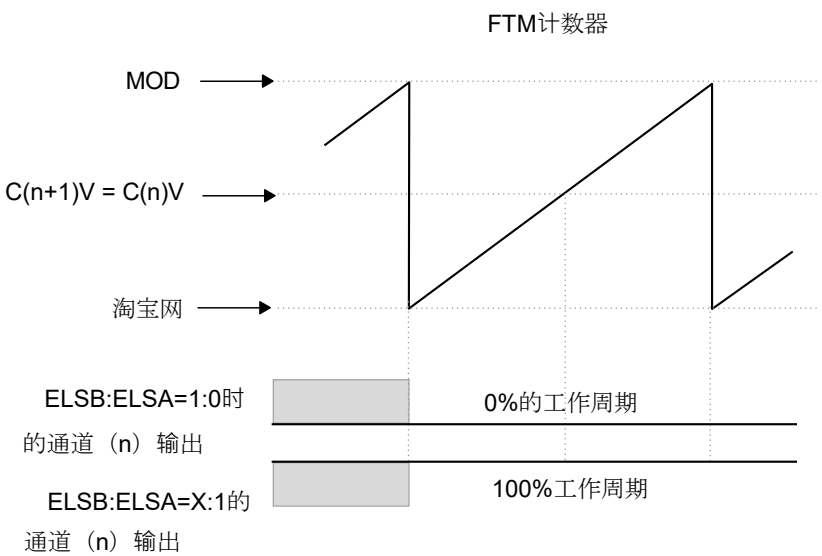



**图42-37.如果（CNTIN < C(n)V < MOD）和（CNTIN < C(n+1)V < MOD）以及（C(n)V = C(n+1)V），通道（n）输出**

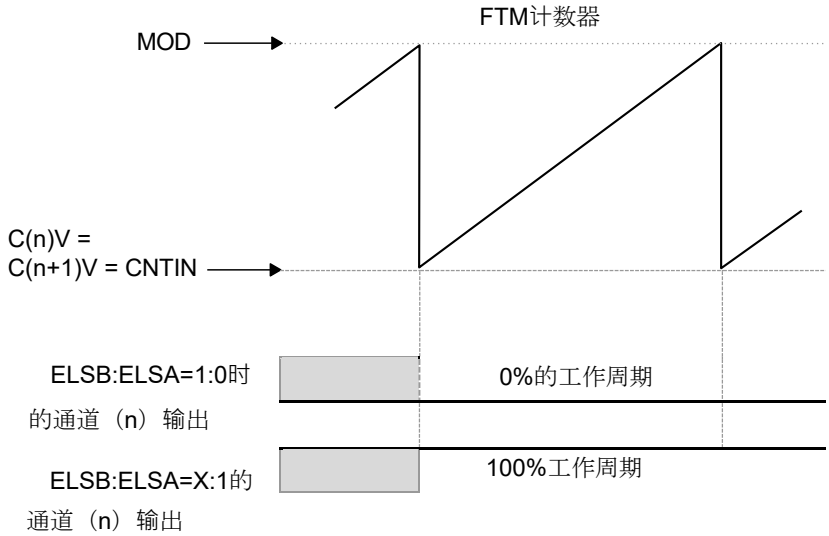



**图42-38.如果（C(n)V = C(n+1)V = CNTIN），通道（n）输出**

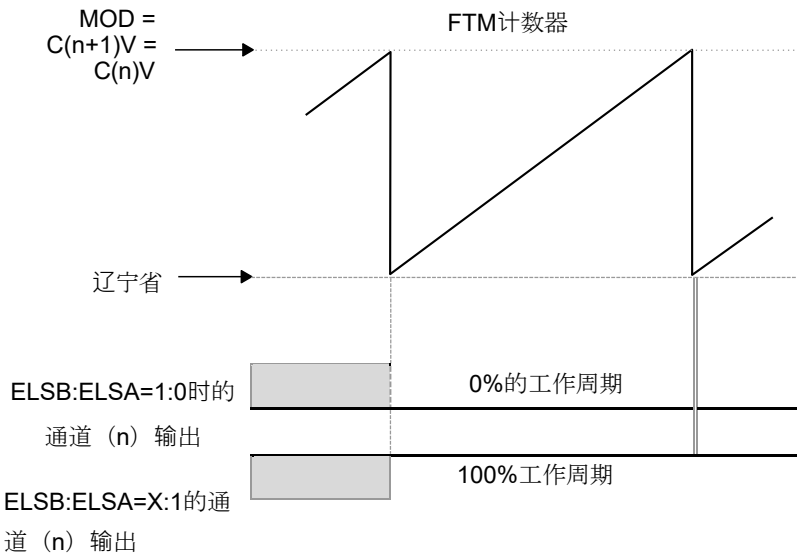
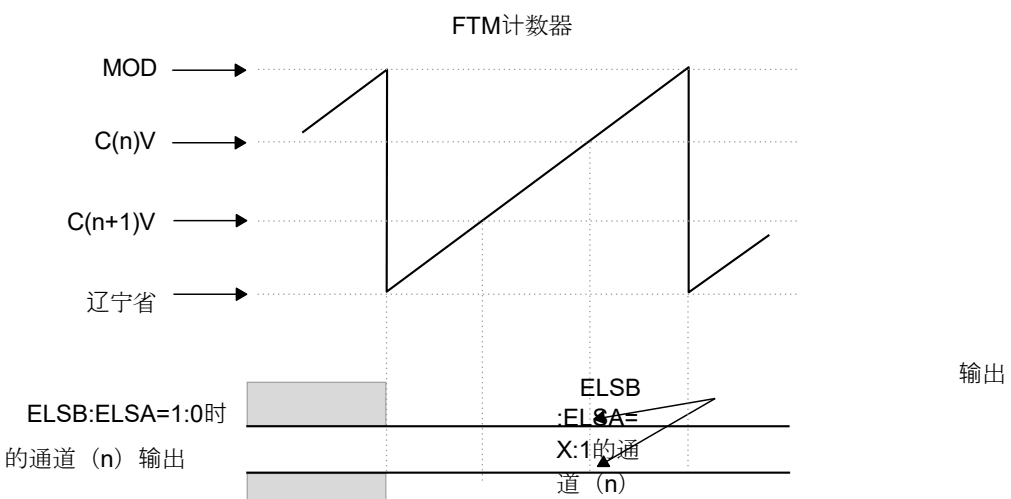



**图42-39.通道（n）输出，如果（C（n）V = C（n+1）V = MOD）**



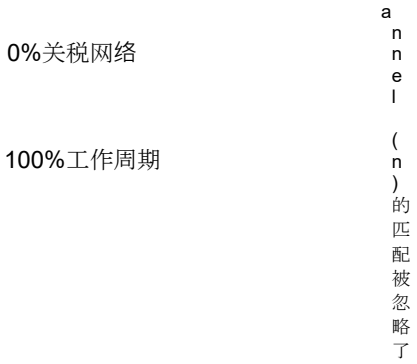



**图42-40.如果（CNTIN < C(n)V < MOD）和（CNTIN < C(n+1)V < MOD）和（C(n)V > C(n+1)V），通道（n）输出**

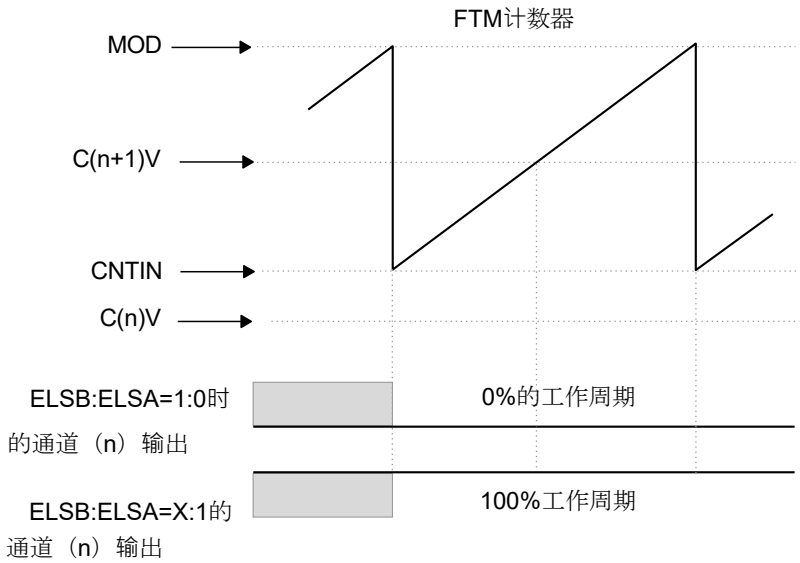



**图42-41.如果(C(n)V < CNTIN)和(CNTIN < C(n+1)V < MOD) 通道(n)输出**

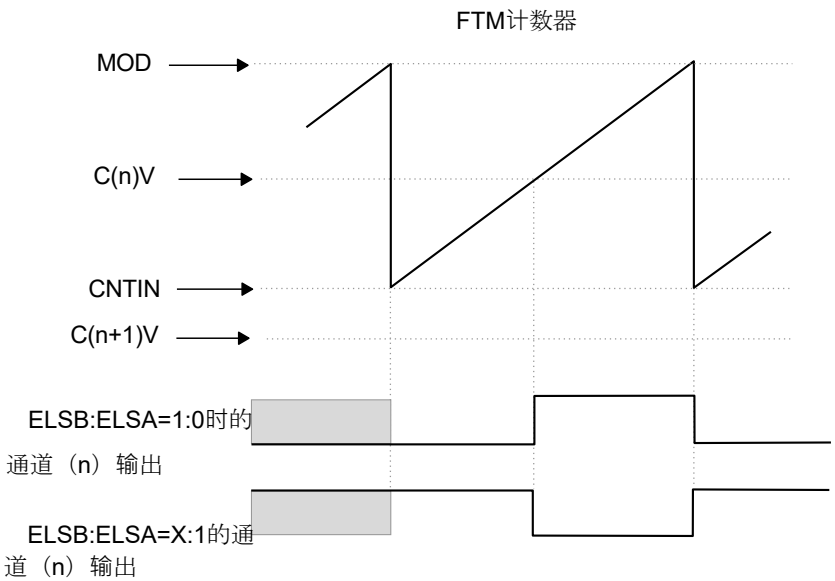



**图42-42.如果（C(n+1)V < CNTIN）和（CNTIN < C(n)V < MOD），通道（n）输出**

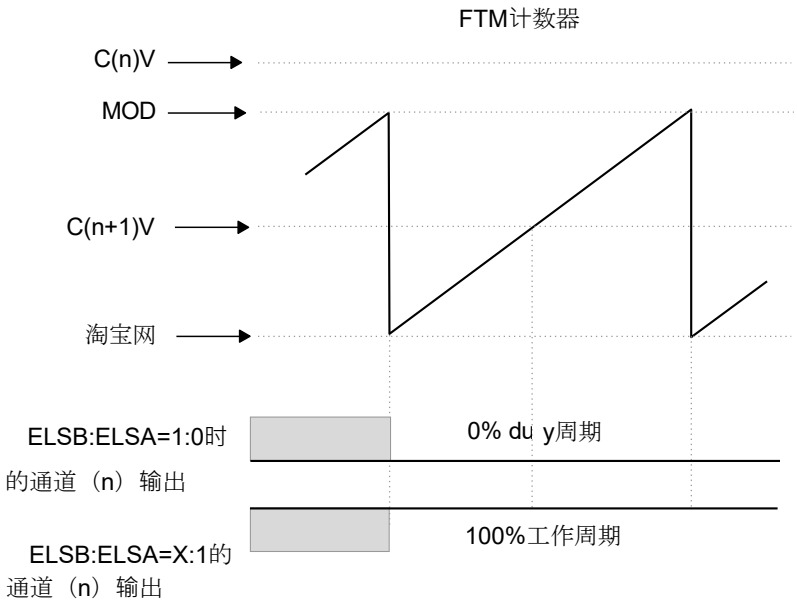



**图42-43.如果(C(n)V > MOD)和(CNTIN < C(n+1)V < MOD)，通道(n)输出**

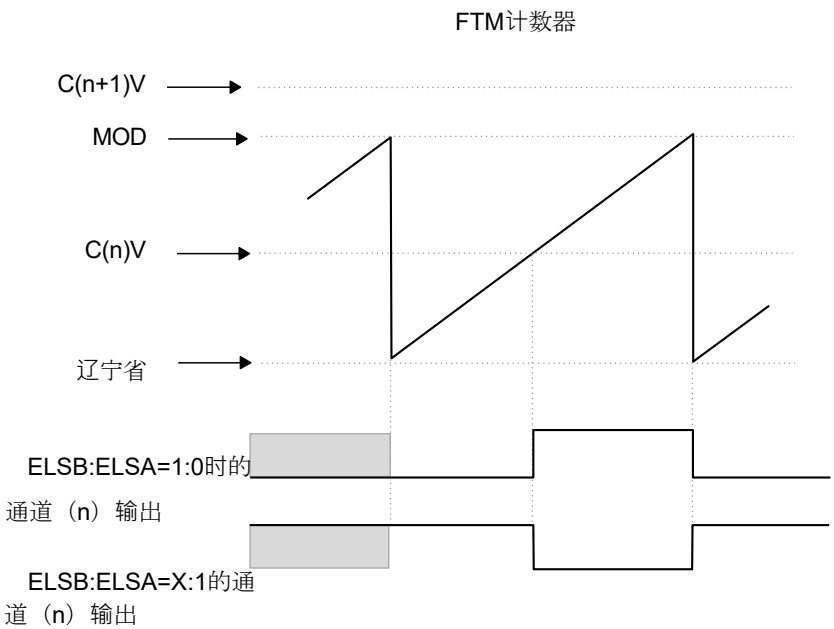



**图42-44.如果(C(n+1)V > MOD)和(CNTIN < C(n)V < MOD)，通道(n)输出**

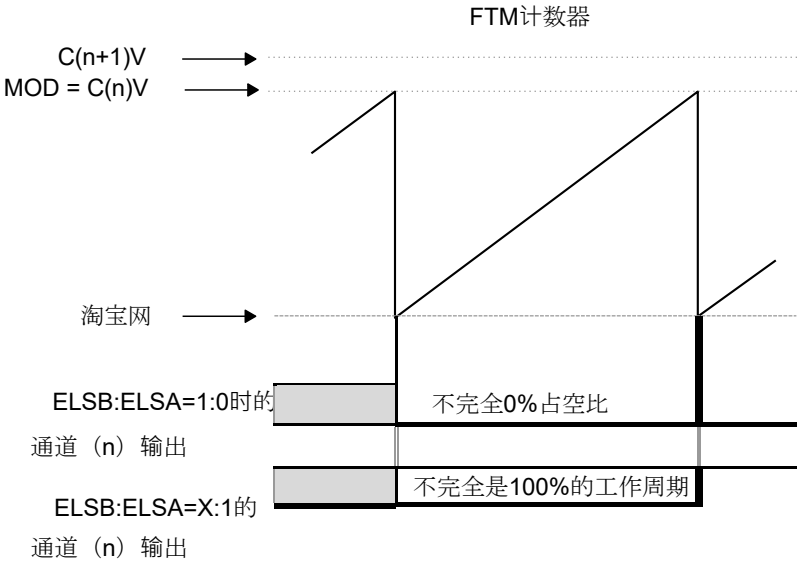



**图42-45.如果(C(n+1)V > MOD)和(CNTIN < C(n)V = MOD)，通道(n)输出**

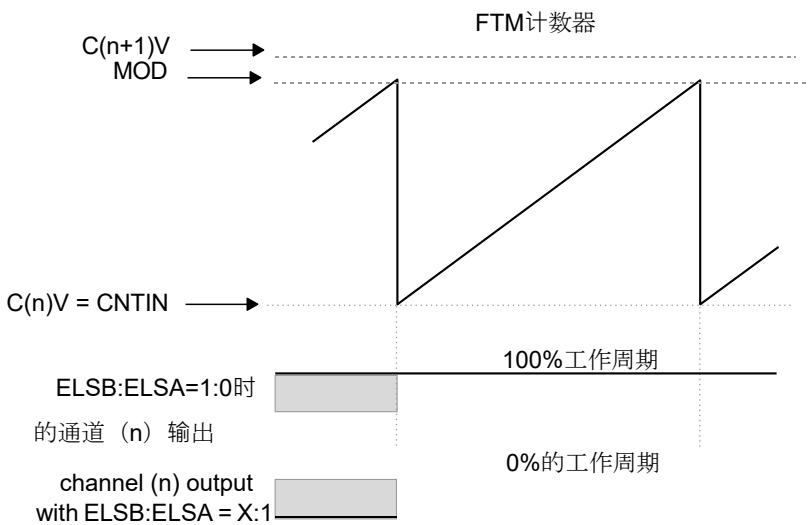



**图42-46.如果(C(n)V = CNTIN)和(C(n+1)V > MOD)，通道(n)输出**

### 42.5.9.1 不对称的PWM

在组合模式和修改后的组合PWM模式中，PWM第一边缘（通道（n）匹配。FTM计数器=C(n)V）与PWM第二边沿（通道（n+1）匹配）无关。FTM计数器=C(n+1)V）。

## 42.5.10 修改后的组合式PWM模式

在以下情况下，选择修改后的组合式PWM模式。

FTM计数器

- QUADEN = 0
- DECAPEN = 0

- MCOMBINE = 1
- COMBINE = 1, 和
- CPWMS = 0

修改后的组合PWM模式是为了支持生成PWM信号，在生成信号时，周期不被修改，但占空比将被改变。在这种模式下，一个偶数通道（n）和相邻的奇数通道（n+1）被结合起来，在通道（n）的输出中产生一个PWM信号。因此，通道（n）的匹配边缘是固定的，通道（n+1）的匹配边缘可以变化。

当一对通道处于修正的组合PWM模式时，建议其他通道也处于修正的组合PWM模式。

在修改后的组合式PWM模式下，假设CNTIN≥0，MOD>0，且CNTIN < MOD:

- PWM周期是由（MOD - CNTIN + 0x0001）决定的。
- 通道（n）的PWM占空比根据下表计算。

**表42-8.修改后的组合式PWM模式 - 占空比**

| 通道（n）PWM占空比 | 状况 |
|---|---|
| 0%的工作周期 | 对于CNTIN≤（C(n)V和C(n+1)V）≤MOD：C(n)V=C(n+1)V |
| 0%和100%之间的工作周期 | 对于CNTIN≤（C（n）V和C（n+1）V）≤MOD。<br>• 如果（C(n)V < C(n+1)V），则占空比为（C(n+1)V - C(n)V）。<br>• 如果（C(n)V>C(n+1)V），则占空比为[（MOD-C(n)V）+（C(n+1)V-CNTIN）+1]。 |
| 100%的工作周期 | CNTIN ≤ C(n)V ≤ MOD 且 C(n+1)V > MOD |

如果通道(n)CHIE=1，在通道(n)匹配时(FTM计数器=C(n)V)，则通道(n)CHF位被设置，其中断被产生。如果通道(n+1)CHIE=1，通道(n+1)匹配时(FTM计数器=C(n+1)V)，通道(n+1)CHF位被设置，其中断被生成。

如果通道(n)ELSB:ELSA=1:0，那么通道(n)输出在通道(n)匹配时被强制为高电平（FTM计数器=C(n)V），在通道(n+1)匹配时被强制为低电平（FTM计数器=C(n+1)V）。

如果通道(n)ELSB:ELSA=X:1，那么通道(n)输出在通道(n)匹配时被强制为低电平(FTM计数器=C(n)V)，在通道(n+1)匹配时被强制为高电平(FTM计数器=C(n+1)V)。

在 Modified Combine PWM 模式下，通道(n+1)的 ELSB:ELSA 位在生成通道(n)和(n+1)输出时不被使用。然而，如果通道(n)ELSB:ELSA = 0:0，则通道(n)的输出不受FTM控制，如果通道(n+1)ELSB:ELSA = 0:0，则通道(n+1)的输出不受FTM控制。

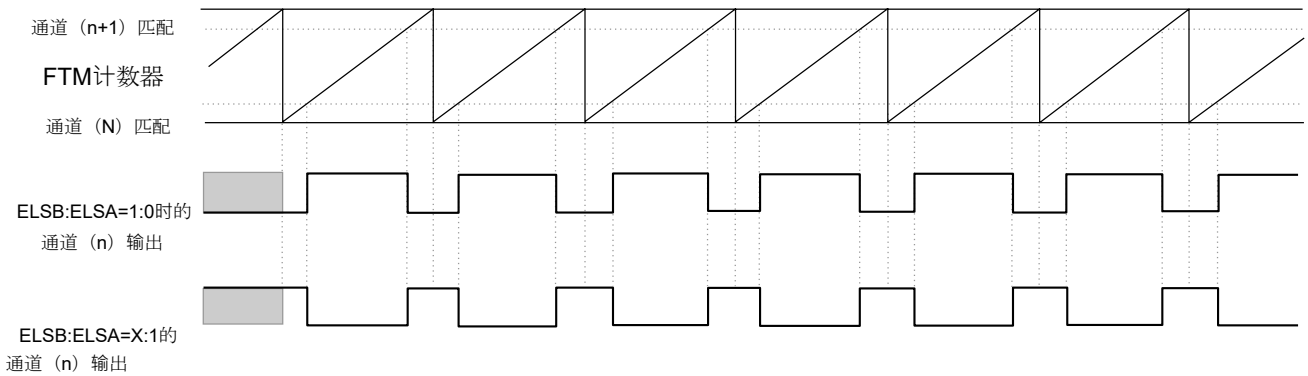



**图42-47.修改后的组合式PWM模式**

修改后的组合式PWM模式允许占空比的增加，因此，在某些情况下，C(n+1)V的匹配可以在下一个FTM计数器周期发生。对于CNTIN ≥0，MOD>0，CNTIN<MOD，这种情况发生在C(n)V>C(n+1)V时。

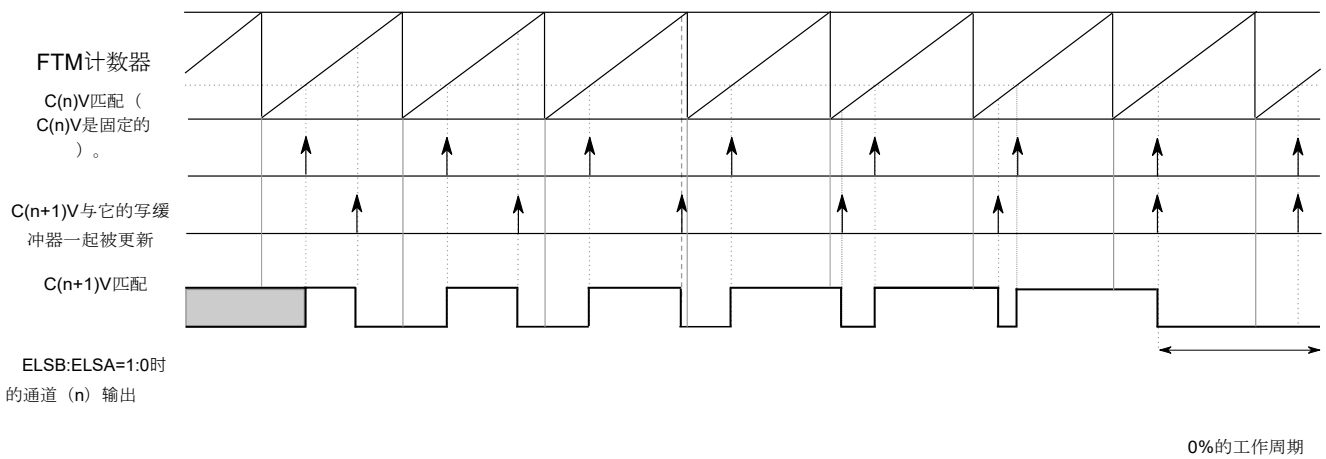



**图42-48.修改后的组合式PWM模式实例**

如果多于一对的通道被配置在修改后的组合PWM模式中，就有可能为每对通道（n）的匹配边缘固定一个相对于其他对的偏移。这种行为在生成照明PWM控制信号时很有用，因为在这种情况下，边缘不与其他对重合，有助于消除噪声的产生。C(n)V寄存器的值是PWM脉冲相对于FTM计数器周期（FTM计数器=CNTIN）开始时的位移。

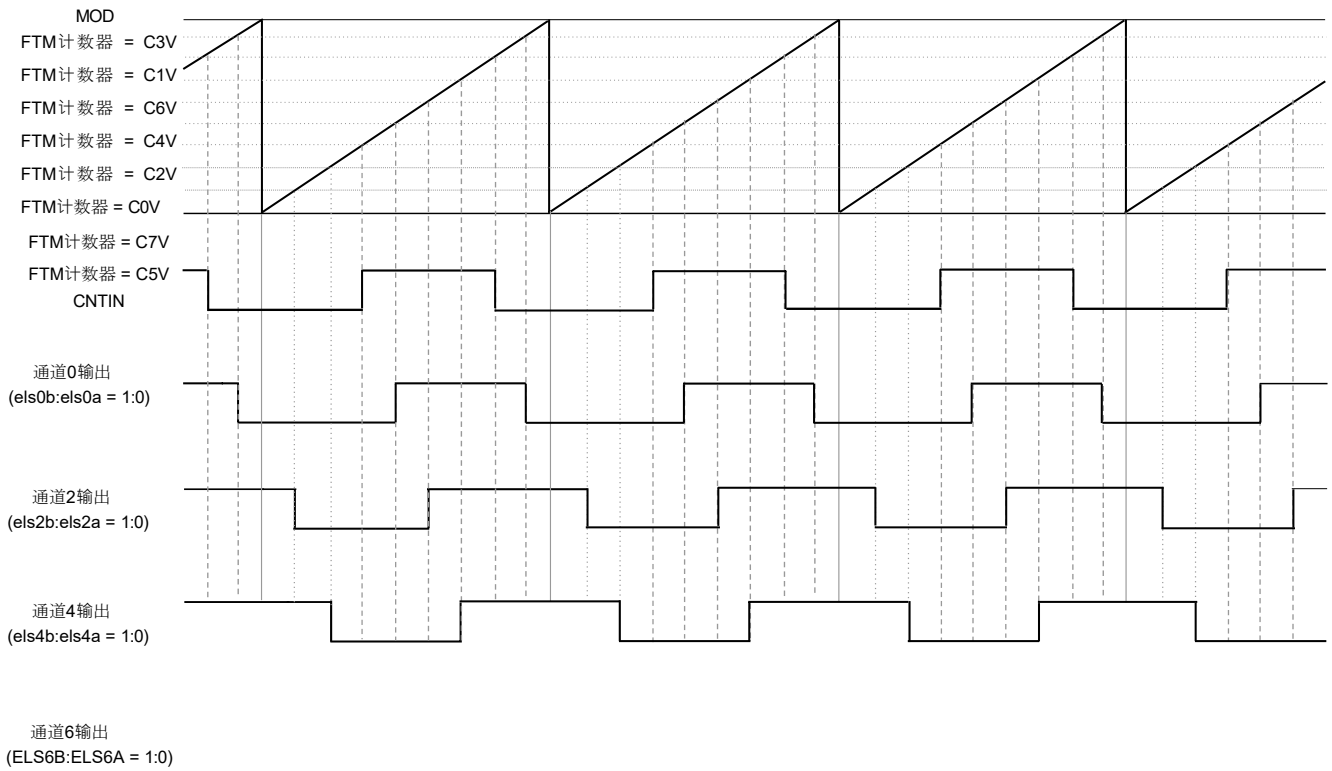



**图42-49.修改后的组合式PWM模式下的四对通道的例子**

### 42.5.10.1 同步性

在修改后的组合模式中，当FTM计数器时钟被禁用（CLKS[1:0] = 0:0）时，应更新下列寄存器。

- CNTIN (CNTIN寄存器更新)
- MOD（MOD和HCR寄存器更新）。
- C(n)V和C(n+1)V（CnV寄存器更新）。

在修改后的组合模式中，如果（FTMEN=1），（CLKS[1:0]≠0:0），并且有一个写到寄存器C(n+1)V的数据，那么寄存器C(n+1)V会在下一个通道（n）的匹配（FTM计数器=C(n)V）上用其写缓冲值进行更新。这个功能允许在这种模式下改变PWM占空比值。

**注意事项**

在修改后的合并模式中，通道(n)和(n+1)的SYNCEN(n)位应该是零位。所以，以下功能对这种模式是不可用的。

- C（n）V和C（n+1）V寄存器同步
- 重新加载点数
- 全球负载

## 42.5.11 互补模式

当选择互补模式时。

- QUADEN = 0
- DECAPEN = 0
- COMP = 1

在互补模式下，通道（n+1）的输出是通道（n）输出的反值。

所以，通道（n+1）的输出与通道（n）的输出相同，当。

- QUADEN = 0
- DECAPEN = 0
- COMP = 0

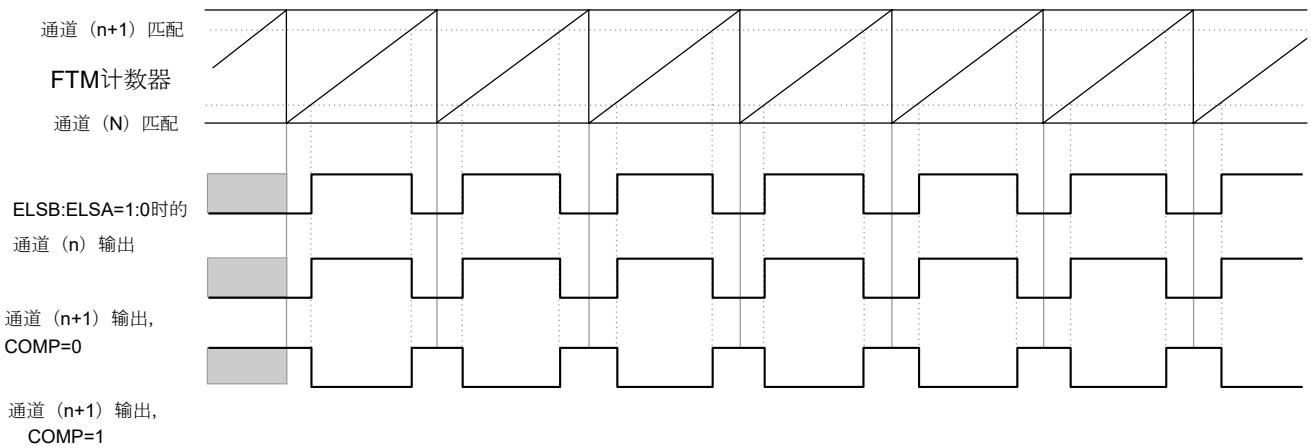



**图42-50.通道(n+1)在互补模式下的输出，(ELSB:ELSA = 1:0)**

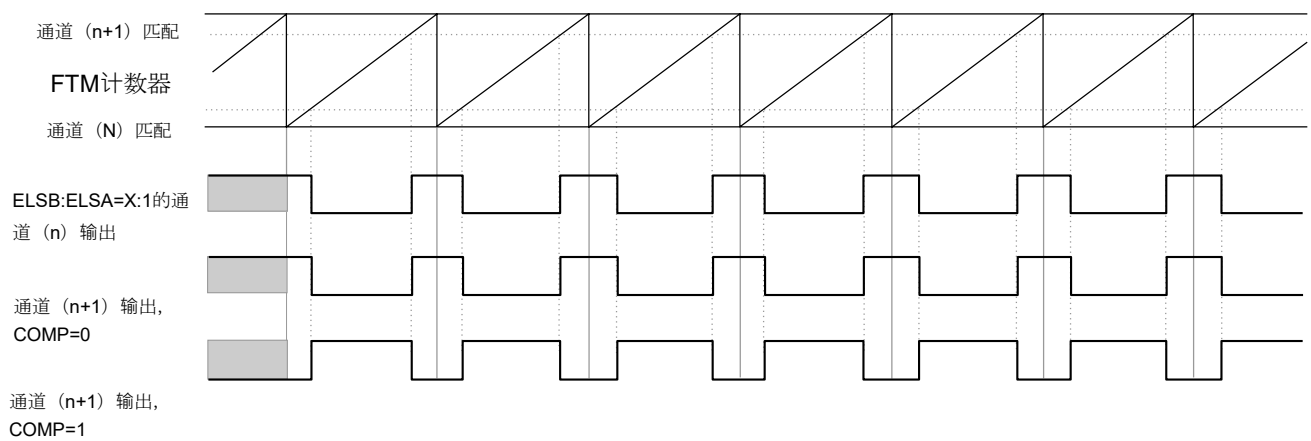



**图42-51.通道(n+1)输出在互补模式下，(ELSB:ELSA = X:1)**

### 注意事项

在输出比较模式下，互补模式是不可用的。

## 42.5.12 从写缓冲区更新的寄存器

### 42.5.12.1 更新CNTIN寄存器

下表描述了CNTIN寄存器被更新的时间。

**表42-9.CNTIN寄存器更新**

| 当 | 然后CNTIN寄存器被更新 |
|---|---|
| clks[1:0] = 0:0 | 当CNTIN寄存器被写入时，与FTMEN位无关。 |
| • FTMEN = 0，或<br>• CNTINC = 0 | 在CNTIN被写入后的下一个FTM输入时钟。 |
| • FTMEN=1。<br>• SYNCMODE = 1, 和<br>• CNTINC = 1 | 由CNTIN寄存器同步。 |
| • CNTINC = 1, 和<br>• LDOK = 1 | 由重装点。 |

### 42.5.12.2 MOD和HCR寄存器的更新

下表描述了 MOD 或 HCR 寄存器的更新时间。

**表42-10.MOD和HCR的更新**

| 当 | 然后MOD或HCR被更新 |
|---|---|
| clks[1:0] = 0:0 | 当MOD（或HCR）被写入时，与FTMEN位无关。 |
| • CLKS[1:0]≠0:0，并且<br>• FTMEN = 0 | 根据CPWMS位，也就是。<br>• 如果选择的模式不是CPWM，那么在MOD（或HCR）寄存器被写入后，MOD（或HCR）被更新，FTM计数器从MOD变为CNTIN。如果 FTM 计数器处于自由运行计数模式，那么当 FTM 计数器从 0xFFFF 变为 0x0000 时，该更新将发生。<br>• 如果选择的模式是CPWM，那么在MOD（或HCR）寄存器被写入后，MOD（或HCR）寄存器被更新，FTM计数器从MOD变为（MOD - 0x0001）。 |
| • CLKS[1:0]≠0:0，并且<br>• FTMEN = 1 | 通过MOD寄存器的同步。HCR在这种情况下遵循MOD寄存器的相同程序。 |
| • LDOK = 1 | 由重装点。 |

### 42.5.12.3 CnV寄存器更新

下表描述了CnV寄存器的更新时间。

表42-11.CnV寄存器更新

| 当 | 然后CnV寄存器被更新 |
| --- | --- |
| clks[1:0] = 0:0 | 当CnV寄存器被写入时，与FTMEN位无关。 |
| • CLKS[1:0]≠0:0，并且<br>• FTMEN = 0 | 根据选定的模式，也就是。<br>• 如果选择的模式是输出比较，那么在CnV寄存器被写入后的下一个FTM计数器变化、预分频器计数结束时，CnV寄存器被更新。<br>• 如果选择的模式是 EPWM，那么在 CnV 寄存器被写入后，CnV 寄存器被更新，FTM 计数器从 MOD 变为 CNTIN。如果 FTM 计数器处于自由运行计数模式，那么当 FTM 计数器从 0xFFFF 变为 0x0000 时，该更新将发生。<br>• 如果选择的模式是 CPWM，那么在 CnV 寄存器被写入后，CnV 寄存器被更新，FTM 计数器从 MOD 变为 (MOD - 0x0001)。 |
| • CLKS[1:0]≠0:0，并且<br>• FTMEN = 1 | 根据选定的模式，也就是。<br>• 如果选择的模式是输出比较，那么 CnV 寄存器将根据 SYNCEN 位进行更新。如果(SYNCEN = 0)，则CnV 寄存器在FTM 计数器的下一次变化，即预分频器计数结束时，被写入CnV 寄存器后更新。如果（SYNCEN = 1），则CnV寄存器由C(n)V和C(n+1)V寄存器同步更新。<br>• 如果选择的模式不是输出比较和（SYNCEN=1），那么CnV寄存器就会被C(n)V和C(n+1)V寄存器同步更新。 |
| • SYNCEN = 1, 和<br>• LDOK = 1 | 由重装点。 |

## 42.5.13 PWM同步

PWM同步提供了一个机会，用它们的缓冲值更新MOD、HCR、CNTIN、CnV、OUTMASK、INVCTRL和SWOCTRL寄存器，并将FTM计数器强制到CNTIN寄存器值。

### 注意事项

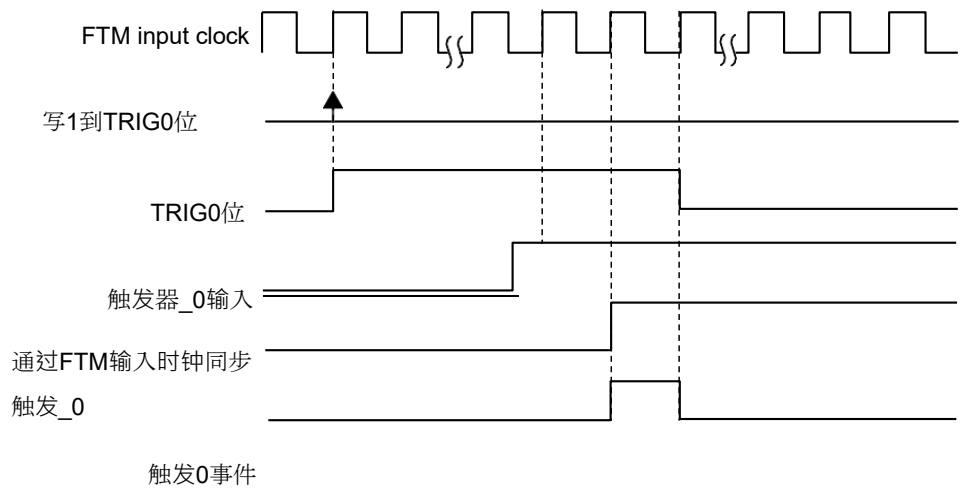
传统的PWM同步（SYNCMODE = 0）是增强型PWM同步（SYNCMODE = 1）的一个子集。因此，只有增强型PWM同步必须被使用。

### 42.5.13.1 硬件触发器

当TRIGn=1时，FTM模块的三个硬件触发信号输入被启用，其中n=0、1或2分别对应于每个输入信号。硬件触发器输入n由FTM输入时钟同步。当在启用的硬件触发输入端检测到一个上升沿时，就开始与硬件触发器进行PWM同步。

如果（HWTRIGMODE = 0），那么当0被写入或检测到触发器n事件时，TRIGn位被清零。

在这种情况下，如果启用了两个或多个硬件触发器（例如，TRIG0和TRIG1=1），并且只发生了触发器1事件，那么只有TRIG1位被清零。如果触发器n事件与设置TRIGn位的写操作一起发生，则同步开始，但TRIGn位由于写操作而保持设置。




注意事项
所有的硬件触发输入都有相同的行为。

**图42-52.硬件触发事件，HWTRIGMODE = 0**

如果HWTRIGMODE=1，那么TRIGn位只有在写入0时才会被清除。

**注意事项**

HWTRIGMODE位必须是1，只有在增强的PWM同步（SYNCMODE = 1）下才可以。

### 42.5.13.2 软件触发器

当1被写入SYNC[SWSYNC]位时，软件触发事件发生。当0被写入SWSYNC位或由软件事件启动的PWM同步完成后，SWSYNC位被清零。

如果在前一个软件触发事件启动的PWM同步结束的同时，发生了另一个软件触发事件（通过向SWSYNC位再写一个1），则新的PWM同步开始，SWSYNC位仍等于1。

如果SYNCMODE=0，那么SWSYNC位也会被FTM根据PWMSYNC和REINIT位清零。在这种情况下，如果（PWMSYNC = 1）或（PWMSYNC = 0和REINIT = 0），那么SWSYNC位在软件触发事件发生后的下一个选定的加载点被清除；见同步点和下图。如果(PWMSYNC=0)和(REINIT=1)，则SWSYNC位在软件触发事件发生时被清零。

如果SYNCMODE=1，那么SWSYNC位也会被FTM根据SWRSTCNT位清零。如果SWRSTCNT=0，则SWSYNC位在软件触发事件发生后的下一个选定的加载点被清零；见下图。如果SWRSTCNT=1，那么SWSYNC位在软件触发事件发生时被清除。

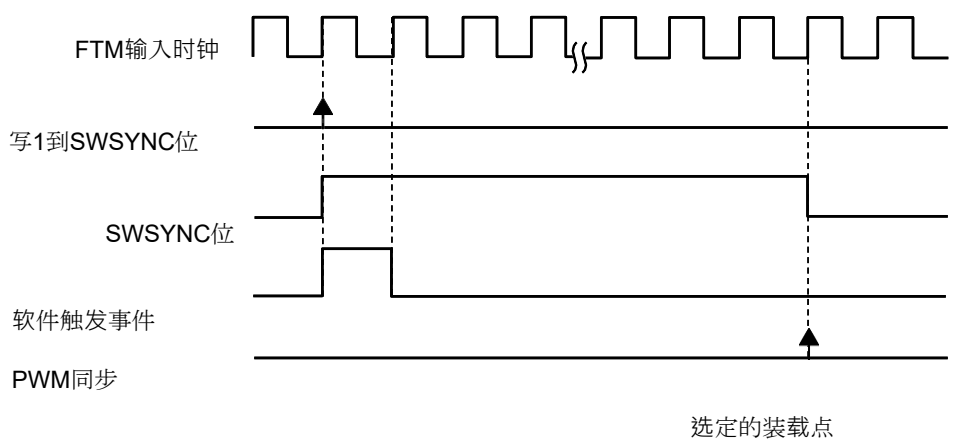



**图42-53.软件触发事件**

### 42.5.13.3 同步点

同步点是寄存器可以通过PWM同步更新其写缓冲器的点。这些同步点是安全点，因为保证了生成的PWM信号的平滑过渡。

在向上计数中，同步点是指FTM计数器从MOD变为CNTIN时。在这种情况下，如果(CNTMIN=1)或(CNTMAX=1)，同步点被启用。

在向上-向下计数中，同步点是。

- 如果（CNTMAX = 1），当FTM计数器从（MOD）变为（MOD - 1）时。
- 如果（CNTMIN = 1），当FTM计数器从（CNTIN）变为（CNTIN + 1）。

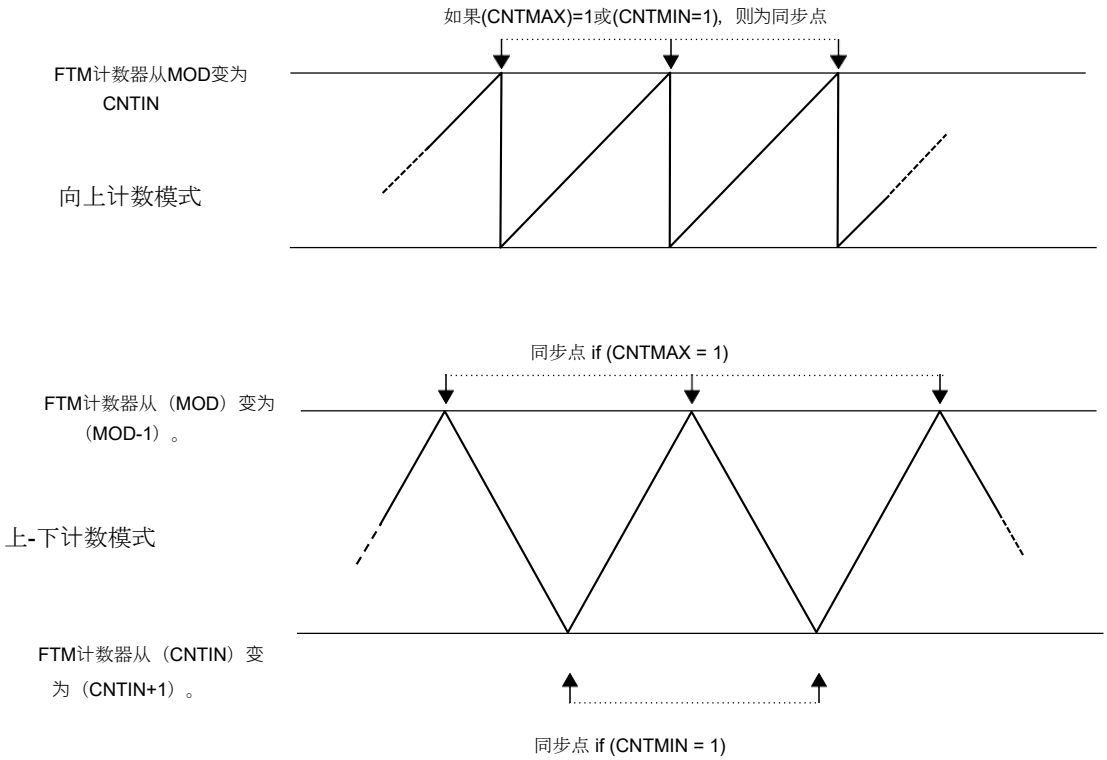



**图42-54.同步点**

### 42.5.13.4 MOD寄存器同步化

MOD寄存器的同步用其缓冲区的值更新MOD寄存器。如果（FTMEN = 1），该同步被启用。

MOD寄存器的同步可以通过增强的PWM同步（SYNCMODE=1）或传统的PWM同步（SYNCMODE
= 0).然而，我们希望MOD寄存器只能通过增强的PWM同步来实现同步。

在增强型PWM同步的情况下，MOD寄存器的同步取决于SWWRBUF、SWRSTCNT、HWWRBUF和HWRSTCNT位，根据这个流程图。

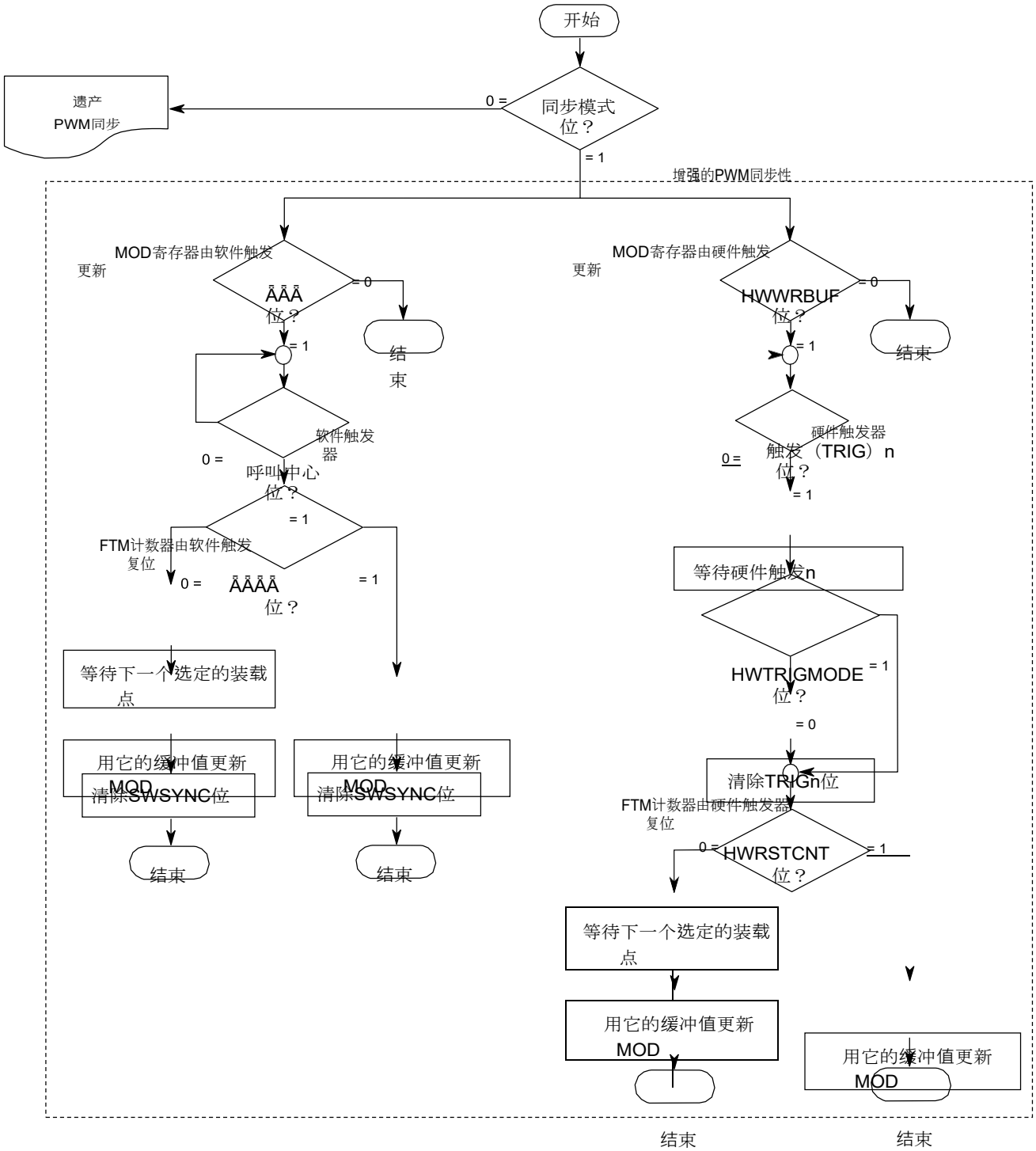



**图42-55.MOD寄存器同步流程图**

在传统的PWM同步的情况下，MOD寄存器的同步取决于PWMSYNC和REINIT位，根据以下描述。

如果(SYNCMODE=0)、(PWMSYNC=0)和(REINIT=0)，那么这种同步是在启用的触发事件发生后的下一个选定的加载点上进行。如果触发事件是一个软件触发，

新值当SWSYNC位在下一个选定的加载点上被清除。

加载点。如果触发事件是硬件触发，那么根据硬件触发，触发使能位（TRIGn）被清零。下面是带有软件和硬件触发器的例子。

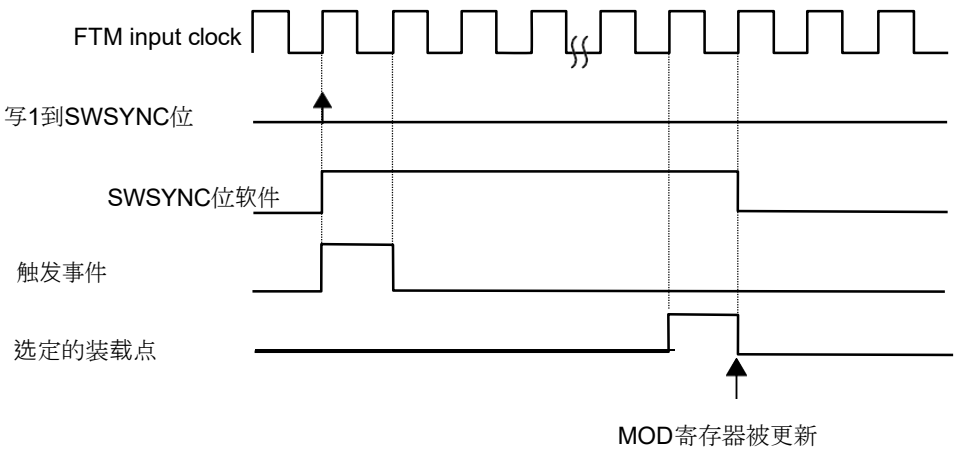



**图42-56.使用（SYNCMODE = 0）、（PWMSYNC = 0）、（REINIT = 0）的MOD同步，并使用软件触发**

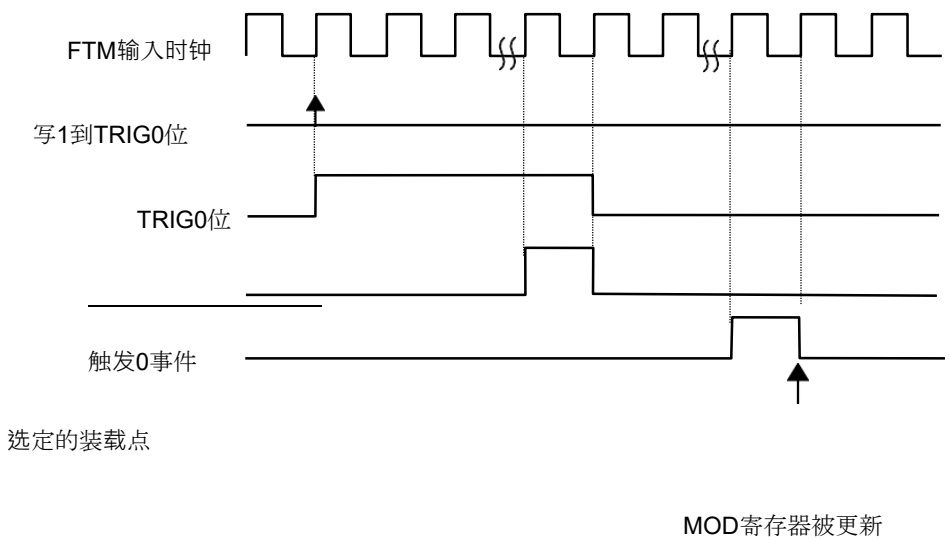



**图42-57.用（SYNCMODE=0）、（HWTRIGMODE=0）、（PWMSYNC=0）、（REINIT=0）和硬件触发器进行MOD同步**

如果(SYNCMODE=0)、(PWMSYNC=0)和(REINIT=1)，那么这种同步是在下一个启用的触发事件上进行的。如果触发事件是一个软件触发器，那么SWSYNC位将按照下面的例子被清除。如果触发事件是一个硬件触发器，那么TRIGn位根据硬件触发器被清零。有软件和硬件触发器的例子如下。

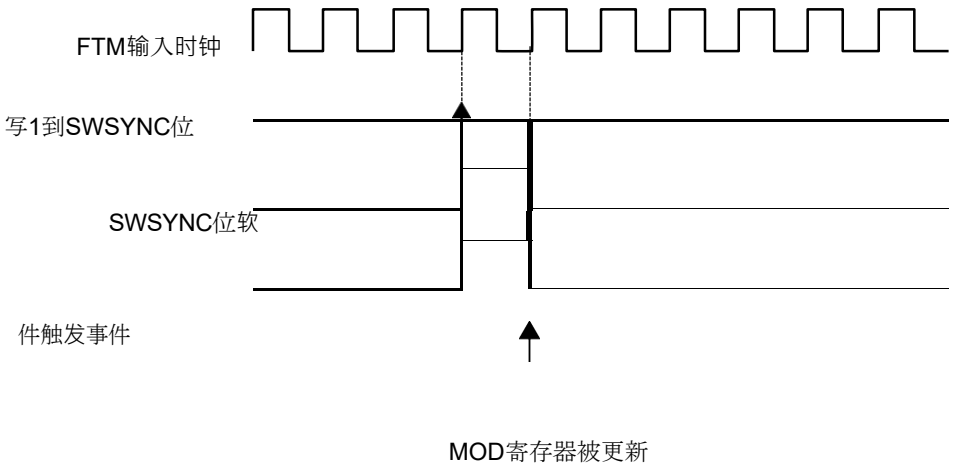



**图42-58.使用(SYNCMODE = 0)、(PWMSYNC = 0)、(REINIT = 1)的MOD同步，并使用软件触发。**

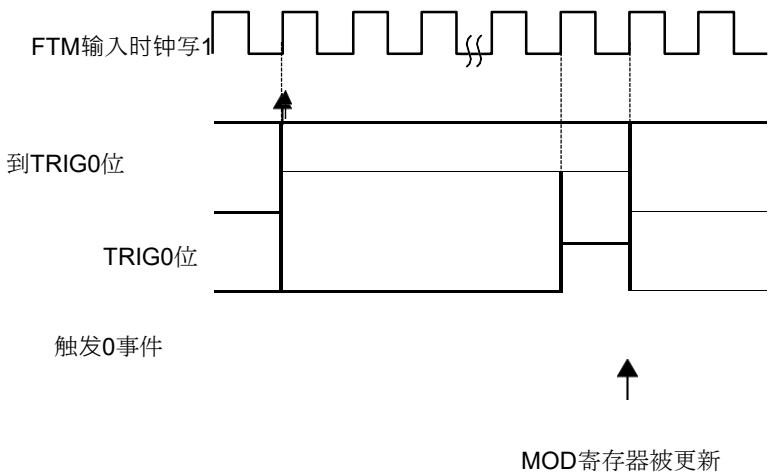
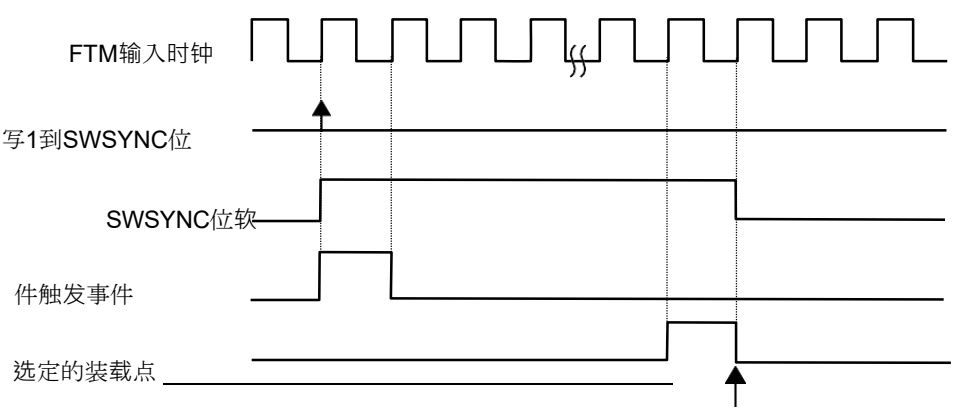



**图42-59.用（SYNCMODE=0）、（HWTRIGMODE=0）、（PWMSYNC=0）、（REINIT=1）和硬件触发器进行MOD同步**

如果(SYNCMODE=0)和(PWMSYNC=1)，则在软件触发事件发生后的下一个选定的加载点进行同步。SWSYNC位在下一个选定的加载点上被清零。



MOD寄存器被更新

**图42-60.在(SYNCMODE = 0)和(PWMSYNC = 1)情况下进行MOD同步**

### 42.5.13.5 CNTIN寄存器同步

CNTIN寄存器的同步以其缓冲值更新CNTIN寄存器。

如果(FTMEN=1)、(SYNCMODE=1)和(CNTINC=1)，则该同步功能被启用。CNTIN寄存器的同步只能由增强型PWM同步（SYNCMODE=1）来完成。该同步机制与由增强型PWM同步完成的MOD寄存器同步相同；参见MOD寄存器同步。

### 42.5.13.6 C（n）V和C（n+1）V寄存器同步

C(n)V和C(n+1)V寄存器同步更新C(n)V和C(n+1)V寄存器的缓冲器值。

如果(FTMEN=1)和(SYNCEN=1)，该同步被启用。该同步机制与MOD寄存器的同步相同。然而，预计C(n)V和C(n+1)V寄存器只能通过增强的PWM同步（SYNCMODE = 1）来同步。

### 42.5.13.7 OUTMASK寄存器同步

OUTMASK寄存器的同步用它的缓冲值更新OUTMASK寄存器。

OUTMASK寄存器可以在FTM输入时钟的每个上升沿更新（SYNCHOM = 0），通过增强的PWM同步（SYNCHOM = 1和SYNCMODE = 1）或通过传统的PWM同步（SYNCHOM = 1和SYNCMODE = 0）。然而，预计OUTMASK寄存器只能通过增强型PWM同步来实现同步。

在增强PWM同步的情况下，OUTMASK寄存器的同步取决于SWOM和HWOM位。见下面的流程图。

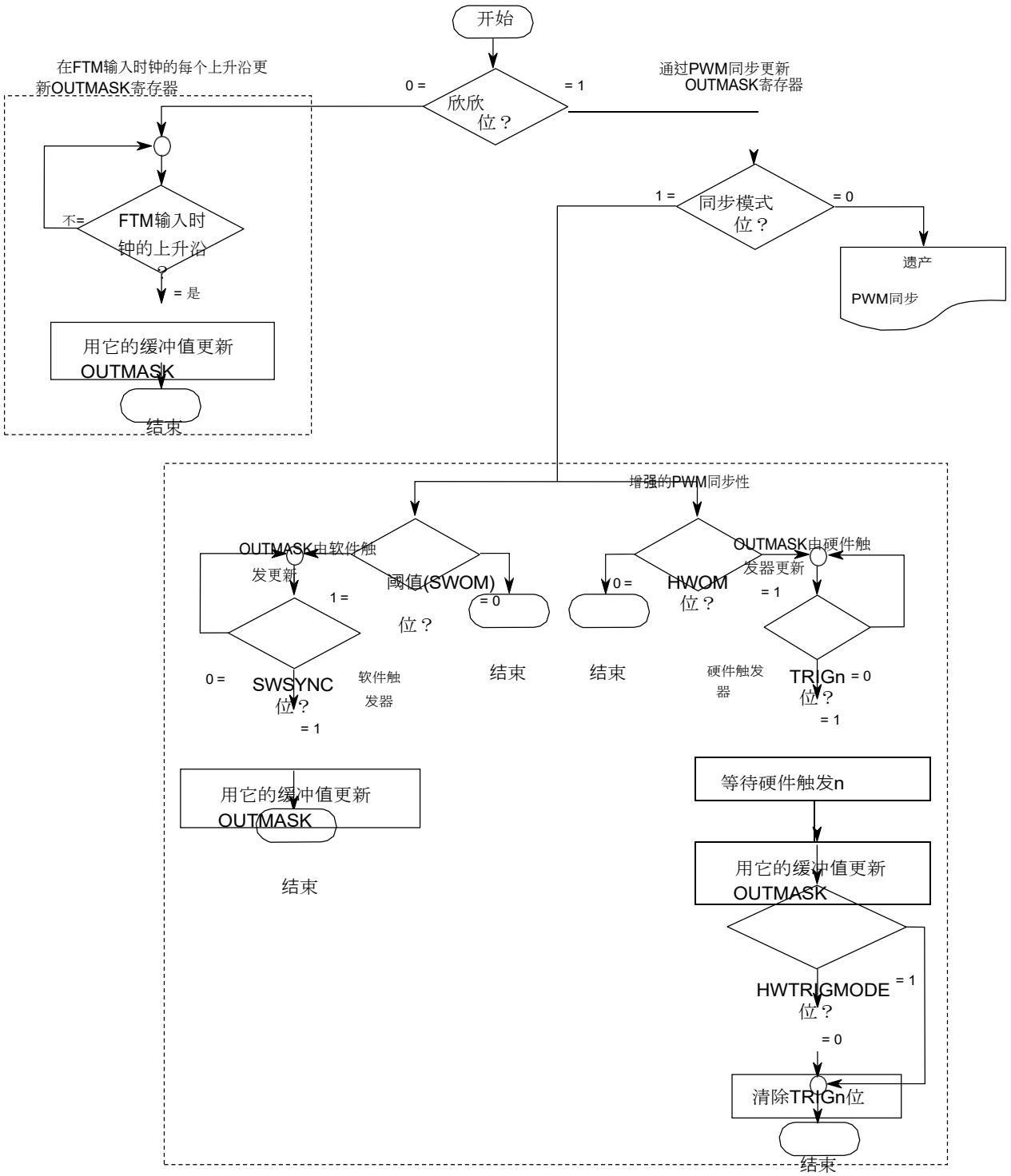



**图42-61.OUTMASK寄存器的同步流程图**

在传统的PWM同步的情况下，OUTMASK寄存器的同步取决于PWMSYNC位，根据以下描述。

如果(SYNCMODE = 0)，(SYNCHOM = 1)，和(PWMSYNC = 0)，那么这个同步是在下一个启用的触发事件上进行的。如果触发事件是一个软件触发器，那么SWSYNC位在下一个选定的加载点上被清除。如果触发事件是硬件触发，那么TRIGn位根据硬件触发被清零。有软件和硬件触发器的例子如下。

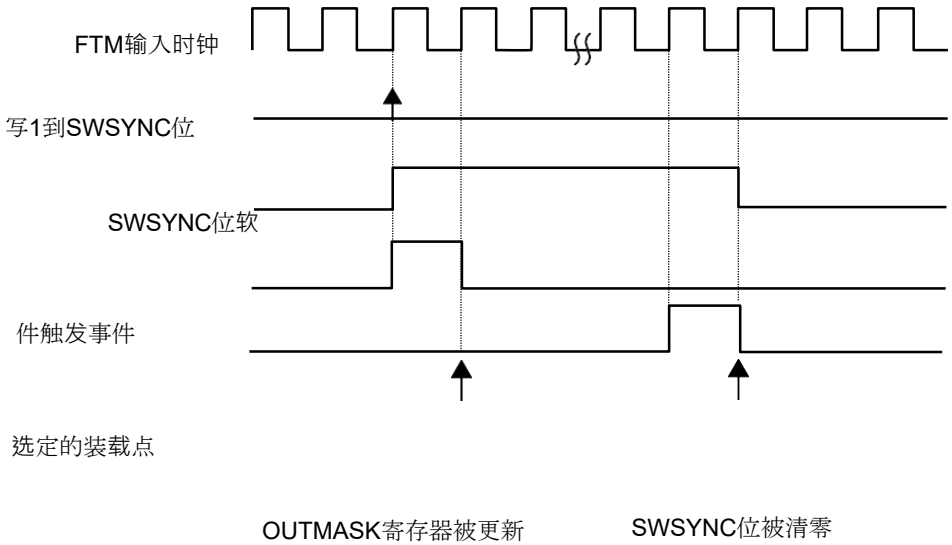



**图42-62.使用（SYNCMODE = 0）、（SYNCHOM = 1）、（PWMSYNC = 0）和软件触发的OUTMASK同步**

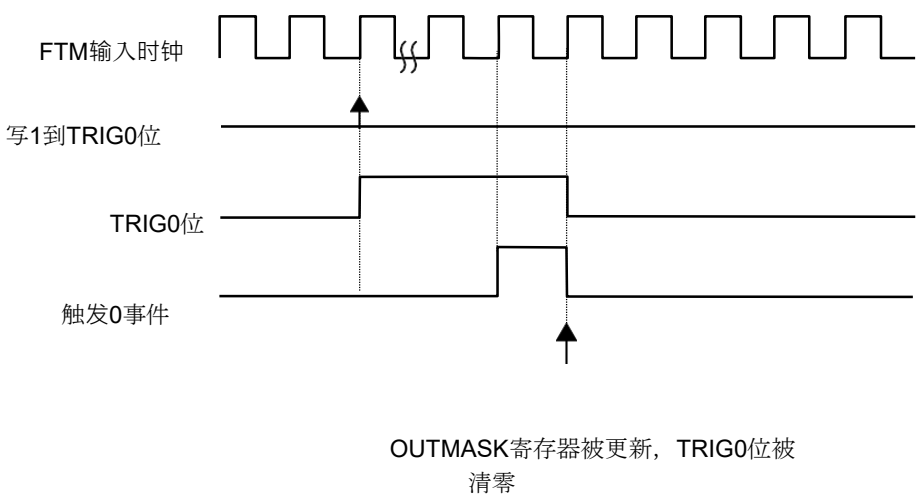



**图42-63.使用(SYNCMODE=0)、(HWTRIGMODE=0)、(SYNCHOM=1)、(PWMSYNC=0)的OUTMASK同步，并使用了硬件触发器**

如果（SYNCMODE = 0），（SYNCHOM = 1）和（PWMSYNC = 1），那么这个同步是在下一个启用的硬件触发器上进行的。TRIGn位根据硬件触发器被清零。下面是一个带有硬件触发器的例子。

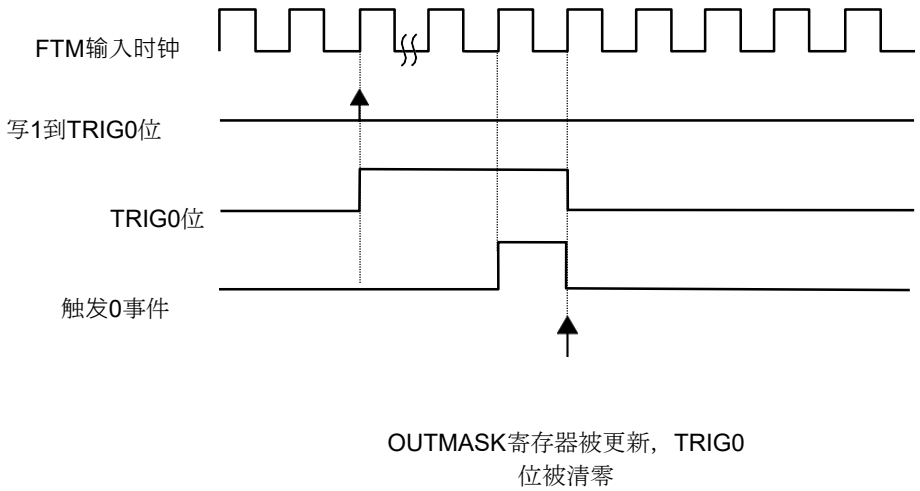



**图42-64.使用(SYNCMODE = 0)、(HWTRIGMODE = 0)、(SYNCHOM = 1)、(PWMSYNC = 1)的OUTMASK同步，并使用了一个硬件触发器**

### 42.5.13.8 INVCTRL寄存器同步

INVCTRL寄存器同步用其缓冲值更新INVCTRL寄存器。

INVCTRL寄存器可以在FTM输入时钟的每个上升沿（INVC=0）或通过增强的PWM同步（INVC=1和SYNCMODE=1）根据以下流程图进行更新。

在增强型PWM同步的情况下，INVCTRL寄存器的同步取决于SWINVC和HWINVC位。

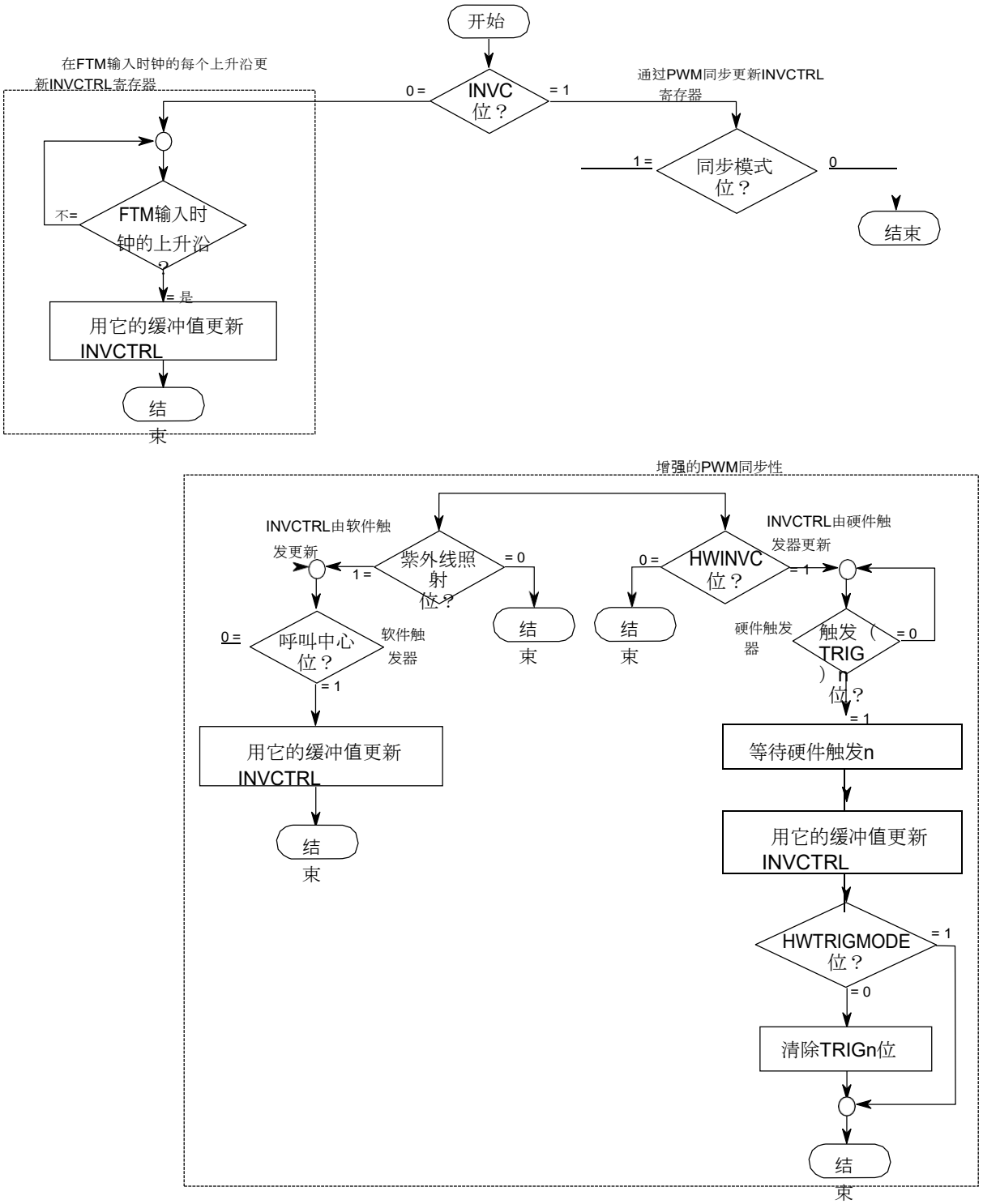



**图42-65.INVCTRL寄存器同步流程图**

## 42.5.13.9 SWOCTRL寄存器的同步

SWOCTRL寄存器的同步用它的缓冲值更新SWOCTRL寄存器。

在FTM输入时钟的每个上升沿，SWOCTRL寄存器可以被更新（SWOC = 0）或通过增强的PWM同步（SWOC = 1和SYNCMODE = 1），根据以下流程图。

在增强型PWM同步的情况下，SWOCTRL寄存器的同步取决于SWSOC和HWSOC位。

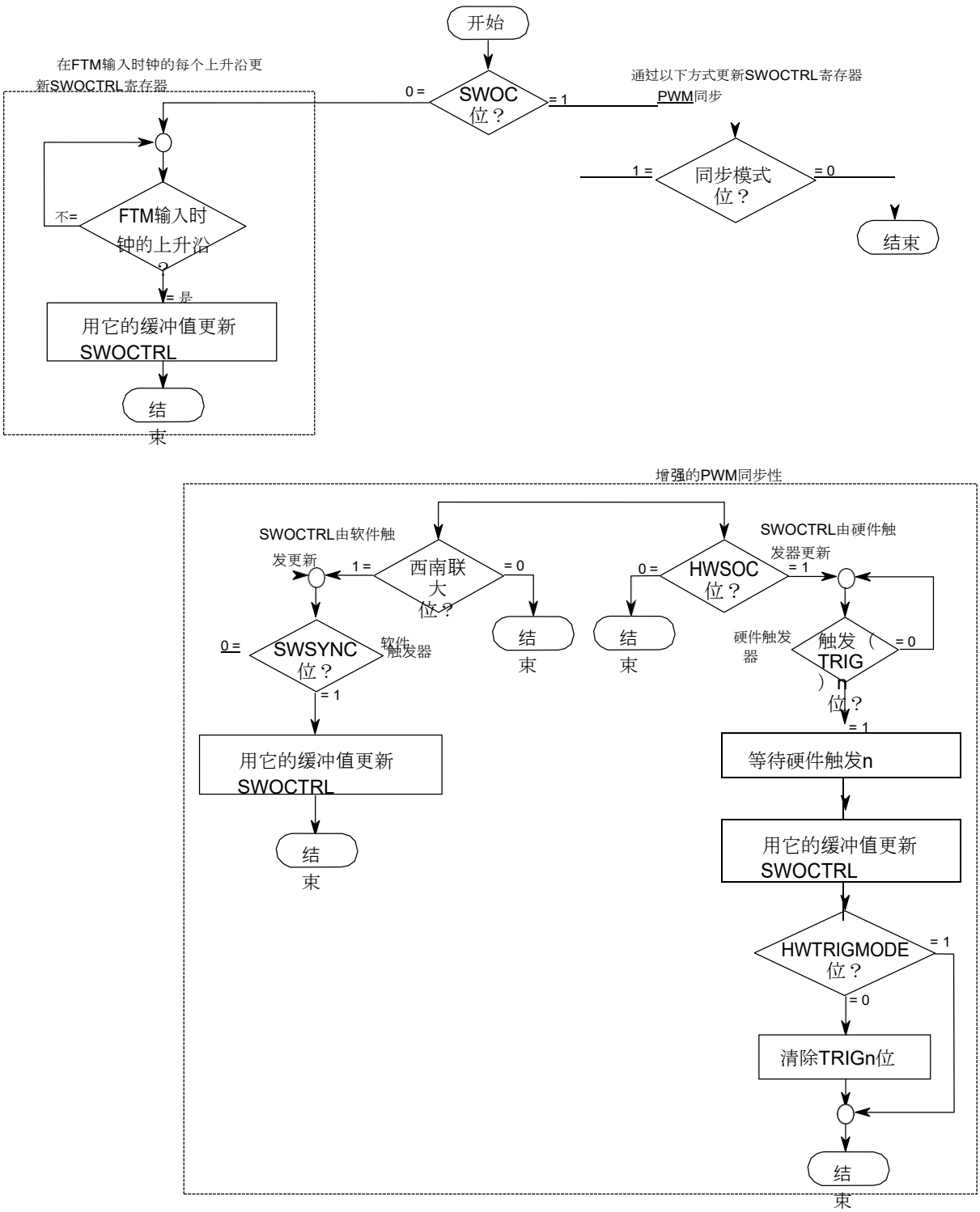


图42-66.SWOCTRL寄存器的同步流程图

### 42.5.13.10 FTM计数器同步

FTM计数器同步是一种机制，允许FTM在PWM周期的某一点上重新启动PWM生成。除了在输出比较模式下的通道外，各通道的输出被强制到其初始值，FTM计数器被强制到由CNTIN寄存器定义的初始计数值。

下图显示了FTM计数器的同步情况。请注意，在同步事件发生后，通道（n）被设置为其初始值，而通道（n
+如果没有选择死区插入，那么在同步事件发生后，通道（n+1）立即过渡到逻辑值1。

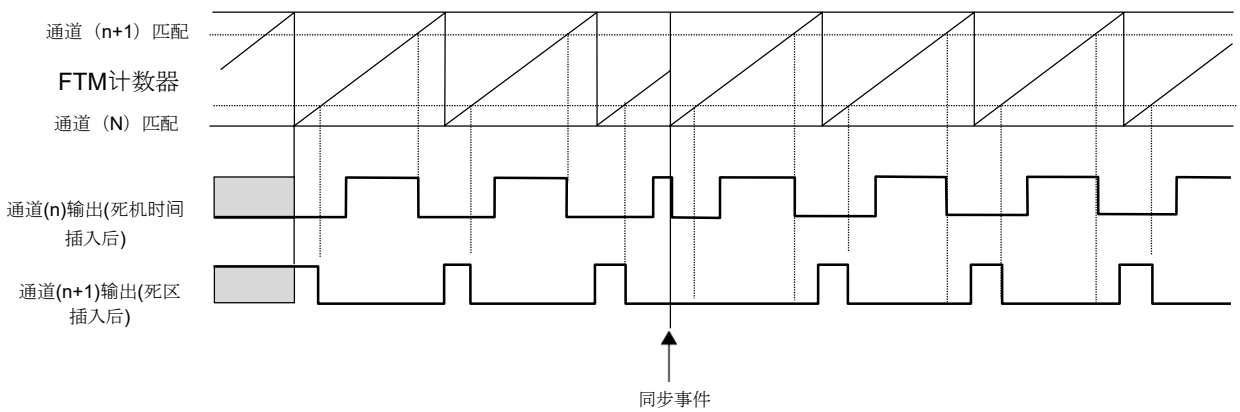



**图42-67.FTM计数器同步**

FTM计数器的同步可以通过增强型PWM同步（SYNCMODE=1）或传统的PWM同步（SYNCMODE
= 0).然而，FTM计数器必须只通过增强的PWM同步来进行同步。

在增强型PWM同步的情况下，根据以下流程图，FTM计数器的同步取决于SWRSTCNT和HWRSTCNT位。

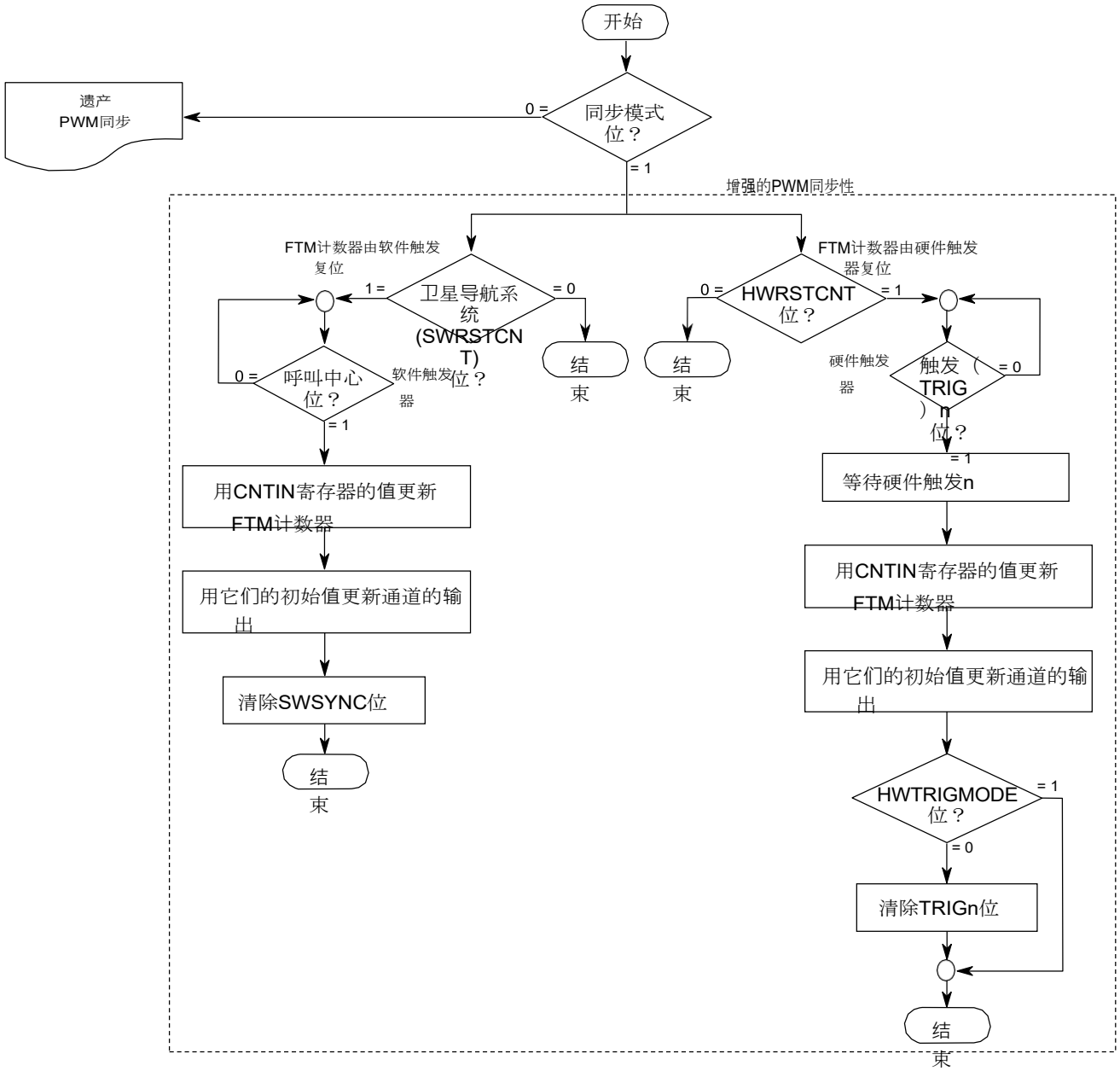



**图42-68.FTM计数器同步流程图**

在传统PWM同步的情况下，FTM计数器的同步取决于REINIT和PWMSYNC位，根据以下描述。

如果(SYNCMODE=0)、(REINIT=1)和(PWMSYNC=0)，则该同步是在下一个启用的触发事件中进行。如果触发事件是一个软件触发器，那么SWSYNC位将根据下面的例子被清除。如果触发事件是一个硬件触发器，那么TRIGn位根据硬件触发器被清零。软件和硬件触发器的例子如下。

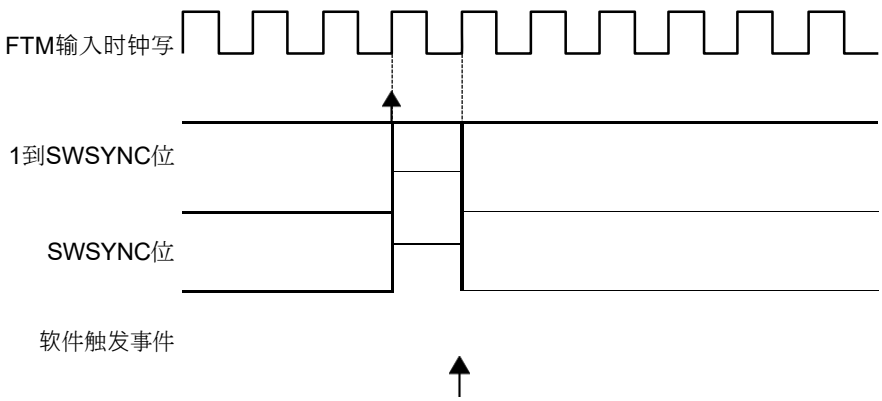



**图42-69.使用(SYNCMODE = 0)、(REINIT = 1)、(PWMSYNC = 0)的FTM计数器同步，并使用软件触发器**

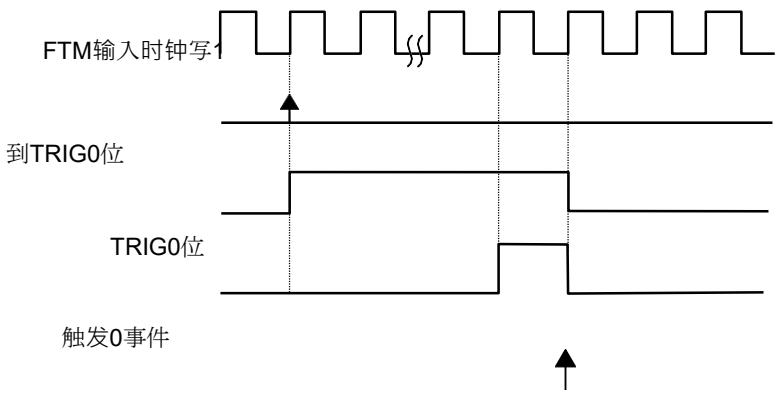
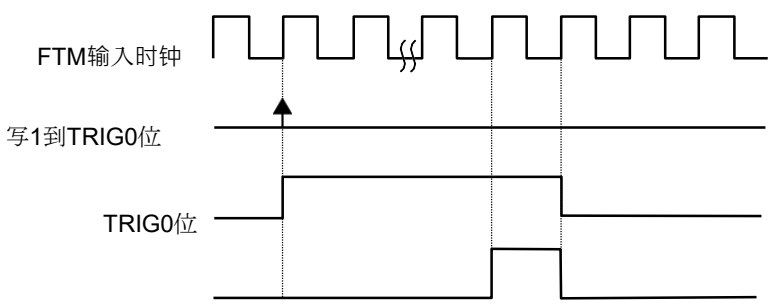



**图42-70.用(SYNCMODE=0)、(HWTRIGMODE=0)、(REINIT=1)、(PWMSYNC=0)进行FTM计数器同步，并使用了硬件触发器**

如果(SYNCMODE = 0)、(REINIT = 1)和(PWMSYNC = 1)，那么这种同步是在下一个启用的硬件触发时进行的。TRIGn位根据硬件触发器被清零。



触发0事件

FTM计数器用CNTIN寄存器的值更新，通道输出被强制到其初始值

**图42-71.使用(SYNCMODE = 0)、(HWTRIGMODE = 0)、(REINIT = 1)、(PWMSYNC = 1)的FTM计数器同步，并使用了一个硬件触发器**

## 42.5.14 倒置

反相功能在通道（n）和通道（n+1）输出之间交换信号。在以下情况下，选择反转操作。

- QUADEN = 0
- DECAPEN = 0
- COMP = 1，和
- INVm = 1 (其中m代表一对通道)

INVCTRL寄存器中的INVm位根据INVCTRL寄存器同步更新其缓冲值。

在通道(n)ELSB:ELSA=1:0的组合模式下，通道(n)的输出在周期开始时被强制为低电平(FTM计数器=CNTIN)，在通道(n)被强制为高电平。
(n)
匹配，并在通道(n+1)匹配时强制为低。如果选择反相，通道（n）的输出行为在PWM周期开始时变为强制高电平，在通道（n）匹配时强制低电平，在通道（n+1）匹配时强制高电平。见下图。

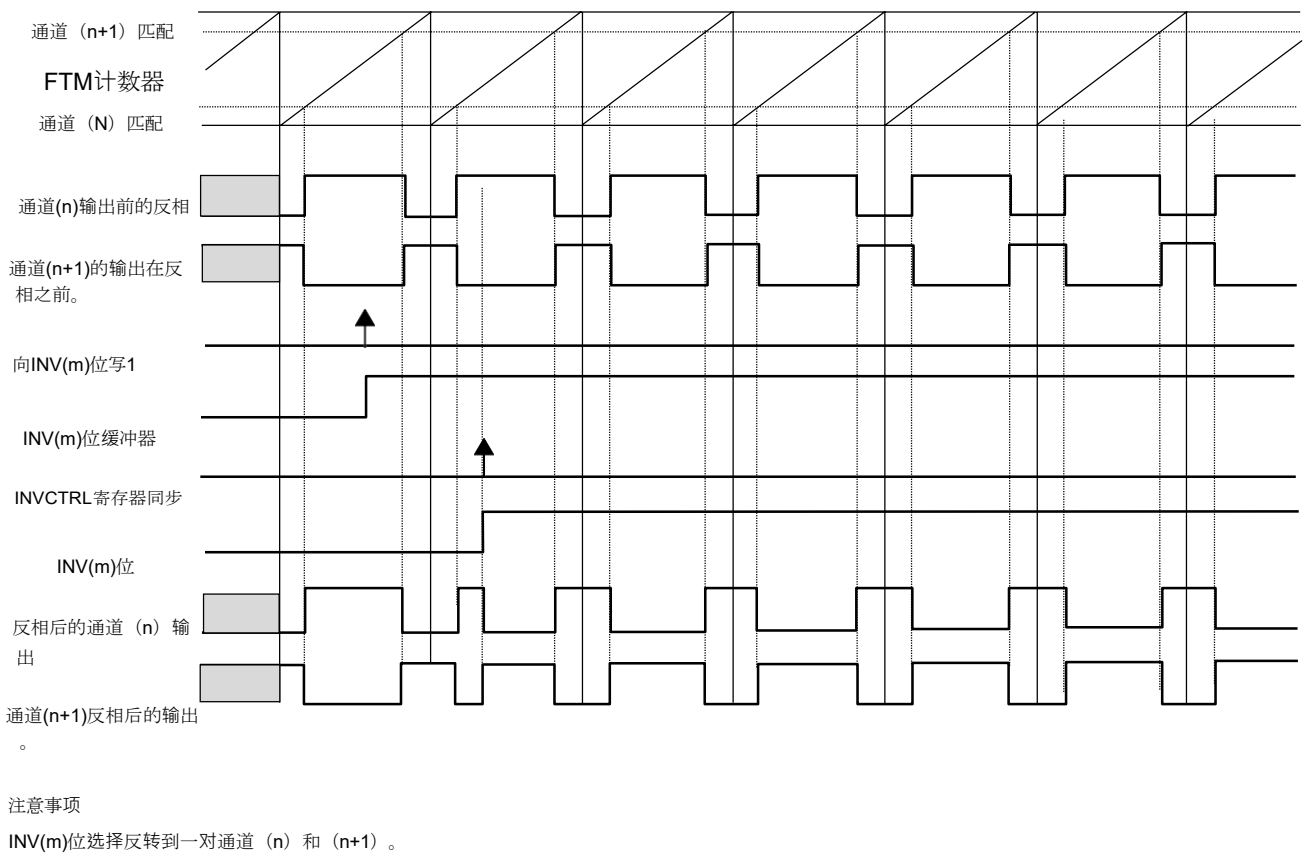



**图42-72.通道(n)和(n+1)在联合模式下反相后的输出，通道(n)ELSB:ELSA=1:0**

请注意，通道(n)的ELSB:ELSA位应该被考虑，因为它们定义了通道输出的有效状态。在通道(n)ELSB:ELSA=X:
1的组合模式下，通道(n)输出在周期开始时被强制为高电平，在通道(n)匹配时被强制为低电平，在通道(n+1)匹配时被强制为高电平。当选择反相时，通道（n）和（n+1）呈现的波形如下图所示。

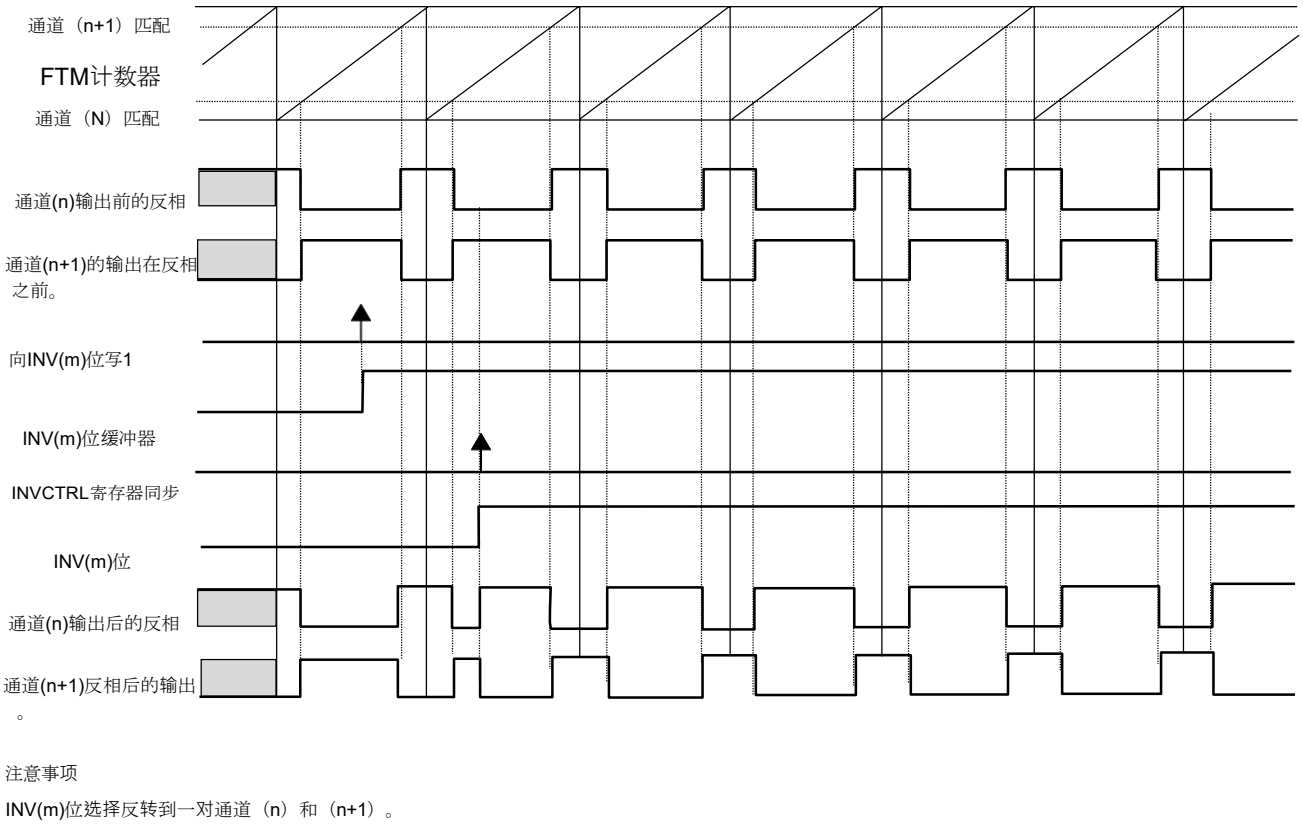



**图42-73.通道(n)和(n+1)在联合模式下反相后的输出，通道(n) ELSB:ELSA=X:1**

**注意事项**

在输出比较模式和修改后的组合PWM模式下，反转是不可用的。

## 42.5.15 软件输出控制模式

软件输出控制迫使通道在PWM生成的特定时间按照软件定义的值进行输出。

当选择软件输出控制时。

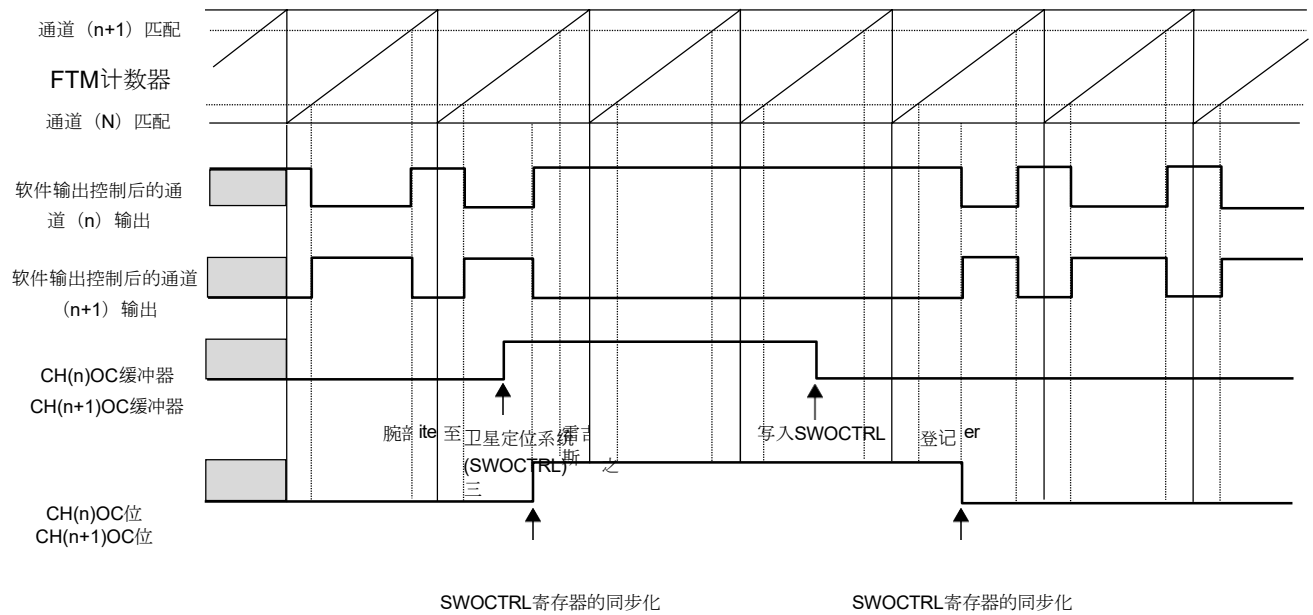
- QUADEN = 0
- DECAPEN = 0，并且

- CH(n)OC = 1

CH(n)OC位启用特定通道输出的软件输出控制，CH(n)OCV选择强制到该通道输出的值。

SWOCTRL寄存器中的CH(n)OC和CH(n)OCV位都被缓冲，并根据SWOCTRL寄存器的同步性更新其缓冲值。

下图显示了使用软件输出控制时的通道(n)和(n+1)输出信号。在这种情况下，通道（n）和（n+1）被设置为组合和互补模式。




注意事项
通道（n）ELSB:ELSA = X:1，CH(n)OCV = 1，CH(n+1)OCV = 0。

**图42-74.合并和互补模式下的软件输出控制实例**

当COMP位为零时，软件输出控制在通道（n）和（n+1）上强制执行以下值。

**表42-12.当（COMP = 0）时的软件输出控制行为**

| CH(n)OC | CH(n+1)OC | CH(n)OCV | CH(n+1)OCV | 通道(n) 输出 | 通道（n+1）输出 |
|---|---|---|---|---|---|
| 0 | 0 | X | X | 不被SWOC修改 | 不被SWOC修改 |
| 1 | 1 | 0 | 0 | 被迫为零 | 被迫为零 |
| 1 | 1 | 0 | 1 | 被迫为零 | 被逼到一个 |
| 1 | 1 | 1 | 0 | 被逼到一个 | 被迫为零 |
| 1 | 1 | 1 | 1 | 被逼到一个 | 被逼到一个 |

当COMP位为1时，软件输出控制在通道（n）和（n+1）上强制执行以下值。

表42-13.当（COMP = 1）时的软件输出控制行为

| CH(n)OC | CH(n+1)OC | CH(n)OCV | CH(n+1)OCV | 通道(n) 输出 | 通道（n+1）输出 |
|---|---|---|---|---|---|
| 0 | 0 | X | X | 不被SWOC修改 | 不被SWOC修改 |
| 1 | 1 | 0 | 0 | 被迫为零 | 被迫为零 |
| 1 | 1 | 0 | 1 | 被迫为零 | 被逼到一个 |
| 1 | 1 | 1 | 0 | 被逼到一个 | 被迫为零 |
| 1 | 1 | 1 | 1 | 被逼到一个 | 被迫为零 |

**注意事项**

- CH(n)OC和CH(n+1)OC位应相等。
- 当软件输出控制被启用时，即CH(n)OC=1和/或CH(n)OC=1时，COMP位不能被修改。
  +1）OC=1。
- 软件输出控制在禁用或启用FTM计数器时具有相同的行为（见状态和控制寄存器中的CLKS字段描述）。

## 42.5.16 死亡时间的插入

当DTEN被设置时，死区插入被启用，并且串联{DTVALEX[3:0], DTVAL[5:0]}为非零。

DEADTIME 寄存器定义了可用于所有 FTM通道的死区延迟。DEADTIME延时的时钟是FTM输入时钟除以DTPS位，{DTVALEX[3:0], DTVAL[5:0]}位定义死区模数，即死区预分频器时钟的数量。

死锁时间的插入确保没有两个互补信号（通道（n）和（n+1））同时驱动活动状态。

如果POL(n)=0，POL(n+1)=0，并且启用了死区时间，那么当通道(n)匹配(FTM计数器=C(n)V)发生时，通道(n)输出保持在低值，直到通道(n)输出被设置的死区时间延迟结束。同样，当通道(n+1)匹配(FTM计数器=C(n+1)V)发生时，通道(n+1)输出保持在低值，直到通道(n+1)输出被设置的死区延时结束。请看下面的数字。

如果POL(n)=1，POL(n+1)=1，并且启用死区时间，那么当通道(n)匹配(FTM计数器=C(n)V)发生时，通道(n)输出保持在高值，直到死区时间延迟结束时通道(n)输出被

清零。类似地。

当通道(n+1)匹配(FTM计数器=C(n+1)V)发生时，通道(n+1)的输出保持在高值，直到死区延迟结束时，通道(n
+1）输出被清除。

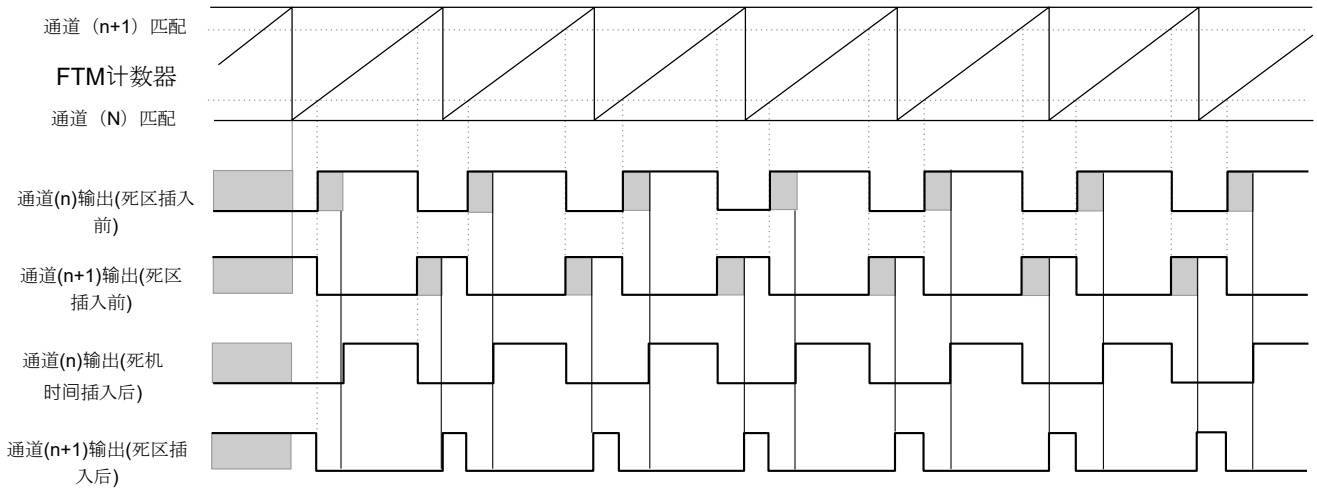



**图42-75.在ELSB:ELSA=1:0，POL(n)=0和POL(n+1)=0的情况下，死机时间插入**

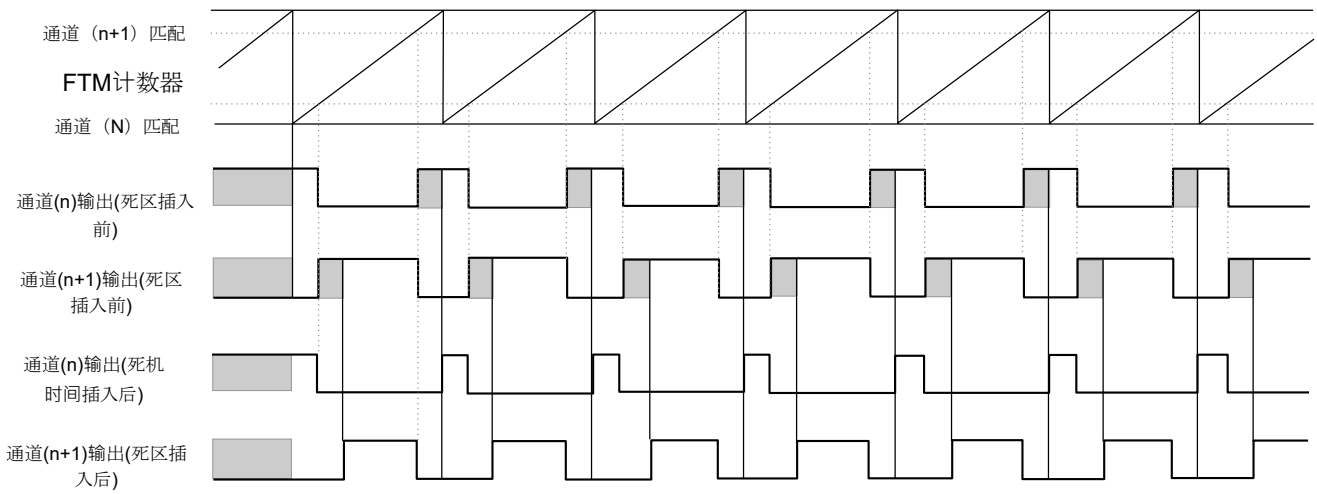



**图42-76.在ELSB:ELSA=X:1，POL(n)=0和POL(n+1)=0的情况下，死机时间插入**

### 注意事项

- 死锁时间功能必须只在互补模式下使用。
- 在输出比较模式下，死锁时间功能不可用。

## 42.5.16.1 按一对通道分开的死期

按一对通道分离的死期值允许每对通道有自己的死期值（即，位DTVALEX[3:0]、DTPS[1:0]和DTVAL[5:0]）。

寄存器PAIR0DEADTIME、PAIR1DEADTIME、PAIR2DEADTIME、和根据FTM配置，PAIR3DEADTIME是可用的。

下图描述了根据FTM配置，每对死机值是如何定义的。

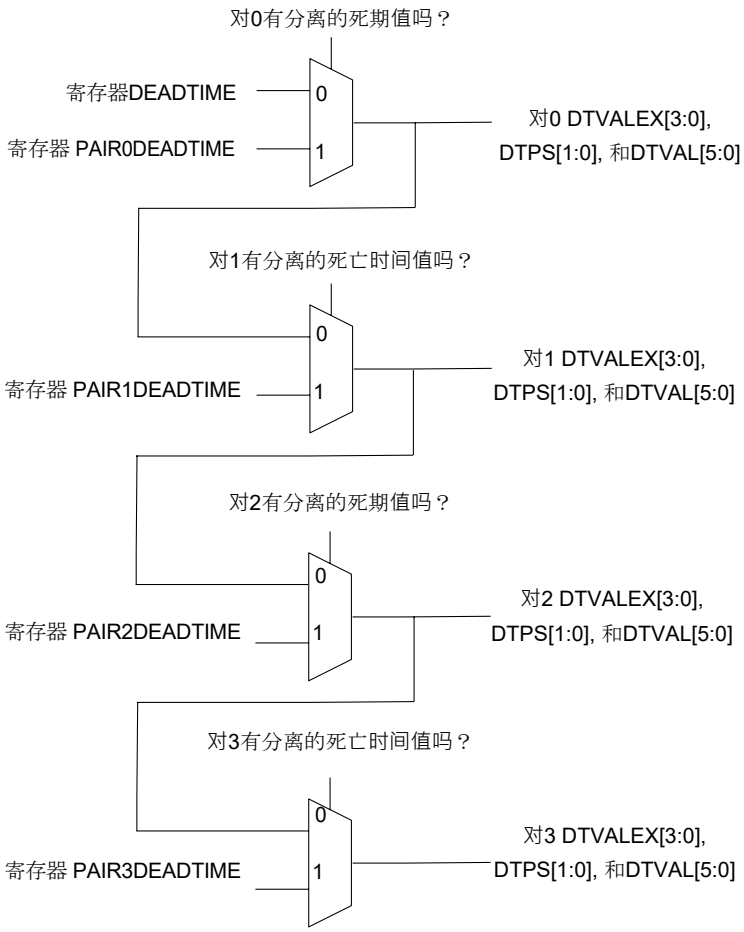



**图42-77.按一对信道分列的死期**

对寄存器DEADTIME的写入也会更新可用的寄存器PAIR(j)DEADTIME（其中j是一对通道）。正因为如此，对寄存器DEADTIME的写入应该在对寄存器PAIR(j)DEADTIME的写入之前完成。

如果通道对0有分离的死区值，那么对寄存器DEADTIME的读取就会返回寄存器PAIR0DEADTIME的读取值。否则，读取的值就是寄存器DEADTIME的值。

### 42.5.16.2 死锁时间插入的角落案例

如果（PS[2:0]被清除），（DTPS[1:0]=0:0或DTPS[1:0]=0:1）。

- 并且死时延迟大于或等于通道（n）的占空比（（C（n+1)V-C(n)V)×FTM输入时钟)，则通道(n)的输出始终为非活动值(POL(n)位值)。
- 并且死区时间延迟大于或等于通道(n+1)的占空比((MOD - CNTIN + 1 - (C(n+1)V - C(n)V))×FTM输入时钟），那么通道（n+1）的输出总是非活动值（POL（n+1）位值）。

虽然，在大多数情况下，死区延迟与通道(n)和(n+1)占空比没有可比性，但以下数字显示了死区延迟与占空比相当的例子。

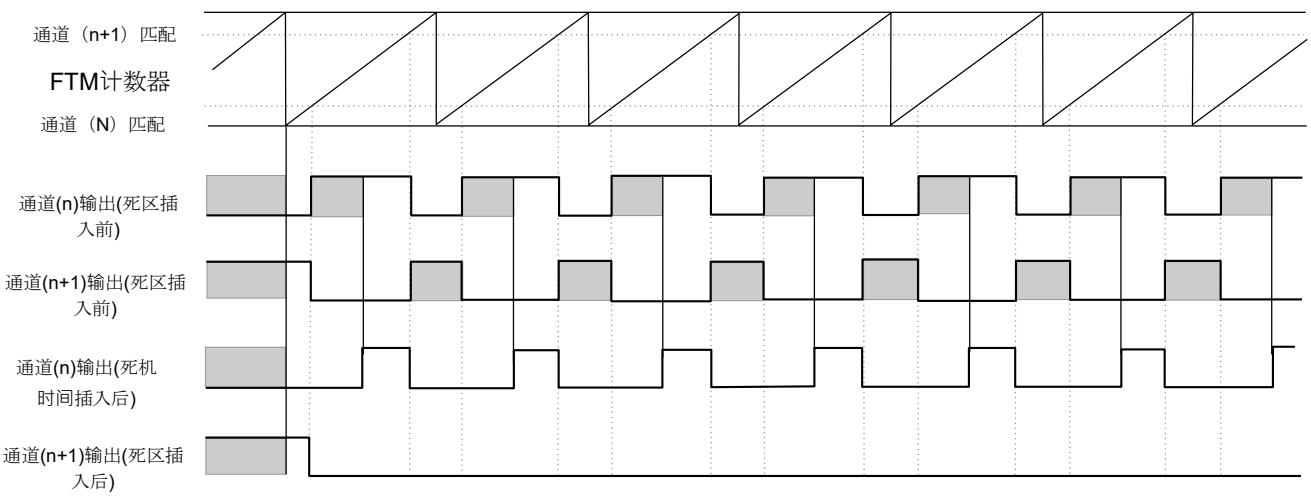



**图42-78.当死区延迟与通道（n+1）占空比相当时，死区插入的例子（通道（n）ELSB:ELSA=1:0，POL(n)=0，POL(n+1)=0）。**

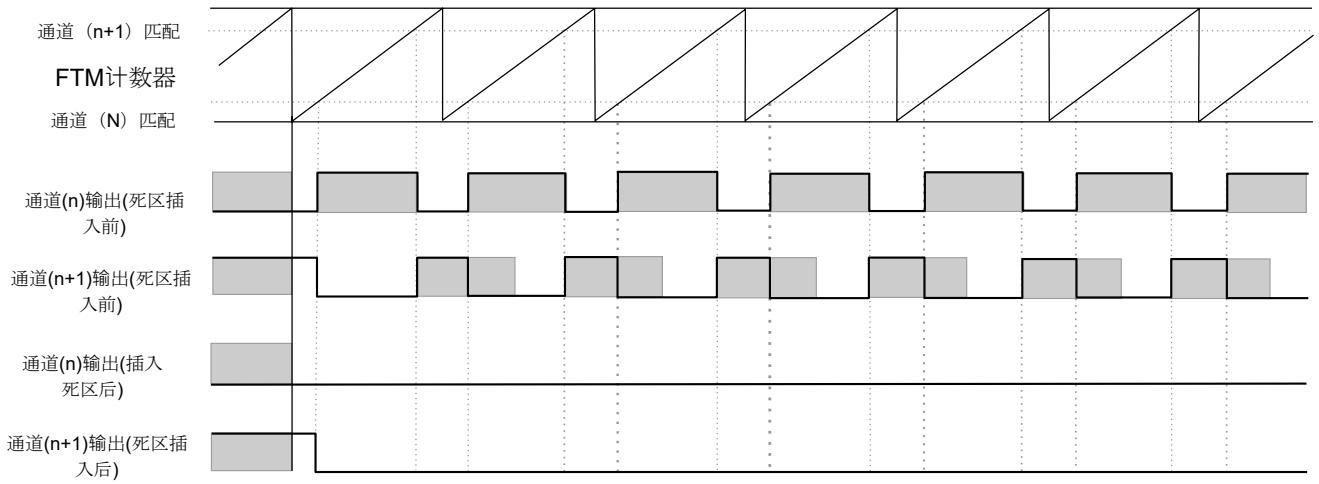



**图42-79.当死区延迟与通道（n）和（n+1）占空比相当时，死区插入的例子（通道（n）ELSB:ELSA=1:0，POL（n）=0，和POL（n+1）=0**

## 42.5.17 输出掩码

输出掩码可用于通过软件强制通道输出到其非活动状态。例如：控制一个BLDC电机。

对OUTMASK寄存器的任何写入都会更新其写入缓冲区。OUTMASK寄存器通过PWM同步更新其缓冲区的值；见OUTMASK寄存器的同步。

如果 CH(n)OM = 1，那么通道(n)输出被强制到其非活动状态（POLn 位值）。如果 CH(n)OM = 0，那么通道(n)的输出不受输出屏蔽的影响。见下图。

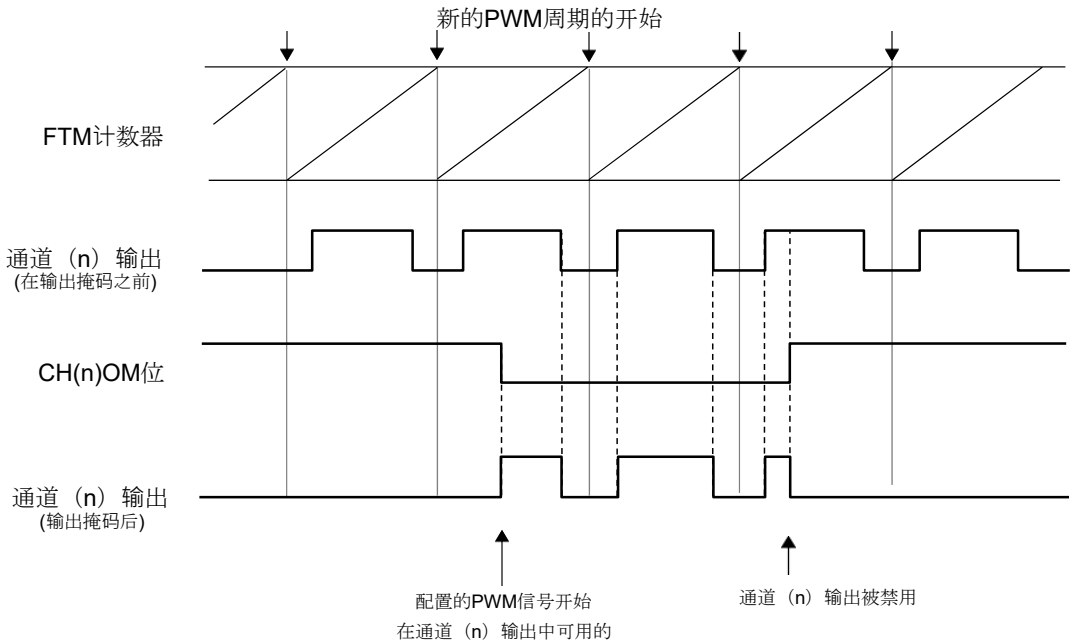



**图42-80.POLn=0时的输出掩码**

下表显示了极性控制前的输出屏蔽结果。

**表42-14.极性控制前的通道（n）的输出屏蔽结果**

| CH(n)OM | 输出掩码输入 | 输出掩码结果 |
|---|---|---|
| 0 | 不活动状态 | 不活动状态 |
| | 活动状态 | 活动状态 |
| 1 | 不活动状态 | 不活动状态 |
| | 活动状态 | |

## 42.5.18 故障控制

如果FAULTM[1:0]≠0:0，则故障控制被启用。

FTM
最多可以有四个故障输入。FAULTnEN位（其中n=0、1、2、3）启用故障输入n，FFLTRnEN位启用故障输入n滤波器。FFVAL[3:0]位选择每个启用的故障输入中的启用滤波器的值。

如果FLTPS[3:0]=0，经FTM输入时钟同步后的故障输入为滤波器输入。

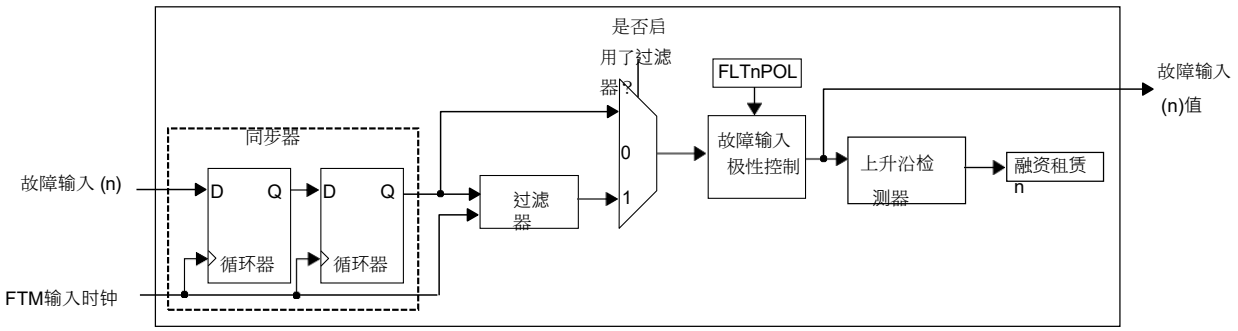



**图42-81.当FLTPS[3:0]=0时的故障控制示意图**

如果FLTPS[3:0]≠0，则经FTM输入时钟同步并经FTM滤波时钟采样的故障输入为滤波输入。

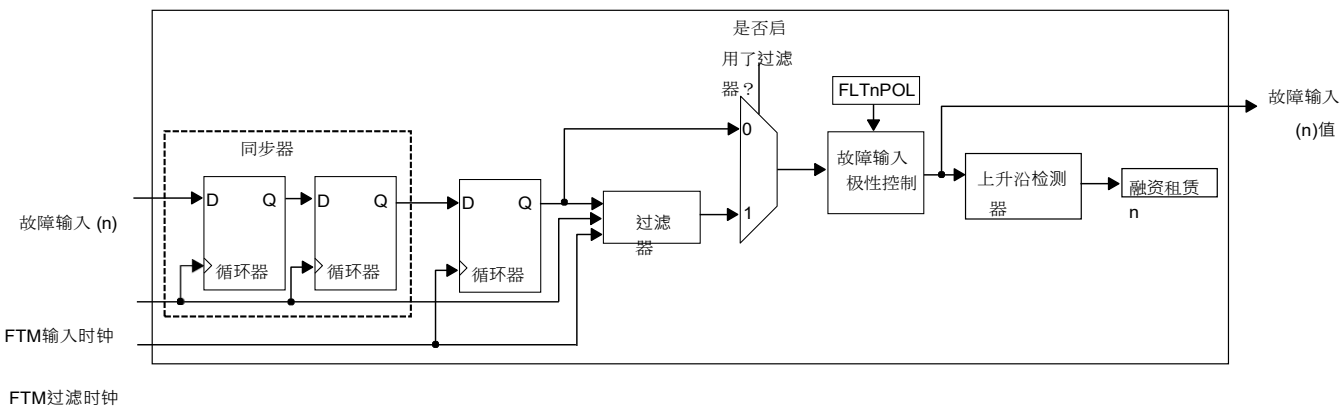



**图42-82.FLTPS[3:0]≠0时的故障控制示意图**

当故障输入（n）的状态发生变化时，计数器被重置并开始向上计数。只要故障输入（n）的新状态是稳定的，计数器继续递增。当计数器等于FFVAL[3:0]位时，新的故障输入（n）值被验证。然后，它被作为一个脉冲传输给边缘检测器。

如果在验证之前，故障输入（n）上出现相反的边沿，计数器被重置。在下一个输入转换时，计数器又开始计数。如果一个脉冲被采样为小于FFVAL[3:0]的连续上升沿的FTM滤波时钟的值，它被视为一个突发事件，不会被传递到边缘检测器。

下表显示了FTM故障输入滤波器根据其配置而增加的延迟。

**表42-15.FTM 故障输入滤波延时**

| FTM 故障输入滤波器 | FLTPS[3:0]位 | 从故障输入的选定边沿到迫使通道输出到其安全值之间的上升边沿数目 |
|---|---|---|
| • 故障输入没有输入滤波器，或<br>• 故障输入滤波器被禁用（FFLTRnEN = 0或FFVAL[3:0] = 0）。 | FLTPS[3:0] = 0 | • FTM输入时钟的3个上升沿 |
| | FLTPS[3:0] ≠ 0 | • FTM输入时钟的3个上升沿，加上<br>• 1个FTM滤波器时钟的上升沿 |
| • 故障输入I有输入滤波器，和<br>• 故障输入滤波器被启用（FFLTRnEN = 1且FFVAL[3:0] ≠ 0）。 | FLTPS[3:0] = 0 | • (4 + FFVAL[3:0]) FTM输入时钟的上升沿 |
| | FLTPS[3:0] ≠ 0 | • FTM输入时钟的4个上升沿，加上<br>• (1 + FFVAL[3:0]) FTM滤波器时钟的上升沿 |

如果故障控制和故障输入(n)被启用，并且在故障输入的选定边沿(n)被检测到，那么就发生了故障状况，FAULTFn位被设置。FAULTF位是FAULTFn[3:0]位的逻辑OR。见下图。

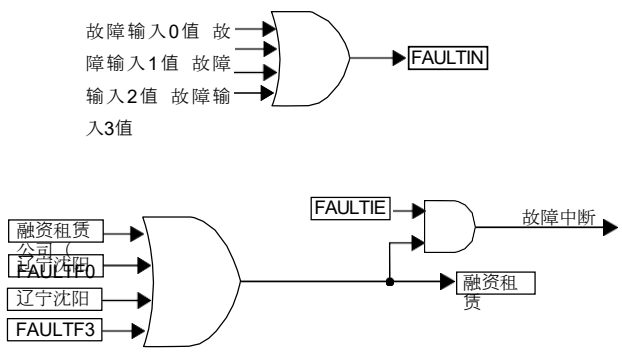



**图42-83.FAULTF和FAULTIN位和故障中断**

如果故障控制被启用（FAULTM[1:0]≠0:0），发生了故障情况，并且（FAULTENj = 1，其中j是通道的一对j），那么通道（n）和（n+1）的输出被强制到其安全值。

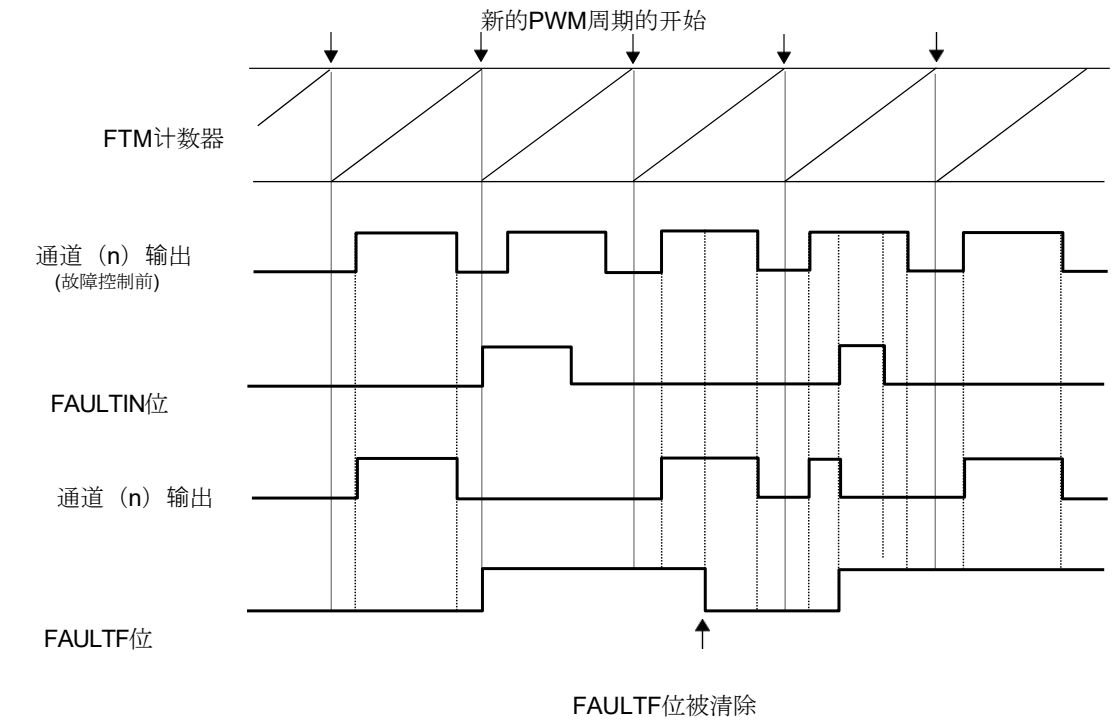
- 通道（n）的输出取POL（n）位值
- 通道(n+1)输出取POL(n+1)位值

当(FAULTF=1)和(FAULTIE=1)时，产生故障中断。这个中断请求一直被设置，直到。

- 软件通过读取FAULTF位为1并将其写入0来清除FAULTF位。
- 软件清除FAULTIE位
- 发生复位

### 42.5.18.1 自动清除故障

如果选择了自动清除故障(FAULTM[1:0] = 1:1)，那么当故障输入信号(FAULTIN)返回到零并且开始一个新的PWM周期时，由故障控制禁用的通道输出又被启用。见下图。

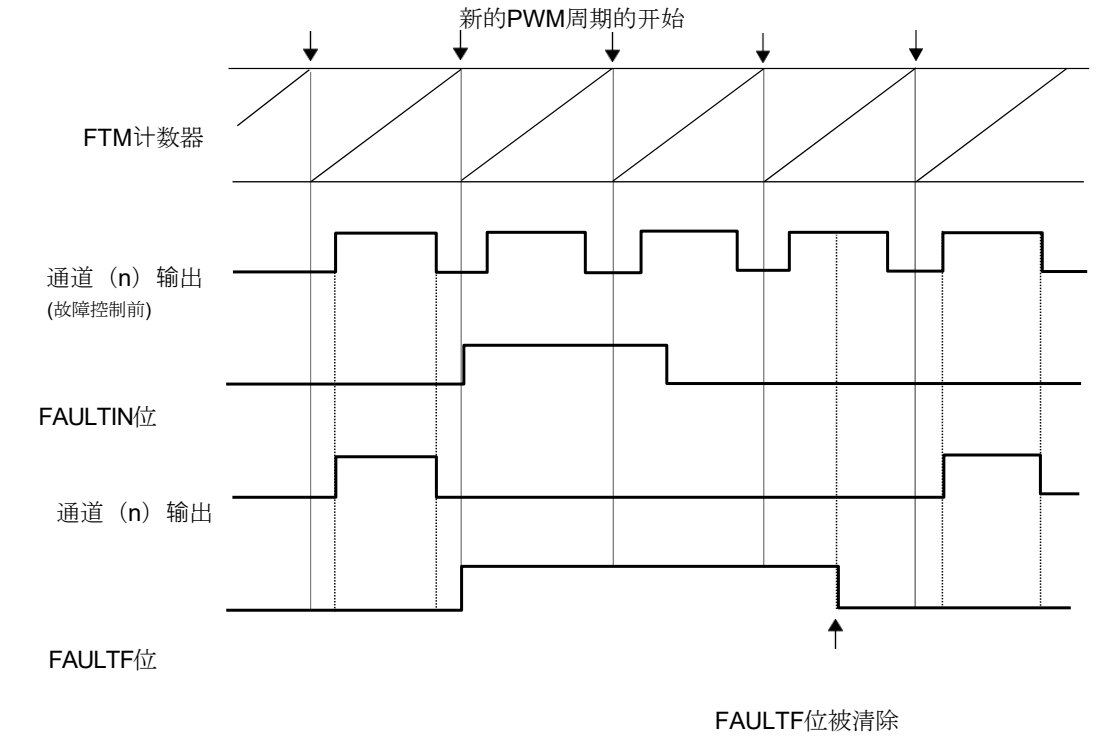



注意事项
通道（n）的输出是在故障控制后，自动清除故障，POLn=0。

**图42-84.具有自动清除故障功能的故障控制**

### 42.5.18.2 手动清除故障

如果选择了手动故障清除（FAULTM[1:0] = 0:1 或 1:0），那么当 FAULTF 位被清除，新的 PWM 周期开始时，由故障控制禁用的通道输出又被启用。见下图。




注意事项
通道（n）的输出是在故障控制后，手动清除故障，POLn=0。

**图42-85.带有手动故障清除功能的故障控制**

### 42.5.18.3 故障输入极性控制

FLTjPOL位选择故障输入j的极性，其中j=0、1、2、3。

- 如果FLTjPOL=0，故障j输入极性为高，所以故障输入j的逻辑1表示有故障。
- 如果FLTjPOL = 1，故障j输入极性为低，所以故障输入j的逻辑零表示有故障。

## 42.5.19 极性控制

POLn位选择通道（n）的输出极性。

- 如果POLn=0，通道（n）的输出极性为高，所以逻辑1为活动状态，逻辑0为非活动状态。
- 如果POLn=1，通道（n）的输出极性为低，所以逻辑零是活动状态，逻辑一是非活动状态。

## 42.5.20 初始化

当1被写入INIT位时，初始化将CH(n)OI位值强制到通道(n)输出。

初始化取决于COMP和DTEN位。下表显示了当COMP和DTEN位为0时，通道（n）和（n+1）被强制初始化的值。

**表42-16.当（COMP = 0和DTEN = 0）时的初始化行为**

| CH(n)OI | CH(n+1)OI | 通道(n) 输出 | 通道（n+1）输出 |
|---|---|---|---|
| 0 | 0 | 被迫为零 | 被迫为零 |
| 0 | 1 | 被迫为零 | 被逼到一个 |
| 1 | 0 | 被逼到一个 | 被迫为零 |
| 1 | 1 | 被逼到一个 | 被逼到一个 |

下表显示了当(COMP = 1)或(DTEN = 1)时，通道(n)和(n+1)通过初始化被强制的数值。

**表42-17.当（COMP = 1或DTEN = 1）时的初始化行为**

| CH(n)OI | CH(n+1)OI | 通道(n) 输出 | 通道（n+1）输出 |
|---|---|---|---|
| 0 | X | 被迫为零 | 被逼到一个 |
| 1 | X | 被逼到一个 | 被迫为零 |

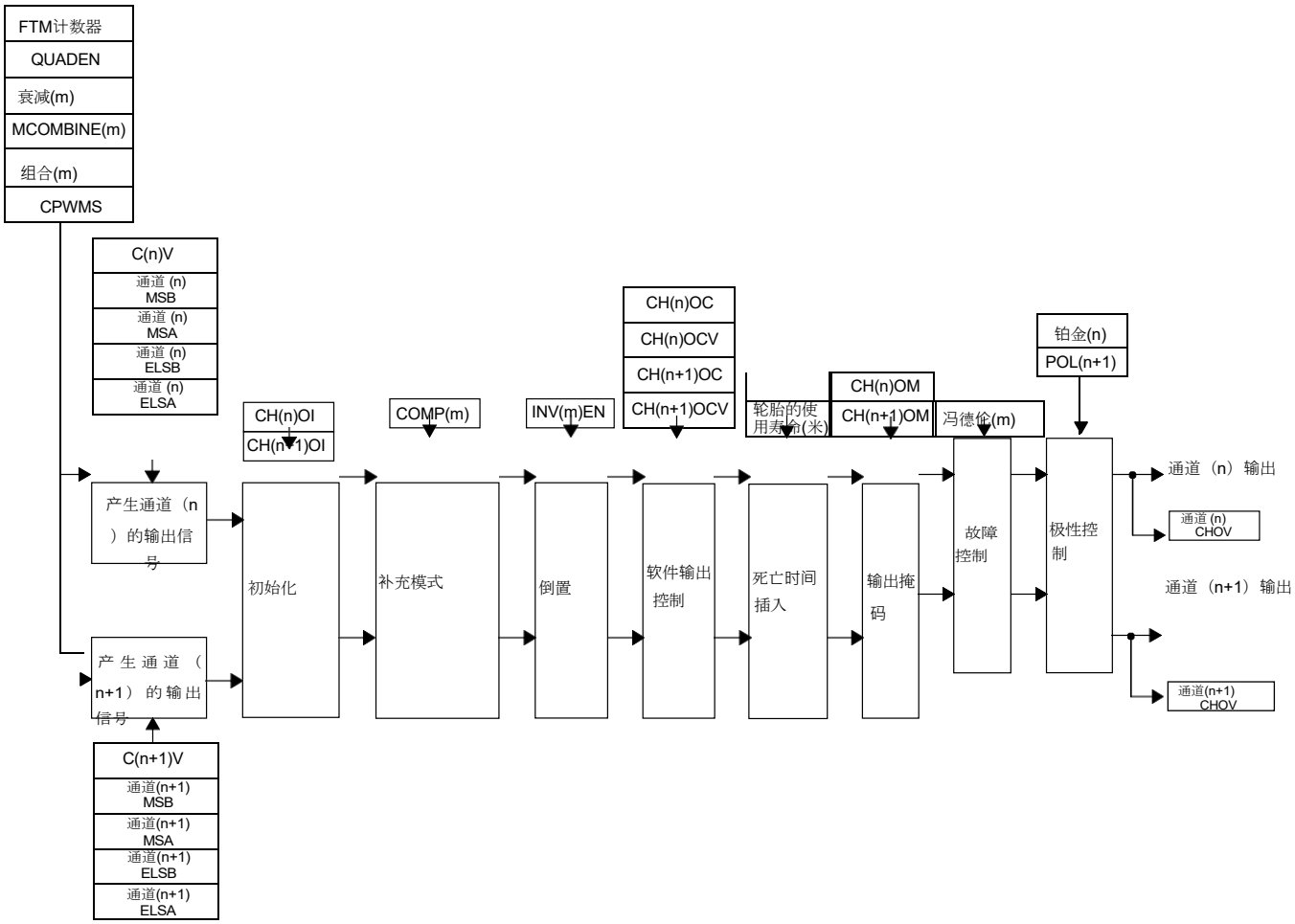
### 注意事项

初始化功能必须只在禁用FTM计数器的情况下使用。参见状态和控制寄存器中CLKS字段的描述。

## 42.5.21 特色优先

下图显示了通道生成时使用的功能的优先次序(n)和(n+1)输出信号。

配对通道（m）--通道（n）和（n+1）。




**请注意。**

通道（n）和（n+1）处于输出比较、EPWM、CPWM、组合或修改的组合PWM模式。

**图42-86.在生成通道（n）和（n+1）输出时使用的特征的优先级**

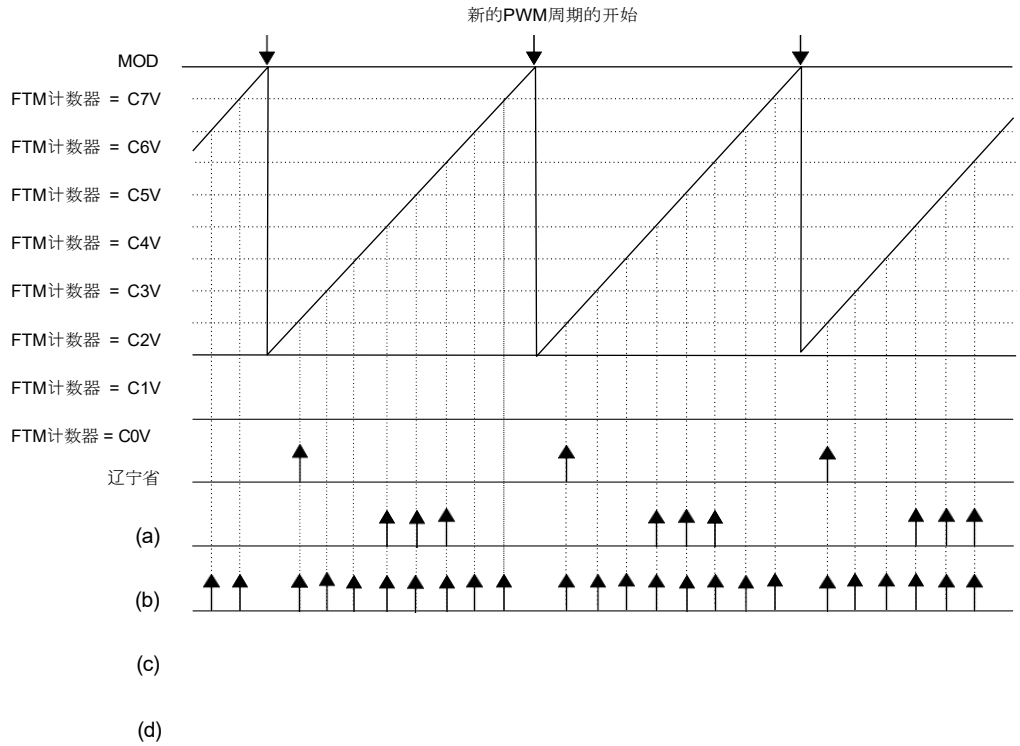
### 注意事项

初始化不得与反转和软件输出控制模式一起使用。

## 42.5.22 外部触发器

如果CH(n)TRIG位（寄存器EXTTRIG）被设置，其中n=0、1、2、3、4、5、6或7，那么当通道（n）匹配发生时（FTM计数器=C(n)V）在FTM外部触发输出产生一个触发。

一个通道（n）触发器的宽度是一个FTM输入时钟，FTM能够在一个PWM周期内产生多个触发器。见下图。




注意事项

(a) ch0trig = 0, ch1trig = 0, ch2trig = 0, ch3trig = 0, ch4trig = 0, ch5trig = 0, ch6trig = 0, ch7trig = 0
(b) ch0trig = 1, ch1trig = 0, ch2trig = 0, ch3trig = 0, ch4trig = 0, ch5trig = 0, ch6trig = 0, ch7trig = 0
(c) ch0trig = 0, ch1trig = 0, ch2trig = 0, ch3trig = 1, ch4trig = 1, ch5trig = 1, ch6trig = 0, ch7trig = 0
(d) ch0trig = 1, ch1trig = 1, ch2trig = 1, ch3trig = 1, ch4trig = 1, ch5trig = 1, ch6trig = 1, ch7trig = 1

**图42-87.外部触发器**

## 42.5.23 初始化触发器

初始化触发允许 FTM 在 FTM
计数器周期的某些特定点上产生一个触发。该功能由位INITTRIGEN和ITRIGR控制。INITTRIGEN位启用初始化触发器的产生，ITRIGR位选择初始化触发器产生的时间。

如果 INITTRIGEN = 1 和 ITRIGR = 1，那么根据重载机会的频率（重载点），当 FTM 计数器达到重载点时就会产生初始化触发。

**注意事项**

对于这种初始化触发的配置，在CPWM模式下，位CNTMAX和CNTMIN选择初始化触发产生的位置。

如果INITTRIGEN=1，ITRIGR=0，那么当FTM计数器用CNTIN寄存器的值更新时就会产生初始化触发。请看下面的情况。

1. 当FTM计数器以CNTIN寄存器值自动更新时。

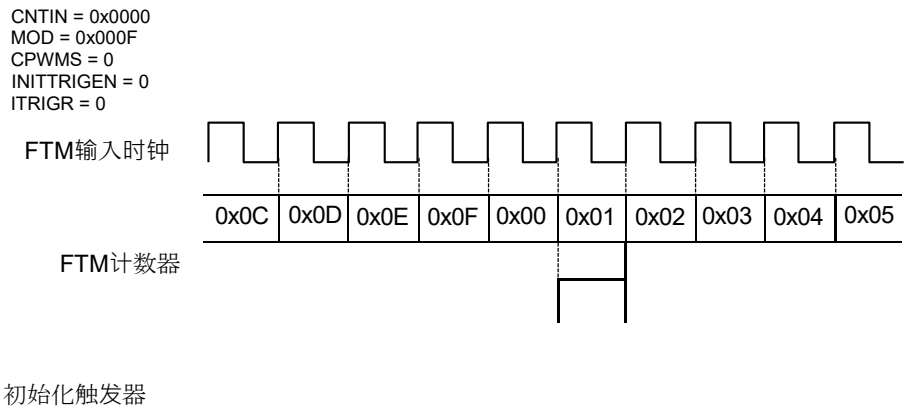



**图42-88.在情况1中生成初始化触发器的例子。**

2. 当有一个写到CNT寄存器时。

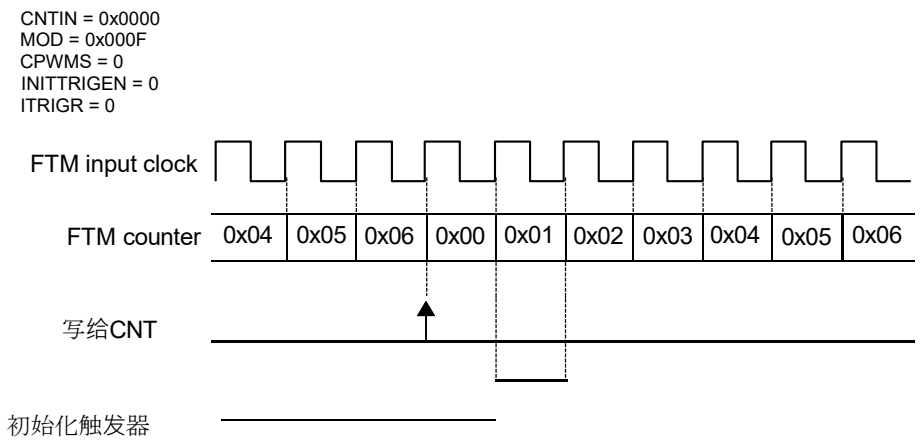



**图42-89.在情况2中生成初始化触发器的例子。**

**注意事项**

这种行为在CPWM模式下是不可用的。

3. 当有FTM计数器同步的时候。

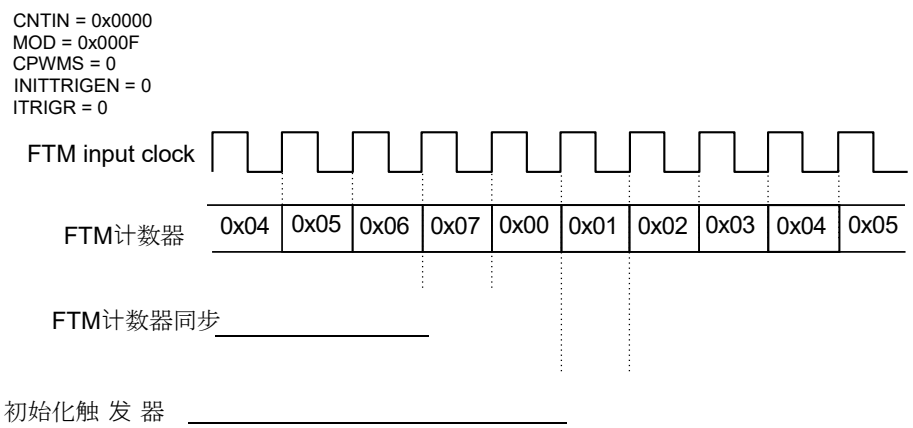


**图42-90.在情况3中生成初始化触发器的例子。**

**注意事项**

这种行为在CPWM模式下是不可用的。

4. 如果（CNT=CNTIN），（CLKS[1:0]=0:0），并且在CLKS[1:0]位上写入一个不同于零的值。

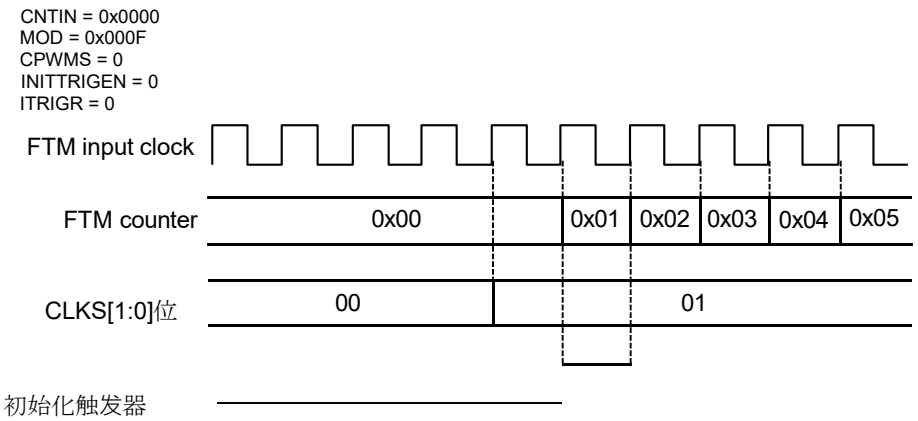



**图42-91.在情况4中生成初始化触发器的例子。**

### 注意事项

这种行为在CPWM模式下是不可用的。

5. 如果通道（n）处于输入捕捉模式，（ICRST=1）并且所选的输入捕捉事件发生在通道（n）的输入端。

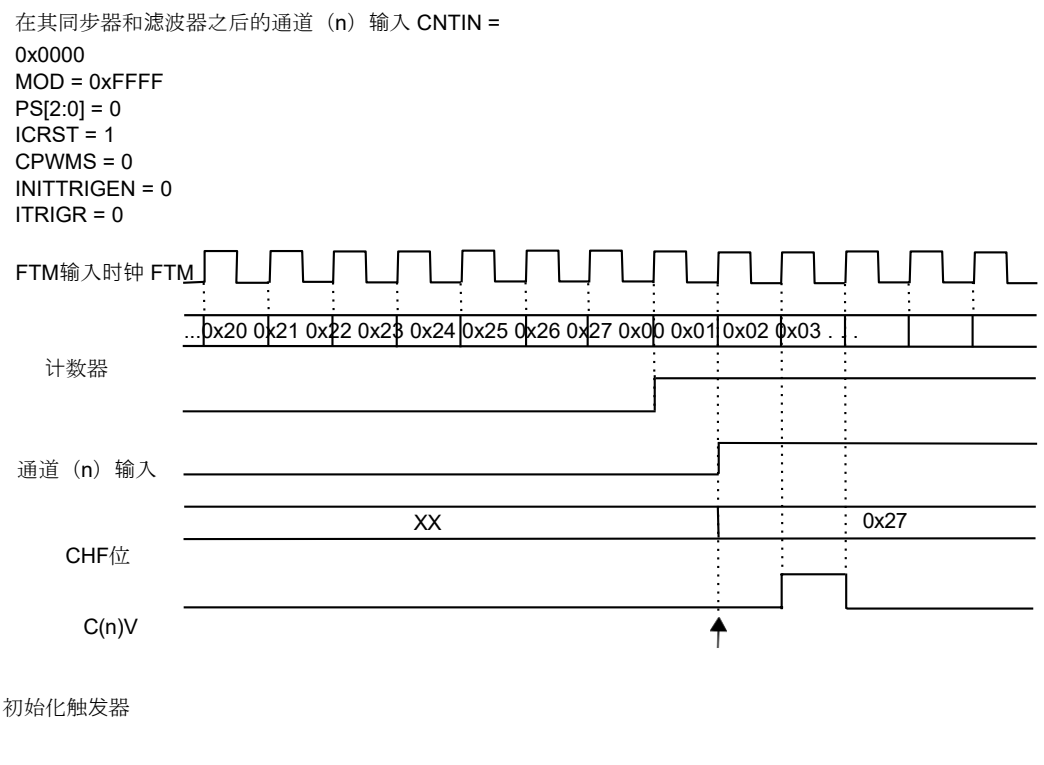


图42-92.案例5中初始化触发器的生成实例。

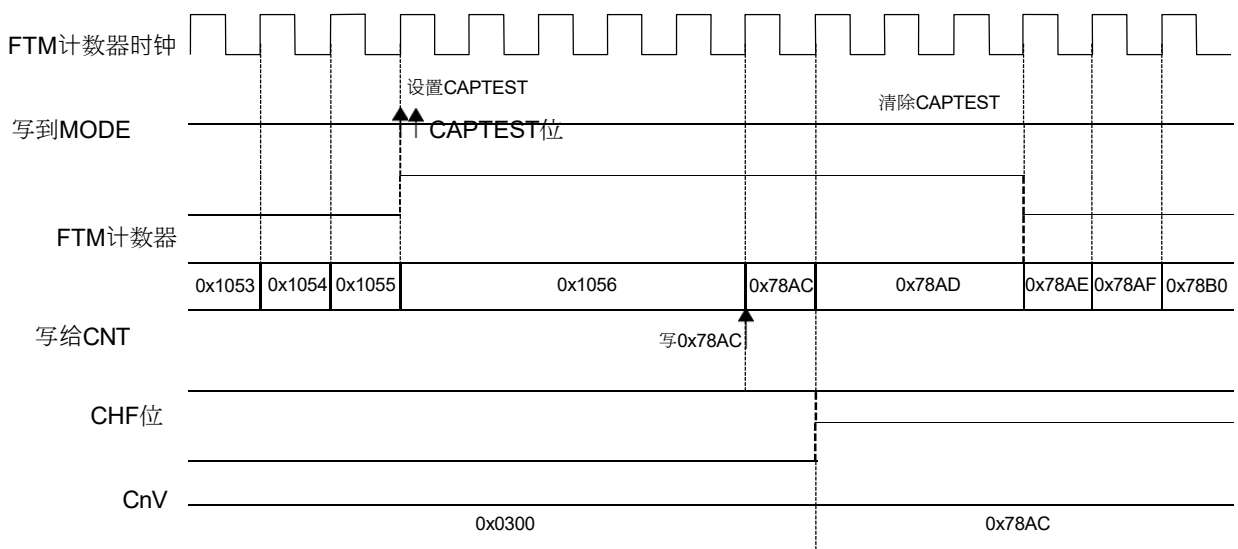
## 42.5.24 捕获测试模式

捕获测试模式允许测试CnV寄存器、FTM计数器以及FTM计数器和CnV寄存器之间的互连逻辑。

在这种测试模式下，所有通道必须配置为输入捕捉模式，FTM计数器必须配置为向上计数。

当捕获测试模式被启用（CAPTEST =
1）时，FTM计数器被冻结，任何对CNT寄存器的写入都直接更新FTM计数器；见下图。
写完后，所有的 CnV 寄存器都被更新为 CNT 寄存器的写入值，CHF
位被设置。因此，FTM
计数器根据其配置更新其下一个值。它的下一个值取决于CNTIN、MOD和写入FTM计数器的值。

对CnV寄存器的下一次读取会返回FTM计数器的写入值，对CNT寄存器的下一次读取会返回FTM计数器的下一个值。




请注意。

- FTM计数器处于自由运行状态
- FTMEN = 1
- FTM通道（n）处于输入捕捉模式

**图42-93.捕获测试模式**

## 42.5.25 DMA

通道根据DMA和CHIE位产生一个DMA传输请求。见下表。

表42-18.通道DMA传输请求

| DMA | 芝华士 | 通道DMA传输请求 | 通道中断 |
|---|---|---|---|
| 0 | 0 | 不产生通道DMA传输请求。 | 通道中断不被产生。 |
| 0 | 1 | 不产生通道DMA传输请求。 | 如果(CHF = 1)，则产生通道中断。 |
| 1 | 0 | 不产生通道DMA传输请求。 | 通道中断不被产生。 |
| 1 | 1 | 如果(CHF = 1)，则产生通道DMA传输请求。 | 通道中断不被产生。 |

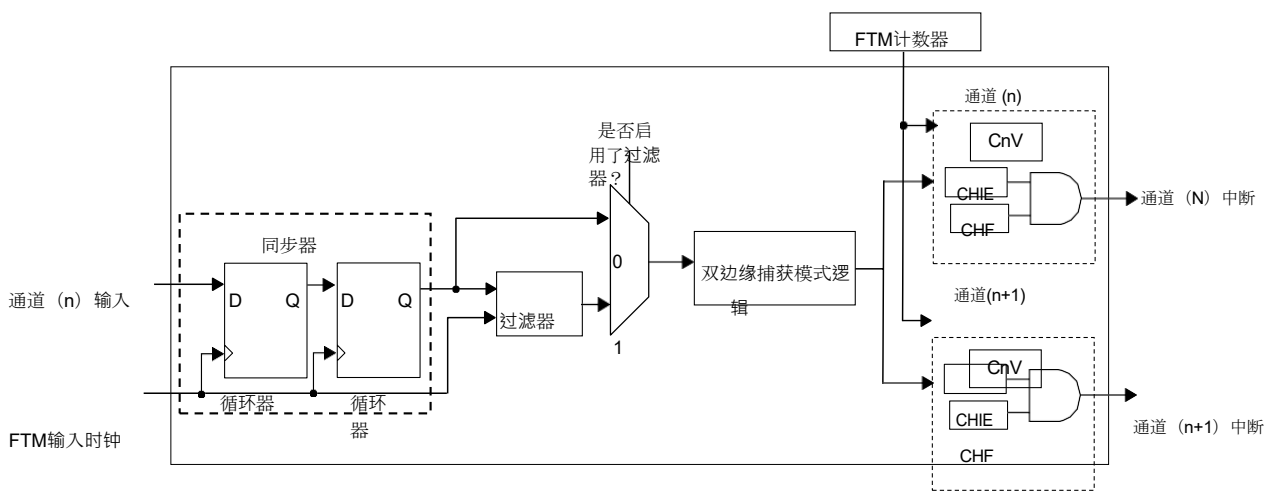
如果DMA=1，CHF位被清除，要么通过通道DMA传输完成，要么在CHF被设置时读取CnSC，然后根据CHIE位向CHF位写0。见下表。

表42-19.当DMA=1时清除CHF位

| 芝华士 | 如何清除CHF位 |
|---|---|
| 0 | CHF位在通道DMA传输完成后被清除，或者在CHF被设置时读取CnSC，然后向CHF位写0。 |
| 1 | 当通道DMA传输完成后，CHF位被清零。 |

## 42.5.26 双边缘捕捉模式

如果DECAPEN=1，则启用双边沿捕获模式。这种模式允许测量通道（n）输入的脉冲宽度或周期，其中n=0、2、4或6。对于n=0或2，通道(n)滤波器可以被启用。



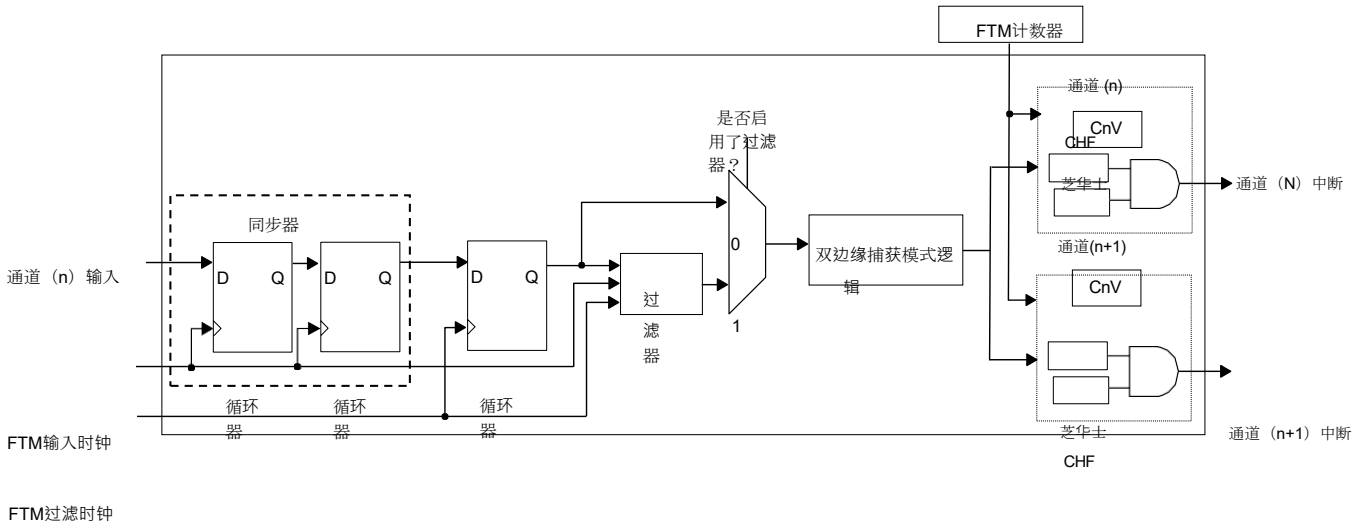



**图42-95.当FLTPS[3:0]≠0时的双边缘捕获模式示意图**

通道(n)MSA位定义了双边缘捕获模式是单发还是连续。

通道(n)ELSB:ELSA位选择由通道(n)捕获的边缘，通道(n+1)ELSB:ELSA位选择由通道(n+1)捕获的边缘。如果通道(n)ELSB:ELSA和通道(n+1)ELSB:ELSA位都选择相同的边，那么它就是周期测量。如果这些位选择不同的边沿，那么就是脉冲宽度测量。

在双边缘捕获模式下，只使用通道（n）的输入，而忽略通道（n+1）的输入。

如果在通道(n)输入处检测到通道(n)位的选择边沿，那么通道(n)CHF位被设置，并产生通道(n)中断（如果通道(n)CHIE =
1）。如果在通道(n)输入处检测到通道(n+1)位的选择边缘，并且(通道(n)CHF
=
1），那么通道（n+1）CHF位被设置，通道（n+1）中断被产生（如果通道（n+1）CHIE = 1）。

当通道(n)输入端检测到通道(n)的选择边沿时，C(n)V寄存器存储FTM计数器的值。C(n+1)V
寄存器存储FTM计数器的值，当通道(n+1)的选择边沿在通道(n)输入处被检测到。

在这种模式下，当C(n)V和C(n+1)V寄存器被读取时，一致性机制（针对通道（n）和（n+1））确保了数据的一致性。唯一的要求是，C(n)V必须在C(n+1)V之前被读取。

## 注意事项

- 双边缘捕获模式必须在通道(n)ELSB:ELSA=0:1或1:0通道(n+1)ELSB:ELSA=0:1的情况下使用。
  0:1或1:0，自由运行计数器中的FTM计数器。

### 42.5.26.1 单次拍摄模式

当(DECAPEN = 1)和(通道(n)MSA = 1)时，选择单次捕获模式。
0).在这种捕获模式下，只有通道（n）输入的一对边沿被捕获。通道(n)ELSB:ELS
A位选择要捕获的第一个边，通道(n+1)ELSB:ELSA位选择要捕获的第二个边。

当 DECAP
位被设置时，边缘捕获被启用。对于单次捕获模式下的每个新测量，首先必须清除通道（n）CHF 和通道（n+1）CHF 位，然后必须设置 DECAP 位。

在这种模式下，当通道(n+1)选择的边沿被捕获时，FTM 自动清除 DECAP
位。因此，当 DECAP
位被设置时，单次捕获正在进行中。当该位被清除时，两个边沿都被捕获，捕获的值可以在C(n)V和C(n+1)V寄存器中读取。

同样，当通道(n+1)CHF位被设置时，两个边沿都被捕获，捕获的值可以在C(n)V和C(n+1)V寄存器中读取。

### 42.5.26.2 连续拍摄模式

当(DECAPEN = 1)和(通道(n)MSA)时，选择连续捕捉模式。
=
1).在这种捕获模式下，通道（n）输入的边沿被连续捕获。通道(n)ELSB:ELSA位选择要捕获的初始边，通道(n+1)ELSB:ELSA位选择要捕获的最终边。

当DECAP位被设置时，边缘捕获被启用。对于初始使用，首先必须清除通道（n）CHF和通道（n+1）CHF位，然后必须设置DECAP位以开始连续测量。

当通道(n+1)CHF位被设置时，两个边沿都被捕获，捕获的值可以在C(n)V和C(n+1)V寄存器中读取。最新的捕获值在这些寄存器中总是可用的，即使在DECAP位被清除之后。

在这种模式下，可以只清除通道(n+1)的CHF位。因此，当通道(n+1)CHF位再次被设置时，最新的捕获值在C(n)V和C(n+1)V寄存器中可用。

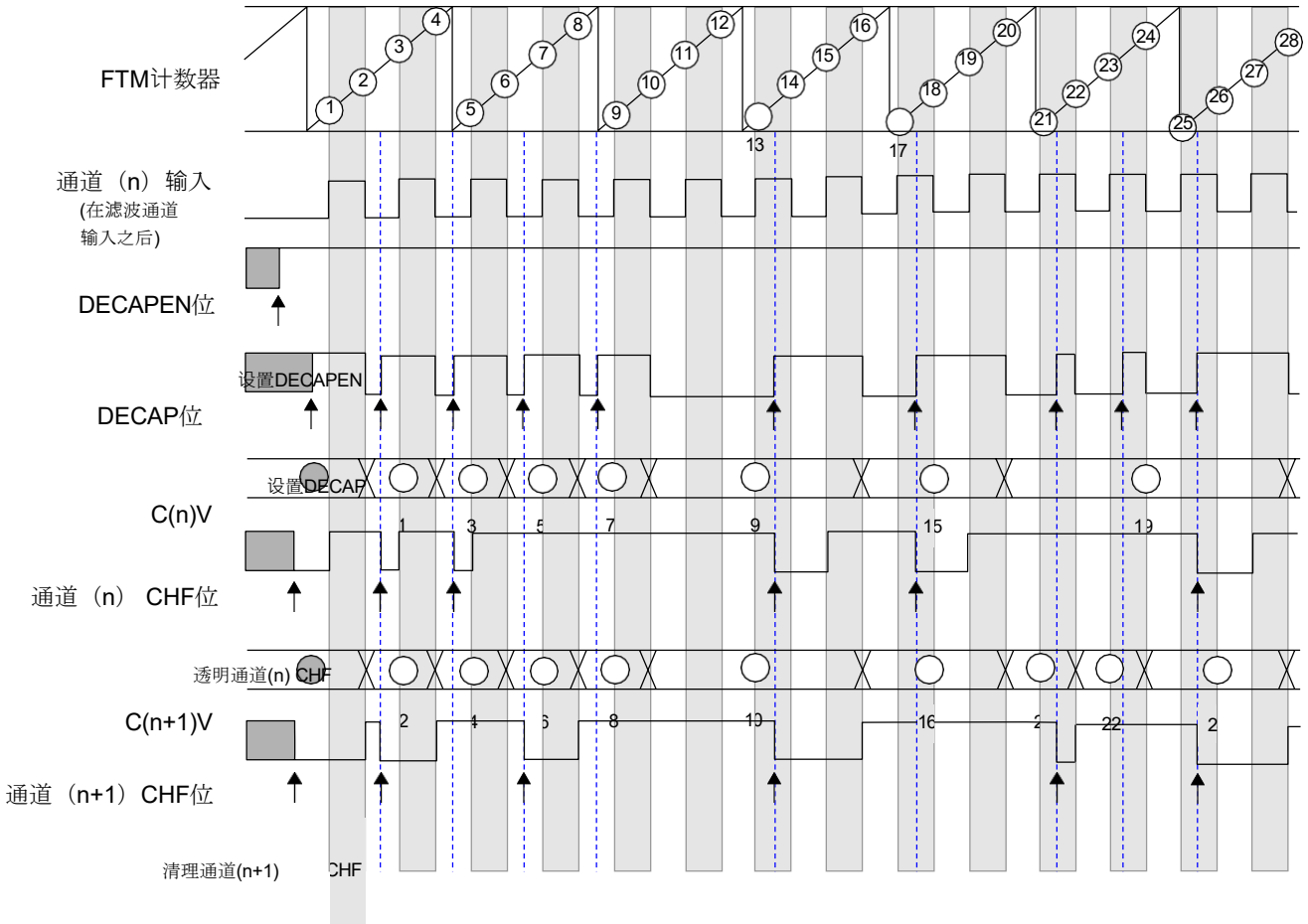
对于双边缘捕获-
连续模式下的新的测量序列，清除通道（n）CHF和通道（n+1）CHF位以开始新的测量。

### 42.5.26.3 脉冲宽度测量

如果通道（n）被配置为捕获上升沿（通道（n）ELSB:ELSA=0:1），通道（n+1）捕获下降沿（通道（n+1）ELSB:ELSA=1:0），则测量正极性脉冲宽度。如果通道（n）被配置为捕获下降沿（通道（n）ELSB:ELSA = 1:0），而通道（n+1）捕获上升沿（通道（n+1）ELSB:ELSA = 0:1），则测量负极性脉宽。

脉宽测量可以在单次捕获模式或连续捕获模式下进行。

下图显示了一个用于测量正极性脉冲宽度的双边沿捕获-单次拍摄模式的例子。DECAPEN位选择了双边缘捕获模式，所以它一直被设置。DECAP位被设置，以启用下一个正极性脉冲宽度的测量。当检测到这个脉冲的第一个边沿时，即通道（n）ELSB:ELSA 位所选择的边沿，通道（n）CHF 位被置位。当检测到这个脉冲的第二个边沿时，通道(n+1)CHF 位被置位，DECAP 位被清零，即通道(n+1) ELSB:ELSA 位所选择的边沿。DECAP和通道(n+1)CHF位都表示当脉冲的两个边沿被捕获时，C(n)V和C(n+1)V寄存器就可以读取了。

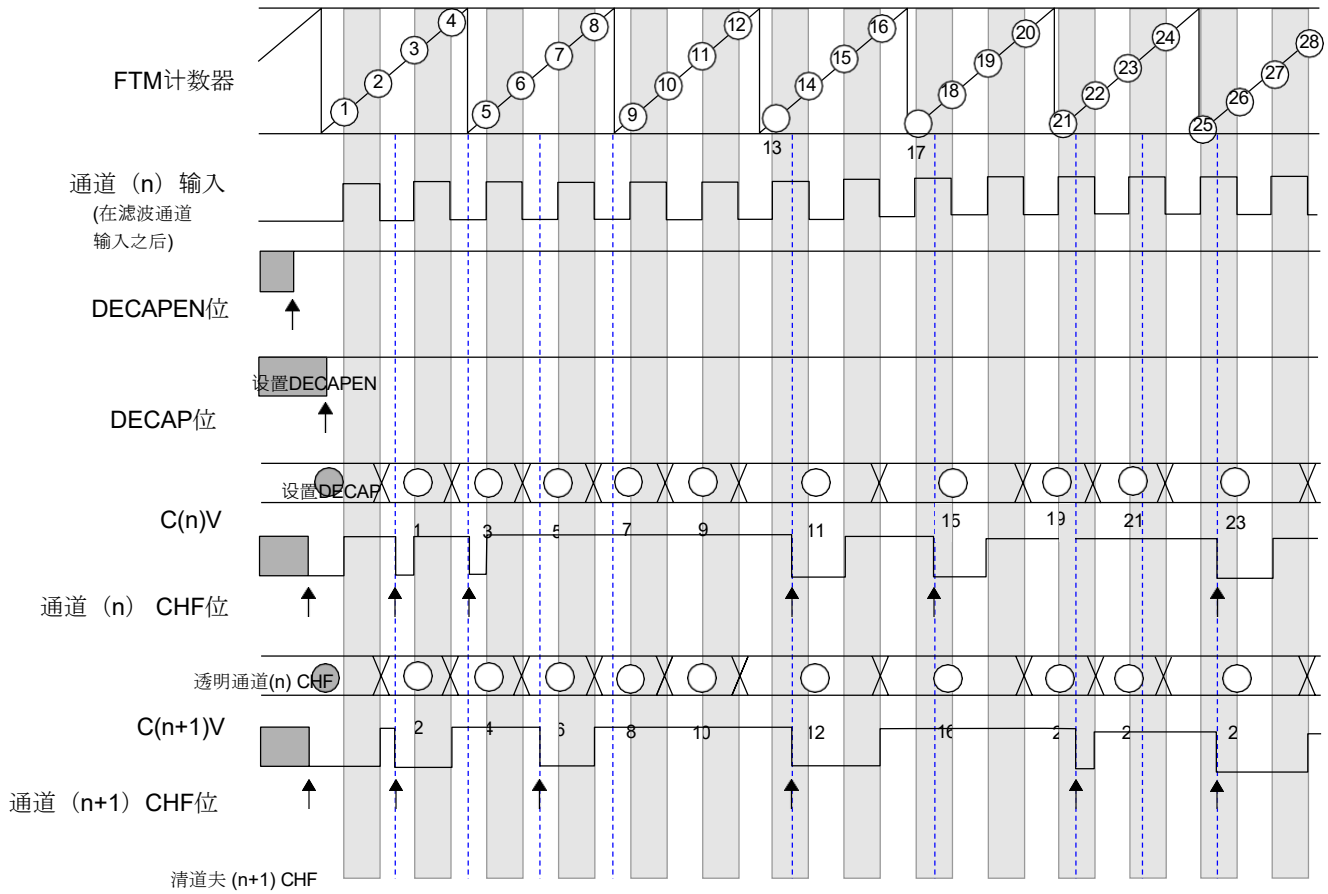



注意事项

- 设置DECAPEN、设置DECAP、清除通道(n)CHF和清除通道(n+1)CHF的命令是由用户作出。
- 问题1：通道（n）输入=1，设置DECAP，不清除通道（n）CHF，清除通道（n+1）CHF。
- 问题2：通道（n）输入=1，设置DECAP，不清除通道（n）CHF，也不清除通道（n+1）CHF。

**图42-96.双边缘捕获 - 用于正极性脉宽测量的单次拍摄模式**

下图显示了一个用于测量正极性脉冲宽度的双边沿捕获-
连续模式的例子。DECAPEN位选择了双边缘捕获模式，所以它一直被设置。当DECAP位被设置时，配置的测量被进行。当检测到正极性脉冲的第一个边沿时，通道(n)CHF位被设置，即通道(n)ELSB:ELSA位选择的边沿。当检测到该脉冲的第二个边沿时，通道(n+1)CHF位被设置，即由通道(n+1)ELSB:ELSA位选择的边沿。通道(n+1)CHF位表示当脉冲的两个边沿被捕获时，C(n)V和C(n+1)V寄存器就可以读取了。




注意事项

- 设置DECAPEN、设置DECAP、清除通道(n)CHF和清除通道(n+1)CHF的命令是由用户作出。

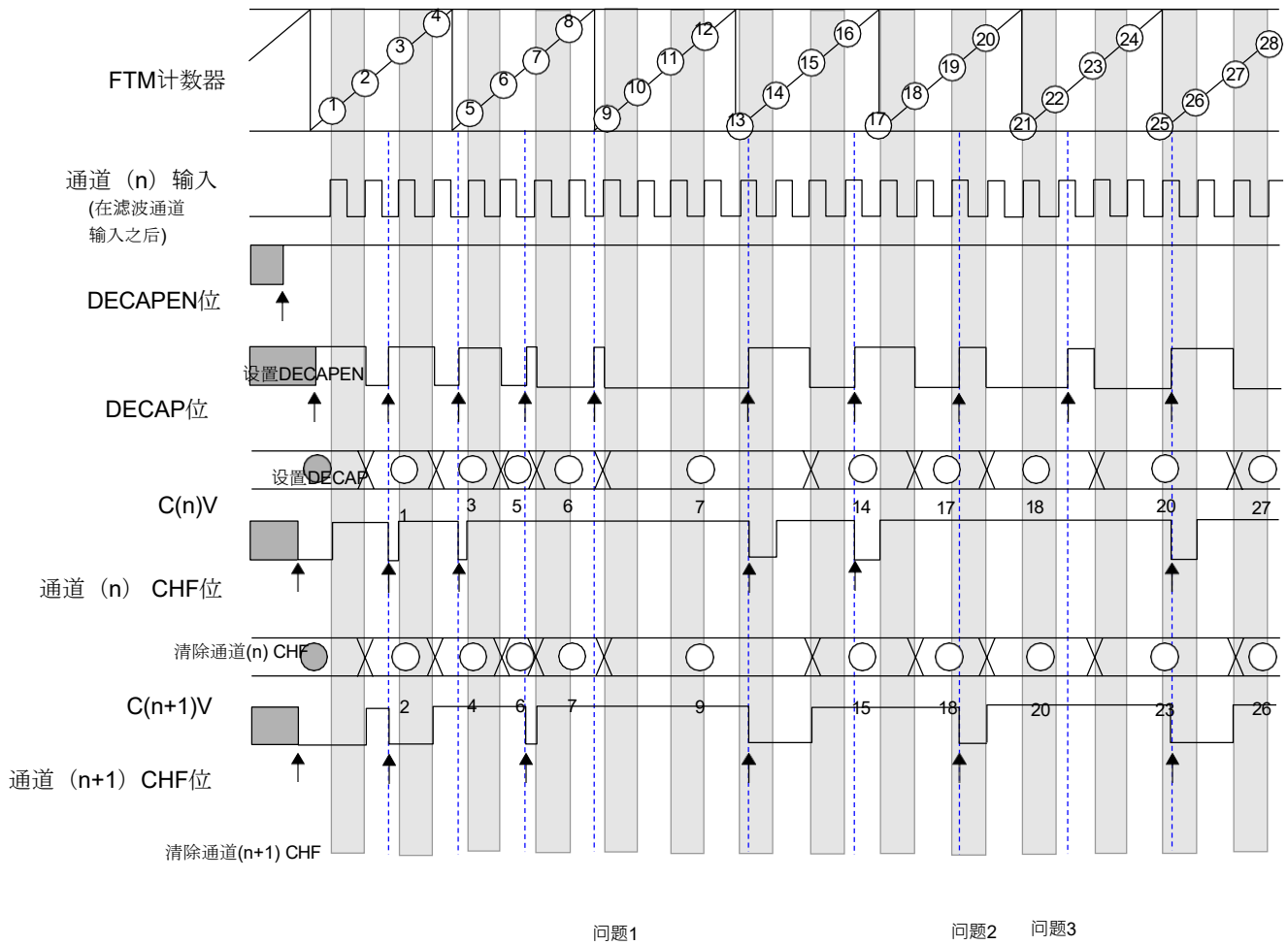
**图42-97.双边缘捕获 - 用于正极性脉冲宽度测量的连续模式**

## 42.5.26.4 期间测量

如果通道(n)和(n+1)被配置为捕获相同极性的连续边沿，那么通道(n)输入信号的周期被测量。如果通道（n）和（n+1）都被配置为捕获上升沿（通道（n）ELSB:ELSA=0:1和通道（n+1）ELSB:ELSA=0:1），那么测量两个连续上升沿之间的周期。如果通道(n)和(n+1)都被配置为捕获下降沿(通道(n)ELSB:ELSA=1:0和通道(n+1)ELSB:ELSA=1:0)，那么就测量两个连续下降沿之间的周期。

周期测量可以在单次拍摄模式或连续拍摄模式下进行。

下图显示了一个双边缘捕获-
单次拍摄模式的例子，用于测量两个连续上升沿之间的周期。DECAPEN位选择了双边沿捕获模式，所以它一直被设置。DECAP位被设置以启用下一个周期的测量。当检测到第一个上升沿时，即由通道(n)ELSB:ELSA 位选择的边沿，通道(n)CHF 位被置位。当检测到第二个上升沿时，通道(n+1)CHF 位被置位，DECAP 位被清零，即由通道(n+1)ELSB:ELSA
位选择的边沿。DECAP和通道(n+1)CHF位都表示当两个选定的边沿被捕获时，C(n)V和C(n+1)V寄存器就可以读取了。




注意事项

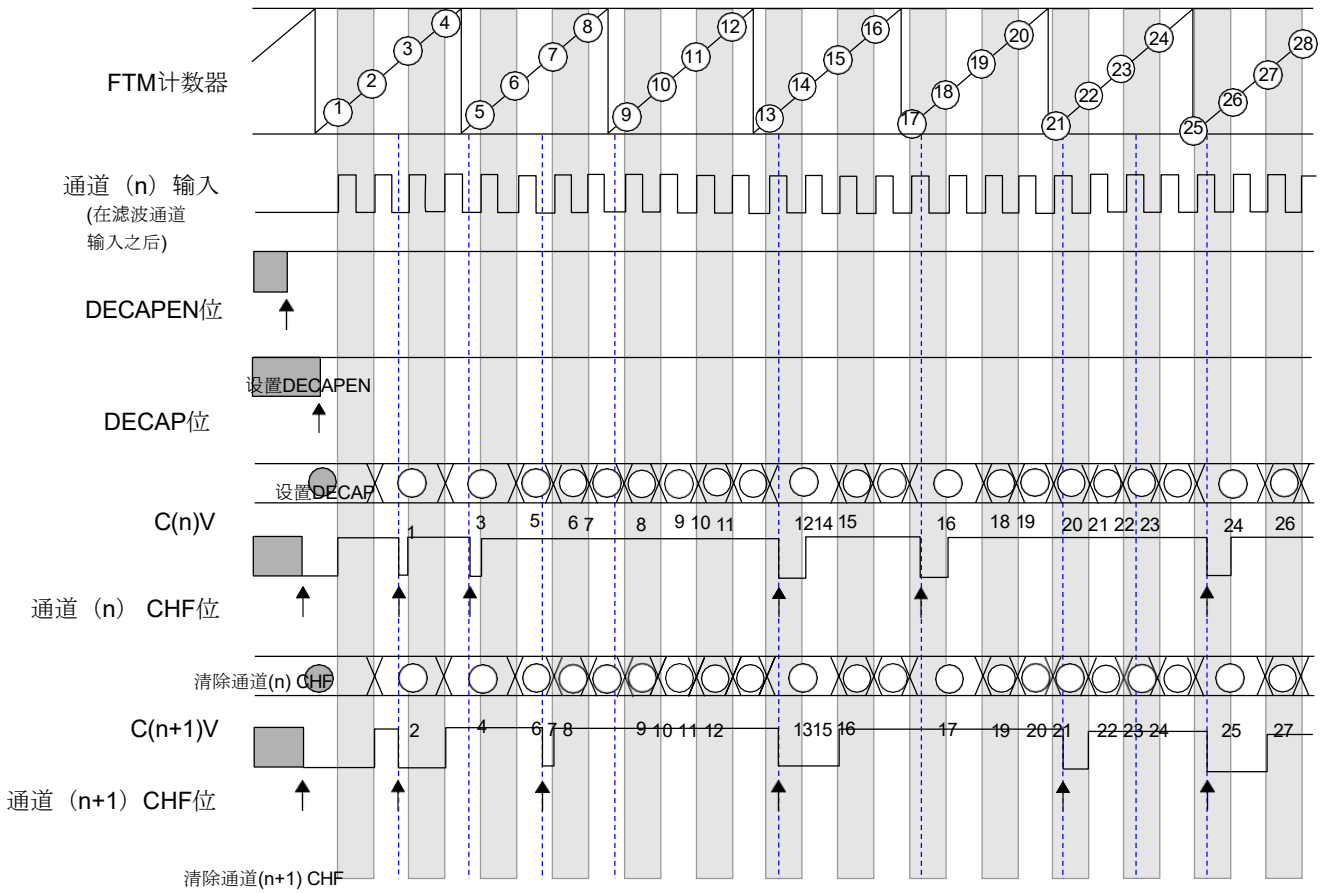
- 设置DECAPEN、设置DECAP、清除通道(n)CHF和清除通道(n+1)CHF的命令是由用户作出。
- 问题1：通道（n）输入=0，设置DECAP，不清除通道（n）CHF，也不清除通道（n+1）CHF。
- 问题2：通道（n）输入=1，设置DECAP，不清除通道（n）CHF，清除通道（n+1）CHF。
- 问题3：通道（n）输入=1，设置DECAP，不清除通道（n）CHF，也不清除通道（n+1）CHF。

**图42-98.双边缘捕获 - 单次拍摄模式，测量两个连续上升沿之间的周期**

下图显示了一个双边缘捕获-

连续模式的例子，用于测量两个连续上升沿之间的周期。DECAPEN位选择了双边沿捕获模式，所以它一直被设置。当DECAP位被设置时，配置的测量被进行。通道（n）CHF位被设置，当第一个上升沿

当检测到第二个上升沿时，即由通道(n)ELSB:ELSA
位选择的边沿。当检测到第二个上升沿时，通道(n+1)CHF位被设置，也就是由通道(n+1)ELSB:ELSA位选择的边沿。通道(n+1)CHF位表示当周期的两个边沿被捕获时，C(n)V和C(n+1)V寄存器就可以被读取。




注意事项

- 设置DECAPEN、设置DECAP、清除通道(n)CHF和清除通道(n+1)CHF的命令是由用户作出。

**图42-99.双边缘捕获 - 连续模式，测量两个连续上升沿之间的周期**

## 42.5.26.5 读取一致性机制

双边缘捕获模式在C(n)V和C(n+1)V寄存器中捕获的FTM计数器值之间实现了一个读一致性机制。下图说明了读一致性机制。在这个例子中，通道（n）和（n+1）处于双边缘捕获-
连续模式，用于正极性脉宽测量。因此，通道（n）被配置为在通道（n）输入信

号有上升沿时捕获FTM计数器值，而通道（n+1）在通道（n）输入信号有下降沿时捕获FTM计数器值。

当通道（n）输入信号出现上升沿时，FTM计数器的值被捕获到通道（n）捕获缓冲器中。当通道(n)输入信号发生下降沿时，通道(n)捕获缓冲器的值被转移到C(n)V寄存器。C(n)V 寄存器有前一个上升沿发生时的 FTM
计数器值，而通道(n)捕获缓冲器有最后一个上升沿发生时的 FTM 计数器值。

当通道（n）输入信号出现下降沿时，FTM计数器的值被捕获到通道（n+1）捕获缓冲器。当C(n)V寄存器被读取时，通道(n+1)捕获缓冲器的值被转移到C(n+1)V寄存器。

在下图中，读取C(n)V返回事件1发生时的FTM计数器值，读取C(n+1)V返回事件2发生时的FTM计数器值。

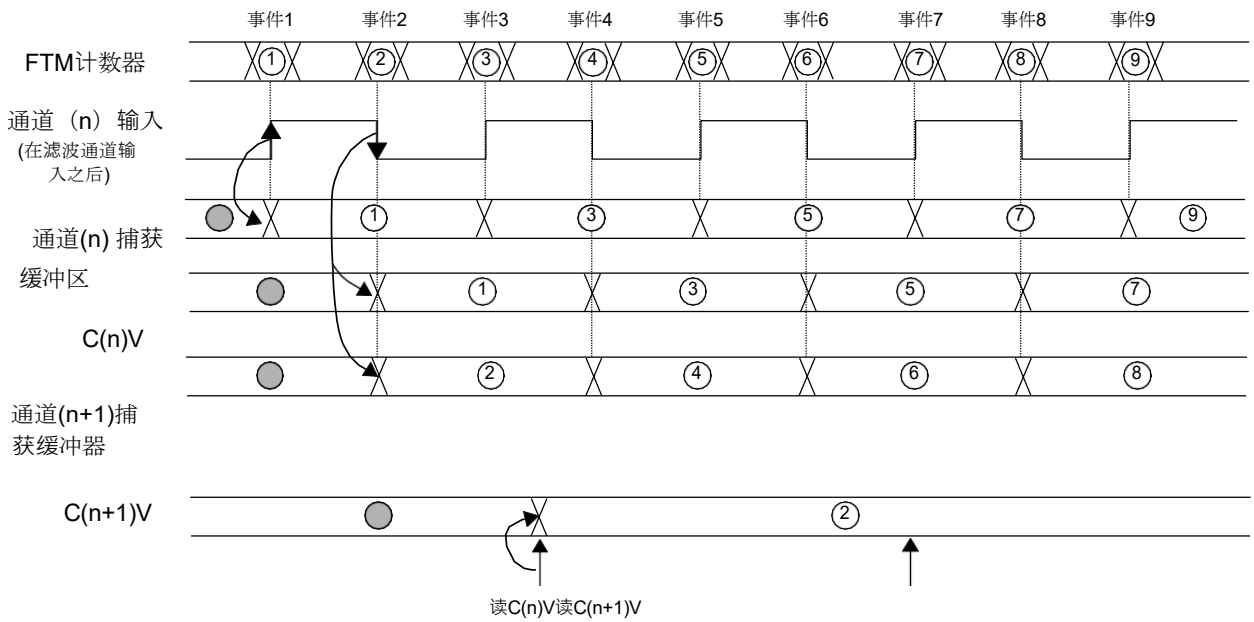



**图42-100.双边缘捕获模式的读取一致性机制**

C(n)V寄存器必须先于C(n+1)V寄存器被读取，在双边缘捕获单次和连续模式下，读取一致性机制才能正常工作。

## 42.5.27 正交解码器模式

如果 QUADEN =
1，则启用正交解码器模式。正交解码器模式使用输入信号A相和B相来控制FTM计数器的增量和减量。

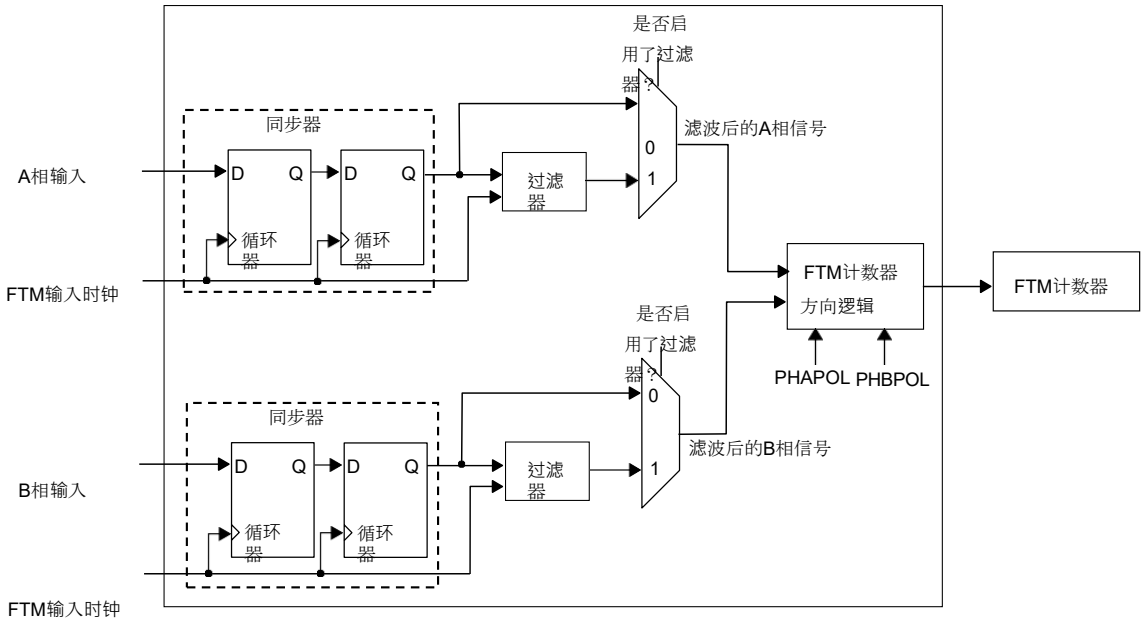



**图42-101.FLTPS[3:0]=0时正交解码器示意图**

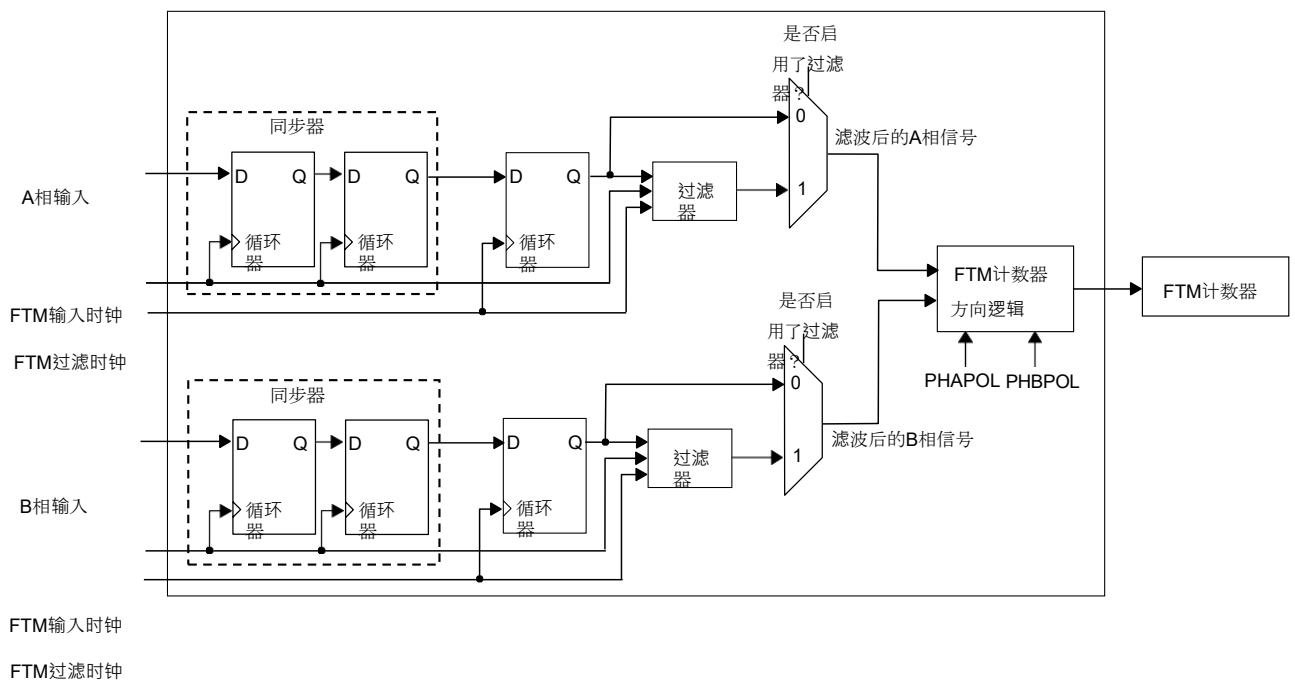



**图42-102.FLTPS[3:0]≠0时的正交解码器示意图**

每一个输入信号A相和B相都有一个滤波器，相当于通道输入滤波器（输入捕捉模式的滤波器）。A相输入滤波器由PHAFLTREN位启用，其值由CH0FVAL[3:0]位定义。B相输入滤波器由PHBFLTREN位启用，其值由CH1FVAL[3:0]位定义。

除了CH0FVAL[3:0]和CH1FVAL[3:0]位，在正交解码器模式下没有使用通道逻辑。

### 注意事项

当启用正交解码器模式时，FTM 计数器由 A 和 B 相输入信号提供时钟。因此，预计正交解码器模式只能在输入捕捉或输出比较模式下的FTM通道使用。

### 注意事项

A相的边不能与B相的边同时出现，反之亦然。

PHAPOL位选择A相输入的极性，PHBPOL位选择B相输入的极性。

QUADMODE选择正交解码器模式中使用的编码模式。如果QUADMODE = 1，则启用计数和方向编码模式；见下图。在这种模式下，B相输入值表示计数方向，A相输入定义了计数率。当A相输入信号有一个上升沿时，FTM计数器被更新。

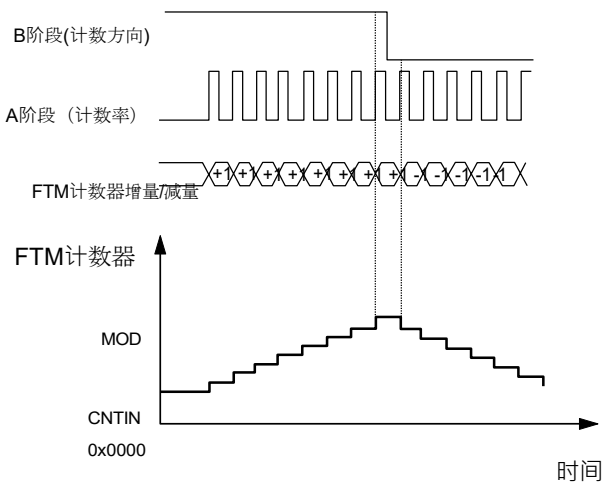



**图42-103.正交解码器 - 计数和方向编码模式**

如果QUADMODE = 0，则启用A相和B相编码模式；见下图。在这种模式下，A相和B相信号之间的关系表示计数方向，A相和B相信号定义计数速率。当A相或B相信号有一个边沿时，FTM计数器被更新。

如果PHAPOL=0，PHBPOL=0，那么FTM计数器的增量发生在以下情况。

- A相信号有一个上升沿，B相信号为逻辑零。
- B相信号有一个上升沿，A相信号为逻辑1。

- B相信号有一个下降沿，A相信号为逻辑零。
- A相信号有一个下降沿，B相信号为逻辑1。

而FTM计数器的递减发生在：。

- A相信号有一个下降沿，B相信号为逻辑零。
- B相信号有一个下降沿，A相信号为逻辑1。
- B相信号有一个上升沿，A相信号为逻辑零。
- A相信号有一个上升沿，B相信号为逻辑一。

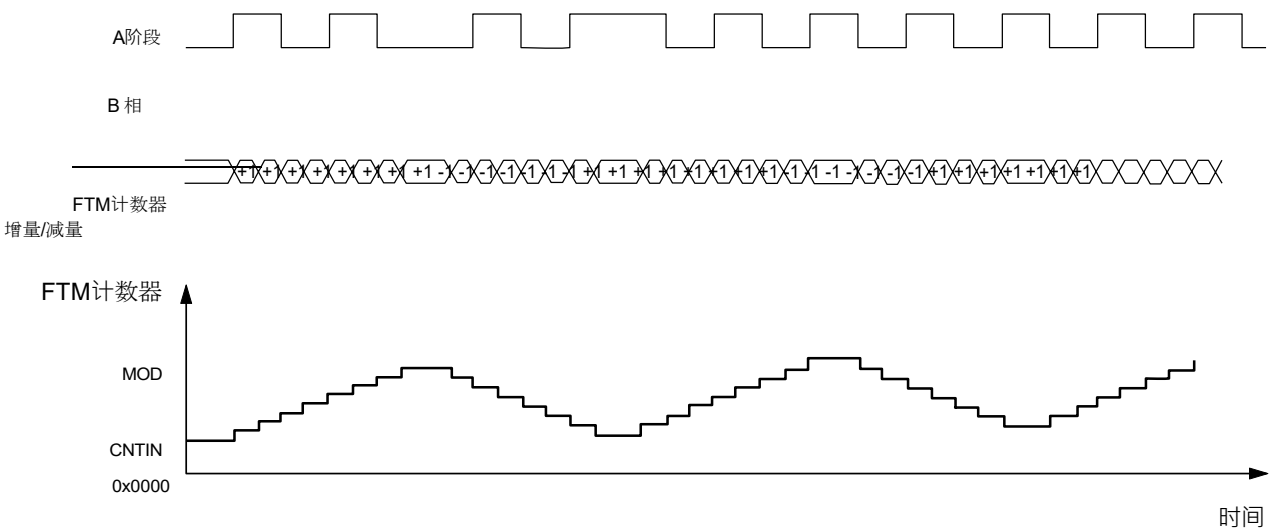



**图42-104.正交解码器 - A相和B相编码模式**

下图显示了FTM计数器在向上计数时的溢出情况。在这种情况下，当 FTM 计数器从 MOD 变为 CNTIN 时，TOF 和 TOFDIR 位被设置。TOF 位表示 FTM 计数器发生溢出。TOFDIR 表示当 FTM 计数器溢出发生时，计数是向上的。

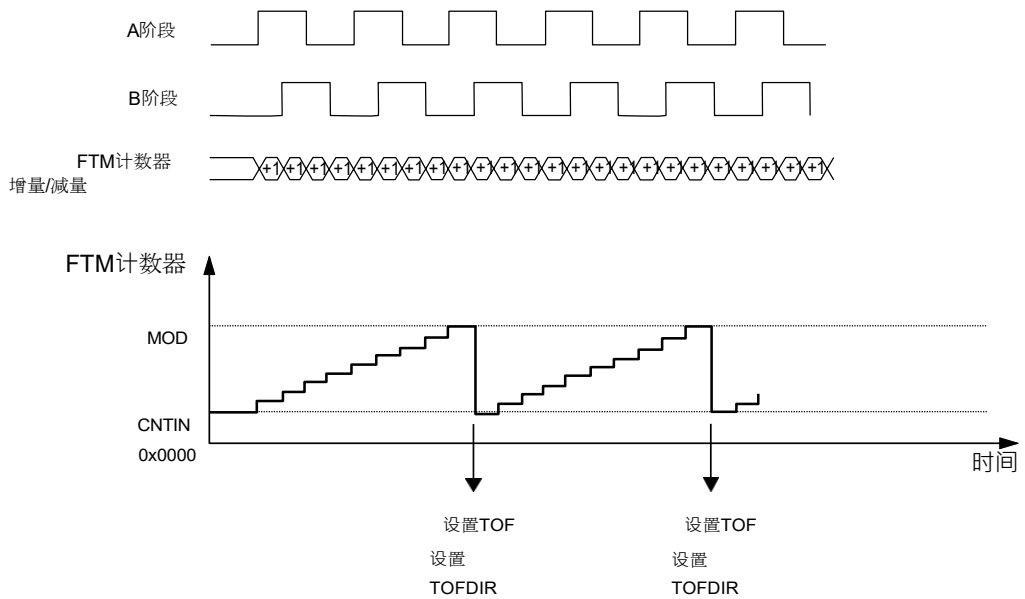



**图42-105.正交解码器模式的上行计数中FTM计数器溢出**

下图显示了下行计数中的 FTM 计数器溢出。在这种情况下，当 FTM 计数器从 CNTIN 变为 MOD 时，TOF 位被设置，TOFDIR 位被清零。TOF 位表示 FTM 计数器发生溢出。TOFDIR 表示当 FTM 计数器溢出发生时，计数正在下降。

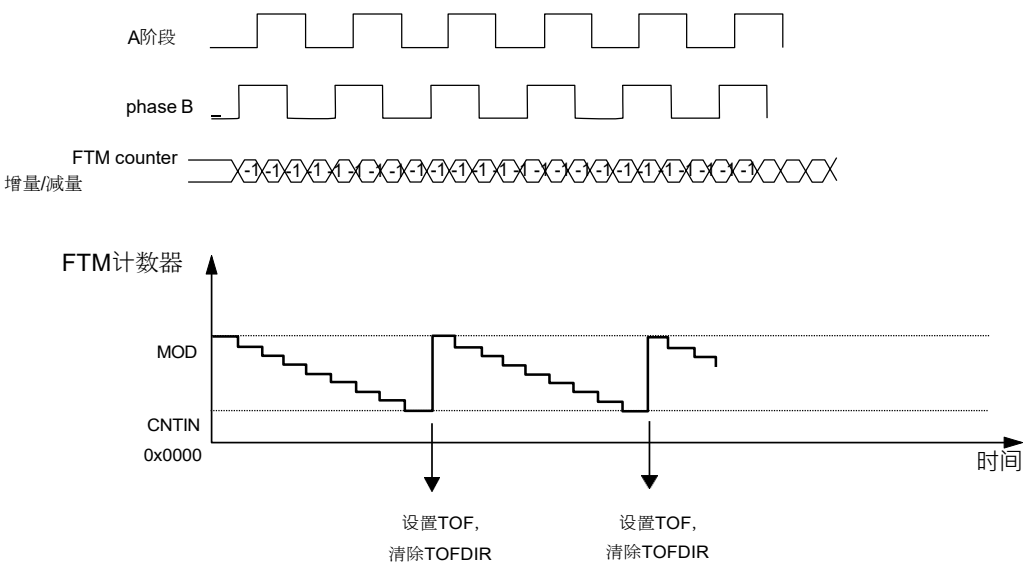



**图42-106.正交解码器模式的下行计数中FTM计数器溢出**

## 42.5.27.1 正交解码器边界条件

下图显示了FTM计数器对电机位置控制应用中典型的电机抖动的反应。

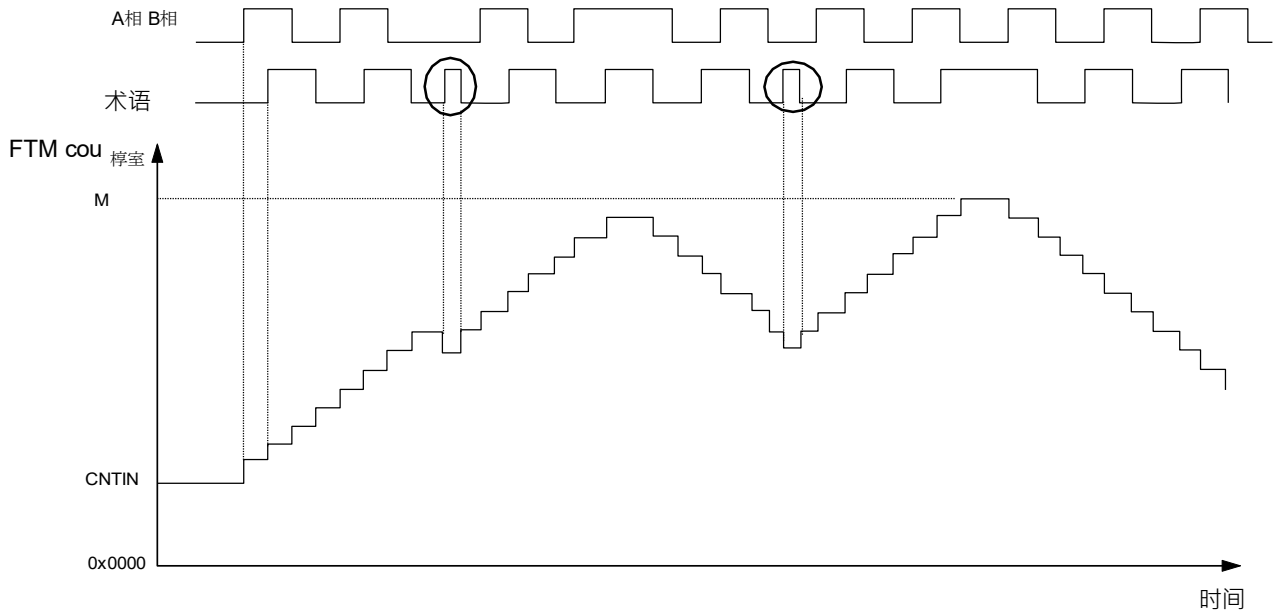



**图42-107.中间计数值的电机位置抖动**

下图显示了由B相和A相脉冲分别产生的电机抖动情况。

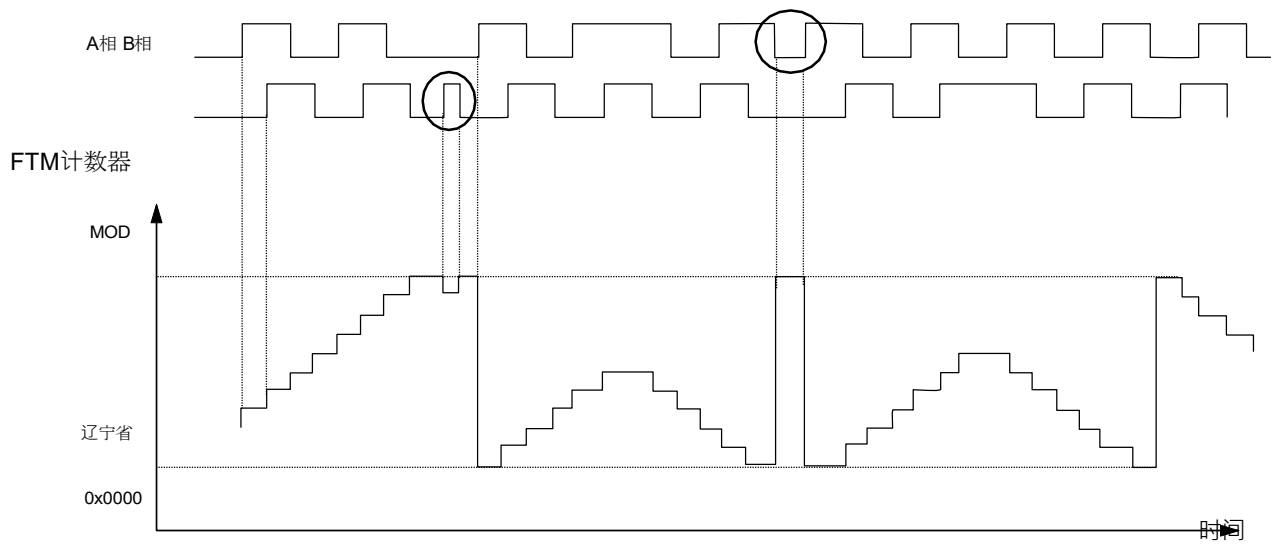



**图42-108.最大和最小计数值附近的电机位置抖动**

第一个突出显示的过渡导致FTM计数器的值在最大计数值（MOD）附近出现抖动。第二个标明的过渡发生在A阶段，导致FTM计数器在由MOD和CNTIN寄存器定义的最大和最小计数值之间过渡。

A相和B相输入滤波器的适当设置对于避免可能导致FTM计数值振荡的毛刺很重要。前面的数字显示了由于输入滤波器设置不当而引起的振荡的例子。因此，保证最小脉冲宽度以避免这些振荡是很重要的。

## 42.5.28 调试模式

当芯片处于调试模式时，BDMMODE[1:0]位根据下表选择FTM计数器、通道（n）CHF位、通道输出的行为，以及对MOD、CNTIN和C(n)V寄存器的写入。

**表42-20.芯片处于调试模式时的FTM行为**

| ÅÂÂ | 基金会计数器 | 通道（n）CHF位 | FTM通道输出 | 对MOD、CNTIN和C(n)V寄存器进行写操作 |
|---|---|---|---|---|
| 00 | 已停止 | 可以设置 | 功能模式 | 对这些寄存器的写入会绕过寄存器缓冲区。 |
| 01 | 已停止 | 没有设置 | 根据POLn位，通道的输出被强制到其安全值 | 对这些寄存器的写入会绕过寄存器缓冲区 |
| 10 | 已停止 | 没有设置 | 当芯片进入调试模式时，各通道的输出被冻结。 | 对这些寄存器的写入会绕过寄存器缓冲区 |
| 11 | 功能模式 | 可以设置 | 功能模式 | 功能模式 |

请注意，如果 BDMMODE[1:0] = 2'b00，那么通道的输出就保持在芯片进入调试模式时的数值，因为 FTM 计数器被停止。然而，以下情况会修改该调试模式下的通道输出。

- 向CNT寄存器写入任何数值；参见计数器复位。在这种情况下，FTM计数器被更新为CNTIN寄存器的值，通道的输出被更新为初始值--除了那些设置为输出比较模式的通道。
- FTM计数器被PWM同步模式复位；见FTM计数器同步。在这种情况下，FTM计数器用CNTIN寄存器的值更新，通道的输出被更新到初始值--除了输出比较模式下的通道。
- 在通道输出初始化中，当数值1被写入INIT位时，通道(n)输出被强制为CH(n)OI位值。见初始化。

**注意事项**

BDMMODE[1:0] = 2'b00
不能与故障控制一起使用。即使故障控制被启用并且存在故障条件，通道的输出也会如上所述被更新。

**注意事项**

如果在调试中CLKS[1:0]=2'b00，在调试中CLKS被写入一个非零值，并且在禁用调试时CnV=CNTIN，那么在禁用调试时CHF位被设置（因为如果通道是一个0%的EPWM信号）。

## 42.5.29 重新加载点数

这个功能允许在选定的重载点用其写缓冲器的值更新寄存器CNTIN、HCR、MOD和C(n)V。

**注意事项**

- 这个功能与PWM同步无关。
- 在这些重载点上，通道输出和FTM计数器都不会改变。软件必须在安全时间点上选择这些重载点。

### 42.5.29.1 重新加载的机会

重载的机会是。

1. 在半周期内

   如果(HCSEL = 1)，这个重载机会被启用，而且它发生在半周期(FTM计数器=HCR寄存器)。软件应该根据FTM计数器的配置计算出半周期的值，然后将这个值写入寄存器HCR中。

2. 在通道（n）匹配

   如果(CH(n)SEL = 1)，该重载机会被启用，并且它发生在通道的(n）匹配（FTM计数器=C（n）V）。

3. 当FTM计数器是一个上升计数器时

   这个重载机会是当FTM计数器从（MOD）变为（CNTIN - 1）时，它总是被启用。

下图显示了在半周期、通道匹配时以及FTM计数器为上升计数器时的重新加载机会的例子。

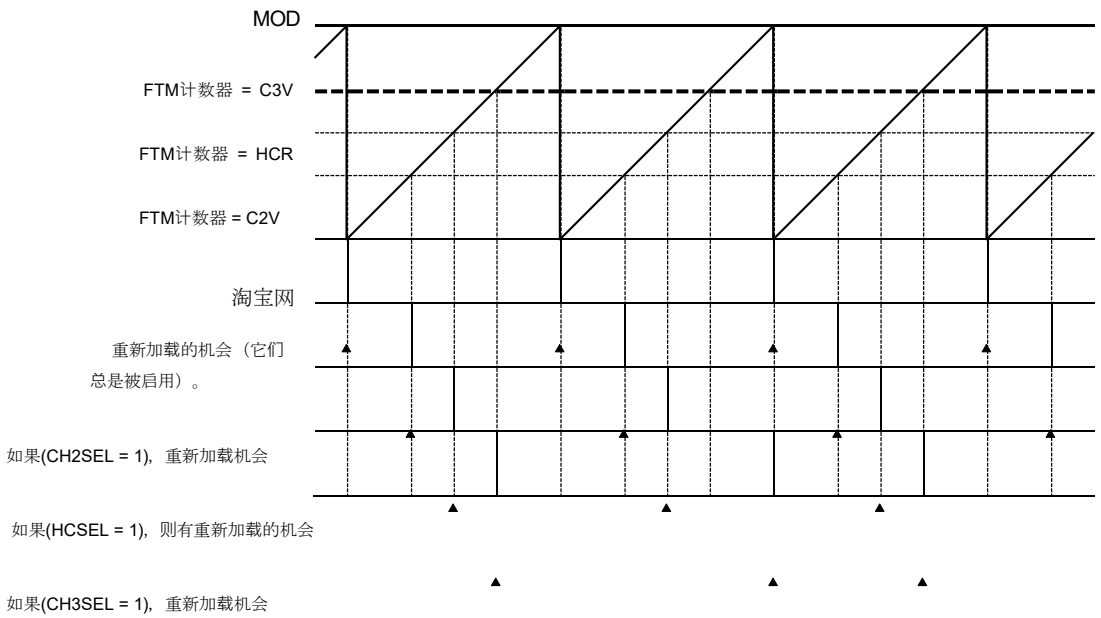



**图42-109.当FTM计数器是一个上升计数器时的重新加载机会**

4. 当FTM计数器是一个向上向下的计数器时

在这种情况下，根据表42-21，通过位CNTMAX和CNTMIN启用重载机会。

**表42-21.当FTM计数器为上-下计数器时，由位CNTMAX和CNTMIN启用的重新加载机会**

| 淘宝网 | ÅÅÅ | 重新加载的机会 |
|---|---|---|
| 0 | 0 | 当FTM计数器从（MOD）变为（MOD-1）时 |
| 0 | 1 | 当FTM计数器从（CNTMIN）变为（CNTMIN+1）时 |
| 1 | 0 | 当FTM计数器从（MOD）变为（MOD-1）时 |
| 1 | 1 | • 当FTM计数器从（MOD）变为（MOD-1）时，以及<br>• 当FTM计数器从（CNTMIN）变为（CNTMIN+1）时 |

下图显示了在半周期、通道匹配时以及FTM计数器为上-下计数器时的重载机会的例子。

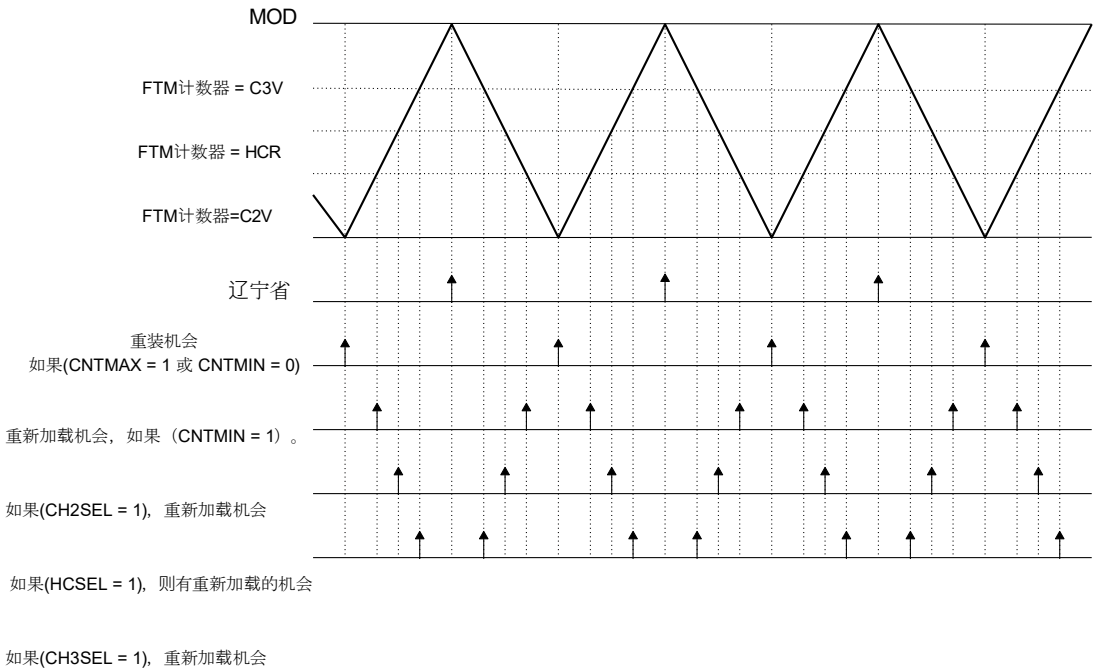



**图42-110.当FTM计数器是一个上-下计数器时的重新加载机会**

## 42.5.29.2 重新加载机会的频率

LDFQ[4:0]位定义了在一个启用的重载机会成为重载点之前应该发生的启用重载机会的数量。下图是LDFQ[4:0]=4时的一个例子。

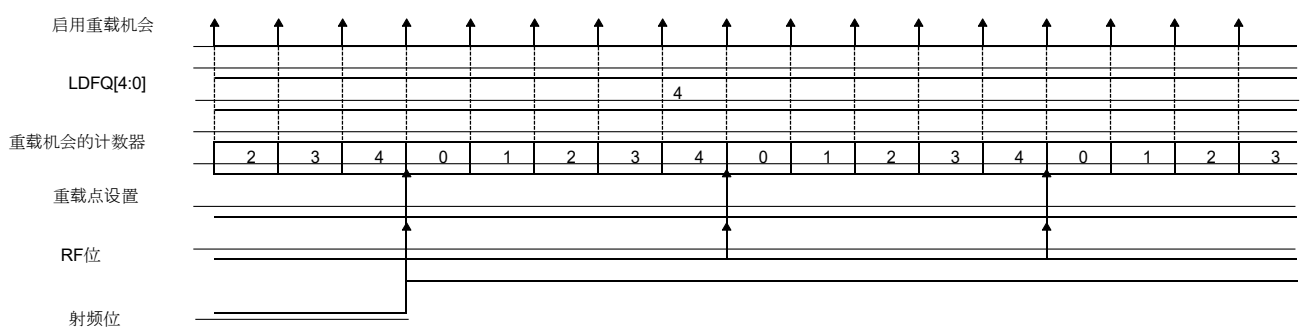



**图42-111.LDFQ[4:0]=4时重装机会的频率**

如果 LDFQ[4:0] = 0，那么所有的重载机会都是重载点。

当有写到寄存器CNT时，重载机会的计数器被重置。

RF位在每个重载点被设置（见上图），与LDOK位的值无关。当(RF=1)和(RIE=1)时，重载点中断被产生。

### 42.5.29.3 登记册的更新

在向带有写缓冲器的寄存器写入新值后，选择其中的哪些寄存器将被更新（根据表42-22），选择重载机会，选择重载机会的频率，因此应设置LDOK位以使这些寄存器在下一个重载点得到更新。

**表42-22.更新寄存器的附加条件**

| 注册 | 附加条件 |
|---|---|
| 辽宁省 | CNTINC = 1 |
| HCR | - |
| MOD | - |
| C(n)V和C(n+1)V | SYNCENm = 1，其中m是一对通道（n）和（n+1） |

## 42.5.30 全球负载

如果在PWMLOAD寄存器中的全局加载OK（GLDOK）位进行了写一操作，全局加载机制允许几个模块在同步事件后同步重新加载其双缓冲寄存器。全局加载可以通过配置PWMLOAD寄存器中的全局加载使能（GLEN）位来启用或禁用。在GLEN启用的情况下，在GLDOK位上写一，与在LDOK位上写一的效果相同。请参考SoC关于全局负载连接的具体信息。

全局加载机制允许 MOD、HCR、CNTIN 和 C(n)V
寄存器在可配置的重新加载点用寄存器缓冲器的内容进行更新。下图显示了FTM全局加载输入和输出之间的连接实例，考虑到GLDOK位是在FTM模块之外实现的。

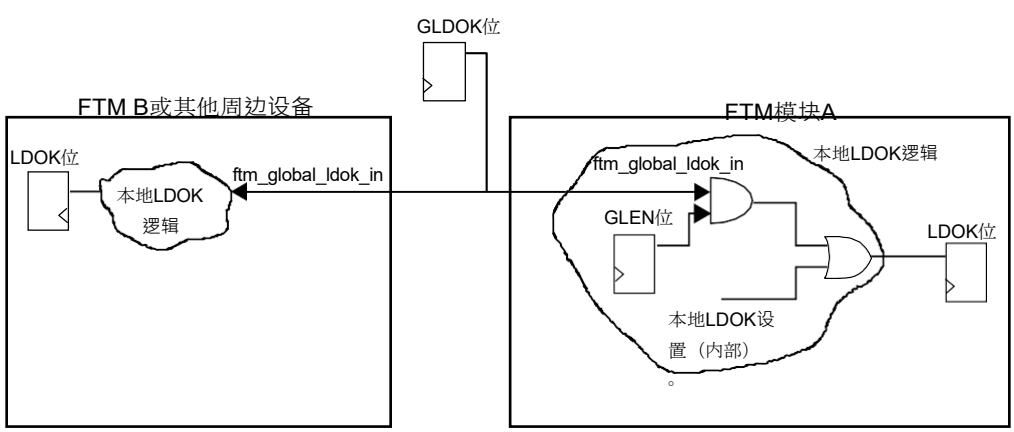


**图42-112.全局负载逻辑**

## 42.5.31 全球时基（GTB）

全局时基（GTB）是一个FTM功能，它允许一个芯片上的多个FTM模块同步。下图显示了一个用于同步两个FTM模块的GTB功能的例子。在这种情况下，FTM A和B通道可以像只使用一个FTM模块一样，即全局时基的行为。

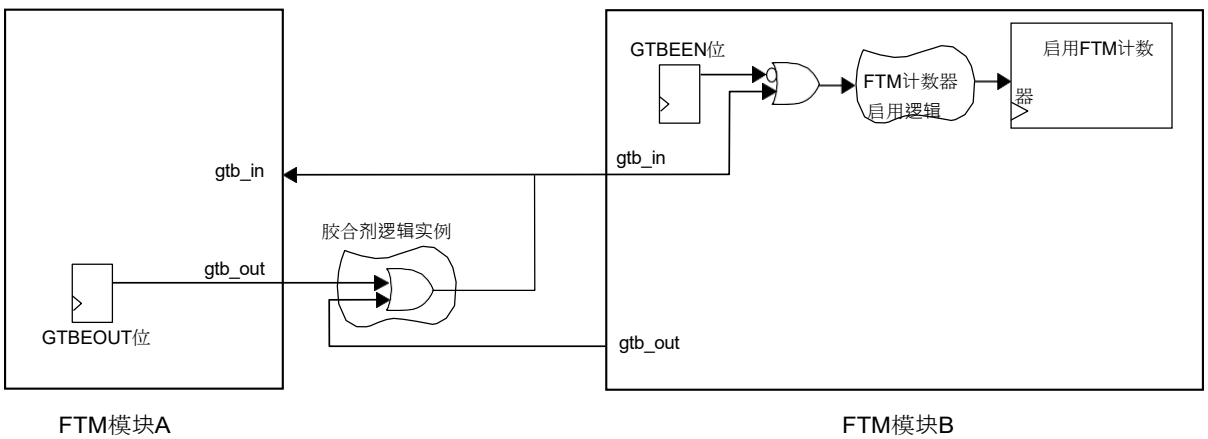



**图42-113.全局时基（GTB）框图**

GTB功能由CONF寄存器中的GTBEEN和GTBEOUT位、输入信号*gtb_in*和输出信号*gtb_out*实现。GTBEEN位使gtb_in控制FTM计数器使能信号。

- 如果GTBEEN=0，每个FTM模块根据其配置的模式独立工作。
- 如果GTBEEN = 1，只有当gtb_in为1时，FTM计数器的更新才被启用。

在上图描述的配置中，如果其中一个FTM模块的gtb_out信号至少有一个为1，则FTM模块A和B的FTM计数器被启用。请注意，这些配置与芯片有关，在FTM模块之外实现。
关于芯片的具体实现，请参见芯片特定的FTM信息。

**注意事项**

- 为了使用GTB信号来同步不同FTM模块的FTM计数器，每个FTM模块的配置应保证其FTM计数器在gtb_in信号为1时就开始计数。
- GTB功能不提供FTM计数器的连续同步，这意味着FTM计数器在FTM操作期间可能失去同步。

GTB功能只允许FTM计数器*启动*
它们的操作是同步的。

### 42.5.31.1 启用全局时基（GTB）。

要启用GTB功能，请对每个参与的FTM模块遵循这些步骤。

1. 停止FTM计数器。写00b到SC[CLKS]。
2. 将FTM编程为预定的配置。FTM计数器的模式需要在所有参与的模块中保持一致。
3. 向CONF[GTBEEN]写1，同时向CONF[GTBEOUT]写0。
4. 在SC[CLKS]中选择预定的FTM计数器时钟源。该时钟源需要在所有参与模块中保持一致。
5. 重置FTM计数器。向CNT寄存器写入任何数值。

为了在上图描述的配置中启动GTB功能，在用作时基的FTM模块中向CONF[GTBEOUT]写1。

## 42.5.32 通道触发输出

通道触发输出提供一个触发信号，该信号在通道（n）输出中具有一个FTM输入时钟周期宽度。

如果 CnSC 寄存器的 TRIGMODE 位被设置（TRIGMODE = 1），当发生匹配时，在通道（n）的输出中会产生一个 FTM 输入时钟宽度的触发脉冲。只有当通道（n）处于 EPWM 或 CPWM 模式时，才允许使用触发模式。

下面的数字显示了通道(n)输出中产生触发的一些情况。

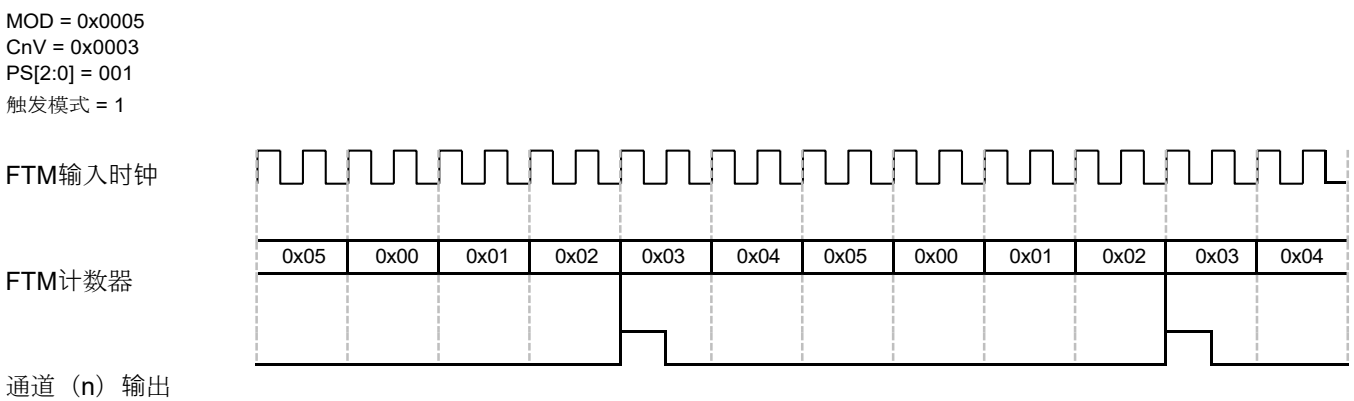


图42-114.在EPWM模式下，通道(n)输出端触发的例子

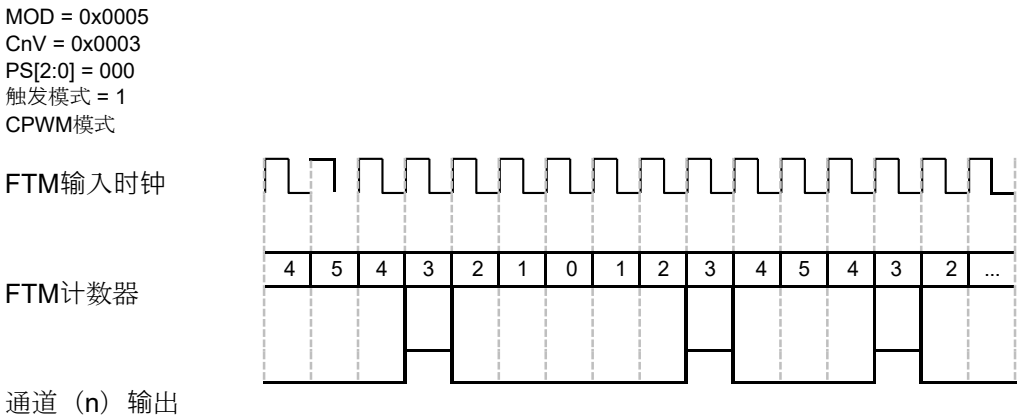



**图42-115.在CPWM模式下，通道(n)输出端触发的例子**

## 42.5.33 通道输出的外部控制

通道(n)的PWMEN位可用于FTM外部逻辑，控制通道(n)输出的最终值。当FSTATE = 1
和通道（n）输出被故障控制禁用时，这个逻辑也可以控制通道（n）输出。下图是这个外部逻辑的一个例子。

术语 "通道（n）输出
"是指极性控制之后的通道（n）输出值。更多细节请参见功能优先和极性控制。

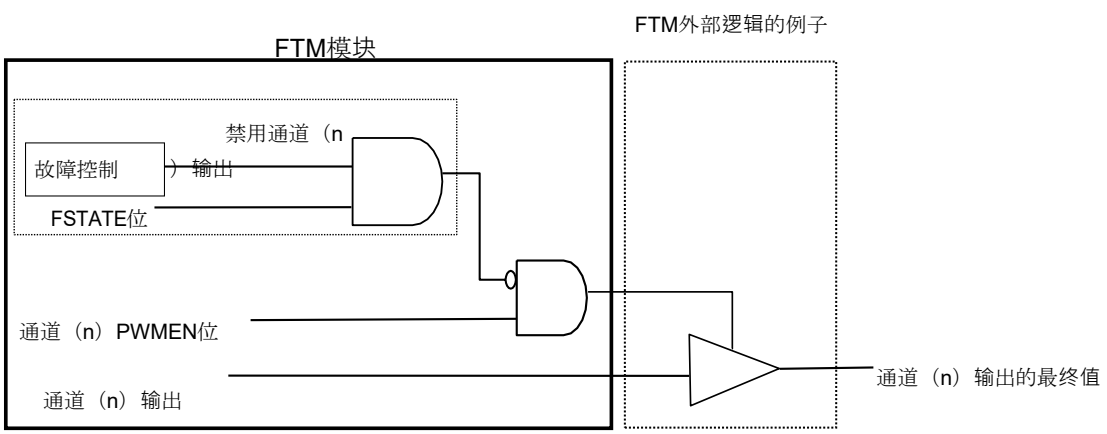



**图42-116.通道（n）输出的外部控制实例**

## 42.5.34 抖动

FTM实现了一个分数延迟，以利用抖动实现对生成的PWM信号的精细分辨率。抖动可以被那些需要比FTM计数器的一个单元更多分辨率的应用所使用。

有两种抖动可供选择。PWM周期抖动和边缘抖动。

### 42.5.34.1 PWM周期抖动

当FRACMOD被写入一个非零值时，PWM周期抖动被启用。在PWM周期抖动中使用的内部累加器在以下情况下被重置。

- 寄存器MOD_MIRROR的字段MOD被更新为其写入的值。
  缓冲。
- FRACMOD被更新为其写缓冲区的值，或
- FTM计数器被停止。

**注意事项**

对于PWM周期抖动，应该使用寄存器MOD_MIRROR来代替寄存器MOD。

为了避免不一致，当寄存器MOD的字段MOD用其写缓冲区的值更新时，字段FRACMOD被清零。

PWM周期抖动是不可用的。

- 当FTM计数器是一个自由运行的计数器时
- 当FTM处于正交解码器模式时

#### 42.5.34.1.1 上数

当FTM计数器是一个向上的计数器，并且启用了PWM周期抖动时，在每个PWM周期结束时，FRACMOD值被添加到一个内部的5位累加器。当这个累加器溢出时（即加法的结果大于或等于0x20），那么FTM计数器的一个单位被加到当前PWM周期的末尾，而累加器仍然是减法的其余部分：（这个加法的结果-0x20）。

由于一个单位的FTM计数器可以添加到PWM周期中，对于无符号计数的PWM周期抖动，MOD的最大有效值是0xFFFE，而对于有符号计数的PWM周期抖动，则是0x7FFE。

下面的数字显示了当FTM计数器是一个上升计数器时，PWM周期抖动的一些例子。

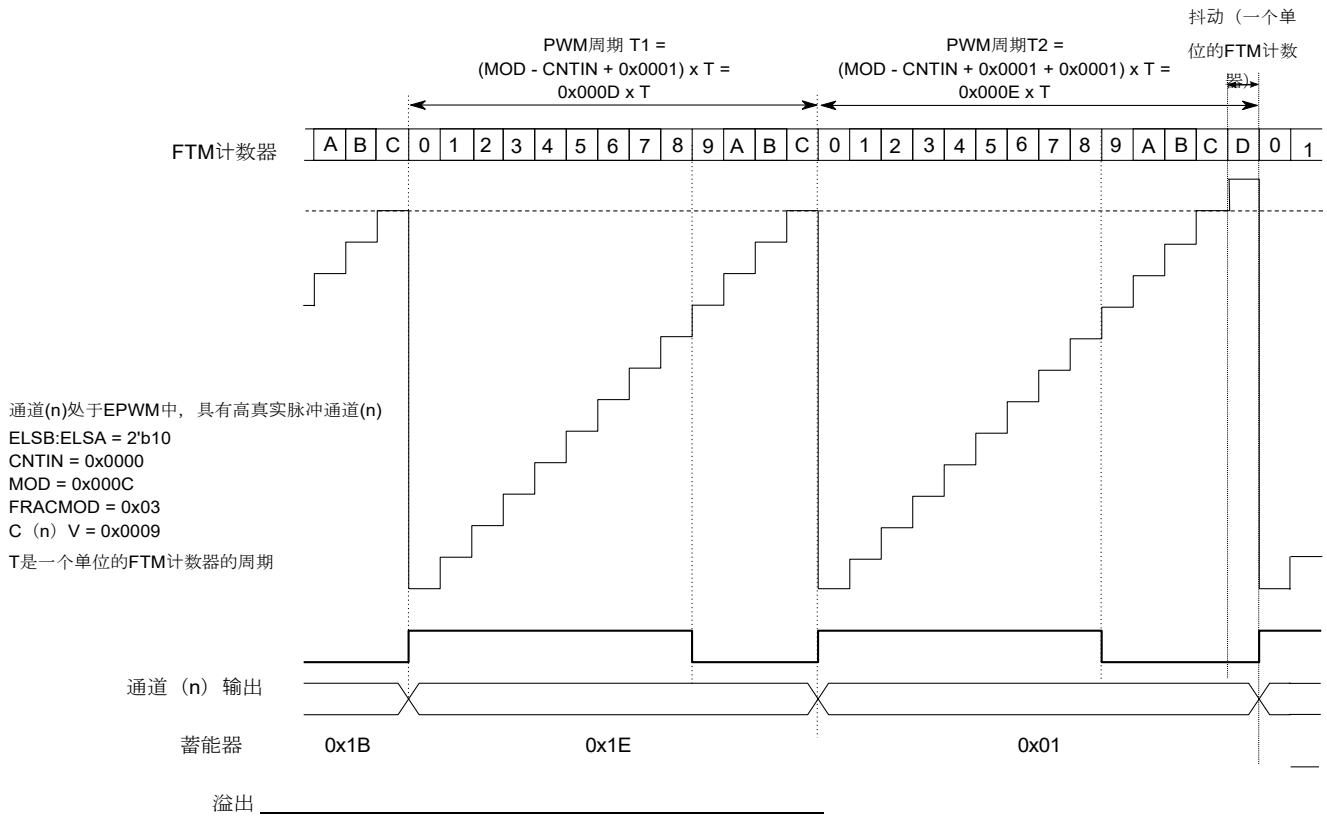



**图42-117.带上行计数的PWM周期抖动**

假设。

- FTM计数器是一个向上的计数器。
- T是一个单位的FTM计数器。
- 没有周期抖动的PWM周期是[(MOD - CNTIN + 1) x T]。
- 抖动周期的PWM周期数为FRACMOD。
- 带周期抖动的PWM周期为[(MOD - CNTIN + 1 + 1) x T]。

因此，平均周期（十进制）是[（MOD-CNTIN+1）+（FRACMOD/32）]x T，其中整数值是（MOD-CNTIN+1），小数值是（FRACMOD/32）。见下面的例子。

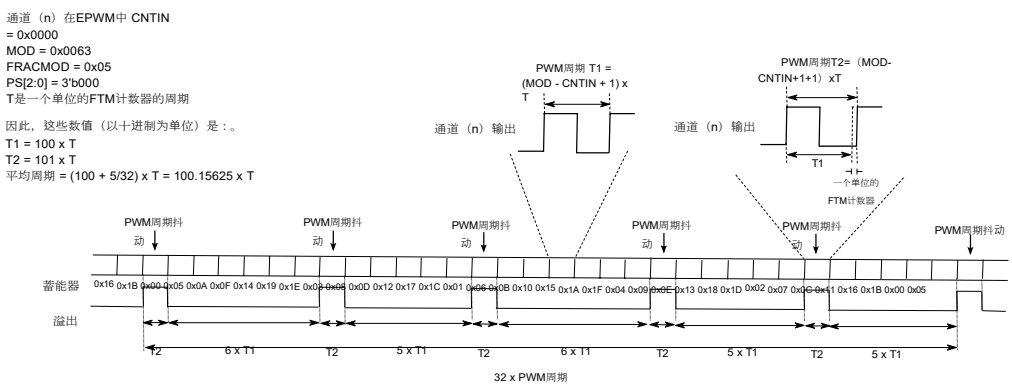


图42-118.当PWM周期抖动与上升计数一起使用时的平均周期示例

注意事项

对于使用EPWM模式和PWM周期抖动在通道（n）中生成100%的PWM信号（通道（n）ELSB:ELSA = 2'b10），建议使用（C（n）>MOD + 1）。

对于使用组合模式和PWM周期抖动在通道（n）中生成PWM信号（通道（n）ELSB:ELSA=2'b10），建议使用

。

- 对于0%的PWM信号：（C(n)V > MOD + 1）和（C(n+1)V > MOD + 1）。
- 对于100%的PWM信号：（C(n)V = CNTIN）和（C(n+1)V > MOD + 1）。

对于在通道(n)中生成PWM信号(与通道(n)ELSB:ELSA=2'b10)，使用修改的组合PWM模式和PWM周期抖动，建议使用。

- 对于0%的PWM信号：（C(n)V > MOD + 1）和（CNTIN ≤ C(n+1)V ≤ MOD）。
- 对于100%的PWM信号：（CNTIN≤C（n）V≤MOD）和（C（n+1）V>MOD+1）。

#### 42.5.34.1.2 向上-向下计数

当FTM计数器是一个上-
下计数器并且启用了PWM周期抖动时，在每个PWM周期结束时，FRACMOD值被加到一个内部的5位累加器中。当这个累加器溢出时(即累加的结果大于或等于0x20)，那么FTM计数器的一个单元被加到当前PWM周期的末端，另一个单元被加到下一个PWM周期的开始(见下图)。在累加器溢出后，累加器仍有剩余的减法：（这个加法的结果-0x20）。

由于有一个单位的FTM计数器可以加到PWM周期中，对于上-
下计数的PWM周期抖动，MOD的最大有效值是0x7FFE（CPWM模式）。

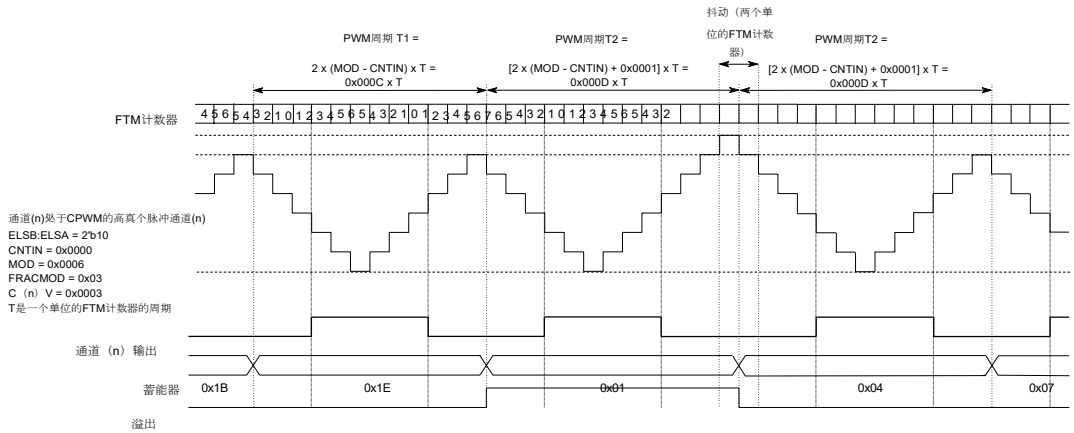



**图42-119.带有上下计数的PWM周期抖动**

### 注意事项

对于使用CPWM模式和PWM周期抖动在通道(n)中生成100%的PWM信号（通道(n)ELSB:ELSA=2'b10），建议使用（C(n)V[15]=0）和（C(n)V>MOD+1）以及（MOD ≠0x0000）。

## 42.5.34.2 PWM边缘抖动

在PWM边缘抖动中使用的通道（n）内部累加器在以下情况下被重置。

- 寄存器C(n)V_MIRROR的字段VAL被更新为其写缓冲器的值。
- FRACVAL被更新为其写缓冲区的值，或
- FTM计数器被停止。

### 注意事项

对于PWM边缘抖动，应该使用寄存器C(n)V_MIRROR来代替寄存器C(n)V。

为了避免不一致，当寄存器C(n)V的字段VAL被更新为其写缓冲区的值时，字段FRACVAL被清零。

PWM边缘抖动是不可用的。

- 到输入模式下的通道，以及
- 到输出比较模式下的通道。

### 42.5.34.2.1 EPWM模式

当非零值被写入通道（n）FRACVAL时，EPWM模式下通道（n）的PWM边缘抖动被启用。

如果通道(n)处于EPWM模式，并且启用了PWM边缘抖动，在每个EPWM周期结束时，通道(n)的FRACVAL值被添加到通道(n)内部的5位累加器。当这个累加器溢出时（即加法的结果大于或等于0x20），累加器仍然是减法的剩余部分：（这个加法的结果-0x20）。

在这种配置下，EPWM占空比的初始边沿发生在(FTM计数器=CNTIN)时，其位置不被PWM边沿抖动所改变。如果在当前的EPWM周期中，通道(n)的累加器没有溢出，那么EPWM占空比的最终边缘发生在通道(n)的匹配上(FTM计数器=C(n)V)，也就是说，它的位置没有被边缘抖动修改。然而，如果在当前 EPWM 期间，通道(n)的累加器有溢出，那么 EPWM 占空比的最后边缘就发生在(FTM counter = C(n)V + 0x0001)。

下面的数字显示了当通道（n）处于EPWM模式时，PWM边缘抖动的一些例子。

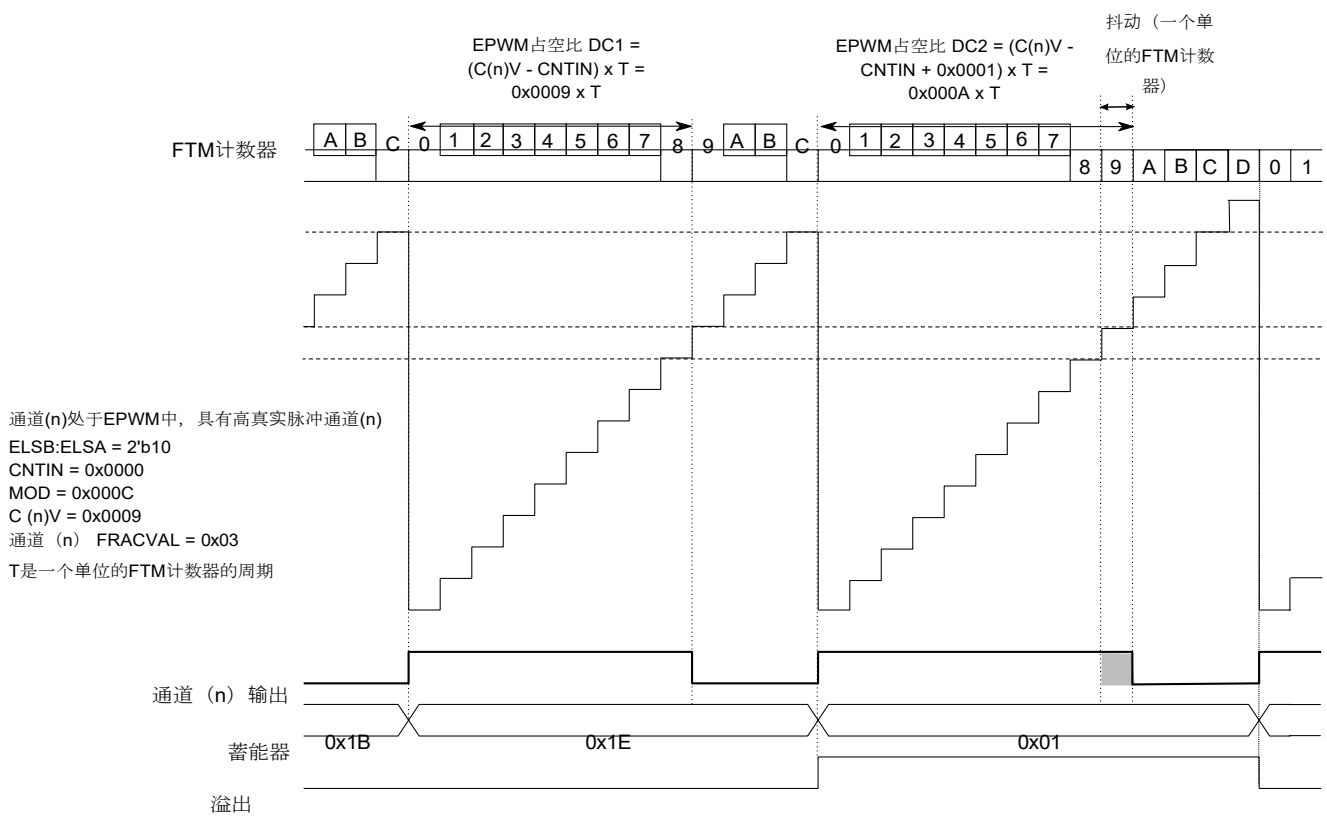



**图42-120.通道(n)处于EPWM模式，具有PWM边缘抖动功能**

假设。

- 该通道（n）处于EPWM模式。
- T是一个单位的FTM计数器。
- 没有边缘抖动的EPWM占空比为[(C(n)V - CNTIN) x T]。
- 具有边沿抖动的PWM周期的数量，即FRACVAL。
- 边缘抖动的EWM占空比为[(C(n)V - CNTIN + 1) x T]。

因此，平均占空比（十进制）是[(C(n)V - CNTIN) + (FRACVAL/32)] x T，其中整数值是(C(n)V - CNTIN)，小数值是(FRACVAL/32)。见下面的例子。

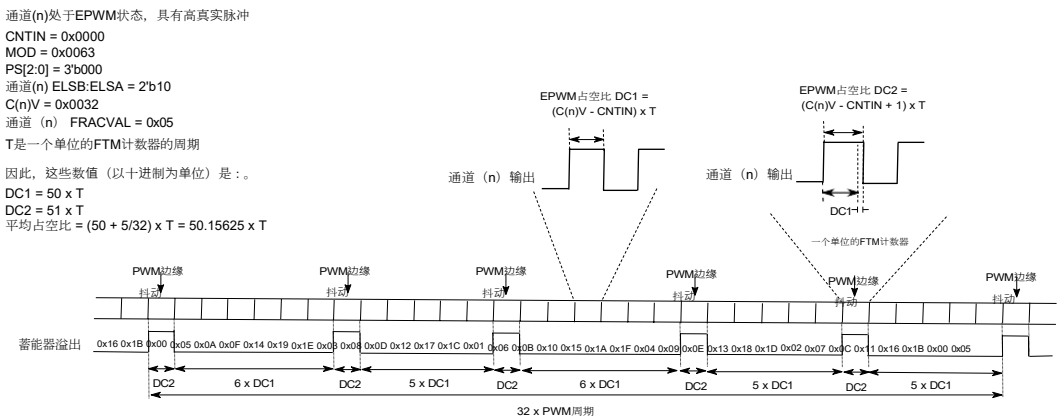



**图42-121.当通道(n)处于EPWM模式下的PWM边缘抖动时的平均占空比示例**

### 42.5.34.2.2 CPWM模式

当非零值被写入通道（n）FRACVAL时，CPWM模式下通道（n）的PWM边缘抖动被启用。

如果通道（n）处于CPWM模式，并且启用了PWM边缘抖动，在每个CPWM周期结束时，通道（n）的FRACVAL值被添加到通道（n）内部的5位累加器。当这个累加器溢出时（即加法的结果大于或等于0x20），累加器仍然是减法的剩余部分：（这个加法的结果-0x20）。

在这种配置下，如果在当前的CPWM周期内没有通道（n）累加器的溢出，那么占空比就不会被PWM边缘抖动修改，也就是说，CPWM占空比的初始边缘发生在通道（n）的匹配上（FTM计数器）。
=
C(n)V），当FTM计数器递减时，通道（n）上CPWM占空比的最后边缘与FTM计数器递增时相匹配。

然而，如果在当前的CPWM周期中，通道(n)累加器出现溢出，那么CPWM占空比的初始边缘发生在(FTM计数器=C(n)V)
+ 0x0001），FTM 计数器在递减，CPWM 占空比的最后边缘时（FTM 计数器 = C(n)V + 0x0001），FTM 计数器在递增。

下图显示了当通道（n）处于CPWM模式时，PWM边缘抖动的一个例子。

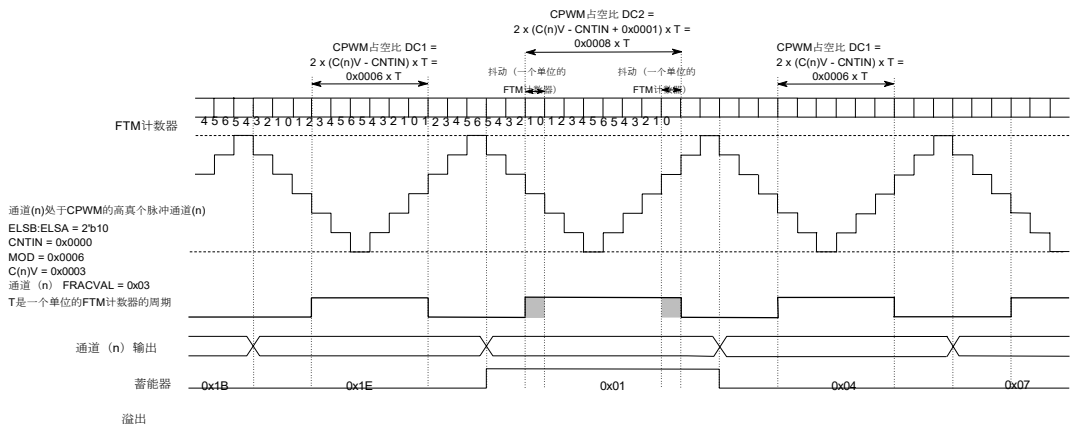



**图42-122.通道（n）处于CPWM模式，带有PWM边缘抖动**

### 42.5.34.2.3 合并模式

在组合模式下，可以进行PWM边缘抖动。

- 在通道（n）匹配（FTM计数器=C（n）V）边缘或
- 在通道（n+1）匹配（FTM计数器=C（n+1）V）边缘。

当一个非零值被写入通道（n）FRACVAL时，通道（n）匹配边缘抖动被启用。

对于通道（n）的匹配边缘抖动，通道（n）有一个内部的5位累积器。在每个PWM周期结束时，通道（n）的FRACVAL值被添加到通道（n）的累加器中。当这个累加器溢出时（即加法的结果大于或等于0x20），累加器仍然是减法的剩余部分：（这个加法的结果-0x20）。

如果在当前的PWM周期内没有发生通道(n)累加器的溢出，则通道(n)的匹配边缘不被修改，也就是说，它发生在通道(n)的匹配。
然而，如果通道(n)的累加器有溢出，通道(n)的匹配边缘发生在(FTM counter = C(n)V + 0x0001)。

下图显示了当通道（n）和（n+1）处于组合模式时，通道（n）匹配边缘抖动的例子。

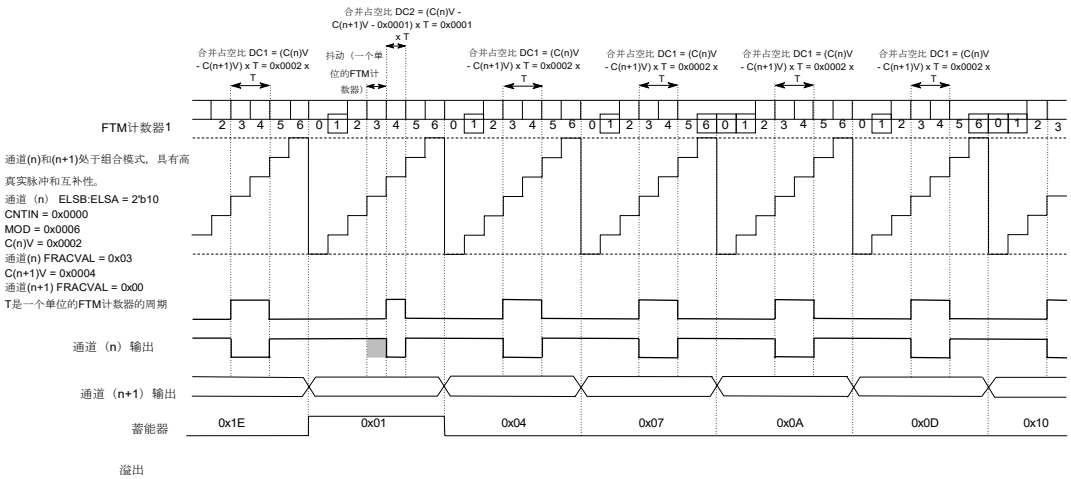


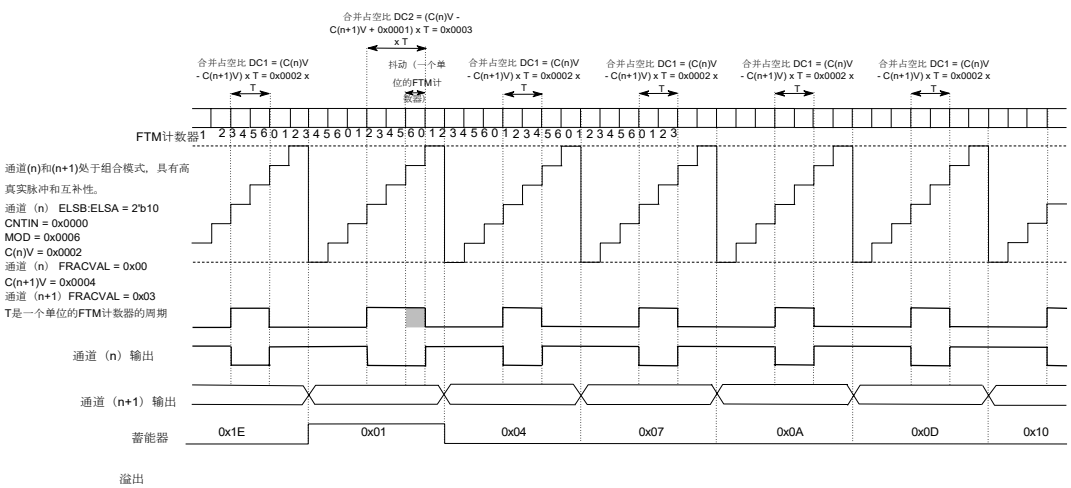

**图42-123.合并模式下的通道（n）匹配边缘抖动**

当一个非零值被写入通道（n+1）FRACVAL时，通道（n+1）匹配边缘抖动被启用。

对于通道（n+1）的匹配边缘抖动，通道（n+1）有一个内部的5位累加器。在每个PWM周期结束时，通道（n+1）的FRACVAL值被添加到通道（n+1）的累加器中。当这个累加器溢出时（即加法的结果大于或等于0x20），累加器仍然是减法的剩余部分：（这个加法的结果-0x20）。

如果在当前PWM周期内没有通道(n+1)累加器的溢出，则通道(n+1)的匹配边缘不被修改，也就是说，它发生在通道(n+1)的匹配。但是，如果通道(n+1)累加器有溢出，则通道(n+1)匹配边沿发生在(FTM计数器=C(n+1)V+0x0001)。

下图显示了当通道（n）和（n+1）处于组合模式时，通道（n+1）匹配边缘抖动的例子。



注意事项

建议每次只使用一个PWM边缘抖动（通道（n）PWM边缘抖动或通道（n+1）PWM边缘抖动）。

对于在通道（n）中生成0%的PWM（与通道(n) ELSB:ELSA = 2'b10)使用组合模式和PWM边缘抖动，建议使用。

- (C(n)V<CNTIN或C(n)V>MOD)和(通道(n)FRACVAL为零)和
- (通道(n+1)FRACVAL为零)。

对于使用组合模式和PWM边缘抖动在通道（n）中生成100%的PWM（通道（n）ELSB:ELSA=2'b10），建议使用。

- (C(n)V=CNTIN)和(通道(n)FRACVAL为零)和
- (C(n+1)V < CNTIN 或 C(n+1)V > MOD) 和 (通道 (n+1）FRACVAL为零）。

#### 42.5.34.2.4 修改后的组合式PWM模式

在修改后的组合式PWM模式下，可以进行PWM边缘抖动。

- 在通道（n）匹配（FTM计数器=C（n）V）边缘或
- 在通道（n+1）匹配（FTM计数器=C（n+1）V）边缘。

当一个非零值被写入通道（n）FRACVAL时，通道（n）匹配边缘抖动被启用。

对于通道（n）的匹配边缘抖动，通道（n）有一个内部的5位累积器。在每个PWM周期结束时，通道（n）的FRACVAL值被添加到通道（n）的累加器中。当这个累加器溢出时（即加法的结果大于或等于0x20），累加器仍然是减法的剩余部分：（这个加法的结果-0x20）。

如果在当前的PWM周期内没有发生通道(n)累加器的溢出，则通道(n)的匹配边缘不被修改，也就是说，它发生在通道(n)的匹配。
然而，如果通道(n)的累加器有溢出，通道(n)的匹配边缘发生在(FTM counter = C(n)V + 0x0001)。

下图显示了当通道（n）和（n+1）处于修改组合PWM模式时，通道（n）匹配边缘抖动的例子。

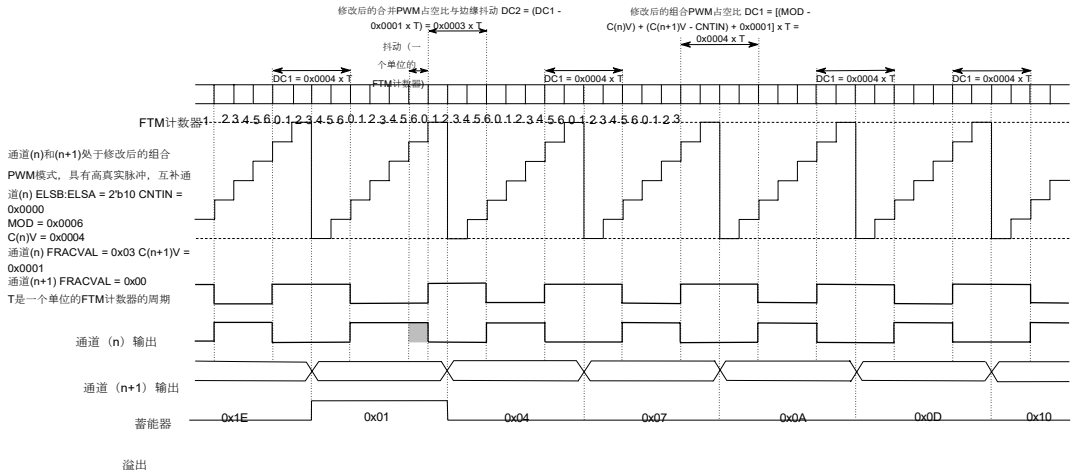



**图42-125.修改后的组合式PWM模式下的通道（n）匹配边缘抖动**

当一个非零值被写入通道（n+1）FRACVAL时，通道（n+1）匹配边缘抖动被启用。

对于通道（n+1）的匹配边缘抖动，通道（n+1）有一个内部的5位累加器。在每个PWM周期结束时，通道（n+1）的FRACVAL值被添加到通道（n+1）的累加器中。当这个累加器溢出时（即加法的结果大于或等于0x20），累加器仍然是减法的剩余部分：（这个加法的结果-0x20）。

如果在当前PWM周期内没有通道(n+1)累加器的溢出，则通道(n+1)的匹配边缘不被修改，也就是说，它发生在通道(n+1)的匹配。但是，如果通道(n+1)累加器有溢出，则通道(n+1)匹配边沿发生在(FTM计数器=C(n+1)V+0x0001)。

下图显示了当通道（n）和（n+1）处于修改组合PWM模式时，通道（n+1）匹配边缘抖动的例子。

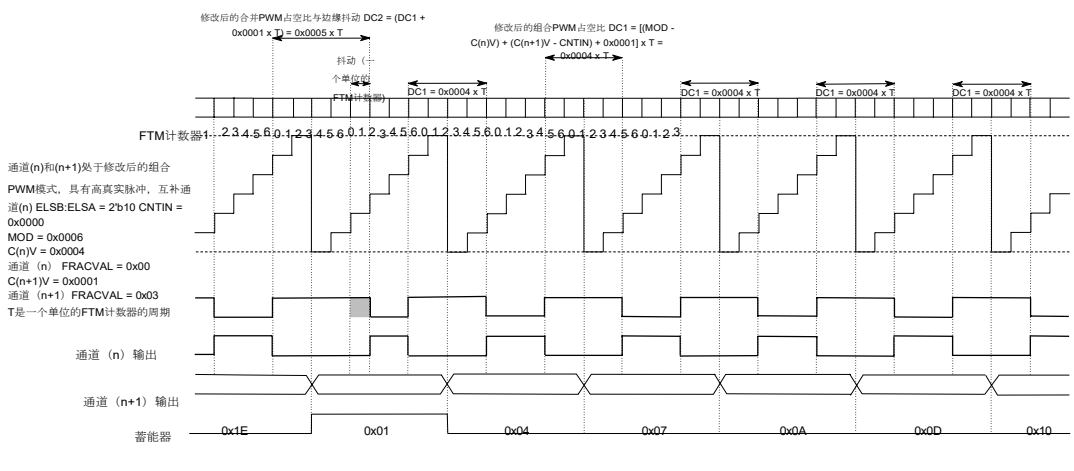


图42-126.通道(n+1)匹配边缘抖动在修改后的组合PWM模式中的应用

注意事项

建议每次只使用一个PWM边缘抖动（通道（n）PWM边缘抖动或通道（n+1）PWM边缘抖动）。

对于在通道（n）中生成0%的PWM（与通道(n) ELSB:ELSA = 2'b10)使用修改的组合PWM模式和PWM边缘抖动，建议使用。

- (C(n)V<CNTIN或C(n)V>MOD)和(通道(n)FRACVAL为零)和
- (CNTIN≤C(n+1)V≤MOD)和(通道(n+1)FRACVAL为零)。

对于使用修改的组合PWM模式和PWM边缘抖动，在通道（n）中生成100%的PWM（通道（n）ELSB:ELSA=2'b10），建议使用。

- (CNTIN≤C(n)V≤MOD)和(通道(n)FRACVAL为零)和
- (C(n+1)V < CNTIN 或 C(n+1)V > MOD) 和 (通道 (n+1）FRACVAL为零）。

## 42.6 重置概述

每当发生任何芯片复位时，FTM

都会被复位。当FTM从复位中退出时。

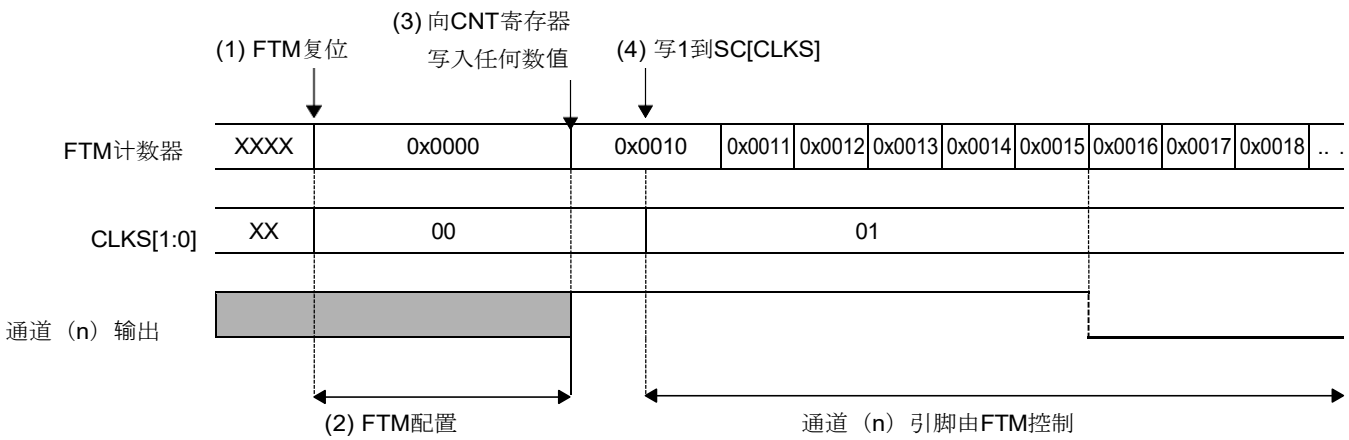
- FTM计数器和预分频器计数器为零并被停止（CLKS[1:0] = 2'b00）。
- 定时器溢出中断为零（定时器溢出中断）。
- 通道中断为零（通道(n)中断）。
- 故障中断为零（Fault Interrupt）。
- 的通道处于输入捕捉模式（输入捕捉模式）。
- 通道的输出为零。
- 通道 ELSB:ELSA=0:0（通道模式），PWMEN=0（通道输出的外部控制）。

下图显示了复位后的FTM行为。复位时（第1项），FTM计数器被禁用（CLKS[1:0] = 2'b00），其值被更新为零，引脚不受FTM控制（通道模式）。

复位后，应配置 FTM（第 2 项）。必须定义 FTM 计数器模式、FTM 计数限制（MOD 和 CNTIN 寄存器值）、通道模式以及根据通道模式确定的 CnV 寄存器值。

因此，建议向CNT寄存器（第3项）写入任何数值。这种写法用CNTIN寄存器的值更新FTM计数器，用它的初始值更新通道输出（除了输出比较模式的通道）（计数器复位）。

下一步是通过CLKS[1:0]位（第4项）来选择FTM计数器的时钟。需要强调的是，只有当CLKS[1:0]位与0不同时，引脚才由FTM控制。

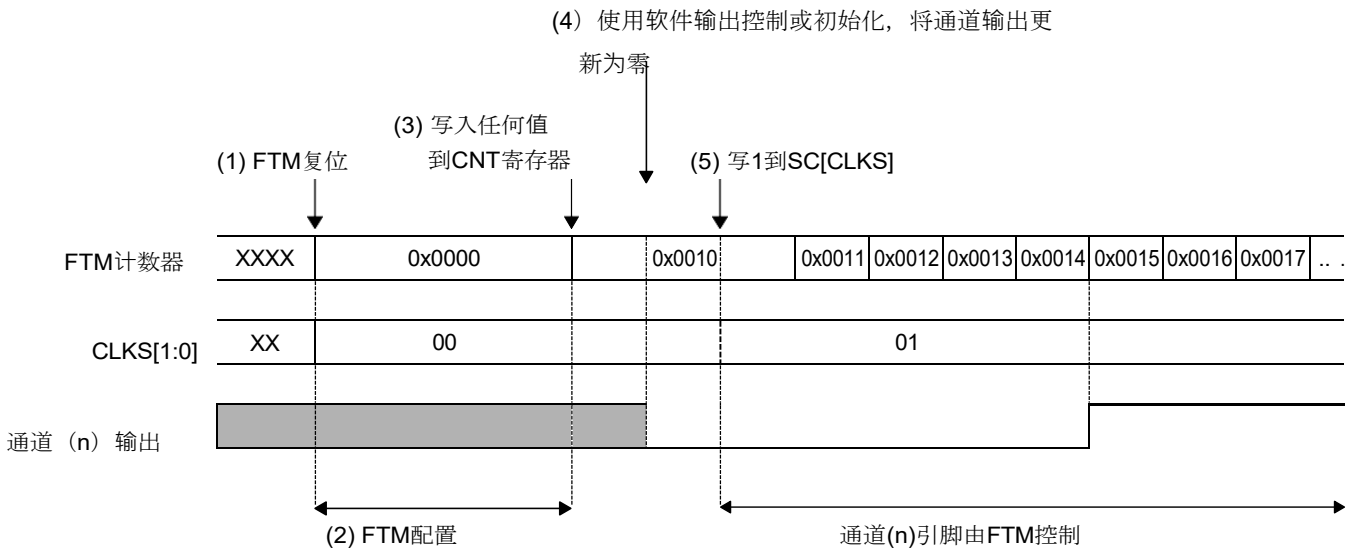



**注意事项。**

– CNTIN = 0x0010
– 通道(n)处于低真个结合模式，CNTIN < C(n)V < C(n+1)V < MOD
– C(n)V = 0x0015

**图 42-127.当通道（n）处于组合模式时，复位后的FTM行为**

下图显示了一个例子，当通道(n)处于输出比较模式时，通道(n)的输出在有匹配时被切换。在输出比较模式下，当有对CNT寄存器（第3项）的写入时，通道的输出不会被更新到初始值。在这种情况下，使用软件输出控制（软件输出控制模式）或初始化（初始化）将通道输出更新到选定的值（第4项）。




**注意事项。**

– CNTIN = 0x0010
– 通道（n）处于输出比较状态，当有匹配时，通道（n）的输出被拨动
– C(n)V = 0x0014

**图42-128.当通道(n)处于输出比较模式时，复位后的FTM行为**

# 42.7 FTM中断

## 42.7.1 定时器溢出中断

当(TOIE=1)和(TOF=1)时，产生定时器溢出中断。

## 42.7.2 重新加载点中断

当(RIE=1)和(RF=1)时，产生重装点中断。

## 42.7.3 通道(n) 中断

当(CHIE=1)和(CHF=1)时，产生通道(n)中断。

## 42.7.4 故障中断

当(FAULTIE = 1)和(FAULTF = 1)时，产生故障中断。

## 42.8 初始化程序

建议采用以下初始化程序来配置FlexTimer。该程序也可用于进行新的配置。

1. 定义POL位。
2. 在写完OUTMASK后的两个时钟，通道的输出处于安全值。
3. (重新)配置FTM计数器和通道以生成周期性信号。
   a. 禁用时钟。禁用正交解码器模式。
   b. (重新)配置的例子。
      - 写给MOD
      - 写给CNTIN
      - 配置将被使用的通道
      - 将输出模式下的通道写入CnV
      - (重新)配置死机时间和故障控制
      - 在没有SW同步的情况下，不要使用SWOC（见第6项）。
      - 在没有SW同步的情况下，不要使用倒置（见第6项）。
      - 请不要使用初始化
      - 请勿改变极性控制
      - 不要配置HW同步。
4. 向CNT写入任何数值。FTM计数器被重置，通道的输出根据新的配置被更新。
5. 启用时钟。向CLKS[1:0]位写入一个不同于零的值。启用正交解码器模式（如果需要的话）。
6. 配置SWOC（如果有必要）、反转（如果有必要）和输出屏蔽（总是）的SW同步。
   a. 选择输出掩码的同步性
      - 写入SYNC（SWSYNC = 0, TRIG2 = 0, TRIG1 = 0, TRIG0 = 0, SYNCHOM = 1, REINIT = 0, CNTMAX = 0, CNTMIN = 0）。
   b. 写给SYNCONF
      - 不能启用HW同步（HWSOC=0，HWINVC=0，HWOM=0，HWWRBUF=0，HWRSTCNT=0，HWTRIGMODE=0）
      - SWOC的SW同步（如果有必要）。SWSOC=[0/1]，SWOC=[0/1]
      - 用于反转的SW同步（如果有必要）。SWINVC=[0/1]和INVC=[0/1]
      - SWOM的同步化（总是）。SWOM = 1

- 不启用写缓冲器的SW同步（因为对有写缓冲器的寄存器的写入是使用CLKS[1:0]=2'b00完成的）。SWWRBUF = 0和CNTINC = 0
- 计数器复位的SW同步（始终）。SWRSTCNT = 1
- 增强同步（总是）。SYNCMODE = 1

c. 如果使用SWOC（SWSOC=1，SWOC=1），则写入SWOCTRL寄存器。

d. 如果使用反转（SWINVC=1，INVC=1），则写入INVCTRL寄存器。

e. 写入OUTMASK以启用被屏蔽的通道。

7. 生成软件触发器
   - 写入SYNC（SWSYNC = 1, TRIG2 = 0, TRIG1 = 0, TRIG0 = 0, SYNCHOM = 1, REINIT = 0, CNTMAX = 0, CNTMIN = 0）。
8. 配置PWMEN位（通道输出的外部控制）。

# 第43章
# 低功耗中断定时器（LPIT）。

## 43.1 特定芯片的LPIT信息

### 43.1.1 实例化信息

S32K1xx包含一个具有四个通道的LPIT模块。该器件不支持低漏电模式和等待模式。关于可用的电源模式的详情，请参见模块在可用的低功率模式下的操作。

当LPIT的功能时钟被禁用时访问它可能会产生传输错误。

### 43.1.2 LPIT/DMA周期性触发器的分配

LPIT产生周期性的触发事件给DMAMUX，如下表所示。

**表43-1.用于定期DMA触发的LPIT通道分配**

| DMA通道编号 | LPIT频道 |
|---|---|
| 0 | 0 |
| 1 | 1 |
| 2 | 2 |
| 3 | 3 |

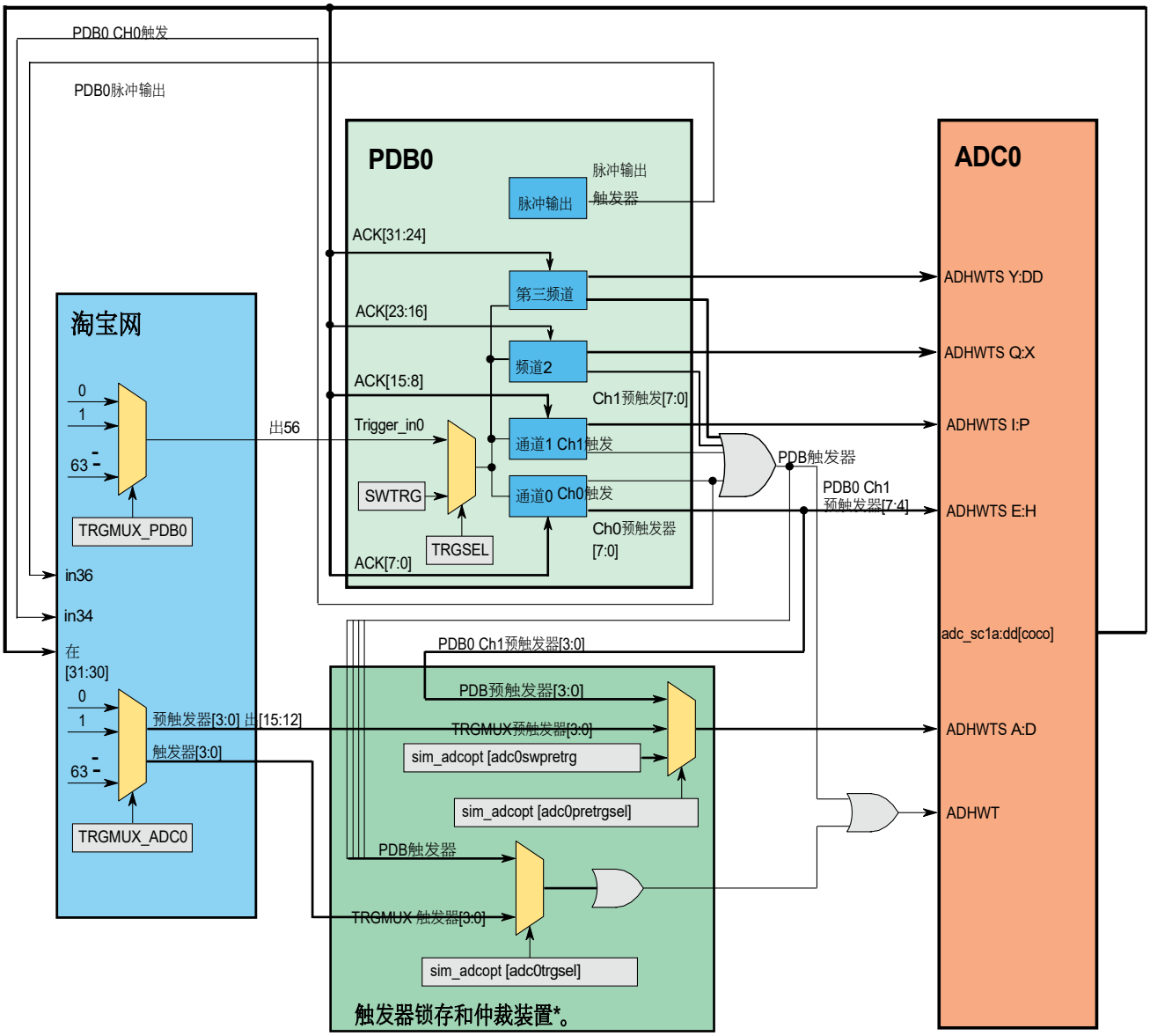
### 43.1.3 LPIT输入触发器

LPIT被设计成在其输入触发器上捕捉小脉冲，而不管其时钟频率如何，正如参考手册所附的S32K14x_Trigger_Muxing.xlsx中提到的。但为了可靠的背靠背操作，这些触发器应间隔至少10个LPIT总线时钟周期。

### 43.1.4 LPIT/ADC触发器

LPIT可以作为一个备用的ADC硬件触发源，其实现是通过TRGMUX。每个LPIT通道支持一个预触发和一个触发。LPIT通道是基于独立的计数器来实现的。当用作ADC触发源时，通道的输出产生ADC硬件触发器和预触发器。下图是一个使用LPIT触发ADC0的例子。

LPIT触发器也可以有多个来源。请参考ADC触发源。




* 该图块描述的是简化的原理图，没有显示详细的锁存。详见ADC章节的 "触发器锁存和仲裁 "一节。

**图43-1.ADC的触发方案**

## 43.2 简介

### 43.2.1 概述

低功率周期性中断定时器（LPIT）是一个多通道定时器模块，产生独立的预触发和触发输出。这些定时器通道可以单独操作，也可以链在一起。如果配置为低功率模式，LPIT可以这样操作。预触发和触发输出可用于触发器件上的其他模块。

每个定时器通道可以被配置为独立运行，并使其在比较或捕获模式下工作。

- **在比较模式下：**计时器在启用时进行递减，并产生一个预触发输出和触发输出。触发输出比预触发脉冲延迟1个时钟周期。每个定时器通道的启动、重载和重启都可以通过控制位来控制。定时器可以被配置为总是从编程的起始值开始递减，或者在选定的触发输入或前一个通道超时时递减（当通道被连锁时）。通过串联定时器通道，应用可以实现更大的超时持续时间。
- **在捕获模式下：**定时器可用于执行测量，因为当选定的触发输入被断言时，定时器值被捕获（在定时器值寄存器）。定时器可以支持一次性或多次测量（如频率测量）。

定时器通道在一个异步时钟上运行，它独立于寄存器读/写访问时钟。时钟域之间的时钟同步确保了正常操作。

## 43.2.2 框图

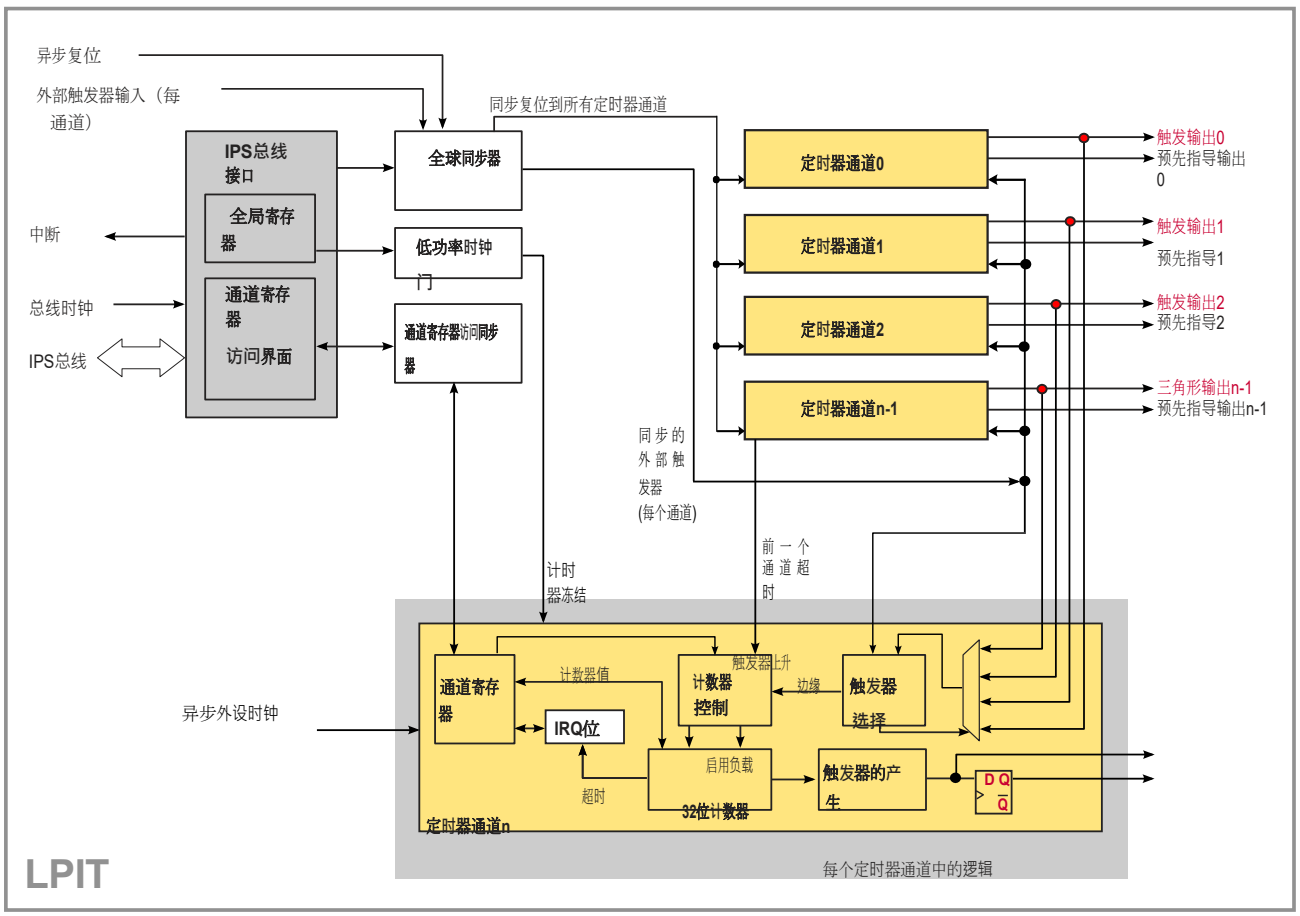



图43-2.顶层方框图

## 43.3 操作模式

LPIT模块支持下列芯片模式。

表43-2.LPIT模块支持的芯片模式

| 芯片模式 | LPIT的运作 |
|---|---|
| 运行 | 正常运行 |
| 停止 | 如果Doze Enable位（MCR[DOZE_EN]）被设置，并且LPIT使用的外部或内部时钟源在停止模式期间保持工作，则可以继续工作。 |
| 调试 | 只要调试启用位（MCR[DBG_EN]）被设置，就可以继续运行。 |

# 43.4 存储器地图和寄存器

## 43.4.1 LPIT寄存器描述

存储器地图由32位对齐的寄存器组成，可以通过8位、16位或32位访问进行访问。对保留位置的写访问将产生一个传输错误。读取保留位置也会产生传输错误，读取数据总线将显示所有的0。内存地图和完整的模块是大端格式。

该模块不会检查寄存器中编程值的正确性，软件必须确保写入的值是正确的。

### 43.4.1.1 LPIT内存图

LPIT基础地址。4003_7000h

| 偏移 | 注册 | 宽度<br>(单位：比特) | 访问 | 重置值 |
|---|---|---|---|---|
| 0h | 版本标识寄存器（VERID） | 32 | RO | 0100_0000h |
| 4h | 参数寄存器（PARAM）。 | 32 | RO | 0000_0404h |
| 8h | 模块控制寄存器（MCR） | 32 | RW | 0000_0000h |
| 丘吉尔 | 模块状态寄存器（MSR） | 32 | W1C | 0000_0000h |
| 10h | 模块中断启用寄存器（MIER） | 32 | RW | 0000_0000h |
| 14h | 设置定时器启用寄存器（SETTEN）。 | 32 | RW | 0000_0000h |
| 18h | 清除定时器启用寄存器（CLRTEN）。 | 32 | 钨钢 | 0000_0000h |
| 20h | 计时器值寄存器（TVAL0） | 32 | RW | 0000_0000h |
| 24h | 当前定时器值（CVAL0） | 32 | RO | FFFF_FFFFh |
| 28h | 定时器控制寄存器（TCTRL0）。 | 32 | RW | 0000_0000h |
| 30h | 计时器值寄存器（TVAL1） | 32 | RW | 0000_0000h |
| 34h | 当前定时器值（CVAL1） | 32 | RO | FFFF_FFFFh |
| 38h | 定时器控制寄存器（TCTRL1）。 | 32 | RW | 0000_0000h |
| 40h | 计时器值寄存器（TVAL2） | 32 | RW | 0000_0000h |
| 44h | 当前定时器值（CVAL2） | 32 | RO | FFFF_FFFFh |
| 48h | 定时器控制寄存器（TCTRL2）。 | 32 | RW | 0000_0000h |
| 50h | 计时器值寄存器（TVAL3） | 32 | RW | 0000_0000h |

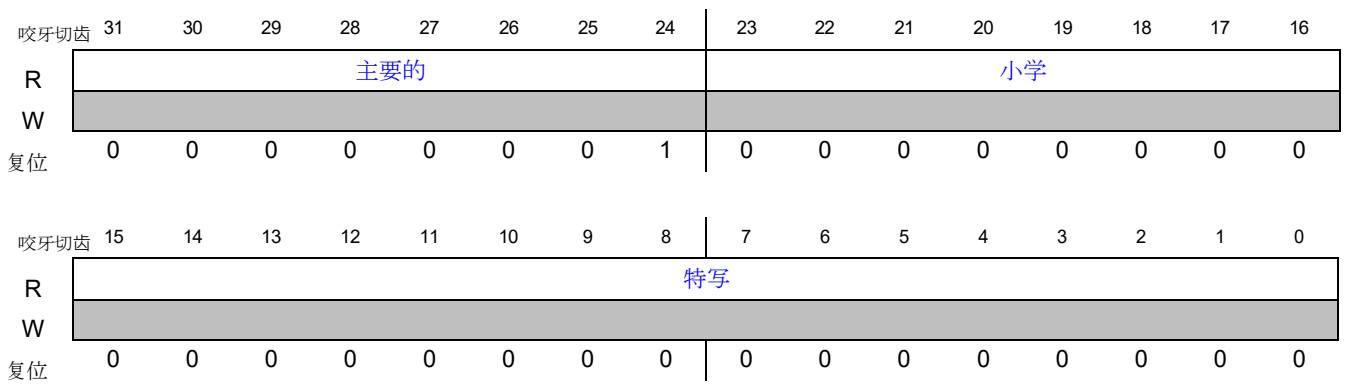
| | | | | |
|---|---|---|---|---|
| 54h | 当前定时器值（CVAL3） | 32 | RO | FFFF_FFFFh |
| 58h | 定时器控制寄存器（TCTRL3） | 32 | RW | 0000_0000h |

## 43.4.1.2 版本标识寄存器（VERID）

### 43.4.1.2.1 偏移

| 注册 | 偏移 |
|---|---|
| 虚实对照 | 0h |

### 43.4.1.2.2 图示




### 43.4.1.2.3 领域

| 场地 | 职能 |
|---|---|
| 31-24<br>主要的 | 主要版本号<br>这个只读字段返回模块规范的主要版本号。 |
| 23-16<br>小学 | 次要版本号<br>这个只读字段返回模块规范的次要版本号。 |
| 15-0<br>特写 | 特征编号<br>这个只读字段返回特征集编号。 |

## 43.4.1.3 参数寄存器（PARAM）。

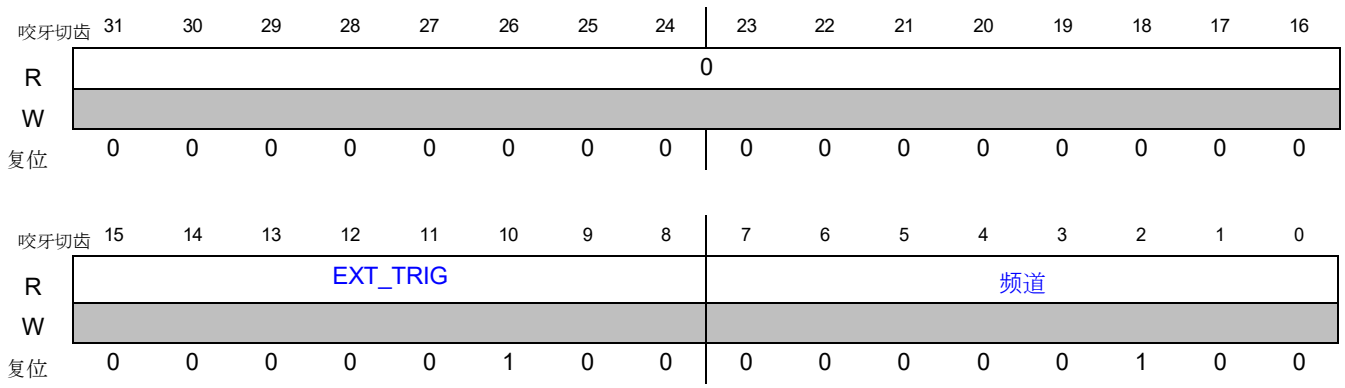
### 43.4.1.3.1 偏移

| 注册 | 偏移 |
| --- | --- |
| 朴实无华 | 4h |

#### 43.4.1.3.2 职能

该寄存器提供了在设备中包含该模块时使用的参数设置的详细信息。

#### 43.4.1.3.3 图示




#### 43.4.1.3.4 领域

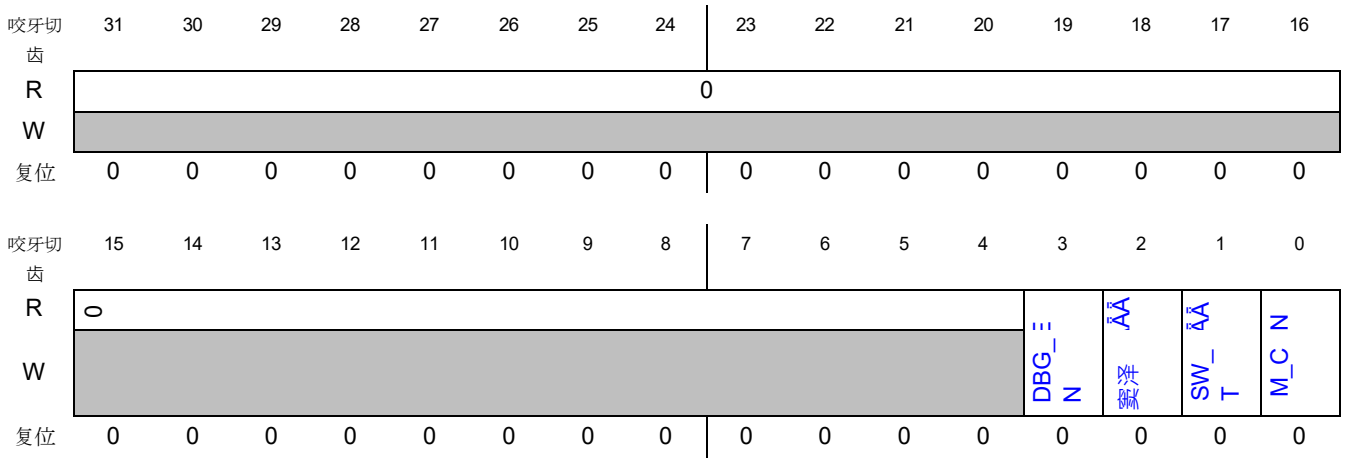
| 场地 | 职能 |
|---|---|
| 31-16<br>- | 这个只读字段是保留的，其值始终为0。 |
| 15-8<br>EXT_TRIG | 外部触发器输入的数量<br>实施的外部触发器的数量。 |
| 7-0<br>频道 | 定时器通道的数量<br>实施的定时器通道的数量。 |

### 43.4.1.4 模块控制寄存器（MCR）

#### 43.4.1.4.1 偏移

| 注册 | 偏移 |
|---|---|
| MCR | 8h |

### 43.4.1.4.2 图示




### 43.4.1.4.3 领域

| 场地 | 职能 |
|---|---|
| 31-4<br>- | 这个只读字段是保留的，其值始终为0。 |
| 3<br>呼叫中心 | 调试启用位<br>允许在设备进入调试模式时停止定时器通道 0b - 在调试模式下停止定时器通道<br>1b - 计时器通道继续在调试模式下运行 |
| 2<br>窦泽成_EN | DOZE模式启用位<br>允许定时器通道在设备进入DOZE模式时停止或继续运行 0b - 定时器通道在DOZE模式下停止运行<br>1b - 定时器通道继续在DOZE模式下运行 |
| 1<br>呼叫中心 | 软件复位位<br>重置所有通道和寄存器，除了模块控制寄存器。在被软件清除之前一直保持设置。<br>**注意：**软件必须等待4个外设时钟（用于时钟同步和复位传播），才能清除这个位。<br>0b - 计时器通道和寄存器不被重置 1b - 计时器<br>通道和寄存器被重置 |
| 0<br>M_CEN | 模块时钟启用<br>启用外围时钟到模块定时器。当写入MSR、SETTEN、CLRTEN、TCTRL和TVAL寄存器时，该位必须被肯定。两个时钟（总线时钟和外设时钟）都必须被启用，以实现时钟同步和寄存器位的更新。<br>**注意：**当M_CEN=0时，对MSR、SETTEN、CLRTEN、TCTRL和TVAL寄存器的写入，将导致该总线周期的传输错误声明。对CVAL和Reserved寄存器的写入将始终产生传输错误。<br>**注意：**在设备中可能有额外的时钟门控位，阻止外围时钟进入该模块。请确保对这些位子进行适当的配置，以启用该模块的外围时钟。<br>0b - 外围时钟到定时器被禁用 1b - 外围时 |

| | |
|---|---|
| | 钟到定时器被启用 |

## 43.4.1.5 模块状态寄存器（MSR）

### 43.4.1.5.1 偏移

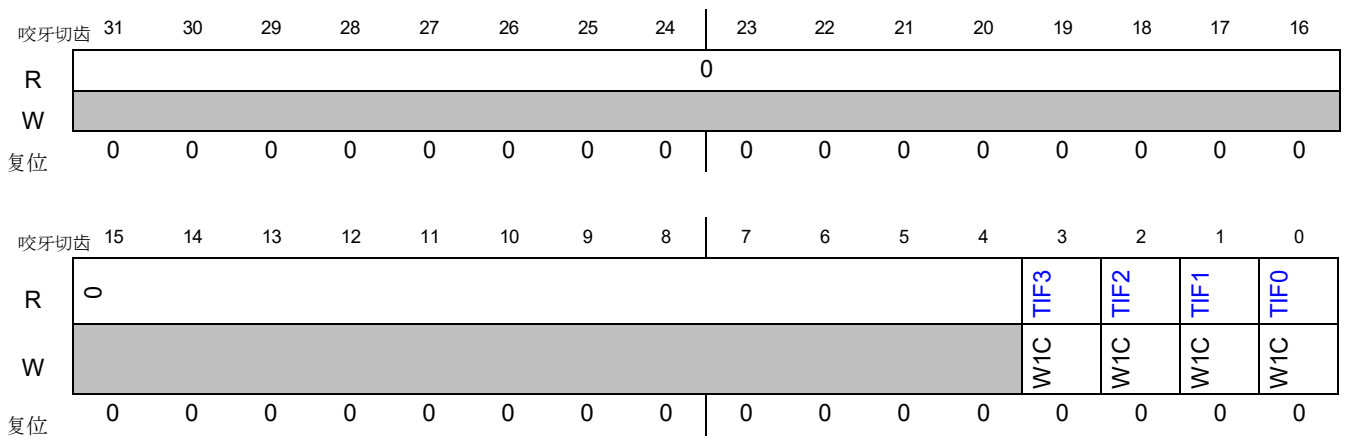
| 注册 | 偏移 |
|---|---|
| ÅÅÅ | 丘吉尔 |

### 43.4.1.5.2 职能

**注意事项**

除非MCR寄存器中的M_CEN位被设置（=1），否则读或写这个寄存器将产生一个传输错误。

### 43.4.1.5.3 图示




### 43.4.1.5.4 领域

| 场地 | 职能 |
|---|---|
| 31-4<br>- | 这个只读字段是保留的，其值始终为0。 |

| | |
|---|---|
| 3<br>TIF3 | 通道3定时器中断标志<br>在比较模式下，在定时器周期结束时设置为1。在捕获模式下，当触发器发出指令时设置为1。把逻辑1写到这个标志上会清除它。写入0则没有效果。<br>0b - 计时器没有超时 1b - 发<br>生超时。 |
| 2 | 通道2定时器中断标志 |

*表继续在下一页...*

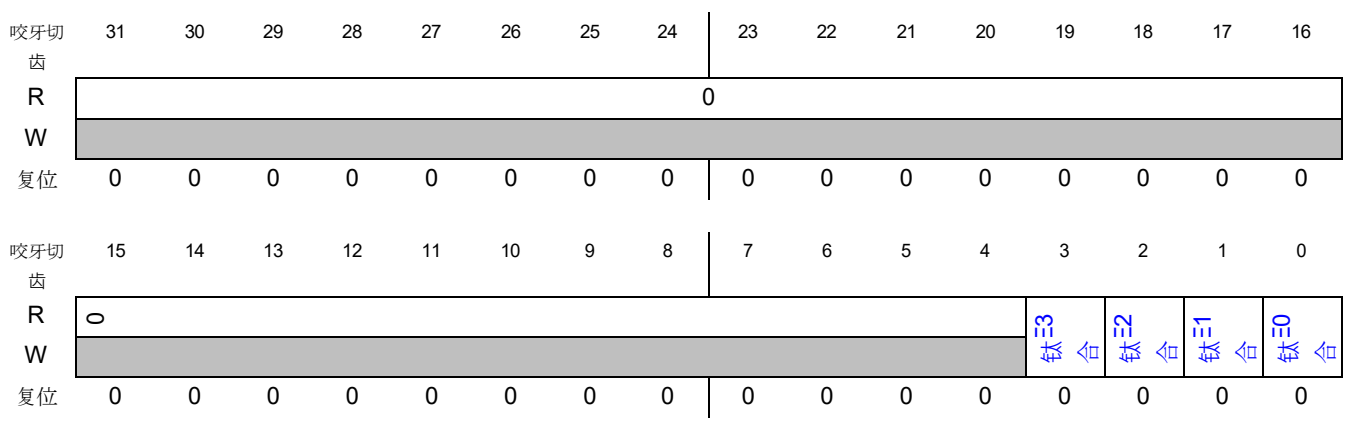
| 场地 | 职能 |
|---|---|
| TIF2 | 在比较模式下，在定时器周期结束时设置为1。在捕获模式下，当触发器发出指令时设置为1。把逻辑1写到这个标志上会清除它。写入0则没有效果。<br>0b - 计时器没有超时 1b - 发<br>生超时。 |
| 1<br>TIF1 | 通道1定时器中断标志<br>在比较模式下，在定时器周期结束时设置为1。在捕获模式下，当触发器发出指令时设置为1。将逻辑1写入该标志会清除它。写入0则没有效果。<br>0b - 计时器没有超时 1b - 发<br>生超时。 |
| 0<br>TIF0 | 通道0定时器中断标志<br>在比较模式下，在定时器周期结束时设置为1。在捕获模式下，当触发器发出指令时设置为1。把逻辑1写到这个标志上会清除它。写入0则没有效果。<br>0b - 计时器没有超时 1b - 发<br>生超时。 |

## 43.4.1.6 模块中断启用寄存器（MIER）

### 43.4.1.6.1 偏移

| 注册 | 偏移 |
|---|---|
| MIER | 10h |

### 43.4.1.6.2 图示



### 43.4.1.6.3 领域

| 场地 | 职能 |
| --- | --- |
| 31-4 | 这个只读字段是保留的，其值始终为0。 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| - | |
| 3<br>TIE3 | 通道3定时器中断启用<br>当该位被设置为1并且相应的定时器中断标志被断言时，启用中断产生。<br>0b - 中断生成被禁用 1b - 中断生成被启用 |
| 2<br>TIE2 | 通道2定时器中断启用<br>当该位被设置为1并且相应的定时器中断标志被断言时，启用中断产生。<br>0b - 中断生成被禁用 1b - 中断生成被启用 |
| 1<br>TIE1 | 通道1定时器中断启用<br>当该位被设置为1并且相应的定时器中断标志被断言时，启用中断产生。<br>0b - 中断生成被禁用 1b - 中断生成被启用 |
| 0<br>TIE0 | 通道0定时器中断启用<br>当该位被设置为1并且相应的定时器中断标志被断言时，启用中断产生。<br>0b - 中断生成被禁用 1b - 中断生成被启用 |

## 43.4.1.7 设置定时器启用寄存器（SETTEN）。

### 43.4.1.7.1 偏移

| 注册 | 偏移 |
|---|---|
| 瑟蒂恩(SETTEN) | 14h |

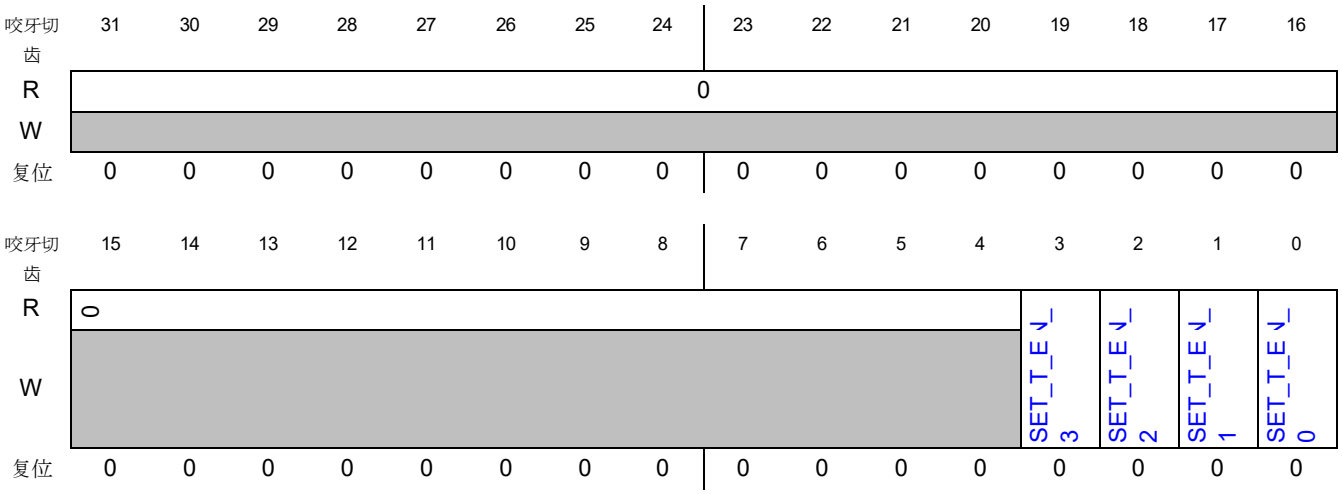
### 43.4.1.7.2 职能

该寄存器允许模拟启用定时器通道。定时器通道可以通过在相应的 TCTRLn 寄存器中的 T_EN 写 "1 "或设置该寄存器中的相应位来启用。在这个寄存器中写 "0 "是没有作用的。CLRTEN 寄存器应该被用来同时禁用定时器通道。

**注意事项**

除非MCR寄存器中的M_CEN位被设置（=1），否则读或写这个寄存器将产生一个传输错误。

### 43.4.1.7.3 图示




### 43.4.1.7.4 领域

| 场地 | 职能 |
|---|---|
| 31-4<br>- | 这个只读字段是保留的，其值始终为0。 |
| 3<br>呼叫中心：<br>SET_T_EN_3 | 设置定时器3启用<br>向该位写 "1 "将启用定时器通道3。这个位可以和TCTRL3寄存器中的T_EN位一起使用。写 "0 "不会使计数器失效。当TCTRL3中的T_EN位被设置为 "0 "或CLRTEN寄存器中的CLR_T_EN_3位被写入 "1 "时，该位将被清零。<br>0b - 没有影响<br>1b - 启用定时器通道3 |
| 2<br>呼叫中心：<br>SET_T_EN_2 | 设置定时器2启用<br>向该位写 "1 "将启用定时器通道2。这个位可以和TCTRL2寄存器中的T_EN位一起使用。写 "0 "不会使计数器失效。当TCTRL2中的T_EN位被设置为 "0 "或CLRTEN寄存器中的CLR_T_EN_2位被写入 "1 "时，该位将被清零。<br>0b - 无影响<br>1b - 启用定时器通道2 |
| 1<br>呼叫中心：<br>SET_T_EN_1 | 设置定时器1启用<br>向该位写 "1 "将启用定时器通道1。这个位可以和TCTRL1寄存器中的T_EN位一起使用。写入 "0 "不会使计数器失效。当TCTRL1中的T_EN位被设置为 "0 "或CLRTEN寄存器中的CLR_T_EN_1位被写入 "1 "时，该位将被清零。<br>0b - 无影响<br>1b - 启用定时器通道1 |

| | |
|---|---|
| 0<br>呼叫中心：<br>SET_T_EN_0 | 设置定时器0启用<br>向该位写 "1 "将启用定时器通道0。这个位可以和TCTRL0寄存器中的T_EN位一起使用。写 "0 "不会使计数器失效。当TCTRL0中的T_EN位被设置为0或CLRTEN寄存器中的CLR_T_EN_0位被写入'1'时，该位将被清零。<br>0b - 没有影响<br>1b - 启用定时器通道0 |

## 43.4.1.8 清除定时器启用寄存器（CLRTEN）。

### 43.4.1.8.1 偏移

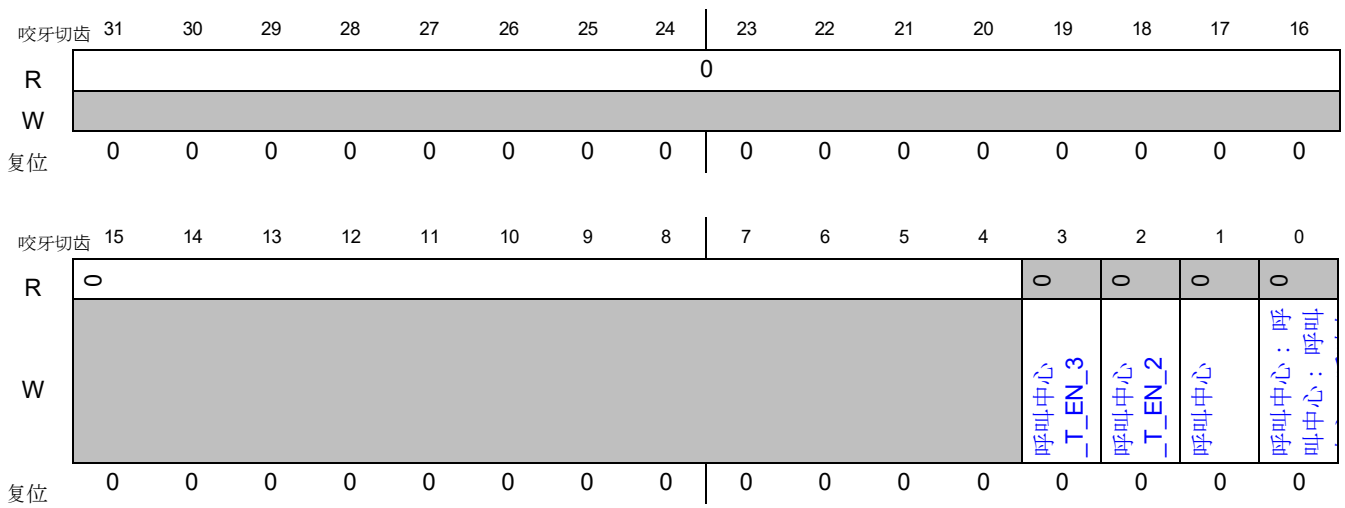
| 注册 | 偏移 |
|---|---|
| 辽宁省 | 18h |

### 43.4.1.8.2 职能

**注意事项**

除非MCR寄存器中的M_CEN位被设置（=1），否则读或写这个寄存器将产生一个传输错误。

### 43.4.1.8.3 图示




### 43.4.1.8.4 领域

| 场地 | 职能 |
|---|---|
| 31-4<br>- | 这个只读字段是保留的，其值始终为0。 |

| | |
|---|---|
| 3<br>呼叫中心<br>_T_EN_3 | 清除定时器3启用<br>向该位写 "1 "将禁用定时器通道3。这个位可以和TCTRL3寄存器中的T_EN位一起使用。写 "1 "不会启用计数器。这个位是自动清零的，总是读0。<br>0b - 没有行动<br>1b - 清除定时器通道3的T_EN位 |
| 2 | 清除定时器2启用 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| 呼叫中心_T_EN_2 | 向该位写 "1 "将禁用定时器通道2。这个位可以和TCTRL2寄存器中的T_EN位一起使用。写 "1 "不会启用计数器。这个位是自动清零的，总是读0。<br>0b - 没有行动<br>1b - 清除定时器通道2的T_EN位 |
| 1<br>呼叫中心 | 清除定时器1启用<br>向该位写 "1 "将禁用定时器通道1。这个位可以和TCTRL1寄存器中的T_EN位一起使用。写 "1 "不会启用计数器。这个位是自动清零的，总是读0。<br>0b - 没有行动<br>1b - 清除定时器通道1的T_EN位 |
| 0<br>呼叫中心： 呼叫中心： 呼叫中心： 呼叫中心：CLR_T_EN_0 | 清除定时器0启用<br>向该位写 "1 "将禁用定时器通道0。这个位可以和TCTRL0寄存器中的T_EN位一起使用。写 "1 "不会启用计数器。这个位是自动清零的，总是读0。<br>0b - 没有行动<br>1b - 清除定时器通道0的T_EN位 |

## 43.4.1.9 计时器值寄存器（TVAL0 - TVAL3）。

### 43.4.1.9.1 偏移

对于n=0到3。

| 注册 | 偏移 |
|---|---|
| ĂĂĂ | 20h + (n × 10h) |

### 43.4.1.9.2 职能

- 在比较模式下，TVALn寄存器选择定时器通道的超时周期。
- 在捕获模式下，当触发器断电时，TVALn寄存器被加载了计数器的值。

**注意事项**

如果MCR寄存器中的M_CEN位没有设置（=1）， 那么读或写TVALn寄存器将产生一个传输错误。

## 43.4.1.1322.3 图示

| 咬牙切齿 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R<br>W | TMR_VAL | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 咬牙切齿 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R<br>W | TMR_VAL | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

### 43.4.1.9.4 领域

| 场地 | 职能 |
|---|---|
| 31-0<br>TMR_VAL | 计时器值<br>• 在比较模式下，TMR_VAL是定时器通道的起始值。定时器将倒计时，直到它达到0，然后定时器将产生一个中断并再次加载TVALn寄存器的值。向TVALn寄存器写入一个新的值不会重新启动定时器通道，相反，新的值将在定时器过期后加载。要中止当前的定时器周期并以新的值开始一个定时器周期，必须禁用定时器通道并再次启用。<br>• 在捕获模式下，每当触发器断电时，TVALn寄存器会存储计数器的反值。<br>0000000000000000000000000000b - 在比较模式下的无效负载值。<br>0000000000000000000000000001b - 在比较模式下的无效加载值。<br>0000000000000000000000000000000000000000000000000000000000000000000000000000<br>0000000000000000000000000000000000000000000000000000000000000000000000000000<br>0000000000000000000000000000000000000000000000000000000000000000000000000000<br>00000000000000000000000000000000000000000000000000000000000000000000000000<br>加载（比较模式）或定时器的值（捕获模式）。 |

## 43.4.1.10 当前定时器值（CVAL0 - CVAL3）。

### 43.4.1.10.1 偏移

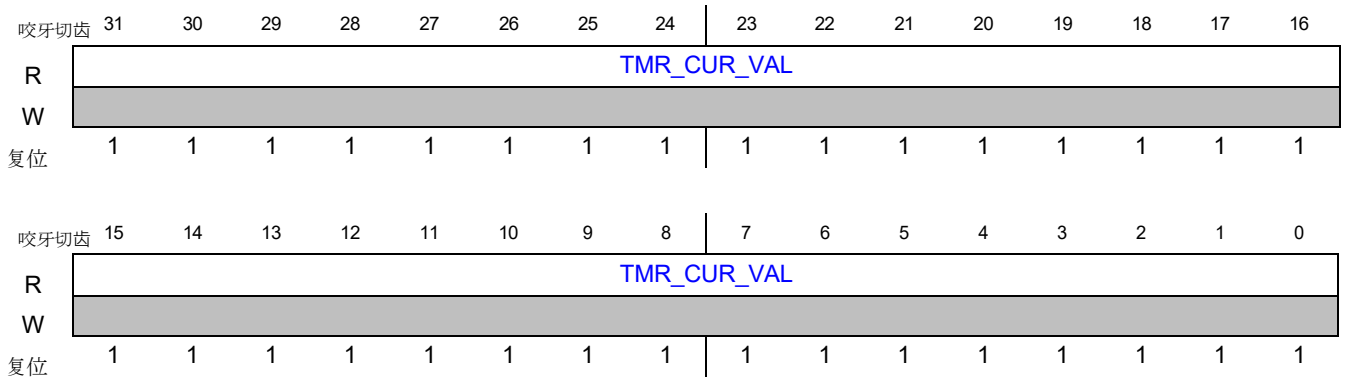
对于n=0到3。

| 注册 | 偏移 |
|---|---|
| ĀĀĀ | 24小时+(n×10小时) |

#### 43.4.1.10.2 职能

这些寄存器显示当前的定时器计数器值。

### 43.4.1. 图示
### 43.4.1.10.3 图示




### 43.4.1.10.4 领域

| 场地 | 职能 |
|---|---|
| 31-0<br>TMR_CUR_VAL | 当前的定时器值<br><br>代表当前的定时器值，如果定时器被启用。 |

## 43.4.1.11 计时器控制寄存器（TCTRL0 - TCTRL3）。

### 43.4.1.11.1 偏移

对于n=0到3。

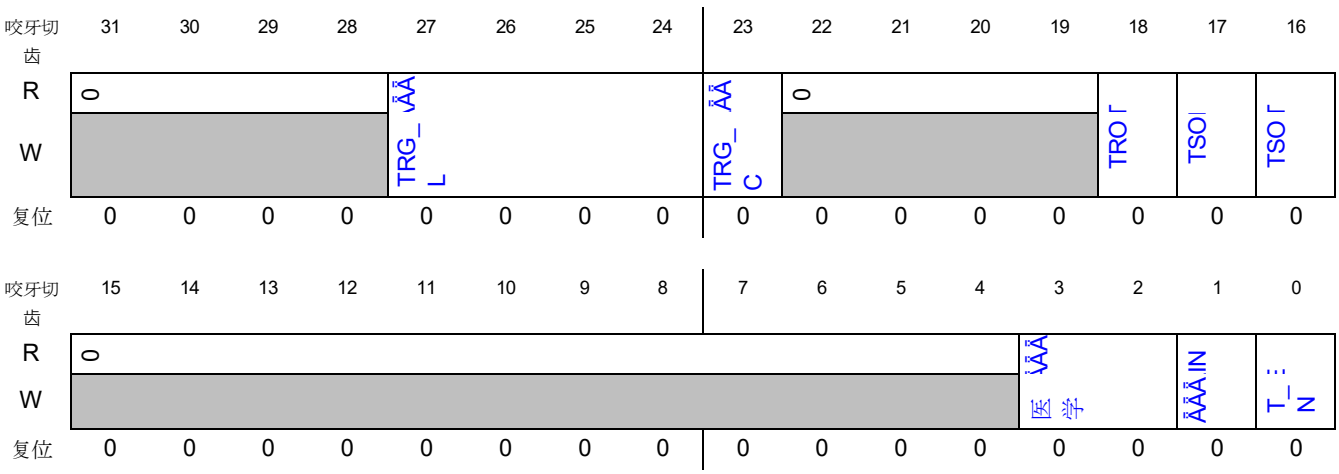
| 注册 | 偏移 |
|---|---|
| 编码：TCTRLn | 28h + (n × 10h) |

### 43.4.1.11.2 职能

这些寄存器包含每个定时器通道的控制位。

**注意事项**

除非MCR寄存器中的M_CEN位被设置（=1），否则读或写这个寄存器将产生一个传输错误。




## 43.4.1.11.4 领域

| 场地 | 职能 |
|---|---|
| 31-28<br>- | 这个只读字段是保留的，其值始终为0。 |
| 27-24<br>TRG_SEL | 触发器选择<br>选择用于启动和/或重新加载LPIT定时器的触发器。这个字段只有在LPIT定时器通道被禁用时才可以改变。TRG_SRC位为每个通道选择内部和外部触发信号。<br>TRG_SEL位从TRG_SRC选择的内部或外部触发器集合中选择一个触发器。<br>0000-0011b - 选择定时器通道0-3的触发源。0100-1111b - 保留。 |
| 23<br>TRG_SRC | 触发源<br>在内部或外部触发源之间进行选择。最终的触发器由TRG_SEL选择，取决于内部触发器或外部触发器中的哪个触发源由TRG_SRC选择。<br>关于可用的外部触发器选项，请参考芯片配置部分。如果一个通道没有相关的外部触发器，那么该通道的这个位应该被设置为1。<br>0b - 在外部选择触发源<br>1b - 选择的触发源是内部触发器 |
| 22-19<br>- | 这个只读字段是保留的，其值始终为0。 |
| 18<br>TROT | 触发器上的定时器重装<br>当设置时，当在选定的触发器输入上检测到一个上升沿时，LPIT定时器将重新加载。如果在调试模式或DOZE模式下禁用LPIT（DOZE_EN或DBGEN），触发输入将被忽略。<br>= 0) |

## 43.4.1. 图示

| | |
|---|---|
| 1326.3 | 0b - 计时器不会在选定的触发器上重新加载<br>1b - 计时器将在选定的触发器上重新加载 |
| 17<br>TSOI | 中断时定时器停止<br>控制通道定时器是否会在它（通道定时器）超时后停止。<br>0b - 通道定时器在超时后不停止。 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | 1b - 通道定时器将在超时后停止，通道定时器将根据TSOT重新启动。当TSOT=0时，通道定时器将在检测到T_EN位的上升沿后重新启动（这意味着定时器通道被禁用，然后启用）；当TSOT=1时，通道定时器将在检测到所选触发器的上升沿后重新启动。 |
| 16<br>TSOT | 触发器上的定时器启动<br>该位控制定时器何时开始递减。<br>0b - 计时器根据重启条件立即开始递减（由TSOI位控制） 1b - 计时器在检测到选定触发器的上升沿时开始递减 |
| 15-4<br>- | 这个只读字段是保留的，其值始终为0。 |
| 3-2<br>模式 | 定时器操作模式<br>配置通道定时器的工作模式。模式位控制定时器如何递减。更多细节见功能描述。<br>00b - 32位周期性计数器<br>01b - 双16位周期性计数器 10b - 32<br>位触发器累积器 11b - 32位触发器<br>输入捕捉器 |
| 1<br>链条 | 链式通道<br>当启用时，当通道N-1触发器断电时，定时器通道将递减。通道0不能被链起来。<br>0b - 通道连锁被禁用。通道定时器独立运行。<br>1b - 启用通道链。计时器在前一个通道的超时时递减 |
| 0<br>T_EN | 定时器启用<br>启用或禁用定时器通道 0b - 定时器通<br>道被禁用 1b - 定时器通道被启用 |

# 43.5 职能描述

## 43.5.1 初始化

可以用以下步骤来初始化LPIT模块

- 通过设置MCR寄存器中的M_CEN位启用外设时钟。

**注意事项**

- 当M_CEN=0时，对某些寄存器的写入将导致对该总线访问的传输错误的断言。这些寄存器是MSR、SETTEN、CLRTEN、TVAL和TCTRL。对CVAL

43.4.1.1328.3 图示 和保留寄存器的写入将导致

无论M_CEN的位值如何，都会产生一个传输错误。对这些寄存器的读取可以不考虑M_CEN的位值。

- 器件中可能有额外的时钟门控位，对该模块的外围时钟进行门控。确保软件在启用该模块的时钟时，除了M_CEN之外还配置了这些位。

- 等待4个外设时钟周期，以便有时间进行时钟同步和复位去势。
- 对于每个要启用的定时器通道，在TCTRLn寄存器中配置定时器的工作模式（MODE位）、触发源选择（TRG_SEL & TRG_SRC）和触发控制位（TROT, TSOT, TSOI位）。
- 通过设置相应通道的TCTRLn寄存器中的CHAIN位来配置要被链起来的通道。
- 对于配置在比较模式下的通道，通过在TVAL寄存器中为这些通道设置适当的值来设置定时器超时值。
- 在MIER寄存器中为那些需要在定时器超时时产生中断的通道配置TIEn位。
- 通过设置MCR寄存器中的DBG_EN和DOZE_EN位，配置模块的低功耗模式功能。这对所有的定时器通道来说是通用的。
- 通过设置相应通道的TCRTLn寄存器中的T_EN位来启用通道定时器。
- 对于配置为捕获模式的通道，当通道超时时，可以从TVALn寄存器中读取定时器值。
- 在任何时候，任何通道的定时器的当前值都可以通过读取相应通道的CVALn寄存器来读取。
- MSR寄存器中的定时器中断标志位（TIFn）在定时器超时时被断言。这些位可以通过写 "1 "来清除。

## 43.5.2 计时器模式

计时器模式是通过在TCTRLn寄存器的MODE位上设置一个适当的值来配置的。支持的定时器模式有

- *32位周期性计数器。*在这种模式下，计数器将加载并递减到0，然后设置定时器中断标志并断言输出预触发。
- *双16位周期性计数器。*在这种模式下，计数器将加载，然后低16位将递减到0，这将断定输出预触发。上面的16位将递减到0，这将否定输出预触发器并设

置定时器中断标志。

- *32位触发器累积器。*在这种模式下，计数器将在第一个触发器上升沿加载，然后在每个触发器上升沿向下递减到0。然后，它将设置定时器中断标志并断言输出预触发。
- *32位触发器输入捕捉。*在这种模式下，计数器将加载0xFFFF_FFFF，然后向下递减到0。如果检测到一个触发器上升沿，那么它将在加载值寄存器中存储当前计数器值的反值，设置定时器中断标志，并断言输出预触发。

触发器控制位（TSOT、TSOI、TROT）进一步控制定时器的操作，这些位控制定时器的加载、重载、启动和重启。

**注意事项**

- 触发输出比预触发输出晚一个外围定时器时钟周期。触发器输出和预触发器输出同时解除引脚。
- 预触发输出被断言为2个时钟周期，触发输出被断言为1个时钟周期（除了在16位周期性计数器模式下，预触发和触发都被断言为许多周期，取决于TMR_VAL[31:16]）。
- *基于外部触发器的*定时器变化在实际外部触发器断言后的4个外围时钟后生效（由于时钟同步、上升沿检测和定时器更新）。

### 43.5.3 计时器的触发器控制

TSOT、TROT、TSOI和TRG_SEL、TRG_SRC位控制触发器输入如何影响定时器的工作。TRG_SEL从所有其他通道的触发输出中选择该通道的输入触发。TRG_SRC进一步选择所选择的内部触发器和外部触发器的输入通道。

选择的触发器会影响基于TROT、TSOI和TSOT位的定时器操作。由于这些位的作用，其行为如下。

- 如果TSOI=1，计数器在TIF断言时停止。需要触发（如果TSOT = 1）或T_EN上升沿（如果TSOT = 0），以重新加载和递减。如果TSOI=0，计数器在超时后不停止。

- 如果TROT =
  1，计数器在每次触发时被加载；否则，计数器在每个T_EN上升沿或超时上升沿被加载（超时不用于捕获模式）。
- 如果TSOT =
  1，计数器将在触发时开始递减。随后的触发会被忽略，直到计数器超时。如果TSOT=0，计数器将从下一个时钟沿立即递减。当通道为链式或捕捉模式时，TSOT没有影响。

这些位在不同的定时器模式下对定时器的操作有不同的影响。

- 在32位周期性计数器和双16位周期性计数器模式下，所有位（TSOT、TSOI和TROT）都会影响上述的定时器操作。
- 在32位触发累积器模式下，只有TSOI位控制定时器功能。TROT和TSOT位对定时器的运行没有影响。
- 在32位输入触发捕捉模式下，TSOI和TROT位控制定时器功能。TSOT位对定时器的运行没有影响。

## 43.5.4 通道链

单个定时器通道可以被链在一起以实现更大的超时值。将定时器通道连锁起来会使它们以 *"嵌套循环"的*方式工作，从而导致有效的超时值为$TVAL_{CHn} \times (TVAL_{CHn-1} + 1)$。

通过设置相应通道的$TCTRL_{CHn}$寄存器中的CHAIN位，可以将这些通道连锁起来。当一个通道被连锁时，该通道的定时器在前一个通道的超时脉冲上递减，与定时器的模式（MODE位）无关。如果通道的定时器（通道'n'）与前一个通道的定时器（通道'n-1'）连锁，则TSOT位没有任何影响。

## 43.5.5 详细的时间安排

**注意事项**

这些部分的时序图不是*周期精确的*（也就是说，有些周期可能没有显示），但时序图确实显示了几个时钟周期内的定时器通道行为。

#### 45.5.5.1 模式=00：32位定期计数器（比较模式）。

模式=00

32位周期性计数器（比较模式）

情况1：TSOT=0,TROT=0,TSOI=0,CHAIN=0

-对于需要重复中断与重载的用例来说

-触发输出将有相等的周期

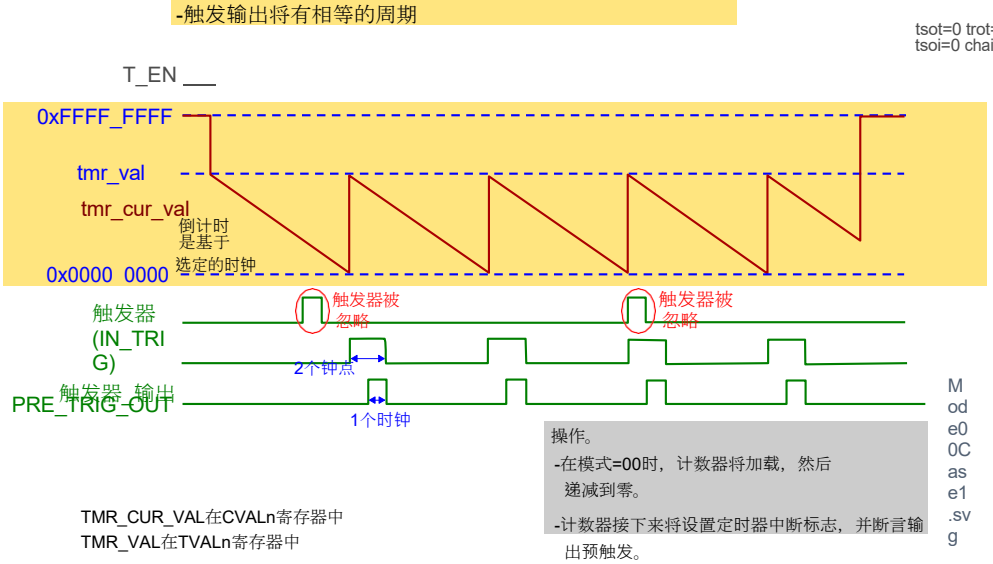



**图43-3.情况1：TSOT = 0；TROT = 0；TSOI = 0；CHAIN = 0**

模式=00

32位周期性计数器（比较模式）

情况2：TSOT=0,TROT=0,TSOI=1,CHAIN=0

-适用于一次性触发模式

-当T_EN被设为1时，定时器将再次启动。

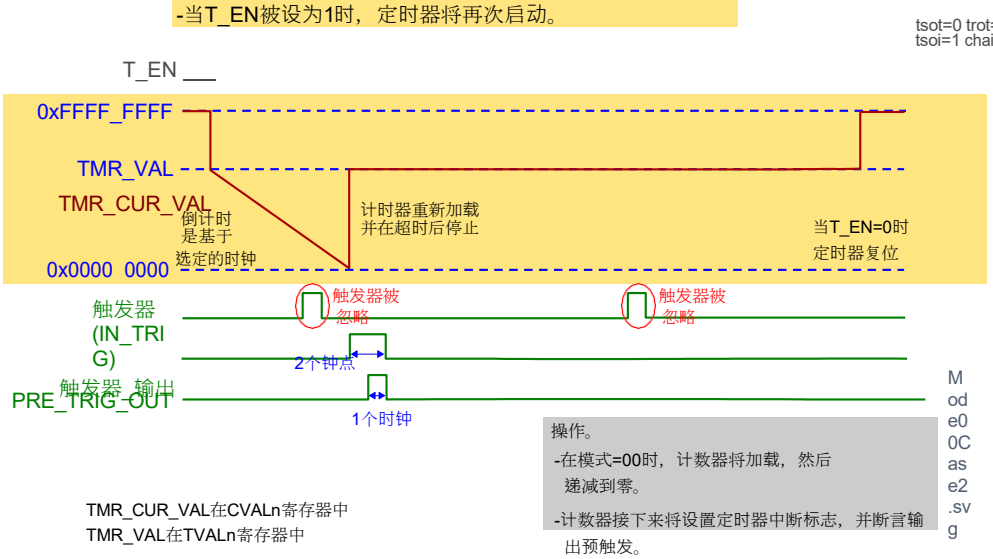



**图43-4.情况2：TSOT = 0；TROT = 0；TSOI = 1；CHAIN = 0**

模式=00

32位周期性计数器（比较模式）

情况3：TSOT=0,TROT=1,TSOI=0,CHAIN=0

-对于需要重复中断与重载的用例来说

-触发输出将有不等的周期

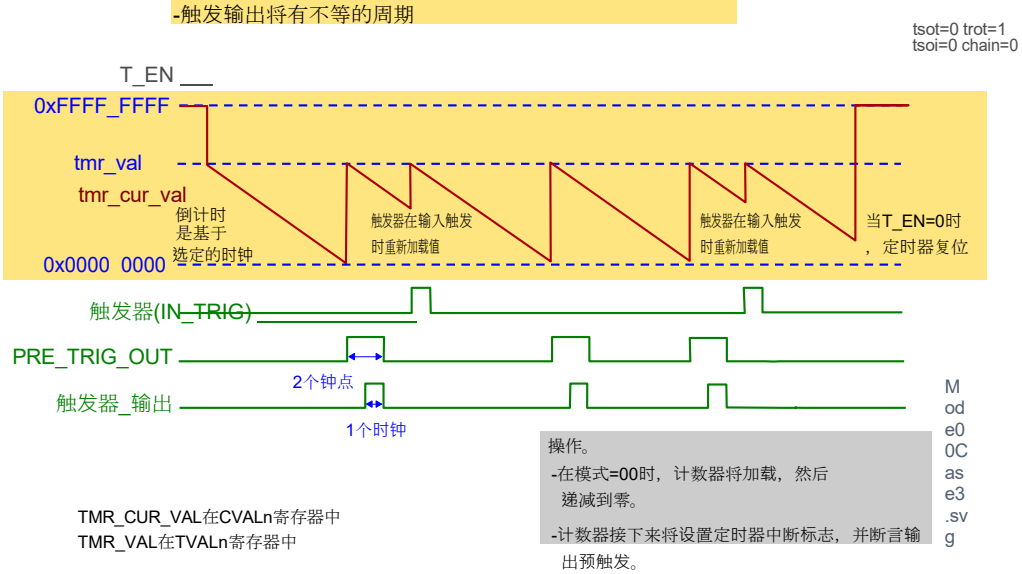



**图43-5.情况3：TSOT = 0；TROT = 1；TSOI = 0；CHAIN = 0**

模式=00

32位周期性计数器（比较模式）

情况4：TSOT=0,TROT=1,TSOI=1,CHAIN=0

-对于在定时器模式超时前重新加载的单发定时器

-当T_EN被设为1时，定时器将再次启动。

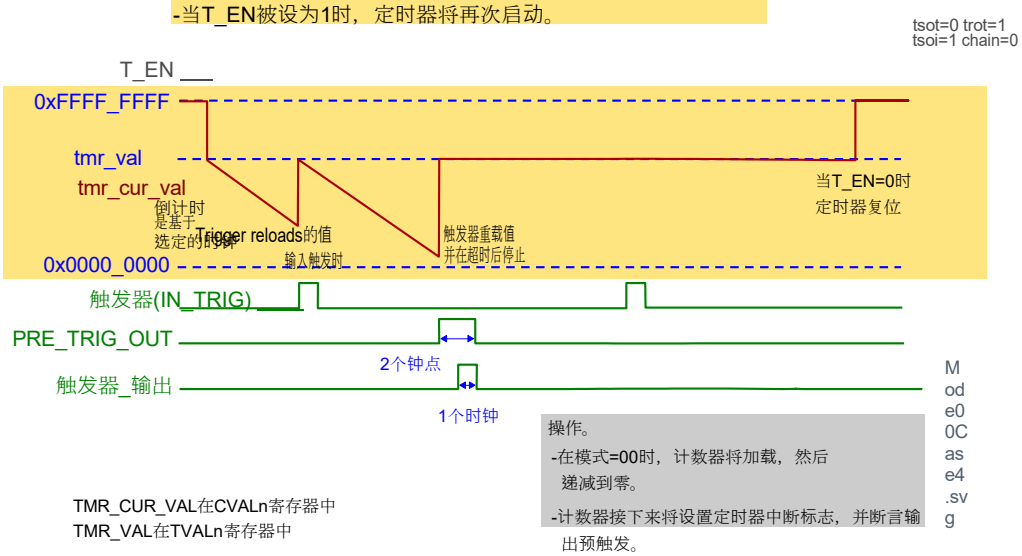



**图43-6.情况4：TSOT = 0；TROT = 1；TSOI = 1；CHAIN = 0**

模式=00

32位周期性计数器（比较模式）

情况5：TSOT=1,TROT=0,TSOI=0,CHAIN=0

-用于在预定义的事件（输入触发）后产生周期性中断。

-输出触发器将有相等的周期

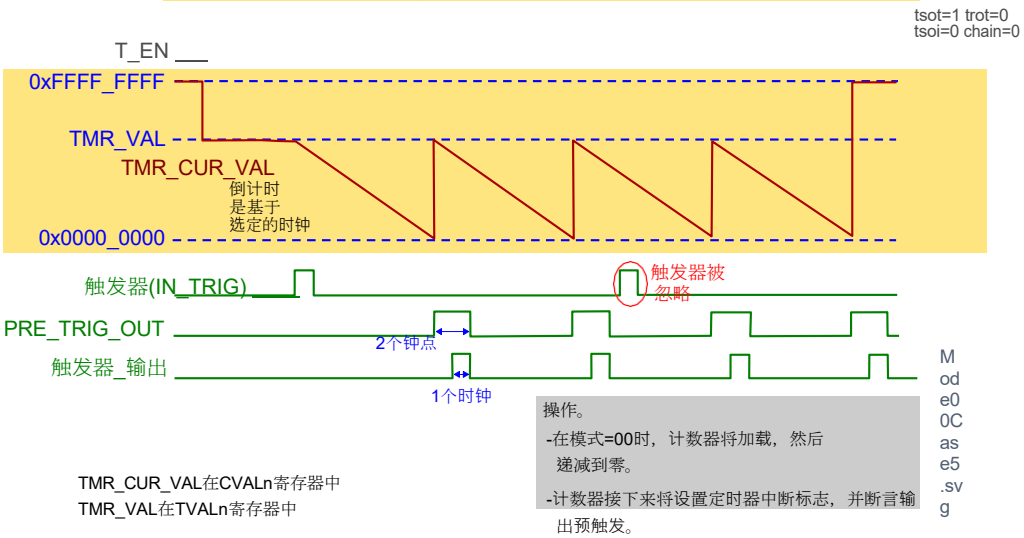



**图43-7.情况5：TSOT = 1；TROT = 0；TSOI = 0；CHAIN = 0**

模式=00

案例6:TSOT=1,Trot=0,TSOI=1,CHAIN=0

32位周期

-触发的单次定时器模式

-输出触发周期将取决于输入触发。

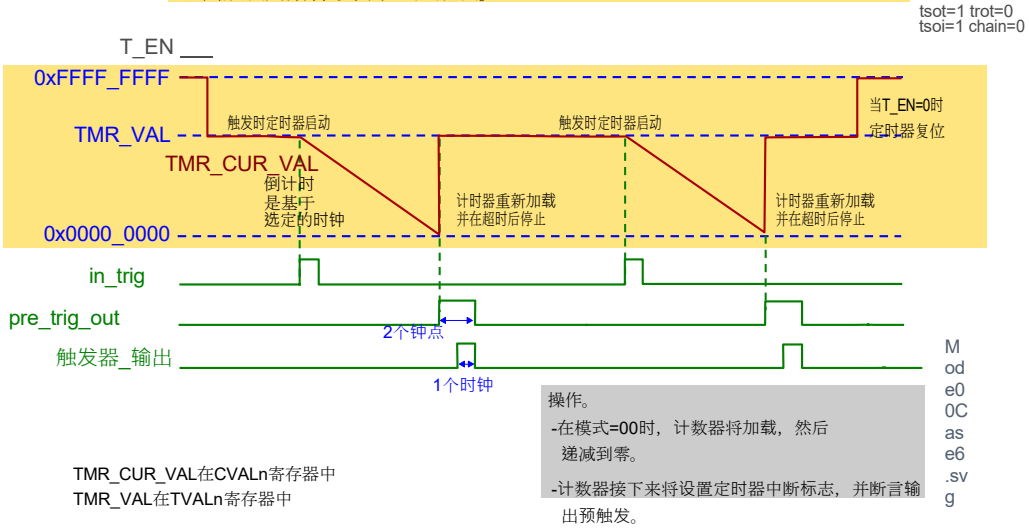



**图43-8.案例6:TSOT = 1; Trot = 0; TSOI = 1; Chain = 0**

模式=00

32位周期性计数器（比较模式）

情况7：TSOT=1,TROT=1,TSOI=0,CHAIN=0

-重复中断与重载定时器模式

-输出触发器将有不平等的周期

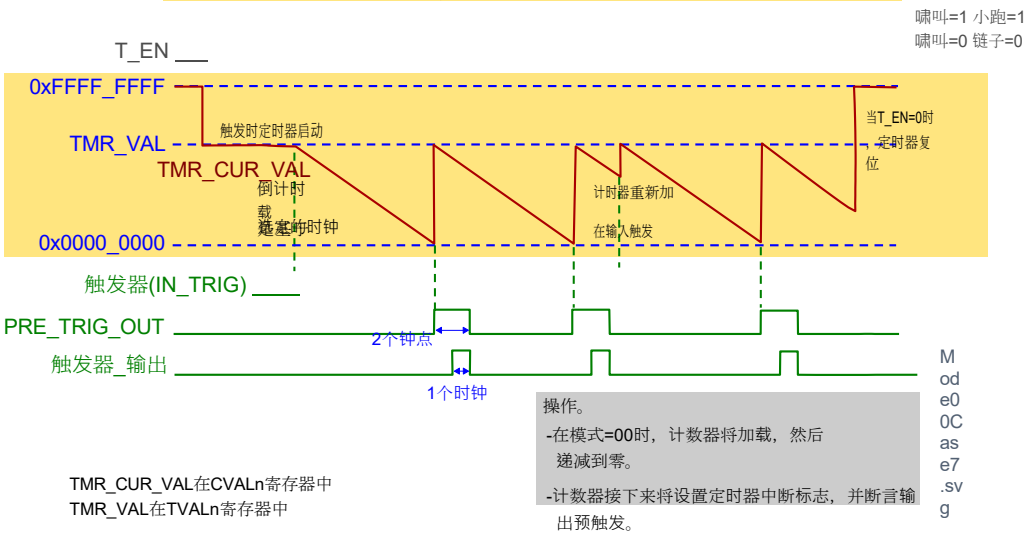



**图43-9.情况7：TSOT = 1；TROT = 1；TSOI = 0；CHAIN = 0**

模式=00

案例8:TSOT=1,Trot=1,TSOI=1,Chain=0

32位周期

-所示情况为非周期性输入触发（可能不是用例）。

-如果输入触发器是周期性的，并且大于定时器的超时，那么这将与TROT=0相同。

-如果输入触发器是周期性的，并且小于定时器的超时，那么定时器将永远不会超时。
并将总是在输入触发时重新加载（不是一个有效的用例）。

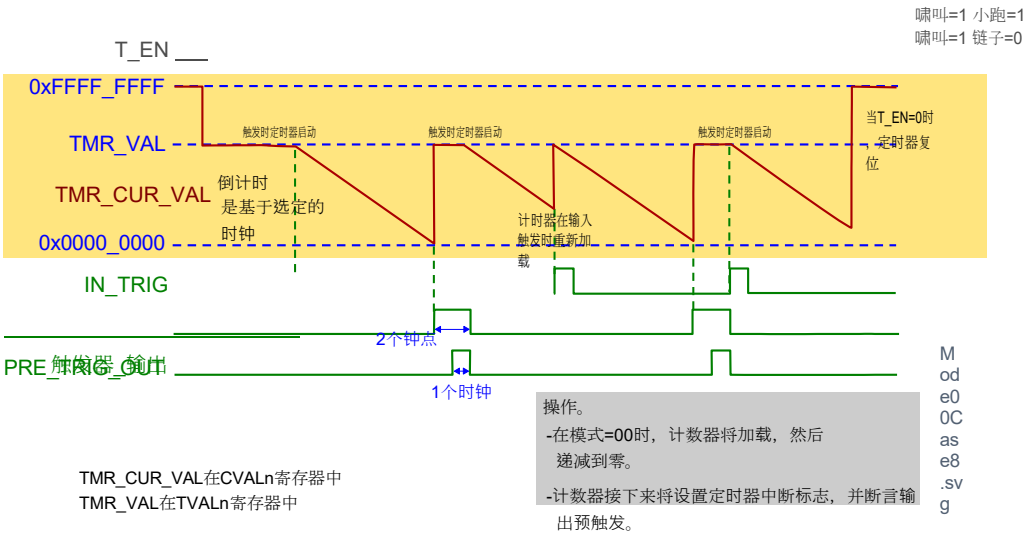


图43-10.案例8:TSOT = 1; Trot = 1; TSOI = 1, Chain = 0

## 43.5.5.2 模式=01：16位双周期计数器（比较模式）。

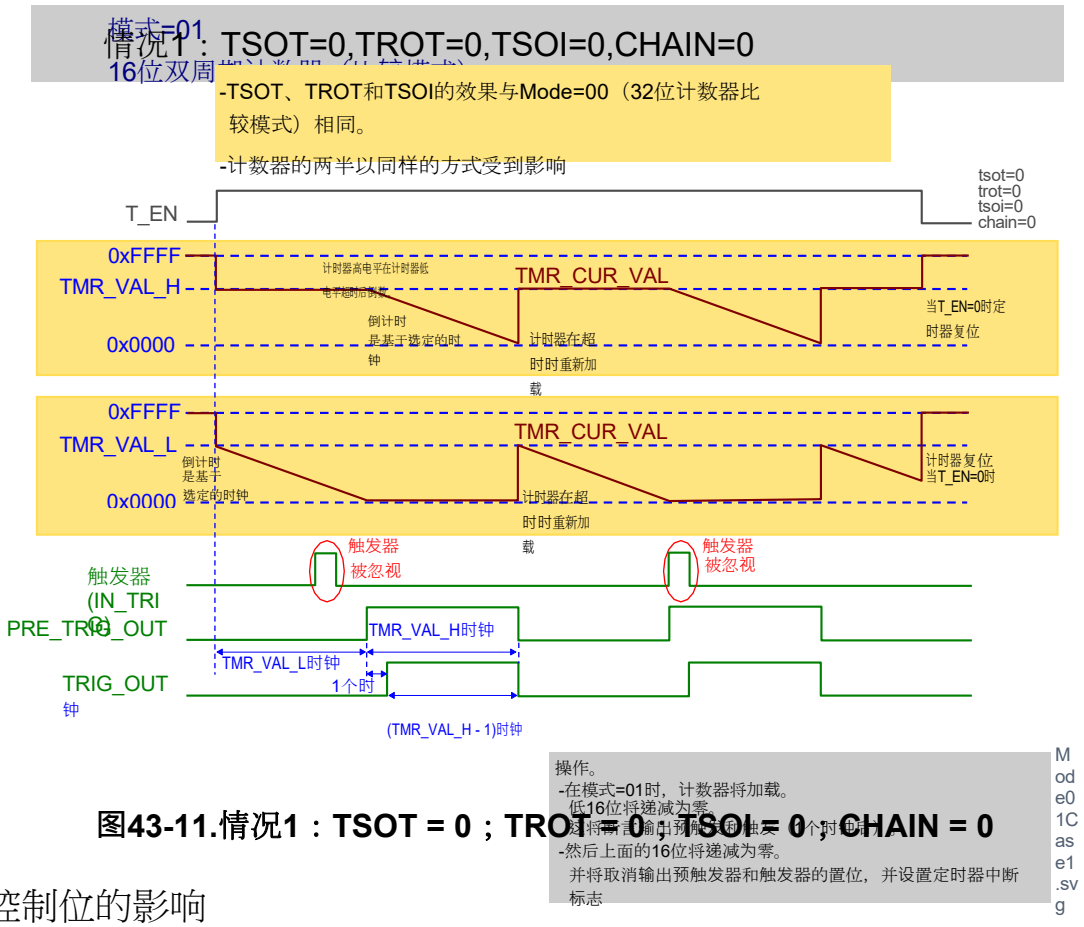



**图43-11.情况1：TSOT = 0；TROT = 0，TSOI = 0，CHAIN = 0**

定时器控制位的影响

- 定时器控制位的效果与Mode=00相同。请参考Mode=00的个别数字。
- 当{TMR_H,TMR_L}=0x0000_0000时，定时器中断（超时）就会发出信号。
- 模式=01的行为在下表中有解释。

**表43-3.模式=01 定时器控制位 功能**

| TSOT | TROT | TSOI | 对定时器的影响 |
|---|---|---|---|
| 0 | 0 | 0 | 此前已涵盖。 |
| 0 | 0 | 1 | 单次拍摄模式。两个定时器在第一次倒计时和超时后停止。计时器不会再计数，直到T_EN再次变成1。 |

*表格在下一页继续...*

## 表43-3.模式=01 定时器控制位 功能（续）

| TSOT | TROT | TSOI | 对定时器的影响 |
|---|---|---|---|
| 0 | 1 | 0 | 两个定时器将在触发器上升沿重新加载TMR_VAL。输出触发器将在重载时清除，如果断言的话 |
| 0 | 1 | 1 | 可重新加载的单发模式。如果触发发生在超时之前，那么两个定时器都会重新加载并倒计时，如图所示，超时后定时器停止。超时后的触发断言只是将TMR_VAL重新加载到定时器中。直到T_EN再次被设置为1，定时器才会再次计数。 |
| 1 | 0 | 0 | 在T_EN上升后，在第一个触发器的上升沿之前，定时器不会启动。后续的触发器将没有任何影响。 |
| 1 | 0 | 1 | 在T_EN上升后，在第一个触发器的上升沿之前，定时器不会启动。计时器在超时断言后停止计数，直到检测到一个新的触发器上升沿才会再次开始计数。 |
| 1 | 1 | 0 | 在T_EN上升后，在第一个触发器的上升沿之前，定时器不会启动。随后的触发器将导致定时器将TMR_VAL重新加载到两个计数器中。如果断言，输出触发器将在重载后清除。 |
| 1 | 1 | 1 | 在T_EN上升后，定时器直到第一个触发器的上升沿才开始。计时器在超时断言后停止计数。触发器的上升沿会使定时器重新加载，然后倒数。 |

## 43.5.5.3 模式=10：32位触发累加器模式

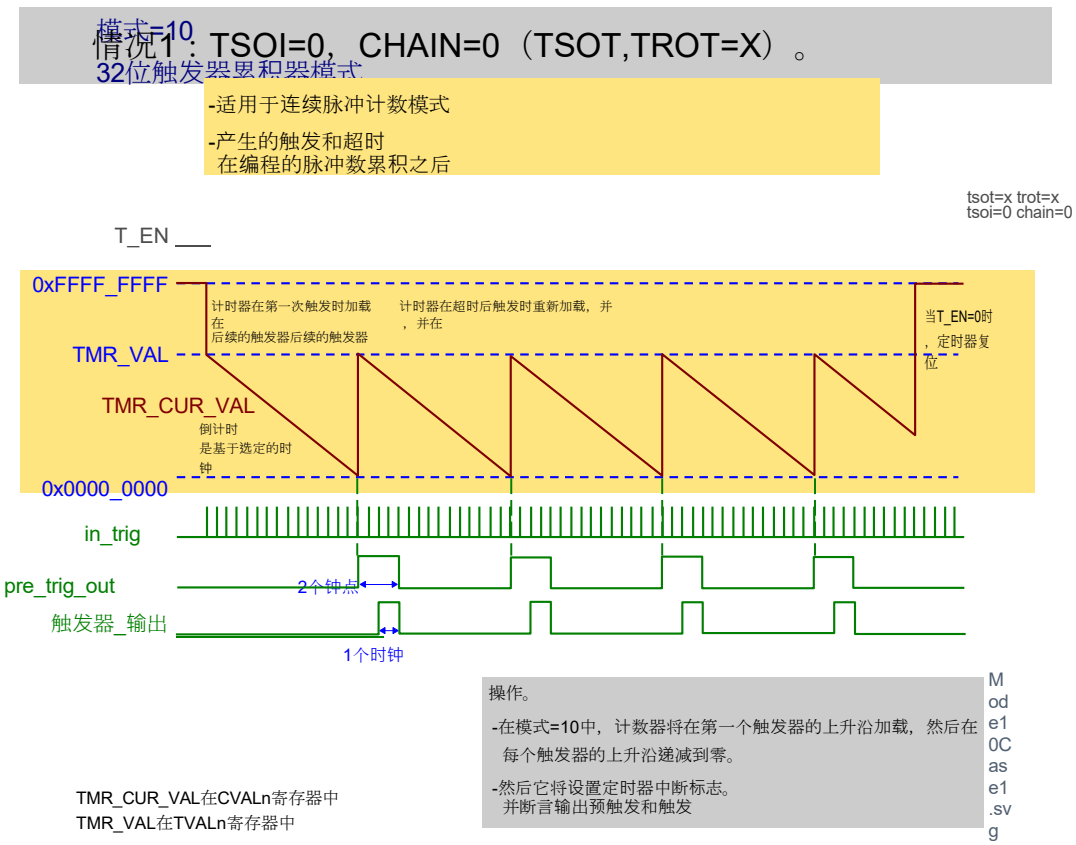



**图43-12.情况 1: TSOI = 0; CHAIN = 0 (TSOT, TROT = X)**

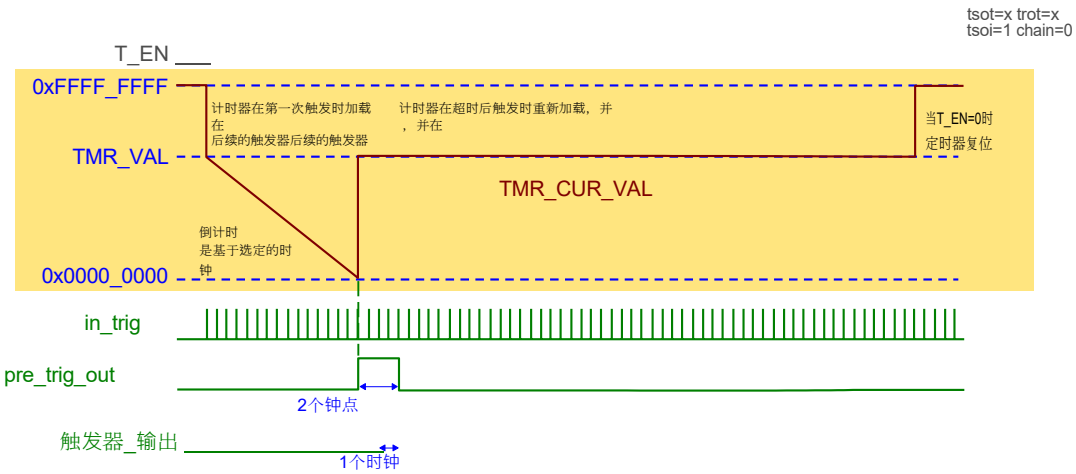
模式=10

32位触发器累积器模式

情况2：TSOI=1，CHAIN=0（TSOT,TROT=X）。

-适用于一-热脉冲计数模式

-产生的触发和超时
在编程的脉冲数累积之后




操作。

-在模式=10中，计数器将在第一个触发器的上升沿加载，然后在
每个触发器的上升沿递减到零。

-然后它将设置定时器中断标志。
并断言输出预触发和触发

Mode10Case2.svg

**图43-13.情况 2: TSOI = 1; CHAIN = 0 (TSOT, TROT = X)**

## 43.5.5.4 模式=11：32位触发器捕获模式

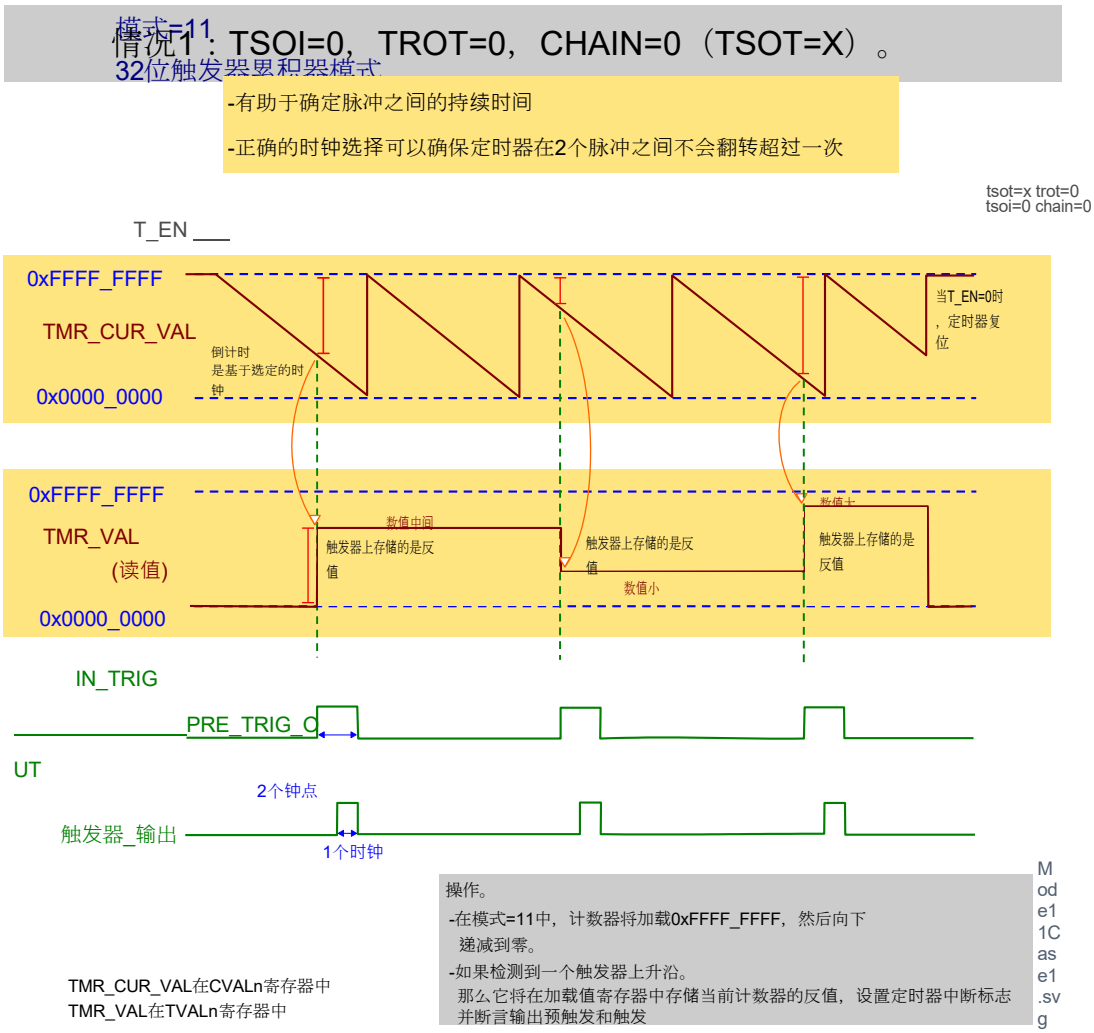



**图43-14.情况 1: TSOI = 0; TROT = 0; CHAIN = 0 (TSOT = X)**

模式=11

32位触发器累积器模式

情况2：TSOI=0，TROT=1，CHAIN=0（TSOT=X）。

-有助于确定脉冲之间的持续时间

-选择一个快速的定时器时钟可以提供精确的测量，但也可能导致脉冲之间的定时器翻转。

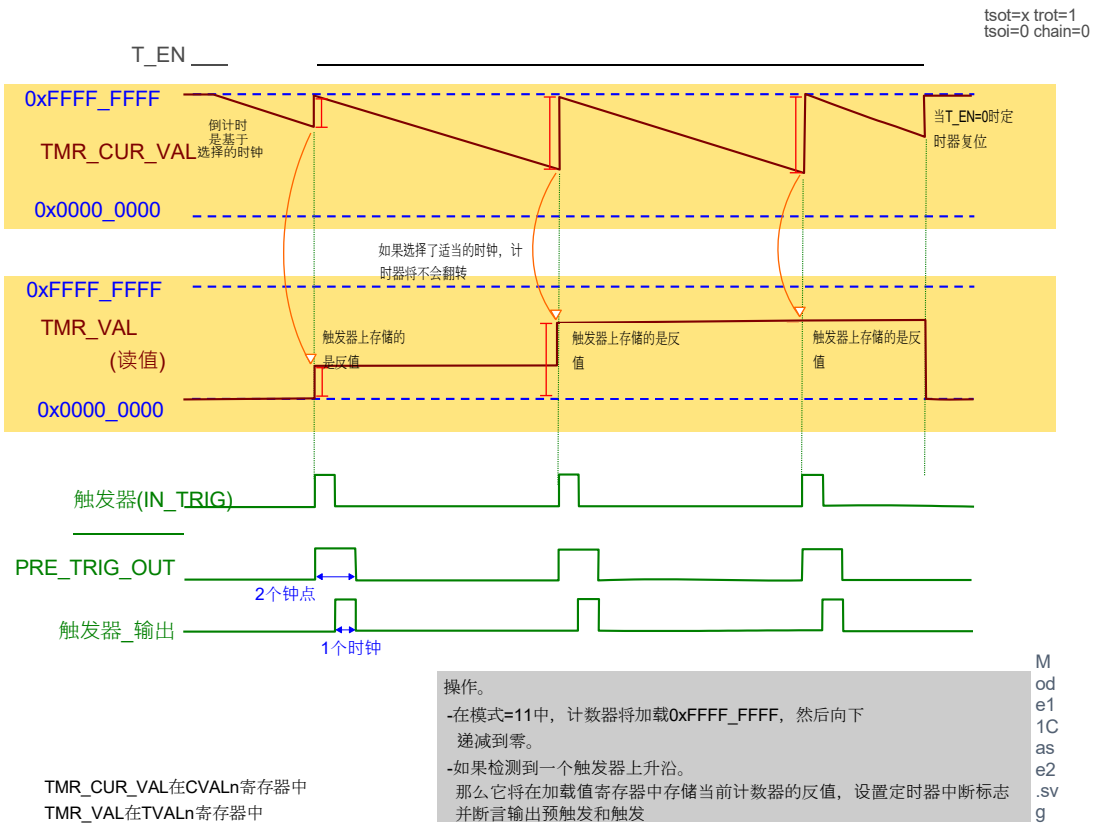



**图43-15.情况 2: TSOI = 0; TROT = 1; CHAIN = 0 (TSOT = X)**

模式=11

32位触发器捕获模式

情况3：TSOI=1，TROT=0，CHAIN=0（TSOT=X）。

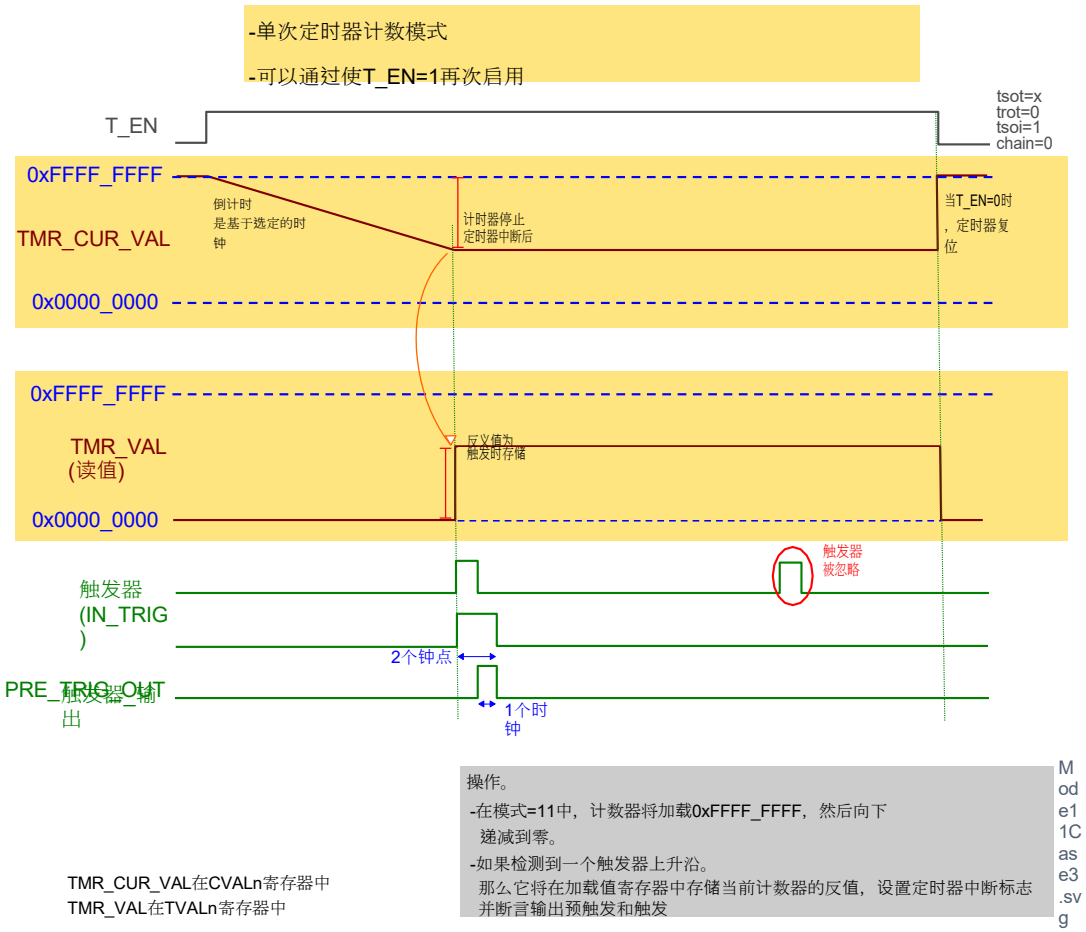



**图43-16.情况 3: TSOI = 1; TROT = 0; CHAIN = 0 (TSOT = X)**

**情况4：TSOI=1；TROT=1；CHAIN=0（TSOT=X）。**

和前面的案例（案例3）一样，只是定时器重新加载到0xFFFF_FFFF，并停止，直到T_EN再次变成1。案例4不是一个非常有用的案例，因为案例3涵盖了这一点。

## 43.5.5.5 定时器链：对定时操作的影响

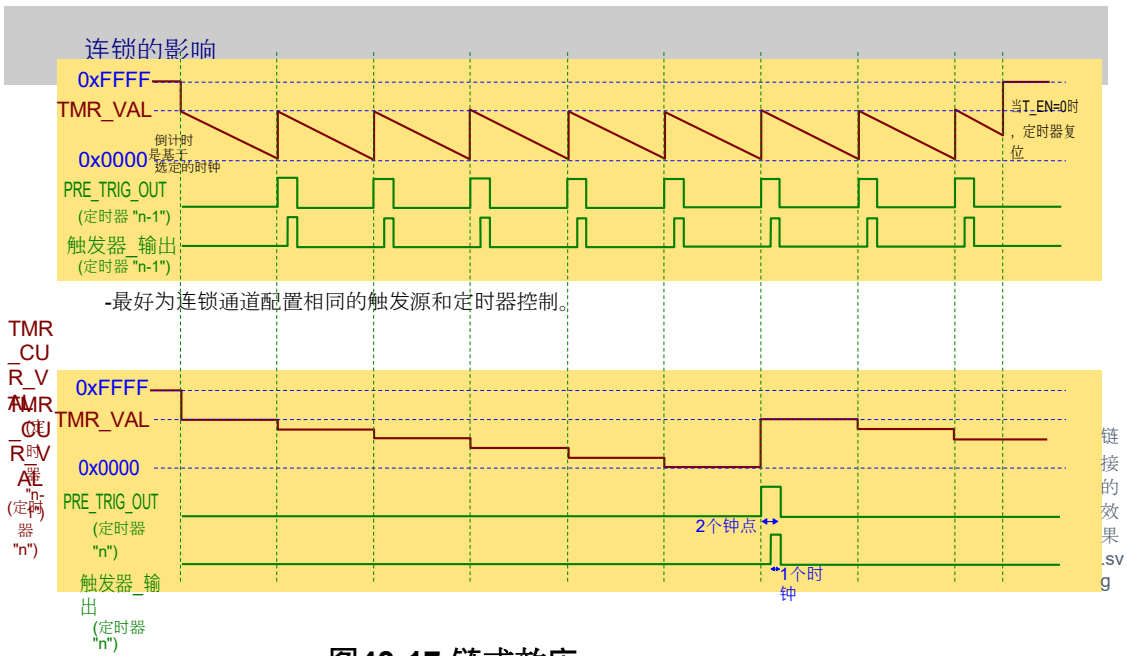



图43-17.链式效应

# 第四十四章
# 低功耗定时器（LPTMR）

## 44.1 芯片特有的LPTMR信息

### 44.1.1 实例化信息

S32K1xx包含一个LPTMR模块，有一个1通道16位时间计数器或1通道16位脉冲计数器。

#### 44.1.1.1 LPT/HSCMP0脉冲计数

LPTMR_ALT0输入是通过TRGMUX对HSCMP0输出（LPT_ALT0 = HSCMP0输出）产生的脉冲进行计数的可选择源。

**注意事项**

- 该器件不支持低漏电模式和等待模式。关于可用的电源模式的详细信息，请参见模块在可用的低功率模式下的操作。
- 关于LPTMR_PSR[PCS]位的选择请参考表。模块时钟部分的外围模块时钟化。

### 44.1.2 LPTMR脉冲计数器输入选项

LPTMR_CSR[TPS]位域配置了在脉冲计数器模式下使用的输入源。下表显示了这个位域的特定芯片输入分配。

**表44-1.脉冲计数器输入选项**

| LPTMR_CSR[TPS] | | 脉冲计数器输入号码 | 芯片输入 |
|---|---|---|---|
| 职能描述 | 00 | 0 | TRGMUX输出 |

*表继续在下一页...*

表44-1.脉冲计数器输入选项（续）

| LPTMR_CSR[TPS] | 脉冲计数器输入号码 | 芯片输入 |
|---|---|---|
| 01 | 1 | LPTMR_ALT1引脚 |
| 10 | 2 | LPTMR_ALT2引脚 |
| 11 | 3 | LPTMR_ALT3引脚 |

## 44.2 简介

低功耗定时器（LPTMR）可被配置为在所有电源模式下（包括低漏电模式）作为一个带有可选的预分频器的时间计数器，或作为一个带有可选的突变滤波器的脉冲计数器运行。它还可以在大多数系统复位事件中继续工作，使其可以作为一个时间计数器使用。

### 44.2.1 特点

LPTMR模块的特点包括。

- 16位时间计数器或带比较的脉冲计数器
  - 可选的中断可以从任何低功率模式下产生异步唤醒功能
  - 硬件触发输出
  - 计数器支持自由运行模式或比较时复位
- 可配置的预分频器/突变滤波器的时钟源
- 可配置的脉冲计数器的输入源
  - 上升的边缘或下降的边缘

### 44.2.2 操作模式

下表描述了LPTMR模块在各种模式下的操作。

表44-2.操作模式

| 模式 | 描述 |
|---|---|
| 运行 | LPTMR运行正常。 |
| 等待 | LPTMR继续正常工作，并且可以配置为通过产生一个中断请求来退出低功耗模式。 |

| | |
|---|---|
| 停止 | LPTMR继续正常工作，并且可以配置为通过产生一个中断请求来退出低功耗模式。 |

*表继续在下一页...*

表44-2.操作模式（续）

| 模式 | 描述 |
| --- | --- |
| 低漏电率 | LPTMR继续正常工作，并且可以配置为通过产生一个中断请求来退出低功耗模式。 |
| 调试 | LPTMR在脉冲计数器模式下运行正常，但在时间计数器模式下计数器不增量。 |

## 44.3 LPTMR信号描述

表44-3.LPTMR信号描述

| 信号 | 输入/输出 | 描述 |
| --- | --- | --- |
| LPTMR_ALTn | I | 脉冲计数器输入引脚 |

### 44.3.1 详细的信号描述

表44-4.LPTMR接口-详细的信号描述

| 信号 | 输入/输出 | 描述 | |
| --- | --- | --- | --- |
| LPTMR_ALTn | I | 脉冲计数器输入<br>LPTMR可以选择其中一个输入引脚用于脉冲计数器模式。 | |
| | | 国家含义 | 断言--如果配置为高电平输入的脉冲计数器模式，则断言会导致CNR增量。<br>断电--如果配置为脉冲计数器模式，输入为有效低电平，那么断电会导致CNR增量。 |
| | | 时间安排 | 断言或取消断言可在任何时间发生；输入可与总线时钟异步断言。 |

## 44.4 内存图和寄存器定义

注意事项

LPTMR寄存器仅在POR或LVD事件中被复位。更多细节请参见LPTMR电源和复位。

## 44.4.1 LPTMR寄存器描述

### 44.4.1.1 LPTMR 内存图

LPTMR0基础地址。4004_0000h

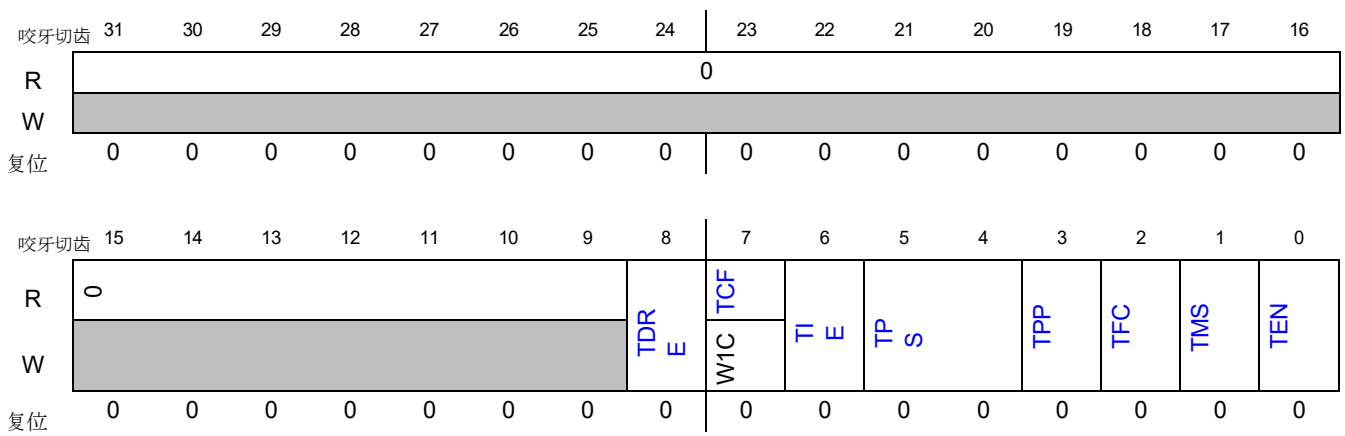
| 偏移 | 注册 | 宽度<br>(单位：比特) | 访问 | 重置值 |
|---|---|---|---|---|
| 0h | 低功耗定时器控制状态寄存器（CSR） | 32 | RW | 0000_0000h |
| 4h | 低功耗定时器预刻度寄存器（PSR） | 32 | RW | 0000_0000h |
| 8h | 低功耗定时器比较寄存器（CMR） | 32 | RW | 0000_0000h |
| 丘吉尔 | 低功耗定时器计数器寄存器（CNR）。 | 32 | RW | 0000_0000h |

### 44.4.1.2 低功耗定时器控制状态寄存器（CSR）

#### 44.4.1.2.1 偏移

| 注册 | 偏移 |
|---|---|
| 企业社会责任 | 0h |

#### 44.4.1.2.2 图示



### 44.4.1.2.3 领域

| 场地 | 职能 |
|---|---|
| 31-9<br>- | 保留 |
| 8<br>贸发会议 | 定时器DMA请求启用<br>当TDRE被设置时，只要TCF也被设置，就会产生LPTMR DMA请求，当DMA控制器完成后，TCF被清零。<br>0b - 禁用定时器DMA请求。1b - 启用定时器DMA请求。 |
| 7<br>TCF | 定时器比较标志<br>当 LPTMR 启用且 CNR 等于 CMR 并递增时，TCF 被设置。当LPTMR被禁用或向其写入一个逻辑1时，TCF被清零。<br>0b - CNR的值不等于CMR和增量。1b - CNR的值等于CMR，并且在增加。 |
| 6<br>TIE | 定时器中断启用<br>当TIE被设置时，只要TCF也被设置，就会产生LPTMR中断。<br>0b - 禁用定时器中断。1b - 启用定时器中断。 |
| 5-4<br>淘宝网 | 计时器引脚选择<br>配置在脉冲计数器模式下使用的输入源。只有当LPTMR被禁用时，才必须改变TPS。输入连接因设备而异。关于这些输入的连接，请参见芯片配置信息。<br>00b - 脉冲计数器输入0被选中。01b - 脉冲计数器输入1被选中。10b - 脉冲计数器输入2被选中。11b - 脉冲计数器输入3被选中。 |
| 3<br>TPP | 计时器引脚极性<br>配置脉冲计数器模式下输入源的极性。只有当LPTMR被禁用时，才必须改变TPP。<br>0b - 脉冲计数器输入源为高电平，CNR将在上升沿递增。1b - 脉冲计数器输入源为低电平，CNR将在下降沿递增。 |
| 2<br>TFC | 定时器自由运行的计数器<br>当清零时，TFC配置CNR在TCF被设置时进行复位。当设置时，TFC将CNR配置为在溢出时复位。只有当LPTMR被禁用时，才必须改变TFC。<br>0b - 只要TCF被设置，CNR就被重置。<br>1b - 溢出时CNR被复位。 |
| 1<br>TMS | 计时器模式选择<br>配置LPTMR的模式。只有当LPTMR被禁用时，TMS必须被改变。<br>0b - 时间计数器模式。1b - 脉冲计数器模式。 |

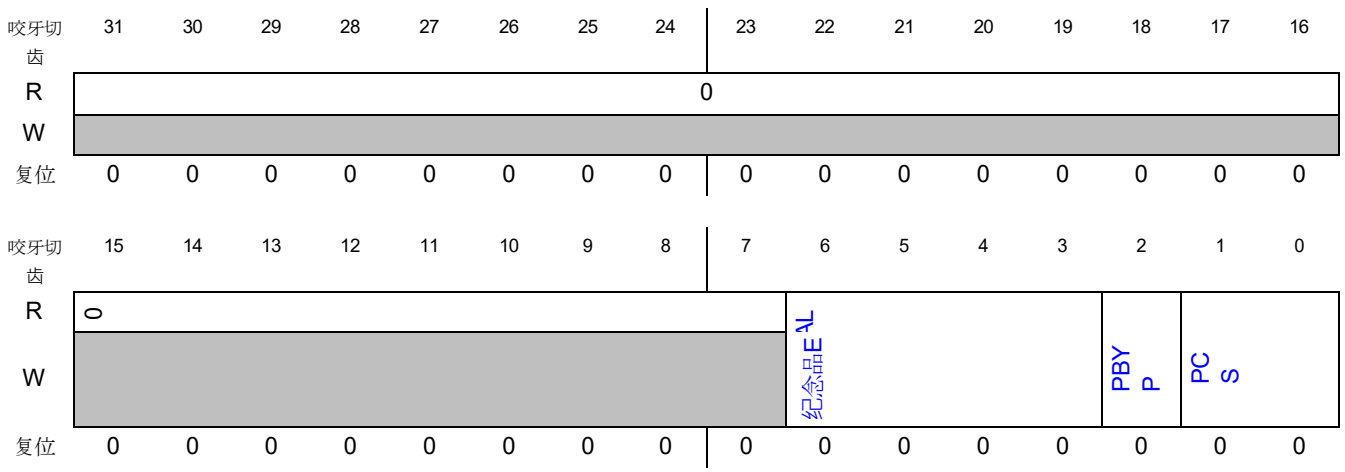
| | |
|---|---|
| 0<br>TEN | 定时器启用<br>当TEN清零时，它将重置LPTMR的内部逻辑，包括CNR和TCF。当TEN被设置时，LPTMR被启用。在向该字段写1时，不能改变CSR[5:1]。<br>0b - LPTMR被禁用，内部逻辑被复位。1b - LPTMR被启用。 |

# 44.4.1.3 低功耗定时器预刻度寄存器（PSR）

## 44.4.1.3.1 偏移

| 注册 | 偏移 |
|---|---|
| 证券交易所 | 4h |

## 44.4.1.3.2 图示




## 44.4.1.3.3 领域

| 场地 | 职能 |
|---|---|
| 31-7<br>- | 保留 |
| 6-3<br>预售 | 标度前值<br>配置时间计数器模式下预分频器的大小或脉冲计数器模式下突波滤波器的宽度。由于脉冲计数器输入的同步性，突波滤波器的宽度可以变化为1个周期。只有当LPTMR被禁用时，才必须改变PRESCALE。<br>0000b - 预分频器将预分频器的时钟除以2；突波滤波器不支持这种配置。0001b - 预分频器将预分频器时钟除以4；突波滤波器识别2个上升时钟沿后输入引脚的变化。<br>0010b - 预分频器将预分频器时钟除以8；突波滤波器识别4个上升时钟沿后输入引脚上的变化。<br>0011b - 预分频器将预分频器时钟除以16；突波滤波器识别8个上升时钟沿后输入引脚上的变化。<br>0100b - 预分频器将预分频器时钟除以32；突波滤波器识别16个上升时钟沿后输入引脚上的变化。<br>0101b - 预分频器将预分频器时钟除以64；突波滤波器识别32个上升时钟沿后输入引脚上的变化。<br>0110b - 预分频器将预分频器时钟除以128；突波滤波器识别输入引脚上的变化<br>在64个上升的时钟边沿之后。 |

*表继续在下一页...*

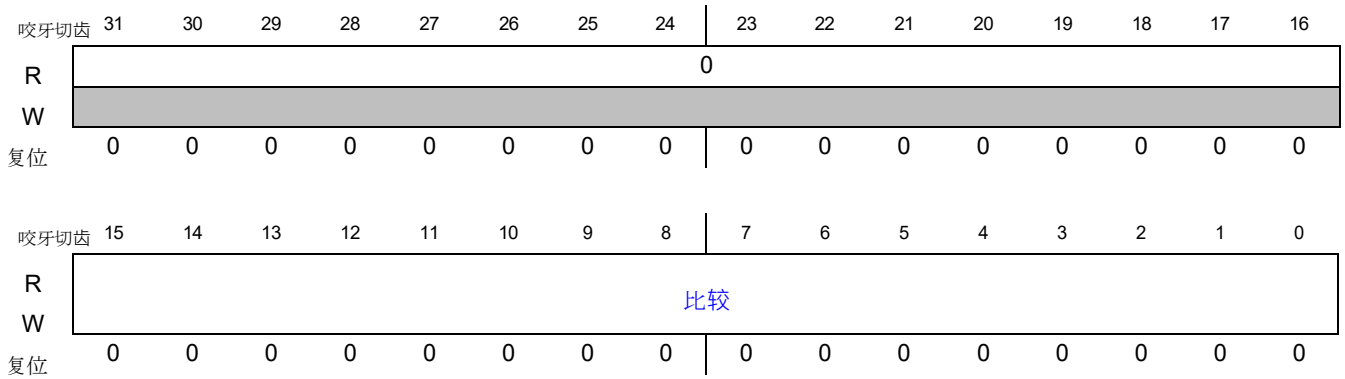
| 场地 | 职能 |
| --- | --- |
| | 0111b - 预分频器将预分频器时钟除以256；突波滤波器识别128个上升时钟沿后输入引脚上的变化。<br>1000b - 预分频器将预分频器时钟除以512；突波滤波器识别256个上升时钟沿后输入引脚上的变化。<br>1001b - 预分频器将预分频器时钟除以1024；突波滤波器识别512个上升时钟沿后输入引脚上的变化。<br>1010b - 预分频器将预分频器时钟除以2048；突波滤波器识别1024个上升时钟沿后输入引脚上的变化。<br>1011b - 预分频器将预分频器时钟除以4096；突波滤波器识别2048个上升时钟沿后输入引脚上的变化。<br>1100b - 预分频器将预分频器时钟除以8192；突波滤波器识别4096个上升时钟沿后输入引脚上的变化。<br>1101b - 预分频器将预分频器时钟除以16,384；突波滤波器识别8192个上升时钟沿后输入引脚上的变化。<br>1110b - 预分频器将预分频器时钟除以32,768；突波滤波器识别16,384个上升时钟沿后输入引脚上的变化。<br>1111b - 预分频器将预分频器时钟除以65,536；突波滤波器识别32,768个上升时钟沿后输入引脚上的变化。 |
| 2<br>PBYP | 预分频器旁路<br>当PBYP被设置时，在时间计数器模式下选定的预分频器时钟或在脉冲计数器模式下选定的输入源直接为CNR提供时钟。当PBYP清零时，CNR由预分频器/突变滤波器的输出来提供时钟。只有当LPTMR被禁用时，才必须改变PBYP。<br>0b - 预分频器/突波滤波器被启用。1b<br>- 预分频器/突变滤波器被绕过。 |
| 1-0<br>辽宁省 | 预分频器时钟选择<br>选择LPTMR预分频器/突变滤波器使用的时钟。在时间计数器模式下，该字段选择预分频器的输入时钟。在脉冲计数模式下，该字段选择输入到突波滤波器的时钟。只有当LPTMR被禁用时，才必须改变PCS。时钟的连接因设备而异。<br>**注意：**关于这些输入的连接信息，请参见芯片配置细节。<br>00b - 选择预分频器/突变滤波器时钟0。01b<br>- 选择预分频器/陷波滤波器时钟1。10b - 选<br>择预分频器/突波滤波器时钟2。11b - 选择<br>预分频器/陷波滤波器时钟3。 |

## 44.4.1.4 低功耗定时器比较寄存器（CMR）

### 44.4.1.4.1 偏移

| 注册 | 偏移 |
| --- | --- |
| CMR | 8h |

#### 44.4.1.4.2 图示




#### 44.4.1.4.3 领域

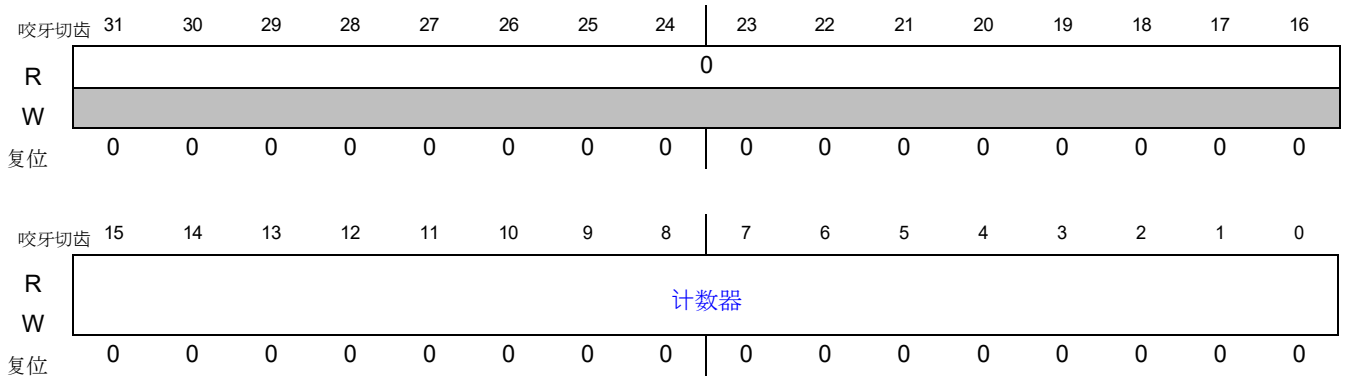
| 场地 | 职能 |
|---|---|
| 31-16<br>- | 保留 |
| 15-0<br>比较 | 比较价值<br>当LPTMR被启用且CNR等于CMR中的值并增量时，TCF被设置，硬件触发器被断电，直到CNR下次增量。如果CMR为0，则硬件触发器将一直被引爆，直到LPTMR被禁用。如果LPTMR被启用，只有当TCF被设置时，CMR必须被改变。 |

### 44.4.1.5 低功耗定时器计数器寄存器（CNR）。

#### 44.4.1.5.1 偏移

| 注册 | 偏移 |
|---|---|
| CNR | 丘吉尔 |

#### 44.4.1.5.2 图示




#### 44.4.1.5.3 领域

| 场地 | 职能 |
|---|---|
| 31-16<br>- | 保留 |
| 15-0<br>计数器 | 计数器值<br>CNR返回最后一次写入该寄存器时LPTMR计数器的当前值。 |

## 44.5 职能描述

### 44.5.1 LPTMR 电源和复位

LPTMR在所有电源模式下都保持供电，包括低漏电模式。如果不要求LPTMR在低功率模式下保持工作状态，那么在进入该模式之前必须禁用它。

LPTMR仅在全局上电复位（POR）或低电压检测（LVD）时被复位。当配置LPTMR寄存器时，在配置PSR和CMR之前，CSR最初必须在禁用定时器的情况下写入。然后，CSR[TIE]必须作为初始化的最后一步进行设置。这可以确保LPTMR配置正确，并且LPTMR计数器在热重置后被重置为零。

### 44.5.2 LPTMR时钟化

LPTMR的预分频器/突变滤波器可以由四个时钟中的一个时钟来控制。时钟源必须在LPTMR被启用之前被启用。

**注意事项**

由PSR[PCS]选择的时钟源可能需要配置为在低功耗模式下保持启用，否则LPTMR在低功耗模式下不会工作。

在绕过预分频器/突变滤波器的脉冲计数器模式下，选定的输入源直接为CNR提供时钟，不需要其他时钟源。为了在这种情况下最大限度地降低功耗，请将预分频器的时钟源配置为非切换的时钟。

**注意事项**

为LPTMR选择的时钟源或脉冲输入源不应该超过器件数据手册中定义的频率$f_{LPTMR}$。

## 44.5.3 LPTMR预分频器/突变滤波器

LPTMR的预分频器和突波滤波器共享相同的逻辑，在时间计数器模式下作为预分频器工作，在脉冲计数器模式下作为突波滤波器工作。

**注意事项**

当LPTMR被启用时，预分频器/突变滤波器的配置不能被改变。

### 44.5.3.1 预分频器启用

在时间计数器模式下，当预分频器被启用时，预分频器的输出直接为CNR提供时钟。当LPTMR被启用时，CNR将每$2^2$至$2^{16}$个预分频器时钟周期递增。在LPTMR启用后，由于同步逻辑，CNR的第一次增量将需要额外的一个或两个预分频器时钟周期。

### 44.5.3.2 预分频器被绕过

在时间计数器模式下，当预分频器被旁路时，所选的预分频器时钟在每个时钟周期都会增加CNR。当LPTMR被启用时，由于同步逻辑，第一次增量将需要额外的一个或两个预分频器时钟周期。

#### 44.5.3.3 突发事件过滤器

在脉冲计数器模式下，当突波滤波器被启用时，突波滤波器的输出直接为CNR提供时钟。当LPTMR首次启用时，突波滤波器的输出被断言，即逻辑1为高电平有效，逻辑0为低电平有效。下表显示了突波滤波器的输出随所选输入源的变化。

| 如果 | 那么 |
| --- | --- |
| 选定的输入源在至少$2^1$至$2^{15}$个连续的预分频器时钟上升沿中保持失电状态 | 突波滤波器的输出也将被取消。 |
| 选定的输入源在至少$2^1$至$2^{15}$个连续的预分频器时钟上升沿中保持断言。 | 突波滤波器的输出也将断言。 |

**注意事项**

输入只在上升的时钟边沿被采样。

CNR将在每次突波滤波器输出断电时增量。在脉冲计数器模式下，CNR的最大增量是每$2^2$至$2^{16}$个预分频器时钟沿的一次。当第一次启用时，由于同步逻辑的原因，突波滤波器将额外等待一个或两个预分频器的时钟沿。

#### 44.5.3.4 绕过了突发事件过滤器

在脉冲计数器模式下，当突波滤波器被绕过时，所选输入源每次断言都会增加CNR。在LPTMR首次启用之前，选定的输入源被强制断言。如果在LPTMR首次启用时，选定的输入源已经断言，这就防止了CNR的增量。

### 44.5.4 LPTMR比较

当CNR等于CMR的值并递增时，会发生以下事件。

- CSR[TCF]被设置。
- 如果CSR[TIE]也被设置，则产生LPTMR中断。
- 产生LPTMR硬件触发。
- 如果CSR[TFC]清零，CNR将被重置。

当LPTMR被启用时，只有当CSR[TCF]被设置时才能改变CMR。当更新CMR时，必须在LPTMR计数器增量超过新的LPTMR比较值之前写入CMR，并且CSR[TCF]必须被清除。

### 44.5.5 LPTMR计数器

CNR每增加一个，就会增加一个。

- 预分频器时钟在时间计数器模式下，预分频器被旁路了
- 在启用预分频器的时间计数器模式下的预分频器输出
- 脉冲计数器模式下的输入源断言，旁通了突波滤波器
- 脉冲计数器模式下的突波滤波器输出，启用突波滤波器

当LPTMR被禁用或计数器寄存器溢出时，CNR被复位。如果CSR[TFC]被清除，那么只要CSR[TCF]被设置，CNR也会被重置。

当核心在调试模式下停止时。
- 如果配置为脉冲计数器模式，CNR继续递增。
- 如果配置为时间计数器模式，CNR停止递增。

CNR不能被初始化，但可以在任何时候被读取。在每次读取CNR时，软件必须首先向CNR写入任何数值。这将使CNR的当前值同步并登记到一个临时寄存器中。每次读CNR时，临时寄存器的内容都会被返回。

读取CNR时，总线时钟必须比LPTMR计数器的增量速度至少快2倍，否则可能会返回错误的数据。

### 44.5.6 LPTMR硬件触发

LPTMR硬件触发器在CSR[TCF]被设置的同时发出信号，可用于触发其他外设的硬件事件而无需软件干预。硬件触发器总是被启用。

| 当 | 那么 |
|---|---|
| 在CSR[TFC]清零的情况下，CMR被设置为0 | LPTMR硬件触发器将在第一次比较时发出指令，并且不会取消指令。 |
| CMR被设置为一个非零值，或者，如果CSR[TFC]被设置为 | LPTMR硬件触发器将在每次比较时发出指令，并在CNR的下一个增量时取消指令。 |

### 44.5.7 LPTMR中断

只要CSR[TIE]和CSR[TCF]被设置，就会产生LPTMR中断。CSR[TCF]通过禁用LPTMR或向其写入逻辑1而被清除。

在LPTMR被启用时，CSR[TIE]可以被改变，CSR[TCF]可以被清除。

LPTMR中断是与系统时钟异步产生的，可以用来从任何低功耗模式下产生唤醒，包括低漏电模式，只要LPTMR被启用为唤醒源。

# 第45章
# 实时时钟（RTC）

## 45.1 芯片特定的RTC信息

### 45.1.1 RTC实例化

该器件上的RTC没有唤醒引脚，因此相关的寄存器位域不适用（如RTC_CR[WPS]、RTC_CR[WPE]和RTC_IER[WPON]）。另外，该器件没有集成电容，因此不能通过软件配置可调谐电容（包含在晶体振荡器中）。

**注意事项**

该器件上没有内部的32.768kHz晶体振荡器。本章中所有提到的32.768kHz时钟都是指RTC_CLK。关于可用的时钟源，见表25-9中的RTC时钟。

### 45.1.2 RTC中断

如RTC_IER寄存器中所述，RTC的中断在复位时被启用。一旦在NVIC中启用，中断就会发生。

### 45.1.3 软件推荐

在启用RTC之前，应向其提供软件复位。

## 45.2 简介

### 45.2.1 特点

RTC模块的功能包括。

- 32位秒计数器，具有翻转保护和32位报警功能
- 16位预分频器，具有补偿功能，可以纠正0.12ppm和3906ppm之间的误差
- 可选择使用1kHz的LPO来增加预分频器（预分频器在每个时钟边沿增加32）。
- 寄存器写入保护
  - 锁定寄存器需要POR或软件复位以实现写入访问
- 可配置的1、2、4、8、16、32、64或128赫兹方波输出，可选中断

### 45.2.2 操作模式

RTC在所有的低功耗模式下都保持功能，并且可以产生一个中断来退出任何低功耗模式。

### 45.2.3 RTC信号描述

**表45-1.RTC信号描述**

| 信号 | 描述 | 输入/输出 |
|---|---|---|
| RTC_CLKOUT | 预分频器方波输出或RTC 32.768 kHz时钟 | O |

#### 45.2.3.1 RTC时钟输出

RTC_CLKOUT信号可以输出方波预分频器输出（可配置为1、2、4、8、16、32、64或128 Hz）或RTC 32.768 kHz时钟。

## 45.3 注册定义

所有的寄存器都必须使用32位写操作，所有的寄存器访问都会产生三个等待状态。

当控制寄存器中的监督员访问位被清除时，非监督员模式的软件对任何寄存器的写访问将以总线错误而终止。

非监管模式软件的读取访问如常完成。

写入受锁定寄存器保护的寄存器不会产生总线错误，但写入不会完成。

### 45.3.1 RTC寄存器描述

#### 45.3.1.1 RTC内存图

RTC基础地址。4003_D000h

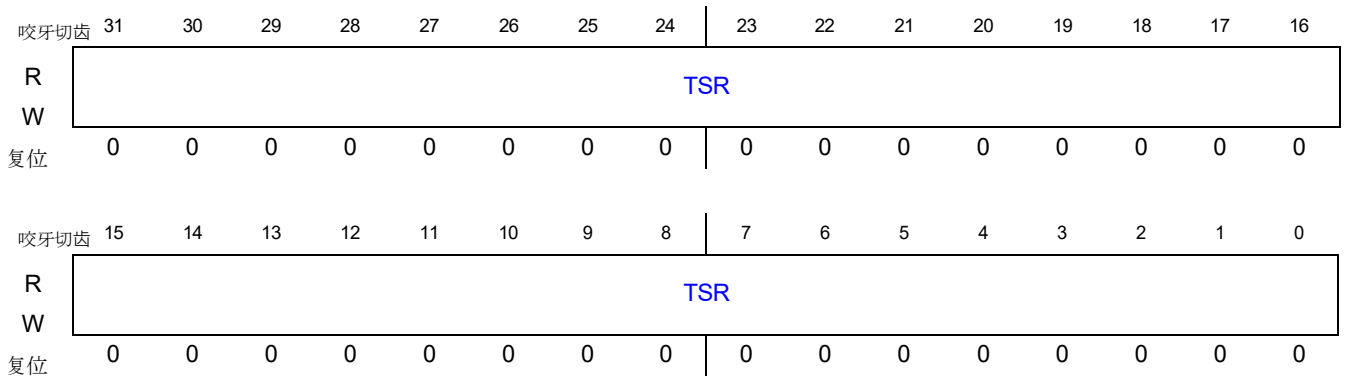
| 偏移 | 注册 | 宽度<br>(单位：比特) | 访问 | 重置值 |
|---|---|---|---|---|
| 0h | RTC时间秒数寄存器（TSR） | 32 | RW | 0000_0000h |
| 4h | RTC时间预分频器寄存器（TPR） | 32 | RW | 0000_0000h |
| 8h | RTC时间报警寄存器（TAR） | 32 | RW | 0000_0000h |
| 丘吉尔 | RTC时间补偿寄存器（TCR） | 32 | RW | 0000_0000h |
| 10h | RTC控制寄存器（CR） | 32 | RW | 0000_0000h |
| 14h | RTC状态寄存器（SR） | 32 | RW | 0000_0001h |
| 18h | RTC锁定寄存器（LR）。 | 32 | RW | 0000_00FFh |
| 1Ch | RTC中断启用寄存器(IER) | 32 | RW | 0000_0007h |

#### 45.3.1.2 RTC时间秒数寄存器（TSR）

#### 45.3.1.2.1 偏移

| 注册 | 偏移 |
|---|---|
| TSR | 0h |

#### 45.3.1.2.2 图示




#### 45.3.1.2.3 领域

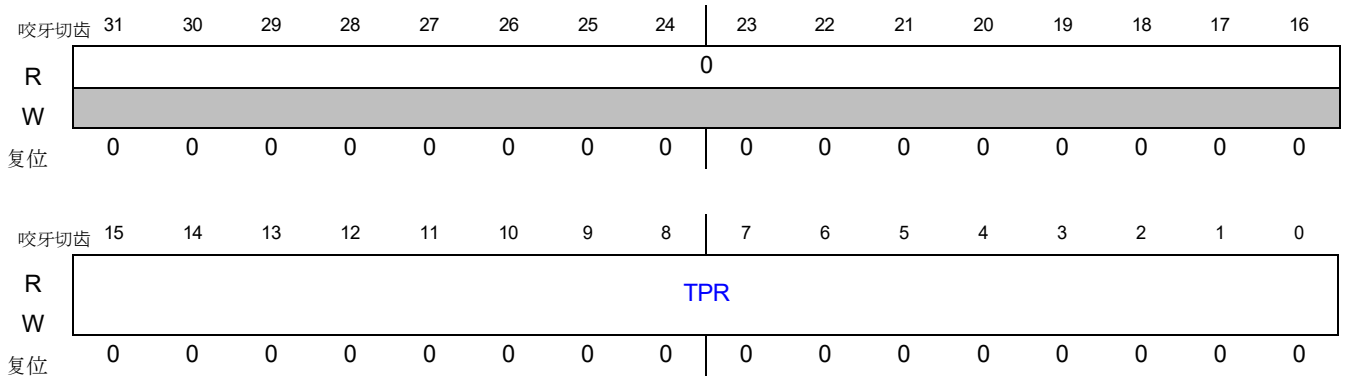
| 场地 | 职能 |
|---|---|
| 31-0<br>TSR | 时间秒数寄存器<br>当启用时间计数器时，只要不设置SR[TOF]或SR[TIF]，TSR就只读并每秒递增一次。当SR[TOF]或SR[TIF]被设置时，时间计数器将读为零。当时间计数器被禁用时，可以读或写TSR。当时间计数器被禁用时，对TSR的写入将清除SR[TOF]和/或SR[TIF]。支持以零写入TSR，但不建议这样做，因为当SR[TIF]或SR[TOF]被设置时，TSR将读为零（表示时间无效）。 |

### 45.3.1.3 RTC时间预分频器寄存器（TPR）

#### 45.3.1.3.1 偏移

| 注册 | 偏移 |
|---|---|
| TPR | 4h |

#### 45.3.1.3.2 图示




#### 45.3.1.3.3 领域

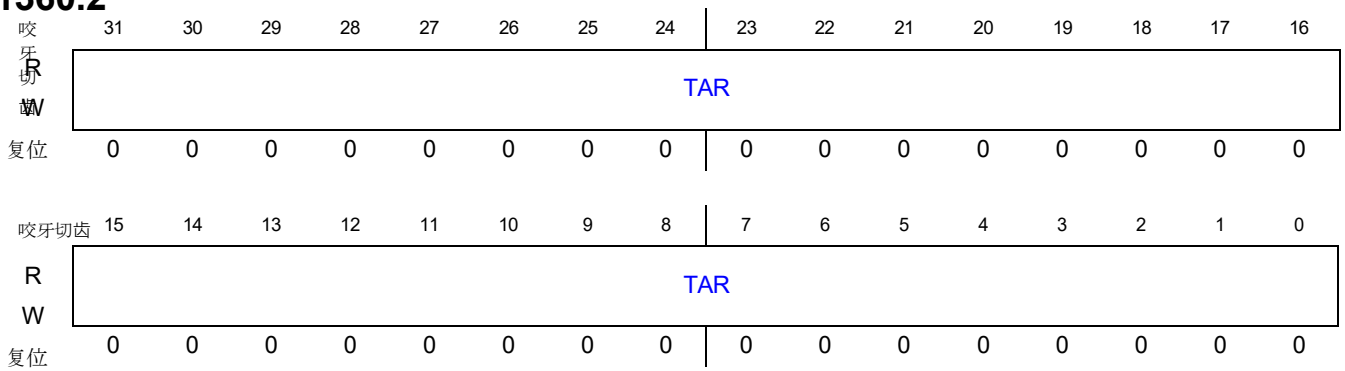
| 场地 | 职能 |
|---|---|
| 31-16<br>- | 保留 |
| 15-0<br>TPR | 时间预分频器寄存器<br>当时间计数器被启用时，TPR是只读的，每32.768kHz时钟周期递增一次。当SR[TOF]或SR[TIF]被设置时，时间计数器将读为零。当时间计数器被禁用时，TPR可以被读取或写入。当TPR的第14位从逻辑一过渡到逻辑零时，TSR[TSR]会增量。 |

### 45.3.1.4 RTC时间报警寄存器（TAR）

#### 45.3.1.4.1 偏移

| 注册 | 偏移 |
|---|---|
| TAR | 8h |

### 45.3.1.1360.2 图示




### 45.3.1.4.3 领域

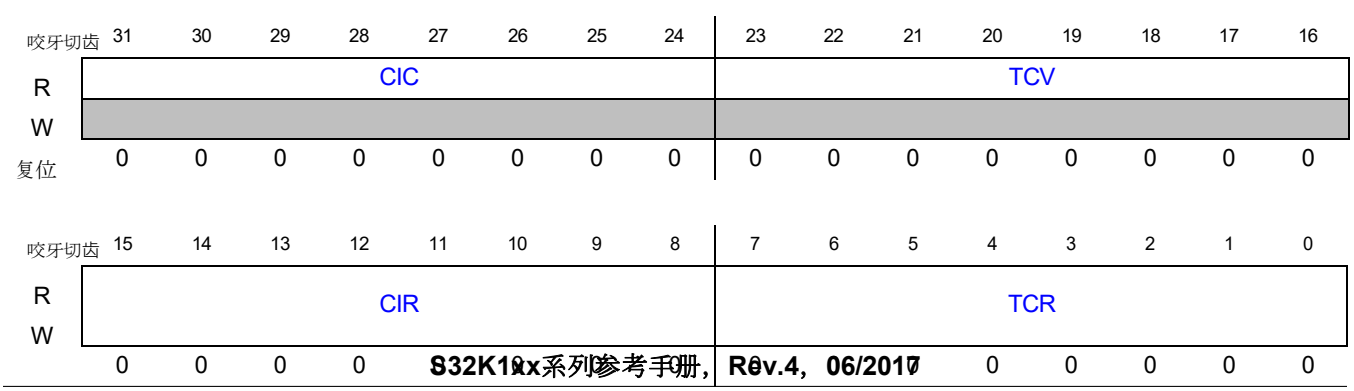
| 场地 | 职能 |
|---|---|
| 31-0<br>TAR | 时间报警寄存器<br>当时间计数器被启用时，只要TAR[TAR]等于TSR[TSR]并且TSR[TSR]递增，SR[TAF]就会被设置。对TAR的写入会清除SR[TAF]。 |

## 45.3.1.5 RTC时间补偿寄存器（TCR）

### 45.3.1.5.1 偏移

| 注册 | 偏移 |
|---|---|
| TCR | 丘吉尔 |

### 45.3.1.5.2 图示



复位

45.3.1.2 图示

45.3.1.5.3 领域

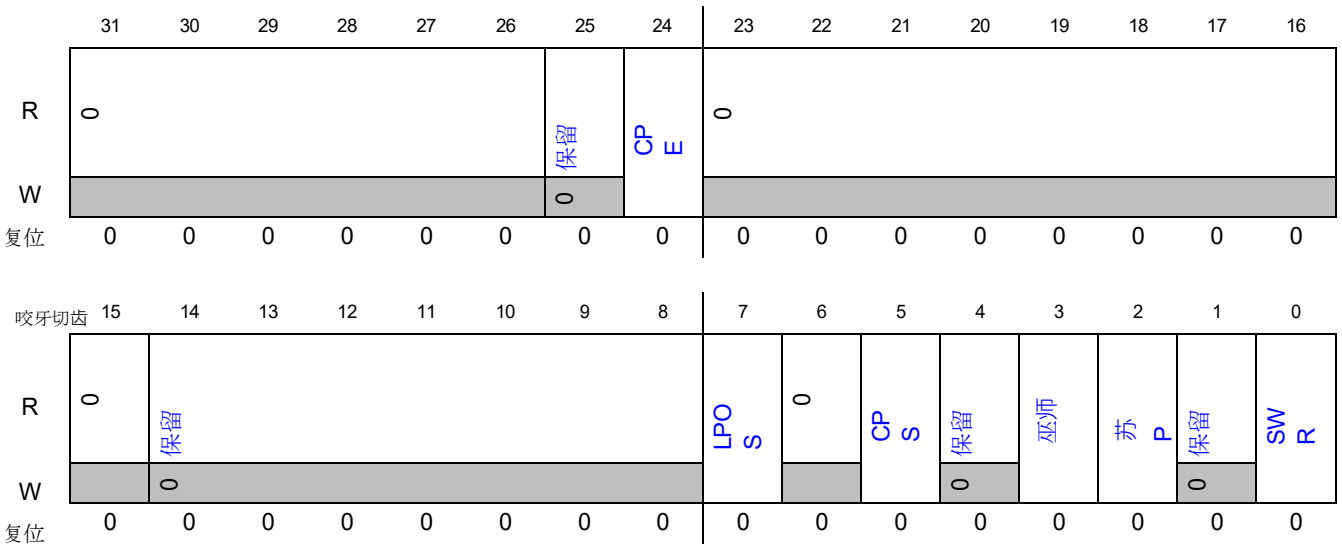
咬牙切齿

| 场地 | 职能 |
|---|---|
| 31-24<br>CIC | 补偿间隔计数器<br>补偿间隔计数器的当前值。如果补偿间隔计数器等于零，那么它将被载入CIR的内容。如果CIC不等于零，那么它将每秒递减一次。在秒计数器增量的同时读取这个寄存器可能会导致读出一个不正确的值。 |
| 23-16<br>TCV | 时间补偿价值<br>补偿逻辑在本秒区间内使用的电流值。如果CIC等于0，则每秒钟更新一次TCR字段的内容。如果CIC不等于0，那么它将被加载为0（该秒增量的补偿不启用）。在秒计数器增量的同时读取该寄存器，可能会导致读取的数值不正确。 |
| 15-8<br>CIR | 补偿间隔寄存器<br>配置以秒为单位的补偿间隔，从1到256，控制TCR在每秒钟内调整32.768kHz周期数的频率。写入的值应比秒数少一个。例如，写0来配置一秒的补偿间隔。这个寄存器是双缓冲的，在当前的补偿间隔结束之前，写入的内容不会产生影响。 |
| 7-0<br>TCR | 时间补偿寄存器<br>配置每秒32.768kHz的时钟周期数，等于32,768-TCR（其中TCR是一个双补号扩展值）。这个寄存器是双缓冲的，在当前的补偿间隔结束前，写入不会产生影响。下面是一些例子的数值。<br>00000000b - 时间预分频器寄存器每32768个时钟周期溢出。00000001b - 时间预分频器寄存器每32767个时钟周期溢出一次。01111110b - 时间预分频器寄存器每32642个时钟周期溢出。01111111b - 时间预分频器寄存器每32641个时钟周期溢出。10000000b - 时间预分频器寄存器每32896个时钟周期溢出。10000001b - 时间预分频器寄存器每32895个时钟周期溢出。11111111b - 时间预分频器寄存器每32769个时钟周期溢出。 |

## 45.3.1.6 RTC控制寄存器（CR）

### 45.3.1.6.1 偏移

| 注册 | 偏移 |
|---|---|
| CR | 10h |




## 45.3.1.6.3 领域

| 场地 | 职能 |
|---|---|
| 31-26<br>- | 保留 |
| 25<br>- | 保留 |
| 24<br>CPE | 启用时钟引脚<br>0b - 禁用RTC_CLKOUT引脚。<br>1b - 启用RTC_CLKOUT引脚。 |
| 23-15<br>- | 保留 |
| 14-8<br>- | 保留 |
| 7<br>LPOS | LPO选择<br>当设置时，RTC 预分频器使用 LPO 的 1 kHz 时钟而不是 RTC 的 32.768 kHz 时钟进行递增。LPO从位TPR[5]（TPR[4:0]被忽略）增加预分频器，支持接近1秒的秒寄存器增量。虽然从LPO时钟支持补偿，但补偿寄存器的TCR[4:0]也被忽略，只有TCR[7:5]设置补偿值（可以在1020到1027个周期后溢出）。<br>0b - RTC预分频器使用32.768 kHz时钟递增。<br>1b - RTC预分频器使用1 kHz LPO递增，预分频器的位[4:0]被忽略。 |
| 6<br>- | 保留 |
| 5<br>CPS | 时钟引脚选择<br>0b - 预分频器输出时钟（由TSIC配置）在RTC_CLKOUT上输出。 |

## 45.3.1.1364.2 图示

| | |
|---|---|
| 咬牙切齿 | 1b - RTC 32.768 kHz的时钟在RTC_CLKOUT上输出，前提是它要输出到其他外设。 |

*表继续在下一页...*

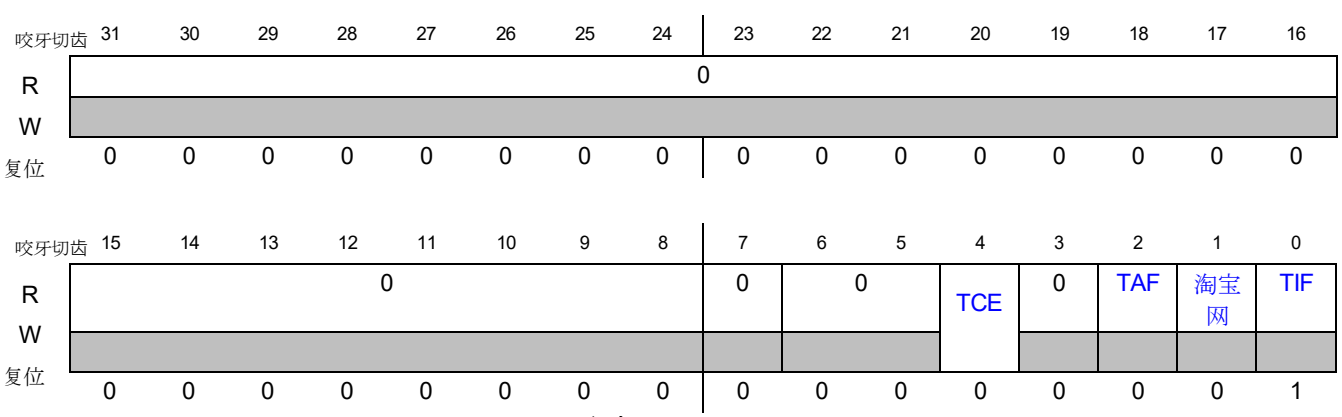
| 场地 | 职能 |
|---|---|
| 4<br>- | 保留 |
| 3<br>巫师 | 更新模式<br>允许写入SR[TCE]，即使在状态寄存器被锁定时。当设置时，如果SR[TIF]或SR[TOF]被设置，或者SR[TCE]被清除，就可以写入SR[TCE]。只要SR[TCE]和CR[UM]都被设置，那么SR[TCE]只能在SR[TIF]或SR[TOF]被设置后被写入。<br>0b - 锁定时不能写入寄存器。<br>1b - 寄存器在有限的条件下锁定后可以被写入。 |
| 2<br>舱内空间 | 监督员访问<br>0b - 不支持非监督者模式的写访问，并产生一个总线错误。1b - 支持非监督员模式的写访问。 |
| 1<br>- | 保留 |
| 0<br>西南联大 | 软件复位<br>0b - 没有影响。<br>1b - 复位所有RTC寄存器，除了SWR位。SWR位由POR和软件明确地清除它。 |

## 45.3.1.7 RTC状态寄存器（SR）

### 45.3.1.7.1 偏移

| 注册 | 偏移 |
|---|---|
| ÃÃÃ | 14h |

### 45.3.1.7.2 图示



#### 45.3.1.7.3 领域

| 场地 | 职能 |
|---|---|
| 31-8<br>- | 保留 |
| 7<br>- | 保留 |
| 6-5<br>- | 保留 |
| 4<br>TCE | 启用时间计数器<br>当时间计数器被禁用时，TSR寄存器和TPR寄存器是可写的，但不增量。当时间计数器被启用时，TSR寄存器和TPR寄存器不可写，但会递增。<br>0b - 时间计数器被禁用。1b - 时间计数器被启用。 |
| 3<br>- | 保留 |
| 2<br>TAF | 时间报警标志<br>当TAR[TAR]等于TSR[TSR]且TSR[TSR]递增时，时间报警标志被设置。该位通过写入TAR寄存器而被清除。<br>0b - 没有发生时间报警。1b - 发生了时间报警。 |
| 1<br>淘宝网 | 时间溢出标志<br>当时间计数器被启用并溢出时，时间溢出标志被设置。当该位被设置时，TSR和TPR不增量，读为零。当时间计数器被禁用时，该位通过写入TSR寄存器被清除。<br>0b - 没有发生时间溢出。<br>1b - 发生了时间溢出，时间计数器被读为零。 |
| 0<br>TIF | 时间无效标志<br>时间无效标志在POR或软件复位时被设置。当该位被设置时，TSR和TPR不增量，读为零。当时间计数器被禁用时，该位通过写入TSR寄存器被清除。<br>0b - 时间有效。<br>1b - 时间无效，时间计数器被读为零。 |

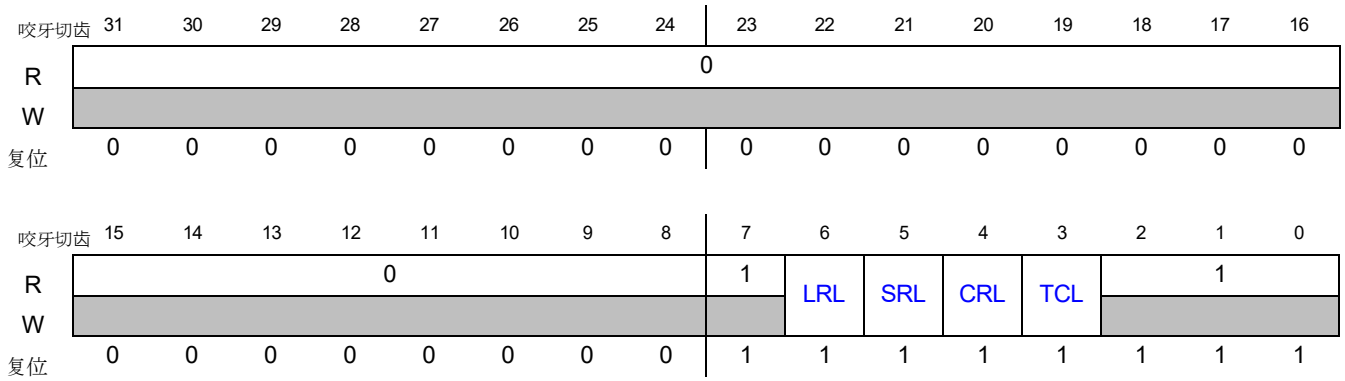
## 45.3.1.8 RTC锁定寄存器（LR）。

### 45.3.1.8.1 偏移

| 注册 | 偏移 |
|---|---|

| | |
|---|---|
| LR | 18h |

## 45.3.1.7.3 ~~45.3.1.8.2~~ 图域




## 45.3.1.8.3 领域

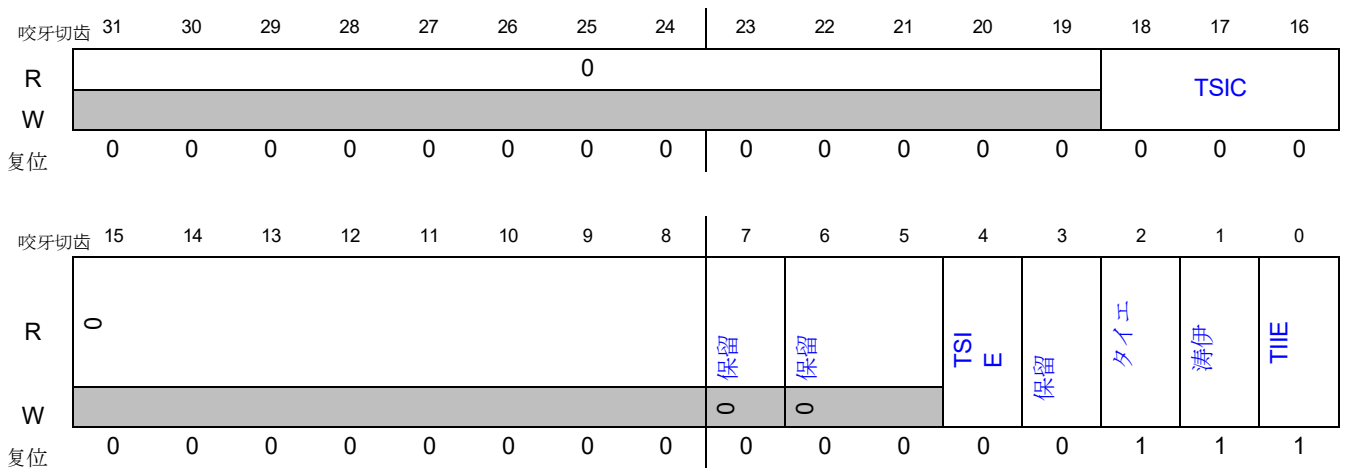
| 场地 | 职能 |
|---|---|
| 31-8<br>- | 保留 |
| 7<br>- | 保留 |
| 6<br>LRL | 锁定 寄存器锁定<br>该位被清零后，只能通过POR或软件复位来设置。<br>0b - 锁定寄存器被锁定，写操作被忽略。<br>1b - 锁定寄存器未被锁定，写操作如常完成。 |
| 5<br>SRL | 状态寄存器锁定<br>该位被清零后，只能通过POR或软件复位来设置。<br>0b - 状态寄存器被锁定，写操作被忽略。<br>1b - 状态寄存器未被锁定，写操作如常完成。 |
| 4<br>CRL | 控制寄存器锁定<br>该位被清零后，只能通过POR来设置。<br>0b - 控制寄存器被锁定，写操作被忽略。<br>1b - 控制寄存器未被锁定，写操作如常完成。 |
| 3<br>TCL | 时间补偿锁<br>该位被清零后，只能通过POR或软件复位来设置。<br>0b - 时间补偿寄存器被锁定，写操作被忽略。<br>1b - 时间补偿寄存器未被锁定，写操作如常完成。 |
| 2-0<br>- | 保留 |

### 45.3.1.9 RTC中断启用寄存器(IER)

## 45.3.1.9.1 偏移

| 注册 | 偏移 |
|---|---|
| IER | 1Ch |

## 45.3.1.9.2 图示




## 45.3.1.9.3 领域

| 场地 | 职能 |
|---|---|
| 31-19<br>- | 保留 |
| 18-16<br>TSIC | 定时器秒数中断配置<br>配置RTC Seconds中断和RTC_CLKOUT预分频器输出的频率。这个字段只应在TSIE清零时改变。<br>000b - 1 Hz。<br>001b - 2 Hz。<br>010b - 4 Hz。<br>011b - 8 Hz。<br>100b - 16 Hz。<br>101b - 32 Hz。<br>110b - 64 Hz。<br>111b - 128 Hz。 |
| 15-8<br>- | 保留 |
| 7<br>- | 保留 |
| 6-5<br>- | 保留 |

| | |
|---|---|
| 4 | 时间秒数 中断启用 |

*表继续在下一页...*

| 45.3.1.9.1 场地 | 偏移 职能 |
| --- | --- |
| TSIE | 秒中断是一个边缘敏感的中断，有一个专门的中断向量。它每秒钟至少产生一次，不需要软件开销（没有相应的状态标志需要清除）。秒中断的频率是由TSIC配置的。<br>0b - 秒针中断被禁用。1b - 秒钟中断被启用。 |
| 3<br>- | 保留 |
| 2<br>タイエ | 时间报警中断启用<br>0b - 时间报警标志不产生一个中断。1b - 时间报警标志会产生一个中断。 |
| 1<br>涛伊 | 时间溢出中断启用<br>0b - 时间溢出标志不产生中断。1b - 时间溢出标志会产生一个中断。 |
| 0<br>TIIE | 时间无效中断启用<br>0b - 时间无效标志不产生一个中断。1b - 时间无效标志会产生一个中断。 |

## 45.4 职能描述

### 45.4.1 电源、时钟和复位

RTC是一个始终通电的模块，在所有的低功耗模式下都保持活动。RTC内的时间计数器默认是由32.768kHz的时钟来计时。
或者，时间计数器可以由一个LPO 1 kHz时钟和预分频器进行时钟控制。
每一个LPO时钟将递增32。

上电复位信号将所有的RTC寄存器初始化为其默认状态。一个软件复位位也可以初始化所有RTC寄存器。

#### 45.4.1.1 振荡器控制

32.768kHz晶体振荡器在POR时被禁用，必须通过软件启用。启用晶体振荡器后，在设置SR[TCE]或使用RTC外部的振荡器时钟之前，请等待振荡器启动时间。

晶体振荡器包括可调谐的电容，可以通过软件进行配置。除非振荡器被禁用，否

则不要改变电容。

#### 45.4.1.2 软件复位

向CR[SWR]写1，迫使RTC模块的其他部分相当于一个POR。CR[SWR]不受软件复位的影响，必须由软件来清除。

#### 45.4.1.3 监督员访问

当监督者访问控制位被清除时，只有监督者模式的软件可以写到RTC寄存器，非监督者模式的软件将产生一个总线错误。监督者和非监督者模式的软件都可以随时读取RTC寄存器。

### 45.4.2 时间计数器

时间计数器包括一个32位的秒计数器，每秒递增一次，以及一个16位的预分频器寄存器，每32.768kHz时钟周期递增一次。也可以选择使用1kHz的LPO对预分频器进行时钟控制，在每个时钟周期将预分频器增加32。

在时间计数器（秒或预分频器）增量时读取它可能会由于读取数据总线的同步性而返回无效数据。如果软件有必要在预分频器或秒计数器增量时读取它们，建议执行两次读取访问，并由软件验证两次读取返回的数据是否相同。

只有当SR[TCE]清零时，才能写入时间秒寄存器和时间预分频器寄存器。在写秒寄存器之前一定要先写预分频器寄存器，因为秒寄存器在预分频器寄存器的第14位下降沿上增量。

只要SR[TCE]被设置，SR[TIF]被清除，SR[TOF]被清除，并且有32.768kHz（或1kHz）的时钟源，时间预分频器寄存器就会增加。启用振荡器后，在设置SR[TCE]之前要等待振荡器的启动时间，以便有时间让振荡器的时钟输出稳定下来。

如果时间秒寄存器溢出，那么SR[TOF]将被设置，时间预分频寄存器将停止递增。通过初始化时间秒寄存器来清除SR[TOF]。只要SR[TOF]被设置，时间秒寄存器和时间预分频器寄存器的读数为零。

SR[TIF]在POR和软件复位时被设置，并通过初始化时间秒寄存器而被清除。只要SR[TIF]被设置，时间秒寄存器和时间预分频器寄存器的读数为零。

### 45.4.3 赔偿

补偿逻辑提供了准确和宽泛的补偿范围，可以纠正高达3906ppm和低至0.12ppm的误差。补偿系数必须在RTC的外部计算，并由软件提供给补偿寄存器。RTC本身并不计算所需的补偿量，尽管1赫兹的时钟被输出到一个外部引脚，以支持外部校准逻辑。

晶体补偿可以通过使用固件和晶体特性来确定补偿量来支持。温度补偿可由固件支持，该固件通过ADC定期测量外部温度并根据指定晶体频率随温度变化的查找表更新补偿寄存器。

补偿逻辑改变了预分频器寄存器溢出并增加时间秒计数器所需的32.768kHz时钟周期的数量。时间补偿值用于调整-127和+128之间的时钟周期数。当预分频器寄存器等于0x3FFF时，周期从预分频器寄存器中添加或减去，然后增量。补偿间隔用于调整时间补偿值的使用频率，即从每秒一次到每256秒一次。

对时间补偿寄存器的更新将不会生效，直到下一次时间秒寄存器递增，并且前一个补偿间隔已经过期。当补偿间隔被设置为非每秒一次时，补偿将应用于第一个秒间隔，其余的秒间隔不接受补偿。

通过将时间补偿寄存器配置为零，可以禁用补偿。

当预分频器被配置为使用1 kHz LPO增量时，有效补偿值被32除以，只能调整-4和+3之间的时钟周期数。

### 45.4.4 时间报警

时间报警寄存器（TAR）、SR[TAF]和IER[TAIE]允许RTC在预定的时间产生一个中断。32位的TAR每次递增时与32位的时间秒数寄存器（TSR）进行比较。当TAR等于TSR并且TSR增量时，SR[TAF]将被设置。

SR[TAF]通过写TAR被清除。这通常是下一个报警值，尽管写一个小于TSR的值，比如0，会阻止SR[TAF]再次设置。SR[TAF]不能被禁用，尽管它产生的中断是由IER[TAIE]启用或禁用。

### 45.4.5 更新模式

控制寄存器（CR[UM]）中的更新模式字段配置了对时间计数器启用（SR[TCE]）字段的软件写入权限。当CR[UM]清零时，只有当LR[SRL]被设置时，SR[TCE]才能被写入。当CR[UM]被设置时，当SR[TCE]被清除或SR[TIF]或SR[TOF]被设置时，SR[TCE]也可以被写入。这使得每当时间失效时，时间秒和预分频器寄存器就会被初始化，同时防止时间秒和预分频器寄存器被临时改变。当LR[SRL]被设置时，CR[UM]对SR[TCE]没有影响。

### 45.4.6 注册锁

锁定寄存器（LR）可以用来阻止对某些寄存器的写访问，直到下一次POR或软件复位。锁定控制寄存器（CR）将使软件复位失效。锁定LR将阻止未来对LR的更新。

对锁定寄存器的写访问被忽略，不产生总线错误。

### 45.4.7 中断

只要一个状态标志和相应的中断使能位都被设置，RTC中断就会被断言。它总是在POR和软件复位时被引爆。RTC中断是通过启用芯片特定的RTC时钟门控制位而在芯片层面上启用的。RTC中断可以用来将芯片从任何低功耗模式下唤醒。

RTC秒中断是一个边沿敏感的中断，有一个专门的中断向量，每秒产生一次，不需要软件开销（没有相应的状态标志需要清除）。它在RTC中通过时间秒中断使能位使能，在芯片级通过设置芯片特定的RTC时钟门控制位使能。秒中断的频率默认为1Hz，但也可以配置为每2、4、8、16、32、64或128Hz触发一次。

# 第46章
# 低功耗串行外设接口（LPSPI）。

## 46.1 特定芯片的LPSPI信息

### 46.1.1 实例化信息

下表总结了该模块在产品系列中各芯片的实现情况。

**表46-1.LPSPI配置**

| 芯片 | 实例 | TX FIFO (字) | RX FIFO (字) | 芯片选择 |
|---|---|---|---|---|
| S32K116 | LPSPI0 | 4 | 4 | TBD |
| S32K118 | LPSPI0 | 4 | 4 | 4 |
| | 锂离子交换器1 | 4 | 4 | 4 |
| S32K142 | LPSPI0 | 4 | 4 | 4 |
| | 锂离子交换器1 | 4 | 4 | 4 |
| S32K144 | LPSPI0 | 4 | 4 | 4 |
| | 锂离子交换器1 | 4 | 4 | 4 |
| | 锂离子电池2 | 4 | 4 | 4 |
| S32K146 | LPSPI0 | 4 | 4 | 4 |
| | 锂离子交换器1 | 4 | 4 | 4 |
| | 锂离子电池2 | 4 | 4 | 4 |
| S32K148 | LPSPI0 | 4 | 4 | 4 |
| | 锂离子交换器1 | 4 | 4 | 4 |
| | 锂离子电池2 | 4 | 4 | 4 |

每个模块的芯片选择的确切数量取决于封装；不是所有的芯片选择在不同的封装上都可用。LPSPI2不支持任何与TRGMUX有关的功能，例如来自TRGMUX的HREQ源或任何对TRGMUX的触发。

所有记录的LPSPI寄存器都存在于设备上的所有LPSPI实例中。

该器件不支持低漏电和等待模式。关于可用的电源模式的详情，请参见模块在可用的低功率模式下的操作。

## 46.2 简介

### 46.2.1 概述

LPSPI是一个低功耗的串行外设接口（SPI）模块，支持作为主站和/或从站与SPI总线的有效接口。只要有适当的时钟，LPSPI就可以在停止模式下继续工作，并且设计成低CPU开销，对FIFO寄存器访问进行DMA卸载。

### 46.2.2 特点

LPSPI支持以下功能。

- 字大小=32位
- 命令/发送FIFO为4个字。
- 接收FIFO为4个字。
- 主机请求输入可用于控制SPI总线传输的开始时间。

## 46.2.3 框图

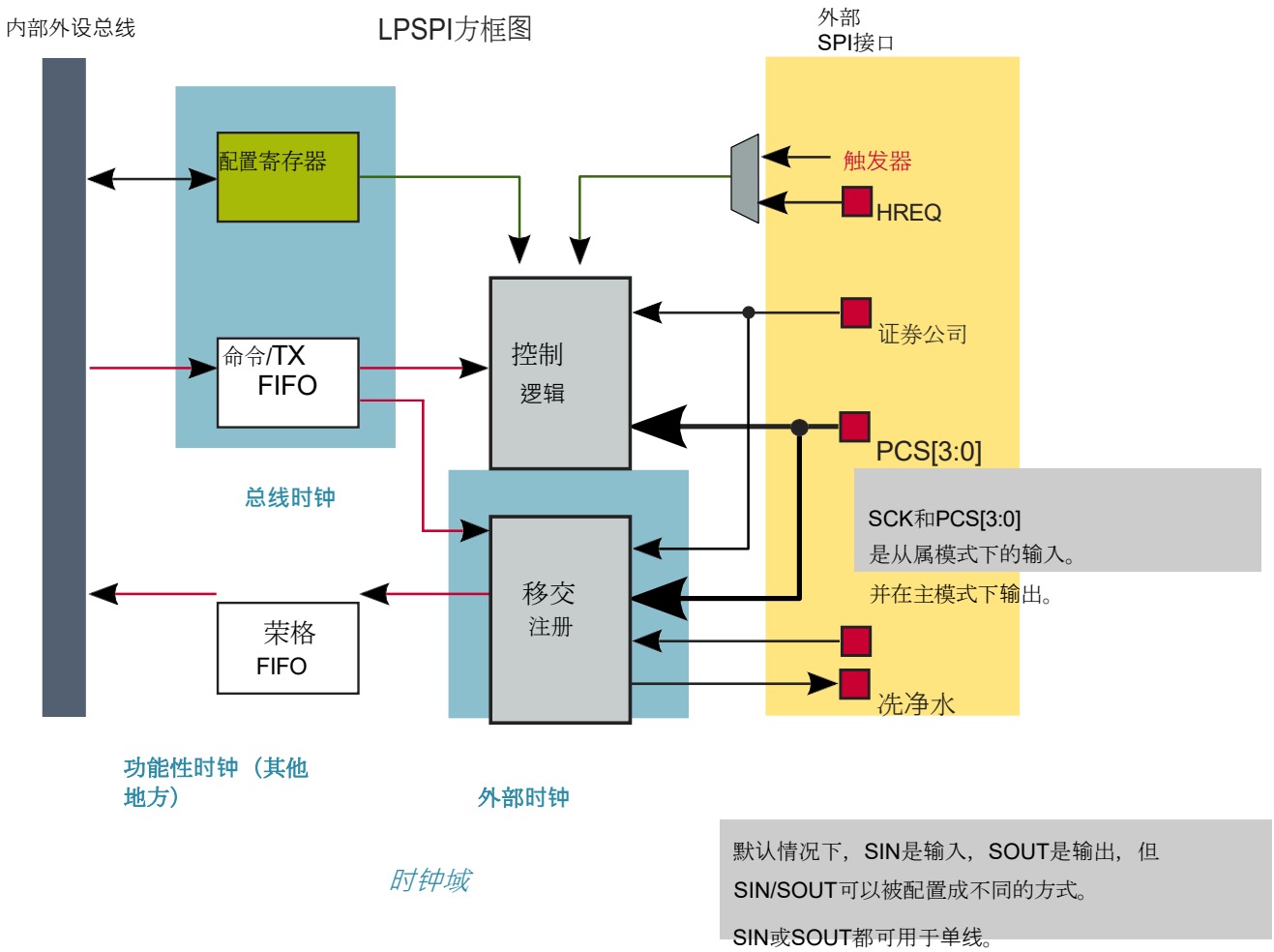



**图46-1.框图**

## 46.2.4 操作模式

LPSPI模块支持下表中所述的芯片模式。

**表46-2.LPSPI模块支持的芯片模式**

| 芯片模式 | LPSPI操作 |
|---|---|
| 运行 | 正常运行 |
| 停止 | 如果Doze Enable位（CR[DOZEN]）被清除，并且LPSPI使用在停止模式期间保持工作的外部或内部时钟源，则可以继续工作。 |
| 调试（核心处于调试/修正模式）。 | 如果调试启用位（CR[DBGEN]）被设置，可以继续运行。 |

### 46.2.5 信号描述

| 信号 | 描述 | 输入/输出 |
|---|---|---|
| 证券公司 | 串行时钟。在从属模式下输入，在主模式下输出。 | 输入/输出 |
| PCS[0] | 周边芯片选择。在从属模式下输入，在主模式下输出。 | 输入/输出 |
| PCS[1] / HREQ | 周边芯片选择或主机请求。当HREN=1和HRSEL=0时，主机请求引脚被选中。 在从属模式下输入，或者当用作主机请求时，在主模式下输出。 | 输入/输出 |
| 个人电脑[2] / 数据[2] | 外围芯片选择或四数据传输时的数据引脚2。在从属模式下输入，在主控模式下输出，在四元数据接收传输中输入，在四元数据发送传输中输出。 | 输入/输出 |
| 个人电脑[3] / 数据[3] | 外围芯片选择或四数据传输时的数据引脚3。在从属模式下输入，在主控模式下输出，在四元数据接收传输中输入，在四元数据发送传输中输出。 | 输入/输出 |
| SOUT / DATA[0] | 串行数据输出。可被配置为串行数据输入信号。在四数据和双数据传输中作为数据引脚0使用。 | 输入/输出 |
| SIN/DATA[1] | 串行数据输入。可被配置为串行数据输出信号。在四数据和双数据传输中作为数据引脚1使用。 | 输入/输出 |

## 46.3 存储器地图和寄存器

### 46.3.1 LPSPI寄存器描述

#### 46.3.1.1 LPSPI内存地图

LPSPI0基地址。4002_C000h

LPSPI1基址。4002_D000h

LPSPI2基址。4002_E000h

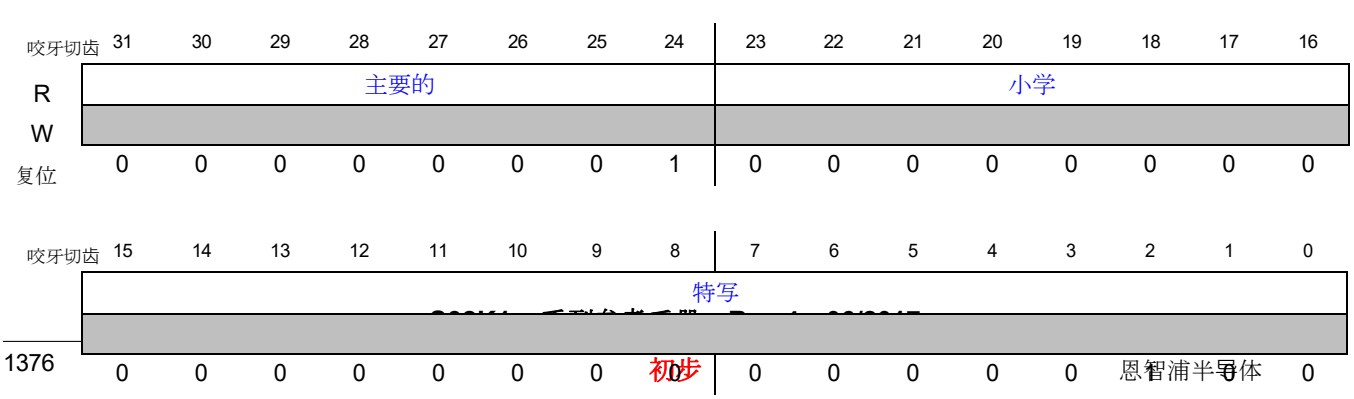
| 偏移 | 注册 | 宽度<br>(单位：比特) | 访问 | 重置值 |
|---|---|---|---|---|
| 0h | 版本标识寄存器（VERID） | 32 | RO | 0100_0004h |
| 4h | 参数寄存器（PARAM）。 | 32 | RO | 0000_0202h |
| 10h | 控制寄存器（CR） | 32 | RW | 0000_0000h |
| 14h | 状态寄存器（SR） | 32 | W1C | 0000_0001h |
| 18h | 中断使能寄存器（IER） | 32 | RW | 0000_0000h |
| 1Ch | DMA启用寄存器（DER） | 32 | RW | 0000_0000h |
| 20h | 配置寄存器0（CFGR0） | 32 | RW | 0000_0000h |
| 24h | 配置寄存器1（CFGR1）。 | 32 | RW | 0000_0000h |
| 30h | 数据匹配寄存器0（DMR0） | 32 | RW | 0000_0000h |
| 34h | 数据匹配寄存器1（DMR1）。 | 32 | RW | 0000_0000h |
| 40h | 时钟配置寄存器（CCR） | 32 | RW | 0000_0000h |
| 58h | FIFO控制寄存器（FCR） | 32 | RW | 0000_0000h |
| 5Ch | FIFO状态寄存器（FSR） | 32 | RO | 0000_0000h |
| 60h | 发送命令寄存器（TCR） | 32 | RW | 0000_001Fh |
| 64h | 发送数据寄存器（TDR） | 32 | 汪洋 | 0000_0000h |
| 70h | 接收状态寄存器（RSR） | 32 | RO | 0000_0002h |
| 74h | 接收数据寄存器（RDR） | 32 | RO | 0000_0000h |

## 46.3.1.2 版本标识寄存器（VERID）

### 46.3.1.2.1 偏移

| 注册 | 偏移 |
|---|---|
| 虚实对照 | 0h |

### 46.3.1.2.2 图示



R
W
复位

#### 46.3.1.2.3 领域

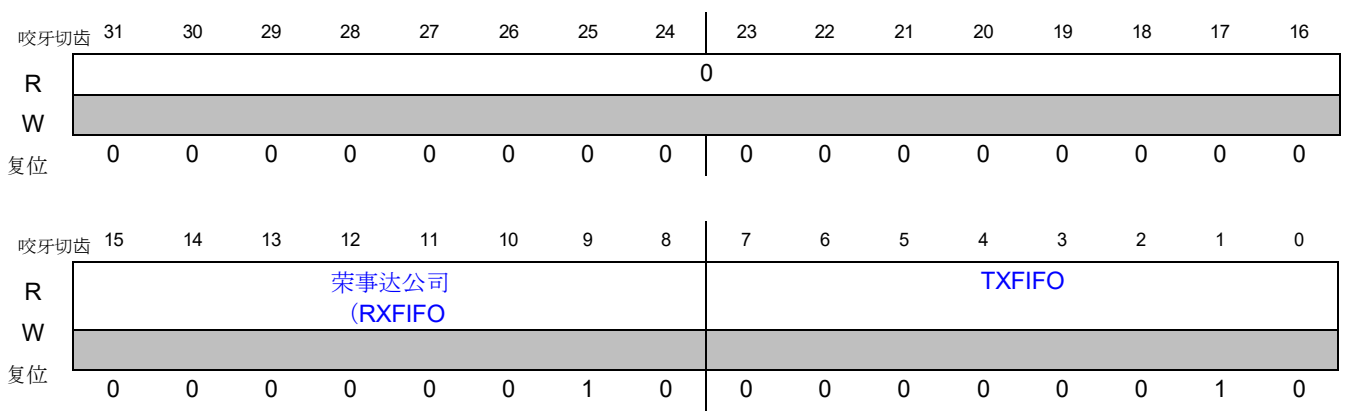
| 场地 | 职能 |
|---|---|
| 31-24<br>主要的 | 主要版本号<br>这个只读字段返回模块规范的主要版本号。 |
| 23-16<br>小学 | 次要版本号<br>这个只读字段返回模块规范的次要版本号。 |
| 15-0<br>特写 | 模块识别号<br>这个只读字段返回特征集的编号。<br>0000000000000100b - 支持32位移位寄存器的标准特征集。 |

### 46.3.1.3 参数寄存器（PARAM）。

#### 46.3.1.3.1 偏移

| 注册 | 偏移 |
|---|---|
| 朴实无华 | 4h |

#### 46.3.1.3.2 图示




#### 46.3.1.3.3 领域

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| 31-16<br>- | 保留 |
| 15-8 | 接收FIFO大小 |

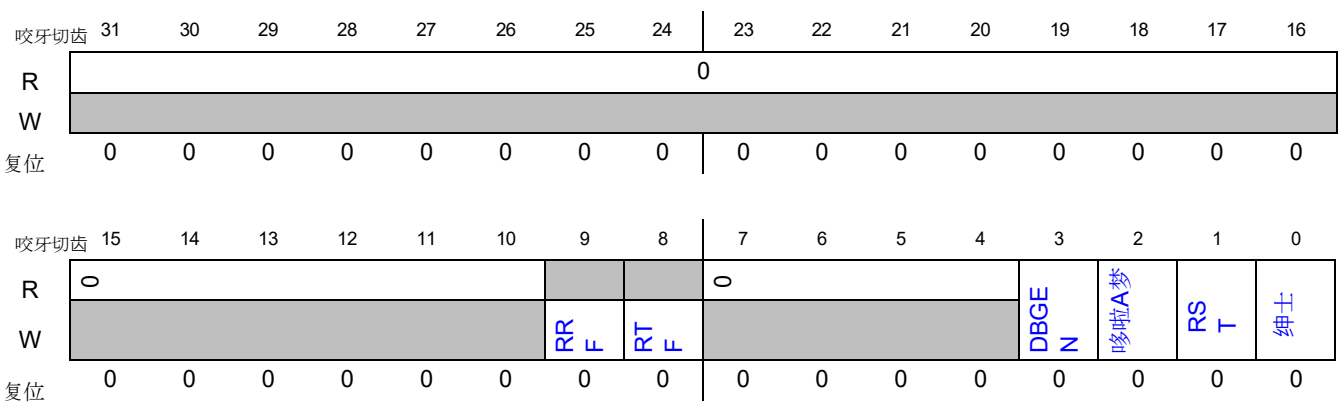
| 场地 | 职能 |
|---|---|
| 荣事达公司（ RXFIFO | 设置接收FIFO中的最大字数，即2RXFIFO。 |
| 7-0 TXFIFO | 发送FIFO大小<br>设置发送FIFO的最大字数，即2TXFIFO。 |

## 46.3.1.4 控制寄存器（CR）

### 46.3.1.4.1 偏移

| 注册 | 偏移 |
|---|---|
| CR | 10h |

### 46.3.1.4.2 图示




### 46.3.1.4.3 领域

| 场地 | 职能 |
|---|---|
| 31-10 - | 保留 |
| 9 RRF | Reset Receive FIFO<br>0b - 无影响。<br>1b - 接收的FIFO被重置。 |
| 8 RTF | 复位发送FIFO 0b - 无<br>影响。<br>1b - 发送FIFO被重置。 |

*表继续在下一页...*

| | |
|---|---|
| 7-4<br>- | 保留 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| 3<br>DBGEN | 调试启用<br>0b - 模块在调试模式下被禁用。1b - 模块在调试模式下被启用。 |
| 2<br>哆啦A梦 | 启用休眠模式<br>启用或禁用休眠模式<br>0b - 模块在休眠模式下被启用。1b - 模块在休眠模式下被禁用。 |
| 1<br>RST | 软件复位<br>重置所有内部逻辑和寄存器，除了控制寄存器。在被软件清除之前一直保持设置。<br>0b - 主控逻辑不被重置。1b - 主控逻辑被复位。 |
| 0<br>绅士 | 模块启用<br>0b - 模块被禁用。<br>1b - 模块已启用。 |

## 46.3.1.5 状态寄存器（SR）

### 46.3.1.5.1 偏移

| 注册 | 偏移 |
|---|---|
| ÃÃÃ | 14h |

### 46.3.1.5.2 图示

| 咬牙切齿 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | MBF | 0 | | | | | | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 咬牙切齿 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | DMF | REF | TEF | TCF | 融资基金 | WCF | 0 | | | | | | RDF | TDF |
| W | | | W1C | W1C | W1C | W1C | W1C | W1C | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

*表继续在下一页...*

### 46.3.1.5.3 领域

| 场地 | 职能 |
|---|---|
| 31-25 | 保留 |

| 场地 | 职能 |
|---|---|
| - | |
| 24<br>MBF | 模块繁忙标志<br>0b - LPSPI处于空闲状态。<br>1b - LPSPI正在忙。 |
| 23-14<br>- | 保留 |
| 13<br>DMF | 数据匹配标志<br>表示收到的数据与MATCFG配置的MATCH0和/或MATCH1字段匹配。<br>0b - 没有收到匹配的数据。 1b - 已收到匹配数据。 |
| 12<br>REF | 接收错误标志<br>当接收机FIFO溢出时，该标志将被设置。当该标志被设置时，建议首先结束传输，清空接收FIFO，清除该标志，然后从头重新开始传输。<br>0b - 接收FIFO没有溢出。 1b - 接收FIFO已经溢出。 |
| 11<br>TEF | 发送错误标志<br>当传输FIFO不足时，该标志将被设置。当该标志被设置时，建议首先结束传输，清除该标志，然后从头重新开始传输。<br>0b - 没有发生发送FIFO不足的情况。 1b - 发生了发送FIFO不足的情况 |
| 10<br>TCF | 转移完成标志<br>在主控模式下，当LPSPI回到空闲状态，发送FIFO为空时，该标志将被设置。<br>0b - 所有转让都没有完成。 1b - 所有转让都已完成。 |
| 9<br>融资基金 | 框架完整的旗帜<br>该标志将在每一帧传输结束时设置，当PCS被否定时。<br>0b - 帧传输未完成。 1b - 帧传输已完成。 |
| 8<br>WCF | 字词完整旗帜<br>当收到的字的最后一位被采样时，这个标志将被设置。<br>0b - 传输字未完成。 1b - 传输字完成。 |
| 7-2<br>- | 保留 |

*表继续在下一页...*

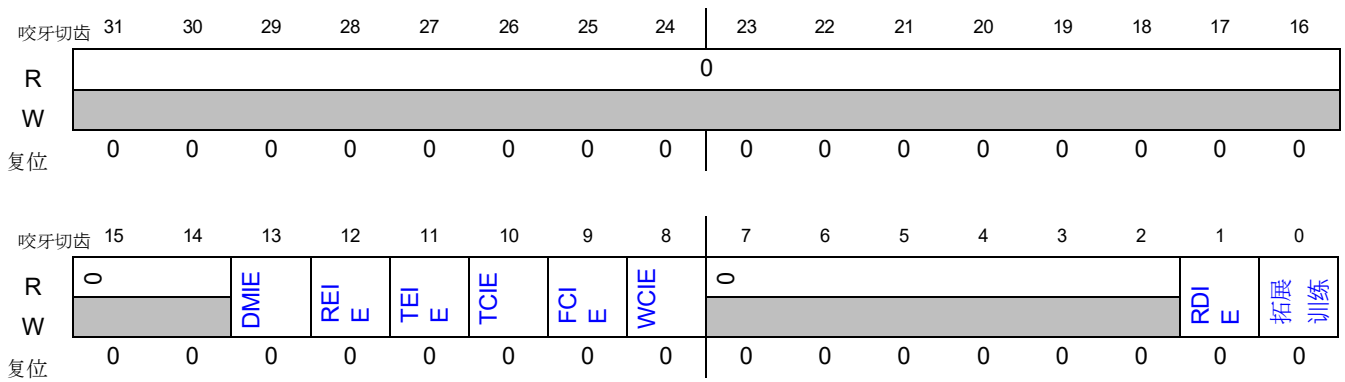
| | |
|---|---|
| 1<br>RDF | 接收数据标志<br>只要接收FIFO中的字数大于RXWATER，就会设置接收数据标志。<br>0b - 接收数据未准备好。1b - 接收数据已准备好。 |
| 0<br>TDF | 发送数据标志<br>每当发送FIFO中的字数等于或少于TXWATER时，发送数据标志就被设置。<br>0b - 不要求传输数据。1b - 发送数据被请求。 |

# 46.3.1.6 中断使能寄存器（IER）

## 46.3.1.6.1 偏移

| 注册 | 偏移 |
|---|---|
| IER | 18h |

## 46.3.1.6.2 图示




## 46.3.1.6.3 领域

| 场地 | 职能 |
|---|---|
| 31-14<br>- | 保留 |
| 13<br>DMIE | 数据匹配中断启用 0b - 中断<br>禁用。1b - 中断启用。 |
| 12<br>リエイ | 接收错误中断启用 0b - 中断禁<br>用。1b - 中断启用。 |
| 11<br>TEIE | 发送错误中断启用 0b - 中断禁<br>用。1b - 中断启用。 |
| 10<br>TCIE | Transfer Complete Interrupt Enable<br>0b - 中断禁用。<br>1b - 中断启用。 |

*表继续在下一页...*

| | |
|---|---|
| 9<br>FCIE | 帧完成中断启用 0b - 中断禁用。<br>1b - 中断启用。 |
| 8 | 字完成中断启用 |

| 场地 | 职能 |
|---|---|
| WCIE | 0b - 中断失效。<br>1b - 中断启用。 |
| 7-2<br>- | 保留 |
| 1<br>种族歧视委员会(RDIE) | 接收数据中断启用 0b - 中断禁用。1b - 中断启用。 |
| 0<br>拓展训练(TDIE) | 发送数据中断启用 0b - 中断禁用。1b - 中断启用 |

## 46.3.1.7 DMA启用寄存器（DER）

### 46.3.1.7.1 偏移

| 注册 | 偏移 |
|---|---|
| DER | 1Ch |

### 46.3.1.7.2 图示

| 咬牙切齿 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 咬牙切齿 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | RÅÅE | TDE |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

### 46.3.1.7.3 领域

| 场地 | 职能 |
|---|---|
| 31-2<br>- | 保留 |

*表继续在下一页...*

| | |
|---|---|
| 1 | 接收数据 DMA启用<br>0b - 禁用DMA请求。 |

*表继续在下一页...*

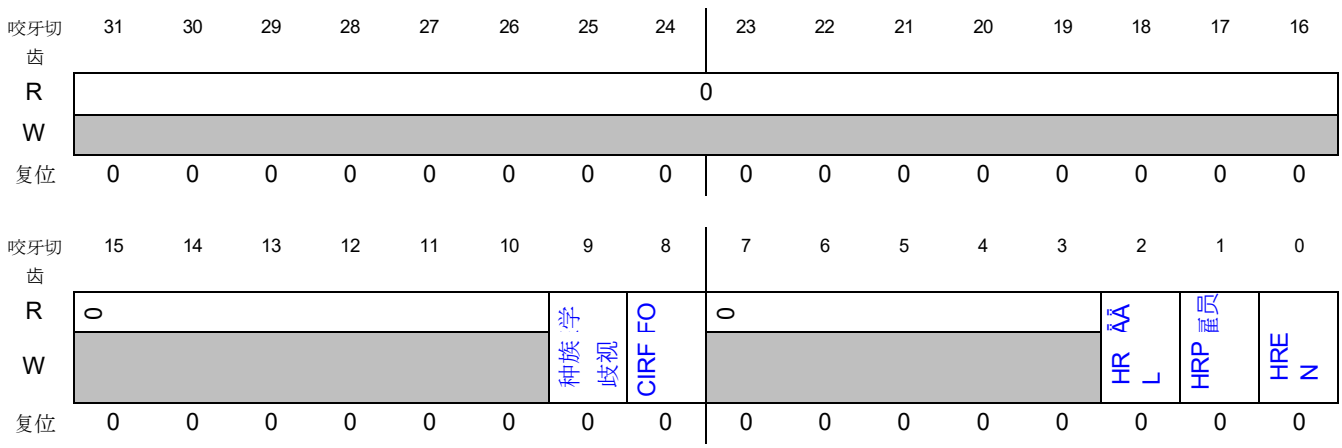
| 场地 | 职能 |
|---|---|
| RDDE | 1b - 启用DMA请求。 |
| 0<br>TDDE | 发送数据DMA启用<br>0b - 禁用DMA请求。1b - 启用DMA请求 |

## 46.3.1.8 配置寄存器0（CFGR0）

### 46.3.1.8.1 偏移

| 注册 | 偏移 |
|---|---|
| CFGR0 | 20h |

### 46.3.1.8.2 图示




### 46.3.1.8.3 领域

| 场地 | 职能 |
|---|---|
| 31-10<br>- | 保留 |

| | |
|---|---|
| 9<br>RDMO | 仅接收数据匹配<br>当启用时，所有没有导致DMF设置的接收数据都被丢弃。一旦DMF被设置，RDMO的配置就会被忽略。RDMO应在LPSPI空闲和DMF被清除时设置。当禁用RDMO时，在清除DMF之前清除RDMO以确保没有接收数据丢失。<br>0b - 接收的数据照常存储在接收FIFO中。1b - 接收的数据被丢弃，除非DMF被设置。 |
| 8 | 循环FIFO启用 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| CIRFIFO | 当启用时，发送FIFO的读指针被保存在一个临时寄存器中。发送FIFO将被正常清空，但一旦LPSPI处于空闲状态且发送FIFO为空，那么读指针的值将从临时寄存器中恢复。这将导致发送FIFO的内容被反复循环使用。<br>0b - 循环FIFO被禁用。 1b - 循环FIFO被启用。 |
| 7-3<br>- | 保留 |
| 2<br>HRSEL | 主机请求选择<br>选择主机请求输入的来源。当用LPSPI_HREQ引脚启用主机请求功能时，LPSPI_PCS[1]功能被禁用。<br>0b - 主机请求输入是引脚LPSPI_HREQ。 1b - 主机请求输入是输入触发。 |
| 1<br>HRPOL | 主机请求极性<br>配置主机请求引脚的极性。<br>0b - 低电平有效。<br>1b - 高电平有效。 |
| 0<br>HREN | 主机请求启用<br>当在主模式下启用时，LPSPI只有在主机请求输入被引出时才会开始新的SPI总线传输。当LPSPI繁忙时，主机请求输入被忽略。<br>0b - 主机请求被禁用。 1b - 主机请求被启用。 |

## 46.3.1.9 配置寄存器1（CFGR1）。

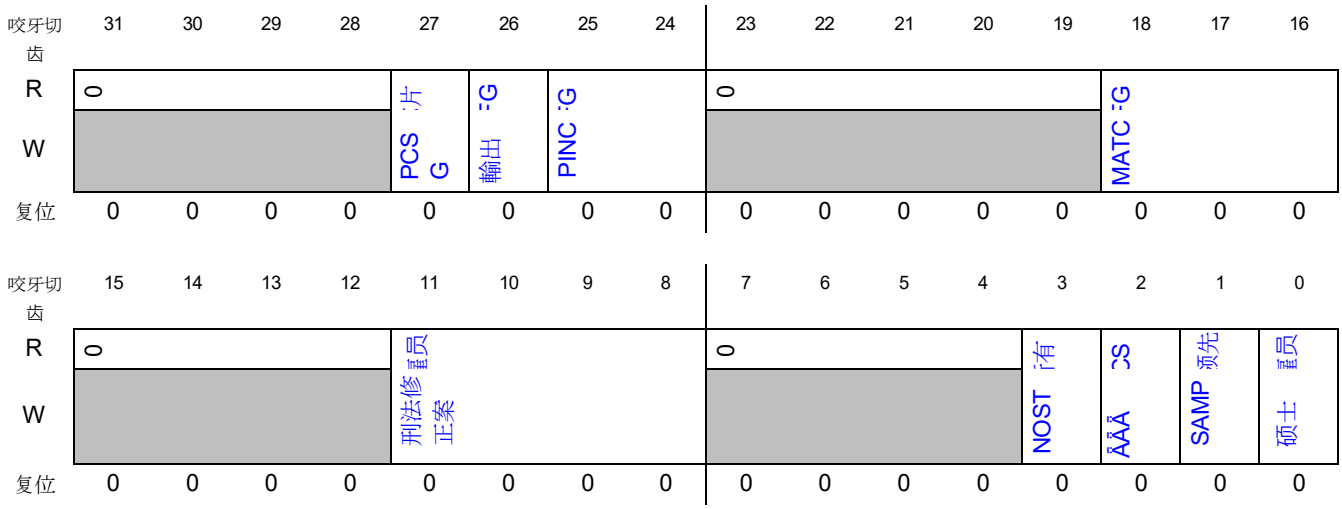
### 46.3.1.9.1 偏移

| 注册 | 偏移 |
|---|---|
| CFGR1 | 24h |

### 46.3.1.9.2 职能

CFGR1只能在LPSPI被禁用时被写入。

### 46.3.1.9.3 图示




### 46.3.1.9.4 领域

| 场地 | 职能 |
|---|---|
| 31-28<br>- | 保留 |
| 27<br>辽宁沈阳 | 周边芯片选择配置<br>如果执行4位传输，必须设置PCSCFG。<br>0b - PCS[3:2]被启用。1b - PCS[3:2]被禁用。 |
| 26<br>OUTCFG | 输出配置<br>配置输出数据在访问之间是否被三态化（LPSPI_PCS被否定）。<br>0b - 芯片选择被否定时，输出数据保留最后的值。1b - 芯片选择被否定时，输出数据被三态化。 |
| 25-24<br>铂尼弗格(PINCFG) | 引脚配置<br>配置哪些引脚在单比特传输期间用于输入和输出数据。<br>00b - SIN用于输入数据，SOUT用于输出数据。01b - SIN同时用于输入和输出数据。<br>10b - SOUT同时用于输入和输出数据。<br>11b - SOUT用于输入数据，SIN用于输出数据。 |
| 23-19<br>- | 保留 |
| 18-16<br>MATCFG | 匹配配置<br>配置将导致DMF设置的条件。 |

| | |
|---|---|
| | **注意：** *语法*。*是布尔值和，+是布尔值或000b - 匹配<br>被禁用。<br>001b - 保留<br>010b - 010b - 如果第一个数据字等于MATCH0或MATCH1，即（第一个数据字），则启用匹配。= match0 + match1)<br>011b - 011b - 如果任何数据字等于MATCH0或MATCH1，即（任何数据字=MATCH0+MATCH1），则启用匹配。 |

*表继续在下一页...*

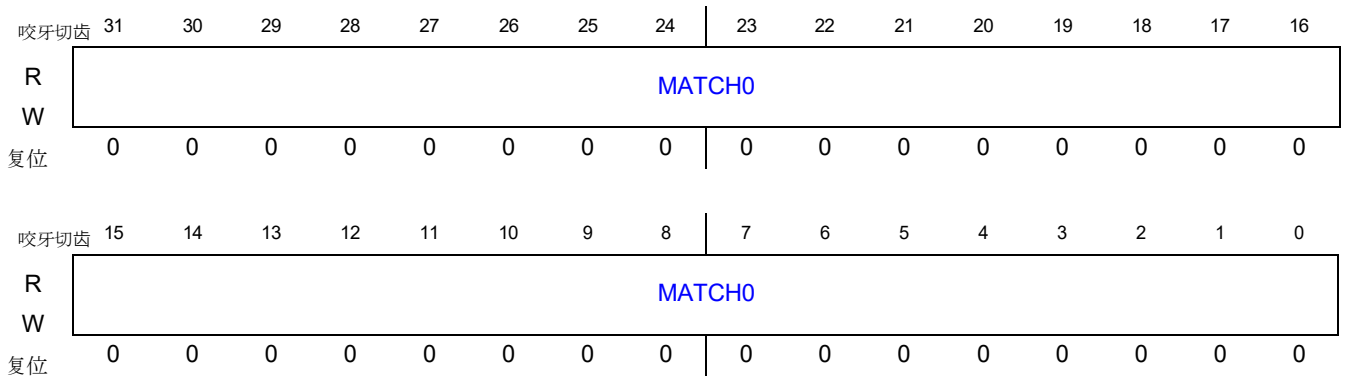
| 场地 | 职能 |
| --- | --- |
| | 100b - 100b - 如果第一个数据字等于MATCH0和第二个数据字等于MATCH1，即[（第一个数据字=MATCH0）*（第二个数据字=MATCH1）]，则启用匹配。<br>101b - 101b - 如果任何数据字等于MATCH0和下一个数据字等于MATCH1，即[（任何数据字=MATCH0）*（下一个数据字=MATCH1）]，则匹配被启用。<br>110b - 110b - 如果（第一个数据字和MATCH1）等于（MATCH0和MATCH1），即[（第一个数据字*MATCH1）=（MATCH0*MATCH1）]，匹配被启用。<br>111b - 111b - 如果（任何数据字和MATCH1）等于（MATCH0和MATCH1），即[（任何数据字*MATCH1）=（MATCH0*MATCH1）]，匹配被启用。 |
| 15-12<br>- | 保留 |
| 11-8<br>PCSPOL | 周边芯片选择极性<br>配置每个外设芯片选择引脚的极性。<br>0000b - PCSx为低电平有效。<br>0001b - PCSx为高电平有效。 |
| 7-4<br>- | 保留 |
| 3<br>不安装 | 没有摊位<br>在主模式下，当发送FIFO为空或接收FIFO为满时，LPSPI将停止传输，确保不会发生发送FIFO不足或接收FIFO超限。设置该位将禁用这一功能。<br>0b - 当发送FIFO为空或接收FIFO为满时，传输将停顿。<br>1b - 传输不会停滞，允许发送FIFO不足或接收FIFO超限发生。 |
| 2<br>呼叫中心 | 自动PCS<br>LPSPI从机通常要求PCS在两帧之间为负数，以便正确操作。当CPHA=1时，设置该位将使LPSPI在每个传输字结束时产生一个内部PCS信号。当该位被设置时，SCK必须在每个字之间保持至少4个LPSPI功能时钟周期（除以PRESCALE配置）的空闲，以确保正确操作。在主模式下，该位被忽略。<br>0b - 禁用自动PCS生成。1b - 启用自动<br>PCS生成。 |
| 1<br>样品 | 样品点<br>当设置时，LPSPI主控器将在一个延迟的LPSPI_SCK边沿上对输入数据进行采样。这改善了数据采样时的设置时间。在主模式下，延迟的LPSPI_SCK边沿的输入数据设置时间等于从模式下的输入数据设置时间。在从属模式下，该位被忽略。<br>0b - 在SCK边缘对输入数据进行采样。<br>1b - 在延迟的SCK边缘对输入数据进行采样。 |
| 0<br>硕士 | 主模式<br>将LPSPI配置为主模式或从模式。该位直接控制LPSPI_SCK和LPCPI_PCS引脚的方向。<br>0b - 从属模式。<br>1b - 主模式。 |

### 46.3.1.10 数据匹配寄存器0（DMR0）

### 46.3.1.10.1 偏移

| 注册 | 偏移 |
|---|---|
| DMR0 | 30h |

### 46.3.1.10.2 图示




### 46.3.1.10.3 领域

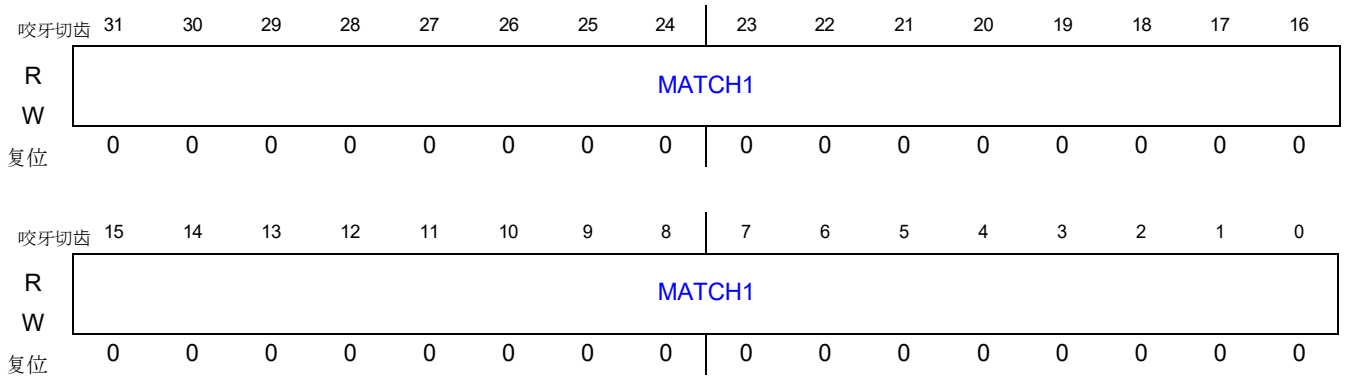
| 场地 | 职能 |
|---|---|
| 31-0<br>MATCH0 | 匹配0值<br>当接收数据匹配被启用时，与接收的数据进行比较。 |

## 46.3.1.11 数据匹配寄存器1（DMR1）。

### 46.3.1.11.1 偏移

| 注册 | 偏移 |
|---|---|
| DMR1 | 34h |

### 46.3.1.11.2 图示




### 46.3.1.11.3 领域

| 场地 | 职能 |
|---|---|
| 31-0<br>MATCH1 | 匹配1的价值<br>当接收数据匹配被启用时，与接收的数据进行比较。 |

## 46.3.1.12 时钟配置寄存器（CCR）

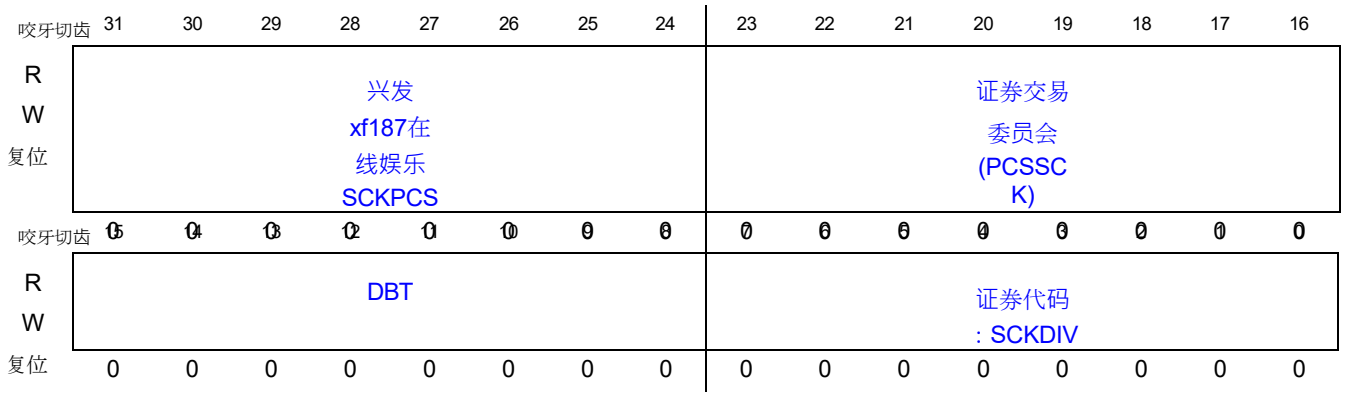
### 46.3.1.12.1 偏移

| 注册 | 偏移 |
|---|---|
| CCR | 40h |

### 46.3.1.12.2 职能

CCR只在主控模式下使用，当LPSPI被启用时不能改变。

### 46.3.1.12.3 图示




### 46.3.1.12.4 领域

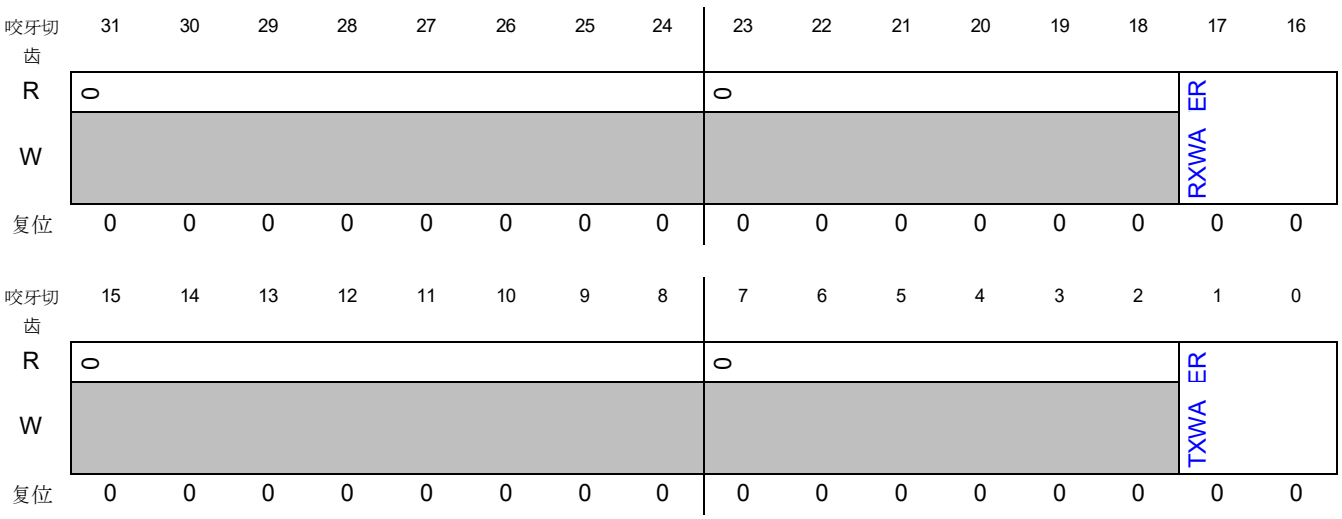
| 场地 | 职能 |
|---|---|
| 31-24<br>兴发xf187在线娱乐 SCKPCS | SCK到PCS的延迟<br>配置主控模式下从最后一个SCK边沿到PCS否定的延迟。该延迟等于LPSPI功能时钟的（SCKPCS + 1）个周期除以PRESCALE配置，最小延迟为1个周期。 |
| 23-16<br>证券交易委员会(PCSSCK) | PCS到SCK的延迟<br>配置主控模式下从PCS断言到第一个SCK边缘的延迟。该延迟等于LPSPI功能时钟的（PCSSCK + 1）周期除以PRESCALE配置，最小延迟为1周期。 |
| 15-8<br>DBT | 转移之间的延迟<br>配置主控模式下从PCS否定到下一次PCS断言的延迟。该延迟等于LPSPI功能时钟的（DBT+2）个周期除以PRESCALE配置，最小延迟为2个周期。注意，一半的延迟发生在PCS断言之前，另一半发生在PCS否定之后，完整的命令字只能在中间更新。<br>还配置了在主控模式下从一个传输字的最后一个SCK边沿到连续传输中下一个传输字的第一个SCK边沿的延迟。该延迟等于LPSPI功能时钟的（DBT+1）个周期除以PRESCALE配置，最小延迟为1个周期。 |
| 7-0<br>证券代码：SCKDIV | SCK分配器<br>配置主模式下SCK引脚的除法比率。SCK周期等于LPSPI功能时钟的（SCKDIV+2）个周期除以PRESCALE配置，最小周期为2个周期。如果周期是奇数，那么周期的前一半将比后一半长一个周期。 |

## 46.3.1.13 FIFO控制寄存器（FCR）

## 46.3.1.13.1 偏移

| 注册 | 偏移 |
|---|---|
| FCR | 58h |

## 46.3.1.13.2 图示




## 46.3.1.13.3 领域

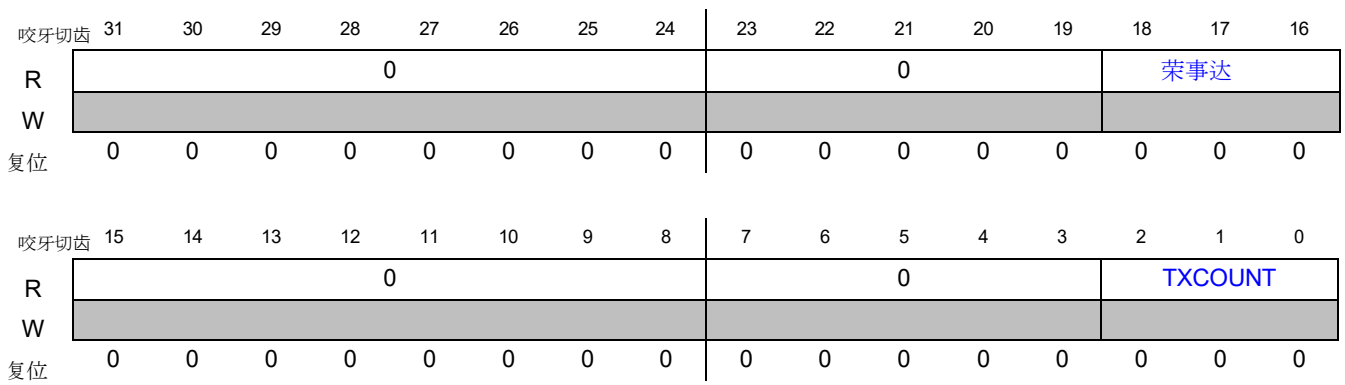
| 场地 | 职能 |
|---|---|
| 31-24<br>- | 保留 |
| 23-18<br>- | 保留 |
| 17-16<br>RXWATER | 接收FIFO水印<br>只要接收FIFO中的字数大于RXWATER，就会设置接收数据标志。写入一个等于或大于FIFO大小的值将被截断。 |
| 15-8<br>- | 保留 |
| 7-2<br>- | 保留 |
| 1-0<br>淡水河谷（<br>TXWATER | 发送FIFO水印<br>只要发送FIFO中的字数等于或小于TXWATER，就会设置发送数据标志。写入一个等于或大于FIFO大小的值将被截断。 |

# 46.3.1.14 FIFO状态寄存器（FSR）

## 46.3.1.14.1 偏移

| 注册 | 偏移 |
|---|---|
| 基金会 | 5Ch |

## 46.3.1.14.2 图示




## 46.3.1.14.3 领域

| 场地 | 职能 |
|---|---|
| 31-24<br>- | 保留 |
| 23-19<br>- | 保留 |
| 18-16<br>荣事达 | 接收FIFO计数<br>返回当前存储在接收FIFO中的字数。 |
| 15-8<br>- | 保留 |
| 7-3<br>- | 保留 |
| 2-0<br>TXCOUNT | 发送FIFO计数<br>返回当前存储在发送FIFO中的字数。 |

## 46.3.1.15 发送命令寄存器（TCR）

### 46.3.1.15.1 偏移

| 注册 | 偏移 |
| --- | --- |
| TCR | 60h |

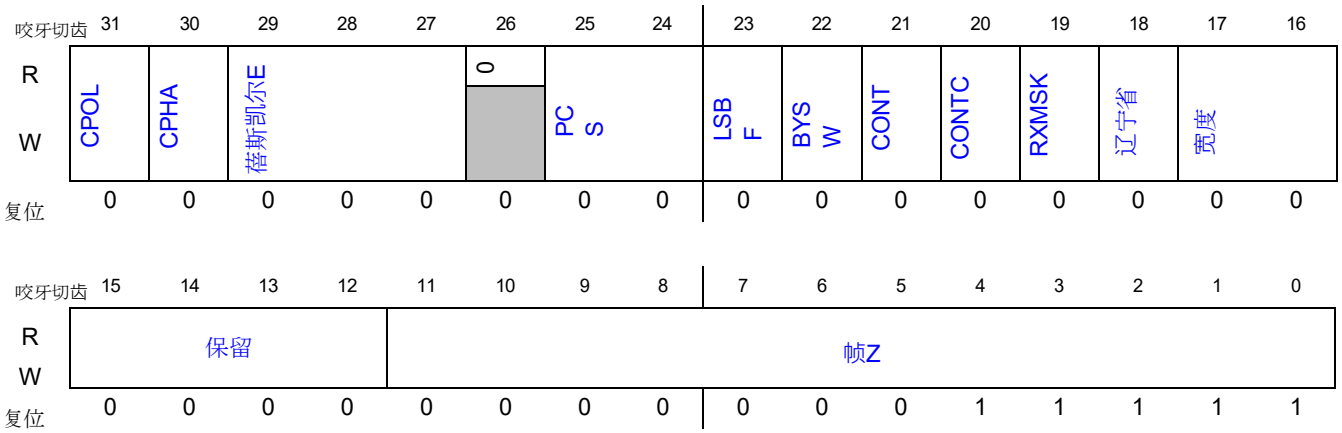
### 46.3.1.15.2 职能

对传输命令寄存器或传输数据寄存器的写入将按写入的顺序将数据推入传输FIFO。命令寄存器的写入将被标记，并导致命令寄存器的更新，在该条目到达FIFO的顶部后。这使得对命令字和传输数据本身的改变可以交错进行。改变命令字将导致所有后续的SPI总线传输使用新的命令字来执行。

- **在主模式下，**写一个新的命令字不会启动新的传输，除非TXMSK被设置。传输是由传输FIFO中的传输数据或新的命令字（设置了TXMSK）启动的。当LPSPI_PCS置位时，硬件将清除TXMSK。
- **在主控模式下，**如果在现有的帧完成之前改变命令字，那么现有的帧将终止，命令字将随之更新。如果新的命令字的CONTC被设置，并且命令字被写在一个帧大小的边界上，那么命令字可以在连续传输过程中被改变。
- **在从属模式下，**只有当LPSPI处于空闲状态并且没有SPI总线传输时，才应改变命令字。

读取发送命令寄存器将返回命令寄存器的当前状态。在发送命令寄存器从发送FIFO
加载的同时读取发送命令寄存器可能会返回不正确的寄存器值。建议在发送FIFO为空时读取发送命令寄存器，或多次读取该寄存器并比较返回值。

### 46.3.1.15.3 图示




### 46.3.1.15.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>CPOL | 时钟极性<br>这个字段只在帧之间更新。<br>0b - SCK的非活动状态值为低。1b - SCK的非活动状态值为高。 |
| 30<br>CPHA | 时钟相位<br>这个字段只在帧之间更新。<br>0b - 数据在SCK的前缘被捕获，在后缘被改变。1b - 数据在SCK的前缘被改变，在后缘被捕获。 |
| 29-27<br>预售 | 预分频器值<br>应用于所有SPI总线传输的时钟配置寄存器的预分频器。这个字段只在帧之间更新。<br>000b - 除以1。 001b - 除以2。 010b - 除以4。 011b - 除以8。 100b - 除以16。 101b - 除以32。 110b - 除以64。 111b - 除以128。 |
| 26<br>- | 保留 |
| 25-24<br>辽宁省 | 周边芯片选择<br>配置用于传输的外设芯片选择。这个字段只在帧之间更新。<br>00b - 使用LPSPI_PCS[0]传输 01b - 使用LPSPI_PCS[1]传输 10b - 使用 |

| | |
|---|---|
| | LPSPI_PCS[2] 传 输 11b - 使 用 LPSPI_PCS[3]传输 |
| 23<br>LSBF | LSB优先<br>0b - 数据以MSB优先传输。1b - 数据以LSB优先传输。 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| 22<br>BYSW | 字节互换<br>字节交换将把[31:24]的内容与[7:0]交换，[23:16]与[15:8]交换，用于从FIFO读取的每个传输数据字和存储到FIFO的每个接收数据字（或与匹配寄存器比较）。<br>0b - 字节互换被禁用。<br>1b - 启用字节互换。 |
| 21<br>CONT | 连续转移<br>在主控模式下，连续传输将在帧大小结束时保持PCS断言，直到收到一个开始新帧的命令字。<br>在从属模式下，当连续传输被启用时，LPSPI将只传输第一个FRAMESZ位，之后它将假设一个32位移位寄存器来传输收到的数据。<br>0b - 禁用连续传输。 1b - 启用连续传输。 |
| 20<br>CONTC | 继续指挥<br>在主控模式下，该位允许在连续传输过程中改变命令字。初始命令字必须启用连续传输（CONT=1），持续命令必须设置该位（CONTC=1），持续命令字必须在帧大小边界上加载。例如，如果连续传输的帧大小为64位，那么必须在64位边界上加载一个继续命令字。<br>0b - 开始新的传输的命令词。 1b - 继续传输的命令字。 |
| 19<br>RXMSK | 接收数据屏蔽<br>当设置时，接收数据被屏蔽（接收数据不被存储在接收FIFO中）。<br>0b - 正常传输。<br>1b - 接收数据被屏蔽。 |
| 18<br>辽宁省 | 发送数据屏蔽<br>当设置时，传输数据被屏蔽（没有数据从传输FIFO加载，输出引脚被三态化）。在主控模式下，该位将启动一个新的传输，不能被另一个命令字中止，该位将在传输结束时被硬件清除。<br>0b - 正常传输。<br>1b - 屏蔽发送数据。 |
| 17-16<br>宽度 | 转移宽度<br>对于2位或4位传输，必须设置RXMSK或TXMSK。<br>00b - 单位传输。<br>01b - 两位数传输。<br>10b - 四位传输。<br>11b - 保留。 |
| 15-12<br>- | 保留。软件只能对该位写0。 |

| | |
|---|---|
| 11-0<br><br>帧Z | 框架尺寸<br><br>配置帧的大小，比特数等于（FRAMESZ + 1）。<br>• 帧的最小尺寸是8比特。<br>• 如果帧的大小大于32位，那么帧就被分成多个32位的字；每个字从发送FIFO加载，并分别存储在接收FIFO中。<br>• 如果帧的大小不能被32整除，那么发送FIFO的最后一次加载和接收FIFO的存储将包含剩余的比特。例如，一个72位的传输将由3个字组成：第1和第2个字是32位，第3个字是8位。<br>• 最小字数为2位；不支持33位（或类似）的帧大小。 |

## 46.3.1.16 发送数据寄存器（TDR）

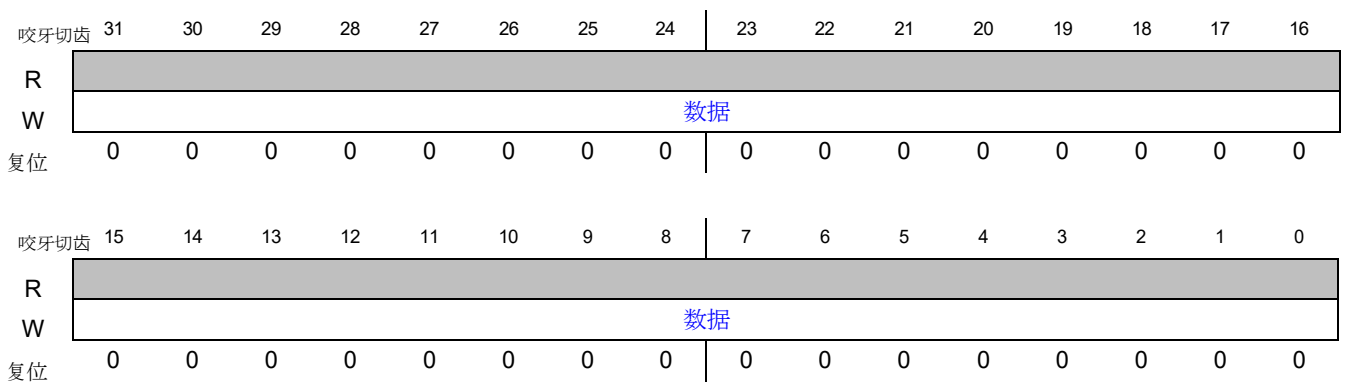
### 46.3.1.16.1 偏移

| 注册 | 偏移 |
|---|---|
| TDR | 64h |

### 46.3.1.16.2 职能

对传输命令寄存器或传输数据寄存器的写入将按照写入的顺序将数据推入传输FIFO。

### 46.3.1.16.3 图示




### 46.3.1.16.4 领域

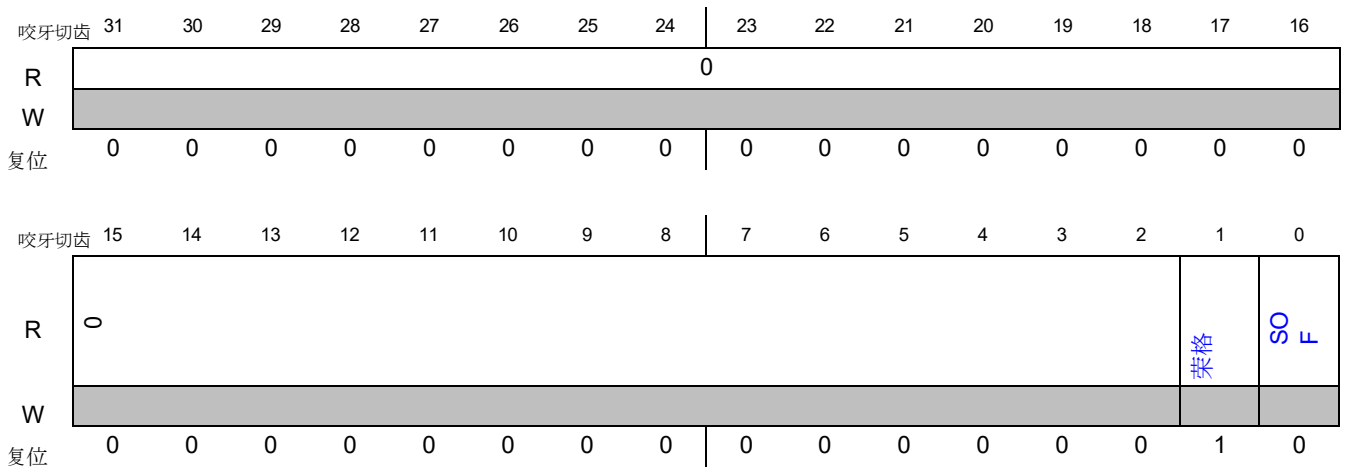
| 场地 | 职能 |
|---|---|
| 31-0<br>数据 | 发送数据<br>8位和16位的发送数据写入都会对写入的数据进行零扩展，并将数据推入发送FIFO。 |

## 46.3.1.17 接收状态寄存器（RSR）

### 46.3.1.17.1 偏移

| 注册 | 偏移 |
|---|---|
| 共和国 | 70h |

### 46.3.1.17.2 图示




### 46.3.1.17.3 领域

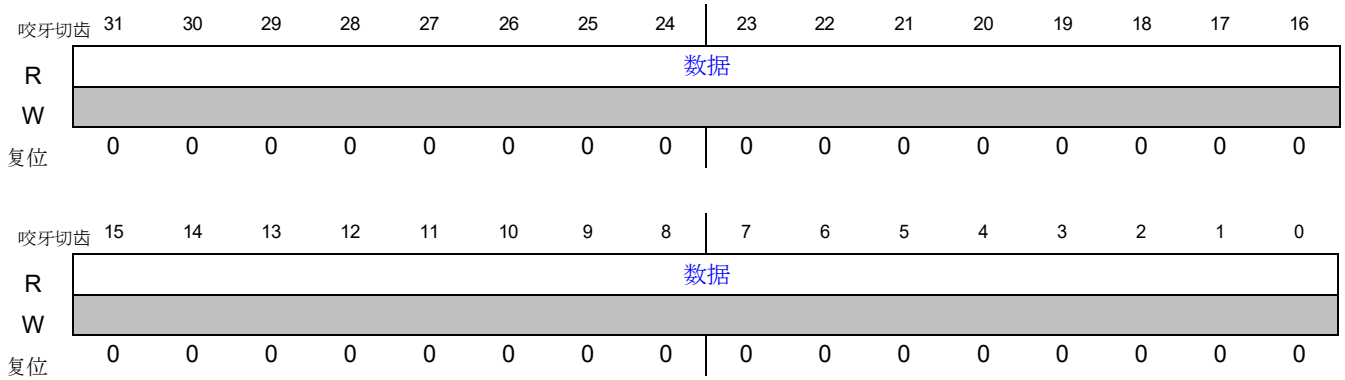
| 场地 | 职能 |
|---|---|
| 31-2<br>- | 保留 |
| 1<br>荣格 | RX FIFO清空<br>0b - RX FIFO不为空。1b - RX FIFO是空的。 |
| 0<br>SOF | 帧的开始<br>表示这是LPSPI_PCS断言后收到的第一个数据字。<br>0b - 在LPSPI_PCS断言后收到的后续数据字。1b - 在LPSPI_PCS断言后收到的第一个数据字。 |

## 46.3.1.18 接收数据寄存器（RDR）

### 46.3.1.18.1 偏移

| 注册 | 偏移 |
| --- | --- |
| RDR | 74h |

### 46.3.1.18.2 图示




### 46.3.1.18.3 领域

| 场地 | 职能 |
|---|---|
| 31-0<br>数据 | 接收数据 |

## 46.4 职能描述

### 46.4.1 时钟和复位

#### 46.4.1.1 功能性时钟

LPSPI功能时钟与总线时钟是异步的，如果它在低功耗模式下保持启用状态，那么LPSPI可以在主从模式下执行SPI总线传输和低功耗唤醒。LPSPI将功能时钟除以一个预分频器，得到的频率必须比SPI外部时钟（LPSPI_SCK）频率至少快2倍。

#### 46.4.1.2 外部时钟

LPSPI移位寄存器直接由LPSPI_SCK时钟驱动，该时钟在主模式下由内部产生，在从模式下由外部提供。

#### 46.4.1.3 总线时钟

总线时钟只用于对控制和配置寄存器的总线访问。总线时钟频率必须足以支持LPSPI寄存器（包括FIFO）的数据带宽要求。

#### 46.4.1.4 芯片复位

LPSPI的逻辑和寄存器在芯片复位时被重置为其默认状态。

#### 46.4.1.5 软件复位

LPSPI在控制寄存器中实现了一个软件复位位。CR[RST]将把所有的逻辑和寄存器复位到其默认状态，但CR本身除外。

#### 46.4.1.6 FIFO复位

LPSPI实现了只写的控制位，重置了发送/命令FIFO（CR[RTF]）和接收FIFO（CR[RRF]）。FIFO被重置后为空。

### 46.4.2 主模式

#### 46.4.2.1 发送和命令FIFO

发送和命令FIFO是一个组合FIFO，包括发送数据和命令字。命令字通过写入发送命令寄存器被存储到发送/命令FIFO中。发送数据字通过写入发送数据寄存器被存储到发送/命令FIFO中。

当一个命令字处于发送/命令FIFO的顶部时，会发生以下动作。

- 如果LPSPI处于帧之间，则命令字从FIFO中拉出并控制所有后续的传输。
- 如果LPSPI很忙，并且现有的CONT位被清除或新的CONTC值被清除，那么SPI帧将在现有字的末尾完成，忽略

FRAMESZ配置。然后，命令字被从FIFO中拉出，控制所有后续的传输（或直到命令字的下一次更新）。

- 如果LPSPI繁忙，现有的CONT位被设置，而新的CONTC值被设置，则在现有帧的最后一个LPSPI_SCK脉冲期间（基于FRAMESZ配置），命令字被从FIFO中拉出，在该帧的其余部分（或直到下次更新命令字）继续使用新的命令值。当CONTC被设置时，只有命令字的低24位被更新。

现有命令字的当前状态可以通过读取发送命令寄存器来读取。发送命令寄存器被写入后至少需要三个LPSPI功能时钟周期才能更新（假设FIFO为空），并且LPSPI必须被启用（CR[MEN]被设置）。

写入发送命令寄存器并不启动SPI总线传输，除非TXMSK位被设置。当TXMSK被设置时，新的命令字将不会被加载，直到现有的帧结束（基于FRAMESZ配置），TXMSK位将在传输结束时被清除。

下表描述了由命令字控制的属性。

**表46-3.主模式下的LPSPI命令字**

| 场地 | 描述 | 转移过程中的修改 |
|---|---|---|
| CPOL | 配置LPSPI_SCK引脚的极性。CPOL值的任何变化都会引起LPSPI_SCK引脚的转换。 | N |
| CPHA | 配置传输的时钟相位。 | N |
| 预售 | 配置用于划分LPSPI功能时钟的预分频器，以产生SPI总线传输的定时参数。与PCS一起改变PRESCALE可以使LPSPI以不同的频率连接到不同的从属设备。 | N |
| 辽宁省 | 配置哪个LPSPI_PCS对传输进行断言，LPSPI_PCS的极性是静态的，由PCSPOL进行配置。如果PCSCFG被设置，那么PCS[3:2]就不应该被选择。 | N |
| LSBF | 配置是先传输/接收LSB（位0）还是MSB（32位字的位31）。 | Y |
| BYSW | 在传输和接收数据时，启用每个32位字的字节交换。在与将数据组织成大的 endean 的设备对接时，可能很有用。 | Y |
| CONT | 配置连续传输，使PCS在各帧之间保持断电状态（如FRAMESZ所配置的）。需要一个新的命令字来导致PCS被否定。还支持在帧大小边界上改变命令字。 | Y |
| CONTC | 表示这是现有连续传输的一个新命令字。CONTC位在没有写入帧边界的发送/命令FIFO时被忽略。 | Y |
| RXMSK | 屏蔽接收数据，不存储到接收FIFO或执行接收数据匹配。对于半双工传输或配置在接收数据匹配期间比较哪些字段是有用的。 | Y |

*表继续在下一页...*

表46-3.主模式下的LPSPI命令字（续）

| 场地 | 描述 | 转移过程中的修改 |
| --- | --- | --- |
| 辽宁省 | 对传输数据进行屏蔽，这样数据就不会从传输FIFO中被拉出，输出数据引脚被三态化（除非由OUTCFG配置）。对半双工传输有用。 | Y |
| 宽度 | 配置每个LPSPI_SCK脉冲上移位的位数。<br>• 1位传输支持半双工或全双工数据格式的传统SPI总线传输。<br>• 2位和4位传输对于连接QuadSPI存储器设备很有用，并且只支持半双工数据格式（TXMSK或RXMSK中至少有一个必须被设置）。 | Y |
| 帧Z | 配置帧的大小，比特数等于（FRAMESZ + 1）。<br>• 帧的最小尺寸是8比特。<br>• 如果帧的大小大于32位，那么帧就被分成多个32位的字；每个字从发送FIFO加载，并分别存储在接收FIFO中。<br>• 如果帧的大小不能被32整除，那么发送FIFO的最后一次加载和接收FIFO的存储将包含剩余的比特。例如，一个72位的传输将由3个字组成：第1和第2个字是32位，第3个字是8位。<br>• 最小字数为2位；不支持33位（或类似）的帧大小。 | Y |

当数据被写入发送FIFO，HREQ引脚被断言（或禁用）并且LPSPI被启用时，LPSPI启动SPI总线传输。SPI总线传输使用传输命令寄存器中配置的属性和时钟配置寄存器中的定时参数来执行传输。一旦达到FRAMESZ配置，或者当一个新的发送命令字处于发送/命令FIFO的顶部时，SPI总线传输就会结束。HREQ输入只有在LPSPI下次空闲时（完成当前的传输且发送/命令寄存器为空）才会被检查。

发送/命令FIFO还支持循环FIFO功能。这允许LPSPI主控器（周期性地）重复一个可以在发送/命令FIFO内的短数据传输，而不需要额外的FIFO访问。当循环FIFO被启用时，FIFO读指针的当前状态被保存，状态标志不更新。一旦发送/命令FIFO被认为是空的，并且LPSPI处于空闲状态，FIFO读指针就会用保存的版本恢复，因此在启用循环FIFO模式时，发送/命令FIFO的内容不会被永久地从FIFO中拉出。

### 46.4.2.2 接收FIFO和数据匹配

接收FIFO用于在SPI总线传输期间存储接收数据。当RXMSK被设置时，接收数据被丢弃，而不是存储在接收FIFO中。

接收数据在帧结束时被写入接收FIFO。在多字或连续传输期间，在从发送FIFO读取新的发送数据的同时，接收数据也被写入接收FIFO。如果在连续传输过程中，发送FIFO是空的，那么只有在发送FIFO被写入或命令寄存器被写入以结束帧后，接收数据才会被写入接收FIFO。

接收数据支持接收数据匹配功能，该功能可以将接收到的数据与两个字中的一个或与一个屏蔽的数据字进行匹配。数据匹配功能也可以被配置为只比较自帧开始以来收到的前一个或两个数据字。由于RXMSK位而已经被丢弃的接收数据不能导致数据匹配被设置，并将推迟对第一个接收数据字的匹配，直到所有被丢弃的数据被接收之后。接收器匹配功能也可以通过CFGR0[RDMO]控制位配置为丢弃所有接收数据，直到检测到数据匹配。在数据匹配后清除CFGR0[RDMO]控制位时，在清除SR[DMF]之前清除CFGR0[RDMO]，以允许所有后续数据被接收。

### 46.4.2.3 计时参数

下表列出了用于所有SPI总线传输的定时参数，这些定时参数是相对于LPSPI功能时钟除以PRESCALE配置的。尽管在LPSPI繁忙时不能改变时钟配置寄存器，但在传输之间可以用命令寄存器改变PRESCALE配置，以支持与不同频率的从属设备的接口。

**表46-4.LPSPI定时参数**

| 场地 | 描述 | 闵行区 | 最大 |
|---|---|---|---|
| 证券代码：SCKDIV | 将LPSPI_SCK的时钟周期配置为（SCKDIV+2）个周期。当配置为奇数周期时，LPSPI_SCK周期的前一半比后一半长一个周期。 | 0 (2个周期) | 255 (257个周期) |
| DBT | 将PCS否定和下一个PCS断言之间的最小延迟配置为（DBT+2）个周期。当命令字在传输之间被更新时，在命令字更新和LPSPI_PCS引脚上的任何变化之间至少有（DBT/2）+1个周期。 | 0 (2个周期) | 255 (257个周期) |

*表继续在下一页...*

表46-4.LPSPI定时参数（续）

| 场地 | 描述 | 闵行区 | 最大 |
|---|---|---|---|
| DBT | 在连续传输过程中，将一个帧的最后一个SCK边沿和继续传输的第一个SCK边沿之间的延迟配置为（DBT + 1）个周期。这在外部从机需要在SPI总线传输的不同字之间有较大延迟的情况下非常有用。 | 0 (1个周期) | 255 (256周期) |
| 证券交易委员会(PCSSCK) | 配置PCS断言和第一个SCK边沿之间的最小延迟为（PCSSCK + 1）周期。 | 0 (1个周期) | 255 (256周期) |
| 兴发xf187在线娱乐 SCKPCS | 配置最后一个SCK边沿和PCS断言之间的最小延迟为（SCKPCS + 1）周期。 | 0 (1个周期) | 255 (256周期) |

### 46.4.2.4 引脚配置

LPSPI_SIN和LPSPI_SOUT引脚可以通过PINCFG配置来交换方向，甚至支持同一引脚的半双工传输。

OUTCFG配置可用于确定输出数据引脚（例如：LPSPI_SOUT）在LPSPI_PCS被否定时是否会三态，或者是否会简单地保留最后的值。当在单比特传输模式下使用同一数据引脚进行半双工传输，或在2比特和4比特传输模式下进行任何传输时，那么输出数据引脚必须被配置为在LPSPI_PCS被否定时三态。

PCSCFG配置用于禁用LPSPI_PCS[3:2]功能，并将其用于四数据传输。在进行四元数据传输时必须启用该选项。

### 46.4.2.5 时钟环回

LPSPI主控器可以被配置为使用两个时钟之一对输入数据（例如：LPSPI_SIN）进行采样，即直接使用LPSPI_SCK输出时钟或该时钟的延迟版本。LPSPI_SCK的延迟版本由LPSPI_SCK引脚的输出延迟加上LPSPI_SCK引脚的输入延迟组成，并通过设置CFGR1[SAMPLE]进行配置。启用LPSPI_SCK的环回版本可以改善从机输入数据的设置时间。

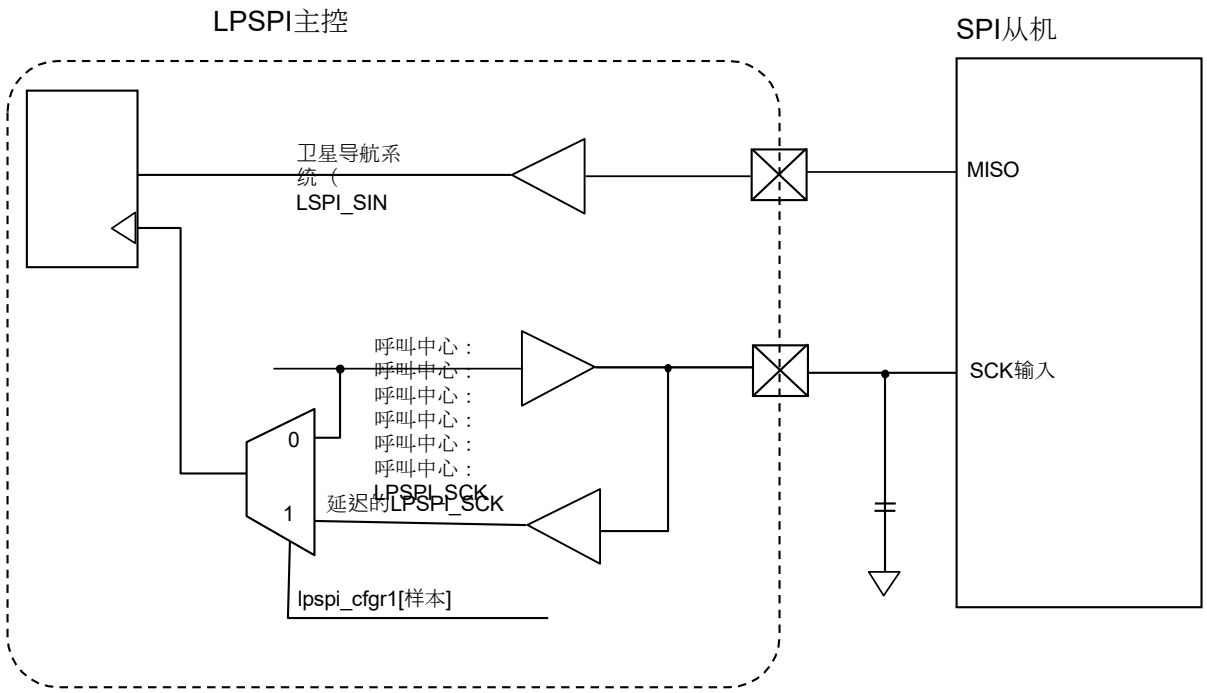



**图46-2.时钟环回**

关于主环回模式下的具体输入设置时间，请参考设备数据手册。

## 46.4.3 从属模式

LPSPI从属模式使用与主模式相同的移位寄存器和逻辑，但不使用时钟配置寄存器，在SPI总线传输过程中，发送命令寄存器必须保持静态。

### 46.4.3.1 发送和命令FIFO

在启用从属模式的LPSPI之前，应先初始化发送命令寄存器，尽管命令寄存器在LPSPI被启用后才会更新。一旦启用，只有在LPSPI处于空闲状态时才应改变发送命令寄存器。下表列出了命令寄存器在从属模式下的功能。

**表46-5.从属模式下的LPSPI命令字**

| 场地 | 描述 |
|---|---|
| CPOL | 配置外部LPSPI_SCK输入的极性。 |
| CPHA | 配置传输的时钟相位。 |
| 预售 | 配置LPSPI功能时钟预分频器。 |

*表继续在下一页...*

表46-5.从属模式下的LPSPI命令字（续）

| 场地 | 描述 |
|---|---|
| 辽宁省 | 配置使用哪个LPSPI_PCS，LPSPI_PCS的极性是静态的，由PCSPOL配置。如果PCSCFG被设置，那么PCS[3:2]就不应该被选择。 |
| LSBF | 配置是先传输/接收LSB（位0）还是MSB（32位字的位31）。 |
| BYSW | 在传输和接收数据时，启用每个32位字的字节交换。在与将数据组织成大的 endean 的设备对接时，可能很有用。 |
| CONT | 当设置时，只有第一个FRAMSZ位会被LPSPI传送/接收。 |
| CONTC | CONTC位在从属模式下被保留。 |
| RXMSK | 屏蔽接收数据，不存储到接收FIFO或执行接收数据匹配。对于半双工传输或配置在接收数据匹配期间比较哪些字段是有用的。 |
| 辽宁省 | 对传输数据进行屏蔽，这样数据就不会从传输FIFO中被拉出，输出数据引脚被三态化（除非由OUTCFG配置）。对半双工传输有用。 |
| 宽度 | 配置每个LPSPI_SCK脉冲上移位的位数。<br>• 1位传输支持半双工或全双工数据格式的传统SPI总线传输。<br>• 2位和4位传输对于连接QuadSPI存储器设备很有用，并且只支持半双工数据格式（TXMSK或RXMSK中至少有一个必须被设置）。 |
| 帧Z | 配置帧的大小，比特数等于（FRAMESZ + 1）。<br>• 帧的最小尺寸是8比特。<br>• 如果帧的大小大于32位，那么帧就被分成多个32位的字；每个字从发送FIFO加载，并分别存储在接收FIFO中。<br>• 如果帧的大小不能被32整除，那么发送FIFO的最后一次加载和接收FIFO的存储将包含剩余的比特。例如，一个72位的传输将由3个字组成：第1和第2个字是32位，第3个字是8位。<br>• 最小字数为2位；不支持33位（或类似）的帧大小。 |

发送FIFO必须在LPSPI_PCS输入被激活之前被发送数据填满，否则发送错误标志将被设置。

### 46.4.3.2 接收FIFO和数据匹配

接收FIFO用于在SPI总线传输期间存储接收数据。当RXMSK被设置时，接收数据被丢弃，而不是存储在接收FIFO中。

接收数据支持一个接收数据匹配功能，该功能可以将接收的数据与两个字中的一个或与一个屏蔽的数据字进行匹配。数据匹配功能也可以被配置为只比较自帧开始以来收到的前一个或两个数据字。由于RXMSK位而已经被丢弃的接收数据不能导致数据匹配被设置，并将推迟对第一个接收数据字的匹配，直到所有被丢弃的数据被接收之后。接收器匹配功能也可以通过CFGR0[RDMO]控制位配置为丢弃所

有接收数据，直到检测到数据匹配。在数据匹配后清除CFGR0[RDMO]控制位时，在清除SR[DMF]之前清除CFGR0[RDMO]，以允许所有后续数据被接收。

*表继续在下一页...*

### 46.4.3.3 时钟接口

LPSPI支持与只提供时钟和数据引脚（不需要LPSPI_PCS）的外部主控器进行接口。这需要使用CPHA=1，将LPSPI_PCS输入配置为总是被断言（配置PCSPOL）并设置AUTOPCS位。当AUTOPCS被设置时，在一个字的最后一个LPSPI_SCK边沿和下一个字的第一个LPSPI_SCK边沿之间至少需要4个LPSPI功能时钟周期（除以PRESCALE配置）。

## 46.4.4 中断和DMA请求

下表说明了可以产生LPSPI中断和LPSPI发送/接收DMA请求的状态标志。

**表46-6.LPSPI中断和DMA请求**

| 旗帜 | 描述 | 中断 | DMA请求 | 低功率唤醒 |
|---|---|---|---|---|
| TDF | 根据TXWATER的配置，数据可以被写入发送FIFO。 | Y | 德克萨斯州 | Y |
| RDF | 根据RXWATER的配置，可以从接收FIFO中读取数据。 | Y | 荣格 | Y |
| WCF | 字完成，字的最后一位已被采样。 | Y | N | Y |
| 融资基金 | 框架完成，PCS已被否定。 | Y | N | Y |
| TCF | 传输完成，PCS已被否定，发送/命令FIFO为空。 | Y | N | Y |
| TEF | 发送错误标志，表示发送/命令FIFO不足。当NOSTALL清零时，该位在主控模式下不能设置。 | Y | N | Y |

| | | | | |
|---|---|---|---|---|
| REF | 接收错误标志，表示接收FIFO溢出。在主模式下，当NOSTALL清零时，该位不能设置。 | Y | N | Y |

*表继续在下一页...*

表46-6.LPSPI中断和DMA请求（续）

| 旗帜 | 描述 | 中断 | DMA请求 | 低功率唤醒 |
|---|---|---|---|---|
| DMF | 数据匹配标志，收到的数据已与配置的数据匹配值相匹配。 | Y | N | Y |
| MBF | LPSPI正忙于执行一个SPI总线传输。 | N | N | N |

## 46.4.5 周边触发器

LPSPI外设触发器与其他外设的连接是由设备决定的。

### 46.4.5.1 输出触发器

LPSPI产生两个输出触发器，可以连接到设备上的其他外设。帧输出触发器在每一帧结束时（当PCS被否定时）发出信号，并在PCS下次发出信号之前保持正向状态。字输出触发器在每个接收到的字结束时发出信号，并在一个LPSPI_SCK周期内保持正向状态。

### 46.4.5.2 输入触发器

可以选择LPSPI输入触发器代替LPSPI_HREQ输入来控制LPSPI总线传输的开始。输入触发器是同步的，必须至少在LPSPI功能时钟除以PRESCALE配置的两个周期内断言才能被检测到。请注意，当LPSPI繁忙时，LPSPI_HREQ输入（以及输入触发器）被忽略。

# 第47章
# 低功耗互连电路（LPI2C）。

## 47.1 芯片特定的LPI2C信息

### 47.1.1 实例化信息

下表总结了该模块在产品系列中各芯片的实现情况。

**表47-1.LPI2C配置**

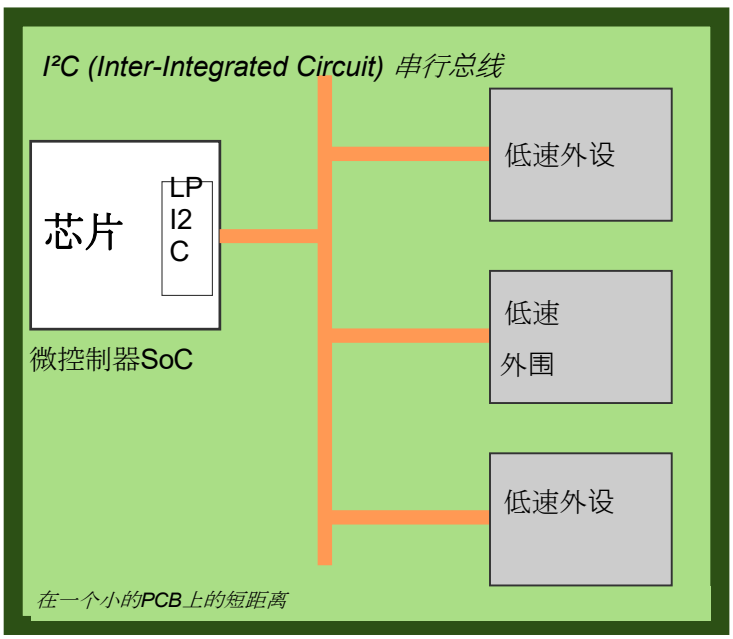
| 芯片 | 实例 | TX FIFO (字) | RX FIFO (字) | SMBus | 启用从属模式 |
|---|---|---|---|---|---|
| S32K116 | LPI2C0 | 4 | 4 | 是 | 是 |
| S32K118 | LPI2C0 | 4 | 4 | 是 | 是 |
| S32K142 | LPI2C0 | 4 | 4 | 是 | 是 |
| S32K144 | LPI2C0 | 4 | 4 | 是 | 是 |
| S32K146 | LPI2C0 | 4 | 4 | 是 | 是 |
| S32K148 | LPI2C0 | 4 | 4 | 是 | 是 |
| | LPI2C1 | 4 | 4 | 是 | 是 |

该器件不支持低漏电和等待模式。关于可用的电源模式的详细信息，请参见模块在可用的低功率模式下的操作。

该设备不支持HS-模式（波特率超过400 kbps）。

LPI2C模块包括SMBus支持和DMA支持。它还具有在停止/VLPS模式下可选的地址匹配唤醒功能。

## 47.2 简介




*维基百科。*

I²C（Inter-Integrated Circuit），发音为I-squared-C，是飞利浦半导体（现在的恩智浦半导体）发明的一种多主、多从、分组交换、单端、串行计算机总线。它通常用于将低速外设IC连接到处理器和微控制器上的短距离板内通信。

**图47-1.I2C串行总线示例**

LPI2C是一个低功耗的互连电路（I2C）模块，支持作为主站和/或从站与I2C总线的有效接口。

- LPI2C实现了对标准模式、快速模式、快速模式+和超快速工作模式的逻辑支持。
- LPI2C被设计为使用很少的CPU开销，对FIFO寄存器访问进行DMA卸载。
- 如果有合适的时钟，LPI2C可以在停止模式下继续工作。

LPI2C模块还符合系统管理总线（SMBus）规范的第二版。来自*维基百科。*系统管理总线（缩写为SMBus或SMB）是一种单端简单的两线制总线，用于轻量级通信的目的。

### 47.2.1 特点

LPI2C支持。

- 支持标准、快速、快速+和超快速模式
- 从属模式下的高速模式（HS）。

- 主模式下的高速模式（HS），如果SCL引脚实现了电流源上拉（特定设备）。
- 支持多主站，包括同步和仲裁。多主意味着可以有任何数量的主节点。此外，主站和从站的角色可以在消息之间改变（在发送STOP之后）。
- 时钟拉伸：来自*维基百科*：多个节点可能同时驱动线路。如果任何节点驱动线路为低电平，那么该线路将为低电平。试图传输逻辑1的I2C节点（即让线路浮动在高位）可以检测到这种情况（线路为低电平），并得出结论，另一个节点在同时活动。
  - 当在SCL线上使用节点检测时，这被称为*时钟拉伸*，它被用作从机的流量控制机制。
  - 当在SDA线上使用节点检测时，这被称为*仲裁*，它确保一次只有一个发射器。
- 一般调用，7位和10位寻址
- 软件复位、START字节和设备ID（也需要软件支持）

LPI2C主站支持。

- 命令/发送FIFO为4个字。
- 接收4个字的FIFO。
- 命令FIFO将等待空闲的I2C总线，然后启动传输。
- 命令FIFO可以启动（重复）START和STOP条件，以及一个或多个主-接收器传输
- STOP条件可以由命令FIFO产生，也可以在发送FIFO为空时自动产生。
- 主机请求输入，控制I2C总线传输的开始时间
- 灵活的接收数据匹配可以在数据匹配时产生中断和/或丢弃不需要的数据
- 标志和可选的中断，以示重复的START条件、STOP条件、仲裁丢失、意外的NACK和命令字错误。
- 支持可配置的总线空闲超时和引脚停顿超时

LPI2C从机支持。

- 独立的I2C从属寄存器，以尽量减少因主/从而产生的软件开销开关
- 支持7位或10位寻址、地址范围、SMBus警报和一般调用地址
- 支持中断或DMA请求的传输数据寄存器
- 支持中断或DMA请求的接收数据寄存器
- 软件可控制ACK或NACK，可选择对ACK/NACK位进行时钟拉伸

- 可配置的时钟拉伸，以避免发送FIFO不足和接收FIFO超限的错误
- 数据包结束时的标志和可选的中断，STOP条件，或位错误检测

## 47.2.2 框图

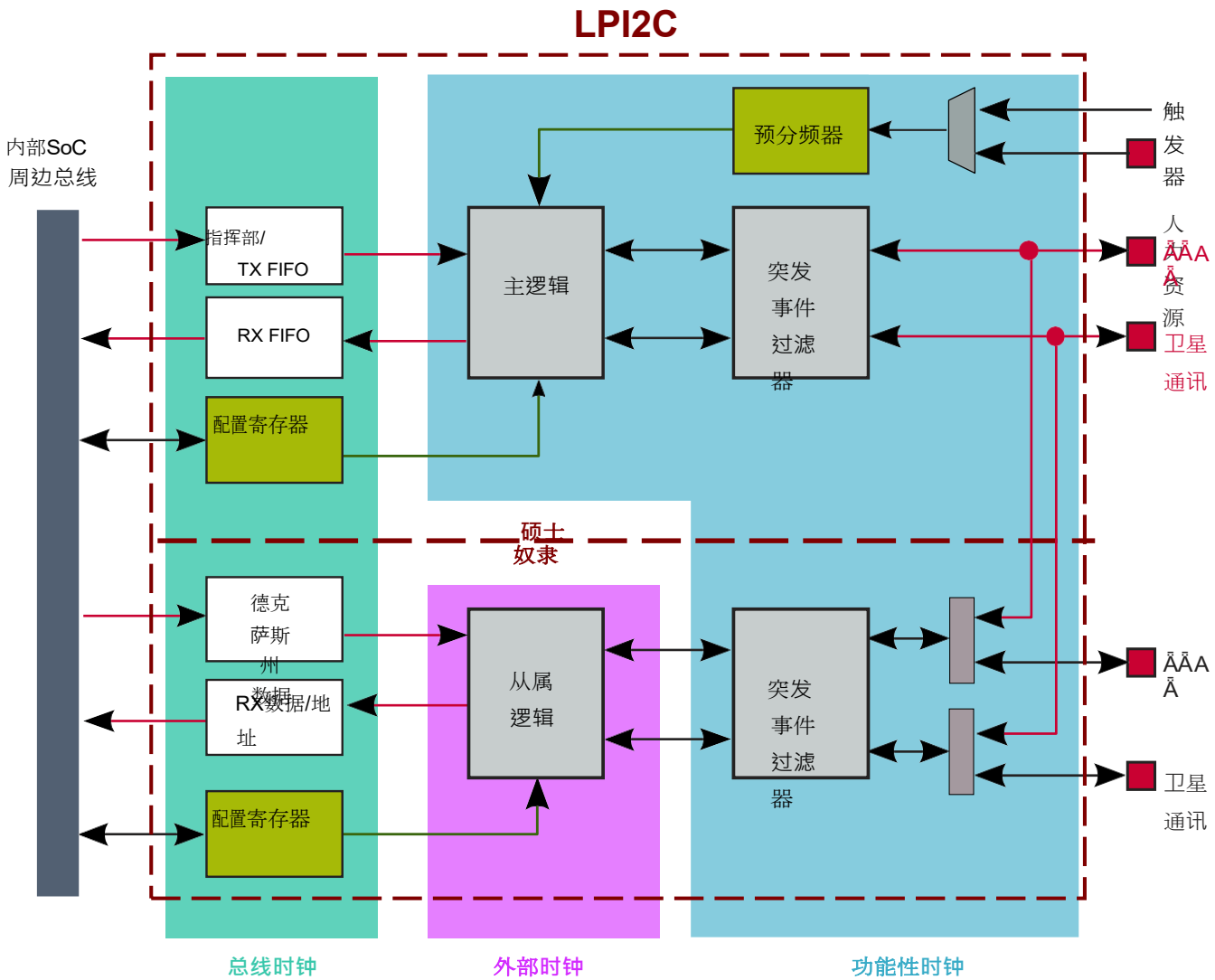



图47-2.LPI2C框图

## 47.2.3 操作模式

表47-2.LPI2C模块支持的芯片模式

| 芯片模式 | LPI2C操作 |
| --- | --- |
| 运行 | 正常运行 |
| 停止 | 如果Doze Enable位（MCR[DOZEN]）被清除，并且LPI2C使用的外部或内部时钟源在停止模式期间保持工作，则可以在停止模式下继续工作。 |
| 调试 | 如果调试启用位（MCR[DBGE]）被设置，可以继续在调试模式下运行。 |

## 47.2.4 信号描述

表47-3.信号

| 信号 | 命名 | 两线制方案 | 四线制方案 | 输入/输出 |
| --- | --- | --- | --- | --- |
| 卫星通讯 | LPI2C时钟线 | 卫星通讯 | 在4线模式下，这是SCL输入引脚。 | 输入/输出 |
| SDA | LPI2C数据线 | SDA | 在4线模式下，这是SDA输入引脚。 | 输入/输出 |
| HREQ | 主机请求 | 如果主机请求被断言，并且I2C总线是空闲的，那么它将启动一个LPI2C主传输。 | | I |
| 证券交易所 | 二级I2C时钟线 | 未使用 | 在4线模式下，这是SCLS输出引脚。如果LPI2C主/从配置为使用单独的引脚，那么这就是LPI2C从属SCL引脚。 | 输入/输出 |
| SDAS | 二级I2C数据线 | 未使用 | 在4线模式下，这是SDAS输出引脚。如果LPI2C主/从配置为使用单独的引脚，那么这就是LPI2C从属SDA引脚。 | 输入/输出 |

## 47.2.5 接线选择

LPI2C可用于实现2线或4线的I2C串行总线。

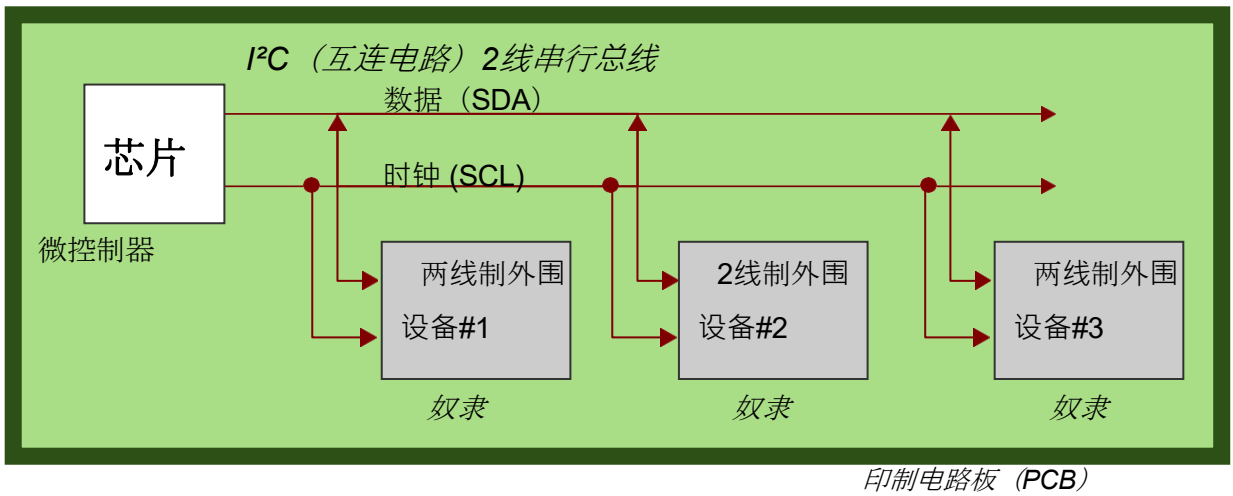



**图47-3.两线制方案**

一些应用可以在I2C总线上提供大量的负载和噪声；为了确保I2C的稳健运行，可以使用与MCU的4线接口，将2条线分成输入和输出。使用一些晶体管、电阻和二极管，设计者可以自己制作廉价的线路驱动器。

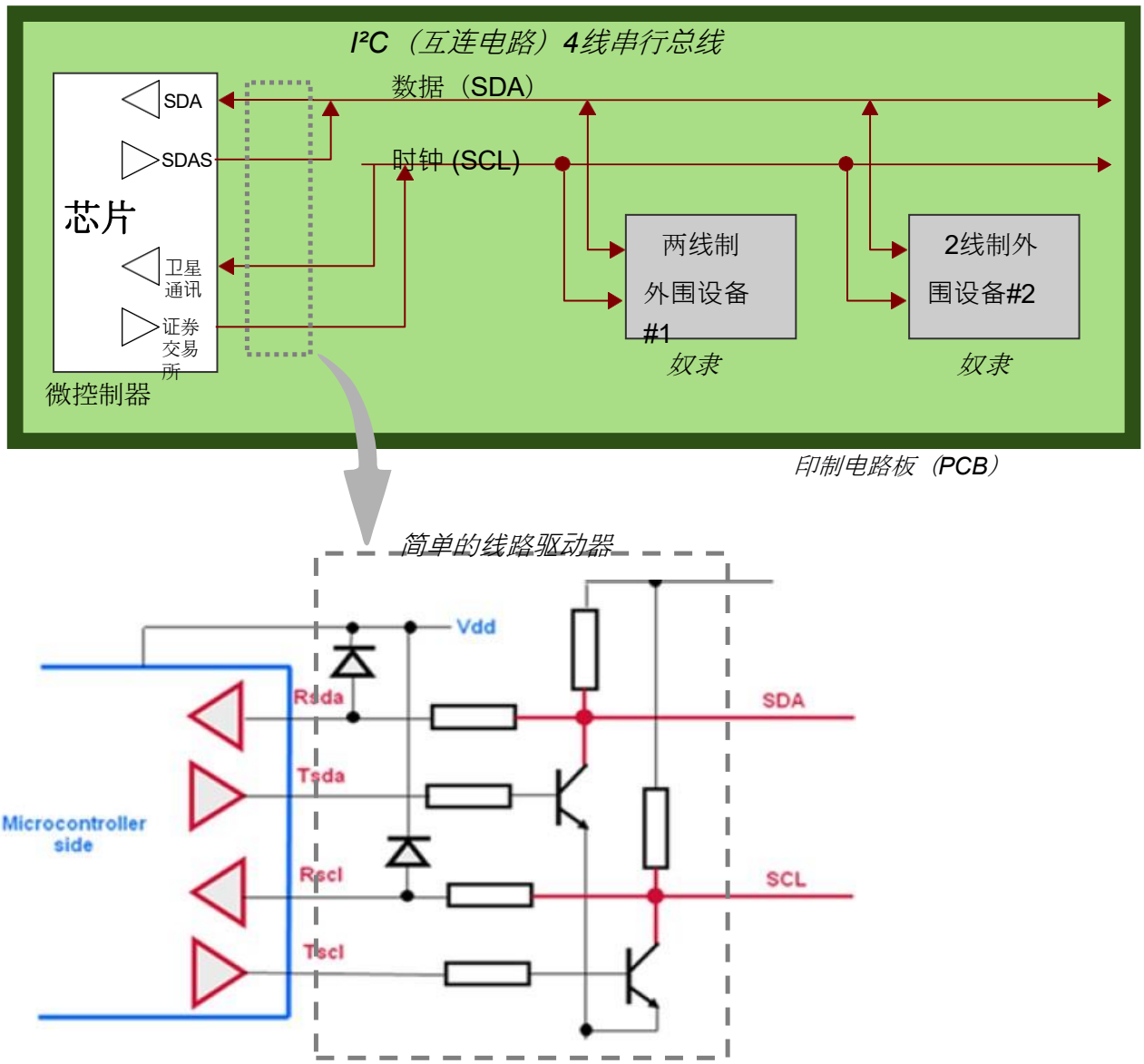



图47-4.4线制方案

## 47.3 存储器地图和寄存器

### 47.3.1 LPI2C寄存器描述

### 47.3.1.1 LPI2C内存地图

LPI2C0基地址。4006_6000h

LPI2C1基础地址。4006_7000h

| 偏移 | 注册 | 宽度<br>(单位：比特) | 访问 | 重置值 |
|---|---|---|---|---|
| 0h | 版本标识寄存器（VERID） | 32 | RO | 0100_0003h |
| 4h | 参数寄存器（PARAM）。 | 32 | RO | 0000_0202h |
| 10h | 主控制寄存器（MCR） | 32 | RW | 0000_0000h |
| 14h | 主状态寄存器（MSR） | 32 | W1C | 0000_0001h |
| 18h | 主中断使能寄存器（MIER） | 32 | RW | 0000_0000h |
| 1Ch | 主DMA启用寄存器（MDER） | 32 | RW | 0000_0000h |
| 20h | 主配置寄存器0（MCFGR0）。 | 32 | RW | 0000_0000h |
| 24h | 主配置寄存器1（MCFGR1）。 | 32 | RW | 0000_0000h |
| 28h | 主配置寄存器2（MCFGR2）。 | 32 | RW | 0000_0000h |
| 2Ch | 主配置寄存器3（MCFGR3）。 | 32 | RW | 0000_0000h |
| 40h | 主数据匹配寄存器（MDMR） | 32 | RW | 0000_0000h |
| 48h | 主时钟配置寄存器0（MCCR0）。 | 32 | RW | 0000_0000h |
| 50h | 主时钟配置寄存器1（MCCR1）。 | 32 | RW | 0000_0000h |
| 58h | 主FIFO控制寄存器（MFCR）。 | 32 | RW | 0000_0000h |
| 5Ch | 主FIFO状态寄存器（MFSR）。 | 32 | RO | 0000_0000h |
| 60h | 主传输数据寄存器（MTDR） | 32 | 汪洋 | 0000_0000h |
| 70h | 主接收数据寄存器（MRDR） | 32 | RO | 0000_4000h |
| 110h | 从属控制寄存器（SCR） | 32 | RW | 0000_0000h |
| 114h | 从属状态寄存器（SSR） | 32 | W1C | 0000_0000h |
| 118h | 从属中断启用寄存器（SIER）。 | 32 | RW | 0000_0000h |
| 11Ch | 从属DMA启用寄存器（SDER）。 | 32 | RW | 0000_0000h |
| 124h | 从属配置寄存器1（SCFGR1）。 | 32 | RW | 0000_0000h |
| 128h | 从属配置寄存器2（SCFGR2）。 | 32 | RW | 0000_0000h |
| 140h | 从属地址匹配寄存器（SAMR） | 32 | RW | 0000_0000h |
| 150h | 从属地址状态寄存器（SASR） | 32 | RO | 0000_4000h |
| 154h | 从机发送ACK寄存器（STAR）。 | 32 | RW | 0000_0000h |
| 160h | 从属传输数据寄存器（STDR） | 32 | 汪洋 | 0000_0000h |
| 170h | 从属接收数据寄存器（SRDR） | 32 | RO | 0000_4000h |

## 47.3.1.2 版本标识寄存器（VERID）

偏移

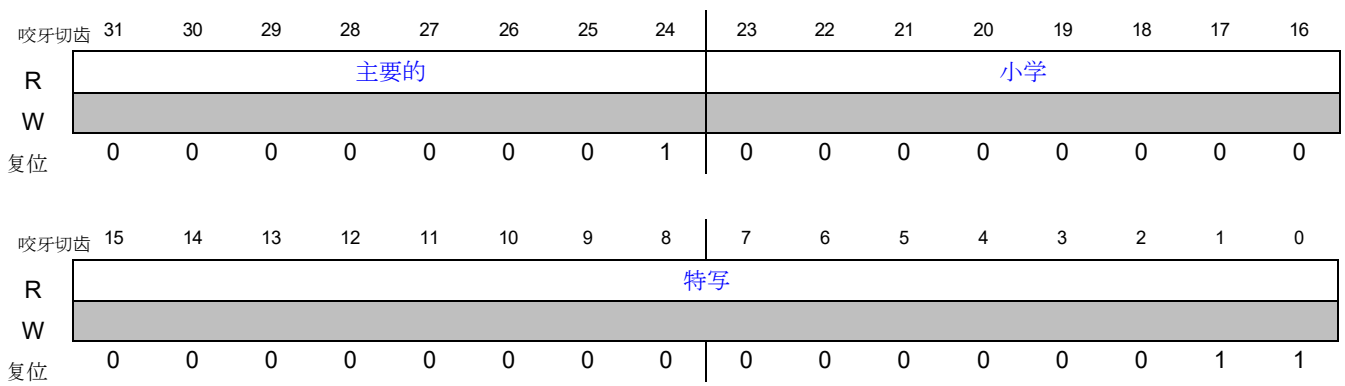
1415.1

| 注册 | 偏移 |
| --- | --- |
| 虚实对照 | 0h |

### 47.3.1.2.2 职能

.

### 47.3.1.2.3 图示




### 47.3.1.2.4 领域

| 场地 | 职能 |
| --- | --- |
| 31-24<br>主要的 | 主要版本号<br>返回模块设计规范的主要版本号。只读字段。 |
| 23-16<br>小学 | 次要版本号<br>返回模块设计规范的次要版本号。只读字段。 |
| 15-0<br>特写 | 特点规格号<br>返回特征集的编号。只读字段。<br>0000000000000010b - 只有主站，有标准功能集<br>0000000000000011b - 主站和从站，有标准功能集 |

## 47.3.1.3 参数寄存器（PARAM）。

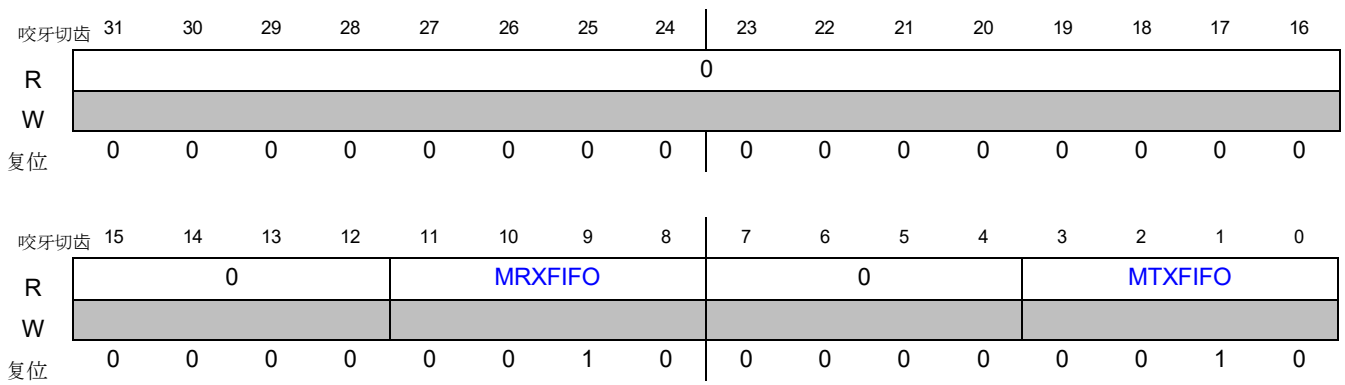
### 47.3.1.3.1 偏移

| 注册 | 偏移 |
|---|---|
| 朴实无华 | 4h |

### 47.3.1.3.2 职能

.

### 47.3.1.3.3 图示




### 47.3.1.3.4 领域

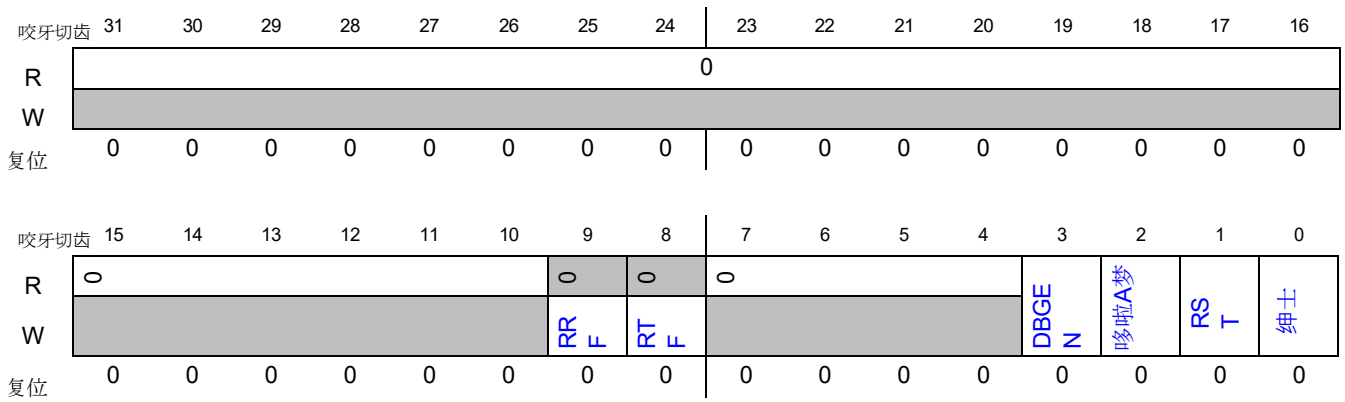
| 场地 | 职能 |
|---|---|
| 31-16<br>- | 保留 |
| 15-12<br>- | 保留 |
| 11-8<br>MRXFIFO | 主站接收FIFO大小<br>将主接收FIFO中的字数配置为2MRXFIFO |
| 7-4<br>- | 保留 |
| 3-0<br>MTXFIFO | 主站发送FIFO大小<br>将主传输FIFO中的字数配置为2MTXFIFO |

## 47.3.1.4 主控制寄存器（MCR）

### 47.3.1.4.1 偏移

| 注册 | 偏移 |
|---|---|
| MCR | 10h |

### 47.3.1.4.2 图示




### 47.3.1.4.3 领域

| 场地 | 职能 |
|---|---|
| 31-10<br>- | 保留 |
| 9<br>RRF | 复位接收FIFO 0b - 无影响<br>1b - 接收的FIFO被重置 |
| 8<br>RTF | 复位发送FIFO 0b - 无影响<br>1b - 发送FIFO被重置 |
| 7-4<br>- | 保留 |
| 3<br>DBGEN | 调试启用<br>0b - 在调试模式下禁用主站 1b - 在调试模式下启用主站 |
| 2<br>哆啦A梦 | 启用休眠模式<br>在静止模式下启用或禁用主站 0b - 在静止模式下启用主站 1b - 在静止模式下禁用主站 |

| | |
|---|---|
| 1<br>RST | 软件复位<br><br>重置所有内部主控逻辑和寄存器，除了主控寄存器。RST保持设置，直到被软件清除。<br>0b - 主逻辑不复位 1b - 主逻辑被复位 |

*表继续在下一页...*

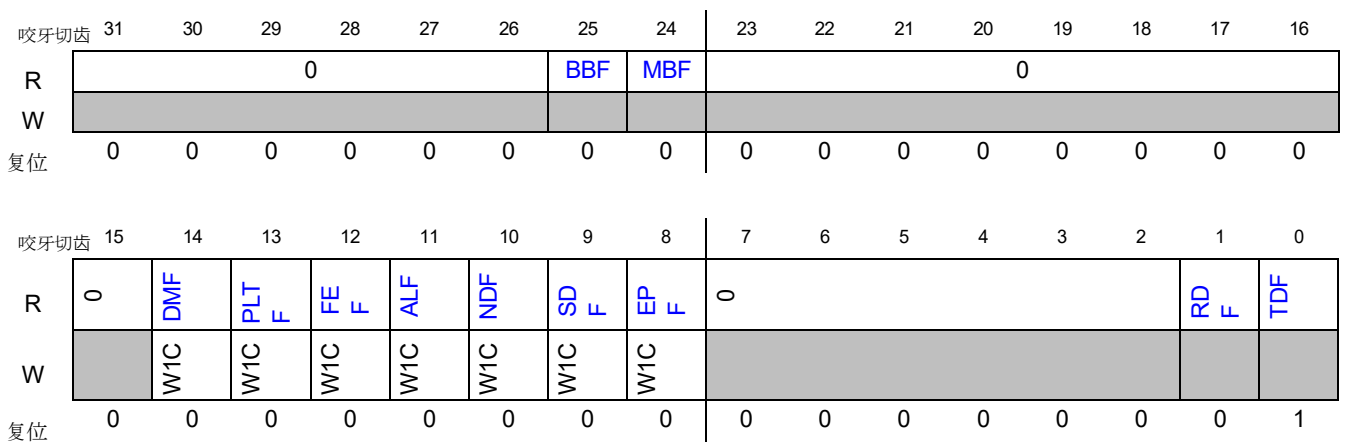
| 场地 | 职能 |
|---|---|
| 0<br>绅士 | 主站启用<br>0b - 主逻辑被禁用 1b - 主逻辑被启用 |

## 47.3.1.5 主状态寄存器（MSR）

### 47.3.1.5.1 偏移

| 注册 | 偏移 |
|---|---|
| ÃÃÃ | 14h |

### 47.3.1.5.2 图示




### 47.3.1.5.3 领域

| 场地 | 职能 |
|---|---|
| 31-26<br>- | 保留 |
| 25<br>BBF | 公交车的标志<br>0b - I2C总线处于空闲状态 1b - I2C总线处于繁忙状态 |

| | |
|---|---|
| 24<br>MBF | 主站繁忙标志<br>0b - I2C主站处于空闲<br>状态 1b - I2C主站处于<br>繁忙状态 |
| 23-15 | 保留 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| - | |
| 14<br>DMF | 数据匹配标志<br>表示收到的数据与MATCH0和/或MATCH1字段相匹配（由MCFGR1[MATCFG]，匹配配置配置）。*由于CMD字段而被丢弃的*接收数据不会导致数据匹配标志被设置。<br>0b - 未收到匹配数据 1b - 已收到匹配数据 |
| 13<br>PLTF | 引脚低电平超时标志<br>当SCL和/或SDA输入低电平超过PINLOW周期时（Pin Low Timeout，MCFGR3[PINLOW]）将被设置，即使LPI2C主站处于空闲状态。<br>• 软件负责解决引脚低电平的问题。<br>• 只要引脚低电平超时持续，就不能清除引脚低电平超时标志。<br>• 在LPI2C启动一个START条件之前，必须清除引脚低电平超时标志。<br>0b - 引脚低电平超时未发生或被禁用 1b - 引脚低电平超时已发生 |
| 12<br>FEF | FIFO错误标志<br>检测试图发送或接收数据而没有首先产生一个（重复）START条件。当AUTOSTOP位被设置时，如果发送FIFO溢出，就会发生这种情况。当FIFO错误标志被设置时，LPI2C主站将发送一个STOP条件（如果忙的话），并且在FIFO错误标志被清除之前不会启动一个新的START条件。<br>0b - 没有错误<br>1b - 主站在没有START条件下发送或接收数据 |
| 11<br>ALF | 仲裁丢失的旗帜<br>镶嵌。<br>• 如果LPI2C主站发送一个逻辑一，并在I2C总线上检测到一个逻辑零的话<br>• 或者当LPI2C主控器正在传输数据时，检测到一个START或STOP条件。<br>当仲裁丢失标志设置时，LPI2C主站将释放I2C总线（进入空闲状态），在仲裁丢失标志被清除之前，LPI2C主站不会启动一个新的START条件。<br>0b - 主站没有失去仲裁 1b - 主站已经失去仲裁 |
| 10<br>NDF | NACK检测标志<br>如果LPI2C主站在传输地址或数据时检测到一个NACK，则设置该标志。当设置时，主站将发送一个STOP条件，并且在NACK检测旗被清除之前不会启动一个新的START条件。如果对一个给定的地址预期有一个NACK（由命令字配置），那么如果没有产生NACK，NACK检测旗将被设置。<br>0b - 没有检测到意外的NACK 1b - 检测到意外的NACK |
| 9<br>SDF | STOP检测标志<br>当LPI2C主站产生一个STOP条件时，设置。<br>0b - 主站未产生STOP条件 1b - 主站已产生STOP条件 |

| | |
|---|---|
| 8<br>EPF | 结束数据包标志<br>当LPI2C主站产生一个重复的START条件或STOP条件时，设置。当主站第一次产生START条件时不设置。<br>0b - 主站没有产生STOP或重复START条件 1b - 主站产生STOP或重复START条件 |
| 7-2 | 保留 |

*表继续在下一页...*

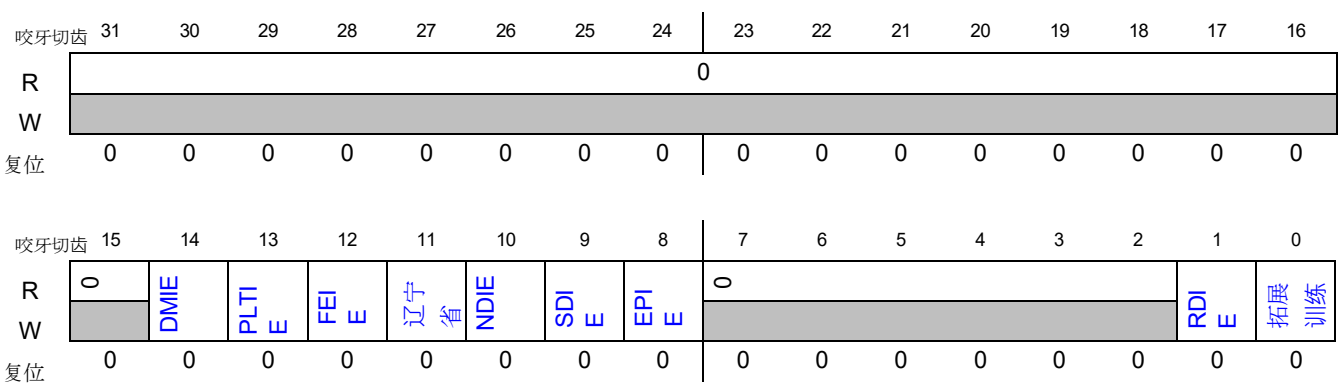
| 场地 | 职能 |
|---|---|
| - | |
| 1<br>RDF | 接收数据标志<br>只要接收FIFO中的字数大于RXWATER，接收数据标志就被设置。<br>0b - 接收数据未准备好 1b - 接<br>收数据已准备好 |
| 0<br>TDF | 发送数据标志<br>每当发送FIFO中的字数等于或少于TXWATER时，发送数据标志就被设置。<br>0b - 不要求传输数据 1b - 要求传输<br>数据 |

## 47.3.1.6 主中断使能寄存器（MIER）

### 47.3.1.6.1 偏移

| 注册 | 偏移 |
|---|---|
| MIER | 18h |

### 47.3.1.6.2 图示




### 47.3.1.6.3 领域

| 场地 | 职能 |
|---|---|
| 31-15<br>- | 保留 |

| | |
|---|---|
| 14 | 数据匹配中断启用 |

*表继续在下一页...*

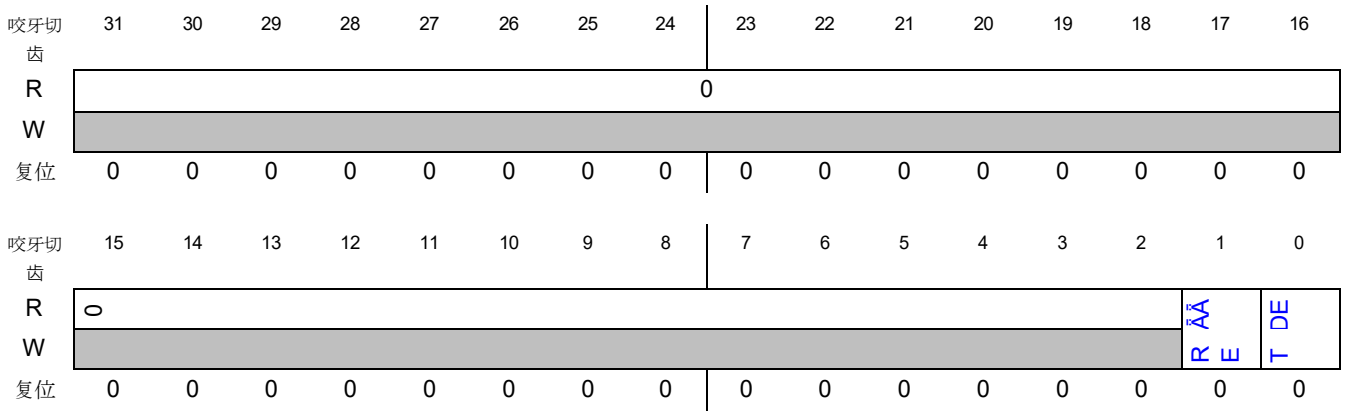
| 场地 | 职能 |
| --- | --- |
| DMIE | 0b - 禁用 1b - 启用 |
| 13 PLTIE | 引脚低电平超时中断启用 0b - 禁用 1b - 已启用 |
| 12 基金会 | FIFO错误中断启用 0b - 已启用 1b - 禁用 |
| 11 辽宁省 | 仲裁丢失中断启用 0b - 禁用 1b - 已启用 |
| 10 NDIE | NACK检测中断启用 0b - 禁用 1b - 已启用 |
| 9 SDIE | STOP检测中断启用 0b - 禁用 1b - 已启用 |
| 8 EPIE | 结束包中断启用 0b - 禁用 1b - 已启用 |
| 7-2 - | 保留 |
| 1 种族歧视委员会(RDIE) | 接收数据中断启用 0b - 禁用 1b - 已启用 |
| 0 拓展训练(TDIE) | 发送数据中断启用 0b - 禁用 1b - 已启用 |

## 47.3.1.7 主DMA启用寄存器（MDER）

### 47.3.1.7.1 偏移

| 注册 | 偏移 |
| --- | --- |
| MDER | 1Ch |

### 47.3.1.7.2 图示




### 47.3.1.7.3 领域

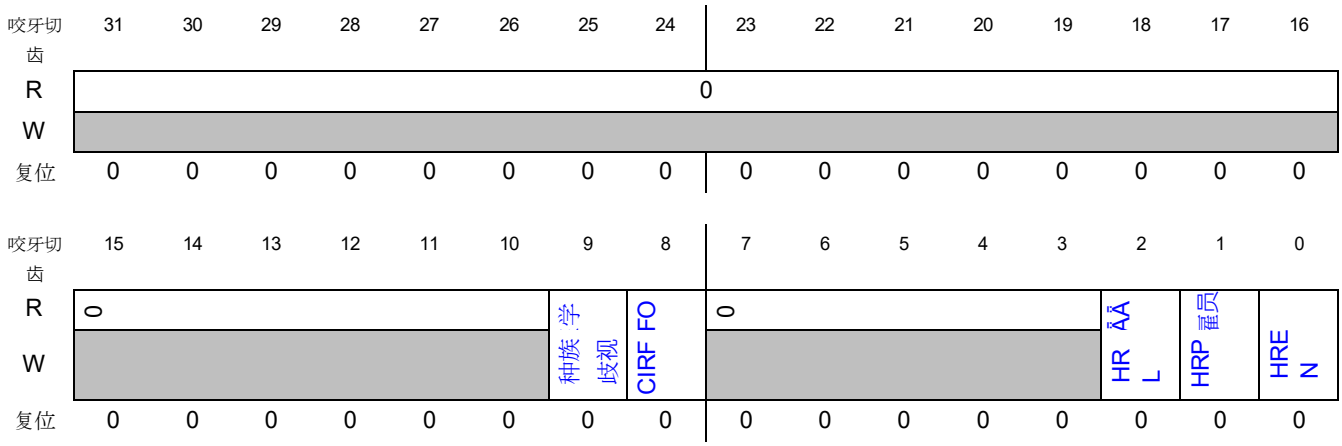
| 场地 | 职能 |
|---|---|
| 31-2<br>- | 保留 |
| 1<br>RDDE | 接收数据 DMA启用<br>0b - DMA请求被禁用 1b - DMA请求被启用 |
| 0<br>TDDE | 发送数据DMA启用<br>0b - DMA请求被禁用 1b - DMA请求被启用 |

## 47.3.1.8 主配置寄存器0（MCFGR0）。

### 47.3.1.8.1 偏移

| 注册 | 偏移 |
|---|---|
| MCFGR0 | 20h |

### 47.3.1.8.2 图示




### 47.3.1.8.3 领域

| 场地 | 职能 |
|---|---|
| 31-10<br>- | 保留 |
| 9<br>RDMO | 仅接收数据匹配<br>当启用时，所有没有导致数据匹配标志（MSR[DMF]）设置的接收数据都被丢弃。在数据匹配标志被设置后，RDMO配置被忽略。当禁用RDMO时，在清除数据匹配标志之前清除RDMO，以确保没有接收数据被丢失。<br>0b - 接收的数据被存储在接收FIFO中<br>1b - 接收的数据被丢弃，除非数据匹配标志（MSR[DMF]）被设置。 |
| 8<br>CIRFIFO | 循环FIFO启用<br>当启用时，发送FIFO的读指针被保存到一个临时寄存器。发送FIFO将被正常清空，但在LPI2C主控器空闲且发送FIFO为空后，则将从临时寄存器恢复读指针值。这将导致发送FIFO的内容被反复循环使用。如果设置了AUTOSTOP，那么每当发送FIFO为空并且读指针被恢复时，就会发送一个STOP条件。<br>0b - 循环 FIFO 被禁用 1b - 循环 FIFO 被启用 |
| 7-3<br>- | 保留 |
| 2<br>HRSEL | 主机请求选择<br>选择主机请求输入的来源。当主机请求输入被启用时，主机请求选择字段应保持静态（主机请求选择不应改变）。<br>0b - 主机请求输入是引脚HREQ 1b - 主机请求输入是输入触发器 |

| | |
|---|---|
| 1<br>HRPOL | 主机请求极性<br>配置主机请求输入引脚的极性。当主机请求输入被启用时，主机请求极性字段应保持静态（主机请求极性不应改变）。<br>0b - 低电平有效<br>1b - 高电平有效 |
| 0<br>HREN | 主机请求启用 |

| 场地 | 职能 |
|---|---|
| | 当启用时，LPI2C主控器只有在主机请求输入被断言且总线空闲时才会启动一个START条件。重复的START不受主机请求的影响。<br>0b - 主机请求输入被禁用 1b - 主机请求输入被启用 |

## 47.3.1.9 主配置寄存器1（MCFGR1）。

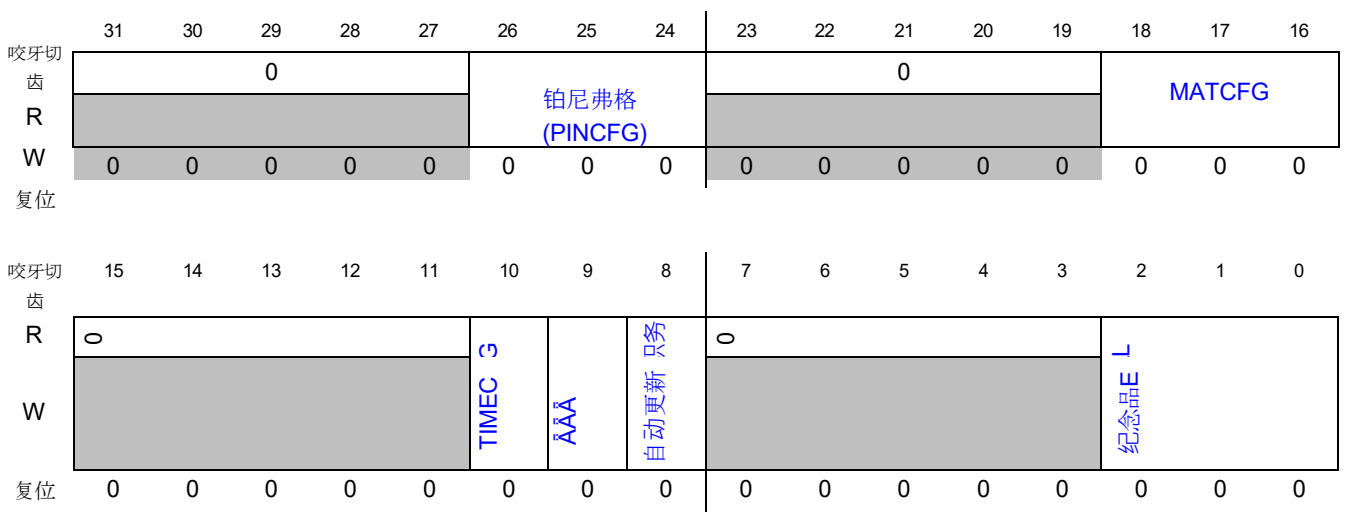
### 47.3.1.9.1 偏移

| 注册 | 偏移 |
|---|---|
| 医学研究中心(MCFGR1) | 24h |

### 47.3.1.9.2 职能

MCFGR1只能在I2C主站被禁用时被写入。

### 47.3.1.9.3 图示




### 47.3.1.9.4 领域

| 场地 | 职能 |
|---|---|

*表继续在下一页...*

| | |
|---|---|
| 31-27<br>- | 保留 |
| 26-24<br>铂尼弗格<br>(PINCFG) | 引脚配置<br>配置LPI2C的引脚模式。 |

| 场地 | 职能 |
| --- | --- |
| | 000b - 2针开漏模式<br>001b - 2针输出模式（超快速模式） 010b - 2针<br>推挽模式<br>011b - 4针推挽模式<br>100b - 带有独立LPI2C从机的2针开漏模式<br>101b - 带有独立LPI2C从机的2针输出模式（超快模式） 110b - 带有独立<br>LPI2C从机的2针推挽模式<br>111b - 4针推挽模式（反转输出）。 |
| 23-19<br>- | 保留 |
| 18-16<br>MATCFG | 匹配配置<br>配置将导致DMF设置的条件。<br>000b - 匹配被禁用 001b -<br>保留<br>010b - 匹配已启用（第一个数据字等于MATCH0或MATCH1） 011b - 匹<br>配已启用（任何数据字等于MATCH0或MATCH1）。<br>100b - 匹配已启用（第一个数据字等于MATCH0，第二个数据字等于MATCH1） 101b - 匹配已启用（任何数据字等于MATCH0，下一个数据字等于MATCH1） 110b - 匹配已启用（第一个数据字和MATCH1等于MATCH0和MATCH1）。<br>111b - 匹配已启用（任何数据字和MATCH1等于MATCH0和MATCH1）。 |
| 15-11<br>- | 保留 |
| 10<br>ÃÃTIMECFG | 超时配置<br>0b - 如果SCL的低电平超过配置的超时时间，引脚低电平超时标志将被设置。<br>1b - 如果SCL或SDA的低电平超过配置的超时时间，引脚低电平超时标志将被设置。 |
| 9<br>ÃÃÃ | ÃÃÃ<br>当设置时，收到的NACK字段被忽略，并假定为ACK。IGNACK位在超快模式下需要被设置。<br>0b - LPI2C主站将正常接收ACK和NACK。<br>1b - LPI2C主站将把收到的NACK当作（NACK）是一个ACK。 |
| 8<br>自动停止 | 自动生成STOP<br>启用后，每当LPI2C主控器繁忙且发送FIFO为空时，就会产生一个STOP条件。STOP条件也可以通过一个发送FIFO命令来产生。<br>0b - 没有影响<br>1b - 每当发送FIFO为空且LPI2C主站繁忙时，自动产生STOP条件。 |
| 7-3<br>- | 保留 |

*表继续在下一页...*

| | |
|---|---|
| 2-0<br><br>预售 | 预分频器<br><br>配置用于所有LPI2C主控逻辑的时钟预分频器，但数字突波滤波器除外。<br>000b - 除以1 001b - 除以2 010b - 除以4 011b - 除以8 100b - 除以16 101b - 除以32 110b - 除以64 111b - 除以128 |

## 47.3.1.10 主配置寄存器2（MCFGR2）。

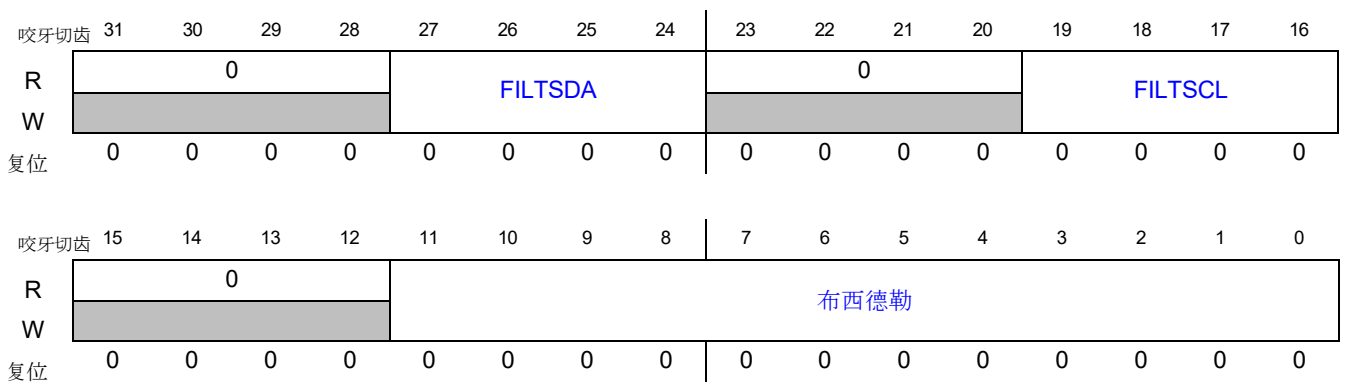
### 47.3.1.10.1 偏移

| 注册 | 偏移 |
|---|---|
| 医学博士（MCFGR2 | 28h |

### 47.3.1.10.2 职能

主配置寄存器2只能在I2C主控器被禁用时被写入。

### 47.3.1.10.3 图示




### 47.3.1.10.4 领域

| 场地 | 职能 |
|---|---|
| 31-28<br>- | 保留 |
| 27-24<br>FILTSDA | 突波滤波器SDA<br>配置SDA输入的I2C主控数字突波滤波器。<br>• 配置为0将禁用突波滤波器。<br>• 等于或小于FILTSDA周期的斑点将被过滤掉，并被忽略。<br>• 通过突波滤波器的延迟等于FILTSDA周期，并且必须配置为小于最小SCL低或高周期<br>• 误码过滤周期数不受PRESCALE配置的影响，在高速模式下，误码过滤周期数被自动绕过 |

*表继续在下一页...*

| | |
|---|---|
| 23-20<br>- | 保留 |

| 场地 | 职能 |
|---|---|
| 19-16<br>FILTSCL | 突波滤波器SCL<br>配置SCL输入的I2C主站数字突波滤波器。<br>• 配置为0将禁用突波滤波器。<br>• 等于或小于FILTSCL周期的斑点将被过滤掉，并被忽略。<br>• 通过突波滤波器的延迟等于FILTSCL周期，并且必须配置为小于最小SCL低或高周期。<br>• 误码过滤周期数不受PRESCALE配置的影响，在高速模式下，误码过滤周期数被自动绕过 |
| 15-12<br>- | 保留 |
| 11-0<br>布西德勒 | 总线空闲超时<br>配置以时钟周期为单位的总线空闲超时周期。<br>• 如果SCL和SDA的高电平都超过BUSIDLE周期，那么I2C总线被认为是空闲的，主站可以产生一个START条件。<br>• 当总线空闲超时被设置为零时，总线空闲超时被禁用。 |

## 47.3.1.11 主配置寄存器3（MCFGR3）。

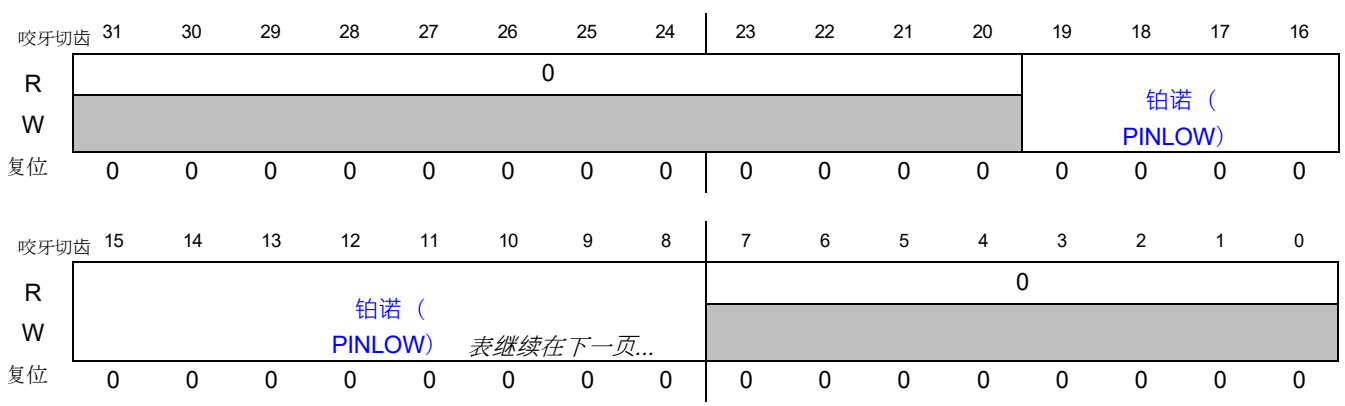
### 47.3.1.11.1 偏移

| 注册 | 偏移 |
|---|---|
| 辽宁沈阳 | 2Ch |

### 47.3.1.11.2 职能

MCFGR3只能在I2C主站被禁用时写入。

### 47.3.1.11.3 图示



### 47.3.1. 1428.4 领域

| 场地 | 职能 |
|---|---|
| 31-20<br>- | 保留 |
| 19-8<br>铂诺（PINLOW）） | 引脚低电平超时<br>以时钟周期配置引脚低电平超时标志。<br>• 如果SCL或SCL或SDA的低电平超过(PINLOW * 256)周期，则PLTF被设置。<br>• 当引脚低电平超时被设置为零时，引脚低电平超时功能被禁用。 |
| 7-0<br>- | 保留 |

## 47.3.1.12 主数据匹配寄存器（MDMR）

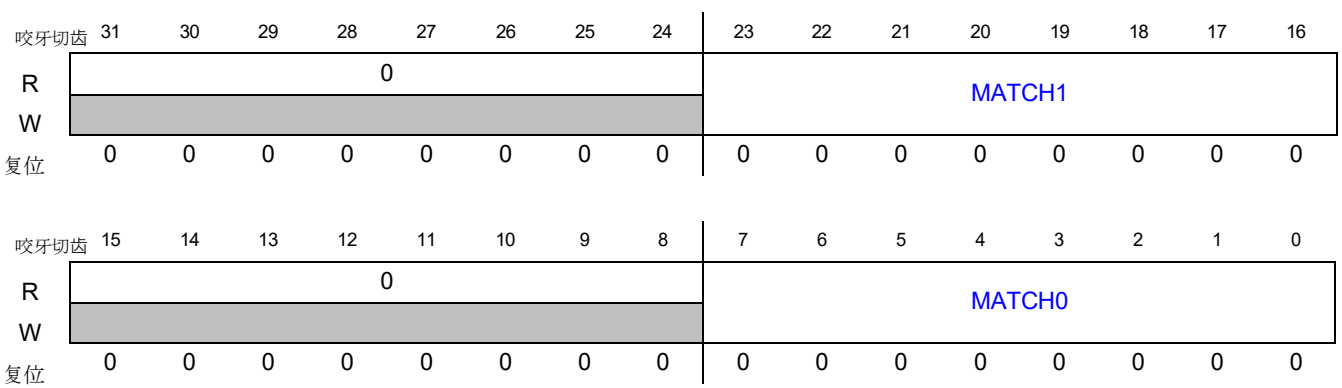
### 47.3.1.12.1 偏移

| 注册 | 偏移 |
|---|---|
| MDMR | 40h |

### 47.3.1.12.2 职能

MDMR只能在I2C主站被禁用或空闲时被写入。

### 47.3.1.12.3 图示



#### 47.3.1.12.4 领域

| 场地 | 职能 |
|---|---|
| 31-24<br>- | 保留 |
| 23-16<br>MATCH1 | 匹配1的价值<br>当接收数据匹配被启用时，与接收的数据进行比较。 |
| 15-8<br>- | 保留 |
| 7-0<br>MATCH0 | 匹配0值<br>当接收数据匹配被启用时，与接收的数据进行比较。 |

## 47.3.1.13 主时钟配置寄存器0（MCCR0）。

### 47.3.1.13.1 偏移

| 注册 | 偏移 |
|---|---|
| MCCR0 | 48h |

### 47.3.1.13.2 职能

当I2C主站被启用时，MCCR0不能被改变，它用于标准、快速、快速模式加和超快速的传输。

### 47.3.1.13.3 图示



### 47.3.1.13.4 领域

| 场地 | 职能 |
|---|---|
| 31-30<br>- | 保留 |
| 29-24<br>ディストフォメーション | 数据有效延时<br>用作SDA的数据保持时间的最小周期数（减去1）。必须配置为小于最小SCL低电平周期。 |
| 23-22<br>- | 保留 |
| 21-16<br>阈值 | 设置保持延时<br>主体使用的最小周期数（减去1）。<br>• 作为START条件的保持时间<br>• 作为重复START条件的设置和保持时间<br>• 作为STOP条件的设置时间<br>设置时间由检测外部SCL引脚的上升沿所需的时间延长。忽略由于外部负载造成的任何额外的电路板延迟，这个时间等于（2 + FILTSCL）/ 2^PRESCALE周期。 |
| 15-14<br>- | 保留 |
| 13-8<br>辽宁省 | 时钟高电平周期<br>SCL时钟被主控器驱动为高电平的最小周期数（减去1）。SCL高电平时间由检测外部SCL引脚上升沿的时间延长。忽略由于外部负载造成的任何额外的电路板延迟，这个时间等于（2 + FILTSCL）/ 2^PRESCALE周期。 |
| 7-6<br>- | 保留 |
| 5-0<br>辽宁省 | 时钟低电平周期<br>SCL时钟被主站驱动为低电平的最小周期数（减去1）。时钟低电平周期值也用于STOP和START条件之间的最小总线空闲时间；这是由检测外部SCL引脚上升沿的时间延长的。忽略由于外部负载造成的任何额外的电路板延迟，这个时间等于（2 + FILTSCL）/ 2^PRESCALE周期。 |

## 47.3.1.14 主时钟配置寄存器1（MCCR1）。

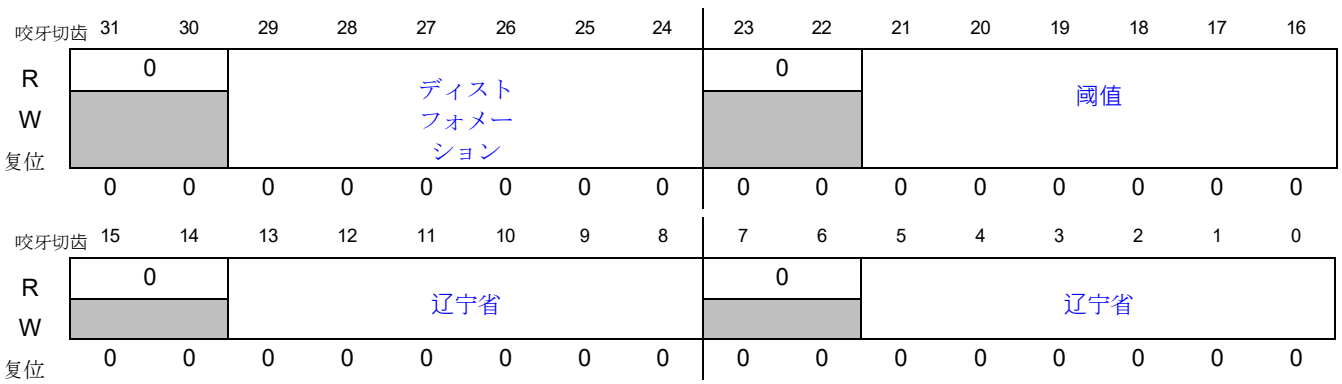
### 47.3.1.14.1 偏移

| 注册 | 偏移 |
|---|---|
| MCCR1 | 50h |

### 47.3.1.14.2 职能

当I2C主控器被启用并用于高速模式传输时，MCCR1不能被改变。高速模式的单独时钟配置允许在快速模式下进行仲裁（由MCCR0配置时序），然后再切换到高速模式（由MCCR1配置时序）。

### 47.3.1.14.3 图示




### 47.3.1.14.4 领域

| 场地 | 职能 |
|---|---|
| 31-30<br>- | 保留 |
| 29-24<br>ディストフォメーション | 数据有效延时<br>用作SDA的数据保持时间的最小周期数（减去1）。数据有效延迟必须被配置为小于最小SCL低电平周期。 |
| 23-22<br>- | 保留 |
| 21-16<br>阈值 | 设置保持延时<br>主站使用的最小周期数（减去1）。<br>• 作为START条件的保持时间<br>• 作为重复START条件的设置和保持时间<br>• 作为STOP条件的设置时间<br>设置时间由检测外部SCL引脚的上升沿所需的时间延长。忽略由于外部负载造成的任何额外的电路板延迟，这个时间等于（2 + FILTSCL）/ 2^PRESCALE周期。 |
| 15-14<br>- | 保留 |

*表继续在下一页...*

| | |
|---|---|
| 13-8<br><br>辽宁省 | 时钟高电平周期<br><br>SCL时钟被主控器驱动为高电平的最小周期数（减去1）。SCL高电平时间由检测外部SCL引脚上升沿的时间延长。忽略由于外部负载造成的任何额外的电路板延迟，这个时间等于（2 + FILTSCL）/ 2^PRESCALE周期。 |

| 场地 | 职能 |
|---|---|
| 7-6<br>- | 保留 |
| 5-0<br>辽宁省 | 时钟低电平周期<br>SCL时钟被主站驱动为低电平的最小周期数（减去1）。时钟低电平周期值也用于STOP和START条件之间的最小总线空闲时间；这是由检测外部SCL引脚上升沿的时间延长的。忽略由于外部负载造成的任何额外的电路板延迟，这个时间等于（2 + FILTSCL）/ 2^PRESCALE周期。 |

## 47.3.1.15 主FIFO控制寄存器（MFCR）。

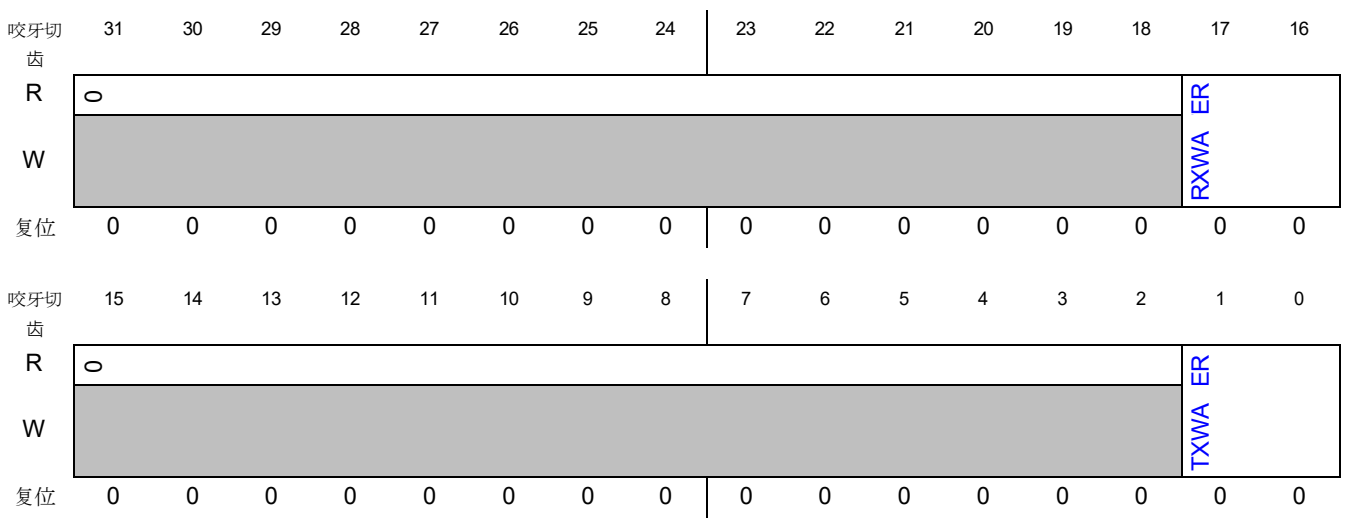
### 47.3.1.15.1 偏移

| 注册 | 偏移 |
|---|---|
| MFCR | 58h |

### 47.3.1.15.2 职能

当MFCR寄存器是静态的（即MFCR寄存器没有变化），主FIFO控制寄存器只在停止模式下使用。

### 47.3.1.15.3 图示




*表继续在下一页...*

#### 47.3.1.15.4 领域

| 场地 | 职能 |
|---|---|
| 31-18<br>- | 保留 |
| 17-16<br>RXWATER | 接收FIFO水印<br>只要接收FIFO中的字数大于RXWATER，就会设置接收数据标志。写入一个等于或大于FIFO大小的值将被截断。 |
| 15-2<br>- | 保留 |
| 1-0<br>淡水河谷（<br>TXWATER | 发送FIFO水印<br>只要发送FIFO中的字数等于或小于TXWATER，就会设置发送数据标志。写入一个等于或大于FIFO大小的值将被截断。 |

## 47.3.1.16 主FIFO状态寄存器（MFSR）

### 47.3.1.16.1 偏移

| 注册 | 偏移 |
|---|---|
| 医学研究中心(MFSR) | 5Ch |

### 47.3.1.16.2 图示

| 咬牙切齿 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | 荣事达 | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 咬牙切齿 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | TXCOUNT | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

### 47.3.1.16.3 领域

| 场地 | 职能 |
|---|---|
| 31-19 | 保留 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| - | |
| 18-16<br>荣事达 | 接收FIFO计数<br>返回接收FIFO中的字数。 |
| 15-3<br>- | 保留 |
| 2-0<br>TXCOUNT | 发送FIFO计数<br>返回发送FIFO中的字数。 |

## 47.3.1.17 主传输数据寄存器（MTDR）

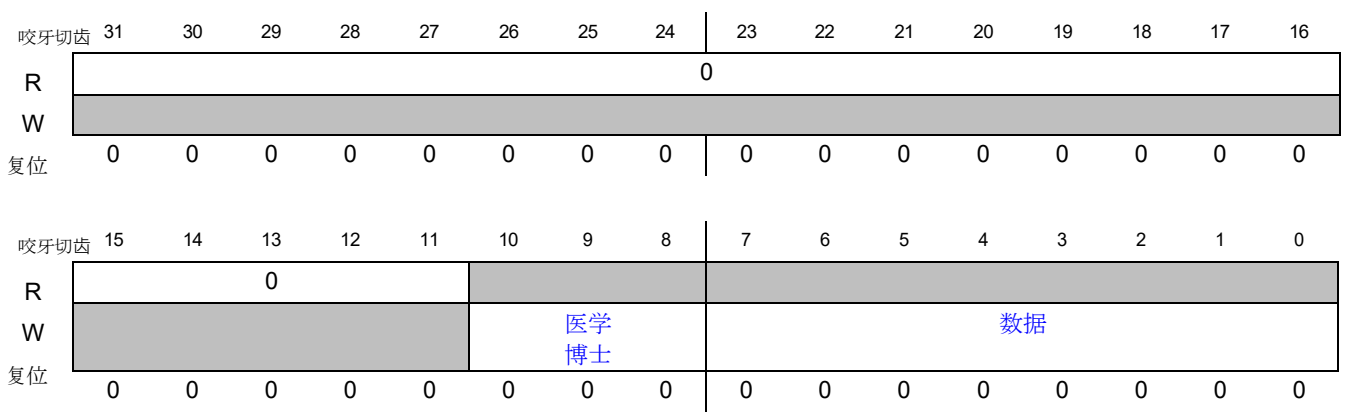
### 47.3.1.17.1 偏移

| 注册 | 偏移 |
|---|---|
| MTDR | 60h |

### 47.3.1.17.2 职能

- 对CMD字段的8位写入将在命令FIFO中存储数据，但不增加FIFO写入指针。
- 对DATA字段的8位写入将使CMD字段扩展为零，除非CMD字段在上次FIFO写入后被单独写入；它（8位写入）也使FIFO写入指针增加。
- 16位或32位将同时写入CMD和DATA字段并递增FIFO。

### 47.3.1.17.3 图示



### 47.3.1.17.4 领域

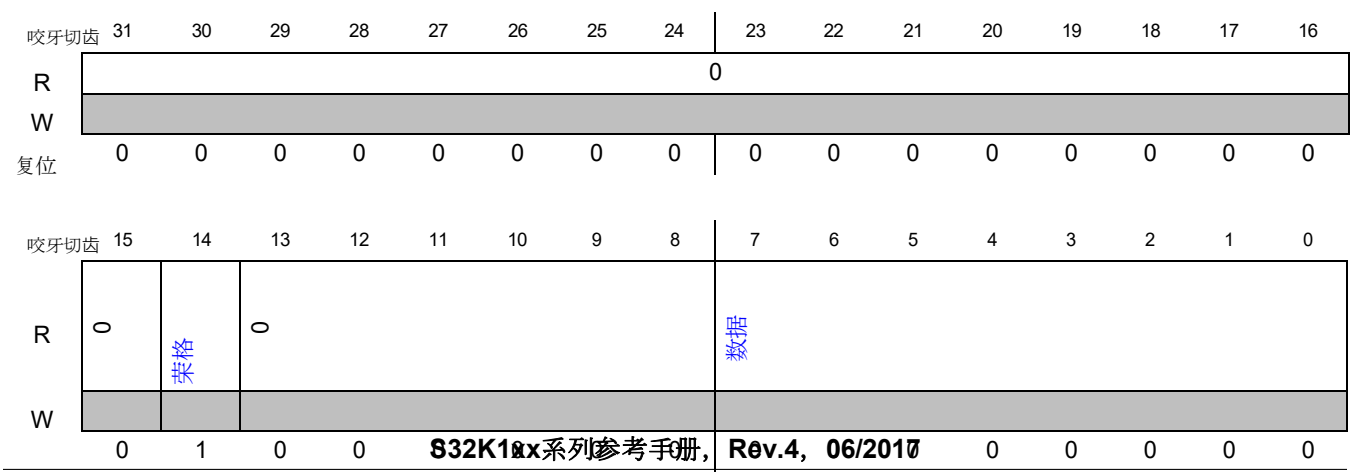
| 场地 | 职能 |
|---|---|
| 31-11<br>- | 保留 |
| 10-8<br>医学博士 | 命令数据<br>000b - 发送数据[7:0]<br>001b - 接收（DATA[7:0]+1）字节 010b - 产生STOP条件<br>011b - 接收并丢弃（DATA[7:0] + 1）个字节<br>100b - 在DATA[7:0]中产生（重复）START和发送地址<br>101b - 在DATA[7:0]中产生（重复）START和传输地址。该传输期望返回一个NACK。<br>110b - 使用高速模式在 DATA[7:0]中产生（重复）START 和发送地址 111b - 使用高速模式在DATA[7:0]中产生（重复）START 和发送地址。该传输期望返回一个NACK。 |
| 7-0<br>数据 | 发送数据<br>对DATA进行8位写操作将使CMD字段扩展为零。 |

## 47.3.1.18 主接收数据寄存器（MRDR）

### 47.3.1.18.1 偏移

| 注册 | 偏移 |
|---|---|
| MRDR | 70h |

### 47.3.1.18.2 图示



#### 47.3.1.18.3 领域

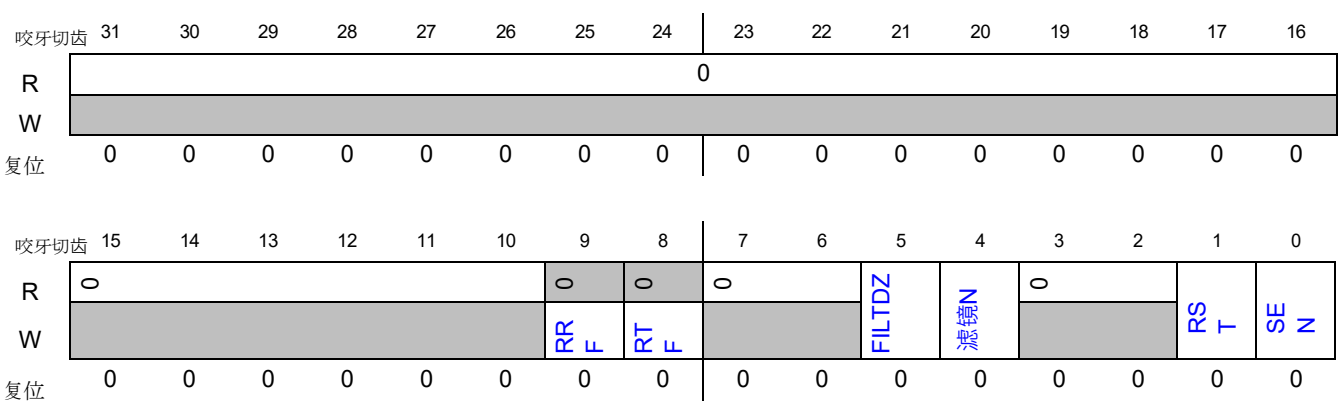
| 场地 | 职能 |
|---|---|
| 31-15<br>- | 保留 |
| 14<br>荣格 | RX空<br>0b - 接收FIFO不为空 1b - 接收FIFO为空 |
| 13-8<br>- | 保留 |
| 7-0<br>数据 | 接收数据<br>读取接收数据寄存器返回I2C主站收到的未被丢弃的数据。接收数据可以由于CMD字段而被丢弃，或者主站可以被配置为丢弃非匹配数据。 |

## 47.3.1.19 从属控制寄存器（SCR）

### 47.3.1.19.1 偏移

| 注册 | 偏移 |
|---|---|
| SCR | 110h |

### 47.3.1.19.2 图示



### 47.3.1.19.3 领域

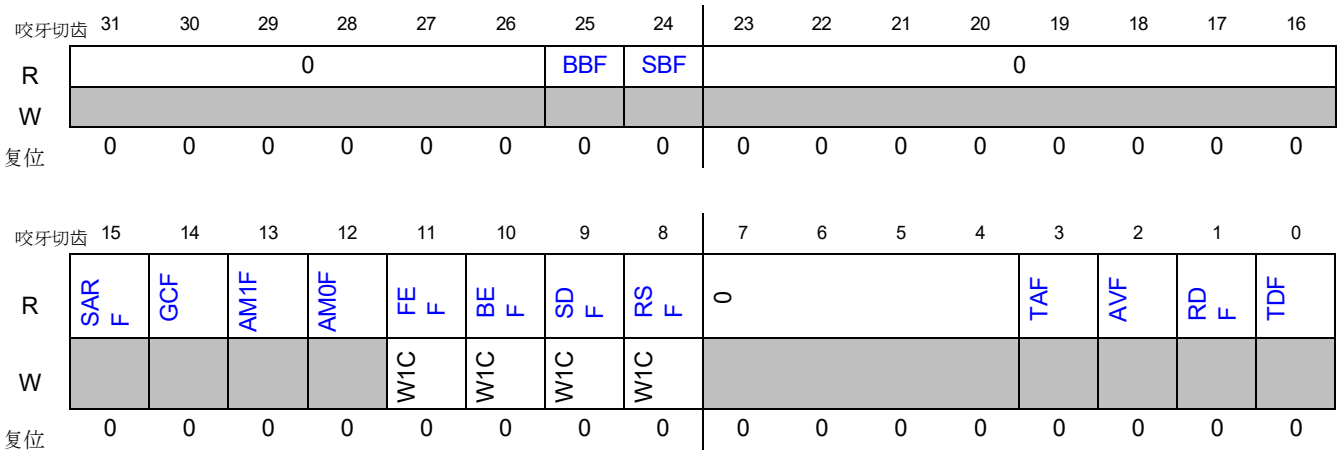
| 场地 | 职能 |
|---|---|
| 31-10<br>- | 保留 |
| 9<br>RRF | 复位接收FIFO 0b - 无<br>影响<br>1b - 接收数据寄存器现在是空的 |
| 8<br>RTF | 复位发送FIFO 0b - 无<br>影响<br>1b - 发送数据寄存器现在是空的 |
| 7-6<br>- | 保留 |
| 5<br>FILTDZ | 启用滤波休眠<br>只有在I2C从机被禁用的情况下，才应更新滤波休眠启用位。<br>0b - 在休眠模式下，过滤器保持启用状态<br>1b - 在休眠模式下，过滤器被禁用 |
| 4<br>菲尔腾 | 启用过滤器<br>滤波器启用位只有在I2C从机被禁用时才应更新。0b - 禁用数字滤波器和<br>输出延迟计数器的从属模式 1b - 启用数字滤波器和输出延迟计数器<br>的从属模式 |
| 3-2<br>- | 保留 |
| 1<br>RST | 软件复位<br>0b - 从属模式逻辑不被重置 1b - 从<br>属模式逻辑被重置 |
| 0<br>秘书 | 从机启用<br>0b - I2C从机模式被禁用 1b - I2C<br>从机模式被启用 |

## 47.3.1.20 从属状态寄存器（SSR）

### 47.3.1.20.1 偏移

| 注册 | 偏移 |
|---|---|
| 证券交易所 | 114h |

## 47.3.1.20.2 图示




## 47.3.1.20.3 领域

| 场地 | 职能 |
|---|---|
| 31-26<br>- | 保留 |
| 25<br>BBF | 公交车的标志<br>指示I2C总线是空闲还是繁忙。<br>0b - I2C总线处于空闲状态 1b - I2C总线处于繁忙状态 |
| 24<br>SBF | 从机忙碌标志<br>指示一个I2C从机是空闲还是繁忙。<br>0b - I2C从机空闲 1b - I2C从机繁忙 |
| 23-16<br>- | 保留 |
| 15<br>SARF | SMBus警报响应标志<br>• 通过读取地址状态寄存器清除SMBus警报响应标志<br>• SMBus警报响应标志不能产生异步唤醒<br>0b - SMBus警报响应被禁用或未检测到 1b - SMBus警报响应被启用并检测到 |

| | |
|---|---|
| 14<br>GCF | 一般呼叫标志<br><br>表明从机是否检测到总调用地址。<br>• 通过读取地址状态寄存器清除一般调用标志<br>• 一般调用标志不能产生异步唤醒<br><br>0b - 从站未检测到总调用地址或总调用地址被禁用 1b - 从站已检测到总调用地址 |
| 13<br>AM1F | 地址匹配1标志<br><br>表示收到的地址与ADDR1字段或ADRCFG配置的ADDR0至ADDR1范围相符。 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | • 地址匹配1标志通过读取地址状态寄存器被清除<br>• 地址匹配1标志不能产生异步唤醒<br><br>0b - 未收到ADDR1或ADDR0/ADDR1范围匹配的地址 1b - 已收到ADDR1或ADDR0/ADDR1范围匹配的地址 |
| 12<br>AM0F | 地址匹配0标志<br>表示收到的地址与ADDRCFG所配置的ADDR0字段相匹配。<br>• 通过读取地址状态寄存器清除地址匹配0标志<br>• 地址匹配0标志不能产生异步唤醒<br><br>0b - 未收到ADDR0匹配地址 1b - 已收到ADDR0匹配地址 |
| 11<br>FEF | FIFO错误标志<br>FIFO错误标志只有在禁用时钟拉伸时才能被设置。<br>0b - 没有检测到FIFO的欠流或溢出 1b - 检测到FIFO的欠流或溢出 |
| 10<br>BEF | 位错误标志<br>如果LPI2C从属设备传输了一个逻辑1并在I2C总线上检测到一个逻辑0，则比特错误标志将被设置。从机将忽略传输的其余部分，直到下一个（重复）START条件。<br>0b - 从机未检测到位错误 1b - 从机检测到位错误 |
| 9<br>SDF | STOP检测标志<br>当LPI2C从机检测到STOP条件时，如果LPI2C从机与最后一个地址字节匹配，STOP检测标志将被设置。<br>0b - 从机未检测到STOP条件 1b - 从机已检测到STOP条件 |
| 8<br>RSF | 重复启动标志<br>当LPI2C从机检测到一个重复的START条件时，如果LPI2C从机与最后一个地址字节相匹配，重复启动标志将被设置。当从机第一次检测到START条件时，重复启动标志不会设置。<br>0b - 从机未检测到重复启动条件 1b - 从机已检测到重复启动条件 |
| 7-4<br>- | 保留 |
| 3<br>TAF | 发送ACK标志<br>通过写入发送 ACK 寄存器清除发送 ACK 标志。<br>0b - 不需要发送ACK/NACK 1b - 需要发送ACK/NACK |
| 2<br>AVF | 地址有效标志<br>地址有效标志通过读取地址状态寄存器而被清除。当接收数据配置（SCFGR1[RXCFG]）被设置时，地址有效标志也会通过读取接收数据寄存器而被清除。 |

| | |
|---|---|
| | 0b - 地址状态寄存器无效 1b - 地址状态寄存器有效 |
| 1<br>RDF | 接收数据标志<br>接收数据标志通过读取接收数据寄存器而被清除。当接收数据配置（SCFGR1[RXCFG]）被设置时，在读取接收数据寄存器和AVF被设置时，接收数据标志不会被清除。<br>0b - 接收数据未准备好 1b - 接收数据已准备好 |

*表继续在下一页...*

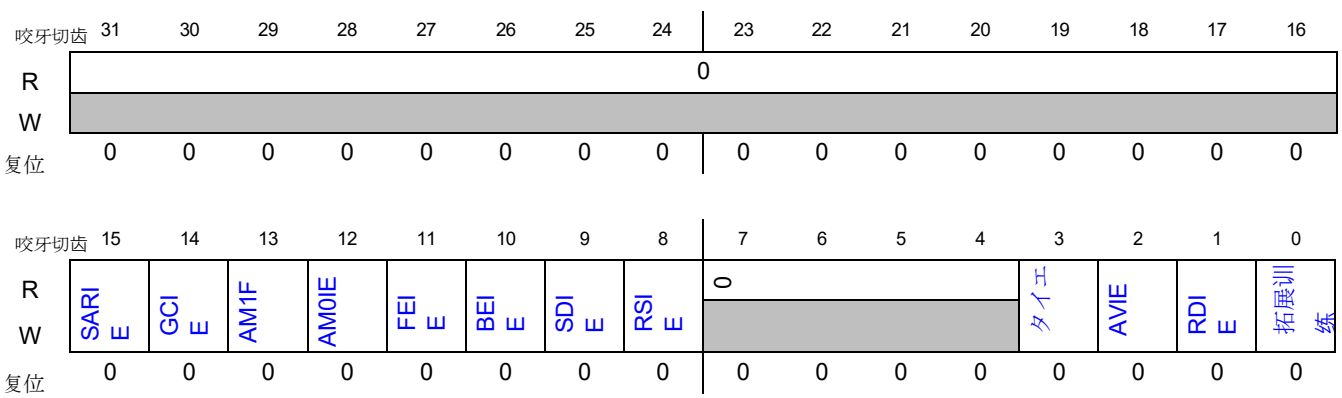
| 场地 | 职能 |
|---|---|
| 0<br>TDF | 发送数据标志<br>发送数据标志通过写入发送数据寄存器而被清除。当传输标志配置（TXCFG）被清除时，如果检测到NACK或重复START或STOP条件，则传输数据标志也被清除。<br>0b - 不要求传输数据 1b - 要求传输数据 |

## 47.3.1.21 从属中断启用寄存器（SIER）。

### 47.3.1.21.1 偏移

| 注册 | 偏移 |
|---|---|
| 证券交易所(SIER) | 118h |

### 47.3.1.21.2 图示




### 47.3.1.21.3 领域

| 场地 | 职能 |
|---|---|
| 31-16<br>- | 保留 |
| 15<br>SARIE | SMBus警报响应中断启用 0b - 禁用<br>1b - 已启用 |

| | |
|---|---|
| 14<br>ĀĀĀ | 一般调用中断启用 0b - 禁用<br>1b - 已启用 |

*表继续在下一页...*

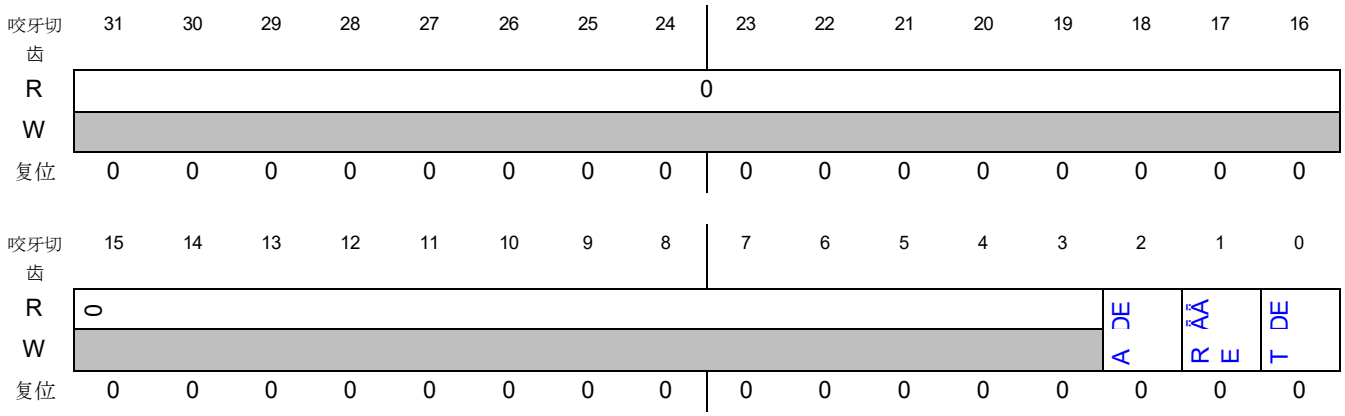
| 场地 | 职能 |
| --- | --- |
| 13<br>AM1F | 地址匹配1中断启用 0b - 禁用<br>1b - 已启用 |
| 12<br>AM0IE | 地址匹配 0 中断启用 0b - 已启用<br>1b - 禁用 |
| 11<br>基金会 | FIFO错误中断启用 0b - 禁用<br>1b - 已启用 |
| 10<br>BEIE | 位错误中断启用 0b - 禁<br>用<br>1b - 已启用 |
| 9<br>SDIE | STOP检测中断启用 0b - 禁用<br>1b - 已启用 |
| 8<br>证券时报 | 重复启动中断启用 0b - 禁用<br>1b - 已启用 |
| 7-4<br>- | 保留 |
| 3<br>タイエ | 发送ACK中断启用 0b - 禁用<br>1b - 已启用 |
| 2<br>AVIE | 地址有效中断启用 0b - 禁用<br>1b - 已启用 |
| 1<br>种族歧视委员会(RDIE) | 接收数据中断启用 0b - 禁用<br>1b - 已启用 |
| 0<br>拓展训练(TDIE) | 发送数据中断启用 0b - 禁用<br>1b - 已启用 |

## 47.3.1.22 从属DMA启用寄存器（SDER）

### 47.3.1.22.1 偏移

| 注册 | 偏移 |
| --- | --- |
| SDER | 11Ch |

### 47.3.1.22.2 图示




### 47.3.1.22.3 领域

| 场地 | 职能 |
|---|---|
| 31-3<br>- | 保留 |
| 2<br>AVDE | 地址有效 DMA启用<br>地址有效DMA请求与接收数据DMA请求共享。如果地址有效DMA请求和接收数据DMA请求都被启用，那么设置接收数据配置（SCFGR1[RXCFG]），以允许DMA从接收数据寄存器读取地址。<br>0b - DMA请求被禁用 1b - DMA请求被启用 |
| 1<br>RDDE | 接收数据 DMA启用<br>0b - DMA请求被禁用 1b - DMA请求被启用 |
| 0<br>TDDE | 发送数据DMA启用<br>0b - DMA请求被禁用 1b - DMA请求被启用 |

## 47.3.1.23 从属配置寄存器1（SCFGR1）。

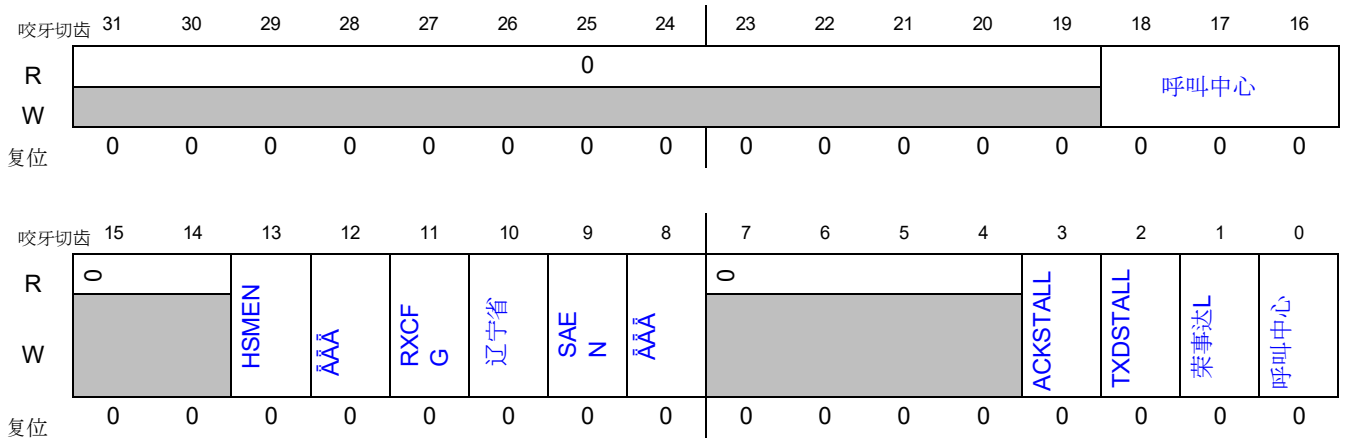
### 47.3.1.23.1 偏移

| 注册 | 偏移 |
|---|---|
| SCFGR1 | 124h |

### 47.3.1.23.2 职能

只有当I2C从机被禁用时，才可以写入从机配置寄存器1。

### 47.3.1.23.3 图示




### 47.3.1.23.4 领域

| 场地 | 职能 |
|---|---|
| 31-19<br>- | 保留 |
| 18-16<br>呼叫中心 | 地址配置<br>配置将导致地址匹配的条件。<br>000b - 地址匹配0（7位） 001b - 地址匹配0（10位）。<br>010b - 地址匹配 0（7 位）或地址匹配 1（7 位） 011b - 地址匹配 0（10 位）或地址匹配 1（10 位） 100b - 地址匹配 0（7 位）或地址匹配 1（10 位） 101b - 地址匹配 0（10 位）或地址匹配 1（7 位）<br>110b - 从地址匹配0（7位）到地址匹配1（7位） 111b - 从地址匹配0（10位）到地址匹配1（10位） |
| 15-14<br>- | 保留 |
| 13<br>HSMEN | 启用高速模式<br>启用检测从属地址0000_1XX的高速模式主码，但不会导致该码的地址匹配。当设置并检测到任何HS模式主码时，FILTEN和ACKSTALL位被忽略，直到检测到下一个STOP条件。<br>0b - 禁用HS模式主代码的检测 1b - 启用HS模式主代码的检测 |

| | |
|---|---|
| 12<br>ĀĀĀ | 忽略NACK<br>当Ignore NACK被设置时，LPI2C从机在检测到NACK后将继续传输。忽略NACK位在超快模式下需要被设置。<br>0b - 当检测到NACK时，从机将结束传输。 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | 1b - 检测到NACK时，从机不会结束传输 |
| 11<br>RXCFG | 接收数据配置<br>0b - 读取接收数据寄存器将返回接收数据并清除接收数据标志（MSR[RDF]）。<br>1b - 当地址有效标志（SSR[AVF]）被设置时读取接收数据寄存器，将返回地址状态寄存器并清除地址有效标志。当地址有效标志被清除时，读取接收数据寄存器，将返回接收数据并清除接收数据标志（MSR[RDF]）。 |
| 10<br>辽宁省 | 发送标志配置<br>在从属传输结束时，传输数据标志总是在检测到NACK之前发出指令。这可能导致一个额外的字被写入发送数据FIFO。<br>• 当TXCFG=0时，当检测到一个从属传输传输时，传输数据寄存器会自动清空。这导致每当检测到一个从属传输传输时，传输数据标志就会被断言，并在从属传输传输结束时导致传输数据标志被否定。<br>• 当TXCFG=1时，当发送数据寄存器为空时，发送数据标志将告警，而当发送数据寄存器为满时，发送数据标志将被否定。这使得传输数据寄存器在检测到从属传输之前被填满，但可能导致传输数据寄存器在检测到从属传输的最后一个字节的NACK之前被写入。<br>0b - 传输数据标志仅在传输数据寄存器为空时，在从属传输传输期间发出指令。<br>1b - 发送数据标志将在发送数据寄存器为空时发出指令。 |
| 9<br>SAEN | SMBus警报启用<br>0b - 禁用SMBus警报的匹配 1b - 启用<br>SMBus警报的匹配 |
| 8<br>ĀĀĀ | 启用一般呼叫<br>0b - 总调用地址被禁用 1b - 总调用地<br>址被启用 |
| 7-4<br>- | 保留 |
| 3<br>ACKSTALL | ACK SCL 滞留<br>在从机发送地址字节和从机接收地址和数据字节期间启用SCL时钟拉伸，以允许软件在发送ACK或NACK之前写入发送ACK寄存器。<br>在传输第9位时发生时钟拉伸，因此与高速模式不兼容。<br>当 ACKSTALL 启用时，不需要设置 RX SCL 停顿（SCFGR1[RXSTALL]）或地址 SCL 停顿（SCFGR1[ADRSTALL]）。<br>0b - 时钟拉伸功能被禁用 1b - 时<br>钟拉伸功能被启用 |
| 2<br>TXDSTALL | TX数据SCL滞留<br>当发送数据标志（SSR[TDF]）在从属传输期间被设置时，启用SCL时钟拉伸。时钟拉伸发生在第9位之后，因此与高速模式兼容。<br>0b - 时钟拉伸功能被禁用 1b - 时<br>钟拉伸功能被启用 |

| | |
|---|---|
| 1<br><br>读取 | RX SCL失速<br><br>当接收数据标志（SSR[RDF]）在从属接收传输期间被设置时，启用SCL时钟拉伸。时钟拉伸发生在第9位之后，因此与高速模式兼容。<br>0b - 时钟拉伸功能被禁用 1b - 时钟拉伸功能被启用 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| 0<br>呼叫中心 | 地址SCL滞留<br>当地址有效标志（SSR[AVF]）被引爆时，启用SCL时钟拉伸。时钟拉伸只发生在第9位之后，因此与高速模式兼容。<br>0b - 时钟拉伸功能被禁用 1b - 时钟拉伸功能被启用 |

## 47.3.1.24 从属配置寄存器2（SCFGR2）。

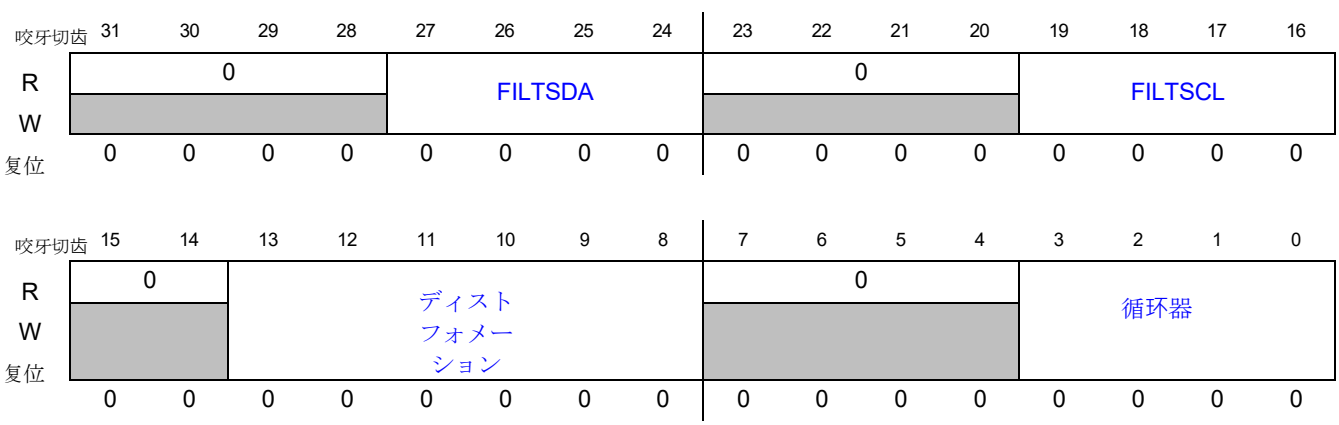
### 47.3.1.24.1 偏移

| 注册 | 偏移 |
|---|---|
| SCFGR2 | 128h |

### 47.3.1.24.2 职能

只有当I2C从机被禁用时，才可以写入从机配置寄存器2。

### 47.3.1.24.3 图示




### 47.3.1.24.4 领域

| 场地 | 职能 |
|---|---|
| 31-28<br>- | 保留 |
| 27-24 | 突波滤波器SDA |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| FILTSDA | 配置SDA输入的I2C从机数字突波滤波器。<br>• 配置为0将禁用突波滤波器。<br>• 等于或小于FILTSDA周期的斑点将被过滤掉，并被忽略。<br>• 通过突波滤波器的延迟等于FILTSDA+3周期，并且必须配置为小于最小SCL低或高周期。<br>• 误码过滤周期计数不受PRESCALE配置的影响，在高速模式下，误码过滤周期计数被禁用。 |
| 23-20<br>- | 保留 |
| 19-16<br>FILTSCL | 突波滤波器SCL<br>配置SCL输入的I2C从机数字突波滤波器。<br>• 配置为0将禁用突波滤波器。<br>• 等于或小于FILTSCL周期的斑点将被过滤掉，并被忽略。<br>• 通过突波滤波器的延迟等于FILTSCL+3周期，必须配置为小于SCL的最小低或高周期<br>• 误码过滤周期计数不受PRESCALE配置的影响，在高速模式下，误码过滤周期计数被禁用。 |
| 15-14<br>- | 保留 |
| 13-8<br>ディストフォメーション | 数据有效延时<br>配置I2C从机的SDA数据有效延迟时间，等于FILTSCL+DATAVD+3周期。<br>• 数据有效延迟必须被配置为小于最小SCL低电平周期。<br>• I2C从机数据有效延迟时间不受PRESCALE配置的影响，在高速模式下，I2C从机数据有效延迟时间被禁用。 |
| 7-4<br>- | 保留 |
| 3-0<br>循环器 | 时钟保持时间<br>配置I2C从机的最小时钟保持时间，当时钟拉伸功能被启用时。<br>• 最小保持时间等于CLKHOLD+3周期<br>• I2C从属时钟保持时间不受PRESCALE配置的影响，在高速模式下I2C从属时钟保持时间被禁用。 |

## 47.3.1.25 从属地址匹配寄存器（SAMR）

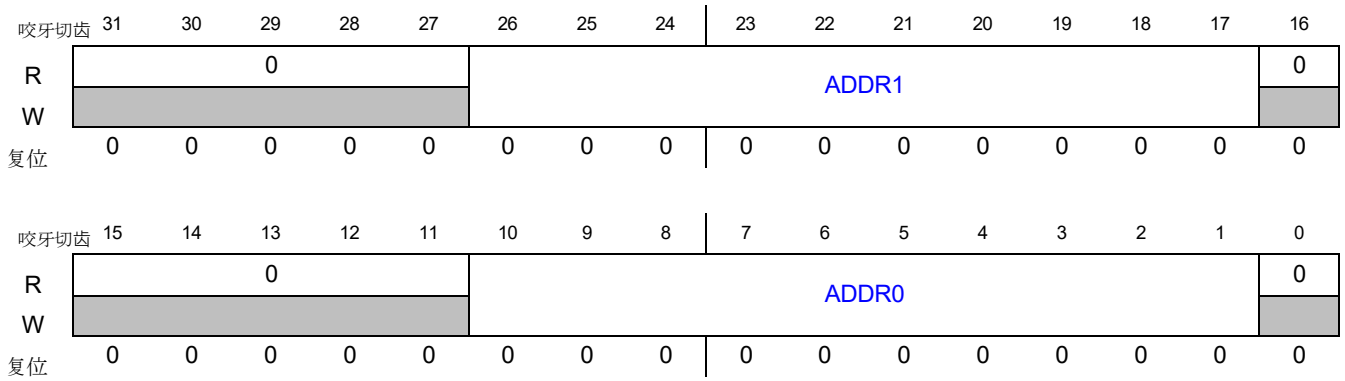
### 47.3.1.25.1 偏移

| 注册 | 偏移 |
| --- | --- |
| 珊珊（SAMR | 140h |

### 47.3.1.25.2 职能

SAMR只能在I2C从机被禁用时被写入。

### 47.3.1.25.3 图示




### 47.3.1.25.4 领域

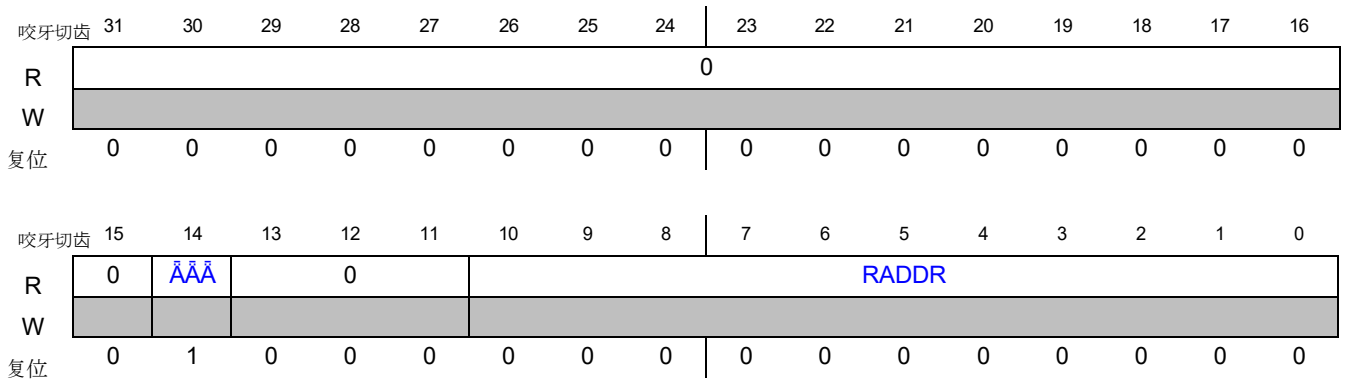
| 场地 | 职能 |
|---|---|
| 31-27<br>- | 保留 |
| 26-17<br>ADDR1 | 地址1 价值<br>与收到的地址进行比较，以检测从属地址。<br>• 在10位模式下，第一个地址字节与{ 11110, ADDR1[10:9] }比较，第二个地址字节与ADDR1[8:1] 比较。<br>• 在7位模式下，地址与ADDR1[7:1]比较。 |
| 16-11<br>- | 保留 |
| 10-1<br>ADDR0 | 地址0 价值<br>与收到的地址进行比较，以检测从属地址。<br>• 在10位模式下，第一个地址字节与{ 11110, ADDR0[10:9] }比较，第二个地址字节与ADDR0[8:1] 比较。<br>• 在7位模式下，地址与ADDR0[7:1]比较。<br>• |
| 0<br>- | 保留 |

## 47.3.1.26 从属地址状态寄存器（SASR）

### 47.3.1.26.1 偏移

| 注册 | 偏移 |
| --- | --- |
| ÃÃÃ | 150h |

### 47.3.1.26.2 图示




### 47.3.1.26.3 领域

| 场地 | 职能 |
|---|---|
| 31-15<br>- | 保留 |
| 14<br>ÃÃÃ | 地址无效<br>0b - 接收的地址（RADDR）有效。<br>1b - 接收的地址（RADDR）无效 |
| 13-11<br>- | 保留 |
| 10-0<br>RADDR | 收件地址<br>只要AMF被设置，接收地址就会更新；AMF通过读取从属地址状态寄存器被清除。<br>• 在7位模式下，地址字节被存储在RADDR[7:0]中。<br>• 在10位模式下，第一个地址字节是{ 11110, RADDR[10:9], RADDR[0] }，第二个地址字节是RADDR[8:1] 。因此R/W位总是存储在RADDR[0]中。 |

## 47.3.1.27 从机发送ACK寄存器（STAR）。

### 47.3.1.27.1 偏移

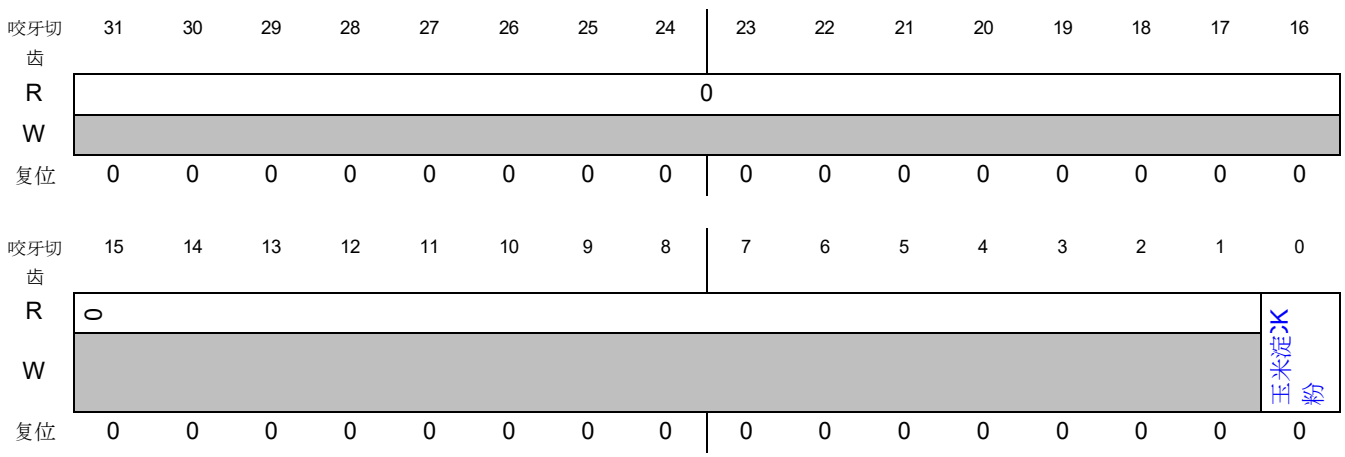
| 注册 | 偏移 |
|---|---|
| 星级 | 154h |

### 47.3.1.27.2 职能

只有当从属配置寄存器1（SCFGR1[ACKSTALL]）中的ACK SCL停顿位被设置时，才能写入从属发送ACK寄存器。

- ACKSTALL位将在ACK/NACK位槽期间启用时钟拉伸，在这段时间内，软件可以写入STAR寄存器。
- 该逻辑确保在STAR寄存器被更新后，时钟拉伸至少持续1个总线时钟周期。
- 这个时钟拉伸时间可以通过时钟保持时间字段（SCFGR2[CLKHOLD]，从属配置寄存器2）进行更多扩展。

### 47.3.1.27.3 图示




### 47.3.1.27.4 领域

| 场地 | 职能 |
|---|---|
| 31-1<br>- | 保留 |
| 0<br>TXNACK | 发送NACK<br>收到每个字后，软件可以传输ACK（逻辑0）或NACK（逻辑1）；传输NACK选择使用哪个。ACK或NACK。<br>• 当ACKSTALL被设置时，必须为每个匹配的地址字节和每个接收的字写一次发送NACK。ACKSTALL必须被设置，因为这将使数据传输停滞，直到软件读取收到的字（并决定是否用ACK或NACK响应）。<br>• 为了配置默认的ACK/NACK，当LPI2C从机被禁用或空闲时，也可以写入发送NACK。<br>0b - 为每个收到的字写一个发送ACK 1b - 为每个收到的字写一个发送NACK |

## 47.3.1.28 从属传输数据寄存器（STDR）

### 47.3.1.28.1 偏移

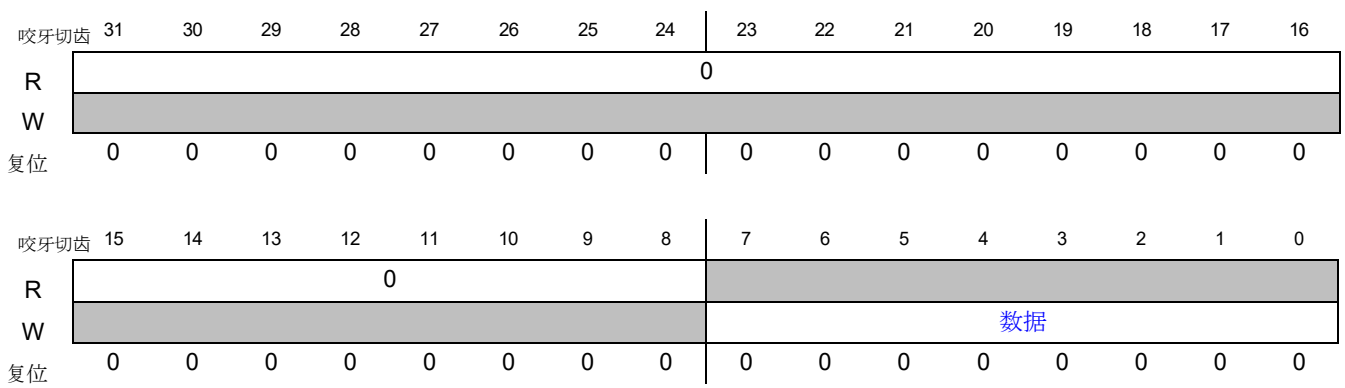
| 注册 | 偏移 |
|---|---|
| STDR | 160h |

### 47.3.1.28.2 职能

时钟拉伸（启用或禁用）会影响传输数据的时间。TXDSTALL位将在从属传输的第一个数据位期间启用时钟拉伸。

- **如果时钟拉伸被启用(TXDSTALL=1)**，那么传输数据的传输就会停滞，直到从属传输寄存器(STDR)被更新。在从属传输数据寄存器（STDR）更新后，时钟拉伸至少延长1个总线时钟周期，使用时钟保持时间字段（SCFGR2[CLKHOLD]，从属配置寄存器2）可以将时钟拉伸延迟得更久。
- **如果时钟拉伸被禁用（TXDSTALL=0）**，那么发送数据应在从属传输开始前写入；否则（即，如果发送数据在从属传输开始前没有写入），FIFO错误标志（SSR[FEF]）将被设置。

### 47.3.1.28.3 图示




### 47.3.1.28.4 领域

| 场地 | 职能 |
|---|---|
| 31-8<br>- | 保留 |

*表继续在下一页...*

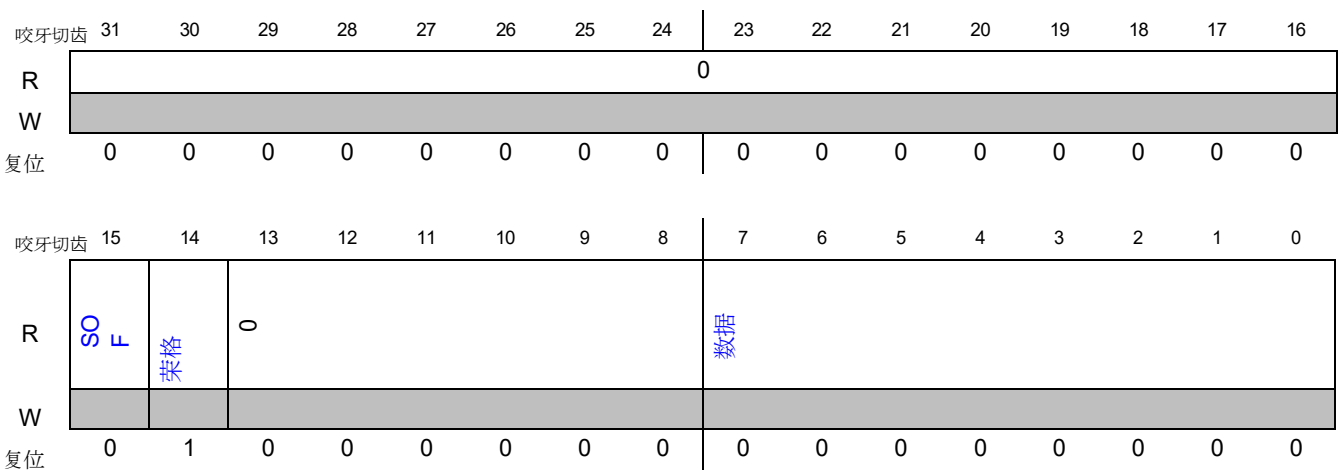
| 场地 | 职能 |
|---|---|
| 7-0<br>数据 | 发送数据<br>写入从属传输数据寄存器（STDR）将*在*从属传输数据寄存器*中*存储I2C从属传输数据。 |

## 47.3.1.29 从属接收数据寄存器（SRDR）

### 47.3.1.29.1 偏移

| 注册 | 偏移 |
|---|---|
| SRDR | 170h |

### 47.3.1.29.2 图示




### 47.3.1.29.3 领域

| 场地 | 职能 |
|---|---|
| 31-16<br>- | 保留 |
| 15<br>SOF | 帧的开始<br>0b - 表示这不是一个（重复的）START或STOP条件后的第一个数据字 1b - 表示这是一个（重复的）START或STOP条件后的第一个数据字 |

| | |
|---|---|
| 14<br>荣格 | RX空<br>0b - 接收数据寄存器不为空 1b - 接收数据寄存器为空 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| 13-8<br>- | 保留 |
| 7-0<br>数据 | 接收数据<br>读取从机接收数据寄存器返回I2C从机接收的数据 |

# 47.4 职能描述

## 47.4.1 时钟和复位

**表47-4.时钟**

| | |
|---|---|
| LPI2C功能时钟 | LPI2C功能时钟与总线时钟是异步的，在低功耗模式下可以保持启用，以支持LPI2C主站的I2C总线传输。功能时钟也被LPI2C从机使用，以支持数字滤波器和数据保持时间配置。LPI2C主控器将功能时钟除以一个预分频器，所得频率必须比I2C总线带宽至少快8倍。 |
| 外部时钟 | LPI2C从机逻辑直接从外部引脚SCL和SDA（或SCLS和SDAS，如果主从机是在不同的引脚上实现的）获得时钟。这使得LPI2C从机即使在LPI2C功能时钟被禁用时也能保持运行。<br>**注意：**如果禁用LPI2C功能时钟，则必须禁用LPI2C从属数字滤波器，这可能会影响对I2C规范中一些定时参数的遵守，如数据保持时间。 |
| 总线时钟 | 总线时钟只用于对控制和配置寄存器的总线访问。总线时钟频率必须足以支持LPI2C主控和从属寄存器的数据带宽要求。 |

**表47-5.重置**

| | |
|---|---|
| 芯片复位 | LPI2C主站和从站的逻辑和寄存器在芯片复位时被重置为其默认状态。 |
| 软件复位 | • LPI2C主站在其控制寄存器中实现了一个软件复位位。MCR[RST]将把所有的主控逻辑和寄存器复位到它们的默认状态，除了MCR本身。<br>• LPI2C从机在其控制寄存器中实现了一个软件复位位。除了SCR本身，SCR[RST]将把所有的从机逻辑和寄存器复位到其默认状态。 |
| FIFO复位 | • LPI2C主站实现了只写的控制位，用于重置发送FIFO（MCR[RTF]）和接收FIFO（MCR[RRF]）。一个FIFO被重置后，该FIFO为空。<br>• LPI2C从机实现了只写的控制位，用于复位发送数据寄存器（SCR[RTF]）和接收数据寄存器（SCR[RRF]）。一个数据寄存器被复位后，该数据寄存器为空。 |

## 47.4.2 主模式

LPI2C主控逻辑独立于从属逻辑运行，在I2C总线上执行所有主控模式传输。

### 47.4.2.1 发送和命令FIFO命令

发送FIFO存储命令数据以启动各种I2C操作。以下操作可以通过发送FIFO中的命令来启动。

- START或重复START条件下的地址字节，期待ACK或NACK。
- 发送数据（这是向发送FIFO写入零扩展字节的默认值）。
- 接收1-256字节的数据（也可以配置为丢弃接收数据，不存储在接收FIFO中）。
- STOP条件（也可以配置为在发送FIFO为空时发送STOP条件）。

在START条件和STOP条件之间可以插入多个发送和接收命令；发送和接收命令不得交错使用（以符合I2C规范）。接收数据命令和接收数据和丢弃命令可以交错使用，以确保只有所需的接收数据被存储在接收FIFO中（或与数据匹配逻辑进行比较）。

除非FIFO中的下一条命令也是一条接收数据命令，否则LPI2C主站会在接收数据命令的最后一个字节上自动发送一个NACK。如果接收数据命令完成时发送FIFO为空，也会自动发送NACK。

LPI2C主站通过一个（重复的）START条件支持10位寻址，然后是一个包含第二个地址字节的发送数据字节，接着是任意数量的带有主站发送数据的数据字节。

一个期待NACK的START或Repeated START条件（例如，HS-模式的主码）之后必须有一个STOP或（重复）START条件。

### 47.4.2.2 主操作

每当LPI2C被启用时，它就会监视I2C总线，以检测I2C总线何时空闲（MSR[BBF]）。如果SCL或SDA为低电平，I2C总线就不再被认为是空闲的，如果检测到STOP条件或总线空闲超时，I2C总线就成为空闲的。

检测到（如MCFGR2[BUSIDLE]所配置的）。在I2C总线空闲后，发送FIFO不为空，并且主机请求被断言或被禁用，那么LPI2C主站将在I2C总线上启动传输。这包括以下步骤。

- 等待总线空闲时间等于（MCCR0[CLKLO] + 1）乘以预分频器（MCFGR1[PRESCALE]）。
- 使用主时钟配置寄存器0（MCCR0）中的时序配置传送START条件和地址字节；如果配置了高速模式传输，则使用主时钟配置寄存器1（MCCR1）的时序配置。
- 按照发送FIFO的配置，进行主发送或主接收的传输。
- 在主控-
  接收传输的最后一个字节上发送NACK，除非发送FIFO中的下一个命令也是一个接收数据命令，并且发送FIFO不是空的。
- 传输FIFO和/或MCFGR1[AUTOSTOP]所配置的重复START或STOP条件。重复START可以改变使用哪个定时配置寄存器。

当LPI2C主控器被禁用时（由于MCR[MEN]被清除或由于模式进入而自动禁用），LPI2C将继续清空发送FIFO，直到传输一个STOP条件。然而，LPI2C将不再拖延I2C总线等待发送或接收FIFO，在发送FIFO被清空后，LPI2C将自动产生一个STOP条件。

在某些情况下，LPI2C主控器可以使I2C总线停滞；这将导致SCL在一个字节的第一位上持续拉低，直到条件消除。

- LPI2C主控器被启用且繁忙，发送FIFO为空，MCFGR1[AUTOSTOP]被清除。
- LPI2C主站启用并接收数据，接收数据没有被丢弃（由于命令或接收数据匹配），并且接收FIFO已满。

### 47.4.2.3 接收FIFO和数据匹配

接收FIFO用于在主站-
接收站传输期间存储接收数据。接收数据也可以配置为丢弃接收数据而不是存储在接收FIFO中；这是由发送FIFO中的命令字配置的。

接收数据支持接收数据匹配功能，该功能可以将接收的数据与两个字节中的一个或与一个屏蔽的数据字节进行匹配。数据匹配功能也可以被配置为只比较自上一次（重复）START条件以来的前一个或两个接收数据字。接收到的数据由于命令

的原因已经被丢弃了

字不能导致数据匹配的设置，并且会将第一个收到的数据字的匹配推迟到收到被丢弃的数据之后。也可以通过MCFGR0[RDMO]控制位将接收器匹配功能配置为丢弃所有接收数据，直到检测到数据匹配。在数据匹配后清除MCFGR0[RDMO]控制位时，在清除MSR[DMF]之前清除MCFGR0[RDMO]，以允许所有后续数据被接收。

### 47.4.2.4 计时参数

以下定时参数可由LPI2C主控器配置。参数是为高速模式（MCCR1）和其他模式（MCCR0）分别配置的。这允许使用常规的定时参数发送高速模式的主控代码，然后切换到高速模式的定时（在重复START之后），直到下一个STOP条件。

LPI2C功能时钟周期中的LPI2C主控定时参数配置如下。它们必须被配置为满足所需模式的I2C定时规范。

**表47-6.计时参数**

| I2C规格时序参数 | I2C规格时序符号 | LPI2C计时参数（LPI2C功能时钟周期）。 |
|---|---|---|
| 保持时间（重复） START条件 | tHD:STA | (SETHOLD +1) x (2 ^ PRESCALE) |
| SCL时钟的低电平周期 | tLOW | (CLKLO + 1) x (2 ^ PRESCALE) |
| SCL时钟的高电平周期 | 高 | (CLKHI + 1 + SCL_LATENCY) x (2 ^ PRESCALE) |
| 重复START条件或STOP条件的设置时间 | tSU:STA, tSU:STO | (SETHOLD + 1 + SCL_LATENCY) x (2 ^ PRESCALE) |
| 数据保持时间 | tHD:DAT | (DATAVD + 1) x (2 ^ PRESCALE) |
| 数据设置时间 | tSU:DAT | (SDA_LATENCY + 1) x (2 ^ PRESCALE) |
| STOP和START条件之间的总线自由时间 | tBUF | (CLKLO + 1 + SDA_LATENCY) x (2 ^ PRESCALE) |
| 数据有效时间，数据有效确认时间 | tVD:DAT, tVD:ACK | (DATAVD + 1) x (2 ^ PRESCALE) |

延迟参数定义在下表中，这些参数假定上升时间小于一个LPI2C功能时钟周期。risetime取决于许多因素，包括I2C总线负载和外部上拉电阻的大小。更大的上升时间将增加信号通过同步器（和突波滤波器）的周期数，从而增加延迟。

**表47-7.同步延时**

| 计时参数 | 时机的定义 |
|---|---|
| 速率(SCL_LATENCY) | 四舍五入 ( (2 + filtscl + scl_risetime) / (2 ^ prescale) ) |
| SDA_LATENCY | 四舍五入 ( (2 + filtsda + sda_risetime) / (2 ^ prescale) ) |

必须执行以下时序限制，以避免I2C总线上出现意外的START或STOP条件，或避免LPI2C主控器检测到的意外START或STOP条件。这些时序限制可以总结为：**当SCL在传输的（重复的）START或STOP条件之外为高电平时，SDA不能改变**。

**表47-8.LPI2C计时参数限制**

| 计时参数 | 最低限度 | 最大 | 评论 |
|---|---|---|---|
| 辽宁省 | 0x03 | - | 另外。CLKLO x (2 ^ PRESCALE) > SCL_LATENCY |
| 辽宁省 | 0x01 | - | 配置CLKHI以满足I2C规范中的占空比要求 |
| 阈值 | 0x02 | - | |
| ディストフォメーション | 0x01 | clklo - sda_latency - 1 | 配置DATAVD以满足I2C规范中的数据保持要求 |
| FILTSCL | 0x00 | [clklo × (2 ^ prescale)] - 3 | FILTSCL和FILTSDA是唯一没有乘以(2 ^ PRESCALE)的参数。 |
| FILTSDA | FILTSCL | [clklo × (2 ^ prescale)] - 3 | 配置FILTSDA大于FILTSCL可以延迟SDA输入，以补偿板级偏移。 |
| 布西德勒 | (clklo+sethold+2) × 2 | - | 也必须大于（CLKHI+1）。 |

必须对定时参数进行配置以满足I2C规范的要求；这将取决于所支持的模式和LPI2C功能时钟频率。当使用不同的时钟配置寄存器在两种模式之间切换时（例如，快速模式和HS模式），PRESCALE系数必须在两种模式之间保持不变。下面提供了一些定时配置的例子。

表47-9.LPI2C示例定时配置

| I2C模式 | 时钟频率 | 波特率1 | 预售 | 滤波器cl / 滤波器da | 阈值 | 辽宁省 | 辽宁省 | ディストフォメーション |
|---|---|---|---|---|---|---|---|---|
| 快速 | 8兆赫 | 400 kbps | 0x0 | 0x0/0x0 | 0x04 | 0x0B | 0x05 | 0x02 |
| 快速+ | 8兆赫 | 1 Mbps | 0x0 | 0x0/0x0 | 0x02 | 0x03 | 0x01 | 0x01 |
| 快速 | 48兆赫 | 400 kbps | 0x0 | 0x1/0x1 | 0x1D | 0x3E | 0x35 | 0x0F |
| 快速 | 48兆赫 | 400 kbps | 0x2 | 0x1/0x1 | 0x07 | 0x11 | 0x0B | 0x03 |
| 快速+ | 48兆赫 | 1 Mbps | 0x2 | 0x1/0x1 | 0x03 | 0x06 | 0x04 | 0x04 |
| HS-模式 | 48兆赫 | 3.2 Mbps | 0x0 | 0x0/0x0 | 0x07 | 0x08 | 0x03 | 0x01 |
| 快速 | 60兆赫 | 400 kbps | 0x1 | 0x2/0x2 | 0x11 | 0x28 | 0x1F | 0x08 |
| 快速+ | 60兆赫 | 1 Mbps | 0x1 | 0x2/0x2 | 0x07 | 0x0F | 0x0B | 0x01 |
| HS-模式 | 60兆赫 | 3.33 Mbps | 0x1 | 0x0/0x0 | 0x04 | 0x04 | 0x02 | 0x01 |

1.你的特定设备可能不支持所有的时钟频率和波特率，请检查模块的实例化和系统信息。

### 47.4.2.5 错误条件

LPI2C主站在活动时将监测错误，下列情况将产生一个错误标志，并阻止新的START条件的发送，直到该标志被软件清除。

- 检测到一个START或STOP条件，并且不由LPI2C主站产生（设置MSR[ALF]）。
- 在SDA上传输数据，并且正在接收不同的数值（设置MSR[ALF]）。
- 发送数据时检测到NACK，MCFGR1[IGNACK]清零（设置MSR[NDF]）。
- 检测到NACK并期待地址字节的ACK，MCFGR1[IGNACK]清零（设置MSR[NDF]）。
- 检测到ACK，并期待地址字节的NACK，MCFGR1[IGNACK]清零（设置MSR[NDF]）。
- 发送FIFO在没有START条件下请求发送或接收数据（设置MSR[FEF]）。
- SCL（或SDA，如果MCFGR1[TIMECFG]被设置）在没有引脚转换的情况下，在（MCFGR2[TIMELOW]*256）预分频器周期内为低电平（设置MSR[PLTF]）。

软件必须对MSR[PTLF]标志做出响应，以干净地（通过清除MCR[MEN]）或突然地（通过设置MCR[SWRST]）终止现有命令。

MCFGR2[BUSIDLE]字段可用于强制I2C总线在SCL和SDA保持高电平（BUSIDLE+1）预分频器周期时被视为空闲。I2C总线通常在LPI2C主控器首次启用时被认为是空闲的，但当BUSIDLE被配置为大于0时，在I2C总线首次被认为是空闲之前，SCL和/或SDA必须保持高电平（BUSIDLE+1）预分频器周期。

#### 47.4.2.6 引脚配置

- **开漏支持。**LPI2C主控器默认为SDA和SCL引脚的开漏配置。对真正开漏的支持取决于具体的设备，并要求LPI2C引脚被混合的引脚支持真正的开漏。
- **高速模式支持。**对高速模式的支持也取决于具体的设备，并要求SCL引脚支持I2C规范中要求的电流源上拉。
- **支持超快模式。**LPI2C主控器还支持使用SDA和SCL引脚的I2C超快模式所需的纯输出推拉功能。对超快模式的支持还要求IGNACK位被设置。
- **推挽式2线支持。**如果LPI2C是唯一的主控器，并且总线上的所有I2C引脚处于相同的电压，那么LPI2C主控器也可以使用推挽式2线配置，可以支持部分高速模式。这将把SCL引脚配置为除第9个时钟脉冲外的每个时钟的推挽，以允许兼容高速模式的从机进行时钟拉伸。在这种模式下，SDA引脚在主站接收数据位和主站发送ACK/NACK位时被三态化，而在其他时间被配置为推挽。为了避免SDA被推拉时出现争用的风险，该引脚可以被配置为开漏操作，作为设备特定配置的一部分。
- **推挽式4线支持。**推挽式4线配置将SCL输入数据和输出数据分离成独立的引脚，将SDA输入数据和输出数据分离成独立引脚。SCL/SDA引脚用于输入数据；SCLS/SDAS引脚用于输出数据，可配置极性。这简化了通过外部电平转换器或分立元件将LPI2C连接到I2C总线时的外部连接。当使用这种4线配置时，LPI2C主逻辑和LPI2C从逻辑不能连接到单独的I2C总线。

### 47.4.3 从属模式

为了在I2C总线上执行所有的从属模式传输，LPI2C从属逻辑与LPI2C主逻辑独立运行。

#### 47.4.3.1 地址匹配

可以对LPI2C从机进行配置。

- 以匹配两个地址中的一个，每个地址使用7位或10位寻址模式
- 来匹配7位或10位寻址模式下的地址范围
- 以匹配总调用地址，并产生适当的标志
- 以匹配SMBus警报地址，并产生适当的标志。
- 检测高速模式下的主码，并禁用数字滤波器和输出有效延迟时间，直到检测到下一个STOP条件。

在一个有效的地址被匹配后，LPI2C从机将自动进行从机发送或从机接收的传输，直到。

- 检测到一个NACK（除非设置了IGNACK）。
- 检测到一个比特错误（LPI2C从机驱动SDA，但采样的数值不同）。
- 检测到一个（重复的）START或STOP条件

#### 47.4.3.2 发送和接收数据

- 发送和接收数据寄存器是双缓冲的，只在从属传输和从属接收传输期间分别更新。
- *收到的*从属地址可以被配置为从接收数据寄存器（例如，当使用DMA传输数据时），或从地址状态寄存器中读取。
- 传输数据寄存器可被配置为只在检测到从属传输后请求数据，或在传输数据寄存器为空时请求新数据。
- 发送数据寄存器只能在发送数据标志被设置时被写入。
- 只有当接收数据标志被设置（或地址有效标志被设置且RXCFG=1）时，才能读取接收数据寄存器。
- 地址状态寄存器只有在地址有效标志被设置时才能被读取。

#### 47.4.3.3 钟表拉伸

LPI2C从机支持许多可配置的选项，以确定何时执行时钟拉伸。以下条件可以被配置为执行时钟拉伸。

- 在地址字节的第9个时钟脉冲期间，并且地址有效标志被设置。

- 在一个从属传输的第9个时钟脉冲期间，并且传输数据标志被设置。
- 在一个从属接收传输的第9个时钟脉冲期间，并且接收数据标志被设置。
- 在一个地址字节或一个从属接收传输的第8个时钟脉冲期间，并且发送ACK标志被设置。在高速模式下，该功能被禁用。
- 时钟拉伸也可以延长CLKHOLD周期，以允许额外的设置时间来对SDA引脚进行外部采样。在高速模式下，该功能被禁用。

除非通过CLKHOLD配置来扩展，否则当时钟拉伸被启用时，时钟拉伸将在SDA更新后延长一个外围总线时钟周期。

#### 47.4.3.4 计时参数

LPI2C从机可以配置下列定时参数。当SCR[FILTEN]清零、SCR[FILTDZ]在Doze模式下被设置，以及LPI2C从机检测到高速模式时，这些参数被禁用。当禁用时，LPI2C从机直接从I2C总线上获得时钟，可能不满足I2C规范的所有时序要求（如标准/快速模式下的SDA最小保持时间）。

- 从SCL否定到SDA更新的SDA数据有效时间
- 启用时钟拉伸时的SCL保持时间，以增加外部采样SDA时的设置时间
- SCL突波过滤时间
- SDA突波过滤时间

LPI2C从机对定时参数有以下限制。

- FILTSDA必须被配置为大于或等于FILTSCL（除非补偿SDA和SCL之间的板级偏移）。
- DATAVD的配置必须小于SCL的最小低周期。

#### 47.4.3.5 错误条件

LPI2C从机可以检测到以下错误情况。

- 当LPI2C从机驱动SDA，但采样值与预期不同时，比特错误标志将被设置。

- FIFO错误标志将由于发送数据不足或接收数据超限而设置。为了消除发生欠载和超载的可能性，请启用时钟拉伸。
- 当RXCFG被设置时，FIFO错误标志也会因地址超限而被设置，否则地址超限不会被标记。为了消除超限发生的可能性，请启用时钟拉伸。

LPI2C从机没有实现由于SCL和/或SDA被卡在低电平而产生的超时。如果需要这种检测，那么应该使用LPI2C主控逻辑，因此软件可以在检测到这种情况时重置LPI2C从机。

## 47.4.4 中断和DMA请求

根据具体的设备，中断和DMA请求可以以某些方式结合起来。

- LPI2C的主中断和从中断可以结合起来
- LPI2C主站和从站发送DMA请求可以结合起来
- LPI2C主站和从站的接收DMA请求可以结合起来

### 47.4.4.1 主模式

下表列出了可以产生LPI2C主控中断和LPI2C主控发送/接收DMA请求的主控模式源。

**表47-10.主站中断和DMA请求**

| 旗帜 | 描述 | 可以通过以下方式产生 | | |
|---|---|---|---|---|
| | | 中断？ | DMA请求？ | 低功率唤醒？ |
| TDF | 数据可被写入传输FIFO，由TXWATER配置。 | Y | 德克萨斯州 | Y |
| RDF | 可以从接收FIFO中读取数据，由RXWATER配置。 | Y | 荣格 | Y |
| EPF | 主站传送了一个重复的START或STOP条件 | Y | N | Y |
| SDF | 主站传送了一个STOP条件 | Y | N | Y |
| NDF | • 在一个地址字节中，主站期待一个ACK，但检测到一个NACK。<br>• 在一个地址字节中，主站期望一个NACK，但检测到一个ACK。<br>• 在一个主站发送的数据字节中，主站检测到一个NACK | Y | N | Y |

*表继续在下一页...*

表47-10.主站中断和DMA请求（续）

| 旗帜 | 描述 | 可以通过以下方式产生 | | |
|---|---|---|---|---|
| | | 中断？ | DMA请求？ | 低功率唤醒？ |
| ALF | • 由于在错误的时间检测到START/STOP条件，主站失去仲裁。<br>• 或者主站在传输数据，但收到的数据与传输的数据不同。 | Y | N | Y |
| FEF | 主站在命令FIFO中期待一个START条件，但命令FIFO中的下一个条目不是一个START条件。 | Y | N | Y |
| PLTF | 引脚低电平超时被启用，且SCL（或SDA，如果配置）的低电平时间超过配置的超时。 | Y | N | Y |
| DMF | 收到的数据与配置的数据匹配，但由于命令FIFO条目，收到的数据没有被丢弃 | Y | N | Y |
| MBF | LPI2C主站正忙于发送/接收数据 | N | N | N |
| BBF | LPI2C主站被启用，I2C总线上检测到活动，但没有检测到STOP条件，也没有发生总线空闲超时（如果启用）。 | N | N | N |

### 47.4.4.2 从属模式

下表列出了可以产生LPI2C从属中断的从属模式源和LPI2C从属发送/接收DMA请求。

表47-11.从属中断和DMA请求

| 旗帜 | 描述 | 可以通过以下方式产生 | | |
|---|---|---|---|---|
| | | 中断？ | DMA请求？ | 低功率唤醒？ |
| TDF | 数据可以被写入传输数据寄存器 | Y | 德克萨斯州 | Y |
| RDF | 可以从接收数据寄存器中读取数据 | Y | 荣格 | Y |
| AVF | 可以从地址状态寄存器中读取地址 | Y | 荣格 | Y |
| TAF | ACK/NACK可以写到发送ACK寄存器。 | Y | N | Y |
| RSF | 从机检测到一个地址匹配，然后是一个重复的START条件。 | Y | N | Y |
| SDF | 从机检测到一个地址匹配，然后是一个STOP条件。 | Y | N | Y |

| | | | | |
|---|---|---|---|---|
| BEF | 从机正在传输数据，但收到的数据与传输的不同。 | Y | N | Y |

*表继续在下一页...*

表47-11.从属中断和DMA请求（续）

| 旗帜 | 描述 | 可以通过以下方式产生 | | |
|---|---|---|---|---|
| | | 中断？ | DMA请求？ | 低功率唤醒？ |
| FEF | • 发送数据不足<br>• 接收数据超限<br>• 地址状态超限（当RXCFG=1时）。<br><br>FEF标志只有在时钟拉伸被禁用时才能设置。 | Y | N | Y |
| AM0F | 从机检测到一个与ADDR0字段相匹配的地址 | Y | N | N |
| AM1F | 从机检测到一个与ADDR1字段匹配的地址，或使用一个地址范围。 | Y | N | N |
| GCF | 从机检测到一个与总调用地址相匹配的地址 | Y | N | N |
| SARF | 从机检测到与SMBus警报地址匹配的地址。 | Y | N | N |
| SBF | LPI2C从机正忙于接收一个地址字节或正在传输/接收数据 | N | N | N |
| BBF | LPI2C从机被启用，并且在I2C总线上检测到一个START条件，但尚未检测到一个STOP条件。 | N | N | N |

## 47.4.5 周边触发器

LPI2C外设触发器与其他外设的连接取决于所使用的具体设备。

表47-12.LPI2C触发器

| 触发器 | 描述 |
|---|---|
| 主输出触发器 | LPI2C主控器产生一个输出触发器，可以连接到设备上的其他外设。主站输出触发器在重复启动或停止条件下都会发出信号，并且主站输出触发器在LPI2C功能时钟除以预分频器的一个周期内保持发出信号。 |
| 从属输出触发器 | LPI2C从机产生一个输出触发器，可以连接到设备上的其他外设。从机输出触发器在从机地址匹配后发生的重复启动或停止条件下都会被引爆，从机输出触发器一直被引爆，直到下一个从机SCL引脚被否定。 |
| 输入触发器 | 为了控制LPI2C总线传输的开始，可以选择LPI2C输入触发器而不是HREQ输入。输入触发器是同步的，要被检测到，输入触发器必须至少在LPI2C功能时钟除以PRESCALE配置的2个周期内断言。<br>当LPI2C繁忙时，HREQ输入（以及输入触发）被忽略。 |

# 第48章
# 低功耗通用异步接收器/发射器（LPUART）。

## 48.1 芯片特有的LPUART信息

### 48.1.1 实例化信息

LPUART模块支持基本的UART，具有DMA接口功能和×4至×32的超采样波特率。下表总结了该模块在产品系列中各芯片的实现情况。

**表48-1.LPUART实例和特性**

| 芯片 | 实例 | TX FIFO (字) | RX FIFO (字) |
|---|---|---|---|
| S32K116 | 呼叫中心 | 4 | 4 |
| | 卫星定位系统(LPUART1) | 4 | 4 |
| S32K118 | 呼叫中心 | 4 | 4 |
| | 卫星定位系统(LPUART1) | 4 | 4 |
| S32K142 | 呼叫中心 | 4 | 4 |
| | 卫星定位系统(LPUART1) | 4 | 4 |
| S32K144 | 呼叫中心 | 4 | 4 |
| | 卫星定位系统(LPUART1) | 4 | 4 |
| | 朴信惠2 | 4 | 4 |
| S32K146 | 呼叫中心 | 4 | 4 |
| | 卫星定位系统(LPUART1) | 4 | 4 |
| | 朴信惠2 | 4 | 4 |
| S32K148 | 呼叫中心 | 4 | 4 |
| | 卫星定位系统(LPUART1) | 4 | 4 |
| | 朴信惠2 | 4 | 4 |

该模块支持LIN主站和从站操作。

该器件不支持低漏电和等待模式。关于可用的电源模式的详细信息，请参见模块在可用的低功率模式下的操作。

## 48.2 简介

### 48.2.1 特点

LPUART模块的特点包括：。

- 全双工，标准的非返回到零（NRZ）格式
- 可编程波特率（13位模数分频器），可配置的过采样率为4倍至32倍
  - 发送和接收波特率可以与总线时钟异步运行。
  - 波特率可以独立于总线时钟频率进行配置
  - 支持在停止模式下操作
- 中断、DMA或轮询操作。
  - 传输数据寄存器为空，传输完成
  - 接收数据寄存器满了
  - 接收超限、奇偶校验错误、成帧错误和噪音错误
  - 空闲的接收器检测
  - 接收引脚上的有效边缘
  - 支持LIN的中断检测
  - 接收数据匹配
- 硬件奇偶校验的生成和检查
- 可编程的7位、8位、9位或10位字符长度
- 可编程的1位或2位停止位
- 三种接收器唤醒方法。
  - 闲置线路唤醒
  - 地址标记唤醒
  - 接收数据匹配
- 自动地址匹配以减少ISR的开销。
  - 地址标记匹配
  - 闲置线路地址匹配
  - 地址匹配开始，地址匹配结束
- 可选的13位断字生成/11位断字检测
- 可配置的空闲长度检测，支持1、2、4、8、16、32、64或128个空闲字符
- 可选择发射器输出和接收器输入极性

- 支持请求发送（RTS）和清除发送（CTS）信号的硬件流量控制
- 可选的IrDA 1.4返回到零转换（RZI）格式，具有可编程的脉冲宽度
- 发送和接收的独立FIFO结构
  - 为接收和发送请求单独配置水印
  - 如果接收FIFO不是空的，接收器可以选择在可配置的空闲字符数之后发出请求。

## 48.2.2 操作模式

### 48.2.2.1 停止模式

只要DOZEEN位被清除，并且异步发送和接收时钟保持启用，LPUART在停止模式期间将保持功能。LPUART可以产生一个中断或DMA请求，以导致从停止模式唤醒。

如果LPUART在停止模式下被禁用，那么如果接收器检测到一个有效边沿，它可以通过RXEDGIF标志产生一个唤醒。

### 48.2.2.2 调试模式

LPUART在调试模式下保持功能。

## 48.2.3 信号描述

| 信号 | 描述 | 输入/输出 |
|---|---|---|
| TXD | 发送数据。这个引脚通常是一个输出，但在单线模式下，每当发射器被禁用或发射方向被配置为接收数据时，这个引脚就是一个输入（三态）。 | 输入/输出 |
| 荣事达 | 接收数据。 | I |
| CTS_B | 清除发送。 | I |
| RTS_B | 要求发送。 | O |

## 48.2.4 框图

下图显示了LPUART的发射器部分。

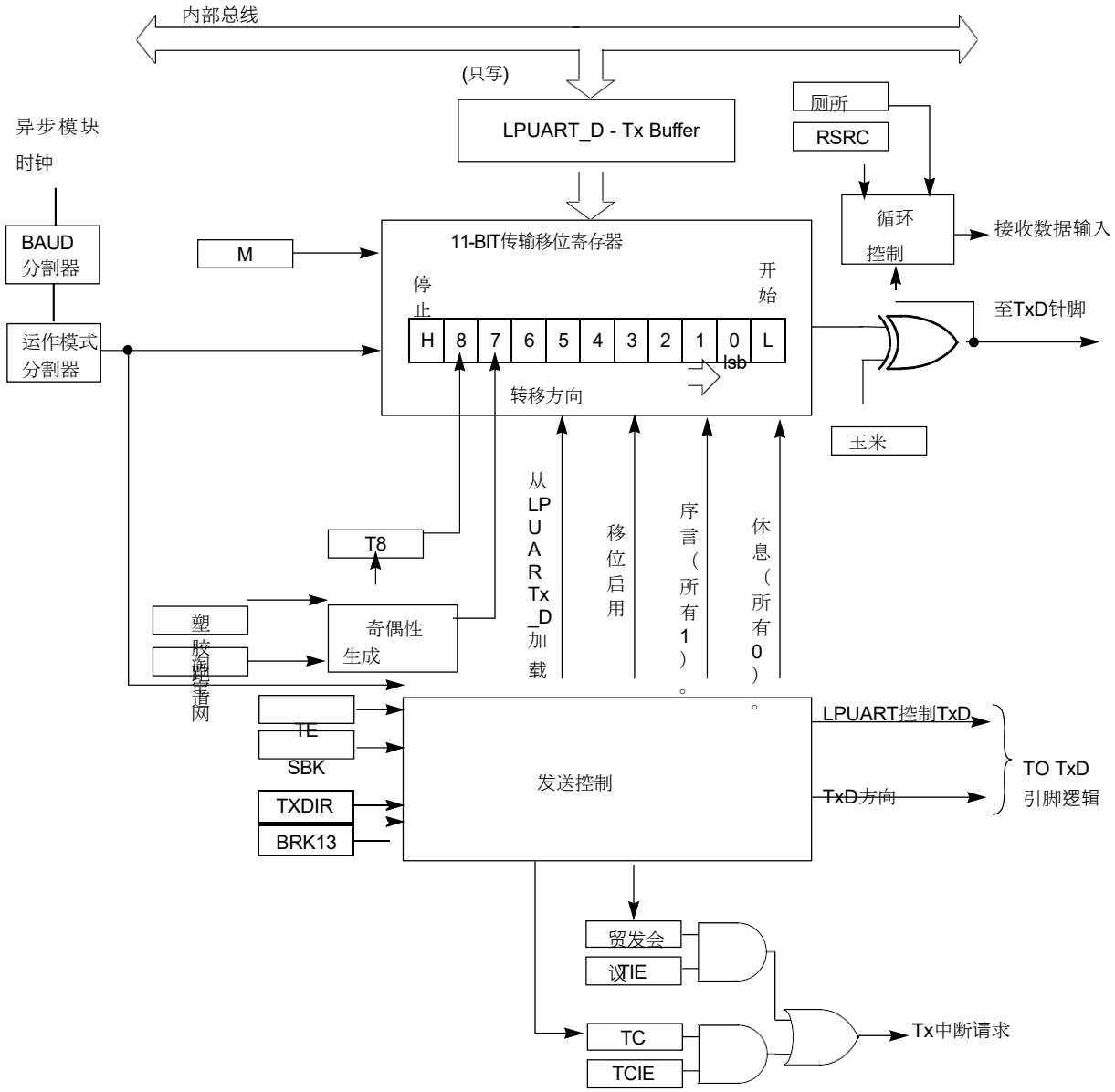



**图48-1.LPUART发射器框图**

下图显示了LPUART的接收器部分。

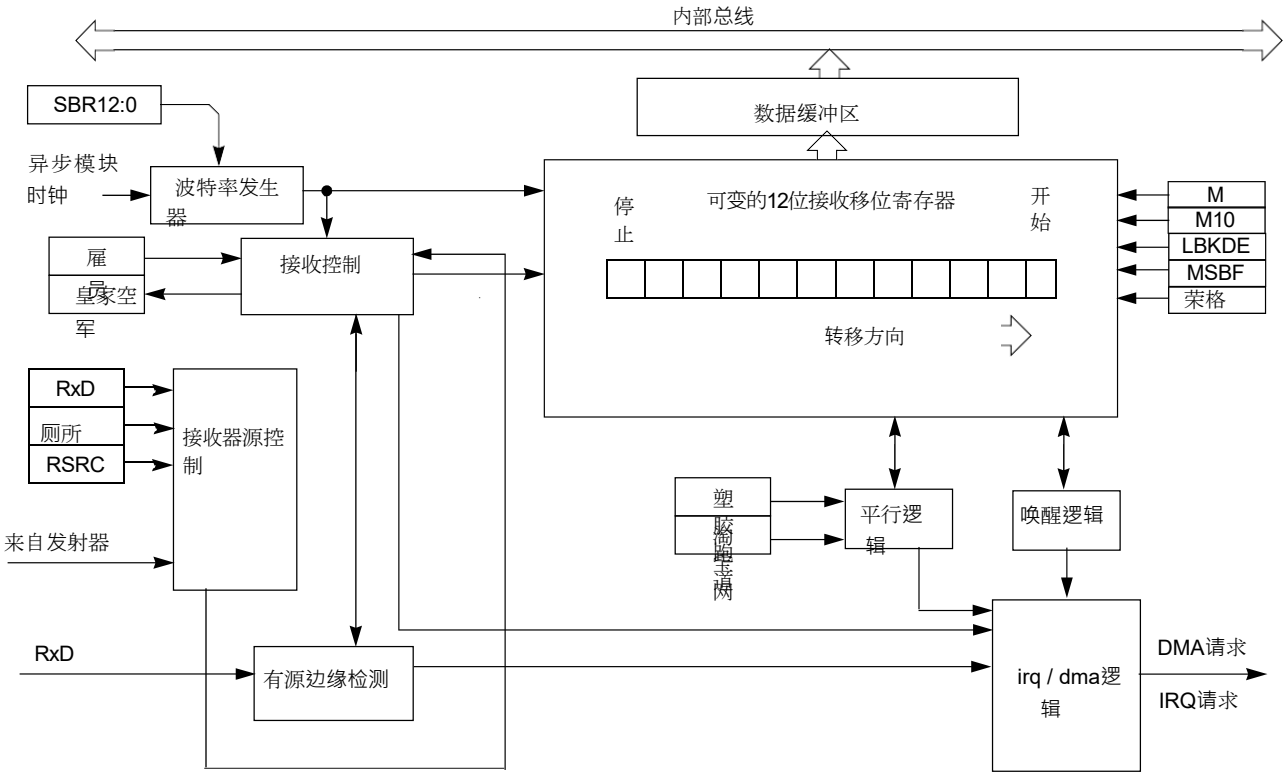



图48-2.LPUART接收器框图

## 48.3 注册定义

LPUART包括控制波特率、选择LPUART选项、报告LPUART状态和用于发送/接收数据的寄存器。对有效内存图之外的地址的访问将产生一个总线错误。

### 48.3.1 LPUART寄存器描述

#### 48.3.1.1 LPUART内存地图

LPUART0基地址。4006_A000h

LPUART1基地址。4006_B000h

LPUART2 基准地址。4006_C000h

注册定义

| 偏移 | 注册 | 宽度<br>(单位：比特) | 访问 | 重置值 |
|---|---|---|---|---|
| 0h | 版本标识寄存器（VERID） | 32 | RO | 0401_0003h |
| 4h | 参数寄存器（PARAM）。 | 32 | RO | 0000_0202h |
| 8h | LPUART全局寄存器（GLOBAL）。 | 32 | RW | 0000_0000h |
| 丘吉尔 | LPUART引脚配置寄存器（PINCFG） | 32 | RW | 0000_0000h |
| 10h | LPUART波特率寄存器（BAUD）。 | 32 | RW | 0F00_0004h |
| 14h | LPUART状态寄存器（STAT） | 32 | RW | 00C0_0000h |
| 18h | LPUART控制寄存器（CTRL） | 32 | RW | 0000_0000h |
| 1Ch | LPUART数据寄存器（DATA） | 32 | RW | 0000_1000h |
| 20h | LPUART匹配地址寄存器（MATCH） | 32 | RW | 0000_0000h |
| 24h | LPUART调制解调器IrDA寄存器（MODIR）。 | 32 | RW | 0000_0000h |
| 28h | LPUART FIFO寄存器（FIFO） | 32 | RW | 00C0_0011h |
| 2Ch | LPUART水印寄存器（WATER） | 32 | RW | 0000_0000h |

## 48.3.1.2 版本标识寄存器（VERID）

### 48.3.1.2.1 偏移

| 注册 | 偏移 |
|---|---|
| 虚实对照 | 0h |

### 48.3.1.2.2 职能

版本ID寄存器表示设备上这个实例所集成的版本，也表示包含/不包含几个可选的功能。

### 48.3.1.2.3 图示



| | |
|---|---|
| W | |
| 复位 | |

#### 48.3.1.2.4 领域

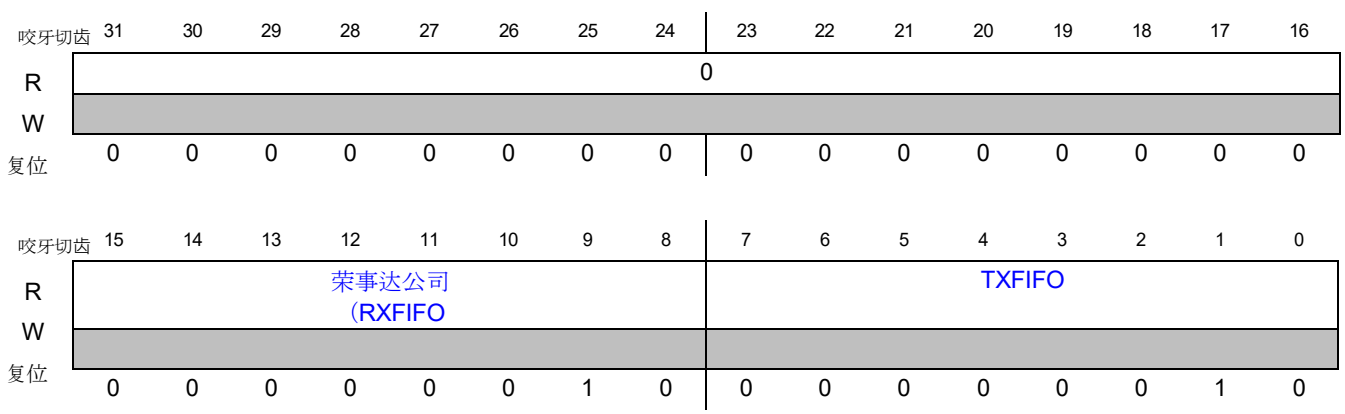
| 场地 | 职能 |
|---|---|
| 31-24<br>主要的 | 主要版本号<br>这个只读字段返回模块规范的主要版本号。 |
| 23-16<br>小学 | 次要版本号<br>这个只读字段返回模块规范的次要版本号。 |
| 15-0<br>特写 | 特征识别码<br>这个只读字段返回特征集编号。0000000000000001b<br>- 标准特征集。<br>0000000000000011b - 支持MODEM/IrDA的标准功能集。 |

### 48.3.1.3 参数寄存器（PARAM）。

#### 48.3.1.3.1 偏移

| 注册 | 偏移 |
|---|---|
| 朴实无华 | 4h |

#### 48.3.1.3.2 图示




#### 48.3.1.3.3 领域

| 场地 | 职能 |
| --- | --- |
| 31-16<br>- | 保留 |

*表继续在下一页...*

注册定义

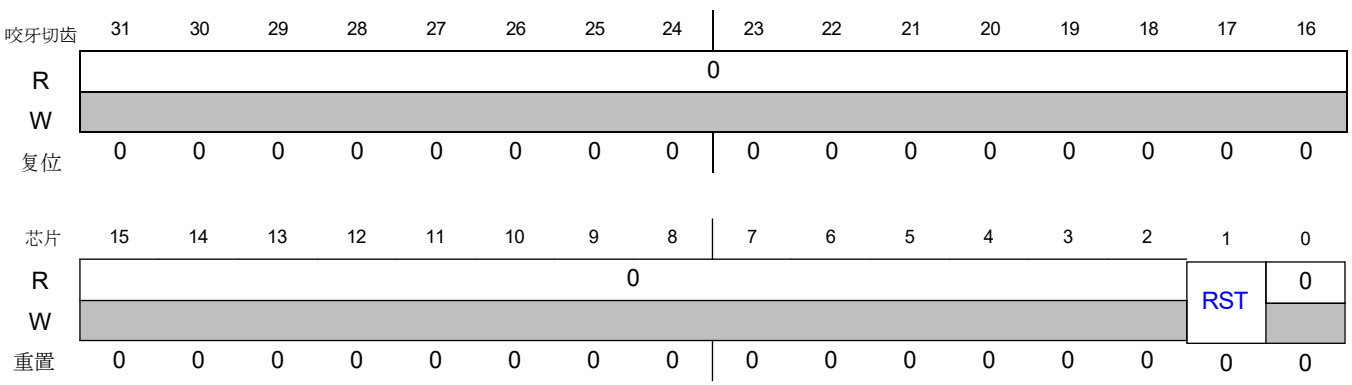
| 场地 | 职能 |
| --- | --- |
| 15-8<br>荣事达公司（<br>RXFIFO | 接收FIFO大小<br>接收FIFO的字数为2^RXFIFO。 |
| 7-0<br>TXFIFO | 发送FIFO大小<br>发送FIFO的字数为2^TXFIFO。 |

## 48.3.1.4 LPUART全局寄存器（GLOBAL）。

### 48.3.1.4.1 偏移

| 注册 | 偏移 |
| --- | --- |
| 全球 | 8h |

### 48.3.1.4.2 图示




### 48.3.1.4.3 领域

| 场地 | 职能 |
| --- | --- |
| 31-2<br>- | 保留 |
| 1<br>RST | 软件复位<br>复位所有的内部逻辑和寄存器，除了全局寄存器。在被软件清除之前一直保持设置。<br>0b - 模块不复位。<br>1b - 模块被重置。 |

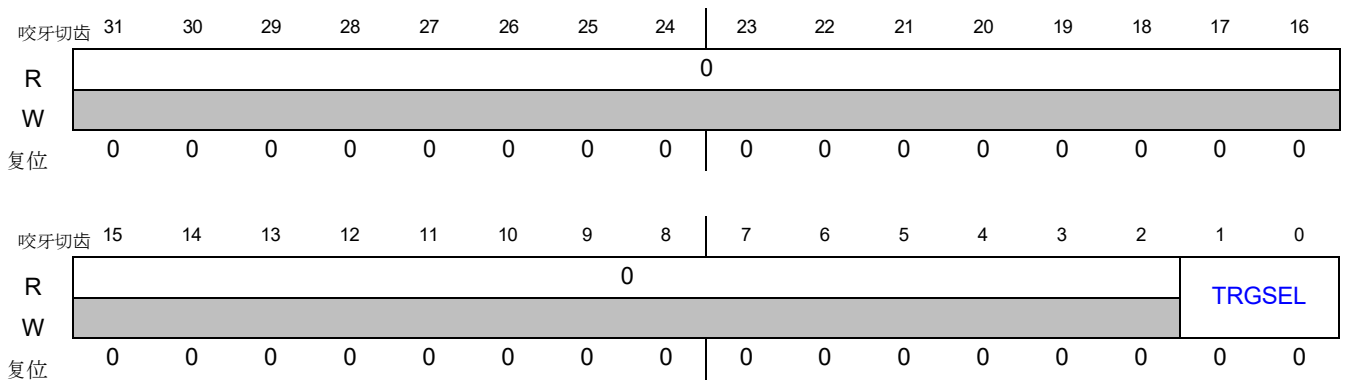
| | |
|---|---|
| 0<br>- | 保留 |

## 48.3.1.5 LPUART引脚配置寄存器（PINCFG）

### 48.3.1.5.1 偏移

| 注册 | 偏移 |
|---|---|
| 铂尼弗格(PINCFG) | 丘吉尔 |

### 48.3.1.5.2 图示




### 48.3.1.5.3 领域

| 场地 | 职能 |
|---|---|
| 31-2<br>- | 保留 |
| 1-0<br>TRGSEL | 触发器选择<br>配置输入触发器的使用。这个字段只有在发射器和接收器都被禁用时才可以改变。<br>00b - 输入触发器被禁用。<br>01b - 使用输入触发器而不是RXD引脚输入。10b - 使<br>用输入触发器而不是CTS_B引脚输入。<br>11b - 输入触发器用于调制TXD引脚的输出。TXD引脚的输出（在TXINV配置后）与输入触发器相加<br>。 |

## 48.3.1.6 LPUART波特率寄存器（BAUD）。

## 48.3.1.6.1 偏移

| 注册 | 偏移 |
|---|---|
| BAUD | 10h |

## 48.3.1.6.2 图示

| 咬牙切齿 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 硕士1 | 闳仁2 | M10 | OSR | | | | | 淘宝网 | 0 | RDMAE | 辽宁省 | MATCFG | | 玻瑟德吉 | 纪念品S |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 咬牙切齿 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | LBKDIE | 荣成市教育局 | SBNS | SBR | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |

## 48.3.1.6.3 领域

| 场地 | 职能 |
|---|---|
| 31<br>硕士1 | 匹配地址模式启用 1 0b - 正常操作。<br>1b - 启用 MATCH[MA1] 的自动地址匹配或数据匹配模式。 |
| 30<br>闳仁2 | 匹配地址模式启用2 0b - 正常操作。<br>1b - 启用 MATCH[MA2] 的自动地址匹配或数据匹配模式。 |
| 29<br>M10 | 10位模式选择<br>M10位使第十位成为串行传输的一部分。该位只有在发射器和接收器都被禁用时才可以改变。<br>0b - 接收器和发射器使用7位至9位数据字符。 1b - 接收器和发射器使用10位数据字符。 |

| | |
|---|---|
| 28-24<br>运作模式 | 过度取样率<br><br>该字段配置接收器的超采样率。这个字段只有在发射器和接收器都被禁用时才可以改变。<br>00000b - 向这个字段写0将导致16的过采样率 00001b - 保留<br>00010b - 保留<br>00011b - 过采样率为4，需要设置BOTHEDGE。00100b - 过采样率为5，需要设置BOTHEDGE。00101b - 过采样率为6，需要设置BOTHEDGE。00110b - 过采样率为7，需要设置BOTHEDGE。<br>00111b - 过采样率为8。 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | 01000b - 9的过采样率。 01001b - 10的过采样率。10010b--过采样率为19。 10011b--过采样率为20。 10100b - 过度取样比率为21.10101b - 逾越了22.10110b - 逾越了23.10111b - 逾越了24.11000b - 过度取样比率为25。11001b - 逾越的比率为26。 11010b - 逾越的比率为27。 11011b - 逾越的比率为28。 11100b - 逾越的比率为29。 11110b--过采样率为31。 11111b--过采样率为32。 |
| 23<br>淘宝网 | 发射器DMA启用<br>TDMAE配置发送数据寄存器空标志，LPUART_STAT[TDRE]，以产生一个DMA请求。<br>0b - 禁用DMA请求。1b - 启用DMA请求。 |
| 22<br>- | 保留 |
| 21<br>RDMAE | 接收器全DMA启用<br>RDMAE配置接收器数据寄存器满标志，LPUART_STAT[RDRF]，以产生一个DMA请求。<br>0b - 禁用DMA请求。1b - 启用DMA请求。 |
| 20<br>辽宁省 | 接收器空闲DMA启用<br>RIDMAE配置接收器空闲标志，LPUART_STAT[IDLE]，以产生DMA请求。当该位被设置时，当DATA[RXEMPT]或DATA[IDLINE]位被设置时，读取LPUART_DATA将产生一个End Of Packet响应，直到现有DMA传输完成。在包结束响应期间，读取LPUART_DATA寄存器将返回0x0000_33FF，并且不从FIFO中提取数据。<br>0b - 禁用DMA请求。1b - 启用DMA请求。 |
| 19-18<br>MATCFG | 匹配配置<br>配置使用的匹配寻址模式。这个字段只应在发射器和接收器都被禁用时改变。<br>00b - 地址匹配唤醒 01b - 空闲匹配唤醒<br>10b - 匹配开和匹配关<br>11b - 启用数据匹配的RWU和发送器CTS输入的匹配开/关。 |

| | |
|---|---|
| 17 | 两种边缘取样 |

*表继续在下一页...*

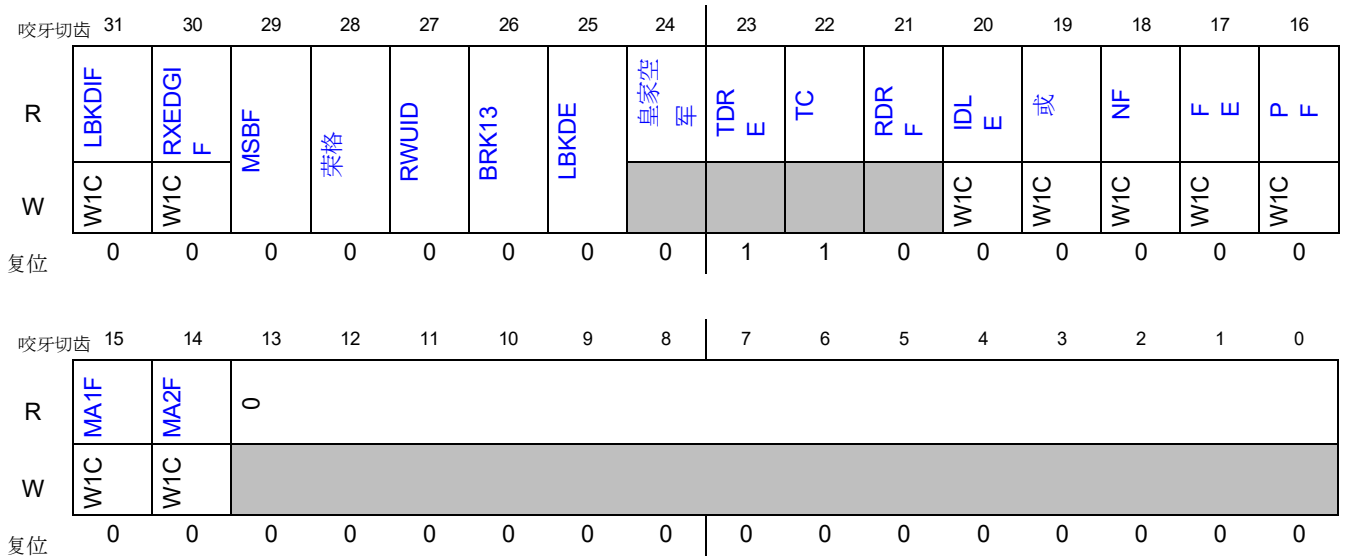
| 场地 | 职能 |
|---|---|
| 玻瑟德吉 | 启用波特率时钟两个边沿的接收数据采样，对于给定的过采样率，接收器对输入数据的采样次数有效增加一倍。该位必须被设置用于x4和x7之间的过采样率，对于更高的过采样率是可选的。该位只应在接收器被禁用时改变。<br>0b - 接收器使用波特率时钟的上升沿采样输入数据。<br>1b - 接收器使用波特率时钟的上升和下降沿对输入数据进行采样。 |
| 16<br>锐意进取 | 禁用重新同步化<br>当设置时，当检测到数据一和数据零的转换时，禁止对接收的数据字进行再同步。该位只应在接收器被禁用时改变。<br>0b - 支持在接收数据字期间进行再同步 1b - 禁用在接收数据字期<br>间进行再同步 |
| 15<br>LBKDIE | LIN中断检测中断启用<br>LBKDIE启用LIN中断检测标志LBKDIF，以产生中断请求。0b - 禁止来自<br>LPUART_STAT[LBKDIF]的硬件中断（使用轮询）。1b - 当<br>LPUART_STAT[LBKDIF]标志为1时请求硬件中断。 |
| 14<br>RXEDGIE | RX输入有效边缘中断启用<br>启用接收输入有效边沿，RXEDGIF，以产生中断请求。当RXEDGIE被设置时，改变CTRL[LOOP]或CTRL[RSRC]会导致RXEDGIF被设置。<br>0b - 禁止来自LPUART_STAT[RXEDGIF]的硬件中断。<br>1b - 当LPUART_STAT[RXEDGIF]标志为1时请求硬件中断。 |
| 13<br>证券时报 | 停止位数选择<br>SBNS决定了数据字符是一个还是两个停止位。该位只有在发射器和接收器都被禁用时才能改变。<br>0b - 一个停止位。<br>1b - 两个停止位。 |
| 12-0<br>SBR | 波特率调制除数。<br>SBR[12:0]中的13位设置波特率发生器的模数除法速率。当SBR为1-8191时，波特率等于 "波特时钟/（（OSR+1）×SBR）"。13位波特率设置[SBR12:SBR0]只有在发射器和接收器都被禁用（LPUART_CTRL[RE]和LPUART_CTRL[TE]都为0）时才必须更新。 |

## 48.3.1.7 LPUART状态寄存器（STAT）

### 48.3.1.7.1 偏移

| 注册 | 偏移 |
|---|---|
| 统计局 | 14h |

## 48.3.1.7.2 图示




## 48.3.1.7.3 领域

| 场地 | 职能 |
|---|---|
| 31<br>LBKDIF | LIN中断检测中断标志<br>LBKDIF在LIN断路检测电路被启用且检测到LIN断路字符时被设置。LBKDIF通过写1被清除。<br>0b - 没有检测到LIN中断字符。1b - 已检测到LIN断裂字符。 |
| 30<br>RXEDGIF | RXD引脚有效边缘中断标志<br>RXEDGIF被设置，只要接收器被启用，并且在RXD引脚上出现活动边缘，如果RXINV=0则下降，如果RXINV=1则上升。RXEDGIF可以通过写1来清除。<br>0b - 接收引脚上没有发生活动边缘。1b - 接收引脚上有一个有效边沿。 |
| 29<br>MSBF | MSB优先<br>设置该位可以颠倒电线上传输和接收的比特的顺序。该位不影响位的极性、奇偶校验位的位置或起始或停止位的位置。只有在发射器和接收器都被禁用的情况下才可以改变这个位。<br>0b - LSB（bit0）是在起始位之后传输的第一个位。此外，在起始位之后收到的第一个位被确定为位0。<br>1b - MSB（bit9、bit8、bit7或bit6）是在起始位之后传输的第一个位，这取决于CTRL[M]、CTRL[PE]和BAUD[M10]的设置。此外，根据CTRL[M]和CTRL[PE]的设置，在起始位之后收到的第一个位被确定为位9、位8、位7或位6。 |

| | |
|---|---|
| 28<br>荣格 | 接收数据反转<br><br>设置该位可以颠倒接收数据输入的极性。该位只应在接收器被禁用时改变。<br><br>**注意：**设置RXINV可以反转所有情况下的RXD输入：数据位、起始位和停止位、中断和空闲。<br>0b - 接收数据不倒置。1b - 接收<br>数据反转。 |
| 27 | 接收 唤醒 闲置检测 |

*表继续在下一页...*

注册定义

| 场地 | 职能 |
| --- | --- |
| RWUID | 对于空闲字符的RWU，RWUID控制唤醒接收器的空闲字符是否设置IDLE位。对于地址匹配唤醒，RWUID控制在地址不匹配时是否设置IDLE位。这个位只应在接收机被禁用时改变。<br>0b - 在接收待机状态（RWU=1），IDLE位在检测到空闲字符时不被设置。在地址匹配唤醒期间，当地址不匹配时，IDLE位不被设置。<br>1b - 在接收待机状态（RWU=1），IDLE位在检测到空闲字符时被设置。在地址匹配唤醒期间，当地址不匹配时，IDLE位会被设置。 |
| 26<br>BRK13 | 破解字符生成长度<br>BRK13选择一个较长的传输中断字符长度。对成帧错误的检测不受该位状态的影响。该位只应在发送器被禁用时改变。通过设置CTRL[SBK]或在DATA[FRETSC]设置和DATA[R9T9]清零的情况下写入发送FIFO，可以发送一个中断字符。<br>0b - 以9至13比特的长度传输中断字符。1b - 以12至15比特的长度传输中断字符。 |
| 25<br>LBKDE | 启用LIN中断检测<br>LBKDE选择一个较长的断裂字符检测长度。当LBKDE被设置时，接收数据不存储在接收数据缓冲区。<br>0b - 禁用LIN断点检测，可以检测正常断点字符。<br>1b - 启用LIN中断检测。在长度为11位次（如果M=0）或12位次（如果M=1）或13位次（M10=1）时检测到LIN中断字符。 |
| 24<br>皇家空军 | 接收器激活标志<br>当接收机检测到一个有效的起始位时，RAF被设置，当接收机检测到一个空闲线路时，RAF自动清除。<br>0b - LPUART接收器闲置，等待启动位。1b - LPUART接收器激活（RXD输入不空闲）。 |
| 23<br>贸发会议 | 发送数据寄存器空标志<br>当发送FIFO被启用时，当发送FIFO（LPUART_DATA）中的数据字数等于或小于LPUART_WATER[TXWATER]指示的数字时，TDRE将被设置。）要清除TDRE，请向LPUART数据寄存器（LPUART_DATA）写入，直到发送FIFO中的字数大于LPUART_WATER[TXWATER]指示的数字。当发送FIFO被禁用时，TDRE将在发送数据寄存器（LPUART_DATA）为空时被设置。要清除TDRE，请向LPUART数据寄存器（LPUART_DATA）写入。<br>TDRE不受正在传输中的字符的影响，它在每个传输的字符开始时被更新。<br>0b - 发送数据缓冲区满。<br>1b - 发送数据缓冲区为空。 |
| 22<br>TC | 变速箱完成标志<br>当有一个传输正在进行中，或者当加载了一个前奏或中断字符时，TC被清零。当发送缓冲区为空，没有数据、序言或中断字符正在传输时，TC被设置。当TC被设置时，发送数据输出信号变为空闲（逻辑1）。通过向 LPUART_DATA 写入以传输新数据，通过清空然后设置 LPUART_CTRL[TE]来排队等待一个前奏，通过向 LPUART_CTRL[SBK]写 1 来排队等待一个中断字符，TC 被清空。<br>0b - 发送器活动（发送数据、序言或中断）。1b - 发送器空闲（传输活动完成）。 |

| | |
|---|---|
| 21<br>RDRF | 接收数据寄存器满标志<br><br>当接收FIFO被启用时，当接收缓冲器中的数据字数大于LPUART_WATER[RXWATER]所指示的数字时，RDRF被设置。要清除 RDRF，请读取 LPUART_DATA，直到接收数据缓冲区中的数据字数等于或小于LPUART_WATER[RXWATER]指示的数字。当接收FIFO被禁用时，当接收缓冲器（LPUART_DATA）满时，RDRF被设置。要清除RDRF，请读取LPUART_DATA寄存器。 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | 一个正在接收的字符不会引起RDRF的变化，直到收到整个字符。即使RDRF被设置，字符将继续被接收，直到收到整个字符后发生超限条件。<br>0b - 接收数据缓冲区为空。1b - 接收数据缓冲区满。 |
| 20<br>怠速 | 闲置线路标志<br>当LPUART接收线在一段时间的活动后成为一个完整的字符时间的空闲时，IDLE被设置。当ILT被清零时，接收器开始计算起始位后的空闲位时间。如果接收字符都是1，这些位时间和停止位时间将计入接收器检测空闲线路所需的逻辑高电平的全字符时间，即10至13位时间。当ILT被设置时，接收器在停止位之后才开始计算空闲位时间。停止位和前一个字符结束时的任何逻辑高位时间不计入接收器检测空闲线路所需的全字符逻辑高位时间。<br>要清除IDLE，将逻辑1写到IDLE标志上。在IDLE被清除后，它不能再次被设置，直到一个新的字符被存储在接收缓冲区或一个LIN中断字符设置了LBKDIF标志后。即使接收线长时间处于空闲状态，IDLE也只被设置一次。<br>0b - 没有检测到空闲线路。<br>1b - 检测到空闲线路。 |
| 19<br>或 | 接收器超限标志<br>当软件未能防止接收数据寄存器溢出数据时，OR被设置。OR位在完全收到溢出缓冲器的数据字的停止位后立即被设置，所有其他错误标志（FE、NF和PF）被阻止设置。移位寄存器中的数据会丢失，但已经在LPUART数据寄存器中的数据不受影响。如果启用LBKDE并检测到LIN中断，如果在收到下一个数据字符之前不清除LBKDIF，OR字段就会被提示。<br>当OR标志被设置时，即使有足够的空间，也不会在数据缓冲器中存储额外的数据。要清除OR，将逻辑1写到OR标志上。<br>0b - 无超限。<br>1b - 接收超限（新的LPUART数据丢失）。 |
| 18<br>NF | 噪声旗<br>接收器中使用的高级采样技术在每个接收比特中取三个样本。如果这些样本中的任何一个与帧中任何比特时间内的其他样本不一致，那么该字符的噪声就被检测出来。每当从LPUART_DATA读取的下一个字符在该字符中检测到噪声时，NF就被设置。要清除NF，向NF写逻辑一。<br>0b - 没有检测到噪音。<br>1b - 在LPUART_DATA的接收字符中检测到噪声。 |
| 17<br>基金会 | 成帧错误标志<br>每当从LPUART_DATA读出的下一个字符被检测到逻辑0时，FE就会被设置，因为在那里预期有一个停止位。要清除FE，请向FE写入逻辑1。<br>0b - 没有检测到成帧错误。这并不保证构架正确。1b - 成帧错误。 |

| | |
|---|---|
| 16<br>基金会 | 奇偶校验错误标志<br>每当从LPUART_DATA读取的下一个字符是在启用奇偶校验（PE=1）时收到的，并且收到的字符中的奇偶校验位与预期的奇偶校验值不一致时，就会设置PF。要清除PF，请向PF写一个逻辑1。<br>0b - 没有奇偶性错误。<br>1b - 奇偶校验错误。 |
| 15<br>MA1F | 匹配1旗 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | 每当从LPUART_DATA读取的下一个字符与MA1匹配时，MA1F就会被设置。要清除MA1F，请向MA1F写一个逻辑1。<br>0b - 接收的数据不等于MA1 1b - 接收的数据等于MA1 |
| 14<br>MA2F | 匹配2旗<br>只要从LPUART_DATA读取的下一个字符与MA2匹配，就会设置MA2F。要清除MA2F，请向MA2F写一个逻辑1。<br>0b - 接收的数据不等于MA2 1b - 接收的数据等于MA2 |
| 13-0<br>- | 保留 |

## 48.3.1.8 LPUART控制寄存器（CTRL）

### 48.3.1.8.1 偏移

| 注册 | 偏移 |
| --- | --- |
| CTRL | 18h |

### 48.3.1.8.2 职能

这个读/写寄存器控制LPUART系统的各种可选功能。只有在发射器和接收器都被禁用的情况下才能改变这个寄存器。

### 48.3.1.8.3 图示

| 咬牙切齿 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| R<br>W | R8T9 | R9T8 | TXDIR | 玉米 | ORIE | NEIE | FEIE | PEIE | TIE | TCIE | RIE | ILIE | TE | RE | RWU | SBK |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 咬牙切齿 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| R<br>W | 闵晖 | 辽宁沈阳 | 0 | | M7 | IDLECFG | | | 厕所 | 窦泽恩 | RSRC | M | 唤醒 | 劳工 | PE | 淘宝 |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

## 48.3.1.8.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>R8T9 | 接收位8/发送位9<br>当LPUART被配置为9位或10位数据格式时，R8是接收的第九个数据位。读取9位或10位数据时，在读取LPUART_DATA之前先读取R8。<br>当LPUART被配置为10位数据格式时，T9是收到的第10个数据位。当写入10位数据时，在写入LPUART_DATA之前写入T9。如果T9不需要从其以前的值中改变，例如当它被用来产生地址标记或奇偶校验时，它们不需要在每次写入LPUART_DATA时都写入它。<br>**注意：**R8是一个只读位，T9是一个只写位，读取的值与写入的值不同。 |
| 30<br>R9T8 | 接收位9/发送位8<br>当LPUART被配置为10位数据格式时，R9是接收的第10个数据位。读取10位数据时，在读取LPUART_DATA之前先读取R9<br>当LPUART被配置为9位或10位数据格式时，T8是接收的第九个数据位。当写入9位或10位数据时，在写入LPUART_DATA之前写入T8。如果T8不需要从其以前的值中改变，例如当它被用来产生地址标记或奇偶校验时，它们不需要在每次写入LPUART_DATA时都写入它。<br>**注意：**R9是一个只读位，T8是一个只写位，读取的值与写入的值不同。 |
| 29<br>TXDIR | 单线模式下的TXD引脚方向<br>当LPUART被配置为单线半双工操作（LOOPS = RSRC = 1）时，该位决定了TXD引脚的数据方向。当清除TXDIR时，发射器将在接收器开始从TXD引脚接收数据之前完成对当前字符（如果有的话）的接收。<br>0b - TXD引脚在单线模式下是一个输入。1b - TXD引脚在单线模式下是一个输出。 |
| 28<br>玉米 | 发送数据反转<br>设置该位可以颠倒传输数据输出的极性。<br>**注意：**设置TXINV可以反转所有情况下的TXD输出：数据位、起始位和停止位、中断和空闲。<br>0b - 发送数据不倒置。1b - 传输数据反转。 |
| 27<br>橙子 | 超限中断启用<br>该位使超限标志（OR）产生硬件中断请求。<br>0b - 禁用OR中断；使用轮询。<br>1b - 当OR被设置时请求硬件中断。 |
| 26<br>NEIE | 噪声错误中断启用<br>该位使噪声标志（NF）产生硬件中断请求。<br>0b - 禁用NF中断；使用轮询。<br>1b - 当NF被设置时请求硬件中断。 |

| | |
|---|---|
| 25<br>基金会 | 帧错误中断启用<br>该位使成帧错误标志（FE）产生硬件中断请求。<br>0b - 禁用FE中断；使用轮询。<br>1b - 当FE被设置时请求硬件中断。 |
| 24<br>PEIE | 奇偶校验错误中断启用<br>该位使奇偶校验错误标志（PF）产生硬件中断请求。<br>0b - 禁用PF中断；使用轮询）。<br>1b - 当PF被设置时请求硬件中断。 |
| 23 | 发送中断启用 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| TIE | 启用STAT[TDRE]，以产生中断请求。<br>0b - 禁止来自TDRE的硬件中断；使用轮询。1b - 当TDRE标志为1时请求硬件中断。 |
| 22<br>TCIE | 传输完成中断启用，用于<br>TCIE启用传输完成标志，TC，以产生中断请求。<br>0b - 禁用TC的硬件中断；使用轮询。1b - 当TC标志为1时请求硬件中断。 |
| 21<br>RIE | 接收器中断启用<br>启用STAT[RDRF]，以产生中断请求。<br>0b - 禁止来自RDRF的硬件中断；使用轮询。1b - 当RDRF标志为1时请求硬件中断。 |
| 20<br>尹丽萍 | 闲置线路中断启用<br>ILIE启用空闲线路标志STAT[IDLE]，以产生中断请求。<br>0b - 禁用来自IDLE的硬件中断；使用轮询。1b - 当IDLE标志为1时请求硬件中断。 |
| 19<br>TE | 发射器启用<br>启用LPUART发射器。通过清除和设置TE，TE也可用于排队等待空闲的前奏。当TE被清除时，该寄存器位将读作1，直到发送器完成当前字符，TXD引脚被三态化。<br>通过在设置DATA[FRETSC]和DATA[R9T9]的情况下向发送FIFO写入一个空闲字符也可以排队。<br>0b - 发送器禁用。1b - 发射器启用。 |
| 18<br>雇员 | 接收器启用<br>启用LPUART接收器。当RE被写为0时，该寄存器位将读为1，直到接收器完成对当前字符（如果有的话）的接收。<br>0b - 接收器禁用。<br>1b - 接收器启用。 |
| 17<br>RWU | 接收器唤醒控制<br>可以设置该字段，使LPUART接收器处于待机状态。当发生RWU事件时，即CTRL[WAKE]清零时的IDLE事件或CTRL[WAKE]设置时的地址匹配与STAT[RWUID]清零，RWU自动清零。<br>**注意：**如果通道目前不是空闲的，RWU必须只在CTRL[WAKE]=0（空闲时唤醒）时设置。这可以通过STAT[RAF]来确定。如果该标志被设置为唤醒IDLE事件，而通道已经处于空闲状态，那么LPUART有可能会丢弃数据。这是因为在检测到IDLE后，必须接收到数据或检测到LIN中断，然后才允许IDLE被重新置位。<br>0b - 接收器正常运行。<br>1b - LPUART接收器处于待机状态，等待唤醒条件。 |

| | |
|---|---|
| 16<br>SBK | 发送休息<br>向SBK写一个1，然后写一个0，在传输数据流中排队一个中断字符。只要SBK被设置，9到13位的额外中断字符，或者12到15位（如果LPUART_STAT[BRK13]被设置）的逻辑0位时间将被排队。根据SBK的设置和清除相对于当前正在传输的字符的时间，在软件清除SBK之前，第二个中断字符可能被排队。<br>在DATA[FRETSC]设置和DATA[R9T9]清零的情况下，也可以通过向发送FIFO写入单个中断字符来进行排队。<br>0b - 发射器正常运行。<br>1b - 排队等待发送的中断字符。 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| 15<br>闳晖 | 匹配1中断启用<br>0b - 禁用 MA1F 中断 1b - 启用 MA1F 中断 |
| 14<br>辽宁沈阳 | 匹配2中断启用<br>0b - 禁用MA2F中断 1b - 启用MA2F中断 |
| 13-12<br>- | 保留 |
| 11<br>M7 | 7比特模式选择<br>只有在发射器和接收器都被禁用的情况下才可以改变这个位。<br>0b - 接收器和发射器使用8位至10位数据字符。 1b - 接收器和发射器使用7位数据字符。 |
| 10-8<br>身份证号：IDLECFG | 闲置配置<br>配置在IDLE标志被设置之前必须收到的空闲字符数。<br>000b - 1个空闲字符 001b - 2个空闲字符 010b - 4个空闲字符 011b - 8个空闲字符 100b - 16个空闲字符 101b - 32个空闲字符 110b - 64个空闲字符 111b - 128个空闲字符 |
| 7<br>厕所 | 循环模式选择<br>当LOOPS被设置时，RXD引脚与LPUART断开，发射器的输出在内部与接收器的输入相连。发送器和接收器必须被启用才能使用循环功能。<br>0b - 正常操作 - RXD和TXD使用单独的引脚。<br>1b - 循环模式或单线模式，发射器输出在内部连接到接收器输入（见RSRC位）。 |
| 6<br>窦泽恩 | 启用打盹功能<br>0b - LPUART在Doze模式下启用。 1b - 在休眠模式下禁用LPUART。 |
| 5<br>RSRC | 接收器来源选择<br>除非LOOPS字段被设置，否则这个字段没有任何意义和作用。当LOOPS被设置时，RSRC字段决定了接收器移位寄存器的输入来源。<br>0b--只要LOOPS被设置，RSRC被清零，选择内部回环模式，LPUART不使用RXD引脚。<br>1b - 单线LPUART模式，TXD引脚连接到发送器输出和接收器输入。 |
| 4<br>M | 9位或8位模式选择<br>0b - 接收器和发射器使用8位数据字符。 1b - 接收器和发射器使用9位数据字符。 |

| | |
|---|---|
| 3<br><br>唤醒 | 接收器唤醒方式选择<br><br>决定当RWU=1时，哪种情况会唤醒LPUART。<br>• 在收到的数据字符的停止位（或启用奇偶校验时的奇偶校验位前的位）前的地址标记，或<br>• 接收引脚输入信号的空闲状态。<br><br>0b - 配置RWU用于空闲线唤醒。<br>1b - 配置具有地址标记唤醒功能的RWU。 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| 2<br>劳工局 | 闲置线路类型选择<br>决定接收器何时开始将逻辑1作为空闲字符位计数。计数在有效的起始位或停止位之后开始。如果在开始位之后开始计数，那么在停止位之前的一串逻辑1会导致空闲字符的错误识别。在停止位之后开始计数可以避免错误的空闲字符识别，但需要适当的同步传输。<br>**注意：**如果LPUART被编程为ILT=1，那么在接收到一个停止位后会自动转换一个逻辑0，因此会重置空闲计数。<br>0b - 闲置字符位数在起始位后开始计算。1b - 空闲字符位数在停止位后开始。 |
| 1<br>塑胶跑道 | 启用奇偶校验<br>启用硬件奇偶校验的生成和检查。当启用奇偶校验时，紧接在停止位之前的位被视为奇偶校验位。<br>0b - 没有硬件奇偶校验生成或检查。1b - 启用奇偶校验。 |
| 0<br>淘宝网 | 奇偶校验类型<br>只要奇偶校验被启用（PE=1），该位就可以选择偶数或奇数校验。奇数奇偶校验意味着数据字符中1的总数，包括奇偶校验位，是奇数。偶数奇偶校验意味着数据字符中的1的总数，包括奇偶校验位，是偶数。<br>0b - 偶数奇偶性。1b - 奇数奇偶性。 |

## 48.3.1.9 LPUART数据寄存器（DATA）

### 48.3.1.9.1 偏移

| 注册 | 偏移 |
|---|---|
| 数据 | 1Ch |

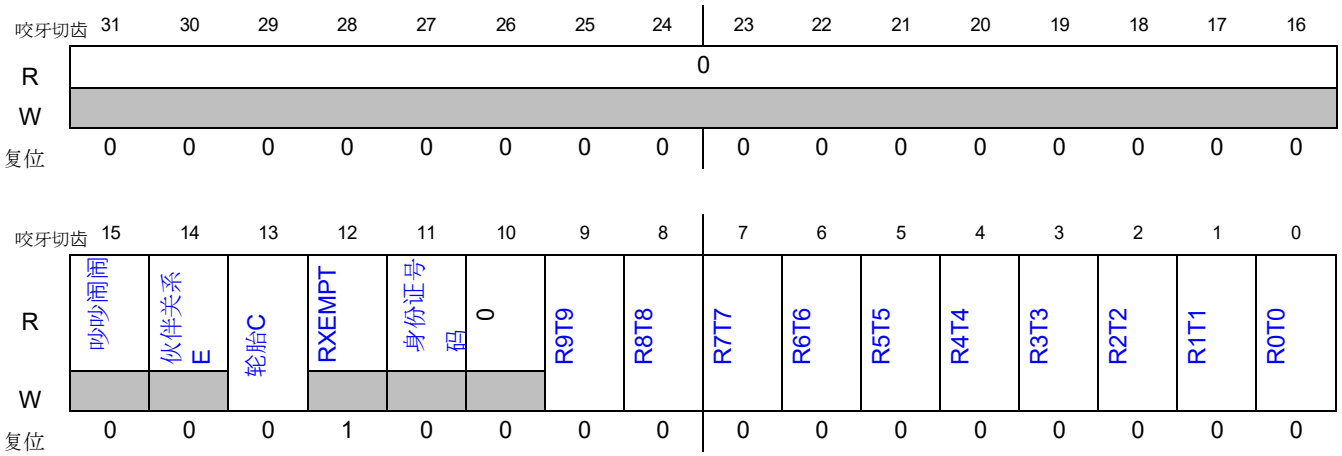
### 48.3.1.9.2 职能

**注意事项**

这个寄存器是两个独立的寄存器。读取时返回只读的接收数据缓冲区的内容，写入时进入只写的发送数据缓冲区。

该寄存器的读和写也参与了一些LPUART状态标志的自动标志清除机制。

### 48.3.1.9.3 图示




### 48.3.1.9.4 领域

| 场地 | 职能 |
|---|---|
| 31-16<br>- | 保留 |
| 15<br>吵吵闹闹 | 吵吵闹闹<br>DATA[R9:R0]中包含的当前接收的数据字是带噪音的。<br>0b - 收到的数据字没有噪音。1b - 收到的数据带有噪音。 |
| 14<br>伙伴关系E | 伙伴关系E<br>当前收到的包含在DATA[R9:R0]中的数据字被接收，并有一个奇偶校验错误。<br>0b - 收到的数据字没有奇偶校验错误。1b - 收到的数据字有一个奇偶校验错误。 |
| 13<br>基金会 | 帧错误/发送特殊字符<br>对于读，表示当前接收到的包含在DATA[R9:R0]中的数据字被接收到，并有一个帧错误。对于写，表示要传输一个中断或空闲字符，而不是DATA[T9:T0]中的内容。T9在0时用于表示中断字符，在1时用于表示空闲字符，DATA[T8:T0]的内容应该是0。<br>0b - 读取时收到的数据字没有帧错误，或写入时传送一个正常的字符。<br>1b - 收到的数据字有帧错误，或在发送时发送一个空闲或中断字符。 |
| 12<br>RXEMPT | 接收缓冲区空<br>当接收缓冲器中没有数据时发出警报。该字段不考虑接收移位寄存器中的数据。<br>0b - 接收缓冲区包含有效数据。<br>1b - 接收缓冲区是空的，读取时返回的数据是无效的。 |

| | |
|---|---|
| 11<br>身份证号码 | 闲置线<br>表示接收器线路在收到DATA[9:0]中的字符之前是空闲的。与IDLE标志不同的是，当接收机首次启用时，该位可以为收到的第一个字符设置。<br>0b - 接收者在收到该字符之前没有闲置。1b - 接收者在收到这个字符之前是空闲的。 |

*表继续在下一页...*

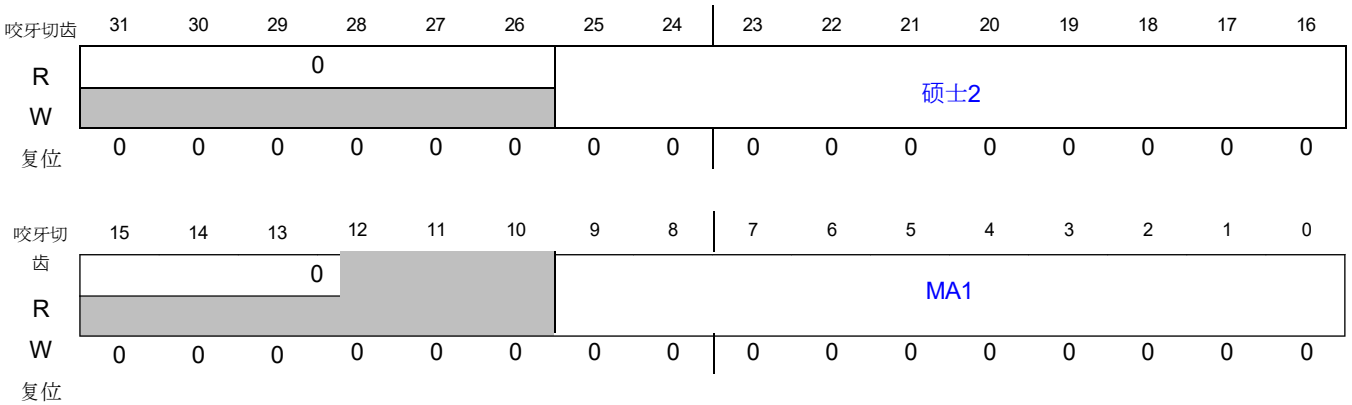
| 48.3.1.9.3 场地 | 图示 职能 |
| --- | --- |
| 10<br>- | 保留 |
| 9<br>R9T9 | R9T9<br>读取接收数据缓冲区9或写入发送数据缓冲区9。 |
| 8<br>R8T8 | R8T8<br>读取接收数据缓冲区8或写入发送数据缓冲区8。 |
| 7<br>R7T7 | R7T7<br>读取接收数据缓冲区7或写入发送数据缓冲区7。 |
| 6<br>R6T6 | R6T6<br>读取接收数据缓冲区6或写入发送数据缓冲区6。 |
| 5<br>R5T5 | R5T5<br>读取接收数据缓冲区5或写入发送数据缓冲区5。 |
| 4<br>R4T4 | R4T4<br>读取接收数据缓冲区4或写入发送数据缓冲区4。 |
| 3<br>R3T3 | R3T3<br>读取接收数据缓冲区3或写入发送数据缓冲区3。 |
| 2<br>R2T2 | R2T2<br>读取接收数据缓冲区2或写入发送数据缓冲区2。 |
| 1<br>R1T1 | R1T1<br>读取接收数据缓冲区1或写入发送数据缓冲区1。 |
| 0<br>R0T0 | R0T0<br>读取接收数据缓冲区0或写入发送数据缓冲区0。 |

## 48.3.1.10 LPUART匹配地址寄存器（MATCH）

### 48.3.1.10.1 偏移

| 注册 | 偏移 |
| --- | --- |
| 匹配 | 20h |

### 48.3.1.10.2 图示




### 48.3.1.10.3 领域

| 场地 | 职能 |
|---|---|
| 31-26<br>- | 保留 |
| 25-16<br>硕士2 | 匹配地址2<br>当最重要的位被设置，相关的BAUD[MAEN]位被设置时，MA1和MA2寄存器与输入数据地址进行比较。如果发生匹配，下面的数据将被传送到数据寄存器。如果匹配失败，下面的数据被丢弃。软件只能在相关的BAUD[MAEN]位清零时写一个MA寄存器。 |
| 15-10<br>- | 保留 |
| 9-0<br>MA1 | 匹配地址1<br>当最重要的位被设置，相关的BAUD[MAEN]位被设置时，MA1和MA2寄存器与输入数据地址进行比较。如果发生匹配，下面的数据将被传送到数据寄存器。如果匹配失败，下面的数据被丢弃。软件只能在相关的BAUD[MAEN]位清零时写一个MA寄存器。 |

## 48.3.1.11 LPUART调制解调器IrDA寄存器（MODIR）。

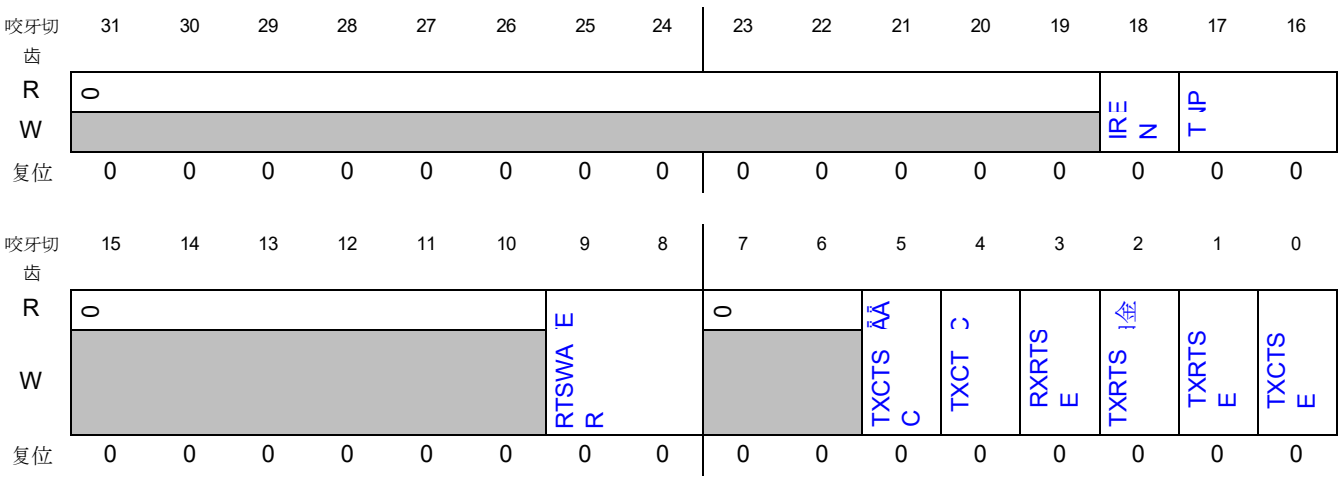
### 48.3.1.11.1 偏移

| 注册 | 偏移 |
|---|---|
| MODIR | 24h |

#### 48.3.1.11.2 职能

MODEM寄存器控制设置调制解调器配置的选项。

## 48.3.1.11.3 图示




## 48.3.1.11.4 领域

| 场地 | 职能 |
|---|---|
| 31-19<br>- | 保留 |
| 18<br>IREN | 红外线启用<br>启用/禁用红外调制/解调功能。该位只有在发射器和接收器都被禁用时才能改变。<br>0b - 禁用IR。<br>1b - 启用IR。 |
| 17-16<br>TNP | 发射器窄脉冲<br>配置LPUART在启用IR时是否传输1/OSR、2/OSR、3/OSR或4/OSR窄脉冲。只有当发射器和接收器都被禁用时，才应改变该位。<br>红外脉冲宽度应配置为小于超采样率的一半。常见的脉冲宽度是比特长度的3/16、1/16、1/32或1/4。这些可以通过选择适当的过采样率和脉冲宽度来配置。<br>00b - 1/OSR。<br>01b - 2/OSR。<br>10b - 3/OSR。<br>11b - 4/OSR。 |
| 15-10<br>- | 保留 |

| | |
|---|---|
| 9-8<br>RTSWATER | 接收RTS配置<br>配置RX RTS_B输出的断言和否定。当接收FIFO中的空字数大于或等于RTSWATER时，RX RTS_B输出被否定。如果RTSWATER被配置为零，RTS_B在接收FIFO满时就会被否定。对于RX RTS_B的产生，当检测到一个起始位时，接收FIFO中的字数会更新。这支持在RTS_B否定和外部发射器停止传输之间的额外延时。如果RX RTS_B和地址/数据匹配都被启用，那么如果没有匹配，RTS_B可以在一个字符的末尾发出信号。 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | 这个字段只有在接收器被禁用时才可以改变。 |
| 7-6<br>- | 保留 |
| 5<br>陕西省商务厅 | 发送CTS来源<br>配置CTS输入的来源。0b - CTS输入是<br>CTS_B引脚。<br>1b - CTS输入是倒置的接收机匹配结果。 |
| 4<br>辽宁沈阳 | 发送CTS配置<br>配置是否在每个字符开始时检查CTS状态或仅在发射器空闲时检查。<br>0b - CTS输入在每个字符开始时被采样。1b - 发送器空<br>闲时对CTS输入进行采样。 |
| 3<br>荣事达 | 启用接收方请求发送功能<br>允许RTS输出控制发送设备的CTS输入，以防止接收器超限。该位只应在接收器被禁用时改变。<br>**注意：**不要同时设置RXRTSE和TXRTSE。<br>0b - 接收器对RTS没有影响。<br>1b - 如果接收机数据寄存器已满或检测到会导致接收机数据寄存器满的启动位，则RTS被置位。如果接收机数据寄存器未满，并且没有检测到会导致接收机数据寄存器满的启动位，则RTS被断言。 |
| 2<br>辽宁省 | 发送器请求发送的极性<br>控制发射器RTS的极性。TXRTSPOL不影响接收器RTS的极性。除非TXRTSE被设置，否则RTS将保持负的低电平状态。该位只应在发射器被禁用时改变。<br>0b - 发送器RTS为低电平有效。1b -<br>发送器RTS为高电平有效。 |
| 1<br>玉米淀粉酶 | 发送器请求发送功能启用<br>控制传输前后的RTS。该位只应在发射器被禁用时改变。<br>0b - 发送器对RTS没有影响。<br>1b - 当一个字符被放入一个空的发送器数据缓冲器时，RTS在传输起始位之前发出一个比特时间。在发送器数据缓冲器和移位寄存器中的所有字符全部发送完毕后，包括最后一个停止位，RTS会发出一个比特时间的中断信号。 |
| 0<br>辽宁省 | 发送器清空发送功能启用<br>TXCTSE控制发射器的操作。TXCTSE的设置可以独立于TXRTSE和RXRTSE的状态。<br>0b - CTS对发射器没有影响。<br>1b - 启用清除发送操作。发送器在每次准备发送一个字符时检查CTS的状态。如果CTS被确认，字符就被发送。如果CTS被置位，信号TXD保持在标记状态，传输被延迟，直到CTS被置位。当一个字符被发送时，CTS的变化并不影响它的传输。 |

## 48.3.1.12 LPUART FIFO寄存器（FIFO）

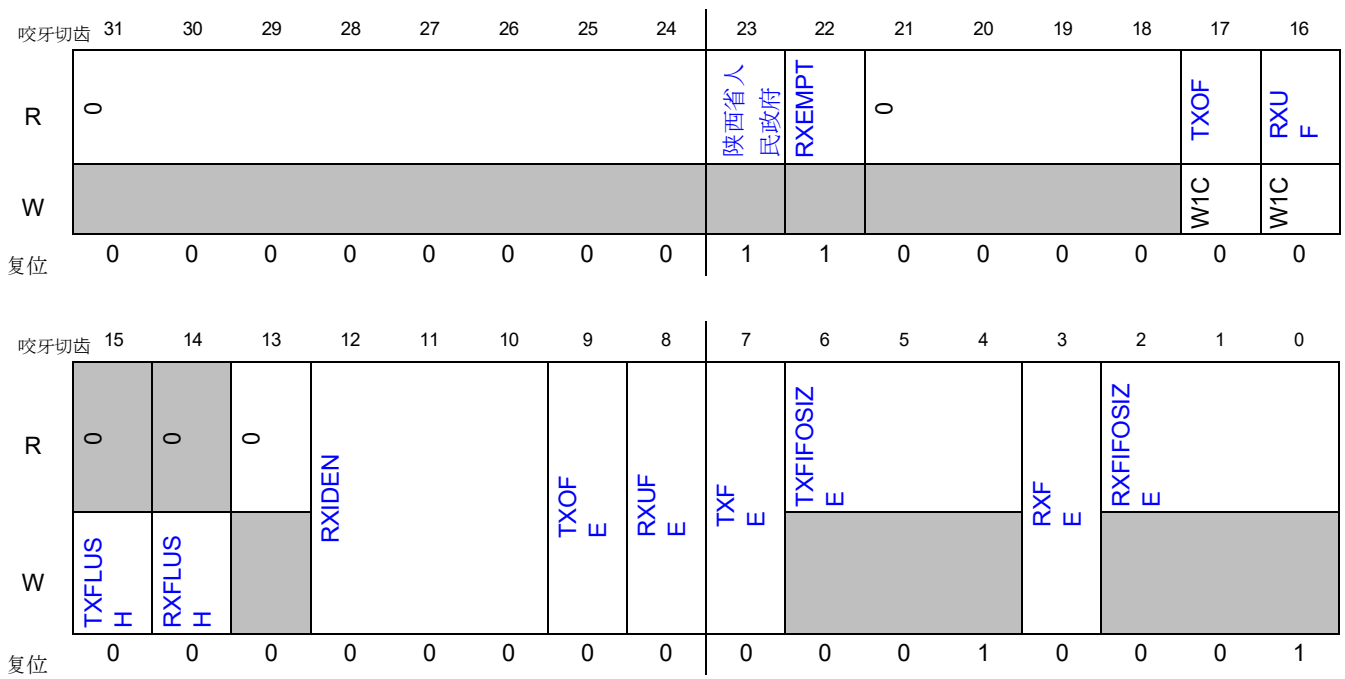
### 48.3.1.12.1 偏移

| 注册 | 偏移 |
|---|---|
| FIFO | 28h |

### 48.3.1.12.2 职能

这个寄存器为编程者提供了打开和关闭FIFO功能的能力。它还提供了已经实现的FIFO的大小。这个寄存器可以在任何时候被读取。只有当CTRL[RE]和CTRL[TE]被清零/未设置，并且数据缓冲区/FIFO为空时，才能写入该寄存器。

### 48.3.1.12.3 图示




### 48.3.1.12.4 领域

| 场地 | 职能 |
|---|---|
| 31-24<br>- | 保留 |
| 23<br>陕西师范大学 | 发送缓冲区/FIFO空<br>当传输FIFO/缓冲区中没有数据时发出警报。该字段不考虑传输移位寄存器中的数据。<br>0b - 发送缓冲区不空。 |

表继续在下一页...

| 场地 | 职能 |
| --- | --- |
| | 1b - 发送缓冲区为空。 |
| 22<br>RXEMPT | 接收缓冲器/FIFO空<br>当接收FIFO/Buffer中没有数据时提示。该字段不考虑接收移位寄存器中的数据。<br>0b - 接收缓冲区不空。1b - 接收<br>缓冲区是空的。 |
| 21-18<br>- | 保留 |
| 17<br>TXOF | 发送器缓冲区溢出标志<br>表示写入传输缓冲区的数据比它能容纳的多。无论TXOFE的值是多少，这个字段都会发出指令。然而，只有当TXOFE被设置时，才会向主机发出中断。这个标志可以通过写1来清除。<br>0b - 自从上次清除该标志后，没有发生发送缓冲区溢出。<br>1b - 自从上次清除该标志后，至少发生了一次发送缓冲区溢出。 |
| 16<br>RXUF | 接收器缓冲区下溢标志<br>表示从接收缓冲区读取的数据比原来多。无论RXUFE的值是多少，这个字段都会发出指令。然而，只有当RXUFE被设置时，才会向主机发出中断。这个标志可以通过写1来清除。<br>0b - 自上次清除该标志后，没有发生接收缓冲区下溢。<br>1b - 自上次清除该标志以来，至少发生了一次接收缓冲区下溢。 |
| 15<br>TXFLUSH | 发送FIFO/缓冲区刷新<br>写入这个字段会导致所有存储在发送FIFO/缓冲区的数据被刷新。这并不影响传输移位寄存器中的数据。<br>0b - 不发生冲刷操作。<br>1b - 发送FIFO/Buffer中的所有数据被清除掉。 |
| 14<br>荣事达 | 接收FIFO/缓冲区刷新<br>写入这个字段会导致所有存储在接收FIFO/缓冲区的数据被刷新。这并不影响在接收移位寄存器中的数据。<br>0b - 不发生冲刷操作。<br>1b - 接收FIFO/缓冲区中的所有数据被清空。 |
| 13<br>- | 保留 |
| 12-10<br>RXIDEN | 接收器空闲启用<br>当设置时，当接收器闲置若干空闲字符且FIFO不为空时，启用RDRF的宣称。<br>000b - 当接收器处于空闲状态时，由于FIFO被部分填满而禁用RDRF断言。<br>001b - 当接收器闲置1个字符时，由于FIFO部分填满而启用RDRF断言。010b - 当接收机空闲2个字符时，由于部分填充的FIFO而使RDRF断言生效。011b - 当接收器空闲4个字符时，由于FIFO部分填满而使RDRF断言生效。100b - 当接收器空闲8个字符时，由于FIFO部分填满而使RDRF断言生效。101b - 当接收机闲置16个字符时，由于FIFO部分填满而使RDRF断言生效。110b - 当接收器空闲32个字符时，由于FIFO部分填满而使RDRF断言生效。111b - 当接收器闲置64个字符时，由于部分填充的FIFO而使RDRF断言生效。 |

| | |
|---|---|
| 9<br>ĀĀĀ | 发送FIFO溢出中断启用<br>当这个字段被设置时，TXOF标志会产生一个中断给主机。<br>0b - TXOF标志不产生对主机的中断。1b - TXOF标志产生一个对主机的中断。 |
| 8 | 接收FIFO下溢中断启用 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| RXUFE | 当这个字段被设置时，RXUF标志会产生一个中断给主机。<br>0b - RXUF标志不产生对主机的中断。1b - RXUF标志产生一个对主机的中断。 |
| 7<br>ÃÃÃ | 发送FIFO启用<br>当这个字段被设置时，发送缓冲区的内置FIFO结构被启用。FIFO结构的大小由TXFIFOSIZE表示。如果这个字段没有设置，无论 TXFIFOSIZE 的值是多少，发送缓冲器都作为一个深度为一个数据字的 FIFO 运行。在改变这个字段之前，CTRL[TE]和CTRL[RE]都必须被清零。<br>0b - 发送FIFO未被启用。缓冲区深度为1。（传统支持）。1b - 发送FIFO被启用。缓冲区的深度由TXFIFOSIZE指示。 |
| 6-4<br>呼叫中心 | 发送FIFO。缓冲区深度<br>发送缓冲器中可存储的最大发送数据字数。这个字段是只读的。<br>000b - 发送FIFO/缓冲区深度=1个数据字。001b - 发送FIFO/缓冲区深度=4个数据字。010b - 发送FIFO/缓冲区深度=8个数据字。011b - 发送FIFO/缓冲区深度=16个数据字。100b - 发送FIFO/缓冲区深度=32个数据字。101b - 发送FIFO/缓冲区深度=64个数据字。110b - 发送FIFO/缓冲区深度=128个数据字。111b - 发送FIFO/缓冲区深度=256个数据字 |
| 3<br>荣事达 | 接收FIFO启用<br>当这个字段被设置时，接收缓冲区的内置FIFO结构被启用。FIFO结构的大小由RXFIFOSIZE字段表示。如果这个字段没有设置，无论RXFIFOSIZE的值是多少，接收缓冲区都作为一个深度为一个数据字的FIFO运行。在改变这个字段之前，CTRL[TE]和CTRL[RE]都必须被清除。<br>0b - 接收FIFO未被启用。(传统支持) 1b - 启用接收FIFO。缓冲区的深度由RXFIFOSIZE指示。 |
| 2-0<br>呼叫中心 | 接收FIFO。缓冲区深度<br>在发生超限之前可以存储在接收缓冲区的最大接收数据字数。这个字段是只读的。<br>000b - 接收FIFO/缓冲区深度=1个数据字。001b - 接收FIFO/缓冲区深度=4个数据字。010b - 接收FIFO/缓冲区深度=8个数据字。011b - 接收FIFO/缓冲区深度=16个数据字。100b - 接收FIFO/缓冲区深度=32个数据字。101b - 接收FIFO/缓冲区深度=64个数据字。110b - 接收FIFO/缓冲区深度=128个数据字。<br>111b - 接收FIFO/缓冲区深度=256个数据字。 |

## 48.3.1.13 LPUART水印寄存器（WATER）

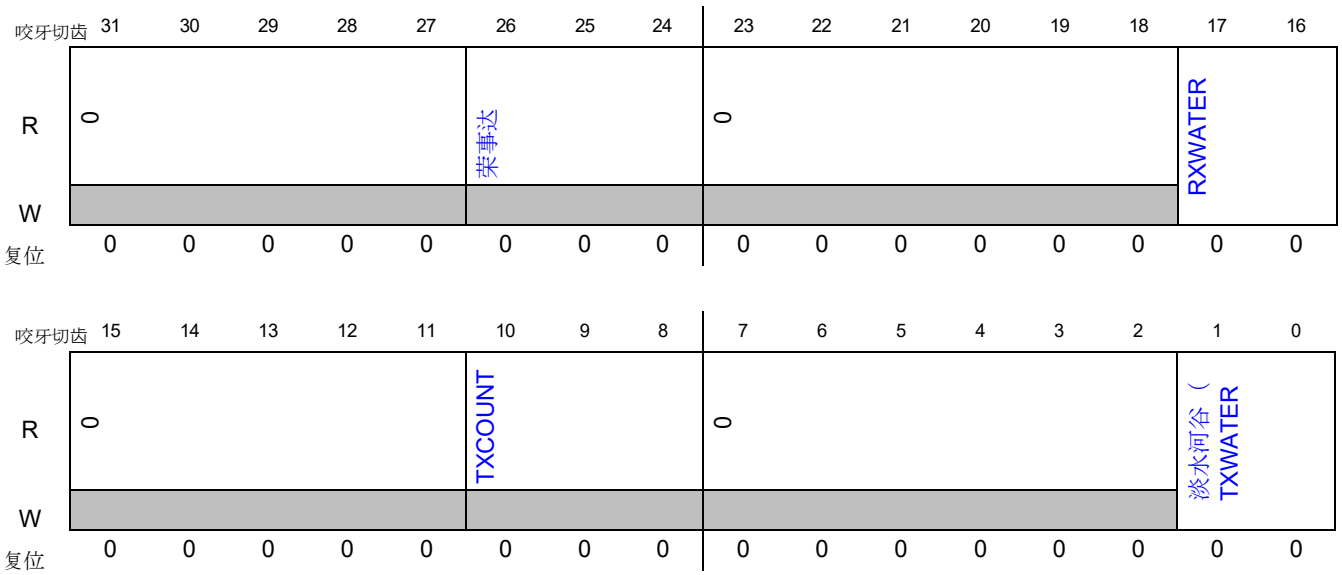
## 48.3.1.13.1 偏移

| 注册 | 偏移 |
| --- | --- |
| 水 | 2Ch |

### 48.3.1.13.2 职能

这个寄存器提供了设置一个可编程阈值的能力，用于通知需要额外的传输数据。
这个寄存器可以在任何时候被读取，但必须在CTRL[TE]没有被设置时才能写入。

### 48.3.1.13.3 图示




### 48.3.1.13.4 领域

| 场地 | 职能 |
|---|---|
| 31-27<br>- | 保留 |
| 26-24<br>荣事达 | 接收计数器<br>该寄存器中的值表示接收FIFO/缓冲区中的数据字数。如果一个数据字正在被接收，也就是在接收移位寄存器中，它不包括在计数中。这个值可以和FIFO[RXFIFOSIZE]一起使用，计算出接收FIFO/缓冲区中还剩下多少空间。 |
| 23-18<br>- | 保留 |
| 17-16<br>RXWATER | 接收水印<br>当接收FIFO/缓冲区的数据字数大于该寄存器字段的值时，将产生一个中断或DMA请求。为了正确操作，RXWATER中的值必须设置为小于FIFO[RXFIFOSIZE]和FIFO[RXFE]所指示的接收FIFO/缓冲区大小，并且必须大于0。 |
| 15-11<br>- | 保留 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| 10-8<br>TXCOUNT | 发送计数器<br>该寄存器中的值表示传输FIFO/缓冲区中的数据字数。如果一个数据字正在传输，也就是在传输移位寄存器中，它不包括在计数中。这个值可以和FIFO[TXFIFOSIZE]一起使用，以计算发送FIFO/缓冲区中还剩下多少空间。 |
| 7-2<br>- | 保留 |
| 1-0<br>淡水河谷（<br>TXWATER | 发送水印<br>当发送FIFO/缓冲区中的数据字数等于或小于该寄存器字段的值时，将产生一个中断或DMA请求。为了正确操作，TXWATER中的值必须设置为小于FIFO[TXFIFOSIZE]和FIFO[TXFE]所指示的发送缓冲器/FIFO大小。 |

## 48.4 职能描述

LPUART支持全双工、异步、NRZ串行通信，包括一个波特率发生器、发射器和接收器块。发送器和接收器独立工作，尽管它们使用相同的波特率发生器。下面介绍LPUART的每个模块。

### 48.4.1 波特率的产生

波特率发生器中的一个13位模数计数器得出接收器和发射器的波特率。写入SBR[12:0]的1至8191的值决定了异步LPUART波特时钟的波特时钟除数。SBR位在LPUART波特率寄存器BDH和BDL中。波特率时钟驱动接收器，而发射器则由比特时钟驱动，比特时钟由波特率时钟除以过采样率产生。根据过采样率的不同，接收器的采集速率为每比特时间4到32个样本。

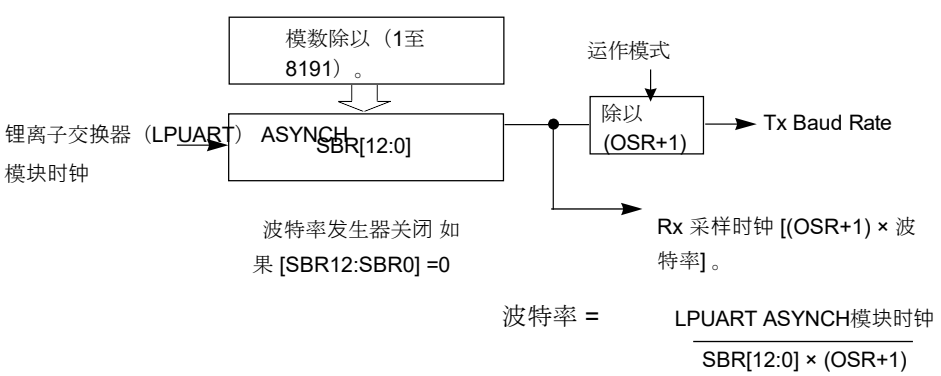


图48-3. LPUART波特率的生成

波特率的产生受制于两个错误来源。

- 异步LPUART波特时钟的整数除法可能无法得到准确的目标频率。
- 与异步的LPUART波特时钟同步会导致相位偏移。

波特率的产生是一个自由运行的计数器，只要发射器或接收器被启用就会继续运行。只要发射器被启用，发射器位时钟就会继续，每个传输的字符将与发射器位时钟的下一个边缘对齐。

### 48.4.2 发射器功能描述

本节介绍了LPUART发送器的总体框图，以及发送中断和空闲字符的专门功能。

发送器输出（TXD）空闲状态默认为逻辑高，CTRL[TXINV]在复位后被清除。通过设置CTRL[TXINV]，发射器输出被反转。通过设置CTRL[TE]位，发射器被启用。这将排队等待一个空闲状态的前导字符，该字符是一个完整的字符帧。然后发射器保持空闲状态，直到发射数据缓冲器中的数据可用。程序通过写入LPUART数据寄存器将数据存储到发送数据缓冲区。

LPUART发送器的核心元件是发送移位寄存器，其长度为9位至13位，取决于CTRL[M]、CTRL[M7]、BAUD[M10]和BAUD[SBNS]控制位中的设置。在本节的其余部分，假设CTRL[M]、CTRL[M7]、BAUD[M10]和BAUD[SBNS]被清零，选择正常的8位数据模式。在8位数据模式下，移位寄存器保存一个起始位、八个数据位和一个停止位。当发送移位寄存器可用于新字符时，发送数据寄存器中等待的值被转移到移位寄存器中，与波特率时钟同步，发送数据寄存器空（STAT[TDRE]）状态标志被设置，表示另一个字符可以被写入LPUART_DATA的发送数据缓冲器。

如果在TXD引脚移出一个停止位后，发送数据缓冲器中没有新的字符等待，发送器就会设置发送完成标志，并进入空闲模式，TXD为高电平，等待更多的字符发送。

向CTRL[TE]写0并不能立即使发射机失效。当前正在进行的发送活动必须首先完成（这可能包括一个数据字符、空闲字符或中断字符），尽管发送器不会开始发送另一个字符。

#### 48.4.2.1 发送中断和排队空闲

LPUART_CTRL[SBK]位发送中断字符，最初是用来吸引老式电传接收机的注意。断点字符是一个完整的逻辑0的字符时间，9位到12位的时间，包括开始和停止位。通过设置LPUART_STAT[BRK13]，可以启用更长的13位中断时间。通常情况下，程序会等待LPUART_STAT[TDRE]成为设置，以表明信息的最后一个字符已移至发送移位器，写入1，然后向LPUART_CTRL[SBK]位写入0。这个动作排队等待一个中断字符，一旦移位器可用，就立即发送。如果在排队的中断字符进入移位器时，LPUART_CTRL[SBK]仍为1，与波特率时钟同步，则会排队发送一个额外的中断字符。当LPUART为接收设备时，接收到的中断字符为所有数据位中的0，并检测到一个成帧错误（LPUART_STAT[FE] = 1）。

也可以通过向LPUART_DATA寄存器写入第13位并清除数据位来传输中断字符。这支持将中断字符作为正常数据流的一部分来传输，也允许DMA传输中断字符。

当使用空闲线唤醒时，在消息之间需要一个完整的空闲字符时间（逻辑1）来唤醒任何沉睡的接收器。通常情况下，程序会等待LPUART_STAT[TDRE]成为设置，以表明消息的最后一个字符已移至发送移位器，然后写入0，再写入1到LPUART_CTRL[TE]位。这个动作使一个空闲字符排队，一旦移位器可用，就立即发送。只要移位器中的字符在LPUART_CTRL[TE]被清零时没有完成，LPUART发送器就不会实际释放TXD引脚的控制。

通过向LPUART_DATA寄存器写入第13位并设置数据位，也可以传输一个空闲字符。这支持将空闲字符作为正常数据流的一部分来传输，也允许DMA传输一个中断字符。

断字的长度受LPUART_STAT[BRK13]、LPUART_CTRL[M]、LPUART_CTRL[M7]、LPUART_BAUD[M10]和
LPUART_BAUD[SNBS]位，如下所示。

**表48-2.断裂字符长度**

| BRK13 | M | M10 | M7 | 证券时报 | 破解字符长度 |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 10位时代 |
| 0 | 0 | 0 | 0 | 1 | 11位时代 |
| 0 | 0 | 0 | 1 | 0 | 9位时代 |
| 0 | 0 | 0 | 1 | 1 | 10位时代 |
| 0 | 1 | 0 | X | 0 | 11位时代 |

表继续在下一页...

表48-2.断裂字符长度（续）

| BRK13 | M | M10 | M7 | 证券时报 | 破解字符长度 |
|---|---|---|---|---|---|
| 0 | 1 | 0 | X | 1 | 12位时代 |
| 0 | X | 1 | X | 0 | 12位时代 |
| 0 | X | 1 | X | 1 | 13位时代 |
| 1 | 0 | 0 | 0 | 0 | 13位时代 |
| 1 | 0 | 0 | 0 | 1 | 13位时代 |
| 1 | 0 | 0 | 1 | 0 | 12位时代 |
| 1 | 0 | 0 | 1 | 1 | 12位时代 |
| 1 | 1 | 0 | X | 0 | 14位时代 |
| 1 | 1 | 0 | X | 1 | 14位时代 |
| 1 | X | 1 | X | 0 | 15位时代 |
| 1 | X | 1 | X | 1 | 15位时代 |

### 48.4.2.2 硬件流量控制

发送器支持硬件流量控制，通过对CTS_B的值进行门控传输。如果清空发送操作被启用，当CTS_B被断言时，字符被传输。如果CTS_B在传输过程中被去掉，而接收器数据缓冲器中还有字符，那么移位寄存器中的字符就会被发送，TXD保持在标记状态，直到CTS_B被重新置位。CTS_B引脚的中断时间必须超过一个比特周期，以保证在发射器空闲有数据要发送时开始新的传输。

如果禁止清空发送操作，发射器将忽略CTS_B的状态。

即使同一个LPUART接收器的RTS_B信号被禁用，发射器的CTS_B信号也可以被启用。

### 48.4.2.3 收发器驱动器启用

发送器可以使用RTS_B作为外部收发器驱动器的启用信号。详见使用RTS_B的收发器驱动器启用。如果请求发送操作被启用，当一个字符被放入一个空的发送器数据缓冲区时，RTS_B在开始位被传送之前会有一个比特时间。在发送器数据缓冲区有任何字符的整个时间内，RTS_B都保持正向。在发送器数据缓冲区和移位寄存器中的所有字符全部结束后，RTS_B会在一个比特的时间内发出中断信号。

包括最后一个停止位。传送一个中断字符也会断言RTS_B，断言和解除断言的时间与传送器数据缓冲区内有一个字符相同。

发送器的RTS_B信号只有在发送器被启用时才会发出信号。然而，发射器的RTS_B信号不受CTS_B信号的影响。RTS_B信号将保持有效，直到传输完成，即使发射器在数据传输的中途被禁用。

### 48.4.2.4 使用RTS_B启用收发器驱动器

RS-
485是一种多降通信协议，其中LPUART收发器的驱动器是3位的，除非LPUART正在驱动。发射器可使用RTS_B信号来启用收发器的驱动器。RTS_B的极性可以与收发器的驱动使能信号的极性相匹配。

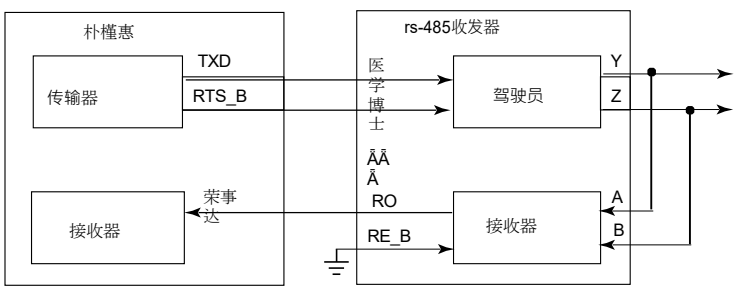



**图48-4.使用RTS_B启用收发器驱动器**

在图中，接收器的使能信号被断言。这种连接的另一种选择是将RTS_B同时连接到DE和RE_B。在驱动时，收发器的接收器被禁用。在这段时间内，一个上拉器可以将RXD拉到一个非浮动值。通过在单线模式下操作LPUART，可以进一步完善这一方案，释放RXD引脚用于其他用途。

## 48.4.3 接收器功能描述

在本节中，接收器框图是整个接收器功能描述的指南。接下来，对用于重建接收机数据的数据采样技术进行了更详细的描述。最后，对接收器唤醒功能的不同变化进行了解释。

通过设置LPUART_STAT[RXINV]，接收器输入被反转。通过设置LPUART_CTRL[RE]位，接收器被启用。字符帧由逻辑0的起始位、7至10个数据位（msb或lsb在前）和逻辑1的一个或两个停止位组成。对于

关于7位、9位或10位数据模式的信息，请参阅数据模式。在本讨论的其余部分，假设LPUART配置为正常的8位数据模式。

在接收到停止位进入接收移位器后，只要接收数据寄存器尚未满，数据字符就会被传送到接收数据寄存器，接收数据寄存器满（LPUART_STAT[RDRF]）状态标志被设置。如果LPUART_STAT[RDRF]已经被设置，表明接收数据寄存器（缓冲区）已经满了，则超限（OR）状态标志被设置，新数据被丢失。由于LPUART接收器是双缓冲器，在LPUART_STAT[RDRF]被设置后，程序有一个完整的字符时间，然后必须读取接收数据缓冲器中的数据以避免接收器超限。

当程序检测到接收数据寄存器已满（LPUART_STAT[RDRF]=1）时，它通过读取LPUART_DATA获得接收数据寄存器中的数据。关于标志清除的详情，请参考中断和状态标志。

#### 48.4.3.1 数据抽样技术

LPUART接收器支持可配置的超采样率，即4倍至32倍的波特率时钟用于采样。接收器首先以超采样率乘以波特率进行逻辑电平采样，以搜索RXD串行数据输入引脚上的下降沿。下降沿被定义为在连续三个逻辑1采样后的一个逻辑0采样。超采样波特率时钟将比特时间分为4到32段，从1到OSR（其中OSR是配置的超采样比率）。当一个下降沿被定位时，在(OSR/2)、(OSR/2)+1和(OSR/2)+2处再取三个样本，以确保这是一个真正的起始位，而不仅仅是噪声。如果这三个采样点中至少有两个是0，那么接收器就认为它已经同步到了一个接收的字符。如果在接收器被认为是同步之前检测到另一个下降沿，接收器将从第一段重新开始采样。

然后，接收器在(OSR/2)、(OSR/2)+1和(OSR/2)+2处对每个比特时间进行采样，包括起始和停止比特，以确定该比特的逻辑电平。逻辑电平被解释为在该位时间内采取的大多数样本的电平。如果一个字符帧中任何位时间的任何样本，包括开始位和停止位，都不能与该位的逻辑电平一致，当接收的字符被传送到接收数据缓冲器时，噪声标志（LPUART_STAT[NF]）被设置。

当LPUART接收器被配置为在波特率时钟的两个边沿上采样时，每个接收位中的段数实际上增加了一倍（从1到OSR×2）。然后在OSR、OSR+1和OSR+2对起始位和数据位进行采样。对于4×到7×的过采样率，必须启用时钟两个边沿的采样，对于更高的过采样率则是可选的。

下降沿检测逻辑不断寻找下降沿。如果检测到一个边沿，采样时钟将重新同步到比特时间（除非重新同步被禁用）。这提高了接收器在存在噪声或不匹配的波特率时的可靠性。它不能改善最坏情况的分析，因为有些字符在字符帧的任何地方都没有任何额外的下降沿。

在成帧错误的情况下，只要收到的字符不是断裂字符，搜索下降沿的采样逻辑就会充满三个逻辑1的采样，这样就可以几乎立即检测到一个新的起始位。

### 48.4.3.2 接收器唤醒操作

接收器唤醒和接收器地址匹配是一种硬件机制，它允许LPUART接收器忽略用于不同接收器的信息中的字符。

在接收机唤醒期间，所有接收机都会评估每条信息的第一个字符，一旦它们确定该信息是为不同的接收机准备的，它们就会向接收机唤醒控制位（LPUART_CTRL[RWU]）写入逻辑1。当RWU位和LPUART_S2[RWUID]位被设置时，除了空闲位IDLE外，与接收器相关的状态标志被抑制设置，从而消除了处理不重要的消息字符的软件开销。在一条消息结束时，或在下一条消息开始时，所有接收机自动将LPUART_CTRL[RWU]强制为0，以便所有接收机及时唤醒，查看下一条消息的第一个（几个）字符。

在接收器地址匹配期间，地址匹配是在硬件中进行的，LPUART接收器将忽略所有不符合地址匹配要求的字符。

**表48-3.接收器唤醒选项**

| RWU | MA1 \| MA2 | MATCFG | 唤醒：RWUID | 接收器唤醒 |
|---|---|---|---|---|
| 0 | 0 | X | X | 正常运行 |
| 1 | 0 | 00 | 00 | 接收器在空闲线路上唤醒，IDLE标志未设置 |
| 1 | 0 | 00 | 01 | 接收器在空闲线路上唤醒，设置IDLE标志 |

*表继续在下一页...*

表48-3.接收器唤醒选项（续）

| RWU | MA1 \| MA2 | MATCFG | 唤醒：RWUID | 接收器唤醒 |
|---|---|---|---|---|
| 1 | 0 | 00 | 10 | 接收器在地址标记时唤醒 |
| 1 | 1 | 11 | X0 | 数据匹配时接收机唤醒 |
| 0 | 1 | 00 | X0 | 地址标记地址匹配，未设置IDLE标志的丢弃的字符 |
| 0 | 1 | 00 | X1 | 地址标记地址匹配，为被丢弃的字符设置IDLE标志 |
| 0 | 1 | 01 | X0 | 闲置线路地址匹配 |
| 0 | 1 | 10 | X0 | 地址匹配开启和地址匹配关闭，对丢弃的字符不设置IDLE标志 |
| 0 | 1 | 10 | X1 | 地址匹配开启和地址匹配关闭，丢弃的字符设置IDLE标志 |

#### 48.4.3.2.1 闲置线唤醒

当唤醒被清除时，接收器被配置为空闲线唤醒。在这种模式下，当接收器检测到空闲线水平的完整字符时间时，LPUART_CTRL[RWU]被自动清零。LPUART_CTRL[M]、LPUART_CTRL[M7]和LPUART_BAUD[M10]控制位选择7位至10位数据模式，LPUART_BAUD[SBNS]位选择1位或2位停止位数，决定了构成一个完整字符时间需要多少位的空闲时间，由于有起始位和停止位，所以是9至13位。

当LPUART_CTRL[RWU]为1且LPUART_STAT[RWUID]为0时，唤醒接收器的空闲条件不会设置LPUART_STAT[IDLE]标志。接收器唤醒后等待下一条信息的第一个数据字符，该信息设置LPUART_STAT[RDRF]标志，如果启用，则产生一个中断。当LPUART_STAT[RWUID]为1时，无论LPUART_CTRL[RWU]是0还是1，任何空闲条件都会设置LPUART_STAT[IDLE]标志，并在启用时产生中断。

空闲线类型（LPUART_CTRL[ILT]）控制位选择两种检测空闲线的方式之一。当
LPUART_CTRL[ILT]被清零时，空闲位计数器在开始位之后开始，因此停止位和字符结束时的任何逻辑 1 都会被计入满值。

空闲的字符时间。当LPUART_CTRL[ILT]被设置时，空闲位计数器在停止位时间之后才开始工作，因此空闲检测不受前一信息最后一个字符中数据的影响。

#### 48.4.3.2.2 地址标记唤醒

当LPUART_CTRL[WAKE]被设置时，接收器被配置为地址标记唤醒。在这种模式下，当接收器检测到接收到的字符的最有效位中的逻辑1时，LPUART_CTRL[RWU]被自动清零。当奇偶校验被启用时，第二个最重要的位被用于地址标记唤醒。

地址标记唤醒允许信息包含空闲字符，但要求保留一个位用于地址帧。地址帧的最重要位（或启用奇偶校验时的第二个最重要位）的逻辑1会清除LPUART_CTRL[RWU]位并设置LPUART_STAT[RDRF]标志。在这种情况下，即使接收器在这个字符的大部分时间内处于睡眠状态，也会收到带有地址标记位的字符。

#### 48.4.3.2.3 数据匹配唤醒

当LPUART_CTRL[RWU]被设置且LPUART_BAUD[MATCFG]等于11时，接收器被配置为数据匹配唤醒。在这种模式下，当接收器检测到与MATCH[MA1]字段相匹配的字符时（当BAUD[MAEN1]被设置时），或与MATCH[MA2]相匹配的字符时（当BAUD[MAEN2]被设置时），LPUART_CTRL[RWU]会自动清零。

#### 48.4.3.2.4 地址匹配操作

当LPUART_BAUD[MAEN1]或LPUART_BAUD[MAEN2]位被设置且LPUART_BAUD[MATCFG]等于00时，地址匹配操作被启用。在该功能中，由RXD引脚接收的最有效位（或启用奇偶校验时的第二最有效位）为逻辑1的字符被视为地址，并与相关的MATCH[MA1]或MATCH[MA2]字段进行比较。只有在比较结果匹配的情况下，该字符才会被传送到接收缓冲区，并且LPUART_STAT[RDRF]被设置。随后收到的所有字符的最有效位（或启用奇偶校验时的第二个最有效位）都被认为是与该地址相关的数据，并被传送到接收数据缓冲器。如果没有发生标记的地址匹配，则不向接收数据缓冲器传输，并且所有在最有效位（或启用奇偶校验时的第二最有效位）上有逻辑0的后续字符也被丢弃。如果LPUART_BAUD[MAEN1]和LPUART_BAUD[MAEN2]位都被否定，则接收器正常工作，所有收到的数据都传输到接收数据缓冲区。

对于MATCH[MA1]和MATCH[MA2]字段，地址匹配操作的功能相同。

- 如果只有LPUART_BAUD[MAEN1]和LPUART_BAUD[MAEN2]中的一个被断言，则标记的地址只与相关的匹配寄存器进行比较，只有在匹配时才将数据传输到接收数据缓冲器。
- 如果LPUART_BAUD[MAEN1]和LPUART_BAUD[MAEN2]被断言，则标记的地址将与两个匹配寄存器进行比较，只有在与任一寄存器匹配时才传输数据。

#### 48.4.3.2.5 空闲匹配操作

当LPUART_BAUD[MAEN1]或LPUART_BAUD[MAEN2]位被设置且LPUART_BAUD[MATCFG]等于01时，空闲匹配操作被启用。在该功能中，空闲线路条件后由RXD引脚接收的第一个字符被视为一个地址，并与相关的MA1或MA2寄存器进行比较。如果比较结果相符，该字符才会被传送到接收缓冲区，并设置LPUART_STAT[RDRF]。所有随后的字符都被认为是与地址相关的数据，并被传送到接收数据缓冲区，直到检测到下一个空闲线路条件。如果没有发生地址匹配，则不向接收数据缓冲器传输，所有后续帧直到下一个空闲条件也被丢弃。如果LPUART_BAUD[MAEN1]和LPUART_BAUD[MAEN2]位都被否定，则接收器正常工作，所有收到的数据都被传输到接收数据缓冲器。

空闲匹配操作对MA1和MA2寄存器的功能相同。

- 如果LPUART_BAUD[MAEN1]和LPUART_BAUD[MAEN2]中只有一个被断言，那么空闲线后的第一个字符只与相关的匹配寄存器进行比较，只有在匹配时才将数据传输到接收数据缓冲器。
- 如果LPUART_BAUD[MAEN1]和LPUART_BAUD[MAEN2]被断言，空闲线路后的第一个字符将与两个匹配寄存器进行比较，只有在与任一寄存器匹配时才传输数据。

#### 48.4.3.2.6 匹配开启 匹配关闭操作

当 LPUART_BAUD[MAEN1]和 LPUART_BAUD[MAEN2]都被设置且 LPUART_BAUD[MATCFG]等于 10
时，匹配开启，匹配关闭操作被启用。在该功能中，由RXD引脚接收的与MATCH[MA1]相匹配的字符被接收并传输到接收缓冲器，LPUART_STAT[RDRF]被设置。所有随后的字符都被认为是数据，也被传送到接收缓冲器。

数据缓冲区，直到收到一个与MATCH[MA2]寄存器匹配的字符。匹配MATCH[MA2]的字符和所有后续字符被丢弃，这种情况一直持续到收到另一个匹配MATCH[MA1]的字符。如果LPUART_BAUD[MAEN1]和LPUART_BAUD[MAEN2]位都被否定，则接收器正常工作，所有收到的数据都被传输到接收数据缓冲器。

**注意事项**

匹配开，匹配关的操作需要LPUART_BAUD[MAEN1]和LPUART_BAUD[MAEN2]都是
被宣称为。

#### 48.4.3.3 硬件流量控制

为了支持硬件流量控制，可以对接收器进行编程，使其自动解除对RTS_B的支持
。

- RTS_B在传输完成前一直保持正向状态，即使在数据传输中途禁用发射器。更多细节请参见使用RTS_B的收发器驱动程序启用。
- 如果接收机请求发送功能被启用，如果接收机数据寄存器中的字符数已满或检测到一个起始位将导致接收机数据寄存器已满，接收机将自动去提示RTS_B。
- 当接收机数据寄存器中的字符数未满，且未检测到会导致接收机数据寄存器满的起始位时，接收机会发出RTS_B。如果STAT[RDRF]被断言，它不受影响。
- 即使RTS_B被去掉，接收器也会继续接收字符，直到接收器数据缓冲区超限。
- 如果接收方请求发送功能被禁用，接收方RTS_B仍为非置位。

#### 48.4.3.4 红外线解码器

红外解码器将收到的字符从IrDA格式转换为接收器使用的NRZ格式。它还有一个OSR超采样波特率时钟计数器，可以过滤噪音，并在收到1的时候指示。

##### 48.4.3.4.1 起始位检测

当STAT[RXINV]被清除时，收到的字符的第一个下降沿对应于起始位。红外线解码器重新设置其计数器。这时，接收器也开始其起始位检测过程。检测到起始位

后，

接收器将其位时间与该起始位时间同步。在其余的字符接收过程中，红外解码器的计数器和接收器的比特时间计数器相互独立计数。

#### 48.4.3.4.2 噪声过滤

在红外解码器计数器的前半段检测到的任何进一步的上升沿都会被解码器忽略。任何小于一个超采样波特时钟的脉冲都可以不被它检测到，不管它是在计数的前半段还是后半段看到的。

#### 48.4.3.4.3 低位检测

在解码器计数的后半部分，一个上升沿被解码为0，被发送到接收器。解码器计数器也被重置。

#### 48.4.3.4.4 高位检测

在前一个上升沿之后的OSR超采样波特率时钟，如果没有看到一个上升沿，那么解码器就会向接收器发送一个1。

如果下一个位是0，而这个0晚点到达，那么根据低位检测，将检测到一个低位。然后，如果一个噪声脉冲在接收器的位时间采样周期之外发生，那么0的延迟就不会被记录为噪声。

### 48.4.4 附加的LPUART功能

下面几节介绍了LPUART的其他功能。

#### 48.4.4.1 数据模式

通过设置LPUART_CTRL[M7]，可以将LPUART发射器和接收器配置为在7位数据模式下工作，通过设置LPUART_CTRL[M]，可以配置为9位数据模式，通过设置LPUART_CTRL[M10]可以配置为10位数据模式。在9位模式下，有第九个数据位，在10位模式下，有第十个数据位。对于发送数据缓冲器，这些位被存储在LPUART_CTRL[T8]和LPUART_CTRL[T9]中。对于接收器，这些位被保存在LPUART_CTRL[R8]和LPUART_CTRL[R9]中。它们也可以通过对LPUART_DATA寄存器的16位或32位访问进行访问。

对于向发送数据缓冲器的连贯8位写入，在写入LPUART_CTRL[T8]和LPUART_CTRL[T9]之前，应写入LPUART_DATA[7:0]。对于对LPUART_DATA寄存器的16位和32位写入，所有10个发送位都同时写入发送数据缓冲区。

如果要作为新字符的第九位和第十位传输的位值与前一个字符的位值相同，则不需要再次向 LPUART_CTRL[T8]和
LPUART_CTRL[T9]写。当数据从发送数据缓冲器传输到发送移位器时，LPUART_CTRL[T8]和LPUART_CTRL[T9]中的值在数据从LPUART_DATA[7:0]传输到移位器的同时被复制。

9位数据模式通常与奇偶校验一起使用，允许8位数据加上第九位的奇偶校验，或者与地址标记唤醒一起使用，这样第九位数据可以作为唤醒位。10位数据模式通常与奇偶校验和地址标记唤醒一起使用，因此第九个数据位可以作为唤醒位，第十个位作为奇偶校验位。在自定义协议中，第九位和/或第十位也可以作为软件控制的标记。

#### 48.4.4.2 闲置长度

空闲字符是指起始位、所有数据位和停止位都处于标记位置的字符。CTRL[ILT]寄存器可以被配置为从上一个起始位（任何数据位和停止位都计入空闲字符检测）或上一个停止位开始检测空闲字符。

使用CTRL[IDLECFG]字段也可以配置在检测到空闲线路条件之前必须收到的空闲字符数。这个字段配置了在STAT[IDLE]标志被设置、STAT[RAF]标志被清除和DATA[IDLINE]标志被设置之前必须收到的空闲字符的数量，与下一个收到的字符一起。

空闲线唤醒和空闲匹配操作也会受到CTRL[IDLECFG]字段的影响。当地址匹配或匹配开/关操作被启用时，设置STAT[RWUID]位将导致任何被丢弃的字符被当作空闲字符处理。

#### 48.4.4.3 循环模式

当LPUART_CTRL[LOOPS]被设置时，同一寄存器中的LPUART_CTRL[RSRC]位会选择循环模式（LPUART_CTRL[RSRC]=0）或单线模式（LPUART_CTRL[RSRC]=1）。循环模式有时被用来检查软件。

独立于外部系统的连接，以帮助隔离系统问题。在这种模式下，发射器输出在内部与接收器输入相连，RXD引脚不被LPUART使用。

#### 48.4.4.4 单线操作

当LPUART_CTRL[LOOPS]被设置时，同一寄存器中的RSRC位会选择循环模式（LPUART_CTRL[RSRC]=0）或单线模式（LPUART_CTRL[RSRC]=1）。单线模式实现了一个半双工的串行连接。接收器内部连接到发射器输出和TXD引脚（RXD引脚不使用）。

在单线模式下，LPUART_CTRL[TXDIR]位控制TXD引脚上串行数据的方向。当LPUART_CTRL[TXDIR]被清除时，TXD 引脚是接收器的输入，发射器暂时与TXD
引脚断开连接，因此外部设备可以向接收器发送串行数据。当LPUART_CTRL[TXDIR]被设置时，TXD引脚是由发射器驱动的输出，内部环回连接被禁用，因此接收器不能接收由发射器发出的字符。

### 48.4.5 红外线接口

LPUART提供了向红外LED发射窄脉冲和接收窄脉冲并将其转换为串行比特的能力，这些比特被发送到LPUART。IrDA物理层规范定义了一个用于交换数据的半双工红外通信链路。完整的标准包括高达16Mbits/s的数据率。本设计仅涵盖2.4kbits/s和115.2kbits/s之间的数据速率。

LPUART有一个红外发射编码器和接收解码器。LPUART传输的串行数据位由红外子模块编码，为每一个零位传输一个窄脉冲。每一个1比特不传输任何脉冲。在接收数据时，红外脉冲用红外光二极管检测，并由LPUART外部的红外接收解码器转化为CMOS电平。然后，狭窄的脉冲被红外接收解码器拉长，回到LPUART接收的串行比特流。传输脉冲和预期接收脉冲的极性可以反转，这样就可以直接连接到使用高电平脉冲的外部IrDA收发模块。

红外线子模块从LPUART接收其时钟源。在红外子模块中选择这两个时钟中的一个，在传输过程中产生1/OSR、2/OSR、3/OSR或4/OSR窄脉冲。

#### 48.4.5.1 红外线发射编码器

红外发射编码器将发射移位寄存器中的串行数据位转换为TXD信号。零位时发送一个窄脉冲，一位时不发送脉冲。窄脉冲在比特开始时发送，持续时间为比特时间的1/OSR、2/OSR、3/OSR或4/OSR。当LPUART_CTRL[TXINV]被清除时，为零位发送一个窄的低脉冲，而当LPUART_CTRL[TXINV]被设置时，为零位发送一个窄的高脉冲。

#### 48.4.5.2 红外线接收解码器

红外线接收块将数据从RXD信号转换到接收移位寄存器。对于收到的每一个零位，预计会有一个窄脉冲，而对于收到的每一个一，预计没有脉冲。当LPUART_STAT[RXINV]被清除时，预计一个零位会有一个窄的低脉冲，而当LPUART_STAT[RXINV]被设置时，预计一个零位会有一个窄的高脉冲。这种接收解码器符合IrDA串行红外物理层规范所定义的边缘抖动要求。

### 48.4.6 中断和状态标志

LPUART发射器有两个状态标志，可以选择性地产生硬件中断请求。发送数据寄存器空LPUART_STAT[TDRE]）指示发送数据缓冲器中何时有空间向LPUART_DATA写入另一个发送字符。如果发送中断使能LPUART_CTRL[TIE]）位被设置，当LPUART_STAT[TDRE]被设置时，将请求一个硬件中断。发送完成（LPUART_STAT[TC]）表示发送器完成了所有数据、序言和中断字符的发送，并在TXD处于非活动水平时处于空闲状态。这个标志在有调制解调器的系统中经常使用，以确定何时可以安全地关闭调制解调器。如果发送完成中断使能（LPUART_CTRL[TCIE]）位被设置，当LPUART_STAT[TC]被设置时，会请求一个硬件中断。如果相应的LPUART_CTRL[TIE]或LPUART_CTRL[TCIE]本地中断掩码被清零，则可使用软件轮询代替硬件中断，以监测LPUART_STAT[TDRE]和LPUART_STAT[TC]状态标志。

当程序检测到接收数据寄存器已满（LPUART_STAT[RDRF]=1）时，它通过读取LPUART_DATA获得接收数据寄存器的数据。LPUART_STAT[RDRF]标志通过读取LPUART_DATA而被清除。

IDLE状态标志包括防止它在RXD线长时间保持空闲时被重复设置的逻辑。IDLE是通过向LPUART_STAT[IDLE]标志写1来清除的。LPUART_STAT[IDLE]被清除后，它不能再次被设置，直到接收器收到至少一个新字符并设置LPUART_STAT[RDRF]。

如果在导致LPUART_STAT[RDRF]被设置的接收字符中检测到相关错误，则错误标志--
噪声标志（LPUART_STAT[NF]）、成帧错误（LPUART_STAT[FE]）和奇偶校验错误标志（LPUART_STAT[PF]）将与LPUART_STAT[RDRF]同时设置。这些标志在超限情况下不被设置。

如果当一个新的字符准备从接收移位器传输到接收数据缓冲器时，LPUART_STAT[RDRF]已经被设置，则超限（LPUART_STAT[OR]）标志被设置，而不是数据连同任何相关的NF、FE或PF条件被丢失。

如果收到的字符与MATCH[MA1]和/或MATCH[MA2]的内容相匹配，那么LPUART_STAT[MA1F]和/或LPUART_STAT[MA2F]标志就会在设置LPUART_STAT[RDRF]的同时设置。

在任何时候，RXD串行数据输入引脚上的有效边沿都会导致LPUART_STAT[RXEDGIF]标志被设置。LPUART_STAT[RXEDGIF]标志通过向其写1来清除。
该功能取决于接收器被启用（LPUART_CTRL[RE]=1）。

## 48.4.7 周边触发器

LPUART外设触发器与其他外设的连接是设备特定的。

### 48.4.7.1 输出触发器

LPUART产生三个输出触发器，可以连接到设备上的其他外设。

- 传输字触发器在每个传输字结束时发出信号，它在一个比特周期后被否定。
- 接收字触发器在每个被写入接收FIFO的接收字结束时发出信号，持续一个超采样时钟周期。
- 接收空闲触发器在空闲标志设置时发出信号，持续一个超采样时钟周期。

### 48.4.7.2 输入触发器

LPUART支持一个外设输入触发器，可以通过以下方式之一进行配置。

- 输入触发器可以代替CTS_B引脚的输入。当发射器空闲有数据要发送时，输入触发器必须断言超过一个比特时钟周期，以保证新的传输。
- 输入触发器可以调制传输数据输出（触发器与TXD输出逻辑上是相通的）。输入触发器应该由一个周期小于比特时钟频率的PWM源产生。
- 输入触发器可以代替RXD引脚的输入。输入触发器预计将由接收数据源产生，如模拟比较器或外部引脚。

# 第四十九章
# 灵活的I/O (FlexIO)

## 49.1 特定芯片的FlexIO信息

### 49.1.1 FlexIO配置

表49-1.FlexIO配置

| | 计时器 | 变速器 | 插销 |
|---|---|---|---|
| 数量 | 4 | 4 | 8 |

该器件不支持低漏电和等待模式。关于可用的电源模式的详情，请参见模块在可用的低功率模式下的操作。

**注意事项**

在禁用FlexIO外设时钟（FlexIO clock）的情况下访问FlexIO寄存器将导致传输错误或使总线停滞。

## 49.2 简介

### 49.2.1 概述

FlexIO是一个高度可配置的模块，提供广泛的功能，包括。

- 仿真各种串行通信协议
- 灵活的16位定时器，支持各种触发、复位、启用和禁用条件

### 49.2.2 特点

FlexIO模块能够支持广泛的协议，包括但不限于以下协议。

- UART
- I2C
- SPI
- I2S
- PWM/波形生成

提供了以下主要特征。

- 具有发送、接收和数据匹配模式的32位移位寄存器阵列
- 双缓冲移位器操作，用于连续数据传输
- 支持大传输量的移位器串联
- 自动生成启动/停止位
- 中断、DMA或轮询的发送/接收操作
- 独立于总线时钟频率的可编程波特率，在停止模式下支持异步操作
- 高度灵活的16位定时器，支持各种内部或外部触发、复位、启用和禁用条件

### 49.2.3 框图

下图给出了FlexIO定时器和移位器配置的高级概述。

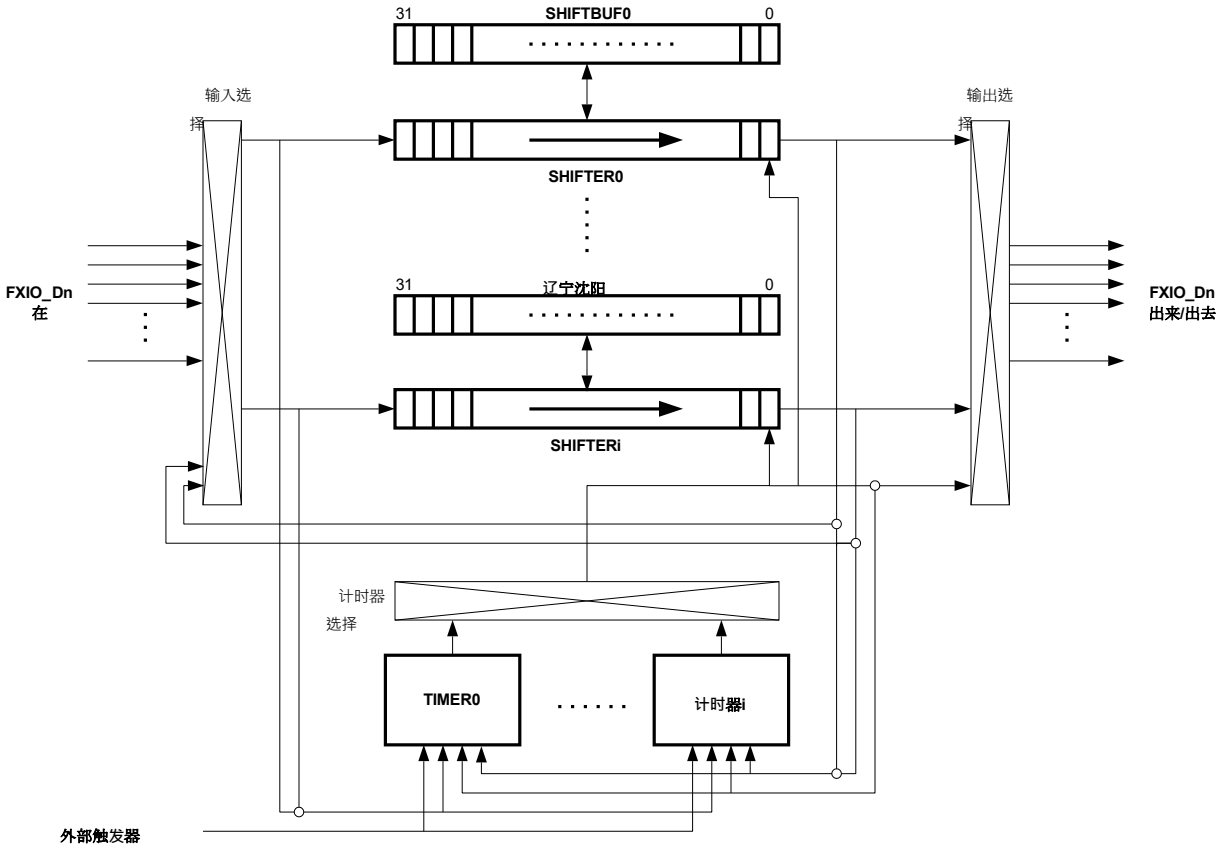



**图49-1.FlexIO方框图**

## 49.2.4 操作模式

FlexIO模块支持下表中描述的芯片模式。

**表49-2.FlexIO模块支持的芯片模式**

| 芯片模式 | FlexIO操作 |
|---|---|
| 运行 | 正常运行 |
| 停止/等待 | 可以继续工作，前提是Doze Enable位（CTRL[DOZEN]）清零，并且FlexIO使用的是外部或内部时钟源，在停止/等待模式下保持工作。 |
| 低漏电停机 | Doze Enable（CTRL[DOZEN]）位被忽略，FlexIO将等待所有定时器完成任何未决操作，然后再确认进入低漏电模式。 |
| 调试 | 只要调试启用位（CTRL[DBGE]）被设置，就可以继续运行。 |

### 49.2.5 FlexIO信号描述

| 信号 | 描述 | 输入/输出 |
|---|---|---|
| FXIO_Dn (n=0...7) | 双向FlexIO移位器和定时器引脚输入/输出 | 输入/输出 |

## 49.3 存储器地图和寄存器

### 49.3.1 FLEXIO寄存器描述

#### 49.3.1.1 FLEXIO内存图

FlexIO基础地址。4005_A000h

| 偏移 | 注册 | 宽度 (单位：比特) | 访问 | 重置值 |
|---|---|---|---|---|
| 0h | 版本标识寄存器（VERID） | 32 | RO | 0101_0000h |
| 4h | 参数寄存器（PARAM）。 | 32 | RO | 0408_0404h |
| 8h | FlexIO控制寄存器（CTRL） | 32 | RW | 0000_0000h |
| 丘吉尔 | 引脚状态寄存器（PIN） | 32 | RO | 0000_0000h |
| 10h | 移位器状态寄存器（SHIFTSTAT）。 | 32 | W1C | 0000_0000h |
| 14h | 移位器错误寄存器（SHIFTERR）。 | 32 | W1C | 0000_0000h |
| 18h | 计时器状态寄存器（TIMSTAT） | 32 | W1C | 0000_0000h |
| 20h | 移位器状态中断启用（SHIFTSIEN）。 | 32 | RW | 0000_0000h |
| 24h | 移位错误中断启用（SHIFTEIEN）。 | 32 | RW | 0000_0000h |
| 28h | 定时器中断使能寄存器（TIMIEN） | 32 | RW | 0000_0000h |
| 30h | 移位器状态DMA启用（SHIFTSDEN）。 | 32 | RW | 0000_0000h |
| 80h - 8Ch | 移位器控制N寄存器（SHIFTCTL0 - SHIFTCTL3）。 | 32 | RW | 0000_0000h |
| 100h - 10Ch | 移位器配置N寄存器（SHIFTCFG0 - SHIFTCFG3）。 | 32 | RW | 0000_0000h |
| 200h - 20Ch | 移位缓冲器N寄存器（SHIFTBUF0 - SHIFTBUF3）。 | 32 | RW | 0000_0000h |
| 280h - 28Ch | 移位缓冲器N位交换寄存器（SHIFTBUFBIS0 - SHIFTBUF BIS3）。 | 32 | RW | 0000_0000h |
| 300h - 30Ch | 移位缓冲器N字节交换寄存器（SHIFTBUFBYS0 - SHIF TBUFBYS3）。 | 32 | RW | 0000_0000h |

| | | | | |
|---|---|---|---|---|
| 380h - 38Ch | 移位缓冲器N位字节交换寄存器（SHIFTBUFBBS0 - SHIF TBUFBBS3）。 | 32 | RW | 0000_0000h |
| 400h - 40Ch | 计时器控制N寄存器（TIMCTL0 - TIMCTL3） | 32 | RW | 0000_0000h |
| 480h - 48Ch | 计时器配置N寄存器（TIMCFG0 - TIMCFG3） | 32 | RW | 0000_0000h |

*表继续在下一页...*

| 49.2.5 FlexIO信号描述 偏移 | 注册 | 宽度 (单位：比特) | 访问 | 重置值 |
|---|---|---|---|---|
| 500h - 50Ch | 定时器比较N寄存器（TIMCMP0 - TIMCMP3）。 | 32 | RW | 0000_0000h |

## 49.3.1.2 版本标识寄存器（VERID）

### 49.3.1.2.1 偏移

| 注册 | 偏移 |
|---|---|
| 虚实对照 | 0h |

### 49.3.1.2.2 图示

| 咬牙切齿 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 主要的 | | | | | | | | 小学 | | | | | | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

| 咬牙切齿 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 特写 | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

### 49.3.1.2.3 领域

| 场地 | 职能 |
|---|---|
| 31-24<br>主要的 | 主要版本号<br>这个只读字段返回模块规范的主要版本号。 |
| 23-16<br>小学 | 次要版本号<br>这个只读字段返回模块规范的次要版本号。 |
| 15-0<br>特写 | 特点规格号<br>这个只读字段返回特征集编号。<br>0000000000000000b - 实施标准功能。0000000000000001b - 支持状态、逻 |

| | |
|---|---|
| | 辑和并行模式。 |

## 49.3.1.3 参数寄存器（PARAM）。

### 49.3.1.3.1 偏移

| 注册 | 偏移 |
|---|---|
| 朴实无华 | 4h |

### 49.3.1.3.2 图示

| 咬牙切齿 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 触发器 | | | | | | | | 密码 | | | | | | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |

| 咬牙切齿 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 计时器 | | | | | | | | 换挡器 | | | | | | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |

### 49.3.1.3.3 领域

| 场地 | 职能 |
|---|---|
| 31-24<br>触发器 | 触发器编号<br>实施的外部触发器的数量。 |
| 23-16<br>密码 | 引脚编号<br>实施的引脚数量。 |
| 15-8<br>计时器 | 计时器编号<br>实施的定时器的数量。 |
| 7-0<br>换挡器 | 变速器编号<br>实施的移位器的数量。 |

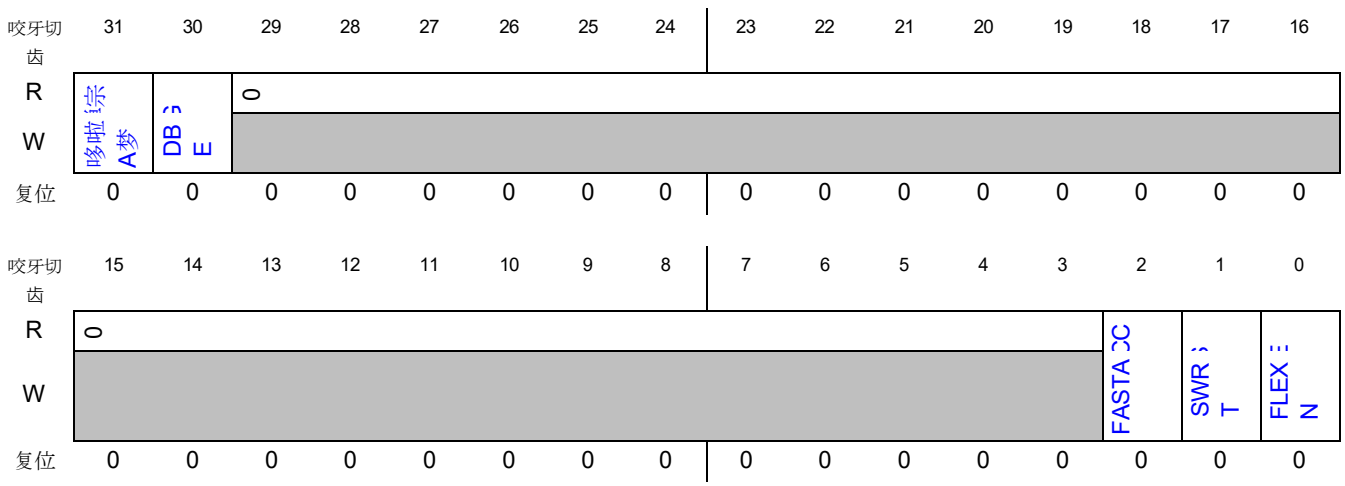
## 49.3.1.4 FlexIO控制寄存器（CTRL）

## 49.3.1. 偏移

1517.1

| 注册 | 偏移 |
|---|---|
| CTRL | 8h |

### 49.3.1.4.2 图示




### 49.3.1.4.3 领域

| 场地 | 职能 |
|---|---|
| 31<br>哆啦A梦 | 启用打盹功能<br>禁用Doze模式下的FlexIO操作。0b - 在休眠模式下启用FlexIO。1b - 在休眠模式下禁用FlexIO。 |
| 30<br>DBGE | 调试启用<br>启用调试模式下的FlexIO操作。<br>0b - 在调试模式下禁用FlexIO。1b - 在调试模式下，FlexIO被启用。 |
| 29-3<br>- | 保留。 |
| 2<br>FASTACC | 快速访问<br>启用对FlexIO寄存器的快速寄存器访问，但要求FlexIO功能时钟的频率至少是总线时钟的两倍。<br>0b - 配置FlexIO的正常寄存器访问 1b - 配置FlexIO的快速寄存器访问 |

| | |
|---|---|
| 1<br>ÃÃÃ | 软件复位<br><br>FlexIO控制寄存器不受软件复位的影响，FlexIO中的所有其他逻辑都受到软件复位的影响，寄存器的访问被忽略，直到该位被清除。该寄存器位将保持设置，直到被软件清除，并且复位在FlexIO时钟域中已被清除。<br>0b - 软件复位被禁用<br>1b - 启用软件复位，除控制寄存器外的所有FlexIO寄存器都被复位。 |

*表继续在下一页...*

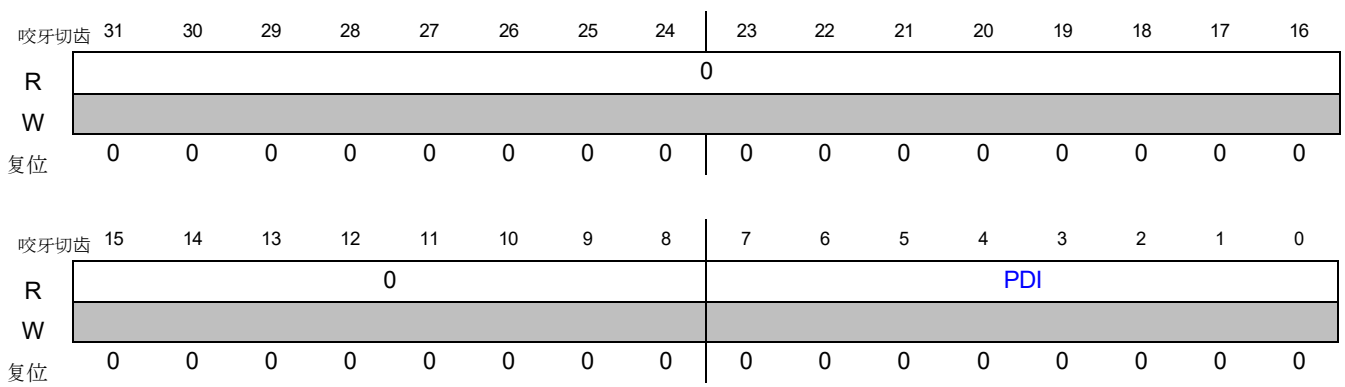
| 49.3.1场地 | 偏移 | 职能 |
| --- | --- | --- |
| 1519.1<br>法丝丽 | 启用FlexIO<br>0b - FlexIO模块被禁用。1b - FlexIO模块被启用。 | |

## 49.3.1.5 引脚状态寄存器（PIN）

### 49.3.1.5.1 偏移

| 注册 | 偏移 |
| --- | --- |
| 密码 | 丘吉尔 |

### 49.3.1.5.2 图示




### 49.3.1.5.3 领域

| 场地 | 职能 |
| --- | --- |
| 31-8<br>- | 保留。 |
| 7-0<br>PDI | 引脚数据输入<br>返回每个FlexIO引脚的输入数据。 |

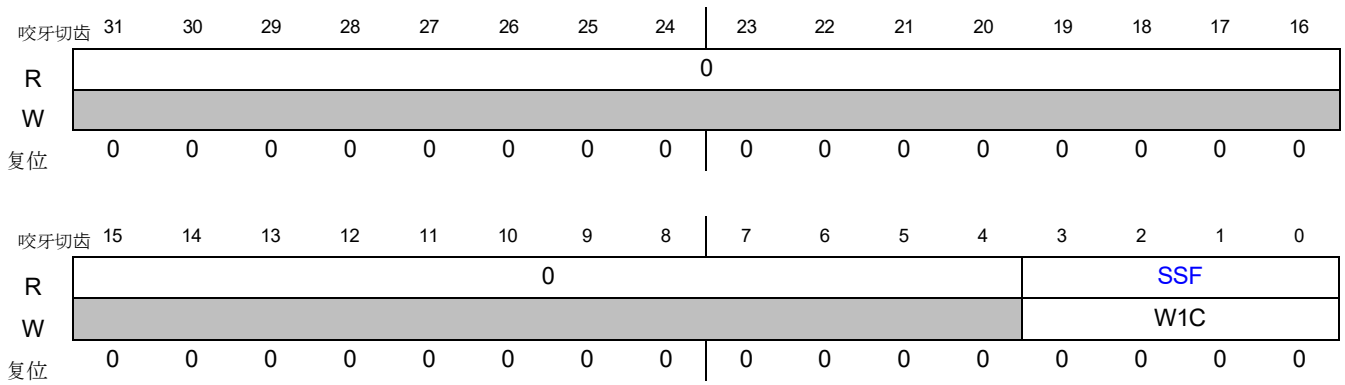
## 49.3.1.6 移位器状态寄存器（SHIFTSTAT）。

## 49.3.1. 偏移

1521.1

| 注册 | 偏移 |
|---|---|
| 湿度统计 | 10h |

## 49.3.1.6.2 图示




## 49.3.1.6.3 领域

| 场地 | 职能 |
|---|---|
| 31-4<br>- | 保留。 |
| 3-0<br>SSF | 移位器状态标志<br>当以下事件之一发生时，移位器状态标志被更新。<br>对于SMOD=Receive，当SHIFTBUF被载入移位器的数据时（SHIFTBUF已满），状态标志被设置，当SHIFTBUF寄存器被读取时，状态标志被清除。<br>对于SMOD=传输，当SHIFTBUF数据被传输到移位器（SHIFTBUF为空）或最初配置为SMOD=传输时，状态标志被设置，当SHIFTBUF寄存器被写入时，状态标志被清除。<br>对于SMOD=匹配存储，当SHIFTBUF和移位器之间发生匹配时，状态标志被设置，而当SHIFTBUF寄存器被读取时，状态标志被清除。<br>对于SMOD=Match Continuous，返回SHIFTBUF和Shifter之间的当前匹配结果。<br>对于除 "连续匹配 "以外的所有模式，状态标志也可以通过向标志写一个逻辑1来清除。<br>0b - 状态标志清除。<br>1b - 状态标志被设置。 |

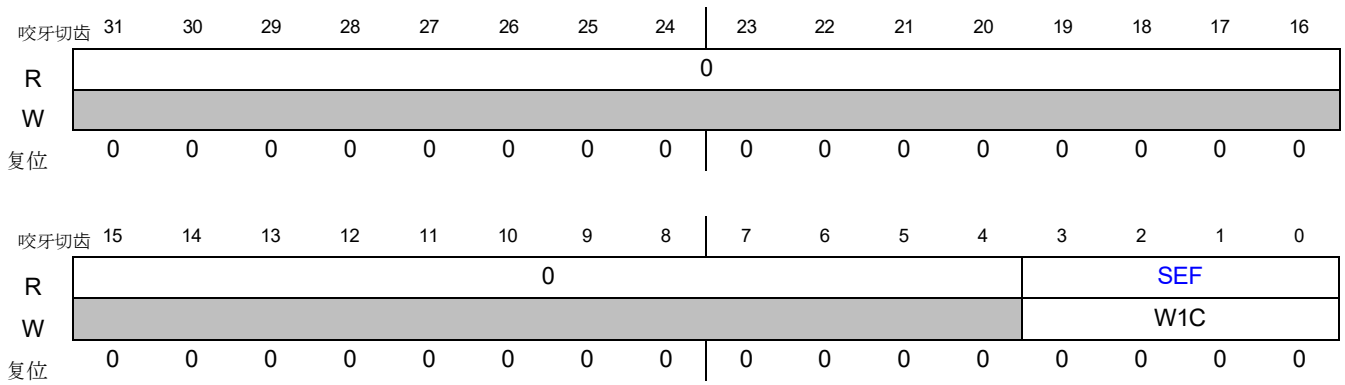
### 49.3.1.7 移位器错误寄存器（SHIFTERR）。

## 49.3.1. 偏移

1520.1

| 注册 | 偏移 |
|---|---|
| 转换器(SHIFTERR) | 14h |

### 49.3.1.7.2 图示




### 49.3.1.7.3 领域

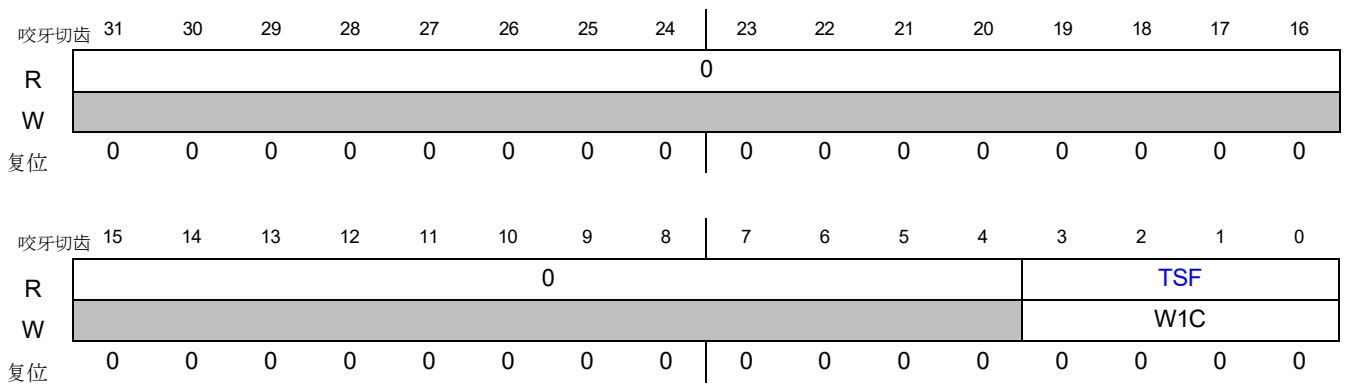
| 场地 | 职能 |
|---|---|
| 31-4<br>- | 保留。 |
| 3-0<br>SEF | 移位器错误标志<br>当以下事件之一发生时，移位器错误标志被设置。<br>对于SMOD=Receive，表示移位器在从SHIFTBUF中读取前一个数据之前，已经准备好将新的数据存储到SHIFTBUF中（SHIFTBUF Overrun），或者表示收到的起始位或停止位与预期值不符。<br>对于SMOD=Transmit，表示移位器在新数据被写入SHIFTBUF之前已经准备好从SHIFTBUF加载新数据（SHIFTBUF Underrun）。<br>对于SMOD=匹配存储，表示在从SHIFTBUF读取前一个匹配数据之前发生了一个匹配事件（SHIFTBUF超限）。<br>对于SMOD=Match Continuous，当SHIFTBUF和Shifter之间发生匹配时，错误标志被设置。<br>可以通过向该标志写逻辑一来清除。对于SMOD=Match Continuous，当SHIFTBUF寄存器被读取时，也可以被清除。<br>0b - 移位器错误标志被清除。<br>1b - 移位器错误标志被设置。 |

# 49.3.1.8 计时器状态寄存器（TIMSTAT）

## 49.3.1.8.1 偏移

| 注册 | 偏移 |
|---|---|
| TIMSTAT | 18h |

## 49.3.1.8.2 图示




## 49.3.1.8.3 领域

| 场地 | 职能 |
|---|---|
| 31-4<br>- | 保留。 |
| 3-0<br>TSF | 计时器状态标志<br>定时器状态标志的设置取决于定时器模式，可以通过向该标志写入逻辑1来清除。<br>在8位计数器模式下，当上层的8位计数器等于0并递减时，定时器状态标志被设置，这也会使计数器与比较寄存器中的值重新加载。<br>在8位PWM模式下，当上层8位计数器等于0并递减时，定时器状态标志被设置，这也导致计数器用比较寄存器中的值重新加载。<br>在16位计数器模式下，当16位计数器等于零并递减时，定时器状态标志被设置，这也会使计数器用比较寄存器中的值重新加载。<br>0b - 计时器状态标志清除。1b - 计时器状态标志被设置。 |

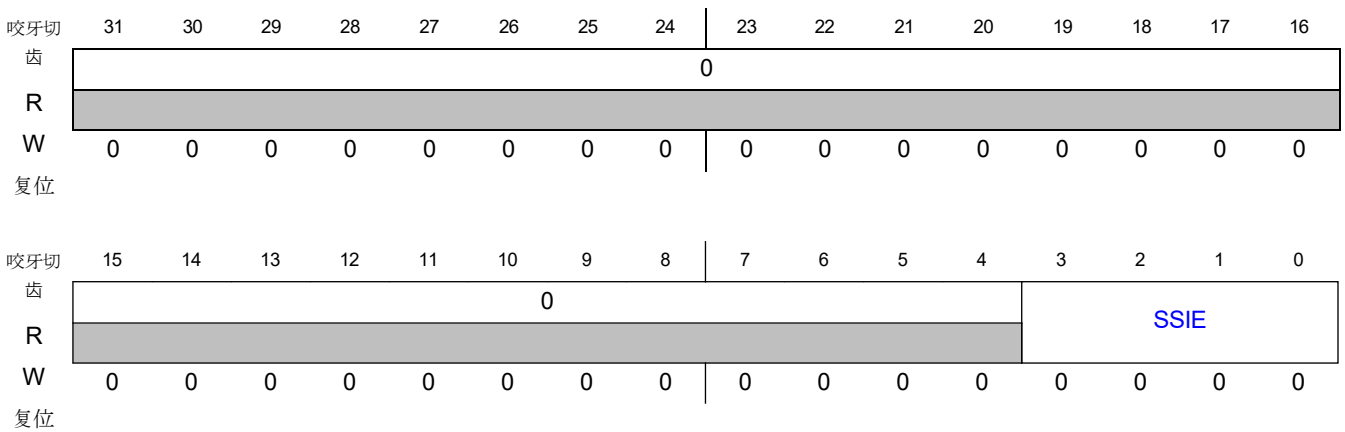
**49.3.1.　　偏移**
**1522.1**

**49.3.1.9　移位器状态中断启用（SHIFTSIEN）。**

## 49.3.1.8 计时器状态寄存器（TIMSTAT）

| 注册 | 偏移 |
|---|---|
| 电手掌柜 | 20h |

### 49.3.1.9.2 图示




### 49.3.1.9.3 领域

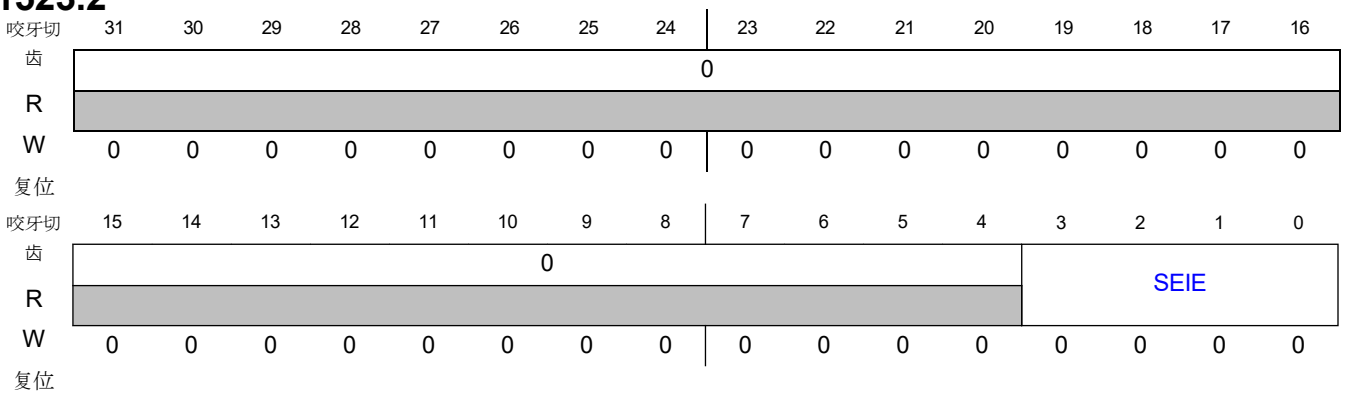
| 场地 | 职能 |
|---|---|
| 31-4<br>- | 保留。 |
| 3-0<br>SSIE | 移位器状态中断启用<br>当相应的SSF被设置时，启用中断生成。<br>0b - 变速器状态标志中断禁用。1b - 变速器状态旗中断启用。 |

## 49.3.1.10 移位错误中断启用（SHIFTEIEN）。

### 49.3.1.10.1 偏移量

| 注册 | 偏移 |
|---|---|
| 薛飞田 | 24h |

### 49.3.1.1523.2 图示




## 49.3.1.10.3 领域

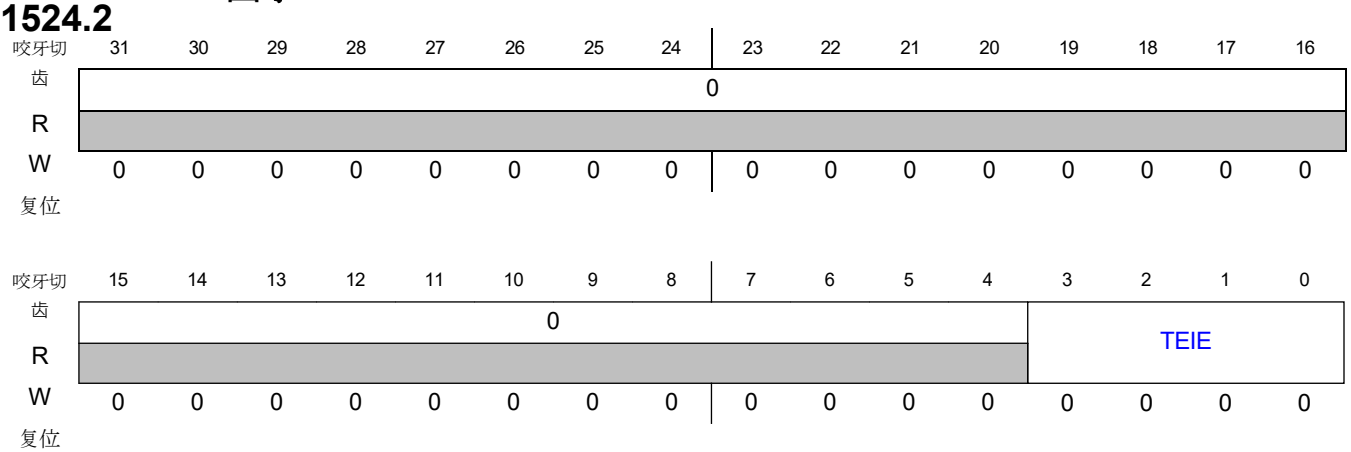
| 场地 | 职能 |
|---|---|
| 31-4<br>- | 保留。 |
| 3-0<br>SEIE | 移位器错误中断启用<br>当相应的SEF被设置时，启用中断生成。<br>0b - 关闭移位器错误标志中断。1b - 启用<br>移位器错误标志中断。 |

# 49.3.1.11 定时器中断使能寄存器（TIMIEN）

## 49.3.1.11.1 偏移量

| 注册 | 偏移 |
|---|---|
| 纪元 | 28h |

## 49.3.1. 1524.2 图示




### 49.3.1.11.3 领域

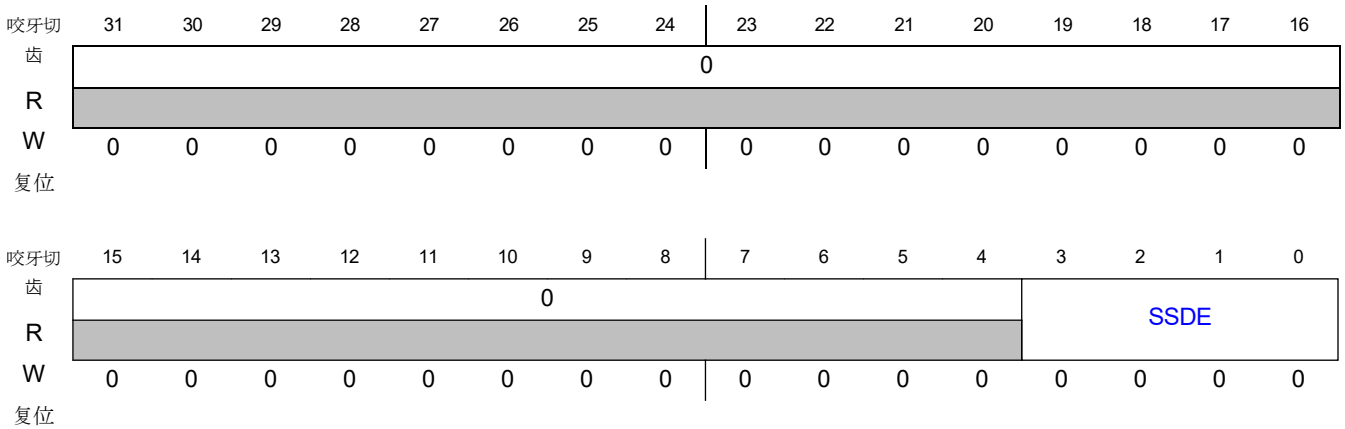
| 场地 | 职能 |
|---|---|
| 31-4<br>- | 保留。 |
| 3-0<br>TEIE | 计时器状态中断启用<br>当相应的TSF被设置时，启用中断生成。<br>0b - 定时器状态标志中断被禁用。1b - 计时器状态标志中断被启用。 |

## 49.3.1.12 移位器状态DMA启用（SHIFTSDEN）。

### 49.3.1.12.1 偏移

| 注册 | 偏移 |
|---|---|
| 甩手掌柜 | 30h |

### 49.3.1.12.2 图示




### 49.3.1.12.3 领域

| 场地 | 职能 |
|---|---|
| 31-4<br>- | 保留。 |
| 3-0<br>SSDE | 变速器状态 DMA启用<br>当相应的SSF被设置时，启用DMA请求的生成。<br>0b - 变速器状态标志DMA请求被禁用。1b - 移位器状态标志DMA请求被启用。 |

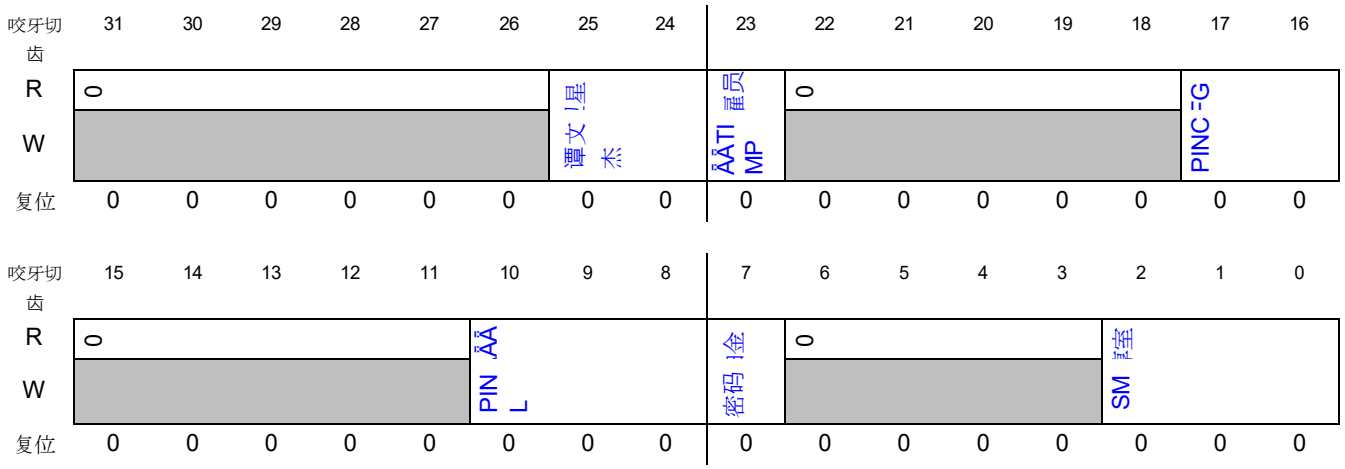
## 49.3.1.13 移位器控制N寄存器（SHIFTCTL0 - SHIFTCTL3）。

### 49.3.1.13.1 偏移

对于a=0到3。

| 注册 | 偏移 |
|---|---|
| SHIFTCTLa | 80h + (a × 4h) |

## 49.3.1.13.2 图示




## 49.3.1.13.3 领域

| 场地 | 职能 |
|---|---|
| 31-26<br>- | 保留。 |
| 25-24<br>秦始皇兵马俑 | 计时器选择<br>选择哪个定时器用于控制逻辑/移位寄存器和产生移位时钟。TIMSEL=i将选择TIMERi。 |
| 23<br>钛宝 | 定时器极性<br>0b - 在移位时钟的正边移位 1b - 在移位时钟的负边移位 |
| 22-18<br>- | 保留。 |
| 17-16<br>铂尼弗格(PINCFG) | 移位器引脚配置<br>00b - 禁用移位器引脚输出<br>01b - 移位器引脚开漏或双向输出使能 10b - 移位器引脚双向输出数据<br>11b - 移位器针脚输出 |
| 15-11<br>- | 保留。 |
| 10-8<br>PINSEL | 移位器针脚选择<br>选择移位器的输入或输出所使用的引脚。PINSEL=i将选择FXIO_Di引脚。 |
| 7<br>PINPOL | 移位器引脚极性<br>0b - 引脚为高电平有效 1b - 引脚为低电平有效 |

| | |
|---|---|
| 6-3<br>- | 保留。 |
| 2-0<br>证券时报 | 变速器模式<br>配置移位器的模式。<br>000b - 禁用。 |

| 场地 | 职能 |
| --- | --- |
| | 001b - 接收模式。在定时器到期时将当前移位器的内容捕获到SHIFTBUF中。<br>010b - 发送模式。在定时器到期时将SHIFTBUF的内容加载到移位器中。011b - 保留。<br>100b - 匹配存储模式。定时器到期时，移位器数据与SHIFTBUF内容进行比较。<br>101b - 匹配连续模式。移位器数据与SHIFTBUF内容连续比较。110b -<br>111b - |

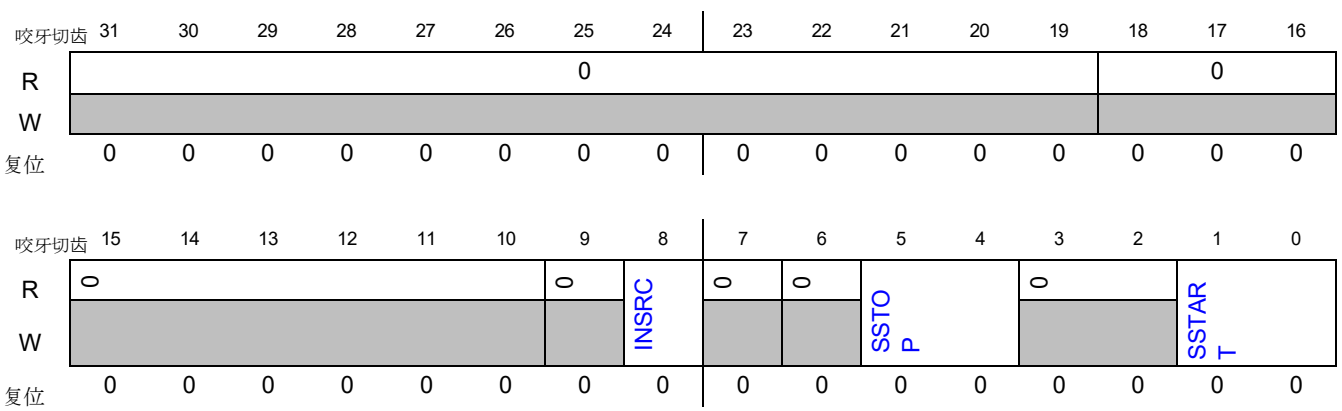
## 49.3.1.14 移位器配置N寄存器（SHIFTCFG0 - SHIFTCFG3）。

### 49.3.1.14.1 偏移

对于a=0到3。

| 注册 | 偏移 |
| --- | --- |
| SHIFTCFGa | 100h + (a × 4h) |

### 49.3.1.14.2 图示




### 49.3.1.14.3 领域

| 场地 | 职能 |
| --- | --- |
| 31-19<br>- | 保留。 |
| 18-16 | 保留。 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| - | |
| 15-10<br>- | 保留。 |
| 9<br>- | 保留。 |
| 8<br>INSRC | 输入源<br>选择移位器的输入源。<br>0b - 引脚<br>1b - 移位器N+1输出 |
| 7<br>- | 保留。 |
| 6<br>- | 保留。 |
| 5-4<br>暂停 | 移位器 停止位<br>对于SMOD=发送，如果选定的定时器也启用了停止位，该字段允许自动插入停止位。<br>对于SMOD=接收或匹配存储，如果所选定时器也启用了停止位，该字段允许自动检查停止位。<br>00b - 发送器/接收器/匹配存储禁用停止位 01b - 发送器/接收器/匹配存储保留。<br>10b - 发送器在存储时输出停止位值0，如果停止位不为0，接收器/匹配存储设置错误标志<br>11b - 发送器在存储时输出停止位值1，如果停止位不是1，接收器/匹配存储设置错误标志 |
| 3-2<br>- | 保留。 |
| 1-0<br>创业板 | 移位器起始位<br>对于SMOD=Transmit，如果所选定时器也启用了启动位，该字段允许自动插入启动位。<br>对于SMOD=接收或匹配存储，如果所选定时器也启用了启动位，该字段允许自动检查启动位。<br>00b - 发送器/接收器/匹配存储禁用起始位，发送器在启用时加载数据 01b - 发送器/接收器/匹配存储禁用起始位，发送器在第一班加载数据 10b - 发送器在第一班加载数据前输出起始位值0，如果起始位不为0，接收器/匹配存储设置错误标志<br>11b - 发送器在第一次移位加载数据前输出起始位值1，如果起始位不是1，接收器/匹配库设置错误标志。 |

## 49.3.1.15 移位缓冲器N寄存器（SHIFTBUF0 - SHIFTBUF3）。

### 49.3.1.15.1 偏移

对于a=0到3。

| 注册 | 偏移 |
|---|---|
| SHIFTBUFa | 200h + (a × 4h) |

## 49.3.1.15.2 图示




## 49.3.1.15.3 领域

| 场地 | 职能 |
|---|---|
| 31-0<br>呼叫中心 | 移位缓冲器<br>根据SMOD的设置，移位缓冲区的数据被用于各种功能。<br>对于 SMOD=Receive，移位器数据在定时器到期时被转入 SHIFTBUF。只有当相应的移位器状态标志被设置，表明有新的移位器数据可用时，才能读取SHIFTBUF寄存器。<br>对于SMOD=发送，SHIFTBUF的数据在定时器开始之前被传送到移位器中。<br>对于 SMOD=匹配存储，SHIFTBUF[31:16]包含要与移位器内容匹配的数据，SHIFTBUF[15:0]可用于屏蔽匹配结果（1=屏蔽，0=无屏蔽）。当定时器过期时，会对匹配进行检查，每当匹配事件发生时，移位器数据[31:16]会被写入SHIFTBUF[31:16]。SHIFTBUF 寄存器只有在相应的移位器状态标志被设置时才能被读取，表明有新的移位器数据可用。<br>对于SMOD=连续匹配，SHIFTBUF[31:16]包含要与移位器内容匹配的数据，SHIFTBUF[15:0]可用于屏蔽匹配结果（1=屏蔽，0=无屏蔽）。 |

# 49.3.1.16 移位缓冲器N位交换寄存器（SHIFTBUFBIS0 - SHIFTBUFBIS3）

#### 49.3.1.16.1 偏移

对于a=0到3。

| 注册 | 偏移 |
| --- | --- |
| 证券交易委员会a | 280h + (a × 4h) |

### 49.3.1.16.2 图示




### 49.3.1.16.3 领域

| 场地 | 职能 |
| --- | --- |
| 31-0<br>辽宁沈阳 | 移位缓冲器<br>与 SHIFTBUF 寄存器有异曲同工之妙，只是对该寄存器的读/写进行了位互换。读取时返回 SHIFTBUF[0:31]。 |

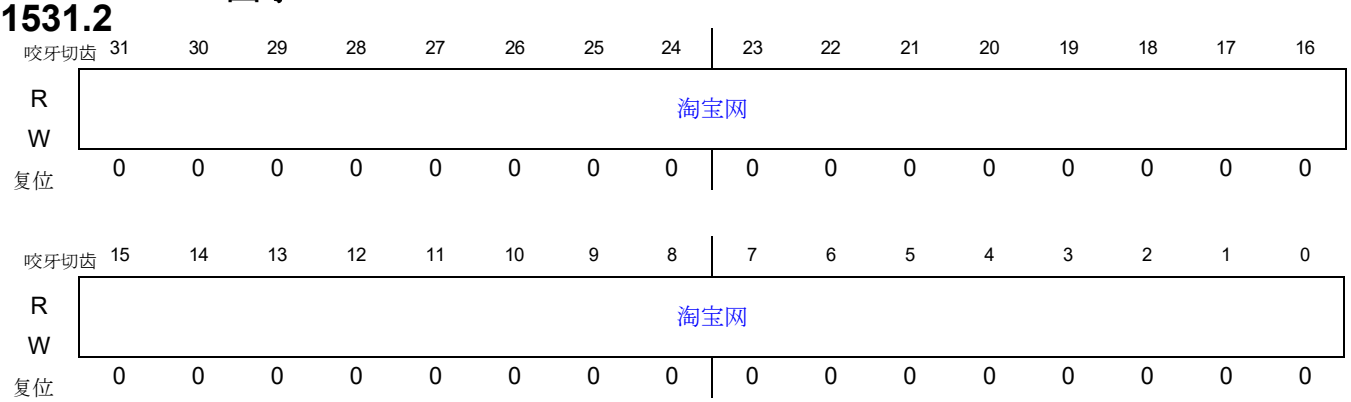
## 49.3.1.17 移位缓冲器N字节交换寄存器（SHIFTBUFBYS0 - SHIFTBUFBYS3）。

### 49.3.1.17.1 偏移量

对于a=0到3。

| 注册 | 偏移 |
| --- | --- |
| 淘宝网 | 300h + (a × 4h) |

### 49.3.1.1531.2 图示




### 49.3.1.17.3 领域

| 场地 | 职能 |
|---|---|
| 31-0<br>淘宝网 | 移位缓冲器<br>与SHIFTBUF寄存器同名，但对该寄存器的读/写是字节交换的。读取时返回{ shiftbuf[7:0], shiftbuf[15:8], shiftbuf[23:16], shiftbuf[31:24] }。 |

## 49.3.1.18 移位缓冲器N位字节交换寄存器（SHIFTBUF BBS0 - SHIFTBUFBBS3）。

### 49.3.1.18.1 偏移

对于a=0到3。

| 注册 | 偏移 |
|---|---|
| SHIFTBUFBBSa | 380h + (a × 4h) |

### 49.3.1.18.2 图示

| 咬牙切齿 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R<br>W | 淘宝网 | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 咬牙切齿 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R<br>W | 淘宝网 | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

### 49.3.1.18.3 领域

| 场地 | 职能 |
|---|---|
| 31-0<br>淘宝网 | 移位缓冲器<br>与SHIFTBUF寄存器的同义词，除了对该寄存器的读/写在每个字节内进行位互换。读取返回 { SHIFTBUF[24:31], SHIFTBUF[16:23], SHIFTBUF[8:15], SHIFTBUF[0:7] }。 |

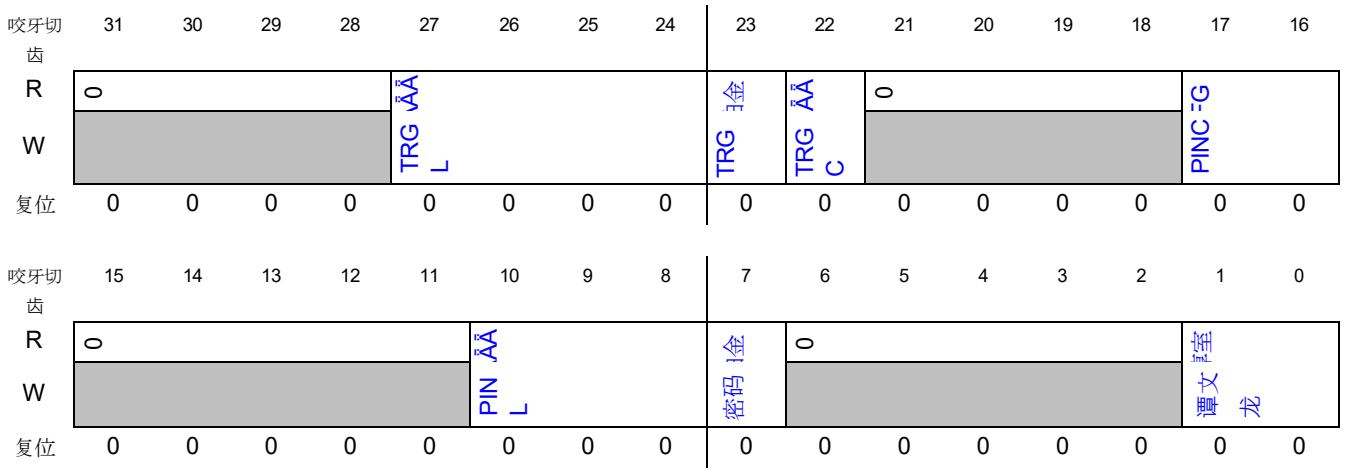
## 49.3.1.19 计时器控制N寄存器（TIMCTL0 - TIMCTL3）

### 49.3.1.19.1 偏移

对于a=0到3。

| 注册 | 偏移 |
|---|---|
| ĀĀĀ | 400h + (a × 4h) |

## 49.3.1.19.2 图示




## 49.3.1.19.3 领域

| 场地 | 职能 |
|---|---|
| 31-28<br>- | 保留。 |
| 27-24<br>TRGSEL | 触发器选择<br>TRGSEL的有效值将取决于FLEXIO_PARAM寄存器。<br>• 当TRGSRC=1时，N的有效值将取决于FLEXIO_PARAM寄存器中的PIN、TIMER、SHIFTER字段。<br>• 当TRGSRC=0时，N的有效值将取决于FLEXIO_PARAM寄存器中的TRIGGER字段。<br>关于外部触发器的选择，请参考芯片配置部分。<br>**注意：**对于一个引脚，N=0到7。对于移位器/定时器，N=0至3。 内部触发器选择配置如下。<br>• 4*N - 引脚2*N输入<br>• 4*N+1 - 移位器N状态标志<br>• 4*N+2 - 引脚2*N+1输入<br>• 4*N+3 - 定时器N触发输出 |
| 23<br>TRGPOL | 触发极性<br>0b - 触发器高电平有效<br>1b - 触发器低电平有效 |
| 22<br>TRGSRC | 触发源<br>0b - 选择外部触发 1b - 选择内部触发 |
| 21-18<br>- | 保留。 |

| | |
|---|---|
| 17-16<br><br>铂尼弗格<br>(PINCFG) | 计时器引脚配置<br><br>配置定时器引脚的方向。配置PINCFG=11将立即生效。<br>00b - 禁用定时器引脚输出<br>01b - 定时器引脚开漏或双向输出使能 10b - 定时器引脚双向输出数据<br>11b - 定时器引脚输出 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| 15-11<br>- | 保留。 |
| 10-8<br>PINSEL | 计时器引脚选择<br>选择定时器的输入或输出所使用的引脚。PINSEL=i将选择FXIO_Di引脚。 |
| 7<br>PINPOL | 计时器引脚极性<br>0b - 引脚为高电平有效<br>1b - 引脚为低电平有效 |
| 6-2<br>- | 保留。 |
| 1-0<br>ÃÃÃ | 计时器模式<br>在8位波特计数器模式下，计数器和比较寄存器的低8位用于配置定时器移位时钟的波特率，而高8位用于配置移位器的位数。<br>在8位PWM高电平模式下，计数器和比较寄存器的低8位用于配置定时器移位时钟的高周期，而高8位用于配置定时器移位时钟的低周期。移位器的位数是用另一个定时器或外部信号来配置的。<br>在16位计数器模式下，计数器和比较寄存器的全部16位被用来配置移位时钟的波特率或移位器的位计数。<br>00b - 定时器被禁用。<br>01b - 双8位计数器波特模式。<br>10b - 双8位计数器PWM高电平模式。11b - 单个16位计数器模式。 |

## 49.3.1.20 计时器配置N寄存器（TIMCFG0 - TIMCFG3）。

### 49.3.1.20.1 偏移

对于a=0到3。

| 注册 | 偏移 |
|---|---|
| TIMCFGa | 480h + (a × 4h) |

### 49.3.1.20.2 职能

当N被4偶数整除时（例如：定时器0），使用定时器N-

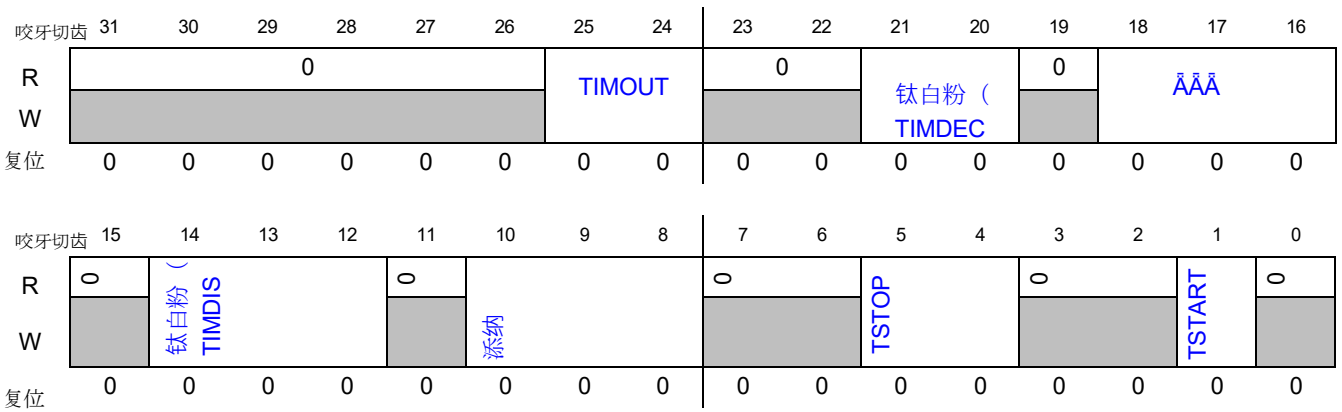
1启用或禁用的选项被保留。

## 注意事项

下面指定的引脚和触发电平/边沿是指被TIMCTL寄存器中的PINPOL或TRGPOL设置修改后的信号状态。例如，"触发器低电平"

意味着如果TRGPOL被设置为1（低电平有效），则触发器实际上处于逻辑电平1。

### 49.3.1.20.3 图示




### 49.3.1.20.4 领域

| 场地 | 职能 |
|---|---|
| 31-26<br>- | 保留。 |
| 25-24<br>TIMOUT | 计时器输出<br>配置定时器输出的初始状态，以及它是否受定时器复位的影响。<br>00b - 计时器输出在启用时为逻辑一，不受定时器复位影响 01b - 计时器输出在启用时为逻辑零，不受定时器复位影响 10b - 计时器输出在启用时和定时器复位时为逻辑一<br>11b - 定时器输出在启用时和定时器复位时为逻辑零 |
| 23-22<br>- | 保留。 |
| 21-20<br>钛白粉（<br>TIMDEC | 计时器递减<br>配置定时器的递减源和移位时钟的源。00b - 在FlexIO时钟上递减计数器，移位时钟等于定时器输出。<br>01b - 触发器输入上的递减计数器（两个边），移位时钟等于定时器输出。10b - 递减计数器在引脚输入（两个边沿），移位时钟等于引脚输入。<br>11b - 在触发器输入（两个边沿）上减少计数器，移位时钟等于触发器输入。 |
| 19<br>- | 保留。 |

| | |
|---|---|
| 18-16<br>ĀĀĀ | 定时器复位<br><br>配置导致定时器计数器（以及可选的定时器输出）被重置的条件。在8位计数器模式下，定时器复位将只复位配置波特率的低8位。在所有其他模式下，定时器复位将重置计数器的全部16位。<br>000b - 定时器永不复位 001b<br>- 保留<br>010b - 定时器引脚上的定时器复位等于定时器输出 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | 011b - 定时器触发器等于定时器输出时的定时器复位 100b<br>- 定时器引脚上升沿的定时器复位<br>101b - 保留<br>110b - 触发器上升沿的定时器复位<br>111b - 触发器上升或下降沿的定时器复位 |
| 15<br>- | 保留。 |
| 14-12<br>钛白粉（<br>TIMDIS | 定时器禁用<br>配置导致定时器被禁用并停止递减的条件。<br>000b - 定时器从未被禁用<br>001b - 在定时器N-1禁用时禁用定时器<br>010b - 在定时器比较时禁用定时器<br>011b - 在定时器比较和触发低电平时禁用定时器 100b - 在<br>引脚上升或下降沿禁用定时器<br>101b - 在引脚上升或下降沿禁用定时器，条件是触发器为高电平 110b - 在触<br>发器下降沿禁用定时器<br>111b - 保留 |
| 11<br>- | 保留。 |
| 10-8<br>添纳 | 定时器启用<br>配置导致定时器被启用并开始递减的条件。<br>000b - 始终启用定时器<br>001b - 定时器N-1使能时启用定时器 010b -<br>触发器高电平时启用定时器<br>011b - 触发器高电平和引脚高电平时启用计时器 100b<br>- 引脚上升沿时启用计时器<br>101b - 在引脚上升沿和触发器高电平时启用定时器 110b -<br>在触发器上升沿启用定时器<br>111b - 触发器上升沿或下降沿启用计时器 |
| 7-6<br>- | 保留。 |
| 5-4<br>TSTOP | 定时器停止位<br>停止位可以在定时器比较（每个字之间）或定时器禁用时添加。当停止位被启用时，配置的移位器将在定时器被禁用时输出停止位的内容。当停止位在定时器禁用时被启用，定时器保持禁用，直到移位时钟的下一个上升沿。如果同时配置了定时器比较和定时器禁用，在定时器禁用时只插入一个停止位。<br>00b - 禁用停止位<br>01b - 定时器比较时启用停止位 10b - 定时<br>器禁用时启用停止位<br>11b - 停止位在定时器比较和定时器禁用时被启用 |

| | |
|---|---|
| 3-2<br>- | 保留。 |
| 1<br>TSTART | 定时器启动位<br>当启动位被启用时，配置的移位器将在定时器被启用时输出启动位的内容，定时器计数器将在移位时钟的第一个上升沿从比较寄存器中重新加载。<br>0b - 禁用启动位 1b -<br>启用启动位 |
| 0 | 保留。 |

| 场地 | 职能 |
|---|---|
| - | |

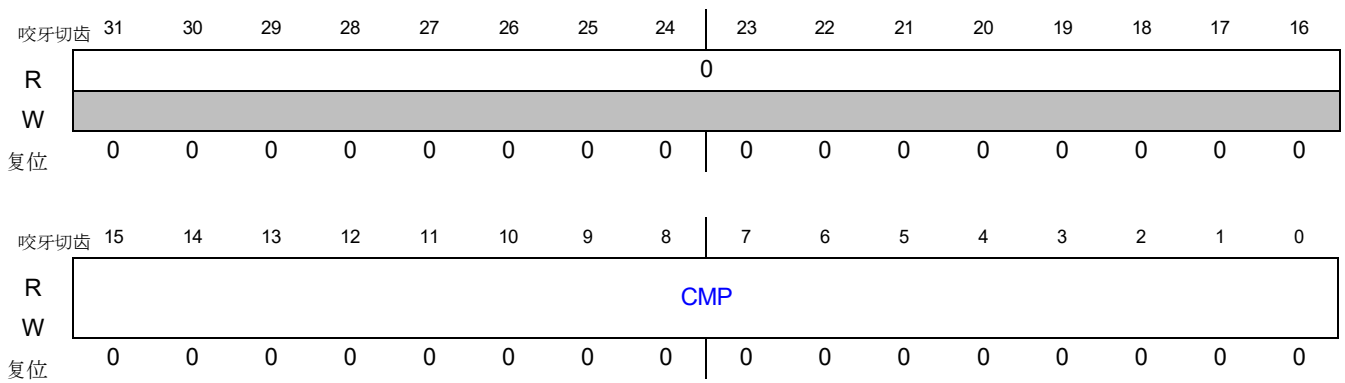
## 49.3.1.21 定时器比较N寄存器（TIMCMP0 - TIMCMP3）。

### 49.3.1.21.1 偏移

对于a=0到3。

| 注册 | 偏移 |
|---|---|
| TIMCMPa | 500h + (a × 4h) |

### 49.3.1.21.2 图示




### 49.3.1.21.3 领域

| 场地 | 职能 |
|---|---|
| 31-16<br>- | 保留。 |

| | |
|---|---|
| 15-0<br>CMP | 定时器比较值<br>定时器的比较值在定时器首次启用时、定时器复位时和定时器递减到零时被加载到定时器计数器。<br>在8位波特计数器模式下，低8位配置波特率分频器，等于（CMP[7:0] + 1）*2。上面的8位配置每个字的位数，等于（CMP[15:8] + 1）/2。<br>在8位PWM高电平模式下，低8位配置输出的高电平周期为（CMP[7:0] + 1），高8位配置输出的低电平周期为（CMP[15:8] + 1）。<br>在16位计数器模式下，比较值可以用来产生波特率分频器（如果移位时钟源是定时器输出），等于（CMP[15:0] + 1）* 2。当移位时钟源为引脚或触发器输入时，比较寄存器被用来设置每个字的位数等于（CMP[15:0] + 1）/2。 |

## 49.4 职能描述

### 49.4.1 时钟和复位

#### 49.4.1.1 功能性时钟

FlexIO功能时钟与总线时钟是异步的，在低功率模式下可以保持启用。在访问任何FlexIO寄存器之前，FlexIO功能时钟必须被启用。只要FlexIO功能时钟比总线时钟至少快2倍，就可以设置CTRL[FASTACC]位以支持快速寄存器访问。

#### 49.4.1.2 总线时钟

总线时钟只用于对控制和配置寄存器的总线访问。

#### 49.4.1.3 芯片复位

FlexIO的逻辑和寄存器在芯片复位时被重置为其默认状态。

#### 49.4.1.4 软件复位

FlexIO在其控制寄存器中实现了一个软件复位位。CTRL[RST]将把所有的逻辑和寄存器复位到其默认状态，除了CTRL本身。

### 49.4.2 变速器操作

移位器负责缓冲并将数据移入或移出FlexIO。移位、加载和存储事件的时间由通过SHIFTCTL[TIMSEL]寄存器分配给移位器的计时器控制。移位器被设计为支持DMA、中断或轮询操作。下面的框图提供了移位器微结构的详细视图。

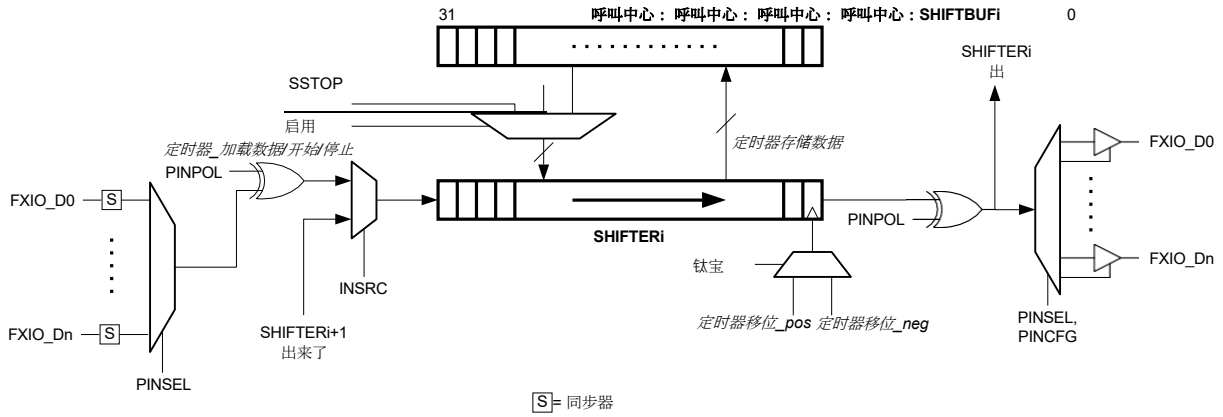



**图49-2.移位器的微结构**

### 49.4.2.1 发送模式

当配置为传输模式（SHIFTCTL[SMOD]=传输）时，移位器将从SHIFTBUF寄存器加载数据，并在指定的定时器发出加载事件信号时将数据移出。通过配置移位器和定时器中的SHIFTCFG[SSTART]、TIMCFG[TSTART]或SHIFTCFG[SSTOP]、TIMCFG[TSTOP]寄存器，也可以在SHIFTBUF数据之前/之后自动加载一个可选的启动/停止位。请注意，如果停止位被启用，当移位器最初被配置为传输模式时，移位器将立即加载停止位。

当数据从 SHIFTBUF
寄存器加载到移位器或移位器最初被配置为传输模式时，移位器状态标志（SHIFTSTAT[SSF]）和任何启用的中断或 DMA 请求将被设置。当新数据被写入 SHIFTBUF 寄存器时，该标志将被清除。

当试图从空的SHIFTBUF寄存器加载数据时，移位器错误标志（SHIFTERR[SEF]）和任何启用的中断将被设置（缓冲区不足）。该标志可以通过写入逻辑1而被清除。

### 49.4.2.2 接收模式

当配置为接收模式（SHIFTCTL[SMOD]=接收）时，当指定的定时器发出存储事件的信号时，移位器将移入数据并将数据存储到SHIFTBUF寄存器。通过配置移位器和定时器中的SHIFTCFG[SSTART]、TIMCFG[TSTART]或SHIFTCFG[SSTOP]、TIMCFG[TSTOP]寄存器，可以在移位器数据被采样之前/之后启用对启动/停止位的检查。

当数据从移位器存储到SHIFTBUF寄存器时，移位器状态标志（SHIFTSTAT[SSF]）和任何启用的中断或DMA请求将被设置。当数据被从SHIFTBUF寄存器中读出时，该标志将被清除。

当试图将数据存储到一个完整的SHIFTBUF寄存器时（缓冲区超限），或在启动/停止位检查中发生不匹配时，移位器错误标志（SHIFTERR[SEF]）和任何启用的中断将被设置。该标志可以通过写逻辑1来清除。

#### 49.4.2.3 匹配商店模式

当配置为匹配存储模式（SHIFTCTL[SMOD]=匹配存储）时，移位器将移入数据，检查匹配结果，并在指定的定时器发出存储事件信号时将匹配数据存储到SHIFTBUF寄存器。通过配置SHIFTCFG[SSTART], TIMCFG[TSTART]或SHIFTCFG[SSTOP], TIMCFG[TSTOP]，可以在移位器数据被采样之前/之后启用对启动/停止位的检查。
移位器和定时器中的寄存器。使用SHIFTBUF[31:16]配置要匹配的数据和SHIFTBUF[15:0]屏蔽匹配结果，可以比较多达16位的数据。

当匹配发生并且匹配数据从移位器存储到SHIFTBUF寄存器时，移位器状态标志（SHIFTSTAT[SSF]）和任何启用的中断或DMA请求将被设置。当匹配的数据从SHIFTBUF寄存器中读出时，该标志将被清除。

当试图将匹配的数据存储到一个完整的SHIFTBUF寄存器时（缓冲区超限），或在启动/停止位检查中发生不匹配时，移位器错误标志（SHIFTERR[SEF]）和任何启用的中断将被设置。该标志可以通过写逻辑1来清除。

#### 49.4.2.4 匹配连续模式

当配置为匹配连续模式（SHIFTCTL[SMOD]=匹配连续）时，移位器将移入数据，并在指定的定时器发出移位事件信号时连续检查匹配结果。使用SHIFTBUF[31:16]配置要匹配的数据，SHIFTBUF[15:0]屏蔽匹配结果，可以比较多达16位的数据。

匹配发生时，移位器状态标志（SHIFTSTAT[SSF]）和任何启用的中断或DMA请求将被设置。一旦移位器数据和SHIFTBUF寄存器之间不再匹配，该标志将自动清除。

当匹配发生时，移位器错误标志（SHIFTERR[SEF]）和任何启用的中断将被设置。当从SHIFTBUF寄存器中读出或以逻辑1写入时，该标志将被清除。

### 49.4.3 定时器操作

FlexIO
16位定时器控制移位寄存器的加载、移位和存储，计数器加载比较寄存器的内容并在FlexIO时钟上递减到零。它们可以执行通用的定时器功能，如产生一个时钟或选择输出或一个PWM波形。计时器可以被配置为在触发器、引脚或移位器条件下启用；总是或仅在触发器或引脚边缘递减；在触发器或引脚条件下复位；在触发器或引脚条件下或在定时器比较时禁用。计时器可以选择性地包括一个启动条件和/或停止条件。

每个定时器都是独立工作的，尽管一个定时器可以被配置为与前一个定时器同时启用或禁用（例如：定时器1可以与定时器0同时启用或禁用），一个定时器的输出可以用来触发任何其他定时器。每个定时器使用的触发器是独立配置的，可以配置为定时器输出、移位器状态标志、引脚输入或外部触发器输入（关于外部触发器连接的细节，请参考芯片配置部分）。触发器的配置与引脚的配置是分开的，引脚可以被配置为输入、输出数据或输出使能。

在设置定时器模式（TIMOD）之前，应配置定时器配置寄存器（TIMCFGn）。

#### 49.4.3.1 定时器8位波特计数器模式

在8位波特计数器模式下，16位计数器被分为两个8位计数器。下面的8位用于配置移位时钟的波特率，上面的8位用于配置传输中移位时钟的边沿数。当低8位递减到零时，定时器输出被切换，低8位从比较寄存器中重新加载。上层8位只有在下层8位等于0时才递减，并递减。

当上面的8位数等于0并递减时，就发生了一个定时器比较事件。定时器状态标志在定时器比较事件中被设置。

### 49.4.3.2 定时器8位高电平PWM模式

在8位高PWM模式下，16位计数器被分成两个8位计数器。低8位用于配置定时器输出高电平周期，高8位用于配置定时器输出低电平周期。当输出为高电平时，低8位会递减。当低8位等于0并递减时，定时器输出被清零，低8位从比较寄存器中重新加载。当输出为低电平时，上面的8位递减。当上面的8位数等于0并且递减时，定时器输出被设置，上面的8位数从比较寄存器中重新加载。

当上面的8位数等于0并递减时，就发生了一个定时器比较事件。定时器状态标志在定时器比较事件中被设置。

### 49.4.3.3 定时器16位计数器模式

在16位计数器模式下，16位计数器可用于配置移位时钟的波特率（如：TIMDEC[1:0] !=10或11）或传输中移位时钟的边缘数（如：TIMDEC[1:0] =10或11）。当16位计数器等于零并递减时，定时器输出切换，计数器从比较寄存器中重新加载。

当16位计数器等于0并递减时，就发生了一个定时器比较事件。定时器状态标志在定时器比较事件中被设置。

### 49.4.3.4 计时器启用和启动位

当TIMOD被配置为所需模式，并且检测到由定时器使能（TIMENA）配置的条件时，就会发生以下事件。

- 计时器将加载比较寄存器的当前值，并按照TIMDEC的配置开始递减。
- 计时器输出将根据TIMOUT配置进行设置。
- 由该定时器控制的传输移位器将输出其起始位值，或从移位缓冲器加载移位寄存器并输出第一个位，如SSTART所配置。

如果定时器启动位被启用，定时器计数器将在定时器开始递减后，在移位时钟的第一个上升沿重新加载比较寄存器。如果在第一个上升沿之前移位时钟没有下降沿（例如，当TIMOUT=1时），被配置为在下降沿移位并在第一个移位上加载的移位器将不能正确加载。

#### 49.4.3.5 计时器的减少和复位

然后，定时器将根据TIMOD和TIMDEC字段产生定时器输出和定时器移位时钟。移位时钟要么等于定时器输出（当TIMDEC != 10或11时），要么等于递减时钟（当TIMDEC = 10或11时）。
当TIMDEC被配置为从引脚或触发器递减时，定时器将在上升沿和下降沿递减。

当定时器按照TIMRST字段的配置进行复位时，那么定时器计数器将再次加载比较寄存器的当前值，定时器的输出也可能受到TIMOUT中配置的复位的影响。

一般来说，当定时器计数器递减到零时，会触发定时器比较事件。定时器比较事件将导致定时器计数器加载定时器比较寄存器的内容，定时器输出切换，任何配置的发送移位寄存器加载，和任何配置的接收移位寄存器存储。根据不同的模式，定时器状态标志也可能被设置。

#### 49.4.3.6 定时器禁用和停止位

当定时器被配置为在每次比较时增加一个停止位时，将发生以下额外事件。

- 由该定时器控制的传输移位器将输出其停止位值（如果由 SSTOP 配置）。
- 由该定时器控制的接收移位器将把移位寄存器的内容存储在其移位缓冲器中，由SSTOP配置。
- 在比较后移位器时钟的第一个上升沿，定时器计数器将重新加载比较寄存器的当前值。

当定时器被配置为在每次比较时插入一个停止位时，传输移位器必须被配置为在第一次移位时加载。

当检测到由定时器禁用（TIMDIS）配置的条件时，会发生以下事件。

- 定时器计数器将重新加载比较寄存器的当前值，并按照TIMDEC的配置开始递减。
- 计时器输出将被清除。
- 由该定时器控制的传输移位器将输出其停止位值（如果由 SSTOP 配置）。
- 由该定时器控制的接收移位器将把移位寄存器的内容存储在其移位缓冲器中

，由SSTOP配置。

如果定时器停止位被启用，定时器计数器将继续递减，直到检测到移位时钟的下一个上升沿，这时它将结束。

定时器使能条件可以在与定时器禁用条件相同的周期内检测到（如果定时器停止位被禁用），或者在禁用条件后的移位时钟的第一个上升沿检测到（如果停止位被启用）。启用了停止位的接收移位寄存器将把移位寄存器的内容存储到移位缓冲器中，并在定时器处于停止状态条件下，在配置的移位边缘验证输入数据的状态。如果在定时器禁用和移位时钟的下一个上升沿之间没有配置的边沿，那么最后的存储和验证就不会发生。

### 49.4.4 引脚操作

每个定时器和移位器的引脚配置可以被配置为使用任何FlexIO引脚的极性。每个定时器和移位器都可以被配置为输入、输出数据、输出使能或双向输出。被配置为输出使能的引脚可以用作漏极开路（反极性，因为输出使能断言会导致引脚上输出逻辑零）或控制双向输出上的使能。任何定时器或移位器都可以被配置为控制一个引脚的输出使能，其中双向输出数据是由另一个定时器或移位器驱动。

#### 49.4.4.1 引脚同步化

当把一个引脚配置为输入时（这包括配置为引脚输入的定时器触发器），在信号被定时器或移位器使用之前，输入信号首先与FlexIO时钟同步。当使用外部引脚输入来产生输出或控制移位器时，这将带来0.5到1.5个FlexIO时钟周期的小延迟。这将最大的设置时间设定为1.5个FlexIO时钟周期。

如果一个输入被一个以上的定时器或移位器使用，那么同步发生一次，以确保任何边缘被使用该输入的所有定时器和移位器在同一周期内看到。

请注意，FlexIO引脚也是内部连接的，配置一个FlexIO移位器或定时器在一个未使用的引脚上输出数据，将建立一个内部连接，允许其他移位器和定时器使用这个引脚作为输入。这允许移位器的输出被用来触发定时器，或者定时器的输出被移位到移位器中。这个路径也是与FlexIO时钟同步的，因此会产生1个周期的延迟。

因此，当使用引脚输入作为定时器触发器、定时器时钟或移位器数据输入时，会出现以下同步延迟。

1. 0.5 - 1.5个FlexIO外部引脚的时钟周期
2. 1个内部驱动引脚的FlexIO时钟周期

关于特定应用（SPI主站、SPI从站、I2C主站、I2S主站、I2S从站）的输出有效时间和输入设置时间等计时注意事项，请参考FlexIO应用信息部分。

### 49.4.5 中断和DMA请求

下表说明了可以产生FlexIO中断和DMA请求的状态标志。

**表49-3.FlexIO中断和DMA请求**

| 旗帜 | 描述 | 中断 | DMA请求 | 低功率唤醒 |
|---|---|---|---|---|
| SSF | 移位器状态标志。 | Y | Y | Y |
| SEF | 移位器错误标志。 | Y | N | Y |
| TSF | 计时器状态标志。 | Y | N | Y |

### 49.4.6 周边触发器

FlexIO外设触发器与其他外设的连接是设备特定的。

#### 49.4.6.1 输出触发器

每个FlexIO定时器产生的输出触发器相当于定时器的输出。输出触发器不受定时器引脚极性配置的影响。

#### 49.4.6.2 输入触发器

FlexIO支持多个外部触发输入，可用于触发一个或多个FlexIO定时器。外部触发器与FlexIO功能时钟同步，必须在FlexIO功能时钟的至少两个周期内断言，才能正确采样。

## 49.5 申请信息

本节提供了各种FlexIO模块应用的例子。

### 49.5.1 UART传输

可以使用一个定时器、一个移位器和一个引脚（如果支持CTS则为两个引脚）来支持UART传输。开始和停止位的插入是自动处理的，并且可以使用DMA控制器支持多次传输。计时器状态标志可以用来指示每个字的停止位何时被传送。

断裂和空闲字符需要软件干预，在传输断裂或空闲字符之前，应该改变SSTART和SSTOP字段，以传输所需的状态，要传输的数据必须等于0xFF或0x00。支持第二个停止位需要使用软件将停止位插入到数据流中（并增加要传输的位数）。请注意，当对SHIFTBUFn（或SHIFTBUFBIS用于传输MSB优先）进行字节写入时，寄存器的其余部分保持不变，允许地址标记位或额外的停止位保持不受干扰。

FlexIO不支持自动插入奇偶校验位。

**表49-4.UART传输配置**

| 注册 | 价值 | 评论 |
|---|---|---|
| SHIFTCFGn | 0x0000_0032 | 配置起始位为0，停止位为1。 |
| SHIFTCTLn | 0x0003_0002 | 使用定时器0在时钟的位置上配置发送，输出数据在引脚上。<br>0.通过设置PINPOL可以反转输出数据，或者通过设置PINPOL=0x1和PINCFG=0x1可以支持开漏。 |
| TIMCMPn | 0x0000_0F01 | 配置8位传输，波特率为FlexIO时钟的4倍。设置TIMCMP[15:8]=（比特数×2）-1。设置TIMCMP[7:0]=（波特率分频器/2）-1。 |
| ÃÃTIMCFGn | 0x0000_2222 | 配置起始位、停止位，在触发器断言时启用，在比较时停用。通过配置TIMEN=0x3，可以支持CTS。 |
| 术语表（TIMCTLn | 0x01C0_0001 | 配置双8位计数器，使用移位器0状态标志作为倒置的内部触发源。通过配置PINSEL=0x1（用于针脚1）和PINPOL=0x1，可以支持CTS。 |
| SHIFTBUFn | 要传输的数据 | 传输数据可以写入SHIFTBUF[7:0]来启动8位传输，使用移位器状态标志来指示何时可以使用中断或DMA请求写入数据。可以通过写到SHIFTBUFBBS[7:0]寄存器来代替支持MSB优先传输。 |

## 49.5.2 UART接收

可以使用一个定时器、一个移位器和一个引脚来支持UART接收（如果支持RTS，则使用两个定时器和两个引脚）。开始和停止位的验证是自动处理的，使用DMA控制器可以支持多次传输。计时器状态标志可用于指示每个字的停止位何时被接收。

FlexIO不支持对接收到的数据进行三重投票，数据只在每个比特的中间被采样一次。另一个定时器可以用来实现对传入数据的突波过滤，另一个定时器也可以用来检测可编程长度的空闲行。断裂字符将导致错误标志的设置，移位缓冲器寄存器将返回0x00。

FlexIO不支持对奇偶校验位的自动验证。

**表49-5.UART接收器配置**

| 注册 | 价值 | 评论 |
|---|---|---|
| 呼叫中心（SHIFTCFGn | 0x0000_0032 | 配置起始位为0，停止位为1。 |
| SHIFTCTLn | 0x0080_0001 | 使用定时器0在时钟的负边上配置接收，输入数据在引脚上。<br>0.可以通过设置PINPOL来反转输入数据。 |
| TIMCMPn | 0x0000_0F01 | 配置8位传输，波特率为FlexIO时钟的4倍。设置TIMCMP[15:8]=（比特数×2）-1。设置TIMCMP[7:0]=（波特率分频器/2）-1。 |
| ÃÃTIMCFGn | 0x0204_2422 | 配置起始位、停止位，在引脚定位时启用，在比较时禁用。用TIMOUT=0x2和TIMRST=0x4启用与接收数据的再同步。 |
| 术语表(TIMCTLn) | 0x0000_0081 | 使用倒置的针0输入配置双8位计数器。 |
| SHIFTBUFn | 要接收的数据 | 收到的数据可以从SHIFTBUFBYS[7:0]读取，使用移位器状态标志来指示何时可以使用中断或DMA请求读取数据。<br>可以通过从SHIFTBUFBIS[7:0]寄存器读取来支持MSB优先传输。 |

带有RTS配置的UART接收器使用一个第二定时器来产生RTS输出。当检测到起始位时，RTS将被断言，而当数据从移位缓冲寄存器中被读取时，RTS将被否定。在

RTS断言时，没有检测到起始位，接收到的数据会被直接忽略。

表49-6.带有RTS配置的UART接收器

| 注册 | 价值 | 评论 |
| --- | --- | --- |
| SHIFTCFGn | 0x0000_0032 | 配置起始位为0，停止位为1。 |
| SHIFTCTLn | 0x0080_0001 | 使用定时器0在时钟的负边上配置接收，输入数据在引脚上。<br>0.可以通过设置PINPOL来反转输入数据。 |
| TIMCMPn | 0x0000_0F01 | 配置8位传输，波特率为FlexIO时钟的4倍。设置TIMCMP[15:8]=（比特数×2）-1。设置TIMCMP[7:0]=（波特率分频器/2）-1。 |
| ÅÅTIMCFGn | 0x0204_2522 | 配置起始位、停止位，在触发器断言时启用引脚位置，在比较时禁用。用TIMOUT=0x2和TIMRST=0x4启用与接收数据的再同步。 |
| 术语表(TIMCTLn) | 0x03C0_0081 | 配置双8位计数器，使用倒置的Pin 0输入。触发器是使用倒置的针脚1输入的内部触发。 |
| TIMCMP(n+1) | 0x0000_FFFF | 永远不要比较。 |
| TIMCFG(n+1) | 0x0030_6100 | 在定时器N启用时启用，在触发器下降沿禁用。触发时递减，以确保没有比较。 |
| TIMCTL(n+1) | 0x0143_0083 | 配置16位计数器并在针脚1上输出。 触发器是使用移位器0标志的内部触发。 |
| SHIFTBUFn | 要接收的数据 | 收到的数据可以从SHIFTBUFBYS[7:0]读取，使用移位器状态标志来指示何时可以使用中断或DMA请求读取数据。<br>可以通过从SHIFTBUFBIS[7:0]寄存器读取来支持MSB优先传输。 |

## 49.5.3 SPI主站

可以使用两个定时器、两个移位器和四个引脚来支持SPI主模式。 可以支持CPHA=0或CPHA=1， 可以使用DMA控制器支持传输。对于CPHA=1， 选择可以在多次传输中保持断言，定时器状态标志可以用来指示传输的结束。

停止位用于保证在从属选择被否定和下一次传输之前至少有一个时钟周期。通过核心或DMA写到传输缓冲区，用于启动每次传输。

由于同步延迟，串行输入数据的设置时间为1.5个FlexIO时钟周期，所以最大波特率要除以4的FlexIO时钟频率。

**表49-7.SPI主站（CPHA=0）配置**

| 注册 | 价值 | 评论 |
|---|---|---|
| SHIFTCFGn | 0x0000_0000 | 启动和停止位禁用。 |
| SHIFTCTLn | 0x0083_0002 | 使用定时器0在时钟的负边配置发送，输出数据在引脚0。 |
| SHIFTCFG(n+1) | 0x0000_0000 | 启动和停止位禁用。 |
| SHIFTCTL(n+1) | 0x0000_0101 | 使用定时器0在时钟的posedge上配置接收，输入数据在引脚1上。 |
| TIMCMPn | 0x0000_3F01 | 配置32位传输，波特率为FlexIO时钟的4倍。设置TIMCMP[15:8]=（比特数×2）-1。<br>设置TIMCMP[7:0]=（波特率分频器/2）-1。 |
| ÅÅTIMCFGn | 0x0100_2222 | 配置起始位、停止位，触发高电平时启用，比较时禁用，初始时钟状态为逻辑0。 |
| 术语表（TIMCTLn | 0x01C3_0201 | 使用引脚2的输出（移位时钟）配置双8位计数器，将移位器0标志作为反相触发器。设置PINPOL以反转输出移位时钟。 |
| TIMCMP(n+1) | 0x0000_FFFF | 永远不要比较。 |
| TIMCFG(n+1) | 0x0000_1100 | 当定时器0启用时启用，当定时器0禁用时禁用。 |
| TIMCTL(n+1) | 0x0003_0383 | 使用倒置的第3针输出（作为从属选择）配置16位计数器（永不比较）。 |
| SHIFTBUFn | 要传输的数据 | 传输数据可以写入SHIFTBUF，使用移位器状态标志来指示何时可以使用中断或DMA请求写入数据。可以通过写到SHIFTBUFBBS寄存器来代替支持MSB优先传输。 |
| SHIFTBUF(n+1) | 要接收的数据 | 收到的数据可以从SHIFTBUFBYS读取，使用移位器状态标志来指示何时可以使用中断或DMA请求读取数据。可以通过从SHIFTBUFBIS寄存器读取来支持MSB优先的传输。 |

**表49-8.SPI主站（CPHA=1）配置**

| 注册 | 价值 | 评论 |
| --- | --- | --- |
| SHIFTCFGn | 0x0000_0021 | 开始位在第一次移位时加载数据。 |

*表继续在下一页...*

## 表49-8.SPI主站（CPHA=1）配置（续）

| 注册 | 价值 | 评论 |
|---|---|---|
| SHIFTCTLn | 0x0003_0002 | 使用定时器0在时钟的posedge上配置发送，输出数据在引脚0上。 |
| SHIFTCFG(n+1) | 0x0000_0000 | 启动和停止位禁用。 |
| SHIFTCTL(n+1) | 0x0080_0101 | 使用定时器0在时钟的负边配置接收，输入数据在引脚1。 |
| TIMCMPn | 0x0000_3F01 | 配置32位传输，波特率为FlexIO时钟的4倍。设置TIMCMP[15:8]=（比特数×2）-1。<br>设置TIMCMP[7:0]=（波特率分频器/2）-1。 |
| ÃÃTIMCFGn | 0x0100_2222 | 配置起始位、停止位，触发高电平时启用，比较时禁用，初始时钟状态为逻辑0。 |
| 术语表（TIMCTLn | 0x01C3_0201 | 使用引脚2的输出（移位时钟）配置双8位计数器，将移位器0标志作为反转触发器。设置PINPOL来反转输出移位时钟。设置TIMDIS=3，只要发送缓冲区内有数据，就保持从属选择断言。 |
| TIMCMP(n+1) | 0x0000_FFFF | 永远不要比较。 |
| TIMCFG(n+1) | 0x0000_1100 | 当定时器0被启用时启用，当定时器0被禁用时禁用。 |
| TIMCTL(n+1) | 0x0003_0383 | 使用倒置的第3针输出（作为从属选择）配置16位计数器（永不比较）。 |
| SHIFTBUFn | 要传输的数据 | 传输数据可以写入SHIFTBUF，使用移位器状态标志来指示何时可以使用中断或DMA请求写入数据。可以通过写到SHIFTBUFBBS寄存器来代替支持MSB优先传输。 |
| SHIFTBUF(n+1) | 要接收的数据 | 收到的数据可以从SHIFTBUFBYS读取，使用移位器状态标志来指示何时可以使用中断或DMA请求读取数据。可以通过从SHIFTBUFBIS寄存器读取来支持MSB优先的传输。 |

## 49.5.4 SPI从机

可以使用一个定时器、两个移位器和四个引脚来支持SPI从属模式。可以支持CPHA=0或CPHA=1，可以使用DMA控制器支持传输。对于CPHA=1，选择可以在多次传输中保持断言，定时器状态标志可以用来指示传输的结束。

发送数据必须在外部从属选择发出信号之前写入发送缓冲寄存器，否则移位器错误标志将被设置。

由于同步延迟，串行输出数据的输出有效时间是2.5个FlexIO时钟周期，所以最大波特率要除以6的FlexIO时钟频率。

**表49-9.SPI从机（CPHA=0）配置**

| 注册 | 价值 | 评论 |
|---|---|---|
| 呼叫中心（SHIFTCFGn | 0x0000_0000 | 启动和停止位禁用。 |
| SHIFTCTLn | 0x0083_0002 | 使用定时器0在移位时钟的下降沿配置发送，输出数据在引脚0。 |
| SHIFTCFG(n+1) | 0x0000_0000 | 启动和停止位禁用。 |
| SHIFTCTL(n+1) | 0x0000_0101 | 使用定时器0在移位时钟的上升沿配置接收，输入数据在引脚1。 |
| TIMCMPn | 0x0000_003F | 配置32位传输。设置TIMCMP[15:0]=（比特数×2）-1。 |
| ÃÃTIMCFGn | 0x0120_6000 | 在触发器上升沿上配置使能，初始时钟状态为逻辑0，在引脚输入上配置递减。 |
| 术语表(TIMCTLn) | 0x06C0_0203 | 使用针脚2输入（移位时钟）配置16位计数器，针脚3输入（从属选择）作为反相触发器。 |
| SHIFTBUFn | 要传输的数据 | 传输数据可以写入SHIFTBUF，使用移位器状态标志来指示何时可以使用中断或DMA请求写入数据。可以通过写到SHIFTBUFBBS寄存器来代替支持MSB优先传输。 |
| SHIFTBUF(n+1) | 要接收的数据 | 收到的数据可以从SHIFTBUFBYS读取，使用移位器状态标志来指示何时可以使用中断或DMA请求读取数据。可以通过从SHIFTBUFBIS寄存器读取来支持MSB优先的传输。 |

表49-10.SPI从机（CPHA=1）配置

| 注册 | 价值 | 评论 |
| --- | --- | --- |
| SHIFTCFGn | 0x0000_0001 | 移位器被配置为在第一个移位时加载，停止位被禁用。 |
| SHIFTCTLn | 0x0003_0002 | 使用定时器0在移位时钟的上升沿配置发送，输出数据在引脚0。 |
| SHIFTCFG(n+1) | 0x0000_0000 | 启动和停止位禁用。 |
| SHIFTCTL(n+1) | 0x0080_0101 | 使用定时器0在移位时钟的下降沿配置接收，输入数据在引脚1。 |
| TIMCMPn | 0x0000_003F | 配置32位传输。设置TIMCMP[15:0]=（比特数×2）-1。 |
| ÃÃTIMCFGn | 0x0120_6602 | 配置启动位，在触发器上升沿启用，在触发器下降沿禁用，初始时钟状态为逻辑0，在引脚输入时递减。 |
| 术语表（TIMCTLn | 0x06C0_0203 | 使用针脚2输入（移位时钟）配置16位计数器，针脚3输入（从属选择）作为反相触发器。 |
| SHIFTBUFn | 要传输的数据 | 传输数据可以写入SHIFTBUF，使用移位器状态标志来指示何时可以使用中断或DMA请求写入数据。可以通过写到SHIFTBUFBBS寄存器来代替支持MSB优先传输。 |
| SHIFTBUF(n+1) | 要接收的数据 | 收到的数据可以从SHIFTBUFBYS读取，使用移位器状态标志来指示何时可以使用中断或DMA请求读取数据。可以通过从SHIFTBUFBIS寄存器读取来支持MSB优先的传输。 |

## 49.5.5 I2C主站

使用两个定时器、两个移位器和两个引脚可以支持I2C主模式。一个定时器用于产生SCL输出，一个定时器用于控制移位器。两个移位器用于每个字的发送和接收，当接收时，发送器必须发送0xFF以三态输出。FlexIO在每个字后插入一个停止位以产生/验证ACK/NACK。FlexIO在启用SCL生成之前，会等待对发送数据缓冲区的第一次写入。可以使用DMA控制器支持数据传输，移位器错误标志将在发送欠载或接收溢出时设置。

第一个定时器为整个数据包（START到Repeated START/STOP）产生位时钟，所以比较寄存器需要用数据包中的总时钟边数（减去1）来编程。当引脚等于输出时，该定时器支持使用复位计数器进行时钟拉伸（尽管这使时钟高电平和时钟低电平周期都至少各增加1个FlexIO时钟周期）。第二个定时器使用SCL输入引脚来控制发送/接收移位寄存器，这使SDA数据保持时间多了2个FlexIO时钟周期。

发送和接收移位器都需要为传输中的每个字提供服务，发送移位器在接收时必须发送0xFF，接收移位器则返回SDA引脚上实际存在的数据。发送移位器将在SCL引脚的最后一个下降沿加载1个额外的字，如果产生STOP条件，这个字应该是0x00，如果产生一个重复的START条件，则应该是0xFF。在主-受传输的最后一个字期间，发送SSTOP位应被软件设置以产生一个NACK。

如果检测到NACK，接收移位寄存器将断言一个错误中断，但软件负责生成STOP或重复START条件。如果在主传输过程中检测到NACK，中断程序应立即将0x00（如果产生STOP）或0xFF（如果产生重复START）写入传输移位寄存器。然后，软件应该等待SCL的下一个上升沿，然后禁用两个定时器。在等待重复START或STOP条件的设置延迟后，发送移位器应被禁用。

由于同步延迟，传输输出的数据有效时间是2个FlexIO时钟周期，所以最大波特率要除以6的FlexIO时钟频率。

I2C主数据的有效时间被延迟了2个周期，因为时钟输出在为发送/接收移位器提供时钟之前要通过一个同步器（以保证一些SDA保持时间）。由于SCL输出与FlexIO时钟同步，同步延迟为1个周期，然后再产生1个周期的输出。

**表49-11.I2C主站配置**

| 注册 | 价值 | 评论 |
|---|---|---|
| SHIFTCFGn | 0x0000_0032 | 启动位启用（逻辑0），停止位启用（逻辑1）。 |
| SHIFTCTLn | 0x0101_0082 | 使用定时器1在时钟上升沿配置发送，引脚0上有倒置的输出使能（漏极开路输出）。 |
| SHIFTCFG(n+1) | 0x0000_0020 | 开始位禁用，停止位启用（逻辑0），用于检测ACK/NACK。 |
| SHIFTCTL(n+1) | 0x0180_0001 | 使用定时器1在时钟的下降沿配置接收，输入数据在引脚0。 |
| TIMCMPn | 0x0000_2501 | 配置2字传输，波特率为FlexIO时钟的4倍。设置 |

*表继续在下一页...*

表49-11.I2C主站配置（续）

| 注册 | 价值 | 评论 |
| --- | --- | --- |
| | | TIMCMP[15:8] = (字数 x 18) + 1.设置TIMCMP[7:0]=（波特率分频器/2）-1。 |
| ÅÅTIMCFGn | 0x0102_2222 | 配置起始位，停止位，触发高电平时启用，比较时禁用，如果输出等于引脚则复位。初始时钟状态为逻辑0，不受复位影响。 |
| 术语表（TIMCTLn | 0x01C1_0101 | 使用引脚1输出使能（SCL开漏）配置双8位计数器，移位器0标志作为反相触发器。 |
| TIMCMP(n+1) | 0x0000_000F | 配置8位传输。设置TIMCMP[15:0]=（比特数×2）-1。 |
| TIMCFG(n+1) | 0x0020_1112 | 当定时器0被启用时启用，当定时器0被禁用时禁用，在每个字的末尾启用开始位和停止位，在引脚输入时减弱。 |
| TIMCTL(n+1) | 0x01C0_0183 | 使用倒置的针脚1输入（SCL）配置16位计数器。 |
| SHIFTBUFn | 要传输的数据 | 发送数据可以写入SHIFTBUFBBS[7:0]，使用移位器状态标志来指示何时可以使用中断或DMA请求写入数据。 |
| SHIFTBUF(n+1) | 要接收的数据 | 收到的数据可以从SHIFTBUFBIS[7:0]读取，使用移位器状态标志来指示何时可以使用中断或DMA请求读取数据。 |

## 49.5.6 I2S主站

使用两个定时器、两个移位器和四个引脚可以支持I2S主模式。一个定时器用于生成位时钟和控制移位器，一个定时器用于生成帧同步。FlexIO在启用位时钟和帧同步生成之前，会等待对传输数据缓冲区的第一次写入。使用DMA控制器可以支持数据传输，移位器错误标志将在发送欠载或接收溢出时设置。

位时钟频率是FlexIO时钟频率的一个偶数整数，初始帧同步断言与第一个位时钟边缘同时发生。计时器使用起始位来确保帧同步在第一个输出数据之前的一个时钟周期产生。

由于同步延迟，接收器输入的设置时间为1.5个FlexIO时钟周期，所以最大波特率

要除以4的FlexIO时钟频率。

## 表49-12.I2S 主站配置

| 注册 | 价值 | 评论 |
|---|---|---|
| SHIFTCFGn | 0x0000_0001 | 在第一次移位时加载传输数据，停止位禁用。 |
| SHIFTCTLn | 0x0003_0002 | 使用定时器0在时钟上升沿配置发送，输出数据在引脚0。 |
| SHIFTCFG(n+1) | 0x0000_0000 | 启动和停止位禁用。 |
| SHIFTCTL(n+1) | 0x0080_0101 | 使用定时器0在时钟的下降沿配置接收，输入数据在针脚1。 |
| TIMCMPn | 0x0000_3F01 | 配置32位传输，波特率为FlexIO时钟的4倍。设置TIMCMP[15:8]=（比特数×2）-1。<br>设置TIMCMP[7:0]=（波特率分频器/2）-1。 |
| ÃÃTIMCFGn | 0x0000_0202 | 配置启动位，在触发高电平时启用，绝不禁用。初始时钟状态为逻辑1。 |
| 术语表(TIMCTLn) | 0x01C3_0201 | 使用引脚2输出（位时钟）配置双8位计数器，将移位器0标志作为反转触发器。设置PINPOL来反转输出移位时钟。 |
| TIMCMP(n+1) | 0x0000_007F | 配置32位传输，波特率为FlexIO时钟的4倍。设置TIMCMP[15:0]=（位数×波特率除数）-1。 |
| TIMCFG(n+1) | 0x0000_0100 | 当定时器0被启用时启用，永远不会被禁用。 |
| TIMCTL(n+1) | 0x0003_0383 | 使用倒置的第3针输出（作为帧同步）配置16位计数器。 |
| SHIFTBUFn | 要传输的数据 | 传输数据可以写入SHIFTBUFBIS，使用移位器状态标志来指示何时可以使用中断或DMA请求写入数据。<br>可以通过写到SHIFTBUF寄存器来支持LSB优先传输。 |
| SHIFTBUF(n+1) | 要接收的数据 | 收到的数据可以从SHIFTBUFBIS读取，使用移位器状态标志来指示何时可以使用中断或DMA请求读取数据。可以通过从SHIFTBUF寄存器读取来支持LSB优先传输。 |

## 49.5.7 I2S从机

可以使用三个定时器、两个移位器和四个引脚支持I2S从属模式（用于单次发送和单次接收，其他发送和接收的组合也是可能的）。

发送数据必须在外部帧同步发出之前写入发送缓冲寄存器，否则移位器错误标志将被设置。

由于同步延迟，串行输出数据的输出有效时间是2.5个FlexIO时钟周期，所以最大波特率要除以6的FlexIO时钟频率。

I2S从机的输出有效时间最大为2.5个周期，因为在时钟同步上有一个最大的1.5个周期延迟，再加上1个周期来输出数据。

计时器2检测到帧同步的下降沿（新帧的开始），并在位时钟的下降沿（当帧同步被正常采样时）之前断言输出。计时器0检测到位时钟的下降沿，而计时器2的输出被断言，并在帧的长度内断言输出。计时器1检测到位时钟的上升沿，而计时器0的输出被确认，并控制移位寄存器进行32位传输。

**表49-13.I2S从机配置**

| 注册 | 价值 | 评论 |
|---|---|---|
| SHIFTCFGn | 0x0000_0000 | 启动和停止位禁用。 |
| SHIFTCTLn | 0x0103_0002 | 使用定时器1在移位时钟的上升沿配置发送，输出数据在引脚0。 |
| SHIFTCFG(n+1) | 0x0000_0000 | 启动和停止位禁用。 |
| SHIFTCTL(n+1) | 0x0180_0101 | 使用定时器1在移位时钟的下降沿配置接收，针脚1上有输入数据。 |
| TIMCMPn | 0x0000_007F | 配置每帧两个32位传输。设置TIMCMP[15:0]=（比特数×4）。- 1. |
| ÃÃTIMCFGn | 0x0020_2500 | 配置在引脚上升沿启用（反转位时钟），触发高电平（定时器2），在比较时禁用，初始时钟状态为逻辑1，在引脚输入（位时钟）时递减。 |
| 术语表(TIMCTLn) | 0x0B40_0283 | 使用倒置的针脚2输入（位时钟）配置16位计数器，以定时器2输出作为触发器。 |
| TIMCMP(n+1) | 0x0000_003F | 配置32位传输。设置TIMCMP[15:0]=（比特数×2）-1。 |
| TIMCFG(n+1) | 0x0020_2500 | 配置在引脚（位时钟）上升沿启用，触发器（定时器0）为高电平，比较时禁用，初始时钟状态为 |

*表继续在下一页...*

表49-13.I2S从机配置（续）

| 注册 | 价值 | 评论 |
| --- | --- | --- |
| | | 逻辑1，并在引脚输入（位时钟）上进行递减。 |
| TIMCTL(n+1) | 0x0340_0283 | 使用倒置的针脚2输入（位时钟）配置16位计数器，以定时器0输出作为触发器。 |
| TIMCMP(n+2) | 0x0000_0000 | 在零点（第一条边）进行比较。 |
| TIMCFG(n+2) | 0x0020_6400 | 配置在反相针（帧同步）上升沿启用，在触发器下降沿禁用（位时钟），初始时钟状态为逻辑1，在反相针输入（帧同步）时递减。 |
| TIMCTL(n+2) | 0x0440_0383 | 使用倒置的针脚3输入（帧同步）配置16位计数器，以针脚2输入（位时钟）作为触发器。 |
| SHIFTBUFn | 要传输的数据 | 传输数据可以写入SHIFTBUFBIS，使用移位器状态标志来指示何时可以使用中断或DMA请求写入数据。<br>可以通过写到SHIFTBUF寄存器来支持LSB优先传输。 |
| SHIFTBUF(n+1) | 要接收的数据 | 收到的数据可以从SHIFTBUFBIS读取，使用移位器状态标志来指示何时可以使用中断或DMA请求读取数据。可以通过从SHIFTBUF寄存器读取来支持LSB优先传输。 |

# 第50章
# FlexCAN

## 50.1 特定芯片的FlexCAN信息

### 50.1.1 实例化信息

下表总结了该模块在产品系列中各芯片的实现情况。

**表50-1.FlexCAN实例和功能**

| 芯片 | 实例 | 信息缓冲区的数量（MB） | ISO CAN FD特点[1] | PNET特征 | 外部时间刻度 | CTRL2寄存器[2]复位值 |
|---|---|---|---|---|---|---|
| S32K116 | 弹性CAN0 | 32 | 是 | 是 | 是 | 0x00A0_0000 |
| S32K118 | 弹性CAN0 | 32 | 是 | 是 | 是 | 0x00A0_0000 |
| S32K142 | 弹性CAN0 | 32 MBs | 是 | 是 | 是 | 0x00A0_0000 |
| | FlexCAN1 | 16 MBs | 没有 | 没有 | 没有 | 0x00B0_0000 |
| S32K144 | 弹性CAN0 | 32 MBs | 是 | 是 | 是 | 0x00A0_0000 |
| | FlexCAN1 | 16 MBs | 没有 | 没有 | 没有 | 0x00B0_0000 |
| | 弹性CAN2 | 16 MBs | 没有 | 没有 | 没有 | 0x00B0_0000 |
| S32K146 | 弹性CAN0 | 32 MBs | 是 | 是 | 是 | 0x00A0_0000 |
| | FlexCAN1 | 32 MBs | 是 | 没有 | 没有 | 0x00A0_0000 |
| | 弹性CAN2 | 16 MBs | 没有 | 没有 | 没有 | 0x00B0_0000 |
| S32K148 | 弹性CAN0 | 32 MBs | 是 | 是 | 是 | 0x00A0_0000 |
| | FlexCAN1 | 32 MBs | 是 | 没有 | 没有 | 0x00A0_0000 |
| | 弹性CAN2 | 32 MBs | 是 | 没有 | 没有 | 0x00A0_0000 |

1. 该功能仅适用于SIM_SDID[6]=1的产品。在CAN经典模式下，SIM_SDID[6]=0的产品仍然可以使用全部数量的信息缓冲区。
2. 位域CTRL2[PREXCEN]和CTRL2[EDFLTDIS]只在支持ISO CAN FD功能的实例中实现。

### 50.1.2 MDIS位的复位值

复位后，CAN_MCR[MDIS]位被设置。因此，FlexCAN模块在复位后被禁用。

### 50.1.3 FlexCAN外部时间刻度

在这个设备上，LPIT通道0的触发输出作为FlexCAN0的外部时间刻度源。

### 50.1.4 FlexCAN中断

FlexCAN有多个中断请求的来源。然而，其中一些来源被OR到一起，产生一个单一的中断请求。各个中断源与中断请求的映射关系见下文。

**表50-2.FlexCAN中断**

| 要求 | 142, 144 来源 | | 148个来源 |
|---|---|---|---|
| | 对于CAN0 (CAN-FD) | 适用于CAN1、CAN2（非FD）。 | 对于CAN0、CAN1、CAN2（FD特征) |
| 信息缓冲区 | 消息缓冲区0-31 | 信息缓冲区0-15 | 消息缓冲区0-31 |
| 公交车下车 | 公交车下车 | 公交车下车 | 公交车下车 |
| 误差 | • 在CAN FD帧的数据阶段出现Bit1错误或Bit1错误<br>• 在CAN FD帧的数据阶段出现比特0错误或比特0错误<br>• 确认错误<br>• 循环冗余检查错误或CAN FD帧的数据阶段的循环冗余<br>• 形式错误或CAN FD帧的数据阶段的形式错误<br>• 在CAN FD帧的数据阶段出现塞满错误或塞满错误<br>• 发送错误警告<br>• 收到错误警告 | • 位1错误<br>• 第0位错误<br>• 确认错误<br>• 循环冗余检查（CRC）错误<br>• 表格错误<br>• 填充错误<br>• 发送错误警告<br>• 收到错误警告 | • 在CAN FD帧的数据阶段出现Bit1错误或Bit1错误<br>• 在CAN FD帧的数据阶段出现比特0错误或比特0错误<br>• 确认错误<br>• 循环冗余检查错误或CAN FD帧的数据阶段的循环冗余<br>• 形式错误或CAN FD帧的数据阶段的形式错误<br>• 在CAN FD帧的数据阶段出现塞满错误或塞满错误<br>• 发送错误警告<br>• 收到错误警告 |

| | | | |
|---|---|---|---|
| | • CAN-FD错误 | | • CAN-FD错误 |
| 发送警告 | 发送警告 | 发送警告 | 发送警告 |
| 接到警告 | 接到警告 | 接到警告 | 接到警告 |

### 50.1.5 低功率模式下的FlexCAN操作

FlexCAN模块可在VLPR模式下运行。在4MHz的系统时钟下（VLPR模式），支持的最快FlexCAN传输速率为250kbps。模块中的位定时参数必须根据新的频率进行调整，但可以实现全部功能。

关于可用的电源模式的详情，请参见模块在可用的低功率模式下的操作。

### 50.1.6 FlexCAN振荡器时钟

FlexCAN振荡器的时钟是SOSCDIV2_CLK，如表25-9所述。

## 50.2 简介

FlexCAN模块是一个根据ISO 11898-1标准和CAN 2.0
B协议规范实现CAN协议的通信控制器。下图是一个总体框图，描述了FlexCAN模块中实现的主要子模块，包括一个用于存储消息缓冲区的相关存储器、接收全局屏蔽寄存器、接收单独屏蔽寄存器、接收FIFO滤波器和接收FIFO
ID滤波器。各个子模块的功能将在后续章节中描述。

**注意事项**

Rx FIFO不能在FD模式下使用。

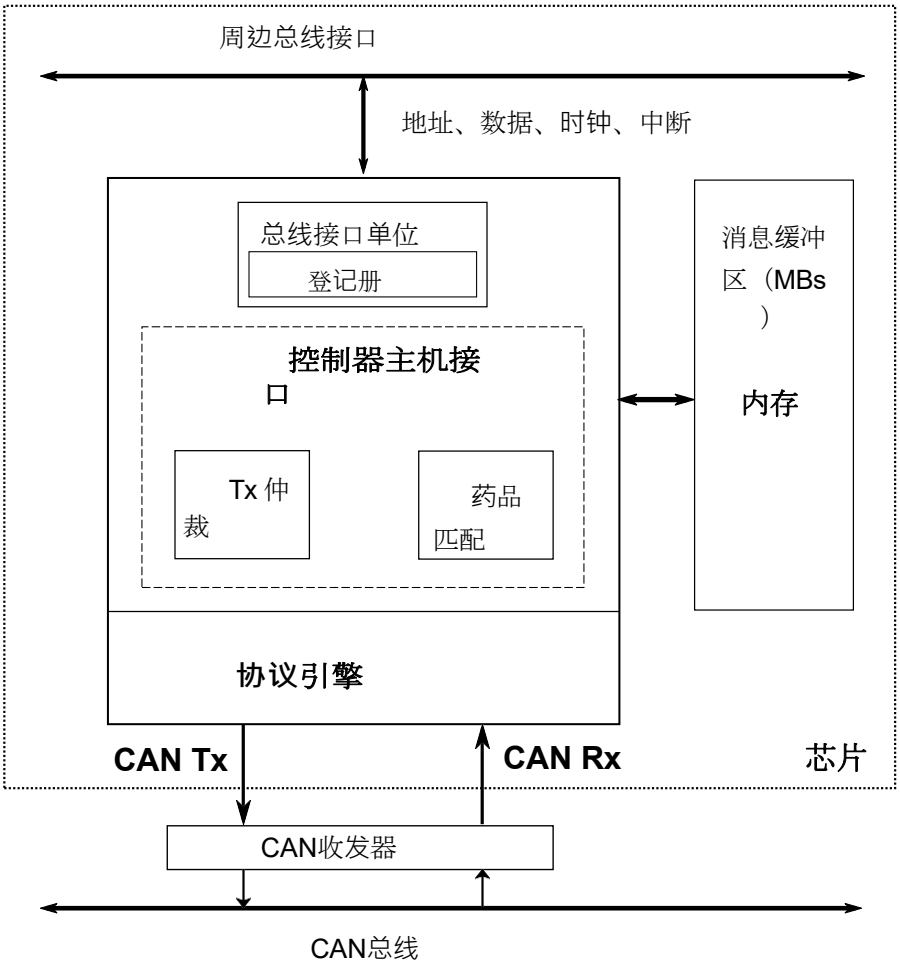



**图50-1.FlexCAN方框图**

## 50.2.1 概述

CAN协议主要被设计用来作为车辆串行数据总线，满足该领域的具体要求。

- 实时处理
- 在车辆的EMI环境中可靠地运行
- 成本效益
- 所需带宽

FlexCAN模块是对CAN协议规范、具有灵活数据速率的CAN（CAN FD）协议和CAN 2.0版本B协议的完全实现，它支持标准和扩展的消息帧和长达64字节的有效载荷以高达8 Mbps的更快速率传输。消息缓冲区存储在FlexCAN模块专用的嵌入式RAM中。有关芯片中配置的消息缓冲区的实际数量，请参见芯片的配置细节。

协议引擎（PE）子模块负责管理CAN总线上的串行通信。

- 请求RAM访问以接收和传送信息帧
- 验证收到的信息
- 进行错误处理
- 检测CAN FD信息

控制器主机接口（CHI）子模块处理接收和传输的消息缓冲区选择，负责CAN FD和非CAN FD消息格式的仲裁和ID匹配算法。

总线接口单元（BIU）子模块控制对内部接口总线的访问，以建立与CPU和其他模块的连接。通过BIU访问时钟、地址和数据总线、中断输出、DMA和测试信号。

## 50.2.2 FlexCAN模块的特点

FlexCAN模块包括这些独特的功能。

- 完全实现具有灵活数据速率的CAN（CAN FD）协议规范和CAN协议规范，2.0版B
  - 标准数据框架
  - 扩展数据框架
  - 零到六十四个字节的数据长度
  - 可编程的比特率（具体的最大速率配置见特定芯片的FlexCAN信息）。
  - 与内容有关的寻址
- 符合ISO 11898-1标准
- 灵活的邮箱可配置为存储0至8、16、32或64字节的数据长度
- 每个邮箱可配置为接收或发送，都支持标准和扩展信息
- 每个邮箱的单独Rx屏蔽寄存器
- 全功能的Rx FIFO，存储容量可达6个帧，并支持DMA的自动内部指针处理。

- 传输中止能力
- 灵活的信息缓冲区（MBs），共有32个信息缓冲区，每个数据长度为8字节，可配置为Rx或Tx。
- 用于CAN协议接口的可编程时钟源，可以是外围时钟或振荡器时钟
- 不被接收或传输结构使用的RAM可以作为通用的RAM空间使用
- 只听模式的能力
- 可编程的回环模式，支持自测试操作
- 可编程的传输优先级方案：最低的ID，最低的缓冲区数量，或最高的优先级
- 基于16位自由运行的计时器的时间戳，可选择外部时间刻度
- 全球网络时间，通过特定的信息进行同步
- 可屏蔽的中断
- 独立于传输介质（假设有一个外部收发器）。
- 由于采用了高优先级信息的仲裁方案，因此延迟时间短
- 低功率模式或与接收到的帧相匹配（假装网络）。
- 以较快的数据速率传输CAN FD报文时，具有收发器延迟补偿功能
- 可自动或通过软件处理远程请求框架
- CAN位时间设置和配置位只能在冻结模式下写入
- Tx邮箱状态（优先级最低的缓冲区或空缓冲区）
- 接收帧的识别器接受过滤命中指标（IDHIT）寄存器
- 状态1寄存器中的错误位可用来通知模块与CAN总线同步。
- 传输信息的CRC状态
- Rx FIFO全局屏蔽寄存器
- 在匹配过程中，可选择邮箱和Rx FIFO之间的优先级

- 强大的Rx FIFO
  ID过滤功能，能够将传入的ID与128个扩展的、256个标准的或512个部分（8位）的ID进行匹配，具有多达32个单独的屏蔽能力。
- 与以前的FlexCAN版本100%向后兼容
- 支持低功率下的假装联网功能：停止模式

### 50.2.3 操作模式

FlexCAN模块有这些功能模式。

- 正常模式（用户或主管）。

  在正常模式下，模块操作接收和/或发送消息帧，错误被正常处理，所有的CAN协议功能都被启用。用户模式和监督者模式在访问一些受限制的控制寄存器方面有所不同。

- 冻结模式。

  当MCR中的FRZ位被断言时，冻结模式被启用。如果启用，当MCR[HALT]被设置或在芯片级请求调试模式且MCR[FRZ_ACK
  ]被FlexCAN断言时，将进入冻结模式。在这种模式下，不进行帧的传输或接收，与CAN总线的同步性也会丧失。更多信息见冻结模式。

- 循环播放模式。

  当控制1寄存器中的LPB字段被断言时，模块就会进入这种模式。在这种模式下，FlexCAN执行内部回环，可用于自检操作。发送器的比特流输出在内部被反馈到接收器输入。Rx CAN输入引脚被忽略，Tx
  CAN输出转为隐性状态（逻辑'1'）。FlexCAN的行为与通常的发送时一样，并将自己发送的信息视为从远程节点接收的信息。在这种模式下，FlexCAN忽略了CAN帧确认字段中ACK槽期间发送的位，以确保正确接收自己的消息。发送和接收中断都会产生。

- 只听模式。

  当控制1寄存器中的LOM字段被断言时，模块进入该模式。在这种模式下，传输被禁止，所有的错误计数器被冻结，模块以CAN错误被动模式运行。只有那些被确认的信息

另一个CAN站将被接收。如果FlexCAN检测到一个没有被确认的信息，它将标记一个BIT0错误（不改变REC），就像它试图确认该信息一样。

- CAN FD 活动模式。

  在这种模式下，FlexCAN能够以交错的方式发送和接收所有根据CAN FD协议和CAN 2.0协议2.0格式的报文。CPU可以通过配置冻结模式下的MCR[FDEN]位域，将FlexCAN设置为CAN FD主动模式。

  了解哪些功能在CAN FD活动模式下可用是很重要的。这个表格列出了FD模式和经典模式之间的区别。

**表50-3.CAN经典和CAN FD之间的区别**

| 特点 | CAN 经典 | CAN FD |
|---|---|---|
| Rx FIFO DMA | 是 | 没有 |
| Rx FIFO | 是 | 没有 |
| 假装的网络支持 | 是 | 没有 |

为了实现低功耗操作，FlexCAN模块具有。

- 模块禁用模式。

  当MCR寄存器中的MDIS位被CPU断言且LPM_ACK被FlexCAN断言时，就会进入这种低功耗模式。当禁用时，该模块要求禁用CAN协议引擎和控制器主机接口子模块的时钟。通过否定MCR寄存器中的MDIS位来退出该模式。更多信息请参见模块禁用模式。

- 停止模式。

  当芯片级请求停止模式，且MCR寄存器中的LPM_ACK位被FlexCAN断言时，就进入这种低功耗模式。当进入 "停止模式 "时，模块将自身置于非活动状态，然后通知CPU可以全局关闭时钟。当停止模式的请求被取消时，就会退出该模式。更多信息见 "停止模式"。

- 假装的网络模式。

  这种模式可以被选择与停止模式一起运行。在进入这些低功率模式之前，MCR寄存器中的PNET_EN位必须被确认。一旦进入低功耗模式，CHI子块的时钟被关闭，而CAN_PE子块的时钟保持不变，这样Rx接收过程仍然处于活动状态

，以过滤进入的数据。

信息（见伪装网络模式下的接收过程），由配置寄存器（见伪装网络控制1寄存器（CTRL1_PN））定义。检测到一个唤醒事件后，会向系统发出一个唤醒中断。

## 50.3 FlexCAN信号描述

FlexCAN模块有两个I/O信号与外部芯片引脚相连。这些信号在下表中进行了总结，并在接下来的小节中进行了详细的描述。

**表50-4.FlexCAN信号描述**

| 信号 | 描述 | 输入/输出 |
|---|---|---|
| CAN Rx | CAN接收引脚 | 输入 |
| CAN Tx | CAN传输引脚 | 输出 |

### 50.3.1 CAN Rx

该引脚是CAN总线收发器的接收引脚。主导状态由逻辑电平0表示。递减状态由逻辑电平1表示。

### 50.3.2 CAN Tx

该引脚是通往CAN总线收发器的发送引脚。主导状态由逻辑电平0表示。递减状态由逻辑电平1表示。

## 50.4 内存图/寄存器定义

本节介绍FlexCAN模块中的寄存器和数据结构。模块的基础地址取决于芯片的特定内存图。

### 50.4.1 FlexCAN内存映射

FlexCAN模块的内存图如下表所示。

FlexCAN占用的地址空间有128个字节，用于从模块基址开始的寄存器，然后是从地址偏移量0x0080开始的嵌入式RAM。

每个单独的寄存器都由其完整的名称和相应的助记符来标识。访问类型可以是Supervisor（S）或Unrestricted（U）。大多数寄存器可以通过在MCR寄存器中的SUPV字段编程，配置为具有监督者或非限制性的访问。这些寄存器在表50-5的访问列中被标识为S/U。

**表50-5.寄存器访问和复位信息**

| 注册 | 访问类型 | 受硬重置的影响 | 受软复位的影响 |
|---|---|---|---|
| 模块配置寄存器（CAN_MCR） | S | 是 | 是 |
| 控制1寄存器（CAN_CTRL1）。 | S/U | 是 | 没有 |
| 自由运行定时器寄存器（CAN_TIMER）。 | S/U | 是 | 是 |
| Rx邮箱全局屏蔽寄存器（CAN_RXMGMASK）。 | S/U | 没有 | 没有 |
| Rx Buffer 14 Mask寄存器（CAN_RX14MASK）。 | S/U | 没有 | 没有 |
| Rx缓冲区15屏蔽寄存器（CAN_RX15MASK）。 | S/U | 没有 | 没有 |
| 错误计数器寄存器（CAN_ECR） | S/U | 是 | 是 |
| 错误和状态1寄存器（CAN_ESR1）。 | S/U | 是 | 是 |
| 中断掩码1寄存器（CAN_IMASK1）。 | S/U | 是 | 是 |
| 中断标志1寄存器（CAN_IFLAG1）。 | S/U | 是 | 是 |
| 控制2寄存器（CAN_CTRL2）。 | S/U | 是 | 没有 |
| 错误和状态2寄存器（CAN_ESR2） | S/U | 是 | 是 |
| CRC寄存器（CAN_CRCR） | S/U | 是 | 是 |
| Rx FIFO全局屏蔽寄存器（CAN_RXFGMASK） | S/U | 没有 | 没有 |
| Rx FIFO信息寄存器（CAN_RXFIR）。 | S/U | 没有 | 没有 |
| CAN位定时寄存器（CAN_CBT）。 | S/U | 是 | 没有 |
| 消息缓冲区 | S/U | 没有 | 没有 |
| Rx单个屏蔽寄存器 | S/U | 没有 | 没有 |
| 假装网络控制1寄存器（CAN_CTRL1_PN） | S/U | 是 | 是 |
| 假装网络控制2寄存器（CAN_CTRL2_PN） | S/U | 是 | 是 |
| 假装网络唤醒匹配寄存器（CAN_WU_MTC）。 | S/U | 是 | 是 |
| 假装网络ID过滤器1寄存器（CAN_FLT_ID1）。 | S/U | 是 | 是 |
| 假装联网DLC过滤寄存器（CAN_FLT_DLC）。 | S/U | 是 | 是 |
| 假装联网有效载荷低滤波1寄存器（CAN_PL1_LO）。 | S/U | 是 | 是 |
| 假装网络有效载荷高过滤1寄存器（CAN_PL1_HI） | S/U | 是 | 是 |
| 假装网络ID过滤2寄存器/ID屏蔽寄存器（CAN_FLT_ID2_IDMASK） | S/U | 是 | 是 |
| 假装联网有效载荷低位过滤2寄存器/有效载荷低位屏蔽寄存器（CAN_PL2_PLMASK_LO）。 | S/U | 是 | 是 |

| | | | |
|---|---|---|---|
| 假装联网有效载荷高过滤2寄存器/有效载荷高屏蔽寄存器（CAN_PL2_PLMASK_HI）。 | S/U | 是 | 是 |

*表继续在下一页...*

表50-5.寄存器访问和复位信息（续）

| 注册 | 访问类型 | 受硬重置的影响 | 受软复位的影响 |
|---|---|---|---|
| 假装联网唤醒信息缓冲区0寄存器（CAN_WMB0） | S/U | 是 | 没有 |
| 假装网络唤醒消息缓冲器1寄存器（CAN_WMB1） | S/U | 是 | 没有 |
| 假装网络唤醒消息缓冲器2寄存器（CAN_WMB2） | S/U | 是 | 没有 |
| 假装网络唤醒消息缓冲器3寄存器（CAN_WMB3） | S/U | 是 | 没有 |
| CAN FD控制寄存器（CAN_FDCTRL）。 | S/U | 是 | 没有 |
| CAN FD位定时寄存器（CAN_FDCBT） | S/U | 是 | 没有 |
| CAN FD CRC寄存器（CAN_FDCRC）。 | S/U | 是 | 是 |

FlexCAN模块可以使用邮箱和Rx FIFO结构存储CAN报文，用于传输和接收。

## 50.4.2 CAN寄存器描述

下表显示了FlexCAN的内存图。

从偏移量0x80到0x27F的地址范围分配了32个128位的消息缓冲区（MB）。

消息缓冲区的内存图在FlexCAN消息缓冲区内存图中。

### 50.4.2.1 CAN内存地图

FlexCAN0基地址。4002_4000h

FlexCAN1基础地址。4002_5000h

FlexCAN2基础地址。4002_B000h

| 偏移 | 注册 | 宽度（单位：比特） | 访问 | 重置值 |
|---|---|---|---|---|
| 0h | 模块配置寄存器（MCR） | 32 | RW | D890_000Fh |
| 4h | 控制1寄存器（CTRL1）。 | 32 | RW | 0000_0000h |
| 8h | 自由运行计时器（TIMER） | 32 | RW | 0000_0000h |
| 10h | Rx邮箱全局屏蔽寄存器（RXMGMASK）。 | 32 | RW | 见描述。 |

| | | | | |
|---|---|---|---|---|
| 14h | Rx 14屏蔽寄存器（RX14MASK） | 32 | RW | 见描述。 |

*表继续在下一页...*

| 偏移 | 注册 | 宽度<br>(单位：比特) | 访问 | 重置值 |
|---|---|---|---|---|
| 18h | Rx 15屏蔽寄存器（RX15MASK） | 32 | RW | 见描述。 |
| 1Ch | 错误计数器（ECR） | 32 | RW | 0000_0000h |
| 20h | 错误和状态1寄存器（ESR1）。 | 32 | W1C | 0000_0000h |
| 28h | 中断掩码1寄存器（IMASK1）。 | 32 | RW | 0000_0000h |
| 30h | 中断标志1寄存器（IFLAG1）。 | 32 | W1C | 0000_0000h |
| 34h | 控制2寄存器（CTRL2）。 | 32 | RW | 00A0_0000h |
| 38h | 错误和状态2寄存器（ESR2）。 | 32 | RO | 0000_0000h |
| 44h | CRC寄存器（CRCR） | 32 | RO | 0000_0000h |
| 48h | Rx FIFO全局屏蔽寄存器（RXFGMASK） | 32 | RW | 见描述。 |
| 4Ch | Rx FIFO信息寄存器（RXFIR）。 | 32 | RO | 见描述。 |
| 50h | CAN位定时寄存器（CBT） | 32 | RW | 0000_0000h |
| 880h - 8FCh | Rx单个屏蔽寄存器（RXIMR0 - RXIMR31） | 32 | RW | 见描述。 |
| B00h | 假装网络控制1寄存器（CTRL1_PN）。 | 32 | RW | 0000_0100h |
| B04h | 假装网络控制2寄存器（CTRL2_PN）。 | 32 | RW | 0000_0000h |
| B08h | 假装网络唤醒匹配寄存器（WU_MTC）。 | 32 | W1C | 0000_0000h |
| B0Ch | 假装的网络ID过滤器1寄存器（FLT_ID1） | 32 | RW | 0000_0000h |
| B10h | 假装联网DLC过滤寄存器（FLT_DLC）。 | 32 | RW | 0000_0008h |
| B14h | 假装的网络有效载荷低滤波1寄存器（PL1_LO）。 | 32 | RW | 0000_0000h |
| B18h | 假装网络有效载荷高过滤1寄存器(PL1_HI) | 32 | RW | 0000_0000h |
| B1Ch | 假装网络ID过滤2寄存器/ID屏蔽寄存器（FLT_ ID2_IDMASK） | 32 | RW | 0000_0000h |
| B20h | 假装联网有效载荷低位过滤2寄存器/有效载荷低位屏蔽寄存器（PL2_PLMASK_LO）。 | 32 | RW | 0000_0000h |
| B24h | 假装联网的有效载荷高过滤2低阶位/有效载荷高屏蔽寄存器（PL2_PLMASK_HI）。 | 32 | RW | 0000_0000h |
| B40h | C/S的唤醒信息缓冲寄存器（WMB0_CS）。 | 32 | RO | 0000_0000h |
| B44h | 唤醒信息缓冲寄存器的ID（WMB0_ID）。 | 32 | RO | 0000_0000h |
| B48h | 唤醒信息缓冲寄存器的数据0-3（WMB0_D03）。 | 32 | RO | 0000_0000h |
| B4Ch | 唤醒信息缓冲区寄存器数据4-7（WMB0_D47）。 | 32 | RO | 0000_0000h |
| B50h | C/S的唤醒信息缓冲寄存器（WMB1_CS）。 | 32 | RO | 0000_0000h |
| B54h | 唤醒信息缓冲器寄存器的ID（WMB1_ID）。 | 32 | RO | 0000_0000h |
| B58h | 唤醒信息缓冲寄存器的数据0-3（WMB1_D03）。 | 32 | RO | 0000_0000h |
| B5Ch | 唤醒信息缓冲区寄存器数据4-7（WMB1_D47）。 | 32 | RO | 0000_0000h |
| B60h | C/S的唤醒信息缓冲寄存器（WMB2_CS）。 | 32 | RO | 0000_0000h |
| B64h | 唤醒信息缓冲器寄存器的ID（WMB2_ID）。 | 32 | RO | 0000_0000h |

| | | | | |
|---|---|---|---|---|
| B68h | 唤醒信息缓冲寄存器的数据0-3（WMB2_D03）。 | 32 | RO | 0000_0000h |
| B6Ch | 唤醒信息缓冲区寄存器数据4-7（WMB2_D47）。 | 32 | RO | 0000_0000h |
| B70h | C/S的唤醒信息缓冲寄存器(WMB3_CS) | 32 | RO | 0000_0000h |

*表继续在下一页...*

| 偏移 | 注册 | 宽度<br>(单位：比特) | 访问 | 重置值 |
|---|---|---|---|---|
| B74h | 唤醒信息缓冲器寄存器的ID（WMB3_ID）。 | 32 | RO | 0000_0000h |
| B78h | 唤醒信息缓冲寄存器的数据0-3（WMB3_D03）。 | 32 | RO | 0000_0000h |
| B7Ch | 唤醒信息缓冲区寄存器数据4-7（WMB3_D47）。 | 32 | RO | 0000_0000h |
| C00h | CAN FD控制寄存器（FDCTRL）。 | 32 | RW | 8000_0100h |
| C04h | CAN FD Bit Timing Register (FDCBT) | 32 | RW | 0000_0000h |
| C08h | CAN FD CRC寄存器（FDCRC） | 32 | RO | 0000_0000h |

## 50.4.2.2 模块配置寄存器（MCR）

### 50.4.2.2.1 偏移

| 注册 | 偏移 |
|---|---|
| MCR | 0h |

### 50.4.2.2.2 职能

该寄存器定义了全局系统配置，如模块操作模式和最大信息缓冲器配置。

### 50.4.2.2.3 图示

| 咬牙切齿 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | MDIS | FRZ | RFEN | 停顿 | 诺德（NOTRDY | 保留 | SOFTRST | 冯-扎克(FRZACK) | SUPV | 保留 | 辽宁沈阳 | LPMACK | 保留 | 保留 | SRXDIS | IRMQ |
| W | | | | | | | | | | | | | | | | |
| 复位 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |

| 咬牙切齿 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | DMA | PNET_EN | LPRIOEN | AEN | FDEN | 0 | IDAM | | 0 | MAXMB | | | | | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |

## 50.4.2.2.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>MDIS | 模块禁用<br>该位控制FlexCAN是否被启用。当禁用时，FlexCAN禁用到CAN协议引擎和控制器主机接口子模块的时钟。该位不受软复位的影响。<br>0b - 启用FlexCAN模块。1b - 禁用FlexCAN模块。 |
| 30<br>FRZ | 冻结启用<br>FRZ位指定了当CAN_MCR寄存器中的HALT位被设置或在芯片级请求调试模式时FlexCAN的行为。当FRZ被断言时，FlexCAN被启用，进入Freeze模式。该位的负值会使FlexCAN从冻结模式中退出。<br>0b - 不启用进入冻结模式。1b - 启用进入冻结模式。 |
| 29<br>伦琴（RFEN） | Rx FIFO启用<br>该位控制Rx FIFO功能是否被启用。当RFEN被设置时，MBs 0到5不能用于正常的接收和传输，因为相应的内存区域（0x80-0xDC）被FIFO引擎以及额外的MBs（最多32个，取决于CAN_CTRL2[RFFN]的设置）使用，它们被用作Rx FIFO ID过滤表元素。RFEN还影响到每个CAN位的最小外设时钟数的定义，如表 "外设时钟频率和CAN位速率之间的最小比率 "中所述（见仲裁和匹配时序）。这个位只能在Freeze模式下写入，因为它在其他模式下被硬件阻断。<br>**注意：**当CAN FD操作被启用时，该位不能被设置（见FDEN位）。<br>0b - 未启用Rx FIFO。1b - 启用Rx FIFO。 |
| 28<br>停顿 | 停止FlexCAN<br>该位的断言使FlexCAN模块进入冻结模式。CPU应在初始化信息缓冲区和控制寄存器CAN_CTRL1和CAN_CTRL2后清除该位。在该位被清除之前，FlexCAN不进行任何接收或传输。当FlexCAN处于低功率模式时，不能进入冻结模式。<br>0b - 没有冻结模式请求。<br>1b - 如果FRZ位被宣称，则进入冻结模式。 |
| 27<br>诺德（NOTRDY） | FlexCAN未准备好<br>这个只读位表示FlexCAN处于禁用模式、停止模式或冻结模式。一旦FlexCAN退出这些模式，它就会被否定。该位不受软复位的影响。<br>0b - FlexCAN模块处于正常模式、只听模式或回环模式。1b - FlexCAN模块处于禁用模式、停止模式或冻结模式。 |
| 26<br>- | 保留<br>当写到这个字段时，总是写重置值。 |

| | |
|---|---|
| 25<br>ĀĀĀ | 软复位<br><br>当该位被断言时，FlexCAN重置其内部状态机和一些内存映射的寄存器。<br><br>当CPU写到MCR寄存器时，SOFTRST位可以直接被断言，但当全局软复位在芯片级被请求时，它也会被断言。因为软复位是同步的，必须遵循跨时钟域的请求/确认程序，所以可能需要一些时间来完全传播其效果。SOFTRST位在复位待定时保持正向，并在复位完成后自动被否定。因此，软件可以通过轮询这个位来了解软复位的完成时间。<br><br>当时钟在低功率模式下关闭时，不能应用软复位。应首先将模块从低功耗模式中移除，然后再应用软复位。该位不受软复位的影响。 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | 0b - 没有复位请求。<br>1b - 复位受软复位影响的寄存器。 |
| 24<br>冯-扎克<br>(FRZACK) | 冻结模式确认<br>这个只读位表示FlexCAN处于Freeze模式，其预分频器被停止。在当前传输或接收过程结束之前，冻结模式请求不能被批准。<br>因此，软件可以轮询FRZACK位，了解FlexCAN何时真正进入Freeze模式。如果冻结模式请求被否定，那么在FlexCAN预分频器再次运行后，该位将被否定。如果在FlexCAN处于低功耗模式时请求Freeze模式，那么只有在退出低功耗模式时，FRZACK位才会被设置。见 "冻结模式 "一节。该位不受软复位的影响。<br>**注意：**FRZACK将在CPU发出冻结模式请求后的178个CAN位内被断言，并在冻结模式请求消除后的2个CAN位内被否定（参见 "协议时序 "部分）。<br>0b - FlexCAN未处于冻结模式，预分频器运行。1b - FlexCAN处于冻结模式，预分频器停止。 |
| 23<br>SUPV | 监督员模式<br>该位将FlexCAN配置为监管者或用户模式。受该位影响的寄存器在模块内存地图的访问类型栏中被标记为S/U。该位的复位值为1，所以受影响的寄存器只能以监督者访问权限开始。这个位只能在冻结模式下写入，因为在其他模式下它被硬件屏蔽了。<br>0b - FlexCAN处于用户模式。受影响的寄存器允许监督员和非限制性访问。1b - FlexCAN处于监督者模式。受影响的寄存器只允许监督员访问。非限制性访问的行为就像访问一个未实现的寄存器位置一样。 |
| 22<br>- | 保留<br>当写到这个字段时，总是写重置值。 |
| 21<br>辽宁沈阳 | 警告中断启用<br>当断言时，该位启用错误和状态寄存器1（ESR1）中TWRNINT和RWRNINT标志的生成。如果WRNEN被否定，TWRNINT和RWRNINT标志将始终为零，与错误计数器的值无关，并且永远不会产生警告中断。这个位只能在冻结模式下写入，因为它在其他模式下被硬件封锁了。<br>0b - TWRNINT和RWRNINT位为零，与错误计数器的值无关。<br>1b - 当各自的错误计数器从小于96过渡到大于或等于96时，TWRNINT和RWRNINT位被设置。 |
| 20<br>LPMACK | 低功耗模式确认<br>这个只读位表示FlexCAN处于低功率模式（禁用模式、停止模式）。在当前所有的传输或接收过程结束之前，不能进入低功耗模式，所以CPU可以轮询LPMACK位，以了解FlexCAN何时真正进入低功耗模式。该位不受软复位的影响。<br>**注意：**LPMACK将在CPU发出低功耗模式请求后的180个CAN位内被断言，并在低功耗模式请求取消后的2个CAN位内被否定（见 "协议时间 "一节）。当FlexCAN处于假装联网模式时，LPMACK将在低功耗模式请求取消后的180个CAN位内被否定。<br>0b - FlexCAN不处于低功耗模式。1b - FlexCAN处于低功耗模式。 |
| 19<br>- | 保留<br>当写到这个字段时，总是写重置值。 |

| | |
|---|---|
| 18<br><br>- | 保留<br><br>当写到这个字段时，总是写重置值。 |
| 17 | 自行接收禁用 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| SRXDIS | 该位定义了FlexCAN是否允许接收自己传输的帧。如果该位被断言，模块传输的帧将不会被存储在任何MB中，无论该MB是否被编程为与传输的帧相匹配的ID，也不会因为帧接收而产生中断标志或中断信号。这个位只能在Freeze模式下写入，因为在其他模式下它被硬件阻断。<br>0b - 启用自我接收。 1b - 禁用自我接收。 |
| 16<br>IRMQ | 单独的Rx屏蔽和队列启用<br>该位指示Rx匹配过程是否基于单独的屏蔽和队列或基于CAN_RXMGMASK、CAN_RX14MASK、CAN_RX15MASK和CAN_RXFGMASK的屏蔽方案。这个位只能在Freeze模式下写入，因为它在其他模式下被硬件屏蔽了。<br>0b - 单个Rx屏蔽和队列功能被禁用。为了向后兼容传统的应用，即使MB是空的，读取C/S字也会锁定它。<br>1b - 启用个别Rx屏蔽和队列功能。 |
| 15<br>DMA | 启用DMA<br>DMA Enable位控制DMA功能是否被启用。DMA功能只能在Rx FIFO中使用，因此CAN_MCR[RFEN]位必须被引爆。当DMA和RFEN被设置时，CAN_IFLAG1[BUF5I]会产生DMA请求，而不会产生RX FIFO中断。这个位只能在Freeze模式下写入，因为它在其他模式下被硬件阻断。<br>0b - 禁用RX FIFO的DMA功能。 1b - 启用RX FIFO的DMA功能。 |
| 14<br>PNET_EN | 假装的网络启用<br>该位启用伪装网络功能。一旦进入停止模式，CAN_PE子块就会保持工作状态，能够处理由伪装网络配置寄存器定义的Rx消息过滤。参见伪装网络模式下的接收过程。这个位只能在Freeze模式下写入。<br>**注意：**这个字段并不是在每个实例中都支持。<br>支持的：弹性CAN0_MCR<br>不支持：FlexCAN1_MCR、FlexCAN2_MCR<br>0b - 假装的网络模式被禁用。 1b - 伪装网络模式被启用。 |
| 13<br>LPRIOEN | 启用本地优先权<br>提供这个位是为了向后兼容传统的应用程序。它控制本地优先级功能是否被启用。它被用来扩展仲裁过程中使用的ID。有了这个扩展的ID概念，仲裁过程是基于完整的32位字进行的，但实际传输的ID对于标准帧来说仍有11位，对于扩展帧来说有29位。这个位只能在Freeze模式下写入，因为它在其他模式下被硬件阻断。<br>0b - 禁用本地优先权。 1b - 启用本地优先权。 |
| 12<br>AEN | 终止启用<br>当断言时，该位启用Tx中止机制。这个机制保证了一个安全的程序来中止一个待定的传输，因此在没有通知的情况下，CAN总线上没有帧被发送。这个位只能在Freeze模式下写入，因为它在其他模式下被硬件阻断。 |

| | |
|---|---|
| | **注意：**当CAN_MCR[AEN]被断言时，只有中止机制（见传输中止机制）必须用于更新配置为传输的邮箱。 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | **注意：**当CAN_MCR[AEN]被断言时，将Abort代码写入Rx邮箱会导致不可预测的结果。<br>0b - 禁止中止。<br>1b - 启用中止功能<br>。 |
| 11<br>FDEN | 启用CAN FD操作<br>该位启用具有灵活数据速率（CAN FD）的CAN操作。该位只能在Freeze模式下写入。<br><br>**注意：**如果FDEN被断言，Rx FIFO Enable（RFEN）位不能被设置。<br>0b - CAN FD被禁用。FlexCAN能够接收和发送CAN 2.0格式的信息。1b - CAN FD已启用。<br>FlexCAN能够以CAN FD和CAN 2.0格式接收和传输信息。 |
| 10<br>- | 保留 |
| 9-8<br>IDAM | ID接受模式<br>这个2位字段确定了Rx FIFO ID过滤表元素的格式。注意，该表的所有元素都是由这个字段同时配置的（它们都是相同的格式）。见 "Rx FIFO结构 "一节。这个字段只能在Freeze模式下写入，因为它在其他模式下被硬件阻断。<br>00b - 格式A：每个ID过滤表元素有一个完整的ID（标准和扩展）。<br>01b - 格式B：每个ID过滤表元素有两个完整的标准ID或两个部分14位（标准和扩展）ID。<br>10b - 格式C：每个ID过滤表元素有四个部分8位标准ID。11b - 格式D。拒绝<br>所有帧。 |
| 7<br>- | 保留 |
| 6-0<br>MAXMB | 最后一个信息缓冲区的编号<br>这个7位字段定义了将参与匹配和仲裁过程的最后一个消息缓冲区的数量。复位值（0x0F）等同于16MB的配置。这个字段只能在Freeze模式下写入，因为它在其他模式下被硬件阻断。<br>最后一个MB的数量=MAXMB<br>**注意：**MAXMB的编程值必须小于或等于可用的消息缓冲区的数量，这在用于CAN FD的FlexCAN内存分区中有所描述。<br>此外，MAXMB值的定义必须考虑到Rx FIFO占用的MB区域以及由CAN_CTRL2寄存器中RFFN位定义的其ID过滤表空间。MAXMB还影响到表 "外围时钟频率和CAN比特率之间的最小比率 "中描述的每个CAN比特的最小外围时钟数的定义（见仲裁和匹配定时）。 |

## 50.4.2.3 控制1寄存器（CTRL1）。

### 50.4.2.3.1 偏移

| 注册 | 偏移 |
| --- | --- |

| | |
|---|---|
| CTRL1 | 4h |

#### 50.4.2.3.2 职能

这个寄存器是为与CAN总线有关的特定FlexCAN控制功能而定义的，如比特率、Rx位内的可编程采样点、回环模式、只听模式、总线关闭恢复行为和中断启用（总线关闭、错误、警告）。它还决定了时钟预分频器的划分系数。

CAN位定时变量（PRESDIV、PROPSEG、PSEG1、PSEG2和RJW）也可以在CAN_CBT寄存器中进行配置，它扩展了所有这些变量的范围。如果CAN_CBT[BTF]被设置，PRESDIV、PROPSEG、PSEG1、PSEG2和RJW领域的CAN_CTRL1成为只读。

**注意事项**

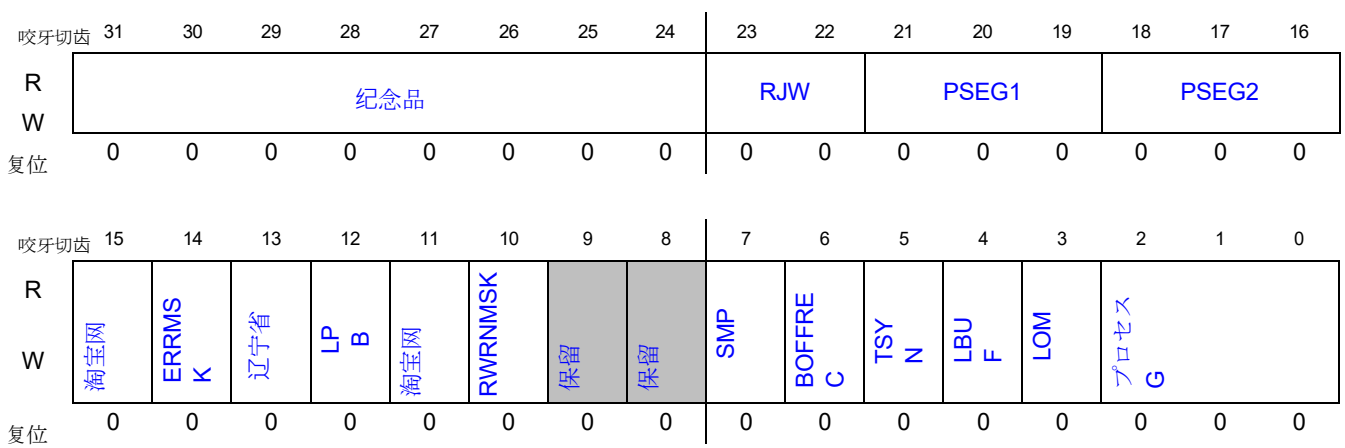
当CAN FD功能被启用时，不要使用CAN_CTRL1寄存器的PRESDIV、RJW、PSEG1、PSEG2和PROPSEG字段来进行CAN位计时。而是使用CAN_CBT寄存器的EPRESDIV、ERJW、EPSEG1、EPSEG2。和EPROPSEG字段。

该寄存器的内容不受软复位的影响。

**注意事项**

CAN_CTRL1和CAN_CBT中的CAN位变量被存储在同一个寄存器中。

#### 50.4.2.3.3 图示



## 50.4.2.3.4 领域

| 场地 | 职能 |
| --- | --- |
| 31-24<br>纪念品 | 预分频器分频系数<br>这个8位字段定义了PE时钟频率和串行时钟（Sclock）频率之间的比率。Sclock周期定义了CAN协议的时间量子。对于复位值，Sclock频率等于PE时钟频率。这个字段的最大值是0xFF，它给出了一个最小的Sclock频率，等于PE时钟频率除以256。参见 "协议计时 "部分。这个字段只能在Freeze模式下写入，因为它在其他模式下被硬件阻断。<br>Sclock频率=PE时钟频率/(PRESDIV + 1) |
| 23-22<br>RJW | 重新同步跳跃宽度<br>这个2位字段定义了一个比特时间通过一次重新同步可以改变的最大时间量子数。一个时间量子等于Sclock周期。可编程的有效值是0-<br>3.这个字段只能在冻结模式下写入，因为它在其他模式下被硬件封锁了。<br>同步跳转宽度 = RJW + 1。 |
| 21-19<br>PSEG1 | 阶段段1<br>这个3位字段定义了位时间中第1相段的长度。可编程的有效值是0-7。这个字段只能在冻结模式下写入，因为它在其他模式下被硬件阻断。<br>相位缓冲段1=（PSEG1+1）×时间-量子。 |
| 18-16<br>PSEG2 | 阶段性片段2<br>这个3位字段定义了位时间中的第2阶段的长度。可编程的有效值是1-7。这个字段只能在冻结模式下写入，因为它在其他模式下被硬件阻断。<br>相位缓冲段2=（PSEG2+1）×时间-量子。 |
| 15<br>淘宝网 | 总线关闭中断屏蔽<br>该位为CAN_ESR1寄存器中的总线关闭中断BOFFINT提供一个屏蔽。<br>0b - 禁用总线关闭中断。1b - 启<br>用总线关闭中断。 |
| 14<br>ÃÃÃ | 错误中断屏蔽<br>该位为CAN_ESR1寄存器中的错误中断ERRINT提供一个屏蔽。<br>0b - 错误中断禁用。1b - 启<br>用错误中断。 |
| 13<br>辽宁省 | CAN引擎时钟源<br>该位选择CAN协议引擎（PE）的时钟源为外围时钟或振荡器时钟。所选择的时钟是送入预分频器以产生串行时钟（Sclock）的时钟。为了保证可靠的操作，该位只能在禁用模式下写入，因为它在其他模式下被硬件阻断。参见 "协议计时"。<br>0b - CAN引擎的时钟源是振荡器时钟。在这种情况下，振荡器的时钟频率必须低于总线时钟。<br>1b - CAN引擎的时钟源是外围时钟。 |

| | |
|---|---|
| 12<br>LPB | 回路模式<br>该位将FlexCAN配置为环回模式运行。在这种模式下，FlexCAN执行内部回环，可用于自检操作。发送器的比特流输出在内部反馈到接收器输入。Rx CAN输入引脚被忽略，Tx CAN输出转为隐性状态（逻辑1）。FlexCAN在传输时的行为与正常情况下一样，并将自己传输的信息视为从远程节点接收的信息。 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | 在这种模式下，FlexCAN忽略了CAN帧确认字段中ACK槽期间发送的位，生成一个内部确认位以确保正确接收自己的信息。发送和接收中断都会产生。该位只能在Freeze模式下写入，因为它在其他模式下被硬件阻断。<br>**注意：**在这种模式下，CAN_MCR[SRXDIS]不能被断言，因为这将阻碍传输信息的自我接收。<br>**注意：**当LPB被断言时，CAN_FDCTRL寄存器中的TDCEN位必须被禁用。<br>0b - 禁止回环。<br>1b - 启用回圈。 |
| 11<br>淘宝网 | Tx警告中断屏蔽<br>该位为与错误和状态寄存器1（ESR1）中的TWRNINT标志相关的Tx警告中断提供屏蔽。当CAN_MCR[WRNEN]位被否定时，该位被读为零。只有当CAN_MCR[WRNEN]位被断言时，这个位才能被写入。<br>0b - 禁用Tx警告中断。1b - 启用Tx警告中断。 |
| 10<br>RWRNMSK | Rx警告中断屏蔽<br>该位为与错误和状态寄存器1（ESR1）中的RWRNINT标志相关的Rx警告中断提供一个屏蔽。当CAN_MCR[WRNEN]位被否定时，该位被读为零。只有当CAN_MCR[WRNEN]位被断言时，这个位才能被写入。<br>0b - 禁用Rx警告中断。1b - 启用Rx警告中断。 |
| 9<br>- | 保留 |
| 8<br>- | 保留 |
| 7<br>SMP | CAN位取样<br>该位定义了Rx输入的CAN位的采样模式。它只能在Freeze模式下被写入，因为在其他模式下它被硬件屏蔽了。<br>**注意：**为了正确的操作，要断言SMP，必须保证CAN_CTRL1[PSEG1]（或CAN_CBT[EPSEG1]）中有2个TQ的最小值。当CAN FD被启用（CAN_MCR[FDEN] = 1）时，该位不能被断言。<br>0b - 只用一个样本来确定比特值。<br>1b - 使用三个样本来确定接收位的值：常规的一个（样本点）和前面的两个样本；使用多数规则。 |
| 6<br>淘宝网 | 公共汽车关闭恢复<br>该位定义了FlexCAN如何从总线关闭状态恢复。如果该位被否定，则根据CAN规范2.0B，自动从总线关闭状态恢复。如果该位被断言，自动恢复总线关闭状态被禁用，模块保持在总线关闭状态，直到该位被用户否定。如果在CAN总线上检测到11个隐性位的128个序列之前发生否定，那么总线关闭的恢复就像BOFFREC位从未被断言过一样。如果在发生了128个11位隐性位的序列后发生否定，那么FlexCAN将在加入总线前等待11位隐性位来重新同步到总线上。否定后，BOFFREC位可以在总线关闭期间再次被重新置位，但它只有在模块下次进入总线关闭时才有效。如果BOFFREC在模块进入总线关闭时被否定，那么在总线关闭时断言它对当前的总线关闭恢复没有效果。<br>0b - 启用从总线关闭状态自动恢复的功能。1b - 禁止从总线关闭状态自动恢复。 |

| | |
|---|---|
| 5<br>TSYN | 计时器同步<br><br>该位启用了一种机制，每次在消息缓冲区0中收到消息时，都会重置自由运行的计时器。 该功能提供了使多个FlexCAN站同步的手段，并有一个特殊的 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | "SYNC "消息，即全球网络时间。如果CAN_MCR中的RFEN位被设置（Rx FIFO启用），根据CAN_CTRL2[RFFN]的设置，第一个可用的邮箱被用于定时器同步，而不是MB0。 这个位只能在Freeze模式下被写入，因为在其他模式下它被硬件阻断。<br>0b - 禁用定时器同步功能 1b - 启<br>用定时器同步功能 |
| 4<br>LBUF | 首先传输最低的缓冲区<br>这个位定义了消息缓冲区传输的排序机制。当断言时，CAN_MCR[LPRIOEN]位不影响优先级仲裁。这个位只能在Freeze模式下写入，因为它在其他模式下被硬件阻断。<br>0b - 优先级最高的缓冲区先被传送。 1b - 最低编号<br>的缓冲区先被传送。 |
| 3<br>LOM | 只听模式<br>该位将FlexCAN配置为只听模式。在这种模式下，传输被禁用，CAN_ECR寄存器中描述的所有错误计数器被冻结，模块以CAN错误被动模式运行。只有被另一个CAN站确认的报文才会被接收。如果FlexCAN检测到一个没有被确认的报文，它将标记一个BIT0错误，而不改变CAN_ECR寄存器中的接收错误计数器（RXERRCNT），就像它正在尝试确认该报文。<br>只听模式被CAN_ESR1[FLTCONF]字段的状态所确认，显示为被动错误。只听模式的请求和确认之间可能会有一些延迟。<br>这个位只能在冻结模式下写入，因为它在其他模式下被硬件屏蔽了。<br>0b - 停用只听模式。<br>1b - FlexCAN模块在只听模式下运行。 |
| 2-0<br>ÃÃÃ | 传播段<br>这个3位字段定义了传播段在比特时间内的长度。可编程的有效值是0-7。这个字段只能在Freeze模式下写入，因为它在其他模式下被硬件阻断。<br>传播段时间=（PROPSEG+1）×时间-量子。时间-量子=一个<br>Sclock周期。 |

## 50.4.2.4 自由运行计时器（TIMER）

### 50.4.2.4.1 偏移

| 注册 | 偏移 |
| --- | --- |
| 计时器 | 8h |

#### 50.4.2.4.2 职能

这个寄存器代表一个16位的自由运行的计数器，可以被CPU读和写。定时器在复位后从0x0开始，线性地计数到0xFFFF，然后绕过。

当CAN_CTRL2寄存器中的TIMER_SRC被断言时，定时器被一个外部的时间刻度持续增加。时间刻度必须与外围时钟同步，最小脉冲宽度为一个时钟周期。

当CAN_CTRL2寄存器中的TIMER_SRC位被否定时，定时器被CAN位时钟递增，它定义了CAN总线的波特率。在信息传输/接收过程中，每接收或传输一个比特，它就递增一个。当总线上没有消息时，它使用先前编程的波特率进行计数。在禁用、停止、假装联网和冻结模式下，定时器不增量。

当任何帧的标识符字段的第二位在CAN总线上时，定时器的值被捕获。在成功接收或发送一个报文后，这个捕获的值被写入报文缓冲区的时间戳条目中。

如果位CAN_CTRL1[TSYN]被断言，根据CAN_CTRL2[RFFN]的设置，每当在第一个可用的邮箱中收到一个信息时，定时器就会被重置。

CPU可以随时写到这个寄存器。但是，如果写入发生在定时器被第一个邮箱中的接收所重置的同时，那么写入的值将被丢弃。

读取这个寄存器会影响到邮箱解锁程序，见 "邮箱锁定机制 "部分。

### 50.4.2.4.3 图示

| 咬牙切齿 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 咬牙切齿 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 计时器 | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

### 50.4.2.4.4 领域

| 场地 | 职能 |
|---|---|
| 31-16<br>- | 保留 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| 15-0<br>计时器 | 计时器值<br>包含自由运行的计数器值。 |

## 50.4.2.5 Rx邮箱全局屏蔽寄存器（RXMGMASK）。

### 50.4.2.5.1 偏移

| 注册 | 偏移 |
|---|---|
| 荣格玛斯克(RXMGMASK) | 10h |

### 50.4.2.5.2 职能

该寄存器位于RAM中。

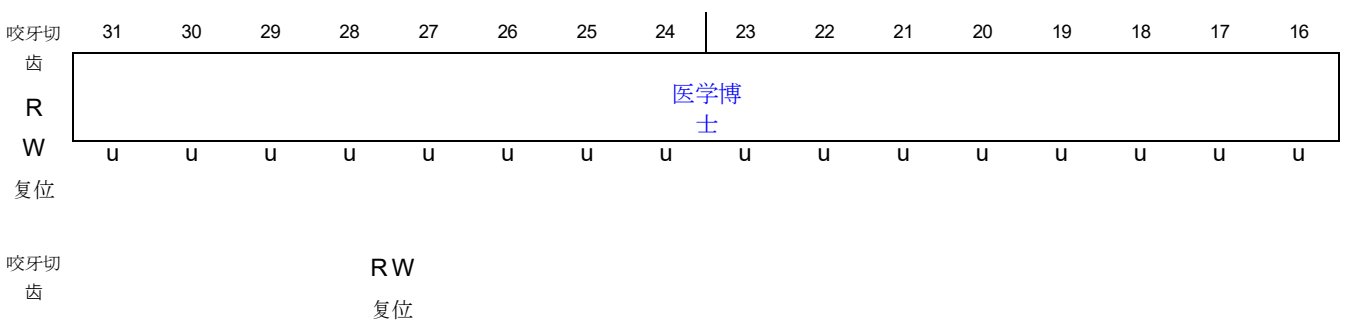
提供RXMGMASK是为了支持传统的应用。

- 当CAN_MCR[IRMQ]位被否定时，RXMGMASK总是有效的（MG字段中的位将屏蔽邮箱过滤位）。
- 当CAN_MCR[IRMQ]位被断言时，RXMGMASK没有作用（MG字段中的位不会屏蔽邮箱过滤位）。

RXMGMASK用于屏蔽所有Rx MBs的过滤字段，不包括MBs 14-15，后者有单独的屏蔽寄存器。

这个寄存器只能在冻结模式下写入，因为它在其他模式下被硬件阻断。

### 50.4.2.5.3 图示






u u u u u u u u

u u u u u u u u

#### 50.4.2.5.4 领域

<table>
<tr><th>场地</th><th>职能</th></tr>
<tr><td>31-0<br>医学博士</td><td>Rx邮箱的全局掩码位<br>这些位掩盖了邮箱的过滤位。请注意，与邮箱ID字的对齐并不完美，因为两个最重要的MG位会影响RTR和IDE字段，它们位于邮箱的控制和状态字中。下表详细说明了哪些MG位屏蔽了每个邮箱过滤字段。
<table>
<tr><th rowspan="2">SMB[RTR]</th><th rowspan="2">can_ctrl2[ rrs]</th><th rowspan="2">can_ctrl2[ eacen] 。</th><th colspan="4">信箱过滤字段</th></tr>
<tr><th>MB[RTR]</th><th>MB[IDE]</th><th>MB[ID]</th><th>保留</th></tr>
<tr><td>0</td><td>-</td><td>0</td><td>备注</td><td>备注</td><td>MG[28:0]</td><td>MG[31:29]</td></tr>
<tr><td>0</td><td>-</td><td>1</td><td>MG[31]</td><td>MG[30]</td><td>MG[28:0]</td><td>MG[29]</td></tr>
<tr><td>1</td><td>0</td><td>-</td><td>-</td><td>-</td><td>-</td><td>MG[31:0]</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>-</td><td>-</td><td>MG[28:0]</td><td>MG[31:29]</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>MG[31]</td><td>MG[30]</td><td>MG[28:0]</td><td>MG[29]</td></tr>
</table>
1. 来电帧的RTR位。它被保存在一个称为Rx串行信息缓冲区（Rx SMB）的辅助MB中。<br>2. 如果CTRL2[EACEN]位被否定，邮箱的RTR位永远不会与传入帧的RTR位比较。<br>3. 如果CAN_CTRL2[EACEN]位被否定，邮箱的IDE位总是与传入帧的IDE位进行比较。<br>0b - 过滤器中的相应位是 "不关心"。1b - 过滤器中的<br>相应位被检查。</td></tr>
</table>

1. 来电帧的RTR位。它被保存在一个称为Rx串行信息缓冲区（Rx SMB）的辅助MB中。
2. 如果CTRL2[EACEN]位被否定，邮箱的RTR位永远不会与传入帧的RTR位比较。
3. 如果CAN_CTRL2[EACEN]位被否定，邮箱的IDE位总是与传入帧的IDE位进行比较。

### 50.4.2.6 Rx 14屏蔽寄存器（RX14MASK）

#### 50.4.2.6.1 偏移

| 注册 | 偏移 |
|---|---|
| 荣格14MASK | 14h |

#### 50.4.2.6.2 职能

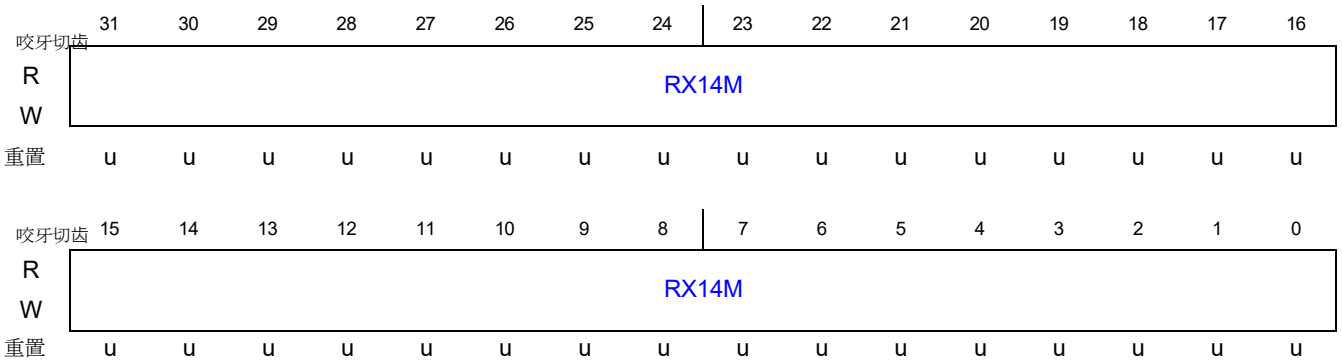
该寄存器位于RAM中。

RX14MASK是为支持传统的应用而提供的。当CAN_MCR[IRMQ]位被断言时，RX14MASK没有作用。

RX14MASK用于屏蔽消息缓冲区14的过滤字段。

这个寄存器只能在模块处于冻结模式时进行编程，因为它在其他模式下被硬件阻断。

#### 50.4.2.6.3 图示




#### 50.4.2.6.4 领域

| 场地 | 职能 |
|---|---|
| 31-0<br>RX14M | Rx缓冲区14个掩码位<br>每个屏蔽位都会屏蔽相应的邮箱14的过滤字段，与RXMGMASK屏蔽其他邮箱的过滤器的方式相同。参见CAN_RXMGMASK寄存器的描述。<br>0b - 过滤器中的相应位是 "不关心"。 1b - 过滤器中的<br>相应位被检查。 |

### 50.4.2.7 Rx 15屏蔽寄存器（RX15MASK）

#### 50.4.2.7.1 偏移

| 注册 | 偏移 |
|---|---|
| 荣格15MASK | 18h |

### 50.4.2.7.2 职能

该寄存器位于RAM中。

RX15MASK是为支持传统的应用而提供的。当CAN_MCR[IRMQ]位被断言时，RX15MASK没有作用。

RX15MASK用于屏蔽消息缓冲区15的过滤字段。

这个寄存器只有在模块处于冻结模式时才能被编程，因为它在其他模式下被硬件阻断。

### 50.4.2.7.3 图示

| 咬牙切齿 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | RX15M | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | u | u | u | u | u | u | u | u | u | u | u | u | u | u | u | u |

| 咬牙切齿 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | RX15M | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | u | u | u | u | u | u | u | u | u | u | u | u | u | u | u | u |

### 50.4.2.7.4 领域

| 场地 | 职能 |
|---|---|
| 31-0<br>RX15M | Rx缓冲区15个掩码位<br>每个屏蔽位都会屏蔽相应的邮箱15的过滤字段，与RXMGMASK屏蔽其他邮箱的过滤器的方式相同。参见CAN_RXMGMASK寄存器的描述。<br>0b - 过滤器中的相应位是 "不关心"。1b - 过滤器中的相应位被检查。 |

## 50.4.2.8 错误计数器（ECR）

## 50.4.2.8.1 偏移

| 注册 | 偏移 |
| --- | --- |
| ECR | 1Ch |

#### 50.4.2.8.2 职能

该寄存器有四个8位字段，反映FlexCAN错误计数器的值。

- 发送错误计数器（TXERRCNT字段）
- 接收错误计数器（RXERRCNT字段）。
- 在设置了BRS位的CAN FD报文的数据阶段检测到的错误的传输错误计数器（TXERRCNT_FAST域）。
- 在设置了BRS位的CAN FD报文的数据阶段检测到的错误的接收错误计数器（RXERRCNT_FAST域）。

TXERRCNT和RXERRCNT计数器考虑了CAN FD和非FD报文格式中的所有错误。TXERRCNT_FAST和RXERRCNT_FAST专门用来计算在设置了BRS位的CAN FD帧的数据阶段发生的错误。

故障封闭状态（Error and Status Register 1 - CAN_ESR1中的FLTCONF字段）只根据TXERRCNT和RXERRCNT计数器更新。TXERRCNT和RXERRCNT计数器只能在Freeze模式下被写入。TXERRCNT_FAST和RXERRCNT_FAST计数器是只读的，除非在Freeze模式下，CPU可以写值为零。增加和减少这些计数器的规则在CAN协议中有所描述，在FlexCAN模块中完全实现。

以下是FlexCAN总线状态转换的基本规则。

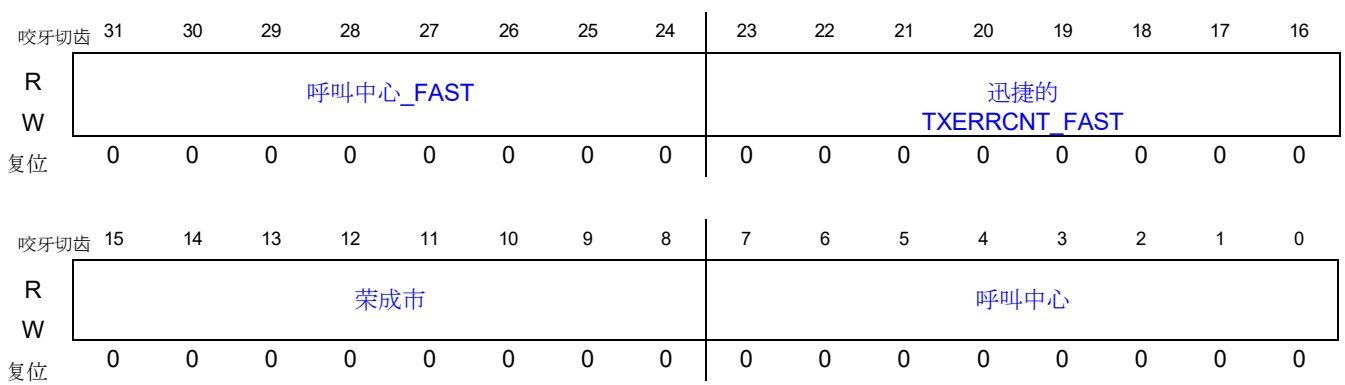
- 如果TXERRCNT或RXERRCNT的值增加到大于或等于128，错误和状态寄存器中的FLTCONF字段将被更新以反映 "错误被动 "状态。
- 如果FlexCAN的状态是 "错误被动"，并且TXERRCNT或RXERRCNT递减到一个小于或等于127的值，而另一个已经满足了这个条件，那么错误和状态寄存器中的FLTCONF字段将被更新，以反映 "错误主动 "状态。
- 如果TXERRCNT的值增加到大于255，错误和状态寄存器中的FLTCONF字段将被更新以反映 "总线关闭 "状态，并可能发出一个中断。然后TXERRCNT的值被重置为零。
- 如果FlexCAN处于 "总线关闭 "状态，那么TXERRCNT将与另一个内部计数器级联，以计算总线上连续11个隐性位的第128次出现。因此，TXERRCNT被重置为零，并以内部计数器计数

11个这样的位的方式进行计数，然后在增加TXERRCNT的同时进行绕行。当TXERRCNT达到128的值时，错误和状态寄存器中的FLTCONF字段被更新为"错误激活"，两个错误计数器被重置为零。在任何情况下，主导位跟随一个小于11的数据流

连续的隐性位，内部计数器自行复位为零，不影响TXERRCNT值。在总线关闭期间，TXERRCNT_FAST计数器被冻结。

- 如果在系统启动期间，只有一个节点在工作，那么它的TXERRCNT在它试图传输的每条信息中都会增加，这是确认错误的结果（由错误和状态寄存器中的ACKERR位指示）。在过渡到 "错误被动
  "状态后，TXERRCNT不会再因确认错误而增加。因此，设备永远不会进入 "总线关闭 "状态。
- 如果RXERRCNT增加到一个大于127的值，它就不再继续增加，即使在作为接收者时检测到更多的错误。在下一次成功接收信息时，计数器被设置为119和127之间的值，恢复到 "错误激活 "状态。
- TXERRCNT_FAST和RXERRCNT_FAST错误计数器的值仅基于在设置了BRS位的CAN
  FD帧的数据阶段检测到的错误而递增和递减，遵循与TXERRCNT和RXERRCNT计数器同样的递增和递减规则。这些计数器不会缠绕并卡在它们的最大值（255）。当FlexCAN处于 "总线关闭
  "状态时，它们会停止计数并保持其数值不变。当FlexCAN离开 "总线关闭
  "状态时，它们会被重置，一旦FlexCAN恢复到 "错误激活
  "状态，它们会重新开始计数。
- 当FlexCAN处于伪装网络模式时，RXERRCNT和RXERRCNT_FAST一直在计算错误，错误标志也被存储。TXERRCNT和TXERRCNT_FAST保留其值，不发生变化，因为在伪装网络模式下不发生传输。当FlexCAN恢复到Normal模式时，在Pretended
  Networking模式下改变值的错误计数器和错误标志会在CAN_ECR和CAN_ESR1中更新。如果FlexCAN处于伪装网络模式，CAN_ESR1寄存器中的FAST错误标志将不会被设置。

### 50.4.2.8.3 图示



#### 50.4.2.8.4 领域

| 场地 | 职能 |
|---|---|
| 31-24<br>荣格_FA ST | 快速位的接收错误计数器<br>接收错误计数器，用于检测在设置了BRS位的CAN FD报文的数据阶段中检测到的错误。<br>RXERRCNT_FAST计数器是只读的，除非在Freeze模式下，CPU可以写一个<br>只有8位零值。 |
| 23-16<br>TXERRCNT_FAST | 用于快速比特的传输错误计数器<br>传输错误计数器，用于在设置了BRS位的传输的CAN FD报文的数据阶段检测到的错误。<br>TXERRCNT_FAST计数器是只读的，除非在Freeze模式下，CPU只能写入一个8位的零值。 |
| 15-8<br>荣成市 | 接收错误计数器<br>接收错误计数器，用于接收信息中检测到的所有错误。RXERRCNT计数器是只读的，除非在冻结模式下，它可以由CPU写入。 |
| 7-0<br>呼叫中心 | 传输错误计数器<br>传输错误计数器，用于检测传输信息中的所有错误。TXERRCNT计数器是只读的，除非在冻结模式下，它可以由CPU写入。 |

## 50.4.2.9 错误和状态1寄存器（ESR1）。

### 50.4.2.9.1 偏移

| 注册 | 偏移 |
|---|---|
| ESR1 | 20h |

### 50.4.2.9.2 职能

该寄存器报告在接收和传输CAN帧时检测到的各种错误情况，以及设备的一些一般状态，它也是一些中断到CPU的来源。

报告的错误条件是BIT1ERR、BIT0ERR、ACKKERR、CRCERR、FRMERR和STFERR，用于在任何格式的CAN帧中检测到的错误，以及BIT1ERR_FAST、BIT0ERR_FAST、CRCERR_FAST、FRMERR_FAST和STFERR。
STFERR_FAST用于在仅设置了BRS位的CAN FD帧的数据阶段检测到的错误。

在一个CAN帧中检测到的错误可以由一个或多个错误标志报告。另外，错误报告

是累积性的，如果在接下来的帧中发生更多的错误事件，而CPU并不试图读取这个寄存器。

TXWRN, RXWRN, IDLE, TX, FLTCONF, RX 和 SYNCH 是状态位。

BOFFINT, BOFFDONEINT, ERRINT, ERRINT_FAST, TWRNINT和RWRNINT为中断位。建议CPU在处理由这些位产生的中断请求时，使用以下程序。

- 读取该寄存器以捕获所有错误条件和状态位。这个动作会清除自上次读取访问后被设置的相应位。
- 写1来清除触发中断请求的中断位。
- 如果ERR_OVR位被设置，写1来清除它。

从所有错误标志清零开始，第一个错误事件设置ERRINT或ERRINT_FAST（只要相应的屏蔽位被断言）。如果后续帧中的其他错误事件发生在CPU满足中断请求之前，ERR_OVR位被设置，以表明来自不同帧的错误已经累积。

| シンチ | 怠速 | 德克萨斯州 | 荣格 | FlexCAN状态 |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 与CAN总线不同步 |
| 1 | 1 | x | x | 闲置 |
| 1 | 0 | 1 | 0 | 传送 |
| 1 | 0 | 0 | 1 | 接收 |

### 50.4.2.9.3 图示

| 咬牙切齿 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | BIT1ERR_FAST | BIT0ERR_FAST | 0 | 呼叫中心：CRCERR_FAST | 基金会 | 呼叫中心 | 0 | | | | ERROVR | 误码率_FAST | 淘宝网 | シンチ | 淘宝网 | RWRNINT |
| W | | | | | | | | | | | W1C | W1C | W1C | | W1C | W1C |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 咬牙切齿 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | BIT1ERR | BIT0ERR | ÃÃÃ | CRCERR | 冯-马R | 汕头市 | 辽宁省 | 荣华富贵 | IDLE | 德克 | FLTCONF | | RX | 勃苏特(BOFFINT | 鄂林T | 0 |
| W | | | | | | | | | | | | | | W1C | W1C | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

### 50.4.2.9.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>BIT1ERR_FAST | Bit1 在设置了BRS位的CAN FD帧的数据阶段的错误<br>该位指示在设置了BRS位的CAN FD帧的数据阶段中，传输和接收的位之间发生不一致的情况。<br>0b - 没有这种情况发生。<br>1b - 至少有一个作为隐性发送的位被作为显性接收。 |
| 30<br>BIT0ERR_FAST | 在设置了BRS位的CAN FD帧的数据阶段出现Bit0错误<br>该位指示在设置了BRS位的CAN FD帧的数据阶段中，传输和接收的位之间发生不一致的情况。<br>0b - 没有这种情况发生。<br>1b - 至少有一个作为显性发送的比特被作为隐性接收。 |
| 29<br>- | 保留 |
| 28<br>呼叫中心：<br>CRCERR_FAST | 设置了BRS位的CAN FD帧的CRC字段的循环冗余检查错误<br>该位表示接收节点在设置了BRS位的CAN FD帧的CRC字段中检测到了CRC错误，也就是说，计算出的CRC与接收到的不同。<br>0b - 没有这种情况发生。<br>1b - 自上次读取该寄存器后，发生了一个CRC错误。 |
| 27<br>迅速 | 设置了BRS位的CAN FD帧在数据阶段的形式错误<br>该位表示接收节点在设置了BRS位的CAN FD帧的数据阶段检测到了形式错误，即固定形式的位域至少包含一个非法位。<br>0b - 没有这种情况发生。<br>1b - 自上次读取该寄存器后，发生了一个表格错误。 |
| 26<br>呼叫中心 | 在设置了BRS位的CAN FD帧的数据阶段出现填塞错误<br>该位表示在设置了BRS位的CAN FD帧的数据阶段检测到了一个塞满错误。<br>0b - 没有这种情况发生。<br>1b - 自上次读取该寄存器后，发生了一个塞满错误。 |
| 25-22<br>- | 保留 |
| 21<br>ĀĀĀ | 错误超限位<br>当任何错误标志已经被设置时，该位表示发生了一个错误状况。该位通过写为1来清除。<br>0b - 没有发生超限。 1b - 已发生<br>超限。 |

*表继续在下一页...*

| | |
|---|---|
| 20<br>误码率_快速 | 错误中断，用于在设置了BRS位的CAN FD帧的数据阶段检测到的错误。<br>该位表示在设置了BRS位的CAN FD帧的数据阶段检测到的至少一个错误位（BIT1ERR_FAST, BIT0ERR_FAST, CRCERR_FAST, FRMERR_FAST或STFERR_FAST）。<br>设置。如果相应的屏蔽位CAN_CTRL2[ERRMSK_FAST]被设置，就会产生一个中断给CPU。这个位可以通过写为1而被清除。写入0没有效果。<br>0b - 没有这种情况发生。<br>1b - 表示在设置了BRS位的CAN FD帧的数据阶段检测到的任何错误位的设置。 |
| 19<br>淘宝网 | 总线关闭完成中断<br>当Tx错误计数器（TXERRCNT）完成了对CAN总线上连续11个隐性位的128次发生的计数并准备离开总线关闭时，该位被设置。如果相应的掩码位在 |

| 场地 | 职能 |
| --- | --- |
| | 控制2寄存器(BOFFDONEMSK)被设置，就会产生一个中断给CPU。这个位可以通过写1来清除。写入0没有效果。<br>0b - 没有这种情况发生。<br>1b - FlexCAN模块已完成总线关闭过程。 |
| 18<br>シンチ | CAN同步状态<br>这个只读标志表示FlexCAN是否与CAN总线同步，是否能够参与通信过程。它是由FlexCAN设置和清除的。请参见整个CAN_ESR1寄存器描述中的表格。<br>0b - FlexCAN不与CAN总线同步。1b - FlexCAN已与CAN总线同步。 |
| 17<br>淘宝网 | Tx警告中断标志<br>如果CAN_MCR中的WRNEN位被断言，当TXWRN标志从0过渡到1时，TWRNINT位被设置，意味着Tx错误计数器达到96。如果控制1寄存器（CAN_CTRL1[TWRNMSK]）中相应的屏蔽位被设置，则会产生一个中断给CPU。这个位可以通过写为1而被清除。当WRNEN被否定时，这个标志被屏蔽。CPU必须在禁用该位之前清除该标志。否则，当WRNEN再次被设置时，它将被设置。写入0没有效果。在总线关闭状态下，该标志不产生。在冻结模式下，该位不被更新。<br>当FlexCAN从伪装网络模式返回到正常模式时（见伪装网络模式下的接收过程），该位不会被更新。<br>0b - 没有这种情况发生。<br>1b - Tx错误计数器从小于96过渡到大于或等于96。 |
| 16<br>RWRNINT | Rx警告中断标志<br>如果CAN_MCR中的WRNEN位被断言，当RXWRN标志从0过渡到1时，RWRNINT位被设置，意味着Rx错误计数器达到96。如果控制1寄存器（CAN_CTRL1[RWRNMSK]）中相应的屏蔽位被设置，就会产生一个中断给CPU。这个位可以通过写为1而被清除。当WRNEN被否定时，这个标志被屏蔽。CPU必须在禁用该位之前清除该标志。否则，当WRNEN再次被设置时，它将被设置。写入0没有效果。在冻结模式下，这个位不会被更新。<br>当FlexCAN从伪装网络模式返回正常模式时（见伪装网络模式下的接收过程），该位被更新以反映Rx错误计数器的状态。<br>0b - 没有这种情况发生。<br>1b - Rx错误计数器从小于96过渡到大于或等于96。 |
| 15<br>BIT1ERR | Bit1 错误<br>该位指示在非CAN FD报文中或在CAN FD报文的仲裁或数据阶段中，传输和接收的位之间发生不一致的情况。<br>当FlexCAN从伪装网络模式返回到正常模式时，该位会被更新。<br>**注意：**在仲裁场或ACK槽的情况下，或在节点发送检测主导位的被动错误标志的情况下，发射器不设置该位。<br>0b - 没有这种情况发生。<br>1b - 至少有一个作为隐性发送的位被作为显性接收。 |

*表继续在下一页...*

| | |
|---|---|
| 14<br>BIT0ERR | Bit0 错误<br>该位指示在非CAN FD报文中或在CAN FD报文的仲裁或数据阶段中，传输和接收的位之间发生不一致的情况。<br>当FlexCAN从伪装网络模式返回到正常模式时，该位会被更新。<br>0b - 没有这种情况发生。<br>1b - 至少有一个作为显性发送的比特被作为隐性接收。 |
| 13 | 确认错误 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| ĂĂĂ | 该位表示发射器节点检测到一个确认错误，即在ACK SLOT期间没有检测到一个主导位。<br>当FlexCAN从伪装网络模式返回到正常模式时，该位会被更新。<br>0b - 没有这种情况发生。<br>1b - 自上次读取该寄存器后发生了一个ACK错误。 |
| 12<br>CRCERR | 循环冗余检查错误<br>该位表示接收节点在非FD报文中或在CAN FD格式的帧的仲裁或数据阶段检测到CRC错误，即计算出的CRC与接收到的不同。<br>当FlexCAN从伪装网络模式返回到正常模式时，该位会被更新。<br>0b - 没有这种情况发生。<br>1b - 自上次读取该寄存器后，发生了一个CRC错误。 |
| 11<br>ĂĂĂ | 表格错误<br>该位表示在非FD报文中检测到形式错误，否则在FD报文的仲裁或数据阶段被接收节点检测到，即固定形式的位域至少包含一个非法位。<br>当FlexCAN从伪装网络模式返回到正常模式时，该位会被更新。<br>0b - 没有这种情况发生。<br>1b - 自上次读取该寄存器后，发生了一个表格错误。 |
| 10<br>ĂĂĂ | 充填错误<br>该位表示接收节点在非FD报文中或在FD报文的仲裁或数据阶段检测到塞满错误。<br>当FlexCAN从伪装网络模式返回到正常模式时，该位会被更新。<br>0b - 没有这种情况发生。<br>1b - 自上次读取该寄存器后，发生了一个塞满错误。 |
| 9<br>辽宁省 | TX错误警告<br>该位指示在信息传输过程中发生的重复错误，仅受CAN_ECR寄存器中TXERRCNT值的影响。该位在冻结模式下不被更新。<br>0b - 没有这种情况发生。<br>1b - TXERRCNT 大于或等于96。 |
| 8<br>荣华富贵 | Rx错误警告<br>该位指示在信息接收过程中何时发生重复错误，仅受CAN_ECR寄存器中RXERRCNT值的影响。该位在冻结模式下不被更新。<br>此外，当FlexCAN从伪装网络模式返回到正常模式时，它会被更新。<br>0b - 没有这种情况发生。<br>1b - RXERRCNT 大于或等于96。 |

*表继续在下一页...*

| | |
|---|---|
| 7<br>怠速 | 怠速<br>该位指示CAN总线何时处于IDLE状态。参见整个CAN_ESR1寄存器描述中的表格。<br>0b - 没有这种情况发生。1b - CAN总线现在是空闲状态。 |
| 6<br>德克萨斯州 | 传输中的FlexCAN<br>该位指示FlexCAN是否正在传输信息。见整个CAN_ESR1寄存器描述中的表格。<br>0b - FlexCAN没有传送信息。1b - FlexCAN正在传输信息。 |

| 场地 | 职能 |
|---|---|
| 5-4<br>FLTCONF | 故障封闭状态<br>这个2位字段表示FlexCAN模块的封闭状态。<br>如果控制寄存器1中的LOM位被断言，在一些取决于CAN位定时的延迟后，FLTCONF字段将显示 "错误被动"。同样的延迟也会影响FLTCONF如何反映CPU对CAN_ECR寄存器的更新。它可能需要一个CAN位的时间来使它们再次连贯。<br>这个位域受到软复位的影响，但是如果LOM位被断言，其复位值只持续一个CAN位。在这段时间之后，FLTCONF报告 "错误被动"。<br>00b - 错误主动 01b - 错误被动 1xb - 总线关闭 |
| 3<br>荣格 | FlexCAN在接待处<br>该位指示FlexCAN是否正在接收一个消息。见整个CAN_ESR1寄存器描述中的表格。<br>0b - FlexCAN没有收到信息。1b - FlexCAN正在接收一个信息。 |
| 2<br>勃芬特<br>(BOFFINT) | 总线关闭中断<br>当FlexCAN进入 "总线关闭 "状态时，该位被设置。如果控制寄存器1(CAN_CTRL1[BOFFMSK])中的相应屏蔽位被设置，就会向CPU产生一个中断。该位可通过写到以下地址来清除<br>1.写0没有效果。<br>0b - 没有这种情况发生。<br>1b - FlexCAN模块进入总线关闭状态。 |
| 1<br>误码率 | 错误中断<br>该位表示至少有一个错误位（BIT1ERR, BIT0ERR, ACKERR. CRCERR, FRMERR 或 STFERR）被设置。如果相应的屏蔽位CAN_CTRL1[ERRMSK]被设置，就会产生一个中断给CPU。这个位可以通过写为1而被清除。写入0没有任何作用。<br>0b - 没有这种情况发生。<br>1b - 表示错误和状态寄存器中任何错误位的设置。 |
| 0<br>- | 保留 |

## 50.4.2.10 中断掩码1寄存器（IMASK1）。

### 50.4.2.10.1 偏移

| 注册 | 偏移 |
|---|---|
| ÃÃÃ | 28h |

表继续在下一页...

### 50.4.2.10.2 职能

该寄存器允许启用或禁用MB31到MB0的32个消息缓冲区的任何数量的中断。它包含每个缓冲区的一个中断屏蔽位，使CPU能够确定在成功传输或接收后，即当相应的CAN_IFLAG1位被设置时，哪个缓冲区会产生一个中断。

### 50.4.2.10.3 图示

| 咬牙切齿 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R<br>W | BUF31TO0M | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 咬牙切齿 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R<br>W | BUF31TO0M | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

### 50.4.2.10.4 领域

| 场地 | 职能 |
|---|---|
| 31-0<br>BUF31TO0M | 缓冲区 MB i 掩码<br>每个位都可以启用或禁用MB31至MB0的相应FlexCAN消息缓冲器中断。<br>**注意：**如果相应的IFLAG1位被设置，设置或清除CAN_IMASK1寄存器中的一个位可以断言或否定一个中断请求。<br>0b - 相应的缓冲区中断被禁用。1b - 相应的缓冲区中断被启用。 |

## 50.4.2.11 中断标志1寄存器（IFLAG1）。

### 50.4.2.11.1 偏移

| 注册 | 偏移 |
|---|---|
| IFLAG1 | 30h |

### 50.4.2.11.2 职能

这个寄存器定义了MB31到MB0的32个消息缓冲区中断的标志。
它包含每个缓冲区的一个中断标志位。每个成功的传输或接收都会设置相应的CAN_IFLAG1位。如果相应的CAN_IMASK1位被设置，将产生一个中断。中断标志必须通过写1来清除。写0没有效果。当Rx
FIFO的DMA被启用时，有一个例外，如下所述。

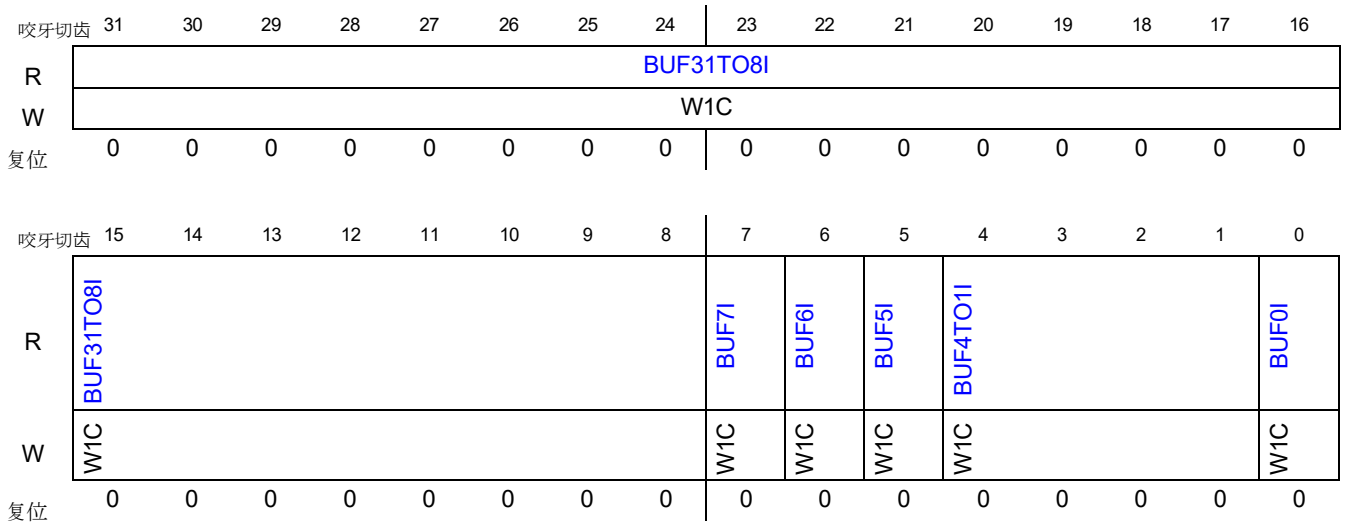
当Rx
FIFO被启用时，BUF7I到BUF5I标志也被用来代表FIFO中断。当位CAN_MCR[RFEN]被设置且位CAN_MCR[DMA]被否定时，8个最不重要的中断标志的功能发生变化。BUF7I、BUF6I和BUF5I表示FIFO的工作状态，BUF0I用于清空FIFO，BUF4I至BUF1I位被保留。

在启用CAN_MCR[RFEN]之前，CPU必须对Rx
FIFO区域的IFLAG位进行服务；参见 "Rx FIFO
"部分。否则，这些IFLAG位会错误地将现在属于FIFO的相关MB显示为有内容要被服务。当CAN_MCR[RFEN]位被否定时，FIFO的标志必须被清除。当选择一个CAN_CTRL2[RFFN]值将Rx
FIFO过滤器扩展到MB7之外时，也必须采取同样的谨慎措施。例如，当RFFN为0x8时，MB0-23的范围被Rx FIFO滤波器占据，相关的IFLAG位必须被清除。

当CAN_MCR[RFEN]和CAN_MCR[DMA]位都被断言时（启用Rx
FIFO的DMA功能），8个最不重要的中断标志（BUF7I -
BUF0I）的功能被改变以支持DMA操作。BUF7I和BUF6I不使用，BUF4I至BUF1I也不使用。BUF5I指示FIFO的工作状态，而BUF0I用于清空FIFO。此外，BUF5I不产生CPU中断，但产生一个DMA请求。当位CAN_MCR[DMA]被启用时，Rx
FIFO区域的IMASK1位不被考虑。此外，当DMA被启用时，CPU不能清除标志BUF5I。在启用位CAN_MCR[DMA]之前，CPU必须对Rx
FIFO区域中的IFLAGs进行服务。当位CAN_MCR[DMA]被否定时，FIFO必须是空的。当CAN_MCR寄存器中的FDEN位被启用时，FIFO必须被禁用。

在更新CAN_MCR[MAXMB]字段之前，CPU必须对MB值大于CAN_MCR[MAXMB]的CAN_IFLAG1位进行服务，否则，它们将保持设置，并与可用的MB数量不一致。

### 50.4.2.11.3 图示




### 50.4.2.11.4 领域

| 场地 | 职能 |
|---|---|
| 31-8<br>BUF31TO8I | 缓冲器MBi中断<br>每个位都标志着MB31至MB8的相应FlexCAN消息缓冲区中断。<br>0b - 相应的缓冲区没有出现成功完成的传输或接收。<br>1b - 相应的缓冲区已成功完成传输或接收。 |
| 7<br>BUF7I | 缓冲器MB7中断或 "Rx FIFO溢出"<br>当CAN_MCR寄存器中的RFEN位被清零时（Rx FIFO被禁用），该位标志着MB7的中断。<br>**注意：**只要CAN_MCR[RFEN]位被CPU写入改变，该标志就会被FlexCAN清零。<br>当CAN_MCR[RFEN]被设置时，BUF7I标志代表 "Rx FIFO溢出"。在这种情况下，该标志表示由于Rx FIFO已满而丢失了一条信息。请注意，当Rx FIFO满了而消息被邮箱捕获时，该标志不会被断言。<br>0b - 当MCR[RFEN]=0时，没有发生MB7完成传输/接收，或当MCR[RFEN]=1时，没有发生Rx FIFO溢出。<br>1b - 当MCR[RFEN]=0时，MB7完成发送/接收，或当MCR[RFEN]=1时，Rx FIFO溢出。 |
| 6<br>BUF6I | 缓冲器MB6中断或 "Rx FIFO警告"<br>当CAN_MCR寄存器中的RFEN位被清零时（Rx FIFO被禁用），该位标志着MB6的中断。<br>**注意：**只要CAN_MCR[RFEN]位被CPU写入改变，该标志就会被FlexCAN清零。 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | 当CAN_MCR[RFEN]被设置时，BUF6I标志代表 "Rx FIFO警告"。在这种情况下，当Rx FIFO内未读信息的数量由于接收到新的信息而从4增加到5时，该标志表示Rx FIFO几乎满了。请注意，如果该标志在未读信息的数量大于4时被清除，它就不会再次断言，直到Rx FIFO内未读信息的数量减少到等于或小于4。<br>0b - 当MCR[RFEN]=0时，没有发生MB6完成传输/接收，或当MCR[RFEN]=1时，Rx FIFO几乎满了。<br>1b - 当MCR[RFEN]=0时，MB6完成发送/接收，或当MCR[RFEN]=1时，Rx FIFO几乎满了。 |
| 5<br>BUF5I | 缓冲器MB5中断或 "Rx FIFO中可用的帧数"<br>当MCR中的RFEN位被清零时（Rx FIFO被禁用），该位标志着MB5的中断。 **注意：**只要MCR[RFEN]位被CPU写入改变，该标志就会被FlexCAN清零。当MCR[RFEN]被设置时（Rx FIFO启用），BUF5I标志代表 "Rx FIFO中可用的帧数"<br>并表示至少有一个帧可以从Rx FIFO中读取。当MCR[DMA]位被启用时，该标志会产生一个DMA请求，CPU不得通过在BUF5I中写1来清除该位。<br>0b - 当MCR[RFEN]=0时，没有发生MB5完成传输/接收，或者当MCR[RFEN]=1时，没有发生FIFO中的可用帧。<br>1b - 当MCR[RFEN]=0时，MB5完成传输/接收，或当MCR[RFEN]=1时，Rx FIFO中可用的帧。在MCR[RFEN]和MCR[DMA]被启用的情况下，它产生一个DMA请求。 |
| 4-1<br>BUF4TO1I | 缓冲区MB i中断或 "保留"<br>当CAN_MCR寄存器中的RFEN位被清零时（Rx FIFO被禁用），这些位标志着MB4到MB1的中断。<br>**注意：**每当CAN_MCR[RFEN]位被CPU写入改变时，这些标志就会被FlexCAN清空。<br>当CAN_MCR[RFEN]被设置时，BUF4TO1I标志被保留。<br>0b - 当MCR[RFEN]=0时，相应的缓冲区没有发生成功完成的发送或接收。<br>1b - 当MCR[RFEN]=0时，相应的缓冲区已经成功完成了传输或接收。 |
| 0<br>BUF0I | 缓冲器MB0中断或清除FIFO位<br>当MCR中的RFEN位被清除时（Rx FIFO被禁用），该位标志着MB0的中断。 如果Rx FIFO被启用，该位被用来触发清除FIFO操作。这个操作会清空FIFO的内容。在执行这个操作之前，CPU必须服务于所有与FIFO相关的IFLAG。当MCR[DMA]位被启用时，该操作也会清除BUF5I标志，从而中止DMA请求。当CPU在BUF0I中写1时，会发生清除FIFO的操作。它仅在冻结模式下允许，在其他条件下被硬件阻止。<br>0b - 当MCR[RFEN]=0时，相应的缓冲区没有发生成功完成的发送或接收。<br>1b - 当MCR[RFEN]=0时，相应的缓冲区已经成功完成了传输或接收。 |

## 50.4.2.12 控制2寄存器（CTRL2）。

### 50.4.2.12.1 偏移

| 注册 | 偏移 |
|---|---|
| CTRL2 | 34h |

### 50.4.2.12.2 职能

该寄存器是对Control1寄存器的补充，为冻结模式下的内存写访问提供控制位，用于扩展FIFO的过滤量，以及调整FlexCAN内部流程的操作，如匹配和仲裁。

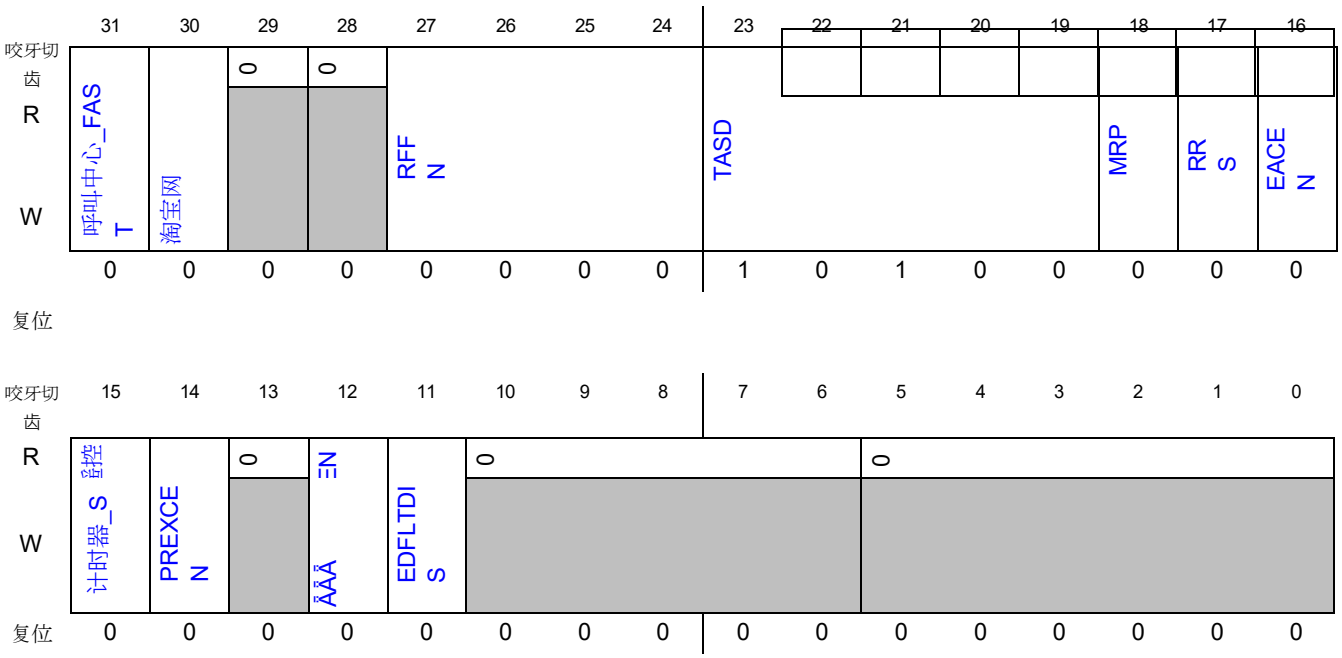
该寄存器的内容不受软复位的影响。

该表显示了Rx
FIFO滤波器结构是如何由CAN_CTRL2[RFFN]的值决定的。更多信息请参见CAN_CTRL2[RFFN]字段描述。

**表50-6.Rx FIFO滤波器：可能的结构**

| RFFN[3:0] | Rx FIFO的数量过滤元件 | Rx FIFO和ID过滤表占用的消息缓冲区 | 剩余的可用邮箱 | Rx FIFO ID过滤表元素受Rx单个掩码的影响 | 受Rx FIFO全局屏蔽影响的Rx FIFO ID过滤表元素 |
|---|---|---|---|---|---|
| 0x0 | 8 | MB 0-7 | MB 8-31 | 元素0-7 | 没有 |
| 0x1 | 16 | MB 0-9 | MB 10-31 | 元素0-9 | 元素 10-15 |
| 0x2 | 24 | MB 0-11 | MB 12-31 | 元素0-11 | 元素 12-23 |
| 0x3 | 32 | MB 0-13 | MB 14-31 | 元素0-13 | 元素 14-31 |
| 0x4 | 40 | MB 0-15 | MB 16-31 | 元素0-15 | 元素16-39 |
| 0x5 | 48 | MB 0-17 | MB 18-31 | 0-17元素 | 元素 18-47 |
| 0x6 | 56 | MB 0-19 | MB 20-31 | 元素0-19 | 元素20-55 |
| 0x7 | 64 | MB 0-21 | MB 22-31 | 元素0-21 | 要素 22-63 |
| 0x8 | 72 | MB 0-23 | MB 24-31 | 元素0-23 | 元素24-71 |
| 0x9 | 80 | MB 0-25 | MB 26-31 | 元素0-25 | 要素 26-79 |
| 0xA | 88 | MB 0-27 | MB 28-31 | 元素0-27 | 元素 28-87 |
| 0xB | 96 | MB 0-29 | MB 30-31 | 元素0-29 | 元素 30-95 |
| 0xC | 104 | MB 0-31 | 没有 | 元素0-31 | 要素 32-103 |

## 50.4.2.12.3 图示




## 50.4.2.12.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>呼叫中心 | 在快速CAN FD帧的数据阶段检测到的错误中断屏蔽<br>该位为CAN_ESR1寄存器中的ERRINT_FAST中断提供一个屏蔽。<br>0b - ERRINT_FAST 错误中断禁用。 1b - ERRINT_FAST 启用错误中断。 |
| 30<br>博夫多纳姆K | 总线关闭完成中断屏蔽<br>该位为CAN_ESR1寄存器中的总线关闭中断提供一个屏蔽。<br>0b - 禁用总线关闭中断。 1b - 启用总线关闭完成中断。 |
| 29<br>- | 保留 |
| 28<br>- | 保留 |

| | |
|---|---|
| 27-24<br>RFFN | Rx FIFO过滤器的数量<br>这个4位字段定义了Rx FIFO滤波器的数量，如表50-6所示。滤波器的最大可选数量由芯片决定。这个字段只能在冻结模式下写入，因为它在其他模式下被硬件屏蔽了。这个字段的编程不能使Rx FIFO和ID过滤器占用的消息缓冲区数量超过CAN_MCR[MAXMB]所定义的现有邮箱数量。<br>**注意：**每组八个过滤器占用的内存空间相当于两个消息缓冲区，这意味着过滤器越多，可用的邮箱就越少。<br>考虑到Rx FIFO占用了原来为MB0-5预留的内存空间，RFFN应该被编程为一个与滤波器数量相对应的值，该值不大于可用的内存字数，可按以下方法计算。<br>(setup_mb - 6) × 4 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | 其中SETUP_MB是参数NUMBER_OF_MB和CAN_MCR[MAXMB]之间的最小值。<br>剩余的可用邮箱数量将是。<br>(setup_mb - 8) - (rffn × 2)<br>如果通过RFFN编程的Rx FIFO滤波器的数量超过了SETUP_MB值（可用的内存空间），超过的部分将无法使用。<br>**注意。** - 最后剩下的可用邮箱的数量由NUMBER_OF_MB减去1和CAN_MCR[MAXMB]字段之间的最小值定义。<br>• 如果不启用 Rx Individual Mask Registers，那么所有 Rx FIFO 过滤器都受 Rx FIFO Global Mask 影响。 |
| 23-19<br>TASD | Tx 仲裁开始延时<br>这个5位字段表示Tx仲裁过程的起始点可以比CAN总线上CRC字段的第一位延迟多少个CAN位。更多细节请参见Tx仲裁启动延迟。这个字段只能在Freeze模式下写入，因为它在其他模式下被硬件屏蔽了。 |
| 18<br>MRP | 信箱 接待优先<br>如果该位被设置，匹配过程从邮箱开始，如果没有匹配，则继续在Rx FIFO上匹配。这个位只能在Freeze模式下写入，因为在其他模式下它会被硬件阻断。<br>0b - 匹配从Rx FIFO开始，在邮箱上继续。1b - 匹配从邮箱开始，在Rx FIFO上继续。 |
| 17<br>RRS | 远程请求存储<br>如果该位被断言，远程请求帧被提交给一个匹配进程，并以数据帧的相同方式存储在相应的消息缓冲区。没有自动的远程响应帧将被生成。<br>如果这个位被否定，远程请求帧就会被提交给一个匹配的过程，如果发现一个CODE=0b1010的消息缓冲区有相同的ID，就会自动生成一个远程响应帧。<br>这个位只能在冻结模式下写入，因为它在其他模式下被硬件封锁了。<br>0b - 产生远程响应帧。1b - 存储远程请求帧。 |
| 16<br>EACEN | 启用Rx邮箱的整个帧仲裁字段比较<br>该位控制Rx邮箱过滤器内的IDE和RTR位与匹配过程中传入帧的相应位的比较。该位不影响 Rx FIFO 的匹配。该位只能在Freeze模式下写入，因为它在其他模式下被硬件阻断。<br>0b - 始终比较Rx Mailbox滤波器的IDE位，尽管有掩码位，但从不比较RTR。1b - 启用Rx Mailbox filter的IDE和RTR位与传入帧中的相应位的比较。屏蔽位适用。 |
| 15<br>计时器_SRC | 定时器来源<br>选择用于自由运行定时器计数器增量的时间刻度源。该位仅在冻结模式下可以写入。<br>**注意：**这个字段并不是在每个实例中都支持。 |

| 支持的 | 不支持 |
| --- | --- |
| FlexCAN0_CTRL2 | FlexCAN1_CTRL2<br>FlexCAN2_CTRL2 |

表继续在下一页...

| 场地 | 职能 |
|---|---|
| | 0b - 自由运行定时器由CAN位时钟提供时钟，它定义了CAN总线上的波特率。<br>1b - 自由运行定时器是由一个外部的时间刻度来计时的。该周期可以调整为与CAN总线上的波特率相等，也可以根据需要调整为不同的值。有关外部时间刻度的细节，请参见设备的具体章节。 |
| 14<br>蓓蕾仙子（PREXCEN） | 启用协议例外<br>该位启用协议例外功能。这个字段只在冻结模式<br>下可写。<br>0b - 协议例外被禁用。1b - 启用协<br>议例外。 |
| 13<br>- | 保留 |
| 12<br>ÃÃÃÃ | ISO CAN FD启用<br>该字段根据ISO规范（ISO 11898-1）启用CAN FD协议（见CAN FD ISO标准）。<br>这个字段只在冻结模式下可写。<br>0b - FlexCAN使用非ISO CAN FD协议运行。<br>1b - FlexCAN使用ISO CAN FD协议（ISO 11898-1）运行。 |
| 11<br>ÃÃÃÃ | 禁用边缘过滤器<br>该位禁用了在总线集成状态下使用的边缘滤波器。当边缘过滤被启用时，需要两个连续的具有主导总线状态的标称时间量子来检测一个导致同步的边缘。当同步发生时，连续11个隐性位的序列的计数被重新开始。边缘过滤器防止短于额定位时间的主导脉冲（在FD帧的数据阶段存在）被误认为是空闲状态。<br>这个字段只在冻结模式下可写。<br>0b - 边缘过滤被启用。1b -<br>边缘过滤被禁用。 |
| 10-6<br>- | 保留 |
| 5-0<br>- | 保留 |

## 50.4.2.13 错误和状态2寄存器（ESR2）。

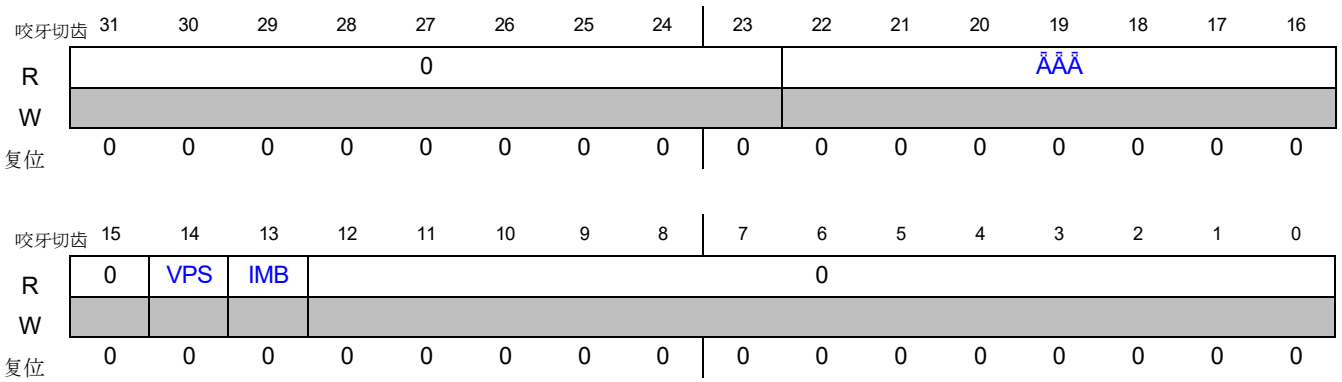
### 50.4.2.13.1 偏移

| 注册 | 偏移 |
|---|---|
| ESR2 | 38h |

### 50.4.2.13.2 职能

该寄存器报告一些一般的状态信息。

### 50.4.2.13.3 图示




### 50.4.2.13.4 领域

| 场地 | 职能 |
|---|---|
| 31-23<br>- | 保留 |
| 22-16<br>ĀĀĀ | 最低优先级的Tx邮箱<br>如果CAN_ESR2[VPS]被断言，这个字段表示最低数量的非活动邮箱（见CAN_ESR2[IMB]位描述）。如果没有不活动的邮箱，那么指示的邮箱就取决于CAN_CTRL1[LBUF]位的值。如果CAN_CTRL1[LBUF]位被否定，那么指示的邮箱是具有最大仲裁值的邮箱（参见 "最高优先级的邮箱优先 "部分）。如果CAN_CTRL1[LBUF]位被断言，那么指示的邮箱是数量最多的活动Tx邮箱。如果一个Tx邮箱正在被传输，则在LPTM计算中不被考虑。如果CAN_ESR2[IMB]没有被断言，并且一个帧被成功传输，LPTM将被更新为其邮箱编号。 |
| 15<br>- | 保留 |
| 14<br>VPS | 有效的优先权状态<br>该位指示CAN_ESR2[IMB]和CAN_ESR2[LPTM]内容当前是否有效。它在每一个完整的Tx仲裁过程中被断言，除非CPU在Tx仲裁过程中向已经被扫描过的邮箱的控制和状态字写入，也就是说，它在Tx仲裁指针后面。如果没有不活动的邮箱，只有一个正在传输的Tx邮箱，那么VPS就不会被断言。该位在每个Tx仲裁过程开始时或在写到任何邮箱的控制和状态字时被否定。<br>**注意：**CAN_ESR2[VPS]不受任何CPU对被中止机制阻断的MB的控制状态（C/S）的写入影响。当CAN_MCR[AEN]被断言时，正在传输的MB的C/S中的中止代码写入（等待中止），或者任何对设置了CAN_IFLAG的Tx MB的写入尝试被阻止。<br>0b - IMB和LPTM的内容是无效的。1b - |

| | |
|---|---|
| | IMB和LPTM的内容是有效的。 |
| 13<br>IMB | 不活跃的邮箱 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | 如果ESR2[VPS]被断言，该位表示是否有任何不活动的邮箱（CODE字段为0b1000或0b0000）。该位在以下情况下被断言。<br>• 在仲裁过程中，如果发现一个CAN_ESR2[LPTM]，并且它是不活动的。<br>• 如果CAN_ESR2[IMB]没有被断言，并且一个帧被成功传输。<br>该位在所有的仲裁开始时都被清除（见 "仲裁过程 "一节）。<br>**注意：**CAN_ESR2[LPTM]机制有以下行为：如果一个MB被成功传输，并且CAN_ESR2[IMB]=0（没有不活动的邮箱），那么CAN_ESR2[VPS]和CAN_ESR2[IMB]被断言，与刚刚传输的MB相关的索引被加载到CAN_ESR2[LPTM]。<br>0b - 如果ESR2[VPS]被断言，ESR2[LPTM]不是一个非活动的邮箱。<br>1b - 如果ESR2[VPS]被断言，至少有一个不活动的邮箱。LPTM的内容是第一个的号码。 |
| 12-0<br>- | 保留 |

## 50.4.2.14 CRC寄存器（CRCR）

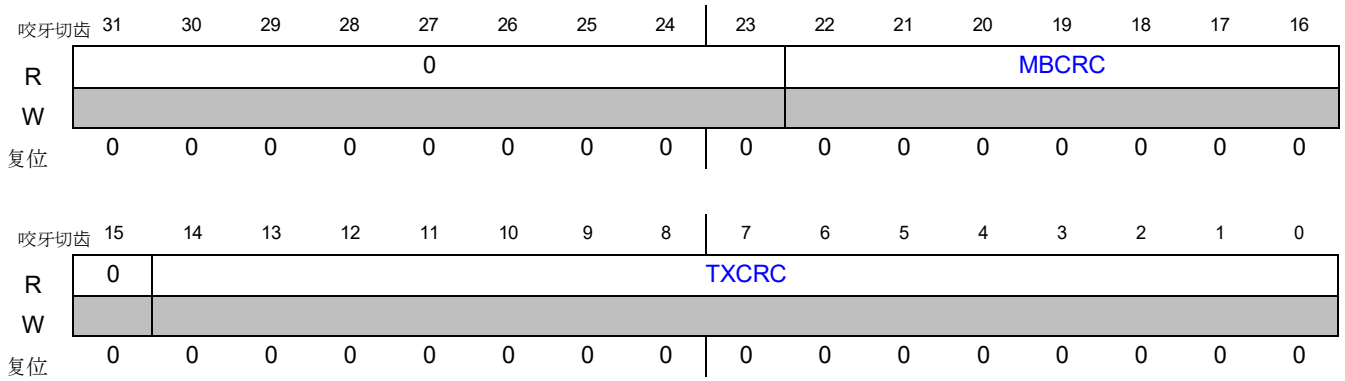
### 50.4.2.14.1 偏移

| 注册 | 偏移 |
| --- | --- |
| CRCR | 44h |

### 50.4.2.14.2 职能

该寄存器提供了关于非FD报文传输的CRC的信息。对于需要17或21位的CAN FD格式的报文，该寄存器只报告CRC计算的15个低阶位。对于CAN FD格式的帧，必须使用CAN_FDCRC寄存器。这个寄存器在Tx中断标志被宣称的同时被更新。

### 50.4.2.14.3 图示




### 50.4.2.14.4 领域

| 场地 | 职能 |
|---|---|
| 31-23<br>- | 保留 |
| 22-16<br>MBCRC | CRC邮箱<br>该字段表示与CAN_CRCR[TXCRC]字段中的值相对应的邮箱的编号。 |
| 15<br>- | 保留 |
| 14-0<br>TXCRC | 传输的CRC值<br>这个字段表示非FD帧的最后一次传输信息的CRC值。对于FD帧，CRC值在CAN_FDCRC寄存器中报告。 |

## 50.4.2.15 Rx FIFO全局屏蔽寄存器（RXFGMASK）

### 50.4.2.15.1 偏移

| 注册 | 偏移 |
|---|---|
| 荣格玛斯克(RXFGMASK) | 48h |

### 50.4.2.15.2 职能

该寄存器位于RAM中。

如果Rx
FIFO被启用，RXFGMASK被用来屏蔽那些根据CAN_CTRL2[RFFN]字段设置没有相应RXIMR的Rx FIFO ID过滤表元素。

这个寄存器只能在冻结模式下写入，因为它在其他模式下被硬件阻断。

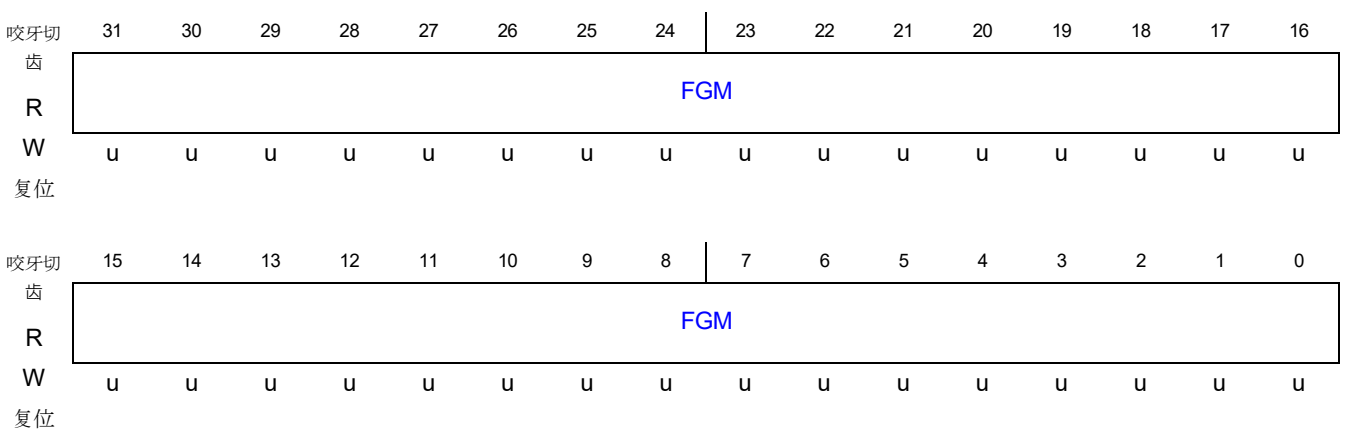
下表显示了FGM位与每个IDAF字段的对应情况。

**表50-7.Rx FIFO全局屏蔽位与IDF字段的对应关系**

| Rx FIFO ID过滤表元素格式（CAN_MCR[IDAM]）。 | 识别器接受过滤字段 | | | | | |
|---|---|---|---|---|---|---|
| | RTR | IDE | RXIDA | RXIDB [1] | RXIDC [2] | 保留 |
| A | 切割女性生殖器官[31] | 切割女性生殖器官[30] | FGM[29:1] | - | - | FGM[0] |
| B | FGM[31], FGM[15] | FGM[30], FGM[14] | - | FGM[29:16]。FGM[13:0] | | - |
| C | - | - | | - | FGM[31:24]。FGM[23:16]。FGM[15:8]。FGM[7:0] | |

1. 如果CAN_MCR[IDAM]字段等同于格式B，只有传入帧的标识符的14个最有效位与Rx FIFO滤波器比较。
2. 如果CAN_MCR[IDAM]字段等同于格式C，只有传入帧的标识符的8个最有效位与Rx FIFO滤波器进行比较。

### 50.4.2.15.3 图示




### 50.4.2.15.4 领域

| 场地 | 职能 |
|---|---|

| | |
|---|---|
| 31-0<br>FGM | Rx FIFO全局掩码位<br><br>这些位掩盖了ID过滤表元素位的完美排列。 |

| 场地 | 职能 |
| --- | --- |
| | 0b - 过滤器中的相应位是 "不关心"。1b - 过滤器中的<br>相应位被检查。 |

## 50.4.2.16 Rx FIFO信息寄存器（RXFIR）。

### 50.4.2.16.1 偏移

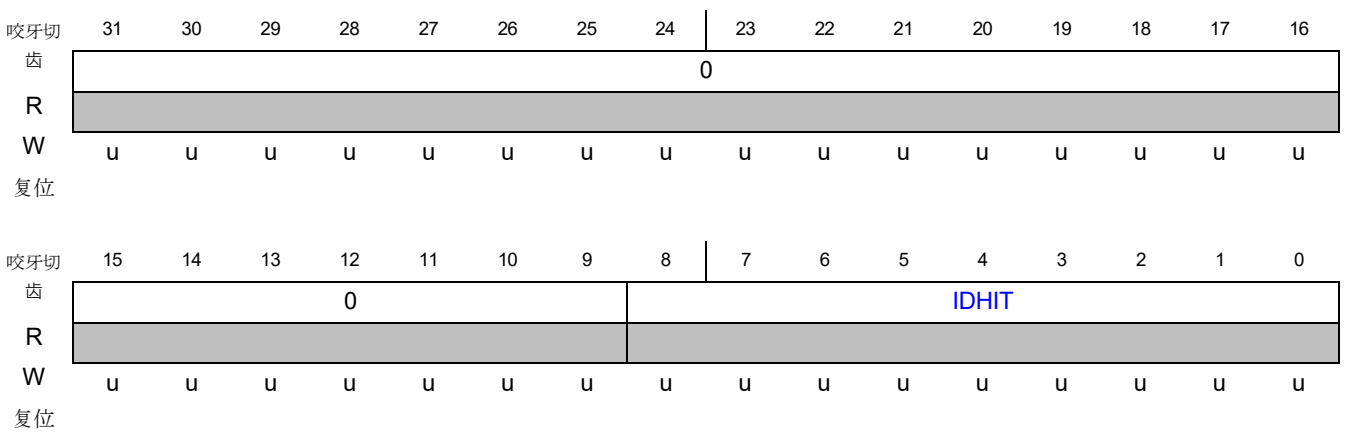
| 注册 | 偏移 |
| --- | --- |
| RXFIR | 4Ch |

### 50.4.2.16.2 职能

RXFIR提供关于Rx FIFO的信息。

该寄存器是CPU访问位于RAM中的RXFIR
FIFO输出的端口。每当有新的信息进入Rx FIFO时，FlexCAN就会写入RXFIR
FIFO，每当Rx
FIFO的输出被更新为下一个信息时，它的输出也会被更新。有关读取该寄存器的说明，请参见 "Rx FIFO "部分。

### 50.4.2.16.3 图示



## 50.4.2.16.4 领域

| 场地 | 职能 |
| --- | --- |
| 31-9 | 保留 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| - | |
| 8-0<br>IDHIT | 识别器接受度过滤器命中指标<br>这个字段表示在Rx FIFO的输出中，哪个识别器接受过滤器被收到的消息击中。如果有多个过滤器与传入的报文ID相匹配，那么将显示由匹配过程发现的第一个匹配的IDAF（最低的数字）。该字段仅在CAN_IFLAG1[BUF5I]被断言时有效。 |

## 50.4.2.17 CAN位定时寄存器（CBT）

### 50.4.2.17.1 偏移

| 注册 | 偏移 |
| --- | --- |
| CBT | 50h |

### 50.4.2.17.2 职能

这个寄存器是存储CAN_CTRL1寄存器中描述的CAN位定时变量的另一种方式。EPRESDIV、EPROPSEG、EPSEG1、EPSEG2和ERJW为分别是PRESDIV、PROPSEG、PSEG1、PSEG2和RJW位域的扩展版本。

BTF位选择使用该寄存器中定义的定时变量。该寄存器的内容不受软复位的影响。

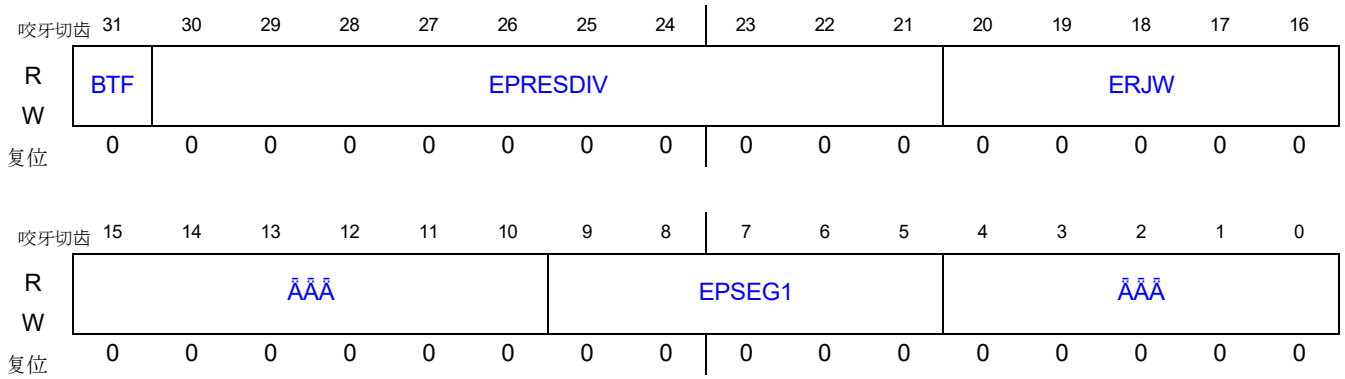
**注意事项**

CAN_CTRL1和CAN_CBT中的CAN位变量被存储在同一个寄存器中。

**注意事项**

当CAN FD功能被启用时（CAN_MCR[FDEN]被设置），总是设置CAN_CBT[BTF]。

### 50.4.2.17.3 图示




### 50.4.2.17.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>BTF | 位计时格式启用<br>启用扩展的CAN位定时字段EPRESDIV、EPROPSEG、EPSEG1、EPSEG2和ERJW，取代CAN_CTRL1寄存器中定义的CAN位定时变量。这个字段只能在Freeze模式下写入。<br>0b - 禁用扩展位时间定义。1b - 启用扩展位时间定义。 |
| 30-21<br>EPRESDIV | 扩展预分频器分频系数<br>这个10位字段定义了当CAN_CBT[BTF]位被断言时，PE时钟频率和串行时钟（Sclock）频率之间的比率，否则它没有任何影响。它扩展了CAN_CTRL1[PRESDIV]的数值范围。<br>Sclock周期定义了CAN协议的时间量子。对于复位值，Sclock频率等于PE时钟频率（见协议计时）。这个字段只能在Freeze模式下写入，因为它在其他模式下被硬件阻断。<br>Sclock频率=PE时钟频率/(EPRESDIV + 1) |
| 20-16<br>ERJW | 扩展同步跳转宽度<br>这个5位字段定义了当CAN_CBT[BTF]位被断言时，一个位时间可以通过一次重新同步来改变的最大时间量子数，否则没有任何作用。它扩展了CAN_CTRL1[RJW]的数值范围。<br>一个时间量子等于Sclock周期。这个字段只能在Freeze模式下写入，因为它在其他模式下被硬件屏蔽了。<br>同步跳转宽度=ERJW+1。 |
| 15-10<br>ÃÃÃ | 扩展传播段<br>这个6位字段定义了当CAN_CBT[BTF]位被断言时，传播段在位时间内的长度，否则它没有任何作用。它扩展了CAN_CTRL1[PROPSEG]的数值范围。这个字段只能在Freeze模式下写入，因为它在其他模式下被硬件阻断。<br>传播段时间=（EPROPSEG+1）×时间-量子。时间-量子=一个Sclock周期。 |

| | |
|---|---|
| 9-5<br>EPSEG1 | 扩展阶段段1 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | 这个5位字段定义了当CAN_CBT[BTF]位被断言时，相位段1在位时间内的长度，否则它没有任何作用。<br>它扩展了CAN_CTRL1[PSEG1]的数值范围。这个字段只能在Freeze模式下写入，因为它在其他模式下被硬件阻断。<br>相位缓冲段1 = (EPSEG1 + 1) × 时间-量子。时间-量子=一<br>个Sclock周期。 |
| 4-0<br>ÃÃÃ | 扩展阶段第2段<br>这个5位字段定义了当CAN_CBT[BTF]位被断言时，相位段2在位时间内的长度，否则它没有任何作用。<br>它扩展了CAN_CTRL1[PSEG2]的数值范围。这个字段只能在Freeze模式下写入，因为它在其他模式下被硬件阻断。<br>相位缓冲段1 = (EPSEG2 + 1) × 时间-量子。时间-量子=一<br>个Sclock周期。 |

## 50.4.2.18 Rx单个屏蔽寄存器（RXIMR0 - RXIMR31）

### 50.4.2.18.1 偏移

对于n=0到31。

| 注册 | 偏移 |
| --- | --- |
| RXIMRn | 880h + (n × 4h) |

### 50.4.2.18.2 职能

RX单个掩码寄存器用于存储Rx MB和Rx FIFO中ID过滤的接受掩码。

当Rx
FIFO被禁用时（CAN_MCR[RFEN]位被否定），为每个可用的Rx邮箱提供一对一的屏蔽。当Rx
FIFO被启用时（CAN_MCR[RFEN]位被断言），根据CAN_CTRL2[RFFN]的设置（见Rx FIFO），为每个Rx FIFO ID过滤表元素提供一对一的屏蔽。

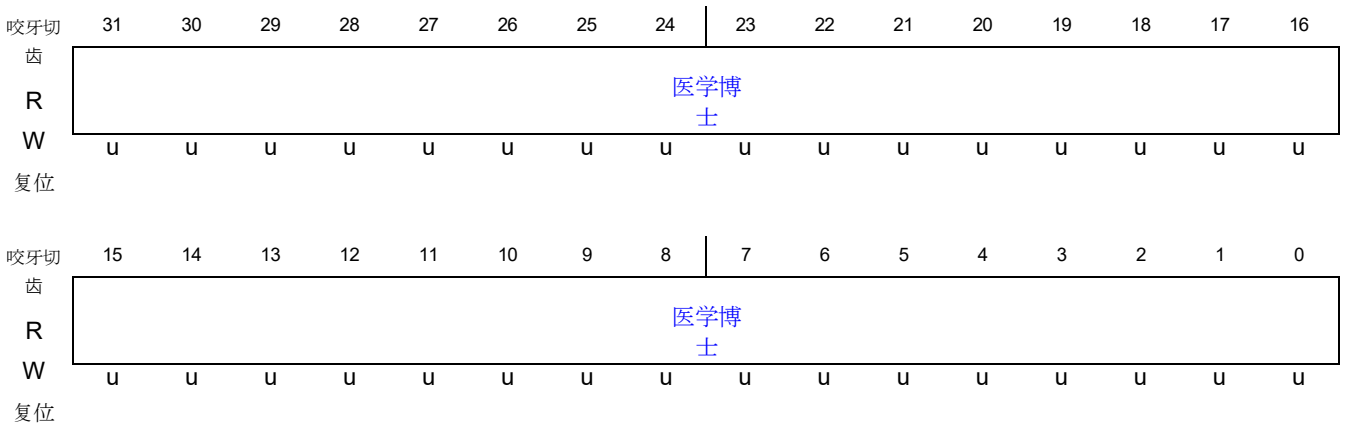
CAN_RXIMR0存储与MB0或ID过滤表元素0相关的单个掩码，CAN_RXIMR1存储与MB1或ID过滤表元素1相关的单个掩码，以此类推。

CAN_RXIMR寄存器只有在模块处于冻结模式时才能被CPU访问，否则，它们会被硬件封锁。这些寄存器不受复位的影响。它们位于RAM中，必须在任何接收之

前明确地进行初始化。

RXIMR内存区域有可能被作为通用内存访问。更多信息见总线接口。

### 50.4.2.18.3 图示




### 50.4.2.18.4 领域

| 场地 | 职能 |
|---|---|
| 31-0<br>医学博士 | 单独的掩码位<br>每个单独的屏蔽位都以不同的方式屏蔽邮箱过滤器和Rx FIFO ID过滤器表元素中的相应位。<br>关于邮箱过滤器，见RXMGMASK寄存器描述。<br>关于Rx FIFO ID过滤表元素，见RXFGMASK寄存器描述。<br>0b - 过滤器中的相应位是 "不关心"。 1b - 过滤器中的相应位被检查。 |

## 50.4.2.19 假装网络控制1寄存器（CTRL1_PN）。

### 50.4.2.19.1 偏移

| 注册 | 偏移 |
|---|---|
| CTRL1_PN | B00h |

### 50.4.2.19.2 职能

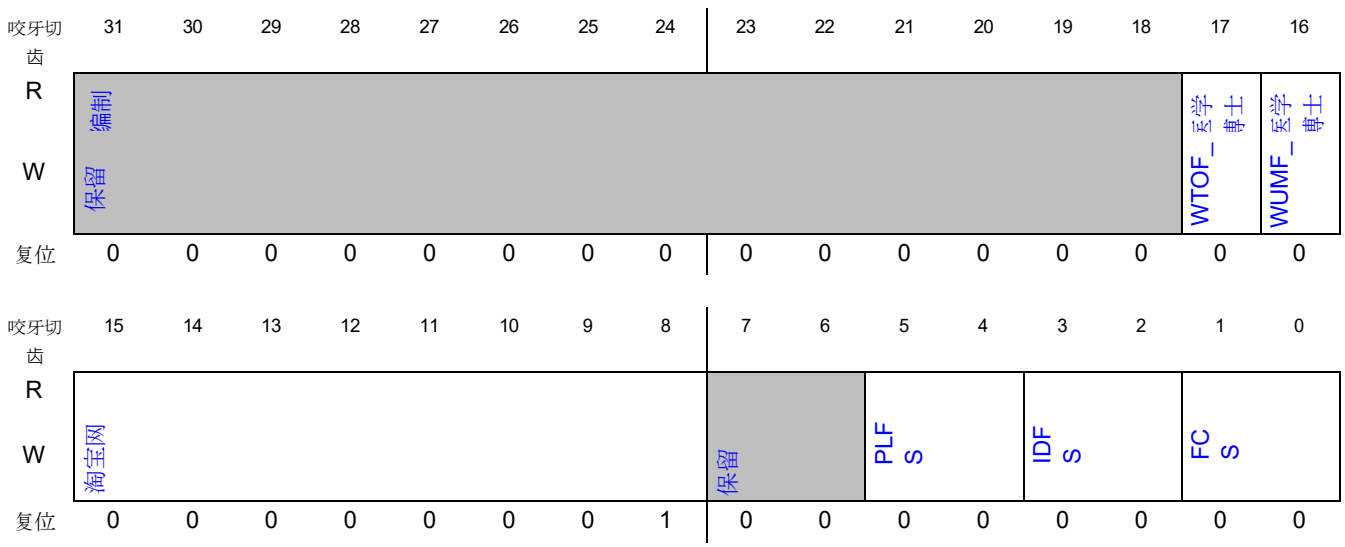
该寄存器包含用于假装网络模式过滤选择的控制位。用接收唤醒信息的过滤标准来配置这个寄存器。除了WTOF_MSK和WUMF_MSK位，它只能在冻结模式下写入。

### 注意事项

这个寄存器不是每个模块实例都使用的。

- 适用于FlexCAN0。
- 不适用于FlexCAN1、FlexCAN2。

## 50.4.2.19.3 图示




## 50.4.2.19.4 领域

| 场地 | 职能 |
|---|---|
| 31-18<br>- | 保留 |
| 17<br>WTOF_MSK | 超时唤醒标志掩码位<br>这个位掩盖了一个由超时引起的唤醒事件的产生。<br>0b - 超时唤醒事件被禁用 1b - 超时唤醒事件被启用 |
| 16<br>WUMF_MSK | 通过匹配标志掩码位唤醒<br>该位掩盖了由成功过滤的Rx信息引起的唤醒事件的产生。<br>0b - 唤醒匹配事件被禁用 1b - 唤醒匹配事件被启用 |

| | |
|---|---|
| 15-8<br>淘宝网 | 符合相同过滤标准的信息数量<br>这个8位字段定义了在产生唤醒事件之前，给定信息必须符合ID和/或PL的预定义过滤标准的次数。这个数量可以通过使用0x01到0xFF的值分别配置在1到255的范围内。<br>00000001b - 在产生唤醒事件之前，收到的信息必须与ID和/或PL的预定义过滤标准匹配一次。<br>00000010b - 收到的信息必须符合预定的ID和/或PL的过滤标准两次。<br>在产生一个唤醒事件之前。 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | 11111111b - 收到的信息必须符合预定的ID和/或PL的过滤标准255次，才能产生唤醒事件。 |
| 7-6<br>- | 保留 |
| 5-4<br>PLFS | 有效载荷过滤选择<br>这个2位字段选择FlexCAN在伪装网络模式下应用的有效载荷过滤级别。当有效载荷过滤激活时，过滤不接受远程信息（RTR=1）。<br>00b - 根据有效载荷内容与精确的目标值进行匹配<br>01b - 在有效载荷值大于或等于指定目标值时进行匹配 10b - 在有效载荷值小于或等于指定目标值时进行匹配<br>11b - 在一个范围内的有效载荷值上进行匹配，大于或等于指定的下限，小于或等于指定的上限 |
| 3-2<br>IDFS | ID过滤选择<br>这个2位字段选择了FlexCAN在伪装网络模式下应用的ID过滤级别。在ID过滤中，如果CAN_FLT_ID2_IDMASK寄存器中的IDE_MSK和RTR_MSK位被设置，IDE和RTR位也被视为接收过滤的一部分。<br>00b - 根据一个ID内容与一个确切的目标值进行匹配<br>01b - 在ID值大于或等于指定目标值时进行匹配 10b - 在ID值小于或等于指定目标值时进行匹配<br>11b - 在一个范围内的ID值进行匹配，大于或等于指定的下限，小于或等于指定的上限。 |
| 1-0<br>FCS | 筛选组合选择<br>这个2位字段用于选择FlexCAN在伪装网络模式下应用的过滤标准。详情请见伪装网络模式下的接收过程。<br>00b - 只过滤消息ID<br>01b - 消息ID过滤和有效载荷过滤<br>10b - 信息ID过滤发生了指定的次数。<br>11b - 消息ID过滤和有效载荷过滤的指定次数 |

# 50.4.2.20 假装网络控制2寄存器（CTRL2_PN）。

## 50.4.2.20.1 偏移

| 注册 | 偏移 |
| --- | --- |
| CTRL2_PN | B04h |

## 50.4.2.20.2 职能

该寄存器包含在假装联网模式下超时值的配置位。它只能在冻结模式下被写入。

### 注意事项

这个寄存器不是每个模块实例都使用的。

- 适用于FlexCAN0。
- 不适用于FlexCAN1、FlexCAN2。

## 50.4.2.20.3 图示




## 50.4.2.20.4 领域

| 场地 | 职能 |
|---|---|
| 31-16<br>- | 保留 |
| 15-0<br>マッセント | 没有符合过滤标准的信息的超时<br>这个16位字段定义了一个超时值，如果CAN_MCR[PNET_EN]被断言，则产生一个唤醒事件。如果超时计数器达到目标值，当FlexCAN处于伪装网络模式下时，就会产生一个唤醒事件。超时限制可以在1到65535之间进行配置，以控制一个内部的16位向上计数定时器，在达到这个配置值时产生一个触发。内部定时器是根据周期性的时间刻度来递增的，这个周期是CAN比特时间单位的64倍。当MATCHTO被配置为0x0000时，超时被禁用。 |

# 50.4.2.21 假装网络唤醒匹配寄存器（WU_MTC）。

## 50.4.2.21.1 偏移

| 注册 | 偏移 |
|---|---|
| 吴国华_MTC | B08h |

### 50.4.2.21.2 职能

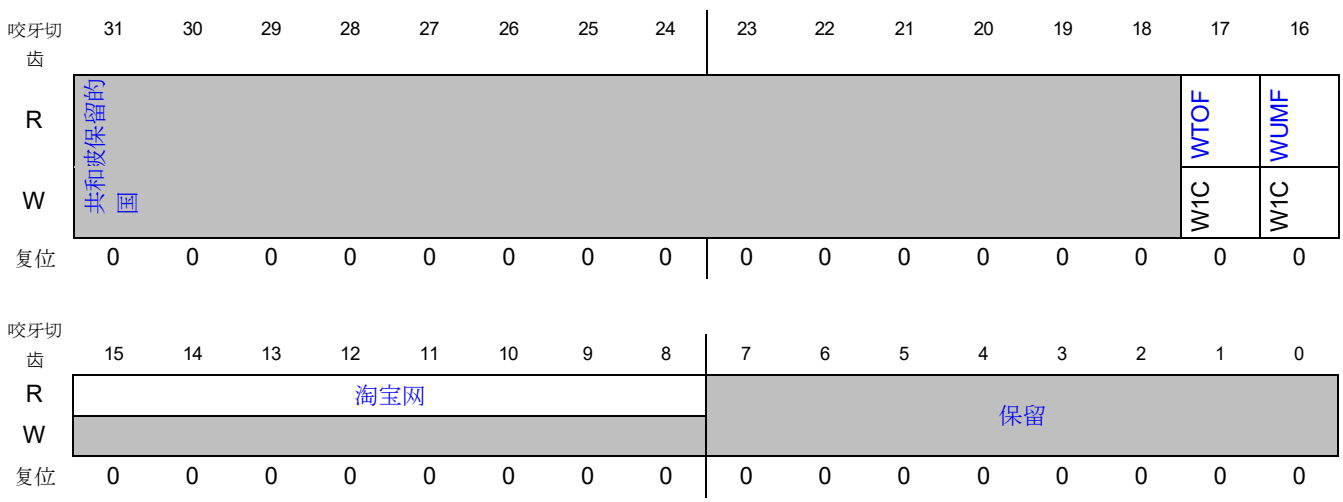
这个只读寄存器包含与FlexCAN在假装网络模式下接收帧时执行的匹配过程有关的唤醒信息。

**注意事项**

这个寄存器不是每个模块实例都使用的。

- 适用于FlexCAN0。
- 不适用于FlexCAN1、FlexCAN2。

### 50.4.2.21.3 图示




### 50.4.2.21.4 领域

| 场地 | 职能 |
|---|---|
| 31-18<br>- | 保留 |
| 17<br>WTOF | 通过超时唤醒标志位<br>该位识别FlexCAN在CAN_CTRL2_PN[MATCHTO]定义的时间间隔内是否检测到超时事件。如果CAN_CTRL1_PN[WTOF_MSK]被启用，该标志会产生一个唤醒事件。<br>0b - 没有检测到超时事件的唤醒 1b - 检测到超时事件的唤醒 |

| | |
|---|---|
| 16<br>WUMF | 用火柴旗位唤醒<br>该位标识FlexCAN是否检测到一个匹配的Rx传入信息，该信息通过了CAN_CTRL1_PN寄存器中指定的过滤标准。如果CAN_CTRL1_PN[WUMF_MSK]被启用，该标志会产生一个唤醒事件。<br>0b - 没有检测到匹配事件而唤醒 1b - 检测到匹配事件而唤醒 |
| 15-8 | 假装联网时的匹配数量 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| 淘宝网 | 这个8位的字段报告了在唤醒事件发生前，一个给定的信息与预定义的ID和/或PL过滤标准相匹配的次数。<br>当FlexCAN进入假装联网模式时，该寄存器被重置，不受软重置的影响。 |
| 7-0<br>- | 保留 |

## 50.4.2.22 假装的网络ID过滤器1寄存器（FLT_ID1）

### 50.4.2.22.1 偏移

| 注册 | 偏移 |
|---|---|
| FLT_ID1 | B0Ch |

### 50.4.2.22.2 职能

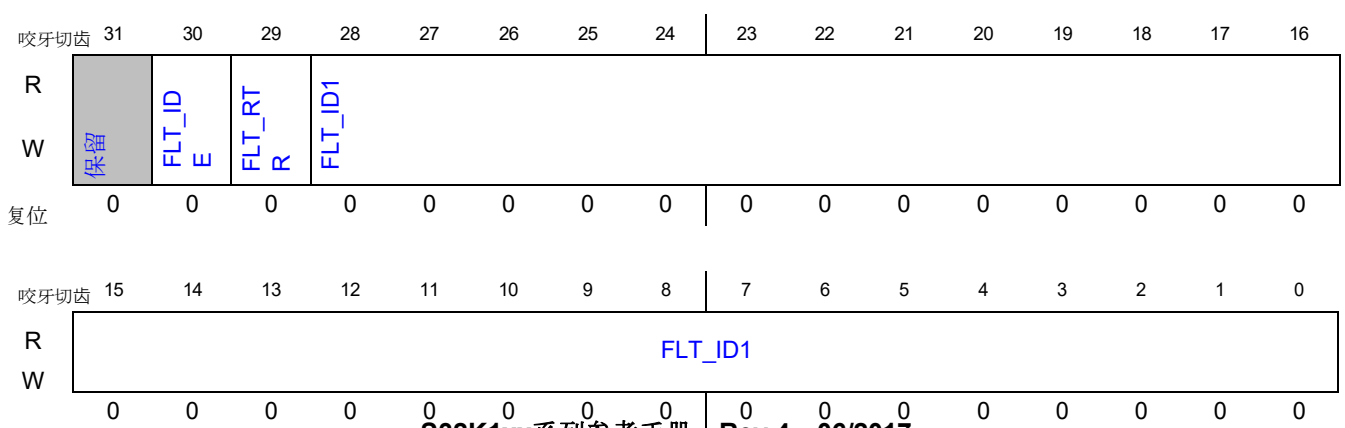
该寄存器包含FLT_ID1目标值，以及IDE和RTR目标值，用于过滤传入的信息ID。FLT_ID1用于等于、小于、大于的比较，或作为ID范围检测的下限值。它只能在冻结模式下写入。

**注意事项**

这个寄存器不是每个模块实例都使用的。

- 适用于FlexCAN0。
- 不适用于FlexCAN1、FlexCAN2。

### 50.4.2.22.3 图示



复位

#### 50.4.2.22.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>- | 保留 |
| 30<br>FLT_IDE | ID扩展过滤器<br>该位识别帧格式是标准还是扩展。它被用作ID接收过滤器的一部分。<br>0b - 接受标准帧格式 1b - 接受扩展帧格式 |
| 29<br>FLT_RTR | 远程传输请求过滤器<br>该位识别帧是否是远程的。它被用作ID接收过滤器的一部分。<br>0b - 拒绝远程帧（接受数据帧） 1b - 接受远程帧 |
| 28-0<br>FLT_ID1 | 假装网络过滤的ID过滤器1<br>这个29位的字段定义了扩展帧格式的29位，考虑所有的位，或者标准帧格式的11位，只考虑最左边的11位。 |

### 50.4.2.23 假装联网DLC过滤寄存器（FLT_DLC）。

#### 50.4.2.23.1 偏移

| 注册 | 偏移 |
|---|---|
| FLT_DLC | B10h |

#### 50.4.2.23.2 职能

该寄存器包含DLC内部范围的目标值（FLT_DLC_LO和FLT_DLC_HI），用于过滤进入的信息。DLC范围只用于有效载荷过滤。它只能在冻结模式下被写入。

**注意事项**

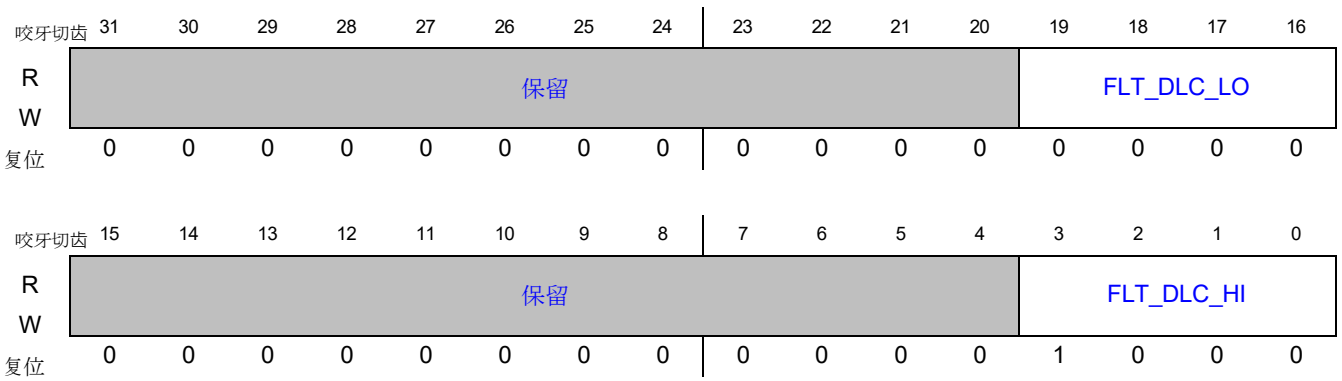
当需要固定数量的数据字节时，FLT_DLC_LO和FLT_DLC_HI都需要配置相同的值，否则会考虑DLC的范围进行过滤（见假装联网模式下的接收过程）。

**注意事项**

这个寄存器不是每个模块实例都使用的。

- 适用于FlexCAN0。
- 不适用于FlexCAN1、FlexCAN2。

### 50.4.2.23.3 图示




### 50.4.2.23.4 领域

| 场地 | 职能 |
|---|---|
| 31-20<br>- | 保留 |
| 19-16<br>FLT_DLC_LO | 数据字节长度的下限过滤器<br>该字段规定了被认为对有效的有效载荷比较的数据字节数的下限。它是作为有效载荷接收过滤器的一部分使用。 |
| 15-4<br>- | 保留 |
| 3-0<br>FLT_DLC_HI | 数据字节长度的上限过滤器<br>该字段规定了被认为对有效的有效载荷比较的数据字节数的上限。它是作为有效载荷接收过滤器的一部分使用。 |

## 50.4.2.24 假装网络有效载荷低滤波1寄存器（PL1_ LO）。

### 50.4.2.24.1 偏移

| 注册 | 偏移 |
| --- | --- |
| PL1_LO | B14h |

### 50.4.2.24.2 职能

该寄存器包含用于过滤传入信息有效载荷的目标值的有效载荷过滤器1的低阶位。它既可用于 "等于"、"小于或等于"、"大于或等于
"的比较，也可作为有效载荷范围检测的下限值。它只能在冻结模式下写入。

**注意事项**

这个寄存器不是每个模块实例都使用的。

- 适用于FlexCAN0。
- 不适用于FlexCAN1、FlexCAN2。

### 50.4.2.24.3 图示

| 咬牙切齿 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R<br>W | 数据_字节_0 | | | | | | | | 数据_byte_1 | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 咬牙切齿 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R<br>W | 数据_字节_2 | | | | | | | | 数据_byte_3 | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

### 50.4.2.24.4 领域

| 场地 | 职能 |
|---|---|
| 31-24<br>数据_字节_0 | 有效载荷过滤1低阶位，用于伪装网络有效载荷过滤，对应于数据字节0。 |
| 23-16<br>数据_byte_1 | 有效载荷过滤1低阶位，用于伪装网络有效载荷过滤，对应于数据字节1。 |
| 15-8<br>数据_字节_2 | 有效载荷过滤1低阶位，用于伪装网络有效载荷过滤，对应于数据字节2。 |
| 7-0<br>数据_byte_3 | 有效载荷过滤1低阶位，用于伪装网络有效载荷过滤，对应于数据字节3。 |

## 50.4.2.25 伪装网络有效载荷高过滤1寄存器(PL1_ HI)

### 50.4.2.25.1 偏移

| 注册 | 偏移 |
|---|---|
| PL1_HI | B18h |

### 50.4.2.25.2 职能

该寄存器包含用于过滤传入信息有效载荷的目标值的有效载荷过滤器1的高阶位。它用于 "等于"、"小于或等于"、"大于或等于"的比较，或作为有效载荷范围检测的下限值。它只能在冻结模式下写入。

**注意事项**

这个寄存器不是每个模块实例都使用的。

- 适用于FlexCAN0。
- 不适用于FlexCAN1、FlexCAN2。

### 50.4.2.25.3 图示

| 咬牙切齿 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R<br>W | 数据_byte_4 | | | | | | | | 数据_byte_5 | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 咬牙切齿 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R<br>W | 数据_byte_6 | | | | | | | | 数据_byte_7 | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

### 50.4.2.25.4 领域

| 场地 | 职能 |
|---|---|
| 31-24<br>数据_byte_4 | 有效载荷过滤1高阶位，用于伪装网络有效载荷过滤，对应于数据字节4。 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| 23-16<br>数据_byte_5 | 有效载荷过滤1高阶位，用于伪装网络有效载荷过滤，对应于数据字节5。 |
| 15-8<br>数据_byte_6 | 有效载荷过滤1高阶位，用于伪装网络有效载荷过滤，对应于数据字节6。 |
| 7-0<br>数据_byte_7 | 有效载荷过滤1高阶位，用于伪装网络有效载荷过滤，对应于数据字节7。 |

## 50.4.2.26 假装网络ID过滤2寄存器/ID屏蔽寄存器（FLT_ID2_IDMASK）

### 50.4.2.26.1 偏移

| 注册 | 偏移 |
| --- | --- |
| FLT_ID2_IDMASK | B1Ch |

### 50.4.2.26.2 职能

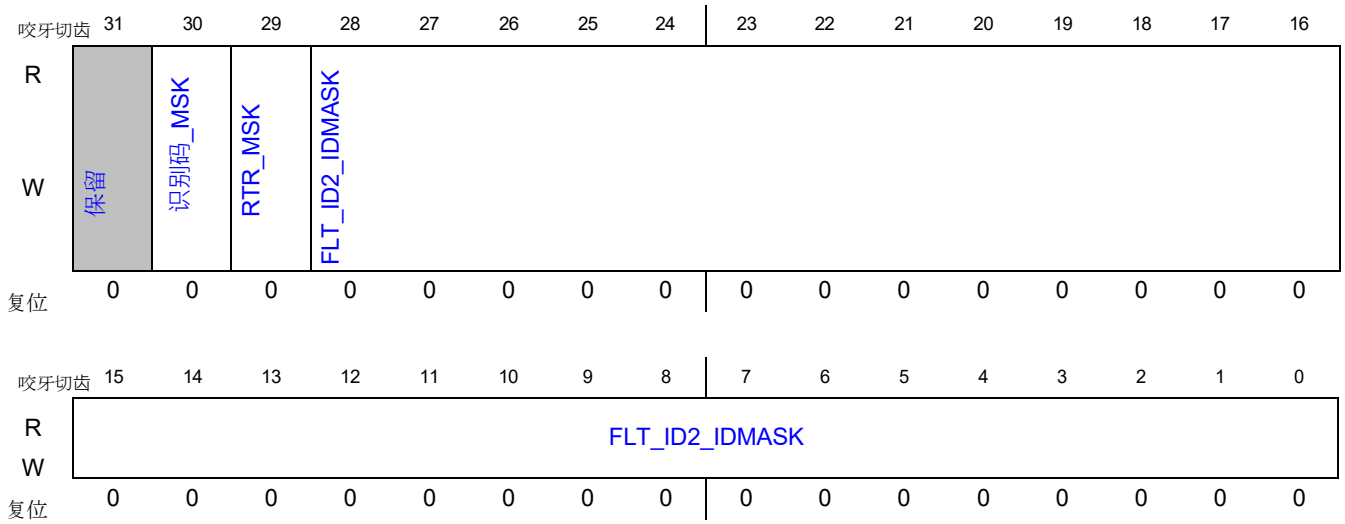
该寄存器包含FLT_ID2目标值，仅作为ID范围检测的上限值。另外，当选择精确的ID过滤标准时，该寄存器被用来存储ID掩码。IDE_MSK和RTR_MSK位同时用于ID过滤（精确和范围），使FLT_IDE和FLT_RTR分别作为ID接收过滤器的一部分使用。该寄存器只能在Freeze模式下写入。

**注意事项**

这个寄存器不是每个模块实例都使用的。

- 适用于FlexCAN0。
- 不适用于FlexCAN1、FlexCAN2。

### 50.4.2.26.3 图示




### 50.4.2.26.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>- | 保留 |
| 30<br>识别码_MSK | ID扩展掩码位<br>该位表示帧格式（标准/扩展）是否被用作ID接收过滤器的一部分。<br>0b - 过滤器中的相应位是 "不关心" 1b - 过滤器中的<br>相应位被检查 |
| 29<br>RTR_MSK | 远程传输请求掩码位<br>该位表示帧类型（数据/远程）是否是ID接收过滤器的一部分。<br>0b - 过滤器中的相应位是 "不关心" 1b - 过滤器中的<br>相应位被检查 |
| 28-0<br>呼叫中心： 呼叫中心： 呼叫中心： 呼叫中心： 呼叫中心： 呼叫中心： 呼叫中心： 呼叫中心 | 用于伪装网络过滤的ID过滤2/用于伪装网络ID过滤的ID掩码比特<br>该寄存器用于定义扩展帧格式的ID过滤值（29位），考虑FLT_ID2[28:0]，或标准帧格式的ID过滤值（11位），考虑FLT_ID2[28:18]（[17:0]范围内的其他位没有意义）。仅在ID过滤的范围内使用。<br>它也可以用来定义扩展帧格式（29位）的掩码值，考虑IDMASK[28:0]，或者标准帧格式（11位），考虑IDMASK[28:18]（[17:0]范围内的其他位没有意义）。仅在精确ID过滤中使用。 |

## 50.4.2.27 假装联网有效载荷低位过滤2寄存器/有效载荷低位屏蔽寄存器（PL2_PLMASK_LO）。

### 50.4.2.27.1 偏移

| 注册 | 偏移 |
|---|---|
| PL2_PLMASK_LO | B20h |

### 50.4.2.27.2 职能

该寄存器有两个功能。首先，它包含了有效载荷过滤2的低阶位，仅作为有效载荷范围检测的上限值。第二，当选择精确的有效载荷过滤标准时，该寄存器被用作有效载荷屏蔽。否则，该寄存器将不被使用。它只能在冻结模式下被写入。

**注意事项**

这个寄存器不是每个模块实例都使用的。

- 适用于FlexCAN0。
- 不适用于FlexCAN1、FlexCAN2。

### 50.4.2.27.3 图示

| 咬牙切齿 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R<br>W | 数据_字节_0 | | | | | | | | 数据_byte_1 | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 咬牙切齿 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R<br>W | 数据_字节_2 | | | | | | | | 数据_byte_3 | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

### 50.4.2.27.4 领域

| 场地 | 职能 |
|---|---|
| 31-24<br>数据_字节_0 | 有效载荷过滤2低阶位/有效载荷屏蔽低阶位，用于伪装网络有效载荷过滤，对应于数据字节0。 |
| 23-16<br>数据_byte_1 | 有效载荷过滤2低阶位/有效载荷屏蔽低阶位，用于伪装网络有效载荷过滤，对应于数据字节1。 |
| 15-8<br>数据_字节_2 | 有效载荷过滤2低阶位/有效载荷屏蔽低阶位，用于伪装网络有效载荷过滤，对应于数据字节2。 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| 7-0<br>数据_byte_3 | 有效载荷过滤2低阶位/有效载荷屏蔽低阶位，用于伪装网络有效载荷过滤，对应于数据字节3。 |

## 50.4.2.28 假装联网的有效载荷高过滤2低阶位/有效载荷高屏蔽寄存器（PL2_PLMASK_HI）。

### 50.4.2.28.1 偏移

| 注册 | 偏移 |
|---|---|
| PL2_PLMASK_HI | B24h |

### 50.4.2.28.2 职能

该寄存器有两个功能。首先，它包含了有效载荷过滤2的高阶位，仅作为有效载荷范围检测的上限值。第二，当选择精确的有效载荷过滤标准时，该寄存器被用作有效载荷屏蔽。否则，该寄存器将不被使用。它只能在冻结模式下被写入。

**注意事项**

这个寄存器不是每个模块实例都使用的。

- 适用于FlexCAN0。
- 不适用于FlexCAN1、FlexCAN2。

### 50.4.2.28.3 图示

| 咬牙切齿 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R<br>W | 数据_byte_4 | | | | | | | | 数据_byte_5 | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 咬牙切齿 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R<br>W | 数据_byte_6 | | | | | | | | 数据_byte_7 | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

#### 50.4.2.28.4 领域

| 场地 | 职能 |
| --- | --- |
| 31-24<br>数据_byte_4 | 有效载荷过滤2高阶位/有效载荷屏蔽高阶位，用于伪装网络有效载荷过滤，对应于数据字节4。 |
| 23-16<br>数据_byte_5 | 有效载荷过滤2高阶位/有效载荷屏蔽高阶位，用于伪装网络有效载荷过滤，对应于数据字节5。 |
| 15-8<br>数据_byte_6 | 有效载荷过滤2高阶位/有效载荷屏蔽高阶位，用于伪装网络有效载荷过滤，对应于数据字节6。 |
| 7-0<br>数据_byte_7 | 有效载荷过滤2高阶位/有效载荷屏蔽高阶位，用于伪装网络有效载荷过滤，对应于数据字节7。 |

### 50.4.2.29 C/S的唤醒信息缓冲寄存器（WMB0_CS - WMB3_CS）。

#### 50.4.2.29.1 偏移

对于a=0到3。

| 注册 | 偏移 |
| --- | --- |
| WMBa_CS | B40h + (a × 10h) |

#### 50.4.2.29.2 职能

四个WMB中的每一个都包含一个寄存器，用于存储传入Rx信息的控制状态（C/S）信息（IDE、RTR和DLC字段）。

**注意事项**

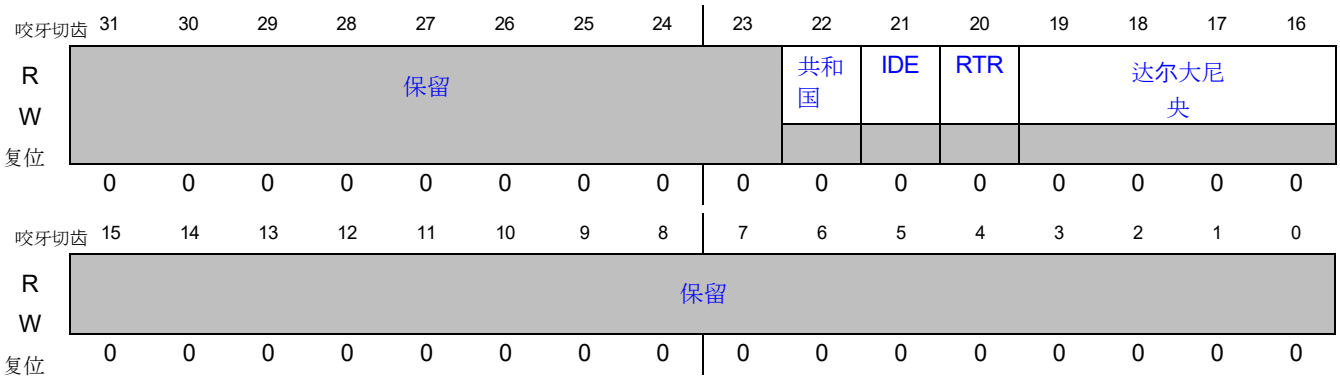
C/S寄存器位于WMB0的0xB40，WMB1的0xB50，WMB2的0xB60和WMB3的0xB70。

**注意事项**

这个寄存器不是每个模块实例都使用的。

- 适用于FlexCAN0。
- 不适用于FlexCAN1、FlexCAN2。

### 50.4.2.29.3 图示




### 50.4.2.29.4 领域

| 场地 | 职能 |
|---|---|
| 31-23<br>- | 保留 |
| 22<br>共和国 | 替代性远程请求<br>这个位可以收到隐性或显性。 |
| 21<br>IDE | ID扩展位<br>该位确定帧格式是标准还是扩展 0b - 帧格式是标准的<br>1b - 帧格式被扩展 |
| 20<br>RTR | 远程传输请求位<br>该位确定该帧是否为远程帧。<br>0b - 帧是数据一（非远程） 1b - 帧<br>是远程一 |
| 19-16<br>达尔优 | 数据的长度，以字节为单位<br>这个4位字段是FlexCAN处于伪装网络模式时收到的Rx数据的长度（以字节为单位）。这个4位字段由FlexCAN模块写入，从收到的帧的DLC（数据长度代码）字段中复制出来。DLC字段表示哪些数据字节是有效的。 |
| 15-0<br>- | 保留 |

## 50.4.2.30 唤醒信息缓冲寄存器的ID（WMB0_ID - WMB3_ID)

### 50.4.2.30.1 偏移

对于a=0到3。

| 注册 | 偏移 |
| --- | --- |
| WMBa_ID | B44h + (a × 10h) |

### 50.4.2.30.2 职能

四个WMB中的每一个都包含一个寄存器，用于存储传入的Rx信息的ID信息。

**注意事项**

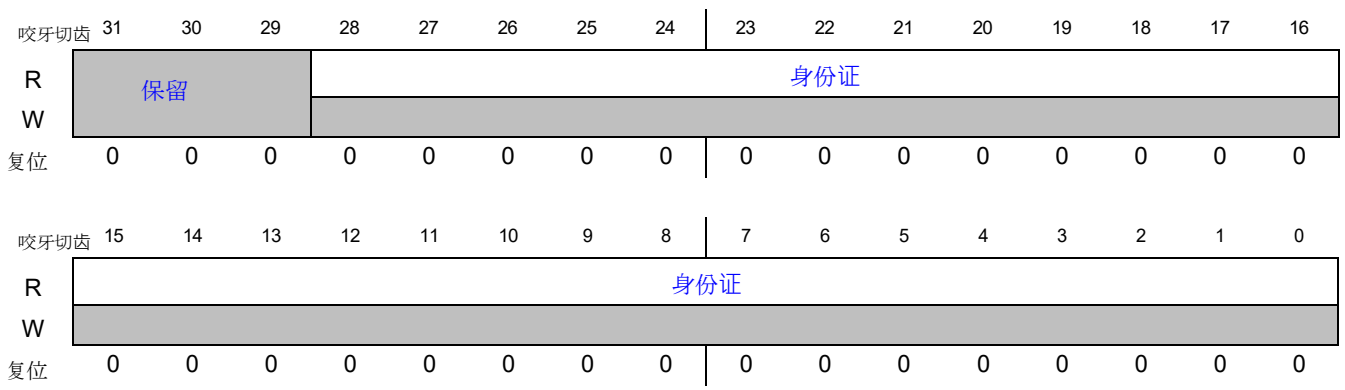
ID寄存器位于WMB0的0xB44，WMB1的0xB54，WMB2的0xB64和WMB3的0xB74。

**注意事项**

这个寄存器不是每个模块实例都使用的。

- 适用于FlexCAN0。
- 不适用于FlexCAN1、FlexCAN2。

### 50.4.2.30.3 图示




### 50.4.2.30.4 领域

| 场地 | 职能 |
| --- | --- |
| 31-29<br>- | 保留 |

| | |
|---|---|
| 28-0<br>身份证 | 在假装联网模式下收到的ID<br>该寄存器存储扩展帧格式的29位（考虑ID[28:0]字段），或标准帧格式的11位（仅考虑ID[28:18]字段，ID[17:0]范围内的其余位没有意义）。 |

## 50.4.2.31 唤醒信息缓冲寄存器的数据0-3（WMB0_D03 - WMB3_D03)

### 50.4.2.31.1 偏移

对于a=0到3。

| 注册 | 偏移 |
|---|---|
| WMBa_D03 | B48h + (a × 10h) |

### 50.4.2.31.2 职能

四个WMB中的每一个都包含一个寄存器，用于存储传入的Rx消息的有效载荷信息的数据字节0到3。当传入的匹配信息是一个远程帧（RTR=1）或一个DLC=0的数据帧时，该寄存器的内容被清零。

**注意事项**

数据0-3寄存器位于WMB0的0xB48，WMB1的0xB58，WMB2的0xB68和WMB3的0xB78。

**注意事项**

这个寄存器不是每个模块实例都使用的。

- 适用于FlexCAN0。
- 不适用于FlexCAN1、FlexCAN2。

### 50.4.2.31.3 图示

| 咬牙切齿 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 数据_字节_0 | | | | | | | | 数据_byte_1 | | | | | | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 咬牙切齿 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 数据_字节_2 | | | | | | | | 数据_byte_3 | | | | | | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

复位

#### 50.4.2.31.4 领域

| 场地 | 职能 |
| --- | --- |
| 31-24<br>数据_字节_0 | 在伪装网络模式下，与数据字节0相对应的收到的有效载荷 |
| 23-16<br>数据_byte_1 | 在伪装网络模式下与数据字节1相对应的接收有效载荷 |
| 15-8<br>数据_字节_2 | 在伪装网络模式下，接收到与数据字节2相对应的有效载荷 |
| 7-0<br>数据_byte_3 | 在伪装网络模式下，与数据字节3对应的接收有效载荷 |

### 50.4.2.32 唤醒信息缓冲区寄存器数据4-7（WMB0_D47 - WMB3_D47）。

#### 50.4.2.32.1 偏移

对于a=0到3。

| 注册 | 偏移 |
| --- | --- |
| WMBa_D47 | B4Ch + (a × 10h) |

#### 50.4.2.32.2 职能

四个WMB中的每一个都包含一个寄存器，用于存储传入的Rx信息的有效载荷信息的数据字节4到7。当传入的匹配信息是一个远程帧（RTR=1）或一个DLC=0的数据帧时，该寄存器的内容被清零。

**注意事项**

数据4-
7寄存器位于WMB0的0xB4C，WMB1的0xB5C，WMB2的0x
B6C和WMB3的0xB7C。

### 注意事项

这个寄存器不是每个模块实例都使用的。

- 适用于FlexCAN0。
- 不适用于FlexCAN1、FlexCAN2。

### 50.4.2.32.3 图示

| 咬牙切齿 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 数据_byte_4 | | | | | | | | 数据_byte_5 | | | | | | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 咬牙切齿 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 数据_byte_6 | | | | | | | | 数据_byte_7 | | | | | | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

### 50.4.2.32.4 领域

| 场地 | 职能 |
|---|---|
| 31-24<br>数据_byte_4 | 在伪装网络模式下，接收到的对应于数据字节4的有效载荷 |
| 23-16<br>数据_byte_5 | 在伪装网络模式下，与数据字节5对应的接收有效载荷 |
| 15-8<br>数据_byte_6 | 在伪装网络模式下，接收到的对应于数据字节6的有效载荷 |
| 7-0<br>数据_byte_7 | 在伪装网络模式下，与数据字节7对应的接收有效载荷 |

## 50.4.2.33 CAN FD控制寄存器（FDCTRL）。

### 50.4.2.33.1 偏移

| 注册 | 偏移 |
|---|---|
| FDCTRL | C00h |

### 50.4.2.33.2 职能

该寄存器包含CAN

FD操作的控制位。它还定义了在RAM（内存块）的不同分区中分配的消息缓冲区的数据大小，如下表所述。

当选择8字节的有效载荷时。

- R0块分配了MB0到MB31。
- R1块分配了MB32到MB63。

当选择超过8字节的有效载荷时，一个块中的最大MB数量将受到限制，如下所述。

**表50-8.消息缓冲器的数量**

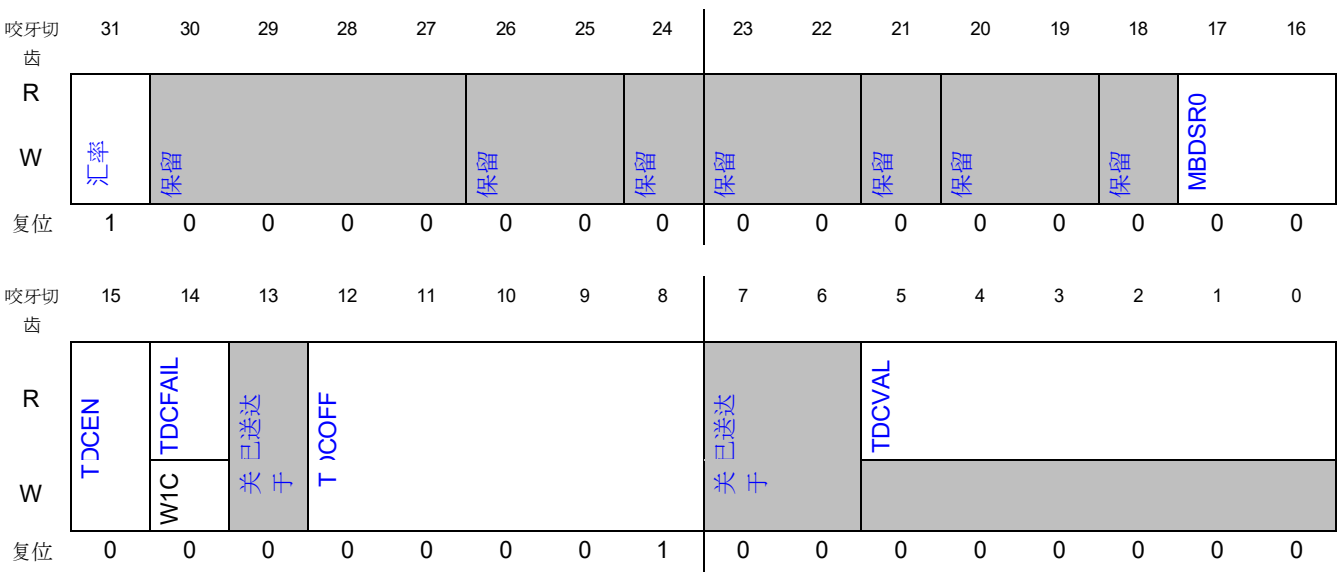
| 有效载荷大小 | 每个RAM块的最大消息缓冲区数量 |
|---|---|
| 8个字节 | 32 |
| 16个字节 | 21 |
| 32字节 | 12 |
| 64字节 | 7 |

**注意事项**

一个内存块正好适合32MB的8字节的有效载荷。对于其他有效载荷大小的选择，在一个块的最后一个MB和下一个块的开始之间可能存在空的内存。这个空的内存相当于小于1MB，不能使用。

该寄存器的内容不受软复位的影响。

### 50.4.2.33.3 图示



## 50.4.2.33.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>汇率 | 启用比特率开关<br>该位在Tx信息的数据阶段启用比特率开关（BRS位）的效果。<br>CPU可以在任何时候写入这个位。但是，只有当CAN总线处于等待总线空闲、总线空闲或总线关闭状态时，或者当前正在接收或传输的帧到达帧间空间时，它的作用才变为有效。<br>通过否定CAN_FDCTRL[FDRATE]位，CPU可以强制CAN FD报文中的所有比特以额定比特率进行传输，尽管Tx MBs的BRS位上的值是如此。<br>0b - 以额定速率传送一帧。Tx MB中的BRS位没有影响。<br>1b - 如果Tx MB中的BRS位是隐性的，则用比特率切换传输一帧。 |
| 30-27<br>- | 保留 |
| 26-25<br>- | 保留 |
| 24<br>- | 保留 |
| 23-22<br>- | 保留 |
| 21<br>- | 保留 |
| 20-19<br>- | 保留 |
| 18<br>- | 保留 |
| 17-16<br>MBDSR0 | 区域0的消息缓冲区数据大小<br>这个2位的字段选择在RAM中分配的消息缓冲区R0区域的数据大小（8、16、32或64字节）。<br>它只能在冻结模式下写入。<br>00b - 每个消息缓冲区选择8个字节。01b - 每个消息缓冲区选择16字节。10b - 每个消息缓冲区选择32字节。11b - 每个消息缓冲区选择64字节。 |
| 15<br>TDCEN | 收发器延时补偿启用<br>这个位可以用来启用和禁用TDC功能。它只能在冻结模式下被写入。<br>**注意：**当启用回环模式时，必须禁用TDC（参见CAN_CTRL1[LPB]寄存器）。<br>0b - TDC被禁用 1b - TDC被启用 |

| | |
|---|---|
| 14<br>TDCFAIL | 收发器延时补偿失败<br>该位表示收发器延迟补偿（TDC）机制超出范围，无法补偿收发器的环路延迟并成功比较延迟的接收位和传输位（见收发器延迟补偿。TDCFAIL在FlexCAN第一次检测到超出范围的情况时设置。CPU需要写1来清除它。<br>0b - 测量的回路延迟在范围内。 |

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| | 1b - 测量的回路延迟超出范围。 |
| 13<br>- | 保留 |
| 12-8<br>贸发局 | 收发器延时补偿偏移<br>该位字段包含要添加到被测收发器环路延迟中的偏移值，以便在比特率切换激活时定义延迟比较点的位置。关于如何进行环路延迟测量的更多细节，请参见收发器延迟补偿。<br>TDCOFF只能在Freeze模式下被写入。它的值可以在协议引擎（PE）时钟周期（CANCLK，更多细节见协议计时）中定义，并且必须选择比数据比特率中的CAN比特持续时间小的值，以便正确操作。<br>**注意：**不建议使用等于零的TDCOFF。 |
| 7-6<br>- | 保留 |
| 5-0<br>TDCVAL | 收发器延时补偿值<br>该寄存器包含了从传输的EDL到R0转换边缘到各自的接收边缘所测得的收发器环路延迟值，再加上CAN_FDCTRL寄存器中指定的TDCOFF值。这个值是协议引擎（PE）时钟周期（CANCLK）的整数倍。<br>关于如何进行环路延迟测量的更多细节，请参阅协议计时。 |

## 50.4.2.34 CAN FD Bit Timing Register (FDCBT)

### 50.4.2.34.1 偏移

| 注册 | 偏移 |
| --- | --- |
| FDCBT | C04h |

### 50.4.2.34.2 职能

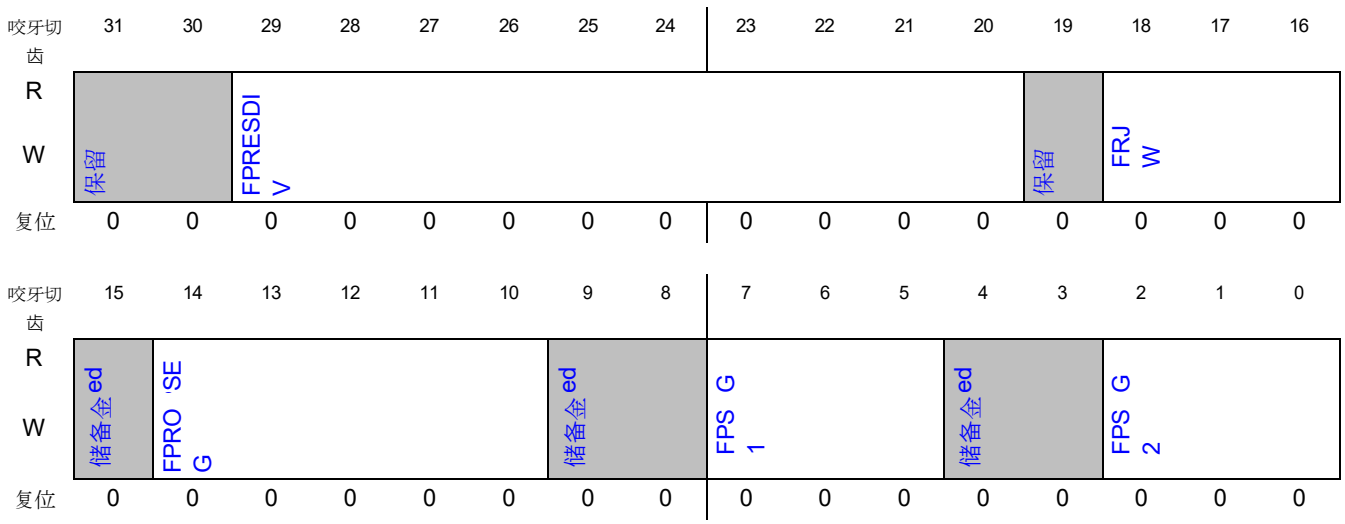
该寄存器存储了CAN_FDCTRL[FDRATE]被设置时，在CAN FD报文的数据阶段使用的CAN位定时变量，与CAN FD规范兼容。FPRESDIV、FPROPSEG、FPSEG1、FPSEG2和FRJW用于定义时间量子持续时间、每个CAN位的时间量子数以及设置了BRS位的CAN FD消息的数据位速率部分的采样点位置。

该寄存器的内容不受软复位的影响。

**注意事项**

快速传播段（FPROPSEG）和快速相位段1（FPSEG1）之和必须至少是两个时间量子。

### 50.4.2.34.3 图示




### 50.4.2.34.4 领域

| 场地 | 职能 |
|---|---|
| 31-30<br>- | 保留 |
| 29-20<br>辽宁省 | 快速预分频器划分系数<br>这个10位字段定义了设置了BRS位的CAN FD报文的数据比特率部分中PE时钟频率和串行时钟（Sclock）频率之间的比率。<br>Sclock周期定义了数据比特率的CAN FD协议的时间量子。这个字段只能在Freeze模式下写入，因为它在其他模式下被硬件屏蔽了。<br>Sclock频率=PE时钟频率/（FPRESDIV + 1）。<br>**注意：**为了在处理FD帧时减少错误，请为FPRESDIV和PRESDIV使用相同的值（在CAN_CBT或CAN_CTRL1中）。更多细节请参考CAN FD帧部分的第一个说明。 |
| 19<br>- | 保留 |

| | |
|---|---|
| 18-16<br>FRJW | 快速重新同步跳跃宽度<br>这个3位字段定义了在设置了BRS位的CAN FD报文的数据比特率部分中，一个比特时间可以通过一次重新同步来改变的最大时间量子数。<br>一个时间量子等于Sclock周期。这个字段只能在Freeze模式下写入，因为它在其他模式下被硬件阻断。<br>重新同步跳转宽度=FSJW+1。 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| 15<br>- | 保留 |
| 14-10<br>基金会 | 快速传播段<br>这个5位字段定义了在设置了BRS位的CAN FD报文的数据比特率部分的比特时间中的传播段的长度。这个字段只能在Freeze模式下写入，因为它在其他模式下被硬件阻断。<br>传播段时间 = FPROPSEG × 时间-量子。时间-量子=一个Sclock周期。 |
| 9-8<br>- | 保留 |
| 7-5<br>FPSEG1 | 快速阶段段1<br>这个3位字段定义了在设置了BRS位的CAN FD报文的数据比特率部分的比特时间中的阶段段1的长度。这个字段只能在Freeze模式下写入，因为它在其他模式下被硬件屏蔽了。<br>相位段1 = (FPSEG1 + 1) × 时间-量子。时间-量子=一个Sclock周期。 |
| 4-3<br>- | 保留 |
| 2-0<br>FPSEG2 | 快速阶段第2段<br>这个3位字段定义了设置了BRS位的CAN FD报文的数据比特率部分的第2相段的长度。这个字段只能在Freeze模式下写入，因为它在其他模式下被硬件屏蔽了。<br>相位段2 = (FPSEG2 + 1) × 时间-量子。时间-量子=一个Sclock周期。 |

## 50.4.2.35 CAN FD CRC寄存器（FDCRC）

### 50.4.2.35.1 偏移

| 注册 | 偏移 |
|---|---|
| FDCRC | C08h |

### 50.4.2.35.2 职能

该寄存器提供关于传输信息的CRC信息。

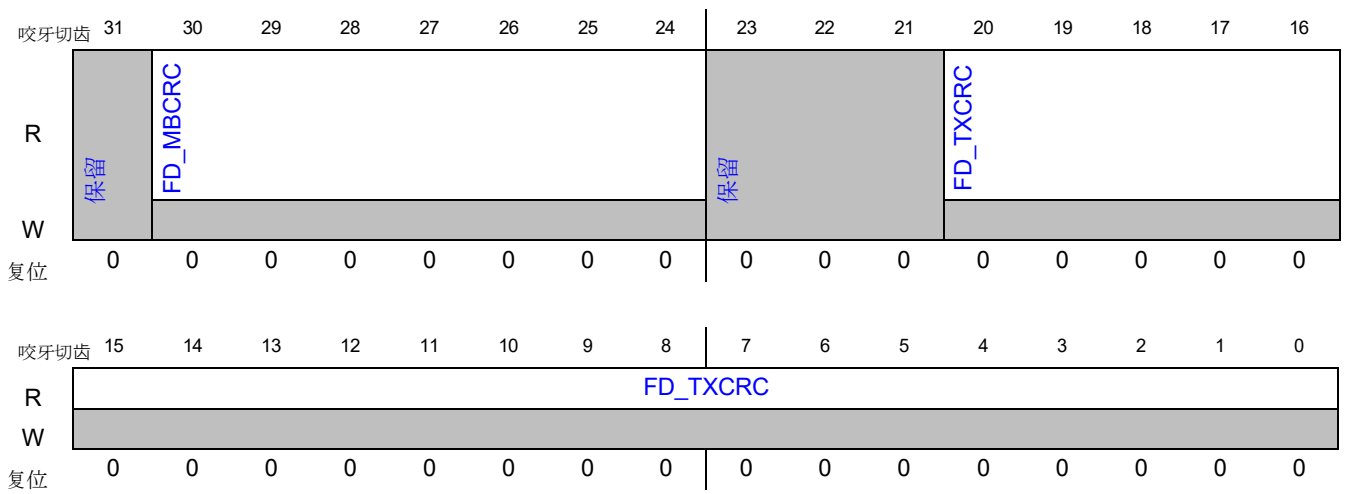
FlexCAN对不同的帧格式使用不同的CRC多项式，如下图所示。

CRC_15多项式用于所有CAN格式的帧。CRC_17多项式用于数据字段不超过16个字节的CAN FD格式的帧。CRC_21多项式用于数据字段长于16字节的CAN FD格式的帧。下面显示的每个多项式的结果是汉明距离为6。
该寄存器在Tx中断标志被宣称的同时被更新。

CRC_15 = 0xC599: (x15 +x14 +x10 +x8 +x7 +x4 +x3+1)

CRC_17 = 0x3685B: (x17+x16 +x14 +x13 +x11 +x6 +x4 +x3 +x1+1) CRC_21 = 0x302899:(x21 +x20 +x13 +x11 +x7 +x4 +x3+1)

### 50.4.2.35.3 图示




### 50.4.2.35.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>- | 保留 |
| 30-24<br>FD_MBCRC | FD_TXCRC的CRC邮箱号码<br>该字段表示与FD_TXCRC字段中的值相对应的邮箱号码，包括FD和非FD帧。<br>它报告的信息与CAN_CRCR寄存器中的MBCRC位域相同。 |
| 23-21<br>- | 保留 |

| | |
|---|---|
| 20-0 | 扩展传输的CRC值 |

| 场地 | 职能 |
|---|---|
| FD_TXCRC | 这个21位的字段包含了对最近传输的信息计算的CRC值。不同的CRC多项式用于不同的帧格式。一个15位的多项式，CRC_15，被用于所有CAN格式的帧。第二个17位的多项式，CRC_17，用于数据字段长达16个字节的CAN FD格式的帧。第三个21位的多项式，CRC_21，用于数据字段长于16字节的CAN FD格式的帧。<br>对于CRC_15和CRC_17，6个最重要的位和4个最重要的位分别报告为零。<br>对于CRC_15，该寄存器的内容与CRC寄存器相同。 |

## 50.4.3 消息缓冲区结构

FlexCAN模块使用的消息缓冲区结构如下图所示。CAN规范（2.0版B部分）中使用的扩展（29位标识符）和标准（11位标识符）帧都有体现。每个单独的MB由16、24、40或72个字节组成，这取决于分配给信息有效载荷的数据字节数量。分别是8、16、32或64个数据字节。

从0x80到0x27F的内存区域被邮箱使用。当启用CAN FD时，每个MB的确切地址取决于其有效载荷的大小。更详细的信息请参见用于CAN FD的FlexCAN内存分区。

**表50-9.消息缓冲区结构--以64字节的有效载荷为例**

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| EDL | BRS | ESI | | 代码 | | 共和国 | IDE | RTR | 达尔大尼央 | 时间戳 | |
| PRIO | | | 身份证(标准/扩展) | | | | | | 身份证 (扩展) | | |
| 数据字节0 | | | | 数据字节1 | | | | | | 数据字节2 | 数据字节3 |
| 数据字节4 | | | | 数据字节5 | | | | | | 数据字节6 | 数据字节7 |
| 数据字节8 | | | | 数据字节9 | | | | | | 数据字节10 | 数据字节11 |
| 数据字节12 | | | | 数据字节13 | | | | | | 数据字节14 | 数据字节15 |
| 数据字节16 | | | | 数据字节17 | | | | | | 数据字节18 | 数据字节19 |
| 数据字节20 | | | | 数据字节21 | | | | | | 数据字节22 | 数据字节23 |
| 数据字节24 | | | | 数据字节25 | | | | | | 数据字节26 | 数据字节27 |
| 数据字节 28 | | | | 数据字节 29 | | | | | | 数据字节30 | 数据字节31 |
| 数据字节32 | | | | 数据字节33 | | | | | | 数据字节34 | 数据字节35 |
| 数据字节36 | | | | 数据字节 37 | | | | | | 数据字节38 | 数据字节39 |
| 数据字节40 | | | | 数据字节41 | | | | | | 数据字节42 | 数据字节43 |

| | | | |
|---|---|---|---|
| 数据字节44 | 数据字节 45 | 数据字节46 | 数据字节47 |
| 数据字节48 | 数据字节49 | 数据字节50 | 数据字节51 |

*表继续在下一页...*

**表50-9.消息缓冲区结构--以64字节的有效载荷为例（续）**

| | | | | |
|---|---|---|---|---|
| 0x3C | 数据字节 52 | 数据字节53 | 数据字节54 | 数据字节55 |
| 0x40 | 数据字节56 | 数据字节 57 | 数据字节 58 | 数据字节59 |
| 0x44 | 数据字节60 | 数据字节61 | 数据字节62 | 数据字节63 |
| | = 未实现或保留 | | | |

**EDL** - 扩展数据长度

这个位区分了CAN格式和CAN FD格式的帧。对于配置为RANSWER的代码字段为0b1010的消息缓冲器，EDL位必须不被设置（见下表）。

**BRS** - 比特率开关

该位定义了在CAN FD格式帧内是否切换了比特率。

**ESI** - 错误状态指示器

该位表示发送节点是错误主动还是错误被动。

**CODE** - 消息缓冲区代码

这个4位字段可以被CPU和FlexCAN模块本身访问（读或写），作为消息缓冲区匹配和仲裁过程的一部分。编码见表50-10和表50-11。其他信息见功能描述。

**表50-10.Rx缓冲区的消息缓冲区代码**

| 代码<br>描述 | 接收新帧前的Rx代码 | SRV1 | 成功接收后的Rx代码2 | RRS3 | 评论 |
|---|---|---|---|---|---|
| 0b0000: 不活跃 - MB是不活跃的。 | 不活跃 | - | - | - | MB不参与匹配过程。 |
| 0b0100: EMPTY - MB是活动的和空的。 | 空白 | - | 全文 | - | 当成功接收到一个帧时（经过Move-in）过程），CODE字段会自动更新为FULL。 |
| 0b0010:满载 - MB已满。 | 全文 | 是 | 全文 | - | 读取C/S字，然后解锁MB（SRV）的行为并没有 |

*表继续在下一页...*

表50-10.Rx缓冲区的消息缓冲区代码（续）

| 代码<br>描述 | 接收新帧前的Rx代码 | SRV1 | 成功接收后的Rx代码2 | RRS3 | 评论 |
|---|---|---|---|---|---|
| | | | | | 使代码返回到EMPTY。它仍然是FULL。如果在MB被维修后，一个新的框架被移到MB上，代码仍然是FULL。与FULL代码有关的匹配细节，见匹配过程。 |
| | | 没有 | 超额完成任务 | - | 如果MB是满的，并且在CPU服务之前有一个新的帧被移到这个MB上，CODE字段会自动更新为OVERRUN。参见匹配过程中，关于超限行为的细节。 |
| 0b0110: OVERRUN - MB是被覆盖成一个完整的缓冲区。 | 超额完成任务 | 是 | 全文 | - | 如果CODE字段显示OVERRUN，并且CPU已经为MB提供了服务，当一个新的帧被移到MB上时，代码会返回到FULL。 |
| | | 没有 | 超额完成任务 | - | 如果CODE字段已经显示OVERRUN，并且另一个新的框架必须被移动，MB将再次被覆盖，代码将保持OVERRUN。关于OVERRUN行为的细节，请看匹配过程。 |

| | | | | | |
|---|---|---|---|---|---|
| 0b1010: RANSWER4 - A 帧被配置为识别一个 | RANSWER | - | TANSWER(0b1110) ) | 0 | 一个远程应答被配置为识别收到的远程请求帧。在这之后 |

*表继续在下一页...*

## 表50-10.Rx缓冲区的消息缓冲区代码（续）

| 代码<br>描述 | 接收新帧前的Rx代码 | SRV1 | 成功接收后的Rx代码2 | RRS3 | 评论 |
|---|---|---|---|---|---|
| 远程请求帧并发送一个响应帧作为回报。[5] | | | | | 一个MB被设定为发送响应帧。代码会自动改为TANSWER（0b1110）。见匹配过程的细节。如果CAN_CTRL2[RRS]为负值，每当收到相同ID的远程请求帧，就发送一个响应帧。 |
| | | - | - | 1 | 该代码在匹配和仲裁过程中被忽略。详见匹配过程。 |
| code[0]=1：忙 - FlexCAN正在更新MB的内容。CPU不能访问该MB。 | 忙6 | - | 全文 | - | 表示MB正在被更新。它将自动被否定，不影响下一个CODE。 |
| | | - | 超额完成任务 | - | |

1. SRV：维修过的MB。MB被读取并通过读取TIMER或其他MB解锁。
2. 一个帧被认为是一个成功的接收后，帧要移动到MB（移入过程）。详见Move-in。
3. 远程请求存储位，详见《控制2寄存器（CAN_CTRL2）》。
4. 代码0b1010不被认为是Tx，具有此代码的MB不应该被中止。
5. 代码0b1010必须用于以CAN FD格式配置的消息缓冲器，其EDL位被设置。
6. 请注意，对于Tx MBs，除了在MCR寄存器中设置AEN位时，BUSY位在读取时应被忽略。如果该位被引爆，相应的MB不参与匹配过程。

## 表50-11.用于Tx缓冲区的消息缓冲区代码

| 代码说明 | 在发送帧之前的发送代码 | MB RTR | 成功传输后的Tx代码 | 评论 |
|---|---|---|---|---|
| 0b1000: 不活跃 - MB不活跃 | 不活跃 | - | - | MB不参与仲裁程序。 |
| 0b1001: ABORT - MB被流产 | 弃权 | - | - | MB不参与仲裁程序。 |

*表继续在下一页...*

表50-11.Tx缓冲区的消息缓冲区代码（续）

| 代码说明 | 在发送帧之前的发送代码 | MB RTR | 成功传输后的Tx代码 | 评论 |
|---|---|---|---|---|
| 0b1100: 数据 - MB是a Tx数据帧（MB RTR必须为0）。 | 数据 | 0 | 不活跃 | 无条件地传输一次数据帧。传输后，MB自动返回到非活动状态。 |
| 0b1100:REMOTE - MB是一个Tx远程请求帧（MB RTR必须为1）。 | 遥控 | 1 | 空白 | 无条件地传送一次远程请求帧。<br>传输后，该MB自动成为具有相同ID的Rx空MB。 |
| 0b1110: TANSWER——。<br>MB是一个来自传入的远程请求帧的Tx响应帧 | TANSWER | - | RANSWER | 这是一个中间代码，由于与远程请求帧的匹配，CHI会自动写入MB中。<br>远程响应帧将被无条件地传送一次，然后代码将自动返回到RANSWER（0b1010）。<br>CPU也可以写这个代码，效果相同。远程响应帧可以是一个数据帧，也可以是另一个远程请求帧，这取决于RTR位的值。详见匹配过程和仲裁过程。 |

**SRR** - 替代性远程请求

固定的隐性位，只在扩展格式中使用。它必须由用户设置为1，用于传输（Tx缓冲区），并将与Rx接收缓冲区在CAN总线上收到的值一起存储。它可以被接收为隐性或显性。如果FlexCAN接收到该位为显性，那么它将被解释为一个仲裁损失。

1 = 隐性值在扩展格式帧中的传输是强制性的 0 =

主导值在扩展格式帧中的传输不是有效值 **IDE** - ID扩展位

该字段确定帧格式是标准还是扩展。1 = 帧格式是扩展的

0 = 帧格式是标准的

**RTR** - 远程传输请求

这个位会影响到远程帧的行为，是接收滤波器的一部分。更多的细节参见表50-10，表50-11，以及控制2寄存器（CAN_CTRL2）中对RRS位的描述。

如果FlexCAN以'1'（隐性）方式传输该位，并以'0'（显性）方式接收该位，则被解释为仲裁丢失。如果该位以'0'（显性）传输，那么如果收到的是'1'（隐性），FlexCAN模块将其视为位错误。如果收到的值与传输的值一致，则认为是成功的位传输。

1 =
如果MB是Tx，表示当前MB可能有一个远程请求帧要传输。如果MB是Rx，那么传入的远程请求帧可以被存储。

0 = 表示当前MB有一个数据帧要传输。在Rx
MB中，它可能被考虑在匹配过程中。

**注意事项**

当配置CAN FD帧时，RTR位必须是负的。

**DLC** - 数据的长度，以字节为单位

这个4位字段是Rx或Tx数据的长度（以字节为单位），它位于MB空间的偏移量0x8至0xF（见表50-9）。在接收时，该字段由FlexCAN模块写入，从接收帧的DLC（数据长度代码）字段复制。在传输时，该字段由CPU写入，与要传输的帧的DLC字段值相对应。当RTR=1时，要传输的帧是一个远程帧，不包括数据字段，与DLC字段无关（见表50-12）。

**时间戳** - 自由运行的计数器时间戳

这个16位的字段是自由运行定时器的副本，在标识符字段的开头出现在CAN总线上时为Tx和Rx帧捕获。

**PRIO**--地方优先

这个3位字段只有在CAN_MCR中LPRIO_EN位被设置时才会使用，而且它只对Tx邮箱有意义。这些位不被传输。它们被附加到常规ID上，以定义传输优先级。参见仲裁过程。

## ID - 帧标识符

在标准帧格式中，只有11个最重要的位（28至18）在接收和发送情况下用于帧识别。18个最小有效位被忽略。在扩展帧格式中，所有位都用于接收和发送情况下的帧识别。

**DATA BYTE 0 to 63** - 数据字段

一个数据帧最多可以使用六十四个字节，这取决于为信息缓冲区选择的有效载荷的大小。

对于Rx帧，数据在从CAN总线上接收到时被存储。DATA BYTE（*n*）只有在*n*小于DLC时才有效，如下表所示。

**表50-12.DATA BYTEs的有效性**

| 达尔优 | 有效的DATA BYTEs |
|---|---|
| 0 | 没有 |
| 1 | DATA BYTE 0 |
| 2 | DATA BYTE 0至1 |
| 3 | DATA BYTE 0至2 |
| 4 | DATA BYTE 0至3 |
| 5 | 数据字节0至4 |
| 6 | 数据字节0至5 |
| 7 | 数据字节0至6 |
| 8 | DATA BYTE 0至7 |
| 9 | 数据字节0至11 |
| 10 | DATA BYTE 0至15 |
| 11 | DATA BYTE 0至19 |
| 12 | DATA BYTE 0至23 |
| 13 | 数据字节0至31 |
| 14 | DATA BYTE 0至47 |
| 15 | DATA BYTE 0至63 |

## 50.4.4 用于CAN FD的FlexCAN内存分区

当CAN FD被启用时，FlexCAN RAM可以被划分为512字节的块。每个块可以容纳一定数量的消息缓冲器，这取决于CAN_FDCTRL[MBDSRn]位域提供的配置，如下表所示。

表50-13.RAM分区

| RAM块 | 8字节的MB数量（默认范围）。 | CAN_FDCTRL寄存器中的尺寸控制位域 | 不同大小的MB数量，每个区块 |
|---|---|---|---|
| 0 | 0至31 | MBDSR0 | MBDSR0=00，32个MBs，有8个字节有效载荷<br>MBDSR0=01，21 MBs，16字节有效载荷<br>MBDSR0=10，12MBs，32个字节有效载荷<br>MBDSR0=11，7个MBs有64个字节有效载荷 |

当在一些或所有的RAM块中配置16、32或64字节的有效载荷大小时，MB的总数和其各自的数量顺序可能与默认配置的8字节不同。例如，假设Block0被配置为8字节的有效载荷，Block1被配置为16字节，那么下表中的信息缓冲区将如何被安排到RAM中。

表50-14.RAM分区的例子

| RAM块 | 有效载荷大小 | RAM块中的MB数量 | 信息缓冲区范围 |
|---|---|---|---|
| 0 | CAN_FDCTRL[MBDSR0]=00，8字节的有效载荷 | 32 | 0至31 |

## 50.4.5 FlexCAN消息缓冲区内存图

FlexCAN内存缓冲区在内存中的分配如下表所示。

表50-15.8字节的消息缓冲区

| 地址偏移（十六进制） | MBDSR=b00<br>8字节的有效载荷 |
|---|---|
| 0080 | MB0 |
| 0090 | MB1 |
| 00A0 | MB2 |
| 00B0 | MB3 |
| 00C0 | MB4 |
| 00D0 | MB5 |

| | |
|---|---|
| 00E0 | MB6 |
| 00F0 | MB7 |
| 0100 | MB8 |

*表继续在下一页...*

表50-15.8字节的消息缓冲区（续）

| 地址偏移（十六进制） | MBDSR=b00<br>8字节的有效载荷 |
|---|---|
| 0110 | MB9 |
| 0120 | MB10 |
| 0130 | MB11 |
| 0140 | MB12 |
| 0150 | MB13 |
| 0160 | MB14 |
| 0170 | MB15 |
| 0180 | MB16 |
| 0190 | MB17 |
| 01A0 | MB18 |
| 01B0 | MB19 |
| 01C0 | MB20 |
| 01D0 | MB21 |
| 01E0 | MB22 |
| 01F0 | MB23 |
| 0200 | MB24 |
| 0210 | MB25 |
| 0220 | MB26 |
| 0230 | MB27 |
| 0240 | MB28 |
| 0250 | MB29 |
| 0260 | MB30 |
| 0270 | MB31 |

表50-16.16字节的消息缓冲区

| 地址偏移（十六进制） | MBDSR=b01<br>16字节的有效载荷 |
|---|---|
| 0080 | MB0 |
| 0098 | MB1 |
| 00B0 | MB2 |
| 00C8 | MB3 |
| 00E0 | MB4 |
| 00F8 | MB5 |
| 0110 | MB6 |
| 0128 | MB7 |
| 0140 | MB8 |
| 0158 | MB9 |

*表继续在下一页...*

表50-15. 表50-16. 16字节的消息缓冲区（续）

| 地址偏移（十六进制） | MBDSR=b01<br>16字节的有效载荷 |
|---|---|
| 0170 | MB10 |
| 0188 | MB11 |
| 01A0 | MB12 |
| 01B8 | MB13 |
| 01D0 | MB14 |
| 01E8 | MB15 |
| 0200 | MB16 |
| 0218 | MB17 |
| 0230 | MB18 |
| 0248 | MB19 |
| 0260 | MB20 |

表50-17.32字节的消息缓冲区

| 地址偏移（十六进制） | MBDSR=b10<br>32字节的有效载荷 |
|---|---|
| 0080 | MB0 |
| 00A8 | MB1 |
| 00D0 | MB2 |
| 00F8 | MB3 |
| 0120 | MB4 |
| 0148 | MB5 |
| 0170 | MB6 |
| 0198 | MB7 |
| 01C0 | MB8 |
| 01E8 | MB9 |
| 0210 | MB10 |
| 0238 | MB11 |

表50-18.64字节的消息缓冲区

| 地址偏移（十六进制） | MBDSR=b11<br>64字节的有效载荷 |
|---|---|
| 0080 | MB0 |
| 00C8 | MB1 |
| 0110 | MB2 |
| 0158 | MB3 |

| | |
|---|---|
| 01A0 | MB4 |

*表继续在下一页...*

表50-18.64字节的消息缓冲区（续）

| 地址偏移（十六进制） | MBDSR=b11<br>64字节的有效载荷 |
|---|---|
| 01E8 | MB5 |
| 0230 | MB6 |

## 50.4.6 Rx FIFO结构

当CAN_MCR[RFEN]位被设置时，从0x80到0xDC的内存区域（通常被MBs 0-5占用）被接收FIFO引擎使用。

区域0x80-0x8C包含FIFO的输出，必须由CPU作为消息缓冲区来读取。这个输出包含已经收到但尚未读取的最古老的消息。区域0x90-0xDC是保留给FIFO引擎内部使用的。

根据CAN_CTRL2[RFFN]字段的设置，一个额外的内存区域从0xE0开始，可以扩展到0x2DC（通常由MBs 6-37占据），包含ID过滤表（可配置8到128个表元素），指定接受帧进入FIFO的过滤标准。

复位后，ID过滤表的灵活内存区域默认为0xE0，只延伸到0xFC，这相当于RFFN=0时的MBs 6至7，以便与以前版本的FlexCAN向后兼容。

以下是Rx FIFO的数据结构。

表50-19.Rx FIFO结构

| IDHIT | 共和国 | IDE | RTR | 达尔大尼央 | 时间戳 |
|---|---|---|---|---|---|
| | 身份证标准 | | | 身份证延期 | |
| 数据字节0 | 数据字节1 | | | 数据字节2 | 数据字节3 |
| 数据字节4 | 数据字节5 | | | 数据字节6 | 数据字节7 |
| 保留 | | | | | |
| 内径过滤表元件0 | | | | | |
| 内径过滤表元件1 | | | | | |

| |
|---|
| 内径过滤表元件2至125 |

*表继续在下一页...*

**表50-19.Rx FIFO结构（续）**

| | |
|---|---|
| 0x2D8 | 内径过滤表元件126 |
| 0x2DC | 内径过滤表元件 127 |
| | = 未实现或保留 |

每个ID过滤表元素占据整个32位字，根据CAN_MCR[IDAM]字段的设置，可以由一个、两个或四个识别器接受过滤器（IDAF）复合而成。下面的数字显示了IDAF的索引情况。

下表显示了ID表元素的三种不同格式。请注意，该表的所有元素必须具有相同的格式。更多信息见Rx FIFO。

**表50-20.ID表结构**

| 样式 | 31 | 30 | 29 ... 24 | 23 ... 16 | 15 | 14 | 13 ... 8 | 7 ... 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| A | RTR | IDE | RXIDA (标准=29-19，扩展=29-1) | | | | | | |
| B | RTR | IDE | 荣信达_0 (标准=29-19，扩展=29-16) | | RTR | IDE | 荣信达_1 (标准=13-3，扩展=13-0) | | |
| C | 荣信达_0 (中/外=31-24) | | 荣信达_1 (STD/EXT = 23-16) | | RXIDC_2 (std/ext = 15-8) | | | RXIDC_3 (std/ext = 7-0) | |
| | = 未实现或保留 | | | | | | | | |

**RTR** - 远程框架

该位指定如果远程帧符合目标ID，是否接受其进入FIFO。

1 = 可以接受远程帧，拒绝数据帧 0 = 拒绝远程帧，可以接受数据帧 **IDE** - 扩展帧

指定是否接受扩展帧或标准帧进入FIFO，如果它们符合目标ID。

1 = 可以接受扩展帧，拒绝标准帧 0 = 拒绝扩展帧，可以接受标准帧 **RXIDA** - Rx 帧标识符（格式 A）

指定一个ID，作为FIFO的接受标准。在标准帧格式中，只有11个最重要的位（29至19）用于帧识别。在扩展帧格式中，所有位都被使用。

**RXIDB_0, RXIDB_1** - Rx Frame Identifier（格式B）。

指定一个ID，作为FIFO的接受标准。在标准帧格式中，11个最重要的位（一个完整的标准ID）（29至19和13至3）用于帧识别。在扩展帧格式中，该字段的所有14位都与收到的ID的14个最有效位进行比较。

**RXIDC_0, RXIDC_1, RXIDC_2, RXIDC_3** - Rx Frame Identifier（格式C）。

指定一个ID，作为FIFO的接受标准。在标准和扩展帧格式中，该字段的所有8位都与收到的ID的8个最重要位进行比较。

**IDHIT** - 标识符接受度过滤命中指标

这个9位字段表示哪个标识符接受过滤器被Rx FIFO的输出中的接收信息击中。更多信息请参见Rx FIFO。

## 50.5 职能描述

FlexCAN模块是一个CAN协议引擎，它有一个非常灵活的邮箱系统来传输和接收CAN帧。该邮箱系统由一组消息缓冲器（MB）组成，用于存储配置和控制数据、时间戳、消息ID和数据（见消息缓冲器结构）。与前38个MB相对应的存储器可以被配置为支持具有强大的ID过滤机制的FIFO接收方案，能够根据ID表（最多128个扩展ID或256个标准ID或512个8位ID片）检查进入的帧，有多达32个ID过滤表元素的单独屏蔽寄存器。

对于经典的CAN帧，支持通过FIFO和邮箱同时接收。对于CAN FD帧，只支持通过邮箱接收。对于邮箱接收，一个匹配算法使得有可能只将接收到的帧存储到在其ID字段上有相同ID编程的MB中。一个掩码方案使得在MB上编程的ID与收到的CAN帧上的ID范围相匹配成为可能。对于传输，一个仲裁算法根据信息ID（可选择增加3个本地优先位）或MB的排序来决定要传输的MB的优先次序。

在继续进行功能描述之前，必须解释一个重要的概念。如果一个消息缓冲区能够参与匹配和仲裁过程，那么它在某一特定时间被称为"活动"。一个代码为0b0000的Rx MB是不活动的（参考表50-10）。同样地，一个代码为0b1000或0b1001的Tx MB也是不活动的（参考表50-11）。

FlexCAN模块也能接收和发送CAN FD格式的信息。消息缓冲区的大小是为了充分存储由CAN_FDCTRL寄存器中MBDSRn位字段选择的数据字节数量。对于给定数量的数据字节，可用的FD MBs的数量被描述为CAN_FDCTRL寄存器。另请参见CAN FD的FlexCAN内存分区。

### 50.5.1 传输过程

为了传输一个CAN帧，CPU必须通过执行以下程序为传输准备一个消息缓冲区。

1. 检查相应的中断位是否被设置，并清除它。
2. 如果MB处于激活状态（传输待定），将ABORT代码（0b1001）写入控制和状态字的CODE字段，请求中止传输。等待相应的IFLAG位通过轮询CAN_IFLAG寄存器或通过中断请求（如果被相应的IMASK位启用）而被断言。然后读回CODE字段以检查传输是否被中止或传送（见传输中止机制）。如果需要向后兼容（CAN_MCR[AEN]位被否定），只需向CODE字段写INACTIVE代码（0b1000）就可以使MB失活，但这时可以在不通知的情况下传输等待的帧（见邮箱失活）。
3. 写下ID字样。
4. 写入数据字节。
5. 写入控制和状态字的DLC、Control和CODE字段以激活MB。当CAN_MCR[FDEN]被设置时，还要写入EDL、BRS和ESI位。

当MB被激活时，它参与了仲裁过程，并最终根据其优先级进行传输。当选择传输的MB中存储的DLC值大于相应的MB有效载荷大小时，FlexCAN会以恒定的0xCC模式添加必要数量的字节，以完成预期的DLC。

在成功传输结束后，自由运行计时器的值被写入时间戳字段，控制和状态字中的CODE字段被更新，CAN_CRC和CAN_FDCRC寄存器都被更新，一个状态标志被设

置在

中断标志寄存器，如果相应的中断屏蔽寄存器位允许，则产生中断。传输后的新CODE字段取决于用于激活MB的代码（见表50-10和表50-11的消息缓冲器结构）。

当中止功能被启用时（CAN_MCR[AEN]被断言），在被配置为发送缓冲区的邮箱的中断标志被断言后，该邮箱被阻塞。因此，CPU不能更新它，直到中断标志被CPU否定。这意味着CPU在开始为新的传输或接收准备这个MB之前必须清除相应的IFLAG位。

### 50.5.2 仲裁程序

仲裁过程扫描邮箱，寻找持有要在下一次机会中发送的信息的Tx邮箱。这个邮箱被称为*仲裁赢家*。

扫描从最低号码的邮箱开始，向更高的邮箱运行。仲裁过程在以下事件中

被触发。

- 从CAN帧的CRC字段开始。起始点取决于CAN_CTRL2[TASD]字段的值。
- 在一个CAN帧的错误分隔符字段中。
- 在一个CAN帧的过载分界符字段中。
- 当赢家未被激活，且CAN总线仍未到达中间场的第一位时。
- 当有CPU写到赢家MB的C/S字时，而且CAN总线仍未到达中间字段的第一位。
- 当CHI处于空闲状态，CPU写到任何MB的C/S字时。
- 当FlexCAN退出总线关闭状态。
- 离开冻结模式或低功率模式时。

如果仲裁过程在CAN总线到达中间字段的第一位之前没有设法评估所有的邮箱，那么临时的仲裁赢家就会失效，FlexCAN将不会在下一次机会中竞争CAN总线。

仲裁过程根据CAN_CTRL1[LBUF]和CAN_MCR[LPRIOEN]位的设置，在扫描结束时在活动的Tx邮箱中选择赢家。

### 50.5.2.1 首先是编号最低的邮箱

如果CAN_CTRL1[LBUF]位被断言，找到的第一个（最低数量）活动Tx邮箱是仲裁赢家。当CAN_CTRL1[LBUF]被断言时，CAN_MCR[LPRIOEN]位没有影响。

### 50.5.2.2 优先级最高的邮箱优先

如果CAN_CTRL1[LBUF]位被否定，那么仲裁过程就会搜索具有最高优先级的活动Tx邮箱，这意味着当多个外部节点同时竞争总线时，该邮箱的帧在CAN总线上赢得仲裁的概率更高。

这种仲裁所考虑的比特序列被称为邮箱的*仲裁值*。最高优先级的Tx邮箱是所有Tx邮箱中仲裁值最低的一个。

如果两个或更多的邮箱有相等的仲裁值，数字最低的邮箱是仲裁赢家。

仲裁值的构成取决于CAN_MCR[LPRIOEN]位的设置。

#### 50.5.2.2.1 地方优先权失效

如果CAN_MCR[LPRIOEN]位被否定，那么仲裁值就会按照在CAN帧中传输的比特的准确顺序建立（见下表），这样一来，本地优先权就被禁用了。

**表50-21.禁用本地优先权时仲裁值的构成**

| 格式 | 信箱仲裁值（32位） | | | | |
|---|---|---|---|---|---|
| 标准 (IDE = 0) | 标准ID（11位） | RTR（1位） | 集成开发环境（1位） | - (18位) | - (1比特) |
| 扩展 (IDE = 1) | 扩展ID[28:18] (11位) | SRR（1位） | 集成开发环境（1位） | 扩展ID[17:0] (18位) | RTR（1位） |

#### 50.5.2.2.2 启用本地优先权

如果需要本地优先权，必须断言CAN_MCR[LPRIOEN]。在这种情况下，邮箱PRIO字段被包括在仲裁值的最左边（见下表）。

**表50-22.启用本地优先权时仲裁值的构成**

| 格式 | 信箱仲裁值（35位） | | | | | |
|---|---|---|---|---|---|---|
| 标准 (IDE = 0) | PRIO（3位） | 标准ID（11位） | RTR（1位） | 集成开发环境（1位） | - (18位) | - (1比特) |
| 扩展 (IDE = 1) | PRIO（3位） | 扩展ID[28:18]（11位）。 | SRR（1位） | 集成开发环境（1位） | 扩展ID[17:0] (18位) | RTR（1位） |

由于PRIO字段是仲裁值中最重要的部分
PRIO值低的邮箱比PRIO值高的邮箱有更高的优先权，而不考虑其仲裁值的其他部分。

请注意，PRIO字段不是CAN总线上帧的一部分。它的目的只是为了影响内部的仲裁过程。

#### 50.5.2.3 仲裁程序（续）

在找到仲裁赢家后，其内容被复制到一个隐藏的辅助MB，称为Tx串行信息缓冲区（Tx
SMB），其结构与普通MB相同，但用户不能访问。这个操作被称为移出，完成后，对相应MB的C/S字的写访问被阻止（如果CAN_MCR寄存器中的AEN位被断言）。在下列情况下，写访问被恢复。

- 在MB被传输并且相应的IFLAG位被CPU清零之后
- FlexCAN进入冻结模式或总线关闭模式
- FlexCAN失去了总线仲裁或在传输过程中出现了错误

在CAN总线上的第一个机会窗口，根据CAN协议规则，Tx SMB上的信息被传输。

仲裁程序可在以下情况下启动。

- 在Rx和Tx帧中，从CAN
  CRC字段到帧的结束。CAN_CTRL2[TASD]位的值可以被改变，以优化仲裁开始点。
- 在CAN总线关闭状态下，从TX_ERR_CNT=124到128。CAN_CTRL2[TASD]

位值可以改变，以优化仲裁起始点。

- 在BusIdle中CPU的C/S写入期间。第一次C/S写启动仲裁过程，在同一仲裁期间的第二次C/S写重新启动该过程。如果其他C/S写被执行，则Tx仲裁过程正在等待。如果仲裁过程结束后没有仲裁赢家，那么TX仲裁机开始一个新的仲裁过程。如果有一个悬而未决的仲裁和BusIdle状态开始，那么就会触发一个仲裁过程。在这种情况下，BusIdle中的第一个和第二个C/S写不会重新启动仲裁过程。有可能在WaitForBusIdle状态下没有足够的时间来完成仲裁，下一个状态是Idle。在这种情况下，扫描不会被打断，而是在BusIdle状态下完成。在这个仲裁过程中，C/S写不会导致仲裁重启。
- 仲裁赢家在有效的仲裁窗口期间停用。
- 在退出Freeze模式时（WaitForBusIdle状态的第一比特）。如果在WaitForBusIdle期间有重新同步，仲裁过程将被重新启动。

仲裁程序在以下情况下停止。

- 所有邮箱都被扫描了
- 在启用最低缓冲区功能的情况下，发现一个Tx活动邮箱。
- 在任何仲裁过程中，仲裁胜出者不活跃或中止。
- 没有足够的时间来完成Tx仲裁过程（例如，在帧结束前进行了停用）。在这种情况下，仲裁过程正在等待。
- 总线中的错误或过载标志
- 闲置状态下的低功率或冻结模式请求

仲裁被认为是待定的，如下所述。

- 不可能及时完成仲裁程序
- 如果在编号低于Tx仲裁指针的MB中进行写，则在仲裁期间进行C/S写。
- 如果没有正在进行的Tx仲裁程序，任何C/S都可以写。
- Rx Match刚刚更新了一个Rx Code to Tx Code
- 进入总线关闭状态

仲裁期间的C/S写入有以下影响。

- 如果在仲裁赢家中进行C/S写，则立即重新启动一个新的进程。
- 如果在一个编号高于Tx仲裁指针的MB中进行C/S写入，正在进行的仲裁过程将正常扫描这个MB。

### 50.5.3 接收过程

为了能够将CAN帧接收到邮箱中，CPU必须通过执行以下步骤为接收做准备。

1. 如果邮箱处于活动状态（无论是Tx还是Rx），请将邮箱灭活（见邮箱灭活），最好是采用安全灭活（见传输中止机制）。
2. 写入ID字样
3. 在控制和状态字的CODE字段中写入EMPTY代码（0b0100）以激活邮箱。EDL、BRS和ESI位不需要设置，它们会被收到的信息中各自的位域覆盖。

在MB被激活后，它将能够接收符合编程的过滤器的帧。在一次成功的接收结束后，邮箱会被*移入*过程更新（见移入），具体如下。

1. 收到的数据字段（对于经典的CAN报文格式，最多8个字节，对于CAN FD报文格式，最多64个字节）被存储。
2. 收到的标识符字段被存储。
3. 帧的标识符字段的第二位时的自由运行计时器的值被写入邮箱的时间戳字段中。
4. 接收的SRR、IDE、RTR、EDL、BRS、ESI和DLC字段被存储。
5. 控制和状态字中的CODE字段被更新（见表50-10和表50-11的消息缓冲区结构）。
6. 在中断标志寄存器中设置一个状态标志，如果相应的中断屏蔽寄存器位允许，则产生一个中断。

推荐使用以下程序为CPU服务（读取）邮箱中收到的帧。

1. 读取该邮箱的控制和状态字。
2. 检查BUSY位是否被去掉，表明邮箱被锁定。重复步骤1），同时它被断言。参见邮箱锁定机制。
3. 读取邮箱的内容。一旦邮箱现在被锁定，其内容将不会被FlexCAN迁入过程所修改。见Move-in。
4. 在IFLAG寄存器中确认适当的标志。

5. 阅读《免费运行计时器》。这是可选的，但建议尽快解锁邮箱并使其可用于接收。

CPU应该通过IFLAG寄存器中特定邮箱的状态标志位来轮询帧的接收，而不是通过该邮箱的CODE字段。轮询CODE字段是不起作用的，因为一旦收到帧，CPU对邮箱进行服务（通过读取C/S字和解锁邮箱），CODE字段将不会返回到EMPTY。如表50-10所解释的，它将保持满值。如果 CPU 试图通过写入 C/S 字来解决这一行为，在读取邮箱后，在没有事先*安全失活的*情况下，强制执行一个 EMPTY 代码，那么新收到的与该邮箱的过滤器相匹配的帧可能会丢失。

**注意事项**

综上所述：*永远不要通过直接读取邮箱的C/S字来做轮询。相反，要读取IFLAG寄存器。*

请注意，收到的帧的标识符字段总是存储在匹配的邮箱中，因此，如果匹配是由于屏蔽造成的，那么邮箱中的ID字段的内容可能会发生变化。当CAN_MCR[SRXDIS]位被断言时，FlexCAN不会将自己传输的帧存储在任何MB中，即使它包含一个匹配的Rx Mailbox，也不会产生中断标志或中断信号。否则，当CAN_MCR[SRXDIS]位被非置位时，如果存在一个匹配的Rx邮箱，FlexCAN可以接收自己传输的帧。

为了能够通过Rx FIFO接收CAN帧，CPU必须在Freeze模式下启用并配置Rx FIFO（参见Rx FIFO）。在收到Rx FIFO中断（参见CAN_IFLAG1寄存器中BUF5I位 "Frames available in Rx FIFO "位的描述）后，CPU应使用以下程序对收到的帧进行服务。

1. 读取控制和状态字（可选：只有在IDE和RTR位使用了屏蔽时才需要）。
2. 读取ID字段（可选：仅在使用掩码时需要）。
3. 读取数据字段
4. 读取CAN_RXFIR寄存器（可选）。
5. 通过向CAN_IFLAG1[BUF5I]位写1清除Rx FIFO中断中的可用帧（强制：释放MB并允许CPU读取下一个Rx FIFO条目）。

当CAN_MCR[DMA]被断言时，在接收到FIFO中的帧时，CAN_IFLAG1[BUF5I]会产生一个DMA请求，而不会产生一个CPU中断（参见DMA操作下的Rx FIFO）。Rx FIFO区域的CAN_IMASK1位不使用。

DMA控制器必须使用以下程序对接收的帧进行服务。

1. 读取控制和状态字（读取0x80地址，可选）。
2. 读取ID字段（读取0x84地址，可选）。
3. 读取所有数据字节（从0x88地址开始读取，可选）
4. 读取最后一个数据字节（读取0x8C地址是强制性的）。

### 50.5.4 匹配过程

匹配过程扫描MB存储器，寻找与从CAN总线上收到的ID相同的编程的Rx MBs。如果FIFO被启用，扫描的优先级可以在邮箱和FIFO过滤器之间选择。匹配从最低编号的消息缓冲器开始，向更高编号的消息缓冲器进行。如果在第一个结构中没有找到匹配的，那么随后就会扫描另一个结构。如果FIFO已经满了，匹配算法总是在FIFO区域外寻找匹配的MB。

在接收帧的过程中，它被存储在一个隐藏的辅助MB中，称为Rx串行消息缓冲器（Rx SMB）。

匹配过程的开始点取决于以下条件。

- 如果收到的帧是一个远程帧，起点是帧的CRC字段。
- 如果收到的帧是一个数据帧，DLC字段等于零，则起点是帧的CRC字段。
- 如果收到的帧是一个数据帧，其DLC字段不同于零，则起始点是帧的DATA字段。

如果在FIFO表中或某个邮箱中发现了匹配的ID，Rx SMB的内容就会被转移到FIFO中或通过移入过程转移到匹配的邮箱中。
如果检测到任何CAN协议错误，那么在接收结束时没有匹配结果被传输到FIFO或匹配的邮箱中。

匹配过程扫描Rx FIFO（如果启用）和活动的Rx邮箱（CODE为空、满、溢出或RANSWER）的所有

匹配元素，寻找与正在接收的Rx SMB的匹配元素的成功比较。

在CAN总线上的帧。Rx
SMB具有与邮箱相同的结构。与比较成功的匹配元素相关的接收结构（Rx FIFO或邮箱）是*匹配结构。匹配的赢家*是在扫描结束时在这些匹配的结构中选择的，取决于前面描述的条件。见下表。

**表50-23.匹配架构**

| 结构 | SMB[RTR] | CTRL2[RRS] | CTRL2[EACEN] | MB[IDE] | MB[RTR] | MB[ID1] | MB[CODE] |
|---|---|---|---|---|---|---|---|
| 邮箱 | 0 | - | 0 | cmp2 | 没有_cmp3 | cmp_msk4 | 空白或<br>满员或超员 |
| 邮箱 | 0 | - | 1 | cmp_msk | cmp_msk | cmp_msk | 空白或<br>满员或超员 |
| 邮箱 | 1 | 0 | - | 凸轮 | 不需要 | 凸轮 | RANSWER |
| 邮箱 | 1 | 1 | 0 | 凸轮 | 不需要 | cmp_msk | 空白或<br>满员或超员 |
| 邮箱 | 1 | 1 | 1 | cmp_msk | cmp_msk | cmp_msk | 空白或<br>满员或超员 |
| FIFO5 | - | - | - | cmp_msk | cmp_msk | cmp_msk | - |

1. 对于邮箱结构，如果 SMB[IDE] 被断言，ID 为 29 位（ID 标准 + ID 扩展）。如果SMB[IDE]被否定，ID只有11位（标准ID）。对于FIFO结构，ID取决于IDAM。
2. cmp。将Rx SMB内容与MB内容进行比较，不考虑掩码。
3. no_cmp。Rx SMB内容不与MB内容比较。
4. cmp_msk:将Rx SMB内容与MB内容进行比较，同时考虑到掩码。
5. 当IDAM为C型时，SMB[IDE]和SMB[RTR]不被考虑。

当满足以下任何一个条件时，一个接收结构就是*自由接收*。

- 信箱的CODE字段是空的
- 信箱的CODE字段是FULL或OVERRUN，并且已经被服务过了（C/S字被CPU读取并解锁，如信箱锁定机制中所述）。
- 信箱的CODE字段是FULL或OVERRUN，并进行灭活处理（见邮箱灭活）。
- Rx FIFO未满

信箱和Rx FIFO的扫描顺序是从编号最低的匹配元件到较高的元件。

信箱的匹配赢家搜索受到CAN_MCR[IRMQ]位的影响。如果它被否定，匹配赢家是第一个匹配的邮箱，不管它是否可以自由接收。如果它被断言，则根据下面的优先级来选择匹配赢家。

1. 第一个自由接收的匹配邮箱。
2. 最后一个非自由接收的匹配邮箱。

可以通过CAN_CTRL2[MRP]位选择邮箱和Rx FIFO之间扫描的优先级。

如果选择的优先级是Rx FIFO优先。

- 如果Rx FIFO是一个匹配的结构，并且是自由接收的，那么Rx FIFO就是匹配的赢家，而不考虑对邮箱的扫描。
- 否则（Rx FIFO不是一个匹配的结构或不是自由接收），那么匹配的赢家将在邮箱中搜索，如上所述。

如果选择的优先级是邮箱优先。

- 如果发现一个自由接收的匹配邮箱，那么无论扫描Rx FIFO，它都是匹配的赢家。
- 如果没有找到匹配的邮箱，那么在扫描Rx FIFO时就会搜索到匹配的赢家
- 如果上述两个条件都不满足，并且发现了一个非自由接收的匹配邮箱，那么匹配赢家的确定就以CAN_MCR[IRMQ]位为条件。
- - 如果CAN_MCR[IRMQ]位被否定，则匹配的赢家是第一个匹配的邮箱。
  - 如果CAN_MCR[IRMQ]位被断言，如果是自由接收匹配结构，则匹配赢家是Rx FIFO；否则，匹配赢家是最后一个非自由接收匹配的邮箱。

匹配的可能性摘要见下表。

**表50-24.匹配的可能性和产生的接收结构**

| 伦琴（RFEN） | IRMQ | MRP | 匹配的是MB | 在FIFO中匹配 | 接待结构 | 描述 |
|---|---|---|---|---|---|---|
| 没有FIFO，只有MB，匹配总是MB优先 | | | | | | |
| 0 | 0 | X1 | 无2 | -3 | 无 | 框体因不匹配而丢失 |
| 0 | 0 | X | 免费4 | - | FirstMB | |
| 0 | 1 | X | 无 | - | 无 | 框体因不匹配而丢失 |
| 0 | 1 | X | 免费 | - | 第一财经 | |
| 0 | 1 | X | 不自由 | - | 最后的MB | 超额完成任务 |
| 启用FIFO，FIFO中没有匹配，就像FIFO不存在一样。 | | | | | | |
| 1 | 0 | X | 无 | 无5 | 无 | 框体因不匹配而丢失 |
| 1 | 0 | X | 免费 | 无 | FirstMB | |

表继续在下一页...

表50-24.匹配的可能性和产生的接收结构（续）

| 伦琴（RFEN） | IRMQ | MRP | 匹配的是MB | 在FIFO中匹配 | 接待结构 | 描述 |
|---|---|---|---|---|---|---|
| 1 | 1 | X | 无 | 无 | 无 | 框体因不匹配而丢失 |
| 1 | 1 | X | 免费 | 无 | 第一财经 | |
| 1 | 1 | X | 不自由 | 无 | 最后的MB | 超额完成任务 |
| **启用FIFO，禁用队列** | | | | | | |
| 1 | 0 | 0 | X | 不满6 | FIFO | |
| 1 | 0 | 0 | 无 | 全7 | 无 | 帧因FIFO满而丢失（FIFO溢出）。 |
| 1 | 0 | 0 | 免费 | 全程 | FirstMB | |
| 1 | 0 | 0 | 不自由 | 全程 | FirstMB | |
| 1 | 0 | 1 | 无 | 未满 | FIFO | |
| 1 | 0 | 1 | 无 | 全程 | 无 | 帧因FIFO满而丢失（FIFO溢出）。 |
| 1 | 0 | 1 | 免费 | X | FirstMB | |
| 1 | 0 | 1 | 不自由 | X | 衬衫b | 超额完成任务 |
| **启用FIFO，启用队列** | | | | | | |
| 1 | 1 | 0 | X | 未满 | FIFO | |
| 1 | 1 | 0 | 无 | 全程 | 无 | 帧因FIFO满而丢失（FIFO溢出）。 |
| 1 | 1 | 0 | 免费 | 全程 | FirstMB | |
| 1 | 1 | 0 | 不自由 | 全程 | 最后一个Mb | 超额完成任务 |
| 1 | 1 | 1 | 无 | 未满 | FIFO | |
| 1 | 1 | 1 | 免费 | X | FirstMB | |
| 1 | 1 | 1 | 不自由 | 未满 | FIFO | |
| 1 | 1 | 1 | 不自由 | 全程 | 最后一个Mb | 超额完成任务 |

1. 这是一个不关心的条件。
2. 在MB中匹配的 "无 "意味着该帧没有匹配任何MB（自由接收或非自由接收）。
3. 这是一个被禁止的条件。
4. 匹配的MB "免费 "意味着该帧至少匹配了一个免费接收的MB，不管它是否匹配了非免费接收的MB。
5. 在FIFO中匹配的 "无 "意味着该帧没有匹配FIFO中的任何滤波器。这就像FIFO不存在一样（CAN_CTRL2[RFEN]=0）。
6. 在FIFO中匹配的 "NotFull "意味着该帧已经匹配了FIFO过滤器，并且有空槽可以接收它。
7. 在FIFO中匹配的 "满 "意味着该帧已经匹配了FIFO过滤器，但无法存储，因为它没有空槽来接收它。

如果在匹配过程中发生了非安全的邮箱失活（见邮箱失活），并且失活的邮箱是临时匹配赢家，那么临时匹配赢家就会失效。匹配元素的扫描不会被停止，也不会

重新启动，它继续正常进行。其后果是，当前的匹配过程就像不激活前比较的匹配元素不存在一样，因此可能会丢失一条信息。

例如，假设FIFO被禁用，IRMQ被启用，有两个具有相同ID的MB，FlexCAN开始接收具有该ID的信息。假设这些MB是阵列中的第二个和第五个。当第一条信息到达时，匹配算法在第2号MB中找到了第一个匹配。这个MB的代码是空的，所以信息被存储在那里。当第二条信息到达时，匹配算法再次找到2号MB，但它不是"自由接收"，所以它继续寻找，找到5号MB并将信息存储在那里。如果又有相同ID的信息到达，匹配算法发现没有匹配的MB是 "可自由接收
"的，所以它决定覆盖最后一个匹配的MB，也就是5号。在这样做的时候，它将MB的CODE字段设置为OVERRUN。

在一个以上的MB中匹配相同的ID的能力可以被利用来实现一个接收队列（除了全功能的FIFO之外），以使CPU有更多的时间为MB服务。通过对一个以上具有相同ID的MB进行编程，收到的信息被排入MB。CPU可以检查MB的时间戳字段以确定信息到达的顺序。

通过使用ID接受掩码，可以与一系列的ID进行匹配。FlexCAN支持每MB的单独屏蔽。参见Rx个体屏蔽寄存器（CAN_RXIMRx）的描述。在匹配算法中，如果一个屏蔽位被断言，那么相应的ID位将被比较。如果屏蔽位被否定，相应的ID位就是"不关心"。请注意，单个屏蔽寄存器是在RAM中实现的，所以它们在复位后不会被初始化。另外，它们只能在模块处于冻结模式时被编程，否则，它们会被硬件封锁。

FlexCAN还支持另一种屏蔽方案，只有四个屏蔽寄存器（CAN_RXFGMASK、CAN_RXMGMASK、CAN_RX14MASK 和
CAN_RX15MASK），以便向后兼容传统的应用。当CAN_MCR寄存器中的IRMQ位被否定时，这个备用的屏蔽方案被启用。

### 50.5.5 假装联网模式下的接收过程

假装网络模式在低功率模式（停止模式）下增加了特定的唤醒功能。当通过断言模块配置寄存器（CAN_MCR）中的PNET_EN位启用假装网络模式（PN）时，FlexCAN在低功耗模式下继续处理Rx
CAN消息，能够通过以下方式检测特定的唤醒消息

使用预先选择的匹配标准，根据ID和有效载荷目标值对它们进行过滤。唤醒功能对CAN FD格式的报文是不可用的。在假装联网模式下，CAN FD格式的报文被忽略。

PN寄存器位于0x0B00 - 0x0B7C的地址范围内，只有在Freeze模式下才能写入。这些寄存器用于在进入伪装网络模式前写入PN配置（包括控制值和目标值），以及在唤醒后返回到正常模式时读取唤醒标志和接收到的信息ID和数据。CPU必须等待CAN_MCR[LPMACK]被否定后才能对FlexCAN PN寄存器进行任何访问。

PN控制寄存器在假装网络控制1寄存器（CAN_CTRL1_PN）和假装网络控制2寄存器（CAN_CTRL2_PN）中描述。配置过滤标准的控制位字段是：。

- PLFS：有效载荷过滤选择。
- IDFS: ID过滤选择。
- FCS：过滤组合选择。PN的目标

值是。

- FLT_IDE：IDE目标值，用于按格式过滤传入的信息。(标准或扩展)。
- FLT_RTR: RTR目标值，用于按类型（数据或远程帧）过滤收到的信息。
- FLT_DLC_HI和FLT_DLC_LO：目标DLC范围，用于过滤传入信息的有效载荷部分的大小。
- FLT_ID1：用于过滤传入信息ID的ID目标值（等于、小于或等于、大于或等于、或ID范围的下限值）。
- FLT_ID2: ID目标值，作为ID范围的上限。
- PL1：用于过滤传入信息有效载荷的有效载荷目标值（等于、小于或等于、大于或等于、或有效载荷范围的下限值）。
- PL2。有效载荷目标值，用作有效载荷范围的上限。

IDE、RTR、ID和有效载荷过滤器有各自的掩码。这些掩码决定哪些位在平等比较中被考虑（某些掩码位置的 "1"），哪些位不被考虑（其他掩码位置的 "0"）。ID和有效载荷掩码只用于精确ID和/或精确有效载荷的比较。

Rx传入信息的ID可以根据以下标准进行过滤。

- 通过检测传入信息的ID字段和目标CAN_FLT_ID1寄存器的内容之间的相等，与准确的ID值相匹配。ID掩码被使用。

- 与ID的最大范围匹配，即任何ID值小于或等于目标CAN_FLT_ID1寄存器内容的报文都被接受。ID掩码不被使用。
- 与ID的最小范围相匹配，即任何ID值大于或等于目标CAN_FLT_ID1寄存器内容的信息都被接受。ID掩码不被使用。
- 接受ID范围内的匹配，即任何ID值大于或等于目标CAN_FLT_ID1寄存器的内容且小于或等于目标CAN_FLT_ID2_IDMASK寄存器内容的报文。ID掩码不被使用。

参见伪装网络控制1寄存器（CAN_CTRL1_PN）中的CAN_CTRL1_PN[IDFS]。

上述ID过滤的标准必须与CAN_FLT_ID1寄存器中的FLT_IDE和FLT_RTR目标值一致。只有各自的IDE和RTR位与FLT_IDE和FLT_RTR位域的内容相匹配的Rx帧将被比较。当选择了ID的范围（CAN_CTRL1_PN[IDFS] = 11），FLT_ID1和FLT_ID2都会被参考到CAN_FLT_ID1寄存器中相同的FLT_IDE和FLT_RTR位。

ID掩码只适用于精确ID比较过滤选项（CAN_CTRL1_PN[IDFS] = 00），以确定在比较中考虑哪些位。对于精确匹配选项，掩码可以选择ID字段内的任何位。对于最大范围、最小范围和范围内的比较，ID掩码不被考虑。

IDE和RTR掩码在精确和范围ID比较过滤选项中都被应用，以确定在比较中考虑哪些位。

与ID标准类似，Rx传入信息的64位数据或有效载荷（PL）可以根据以下标准进行过滤。

- 通过检测传入信息的有效载荷字段和PL1寄存器的内容是否相等，与准确的有效载荷值相匹配。使用有效载荷掩码。
- 与有效载荷的最大范围相匹配，即任何有效载荷值小于或等于PL1寄存器内容的信息都被接受。有效载荷掩码不被使用。
- 与有效载荷的最小范围相匹配，即任何有效载荷值大于或等于PL1寄存器内容的信息都被接受。有效载荷掩码不被使用。
- 在有效载荷范围内的匹配，即任何有效载荷值大于或等于PL1寄存器的内容且小于或等于PL2寄存器的内容的信息都被接受。有效载荷掩码不被使用。

参见伪装网络控制1寄存器（CAN_CTRL1_PN）中的CAN_CTRL1_PN[PLFS]。

上述有效载荷过滤的标准必须与CAN_FLT_DLC寄存器中的FLT_DLC上限和下限值相一致。只有当Rx传入信息的DLC值在DLC范围内时，Rx传入信息的有效载荷才会按照选定的标准进行过滤。

- 大于或等于FLT_DLC_LO（下限）和
- 低于或等于FLT_DLC_HI（上限）。

反之，超出指定范围的DLC值将导致不匹配。通过使FLT_DLC_LO = FLT_DLC_HI，只有指定数量的字节的有效载荷将被过滤。DLC是不可屏蔽的。

当选择了有效载荷范围内的选项（CAN_CTRL1_PN[PLFS] = 11），PL1和PL2都被认为是8字节的数据长度。所有被接收信息的DLC排除的数据字节都被认为是零值。

有效载荷掩码仅用于精确匹配选项（CAN_CTRL1_PN[PLFS] = 00），以选择Rx传入信息的8字节数据字段和PL1寄存器的内容中的哪些位或字节进行匹配。掩码长度必须符合DLC值的预期范围。对于最大范围、最小范围和内部范围的比较，不考虑有效载荷屏蔽。

当FlexCAN收到一个远程帧，并且CAN_CTRL1[FCS]位被配置为选择有效载荷比较时，不考虑有效载荷过滤，比较结果为不匹配。

Rx传入的信息也可以根据信息接收的数量和速率进行过滤，具体而言。

- 符合ID或有效载荷的过滤标准的几条信息的预定数量。这个数量可以在1到255的范围内配置。参见假装网络控制1寄存器（CAN_CTRL1_PN）。
- 在超时触发前，没有符合ID或有效载荷过滤标准的消息。也就是说，在规定的时间内没有接收到匹配的消息。参见假装网络控制2寄存器（CAN_CTRL2_PN）。

FlexCAN可以从内部定时器产生一个唤醒超时事件，该定时器具有相关的比较器电路，能够在计数达到预先定义的超时值时产生一个超时标志，如CAN_CTRL2_PN[MATCHTO]中规定的那样。

上述过滤标准可以一起使用，具体如下。

- 仅限消息ID过滤。

- 消息ID过滤和有效载荷过滤。
- 信息ID过滤只发生N次。
- 消息ID过滤和有效载荷过滤发生了N次。

超时计数器与接收过滤过程同时运行。两个引擎，超时计数器和消息过滤，都是独立的。如果一个传入的消息符合选定的过滤标准，超时计数器就会一直计数，直到CPU被唤醒。
反之，如果超时计数器达到目标值，那么消息过滤过程将继续过滤传入的消息，直到CPU被唤醒。CAN_WU_MTC[MCOUNTER]字段将报告在伪装网络模式下发生的匹配信息的数量，直到CPU被唤醒的那一刻。

在假装联网模式下，可能发生的唤醒事件将设置相应的唤醒标志（见假装联网唤醒匹配寄存器（CAN_WU_MTC））。

- 如果按照选定的过滤标准成功匹配，CAN_WU_MTC[WUMF]。
- 在超时触发的情况下，CAN_WU_MTC[WTOF]。

只要各自的屏蔽位被启用（CAN_CTRL1_PN寄存器中的WUMF_MSK或WTOF_MSK），这些标志中的任何一个都会向CPU产生中断。

有四个WMB（唤醒信息缓冲区），用于在伪装网络模式下存储传入的信息。最多可存储四条信息（见伪装网络唤醒信息缓冲器寄存器（CAN_WMB0 - CAN_WMB3））。当CTRL1_PN[NMATCH]值为1时，如果符合过滤条件，则只接收一个信息，这个信息被存储在CAN_WMB0中。
如果NMATCH值在2到4之间，则CAN_WMB1、CAN_WMB2和CAN_WMB3分别用来存储第二、第三和第四个匹配信息。如果NMATCH大于4，则最后四个匹配的报文被存储在WMB中，尊重WMB索引以表示到达顺序，最新的被存储在CAN_WMB3中。在WMBs的数据字段中，只存储传入的匹配信息的有效数据字节。非有效的数据字节被读为零。在DLC=0和RTR=1的情况下，数据字段被填充为0。在上述任何一种情况下，当过滤标准完成并且CAN_CTRL1_PN[WUMF_MSK]被启用时，就会产生唤醒中断。

当发生非匹配的唤醒事件（超时或外部）且MCOUNTER寄存器等于或大于4时，存储在WMB0的信息没有有效内容。
CAN_WMB0被用作CAN总线上当前消息的缓冲区。在Pretended Networkig模式下收到的报文没有时间戳，WMB结构中的相应字段必须被忽略。

在低功耗模式（停止）下，除了CAN_PE子块内的PN功能外，所有进程都被关闭，该功能由振荡器时钟保持时钟（见时钟域和限制）。FlexCAN继续接收传入的Rx报文，只是将它们与预定义的目标值进行比较，并符合选定的过滤标准。通常在正常模式下的匹配、仲裁、移入和移出过程，在假装网络模式下不执行。

假装联网下的FlexCAN对CAN总线上的信息的反应与正常模式下的反应相同（即产生确认位，检测和计算错误等）。

## 50.5.6 搬迁过程

搬家过程有两种类型：迁入和迁出。

### 50.5.6.1 迁入

移入过程是将Rx
SMB收到的消息复制到与之匹配的Rx邮箱或FIFO。如果移动目的地是Rx
FIFO，消息的属性也被复制到CAN_RXFIR FIFO。每个Rx
SMB都有自己的移入过程，但在一个给定的时间内只有一个被执行，如前面所述。只有当Rx
SMB持有的报文有相应的匹配赢家时（见匹配过程），并且以下所有条件为真时，才开始移入。

- CAN总线已经达到或让过了这两种情况。
  - 在Rx SMB中携带信息的帧旁边的Intermission字段的第二位。
  - 在Rx SMB中携带信息的帧旁边的过载帧的第一比特。
- 没有持续的匹配过程
- 目标邮箱未被CPU锁定
- 没有正在进行的从另一个Rx
  SMB移入的过程。如果有一个以上的迁入进程同时启动，则两个进程都会被执行，最新的进程将取代最旧的进程。

整个文件中使用了*待入住*一词，它代表了仍未满足上述所有条件的待入住者。

如果满足以下任何一个条件，移入被取消，Rx SMB能够接收另一个信息。

- 在CAN总线到达携带该信息的帧旁边的Intermission字段的第一位并完成其匹配过程后，目的地邮箱将被停用。
- 以前有一个待处理的迁入同一目标邮箱的情况
- Rx
  SMB正在接收一个由FlexCAN本身传输的帧，并且自我接收被禁用（CAN _MCR[SRXDIS]位被断言）。
- 检测到任何CAN协议错误

请注意，如果模块进入冻结或低功耗模式，等待中的移动不会被取消。它只是停留在等待退出冻结和低功耗模式并被解锁的状态。如果MB在冻结模式下被解锁，则移动会立即发生。

迁入过程是由FlexCAN执行以下步骤。

1. 如果信息的目的地是Rx FIFO，将IDHIT推入RXFIR FIFO。
2. 根据为Rx存储元件选择的有效载荷大小，从Rx SMB读取所有数据字。
3. 按照为Rx存储元素选择的有效载荷大小，将所有数据字写到Rx邮箱。如果存储元素的数据大小小于消息的DLC字段中描述的原始有效载荷大小，则有效载荷被截断，不适合目的地大小的高阶字节被丢失。
4. 从Rx SMB读取控制/状态和ID字。
5. 向Rx邮箱写入控制/状态和ID字，并更新CODE字段。

迁入过程不是原子性的，在这种情况下，它立即被目的地邮箱的失活所取消（见邮箱失活），在这种情况下，邮箱可能留下部分更新，因此不连贯。例外的情况是，如果迁入目的地是一个Rx FIFO消息缓冲区，那么这个过程不能被取消。

在移入过程中，目的地消息缓冲区的BUSY位（CODE字段的最小有效位）被断言，以提醒CPU消息缓冲区的内容暂时不连贯。

### 50.5.6.2 迁出

迁出过程是指当有信息可供传输时，将内容从Tx邮箱复制到Tx SMB（见"仲裁过程 "部分）。迁出发生在以下条件下。

- 中场休息字段的第一位
- 在总线关闭状态下，当TX错误计数器在124至128范围内时
- 在总线空闲状态下
- 在等待总线空闲状态下

迁出过程不是原子性的。只有CPU有优先权在总线空闲状态下同时访问内存。在总线空闲状态下，移出过程对并发的内存访问具有最低的优先权。

### 50.5.7 数据的一致性

为了保持数据的一致性和FlexCAN的正常运行，CPU必须遵守传输过程和接收过程中描述的规则。

#### 50.5.7.1 传输中止机制

终止机制提供了一种安全的方式来请求中止一个待定的传输。如果传输被中止，或者帧不能被中止而被传输，则提供一个反馈机制来通知CPU。

为了中止传输，必须满足两个主要条件。

- CAN_MCR[AEN]位必须被断言
- CPU的第一个动作必须是将中止代码（0b1001）写入控制和状态字的CODE字段中。

为传输而配置的活动MB必须首先被中止，然后才能更新。如果中止代码被写入当前正在传输的邮箱或已经加载到Tx
SMB进行传输的邮箱，则写入操作被阻止，传输不会受到干扰。然而，中止请求被捕获并保持待定，直到满足以下条件之一。

- 该模块失去了总线仲裁
- 在传输过程中出现了一个错误
- 模块进入冻结模式
- 该模块进入总线关闭状态
- 有一个过载的框架

如果上述条件都没有达到，则MB被正确传输，中断标志在IFLAG寄存器中被设置，并产生一个对CPU的中断（如果启用）。
当中断标志被设置时，中止请求被自动清除。另一方面，如果达到上述条件之一，则不传输帧；因此，中止代码被写入CODE字段，中断标志在IFLAG中被设置，并（可选择地）产生一个中断给CPU。

如果CPU在内部开始传输之前写下中止代码，那么写操作就不会被阻止；因此，MB被更新，中断标志被设置。这样一来，CPU只需要读取中止代码，以确保活动的MB被*安全地灭活*。尽管AEN位被断言，而且CPU写了中止代码，但在这种情况下，MB被灭活而不是中止，因为传输还没有开始。
一个邮箱只有在中止请求被捕获后才会被中止，并保持待定，直到满足前面的一个条件。

终止程序可以总结为以下几点。

- CPU检查相应的IFLAG，如果断言，将其清除。
- CPU将0b1001写入C/S字的CODE字段中。
- CPU等待相应的IFLAG，表明该帧已被传送或中止。
- CPU读取CODE字段以检查该帧是被传送（CODE=0b1000）还是被中止（CODE=0b1001）。
- 有必要清除相应的IFLAG，以便允许对MB进行重新配置。
- 在再次传输之前，有必要重新配置已放弃的MB的EDL、BRS和ESI字段。

### 50.5.7.2 信箱失活

停用是为保护邮箱不被FlexCAN内部进程更新而提供的一种机制，因此，即使在正常模式下，CPU也可以在更新后依赖邮箱数据的一致性。

必须在MCR[AEN]位去干扰时执行传输邮箱的停用。

如果一个邮箱被停用，它既不参与仲裁过程也不参与匹配过程，直到它被重新激活。关于如何灭活和重新激活一个邮箱的更多详细说明，请参见发送过程和接收过程。

要使一个邮箱不活跃，CPU必须将其CODE字段更新为INACTIVE（0b0000或0b1000）。

由于用户无法将CODE字段的更新与FlexCAN内部流程同步，因此不激活会产生以下后果。

- 总线中与未激活的Rx邮箱的过滤相匹配的帧可能会在没有通知的情况下丢失，即使有其他具有相同过滤的邮箱。
- 含有未激活的Tx邮箱内的信息的帧可以被传送，而不需要设置相应的IFLAG

为了进行*安全的失活*，避免Tx邮箱的上述后果，CPU必须使用传输中止机制（见传输中止机制）。

灭活会自动解锁邮箱（见邮箱锁定机制）。

**注意事项**

作为Rx FIFO一部分的消息缓冲器不能被激活。FlexCAN对FIFO区域没有写保护。当RFEN被断电时，CPU必须保持FIFO区域的数据一致性。

### 50.5.7.3 信箱锁机制

除了邮箱失活外，FlexCAN在接收过程中还有一种数据一致性机制。当CPU读取代码为FULL或OVERRUN的Rx MB的控制和状态字时，FlexCAN认为CPU想在一个原子操作中读取整个MB，因此它为该MB设置一个内部锁标志。当CPU读取自由运行定时器（全局解锁操作），或读取另一个MB的控制和状态字（无论其代码如何）时，该锁就会被释放。CPU对C/S字的写入也会解除对MB的锁定，但这个过程不建议用于正常的解锁，因为它取消了一个待定的动作，并有可能丢失一个已收到的信息。MB锁定可以防止CPU在读取MB时将一个新的帧写入其中。

**注意事项**

锁定机制只适用于不属于FIFO并且代码不同于INACTIVE（0b0000）或EMPTY1（0b0100）的Rx MBs。另外，Tx MBs不能被锁定。

例如，假设FIFO被禁用，阵列中的第二个和第五个MB被编程为相同的ID，而且FlexCAN已经接收并存储了信息到这两个MB。假设现在CPU决定读取第5个MB，同时有另一个相同ID的信息到达。当CPU读取MB编号5的控制和状态字时，这个MB被锁定。新信息到达后，匹配算法发现没有 "可自由接收 "的MB，所以它决定覆盖5号MB。然而，这个MB是被锁定的，所以新的信息不能被写入那里。它将留在Rx SMB中，等待MB被解锁，然后才会被写入MB。

1. 在以前的FlexCAN版本中，读取C/S字会锁定MB，即使它是空的。当IRMQ位被否定时，这一行为被保持。

如果MB没有及时解锁，又有一个相同ID的新信息到达，那么新信息就会覆盖Rx SMB上的信息，在MB的CODE字段或错误和状态寄存器中都不会有丢失信息的提示。

当信息从Rx SMB移入到MB时，CODE字段上的BUSY位被断言。如果CPU读取控制和状态字并发现BUSY位被设置，它应该推迟访问MB，直到BUSY位被否定。

**注意事项**

如果BUSY位被断言或者MB是空的，那么读控制和状态字不会锁定MB。

激活优先于锁定。如果CPU不激活一个锁定的Rx MB，那么它的锁定状态就会被否定，并且该MB在当前的匹配轮中被标记为无效。Rx SMB上的任何未决信息将不再被传输到该MB。当CPU读取自由运行定时器寄存器（见"自由运行定时器寄存器（CAN_TIMER）"一节），或另一个MB的C/S字时，一个MB被解锁。

锁定和解锁机制在正常和冻结模式下都有相同的功能。

在"正常"或"冻结"模式下的解锁，会导致待处理信息的移入。然而，如果解锁发生在低功率模式下，则延迟进入（见操作模式），只有当模块恢复到正常或冻结模式时才会发生。

## 50.5.8 Rx FIFO

Rx FIFO是只接收的，通过断言CAN_MCR[RFEN]位来启用。该位的复位值为0，以保持与以前没有FIFO功能的模块版本的软件向后兼容。

**注意事项**

当启用CAN FD功能时，Rx FIFO不能被启用。

FIFO的深度为6个消息。FIFO结构（包括Message Buffers和FIFO引擎）占用的内存区域在Rx FIFO结构中描述。CPU可以通过重复读取FIFO输出端的Message

Buffer结构，按照接收到的消息的顺序依次读取。

当至少有一个帧可以从FIFO中读取时，CAN_IFLAG1[BUF5I]（Rx FIFO中可用的帧）被引爆。如果相应的屏蔽位使能，就会产生一个中断。收到中断后，CPU可以读取消息（作为消息缓冲区访问FIFO的输出）和CAN_RXFIR寄存器，然后清除中断。如果FIFO中还有更多的消息，清除中断的行为就会用下一个消息更新FIFO的输出，并用该消息的属性更新CAN_RXFIR，重新向CPU发出中断。否则，该标志保持为负值。FIFO的输出只有在CAN_IFLAG1[BUF5I]被确认时才有效。

CAN_IFLAG1[BUF6I]（Rx FIFO警告）在Rx FIFO内的未读报文数量由于接收到新的报文而从4个增加到5个时被引爆，这意味着Rx FIFO几乎已满。在CPU清除之前，该标志一直被置位。

CAN_IFLAG1[BUF7I]（Rx FIFO溢出）在传入的消息因Rx FIFO满而丢失时被断言。请注意，当Rx FIFO满了而消息被邮箱捕获时，该标志不会被置位。该标志会一直持续，直到CPU将其清除。

清除这三个标志中的一个并不影响其他两个的状态。

如果一个IFLAG位被断言，并且相应的屏蔽位也被断言，就会产生一个中断。

提供了一个强大的过滤方案，只接受用于目标应用的帧，减少中断服务的工作负荷。根据CAN_CTRL2[RFFN]的设置，过滤标准是由一个多达128个32位寄存器的编程来指定的，它可以被配置为以下格式之一（参见Rx FIFO结构）。

- 格式A：128个IDAF（扩展或标准ID，包括IDE和RTR）。
- 格式B：256个IDAF（标准ID或包括IDE和RTR的扩展14位ID片）。
- 格式C：512个IDAF（标准或扩展的8位ID片）。

**注意事项**

所选择的格式被应用于过滤表的所有条目。不可能在表中混合格式。

FIFO中每个可用的帧都有一个相应的IDHIT（识别器接受过滤命中指标），可以从C/S字中读取IDHIT字段，如Rx FIFO结构描述中所示。CPU获得该信息的另一种方式是通过

访问CAN_RXFIR寄存器。CAN_RXFIR[IDHIT]字段指的是FIFO输出的信息，并且在CAN_IFLAG1[BUF5I]标志被断言时有效。CAN_RXFIR寄存器必须在清除标志之前被读取，这保证了信息是指FIFO内正确的帧。

根据CAN_CTRL2[RFFN]的设置，过滤表中最多有32个元素被单独的屏蔽寄存器（CAN_RXIMRx）所影响，允许定义非常强大的过滤标准。如果CAN_MCR[IRMQ]位被否定，那么FIFO过滤表会受到CAN_RXFGMASK的影响。

**注意事项**

关于FD和非FD之间关于此功能的区别的更多信息，见表50-3。

#### 50.5.8.1 DMA操作下的Rx FIFO

只接收的FIFO可以支持DMA，这个功能可以通过断言CAN_MCR[RFEN]和CAN_MCR[DMA]位来启用。CAN_MCR[DMA]位的复位值为0，以保持与以前没有DMA功能的模块版本的向后兼容性。

DMA控制器可以通过在FIFO输出端口读取0x80-0x8C地址范围内的消息缓冲结构来读取收到的消息。

当CAN_MCR[DMA]被断言时，CPU不得访问FIFO输出端口地址范围。在启用CAN_MCR[DMA]之前，CPU必须对Rx FIFO区域中的IFLAGs进行服务。否则，这些IFLAG可能会显示FIFO有数据要被服务，并错误地产生一个DMA请求。在禁用CAN_MCR[DMA]之前，CPU必须执行一个清除FIFO的操作。

当至少有一个帧可以从FIFO中读取时，CAN_IFLAG1[BUF5I]（Rx FIFO中可用的帧）被引爆，因此同时产生一个DMA请求。收到请求后，DMA控制器可以读取信息（将FIFO的输出作为信息缓冲区来访问）。DMA读取过程必须通过读取地址0x8C来结束，这就清除了CAN_IFLAG1[BUF5I]，并且用下一个消息更新FIFO输出（如果FIFO不是空的），用新消息的属性更新CAN_RXFIR寄存器。如果FIFO中存储有更多的消息，CAN_IFLAG1[BUF5I]将被重新置位，并发出另一个DMA请求。否则，该标志保持为负值。

**注意事项**

在DMA完成FIFO读取后，CAN_RXFIR寄存器的内容不能被读取。IDHIT信息也是

在地址0x080的C/S字中可用（见Rx FIFO结构。

当DMA功能被启用时，CAN_IFLAG1[BUF6I]和CAN_IFLAG1[BUF7I]不被使用。

当FlexCAN使用DMA工作时，CPU不会收到任何Rx FIFO的中断，也不得清除相关的IFLAGs。此外，相关的IMASK不用于屏蔽DMA请求的产生。

**注意事项**

关于FD和非FD之间关于此功能的区别的更多信息，见表50-3。

### 50.5.8.2 清除FIFO操作

当CAN_MCR[RFEN]被断言时，清除FIFO操作是用来清空FIFO内容的功能。当CAN_MCR[RFEN]被断言时，当CPU在CAN_IFLAG1[BUF0I]中写1时，清空FIFO发生。这个操作只能在Freeze模式下执行，在其他模式下被硬件阻断。这个操作并不清除FIFO的IFLAG，因此在执行清除FIFO任务之前，CPU必须服务所有的FIFO IFLAG。

当Rx FIFO与DMA一起工作时，清除FIFO的操作会清除CAN_IFLAG1[BUF5I]，DMA请求被取消。

**注意事项**

*清除FIFO操作并不清除IFLAG，除了当CAN_MCR[DMA]被断言时，在这种情况下只有CAN_IFLAG1[BUF5I]被清除。*

## 50.5.9 CAN协议的相关功能

本节介绍了与CAN协议相关的功能。

### 50.5.9.1 符合CAN FD ISO标准

CAN FD协议已经被改进，以提高原始CAN

FD协议中的故障检测能力，该协议也被CAN in Automation（CiA）称为非ISO CAN FD。在改进后的CAN FD协议（现在称为ISO CAN FD）中，已经引入了一个三比特的东西计数器和一个奇偶校验位。CRC的计算也已经

已被修改。所有这些改进使得ISO CAN FD协议与非FD CAN FD协议不兼容。FlexCAN仍然支持非ISO CAN FD，因此它可以主要在中间阶段使用，用于评估和开发目的。

因此，强烈建议通过设置CAN_CTRL2寄存器中的ISOCANFDEN字段将FlexCAN配置为ISO CAN FD协议。

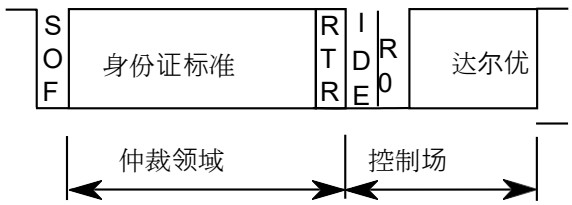
### 50.5.9.2 CAN FD框架

ISO 11898-1标准规定了符合ISO 11898-1标准的经典框架格式。11898-1（2003），并引入了CAN灵活数据速率帧格式。经典帧格式允许比特率达到1 Mbit/s，每帧有效载荷达到8字节。灵活数据速率帧格式允许比特率高于1 Mbit/s，每帧的有效载荷长于8字节。FlexCAN可以接收和发送与经典CAN报文交错的CAN FD报文。

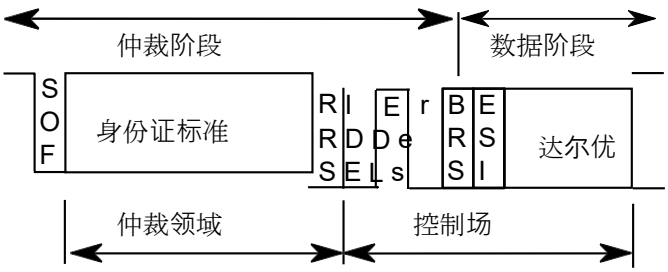
在CAN FD帧中有三个额外的控制位。扩展数据长度（EDL）位可以通过不同的数据长度编码实现更长的数据有效载荷。比特率开关（BRS）位决定比特率是否在CAN FD格式帧内被切换。错误状态指示器（ESI）标志由错误主动节点主导传输，由错误被动节点隐性传输。在CAN FD格式中没有远程帧（见远程帧）。被配置为传输远程帧的报文总是以经典的CAN格式发送出去。当收到一个FD帧并与一个邮箱相匹配时，接收消息缓冲区中的RTR位被否定。RTR位必须只在经典帧中被考虑。

CAN FD报文可以被格式化为数据字段超过8个字节的长帧，其范围可以从12到64个字节。它们也可以被配置为支持比特率切换，其中CAN帧的控制字段、数据字段和CRC字段以比帧的开始和结束更高的比特率传输。经典CAN格式的报文在标称速率下被限制为传输8字节的最大有效载荷。下图说明了具有标准或扩展ID的经典和FD帧的信息格式。

CAN标准格式




CAN FD标准格式




CAN扩展格式




CAN FD扩展格式




**图50-2.CAN报文格式**

接收和发送CAN
FD信息的能力由CAN_MCR[FDEN]位启用。无论是具有11位标识符的CAN帧中的隐性R0位还是具有29位标识符的CAN帧中的隐性R1位都被解码为EDL位（不是

保留位）。一个CAN FD帧是由一个隐性的EDL位来识别的，而一个

经典的CAN帧是由一个主导的EDL位识别的。BRS位指定该帧是否在其数据阶段切换比特率。一个长的帧是根据DLC字段的值来解码的（见消息缓冲区结构中的DLC定义）。

CAN FD报文可以用两种不同的比特率进行传输。CAN FD帧的第一部分，从帧开始（SOF）位到比特率开关（BRS）位，也称为仲裁阶段，以基于一组标称CAN比特定时配置值的标称比特率进行传输。第二部分，从BRS位到CRC分界符位，也被称为数据阶段，以第二组CAN数据位定时配置值定义的数据比特率进行传输。最后，从CRC分界符到中断位，传输恢复到额定比特率。在具有比特率切换的CAN FD帧中，如果BRS位是隐性的，则在帧内的采样点上改变比特计时。在BRS位之前，在CAN FD仲裁阶段，使用由CAN_CBT寄存器定义的名义CAN位定时（为了向后兼容，也由CAN_CTRL1寄存器定义）。当检测到一个隐性的BRS位时，CAN数据位的时序将按照CAN_FDCBT寄存器的定义使用。

**注意事项**

如果标称位定时中的时间量子的长度和数据位定时中的时间量子的长度不一致，可能会出现仲裁相的一个时间量子的量化误差，作为相位误差。这种情况可能发生在从仲裁阶段切换到数据阶段之后，并将持续到下一个同步事件。因此，在标称和数据位计时中，时间量子的长度应该是相同的，以尽量减少CAN总线上出现错误帧的机会，并优化使用FD帧的网络中的时钟容差。

如果选定的Tx MB中的BRS位被设置，CAN_FDCTRL[FDRATE]使所有帧的传输具有比特率切换。如果FDRATE被否定，那么无论BRS位的值如何，都会以额定速率进行传输。CAN_FDCTRL[FDRATE]位可以在任何时候写入，但只对传输或接收的下一个信息生效。

在CRC定界位的采样点或检测到错误时（以先发生者为准）恢复名义位计时。下图描述了当BRS位为隐性时进入和离开数据阶段的机制。

CAN FD框架

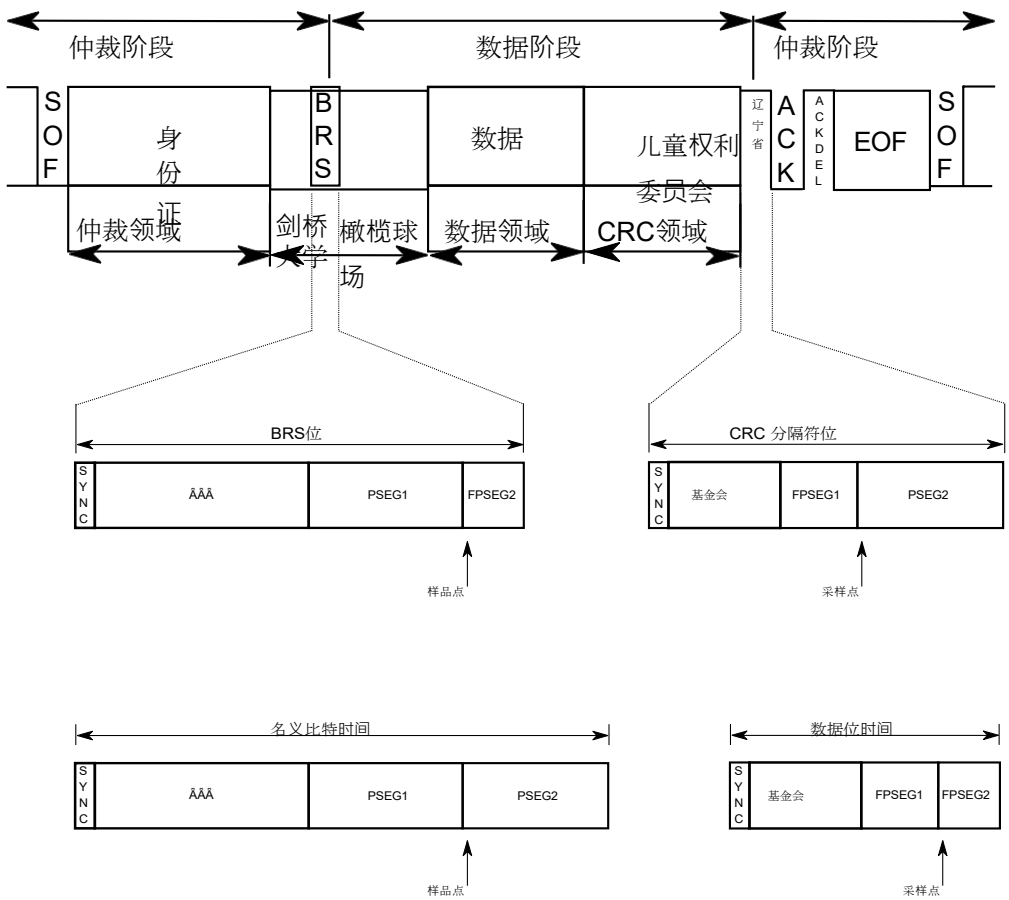



**图50-3.CAN FD报文的比特率切换机制**

## 注意事项

在经典的CAN帧中，CRC定界符是一个单一的隐性位。在CAN
FD帧中，CRC定界符可以由一个或两个隐性位组成。FlexCAN只发送一个隐性位作为CRC定界符，但在开始确认槽的隐性位到显性位的边缘之前，它可以接受两个隐性位。作为一个接收器，FlexCAN在第一个CRC定界位之后发送其确认位。在CAN
FD帧中，FlexCAN接受一个两比特的显性ACK槽作为有效的ACK，以补偿接收机之间的相位偏移。

CAN
FD数据阶段的最大可配置比特率取决于CAN_PE子块的时钟频率。例如，如果CA

N_PE的时钟频率为40MHz，最短的可配置比特时间为5个时间量子，那么数据阶段的比特率为8 Mbit/s。[2]

ESI位的值由发射器在传输开始时的错误状态决定，如果该帧起源于FlexCAN节点，则由原发射节点决定，如果FlexCAN作为消息的网关。如果发射器是错误被动的，ESI是隐性传输；否则，是显性传输。

对于不同的CAN帧格式有不同的CRC多项式。第一个多项式，CRC_15，用于所有经典CAN格式的帧。第二个，CRC_17，用于数据字段长达16个字节的CAN FD格式的帧。第三个，CRC_21，用于数据字段长于16个字节的CAN FD格式的帧。每个多项式的结果都是汉明距离为6。在帧的开始，所有三个CRC多项式都被同时计算。要传输的CRC序列由EDL位和DLC位域的值来选择。当接收信息时，FlexCAN对EDL和DLC进行解码，选择适当的CRC多项式来检查CRC错误。

在CAN FD格式帧中，为计算CRC而在比特流中包括了填充位。在经典的CAN格式帧中，不包括这些位。在传输了与CRC计算有关的最后一个比特后，CAN_FDCRC寄存器存储了所传输报文的CRC计算结果，其长度与报文类型相符，适用于CAN FD和非FD报文。CAN_CRCR寄存器仅对经典的CAN报文报告有效的CRC。

在CAN FD格式的帧中，CRC序列的CAN位填充方法被改变，因此填充位被插入到固定位置。当FlexCAN传输CAN FD帧时，在CRC序列的第一位之前插入一个固定的填充位，即使前面字段的最后几位不满足CAN填充条件。在CRC序列的每一个第四位之后会插入额外的填充位。任何固定位的值都是其前面位的反值。当FlexCAN接收一个CAN FD帧时，它从位流中丢弃固定的东西位以进行CRC检查。如果固定位的值与前一个位的值相同，则检测到 "错误"。

FlexCAN检测CAN FD帧的错误的方式与检测经典CAN帧的方式相同。CAN_ECR寄存器中的错误计数器RXERRCNT和TXERRCNT分别累积FD和非FD帧的Rx和Tx错误计数。有两个额外的错误计数器（RXERRCNT_FAST和TXERRCNT_FAST），累积在只设置了BRS位的CAN FD帧的数据阶段发生的Rx和Tx错误。更新错误计数器的规则对于CAN FD和非FD帧都是一样的（参见CAN_ECR寄存器）。

2. 本例中使用的频率可能不被该芯片支持；显示它只是为了说明最大可配置比特率是如何计算的。

错误标志BITERR1, BITERR0, ACKERR, CRCERR, FRMERR和STFERR在ESR1寄存器报告CAN
FD和非FD帧的错误。如果CAN_CTRL1[ERRMSK]被断言，它们还会产生ERRINT中断。CAN_ESR1寄存器有额外的错误标志（BITERR1_FAST、BITERR0_FAST、CRCERR_FAST、FRMERR_FAST和STFERR_FAST）来单独指示在BRS位被设置的CAN FD帧的数据阶段发生的错误。在CAN
FD帧的数据阶段没有检测到ACKERR。在CAN_ESR1[FLTCONF]中报告的故障限制状态对于CAN
FD和经典CAN帧是相同的，并且只基于RXERRCNT和TXERRCNT错误计数器。RXERRCNT_FAST和TXERRCNT_FAST计数器中包含的信息可被视为状态，以帮助检测与比特率值有关的错误性质。

当FlexCAN处于数据阶段，无论是发送还是接收CAN
FD报文，如果检测到错误，它将立即切换回仲裁阶段，并切换到额定速率，启动错误标志。

重新同步和硬同步在CAN
FD帧中的发生方式与在经典CAN帧中的发生方式相同。此外，在CAN
FD格式帧中，从EDL到R0的隐性到显性边缘也会进行硬同步。FlexCAN在CAN
FD数据阶段传输时不进行再同步。

### 50.5.9.3 收发器延时补偿

当报文的BRS位被设置时，CAN
FD协议允许以高于仲裁阶段使用的标称速率的比特率传输和接收数据。
该功能使使用的速率高达8Mbps。

在CAN
FD帧的数据阶段，如果发射器不能在该位的采样点上接收到自己的最新传输位，就会检测到一个位错误。当比特率切换被启用时（BRS位被断言），数据阶段的CAN比特时间长度可以变得比收发器的环路延迟短，从而阻碍了在当前CAN比特时间间隔内正确比较传输比特和接收比特。

FlexCAN支持一个可选的收发器延迟补偿（TDC）机制，该机制定义了一个二级采样点，在这个采样点上，传输的比特与接收的比特进行正确的比较，以检查比特错误。

TDC机制可以通过CAN_FDCTRL[TDCEN]位启用，并且只在设置了BRS位的FD帧的数据阶段有效。它对非FD帧或以正常比特率传输的FD帧都没有影响。TDC从BR

S位的采样点到CRC定界位的采样点都是有效的，前提是

正在传输的相应信息有BRS位被设置。当它被激活时，在实际接收的比特和延迟传输的比特之间进行比较，其中延迟是根据测量的收发器环路延迟计算的。

**注意事项**

使用收发器延迟补偿机制的发射器不考虑CRC定界位的实际值。CRC字段末尾的全局错误将导致接收机发送错误帧，发射机将在确认或帧末检测到。

对于每一个设置了BRS位的传输FD帧，延迟测量是由从隐性EDL位到显性R0位的转换触发的（如下图所示）。环路延迟是以协议引擎（PE）的时钟周期（CANCLK，见协议计时）来测量的，从传输的EDL-R0边缘到收到的EDL-R0边缘。二次采样点的位置由测量的环路延迟时间加上在CAN_FDCTRL[TDCOFF]中指定的偏移值来定义。

CAN_FDCTRL[TDCVAL]位域存储了这个计算的结果。当延迟测量时间过长时，TDCVAL值会在其最大值15 CANCLK时达到饱和。

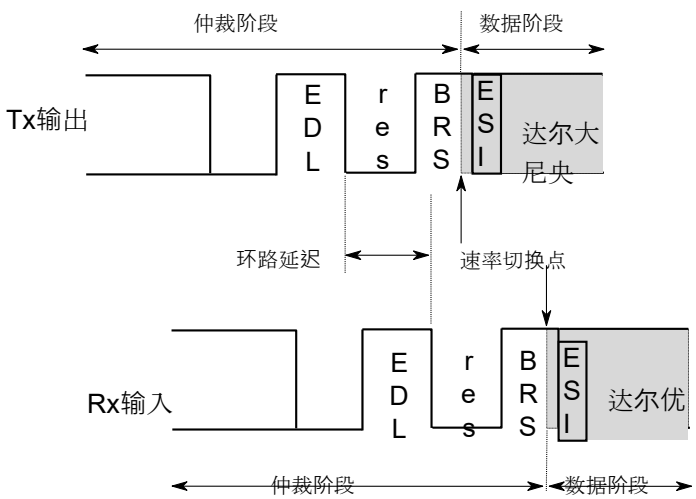



**图50-4.收发器环路延迟测量**

测量的环路延迟不足以用来定义二次采样点，因为它与CAN位的边缘有关。收发器延迟补偿偏移TDCOFF用于将二次采样点从边缘转移到位时间内的一个中间点（例如数据阶段的一半位时间），远离其边缘。因此，TDCOFF值不能大于数据阶段的CAN位时间。

在启用了比特率切换的CAN FD帧的数据阶段，在每个Tx CAN位开始时，传输的Tx位值被暂时存储在一个缓冲器中，并开始一个基于TDC VAL的时间倒计时，该倒计时以接收的Rx位（由外部环路延迟加上指定的偏移量延迟）与存储的Tx位的比较为结束。如果在二级采样点检测到位错误，FlexCAN会在下一个采样点向CAN总线发出一个错误标志。

在仲裁阶段，延迟补偿总是被禁用。在数据阶段，FlexCAN的收发器延迟补偿所能补偿的最大延迟是3个CAN位次-2个Tq。超过这个限制，CAN_FDCTRL[TDCFAIL]标志会被设置，表示收发器延迟补偿机制超出范围，无法补偿收发器环路延迟。

### 50.5.9.4 远程框架

远程帧是一种特殊的帧。用户可以通过将邮箱配置为传输，并将RTR位设置为'1'，从而将邮箱编程为远程请求帧。在远程请求帧传输成功后，该邮箱成为接收消息缓冲区，其ID与之前相同。

当FlexCAN收到远程请求帧时，可以用三种方式处理，这取决于远程请求存储（CTRL2[RRS]）和Rx FIFO启用（MCR[RFEN]）位。

- 如果RRS被否定，该帧的ID将与CODE字段0b1010的发送消息缓冲区的ID进行比较。如果有一个匹配的ID，那么这个邮箱帧将被传送。注意，如果匹配的邮箱有RTR位设置，那么FlexCAN将传送一个远程帧作为响应。收到的远程请求帧不存储在接收缓冲区。它只用于触发响应帧的传输。在远程帧匹配中不使用掩码寄存器，收到的帧的所有ID位（除RTR外）都应匹配。在收到远程请求帧并与邮箱匹配的情况下，这个消息缓冲区立即进入内部仲裁过程，但被视为普通的Tx邮箱，没有更高的优先级。该帧的数据长度与启动其传输的远程帧中的DLC字段无关。

- 如果RRS被断言，该帧的ID将与CODE字段为0b0100、0b0010或0b0110的接收邮箱的ID进行比较。如果有一个匹配的ID，那么这个邮箱将以数据帧的相同方式存储远程帧。不会产生自动的远程响应帧。掩码寄存器在匹配过程中使用。
- 如果RFEN被断言，FlexCAN将不会对符合FIFO过滤条件的远程请求帧产生自动响应。如果远程帧与目标ID之一相匹配，它将被存储在FIFO中并呈现给CPU。注意，对于过滤格式A和B，可以选择是否接受远程帧。对于格式C，总是接受远程帧（如果它们与ID匹配）。远程请求帧被认为是正常帧，当成功接收后，FIFO已经满了，就会产生FIFO溢出。

**注意事项**

在CAN
FD格式中没有远程帧。RTR位被一个固定的主导RRS位所取代。FlexCAN以经典CAN格式接收和发送远程帧。

### 50.5.9.5 过载框架

由于检测到CAN总线上的以下情况，FlexCAN确实会发送过载帧。

- 探测到中断期第一/第二位的优势位
- 检测帧结束字段第7位（最后一位）的主导位（Rx帧）。
- 检测错误帧定界符或过载帧定界符的第8位（最后一位）的优势位

### 50.5.9.6 时间戳

自由运行定时器的值在CAN总线上的Identifier字段开始时被采样，并在 "move-in "结束时存储在TIME STAMP字段，提供与时间有关的网络行为。

当CAN_CTRL2寄存器中的TIMER_SRC位被断言时，自由运行的定时器被外部的时间刻度连续计时。

当CAN_CTRL2寄存器中的TIMER_SRC位被否定时，自由运行定时器由FlexCAN位时钟提供时钟，它定义了CAN总线的波特率。
在信息传输/接收期间，每接收或传输一个比特，它就递增一个。当总线上没有信息时，它使用先前编程的波特率进行计数。

在禁用、停止和冻结模式下，自由运行定时器不会被递增。它可以在一个特定的帧接收时被重置，实现网络时间同步。参见控制1寄存器（CAN_CTRL1）中的TSYN描述。

### 50.5.9.7 协议时间

下图显示了为CAN协议引擎（PE）子模块供电的时钟生成电路的结构。CAN_CTRL1寄存器中的时钟源位CLKSRC定义了内部时钟是连接到晶体振荡器的输出（振荡器时钟）还是连接到外围时钟。为了保证可靠的操作，时钟源应该在模块处于禁用模式时选择（模块配置寄存器中的MDIS位被设置）。

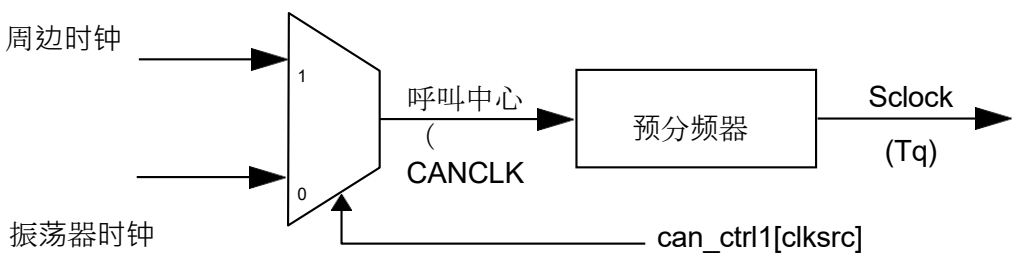



**图50-5.CAN引擎的时钟方案**

只要CAN总线定时需要严格的公差（最高0.1%），就应该选择振荡器时钟。水晶振荡器时钟比外围时钟有更好的抖动性能。

FlexCAN模块支持多种手段来设置CAN协议所要求的位定时参数。控制1寄存器（CAN_CTRL1）有各种字段，用于控制位定时参数。PRESDIV、PROPSEG、PSEG1、PSEG2和RJW。

CAN位定时寄存器（CAN_CBT）扩展了CAN_CTRL1中CAN位定时变量的范围。CAN
FD位定时寄存器（CAN_FDCBT）提供了第二套CAN位定时变量，在设置了比特率开关（BRS）的CAN FD帧的数据阶段应用。

**注意事项**

当启用CAN FD功能时，一定要设置CAN_CBT[BTF]并在CAN_CBT中配置CAN位定时变量。参见CAN位定时寄存器（CBT）。

PRESDIV字段（以及其扩展范围EPRESDIV和FDPRESDIV，用于CAN FD报文的数据相位）定义了产生串行时钟（Sclock）的预分频器值（见下面的公式），其周期定义了用于组成CAN波形的"时间量子"。一个时间量子（Tq）是CAN引擎处理的时间的原子单位。

$$Tq = \frac{(\text{预设值} + 1)}{f_{\text{呼叫中心（CANCLK}}}$$

比特率定义了CAN信息的接收或传输速率，由公式给出。

CAN比特时间=（1比特时间内的时间量子数）*Tq

$$\text{比特率} = \frac{1}{\text{CAN位时间}}$$

一个比特时间被细分为三个部分1（见图50-6、图50-7和表50-25）。

- SYNC_SEG：该段有一个固定长度的时间量子。预计信号边缘将在这一段内发生
- 时间段1：该段包括CAN标准中的传播段和相位段1。它可以通过设置CAN_CTRL1寄存器的PROPSEG和PSEG1字段，使它们的总和（加2）在2到16个时间段的范围内进行编程。当CAN_CBT[BTF]位被断言时，FlexCAN使用CAN_CBT寄存器中的EPROPSEG和EPSEG1字段，使它们的总和（加2）在2到16个时间量子之间。

1.关于基本概念的进一步解释，见ISO 11898-1。关于比特计时，也请参见CAN 2.0A/B协议规范。

是在2到96个时间量子的范围内。对于设置了BRS位的CAN FD格式的报文，FlexCAN使用CAN_FDCBT的FDPROPSEG和FDPSEG1来代替，这样它们的总和（加1）就在2到39个时间量子的范围内。

- 时间段2：该段代表CAN标准的相位段2。它可以通过设置CAN_CTRL1寄存器的PSEG2字段（加1）来编程，长度为2到8个时间段。当CAN_CBT[BTF]位被断言时，FlexCAN使用CAN_CBT寄存器的EPSEG2字段，使其值（加1）在2至32个时间量子范围内。对于设置了BRS位的CAN FD格式的报文，FlexCAN使用CAN_FDCBT的FDPSEG2，这样它的值（加1）就在2到8个时间量子的范围内。时间段2不能小于信息处理时间（IPT），在FlexCAN中IPT的值是2个时间量子。

**注意事项**

上述时间段定义的比特时间不得小于5个时间量子。对于比特时间的计算，请使用信息处理时间（IPT）为2，这是FlexCAN模块中实现的数值。

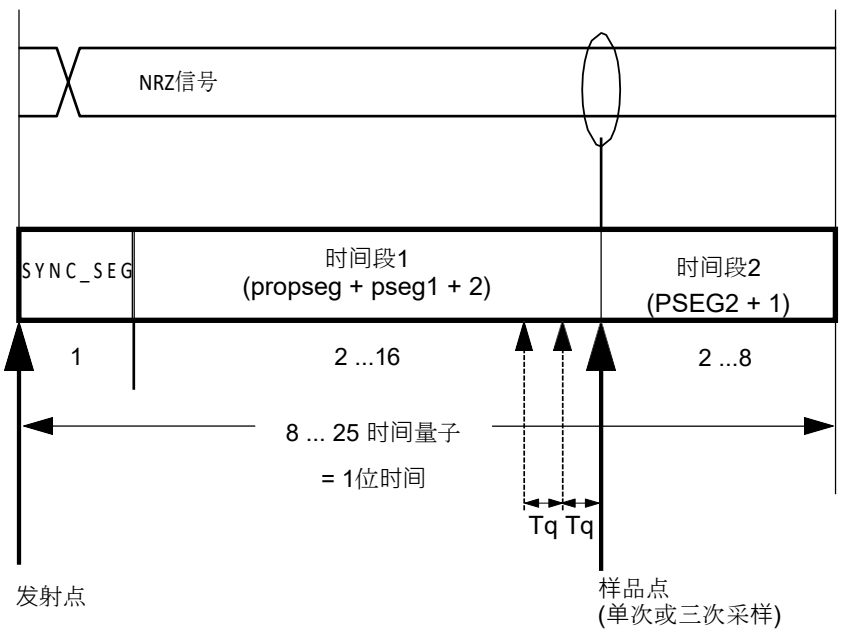



**图50-6.位时间内的分段（使用经典CAN格式的CAN_CTRL1位定时变量的例子）**

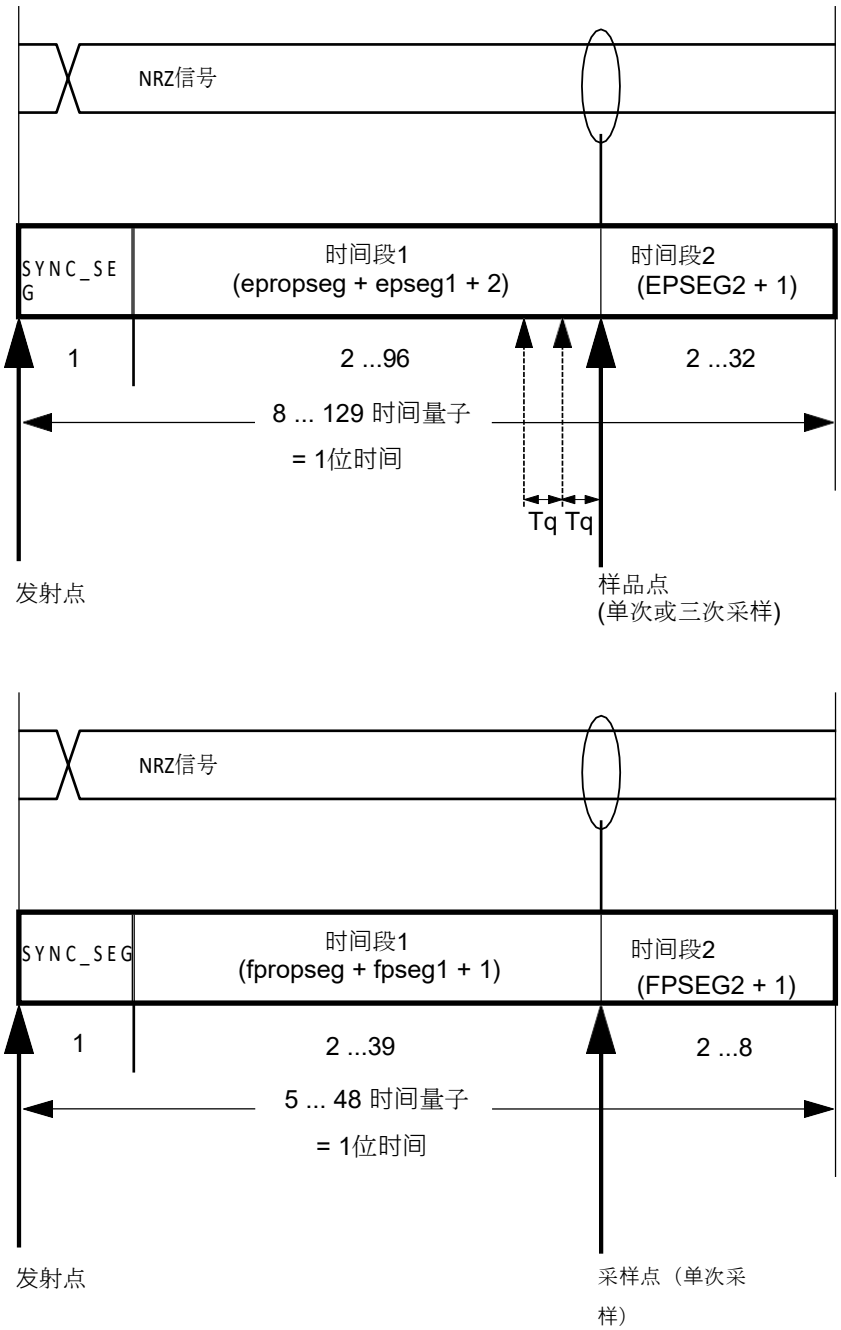



**图50-7.位时间内的分段（使用CAN_CBT和CAN_FDCBT位定时变量的CAN FD格式的例子）**

**表50-25.时间段的语法**

| 语法 | 描述 |
|---|---|
| 呼叫中心 | 系统希望在此期间总线上发生转换。 |
| TSEG1 | 相当于PROPSEG和PSEG1之和。 |
| TSEG2 | 对应于PSEG2值。 |

表继续在下一页...

表50-25.时间段的语法（续）

| 语法 | 描述 |
| --- | --- |
| 发射点 | 一个处于发送模式的节点在这时将一个新的数值传输到CAN总线上。 |
| 样品点 | 一个节点在此点对总线进行采样。如果选择了每比特三个样本的选项，那么这个点就标志着第三个样本的位置。 |

下表给出了经典CAN格式（Bosch CAN 2.0B）（非FD）报文的一些CAN兼容段设置的例子。

表50-26.符合博世CAN 2.0B标准的位时间段设置

| 时间段1 | 时间段2 | 重新同步化的跳转宽度 |
| --- | --- | --- |
| 5 .. 10 | 2 | 1 .. 2 |
| 4 .. 11 | 3 | 1 .. 3 |
| 5 .. 12 | 4 | 1 .. 4 |
| 6 .. 13 | 5 | 1 .. 4 |
| 7 .. 14 | 6 | 1 .. 4 |
| 8 .. 15 | 7 | 1 .. 4 |
| 9 .. 16 | 8 | 1 .. 4 |

## 注意事项

用户必须确保位时间设置符合CAN协议标准（ISO 11898-1）。

每当CAN位被用作时间长度的衡量标准时（例如，估计消息中CAN位事件的发生），一个CAN位中的外围时钟数（NumClkBit）可以计算为：。

$$\text{NumClkBit} = \frac{f_{SYSX}}{CANCLK} \text{ (PRESDIV + 1) X (PROPSEG + PSEG1 + PSEG2 + 4) } f$$

其中。

- NumClkBit是一个CAN位的外设时钟数。
- fCANCLK是协议引擎（PE）时钟（见图"CAN引擎时钟方案"），单位为Hz。
- $f_{SYS}$ 是系统（CHI）时钟的工作频率，单位为Hz。
- PSEG1是CAN_CTRL1[PSEG1]字段中的数值。
- PSEG2是CAN_CTRL1[PSEG2]字段中的数值。

- PROPSEG是CAN_CTRL1[PROPSEG]字段的值。
- PRESDIV是CAN_CTRL1[PRESDIV]字段中的数值。

上述公式也适用于CAN位定时寄存器（CAN_CBT）和CAN FD位定时寄存器（CAN_FDCBT）中描述的替代CAN位定时变量。

例如，180个CAN位=（180×NumClkBit）外围时钟周期。

### 50.5.9.8 仲裁和匹配时间

在正常的接收和传输过程中，匹配、仲裁、移入和移出过程是在CAN帧内的某些时间窗口中执行的，如以下数字所示。

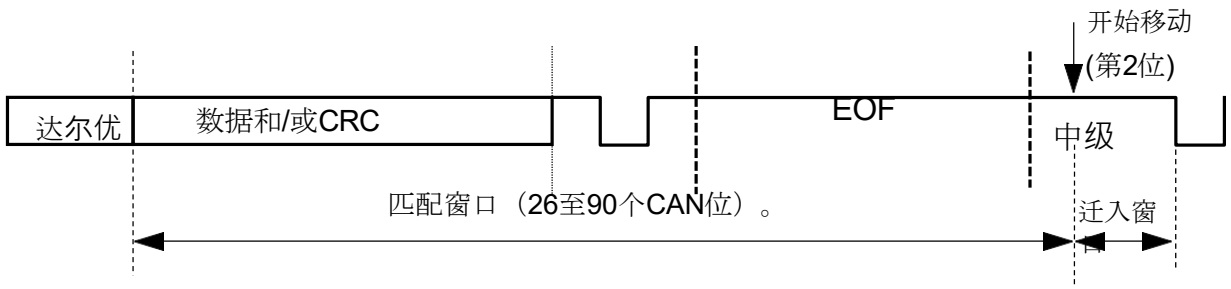



**图50-8.匹配和迁入的时间窗口**

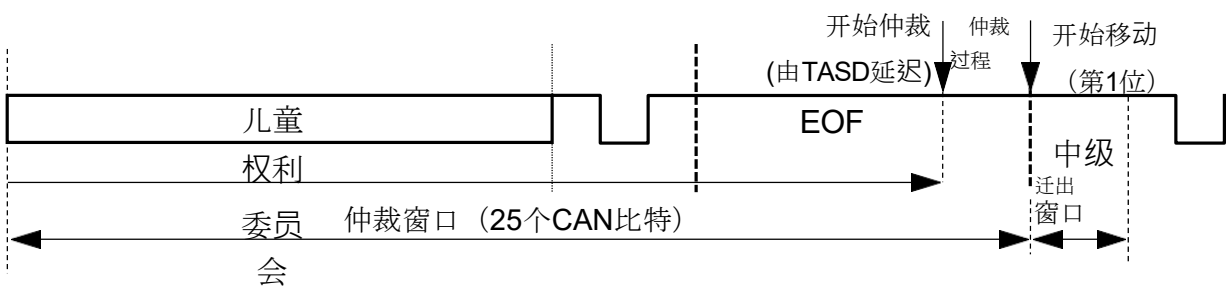



**图50-9.仲裁和迁出时间窗口**

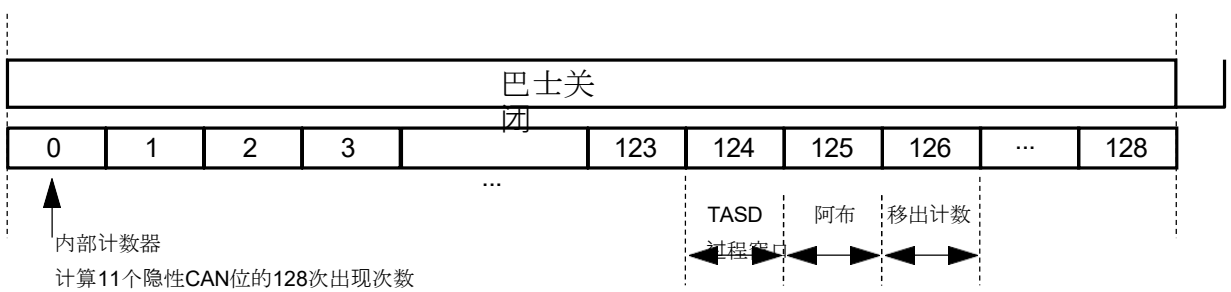



**图50-10.在总线关闭和移出时间窗口结束时进行仲裁**

## 注意事项

在前面的数字中，匹配和仲裁的时间没有考虑到由同时进行的

由于CPU或其他内部FlexCAN子块导致的内存访问。

### 50.5.9.9 Tx 仲裁启动延迟

控制2寄存器（CAN_CTRL2[TASD]）中的Tx Arbitration Start Delay（TASD）位域是一个变量，它表示FlexCAN用于延迟Tx Arbitration过程起点与当前帧的CRC字段第一位的CAN位数。这个变量只能在Freeze模式下写入，因为它在其他模式下被硬件阻断。

传输性能受到CPU在内部仲裁过程结束后重新配置信息缓冲区（MB）进行传输的能力的影响，FlexCAN在内部仲裁过程中找到了获胜的MB进行传输（见仲裁过程）。如果仲裁在中断字段的第一位前过早结束，那么CPU就有可能重新配置一些Tx MBs，获胜的MB就不再是最佳的传输候选者。

TASD有助于通过定义仲裁开始点来优化传输性能，如下图所示，基于以下因素。

- 外围设备与振荡器的时钟比率
- 决定CAN比特率的CAN比特计时变量
- 匹配和仲裁过程中使用的消息缓冲区（MB）的数量。

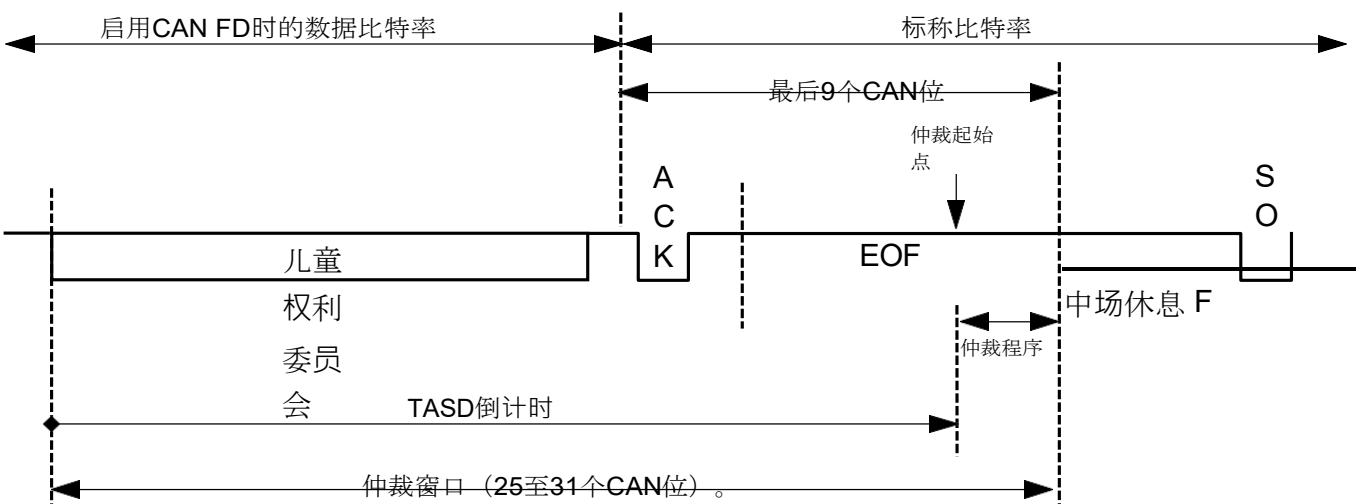



**图50-11.最佳的Tx仲裁开始点**

以CAN比特为单位，仲裁过程的持续时间与可用的MB数量和CAN比特率成正比，与外围时钟频率成反比。

最佳的仲裁时间是在一个CAN帧的中间字段的第一位之前扫描最后一个MB。例如，如果MB很少，外设/振荡器的时钟比率很高，而CAN波特率很低，那么仲裁可以放在更靠近帧的末端，为其起始点增加更多的延迟，反之亦然。

如果TASD被设置为0，那么仲裁的启动就不会被延迟，并且为仲裁保留更多的时间。另一方面，如果TASD接近24，那么CPU可以更晚地配置Tx MB，为仲裁保留的时间就更少。如果为仲裁预留的时间太少，FlexCAN可能无法及时找到获胜的MB，从而无法赢得与CAN总线上外部节点进行总线仲裁的最佳机会。

最佳的TASD值可按以下方式计算。

对于CAN FD帧和 $(MAXMB + 1) \leq NMB_{END}$

$$TASD = 31 - \frac{2 * ( maxmb + 1 ) + 4}{CPCB_N}$$

对于CAN FD帧和 （MAXMB + 1）

$>NMB_{END}$ $TASD = 22 - \frac{2 * ( MAXMB + 1 ) - NMB_{END}}{CPCB_F}$

对于非FD框架

$$TASD = 25 - \frac{2 * ( maxmb + 1 ) + 4}{CPCB}$$

其中。

$$NMB_{结束} = \frac{(9 * cpcb_N) - 4}{2}$$

$$BITRATE_N = \left( \frac{fCANCLK}{[1 + (EPSEG1 + 1) + (EPSEG2 + 1) + (EPROPSEG + 1)] \times (EPRESDIV + 1)} \right)$$

$$BITRATE_F = \left( \frac{fCANCLK}{[1 + (FPSEG1 + 1) + (FPSEG2 + 1) + FPROPSEG] \times (FPRESDIV + 1)} \right)$$

$$CPCB_N = \frac{f_{SYS}}{BITRATE_N}$$

$$CPCB_F = \frac{f_{SYS}}{BITRATE_F}$$

$$CPCB = CPCB_N$$

- MAXMB是CAN_CTRL1[MAXMB]字段中的数值。
- $NMB_{END}$
  是指在一帧结束时最后9个CAN比特期间，仲裁过程可以扫描的消息缓冲区的数量，见上图。
- $BITRATE_N$
  是由标称的CAN比特时间变量计算出的CAN比特率，单位为比特/秒。
- $BITRATE_F$
  是由数据CAN比特时间变量计算出的CAN比特率，单位为每秒比特。
- $CPCB_N$ 是CAN FD帧的名义比特率中每个CAN比特的外设时钟数。
- $CPCB_F$ 是CAN FD帧的数据比特率中每个CAN比特的外设时钟数。
- CPCB是指非FD帧的每个CAN位的外设时钟数。
- fCANCLK是振荡器时钟，单位为Hz
- $f_{SYS}$ 是外围时钟，单位为Hz
- EPSEG1是CAN_CBT[EPSEG1]字段中的值（也可使用CAN_CTRL1[PSEG1]）。
- EPSEG2是CAN_CBT[EPSEG2]字段中的值（也可使用CAN_CTRL1[PSEG2]）。
- EPROPSEG是CAN_CBT[EPROPSEG]字段中的值（也可使用CAN_CTRL1[PROPSEG]）。

- EPRESDIV是CAN_CBT[EPRESDIV]字段中的值（也可使用CAN_CTRL1[PRESDIV]）。
- FPSEG1是CAN_FDCBT[FPSEG1]字段中的数值。
- FPSEG2是CAN_FDCBT[FPSEG2]字段中的数值。
- FPROPSEG是CAN_FDCBT[FPROPSEG]字段中的数值。
- FPRESDIV是CAN_FDCBT[FPRESDIV]字段中的值，更

多细节请参见协议计时。

下表给出了一些配置情况下的TASD值计算。

案例1:

- 时钟比率=2:1（例如：外围时钟80MHz，振荡器时钟40MHz）。
- 仲裁阶段的比特率=1 Mbaud

**表50-27.TASD值。**

| 信息缓冲区的数量 | TASD值 | 数据阶段的最大比特率(Mbaud) |
|---|---|---|
| 16 | 24 | 无效 |
| 32 | 24 | 8.0 |

案例2:

- 时钟比率=1:1（例如：外围时钟40MHz和振荡器时钟40MHz）。
- 仲裁阶段的比特率=1 Mbaud

**表50-28.TASD值。**

| 信息缓冲区的数量 | TASD值 | 数据阶段的最大比特率(Mbaud) |
|---|---|---|
| 16 | 24 | 无效 |
| 32 | 23 | 6.67 |

案例3:

- 时钟比率=2:1（例如：外围时钟40MHz，振荡器时钟20MHz）。
- 仲裁阶段的比特率=1 Mbaud

**表50-29.TASD值。**

| 信息缓冲区的数量 | TASD值 | 数据阶段的最大比特率(Mbaud) |
|---|---|---|
| 16 | 24 | 无效 |

| | | |
|---|---|---|
| 32 | 23 | 4.0 |

## 50.5.10 时钟域和限制

FlexCAN模块有两个互相不同步的时钟域。

- 总线域为控制主机接口（CHI）子模块提供能量，并来自外设时钟。
- 振荡器域为CAN协议引擎（PE）子模块提供信号，并直接来自晶体振荡器时钟，因此可以在CAN总线上实现非常低的抖动性能。

当CAN_CTRL1[CLKSRC]位被设置时，会发生同步操作，因为两个域都连接到外设时钟（在外设和振荡器时钟之间形成1：1的比例）。

当两个域连接到具有不同频率和/或相位的时钟时，对两个时钟域之间的频率关系有限制。在异步运行的情况下，总线域的时钟频率必须总是大于振荡域的时钟频率。

**注意事项**

不允许外设和振荡器时钟之间的比例为1：1的异步操作。

在进行匹配和仲裁时，FlexCAN需要在一个CAN帧的时隙内扫描整个Message Buffer存储器，该帧由若干个CAN比特组成。为了有足够的时间来做这件事，必须遵守以下要求。

- 外围时钟频率不能小于振荡器的时钟频率。
- 每个CAN位必须有一个最小的外围时钟数量，如下表所示

**表50-30.经典CAN格式的每个CAN位的最小外设时钟数**

| 信箱的数量 | CAN_MCR[RFEN]位的值 | 每个CAN位的外围时钟的最小数量 |
|---|---|---|
| 16 | 0 | 16 |
| 32 | 0 | 16 |
| 16 | 1 | 16 |
| 32 | 1 | 17 |

对于经典帧格式，上表中规定的每个CAN位的最小外设时钟数决定了给定数量的邮箱和预期CAN位速率的最小外设时钟频率。CAN比特率取决于

关于一个CAN位的时间量子的数量，可以通过调整控制1寄存器（CAN_CTRL1）或CAN位时间寄存器（CAN_CBT）中包含的一个或多个位计时值来定义。时间量子（Tq）是在协议计时中定义的。每个CAN位的最小时间量子数必须是8；因此，振荡器的时钟频率应该至少是CAN位速率的8倍。

对于CAN FD帧格式，有一些需要满足的约束。在标称比特率中每个CAN比特的外设时钟数（NumClkNomBit）可以通过以下公式计算。

$$\text{NumClkNomBit} = \frac{f_{SYSX}}{CANCLK} \text{ (PRESDIV + 1) X (PROPSEG + PSEG1 + PSEG2 + 4) } f$$

$$\frac{f_{SYS}}{\text{编码率（NomBitRate}}$$

其中PRESDIV、PSEG1和PSEG2是CTRL1寄存器中的CAN位时间值。另外，也可以使用CBT寄存器中的EPRESDIV、EPSEG1和EPSEG2值来代替。NumClkNomBit也可以作为仲裁阶段使用的预期额定比特率（NomBitRate）的函数来计算，如上式所示。

设置了BRS位的FD帧的数据阶段中的CAN位数（简称快速CAN位）取决于有效载荷中的数据字节数。快速CAN比特的数量（NumOfFastBits）可以在下表中确定。数据字节数越少，快速CAN位的数量就越少，在内部匹配和仲裁过程中，FlexCAN可用于扫描整个Message Buffer内存的时间就越少。

**表50-31.一个CAN FD帧中快速CAN比特的数量**

| 数据字节的最小数量 | DLC领域 | 快速比特数（NumOfFastBits |
|---|---|---|
| 0 | 0x0 | 21 |
| 1 | 0x1 | 29 |
| 2 | 0x2 | 37 |
| 3 | 0x3 | 45 |
| 4 | 0x4 | 53 |
| 5 | 0x5 | 61 |
| 6 | 0x6 | 69 |
| 7 | 0x7 | 77 |
| 8 | 0x8 | 85 |
| 12 | 0x9 | 117 |
| 16 | 0xA | 149 |

*表继续在下一页...*

表50-31.一个CAN FD帧中的快速CAN比特数（续）

| 数据字节的最小数量 | DLC领域 | 快速比特数（NumOfFastBits |
|---|---|---|
| 20 | 0xB | 186 |
| 24 | 0xC | 218 |
| 32 | 0xD | 282 |
| 48 | 0xE | 410 |
| 64 | 0xF | 538 |

CAN
FD帧的关键部分是在数据阶段，这里的CAN比特率要比仲裁阶段快。可以计算出每个快速CAN位的最小外设时钟数（MinNumClkFastBit），以保证FlexCAN在接收和传输期间有足够的时间来扫描消息缓冲区内存。下面的公式可以计算出这个约束条件。

$$MinNumClkFastBit_A = \frac{(8.5 \times MaxNumOfMb) + 64 - (9 \times NumClkNomBit)}{NumOfFastBits}$$

其中MaxNumOfMb是CAN_MCR[MAXMB]中定义的最大可用邮箱数量。

CHI和PE子块之间的时钟域交叉电路也规定了每个快速CAN位的最小外设时钟数，以便握手机制正常工作而不会通过接口丢失状态信息，如下式所示。

$$MinNumClkFastBit_B = 3 \times \left(1 + \frac{f_{SYS}}{f_{CANCLK}}\right)$$

因此，每个快速CAN位的最小外设时钟数（MinNumClkFastBit）是由上述两个计算值中较大的一个决定的。

$$MinNumClkFastBit = Maximum(MinNumClkFastBit_A, MinNumClkFastBit_B)$$

然后，在CAN
FD帧的数据阶段的最大CAN比特率（DataBitRateMAX）可以按以下方式计算。

$$\text{数据比特率}_{MAX} = \frac{f_{CANCLK}}{\text{循环播放}\left\langle \frac{\text{MinNumClkFastBit} \times f_{CANCLK}}{f_{SYS}} \right\rangle}$$

外设和振荡器的时钟频率、邮箱的最大数量和预期的额定比特率会影响FlexCAN在CAN
FD模式下可达到的最大数据比特率。此外，数据比特率还取决于特定应用中使用的FD帧的最小有效载荷大小。

为了说明CAN
FD比特率是如何受FlexCAN变量配置的影响，我们考虑了一个应用实例，将外设和振荡器的时钟频率分别设置为50MHz和40MHz。

第1步--考虑到标称比特率为1
Mbps，标称比特率中每个CAN比特的外围时钟数量计算如下。

$$\text{NumClkNomBit} = \frac{50 \times 10^{6}}{1 \times 10^{6}} = 50$$

第2步--
快速CAN比特的数量（NumOfFastBits）在上面的表格中确定。例如，如果FD帧的最小有效载荷是8字节，那么在数据阶段就有85个CAN比特。

第3步 -
假设邮箱的最大数量是96，可以计算出每个快速CAN位的最小外设时钟数（MinNumClkFastBit）。

$$\text{MinNumClkFastBit}_{A} = \frac{(8.5 \times 96) + 64 - (9 \times 50)}{85} = 5.06$$

$$\text{MinNumClkFastBit}_{B} = 3 \times \left\langle 1 + \frac{50}{40} \right\rangle = 6.75$$

MinNumClkFastBit = Maximum ( 5.06, 6.75 ) = 6.75

第4步 - 最终可以找到数据阶段的最大CAN比特率。

$$\text{数据比特率}_{MAX} = \frac{40 \times 10^6}{\text{循环播放}\left\langle \frac{6.75 \times 40 \times 10^6}{50 \times 10^6} \right\rangle} = 6.667 \text{ Mbps}$$

如本例所示，即使振荡器的时钟频率（40 MHz）足以在CAN FD模式下产生8 Mbps的数据速率，但具体的FlexCAN配置将这一速率限制在6.667 Mbps。这种限制主要是由于低的外设时钟频率强加了MinNumClkFastBitB的约束。

下表显示了根据时钟频率、有效载荷大小和可用邮箱数量的不同，CAN FD的最大数据速率。从这个表中可以看到，对于某些情况，如果减少可用邮箱的数量，那么FlexCAN就可以实现高达8Mbps的数据速率。

**表50-32.CAN FD帧上数据阶段的最大CAN比特率**

| 外围时钟频率(MHz) | 有效载荷大小 | 可用邮箱的数量 | 最大数据速率（Mbps） |
|---|---|---|---|
| 40 | 8 | 94 | 6.667 |
| 40 | 8 | 114 | 5.0 |
| 40 | 12 | 117 | 6.667 |
| 40 | 12 | 128 | 5.714 |
| 50 | 12至64岁 | 128 | 6.667 |
| 60 | 8 | 126 | 8.0 |
| 60 | 12 | 128 | 8.0 |
| 67 | 6 | 128 | 8.0 |
| 80 | 3 | 128 | 8.0 |
| 100 | 0 | 128 | 8.0 |

## 50.5.11 操作模式的细节

FlexCAN模块有功能模式和低功耗模式。请参阅操作模式，了解所有操作模式的介绍性说明。下面的小节包含冻结模式和低功耗模式的功能细节。

**注意事项**

FlexCAN不支持CAN总线线路上的 "永久主导
"故障。如果在 "永久主导
"期间进行低功率请求或冻结模式请求，相应的确认将
永远不会被断言。

### 50.5.11.1 冻结模式

这种模式是由CPU通过CAN_MCR寄存器中的HALT位的断言来请求的，或者当芯片进入调试模式时。在这两种情况下，都需要CAN_MCR寄存器中的FRZ位被断言，并且模块不处于低功耗模式。

确认是通过FlexCAN对同一寄存器中FRZ_ACK位的断言获得的。只有当请求和确认条件都满足时，CPU才会认为FlexCAN处于Freeze模式。

当请求冻结模式时，FlexCAN会做以下工作。

- 等待处于间歇、被动错误、总线关闭或闲置状态中。
- 等待所有内部活动，如仲裁、匹配、迁入和迁出完成。一个待定的迁入并不妨碍进入冻结模式。
- 忽略Rx输入引脚，将Tx引脚驱动为隐性引脚
- 停止预分频器，从而停止所有CAN协议活动
- 允许写入错误计数器寄存器，在其他模式下是只读的。
- 设置CAN_MCR中的NOT_RDY和FRZ_ACK位。

在请求冻结模式后，用户必须等待CAN_MCR中的FRZ_ACK位被断言，然后再执行其他操作，否则FlexCAN可能以不可预测的方式运行。在Freeze模式下，除了CAN_CTRL1[CLKSRC]位可以读取但不能写入外，所有内存映射的寄存器都可以访问。

退出冻结模式是通过以下方式之一进行的。

- CPU否定了CAN_MCR寄存器中的FRZ位
- 芯片从调试模式中移除和/或HALT位被否定

协议引擎确认冻结请求被否定后，FRZ_ACK位被否定。当脱离冻结模式时，FlexCAN试图通过等待11个连续的隐性位来重新同步到CAN总线上。

### 50.5.11.2 模块禁用模式

这种低功耗模式通常用于暂时禁用一个完整的FlexCAN块，没有功耗。CPU通过CAN_MCR[MDIS]位的断言来请求该模式，并通过FlexCAN对CAN_MCR[LPMACK]位的断言来获得确认信息。只有当请求和确认条件都满足时，CPU才必须考虑FlexCAN处于禁用模式。

如果模块在冻结模式下被禁用，它请求禁用PE和CHI子模块的时钟，设置LPMACK位并否定FRZACK位。

不建议在假装联网模式下使用模块禁用模式。在设置CAN_MCR[PNET_EN]之前，将MDIS位置负，等待LPMACK置负。

如果模块在传输或接收过程中被禁用，FlexCAN会进行以下操作。

- 等待处于空闲或总线关闭状态，否则等待第三位 -->内切，然后检查其是否为隐性。
- 等待所有内部活动，如仲裁、匹配、迁入和迁出完成。待定的迁入不会被考虑在内。
- 忽略其Rx输入引脚，将其Tx引脚驱动为隐性。
- 关掉通往PE和CHI子模块的时钟
- 设置CAN_MCR中的NOTRDY和LPMACK位。

总线接口单元继续工作，使CPU能够访问内存映射的寄存器，除了Rx邮箱全局屏蔽寄存器、Rx缓冲区14号屏蔽寄存器、Rx缓冲区15号屏蔽寄存器、Rx FIFO全局屏蔽寄存器。当模块处于禁用模式时，不能访问Rx FIFO信息寄存器、消息缓冲器、Rx单个屏蔽寄存器和RAM内的保留字。退出该模式的方法是由CPU否定MDIS位，这将导致FlexCAN请求恢复时钟，并在CAN协议引擎识别到CPU请求的禁用模式的否定后否定LPMACK位。

### 50.5.11.3 停止模式

这是一种系统的低功耗模式，在这种模式下，所有的芯片时钟都可以停止，以最大限度地节省功耗。停止模式由CPU全局请求，通过FlexCAN发出的停止确认信号获得确认。只有当请求和确认条件都满足时，CPU才必须考虑FlexCAN处于停止模式。

如果FlexCAN在冻结模式下收到全局停止模式请求，它会设置LPMACK位，否定FRZACK位，然后向CPU发送停止确认信号，以便全局关闭时钟。

如果在传输或接收过程中请求停止模式，FlexCAN会进行以下操作。

- 等待处于空闲或总线关闭状态，否则就等待间歇期的第三位，并检查其是否为隐性。
- 等待所有内部活动，如仲裁、匹配、迁入和迁出完成。待定的迁入不会被考虑在内。
- 忽略其Rx输入引脚，将其Tx引脚驱动为隐性。
- 设置CAN_MCR中的NOTRDY和LPMACK位。
- 向CPU发送一个停止确认信号，以便它能在全球范围内关闭时钟。

当CPU恢复时钟并删除停止模式的请求时，停止模式就退出了。

在CAN协议引擎识别到停止模式请求的否定后，FlexCAN会否定LPMACK位。然后FlexCAN将等待11个连续的隐性位来同步到CAN总线上。

### 50.5.11.4 假装的网络模式

这是一种特殊的低功率模式，用于接收低功率的唤醒信息。这种模式可以被选择与停止模式一起运行。在进入这些低功率模式之前，CAN_MCR寄存器中的PNET_EN位必须被确认。一旦进入低功耗模式，CHI子块被关闭，而CAN_PE子块保持活动状态，这样Rx接收过程仍然处于活动状态，以过滤传入的信息（参见伪装网络模式下的接收过程），这一点由配置寄存器定义（参见伪装网络控制1寄存器（CAN_CTRL1_PN））。在检测到一个唤醒事件时，会向系统发出一个唤醒中断。

要进入假装联网模式，FlexCAN必须处于正常模式（既不是冻结模式，也不是禁用模式）。在假装联网模式下，FlexCAN在停止模式下保持与CAN BUS的同步。然后，当要求停止模式时，FlexCAN会执行以下步骤。

- 等待处于空闲状态，否则就等待第三位Intermission，然后检查其是否为隐性。
- 设置CAN_MCR寄存器中的LPM_ACK位。
- 要求关闭CHI子模块的时钟，同时保持PE子模块的时钟有效。

FlexCAN可以通过以下方式退出假装联网模式。

- CPU删除停止模式请求。
- FlexCAN将等到总线空闲或中间状态的第三位来否定CAN_MCR[LPM_ACK]位。

上述退出方式可以由FlexCAN检测到唤醒事件并发出相应的中断后的动作触发，也可以由CPU本身被其他方式唤醒后触发。因此，FlexCAN会等待到总线空闲状态或中间状态的第三位来否定CAN_MCR[LPM_ACK]位，并恢复到正常模式。这个过程确保了FlexCAN在退出假装联网模式后将同步到CAN总线上。CPU必须等待CAN_MCR[LPM_ACK]位被否定后才能对FlexCAN执行任何访问。

**注意事项**

关于FD和非FD之间关于此功能的区别的更多信息，见表50-3。

## 50.5.12 中断

该模块有许多中断源：由于消息缓冲器的中断和由于MB的ORed中断、总线关闭、总线关闭完成、错误、错误快速（在BRS位设置的CAN FD格式消息的数据阶段检测到的错误）、唤醒匹配、唤醒超时、Tx警告和Rx警告。

每一个消息缓冲区都可以是一个中断源，如果其相应的IMASK位被设置。对于一个特定的缓冲区，在缓冲区被初始化为传输或接收的假设下，不存在Tx和Rx中断的区别。

每个缓冲区在CAN_IFLAG寄存器中都有一个指定的标志位。当相应的缓冲区完成成功的传输时，该位被设置，当CPU将其写为1时被清除（除非同时产生了另一个中断）。

**注意事项**

必须保证CPU只清除引起当前中断的位。由于这个原因，不能使用位操作指令（BSET）来清除中断标志。这些指令可能会导致意外清除进入当前中断服务程序后设置的中断标志。

如果Rx FIFO被启用（CAN_MCR[RFEN] = 1），DMA被禁用（CAN_MCR[DMA] = 0），那么对应于MBs
0到7的中断有不同的含义。CAN_IFLAG1寄存器的第7位成为 "FIFO溢出"标志；第6位成为 "FIFO警告 "标志，第5位成为 "FIFO可用帧 "标志，第4-0位未使用。更多信息请参见中断标志1寄存器（CAN_IFLAG1）的描述。

如果Rx
FIFO和DMA都被启用（CAN_MCR[RFEN]和CAN_MCR[DMA]=1），FlexCAN就不会产生任何FIFO中断。CAN_IFLAG1寄存器的第5位仍然指示
"FIFO中可用的帧"，并产生一个DMA请求。第7、6、4-0位未使用。

**注意事项**

当CAN FD功能被启用时，FIFO不能被启用。

对于多个MB中断源被OR在一起的组合中断，当任何一个相关的MB（或FIFO，如果适用）产生中断时，就会产生中断。在这种情况下，CPU必须读取CAN_IFLAG寄存器以确定哪个MB或FIFO源引起了中断。

总线关闭、总线关闭完成、错误、快速错误、Tx警告和Rx警告的中断源与MB中断源一样产生中断，并可以从CAN_ESR1寄存器中读取。总线关闭、错误、Tx警告和Rx警告的中断屏蔽位位于CAN_CTRL1寄存器中。

假装联网的中断源（通过匹配标志唤醒和通过超时标志唤醒）可以在CAN_WU_MTC寄存器中读取，相应的中断屏蔽位位于CAN_CTRL1_PN寄存器中。

## 50.5.13 总线接口

CPU对FlexCAN寄存器的访问要遵守以下规则。

- 对监督员寄存器（在监督员模式下表 "模块内存地图
  "中用S/U标识的寄存器或仅用S标识的寄存器）的无限制读写访问会导致访问错误。

- 对已实施的保留地址空间的读写访问会导致访问错误。
- 对当前所有位都是只读的位置进行写访问，会导致访问错误。如果至少有一个位不是只读的，则不会产生访问错误。对位置或其某些位的写入权限可以根据操作模式或临时状态而改变。详情请参考寄存器和位的描述。
- 对未实现的地址空间的读写访问导致访问错误。
- 在低功率模式下对RAM位置的读写访问会导致访问错误。
- RXIMR内存区域有可能被认为是通用内存，可供访问。有两种方法可以做到这一点。
  a. 如果CAN_MCR[IRMQ]被清零，单个掩码（RXIMR）被禁用。在这种情况下，RXIMR存储器区域被认为是通用存储器。
  b. 如果CAN_MCR[MAXMB]的编程值小于可用的MB数，那么未使用的内存空间就可以作为通用的RAM空间使用。请注意，RAM内的保留字不能被使用。举个例子，假设FlexCAN的RAM最多可以支持16MB，CAN_CTRL2[RFFN]为0x0，CAN_MCR[MAXMB]被编程为0。这种情况下，最大的MB数变成了1。RAM从0x0080开始，从0x0080到0x008F的空间被这一个MB使用。从0x0090到0x017F的内存空间可以使用。0x0180到0x087F之间的空间被保留。从0x0880到0x0883的空间被一个单独的掩码使用，掩码寄存器空间中的可用内存将是0x0884到0x08BF。从0x08C0到0x09DF有保留字供内部使用，不能作为一般用途的RAM。一般来说，用于一般用途的自由内存空间只取决于MAXMB。

## 50.6 初始化/应用信息

本节提供初始化FlexCAN模块的说明。

### 50.6.1 FlexCAN初始化顺序

FlexCAN模块可以通过三种方式复位。

- 芯片级硬复位，异步重置所有内存映射的寄存器
- 在MCR中的SOFTRST位，可以同步复位一些内存映射的寄存器。参见表50-5，看看哪些寄存器会受到软复位的影响。
- 芯片级软复位，其效果与MCR中的SOFTRST位相同

软复位是同步的，必须遵循一个跨时钟域的内部请求/确认程序。因此，它可能需要一些时间来完全传播其效果。在软复位待定时，CAN_MCR[SOFTRST]位仍然是正向的，所以软件可以通过轮询这个位来了解复位的完成时间。另外，当时钟在低功耗模式下被关闭时，不能应用软复位。在应用软复位之前，应该退出低功耗模式并恢复时钟。

应在模块处于禁用模式时选择时钟源（见CAN_CTRL1[CLKSRC]位）。在选择了时钟源并启用了模块后（CAN_MCR[MDIS]位被否定），FlexCAN自动进入Freeze模式。在Freeze模式下，FlexCAN与CAN总线不同步，CAN_MCR寄存器中的HALT和FRZ位被设置，内部状态机被禁用，CAN_MCR寄存器中的FRZACK和NOTRDY位被设置。Tx引脚处于隐性状态，FlexCAN不启动任何CAN帧的传输或接收。请注意，消息缓冲器和Rx个体屏蔽寄存器不受复位的影响，所以它们不会被自动初始化。

对于任何配置的改变/初始化，都要求FlexCAN进入Freeze模式（见Freeze模式）。以下是适用于FlexCAN模块的通用初始化顺序。

- 初始化模块配置寄存器（CAN_MCR）。
  - 通过设置IRMQ位启用每MB的单独过滤和接收队列功能
  - 通过设置WRNEN位启用警告中断
  - 如果需要，通过设置SRXDIS位来禁止帧自接收
  - 通过设置RFEN位启用Rx FIFO
  - 如果启用了Rx FIFO并且需要DMA，则设置DMA位
  - 如果需要假装联网模式，请设置PNET_EN位
  - 通过设置AEN位启用中止机制
  - 通过设置LPRIOEN位启用本地优先功能
- 初始化控制1寄存器（CAN_CTRL1）和可选的CAN位定时寄存器（CAN_CBT）。也要初始化CAN FD CAN Bit Timing Register（CAN_FDCBT）。
  - 确定位定时参数。propseg, pseg1, pseg2, rjw

    - 可选择确定位定时参数。epropseg, epseg1, epseg2, erjw
    - 确定CAN FD位定时参数。Fpropseg, Fpseg1, Fpseg2, frjw
    - 通过编程PRESDIV字段和可选的EPRESDIV字段确定比特率
    - 通过编程FPRESDIV字段来确定CAN FD比特率
    - 确定内部仲裁模式（LBUF位）。
- 初始化消息缓冲区
    - 所有消息缓冲器的控制和状态字必须被初始化
    - 如果启用了Rx FIFO，必须初始化ID过滤表。
    - 每个消息缓冲区的其他条目应根据需要进行初始化
- 初始化Rx个体屏蔽寄存器（CAN_RXIMRn）。
- 在CAN_IMASK寄存器（对于所有的MB中断）和CAN_CTRL1 / CAN_CTRL2寄存器（对于总线关闭和错误中断）中设置所需的中断屏蔽位。
- 如果启用了假装联网模式，为选择性唤醒配置必要的寄存器。
- 屏蔽CAN_MCR中的HALT位

在上面列出的最后一步之后，FlexCAN试图与CAN总线同步。

# 第51章
# 同步音频接口（SAI）

## 51.1 特定芯片的SAI信息

S32K148有两个SAI模块的实例。S32K1xx系列的其他产品没有SAI模块。

### 51.1.1 SAI配置

- S32K148有2个SAI模块。SAI0、SAI1。
  - SAI0和SAI1将始终相对于对方异步运行，这意味着SAI0将相对于SAI1异步运行，而SAI1将相对于SAI0异步运行。
  - SAI_BCLK（SAI_TX_BCLK，SAI_RX_BCLK）、SAI_SYNC（SAI_TX_SYNC，SAI_RX_SYNC）和SAI_DATA（SAI_TX_DATA。
    SAI_RX_DATA）引脚在S32K148设备的每个模块实例（SAI0，SAI1）的接收器和发射器之间共享。详情见本参考手册所附的IO信号描述输入复用表。在任何特定的时间，SAI接收器或SAI发射器都应该是激活的，或者它们应该同步运行（每次只有一个激活）。SAI发射器和SAI接收器不应配置为同时运行，配置不一致（如前所述）。
- S32K14x系列的其他器件没有任何SAI模块；只有S32K148器件有SAI模块。

**表51-1.SAI模块的配置**

| 特点 | SAI0模块 | SAI1模块 |
|---|---|---|
| 支持的频道 | 4 | 1 |
| FIFO大小 | 8个字 | 8个字 |
| 框架长度 | 最大=每帧16个字 | 最大=每帧8个字 |
| 帧同步 | I2S、AC97和CODEC/DSP接口 | I2S、AC97和CODEC/DSP接口 |

**注意事项**

- 在STOP/VLPS模式下不支持SAI功能。
- 在使用SAI_TX_DATAx (SAI_Dx)、SAI_SYNC或SAI_BCLK进行输入配置之前，必须对相应的引脚进行GPIO_PDDR配置。

### 51.1.2 芯片特定的寄存器信息

两个SAI模块实例的PARAM寄存器的复位值是不同的。下表显示了特定实例的复位值。

**表51-2.SAI寄存器的复位值**

| 实例 | PARAM复位值 | 评论 |
|---|---|---|
| SAI0 | 0004_0304 | PARAM[DATALINE]反映对4个通道的支持 |
| SAI1 | 0004_0301 | PARAM[DATALINE]反映对1个通道的支持 |

## 51.2 简介

I2S（或I2S）模块提供一个同步音频接口（SAI），支持具有帧同步的全双工串行接口，如I2S、AC97、TDM和编解码/DSP接口。

### 51.2.1 特点

请注意，有些功能不是所有的SAI实例都支持的；请参见本章第一节中的芯片具体信息。

- 带有独立位时钟和帧同步的发射器，支持4条数据线
- 带有独立位时钟和帧同步的接收器，支持4条数据线
- 每条数据线可以支持最大16个字的帧大小
- 字大小在8位和32位之间
- 为框架中的第一个字和其余的字分别配置的字数大小
- 每个发送和接收数据线都有异步的8×32位FIFO
- 支持FIFO错误后的优雅重启
- 支持FIFO错误后自动重启，无需软件干预
- 支持将8位和16位数据打包到每个32位FIFO字中

- 支持将多个数据线FIFO合并为单个数据线FIFO

## 51.2.2 框图

下面的方框图也显示了模块的时钟。

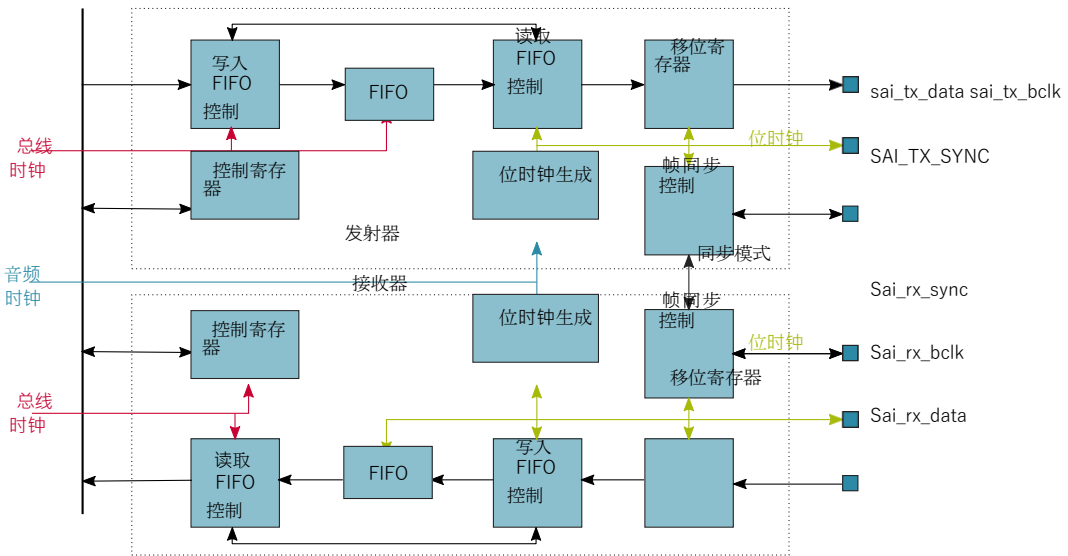



**图51-1.I2S/SAI框图**

## 51.2.3 操作模式

模块电源模式包括运行模式，和调试模式。

### 51.2.3.1 运行模式

在运行模式下，SAI发射器和接收器正常运行。

### 51.2.3.2 调试模式

在调试模式下，只要调试启用位被设置，SAI发射器和/或接收器可以继续工作。当TCSR[DBGE]或RCSR[DBGE]位清零并进入调试模式时，SAI在完成当前的发送或接收帧后被禁用。发送器和接收器的位时钟不受调试模式的影响。

## 51.3 外部信号

| 命名 | 职能 | 输入/输出 |
| --- | --- | --- |
| SAI_TX_BCLK | 发送位时钟。位时钟在外部产生时是一个输入，在内部产生时是一个输出。 | 输入/输出 |
| SAI_TX_SYNC | 发送帧同步。当外部产生时，帧同步是一个由位时钟同步采样的输入，当内部产生时，帧同步是一个由位时钟同步产生的输出。 | 输入/输出 |
| sai_tx_data[3:0] | 发送数据。发送数据由位时钟同步产生，每当不发送一个字时就会三态化。 | O |
| SAI_RX_BCLK | 接收比特时钟。位时钟在外部产生时是一个输入，在内部产生时是一个输出。 | 输入/输出 |
| 呼叫中心：SAI_RX_SYNC | 接收帧同步。当外部产生时，帧同步是一个由位时钟同步采样的输入，当内部产生时，帧同步是一个由位时钟同步产生的输出。 | 输入/输出 |
| sai_rx_data[3:0] | 接收数据。接收数据是由位时钟同步采样的。 | I |
| 呼叫中心：MCLK_EN | 音频主时钟。 | I |

## 51.4 内存图和寄存器定义

对偏移量为0x100及以上的地址进行读或写访问将导致总线错误。

### 51.4.1 I2S寄存器描述

#### 51.4.1.1 I2S内存图

SAI0基础地址。4005_4000h

| 偏移 | 注册 | 宽度<br>(单位：比特) | 访问 | 重置值 |
|---|---|---|---|---|
| 0h | 版本标识寄存器（VERID） | 32 | RO | 0300_0000h |
| 4h | 参数寄存器（PARAM）。 | 32 | RO | 0004_0304h |
| 8h | SAI传输控制寄存器（TCSR） | 32 | RW | 0000_0000h |
| 丘吉尔 | SAI传输配置1寄存器（TCR1）。 | 32 | RW | 0000_0000h |
| 10h | SAI传输配置2寄存器（TCR2）。 | 32 | RW | 0000_0000h |
| 14h | SAI传输配置3寄存器（TCR3）。 | 32 | RW | 0000_0000h |
| 18h | SAI传输配置4寄存器（TCR4）。 | 32 | RW | 0000_0000h |
| 1Ch | SAI传输配置5寄存器（TCR5）。 | 32 | RW | 0000_0000h |
| 20h - 2Ch | SAI传输数据寄存器（TDR0 - TDR3）。 | 32 | 钨钢 | 0000_0000h |
| 40h - 4Ch | SAI 发送 FIFO 寄存器（TFR0 - TFR3）。 | 32 | RO | 0000_0000h |
| 60h | SAI传输掩码寄存器（TMR） | 32 | RW | 0000_0000h |
| 88h | SAI接收控制寄存器（RCSR）。 | 32 | RW | 0000_0000h |
| 8Ch | SAI接收配置1寄存器（RCR1）。 | 32 | RW | 0000_0000h |
| 90h | SAI接收配置2寄存器（RCR2）。 | 32 | RW | 0000_0000h |
| 94h | SAI接收配置3寄存器（RCR3）。 | 32 | RW | 0000_0000h |
| 98h | SAI接收配置4寄存器（RCR4）。 | 32 | RW | 0000_0000h |
| 9Ch | SAI接收配置5寄存器（RCR5）。 | 32 | RW | 0000_0000h |
| A0h - ACh | SAI接收数据寄存器（RDR0 - RDR3）。 | 32 | RO | 0000_0000h |
| C0h - CCh | SAI接收FIFO寄存器（RFR0 - RFR3）。 | 32 | RO | 0000_0000h |
| E0h | SAI接收屏蔽寄存器（RMR）。 | 32 | RW | 0000_0000h |

## 51.4.1.2 版本标识寄存器（VERID）

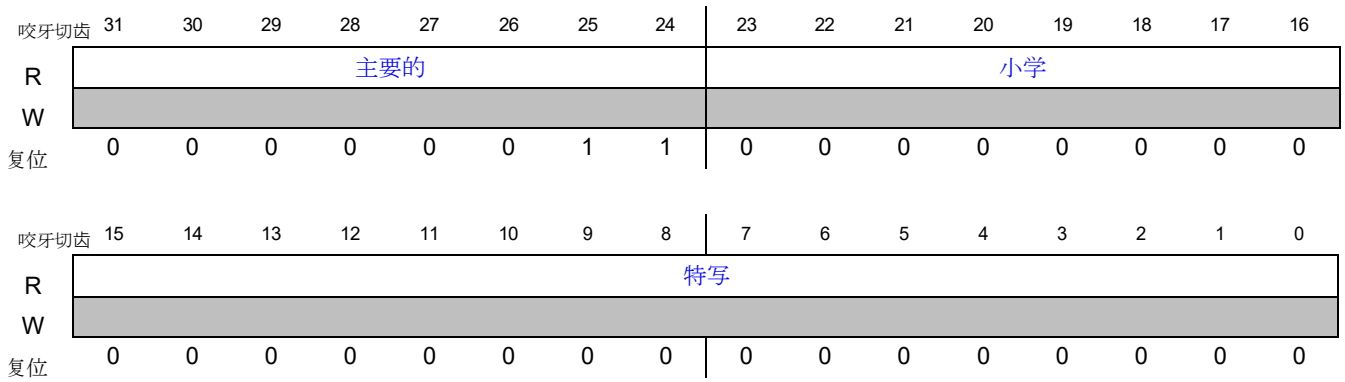
### 51.4.1.2.1 偏移

| 注册 | 偏移 |
|---|---|
| 虚实对照 | 0h |

### 51.4.1.2.2 职能

.

#### 51.4.1.2.3 图示




#### 51.4.1.2.4 领域

| 场地 | 职能 |
|---|---|
| 31-24<br>主要的 | 主要版本号<br>这个只读字段返回该规范的主要版本号。 |
| 23-16<br>小学 | 次要版本号<br>这个只读字段返回规范的次要版本号。 |
| 15-0<br>特写 | 特点规格号<br>这个只读字段返回特征集编号。0000000000000000b - 标准特征集。 |

### 51.4.1.3 参数寄存器（PARAM）。

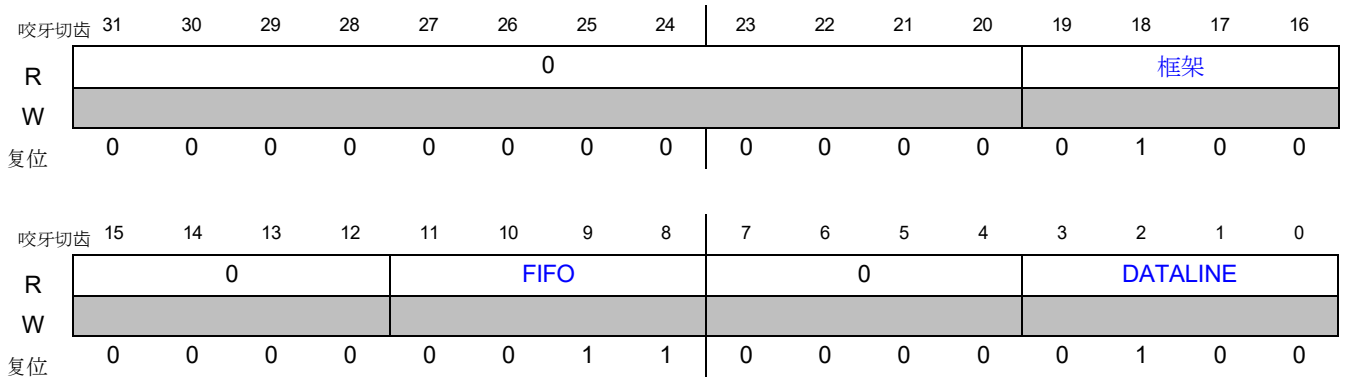
#### 51.4.1.3.1 偏移

| 注册 | 偏移 |
|---|---|
| 朴实无华 | 4h |

#### 51.4.1.3.2 职能

.

#### 51.4.1.3.3 图示




#### 51.4.1.3.4 领域

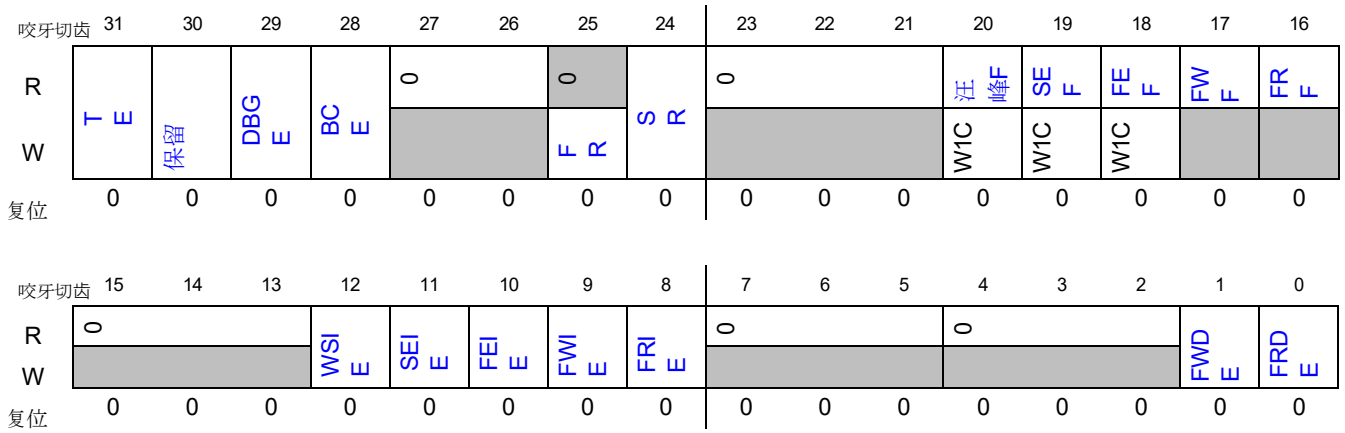
| 场地 | 职能 |
|---|---|
| 31-20<br>- | 保留 |
| 19-16<br>框架 | 框架尺寸<br>每一帧的最大槽位数为2^FRAME。 |
| 15-12<br>- | 保留 |
| 11-8<br>FIFO | FIFO大小<br>每个FIFO的字数为2^FIFO。 |
| 7-4<br>- | 保留 |
| 3-0<br>DATALINE | 数据线的数量<br>实施的数据线的数量。 |

### 51.4.1.4 SAI传输控制寄存器（TCSR）

#### 51.4.1.4.1 偏移

| 注册 | 偏移 |
|---|---|
| TCSR | 8h |

### 51.4.1.4.2 图示




### 51.4.1.4.3 领域

| 场地 | 职能 |
|---|---|
| 31<br>TE | 发射器启用<br>启用/停用发射机。当软件清除这个字段时，发射器保持启用，这个位保持设置，直到当前帧结束。<br>0b - 发送器被禁用。<br>1b - 发射器被启用，或发射器已被禁用且尚未达到帧的终点。 |
| 30<br>- | 保留。软件只能给这个保留位写0。 |
| 29<br>DBGE | 调试启用<br>启用/禁用调试模式下的发送器操作。发送位时钟不受调试模式的影响。<br>0b - 发送器在调试模式下被禁用，在完成当前帧后。1b - 发送器在调试模式下被启用。 |
| 28<br>BCE | 位时钟启用<br>启用发送位时钟，与TE分开。只要TE被设置，这个字段就会自动设置。当软件清除这个字段时，发送位时钟保持启用，这个位保持设置，直到当前帧的结束。<br>0b - 传输位时钟被禁用。1b - 发送位时钟被启用。 |
| 27-26<br>- | 保留 |
| 25<br>共和国 | FIFO复位<br>重置FIFO指针。读取这个字段将总是返回零。只有当发射器被禁用或FIFO错误标志被设置时，FIFO指针才应被重置。<br>0b - 没有影响。<br>1b - FIFO复位。 |

| | |
|---|---|
| 24<br>ĀĀĀ | 软件复位<br><br>当设置时，重置内部发射器逻辑，包括FIFO指针。软件可见的寄存器不受影响，除了状态寄存器。<br>0b - 没有影响。 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| | 1b - 软件复位。 |
| 23-21<br>- | 保留 |
| 20<br>WSF | 字的开始标志<br>表示已经检测到配置的字的开始。给这个字段写一个逻辑1来清除这个标志。<br>0b - 未检测到字的开始。1b - 检测到字的开始。 |
| 19<br>SEF | 同步错误标志<br>表示已经检测到外部生成的帧同步中的错误。给这个字段写一个逻辑1来清除这个标志。<br>0b - 未检测到同步错误。1b - 检测到帧同步错误。 |
| 18<br>FEF | FIFO错误标志<br>表示一个已启用的发送FIFO有不足。给这个字段写一个逻辑1来清除这个标志。<br>0b - 未检测到发送不足。1b - 检测到传输不足。 |
| 17<br>基金会 | FIFO警告标志<br>表示一个启用的发送FIFO是空的。<br>0b - 没有启用发送FIFO是空的。1b - 启用的发送FIFO是空的。 |
| 16<br>基金会 | FIFO请求标志<br>表示启用的发送通道FIFO中的字数小于或等于发送FIFO水印。<br>0b - 未达到发送FIFO水印。1b - 已达到发送FIFO的水印。 |
| 15-13<br>- | 保留 |
| 12<br>宇宙飞船 | 字启动中断启用<br>启用/禁用字启动中断。<br>0b - 禁用中断。<br>1b - 启用中断。 |
| 11<br>SEIE | 同步错误中断启用<br>启用/停用同步错误中断。<br>0b - 禁用中断。<br>1b - 启用中断。 |
| 10<br>基金会 | FIFO错误中断启用<br>启用/停用FIFO错误中断。<br>0b - 禁用该中断。1b - 启用该中断。 |

| | |
|---|---|
| 9<br>FWIE | FIFO警告中断启用<br>启用/禁用FIFO警告中断。<br>0b - 禁用该中断。1b - 启用该中断。 |
| 8<br>基金会 | FIFO请求中断启用<br>启用/禁用FIFO请求中断。 |

*表继续在下一页...*

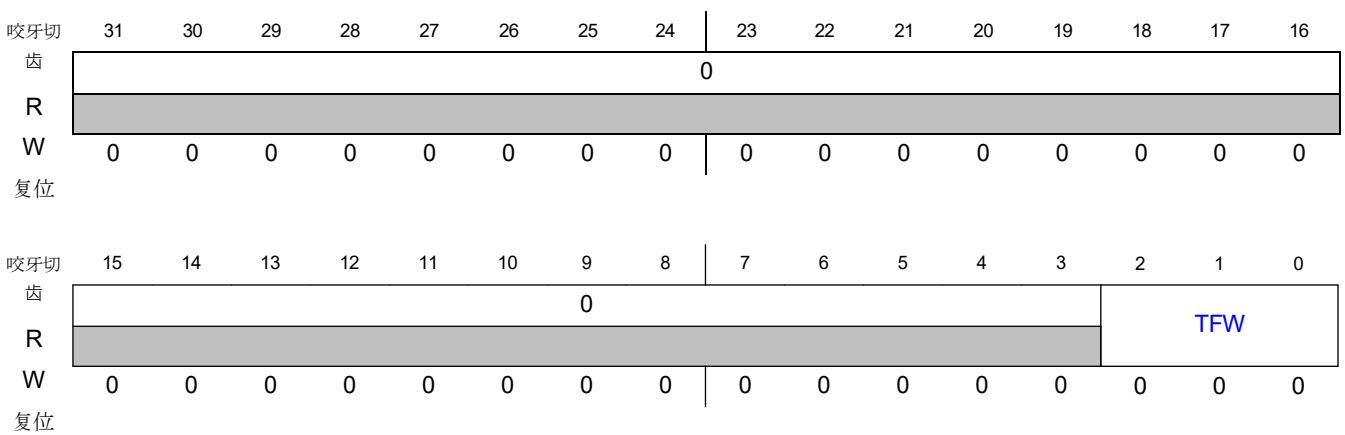
| 场地 | 职能 |
| --- | --- |
| | 0b - 禁用该中断。1b - 启用<br>该中断。 |
| 7-5<br>- | 保留 |
| 4-2<br>- | 保留 |
| 1<br>FWDE | FIFO警告DMA启用<br>启用/停用DMA请求。<br>0b - 禁用DMA请求。1b - 启用<br>DMA请求。 |
| 0<br>FRDE | FIFO请求DMA启用<br>启用/停用DMA请求。<br>0b - 禁用DMA请求。1b - 启用<br>DMA请求。 |

## 51.4.1.5 SAI传输配置1寄存器（TCR1）。

### 51.4.1.5.1 偏移

| 注册 | 偏移 |
| --- | --- |
| TCR1 | 丘吉尔 |

### 51.4.1.5.2 图示



#### 51.4.1.5.3 领域

| 场地 | 职能 |
|---|---|
| 31-3<br>- | 保留 |
| 2-0<br>TFW | 发送FIFO水印<br>配置所有启用的传输通道的水印水平。 |

### 51.4.1.6 SAI传输配置2寄存器（TCR2）。

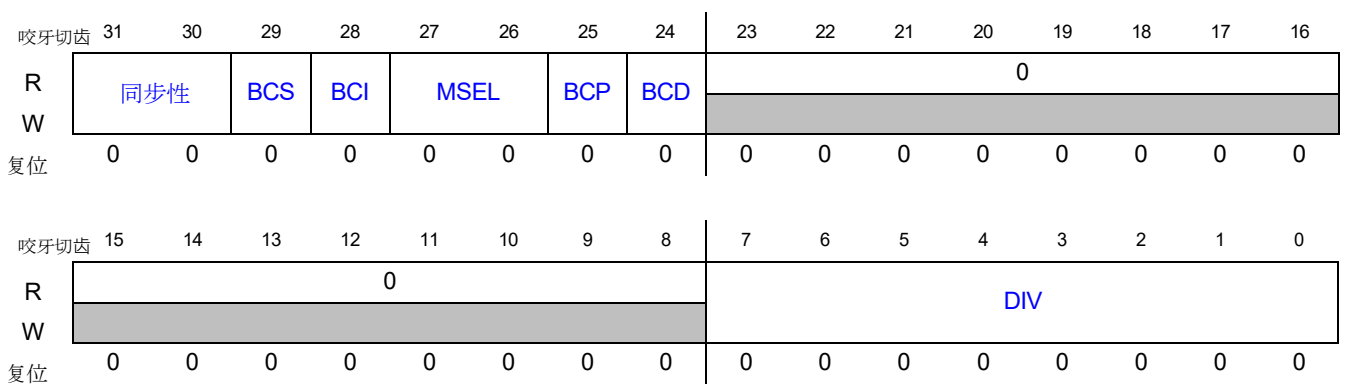
#### 51.4.1.6.1 偏移

| 注册 | 偏移 |
|---|---|
| TCR2 | 10h |

#### 51.4.1.6.2 职能

当TCSR[TE]被设置时，这个寄存器不能被改变。

#### 51.4.1.6.3 图示




#### 51.4.1.6.4 领域

*表继续在下一页...*

| 场地 | 职能 |
| --- | --- |
| 31-30<br>同步性 | 同步模式<br>在异步和同步操作模式之间进行配置。当配置为同步操作模式时，接收器必须配置为异步操作。<br>00b - 异步模式。 |

| 场地 | 职能 |
| --- | --- |
| | 01b - 与接收器同步。10b - 保留。<br>11b - 保留。 |
| 29<br>BCS | 位时钟互换<br>这个字段交换了发射器使用的位时钟。当发射器被配置为异步模式且该位被设置时，发射器由接收器的位时钟（SAI_RX_BCLK）提供时钟。这允许发射器和接收器共享相同的位时钟，但发射器继续使用发射帧同步（SAI_TX_SYNC）。<br>当发射器被配置为同步模式时，发射器BCS字段和接收器BCS字段必须被设置为相同的值。当两者都被设置时，发射器和接收器都由发射器位时钟（SAI_TX_BCLK）提供时钟，但使用接收器帧同步（SAI_RX_SYNC）。<br>0b - 使用正常的位时钟源。1b - 调换<br>位时钟源。 |
| 28<br>BCI | 位时钟输入<br>当这个字段被设置并在同步或异步模式下使用内部产生的位时钟时，发射器实际使用的位时钟被垫子输出延迟（发射器被垫子输入时钟，就像时钟由外部产生一样）。这具有减少数据输入设置时间的效果，但增加了数据输出有效时间。<br>当该位被设置时，发射器应使用数据手册中的从属模式计时。在同步模式下，该位允许发射器使用数据表中的从属模式计时，而接收器使用主模式计时。当配置为外部产生的位时钟时，这个字段没有影响。<br>0b - 没有影响。<br>1b - 内部逻辑的时钟就像外部产生的位时钟一样。 |
| 27-26<br>MSEL | 选择MCLK<br>选择用于生成内部产生的比特时钟的音频主时钟选项。当配置为外部产生的位时钟时，这个字段没有影响。<br>**注意：**根据设备的不同，一些主时钟选项可能无法使用。关于每个选项的含义，请参见芯片的具体信息。<br>00b - 选择总线时钟。<br>01b - 选择了主时钟(MCLK)1选项。10b - 选择了主时钟(MCLK)2选项。11b - 已选择主时钟(MCLK)3选项。 |
| 25<br>BCP | 位时钟极性<br>配置位时钟的极性。<br>0b - 位时钟为高电平，驱动输出在上升沿，采样输入在下降沿。1b - 位时钟为低电平，驱动输出在下降沿，采样输入在上升沿。 |
| 24<br>BCD | 位时钟方向<br>配置位时钟的方向。<br>0b - 位时钟在从属模式下由外部产生。1b - 位时钟在主模式下由内部产生。 |

*表继续在下一页...*

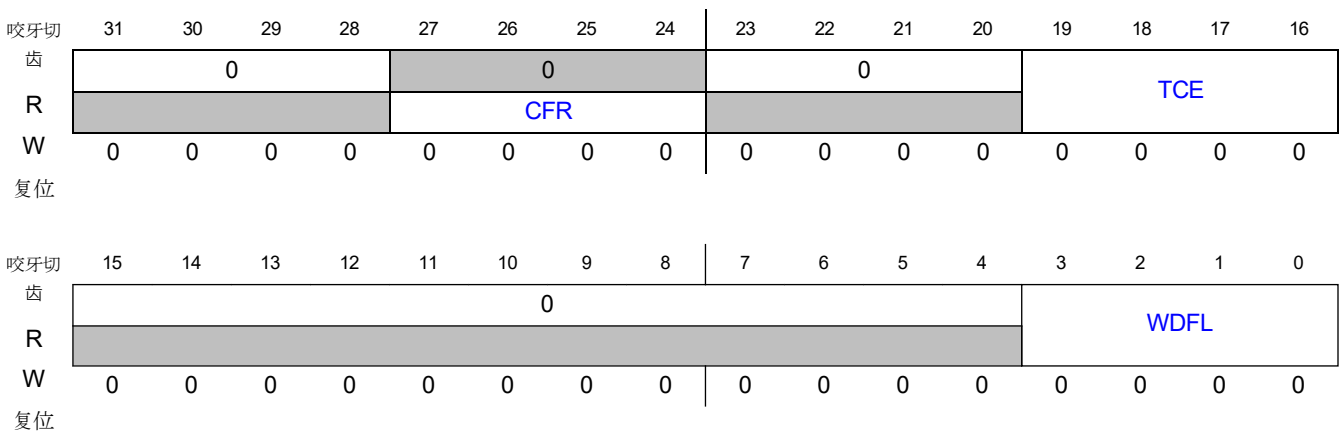
| | |
|---|---|
| 23-8<br>- | 保留 |
| 7-0<br>DIV | 位时钟分频<br>当配置为内部位时钟时，将音频主时钟向下划分以产生位时钟。分割值为（DIV+1）*2。 |

## 51.4.1.7 SAI传输配置3寄存器（TCR3）。

### 51.4.1.7.1 偏移

| 注册 | 偏移 |
|---|---|
| TCR3 | 14h |

### 51.4.1.7.2 图示




### 51.4.1.7.3 领域

| 场地 | 职能 |
|---|---|
| 31-28<br>- | 保留 |
| 27-24<br>CFR | 通道FIFO复位<br>重置一个特定通道的FIFO指针。读取这个字段将总是返回0。只有当一个通道被禁用或FIFO错误标志被设置时，FIFO指针才应被重置。<br>CFR字段的宽度=发送通道的数量（称之为N）。例如，如果CFR是2位宽，那么第24位是指发送通道1 FIFO指针，第25位是指发送通道2 FIFO指针。设置位24重置发送通道1 FIFO指针，设置位25启用发送通道2 FIFO指针。设置位 N 将重置发送通道 N FIFO 指针。<br>0b - 没有影响。<br>1b - 发送数据通道N FIFO被重置。 |

*表继续在下一页...*

| | |
|---|---|
| 23-20<br>- | 保留 |
| 19-16<br>TCE | 发送通道启用<br>启用相应的数据通道进行传输操作。改变TCE字段将立即生效，以产生FIFO请求和警告标志，但在每一帧结束时进行传输操作。 |

| 场地 | 职能 |
|---|---|
| | TCE字段的宽度=发送通道的数量（称之为N）。例如，如果TCE字段是2位宽，那么第16位是指发送通道1，第17位是指发送通道2。设置第16位使发射通道1，设置第17位使发射通道2。设置位N将启用传输通道N。<br>0b - 发送数据通道N被禁用。1b - 发送数据通道N被启用。 |
| 15-4<br>- | 保留 |
| 3-0<br>WDFL | 字旗配置<br>配置哪个字设置字的开始标志。写入的值必须比字数少一个。例如，写0可以配置帧中的第一个字。当配置为一个大于TCR4[FRSZ]的值时，那么字的开始标志永远不会被设置。 |

## 51.4.1.8 SAI传输配置4寄存器（TCR4）。

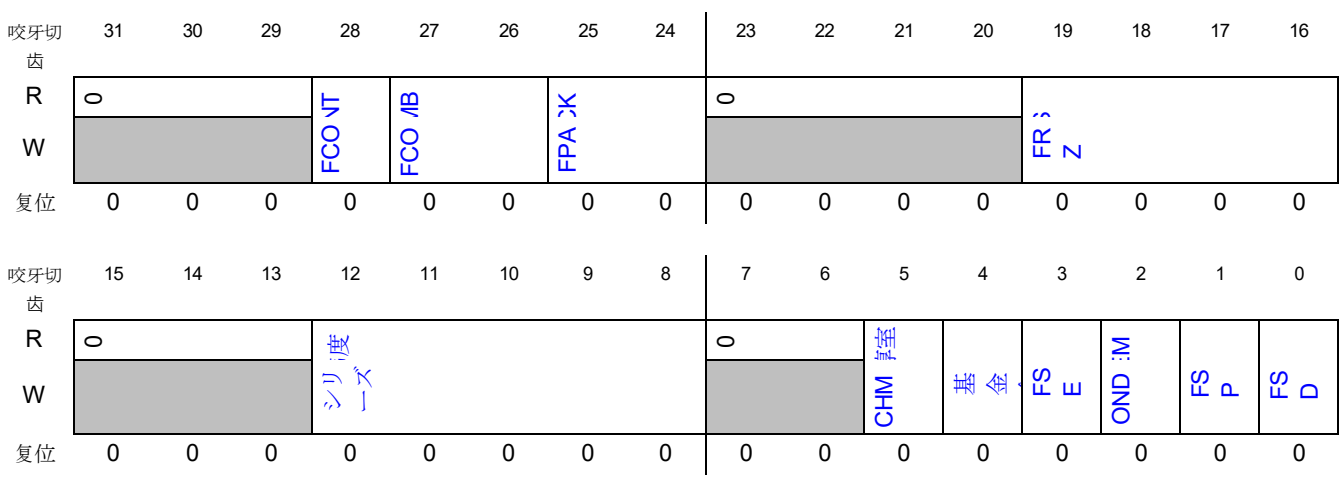
### 51.4.1.8.1 偏移

| 注册 | 偏移 |
|---|---|
| TCR4 | 18h |

### 51.4.1.8.2 职能

当TCSR[TE]被设置时，这个寄存器不能被改变。

### 51.4.1.8.3 图示




*表继续在下一页...*

### 51.4.1.8.4 领域

| 场地 | 职能 |
|---|---|
| 31-29<br>- | 保留 |
| 28<br>FCONT | 错误时FIFO继续<br>配置SAI在检测到FIFO错误后何时继续传输。<br>0b - 在FIFO错误时，SAI将在FIFO错误标志被清除后从下一帧开始继续。<br>1b - 在FIFO错误时，在FIFO警告标志被清除后，SAI将从导致FIFO错误的同一个字继续设置。 |
| 27-26<br>FCOMB | FIFO组合模式<br>当FIFO组合模式对FIFO写启用时，软件对任何FIFO数据寄存器的写将在启用的数据通道FIFO之间交替进行。例如，如果两个数据通道被启用，那么第一次写入将被执行到第一个启用的数据通道FIFO，第二次写入将被执行到第二个启用的数据通道FIFO。重置FIFO或禁用FIFO组合模式的FIFO写入将重置指针回到第一个启用的数据通道。<br>当FIFO组合模式被启用用于从传输移位寄存器的FIFO读取时，传输数据通道的输出将在启用的数据通道FIFO之间交替进行。例如，如果两个数据通道被启用，那么第一个未屏蔽的字将从第一个启用的数据通道FIFO传输，第二个未屏蔽的字将从第二个启用的数据通道FIFO传输。由于帧的第一个字总是从第一个启用的数据通道FIFO中传输，因此建议每帧的未屏蔽字的数量均匀地除以启用的数据通道的数量。<br>00b - 禁用FIFO组合模式。<br>01b - 在FIFO读时启用FIFO组合模式（从传输移位寄存器）。10b - 在FIFO写时启用FIFO组合模式（通过软件）。<br>11b - 在FIFO读取（来自传输移位寄存器）和写入（通过软件）时启用FIFO组合模式。 |
| 25-24<br>ÃÃÃ | FIFO打包模式<br>启用将8位数据或16位数据打包到每个32位FIFO字中。如果字的大小大于8-位或16-位，那么只从FIFO中加载第一个8-位或16-位。每一帧中的第一个字总是以一个新的32位FIFO字开始，第一个移位必须配置在第一个打包字内。当FIFO打包被启用时，FIFO写指针只有在软件写满32位FIFO字后才会递增。<br>00b - 禁用FIFO包装 01b - 保留<br>10b - 启用8位FIFO包装 11b - 启用16位FIFO包装 |
| 23-20<br>- | 保留 |
| 19-16<br>FRSZ | 框架尺寸<br>配置每个帧中的字数。写入的值必须比帧中的字数少一个。例如，写0表示每帧有一个字。支持的最大帧大小是16个字。 |
| 15-13<br>- | 保留 |

| | |
|---|---|
| 12-8<br>シリーズワールド | 同步宽度 |

表继续在下一页...

| 场地 | 职能 |
| --- | --- |
| | 配置帧同步的长度，单位为比特时钟数。写入的值必须比位时钟的数量少一个。例如，如果写成0，则帧同步只断言一个比特时钟。配置的同步宽度不能长于帧的第一个字。 |
| 7-6<br>- | 保留 |
| 5<br>CHMOD | 频道模式<br>配置发送数据引脚是否配置为TDM模式或输出模式。<br>0b - TDM模式，当槽位被屏蔽或通道被禁用时，发送数据引脚是三态的。1b - 输出模式，当槽位被屏蔽或通道被禁用时，发送数据引脚从不被三振，并将输出零。 |
| 4<br>基金会 | MSB优先<br>配置是先传输LSB还是MSB。<br>0b - 先传输LSB。1b - MSB<br>先被传送。 |
| 3<br>FSE | 帧同步早期<br>0b - 帧同步与帧的第一位一起断电。<br>1b - 帧同步在帧的第一位之前断言。 |
| 2<br>ONDEM | 按需模式<br>当设置时，并且帧同步是在内部产生的，只有当FIFO警告标志清除时才会产生帧同步。<br>0b - 持续产生内部帧同步。<br>1b - 当FIFO警告标志清除时产生内部帧同步。 |
| 1<br>FSP | 帧同步极性<br>配置帧同步的极性。<br>0b - 帧同步为高电平有效。1b -<br>帧同步为低电平有效。 |
| 0<br>FSD | 帧同步方向<br>配置帧同步的方向。<br>0b - 帧同步在从属模式下由外部产生。1b - 帧同步在主模式下由内部产生。 |

# 51.4.1.9 SAI传输配置5寄存器（TCR5）。

## 51.4.1.9.1 偏移

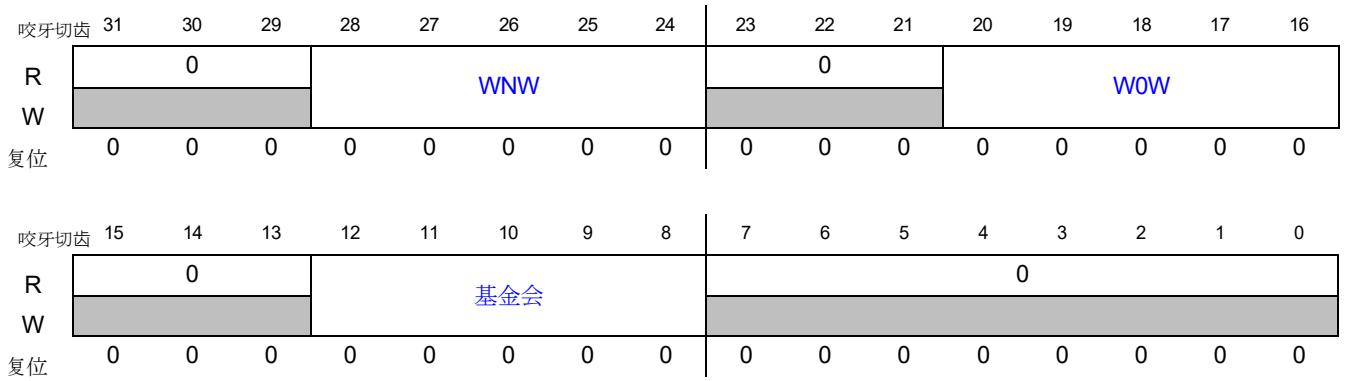
| 注册 | 偏移 |
| --- | --- |
| TCR5 | 1Ch |

#### 51.4.1.9.2 职能

当TCSR[TE]被设置时，这个寄存器不能被改变。

*表继续在下一页...*

### 51.4.1.9.3 图示




### 51.4.1.9.4 领域

| 场地 | 职能 |
|---|---|
| 31-29<br>- | 保留 |
| 28-24<br>WNW | 字N宽度<br>配置每个字的位数，除了帧中的第一个字外，每个字的位数。写入的值必须比每个字的比特数少一个。不支持字宽小于8位的情况。 |
| 23-21<br>- | 保留 |
| 20-16<br>W0W | 字0宽度<br>配置每个帧中第一个字的位数。写入的值必须比第一个字的位数少一个。如果每帧只有一个字，则不支持小于8位的字宽。 |
| 15-13<br>- | 保留 |
| 12-8<br>基金会 | 第一比特移位<br>配置帧中每个字的第一个传输位的索引。如果配置为MSB优先，则传输的下一个位的索引比传输的当前位少一个。如果配置为LSB优先，则传输的下一个位的索引比传输的当前位多一个。当配置为MSB优先时，写入的值必须大于或等于字宽。当配置为LSB优先时，写入的值必须小于或等于31字的宽度。 |
| 7-0<br>- | 保留 |

## 51.4.1.10 SAI传输数据寄存器（TDR0 - TDR3）。

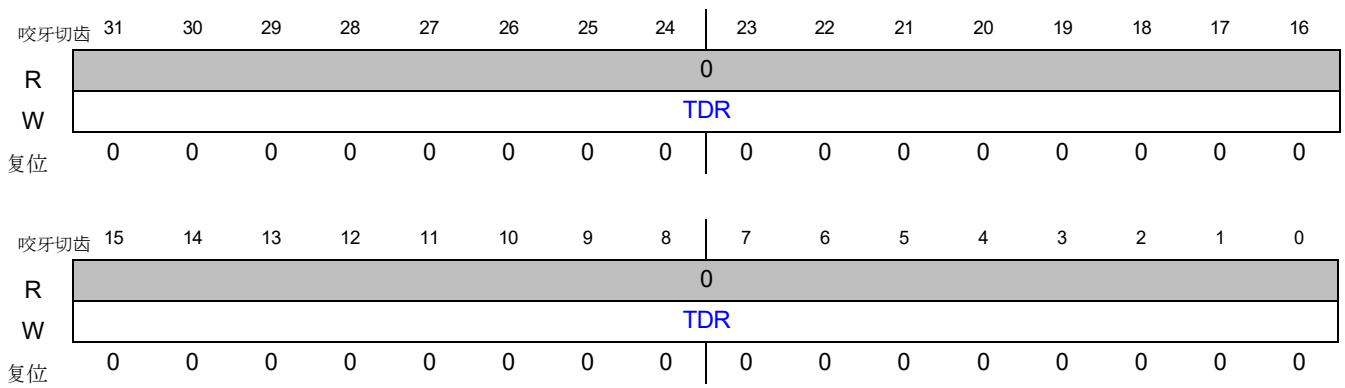
## 51.4.1. 偏移

1722.1

对于a=0到3。

| 注册 | 偏移 |
|---|---|
| TDRa | 20h + (a × 4h) |

### 51.4.1.10.2 图示




### 51.4.1.10.3 领域

| 场地 | 职能 |
|---|---|
| 31-0<br>TDR | 发送数据寄存器<br>当发送FIFO未满时，对该寄存器的写入将把写入的数据推入发送数据FIFO。当发送FIFO满时，对该寄存器的写入将被忽略。 |

## 51.4.1.11 SAI 发送 FIFO 寄存器（TFR0 - TFR3）。

### 51.4.1.11.1 偏移

对于a=0到3。

| 注册 | 偏移 |
|---|---|
| TFRa | 40h + (a × 4h) |

### 51.4.1.11.2 职能

读和写指针的MSB被用来区分FIFO满和空的情况。如果读和写指针相同，那么FIFO是空的。如果读和写指针除了MSB以外都是相同的，那么FIFO是满的。

### 51.4.1.11.3 图示

| 咬牙切齿 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | WCP | 0 | | | | | | | | | | | WFP | | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 咬牙切齿 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | RFP | | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

### 51.4.1.11.4 领域

| 场地 | 职能 |
|---|---|
| 31<br>WCP | 写入通道指针<br>当FIFO组合模式被启用写入时，表示该数据通道是下一个要写入的FIFO。<br>0b - 没有影响。<br>1b - FIFO组合被启用，用于FIFO的写入，这个FIFO将在下一次FIFO写入时被写入。 |
| 30-20<br>- | 保留 |
| 19-16<br>WFP | 写FIFO指针<br>发送数据通道的FIFO写指针。 |
| 15-4<br>- | 保留 |
| 3-0<br>RFP | 读取FIFO指针<br>发送数据通道的FIFO读取指针。 |

## 51.4.1.12 SAI传输掩码寄存器（TMR）

### 51.4.1.12.1 偏移

| 注册 | 偏移 |
|---|---|
| TMR | 60h |

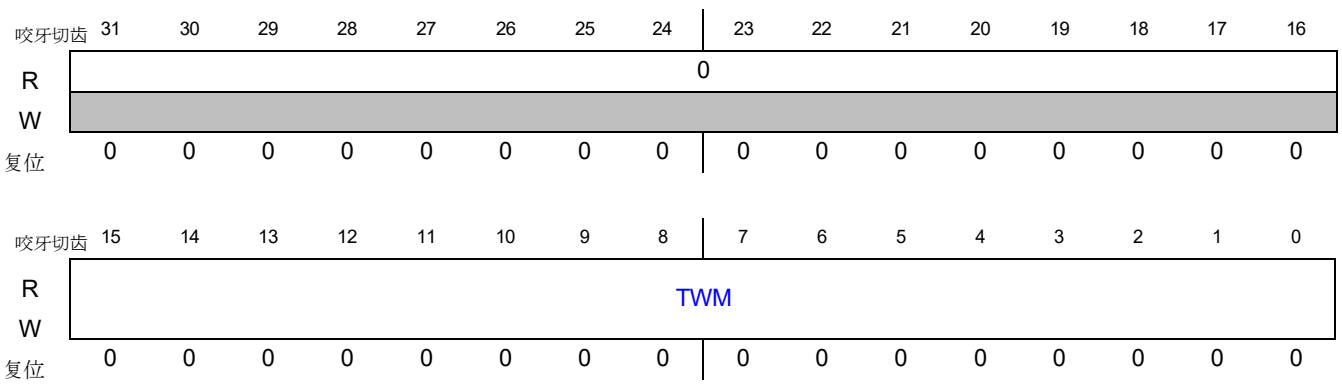
### 51.4.1.12.2 职能

这个寄存器是双缓冲的，并进行更新。

1. 当TCSR[TE]被首次设置时
2. 在每一帧结束时。

这允许每一帧中的遮蔽词在每一帧中发生变化。

### 51.4.1.12.3 图示




### 51.4.1.12.4 领域

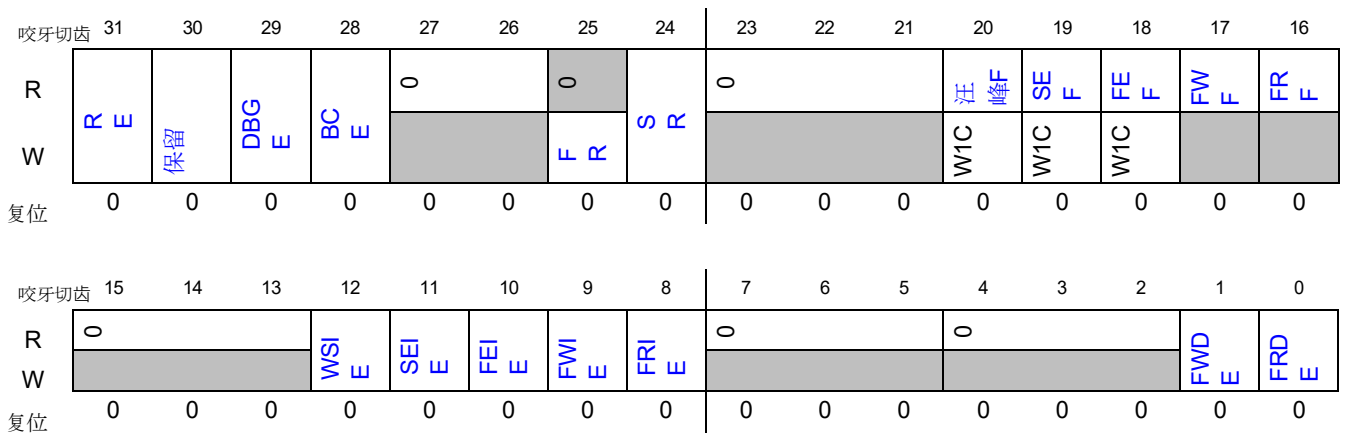
| 场地 | 职能 |
|---|---|
| 31-16<br>- | 保留 |
| 15-0<br>TWM | 发送字屏蔽<br>配置帧中相应字的发送字是否被屏蔽（发送数据引脚为三态或驱动零，发送数据不从FIFO读取）。<br>0000000000000000b - 字N被启用。<br>0000000000000001b - 字N被屏蔽。发送数据引脚在被屏蔽时是三态的或驱动零。 |

# 51.4.1.13 SAI接收控制寄存器（RCSR）。

## 51.4.1.13.1 偏移

| 注册 | 偏移 |
|---|---|
| RCSR | 88h |

## 51.4.1.13.2 图示




## 51.4.1.13.3 领域

| 场地 | 职能 |
|---|---|
| 31<br>雇员 | 接收器启用<br>启用/停用接收器。当软件清除该字段时，接收器保持启用状态，该位保持设置，直到当前帧的结束。<br>0b - 接收器被禁用。<br>1b - 接收器已启用，或接收器已被禁用且尚未达到帧结束。 |
| 30<br>- | 保留。软件只能给这个保留位写0。 |
| 29<br>DBGE | 调试启用<br>启用/禁用调试模式下的接收器操作。接收位时钟不受调试模式的影响。<br>0b - 接收器在调试模式下被禁用，在完成当前帧后。1b - 接收器在调试模式下被启用。 |
| 28<br>BCE | 位时钟启用<br>启用接收位时钟，与RE分开。只要设置了RE，这个字段就会自动设置。当软件清除该字段时，接收位时钟保持启用，该字段保持设置，直到当前帧结束。<br>0b - 接收位时钟被禁用。1b - 接收 |

| | |
|---|---|
| | 位时钟被启用。 |
| 27-26 | 保留 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| - | |
| 25<br>共和国 | FIFO复位<br>重置FIFO指针。读取这个字段将总是返回0。只有当接收器被禁用或FIFO错误标志被设置时，FIFO指针才应被重置。<br>0b - 没有影响。<br>1b - FIFO复位<br>。 |
| 24<br>ÃÃÃ | 软件复位<br>重置内部接收器逻辑，包括FIFO指针。软件可见的寄存器不受影响，除了状态寄存器。<br>0b - 没有影响。<br>1b - 软件复位。 |
| 23-21<br>- | 保留 |
| 20<br>WSF | 字的开始标志<br>表示已经检测到配置的字的开始。给这个字段写一个逻辑1来清除这个标志。<br>0b - 未检测到字的开始。1b - 检<br>测到字的开始。 |
| 19<br>SEF | 同步错误标志<br>表示已经检测到外部生成的帧同步中的错误。给这个字段写一个逻辑1来清除这个标志。<br>0b - 未检测到同步错误。1b -<br>检测到帧同步错误。 |
| 18<br>FEF | FIFO错误标志<br>表示一个启用的接收FIFO已经溢出。给这个字段写一个逻辑1来清除这个标志。<br>0b - 未检测到接收溢出。1b - 检测<br>到接收溢出。 |
| 17<br>基金会 | FIFO警告标志<br>表示一个启用的接收FIFO已满。<br>0b - 没有启用的接收FIFO是满的。1b<br>- 启用的接收FIFO已满。 |
| 16<br>基金会 | FIFO请求标志<br>表示启用的接收通道FIFO中的字数大于接收FIFO水印。<br>0b - 未达到接收FIFO水印。<br>1b - 已达到接收FIFO水印。 |
| 15-13<br>- | 保留 |
| 12<br>宇宙飞船 | 字启动中断启用<br>启用/禁用字启动中断。<br>0b - 禁用中断。<br>1b - 启用中断。 |

| | |
|---|---|
| 11<br>SEIE | 同步错误中断启用<br><br>启用/停用同步错误中断。<br>0b - 禁用中断。<br>1b - 启用中断。 |
| 10 | FIFO错误中断启用 |

*表继续在下一页...*

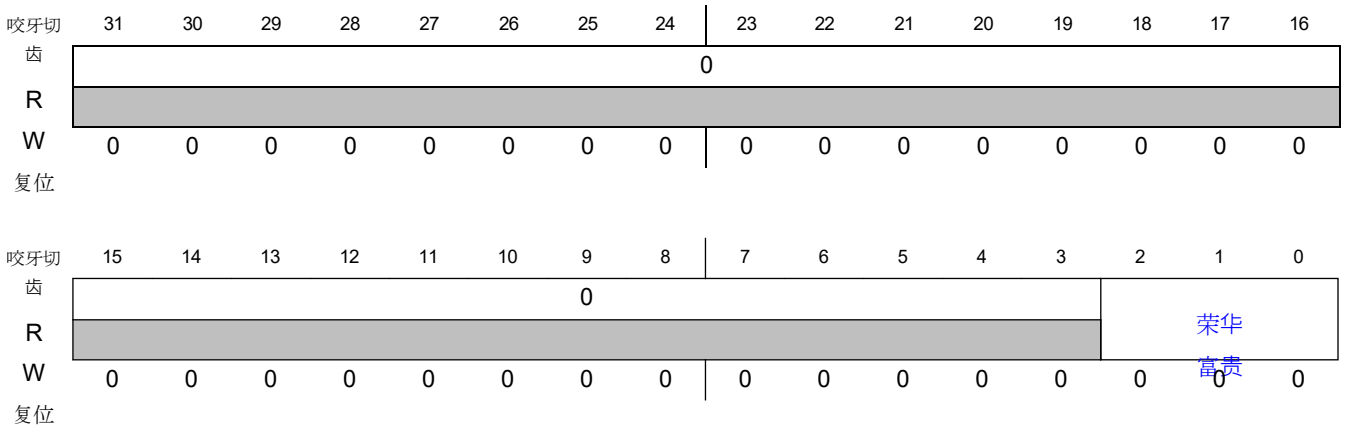
| 场地 | 职能 |
|---|---|
| 基金会 | 启用/停用FIFO错误中断。<br>0b - 禁用该中断。1b - 启用该中断。 |
| 9<br>FWIE | FIFO警告中断启用<br>启用/停用FIFO警告中断。<br>0b - 禁用该中断。1b - 启用该中断。 |
| 8<br>基金会 | FIFO请求中断启用<br>启用/禁用FIFO请求中断。<br>0b - 禁用该中断。1b - 启用该中断。 |
| 7-5<br>- | 保留 |
| 4-2<br>- | 保留 |
| 1<br>FWDE | FIFO警告DMA启用<br>启用/停用DMA请求。<br>0b - 禁用DMA请求。1b - 启用DMA请求。 |
| 0<br>FRDE | FIFO请求DMA启用<br>启用/停用DMA请求。<br>0b - 禁用DMA请求。1b - 启用DMA请求。 |

## 51.4.1.14 SAI接收配置1寄存器（RCR1）。

### 51.4.1.14.1 偏移

| 注册 | 偏移 |
|---|---|
| RCR1 | 8Ch |

#### 51.4.1.14.2 图示




#### 51.4.1.14.3 领域

| 场地 | 职能 |
|---|---|
| 31-3<br>- | 保留 |
| 2-0<br>荣华富贵 | 接收FIFO水印<br>配置所有启用的接收通道的水印等级。 |

### 51.4.1.15 SAI接收配置2寄存器（RCR2）。

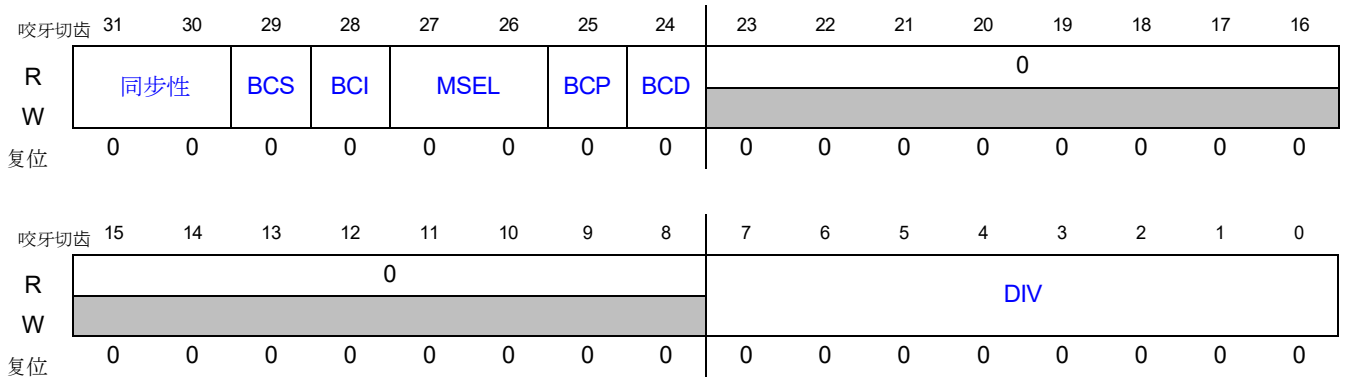
#### 51.4.1.15.1 偏移

| 注册 | 偏移 |
|---|---|
| RCR2 | 90h |

#### 51.4.1.15.2 职能

当 RCSR[RE]被设置时，这个寄存器不能被改变。

### 51.4.1.15.3 图示




### 51.4.1.15.4 领域

| 场地 | 职能 |
|---|---|
| 31-30<br>同步性 | 同步模式<br>在异步和同步操作模式之间进行配置。当配置为同步操作模式时，发射器必须配置为异步操作。<br>00b - 异步模式。<br>01b - 与发射器同步。10b - 保留。<br>11b - 保留。 |
| 29<br>BCS | 位时钟互换<br>这个字段交换了接收器使用的位时钟。当接收机配置为异步模式且该位被设置时，接收机由发射机位时钟（SAI_TX_BCLK）提供时钟。这允许发射器和接收器共享相同的位时钟，但接收器继续使用接收器的帧同步（SAI_RX_SYNC）。<br>当接收机被配置为同步模式时，发射机BCS字段和接收机BCS字段必须被设置为相同的值。当两者都被设置时，发射器和接收器都由接收器位时钟（SAI_RX_BCLK）提供时钟，但使用发射器帧同步（SAI_TX_SYNC）。<br>0b - 使用正常的位时钟源。1b - 调换<br>位时钟源。 |
| 28<br>BCI | 位时钟输入<br>当这个字段被设置并在同步或异步模式下使用内部产生的位时钟时，接收器实际使用的位时钟被垫子输出延迟（接收器被垫子输入时钟，就像时钟是外部产生的一样）。这样做的效果是减少了数据输入的设置时间，但增加了数据输出的有效时间。<br>当该位被设置时，接收器应使用数据手册中的从属模式计时。在同步模式下，该位允许接收器使用数据表中的从属模式计时，而发射器使用主模式计时。当配置为外部产生的位时钟时，该字段没有影响。<br>0b - 没有影响。<br>1b - 内部逻辑的时钟就像外部产生的位时钟一样。 |

| | |
|---|---|
| 27-26<br>MSEL | 选择MCLK<br><br>选择用于生成内部产生的比特时钟的音频主时钟选项。当配置为外部产生的位时钟时，这个字段没有影响。 |

*表继续在下一页...*

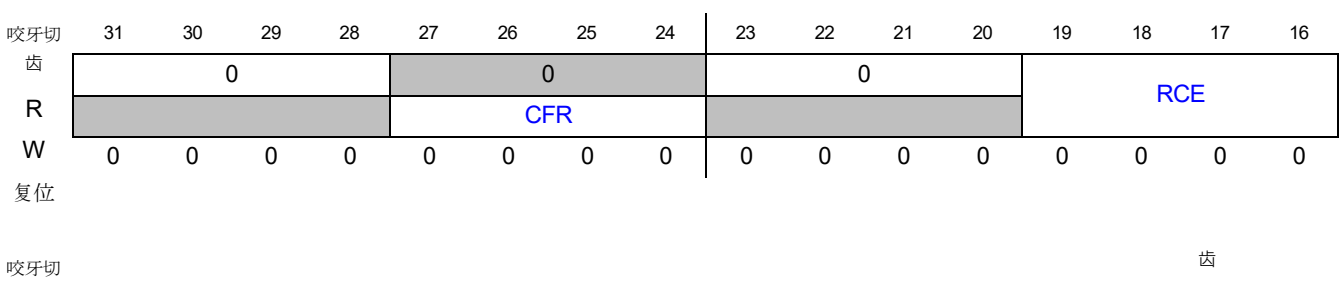
| 场地 | 职能 |
| --- | --- |
| | **注意：**根据设备的不同，一些主时钟选项可能不可用。关于每个选项的可用性和芯片的具体含义，请参见芯片的具体信息。<br>00b - 选择总线时钟。<br>01b - 选择了主时钟(MCLK)1选项。10b - 选择了主时钟(MCLK)2选项。11b - 已选择主时钟(MCLK)3选项。 |
| 25<br>BCP | 位时钟极性<br>配置位时钟的极性。<br>0b - 位时钟为高电平，驱动输出在上升沿，采样输入在下降沿。1b - 位时钟为低电平，驱动输出在下降沿，采样输入在上升沿。 |
| 24<br>BCD | 位时钟方向<br>配置位时钟的方向。<br>0b - 位时钟在从属模式下由外部产生。1b - 位时钟在主模式下由内部产生。 |
| 23-8<br>- | 保留 |
| 7-0<br>DIV | 位时钟分频<br>当配置为内部位时钟时，将音频主时钟向下划分以产生位时钟。除法值为（DIV+1）*2。 |

## 51.4.1.16 SAI接收配置3寄存器（RCR3）。

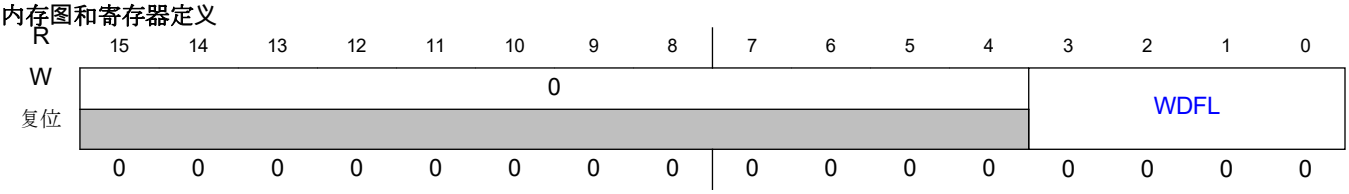
### 51.4.1.16.1 偏移

| 注册 | 偏移 |
| --- | --- |
| RCR3 | 94h |

### 51.4.1.16.2 图示



内存图和寄存器定义



### 51.4.1.16.3 领域

| 场地 | 职能 |
|---|---|
| 31-28<br>- | 保留 |
| 27-24<br>CFR | 通道FIFO复位<br>重置一个特定通道的FIFO指针。读取这个字段将总是返回0。只有当一个通道被禁用或FIFO错误标志被设置时，FIFO指针才应被重置。<br>CFR字段的宽度=接收通道的数量（称之为N）。例如，如果CFR是2位宽，那么第24位是指接收通道1的FIFO指针，第25位是指接收通道2的FIFO指针。设置位24重置接收通道1的FIFO指针，设置位25启用接收通道2的FIFO指针。设置位 N 将重置接收通道 N FIFO 指针。<br>0b - 没有影响。<br>1b - 接收数据通道N FIFO被重置。 |
| 23-20<br>- | 保留 |
| 19-16<br>RCE | 接收通道启用<br>启用相应的数据通道进行接收操作。改变这个字段将立即生效，以产生FIFO请求和警告标志，但在接收操作的每一帧结束时生效。<br>RCE字段的宽度=接收通道的数量（称之为N）。例如，如果RCE字段是2位宽，那么第16位是指接收通道1，第17位是指接收通道2。设置位16启用接收通道1，设置位17启用接收通道2。设置位N将启用接收通道N。<br>0b - 接收数据通道N被禁用。1b - 接收数据通道N被启用。 |
| 15-4<br>- | 保留 |
| 3-0<br>WDFL | 字旗配置<br>配置字开始标志被设置的字。写入的值应比字数少一个（例如，写0来配置帧中的第一个字）。当配置为一个大于帧大小字段的值时，那么字的开始标志就永远不会被设置。 |

## 51.4.1.17 SAI接收配置4寄存器（RCR4）。

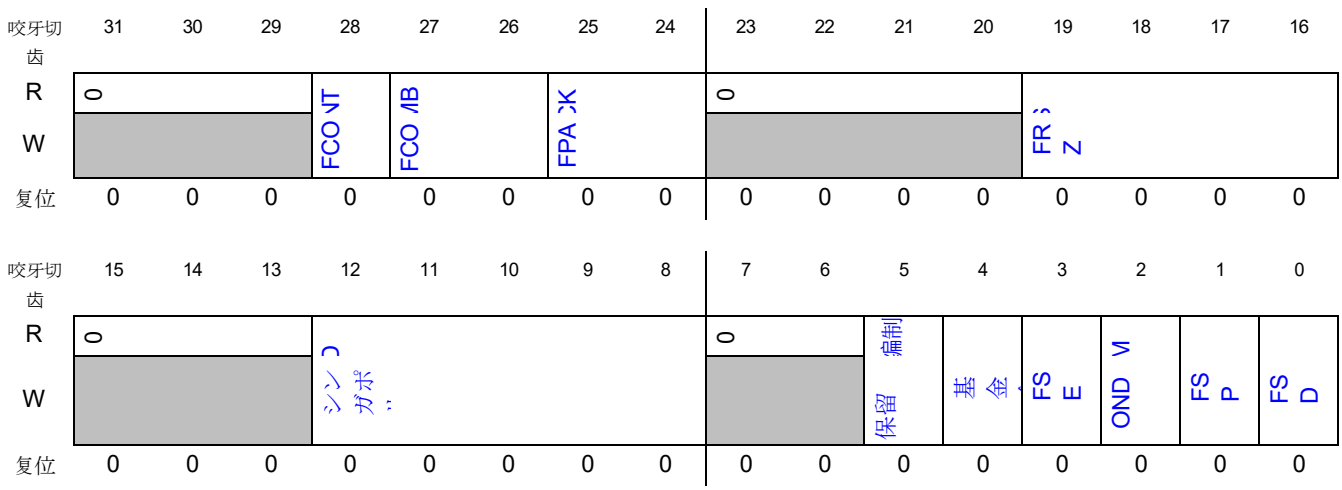
### 51.4.1.17.1 偏移

| 注册 | 偏移 |
|---|---|
| RCR4 | 98h |

### 51.4.1.17.2 职能

当 RCSR[RE]被设置时，这个寄存器不能被改变。

### 51.4.1.17.3 图示




### 51.4.1.17.4 领域

| 场地 | 职能 |
|---|---|
| 31-29<br>- | 保留 |
| 28<br>FCONT | 错误时FIFO继续<br>配置SAI在检测到FIFO错误后何时继续接收。<br>0b - 在FIFO错误时，SAI将在FIFO错误标志被清除后从下一帧开始继续。<br>1b - 在FIFO错误时，在FIFO警告标志被清除后，SAI将从导致FIFO错误的同一个字继续设置。 |

| | |
|---|---|
| 27-26<br>FCOMB | FIFO组合模式<br><br>当FIFO组合模式为FIFO读取所启用时，软件读取任何FIFO数据寄存器将在启用的数据通道FIFO中交替进行读取。例如，如果两个数据通道被启用，那么第一个读取将被执行到第一个启用的数据通道FIFO，第二个读取将被执行到第二个启用的数据通道FIFO。重置FIFO或禁用FIFO组合模式的FIFO读取将重置指针回到第一个启用的数据通道。<br><br>当接收移位寄存器的FIFO组合模式被启用时，第一个启用的数据通道输入将在启用的数据通道FIFO之间交替进行。例如，如果两个数据通道被启用，那么第一个未屏蔽的接收字将被存储在第一个启用的数据通道FIFO中，第二个未屏蔽的接收字将被存储在第二个启用的数据通道FIFO中。由于帧的第一个字总是存储在第一个启用的数据通道FIFO中，因此建议每帧的未屏蔽字的数量与启用的数据通道的数量成整数关系。<br><br>00b - 禁用FIFO组合模式。<br>01b - 在FIFO写入时启用FIFO组合模式（来自接收移位寄存器）。10b - 在FIFO读取时启用FIFO组合模式（通过软件）。<br>11b - 在FIFO写入（来自接收移位寄存器）和读取（通过软件）时启用FIFO组合模式。 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| 25-24<br>ÃÃÃ | FIFO打包模式<br>启用将8位数据或16位数据打包到每个32位FIFO字中。如果字的大小大于8-位或16-位，那么只有第一个8-位或16-位被存储到FIFO中。每一帧中的第一个字总是以一个新的32位FIFO字开始，第一个移位必须配置在第一个打包字内。当FIFO打包被启用时，FIFO读指针只有在软件读取完整的32位FIFO字时才会递增。<br>00b - 禁用FIFO包装 01b - 保留<br>。<br>10b - 启用8位FIFO包装 11b - 启用16<br>位FIFO包装 |
| 23-20<br>- | 保留 |
| 19-16<br>FRSZ | 框架尺寸<br>配置每个帧中的字数。写入的值必须比帧中的字数少一个。例如，写0表示每帧有一个字。支持的最大帧大小是16个字。 |
| 15-13<br>- | 保留 |
| 12-8<br>シリーズワールド | 同步宽度<br>配置帧同步的长度，以比特时钟的数量为单位。写入的值必须比位时钟的数量少一个。例如，如果写成0，则帧同步只断言一个比特时钟。配置的同步宽度不能长于帧的第一个字。 |
| 7-6<br>- | 保留 |
| 5<br>- | 保留。软件只能对该位写0。 |
| 4<br>基金会 | MSB优先<br>配置是先接收LSB还是MSB。<br>0b - 先接收LSB。 1b - 先接<br>收MSB。 |
| 3<br>FSE | 帧同步早期<br>0b - 帧同步与帧的第一位一起断电。<br>1b - 帧同步在帧的第一位之前断言。 |
| 2<br>ONDEM | 按需模式<br>当设置时，并且帧同步是在内部产生的，只有当FIFO警告标志清除时才会产生帧同步。<br>0b - 连续产生内部帧同步。<br>1b - 当FIFO警告标志清除时产生内部帧同步。 |
| 1<br>FSP | 帧同步极性<br>配置帧同步的极性。<br>0b - 帧同步为高电平有效。 1b -<br>帧同步为低电平有效。 |

| | |
|---|---|
| 0<br>FSD | 帧同步方向<br>配置帧同步的方向。<br>0b - 帧同步在从属模式下由外部产生。1b - 帧同步在主模式下由内部产生。 |

## 51.4.1.18 SAI接收配置5寄存器（RCR5）。

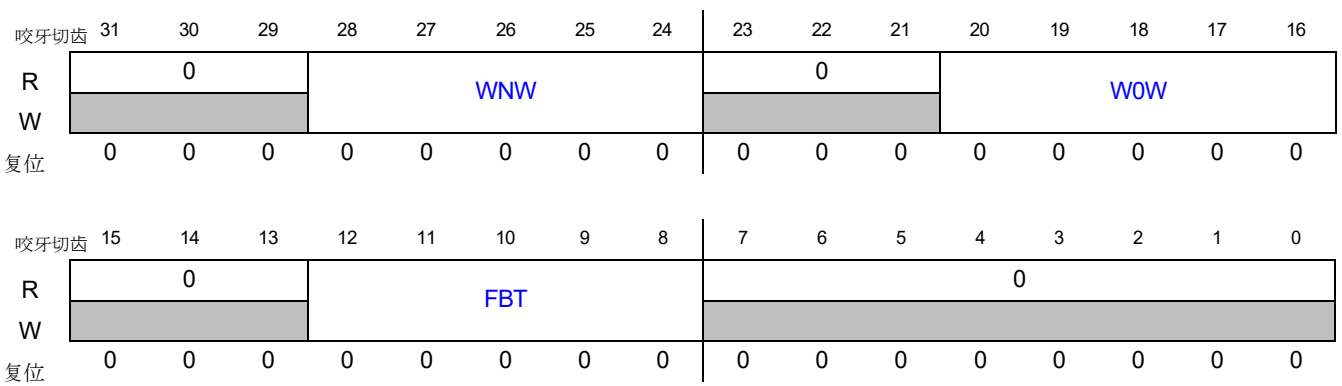
### 51.4.1.18.1 偏移

| 注册 | 偏移 |
|---|---|
| RCR5 | 9Ch |

### 51.4.1.18.2 职能

当 RCSR[RE]被设置时，这个寄存器不能被改变。

### 51.4.1.18.3 图示




### 51.4.1.18.4 领域

| 场地 | 职能 |
|---|---|
| 31-29<br>- | 保留 |
| 28-24<br>WNW | 字N宽度<br>配置每个字的位数，除了帧中的第一个字外，每个字的位数。写入的值必须比每个字的比特数少一个。不支持字宽小于8位的情况。 |
| 23-21<br>- | 保留 |
| 20-16<br>W0W | 字0宽度<br>配置每个帧中第一个字的位数。写入的值必须比第一个字的位数少一个。如果每帧只有一个字，则不支持小于8位的字宽。 |

| | |
|---|---|
| 15-13 | 保留 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| - | |
| 12-8<br>基金会 | 第一比特移位<br>配置帧中每个字的第一个接收位的索引。如果配置为MSB优先，收到的下一个比特的索引比收到的当前比特少一个。如果配置为LSB优先，则收到的下一个位的索引比当前收到的位多一个。当配置为MSB优先时，写入的值必须大于或等于字宽。当配置为LSB优先时，写入的值必须小于或等于31字的宽度。 |
| 7-0<br>- | 保留 |

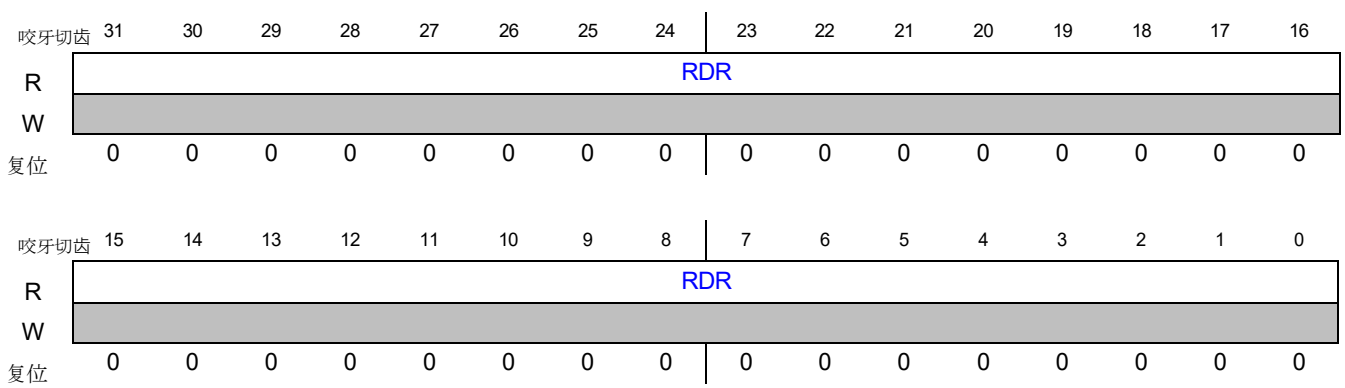
## 51.4.1.19 SAI接收数据寄存器（RDR0 - RDR3）。

### 51.4.1.19.1 偏移

对于a=0到3。

| 注册 | 偏移 |
|---|---|
| RDRa | A0h + (a × 4h) |

### 51.4.1.19.2 图示




### 51.4.1.19.3 领域

| 场地 | 职能 |
|---|---|

| | |
|---|---|
| 31-0<br>RDR | 接收数据寄存器<br>当接收FIFO不为空时，从该寄存器的读数将返回接收FIFO顶部的数据。当接收FIFO为空时，从该寄存器的读数将被忽略。 |

## 51.4.1.20 SAI接收FIFO寄存器（RFR0 - RFR3）。

### 51.4.1.20.1 偏移

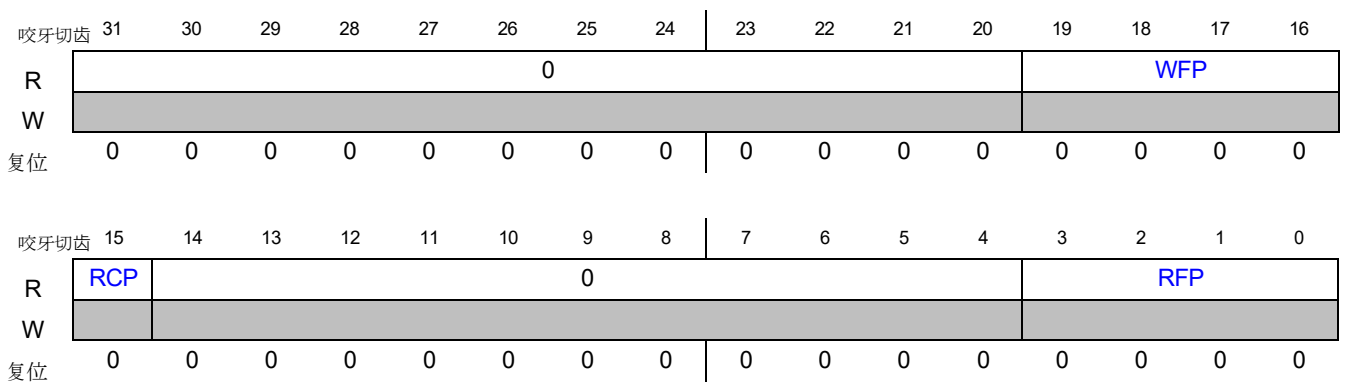
对于a=0到3。

| 注册 | 偏移 |
|---|---|
| 融资租赁a | C0h + (a × 4h) |

### 51.4.1.20.2 职能

读和写指针的MSB被用来区分FIFO满和空的情况。如果读和写指针相同，那么FIFO是空的。如果读和写指针除了MSB以外都是相同的，那么FIFO是满的。

### 51.4.1.20.3 图示




### 51.4.1.20.4 领域

| 场地 | 职能 |
|---|---|
| 31-20<br>- | 保留 |
| 19-16<br>WFP | 写FIFO指针<br>接收数据通道的FIFO写指针。 |

| | |
|---|---|
| 15<br>RCP | 接收通道指针<br><br>当启用FIFO组合模式进行读取时，表示该数据通道是下一个要读取的FIFO。<br>0b - 没有影响。<br>1b - FIFO组合被启用，用于FIFO的读取，这个FIFO将在下一次FIFO读取时被读取。 |

*表继续在下一页...*

| 场地 | 职能 |
|---|---|
| 14-4<br>- | 保留 |
| 3-0<br>RFP | 读取FIFO指针<br>接收数据通道的FIFO读取指针。 |

## 51.4.1.21 SAI接收屏蔽寄存器（RMR）。

### 51.4.1.21.1 偏移

| 注册 | 偏移 |
|---|---|
| 风险管理 | E0h |

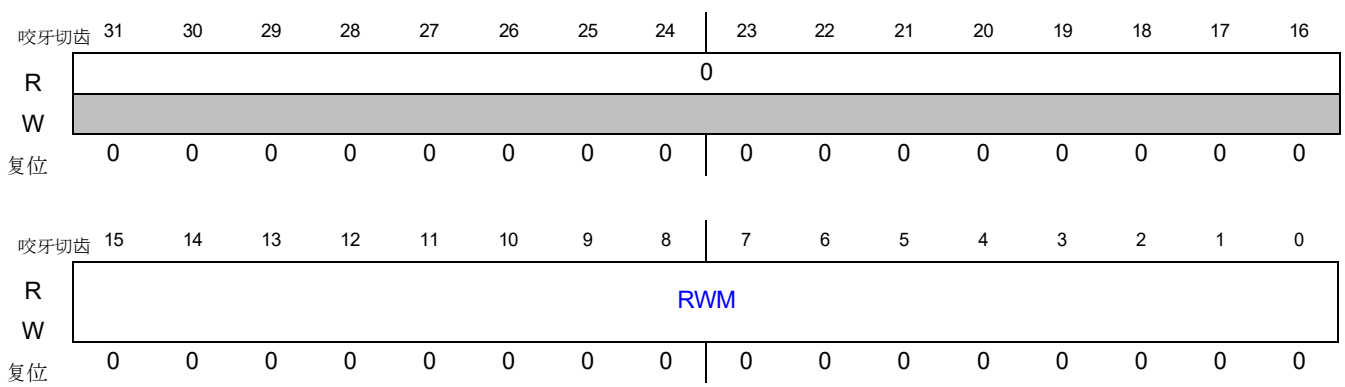
### 51.4.1.21.2 职能

这个寄存器是双缓冲的，并进行更新。

1. 当RCSR[RE]被首次设置时
2. 在每一帧结束时

这允许每一帧中的遮蔽词在每一帧中发生变化。

### 51.4.1.21.3 图示



#### 51.4.1.21.4 领域

| 场地 | 职能 |
|---|---|
| 31-16<br>- | 保留 |
| 15-0<br>RWM | 接收字屏蔽<br>配置帧中相应字的接收字是否被屏蔽（接收数据被忽略，不写入接收FIFO）。<br>0000000000000000b - 字N被启用。<br>0000000000000001b - 字N被屏蔽。 |

## 51.5 职能描述

本节提供了该块的完整功能描述。

### 51.5.1 SAI时钟

SAI的时钟包括。

- 音频主时钟
- 位时钟
- 总线时钟

#### 51.5.1.1 音频主时钟

当接收器或发射器被配置为内部产生的位时钟时，音频主时钟被用来产生位时钟。发射器和接收器可以独立选择总线时钟和最多三个音频主时钟来产生比特时钟。

音频主时钟的生成和选择是特定于芯片的。请参考特定芯片的时钟信息，了解音频主时钟是如何产生的。

#### 51.5.1.2 位时钟

SAI发射器和接收器支持异步自由运行的比特时钟，这些比特时钟可以从音频主时钟内部产生或从外部提供。也可以选择在接收器和发射器之间或在多个SAI外围设备之间进行同步位时钟和帧同步操作。

- 如果发射器和接收器都被配置为异步操作，那么发射器和接收器将各自使用*自己的*位时钟和帧同步。
- 如果*发射器*被配置为异步模式，而接收器被配置为同步模式，那么发射器和接收器都将使用*发射器*位时钟和帧同步。
- 如果*接收器*被配置为异步模式，而发射器被配置为同步模式，那么发射器和接收器都将使用*接收器的*位时钟和帧同步。

请注意，软件配置的是同步或异步模式，选择的是所使用的位时钟/帧同步。

外部产生的位时钟必须是。

- 在启用SAI发射器或接收器之前就已启用
- 在SAI发射器或接收器被禁用并完成其当前帧后禁用

如果SAI发射器或接收器在异步模式下使用外部产生的位时钟，而该位时钟是由在停止模式下被禁用的SAI产生的，那么在进入停止模式前应通过软件禁用发射器或接收器。当发射器或接收器处于同步模式时，这个问题并不适用，因为所有的同步SAI是同时启用和禁用的。

#### 51.5.1.3 总线时钟

总线时钟被控制和配置寄存器使用，并用于产生同步中断和DMA请求。

**注意事项**

虽然没有规定具体的最低总线时钟频率，但总线时钟频率必须足够快（相对于位时钟频率），以确保FIFO能够得到服务，而不产生发送器FIFO欠载或接收器FIFO溢出的情况。

### 51.5.2 SAI复位

SAI在系统复位时被异步复位。SAI有一个软件复位和一个FIFO复位。

#### 51.5.2.1 软件复位

SAI发射机包括一个软件复位，复位所有发射机的内部逻辑，包括位时钟生成，状态标志和FIFO指针。它不重置配置寄存器。软件复位在被软件清除之前一直保持正向状态。

SAI接收机包括一个软件复位功能，可以重置所有接收机的内部逻辑，包括位时钟生成、状态标志和FIFO指针。它并不重置配置寄存器。软件复位在被软件清除之前一直保持正向状态。

#### 51.5.2.2 FIFO复位

SAI发射器包括一个FIFO复位，使FIFO写指针同步到与FIFO读指针相同的值。这将清空FIFO的内容，在TCSR[FEF]被设置后，在FIFO被重新初始化和TCSR[FEF]被清除前使用。FIFO复位仅在一个周期内被确认。

SAI发射器也可以通过设置相应的TCR3[CFR]位来重置各个数据通道的FIFO。这只能在相应的TCR3[TCE]位清零时进行。

SAI 接收器包括一个 FIFO 复位，将 FIFO 读取指针同步到与 FIFO 写入指针相同的值。这将清空FIFO的内容，并在RCSR[FEF]被设置和任何剩余数据从FIFO读出后，在RCSR[FEF]被清除前使用。FIFO复位仅在一个周期内被确认。

SAI接收器也可以通过设置相应的RCR3[CFR]位来重置单个数据通道的FIFO。这只能在相应的RCR3[RCE]位清零时进行。

### 51.5.3 同步模式

SAI的发射器和接收器可以相互同步运行。

#### 51.5.3.1 同步模式

SAI发射器和接收器可以被配置为与同步位时钟和帧同步操作。

如果发射器位时钟和帧同步要被发射器和接收器同时使用。

- 发射器必须配置为异步操作，接收器配置为同步操作。
- 在同步模式下，只有当发射器和接收器都被启用时，接收器才被启用。
- 建议发射器是最后一个启用，第一个禁用。

如果接收器的位时钟和帧同步要同时被发射器和接收器使用。

- 接收器必须配置为异步操作，发射器配置为同步操作。
- 在同步模式下，只有当接收器和发射器都被启用时，发射器才会被启用。
- 建议接收器是最后一个启用，第一个禁用。

当在同步模式下工作时，只有位时钟、帧同步和发送器/接收器的启用是共享的。发送器和接收器独立工作，尽管配置寄存器必须在发送器和接收器上一致配置。

### 51.5.4 帧同步配置

当启用时，SAI连续发送和/或接收数据帧。每个帧由固定数量的字组成，每个字由固定数量的比特组成。在每一帧中，任何给定的字都可以被屏蔽，导致接收器忽略该字，发射器在该字的持续时间内处于三态。

帧同步信号用于指示每个帧的开始。一个有效的帧同步需要检测到一个上升沿（如果是高电平有效）或下降沿（如果是低电平有效），并且发射器或接收器不能忙于前一个帧。在启用发射器或接收器后的前四个比特时钟周期内，有效的帧同步也会被忽略（从属模式）或不产生（主模式）。

发送器和接收器的帧同步可以用以下任何一个选项独立配置。

- 外部产生或内部产生
- 高电平有效或低电平有效
- 以帧中的第一个位为断言，或提前一个位断言
- 在1个比特时钟和第一个字长之间的持续时间内发出断言
- 每帧长度从1到16个字
- 字长支持每字8至32位
  - 第一个字的长度和其余字的长度可以单独配置
- 字可以被配置为先发送/接收MSB或先接收LSB

在启用SAI发射器或接收器后，这些配置选项不能被改变。

## 51.5.5 数据FIFO

每个发送和接收通道包括一个大小为8×32位的FIFO。FIFO的数据是通过SAI发送/接收数据寄存器访问的。

### 51.5.5.1 数据调整

通过使用第一位移位配置字段，FIFO中的数据可以在32位宽的寄存器中的任何位置对齐，该字段选择了第一位移位的比特索引（31和0之间）。

图51-2为LSB优先配置，图51-3为MSB优先配置，说明了支持的数据排列和所需的第一比特移位配置的例子。

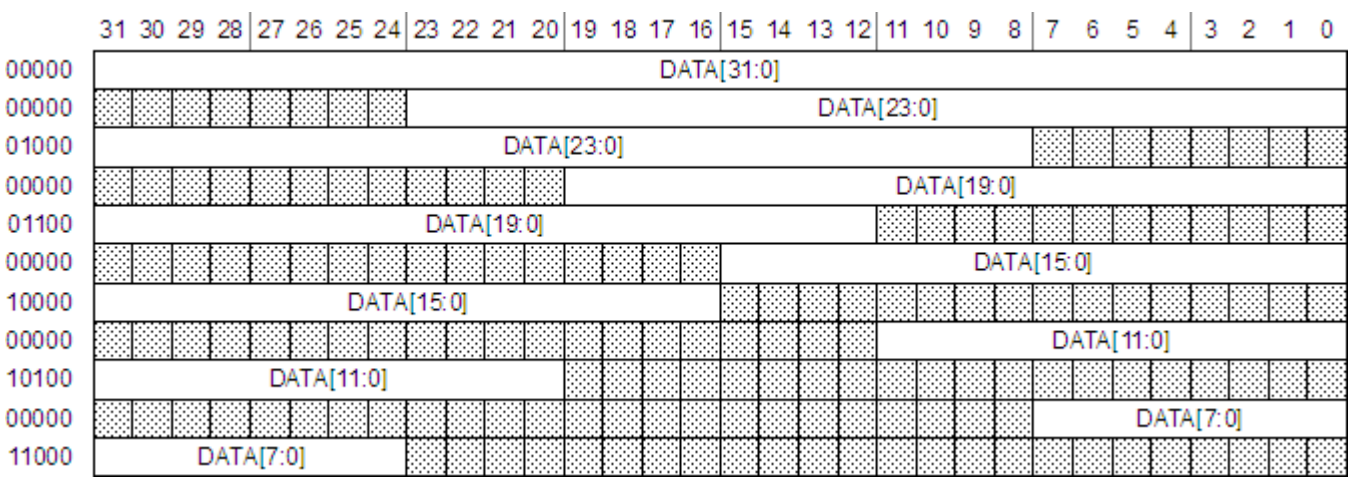



**图51-2.SAI第一比特移位，LSB在前**

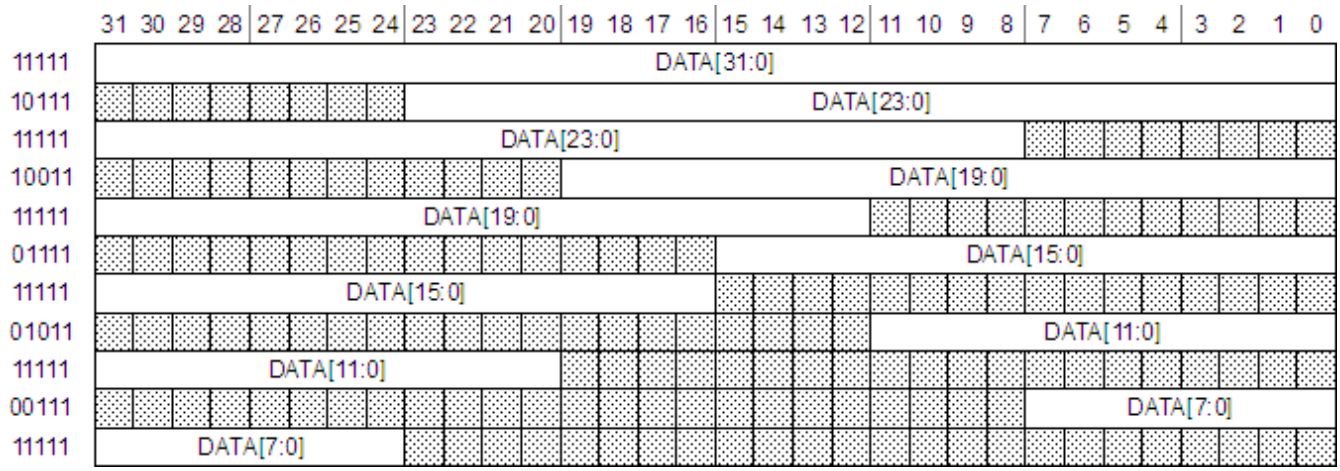



**图51-3.SAI第一比特移位，MSB在前**

### 51.5.5.2 FIFO指针

当向TDR写入时，相应TFR的WFP在每次有效写入后都会递增。SAI支持对TDR进行8位、16位和32位的写入，每次单独写入后FIFO指针都会递增。请注意，8位写入只能在传输8位数据时使用，16位写入只能在传输16位数据时使用。

如果FIFO已满，对TDR的写入将被忽略。如果传输FIFO是空的，则必须在下一个未屏蔽字开始前至少三个位时钟写入TDR，以避免FIFO欠载。

读取RDR时，每次有效读取后，相应RFR的RFP会递增。SAI支持从RDR读取8位、16位和32位的数据，FIFO指针在每次单独读取后都会递增。注意，8位读数只能在接收8位数据时使用，16位读数只能在接收16位数据时使用。

如果FIFO是空的，从RDR中的读取将被忽略。如果接收FIFO是满的，RDR必须在一个未屏蔽的字结束前至少读三个位时钟，以避免FIFO超限。

### 51.5.5.3 FIFO包装

FIFO包装支持在一个32位FIFO字中存储多个8位或16位数据字，用于发射器和/或接收器。虽然这可以通过调整每个字的位数和每帧的字数来模拟（例如，一个32位的字

与每帧两个16位字相比，FIFO打包不需要每帧字的倍数，并完全支持字屏蔽。当启用FIFO包装时，FIFO指针只在软件写入（传输）或读取（接收）完整的32位FIFO字时才会递增，支持每帧内不同字被加载/存储在内存的不同区域的情况。

当16位FIFO包装被启用用于传输时，传输移位寄存器在每一帧的开始和每第二个未屏蔽的传输字之后被加载。发送的第一个字取自字节偏移量为0的16位字（第一位由TCFG5[FBT]选择必须配置在这个16位字内），发送的第二个字取自字节偏移量为2的16位字（第一位由TCSR5[FBT][3：0]选择）。一旦16位字传输完毕，发射器将传输逻辑零，直到下一个字的开始。

当16位FIFO包装被启用用于接收时，接收移位寄存器在每第二个未屏蔽的接收字之后被存储，如果每帧中有奇数的未屏蔽的接收字，则在每帧结束时被存储。收到的第一个字存储在字节偏移量为0的16位字中（第一位由RCFG5[FBT]选择，必须配置在这个16位字中），收到的第二个字存储在字节偏移量为2的16位字中（第一位由RCSR5[FBT][3:0]选择）。一旦收到16位字，接收器将忽略收到的数据，直到下一个字的开始。

8位FIFO包装与16位包装类似，只是在每个32位FIFO字中加载或存储四个字。第一个字被加载/存储在字节偏移量$0，第二个字在字节偏移量$1，第三个字在字节偏移量$2，第四个字在字节偏移量
3.TCFG5[FBT]和/或RCFG5[FBT]必须配置在字节偏移量$0以内。

### 51.5.5.4 FIFO组合

FIFO组合模式允许将多个数据通道的独立FIFO作为单个FIFO使用，用于软件访问或单个数据通道或两者。注意，启用FIFO组合模式时，启用的数据通道必须是连续的，而且数据通道0必须被启用。

结合FIFO进行软件访问（写入发送FIFO寄存器，读取接收FIFO寄存器），允许DMA控制器或软件读取或写入多个FIFO，而不递增被访问的地址。一旦启用，对FIFO寄存器的第一次软件访问将访问第一个启用的通道FIFO，而对FIFO寄存器的第二次访问将访问第二个启用的通道FIFO。这种情况一直持续到软件

访问最后一个启用的通道FIFO，指针重新设置到第一个启用的通道FIFO。为了手动重置指针，软件可以重置FIFO，或者在软件访问时禁用FIFO组合。

结合传输数据通道的FIFO，允许一个数据通道使用所有启用的通道FIFO的FIFO，每个启用的数据通道上有相同的数据输出。所有启用的数据通道的传输移位寄存器在每一帧的开始和每N个未屏蔽的字（其中N是启用的数据通道的数量）被加载。传输的第一个字从第一个启用的通道FIFO加载，而传输的第二个字从第二个启用的通道FIFO加载，以此类推，直到帧的结束。由于每一帧中的第一个字总是从第一个启用的数据通道加载，建议每一帧中未被屏蔽的字的数量均匀地除以启用的数据通道数量。

组合接收数据通道的FIFO允许一个数据通道使用所有启用的通道FIFO的FIFO，从通道0接收的数据存储到每个启用的数据通道。所有启用的数据通道的接收移位寄存器在每N个未屏蔽的字（其中N是启用的数据通道的数量）后存储。收到的第一个字被存储到第一个启用的通道FIFO，而收到的第二个字被存储到第二个启用的通道FIFO，以此类推直到帧的结束。由于每一帧中的第一个字总是存储在第一个启用的数据通道中，因此建议每一帧中未屏蔽的字的数量要均匀地除以启用的数据通道的数量。

注意，合并数据通道的FIFO将在同一时间加载或存储每个通道的FIFO。这意味着每隔N个字（N为启用的数据通道数）才检查一次FIFO错误条件，如果任何一个启用的数据通道满足警告标志或请求标志条件，FIFO警告和请求标志将被断言。

### 51.5.6 字掩码寄存器

SAI发射器和接收器各自包含一个字屏蔽寄存器，即TMR和RMR，可用于屏蔽帧中的任何字。由于字屏蔽寄存器是双缓冲的，软件可以在每一帧结束前更新它，以屏蔽下一帧中的特定字。

TMR使传输数据引脚在每个选定的字的长度上被三振，传输FIFO不被读取屏蔽的字。

RMR使每个选定字的接收数据被丢弃，不写入接收FIFO。

## 51.5.7 中断和DMA请求

SAI发射器和接收器产生单独的中断和单独的DMA请求，但支持相同的状态标志。

### 51.5.7.1 FIFO请求标志

FIFO请求标志是根据FIFO中的条目数和FIFO水印配置来设置的。

当任何一个启用的发送FIFO中的条目数小于或等于发送FIFO水印配置时，发送FIFO请求标志被设置，当每个启用的发送FIFO中的条目数大于发送FIFO水印配置时被清除。

当任何一个启用的接收FIFO中的条目数大于接收FIFO水印配置时，接收FIFO请求标志被设置，当每个启用的接收FIFO中的条目数小于或等于接收FIFO水印配置时被清除。

FIFO请求标志可以产生一个中断或一个DMA请求。

### 51.5.7.2 FIFO警告标志

FIFO警告标志是根据FIFO中的条目数来设置的。

当任何一个启用的发送FIFO中的条目数为空时，发送警告标志被设置，当每个启用的发送FIFO中的条目数不为空时，发送警告标志被清零。

当任何一个启用的接收FIFO中的条目数满时，接收警告标志被设置，当每个启用的接收FIFO中的条目数未满时，接收警告标志被清除。

FIFO警告标志可以产生一个中断或一个DMA请求。

### 51.5.7.3 FIFO错误标志

当任何一个被启用的发送FIFO出现下溢时，发送FIFO错误标志被设置。在它被设置后，在TCSR[FEF]被清除前，所有被启用的发送通道将发送零数据。

当TCR4[FCONT]被设置时，FIFO将在下溢后继续传输数据，无需软件干预。为了确保数据以正确的顺序传输，发射器将从导致FIFO溢出的帧中的相同字号开始继续传输，但只有在新的数据被写入发射FIFO之后。软件仍应清除TCSR[FEF]标志，但不必重新初始化发送FIFO。

当任何一个启用的接收FIFO溢出时，RCSR[FEF]被设置。在它被设置后，所有启用的接收通道都会丢弃收到的数据，直到
RCSR[FEF]被清除，下一个下一个接收帧开始。在RCSR[FEF]被清除之前，所有启用的接收FIFO都应该被清空。

当RCR4[FCONT]被设置时，FIFO将在溢出后继续接收数据，无需软件干预。为了确保数据以正确的顺序被接收，接收器将从导致FIFO溢出的帧中的相同字号继续接收，但只有在数据从接收FIFO中被读出之后。软件仍应清除RCSR[FEF]标志，但不清空接收FIFO。

FIFO错误标志只能产生一个中断。

### 51.5.7.4 同步错误标志

同步错误标志，TCSR[SEF]或RCSR[SEF]，在配置为外部产生的帧同步时被设置，当发射器或接收器忙于前一帧时，外部帧同步断言。外部帧同步的断言被忽略，同步错误标志被设置。当同步错误标志被设置时，发射器或接收器在空闲时或在每一帧结束时继续检查帧同步的断言。

同步错误标志只能产生一个中断。

### 51.5.7.5 字开始标志

字开始标志在所选字的第二个位时钟开始时被设置，这是由字标志寄存器字段配置的。

字开始标志只能产生一个中断。

# 第五十二章
# 以太网MAC (ENET)

## 52.1 特定芯片的ENET信息

S32K148有一个ENET模块的实例。S32K1xx系列的其他产品没有ENET模块。

**注意事项**

- 在使用ENET_MDIO或ENET_TMRx之前，需要为相应的引脚配置GPIO_PDDR。
- 通过设置PORT_PCRn寄存器的DSE字段，为ENET的发送焊盘启用驱动强度。
- 只支持从STOPx模式下通过魔术包唤醒，而不支持从VLPS模式下唤醒。
- 对于RMII模式，IO信号描述输入复用表中的MII_RMII_TX_CLK被称为RMII_REF_CLK。

### 52.1.1 ENET运行期间的软件指南

在ENET操作过程中，系统频率不能改变，因为这可能导致FIFO欠费状态。

## 52.2 简介

MAC-NET核心与10/100-Mbit/s
MAC结合，实现了第三层的网络加速功能。这些功能旨在加速处理各种常见的网络协议，如IP、TCP、UDP和ICMP，向客户应用提供线速服务。

## 52.3 概述

该核心实现了一个符合IEEE802.3-2002标准的双速10/100-Mbit/s以太网MAC。MAC层提供与半双工或全双工10/100-Mbit/s以太网LAN的兼容性。

MAC操作是完全可编程的，可用于网络接口卡（NIC）、桥接或交换应用。该核心根据IETF RFC 2819实现了远程网络监控（RMON）计数器。

该核心还实现了一个硬件加速块，以优化提供TCP/IP、UDP和ICMP协议服务的网络控制器的性能。加速块在硬件中执行关键功能，这些功能的实现通常需要大量的软件开销。

该核心实现了可编程的嵌入式FIFO，可以在接收路径上提供缓冲，实现无损流量控制。

先进的电源管理功能有神奇的数据包检测和可编程的断电模式。

ENET模块内部的统一DMA（uDMA）优化了ENET内核和SoC之间的数据传输，并支持增强的缓冲区描述符编程模型以支持IEEE 1588功能。

带有IEEE 1588的可编程以太网MAC将标准的IEEE 802.3以太网MAC与一个时间戳模块整合在一起。IEEE 1588标准为工业自动化应用的分布式控制节点提供精确的时钟同步。

### 52.3.1 特点

#### 52.3.1.1 以太网MAC功能

- 执行完整的802.3规范，包括序言/SFD生成、帧填充生成、CRC生成和检查。
- 支持零长度序言
- 可动态配置以支持10/100-Mbit/s操作
- 支持10/100 Mbit/s全双工和可配置的半双工操作
- 符合AMD神奇的数据包检测与中断，用于节点远程电源管理
- 通过以下之一与商业以太网PHY设备的无缝接口。

  - 一个4位媒体独立接口（MII），工作频率为2.5/25MHz。
  - 4位非标准的MII-Lite（没有CRS和COL信号的MII），工作频率为2.5/25MHz。
  - 一个工作频率为50MHz的2位精简MII（RMII）。
- 简单的64位FIFO用户应用界面
- CRC-32全速检查，可选择将帧校验序列（FCS）字段转发给客户端
- 在传输或转发用户应用提供的FCS时生成和附加CRC-32，可在每个帧基础上选择
- 在全双工模式下。
  - 实施自动暂停帧（802.3 x31A）的生成和终止，提供流量控制，无需用户应用干预
  - 用于形成暂停帧的暂停量子 - 可动态编程
  - 暂停帧的生成还可由用户应用控制，提供灵活的流量控制
  - 可选择将收到的暂停帧转发给用户应用程序
  - 实施标准的流量控制机制
- 在半双工模式下：提供全面的碰撞支持，包括干扰、回退和自动重传。
- 支持符合IEEE 802.1Q的VLAN标记的帧
- 可编程的MAC地址。发送时插入；接收时丢弃目标地址不匹配的帧（广播和暂停帧除外）
- 支持可编程的混杂模式，以省略接收时的MAC目标地址检查
- 基于64个条目的哈希表在接收时进行多播和单播地址过滤，减少上层的处理负荷
- 可编程的框架最大长度，提供对任何标准或专有框架长度的支持
- 帧流量和错误（对齐、CRC、长度）以及暂停帧的统计指标，为IEEE 802.3基本和强制性管理信息数据库（MIB）包和远程网络监控提供统计指标（RFC 2819）。
- 简单的握手用户应用FIFO接口，具有完全可编程的深度和阈值水平
- 在用户界面上为每个收到的帧提供单独的状态字，提供诸如帧长度、帧类型、VLAN标签和错误信息等信息
- 多种内部回环选项
- 用于PHY设备配置和管理的MDIO主接口支持两个可编程的MDIO基本地址，以及标准（IEEE 802.3第22条）和扩展（第45条）MDIO帧格式
- 支持传统的FEC缓冲区描述符

### 52.3.1.2 IP协议性能优化功能

- 只对TCP/IP和UDP/IP以及ICMP/IP协议数据或IP头进行操作
- 实现线速处理
- 支持IPv4和IPv6
- 透明地传递其他类型和协议的帧
- 支持符合IEEE 802.1q的VLAN标签帧，透明转发VLAN标签和控制域
- 接收时自动计算IP头和有效载荷（特定协议）的校验和，并进行验证
- 自动生成IP头和有效载荷（特定协议）校验和，并在传输时自动插入，可按帧配置
- 支持IP和TCP、UDP、ICMP数据的检查和生成和检查
- 支持IPv4和TCP协议头的全部头选项
- 为只有基本头的数据报提供IPv6支持--
  带有扩展头的数据报不经修改/不经检查就透明地通过。
- 提供收到的IP和协议错误的统计信息
- 可配置的错误帧的自动丢弃
- 可配置的主机到网络（RX）和网络到主机（TX）的字节顺序自动转换，用于帧内的IP和TCP/UDP/ICMP头。
- 可配置的接收短IP数据报的填充删除
- 可配置的以太网有效载荷对齐，允许32位字对齐的头和有效载荷处理
- 带有时钟和速率去耦FIFO的可编程存储和转发操作

### 52.3.1.3 IEEE 1588的特点

- 支持所有的IEEE 1588框架。
- 允许独立于网络速度选择参考时钟。
- 可通过软件编程对入站和出站帧进行精确的时间标记

- 用于系统校准和定时精度管理的定时器监测功能
- 通过可编程的中断生成对外部事件进行精确的时间标记
- 用于外部系统控制的可编程事件和中断生成
- 支持硬件和软件控制的定时器同步。
- 提供一个4通道的IEEE 1588定时器。每个通道都支持使用1588计数器的输入捕获和输出比较。

## 52.3.2 框图

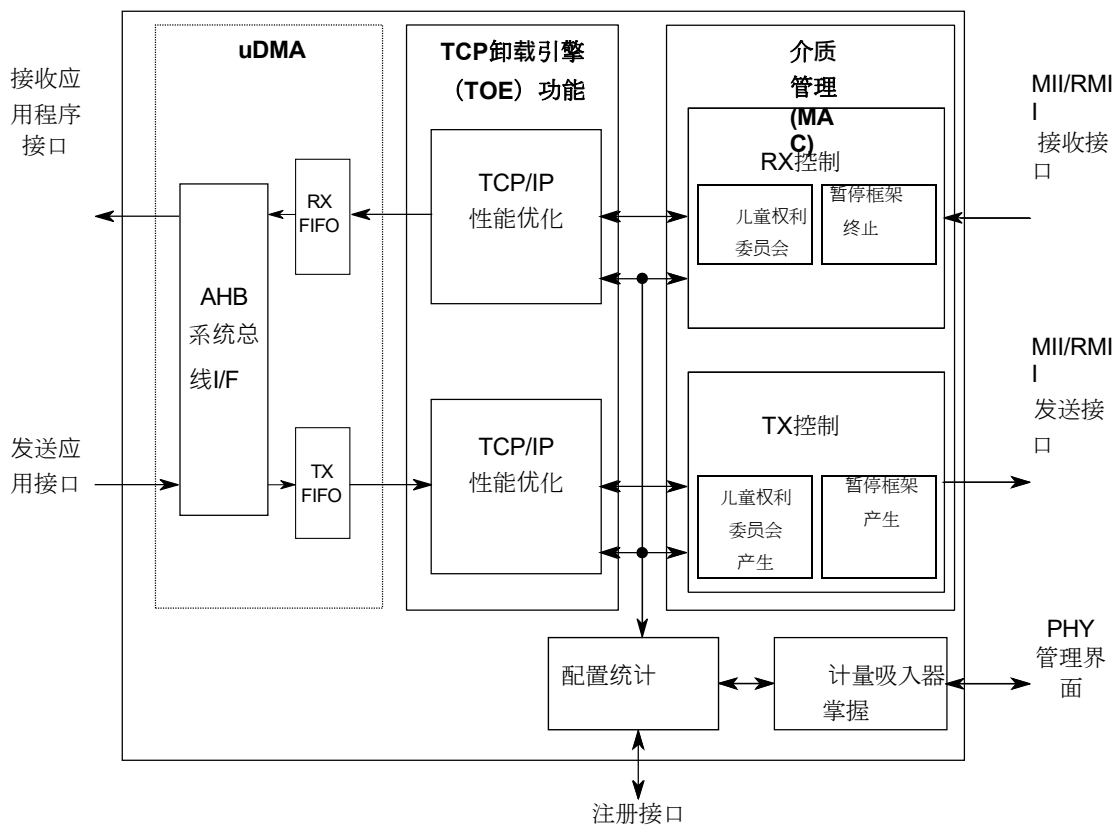



**图52-1.以太网MAC-NET核心框图**

## 52.4 外部信号描述

### 注意事项

MII列只与支持MII的设备有关。

| MII | 风险管理 | 描述 | 输入/输出 |
|---|---|---|---|
| MII_COL | - | 在检测到碰撞时被断言，并在碰撞持续存在时保持断言。这个信号在全双工模式下没有定义。 | I |
| 呼叫中心 | - | 载波感应。当断言时，表明发送或接收介质不是空闲的。<br>在RMII模式下，该信号出现在RMII_CRS_DV引脚上。 | I |
| MII_MDC | 编码：RMII_MDC | 输出时钟为PHY提供定时参考，用于MDIO信号上的数据传输。 | O |
| 呼叫中心：MII_MDIO | RMII_MDIO | 在外部PHY和媒体访问控制器之间传输控制信息。数据与MDC同步。该信号在复位后是一个输入。 | 输入/输出 |
| MII_RXCLK | - | 在MII模式下，为RXDV、RXD[3:0]和RXER提供定时参考。 | I |
| 呼叫中心：MII_RXDV | 呼叫中心： 呼叫中心： 呼叫中心： 呼叫中心： 呼叫中心： 呼叫中心： 呼叫中心： 呼叫中心： | 该输入信号表明PHY在MII上有有效的小数点存在。RXDV必须在帧的第一个恢复的小数点到最后一个小数点之间保持正向。RXDV必须在不晚于SFD的情况下开始，并且不包括任何EOF。<br>在RMII模式下，该引脚也产生CRS信号。 | I |
| MII_RXD[3:0] | RMII_RXD[1:0] | 包含当RXDV被断言时从PHY传输到媒体访问控制器的以太网输入数据。 | I |
| 呼叫中心 | 呼叫中心 | 当与RXDV一起发出时，表明PHY在当前帧中检测到一个错误。 | I |
| MII_TXCLK | - | 输入时钟，为TXEN、TXD[3:0]和TXER提供定时参考。 | I |
| MII_TXD[3:0] | RMII_TXD[1:0] | 串行输出以太网数据。仅在TXEN断言期间有效。 | O |

| | | | |
|---|---|---|---|
| 呼叫中心：MII_TXEN | RMII_TXEN | 指示MII上何时出现有效的nibbles。该信号是随着 | O |

*表继续在下一页...*

| MII | 风险管理 | 描述 | 输入/输出 |
|---|---|---|---|
| | | 帧的最后一个字节之后的第一个TXCLK之前被解除。 | |
| 呼叫中心 | - | 当TXEN也被断定为一个或多个时钟周期时，PHY将发送一个或多个非法符号。 | O |
| - | 呼叫中心：RMII_REF_CLK | 在RMII模式下，该信号是接收、发送和控制接口的参考时钟。 | I |
| 1588_TMRn | 1588_TMRn | 捕获/比较块输入/输出事件总线。<br>当配置为捕获并检测到上升沿时，当前的定时器值被锁定并传输到相应的ENET_TCCRn寄存器，供软件检查。<br>当配置为比较时，当定时器达到ENET_TCCRn中编程的比较值时，相应的信号1588_TMRn被引爆一个周期。<br>如果ENET_TCSRn[TIE]被设置，可以触发中断。<br>如果ENET_TCSRn[TDRE]被设置，可以触发DMA请求。 | 输入/输出 |
| 辽宁沈阳 | 辽宁沈阳 | 替代IEEE 1588以太网时钟输入；时钟周期应该是纳秒的整数。 | I |

## 52.5 内存图/寄存器定义

ENET寄存器必须以32位访问方式读或写。非32位的访问将以一个错误结束。保留位应写为0，读时忽略。未使用的寄存器读数为0，写入时没有任何影响。该表显示了以太网寄存器的组织。

## 表52-1.寄存器图摘要

| 偏移地址 | 科目 | 描述 |
|---|---|---|
| 0x0000 - 0x01FF | 配置 | 核心控制和状态寄存器 |
| 0x0200 - 0x03FF | 统计计数器 | MIB和远程网络监控（RFC 2819）登记册 |
| 0x0400 - 0x0430 | 1588控制 | 1588可调定时器（TSM）和1588帧控制 |
| 0x0600 - 0x07FC | 捕获/比较块 | 捕获/比较块的寄存器 |

## ENET内存图

| 地址偏移（十六进制） | 注册名称 | 宽度（单位：比特） | 访问 | 重置值 | 节/页 |
|---|---|---|---|---|---|
| 4 | 中断事件寄存器（ENET_EIR）。 | 32 | w1c | 0000_0000h | 52.5.1/1760 |
| 8 | 中断屏蔽寄存器（ENET_EIMR）。 | 32 | R/W | 0000_0000h | 52.5.2/1763 |
| 10 | 接收描述符活动寄存器（ENET_RDAR）。 | 32 | R/W | 0000_0000h | 52.5.3/1766 |
| 14 | 发送描述符激活寄存器（ENET_TDAR）。 | 32 | R/W | 0000_0000h | 52.5.4/1766 |
| 24 | 以太网控制寄存器（ENET_ECR） | 32 | R/W | F000_0000h | 52.5.5/1767 |
| 40 | MII管理帧寄存器（ENET_MMFR）。 | 32 | R/W | 0000_0000h | 52.5.6/1769 |
| 44 | MII速度控制寄存器（ENET_MSCR）。 | 32 | R/W | 0000_0000h | 52.5.7/1770 |
| 64 | MIB控制寄存器（ENET_MIBC）。 | 32 | R/W | C000_0000h | 52.5.8/1772 |
| 84 | 接收控制寄存器（ENET_RCR）。 | 32 | R/W | 05EE_0001h | 52.5.9/1773 |
| C4 | 发送控制寄存器（ENET_TCR） | 32 | R/W | 0000_0000h | 52.5.10/1776 |
| E4 | 物理地址下限寄存器（ENET_PALR）。 | 32 | R/W | 0000_0000h | 52.5.11/1778 |
| E8 | 物理地址上位寄存器（ENET_PAUR）。 | 32 | R/W | 0000_8808h | 52.5.12/1778 |
| EC | 操作码/暂停持续时间寄存器（ENET_OPD）。 | 32 | R/W | 0001_0000h | 52.5.13/1779 |
| 118 | 描述符个人上位地址寄存器（ENET_IAUR）。 | 32 | R/W | 0000_0000h | 52.5.14/1779 |
| 11C | 描述符个人下位地址寄存器（ENET_IALR）。 | 32 | R/W | 0000_0000h | 52.5.15/1780 |
| 120 | 描述符组上位地址寄存器（ENET_GAUR）。 | 32 | R/W | 0000_0000h | 52.5.16/1780 |
| 124 | 描述符组下位地址寄存器（ENET_GALR）。 | 32 | R/W | 0000_0000h | 52.5.17/1781 |
| 144 | 发送FIFO水印寄存器（ENET_TFWR）。 | 32 | R/W | 0000_0000h | 52.5.18/1781 |
| 180 | 接收描述符环形启动寄存器（ENET_RDSR）。 | 32 | R/W | 0000_0000h | 52.5.19/1782 |
| 184 | 发送缓冲区描述符环形启动寄存器（ENET_TDSR）。 | 32 | R/W | 0000_0000h | 52.5.20/1783 |
| 188 | 最大接收缓冲区大小寄存器（ENET_MRBR）。 | 32 | R/W | 0000_0000h | 52.5.21/1784 |

表继续在下一页...

## ENET内存图（续）

| 地址偏移（十六进制） | 注册名称 | 宽度（单位：比特） | 访问 | 重置值 | 节/页 |
|---|---|---|---|---|---|
| 190 | 接收FIFO部分满阈值（ENET_RSFL）。 | 32 | R/W | 0000_0000h | 52.5.22/1785 |
| 194 | 接收FIFO部分空阈值（ENET_RSEM）。 | 32 | R/W | 0000_0000h | 52.5.23/1785 |
| 198 | 接收FIFO几乎清空的阈值（ENET_RAEM）。 | 32 | R/W | 0000_0004h | 52.5.24/1786 |
| 19C | 接收FIFO几乎满的阈值（ENET_RAFL）。 | 32 | R/W | 0000_0004h | 52.5.25/1786 |
| 1A0 | 发送FIFO部分空阈值（ENET_TSEM）。 | 32 | R/W | 0000_0000h | 52.5.26/1787 |
| 1A4 | 发送FIFO几乎为空的阈值（ENET_TAEM）。 | 32 | R/W | 0000_0004h | 52.5.27/1787 |
| 1A8 | 发送FIFO几乎满的阈值（ENET_TAFL）。 | 32 | R/W | 0000_0008h | 52.5.28/1788 |
| 1AC | 发送包间间隙 (ENET_TIPG) | 32 | R/W | 0000_000Ch | 52.5.29/1788 |
| 1B0 | 帧截断长度 (ENET_FTRL) | 32 | R/W | 0000_07FFh | 52.5.30/1789 |
| 1C0 | 发送加速器功能配置（ENET_TACC）。 | 32 | R/W | 0000_0000h | 52.5.31/1789 |
| 1C4 | 接收加速器功能配置（ENET_RACC）。 | 32 | R/W | 0000_0000h | 52.5.32/1790 |
| 200 | 保留统计寄存器（ENET_RMON_T_DROP）。 | 32 | R | 0000_0000h | 52.5.33/1791 |
| 204 | Tx数据包计数统计寄存器（ENET_RMON_T_PACKETS）。 | 32 | R | 0000_0000h | 52.5.34/1792 |
| 208 | Tx广播包统计寄存器（ENET_RMON_T_BC_PKT）。 | 32 | R | 0000_0000h | 52.5.35/1792 |
| 20C | Tx多播包统计寄存器(ENET_RMON_T_MC_PKT) | 32 | R | 0000_0000h | 52.5.36/1793 |
| 210 | CRC/Align错误统计寄存器（ENET_RMON_T_CRC_ALIGN）的Tx数据包。 | 32 | R | 0000_0000h | 52.5.37/1793 |
| 214 | Tx包小于字节和良好CRC统计寄存器（ENET_RMON_T_UNDERSIZE）。 | 32 | R | 0000_0000h | 52.5.38/1793 |
| 218 | Tx 数据包 GT MAX_FL 字节和良好 CRC 统计寄存器 (ENET_RMON_T_OVERSIZE) | 32 | R | 0000_0000h | 52.5.39/1794 |
| 21C | 小于64字节的Tx数据包和坏的CRC统计寄存器（ENET_RMON_T_FRAG）。 | 32 | R | 0000_0000h | 52.5.40/1794 |
| 220 | Tx 数据包大于 MAX_FL 字节和坏 CRC 统计寄存器 (ENET_RMON_T_JAB) | 32 | R | 0000_0000h | 52.5.41/1795 |
| 224 | Tx碰撞计数统计寄存器（ENET_RMON_T_COL）。 | 32 | R | 0000_0000h | 52.5.42/1795 |
| 228 | Tx 64字节数据包统计寄存器（ENET_RMON_T_P64）。 | 32 | R | 0000_0000h | 52.5.43/1795 |

**ENET内存图（续）**

*表继续在下一页...*

## ENET内存图（续）

| 地址偏移（十六进制） | 注册名称 | 宽度（单位：比特） | 访问 | 重置值 | 节/页 |
|---|---|---|---|---|---|
| 22C | Tx 65-127字节数据包统计寄存器 (ENET_RMON_T_P65TO127) | 32 | R | 0000_0000h | 52.5.44/ 1796 |
| 230 | Tx 128-255字节数据包统计寄存器（ ENET_RMON_T_P128TO255）。 | 32 | R | 0000_0000h | 52.5.45/ 1796 |
| 234 | Tx 256-至511字节数据包统计寄存器 (ENET_RMON_T_P256TO511) | 32 | R | 0000_0000h | 52.5.46/ 1797 |
| 238 | Tx 512-至1023字节数据包统计寄存器 (ENET_RMON_T_P512TO1023) | 32 | R | 0000_0000h | 52.5.47/ 1797 |
| 23C | Tx 1024-至2047字节数据包统计寄存器 (ENET_RMON_T_P1024TO2047) | 32 | R | 0000_0000h | 52.5.48/ 1798 |
| 240 | 大于2048字节的Tx数据包统计寄存器 (ENET_RMON_T_P_GTE2048) | 32 | R | 0000_0000h | 52.5.49/ 1798 |
| 244 | Tx Octets统计寄存器（ENET_RMON_T_OCTETS）。 | 32 | R | 0000_0000h | 52.5.50/ 1798 |
| 248 | 保留统计寄存器（ENET_IEEE_T_DROP）。 | 32 | R | 0000_0000h | 52.5.51/ 1799 |
| 24C | 帧传输OK统计寄存器（ENET_IEEE_T_FRAME_OK）。 | 32 | R | 0000_0000h | 52.5.52/ 1799 |
| 250 | 用单次碰撞统计寄存器传输的帧（ENET_IEEE_T_1COL）。 | 32 | R | 0000_0000h | 52.5.53/ 1800 |
| 254 | 多次碰撞传输的帧统计寄存器（ENET_IEEE_T_MCOL）。 | 32 | R | 0000_0000h | 52.5.54/ 1800 |
| 258 | 延迟延迟后传输的帧统计寄存器（ENET_IEEE_T_DEF）。 | 32 | R | 0000_0000h | 52.5.55/ 1800 |
| 25C | 用晚期碰撞统计寄存器传输的帧（ENET_IEEE_T_LCOL）。 | 32 | R | 0000_0000h | 52.5.56/ 1801 |
| 260 | 碰撞过多时传输的帧数统计寄存器（ENET_IEEE_T_EXCOL）。 | 32 | R | 0000_0000h | 52.5.57/ 1801 |
| 264 | 用Tx FIFO Underrun统计寄存器传输的帧（ ENET_IEEE_T_MACERR）。 | 32 | R | 0000_0000h | 52.5.58/ 1802 |
| 268 | 用载波感应错误统计寄存器传输的帧（ ENET_IEEE_T_CSERR）。 | 32 | R | 0000_0000h | 52.5.59/ 1802 |
| 26C | 保留统计寄存器（ENET_IEEE_T_SQE）。 | 32 | R (读数0) | 0000_0000h | 52.5.60/ 1802 |
| 270 | 流量控制暂停帧传输统计寄存器（ENET_IEEE_T_FDXFC）。 | 32 | R | 0000_0000h | 52.5.61/ 1803 |
| 274 | 误差统计寄存器（ENET_IEEE_T_OCTETS_OK）中传输的帧的八位数。 | 32 | R | 0000_0000h | 52.5.62/ 1803 |
| 284 | Rx数据包计数统计寄存器（ENET_RMON_R_PACKETS）。 | 32 | R | 0000_0000h | 52.5.63/ 1804 |
| 288 | Rx广播包统计寄存器（ENET_RMON_R_BC_PKT）。 | 32 | R | 0000_0000h | 52.5.64/ 1804 |

**ENET内存图（续）**

| | | | | | |
|---|---|---|---|---|---|
| 28C | Rx多播包统计寄存器（ENET_RMON_R_MC_PKT）。 | 32 | R | 0000_0000h | 52.5.65/ 1804 |

*表继续在下一页...*

## ENET内存图（续）

| 地址偏移（十六进制） | 注册名称 | 宽度（单位：比特） | 访问 | 重置值 | 节/页 |
|---|---|---|---|---|---|
| 290 | CRC/Align错误统计寄存器（ENET_RMON_R_CRC_ALIGN）的Rx数据包。 | 32 | R | 0000_0000h | 52.5.66/1805 |
| 294 | 小于64字节的Rx数据包和良好的CRC统计寄存器（ENET_RMON_R_UNDERSIZE）。 | 32 | R | 0000_0000h | 52.5.67/1805 |
| 298 | Rx数据包大于MAX_FL和良好CRC统计寄存器（ENET_RMON_R_OVERSIZE）。 | 32 | R | 0000_0000h | 52.5.68/1806 |
| 29C | 小于64字节的Rx数据包和坏CRC统计寄存器（ENET_RMON_R_FRAG）。 | 32 | R | 0000_0000h | 52.5.69/1806 |
| 2A0 | 大于 MAX_FL 字节的 Rx 包和坏 CRC 统计寄存器 (ENET_RMON_R_JAB) | 32 | R | 0000_0000h | 52.5.70/1806 |
| 2A4 | 保留统计寄存器（ENET_RMON_R_RESVD_0）。 | 32 | R (读数 0) | 0000_0000h | 52.5.71/1807 |
| 2A8 | Rx 64字节数据包统计寄存器（ENET_RMON_R_P64）。 | 32 | R | 0000_0000h | 52.5.72/1807 |
| 2AC | Rx 65-127-字节数据包统计寄存器 (ENET_RMON_R_P65TO127) | 32 | R | 0000_0000h | 52.5.73/1808 |
| 2B0 | Rx 128-255字节数据包统计寄存器 (ENET_RMON_R_P128TO255) | 32 | R | 0000_0000h | 52.5.74/1808 |
| 2B4 | Rx 256- to 511-Byte 数据包统计寄存器 (ENET_RMON_R_P256TO511) | 32 | R | 0000_0000h | 52.5.75/1808 |
| 2B8 | Rx 512- to 1023-Byte 数据包统计寄存器 (ENET_RMON_R_P512TO1023) | 32 | R | 0000_0000h | 52.5.76/1809 |
| 2BC | Rx 1024-至2047-字节数据包统计寄存器 (ENET_RMON_R_P1024TO2047) | 32 | R | 0000_0000h | 52.5.77/1809 |
| 2C0 | 大于2048字节的Rx数据包统计寄存器 (ENET_RMON_R_P_GTE2048) | 32 | R | 0000_0000h | 52.5.78/1810 |
| 2C4 | Rx Octets统计寄存器（ENET_RMON_R_OCTETS）。 | 32 | R | 0000_0000h | 52.5.79/1810 |
| 2C8 | 未正确计数的帧统计寄存器（ENET_IEEE_R_DROP）。 | 32 | R | 0000_0000h | 52.5.80/1810 |
| 2CC | 帧接收OK统计寄存器（ENET_IEEE_R_FRAME_OK）。 | 32 | R | 0000_0000h | 52.5.81/1811 |
| 2D0 | CRC错误统计寄存器（ENET_IEEE_R_CRC）收到的帧。 | 32 | R | 0000_0000h | 52.5.82/1811 |
| 2D4 | 收到的帧中有对齐错误统计寄存器（ENET_IEEE_R_ALIGN）。 | 32 | R | 0000_0000h | 52.5.83/1812 |
| 2D8 | 接收FIFO溢出计数统计寄存器（ENET_IEEE_R_MACERR）。 | 32 | R | 0000_0000h | 52.5.84/1812 |
| 2DC | 流量控制暂停帧接收统计寄存器（ENET_IEEE_R_FDXFC）。 | 32 | R | 0000_0000h | 52.5.85/1812 |
| 2E0 | 无错误接收的帧的八位数统计寄存器（ | 32 | R | 0000_0000h | 52.5.86/1813 |

**ENET内存图（续）**

| | | | | | |
|---|---|---|---|---|---|
| | ENET_IEEE_R_OCTETS_OK）。 | | | | |
| 400 | 可调定时器控制寄存器（ENET_ATCR）。 | 32 | R/W | 0000_0000h | 52.5.87/1813 |

*表继续在下一页...*

ENET内存图（续）

| 地址偏移（十六进制） | 注册名称 | 宽度（单位：比特） | 访问 | 重置值 | 节/页 |
|---|---|---|---|---|---|
| 404 | 计时器值寄存器（ENET_ATVR）。 | 32 | R/W | 0000_0000h | 52.5.88/1815 |
| 408 | 定时器偏移寄存器（ENET_ATOFF）。 | 32 | R/W | 0000_0000h | 52.5.89/1816 |
| 40C | 定时器周期寄存器（ENET_ATPER）。 | 32 | R/W | 3B9A_CA00h | 52.5.90/1816 |
| 410 | 计时器校正寄存器（ENET_ATCOR）。 | 32 | R/W | 0000_0000h | 52.5.91/1817 |
| 414 | 时间戳时钟周期寄存器（ENET_ATINC）。 | 32 | R/W | 0000_0000h | 52.5.92/1817 |
| 418 | 最后传输的帧的时间戳 (ENET_ATSTMP) | 32 | R | 0000_0000h | 52.5.93/1818 |
| 604 | 计时器全局状态寄存器（ENET_TGSR）。 | 32 | R/W | 0000_0000h | 52.5.94/1818 |
| 608 | 定时器控制状态寄存器（ENET_TCSR0） | 32 | R/W | 0000_0000h | 52.5.95/1819 |
| 60C | 定时器比较捕获寄存器（ENET_TCCR0）。 | 32 | R/W | 0000_0000h | 52.5.96/1820 |
| 610 | 定时器控制状态寄存器（ENET_TCSR1） | 32 | R/W | 0000_0000h | 52.5.95/1819 |
| 614 | 定时器比较捕获寄存器（ENET_TCCR1）。 | 32 | R/W | 0000_0000h | 52.5.96/1820 |
| 618 | 定时器控制状态寄存器（ENET_TCSR2） | 32 | R/W | 0000_0000h | 52.5.95/1819 |
| 61C | 定时器比较捕获寄存器（ENET_TCCR2）。 | 32 | R/W | 0000_0000h | 52.5.96/1820 |
| 620 | 定时器控制状态寄存器（ENET_TCSR3） | 32 | R/W | 0000_0000h | 52.5.95/1819 |
| 624 | 定时器比较捕获寄存器（ENET_TCCR3）。 | 32 | R/W | 0000_0000h | 52.5.96/1820 |

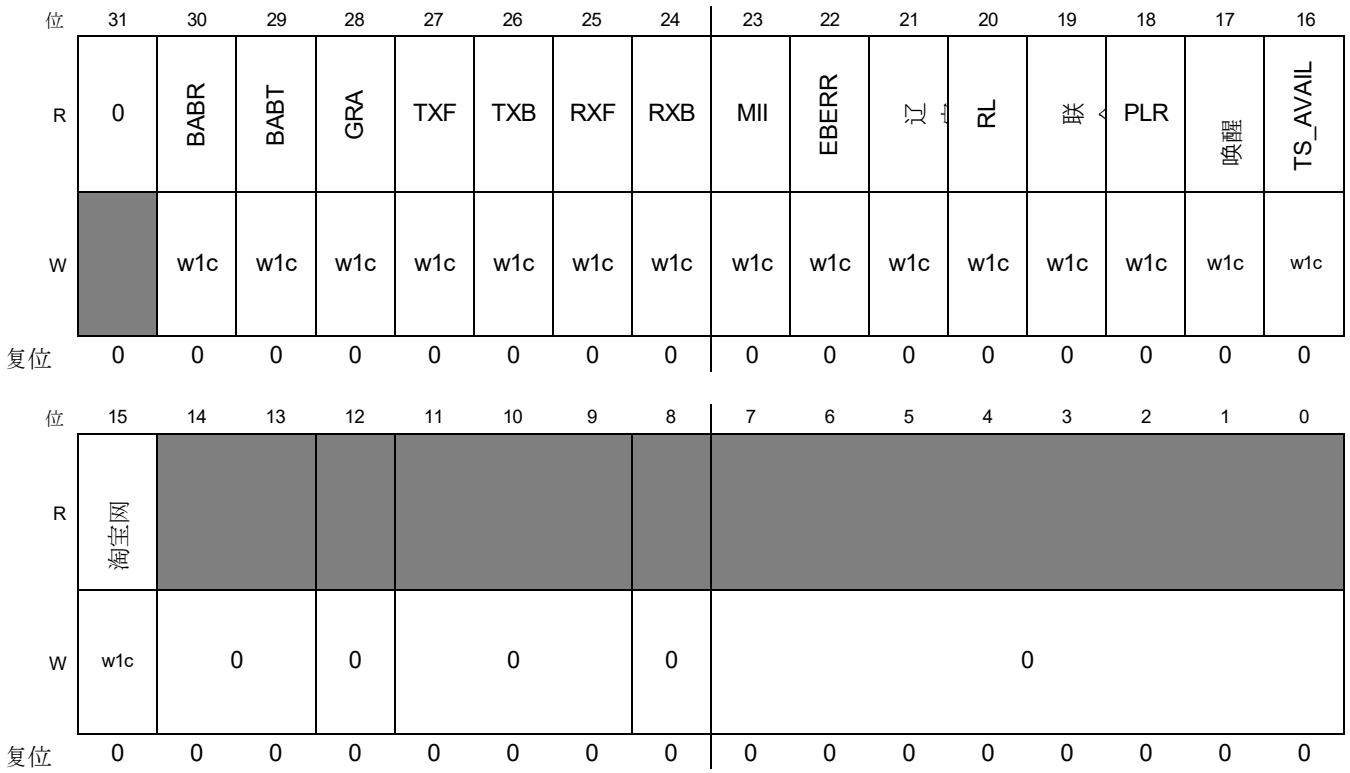
## 52.5.1 中断事件寄存器（ENET_EIR）。

当一个事件发生并设置了EIR中的一个位时，如果中断屏蔽寄存器（EIMR）中的相应位也被设置，则会发生中断。在EIR位上写1可以清除它，写0则没有效果。这个寄存器在硬件复位时被清空。

**注意事项**

TxBD[INT]和RxBD[INT]必须设置为1，以便在增强模式下设置相应的EIR寄存器标志，ENET_ECR[EN1588] = 1。遗留模式不需要启用这些标志。

地址。0h基数+4h偏移量=4h




## ENET_EIR字段描述

| 场地 | 描述 |
|---|---|
| 31<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 30<br>BABR | 咿呀学语接收错误<br><br>表示收到了一个长度超过RCR[MAX_FL]字节的帧。 |
| 29<br>BABT | 咿咿呀呀的传输错误<br><br>表示传输的帧长度超过了RCR[MAX_FL]字节。通常这种情况是由一个太长的帧被放入传输数据缓冲区引起的。截断不会发生。 |
| 28<br>GRA | 优雅的停顿完成<br><br>该中断在发射器完成当前正在传输的帧后进入暂停状态后被断言。关于导致优雅停止的条件，请参阅优雅传输停止（GTS）。<br><br>**注意：**GRA中断只有在TX转换到停止状态时才被引爆。如果该位通过写1被清除，而TX仍处于停止状态，则该位不会被再次设置。 |
| 27<br>TXF | 发送帧中断<br><br>表示一个帧已经被传输，最后一个相应的缓冲区描述符已经被更新。 |
| 26<br>TXB | 发送缓冲器中断 |

| | |
|---|---|
| | 表示一个传输缓冲区描述符已被更新。 |
| 25<br>RXF | 接收帧中断<br><br>表示已经收到一个帧，并且最后一个相应的缓冲区描述符已经被更新。 |

*表继续在下一页...*

## ENET_EIR字段描述（续）

| 场地 | 描述 |
|---|---|
| 24<br>RXB | 接收缓冲器中断<br><br>表示一个接收缓冲区描述符不是帧中的最后一个，已被更新。 |
| 23<br>MII | MII中断。<br><br>表示MII已经完成了请求的数据传输。 |
| 22<br>EBERR | 以太网总线错误<br><br>表示当一个uDMA事务正在进行时，发生了一个系统总线错误。当该位被设置时，ECR[ETHEREN]被清零，停止MAC的帧处理。当这种情况发生时，软件必须确保适当的行动，可能会重置系统，以恢复正常操作。 |
| 21<br>辽宁省 | 迟到的碰撞<br><br>表示在半双工模式下，碰撞发生在碰撞窗口（插槽时间）之外。帧因CRC不良而被截断，帧的剩余部分被丢弃。 |
| 20<br>RL | 碰撞重试限制<br><br>表示在16次连续的帧传输尝试中，每次都发生了碰撞。该帧没有被传输就被丢弃，开始传输下一个帧。这种错误只能在半双工模式下发生。 |
| 19<br>联合国 | 发送FIFO欠费<br><br>表示在传输完整的帧之前，发送FIFO已经空了。一个坏的CRC被附加到帧的片段上，帧的剩余部分被丢弃。 |
| 18<br>PLR | 有效载荷接收错误<br><br>表示收到的帧有一个有效载荷长度错误。参见帧长度/类型验证。欲了解更多信息，请参见帧长度/类型验证：有效载荷长度检查。 |
| 17<br>唤醒 | 节点唤醒请求指示<br><br>只读状态位，表示已经检测到一个神奇的数据包。只有当ECR[MAGICEN]被设置时才会发挥作用。 |
| 16<br>TS_AVAIL | 可用的传输时间戳<br><br>表示在ATSTMP寄存器中可以得到最后传输的定时帧的时间戳。 |
| 15<br>淘宝网 | 时间戳定时器<br><br>可调定时器到达周期事件。如果ATCR[PEREN]被设置，并且定时器按照ATPER寄存器中的周期设置进行包装，就可以产生一个周期事件中断。在设置ATCR[PEREN]之前，先设置定时器的周期值。 |
| 14-13<br>保留 | 这个字段是保留的。<br>这个只写的字段是保留的。它必须始终以0的值写入。 |
| 12<br>保留 | 这个字段是保留的。<br>这个只写的字段是保留的。它必须始终以0的值写入。 |
| 11-9<br>保留 | 这个字段是保留的。 |

| | |
|---|---|
| | 这个只写的字段是保留的。它必须始终以0的值写入。 |
| 8<br>保留 | 这个字段是保留的。<br>这个只写的字段是保留的。它必须始终以0的值写入。 |
| 保留 | 这个字段是保留的。<br>这个只写的字段是保留的。它必须始终以0的值写入。 |

## 52.5.2 中断屏蔽寄存器（ENET_EIMR）。

EIMR控制哪些中断事件被允许产生实际中断。一个硬件复位会清除这个寄存器。如果EIR和EIMR寄存器中的相应位被设置，就会产生一个中断。中断信号会一直保持，直到1被写入EIR字段（写1来清除）或0被写入EIMR字段。

地址。0h基数+8h偏移量=8h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | | BABR | BABT | GRA | TXF | TXB | RXF | RXB | MII | EBERR | 辽宁省 | RL | 联合国 | PLR | 唤醒 | TS_AVAIL |
| W | 0 | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 位 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 淘宝网 | | | | | | | | | | | | | | | |
| W | | 0 | | 0 | | 0 | | 0 | | | | 0 | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**ENET_EIMR字段描述**

| 场地 | 描述 |
|---|---|
| 31<br>保留 | 这个字段是保留的。<br>这个只写的字段是保留的。它必须始终以0的值写入。 |
| 30<br>BABR | BABR中断屏蔽<br>对应于中断源EIR[BABR]，决定一个中断条件是否可以产生中断。在每个模块时钟，EIR对中断源产生的信号进行采样。对应的EIR BABR字段反映了中断信号的状态，即使对应的EIMR字段被清除。<br>0 相应的中断源被屏蔽了。<br>1 相应的中断源没有被屏蔽。 |
| 29<br>BABT | BABT中断屏蔽<br>对应于中断源EIR[BABT]，决定一个中断条件是否可以产生一个中断。在每个模块时钟，EIR对中断源产生的信号进行采样。对应的EIR BABT字段反映了中断信号的状态，即使对应的EIMR字段被清除。<br>0 相应的中断源被屏蔽了。<br>1 相应的中断源没有被屏蔽。 |
| 28<br>GRA | GRA中断屏蔽<br>对应于中断源EIR[GRA]，决定一个中断条件是否可以产生一个中断。在每个模块时钟，EIR对中断源产生的信号进行采样。该 |

*表继续在下一页...*

## ENET_EIMR字段描述（续）

| 场地 | 描述 |
| --- | --- |
| | 对应的EIR GRA字段反映了中断信号的状态，即使对应的EIMR字段被清除。<br><br>0 相应的中断源被屏蔽了。<br>1 相应的中断源没有被屏蔽。 |
| 27<br>TXF | TXF中断屏蔽<br><br>对应于中断源EIR[TXF]，决定一个中断条件是否可以产生中断。在每个模块时钟，EIR对中断源产生的信号进行采样。对应的EIR TXF字段反映了中断信号的状态，即使对应的EIMR字段被清除。<br><br>0 相应的中断源被屏蔽了。<br>1 相应的中断源没有被屏蔽。 |
| 26<br>TXB | TXB中断屏蔽<br><br>对应于中断源EIR[TXB]，决定一个中断条件是否可以产生中断。在每个模块时钟，EIR对中断源产生的信号进行采样。对应的EIR TXF字段反映了中断信号的状态，即使对应的EIMR字段被清除。<br><br>0 相应的中断源被屏蔽了。<br>1 相应的中断源没有被屏蔽。 |
| 25<br>RXF | RXF中断屏蔽<br><br>对应于中断源EIR[RXF]，决定一个中断条件是否可以产生中断。在每个模块时钟，EIR对由中断源产生的信号进行采样。对应的EIR RXF字段反映了中断信号的状态，即使对应的EIMR字段被清除。 |
| 24<br>RXB | RXB中断屏蔽<br><br>对应于中断源EIR[RXB]，决定一个中断条件是否可以产生一个中断。在每个模块时钟，EIR对中断源产生的信号进行采样。对应的EIR RXB字段反映了中断信号的状态，即使对应的EIMR字段被清除。 |
| 23<br>MII | MII中断屏蔽<br><br>对应于中断源EIR[MII]，决定一个中断条件是否可以产生一个中断。在每个模块时钟，EIR对由中断源产生的信号进行采样。对应的EIR MII字段反映了中断信号的状态，即使对应的EIMR字段被清除。 |
| 22<br>EBERR | EBERR中断屏蔽<br><br>对应于中断源EIR[EBERR]，决定一个中断条件是否可以产生一个中断。在每个模块时钟，EIR对中断源产生的信号进行采样。对应的EIR EBERR字段反映了中断信号的状态，即使对应的EIMR字段被清除。 |
| 21<br>辽宁省 | LC中断屏蔽<br><br>对应于中断源EIR[LC]，决定一个中断条件是否可以产生一个中断。在每个模块时钟，EIR对由中断源产生的信号进行采样。即使相应的EIMR字段被清除，相应的EIR LC字段也反映了中断信号的状态。 |

*表继续在下一页...*

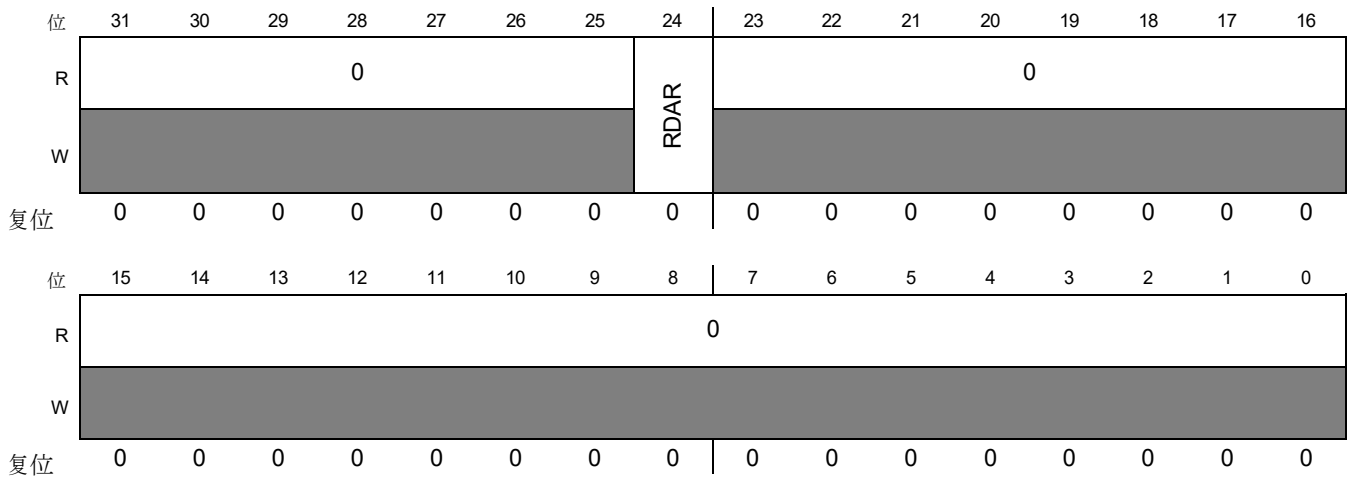
## ENET_EIMR字段描述（续）

| 场地 | 描述 |
|---|---|
| 20<br>RL | RL中断屏蔽<br><br>对应于中断源EIR[RL]，决定一个中断条件是否可以产生一个中断。在每个模块时钟，EIR对由中断源产生的信号进行采样。即使相应的EIMR字段被清除，相应的EIR RL字段也反映了中断信号的状态。 |
| 19<br>联合国 | 联合国中断屏蔽<br><br>对应于中断源EIR[UN]，决定一个中断条件是否可以产生一个中断。在每个模块时钟，EIR对由中断源产生的信号进行采样。即使相应的EIMR字段被清除，相应的EIR UN字段也反映了中断信号的状态。 |
| 18<br>PLR | PLR中断屏蔽<br><br>对应于中断源EIR[PLR]，决定一个中断条件是否可以产生一个中断。在每个模块时钟，EIR对中断源产生的信号进行采样。对应的EIR PLR字段反映了中断信号的状态，即使对应的EIMR字段被清除。 |
| 17<br>唤醒 | WAKEUP中断屏蔽<br><br>对应于中断源EIR[WAKEUP]寄存器，决定一个中断条件是否可以产生中断。在每个模块时钟，EIR对中断源产生的信号进行采样。即使相应的EIMR字段被清除，相应的EIR WAKEUP字段也反映了中断信号的状态。 |
| 16<br>TS_AVAIL | TS_AVAIL中断屏蔽<br><br>对应于中断源EIR[TS_AVAIL]寄存器，决定一个中断条件是否可以产生中断。在每个模块时钟，EIR对中断源产生的信号进行采样。对应的EIR TS_AVAIL字段反映了中断信号的状态，即使对应的EIMR字段被清除。 |
| 15<br>淘宝网 | TS_TIMER中断屏蔽<br><br>对应于中断源EIR[TS_TIMER]寄存器，决定一个中断条件是否可以产生中断。在每个模块时钟，EIR对中断源产生的信号进行采样。对应的EIR TS_TIMER字段反映了中断信号的状态，即使对应的EIMR字段被清除。 |
| 14-13<br>保留 | 这个字段是保留的。<br>这个只写的字段是保留的。它必须始终以0的值写入。 |
| 12<br>保留 | 这个字段是保留的。<br>这个只写的字段是保留的。它必须始终以0的值写入。 |
| 11-9<br>保留 | 这个字段是保留的。<br>这个只写的字段是保留的。它必须始终以0的值写入。 |
| 8<br>保留 | 这个字段是保留的。<br>这个只写的字段是保留的。它必须始终以0的值写入。 |
| 保留 | 这个字段是保留的。<br>这个只写的字段是保留的。它必须始终以0的值写入。 |

## 52.5.3 接收描述符活动寄存器（ENET_RDAR）。

RDAR是一个命令寄存器，由用户写入，表示接收描述符环已被更新，也就是说，驱动程序产生了空位的接收缓冲区。

地址。0h基数+10h偏移量=10h




**ENET_RDAR字段描述**

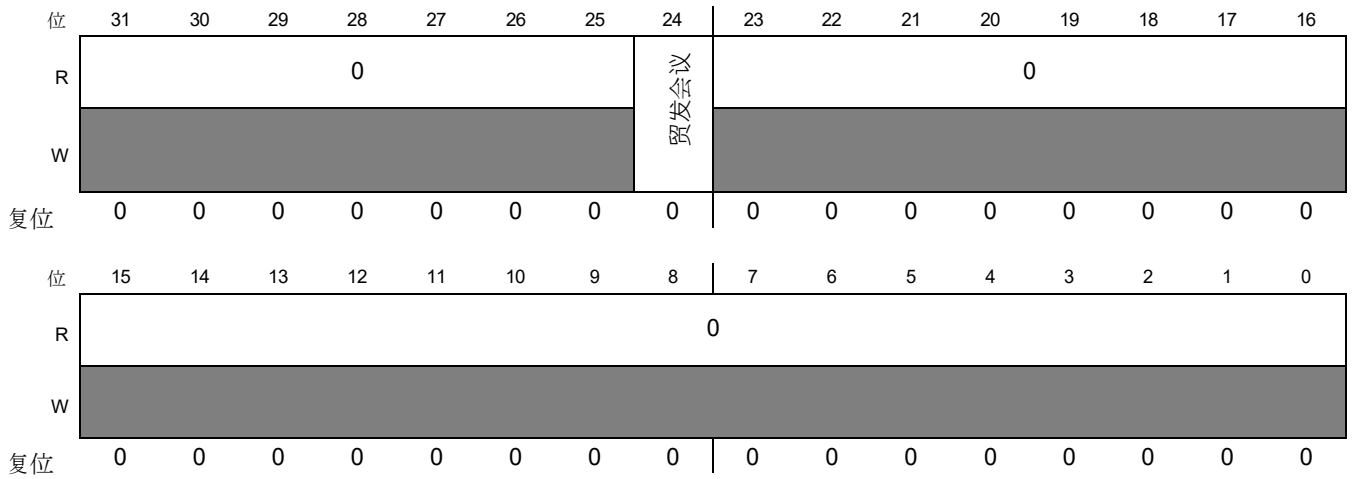
| 场地 | 描述 |
|---|---|
| 31-25<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 24<br>RDAR | 接收描述符激活<br><br>当此寄存器被写入时，无论写入的值是多少，总是设置为1。当接收环中没有其他空描述符时，该字段被MAC设备清零。当ECR[ETHEREN]从设置过渡到清除或ECR[RESET]被设置时，它也被清除。 |
| 保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |

## 52.5.4 发送描述符激活寄存器（ENET_TDAR）。

TDAR是一个命令寄存器，用户写它来表明发送描述符环已被更新，也就是说，发送缓冲区已被驱动器产生，缓冲区描述符中的就绪位被设置。

在复位时，当ECR[ETHEREN]从设置过渡到清零时，或者ECR[RESET]被设置时，TDAR寄存器被清零。

地址。0h基数+14h偏移量=14h



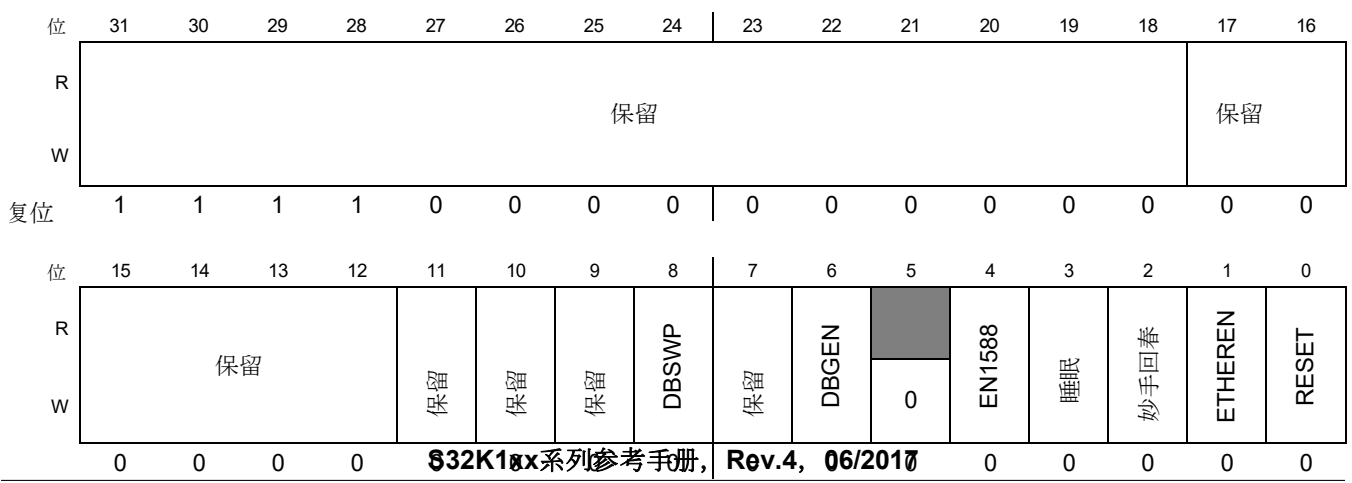

**ENET_TDAR字段描述**

| 场地 | 描述 |
|---|---|
| 31-25<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 24<br>贸发会议 | 传输描述符激活<br><br>当这个寄存器被写入时，无论写入的值是多少，总是被设置为1。当发送环中没有额外的就绪描述符时，该位被MAC设备清零。当ECR[ETHEREN]从设置过渡到清零或ECR[RESET]被设置时也会被清零。 |
| 保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |

## 52.5.5 以太网控制寄存器（ENET_ECR）

ECR是一个读/写用户寄存器，尽管硬件也可以改变这个寄存器中的字段。它控制以太网MAC的许多高级功能，包括通过EN1588字段支持传统的FEC。

地址。0h基数+24h偏移量=24h



复位

## ENET_ECR字段描述

| 场地 | 描述 |
|---|---|
| 31-18<br>保留 | 这个字段是保留的。<br>始终将111100000000b写到这个字段。 |
| 17-12<br>保留 | 这个字段是保留的。<br>始终将0写到这个字段。 |
| 11<br>保留 | 这个字段是保留的。<br>始终将0写到这个字段。 |
| 10<br>保留 | 这个字段是保留的。<br>始终将0写到这个字段。 |
| 9<br>保留 | 这个字段是保留的。<br>始终将0写到这个字段。 |
| 8<br>DBSWP | 描述符字节交换启用<br><br>调换缓冲区描述符的字节位置。<br><br>**注意：**复位后该字段必须写为1。<br><br>0　缓冲区描述符的字节没有被交换，以支持big-endian设备。<br>1　缓冲区描述符的字节被调换，以支持小-endian设备。 |
| 7<br>保留 | 这个字段是保留的。<br>始终将0写到这个字段。 |
| 6<br>DBGEN | 调试启用<br><br>当设备进入调试模式时，使MAC进入硬件冻结模式。<br><br>0　MAC在调试模式下继续运行。<br>1　当处理器处于调试模式时，MAC会进入硬件冻结模式。 |
| 5<br>保留 | 这个字段是保留的。<br>这个只写的字段是保留的。它必须始终以0的值写入。 |
| 4<br>EN1588 | EN1588启用<br><br>使得MAC的功能得到加强。<br><br>0　启用传统的FEC缓冲区描述符和功能。<br>1　启用了增强的帧时间标记功能。 |
| 3<br>睡眠 | 启用睡眠模式<br><br>0　正常操作模式。<br>1　睡眠模式。 |

| | |
|---|---|
| 2<br>妙手回春 | 启用魔法数据包检测 启用/禁用魔法数据包检<br><br>测。<br><br>**注意：**MAGICEN只有在设置了SLEEP字段时才有意义。如果MAGICEN被设置，改变SLEEP字段<br>启用/禁用睡眠模式和魔法数据包检测。<br><br>0 魔法检测逻辑失效。<br>1 MAC核心检测神奇的数据包，当检测到一个帧时，会发出EIR[WAKEUP]。 |
| 1<br>ETHEREN | 启用以太网 |

*表继续在下一页...*

**ENET_ECR字段描述（续）**

| 场地 | 描述 |
|---|---|
| | 启用/停用以太网MAC。当MAC被禁用时，中止传输帧的缓冲区描述符不会被更新。uDMA、缓冲区描述符和FIFO控制逻辑被重置，包括缓冲区描述符和FIFO指针。<br>硬件在以下条件下清除这个字段。<br>• RESET是由软件设置的<br>• 一个错误条件导致EBERR字段被设置。<br>**注意。** - ETHEREN必须在ENET配置/设置/初始化过程中的最后一步进行设置，只有*在*所有其他与ENET相关的寄存器都配置完毕*后才能*设置。<br>• 如果ETHEREN被软件清除为0，那么下次ETHEREN被设置时，由于之前的待处理中断，EIR中断必须被清除为0。<br>0 在任何当前传输的帧上附加一个坏的CRC后，接收立即停止，传输停止。<br>1 启用MAC，可以进行接收和传输。 |
| 0<br>RESET | 以太网MAC复位<br>当这个字段被设置时，它将清除ETHEREN字段。 |

## 52.5.6 MII管理帧寄存器（ENET_MMFR）。

除非MSCR被编程为零，否则写到MMFR会触发一个管理帧事务到PHY设备。

如果在对MMFR的写入过程中，MSCR从零变为非零，则会产生一个MII帧，其中包含之前写入MMFR的数据。如果MSCR当前为零，这允许MMFR和MSCR以任何顺序被编程。

如果MMFR寄存器在帧生成过程中被写入，帧的内容就会被改变。软件必须使用EIR[MII]中断指示来避免在帧生成过程中写入MMFR寄存器。

地址。0h基数+40h偏移量=40h

| 位 | 31 30 29 | 28 | 27 26 25 24 23 | 22 21 20 19 18 | 17 16 | 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0 |
|---|---|---|---|---|---|---|
| R<br>W | ÃÃÃ | 职务 | PA | RA | 淘宝网 | 数据 |
| 复位 | 0 0 0 | 0 | 0 0 0 0 0 | 0 0 0 0 0 | 0 0 | 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 |

**ENET_MMFR字段描述**

| 场地 | 描述 |
|---|---|
| 31-30<br>ÃÃÃ | 帧的起始分隔符<br>正确值见表52-39（条款22）或表52-41（条款45）。 |

| | |
|---|---|
| 29-28<br>职务 | 操作代码 |

*表继续在下一页...*

**ENET_MMFR字段描述（续）**

| 场地 | 描述 |
|---|---|
| | 正确值见表52-39（条款22）或表52-41（条款45）。 |
| 27-23<br>PA | PHY地址<br><br>正确值见表52-39（条款22）或表52-41（条款45）。 |
| 22-18<br>RA | 注册地址<br><br>正确值见表52-39（条款22）或表52-41（条款45）。 |
| 17-16<br>淘宝网 | 转身<br><br>这个字段必须被编程为10，以产生一个有效的MII管理帧。 |
| 数据 | 管理框架数据<br><br>这是一个用于向PHY寄存器写入或从其读取数据的字段。 |

## 52.5.7 MII速度控制寄存器（ENET_MSCR）。

MSCR提供了对MII时钟（MDC引脚）频率的控制，并允许在MII管理帧上有一个前导音的下降。

MII_SPEED字段必须被编程为一个值，以提供一个小于或等于2.5MHz的MDC频率，以符合IEEE 802.3
MII规范。MII_SPEED必须被设置为一个非零值，以作为读或写管理帧的来源。在管理帧完成后，MSCR寄存器可以选择清零以关闭MDC。产生的MDC信号有50%的占空比，除非在操作过程中MII_SPEED发生变化。这种变化在MDC的上升沿或下降沿之后生效。

例如，如果内部模块时钟(即外围总线时钟)为25
MHz，将MII_SPEED编程为0x4会导致MDC如下式所示。

MII时钟频率=25MHz/（（4+1）×2）=2.5MHz

下表显示了MII_SPEED的最佳值与IPS总线时钟频率的关系。

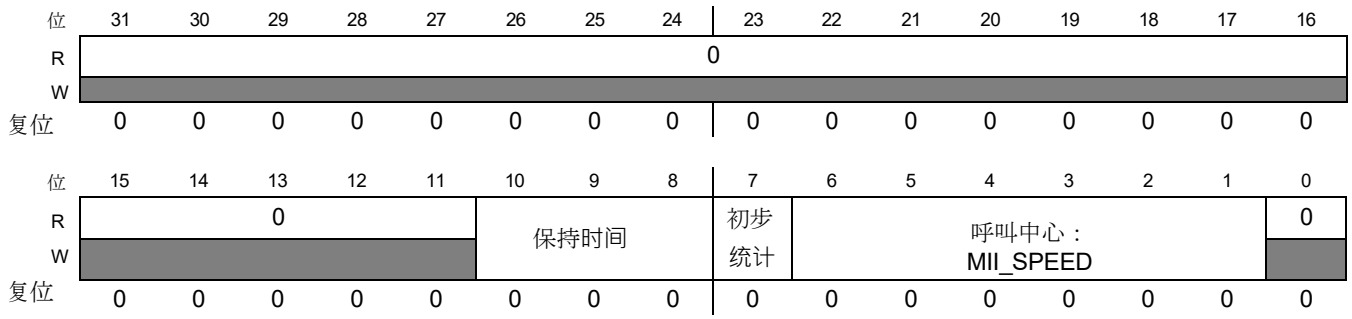
**表52-2.MSCR的编程实例**

| 内部模块时钟频率 | mscr [mii_speed] | MDC频率 |
|---|---|---|
| 25兆赫 | 0x4 | 2.50 MHz |
| 33兆赫 | 0x6 | 2.36 MHz |
| 40兆赫 | 0x7 | 2.50 MHz |
| 50兆赫 | 0x9 | 2.50 MHz |

| | | |
|---|---|---|
| 66兆赫 | 0xD | 2.36 MHz |

**内存图/寄存器定义**

地址。0h基数+44h偏移量=44h



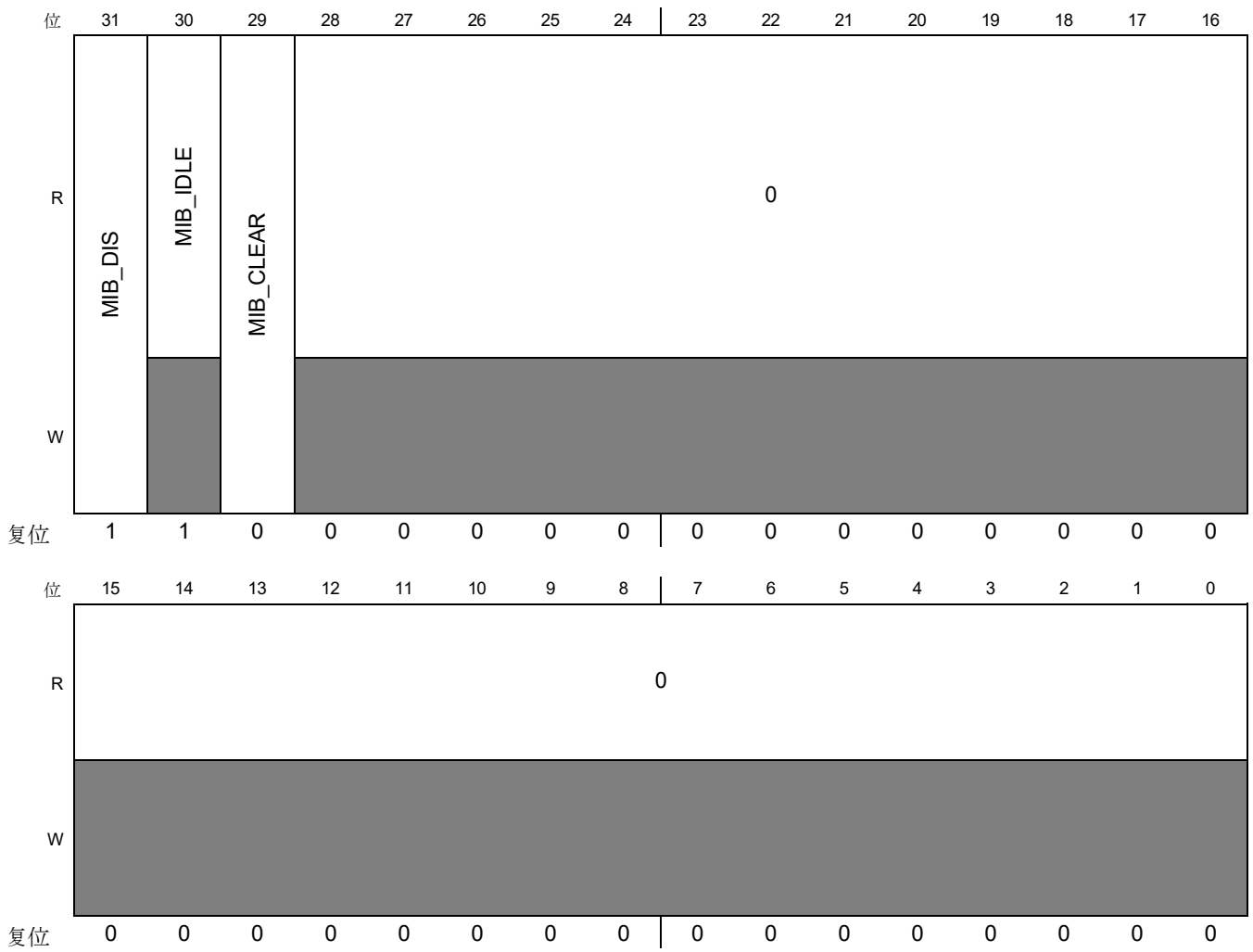

## ENET_MSCR字段描述

| 场地 | 描述 |
|---|---|
| 31-11<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 10-8<br>保持时间 | MDIO输出的保持时间<br><br>IEEE802.3第22条规定，MDIO输出的保持时间至少为10ns。根据主机总线的频率，该设置可能需要增加。<br><br>00 1个内部模块时钟周期<br>01 2个内部模块时钟周期<br>0103 内部模块时钟周期<br>1118个内部模块时钟周期 |
| 7<br>帐户 | 禁用序言<br><br>启用/禁用在MII管理帧中预留序言。如果所连接的PHY设备不需要，MII标准允许放弃前奏。<br><br>0 启用序言。<br>1 序言(32个1)没有预加到MII管理帧中。 |
| 6-1<br>呼叫中心：<br>MII_SPEED | MII速度<br><br>控制MII管理接口时钟（MDC）相对于内部模块时钟的频率。这个字段的值为0，则关闭MDC，使其处于低电压状态。任何非零值都会导致MDC的频率为。<br><br>内部模块时钟频率的1/((MII_SPEED + 1) x 2) |
| 0<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |

## 52.5.8 MIB控制寄存器（ENET_MIBC）。

MIBC是一个读/写寄存器，控制和观察MIB块的状态。访问这个寄存器可以禁用MIB块的操作或清除MIB计数器。MIB_DIS字段重设为1。

地址。0h base + 64h offset = 64h




**ENET_MIBC字段描述**

| 场地 | 描述 |
|---|---|
| 31<br>MIB_DIS | 禁用MIB逻辑<br><br>如果这个控制字段被设置。<br><br>0MIB逻辑被启用。<br>1MIB逻辑被禁用。MIB逻辑停止，不更新任何MIB计数器。 |

*表继续在下一页...*

**ENET_MIBC字段描述（续）**

| 场地 | 描述 |
|---|---|
| 30<br>MIB_IDLE | MIB闲置<br>0 该MIB块正在更新MIB计数器。<br>1 该MIB块目前没有更新任何MIB计数器。 |
| 29<br>MIB_CLEAR | MIB清除<br>**注意：**这个字段不是自动清除的。要清除MIB计数器，必须先设置然后再清除这个字段。<br>0见上注。<br>1所有统计计数器被重置为0。 |
| 保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |

## 52.5.9 接收控制寄存器（ENET_RCR）。

地址。0h基数+84h偏移量=84h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | GRS | NLC | MAX_FL | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 |

| 位 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | CFEN | CRCFWD | PAUFWD | PADEN | | | 湿度 | RMII_MODE | | | FCE | BC_REJ | PROM | MII_MODE | ÄÄDRT | LOOP |
| W | | | | | 0 | | | | 0 | 0 | | | | | | |
| 1780 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

复位

## ENET_RCR字段描述

| 场地 | 描述 |
|---|---|
| 31<br>GRS | 优雅的接收已停止<br><br>只读状态，表示MAC接收数据通路已经停止。 |
| 30<br>NLC | 有效载荷长度检查 禁用 启用/禁用有效载荷<br><br>长度检查。<br><br>0 有效载荷长度检查被禁用。<br>1 内核通过帧长度/类型字段检查帧的有效载荷长度。错误会在EIR[PLC]字段中显示。 |
| 29-16<br>MAX_FL | 最大框架长度<br><br>重置为十进制1518。长度从DA开始测量，包括帧末的CRC。发送帧长于MAX_FL会导致BABT中断发生。长于MAX_FL的接收帧会导致BABR中断发生，并在帧末接收缓冲描述符中设置LG字段。如果支持VLAN标签，建议编程的默认值为1518或1522。 |
| 15<br>CFEN | MAC控制帧启用 启用/禁用MAC控制帧。<br><br>0 除0x0001（暂停帧）以外的任何操作码的MAC控制帧都被接受并转发到客户接口。<br>1 具有0x0001（暂停帧）以外的任何操作码的MAC控制帧会被默默地丢弃。 |
| 14<br>CRCFWD | 终止/转发接收的CRC<br><br>指定接收到的帧的CRC字段是被传送还是被剥离。<br><br>**注意：**如果填充功能被启用（PADEN=1），CRCFWD被忽略，CRC字段被检查并总是被终止和删除。<br><br>0 收到的帧的CRC字段被传送到用户应用程序。<br>1 CRC字段被从帧中剥离。 |
| 13<br>PAUFWD | 终止/转发暂停的帧数<br><br>指定暂停帧是被终止还是被转发。<br><br>0 暂停帧在MAC中被终止和丢弃。<br>1 暂停帧被转发到用户应用程序。 |
| 12<br>PADEN | 启用帧填充 在接收时删除<br><br>指定MAC是否从接收的帧中移除填充物。<br><br>0 MAC在接收时不去除填充物。<br>1 从收到的帧中去除填充物。 |
| 11-10<br>保留 | 这个字段是保留的。<br>这个只写的字段是保留的。它必须始终以0的值写入。 |

| | |
|---|---|
| 9<br>湿度 | 启用RMII的10-Mbit/s模式 .<br><br>0 100-Mbit/s操作。<br>1 10-Mbit/s操作。 |

*表继续在下一页...*

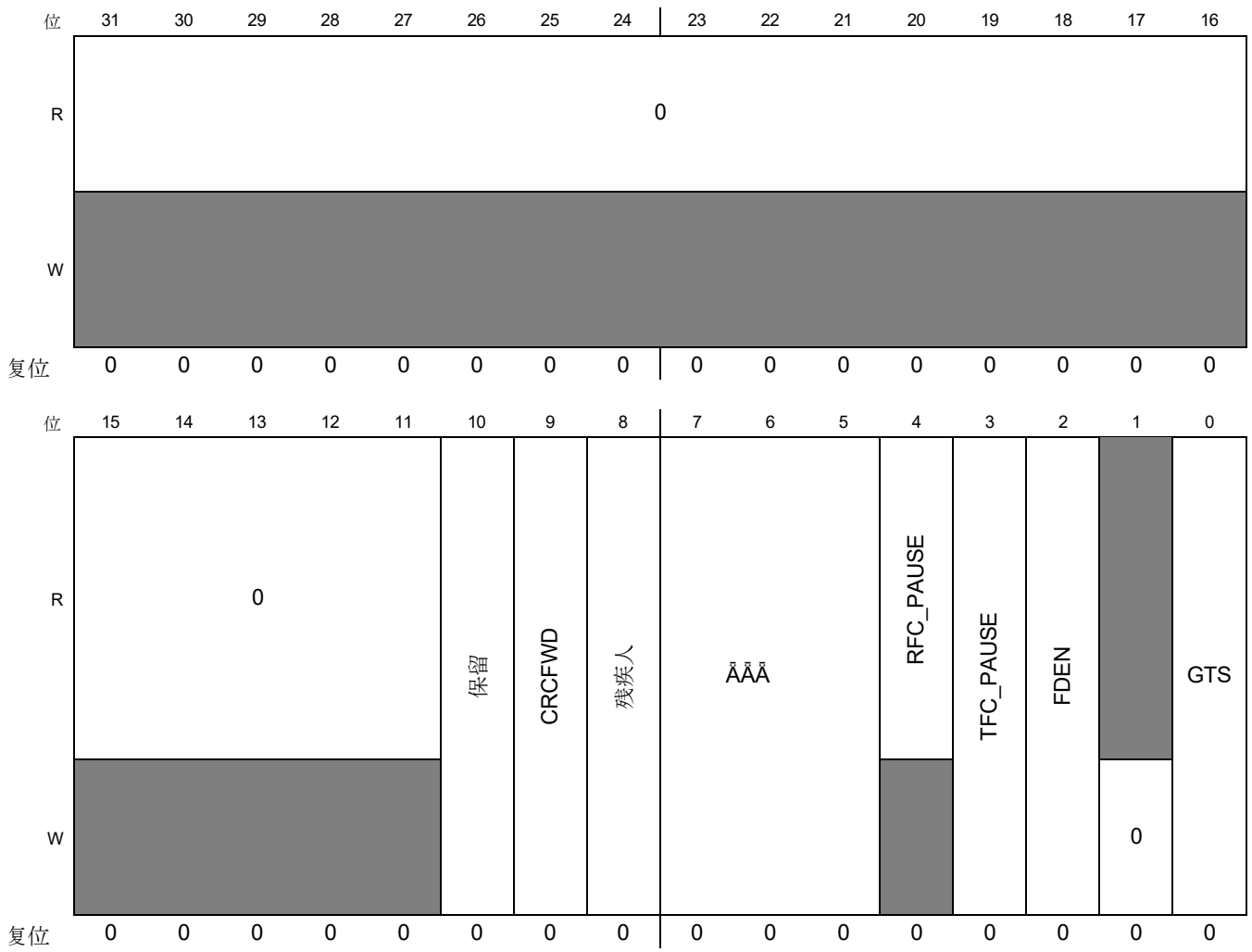
## ENET_RCR字段描述（续）

| 场地 | 描述 |
|---|---|
| 8<br>RMII_MODE | 启用RMII模式<br><br>指定MAC是否被配置为MII模式或RMII操作。<br><br>0 为MII模式配置的MAC。<br>1 为RMII操作配置的MAC。 |
| 7<br>保留 | 这个字段是保留的。<br>这个只写的字段是保留的。它必须始终以0的值写入。 |
| 6<br>保留 | 这个字段是保留的。<br>这个只写的字段是保留的。它必须始终以0的值写入。 |
| 5<br>FCE | 启用流量控制<br><br>如果设置，接收器检测PAUSE帧。在检测到PAUSE帧时，发射器在给定的时间内停止传输数据帧。 |
| 4<br>BC_REJ | 广播帧拒绝<br><br>如果设置，除非PROM字段被设置，否则目的地地址（DA）等于0xFFFF_FFFF_FFFF的帧会被拒绝。如果BC_REJ和PROM被设置，具有广播DA的帧被接受，并且在接收缓冲区描述符中设置MISS（M）。 |
| 3<br>PROM | 淫乱模式<br><br>无论地址是否匹配，所有帧都被接受。<br><br>0 残废。<br>1 已启用。 |
| 2<br>MII_MODE | 媒体独立接口模式 这个字段必须始终<br><br>被设置。<br><br>0 保留。<br>1 MII或RMII模式，由RMII_MODE字段表示。 |
| 1<br>ÃÃDRT | 禁用发送时的接收<br><br>0 接收路径与发射路径独立运行（即全双工模式）。也可用于监测半双工模式下的发射活动。<br>1 发送时禁止接收帧。(通常用于半双工模式）。 |
| 0<br>LOOP | 内部环回<br><br>这是一个MII内部环回，因此MII_MODE必须写为1，RMII_MODE必须写为0。<br><br>0 禁止环回。<br>1 传输的帧在设备内部循环，传输的MII输出信号不被主张。DRT必须被清除。 |

## 52.5.10 发送控制寄存器（ENET_TCR）

TCR是读/写的，用于配置传输块。该寄存器在系统复位时被清零。只有当ECR[ETHEREN]被清零时，FDEN才能被修改。

地址。0h基数 + C4h偏移量 = C4h




**ENET_TCR字段描述**

| 场地 | 描述 |
|---|---|
| 31-11<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 10<br>保留 | 这个字段是保留的。<br>这个字段是读/写，必须设置为0。 |
| 9<br>CRCFWD | 从应用中转发带有CRC的帧 |

*表继续在下一页...*

## ENET_TCR字段描述（续）

| 场地 | 描述 |
|---|---|
| | 0 TxBD[TC]控制帧是否有来自应用程序的CRC。<br>1 发送器不在传输的帧上附加任何CRC，因为它期待着来自应用的带有CRC的帧。 |
| 8<br>残疾人 | 发送时设置MAC地址<br><br>0 源MAC地址不被MAC修改。<br>1 MAC根据ADDSEL将源MAC地址与编程的MAC地址进行覆盖。 |
| 7-5<br>ÃÃÃ | 发送时选择源MAC地址<br><br>如果ADDINS被设置，表示覆盖源MAC地址的MAC地址。000 在PADDR1/2寄存器上<br><br>编程的节点 MAC地址。<br>100 保留。<br>101 保留。<br>110 保留。 |
| 4<br>RFC_PAUSE | 接收帧控制暂停<br><br>当收到全双工流量控制暂停帧时，这个状态字段被设置，发射器暂停的时间在这个暂停帧中定义。当暂停持续时间完成时，该字段自动清除。 |
| 3<br>TFC_PAUSE | 发送帧控制暂停<br><br>暂停帧传输。当这个字段被设置时，EIR[GRA]被设置。随着数据帧传输的停止，MAC传输一个MAC控制PAUSE帧。接下来，MAC清除TFC_PAUSE并恢复传输数据帧。如果发射器因用户断言GTS或接收PAUSE帧而暂停，MAC可以继续发射一个MAC控制PAUSE帧。<br><br>0 没有传输PAUSE帧。<br>1 MAC在当前传输完成后停止传输数据帧。 |
| 2<br>FDEN | 全双工启用<br><br>如果这个字段被设置，帧的传输与载波感应和碰撞输入无关。只有当ECR[ETHEREN]被清零时，才能修改该位。 |
| 1<br>保留 | 这个字段是保留的。<br>这个只写的字段是保留的。它必须始终以0的值写入。 |
| 0<br>GTS | 优雅的传输停止<br><br>当这个字段被设置时，MAC在当前传输的任何帧完成后停止传输，EIR[GRA]被设置。如果当前没有进行帧传输，GRA中断立即被断言。传输结束后，清除GTS重新开始。然后，传输FIFO中的下一个帧被传输。如果当GTS被设置时，在传输过程中发生早期碰撞，碰撞后传输停止。在GTS被清除后，该帧再次被传输。发送FIFO中可能有旧的帧，当GTS被重新置位时发送。为了避免这种情况，在GRA中断后清除ECR[ETHEREN]。 |

## 52.5.11 物理地址下限寄存器（ENET_PALR）。

PALR包含地址识别过程中使用的48位地址的低32位（字节0、1、2、3），与具有单独DA的接收帧的目的地址（DA）字段进行比较。此外，在传输PAUSE帧时，该寄存器被用于六字节源地址字段的第0至3字节。

地址。0h基数+E4h偏移量=E4h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R<br>W | PADDR1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**ENET_PALR字段描述**

| 场地 | 描述 |
|---|---|
| PADDR1 | 暂停地址<br><br>6个字节的个人地址中的字节0（位31：24）、1（位23：16）、2（位15：8）和3（位7：0）用于精确匹配和PAUSE帧中的源地址域。 |

## 52.5.12 物理地址上位寄存器（ENET_PAUR）。

PAUR包含地址识别过程中使用的48位地址的前16位（字节4和5），用于与具有独立DA的接收帧的目的地址（DA）字段进行比较。此外，该寄存器在传输PAUSE帧时用于6字节源地址字段的第4和第5字节。PAUR的第15:0位包含一个用于传输PAUSE帧的常数类型字段（0x8808）。

地址。0h基数+E8h偏移量=E8h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | PADDR2 | | | | | | | | | | | | | | | | 类型 | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |

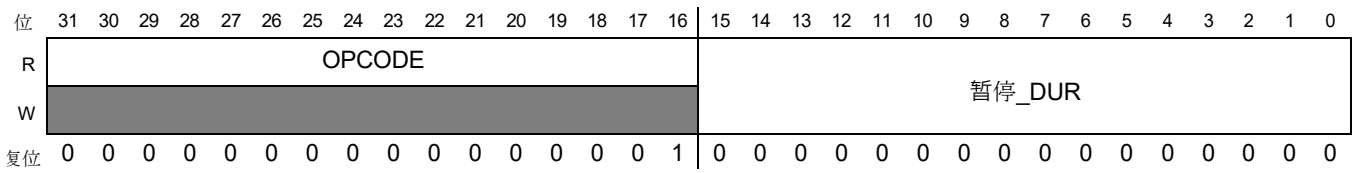
**ENET_PAUR字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>PADDR2 | 用于精确匹配的6字节个人地址的第4字节（第31:24位）和第5字节（第23:16位），以及PAUSE帧中的源地址域。 |
| 类型 | 暂停帧中的类型字段<br><br>这些字段的恒定值为0x8808。 |

## 52.5.13 操作码/暂停持续时间寄存器（ENET_OPD）。

OPD是可以读/写的。该寄存器包含在传输PAUSE帧时使用的16位操作码和16位暂停时间字段。操作码字段是一个常量值，0x0001。当另一个节点检测到一个PAUSE帧时，该节点会暂停传输，持续时间在暂停持续时间字段中指定。这个寄存器的低16位没有被重置，你必须初始化它。

地址。0h基数 + ECh偏移量 = ECh



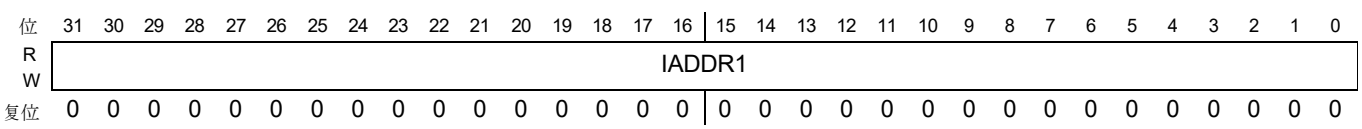

**ENET_OPD字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>OPCODE | PAUSE帧中的操作码字段<br><br>这些字段的恒定值为0x0001。 |
| 暂停_DUR | 暂停时间<br><br>在PAUSE帧中使用的暂停持续时间字段。 |

## 52.5.14 描述符个人上位地址寄存器（ENET_IAUR）。

IAUR包含64位独立地址哈希表的前32位。地址识别过程使用这个表来检查是否与带有单个DA的接收帧的目标地址（DA）字段可能匹配。这个寄存器没有被重置，你必须初始化它。

地址。0h基数+118h偏移量=118h




**ENET_IAUR字段描述**

| 场地 | 描述 |
|---|---|
| IADDR1 | 包含64位哈希表的前32位，用于地址识别过程中的单播地址接收帧。IADDR1的第31位包含哈希索引位63。IADDR1的第0位包含哈希索引位32。 |

## 52.5.15 描述符个人下位地址寄存器（ENET_IALR）。

IALR包含64位个人地址哈希表的低32位。地址识别过程使用这个表来检查是否与具有独立DA的接收帧的DA字段可能匹配。这个寄存器没有被重置，你必须初始化它。

地址。0h基数+11Ch偏移量=11Ch

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R<br>W | IADDR2 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**ENET_IALR字段描述**

| 场地 | 描述 |
|---|---|
| IADDR2 | 包含64位哈希表的低32位，用于地址识别过程中的单播地址接收帧。IADDR2的第31位包含哈希索引位31。IADDR2的第0位包含哈希索引位0。 |

## 52.5.16 描述符组上位地址寄存器（ENET_GAUR）。

GAUR包含64位哈希表的前32位，用于接收带有多播地址的帧的地址识别过程。你必须初始化这个寄存器。

地址。0h base + 120h offset = 120h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R<br>W | GADDR1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

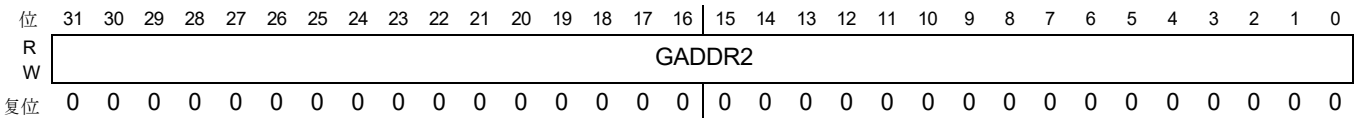
**ENET_GAUR字段描述**

| 场地 | 描述 |
|---|---|
| GADDR1 | 包含64位哈希表的前32位，用于接收具有多播地址的帧的地址识别过程。GADDR1的第31位包含哈希索引位63。GADDR1的第0位包含哈希索引位32。 |

## 52.5.17 描述符组下位地址寄存器（ENET_GALR）。

GALR包含64位哈希表的低32位，该表在地址识别过程中用于接收带有多播地址的帧。你必须初始化这个寄存器。

地址。0h base + 124h offset = 124h




**ENET_GALR字段描述**

| 场地 | 描述 |
|---|---|
| GADDR2 | 包含64位哈希表的低32位，用于地址识别过程，用于接收具有多播地址的帧。GADDR2的第31位包含哈希索引位31。GADDR2的第0位包含哈希索引位0。 |

## 52.5.18 发送FIFO水印寄存器（ENET_TFWR）。

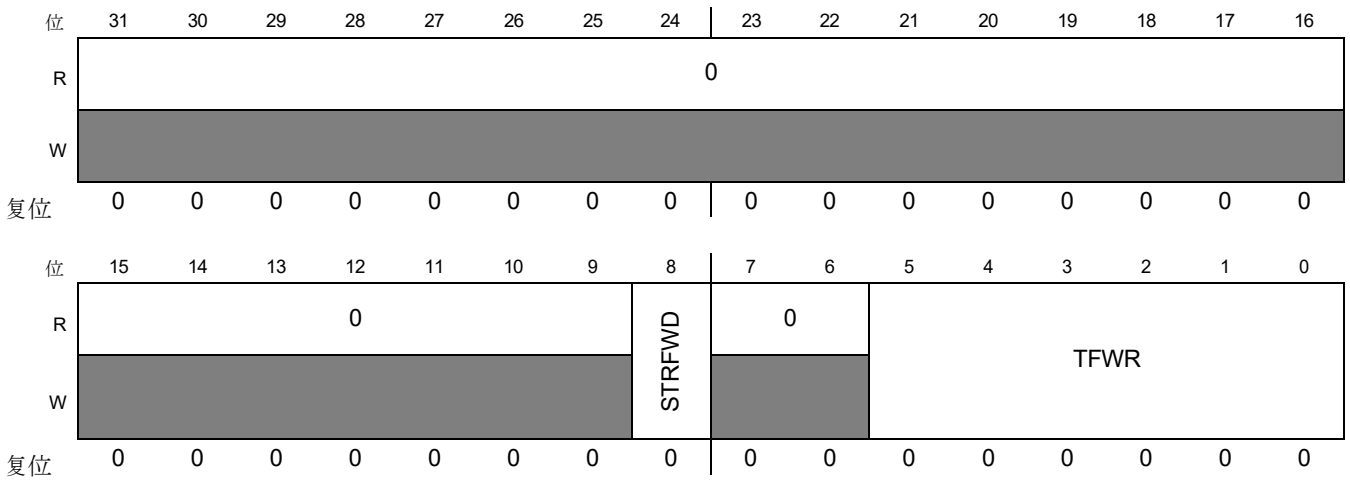
如果TFWR[STRFWD]被清除，TFWR[TFWR]控制在开始传输帧之前，发送FIFO中需要的数据量。这允许你最大限度地减少传输延迟（TFWR=00或01）或允许由于对系统总线的争夺而产生较大的总线访问延迟（TFWR=11）。将水印设置为高值可以最大限度地减少由于争夺系统总线而导致的发送FIFO不足的风险。与TFWR字段相关的字节数可能需要修改，以符合特定的系统要求，例如，传输数据uDMA通道的最坏情况下的总线访问延迟。

当FIFO水平达到TFWR字段的值，并且STR_FWD被设置为'0'时，MAC发送控制逻辑甚至在FIFO中可用的帧末之前就开始传输帧（cut-through操作）。

如果一个完整的帧的大小小于用TFWR编程的阈值，MAC也将该帧传送到线路上。

要在传输路径上启用存储和转发，将STR_FWD设为"1"。在这种情况下，MAC只有在发送FIFO中存储了一个完整的帧时才开始传输数据。

地址。0h基数+144h偏移量=144h




**ENET_TFWR字段描述**

| 场地 | 描述 |
|---|---|
| 31-9<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 8<br>STRFWD | 启用存储和转发功能<br><br>0 复位。传输开始阈值在TFWR[TFWR]中编程。<br>1 已启用。 |
| 7-6<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| TFWR | 发送FIFO写入<br><br>如果TFWR[STRFWD]被清除，这个字段表示在开始传输一帧之前，写入传输FIFO的字节数，以64字节为单位。<br><br>**注意：**如果一个小于阈值的帧被写入，它仍然被发送，与这个阈值设置无关。只有当帧大于给定的阈值时，该阈值才是相关的。<br><br>00000064字节写。<br>00000164字节写。<br>000010128字节写。<br>000011192字节写入。<br>... ...<br>0111111984字节写。 |

## 52.5.19 接收描述符环形启动寄存器（ENET_RDSR）。

RDSR指向外部存储器中循环接收缓冲区描述符队列的开始。这个指针必须是64位

对齐（第2-0位必须是0）；但是，建议是128位对齐，也就是均匀地除以16。

### 注意事项

该寄存器必须在操作前被初始化

地址。0h基数+180h偏移量=180h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R<br>W | R_DES_START | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 位 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | R_DES_START | | | | | | | | | | | | | | 0 | |
| W | | | | | | | | | | | | | | 0 | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**ENET_RDSR字段描述**

| 场地 | 描述 |
|---|---|
| 31-3<br>R_DES_START | 指向接收缓冲区描述符队列开始的指针。 |
| 2<br>保留 | 这个字段是保留的。<br>这个只写的字段是保留的。它必须始终以0的值写入。 |
| 保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |

## 52.5.20 发送缓冲区描述符环形启动寄存器（ENET_TDSR）。

TDSR提供了一个指向外部存储器中循环传输缓冲区描述符队列开始的指针。这个指针必须是64位对齐的（第2-
0位必须是0）；但是，建议是128位对齐的，也就是均匀地除以16。

**注意事项**

该寄存器必须在操作前被初始化。

地址。0h base + 184h offset = 184h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R<br>W | X_DES_START | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 位 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | X_DES_START | | | | | | | | | | | | | | 0 | |
| W | | | | | | | | | | | | | | 0 | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**ENET_TDSR字段描述**

| 场地 | 描述 |
|---|---|

| | |
|---|---|
| 31-3<br>X_DES_START | 指向发送缓冲区描述符队列开始的指针。 |

*表继续在下一页...*

**ENET_TDSR字段描述（续）**

| 场地 | 描述 |
|---|---|
| 2<br>保留 | 这个字段是保留的。<br>这个只写的字段是保留的。它必须始终以0的值写入。 |
| 保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |

## 52.5.21 最大接收缓冲区大小寄存器（ENET_MRBR）。

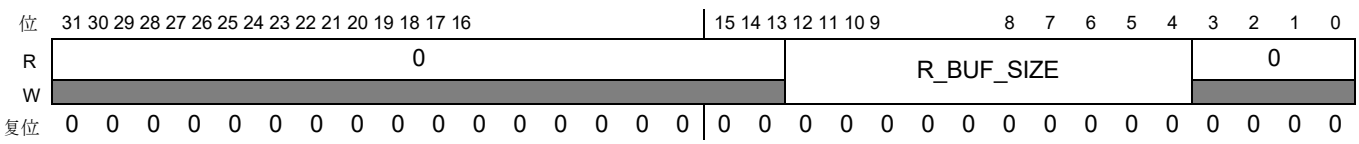
MRBR是一个用户可编程的寄存器，决定了所有接收缓冲区的最大尺寸。这个值应该考虑到接收CRC总是被写入最后一个接收缓冲区中。

- R_BUF_SIZE与该寄存器的四个最低位相连接，作为最大接收缓冲区大小。
- 为了允许每个缓冲区有一个最大尺寸的帧，MRBR必须被设置为RCR[MAX_FL]或者更大。
- 为了正确对齐缓冲区，MRBR必须均匀地被16除以。为了确保这一点，设备将低四位设置为零。
- 为了尽量减少总线的使用（描述符的获取），设置MRBR大于或等于256字节。

**注意事项**

该寄存器在操作前必须被初始化。

地址。0h base + 188h offset = 188h

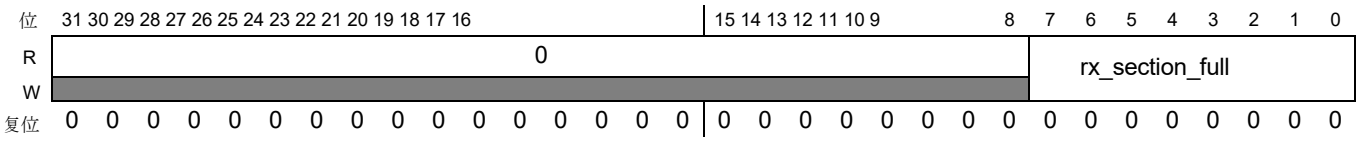



**ENET_MRBR字段描述**

| 场地 | 描述 |
|---|---|
| 31-14<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 13-4<br>R_BUF_SIZE | 接收缓冲区大小，以字节为单位。该值与该寄存器的四个最低位（始终为零）相连接，是有效的最大接收缓冲区大小。 |
| 保留 | 这个字段始终为零，是最大接收缓冲区大小的四个最小有效位。<br><br>这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |

## 52.5.22 接收FIFO部分满阈值（ENET_RSFL）。

地址。0h基数+190h偏移量=190h

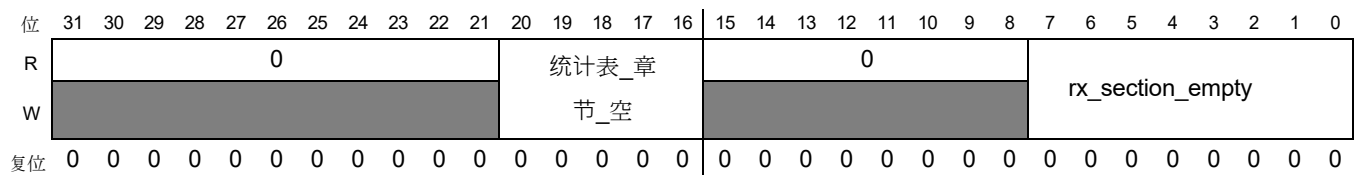



**ENET_RSFL字段描述**

| 场地 | 描述 |
|---|---|
| 31-8<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 充满了<br>rx_section_ | 接收FIFO部分满阈值的值<br>接收FIFO部分满的阈值，单位是64位字。清除这个字段以启用RX FIFO上的存储和转发。当编程一个大于0的值时（切入操作），它必须大于RAEM[RX_ALMOST_EMPTY]。<br>当FIFO水平达到该字段的值时，接收FIFO中的数据就可以使用了（切入操作）。 |

## 52.5.23 接收FIFO部分空阈值（ENET_RSEM）。

地址。0h base + 194h offset = 194h




**ENET_RSEM字段描述**

| 场地 | 描述 |
|---|---|
| 31-21<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 20-16<br>统计表_章节_空 | RX状态FIFO部分空阈值<br>定义接收FIFO中可以接受的帧数，与它的大小无关。如果达到极限，接收将继续正常进行，但会触发一个暂停帧，向远程设备提示可能的拥堵，以避免FIFO溢出。值为0时，禁止自动生成暂停帧。 |
| 15-8<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |

| | |
|---|---|
| Rx_section_empty | 接收FIFO部分空阈值的值<br><br>接收FIFO部分的空阈值，单位是64位字。当FIFO达到这个水平时，将发出一个暂停帧。 |

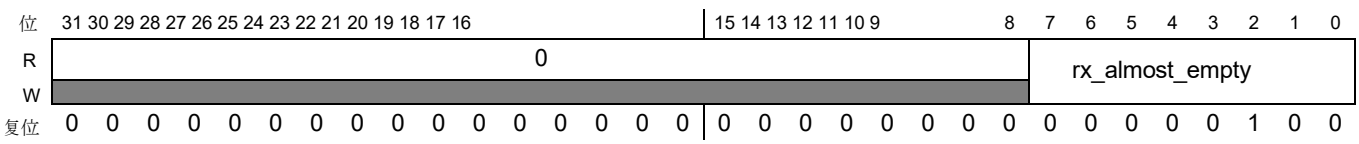
*表继续在下一页...*

**ENET_RSEM字段描述（续）**

| 场地 | 描述 |
|---|---|
| | 值为0时，禁止自动生成暂停帧。<br>当FIFO水平低于该字段编程的值时，会发出一个XON暂停帧，以表示FIFO拥塞被清除给远程以太网客户端。<br>**注意：**来自两个FIFO的部分空阈值指示被OR，以导致XOFF暂停帧的产生。 |

## 52.5.24 接收FIFO几乎清空的阈值（ENET_RAEM）。

地址。0h基数+198h偏移量=198h

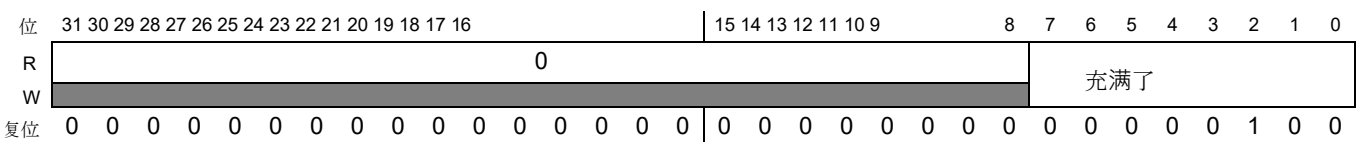



**ENET_RAEM字段描述**

| 场地 | 描述 |
|---|---|
| 31-8<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 几乎为空 | 接收FIFO几乎清空的阈值<br><br>接收FIFO几乎为空的阈值，单位为64位字。当FIFO水平达到在这个字段中编程的值，并且还没有收到该帧的帧末，核心接收读取控制停止FIFO读取（并随后停止向MAC客户端应用传输数据）。如果FIFO中又有比阈值或帧末更多的数据，它将继续传送该帧。应设置最小值为4。 |

## 52.5.25 接收FIFO几乎满的阈值（ENET_RAFL）。

地址。0h基数+19ch偏移量=19ch




**ENET_RAFL字段描述**

| 场地 | 描述 |
|---|---|
| 31-8<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |

| | |
|---|---|
| 几乎满载的Rx_almost_full | 接收FIFO几乎满的阈值 |

*表继续在下一页...*

**ENET_RAFL字段描述（续）**

| 场地 | 描述 |
|---|---|
| | 接收FIFO几乎满的阈值，单位是64位字。当FIFO水平接近最大值，以至于没有更多的空间容纳至少RX_ALMOST_FULL字数时，MAC停止在FIFO中写入数据，并截断接收的帧以避免FIFO溢出。当帧被传递给应用程序时，相应的错误状态将被设置。应设置最小值为4。 |

## 52.5.26 发送FIFO部分空阈值（ENET_TSEM）。

地址。0h基数+1A0h偏移量=1A0h

| 位 | 31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 | 15 14 13 12 11 10 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | tx_section_empty | |
| W | | | | | | | | | | | |
| 复位 | 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 | 0 0 0 0 0 0 0 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**ENET_TSEM字段描述**

| 场地 | 描述 |
|---|---|
| 31-8<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| tx_section_empty | 发送FIFO部分空阈值的值<br><br>发送FIFO部分的空阈值，单位是64位字。更多信息请参见发送FIFO。 |

## 52.5.27 发送FIFO几乎为空的阈值（ENET_TAEM）。

地址。0h基数+1A4h偏移量=1A4h

| 位 | 31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 | 15 14 13 12 11 10 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | tx_almost_empty | |
| W | | | | | | | | | | | |
| 复位 | 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 | 0 0 0 0 0 0 0 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |

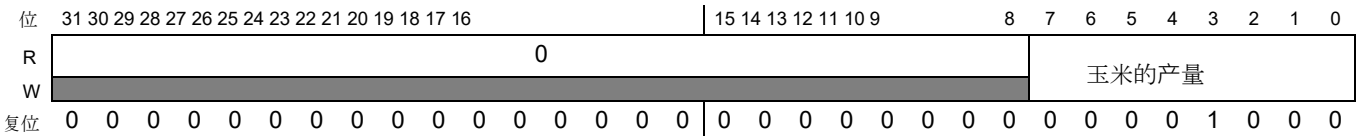
**ENET_TAEM字段描述**

| 场地 | 描述 |
|---|---|
| 31-8<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |

| | |
|---|---|
| 几乎为空 | 发送FIFO几乎为空阈值的值<br><br>发送FIFO几乎为空的阈值，单位为64位字。<br><br>当FIFO水平达到该字段中编程的值，并且没有帧末可用时，MAC发送逻辑为避免FIFO溢出，停止读取FIFO并发送一个带有MII错误指示的帧。更多信息请参见发送FIFO。<br><br>应设定最小值为4。 |

## 52.5.28 发送FIFO几乎满的阈值（ENET_TAFL）。

地址。0h基数+1A8h偏移量=1A8h

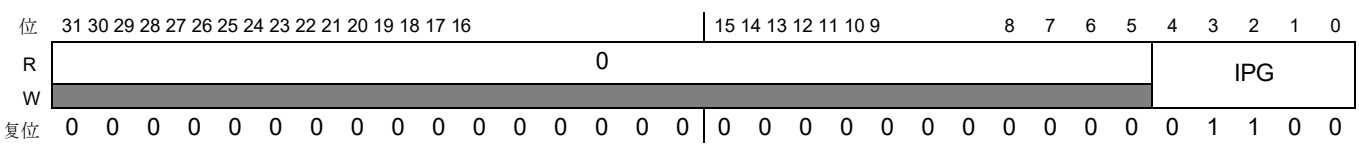



**ENET_TAFL字段描述**

| 场地 | 描述 |
|---|---|
| 31-8<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 几乎满员 | 发送FIFO几乎满的阈值<br><br>发送FIFO几乎满的阈值，单位是64位字。要求最小值为6。推荐值至少为8，允许应用有两个时钟周期的延迟。如果需要更多的延迟，可以根据需要增加该值（延迟=TAFL-5）。<br><br>当FIFO水平接近最大值时，以至于没有更多的空间容纳至少TX_ALMOST_FULL数量的字，引脚ff_tx_rdy被去势。如果应用没有对这个信号做出反应，FIFO的写控制逻辑，为了避免FIFO溢出，会截断当前的帧并设置错误状态。因此，该帧在传输时将出现GMII/MII错误指示。更多信息请参见传输FIFO。<br><br>**注意：**FIFO溢出是一个致命的错误，需要在传输数据通路上进行全局复位，或者至少是去掉ETHEREN的信号。 |

## 52.5.29 发送包间间隙 (ENET_TIPG)

地址。0h基数+1ACh偏移量=1ACh



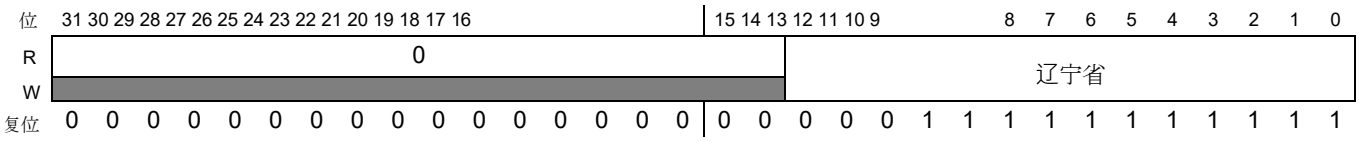

**ENET_TIPG字段描述**

| 场地 | 描述 |
|---|---|
| 31-5<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| IPG | 发送包间间隙<br><br>表示传输帧之间的IPG，以字节为单位。有效值范围为8至26。如果写入值小于8或大于26，内部（有效）IPG为12。 |

| | |
|---|---|
| | **注意：**读取的IPG值将是写入的值，即使它超出了范围。 |

## 52.5.30 帧截断长度 (ENET_FTRL)

地址。0h基数+1B0h偏移量=1B0h




**ENET_FTRL字段描述**

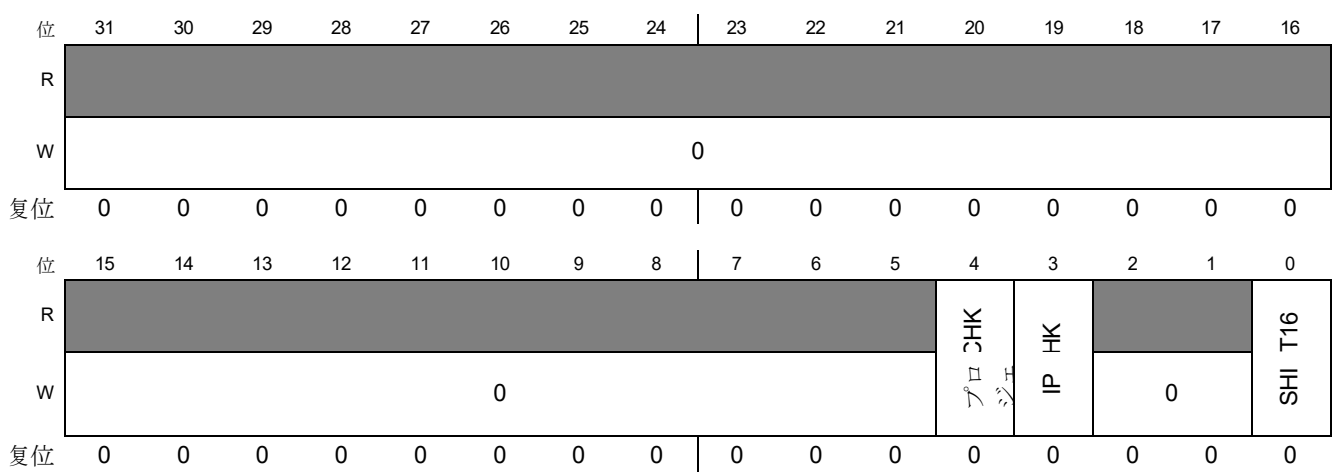
| 场地 | 描述 |
|---|---|
| 31-14<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 辽宁省 | 帧截断长度<br><br>表示一个接收帧被截断的值，如果它大于这个值。必须大于或等于 RCR[MAX_FL]。<br><br>**注意：**截断发生在TRUNC_FL。然而，当截断发生时，应用程序（FIFO）可能会收到更少的数据，保证它收到的数据永远不会超过设定的限制。 |

## 52.5.31 发送加速器功能配置（ENET_TACC）。

TACC控制发送帧时的加速器动作。该寄存器可以在每一帧之前或之后被改变，但在向发送FIFO写入帧时必须保持不被修改。

TFWR[STRFWD]字段必须被设置以使用校验功能。

地址。0h基数 + 1C0h偏移量 = 1C0h

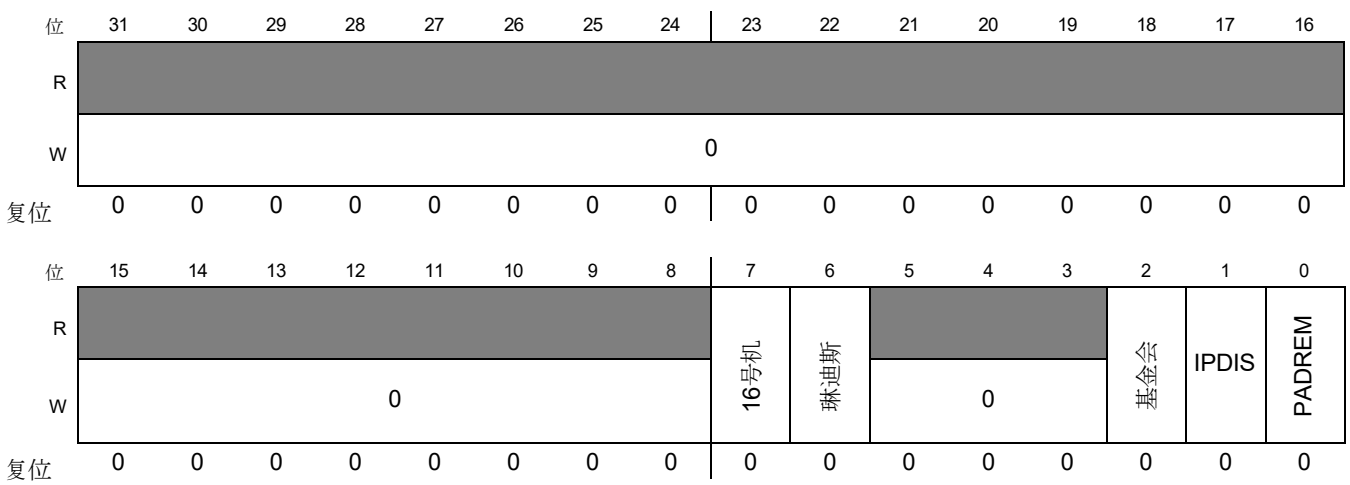


**ENET_TACC字段描述**

| 场地 | 描述 |
|---|---|
| 31-5<br>保留 | 这个字段是保留的。<br>这个只写的字段是保留的。它必须始终以0的值写入。 |
| 4<br>プロジェクト | 启用插入协议检查和。<br>0 未插入校验和。<br>1 如果传输的是已知协议的IP帧，校验和会自动插入帧中。校验和字段必须被清除。其他帧不被修改。 |
| 3<br>IPCHK | 启用插入IP头检查和。<br>0 没有插入校验和。<br>1 如果一个IP帧被传输，校验和会自动插入。IP头的校验和字段必须被清除。如果传输的是一个非IP帧，该帧不会被修改。 |
| 2-1<br>保留 | 这个字段是保留的。<br>这个只写的字段是保留的。它必须始终以0的值写入。 |
| 0<br>16号机 | TX FIFO Shift-16<br>0 残废。<br>1 向传输数据FIFO表明，写入的帧在帧数据之前包含两个额外的八位数。这意味着实际的帧从写进FIFO的第一个字的第16位开始。这个功能允许将帧有效载荷放在内存的32位边界上，因为14字节的以太网头被扩展为16字节的头。 |

## 52.5.32 接收加速器功能配置（ENET_RACC）。

地址。0h基数 + 1C4h偏移量 = 1C4h




**ENET_RACC字段描述**

| 场地 | 描述 |
|---|---|

| | |
|---|---|
| 31-8<br>保留 | 这个字段是保留的。<br>这个只写的字段是保留的。它必须始终以0的值写入。 |

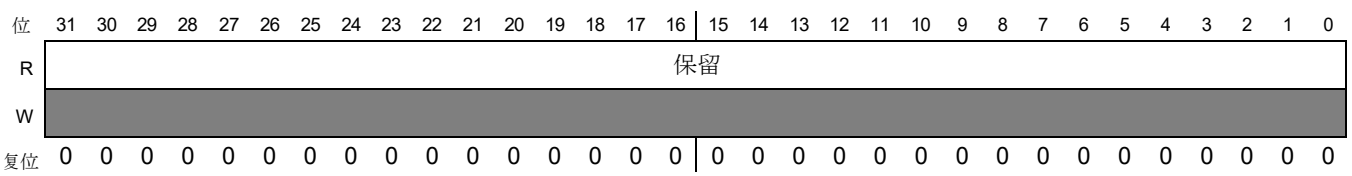
*表继续在下一页...*

**ENET_RACC字段描述（续）**

| 场地 | 描述 |
|---|---|
| 7<br>16号机 | RX FIFO Shift-16<br>当这个字段被设置时，实际的帧数据从从RX FIFO读取的第一个字的第16位开始，在32位边界上对齐以太网有效载荷。<br>**注意：**这个功能只影响FIFO的存储，对统计数据没有影响，因为统计数据使用的是接收到的帧的实际长度。<br>0 残废。<br>1 指示MAC在接收到的每个帧的前面再写两个字节到RX FIFO。 |
| 6<br>琳迪斯 | 启用丢弃有MAC层错误的帧的功能<br>0 有错误的帧不会被丢弃。<br>1 任何收到的带有CRC、长度或PHY错误的帧都会被自动丢弃，不会转发到用户应用接口。 |
| 5-3<br>保留 | 这个字段是保留的。<br>这个只写的字段是保留的。它必须始终以0的值写入。 |
| 2<br>基金会 | 启用丢弃有错误协议校验和的帧的功能<br>0 具有错误校验和的帧不会被丢弃。<br>1 如果收到的TCP/IP、UDP/IP或ICMP/IP帧有错误的TCP、UDP或ICMP校验和，该帧将被丢弃。只有当RX FIFO工作在存储和转发模式（RSFL被清除）时，丢弃才可用。 |
| 1<br>IPDIS | 启用丢弃有错误的IPv4头校验和的帧<br>0 具有错误的IPv4头校验和的帧不会被丢弃。<br>1 如果收到的IPv4帧有一个不匹配的帧头校验和，该帧将被丢弃。IPv6没有报头校验，不受此设置影响。只有当RX FIFO工作在存储和转发模式（RSFL被清除）时，丢弃才可用。 |
| 0<br>PADREM | 启用短IP帧的填充移除功能<br>0 衬垫未被移除。<br>1 帧的IP有效载荷部分之后的任何字节都从帧中删除。 |

## 52.5.33 保留统计寄存器（ENET_RMON_T_DROP）。

地址。0h基数+200h偏移量=200h



## ENET_RMON_T_DROP字段描述

| 场地 | 描述 |
| --- | --- |
| 保留 | 这个只读字段的值始终为0。<br><br>这个字段是保留的。 |

## 52.5.34 Tx数据包计数统计寄存器（ENET_RMON_T_PACKETS）。

地址。0h base + 204h offset = 204h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | | TXPKTS | | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**ENET_RMON_T_PACKETS字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| TXPKTS | 数据包数量<br><br>发送数据包数量 |

## 52.5.35 Tx广播包统计寄存器（ENET_RMON_T_BC_PKT）。

RMON Tx广播包

地址。0h base + 208h offset = 208h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | | TXPKTS | | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**ENET_RMON_T_BC_PKT字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| TXPKTS | 广播数据包 |

## 52.5.36 Tx多播包统计寄存器 (ENET_RMON_T_MC_PKT)

地址。0h基数+20Ch偏移量=20Ch

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | | TXPKTS | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**ENET_RMON_T_MC_PKT字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| TXPKTS | 多播数据包 |

## 52.5.37 CRC/Align错误统计寄存器（ ENET_RMON_T_CRC_ALIGN）的Tx数据包。

地址。0h base + 210h offset = 210h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | | TXPKTS | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**ENET_RMON_T_CRC_ALIGN字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| TXPKTS | 有CRC/对齐错误的数据包 |

## 52.5.38 Tx包小于字节和良好CRC统计寄存器（ ENET_RMON_T_UNDERSIZE）。

地址。0h基数+214h偏移量=214h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | | TXPKTS | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

R

W

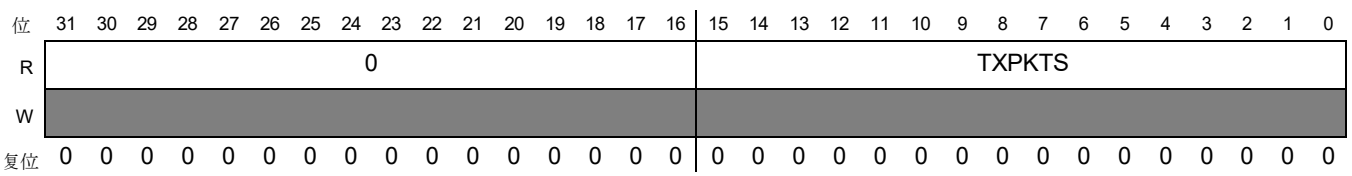
复位

**ENET_RMON_T_UNDERSIZE字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| TXPKTS | 具有良好CRC的小于64字节的传输数据包的数量 |

## 52.5.39 Tx 数据包 GT MAX_FL 字节和良好 CRC 统计寄存器 (ENET_RMON_T_OVERSIZE)

地址。0h基数+218h偏移量=218h



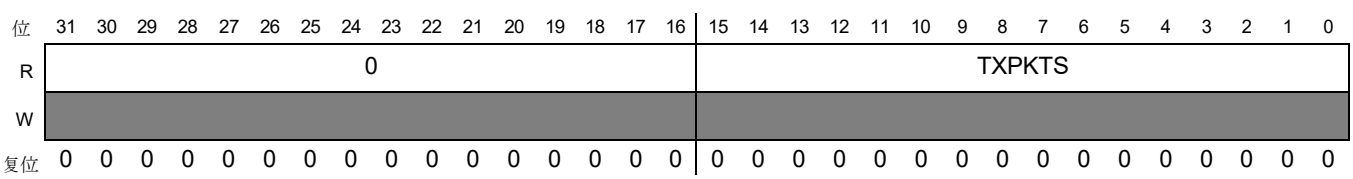

**ENET_RMON_T_OVERSIZE字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| TXPKTS | 大于MAX_FL字节的传输数据包的数量，具有良好的CRC |

## 52.5.40 小于64字节的Tx数据包和坏的CRC统计寄存器（ENET_RMON_T_FRAG）。

.

地址。0h基数+21Ch偏移量=21Ch




**ENET_RMON_T_FRAG字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |

| | |
|---|---|
| TXPKTS | CRC值小于64字节的数据包的数量。 |

## 52.5.41 Tx 数据包大于 MAX_FL 字节和坏 CRC 统计寄存器 (ENET_RMON_T_JAB)

地址。0h基数+220h偏移量=220h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | | TXPKTS | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**ENET_RMON_T_JAB字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| TXPKTS | 大于MAX_FL字节的传输数据包和不良CRC的数量 |

## 52.5.42 Tx碰撞计数统计寄存器（ENET_RMON_T_COL）。

地址。0h基数+224h偏移量=224h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | | TXPKTS | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**ENET_RMON_T_COL字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| TXPKTS | 发送碰撞的数量 |

## 52.5.43 Tx 64字节数据包统计寄存器（ENET_RMON_T_P64）。

.

地址。0h基数+228h偏移量=228h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | | TXPKTS | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

R

W

复位

**ENET_RMON_T_P64字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| TXPKTS | 64字节的传输数据包的数量 |

## 52.5.44 Tx 65-127字节数据包统计寄存器 (ENET_RMON_T_P65TO127)

地址。0h基数+22Ch偏移量=22Ch

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | | TXPKTS | | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**ENET_RMON_T_P65TO127字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| TXPKTS | 65至127字节的传输数据包的数量 |

## 52.5.45 Tx 128-255字节数据包统计寄存器（ ENET_RMON_T_P128TO255）。

地址。0h base + 230h offset = 230h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | | TXPKTS | | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**ENET_RMON_T_P128TO255字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| TXPKTS | 128至255字节的传输数据包的数量 |

## 52.5.46 Tx 256-至511字节数据包统计寄存器 (ENET_RMON_T_P256TO511)

地址。0h base + 234h offset = 234h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | | TXPKTS | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**ENET_RMON_T_P256TO511字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| TXPKTS | 256至511字节的传输数据包的数量 |

## 52.5.47 Tx 512-至1023字节数据包统计寄存器 (ENET_RMON_T_P512TO1023)

.

地址。0h base + 238h offset = 238h

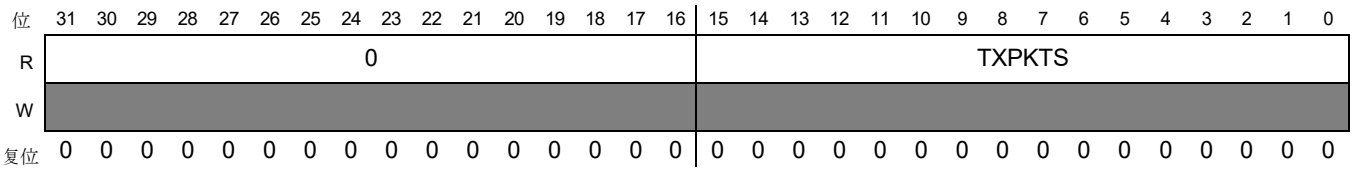
| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | | TXPKTS | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**ENET_RMON_T_P512TO1023字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| TXPKTS | 512至1023字节的传输数据包的数量 |

## 52.5.48 Tx 1024-至2047字节数据包统计寄存器 (ENET_RMON_T_P1024TO2047)

地址。0h基数+23Ch偏移量=23Ch

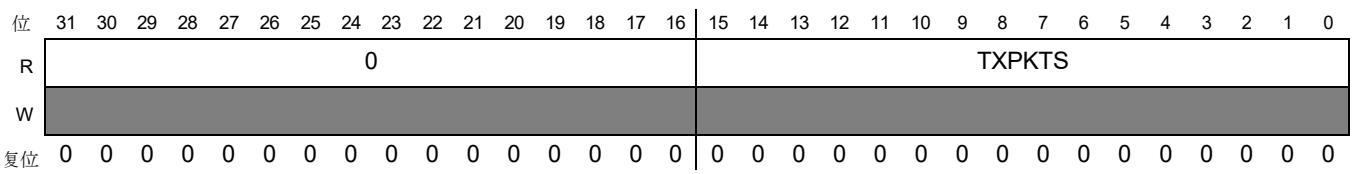



**ENET_RMON_T_P1024TO2047字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| TXPKTS | 1024至2047字节的传输数据包的数量 |

## 52.5.49 大于2048字节的Tx数据包统计寄存器 (ENET_RMON_T_P_GTE2048)

地址。0h base + 240h offset = 240h




**ENET_RMON_T_P_GTE2048字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| TXPKTS | 大于2048字节的传输数据包的数量 |

## 52.5.50 Tx Octets统计寄存器（ENET_RMON_T_OCTETS）。

地址。0h基数+244h偏移量=244h



W

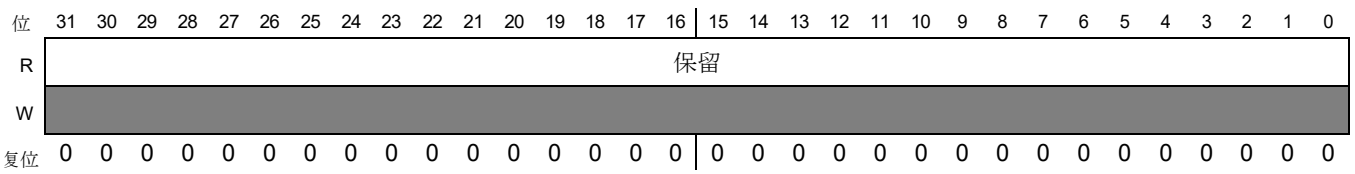
复位

**ENET_RMON_T_OCTETS字段描述**

| 场地 | 描述 |
|---|---|
| 芵迮郇郎郅郎泄邽迮郴邽郭 | 发送八位数的数量 |

## 52.5.51 保留统计寄存器（ENET_IEEE_T_DROP）。

地址。0h base + 248h offset = 248h

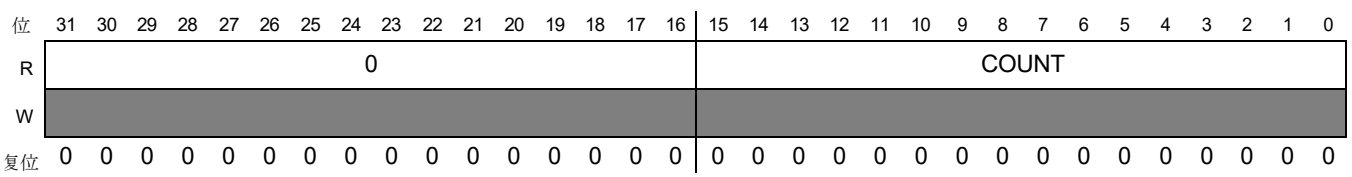



**ENET_IEEE_T_DROP字段描述**

| 场地 | 描述 |
|---|---|
| 保留 | 这个只读字段的值始终为0。<br><br>这个字段是保留的。 |

## 52.5.52 帧传输OK统计寄存器（ ENET_IEEE_T_FRAME_OK）。

地址。0h基数+24Ch偏移量=24Ch




**ENET_IEEE_T_FRAME_OK字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| COUNT | 传输的帧数 OK<br><br>**注意：在**接收控制寄存器（RCR）中，当广播拒绝被启用且混杂模式被禁用时，不会为广播帧增量。 |

## 52.5.53 用单次碰撞统计寄存器传输的帧（ENET_IEEE_T_1COL）。

地址。0h基数+250h偏移量=250h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | | COUNT | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**ENET_IEEE_T_1COL字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| COUNT | 传送一个碰撞的帧数 |

## 52.5.54 多次碰撞传输的帧统计寄存器（ENET_IEEE_T_MCOL）。

地址。0h base + 254h offset = 254h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | | COUNT | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**ENET_IEEE_T_MCOL字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| COUNT | 传输的帧数有多个碰撞 |

## 52.5.55 延迟延迟后传输的帧统计寄存器（ENET_IEEE_T_DEF）。

地址。0h基数 + 258h偏移量 = 258h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | | COUNT | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**ENET_IEEE_T_DEF字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| COUNT | 延迟传送的帧数 |

## 52.5.56 用晚期碰撞统计寄存器传输的帧（ENET_IEEE_T_LCOL）。

地址。0h基数+25Ch偏移量=25Ch

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | | COUNT | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**ENET_IEEE_T_LCOL字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| COUNT | 迟来的碰撞所传送的帧的数量 |

## 52.5.57 碰撞过多时传输的帧数统计寄存器（ENET_IEEE_T_EXCOL）。

地址。0h基数+260h偏移量=260h

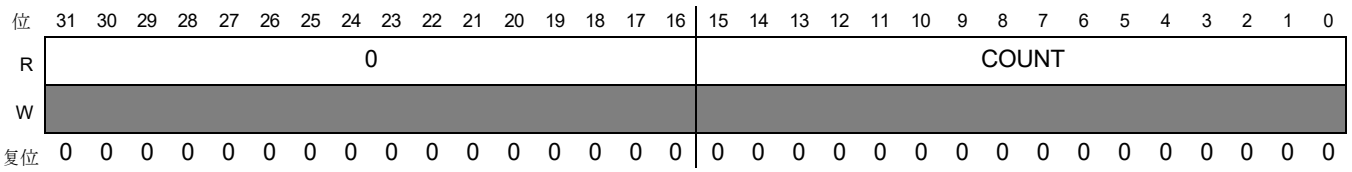
| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | | COUNT | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**ENET_IEEE_T_EXCOL字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| COUNT | 传输的帧数有过度碰撞 |

## 52.5.58 用Tx FIFO Underrun统计寄存器传输的帧（ENET_IEEE_T_MACERR）。

地址。0h基数+264h偏移量=264h

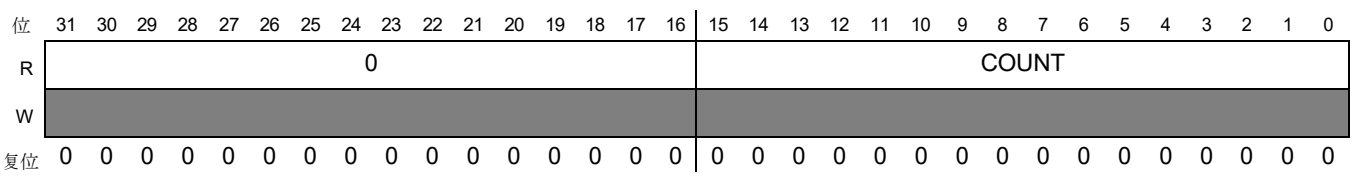



**ENET_IEEE_T_MACERR字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| COUNT | 传输FIFO不足时传输的帧数 |

## 52.5.59 用载波感应错误统计寄存器传输的帧（ENET_IEEE_T_CSERR）。

地址。0h基数 + 268h偏移量 = 268h

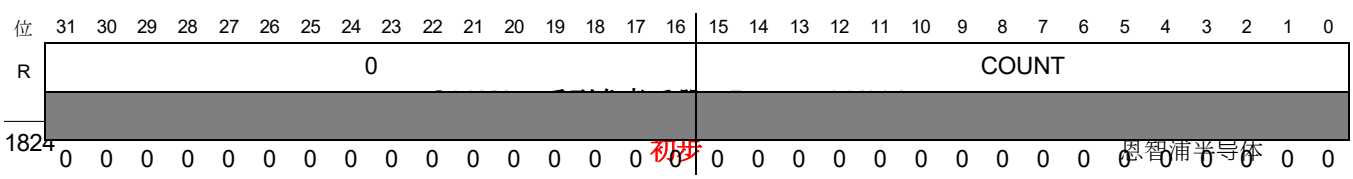



**ENET_IEEE_T_CSERR字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| COUNT | 带载波感应错误的传输帧数 |

## 52.5.60 保留统计寄存器（ENET_IEEE_T_SQE）。

地址。0h基数+26Ch偏移量=26Ch





初步 恩智浦半导体

W

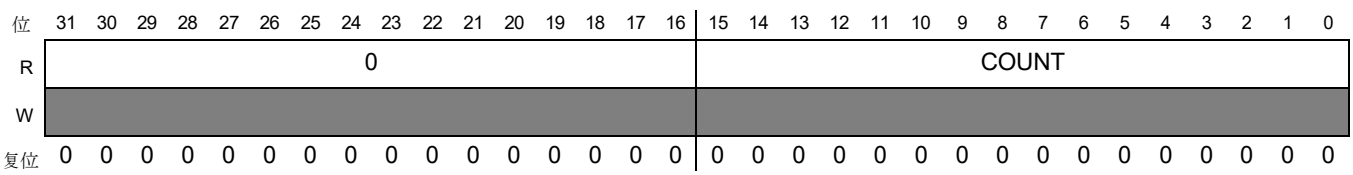
复位

**ENET_IEEE_T_SQE字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| COUNT | 这个只读字段是保留的，其值始终为0。<br><br>**注：**由于没有SQE信息，所以没有实施计数器。 |

## 52.5.61 流量控制暂停帧传输统计寄存器（ENET_IEEE_T_FDXFC）。

地址。0h基数+270h偏移量=270h

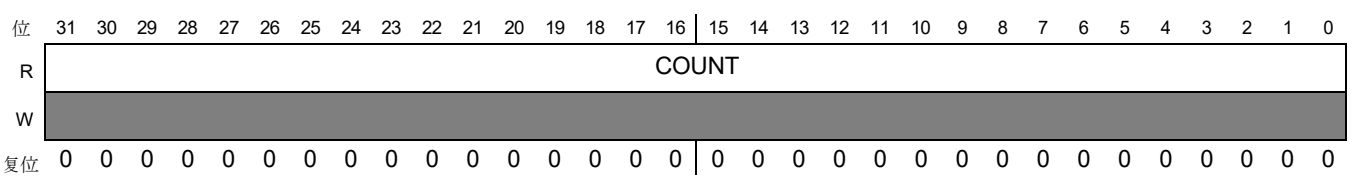



**ENET_IEEE_T_FDXFC字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| COUNT | 传输的流控暂停帧的数量 |

## 52.5.62 误差统计寄存器（ENET_IEEE_T_OCTETS_OK）中传输的帧的八位数。

地址。0h基数+274h偏移量=274h




**ENET_IEEE_T_OCTETS_OK字段描述**

| 场地 | 描述 |
|---|---|
| COUNT | 无错误传输的帧的八位数<br><br>**注意事项** |

| | |
|---|---|
| | 计算总的字节数（包括头和FCS字段）。 |

## 52.5.63 Rx数据包计数统计寄存器（ENET_RMON_R_PACKETS）。

地址。0h基数 + 284h偏移量 = 284h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | | COUNT | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**ENET_RMON_R_PACKETS字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| COUNT | 收到的数据包数量 |

## 52.5.64 Rx广播包统计寄存器（ENET_RMON_R_BC_PKT）。

地址。0h基数+288h偏移量=288h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | | COUNT | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**ENET_RMON_R_BC_PKT字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| COUNT | 接收广播数据包的数量 |

| 位 | |
|---|---|
| R | |
| W | |
| 复位 | |

## 52.5.65 Rx多播包统计寄存器（ENET_RMON_R_MC_PKT）。

地址。0h基数+28Ch偏移量=28Ch

**ENET_RMON_R_MC_PKT字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| COUNT | 接收多播数据包的数量 |

## 52.5.66 CRC/Align错误统计寄存器（ENET_RMON_R_CRC_ALIGN）的Rx数据包。

地址。0h base + 290h offset = 290h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | | COUNT | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**ENET_RMON_R_CRC_ALIGN字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| COUNT | 有CRC或对齐错误的接收数据包的数量 |

## 52.5.67 小于64字节的Rx数据包和良好的CRC统计寄存器（ENET_RMON_R_UNDERSIZE）。

地址。0h base + 294h offset = 294h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | | COUNT | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**ENET_RMON_R_UNDERSIZE字段描述**

| 场地 | | 描述 |
|---|---|---|
| 位<br>R | 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| W | COUNT | 小于64字节且CRC良好的接收数据包的数量 |
| 复位 | | |

## 52.5.68 Rx数据包大于MAX_FL和良好CRC统计寄存器（ENET_RMON_R_OVERSIZE）。

地址。0h base + 298h offset = 298h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | | COUNT | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

### ENET_RMON_R_OVERSIZE字段描述

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| COUNT | 大于MAX_FL且CRC良好的接收数据包的数量 |

## 52.5.69 小于64字节的Rx数据包和坏CRC统计寄存器（ENET_RMON_R_FRAG）。

地址。0h基数+29Ch偏移量=29Ch

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | | COUNT | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

### ENET_RMON_R_FRAG字段描述

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| COUNT | 小于64字节和CRC不良的接收数据包的数量 |

## 52.5.70 大于 MAX_FL 字节的 Rx 包和坏 CRC 统计寄存器 (ENET_RMON_R_JAB)

地址。0h基数+2A0h偏移量=2A0h

位

R

W

复位

**ENET_RMON_R_JAB字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| COUNT | 大于MAX_FL和坏CRC的接收数据包数量 |

## 52.5.71 保留统计寄存器（ENET_RMON_R_RESVD_0）。

地址。0h基数+2A4h偏移量=2A4h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | | 0 | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**ENET_RMON_R_RESVD_0字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |

## 52.5.72 Rx 64字节数据包统计寄存器（ENET_RMON_R_P64）。

地址。0h基数+2A8h偏移量=2A8h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | | COUNT | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**ENET_RMON_R_P64字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| COUNT | 64字节的接收数据包的数量 |

## 52.5.73 Rx 65-127-字节数据包统计寄存器 (ENET_RMON_R_P65TO127)

地址。0h基数+2ACh偏移量=2ACh

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | | COUNT | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**ENET_RMON_R_P65TO127字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| COUNT | 65至127字节的接收数据包的数量 |

## 52.5.74 Rx 128-255字节数据包统计寄存器 (ENET_RMON_R_P128TO255)

地址。0h基数 + 2B0h偏移量 = 2B0h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | | COUNT | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**ENET_RMON_R_P128TO255字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| COUNT | 128至255字节的接收数据包的数量 |

| 位 | |
|---|---|
| R | |
| W | |
| 复位 | |

## 52.5.75 Rx 256-至511字节数据包统计寄存器 (ENET_RMON_R_P256TO511)

地址。0h基数 + 2B4h偏移量 = 2B4h

**ENET_RMON_R_P256TO511字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| COUNT | 256至511字节的接收数据包的数量 |

## 52.5.76 Rx 512- to 1023-Byte 数据包统计寄存器 (ENET_RMON_R_P512TO1023)

地址。0h基数 + 2B8h偏移量 = 2B8h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | | COUNT | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**ENET_RMON_R_P512TO1023字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| COUNT | 512-1023字节的接收数据包的数量 |

## 52.5.77 Rx 1024-至2047-字节数据包统计寄存器 (ENET_RMON_R_P1024TO2047)

地址。0h基数 + 2BCh偏移量 = 2BCh

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | | COUNT | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**ENET_RMON_R_P1024TO2047字段描述**

| 场地 | | 描述 |
|---|---|---|
| 位<br>R | 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| W | COUNT | 1024至2047字节的接收数据包的数量 |

复位

## 52.5.78 大于2048字节的Rx数据包统计寄存器 (ENET_RMON_R_P_GTE2048)

地址。0h基数 + 2C0h偏移量 = 2C0h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | | COUNT | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**ENET_RMON_R_P_GTE2048字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| COUNT | 大于2048字节的接收数据包的数量 |

## 52.5.79 Rx Octets统计寄存器（ENET_RMON_R_OCTETS）。

地址。0h基数 + 2C4h偏移量 = 2C4h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | COUNT | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**ENET_RMON_R_OCTETS字段描述**

| 场地 | 描述 |
|---|---|
| COUNT | 接收八位数的数量 |

## 52.5.80 未正确计数的帧统计寄存器（ ENET_IEEE_R_DROP）。

如果检测到带有无效或缺失SFD字符的帧并被丢弃，则计数器会增加。如果该计数器递增，其他计数器都不会递增。

地址。0h基数 + 2C8h偏移量 = 2C8h

位

R

W

复位

**ENET_IEEE_R_DROP字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| COUNT | 帧数 |

## 52.5.81 帧接收OK统计寄存器（ENET_IEEE_R_FRAME_OK）。

地址。0h基数+2CCh偏移量=2CCh

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | | COUNT | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**ENET_IEEE_R_FRAME_OK字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| COUNT | 收到的帧的数量 OK |

## 52.5.82 CRC错误统计寄存器（ENET_IEEE_R_CRC）收到的帧。

地址。0h base + 2D0h offset = 2D0h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | | COUNT | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**ENET_IEEE_R_CRC字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| COUNT | 收到的有CRC错误的帧数 |

## 52.5.83 收到的帧中有对齐错误统计寄存器（ENET_IEEE_R_ALIGN）。

地址。0h基数+2D4h偏移量=2D4h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | | COUNT | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**ENET_IEEE_R_ALIGN字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| COUNT | 收到的有对齐错误的帧数 |

## 52.5.84 接收FIFO溢出计数统计寄存器（ENET_IEEE_R_MACERR）。

地址。0h base + 2D8h offset = 2D8h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | | COUNT | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**ENET_IEEE_R_MACERR字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| COUNT | 接收FIFO溢出计数 |

| 位 | |
|---|---|
| R | |
| W | |
| 复位 | |

## 52.5.85 流量控制暂停帧接收统计寄存器（ENET_IEEE_R_FDXFC）。

地址。0h base + 2DCh offset = 2DCh

**ENET_IEEE_R_FDXFC字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| COUNT | 收到的流控暂停帧的数量 |

## 52.5.86 无错误接收的帧的八位数统计寄存器（ENET_IEEE_R_OCTETS_OK）。

地址。0h基数 + 2E0h偏移量 = 2E0h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | COUNT | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**ENET_IEEE_R_OCTETS_OK字段描述**

| 场地 | 描述 |
|---|---|
| COUNT | 无错误接收的帧的八位数<br><br>**注意：**计算总八位数（包括头和FCS字段）。在接收控制寄存器（RCR）中，当广播拒绝被启用和混杂模式被禁用时，不会为广播帧增量。 |

## 52.5.87 可调定时器控制寄存器（ENET_ATCR）。

ATCR命令字段可以直接触发相应的事件。当命令字段在寄存器中被设置时，不需要保留任何配置字段，也就是说，不需要读-改-写。

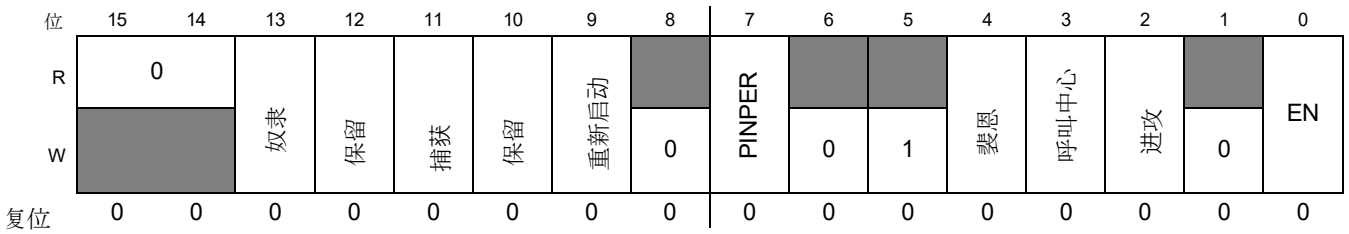
**注意事项**

CAPTURE和RESTART字段和第12位和第10位必须为0，才能写到这个寄存器的其他字段。

地址。0h base + 400h offset = 400h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

复位




## ENET_ATCR字段描述

| 场地 | 描述 |
|---|---|
| 31-14<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 13<br>奴隶 | 启用定时器从属模式<br><br>0 定时器处于激活状态，该寄存器中的所有配置字段都是相关的。<br>1 内部定时器被禁用，使用外部提供的定时器值。除CAPTURE外，该寄存器中的所有其他字段都没有作用。CAPTURE仍然可以用来捕获当前的定时器值。 |
| 12<br>保留 | 这个字段是保留的。<br>始终将0写到这个字段。 |
| 11<br>捕获 | 捕获定时器值<br><br>当这个字段被设置时，所有其他字段在写入过程中被忽略。命令完成后，该字段自动清除为0。<br><br>**注意：**为确保从 ATVR 寄存器中读取正确的时间值，从发出该命令到读取 ATVR 寄存器必须经过一个最小的时间。这个最短的时间是由6个寄存器时钟周期或6个1588/时间戳时钟周期中的较大者来定义。<br><br>0 没有影响。<br>1 当前的时间被捕获，可以从ATVR寄存器中读取。 |
| 10<br>保留 | 这个字段是保留的。<br>始终将0写到这个字段。 |
| 9<br>重新启动 | 重置定时器<br><br>将定时器重置为零。这对计数器的启用没有影响。如果这个字段被设置时，计数器被启用，则定时器被重置为零并从那里开始计数。当设置时，所有其他字段在写入时被忽略。命令完成后，该字段自动清零。<br><br>**注意：**复位定时器命令需要至少6个时钟周期的寄存器时钟或1588/时间戳时钟（以较大者为准）才能完成。 |
| 8<br>保留 | 这个字段是保留的。 |
| 7<br>PINPER | 在周期性事件中启用事件信号输出断言。<br><br>**注意：**不是所有的器件都包含事件信号输出。参见芯片配置细节。<br><br>0 禁用。<br>1 启用。 |
| 6<br>保留 | 这个字段是保留的。 |

| | |
|---|---|
| 5<br>保留 | 这个字段是保留的。 |

*表继续在下一页...*

**ENET_ATCR字段描述（续）**

| 场地 | 描述 |
|---|---|
| | **注意：**该字段必须始终写为1。 |
| 4<br>裴恩 | 启用周期性事件<br>0 禁用。<br>1 可以产生一个周期性的事件中断（EIR[TS_TIMER]），当定时器按照周期设置ATPER缠绕时，事件信号输出被断言。在设置这个位之前，必须先设置定时器的周期值。<br>**注意：**不是所有的器件都包含事件信号输出。参见芯片配置细节。 |
| 3<br>呼叫中心 | 在偏移事件中重置定时器<br>0 计时器不受影响，当达到偏移量时，除了清除OFFEN外，不发生任何动作。<br>1 如果OFFEN被设置，当达到偏移量设置时，定时器复位为零。偏移事件不会引起定时器中断。 |
| 2<br>进攻 | 启用单次偏移事件<br>0 禁用。<br>1 当达到给定的偏移时间时（偏移事件），定时器可以被重置为零。当达到偏移事件时，该字段被清除，因此在再次设置该字段之前，不会有进一步的事件发生。在设置这个字段之前，必须先设置定时器的偏移值。 |
| 1<br>保留 | 这个字段是保留的。 |
| 0<br>EN | 启用定时器<br>0 计时器停止在当前值。<br>1 计时器开始递增。 |

## 52.5.88 计时器值寄存器（ENET_ATVR）。

地址。0h基数+404h偏移量=404h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R<br>W | ATIME | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**ENET_ATVR字段描述**

| 场地 | 描述 |
|---|---|
| ATIME | 写入设置定时器。读取则返回最后的捕获值。要读取当前值，请在读取此寄存器之前发出一个捕获命令（即设置ATCR[CAPTURE]）。 |

## 52.5.89 定时器偏移寄存器（ENET_ATOFF）。

地址。0h base + 408h offset = 408h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R<br>W | 关节 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**ENET_ATOFF字段描述**

| 场地 | 描述 |
|---|---|
| 关节 | 用于生成一次性事件的偏移值。当定时器达到该值时，可以生成一个事件来重置计数器。如果ATINC中的增量值是以真纳秒为单位给出的，那么这个值也是以真纳秒为单位。 |

## 52.5.90 定时器周期寄存器（ENET_ATPER）。

地址。0h基数+40Ch偏移量=40Ch

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R<br>W | 期限 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**ENET_ATPER字段描述**

| 场地 | 描述 |
|---|---|
| 期限 | 生成周期性事件的值。每当定时器达到这个值时，周期事件就会发生，定时器就会重新启动。如果ATINC中的增量值是以真纳秒为单位，那么这个值也是以真纳秒为单位。该值应初始化为1,000,000,000(1×109)，以代表一秒钟的定时器绕行。ATINC中设置的增量值应该被设置为时钟ts_clk周期的真纳秒，因此实现了一个真正的1秒计数器。<br><br>**注意：**PERIOD的值有以下约束。<br><br>$2^{32}$ - enet_atinc[inc_cor] - 3×enet_atinc[inc] ≥ period > 0。 |

## 52.5.91 计时器校正寄存器（ENET_ATCOR）。

地址。0h base + 410h offset = 410h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | COR | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 位 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | COR | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

### ENET_ATCOR字段描述

| 场地 | 描述 |
|---|---|
| 31<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| COR | 纠正计数器的环绕值<br><br>定义在多少个定时器时钟周期(ts_clk)后，校正计数器应被重置，并触发定时器的校正增量。修正量在ATINC[INC_CORR]中定义。值为0时，校正计数器失效，不发生校正。<br><br>**注意：**这个值是以时钟周期为单位的，而不是像其他所有值那样以纳秒为单位。 |

## 52.5.92 时间戳时钟周期寄存器（ENET_ATINC）。

地址。0h基数+414h偏移量=414h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 位 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | INC_CORR | | | | | | | 0 | 基金会 | | | | | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

### ENET_ATINC字段描述

| 场地 | 描述 |
|---|---|
| 31-15<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |

| | |
|---|---|
| 14-8<br>INC_CORR | 矫正增量值<br><br>这个值在每次校正定时器过期时（ATCOR中给出的每一个时钟周期）都会加入。小于INC的值会减慢定时器的速度。大于INC的值会加快定时器的速度。 |

*表继续在下一页...*

**ENET_ATINC字段描述（续）**

| 场地 | 描述 |
|---|---|
| 7<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 基金会 | 时间戳时钟（ts_clk）的时钟周期（纳秒）。<br><br>定时器在每个时钟周期都以这个数量递增。例如，设置为10代表100MHz，8代表125MHz，5代表200MHz。<br><br>**注意：**为了获得最高精度，请使用ATPER中设定的周期的整数部分的值。 |

## 52.5.93 最后传输的帧的时间戳 (ENET_ATSTMP)

地址。0h基数+418h偏移量=418h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | TIMESTAMP | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**ENET_ATSTMP字段描述**

| 场地 | 描述 |
|---|---|
| TIMESTAMP | 由核心传输的最后一帧的时间戳，该帧设置了TxBD[TS]。这个寄存器只有在EIR[TS_AVAIL]被设置时才有效。 |

## 52.5.94 计时器全局状态寄存器（ENET_TGSR）。

地址。0h base + 604h offset = 604h

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 位 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | 0 | | | | | | | | | | | | TF3 | TF2 | TF1 | TF0 |
| W | | | | | | | | | | | | | w1c | w1c | w1c | w1c |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**ENET_TGSR字段描述**

| 场地 | 描述 |
| --- | --- |
| 31-4<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |

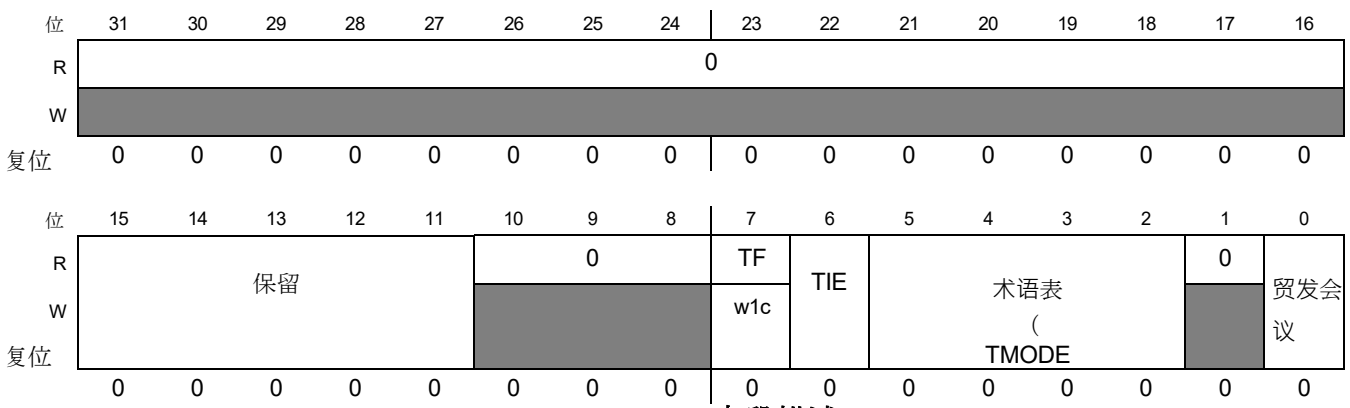
*表继续在下一页...*

**ENET_TGSR字段描述（续）**

| 场地 | 描述 |
|---|---|
| 3<br>TF3 | 频道3的定时器标志的复制<br><br>0 通道3的定时器标志已清除<br>1 通道3的定时器标志被设置 |
| 2<br>TF2 | 通道2的定时器标志的拷贝<br><br>0 通道2的定时器标志已清除<br>1 通道2的定时器标志被设置 |
| 1<br>TF1 | 通道1的定时器标志的拷贝<br><br>0 通道1的定时器标志已清除<br>1 通道1的定时器标志被设置 |
| 0<br>TF0 | 通道0的定时器标志的拷贝<br><br>0 通道0的定时器标志已清除<br>1 通道0的定时器标志被设置 |

## 52.5.95 计时器控制状态寄存器(ENET_TCSRn)

地址。0h基数+608h偏移量+（8d×i），其中i=0d至3d




**ENET_TCSRn字段描述**

| 场地 | 描述 |
|---|---|
| 31-16<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 15-11<br>保留 | 这个字段是保留的。<br>这个字段必须写成0。 |
| 10-8<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |

| | |
|---|---|
| 7<br>TF | 计时器标志<br><br>设定输入捕获或输出比较发生的时间。这个标志在模块时钟和1588时钟域之间是双缓冲的。当这个字段为1时，可以通过写1将其清除为0。 |

*表继续在下一页...*

**ENET_TCSRn字段描述（续）**

| 场地 | 描述 |
|---|---|
| | 0 输入捕捉或输出比较没有发生。<br>1 发生了输入捕捉或输出比较。 |
| 6<br>TIE | 定时器中断启用<br>0 中断被禁用<br>1 中断被启用 |
| 5-2<br>术语表（TMODE | 计时器模式<br>更新定时器模式字段需要几个周期来注册，因为它是与1588时钟同步的。读取时返回的定时器模式的版本是来自 1588 时钟域。当改变定时器模式时，一定要禁用该通道，并读取该寄存器以验证该通道是否被禁用。<br>0000Timer Channel被禁用。<br>0001Timer 通道被配置为上升沿的输入捕捉。0010Timer 通道被配置为下降沿的输入捕捉。0011定时器通道被配置为 在两个边沿上进行输入捕捉。0100Timer通道被配置为输出比较 - 仅限软件。<br>0101Timer Channel被配置为输出比较--在比较时切换输出。0110Timer Channel被配置为输出比较--清除比较时的输出。0111Timer Channel被配置为Output Compare - 设定比较时的输出。<br>1000 保留<br>1010Timer通道被配置为输出比较 - 比较时清除输出，溢出时设置输出。<br>10X1定时器通道被配置为输出比较--比较时设置输出，溢出时清除输出。<br>110X 保留<br>1110Timer通道被配置为输出比较--在一个1588-时钟周期内，比较时输出低电平脉冲。<br>1111Timer通道被配置为输出比较--在一个1588-时钟周期内，比较时输出高电平脉冲。 |
| 1<br>保留 | 这个字段是保留的。<br>这个只读字段是保留的，其值始终为0。 |
| 0<br>贸发会议 | 定时器DMA请求启用<br>0 DMA请求被禁用<br>1 DMA请求被启用 |

## 52.5.96 定时器比较捕获寄存器（ENET_TCCRn）。

地址。0h基数+60Ch偏移量+（8d×i），其中i=0d至3d

| 位 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R<br>W | TCC | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 复位 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**ENET_TCCRn字段描述**

| 场地 | 描述 |
|---|---|
| TCC | 定时器捕获比较<br><br>该寄存器在模块时钟和1588时钟域之间是双缓冲的。<br><br>当配置为比较时，每当定时器通道首次启用时，1588时钟域就会与模块时钟域的值一起更新，并在随后的每次比较中更新。在启用定时器通道之前，将第一个比较值写入该寄存器。当定时器通道被启用时，立即写入第二个比较值，或者至少在第一个比较发生之前。在每次比较之后，在前一次比较发生之前，在清除定时器标志之前，写入下一个比较值。<br><br>比较发生在IEEE 1588计数器在1588时钟域中增量超过比较值后的一个1588时钟周期。如果比较值小于定时器通道首次启用时的1588计数器的值，那么比较就不会发生，直到1588计数器的下一次溢出之后。如果在1588计数器溢出时，比较值大于IEEE 1588计数器的值，或者溢出后比较值小于IEEE 1588计数器的值，那么在溢出后的一个1588时钟周期内发生比较。<br><br>当配置为捕获时，IEEE 1588计数器的值被捕获到1588时钟域，然后更新到模块时钟域，只要定时器标志被清除。在清除定时器标志之前，一定要读取捕获值。 |

## 52.6 职能描述

本节提供了MAC-NET内核的完整功能描述。

### 52.6.1 以太网MAC帧格式

- 最小长度为64字节
- 最大长度为1518字节，不包括序言和起始帧定界符（SFD）字节

一个以太网帧由以下字段组成。

- 七个字节的序言
- 起始帧定界符（SFD）
- 两个地址栏
- 长度或类型字段
- 数据领域

- 帧校验序列（CRC值）。




**图52-2.MAC帧格式概述**

另外，MAC帧可以通过在MAC源地址和类型/长度字段之间插入一个额外的四字节字段来进行VLAN标记。VLAN标记是由IEEE P802.1q规范定义的。带VLAN标签的帧最大长度为1522字节，不包括前言和SFD字节。




**图52-3.VLAN标记的MAC帧格式概述 表52-3.MAC帧定义**

| 术语 | 描述 |
|---|---|
| 框架长度 | 定义不含序言和SFD的完整帧的长度，以八位字节为单位。如果一个帧至少包含64个八位字节，并且不超过编程的最大长度，则该帧具有有效长度。 |
| 有效载荷长度 | 长度/类型字段表示帧的有效载荷部分的长度。最重要的字节被首先发送/接收。<br>• 如果长度/类型字段被设置为小于46的值，则有效载荷被填充，以便满足最小帧长度要求（64字节）。对于VLAN标记的帧，小于42的值表示有填充的帧。<br>• 如果长度/类型字段被设置为一个大于编程的帧最大长度（例如1518）的值，它将被解释为一个类型字段。 |
| 目的地和源地址 | 48位MAC地址。最不重要的字节首先被发送/接收，MAC地址的前两个最不重要的比特区分MAC帧，详见MAC地址检查。 |

### 注意事项

尽管IEEE规范定义了一个最大的帧长度，但MAC核心提供了灵活性，可以为帧的最大长度编程任何值。

#### 52.6.1.1 暂停帧数

接收设备产生一个暂停帧，向发射设备表示拥堵，发射设备应停止发送数据。

暂停帧由长度/类型设置为0x8808表示。暂停帧的前两个字节在类型之后，定义了一个16位的操作码字段，始终设置为0x0001。一个16位的暂停量子被定义在帧有效载荷字节2（P1）和3（P2）中，如下表所定义。P1暂停量子字节是最重要的。

**表52-4.暂停帧格式（数值为十六进制）**

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 55 | 55 | 55 | 55 | 55 | 55 | 55 | D5 | 01 | 80 | C2 | 00 | 00 | 01 |
| 序言 | | | | | | | 证券日报 | 多播目的地地址 | | | | | |
| 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 -68 | |
| 00 | 00 | 00 | 00 | 00 | 00 | 88 | 08 | 00 | 01 | 嗨 | 掠夺 | 00 | |
| 来源地址 | | | | | | 类型 | | 操作码 | | P1 | P2 | 垫子 (42) | |
| 69 | 70 | 71 | 72 | | | | | | | | | | |
| 26 | 6B | AE | 0A | | | | | | | | | | |
| CRC-32 | | | | | | | | | | | | | |

暂停帧内没有有效载荷长度字段，暂停帧总是用42字节（0x00）填充。

如果收到一个暂停值大于零的暂停帧（XOFF条件），MAC在当前帧传输完成后立即停止传输数据。MAC停止传输数据的值是以暂停量子定义的。一个暂停量子的分数是指512比特时间。

如果收到一个暂停值为零的暂停帧（XON条件），允许发射器立即发送数据（详见全双工流控操作）。

#### 52.6.1.2 神奇的数据包

魔法数据包是一个单播、多播或广播数据包，它在有效载荷部分携带一个定义的

序列。

魔法数据包仅在特定条件下被接收和检查。

解码魔法数据包的定义序列是由一个同步流组成的，该同步流由六个连续的0xFF字节组成，然后是要被唤醒的节点的十六个连续单播MAC地址序列。

这个序列可以位于魔法包有效载荷的任何地方。魔法数据包由标准的以太网头、可选的填充物和CRC组成。

## 52.6.2 IP和更高层次的帧格式

例如，一个IP数据报指定一个以太网帧的有效载荷部分。TCP数据报指定了IP数据报中的有效载荷部分。

### 52.6.2.1 以太网类型

IP数据报是在以太网帧的有效载荷部分进行的。以太网帧类型/长度字段可区分几种数据报类型。

下表列出了兴趣的类型。

**表52-5.以太网类型值示例**

| 类型 | 描述 |
|---|---|
| 0x8100 | VLAN标记的帧。实际的类型是在帧的4个八位字节之后发现的。 |
| 0x0800 | IPv4 |
| 0x0806 | ARP |
| 0x86DD | ÃÃÃ |

### 52.6.2.2 IPv4数据报格式

下图显示了IP第四版（IPv4）报头，它位于IP数据报的开头。它以32位字组织。发送/接收的第一个字节是第一个字的最左边的字节（换言之，版本/IHL字段）。

IP头可以包含更多的选项，如果有必要，这些选项总是被填充，以保证头之后的有效载荷被对齐到32位边界。

IP头之后紧接着是有效载荷，它可以包含进一步的协议头（例如，TCP或UDP,

由协议字段值表示）。完整的IP数据报在以太网帧的有效载荷部分进行传输。

**表52-6.IPv4报头格式**

| 31 30 29 28 | 27 26 25 24 | 23 22 21 20 | 19 18 17 16 | 15 14 13 12 | 11 10 9 8 | 7 6 5 4 | 3 2 1 0 |
|---|---|---|---|---|---|---|---|
| 版本 | IHL | TOS | | 长度 | | | |
| 片段ID | | | | 旗帜 | 片段偏移 | | |
| TTL | | 议定书 | | 头部检查和 | | | |
| 来源地址 | | | | | | | |
| 目的地地址 | | | | | | | |
| 选项 | | | | | | | |

**表52-7.IPv4报头字段**

| 领域名称 | 描述 |
|---|---|
| 版本 | 4位IP版本信息。0x4用于IPv4帧。 |
| IHL | 4位互联网头的长度信息。确定在IP头中发现的32位字的数量。如果没有选项，默认值为0x5。 |
| TOS | 服务类型/DiffServ字段。 |
| 长度 | 数据报的总长度，以字节为单位，包括报头和有效载荷的所有八位字节。 |
| 片段ID，标志，片段偏移量 | 用于IP分片的字段。 |
| TTL | 存活时间。实际上，在每个路由器到达时被递减。如果为零，数据报必须被丢弃。 |
| 议定书 | 数据报中后续协议的标识符。 |
| 头部检查和 | IP头的校验和。出于计算的目的，这个字段的值为零。 |
| 来源地址 | 源IP地址。 |
| 目的地地址 | 目的地IP地址。 |

### 52.6.2.3 IPv6数据报格式

下图显示了IP版本6（IPv6）的头，它位于IP数据报的开头。它以32位字组织，有10个字（40字节）的固定长度。下一个头字段确定了IPv6头之后的头的类型。它的定义类似于IPv4内的协议标识符，具有识别扩展头的新定义。这些头可以插入到IPv6头和协议头之间，这将相应地转移协议头。加速器目前只支持没有扩展头的IPv6（换句话说，下一个头指定TCP、UDP或IMCP）。

发送/接收的第一个字节是第一个字的最左边的字节（换言之，版本/流量类别字段）。

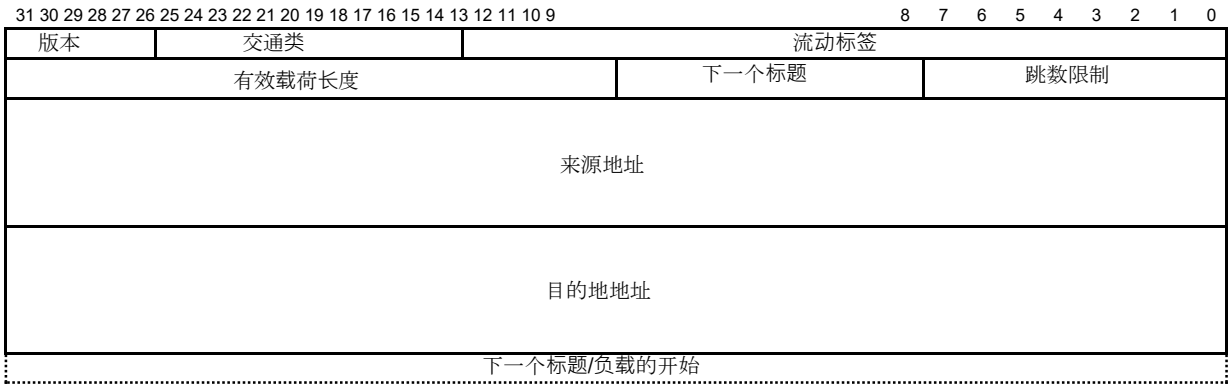



**图52-4.IPv6头的格式 表52-8.IPv6头的字段**

| 领域名称 | 描述 |
|---|---|
| 版本 | 4位IP版本信息。0x6用于所有IPv6帧。 |
| 交通类 | 定义流量类别的8位字段。 |
| 流动标签 | 20位流量标签，识别同一流量的帧。 |
| 有效载荷长度 | 数据报有效载荷的16位长度（字节）。它包括IPv6头之后的所有八位字节。 |
| 下一个标题 | 标明IPv6头之后的头。这可以是协议头或任何IPv6定义的扩展头。 |
| 跳数限制 | 跳数，由转发该帧的每个站减一。如果跳数限制为0，该帧必须被丢弃。 |
| 来源地址 | 128位IPv6源地址。 |
| 目的地地址 | 128位IPv6目标地址。 |

## 52.6.2.4 互联网控制信息协议（ICMP）数据报格式

如果协议标识符为1，则在IP头之后发现一个互联网控制信息协议（ICMP）。ICMP数据报有一个四字节的头，后面是附加的信息数据。

**表52-9.ICMP头的格式**

| 31 30 29 28 | 27 26 25 24 | 23 22 21 20 | 19 18 17 16 | 15 14 13 12 | 11 10 9 8 | 7 6 5 4 | 3 2 1 0 |
|---|---|---|---|---|---|---|---|
| 类型 | | 编码 | | 校验和 | | | |
| ICMP消息数据 | | | | | | | |

**表52-10.IP报头字段**

| 领域名称 | 描述 |
|---|---|
| 类型 | 8位类型信息 |

| | |
|---|---|
| 编码 | 与信息类型有关的8位代码 |

*表继续在下一页...*

表52-10.IP头字段（续）

| 领域名称 | 描述 |
| --- | --- |
| 校验和 | 对完整的ICMP数据报进行16位的一补校验。 |

### 52.6.2.5 用户数据报协议（UDP）数据报格式

在IP头之后发现一个用户数据报协议头，此时协议标识符为17。

数据报的有效载荷在UDP报头之后。头部的字节顺序遵循上述IP头的惯例。

表52-11.UDP报头格式

| 31 30 29 28 | 27 26 25 24 | 23 22 21 20 | 19 18 17 16 | 15 14 13 12 | 11 10 9 8 | 7 6 5 4 | 3 2 1 0 |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 来源端口 | | | | 目的地端口 | | | |
| 长度 | | | | 校验和 | | | |

表52-12.UDP报头字段

| 领域名称 | 描述 |
| --- | --- |
| 来源端口 | 源应用程序端口 |
| 目的地端口 | 目的地应用程序端口 |
| 长度 | 紧随报头的用户数据的长度，包括UDP报头（即最小值为8）。 |
| 校验和 | 对完整的数据报和一些IP头信息进行校验 |

### 52.6.2.6 TCP数据报格式

当协议标识符的值为6时，在IP头之后会发现一个TCP头。

TCP有效载荷紧随TCP头。

表52-13.TCP报头格式

| 31 30 29 28 | 27 26 25 24 | 23 22 21 20 | 19 18 17 16 | 15 14 13 12 | 11 10 9 8 | 7 6 5 4 | 3 2 1 0 |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 来源端口 | | | | 目的地端口 | | | |
| 顺序号 | | | | | | | |
| 鸣谢号码 | | | | | | | |

*表继续在下一页...*

表52-13.TCP报头格式（续）

| | | | |
|---|---|---|---|
| 偏移 | 保留 | 旗帜 | 窗口 |
| 校验和 | | | 紧急指示器 |
| 选择 | | | |

表52-14.TCP报头字段

| 领域名称 | 描述 |
|---|---|
| 来源端口 | 源应用程序端口 |
| 目的地端口 | 目的地应用程序端口 |
| 序列数 | 发送序列号 |
| 呼叫号码 | 接收序列号 |
| 偏移 | 数据偏移量，是TCP头中32位字的数量 - 如果没有选择，默认值为5 |
| 旗帜 | URG, ACK, PSH, RST, SYN, FIN标志 |
| 窗口 | TCP接收窗口大小信息 |
| 校验和 | 对完整的数据报（TCP报头和数据）和IP报头信息进行校验 |
| 选择 | 用于协议选项的额外32位字 |

## 52.6.3 IEEE 1588信息格式

以下章节描述了IEEE 1588的信息格式。

### 52.6.3.1 运输封装

精确时间协议（PTP）数据报使用UDP/IP传输机制封装在以太网帧中，也可以选择使用较新的1588v2直接封装在以太网帧中（第二层）。

通常情况下，多播地址被用来实现同步信息的有效分发。

#### 52.6.3.1.1 UDP/IP

1588消息（v1和v2）可以使用UDP/IP组播消息进行传输。

表52-15显示了为PTP定义的IP组播组。该表还根据RFC 1112显示了它们各自的MAC层组播地址映射（IP的最后三个字节遵循01-00-5E的固定值）。

**表52-15.UDP/IP多播域**

| 命名 | IP地址 | MAC地址映射 |
|---|---|---|
| 默认的PTPdomain | 224.0.1.129 | 01-00-5E-00-01-81 |
| 备用PTPdomain1 | 224.0.1.130 | 01-00-5E-00-01-82 |
| 备用PTPdomain2 | 224.0.1.131 | 01-00-5E-00-01-83 |
| 备用PTPdomain3 | 224.0.1.132 | 01-00-5E-00-01-84 |

**表52-16.UDP端口号**

| 信息类型 | UDP端口 | 注意事项 |
|---|---|---|
| 活动 | 319 | 用于SYNC和DELAY_REQUEST信息 |
| 一般 | 320 | 所有其他信息（例如，跟进、延迟回复）。 |

#### 52.6.3.1.2 本地以太网（PTPv2）

除了使用UDP/IP帧外，IEEE 1588v2还定义了一种本地以太网帧格式，使用ethertype = 0x88F7。以太网帧的有效载荷立即包含PTP数据报，从PTPv2的头开始。

除其他外，版本2增加了一个对等延迟机制，允许在多个节点的路径上的单个点对点链接之间进行延迟测量。在PTPv2中还定义了以下多播域。

**表52-17.PTPv2多播域**

| 命名 | MAC地址 |
|---|---|
| 正常信息 | 01-1B-19-00-00-00 |
| 同行延迟信息 | 01-80-C2-00-00-0E |

### 52.6.3.2 PTP标头

所有的PTP帧都包含一个确定协议版本和信息类型的通用头，它定义了信息的其余内容。

所有的多八位数字段都是以大数顺序传输的（最重要的字节先被传输/接收）。

对于PTPv1和PTPv2报头，versionPTP的最后四位处于相同的位置（第二个字节）。这允许通过检查信息的前两个字节进行准确识别。

#### 52.6.3.2.1 PTPv1头

**表52-18.常见的PTPv1消息头**

| 偏移 | 八位数 | 咬牙<br>切齿 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0 | 2 | 版本PTP = 0x0001 | | | | | | | |
| 2 | 2 | 版本网络 | | | | | | | |
| 4 | 16 | 子域 | | | | | | | |
| 20 | 1 | 消息类型 | | | | | | | |
| 21 | 1 | 来源：通信技术 | | | | | | | |
| 22 | 6 | sourceUuid | | | | | | | |
| 28 | 2 | sourcePortId | | | | | | | |
| 30 | 2 | sequenceId | | | | | | | |
| 32 | 1 | 控制 | | | | | | | |
| 33 | 1 | 0x00 | | | | | | | |
| 34 | 2 | 旗帜 | | | | | | | |
| 36 | 4 | 保留 | | | | | | | |

消息的类型在messageType和control字段中进行编码，如表52-19所示。

**表52-19.PTPv1消息类型识别**

| 消息类型 | 控制 | 信息名称 | 留言 |
|---|---|---|---|
| 0x01 | 0x0 | 同步性 | 事件消息 |
| 0x01 | 0x1 | 呼叫中心：DELAY_REQ | 事件消息 |
| 0x02 | 0x2 | 跟进_UP | 一般信息 |
| 0x02 | 0x3 | 呼叫中心 | 一般信息 |
| 0x02 | 0x4 | 管理部门 | 一般信息 |
| 其他 | 其他 | - | 保留 |

字段sequenceId用于非明确地识别一条信息。

#### 52.6.3.2.2 PTPv2标头

表52-20.常见的PTPv2消息头

| 偏移 | 八位数 | 咬牙<br>切齿 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0 | 1 | 特定运输 | | | | messageId | | | |
| 1 | 1 | 保留 | | | | 版本PTP = 0x2 | | | |
| 2 | 2 | 消息长度 | | | | | | | |
| 4 | 1 | 域名 | | | | | | | |
| 5 | 1 | 保留 | | | | | | | |
| 6 | 2 | 旗帜 | | | | | | | |
| 8 | 8 | 更正字段 | | | | | | | |
| 16 | 4 | 保留 | | | | | | | |
| 20 | 10 | 源端口名称（sourcePortIdentity | | | | | | | |
| 30 | 2 | sequenceId | | | | | | | |
| 32 | 1 | 控制 | | | | | | | |
| 33 | 1 | 讯息的平均时间 | | | | | | | |

消息的类型在字段messageId中编码如下。

表52-21.PTPv2消息类型识别

| messageId | 信息名称 | 留言 |
|---|---|---|
| 0x0 | 同步性 | 事件消息 |
| 0x1 | 呼叫中心：DELAY_REQ | 事件消息 |
| 0x2 | PATH_DELAY_REQ | 事件消息 |
| 0x3 | 路径_延时_响应 | 事件消息 |
| 0x4-0x7 | - | 保留 |
| 0x8 | 跟进_UP | 一般信息 |
| 0x9 | 呼叫中心 | 一般信息 |
| 0xa | 路径_延时_跟进 | 一般信息 |
| 0xb | 告知 | 一般信息 |
| 0xc | 讯号 | 一般信息 |
| 0xd | 管理部门 | 一般信息 |

PTPv2标志字段包含关于信息类型的进一步细节，特别是如果使用单步或两步的实现。单步或两步实现由标志字段第一个八位中的TWO_STEP位控制，如下所示。

保留位被清空。

表52-22.PTPv2消息标志的字段定义

| 位 | 命名 | 描述 |
|---|---|---|
| 0 | 替代品 | 见IEEE 1588第17.4条 |
| 1 | 双轨制 | 1 两步式时钟<br>0 单步式时钟 |
| 2 | 贸发会议 | 1 传输层地址使用单播目的地址<br>0 使用组播 |
| 3 | - | 保留 |
| 4 | - | 保留 |
| 5 | 具体概况 | |
| 6 | 具体概况 | |
| 7 | - | 保留 |

## 52.6.4 MAC接收

- 检查框架构架
- 删除帧序言和帧SFD字段
- 根据帧目的地地址字段丢弃帧
- 终止暂停帧
- 检查框架长度
- 如果存在有效载荷的填充物，则删除它
- 计算和验证CRC-32
- 在核心接收FIFO中写入收到的帧

如果MAC被编程为在半双工模式下工作，它也将检查是否收到了碰撞的帧。

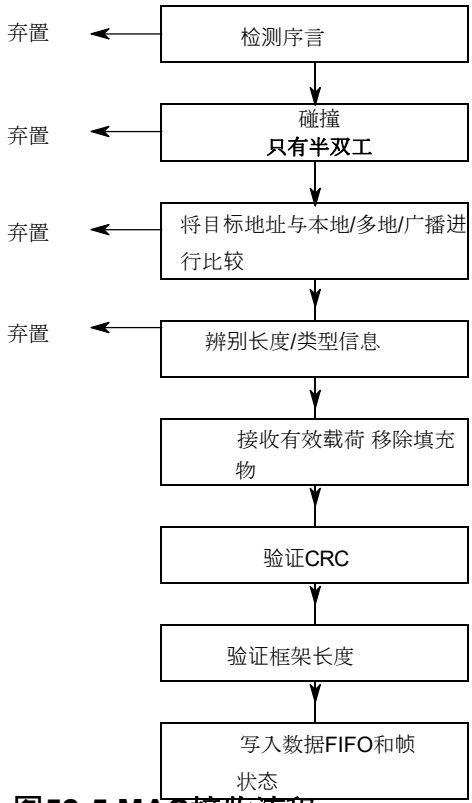



**图52-5.MAC接收流程**

### 52.6.4.1 半双工模式下的碰撞检测

如果数据包在接收前64个字节时检测到碰撞，数据包将被丢弃（如果帧大小小于~14个八位字节）或以错误和RxBD[CE]设置传送给用户应用程序。

### 52.6.4.2 序言处理

IEEE
802.3标准允许序言的最大尺寸为56比特（7字节），而MAC核心允许任何序言长度，包括零长度的序言。

MAC核心检查起始帧定界符（SFD）字节。如果序言的下一个字节，与0x55不同，不是0xD5，则该帧被丢弃。

尽管IEEE规范规定，内包间隙应至少为96位，但MAC核心被设计为只接受由64个10/100-Mbit/s操作（MII）位分隔的帧。

MAC核心删除序言和SFD字节。

#### 52.6.4.3 MAC地址检查

目标地址第0位区分了多播和单播地址。

- 如果第0位为0，MAC地址是一个单独的（单播）地址。
- 如果第0位为1，MAC地址定义了一个组（组播）地址。
- 如果MAC地址的所有48位都被设置，则表明是一个广播地址。

##### 52.6.4.3.1 单播地址检查

如果收到一个单播地址，目的MAC地址将与主机在PADDR1/2寄存器中编程的节点MAC地址进行比较。

如果目标地址与任何一个编程的MAC地址相匹配，则接受该帧。

如果启用混杂模式（RCR[PROM] =
1），则不进行地址检查，所有单播帧都被接受。

##### 52.6.4.3.2 多播和单播地址解析

组和个人哈希过滤中使用的哈希表算法操作如下。

- 48位目的地址被映射成64位中的一位，由ENETn_GAUR/GALR（组地址哈希匹配）或ENETn_IAUR/IALR（个别地址哈希匹配）中的64位表示。
- 这种映射是通过将48位地址通过片上32位CRC发生器，并选择CRC编码结果的六个最重要的位来产生一个0到63之间的数字。
- CRC结果的msb选择ENETn_GAUR（msb = 1）或ENETn_GALR（msb = 0）。
- 哈希结果的五个lsbs选择所选寄存器内的位。
- 如果CRC生成器选择了哈希表中的一个比特集，则该帧被接受；否则，它被拒绝。

例如，如果在散列表中存储了8个组地址，并且收到了随机组地址，散列表阻止了大约56/64（或87.5%）的组地址帧到达内存。那些到达内存的帧必须由处理器进一步过滤，以确定它们是否真的包含八个所需地址之一。

哈希表的有效性随着地址数量的增加而下降。

用户必须初始化哈希表的寄存器。使用这个CRC32多项式来计算哈希值。

- FCS(x) = x32+ x26+ x23+ x22+ x16+ x12+ x11+ x10+ x8+ x7+ x5+ x4+ x2+ x1+ 1

如果启用混杂模式(ENETn_RCR[PROM] = 1)，无论ENETn_GAUR/GALR和ENETn_IAUR/IALR设置如何，所有单播和多播帧都被接受。

#### 52.6.4.3.3 拒绝广播地址

如果BC_REJ被清除或ENETn_RCR[PROM]被设置，所有的广播帧都被接受。如果ENETn_RCR[BC_REJ]被设置时，PROM被清除，所有的广播帧都被拒绝。

**表52-23.广播地址拒绝编程**

| PROM | BC_REJ | 广播框架 |
|---|---|---|
| 0 | 0 | 已接受 |
| 0 | 1 | 被拒绝的 |
| 1 | 0 | 已接受 |
| 1 | 1 | 已接受 |

#### 52.6.4.3.4 错位执行

对于上层过滤的目的，RxBD[M]表示当MAC在混杂模式下运行并接受一个本来会被拒绝的帧时，会出现地址丢失。

如果没有出现组/个体散列或精确匹配，并且启用了淫乱模式（RCR[PROM] = 1），则接受该帧，并在缓冲区描述符中设置M位；否则，拒绝该帧。

这意味着在淫乱模式下，状态位在以下任何一种情况下被设置。

- 当BC_REJ被设置时，收到一个广播帧
- 收到的单播信号也不匹配。

- 节点地址（PALR[PADDR1]和PAUR[PADDR2]）。
- 单播的哈希表（IAUR[IADDR1] 和 IALR[IADDR2]）。

- 收到一个与GAUR[GADDR1]和GALR[GADDR2]哈希表项不匹配的多播。

#### 52.6.4.4 帧长度/类型验证：有效载荷长度检查

如果长度/类型小于0x600且NLC被设置，MAC检查有效载荷长度，并在帧状态字和中断位PLR中报告任何错误。

如果长度/类型大于或等于0x600，MAC将该字段解释为一个类型，不进行有效载荷长度检查。

长度检查在VLAN和堆叠的VLAN帧上执行。如果收到一个填充的帧，由于扩展的帧有效载荷，不能进行长度检查，因为填充的帧永远不会有有效载荷长度错误。

#### 52.6.4.5 帧长/类型验证：帧长检查

当接收帧长度超过MAX_FL字节时，BABR中断产生，RxBD[LG]位被设置。

除非帧的长度超过
ENETn_FTRL[TRUNC_FL]中编程的值，否则帧不会被截断。如果帧被截断，RxBD[TR]被设置。此外，截断的帧总是有CRC错误指示被设置（RxBD[CR]）。

#### 52.6.4.6 VLAN帧处理

VLAN帧的长度/类型字段设置为0x8100，紧接着是一个16位的VLAN控制信息字段。

带VLAN标签的帧作为正常帧接收，因为VLAN标签不被MAC功能解释，并与VLAN标签一起完整推送给用户应用。如果VLAN标记的帧的长度/类型字段（在帧的四个八位字节后发现）小于42，则填充物被删除。此外，帧状态字（RxBD[NO]）表明当前帧是VLAN标记的。

### 52.6.4.7 暂停帧终止

接收引擎终止暂停帧，不将其转移到接收FIFO。量子被提取出来，并通过一个小型的内部时钟速率解耦异步FIFO发送到MAC发送路径。

只有当控制状态机检测到正确的CRC和帧长时，才会写入量子。如果不是，量子被丢弃，MAC传输路径不会暂停。

如果ENETn_RCR[FCE]被清除，良好的暂停帧会被忽略，而当ENETn_RCR[PAUFWD]被设置时，良好的暂停帧会被转发到客户端接口。

### 52.6.4.8 CRC检查

如果ENETn_RCR[CRCFWD]被清零且ENETn_RCR[PADEN]被设置，则CRC-32字段被检查并转发到核心FIFO接口。

当CRCFWD被设置时（与PADEN无关），CRC-32字段被检查并终止（不传送到FIFO）。

在802.3标准中规定的CRC多项式是：。

- FCS(x) = x32+ x26+ x23+ x22+ x16+ x12+ x11+ x10+ x8+ x7+ x5+ x4+ x2+ x1+ 1

CRC值的32位被放在帧校验序列（FCS）字段中，x31项为第一个八位的最右位。因此，CRC位按以下顺序接收：x31，x30，...，x1，x0。

如果检测到CRC错误，该帧被标记为无效，RxBD[CR]被设置。

### 52.6.4.9 框架填充物的移除

当收到一个有效载荷长度字段设置为小于46的帧时（对于VLAN标记的帧为42，对于堆叠VLAN的帧为38），在帧被写入数据FIFO之前，可以根据ENETn_RCR[PADEN]的设置去除零填充。

**注意事项**

如果收到一个带有多余填充物的帧（换句话说，长度字段如上所述被设置，但该帧有超过64个八位字节）并且填充物移除被启用，那么

如果该帧在其他方面是正确的（例如：良好的CRC，小于最大长度，并且没有其他错误），则填充物将被正常删除，并且不报告错误。

### 52.6.5 MAC传输

当发送FIFO容纳足够的数据时，帧传输开始。

在传输开始后，MAC传输功能执行以下任务。

- 在帧传输前，生成预载波和SFD字段
- 如果接收FIFO报告拥堵，或者ENETn_TCR[TFC_PAUSE]被设置为ENETn_OPD[PAUSE_DUR]的非零值，则生成XOFF暂停帧。
- 如果接收FIFO拥堵条件被清除，或者TFC_PAUSE被设置且PAUSE_DUR被清除，则生成XON暂停帧。
- 如果从MAC接收路径收到一个非零的暂停量子，则暂停以太网帧传输（XOFF）
- 如果需要，为框架添加填充物
- 计算并将CRC-32附加到传输的帧上
- 以正确的包间间隙（IPG）发送帧（推迟）。

当MAC被配置为在半双工模式下运行时，将执行以下额外任务。

- 碰撞检测
- 帧重传在后退计时器过期后

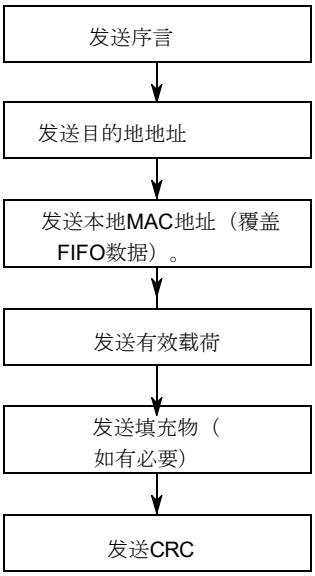



**图52-6.帧传输概述**

### 52.6.5.1 帧有效载荷填充

IEEE规范定义的最小帧长为64字节。

如果从用户应用程序发送到MAC的帧的大小小于60字节，MAC会自动添加填充字节（0x00）以符合以太网最小帧长度规范。传输填充总是被执行，不能被禁用。

如果MAC不允许附加CRC（TxBD[TC]=1），用户应用负责提供最小长度为64个字节的帧。

### 52.6.5.2 插入MAC地址

在每个从核心发送FIFO接口收到的帧上，源MAC地址是：。

- 由PADDR1/2字段中编程的地址取代（ENETn_TCR[ADDINS] = 1）。
- 透明地转发到以太网线路（ENETn_TCR[ADDINS] = 0）。

### 52.6.5.3 CRC-32生成

CRC-32字段可选择生成并附加在帧的末端。

在802.3标准中规定的CRC多项式是：。

- FCS(x) = x32+ x26+ x23+ x22+ x16+ x12+ x11+ x10+ x8+ x7+ x5+ x4+ x2+ x1+ 1

CRC值的32位被放在FCS字段中，因此x31项是第一个八位数的最右边的位。因此，CRC位按以下顺序传输：x31，x30，...，x1，x0。

### 52.6.5.4 包间间隙（IPG）

在全双工模式下，在帧传输后和传输新帧前，保持一个IPG（在ENETn_TIPG中编程）。最小的IPG可以在8到26个字节-次（64到208个比特-次）之间编程。

在半双工模式下，核心不断监测线路。只有在闲置了96位的时间段，并且满足了任何回退时间的要求时，才会将数据实际传输到网络上。根据标准，核心从CRS的去assertion开始测量IPG。

### 52.6.5.5 碰撞检测和处理 - 仅限半双工操作

在半双工网络中，当两个或多个节点同时进行传输时，就会发生碰撞。在传输过程中，核心监控线路状况，并在PHY设备发出COL信号时检测到碰撞。

当核心在传输过程中检测到碰撞时，它会停止传输数据，并传输一个32位的干扰模式。如果在序言或SFD传输过程中检测到碰撞，则在完成SFD后传输干扰模式，从而形成一个最小的96位片段。干扰模式是一个固定的模式，不与实际的帧CRC相比较，而且干扰模式与CRC相同的概率非常低（0.532）。

如果在传输64个字节（包括序言和SFD）之前发生碰撞，MAC核心等待后退期，并重新传输已经在线路上发送的数据包（存储在64个字节的重新传输缓冲区）。反冲期是由一个伪随机过程产生的（截断的二进制指数反冲）。

如果在传输64个字节（包括序言和SFD）后发生碰撞，MAC会丢弃剩余的帧，可选择设置LC中断位，并设置TxBD[LCE]。

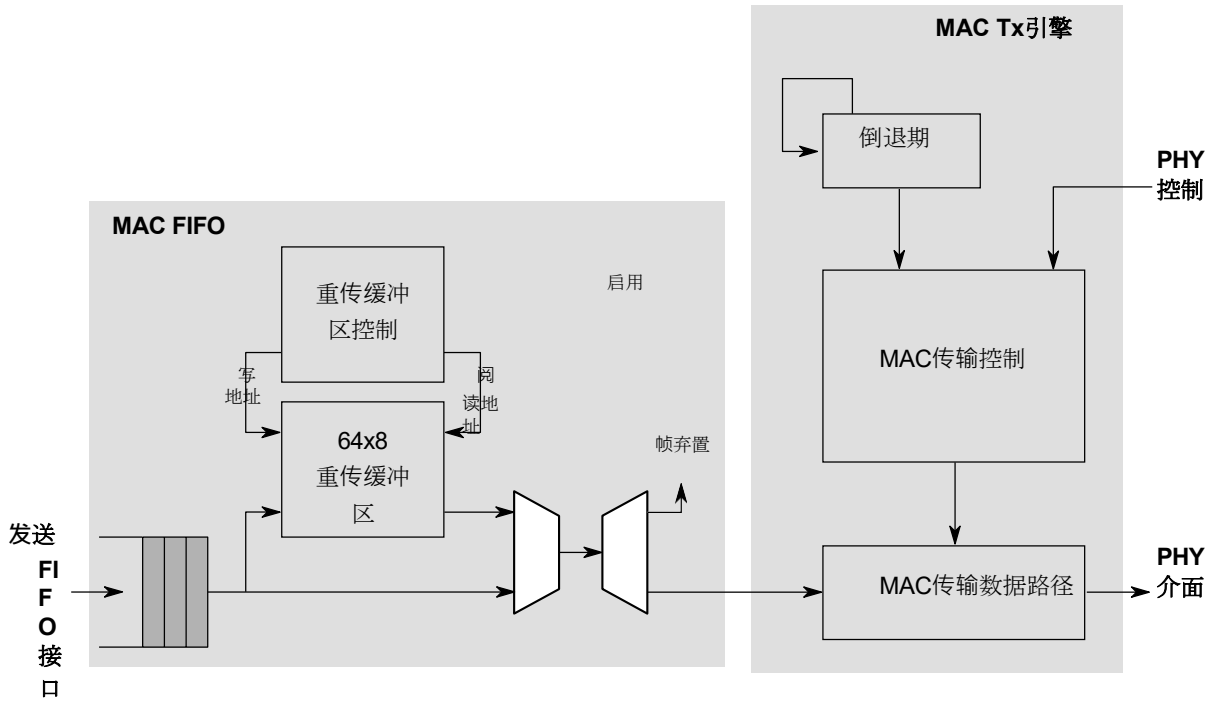



**图52-7.数据包重传概述**

后退时间由槽次的整数倍表示。一个时隙等于一个512位的时间段。在$n^{th}$次重传尝试之前，延迟槽次的数量被选为范围内的一个均匀分布的随机整数。

- $0 < r < 2k$
- $k = \min(n, N)$; 其中$n$是重传的数量，$N = 10$

例如，在第一次碰撞后，退避期为0或1个槽位时间。如果在第一次重传时发生碰撞，退避期为0、1、2或3，以此类推。

按照IEEE 802.3标准的规定，最大退避时间（以512位时间槽为单位）由N = 10限制。

如果在连续16次重传后发生碰撞，核心会报告一个过度碰撞条件（ENETn_EIR[RL]中断字段和TxBD[EE]）并丢弃FIFO中的当前数据包。

在违反标准要求的网络中，在传输完前64个字节后可能会发生碰撞。在这种情况下，内核停止当前的数据包传输，并丢弃发送FIFO中的其余数据包。内核用内核发送FIFO中可用的下一个数据包恢复传输。

**警告**

支持SQE测试或 "心跳
"功能的以太网PHY必须禁用此功能。当该功能被启用时。

PHY在传输完一帧后会发出碰撞信号，以向ENET表明PHY的碰撞逻辑正在工作。这可能导致来自ENET的下一个帧的数据损坏。这个损坏的帧最多包含21个零字节，这些零字节从MAC目的地址字段的某个地方开始。然而，ENET仍将产生一个良好的FCS（CRC-32），但有损坏的数据。

## 52.6.6 全双工流量控制操作

内核的流量控制引擎处理三种情况。

- 远程设备拥堵 -
  连接到与核心网段相同的远程设备报告拥堵，要求核心网停止发送数据。
- 核心FIFO拥堵 -
  当核心的接收FIFO达到用户可编程的阈值（RX部分为空）时，核心会向远程设备发送一个暂停帧，要求停止数据传输。
- 本地设备拥堵 -
  任何连接到核心的设备都可以要求（通常是通过主机处理器）远程设备停止传输数据。

### 52.6.6.1 远程设备拥堵

当MAC发送控制从接收路径得到一个有效的暂停量子，并且如果ENETn_RCR[FCE]被设置，MAC发送逻辑。

- 完成当前帧的传输。
- 停止发送数据，时间由暂停量指定，以512比特时间为增量。
- 设置ENETn_TCR[RFC_PAUSE]。

当量子指定的时间过后，如果没有收到新的量子值，或者收到量子值设置为0x0000的新暂停帧，帧传输就会恢复。MAC还将RFC_PAUSE重置为零。

如果ENETn_RCR[FCE]清零，MAC将忽略收到的暂停帧。

如果设置了 PAUFWD，暂停帧将被转发到客户接口，这与 ENETn_RCR[FCE] 无关。

### 52.6.6.2 本地设备/FIFO拥堵

当本地接收FIFO无法接收超过预定的字数（FIFO可编程阈值）或本地主机处理器要求生成暂停帧时，MAC发送引擎会生成暂停帧。

- 为了产生一个暂停帧，主机处理器设置ENETn_TCR[TFC_PAUSE]。当当前的帧传输完成后，会产生一个暂停帧，TFC_PAUSE被自动清除。也可以选择生成一个中断。
- 当接收FIFO发出空段标志（内部）时，会产生一个XOFF暂停帧。当当前的帧传输完成后，会自动生成一个XOFF暂停帧。
- 当接收FIFO取消其部分空标志（内部）时，将产生一个XON暂停帧。当当前的帧传输完成后，会自动生成一个XON暂停帧。

当产生一个 XOFF 暂停帧时，暂停量子（有效载荷字节 P1 和 P2）被填充为ENETn_OPD[PAUSE_DUR]中编程的值。

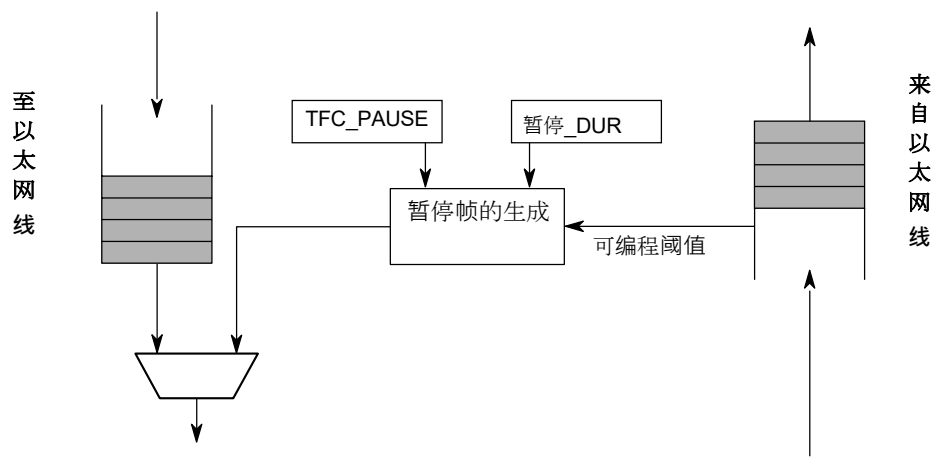



**图52-8.暂停帧生成概述**

#### 注意事项

尽管流量控制机制应该防止MAC核心接收路径上的任何FIFO溢出，但核心接收FIFO是受保护的。当检测到接收FIFO的溢出时，当前帧被截断，并在帧状态字中设置错误指示。该帧随后应被用户应用丢弃。

## 52.6.7 神奇的数据包检测

魔法数据包检测唤醒被节点管理代理置于断电模式的节点。只有当MAC被配置为睡眠模式时，才支持魔法数据包检测。

### 52.6.7.1 睡眠模式

要使 MAC 处于睡眠模式，请设置 ENETn_ECR[SLEEP]。同时ENETn_ECR[MAGICEN]也应被设置，以启用魔法数据包检测。

此外，当处理器处于停止模式时，进入睡眠模式，不影响ENETn_ECR寄存器位。

当MAC处于睡眠模式时。

- 发送逻辑被禁用。
- FIFO的接收/发送功能被禁用。
- 接收逻辑保持在正常模式，但它忽略了来自线路的所有流量，除了魔术包。它们被检测到，以便远程代理可以唤醒节点。

### 52.6.7.2 神奇的数据包检测

该核心被设计用来检测目标地址设置为的魔法数据包。

- 任何多播地址
- 广播地址
- 在PADDR1/2中编程的单播地址

当检测到一个神奇的数据包时，EIR[WAKEUP]被设置，没有一个统计寄存器被递增。

### 52.6.7.3 唤醒

当检测到魔法数据包时，由
ENETn_EIR[WAKEUP]指示，ENETn_ECR[SLEEP]应被清除，以恢复 MAC 的正常运行。清除 SLEEP 位会自动屏蔽
ENETn_ECR[MAGICEN]，禁用魔法包检测。

## 52.6.8 IP加速器功能

### 52.6.8.1 校验和计算

IP和ICMP、TCP、UDP的校验和是用一补的算术计算出来的，是16位的值。

- 对于ICMP，校验是通过完整的ICMP数据报计算的，换句话说，没有IP头。
- 对于TCP和UDP来说，校验包含了IP头和数据部分以及IP头的值，它可以被看作是数据流中实际不存在的一个伪头。

**表52-24.用于计算校验和的IPv4伪报头**

| 31 30 29 28 | 27 26 25 24 | 23 22 21 20 | 19 18 17 16 | 15 14 13 12 | 11 10 9 8 | 7 6 5 4 | 3 2 1 0 |
|---|---|---|---|---|---|---|---|
| 来源地址 | | | | | | | |
| 目的地地址 | | | | | | | |
| 零度 | | 议定书 | | TCP/UDP长度 | | | |

**表52-25.用于计算校验和的IPv6伪报头**

| 31 30 29 28 | 27 26 25 24 | 23 22 21 20 | 19 18 17 16 | 15 14 13 12 | 11 10 9 8 | 7 6 5 4 | 3 2 1 0 |
|---|---|---|---|---|---|---|---|
| 来源地址 | | | | | | | |
| 目的地地址 | | | | | | | |
| TCP/UDP长度 | | | | | | | |
| 零度 | | | | | | 下一个标题 | |

TCP/UDP长度值是TCP或UDP数据报的长度，它相当于IP数据报的有效载荷。它是通过从IP头（IPv4）中给出的完整的IP数据报长度中减去IP头的长度，或者直接从IP头（IPv6）中获取的。协议字段是来自IP头的相应值。
零字段都是零。

对于IPv6，考虑的是完整的128位地址。下一个头值确定了上层协议是TCP还是UDP。如果在协议头之前插入了扩展头，它可能与IPv6头的下一个头值不同。

校验和计算使用网络字节顺序的16位字。发送/接收的第一个字节是MSB，发送/接收的第二个字节是16位值的LSB，添加到校验和中。如果帧以奇数字节结束，则附加一个零字节，仅用于计算校验和，而不是实际传输。

### 52.6.8.2 额外的填充处理

根据IEEE 802.3，任何以太网帧的最小长度必须是64个字节。

当收到带有长度信息的帧时，MAC通常会在接收时移除填充物。因为IP帧有一个类型值而不是长度，MAC对短的IP帧不去除填充，因为它不知道帧的内容。

可以对IP加速器功能进行配置，以去除可能跟随IP数据报的以太网填充字节。

在传输过程中，MAC会根据需要自动添加填充物，使任何帧的长度达到64字节。

### 52.6.8.3 32位以太网有效载荷对齐

数据FIFO允许在一个帧的前面插入两个额外的任意字节。这将14字节的以太网头扩展为16字节的头，这导致以太网有效载荷在以太网头之后以32位边界对齐。

该功能可以通过ENETn_TACC和ENETn_RACC寄存器中相应的SHIFT16位独立启用发送和接收。

启用后，有效帧数据的排列方式如表52-26所示。

**表52-26.启用SHIFT16后的64位接口数据结构**

| 63 56 | 55 48 | 47 40 | 39 32 | 31 24 | 23 16 | 15 8 | 7 0 |
|---|---|---|---|---|---|---|---|
| 字节5 | 字节4 | 字节3 | 字节2 | 字节1 | 字节0 | 任何价值 | 任何价值 |
| 字节13 | 字节12 | 字节11 | 字节10 | 字节9 | 字节8 | 字节7 | 字节6 |
| ... | | | | | | | |

#### 52.6.8.3.1 接收处理

当ENETn_RACC[SHIFT16]被设置时，每个帧在接收时前面有两个附加字节。

用户应用必须忽略这前两个字节，在RX FIFO的第一个字的第23-16位找到帧的第一个字节。

**注意事项**

SHIFT16必须在初始化过程中被设置，并在整个操作过程中保持设置，因为它影响到FIFO的写入行为。

#### 52.6.8.3.2 发送处理

当ENETn_TACC[SHIFT16]被设置时，写入的第一个字的前两个字节（位15-0）被FIFO写入逻辑立即丢弃。

如果需要，SHIFT16位可以为每一帧单独启用/禁用，但只能在帧之间改变。

### 52.6.8.4 接收到的帧被丢弃

因为接收FIFO必须在存储和转发模式下运行（ENETn_RSFL清零），所以可以根据以下错误丢弃接收的帧。

- MAC功能收到的帧有错误。
  - 该帧有一个无效的有效载荷长度
  - 帧长度大于MAX_FL
  - 收到有CRC-32错误的帧
  - 由于接收FIFO溢出，帧被截断
  - 帧被破坏，因为PHY发出了一个错误信号（接收过程中发出RX_ERR信号）。
- 检测到一个IP帧，并且IP头的校验和是错误的。
- 检测到一个具有有效的IP头和有效的IP头校验和的IP帧，协议是已知的，但协议特定的校验和是错误的

如果发生其中一个错误，并且IP加速器功能被配置为丢弃帧（ENETn_RACC），则该帧被自动丢弃。统计数据被正常维护，不受这个丢弃功能的影响。

#### 52.6.8.5 IPv4片段

当收到一个IPv4
IP片段帧时，只检查IP头和验证其校验。32位对齐方式在片段上的操作与在正常IP帧上的操作相同。

IP片段帧的有效载荷没有被检查出任何协议头。因此，协议头只存在于第一个片段中。为了协助特定协议的校验，在IP有效载荷（IP头之后的所有字节）上计算一补和，并与帧状态字一起提供。

帧碎片状态字段（RxBD[FRAG]）被设置为指示碎片接收，IP有效载荷的一补和在RxBD[有效载荷校验]中可用。

**注意事项**

在所有片段被接收并重新组装后，应用软件可以利用与帧的状态字一起交付的有效载荷校验和来计算数据报的特定协议校验和。

例如，如果一个TCP有效载荷由多个IP片段传递，应用软件可以从第一个片段计算出伪头校验值，并将所有片段的有效载荷校验值与状态相加，以验证TCP数据报的校验值。

#### 52.6.8.6 支持IPv6

以下章节描述了对IPv6的支持。

##### 52.6.8.6.1 接收处理

类型为0x86DD的以太网帧标识了一个IP版本6帧（IPv6）帧。如果收到一个IPv6帧，将检查第一个IP头（前十个字），每个IPv6帧中都有这个头。

如果接收SHIFT16功能被启用，IP头将在32位边界上对齐，允许更有效的处理。

对于TCP和UDP数据报，要进行伪头校验计算并进行验证。

为了协助特定协议的校验，一补之和总是在IP有效载荷（IP头之后的所有字节）上计算，并与帧状态字一起提供。例如，如果存在扩展头，它们的总和可以在软件中从校验和中减去，以便在需要时分离出TCP/UDP数据报校验和。

#### 52.6.8.6.2 发送处理

对于IPv6传输，支持SHIFT16功能来处理32位对齐的数据报。

IPv6没有IP头校验；因此，IP校验插入配置被忽略了。

只有当IP头的下一个头是一个已知的协议（TCP、UDP或ICMP）时，才会插入协议校验和。如果检测到一个已知的协议，将计算IP头之后所有字节的校验和，并插入到正确的位置。

对TCP和UDP数据报相应地进行伪头校验计算。

### 52.6.9 复位和停止控制

#### 52.6.9.1 硬件复位

要重置以太网模块，设置ENETn_ECR[RESET]。

#### 52.6.9.2 软复位

当ENETn_ECR[ETHER_EN]在运行过程中被清零时，会发生以下情况。

- uDMA、缓冲区描述符和FIFO控制逻辑被重置，包括缓冲区描述符和FIFO指针。
- 目前正在进行的传输是通过向PHY断言TXER来终止的。

- 当前正在进行的来自应用程序的发送FIFO写入被终止，停止对FIFO的写入，所有来自应用程序的进一步数据被忽略。所有后续的写入都被忽略，直到重新启用。
- 当前正在进行的接收FIFO读取被终止。在这种情况下，RxBD有任意的值。

#### 52.6.9.3 硬件冻结

当处理器进入调试模式并且ECR[DBGEN]被设置时，MAC进入冻结状态，它将优雅地停止所有发送和接收活动。

当MAC进入硬件冻结状态时，会发生以下情况。

- 接收应用接口上当前正在进行的接收交易如常完成。不再从FIFO中读取帧。
- 发送应用接口上当前正在进行的发送事务会像正常一样完成（换句话说，直到写完包（EOP））。
- 目前正在进行的帧接收正常完成，此后不再接受来自MII的帧。
- 当前正在进行的帧传输正常完成，之后不再传输任何帧。

#### 52.6.9.4 优雅的停止

在优雅停止期间，任何当前正在进行的交易都会正常完成，不再接受任何帧。MAC可以从优雅停止中恢复，而不需要复位（例如，不需要清除ETHER_EN）。

以下情况会导致MAC发送或接收数据通路的优雅停止。

##### 52.6.9.4.1 优雅的传输停止（GTS）

当优雅地停止时，MAC不再从发送FIFO中读取帧数据，并且已经完成了任何正在进行的传输。

在以下任何一种情况下，发送数据通路在正在进行的帧传输正常完成后停止。

- ENETn_TCR[GTS]由软件设置。
- ENETn_TCR[TFC_PAUSE]被软件设置，请求暂停帧传输。暂停帧发送完毕后，该状态（和寄存器位）被清零。
- 收到一个暂停帧，使发射机停止工作。当暂停定时器过期或收到一个零量子的暂停帧时，停止的情况就会终止。
- MAC通过软件或处理器进入停止模式而进入睡眠模式（见睡眠模式）。
- MAC处于硬件冻结模式。

当发射机达到其停止状态时，以下事件发生。

- 当过渡到停止时，GRA中断被断言。
- 在硬件冻结模式下，GRA中断不等待应用程序的写入完成，当发送状态机（换句话说，TX FIFO的行侧）达到停止状态时，GRA中断就会发出指令。

#### 52.6.9.4.2 优雅的接收停止（GRS）

当优雅地停止时，MAC不再向接收FIFO写入帧。

在任何正在进行的帧接收正常完成后，如果出现以下任何情况，接收数据通路就会停止。

- MAC被软件置于睡眠模式，或者处理器处于停止模式）。MAC继续接收帧，并在启用的情况下搜索魔法包（见魔法包检测）。然而，没有帧被写入接收FIFO，因此不会被转发给应用程序。
- MAC处于硬件冻结模式。MAC不接受来自MII的任何帧。

当接收数据通路被停止时，会发生以下事件。

- 如果RX处于停止状态，RCR[GRS]被设置为

- 当发射器和接收器被停止时，GRA中断被断言。
- 任何正在进行的对应用程序的接收事务（RX FIFO读取）都会正常进行，直到达到帧的结束包（EOP）。在这之后，会发生以下情况。
  - 当睡眠模式激活时，所有进一步的帧被丢弃，冲刷RX FIFO。
  - 在硬件冻结模式下，没有进一步的帧被传递给应用程序，它们留在接收FIFO中。

**注意事项**

GRS的断言并不等待FIFO应用端正在进行的FIFO读取交易（FIFO读取）。

#### 52.6.9.4.3 优雅停止中断（GRA）。

优雅停止中断（GRA）在下列情况下被引爆。

- 在睡眠模式下，只有在TX和RX数据通路都停止后，中断才会提示。
- 在硬件冻结模式下，只有在TX和RX数据通路都停止后，中断才会被提示。
- MAC发送数据通路因任何其他条件（GTS、TFC_PAUSE、收到暂停）而停止。

GRA中断只在进入停止状态时被触发一次。如果在停止状态持续的情况下清除中断，则不会再触发中断。

## 52.6.10 IEEE 1588功能

为了允许IEEE
1588或类似的时间同步协议的实现，MAC与一个时间戳模块相结合，支持对传入和传出的帧进行精确的时间戳。设置ENETn_ECR[EN1588]以启用1588支持。

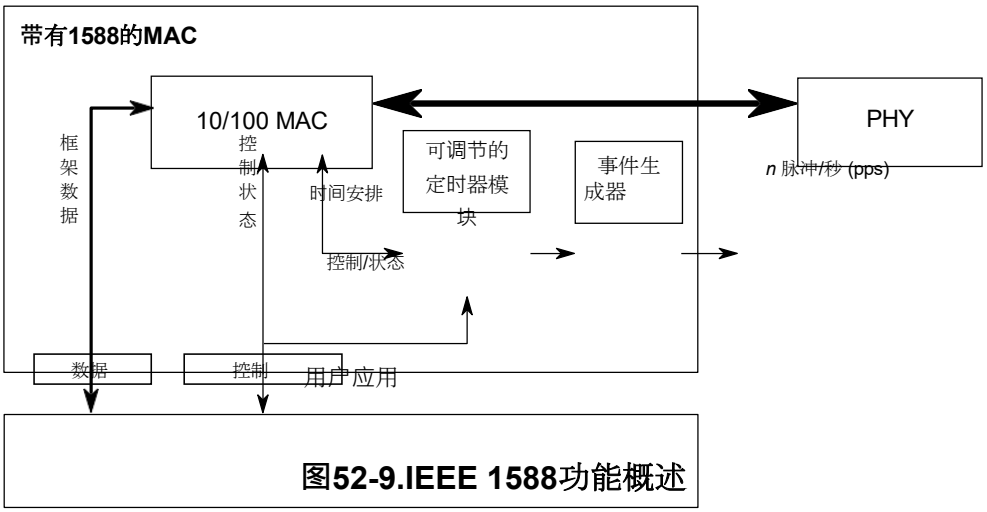



图52-9.IEEE 1588功能概述

### 52.6.10.1 可调节的定时器模块

可调定时器模块（TSM）实现了自由运行计数器（FRC）， 它产生了时间戳。FRC与时间戳时钟一起工作，可以根据你的系统要求设置为任何数值。

通过专用的校正逻辑，可以调整定时器以允许与远程主站同步，并为本地系统提供同步的定时参考。计时器可以被配置为在一个固定的时间段后引起中断，以允许软件计时器的同步或执行其他同步系统的功能。

定时器通常用于实现一秒钟的周期；因此，其数值范围为0至（$1\times10^{9}$）-
1。周期事件可以触发一个中断，软件可以根据需要维持秒和小时的时间值。

#### 52.6.10.1.1 可调节的定时器实施

可调式定时器由一个可编程的计数器/累积器和一个校正计数器组成。这两个计数器的周期和它们的增量率都可以自由配置，允许对定时器进行非常精细的调整。

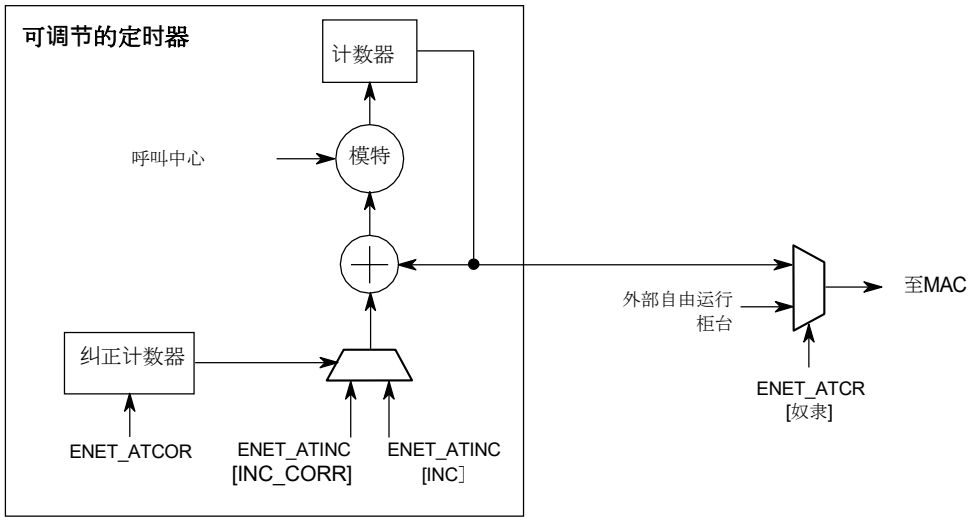



**图52-10.可调式定时器实施细节**

该计数器产生当前时间。在每个时间戳的时钟周期中，按照ENETn_ATINC中的编程，一个常量值被添加到当前时间中。该值取决于所选择的时间戳时钟频率。例如，如果它工作在125MHz，将增量设置为8代表8ns。

在ENETn_ATPER中配置的周期，定义了计数器缠绕时的模数。在一个典型的实现中，周期被设置为$1\times^{109}$，这样，计数器每秒钟都会被包裹，因此，所有的时间戳都代表一秒钟内的绝对纳秒数。当周期达到时，计数器会按照周期的模数重新开始。这意味着它不一定从零开始，而是用值（Current + Inc -(1 × $^{109}$)）加载计数器，假设周期被设定为1 × $^{109}$。

校正计数器完全独立运行，在每个时间戳的时钟周期内增量为1。当它达到ENETn_ATCOR中配置的值时，它重新启动，并指示定时器按校正值增量一次，而不是正常值。

正常增量和校正增量在ENETn_ATINC中配置。要加快定时器的速度，将校正增量设置为大于正常增量值。要减慢定时器的速度，将校正增量设置为小于正常增量值。

修正计数器只定义了修正动作的距离，而不是数量。这允许非常精细的修正和低抖动（在1ns范围内），与所选择的时钟频率无关。

通过启用从属模式（ENETn_ATCR[SLAVE]=1），定时器被忽略，当前时间由外部的一个模块提供。关于使用哪种时钟源，请参见芯片配置的细节。如果有多个模块，这很有用

系统内必须从一个单一的定时器进行操作。当启用从属模式时，仍然必须将ENETn_ATINC[INC]设置为主站的值，因为它被用于内部比较。

### 52.6.10.2 发送时间戳

只有1588事件帧需要在发送时打上时间戳。客户端应用程序（例如，MAC驱动程序）应检测1588事件帧，并将TxBD[TS]与该帧一起设置。

如果 TxBD[TS]被设置，MAC 会在 ENETn_ATSTMP
中记录该帧的时间戳。ENETn_EIR[TS_AVAIL]被设置，表示有新的时间戳可用。

软件通过在传输需要时间戳的帧时设置TxBD[TS]来实现握手程序，然后等待ENETn_EIR[TS_AVAIL]来确定时间戳的可用时间。然后从 ENETn_ATSTMP
读取时间戳。这是对所有事件帧的操作。其他帧不使用
TxBD[TS]，因此不会干扰时间戳的捕获。

### 52.6.10.3 接收时间戳

当收到一个帧时，MAC在检测到该帧的帧开始分隔符（SFD）字段时锁住计时器的值，并在RxBD[1588
timestamp]上提供捕获的时间戳。这是对所有收到的帧进行的。

### 52.6.10.4 时间同步

可调定时器模块可用于将节点的本地时钟与远程主站同步。它实现了一个自由运行的32位计数器，还包含一个额外的校正计数器。

修正计数器增加或减少自由运行计数器的速率，使同步的定时器有非常细微的变化，但在进行修正时只增加非常低的抖动。

在从属情况下，应用软件实现所需的控制算法，设置校正以补偿本地振荡器的漂移，并将定时器锁定在网络上的远程主时钟。

计时器和所有与时间戳有关的信息应被配置为显示一秒钟的真正纳秒值（换句话说，计时器被配置为有一秒钟的周期）。因此，数值范围从0到（$1\times^{109}$）-1。在这个应用中，秒计数器是用一个中断函数在软件中实现的，当纳秒计数器包裹在$1\times^{109}$时，这个中断函数被执行。

### 52.6.10.5 输入捕捉和输出比较

输入捕获输出比较块可以用来为输入和输出事件提供精确的硬件计时。

#### 52.6.10.5.1 输入捕捉

当相应的外部事件发生时，TCCRn捕获寄存器会锁住时间值。事件可以是1588_TMRn 信号中的一个上升沿、下降沿或任一上升沿。一个事件将导致相应的TCSRn[TF]定时器标志被设置，表明一个输入捕获已经发生。如果相应的中断被TCSRn[TIE]字段启用，就可以产生一个中断。

#### 52.6.10.5.2 输出比较

TCCRn比较寄存器中加载了相应事件应该发生的时间。当 ENET自由运行计数器的值与 TCCRn 寄存器中的输出比较参考值相匹配时，相应的标志TCSRn[TF]被设置，表明发生了输出比较。如果TCSRn[TIE]使能，将产生相应的中断。相应的 1588_TMRn 输出信号将根据TCSRn[TMODE]的要求被引出。

#### 52.6.10.5.3 DMA请求

通过设置 TCSRn[TDRE]可以启用一个 DMA请求。当TCSRn[TF]定时器标志被设置时，相应的DMA请求被产生。当DMA完成后，相应的TCSRn[TF]标志被清除。

## 52.6.11 FIFO阈值

核心的FIFO阈值是完全可编程的，可以动态地改变FIFO的操作。

例如，存储和前向传输可以通过简单改变FIFO阈值寄存器来启用。

阈值是以64位字定义的。

接收和发送FIFO都有256个字的深度。

### 52.6.11.1 接收FIFO

有四个可编程的阈值，可以设置为任何数值来控制核心的运作。

**表52-27.接收FIFO阈值定义**

| 注册 | 描述 |
|---|---|
| ENETn_RSFL [rx_section_f ull] 。 | 当FIFO水平达到ENETn_RSFL值时，MAC状态信号被引爆，表示接收FIFO中的数据可用（直通操作）。一旦断言，如果FIFO清空到ENETn_RAEM设定的阈值以下，并且如果帧末尚未存储在FIFO中，则状态信号再次被断言。<br>如果一个帧的大小小于阈值（换句话说，该帧有一个帧末），则该状态也被断言。<br>要在接收路径上启用存储和转发，请清除ENETn_RSFL。只有当接收FIFO中存储了一个完整的帧时，MAC状态信号才会被引出。<br>当对 ENETn_RSFL（直通式操作）进行非零值编程时，它应大于 ENETn_RAEM。 |
| 呼叫中心 [rx_almost_e mpty]。 | 当FIFO水平达到ENETn_RAEM值，并且没有收到帧末，核心接收读取控制就会停止FIFO读取（并随后停止向MAC客户应用传输数据）。<br>如果FIFO中又有比阈值更多的数据或帧末的数据，它就继续传送帧。<br>将ENETn_RAEM设置为至少6个。 |
| 呼叫中心 [rx_almost_f ull]。 | 当FIFO水平接近最大值，并且没有更多的空间可以容纳至少ENETn_RAFL字数时，MAC控制逻辑停止在FIFO中写入数据并截断接收帧以避免FIFO溢出。<br>相应的错误状态在帧被传送到应用程序时被设置。将ENETn_RAFL设置为最小值4。 |
| ENETn_RSEM [rx_section_e mpty] 。 | 当FIFO水平达到ENETn_RSEM值时，一个指示被发送到MAC传输逻辑，它产生一个XOFF暂停帧。这表明FIFO拥堵到了远程以太网客户端。<br>当FIFO水平低于ENETn_RSEM中编程的值时，一个指示被发送到MAC发送逻辑，它产生一个XON暂停帧。这表明FIFO的拥堵被清除给远程以太网客户端。<br>清除ENETn_RSEM后，将禁止生成任何暂停帧。 |

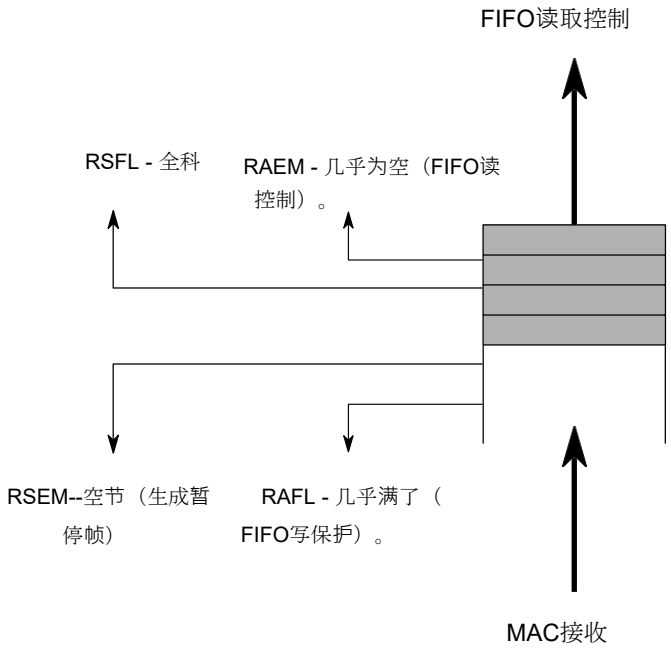



**图52-11.接收FIFO概述**

## 52.6.11.2 发送FIFO

有四个可编程的阈值，可以控制核心的操作。

**表52-28.发送FIFO阈值定义**

| 注册 | 描述 |
|---|---|
| ENETn_TAEM[TX_ALMOST_EMPTY] | 当FIFO水平达到ENETn_TAEM值，并且没有帧末可用时，MAC传输逻辑通过停止FIFO的读取和传输带有MII错误指示的以太网帧来避免FIFO下溢。<br>将ENETn_TAEM设置为至少4。 |
| ENETn_TAFL[TX_ALMOST_FULL] | 当FIFO水平接近最大值，以至于没有更多的空间容纳至少ENETn_TAFL数量的字时，MAC会向应用程序发出控制信号。<br>如果应用没有对这个信号做出反应，FIFO写控制逻辑通过截断当前帧并设置错误状态来避免FIFO溢出。因此，该帧的传输带有MII错误指示。<br>较大的值允许应用程序在帧被截断之前对MAC控制信号的中断作出更多的反应。一个典型的设置是8，它为应用提供了3-4个时钟周期的延迟，以便对MAC控制信号的中断作出反应。 |
| ENETn_TSEM[TX_SECTION_EMPTY] | 当FIFO水平达到ENETn_TSEM值时，一个MAC状态信号被去引，以表明发送FIFO已满。这给了ENET模块一个指示，以减缓或停止其写交易，避免缓冲区溢出。这是对应用的纯指示功能。它在MAC内部没有任何作用。<br>当ENETn_TSEM为0时，该信号永远不会被反送。 |
| 呼叫中心 | 当FIFO水平达到ENETn_TFWR值且STRFWD被清零时，MAC发送控制逻辑在FIFO中可用的帧末之前开始帧传输（cut-through操作）。 |

表52-28.发送FIFO阈值定义

| 注册 | 描述 |
| --- | --- |
| | 如果一个完整的帧的大小小于ENETn_TFWR阈值，MAC也会将该帧传送到线路上。<br>要在发送路径上启用存储和转发，请设置STRFWD。在这种情况下，MAC只有在发送FIFO中存储了一个完整的帧时才开始传输数据。 |

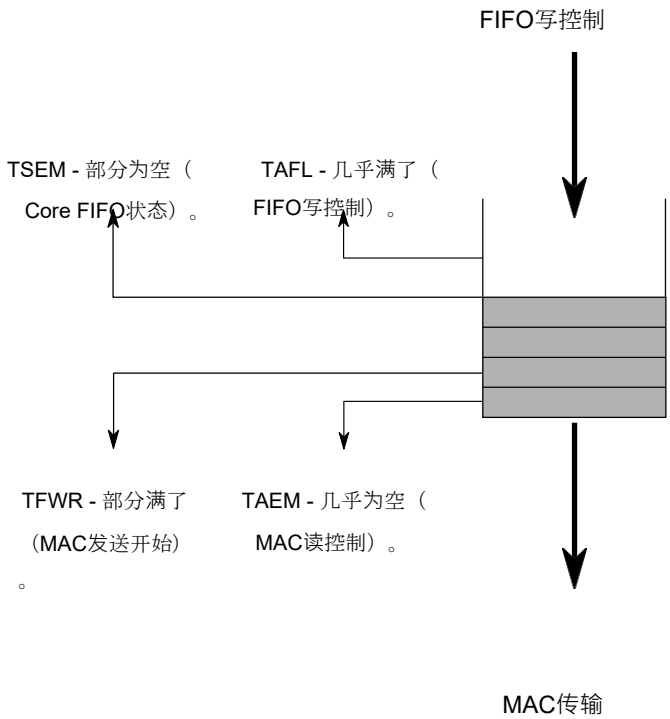



图52-12.发送FIFO概述

## 52.6.12 回路选项

内核实现了外部和内部环回选项，由这里的ENETn_RCR寄存器字段控制。

表52-29.回路选项

| 注册领域 | 描述 |
| --- | --- |
| LOOP | 内部MII环回。MAC发送被返回到MAC接收。没有数据被传输到外部接口。<br>在MII内部环回中，MII_TXCLK和MII_RXCLK必须提供一个时钟信号（10 Mbit/s为2.5 MHz，100 Mbit/s为25 MHz）） |

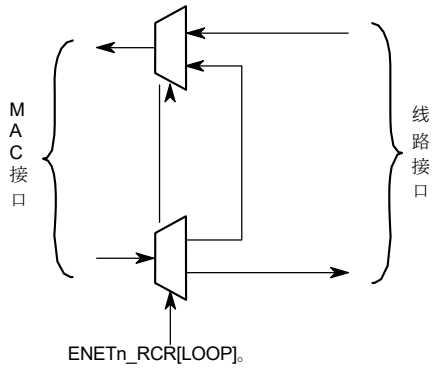



**图52-13.回路选项**

## 52.6.13 遗留的缓冲区描述符

为了支持以前芯片上的以太网控制器，可以使用传统的FEC缓冲区描述符。要启用传统的支持，请将0写入ENETn_ECR[1588EN]。

**注意事项**

- 传统的缓冲区描述符表显示了little-endian芯片的字节顺序。复位后DBSWP必须被设置为1，以启用小-endian模式。

### 52.6.13.1 遗留的接收缓冲区描述符

下表显示了传统的FEC接收缓冲区描述符。表52-33包含每个字段的描述。

**表52-30.遗留FEC接收缓冲区描述符（RxBD）**

| | 字节1 | | | | | | | | 字节0 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 偏移量+0 | 数据长度 | | | | | | | | | | | | | | | |
| 偏移量+2 | E | RO1 | W | RO2 | L | - | - | M | BC | 医学会 | 铃木 | 没有 | - | CR | OV | ÃÃÃ |
| 偏移量+4 | Rx数据缓冲器指针 - 低半字 | | | | | | | | | | | | | | | |
| 偏移量+6 | Rx数据缓冲器指针 - 高半字 | | | | | | | | | | | | | | | |

### 52.6.13.2 传统的传输缓冲区描述符

下表显示了传统的FEC发送缓冲区描述符。表52-35包含每个字段的描述。

**表52-31.遗留FEC发送缓冲区描述符（TxBD）**

| | 字节1 | | | | | | | | 字节0 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 偏移量+0 | 数据长度 | | | | | | | | | | | | | | | |
| 偏移量+2 | R | TO1 | W | TO2 | L | TC | ABC1 | - | - | - | - | - | - | - | - | - |
| 偏移量+4 | Tx数据缓冲器指针 - 低半字 | | | | | | | | | | | | | | | |
| 偏移量+6 | Tx数据缓冲器指针 - 高半字 | | | | | | | | | | | | | | | |

1.uDMA不支持这个字段。

## 52.6.14 增强的缓冲区描述符

本节介绍了通过缓冲区描述符对驱动/uDMA的增强操作。

接下来是对接收和发送描述符字段的详细描述。要启用增强功能，请设置ENETn_ECR[1588EN]。

**注意事项**

增强的缓冲区描述符表显示了little-endian芯片的字节顺序。复位后，DBSWP必须设置为1，以启用小序数模式。

### 52.6.14.1 增强的接收缓冲区描述符

下表显示了增强的uDMA接收缓冲区描述符。表52-33包含了每个字段的描述。

**表52-32.增强型uDMA接收缓冲区描述符（RxBD）**

| | 字节1 | | | | | | | | 字节0 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 偏移量+0 | 数据长度 | | | | | | | | | | | | | | | |
| 偏移量+2 | E | RO1 | W | RO2 | L | - | - | M | BC | 医学会 | 铃木 | 没有 | - | CR | OV | ÃÃÃ |
| 偏移量+4 | Rx数据缓冲器指针 - 低半字 | | | | | | | | | | | | | | | |

| 职能描述 | |
|---|---|
| 偏移量+6 | Rx数据缓冲器指针 - 高半字 |

*表继续在下一页...*

## 表52-32.增强型uDMA接收缓冲区描述符（RxBD）（续）

| | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 偏移量+8<br>偏移 + A | VPCP | | | - | - | - | - | - | - | - | ÃÃÃ | 细菌性阴道炎 | - | VLAN | IPV6 | FRAG |
| 偏移 + C<br>偏移 + E | 医学 | - | - | - | - | 塑胶跑道 | 剑桥大学 | UC | INT | - | - | - | - | - | - | - |
| 偏移 + 10 | 有效载荷检查和 | | | | | | | | | | | | | | | |
| 偏移量+12 | 标头长度 | | | | | - | - | - | 协议类型 | | | | | | | |
| 偏移量+14 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 偏移量+16 | BDU | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 偏移 + 18 | 1588 时间戳 - 低半字 | | | | | | | | | | | | | | | |
| 偏移 + 1A | 1588时间戳 - 高半字 | | | | | | | | | | | | | | | |
| 偏移 + 1C | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 偏移量+1E | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |

## 表52-33.接收缓冲区描述符字段定义

| 词语 | 场地 | 描述 |
|---|---|---|
| 偏移量+0 | 15-0<br>数据长度 | 数据长度。由MAC写入。数据长度是指如果L被清除（该值等于EMRBR），MAC写入该BD的数据缓冲区的八位字节数，如果L被设置，则是包括CRC在内的帧的长度。当BD关闭时，它被MAC写入一次。 |
| 偏移量+2 | 15<br>E | 空的。由MAC（=0）和用户（=1）写入。<br>0 与该BD相关的数据缓冲区已被接收的数据填满，或由于错误情况而中止了数据接收。状态和长度字段已按要求更新。<br>1 与此BD相关的数据缓冲区是空的，或者目前正在接收中。 |
| 偏移量+2 | 14<br>RO1 | 接收软件所有权。这个字段是保留给软件使用的。这个读/写字段不被硬件修改，其值也不影响硬件。 |
| 偏移量+2 | 13<br>W | 包裹。由用户撰写。<br>0 在连续的位置找到下一个缓冲区描述符。<br>1 在ENETn_RDSR中定义的位置找到下一个缓冲区描述符。 |
| 偏移量+2 | 12<br>RO2 | 接收软件所有权。这个字段是保留给软件使用的。这个读/写字段不被硬件修改，其值也不影响硬件。 |
| 偏移量+2 | 11<br>L | 最后一帧。由uDMA编写。<br>0 缓冲区不是一帧中的最后一个。<br>1 缓冲区是一帧中的最后一个。 |
| 偏移量+2 | 10-9 | 保留，必须被清除。 |

| 职能描述 | | |
|---|---|---|
| 偏移量+2 | 8<br>M | 小姐。由MAC写入。该字段由MAC设置，用于在混杂模式下接受的帧，但被内部地址识别标记为错过。因此，在混杂模式下，你可以使用这个字段来快速确定帧是否以该站为目的地。该字段只有在 L 和 PROM 位被设置时才有效。<br>0 帧的接收是由于地址识别命中。<br>1 帧的接收是由于杂交模式。 |

*表继续在下一页...*

表52-33.接收缓冲区描述符字段定义（续）

| 词语 | 场地 | 描述 |
|---|---|---|
| | | 这个字段所需的信息来自 promiscuous_miss(ff_rx_err_stat[26]) 边带信号。 |
| 偏移量+2 | 7<br>BC | 如果DA是广播式的，则设置（FFFF_FFFF_FFFF）。 |
| 偏移量+2 | 6<br>医学会 | 如果DA是多播而不是BC，则设置。 |
| 偏移量+2 | 5<br>铃木 | 接收帧长度违反。由MAC写入。识别出的帧长度大于RCR[MAX_FL]。这个字段只有在L字段被设置时才有效。除非长度超过TRUNC_FL字节，否则接收数据不会以任何方式改变。 |
| 偏移量+2 | 4<br>没有 | 接收非八位数对齐的帧。由MAC写入。收到一个包含不被8整除的比特数的帧，并且在前一个字节边界发生的CRC检查产生了一个错误或发生了PHY错误。这个字段只有在L字段被设置时才有效。如果这个字段被设置，CR字段就不会被设置。 |
| 偏移量+2 | 3 | 保留，必须被清除。 |
| 偏移量+2 | 2<br>CR | 接收CRC或帧错误。由MAC写入。该帧包含一个PHY或CRC错误，长度为一个整数的八位字节。这个字段只有在L字段被设置时才有效。 |
| 偏移量+2 | 1<br>OV | 超限。由MAC写入。在帧接收期间发生了接收FIFO超限。如果该字段被设置，其他状态字段M、LG、NO和CR失去其正常意义，为零。该字段只有在L字段被设置时才有效。 |
| 偏移量+2 | 0<br>ÃÃÃ | 如果接收帧被截断（帧长>TRUNC_FL），则设置。如果TR字段被设置，该帧必须被丢弃，其他错误字段必须被忽略，因为它们可能不正确。 |
| 偏移量+4 | 15-0<br>数据缓冲器指针为低电平 | 接收数据缓冲器指针，低半字 |
| 0ffset + 6 | 15-0<br>数据缓冲器指针为高电平 | 接收数据缓冲器指针，高半字1 |
| 偏移量+8 | 15-13<br>VPCP | VLAN优先级代码点。这个字段由uDMA写入，表示帧的优先级。有效值从0（最佳努力）到7（最高）。该值可用于确定不同类别流量的优先级（如语音、视频、数据）。这个字段只有在L字段被设置后才有效。 |
| 偏移量+8 | 12-6 | 保留，必须被清除。 |
| 偏移量+8 | 5<br>ÃÃÃ | IP头检查和错误。这是一个加速器选项。这个字段是由uDMA写的。当收到一个非IP帧或IP头校验和无效时设置。一个有效载荷小于3字节的IP帧被认为是一个无效的IP帧。这个字段只有在L字段被设置时才有效。 |
| 偏移量+8 | 4<br>细菌性阴道炎 | 协议检查和错误。这是一个加速器选项。该字段由uDMA写入。当协议的校验和无效或发现未知协议且无法进行校验时设置。这个字段只有在L字段被设置时才有效。 |
| 偏移量+8 | 3 | 保留，必须被清除。 |

| | | |
|---|---|---|
| 偏移量+8 | 2<br>VLAN | VLAN。这是一个加速器选项。这个字段是由uDMA写入的。它意味着该帧有一个VLAN标签。这个字段只有在L字段被设置时才有效。 |

*表继续在下一页...*

表52-33.接收缓冲区描述符字段定义（续）

| 词语 | 场地 | 描述 |
|---|---|---|
| 偏移量+8 | 1<br>IPV6 | IPV6帧。这个字段是由uDMA写入的。这个字段表示该帧有一个IPv6帧类型。如果这个字段没有设置，意味着收到的是一个IPv4或其他协议的帧。这个字段只有在L字段被设置时才有效。 |
| 偏移量+8 | 0<br>FRAG | IPv4片段。这是一个加速器选项。这个字段由uDMA写入。它表明该帧是一个IPv4片段帧。这个字段只有在L字段被设置时才有效。 |
| 偏移量+A | 15<br>医学 | MAC错误。这个字段是由uDMA写的。这个字段意味着存储在系统内存中的帧被接收时出现了错误（通常是接收FIFO溢出）。这个字段只有在L字段被设置时才有效。 |
| 偏移量+A | 14-11 | 保留，必须被清除。 |
| 偏移量+A | 10<br>塑胶跑道 | PHY错误。该字段由uDMA写入。当在PHY接口上收到带有错误字符的帧时，设置为 "1"。该帧是无效的。该字段只有在L字段被设置时才有效。 |
| 偏移量+A | 9<br>剑桥大学 | 碰撞。该字段由uDMA写入。当收到的帧在接收过程中检测到碰撞时设置。该帧无效并被发送到用户应用程序。这个字段只有在L字段被设置时才有效。 |
| 偏移量+A | 8<br>UC | 单播。该字段由uDMA写入，表示该帧为单播。无论L字段是否被设置，该字段都是有效的。 |
| 偏移量+A | 7<br>INT | 产生RXB/RXF中断。这个字段由用户设置，表示uDMA要在*dma_int_rxb* / *dma_int_rxfevent*上产生一个中断。 |
| 偏移量+A | 6-0 | 保留，必须被清除。 |
| 偏移量+C | 15-0<br>有效载荷检查和 | 互联网有效载荷检查和。这是一个加速器选项。它是IP帧的有效载荷部分的一补之和。这个和是在IP头之后的所有数据上计算的，直到IP有效载荷的结束。这个字段只有在L字段被设置时才有效。 |
| 偏移量+E | 15-11<br>标头长度 | 头部长度。这是一个加速器选项。这个字段是由uDMA写入的。这个字段是在IP和它下面的协议头中发现的32位字的总和。如果发现一个具有未知协议的IP数据报，那么这个值就是IP头的长度。如果没有发现IP帧或错误的IP头，则该值为0。 如果在各自的头中没有头选项存在，则以下值为最小值。<br>• ICMP/IP：6（5个IP头，1个ICMP头）。<br>• UDP/IP：7（5个IP头，2个UDP头）。<br>• TCP/IP：10（5个IP头，5个TCP头）。<br>这个字段只有在L字段被设置后才有效。 |
| 偏移量+E | 10-8 | 保留，必须被清除。 |
| 偏移量+E | 7-0<br>协议类型 | 协议类型。这是一个加速器选项。在帧的IP头内发现的8位协议字段。只有当ICE被清除时，它才有效。这个字段只有在L字段被设置时才有效。 |
| 偏移量+10 | 15-0 | 保留，必须被清除。 |
| 偏移量+12 | 15<br>BDU | 最后一个缓冲区描述符更新完成。表示最后一个BD数据已经被uDMA更新。该字段由用户（=0）和uDMA（=1）写入。 |
| 偏移量+12 | 14-0 | 保留，必须被清除。 |

| | | |
|---|---|---|
| 偏移量+14 | 15-0 | 这个值是由uDMA写入的。它只有在L字段被设置时才有效。 |
| 偏移量+16 | | |

*表继续在下一页...*

表52-33.接收缓冲区描述符字段定义（续）

| 词语 | 场地 | 描述 |
|---|---|---|
| | 1588<br>时间戳 | |
| 偏移量+18<br>-<br>偏移量+1E | 15-0 | 保留，必须被清除。 |

1.含有相关数据缓冲区地址的接收缓冲区指针，必须总是被16均匀地除以。该缓冲区必须位于MAC外部的存储器中。以太网控制器从不修改这个值。

## 52.6.14.2 增强的传输缓冲区描述符

表52-34.增强型发送缓冲区描述符（TxBD）

| | 字节1 | | | | | | | | 字节0 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 偏移量+0 | 数据长度 | | | | | | | | | | | | | | | |
| 偏移量+2 | R | TO1 | W | TO2 | L | TC | - | - | - | - | - | - | - | - | - | - |
| 偏移量+4 | Tx数据缓冲器指针 - 低半字 | | | | | | | | | | | | | | | |
| 偏移量+6 | Tx数据缓冲器指针 - 高半字 | | | | | | | | | | | | | | | |
| 偏移量+8 | TXE | - | UE | EE | 基金会 | LCE | OE | 东交所 | - | - | - | - | - | - | - | - |
| 偏移量+A | - | INT | TS | PINS | IINS | - | - | - | - | - | - | - | - | - | - | - |
| 偏移量+C | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 偏移量+E | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 偏移量+10 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 偏移量+12 | BDU | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 偏移量+14 | 1588 时间戳 - 低半字 | | | | | | | | | | | | | | | |
| 偏移量+16 | 1588时间戳 - 高半字 | | | | | | | | | | | | | | | |
| 偏移量+18 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 偏移量+1A | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 偏移量 + 1C | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| 偏移量+1E | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |

表52-35.增强型发送缓冲区描述符字段定义

| 词语 | 场地 | 描述 |
|---|---|---|
| 偏移量+0 | 15-0<br>数据长度 | 数据长度，由用户写入。<br>数据长度是MAC应该从这个BD的数据缓冲区传输的八位数。它从不被MAC修改。 |

| | | |
|---|---|---|
| 偏移量+2 | 15<br>R | 准备好了。由MAC和你撰写。 |

*表继续在下一页...*

表52-35 增强型发送缓冲区描述符字段定义（续）

表52-33 接收缓冲区描述符字段定义（续）

| 词语 | 场地 | 描述 |
| --- | --- | --- |
| | | 0 与该BD相关的数据缓冲区还没有准备好进行传输。你可以自由操作这个BD或其相关的数据缓冲区。MAC在缓冲区被传输后或遇到错误情况后，会清除这个字段。<br>1 由你准备传输的数据缓冲区，还没有被传输或目前正在传输。在这个字段被设置后，你可以不写这个BD的字段。 |
| 偏移量+2 | 14<br>TO1 | 传递软件所有权。这个字段是保留给软件使用的。这个读/写字段不被硬件修改，其值不影响硬件。 |
| 偏移量+2 | 13<br>W | 包裹。由用户撰写。<br>0 在连续的位置找到下一个缓冲区描述符<br>1 在ETDSR中定义的位置找到下一个缓冲区描述符。 |
| 偏移量+2 | 12<br>TO2 | 传递软件所有权。该字段保留给软件使用。这个读/写字段不被硬件修改，其值不影响硬件。 |
| 偏移量+2 | 11<br>L | 最后一次入镜。由用户撰写。<br>0 缓冲区不是传输帧中的最后一个<br>1 缓冲区是传输帧中的最后一个。 |
| 偏移量+2 | 10<br>TC | 发送CRC。由用户写入，仅当L被设置时有效。<br>0 在最后一个数据字节后立即结束传输<br>1 在最后一个数据字节后传输CRC序列 该字段仅在L字段被设置时有效。 |
| 偏移量+2 | 9<br>美国广播公司 | 追加坏的CRC。<br>**注意：**这个字段不被uDMA支持，会被忽略。 |
| 偏移量+2 | 8-0 | 保留，必须被清除。 |
| 偏移量+4 | 15-0<br>数据缓冲器指针为低电平 | Tx数据缓冲器指针，低半字 |
| 偏移量+6 | 15-0<br>数据缓冲器指针为高电平 | Tx数据缓冲器指针，高半字。发送缓冲区指针，包含相关数据缓冲区的地址，必须总是被8均匀整除。 缓冲区必须位于MAC外部的存储器中。这个值永远不会被以太网控制器修改。 |
| 偏移量+8 | 15<br>TXE | 发生传输错误。这个字段是由uDMA写的。这个字段表示有一个与帧一起报告的某种传输错误。实际上，这个字段是其他错误字段的OR，包括UE、EE、FE、LCE、OE和TSE。这个字段只有在L字段被设置时才有效。 |
| 偏移量+8 | 14 | 保留，必须被清除。 |
| 偏移量+8 | 13<br>UE | 下溢错误。这个字段是由uDMA写的。该字段表示MAC在发送时报告了一个下溢错误。这个字段只有在L字段被设置时才有效。 |

| | | |
|---|---|---|
| 偏移量+8 | 12<br>EE | 超额碰撞错误。这个字段是由uDMA写的。这个字段表示MAC在传输中报告了一个多余的碰撞错误。这个字段只有在L字段被设置时才有效。 |

*表继续在下一页...*

表52-35.增强型发送缓冲区描述符字段定义（续）

| 词语 | 场地 | 描述 |
|---|---|---|
| 偏移量+8 | 11<br>基金会 | 有错误的帧。该字段由uDMA写入。该字段表示MAC报告说uDMA在提供数据包时报告了一个错误。这个字段只有在L字段被设置时才有效。 |
| 偏移量+8 | 10<br>LCE | 迟来的碰撞错误。这个字段是由uDMA写的。这个字段表示MAC报告说在发送时有一个Late Collision。这个字段只有在L字段被设置时才有效。 |
| 偏移量+8 | 9<br>OE | 溢出错误。这个字段是由uDMA写的。这个字段表示MAC报告说在发送时有一个FIFO溢出条件。这个字段只有在L字段被设置时才有效。 |
| 偏移量+8 | 8<br>东交所 | 时间戳错误。这个字段是由uDMA写的。该字段表示MAC报告的帧类型与时间戳帧不同。这个字段只有在L字段被设置时才有效。 |
| 偏移量+8 | 7-0 | 保留，必须被清除。 |
| 偏移量+A | 15 | 保留，必须被清除。 |
| 偏移量+A | 14<br>INT | 产生中断标志。这个字段是由用户写的。无论L字段如何，该字段都是有效的，对于一个给定帧的所有EBD必须是相同的。uDMA不更新这个值。 |
| 偏移量+A | 13<br>TS | 时间戳。这个字段是由用户写的。这表明uDMA要生成一个时间戳帧给MAC。这个字段是有效的，与L字段无关，对给定帧的所有EBD必须是相同的。uDMA不更新此值。 |
| 偏移量+A | 12<br>PINS | 插入特定协议的校验和。该字段由用户写入。如果设置，MAC的IP加速器会计算协议校验和，并用计算值覆盖相应的校验和字段。校验和字段必须由生成帧的应用程序清除。uDMA不更新这个值。无论L字段如何，该字段都是有效的，对于一个给定帧的所有EBD必须是相同的。 |
| 偏移量+A | 11<br>IINS | 插入IP头检查和。这个字段由用户写入。如果设置，MAC的IP加速器会计算IP头校验和，并用计算值覆盖相应的头字段。检查和字段必须由生成帧的应用程序清除。uDMA不更新这个值。这个字段是有效的，与L字段无关，对一个给定帧的所有EBD必须是相同的。 |
| 偏移量+A | 10-0 | 保留，必须被清除。 |
| 偏移量+C | 15-0 | 保留，必须被清除。 |
| 偏移量+E | 15-0 | 保留，必须被清除。 |
| 偏移量+10 | 15-0 | 保留，必须被清除。 |
| 偏移量+12 | 15<br>BDU | 最后一个缓冲区描述符更新完成。表示最后一个BD数据已经被uDMA更新。该字段由用户（=0）和uDMA（=1）写入。 |
| 偏移量+12 | 14-0 | 保留，必须被清除。 |
| 偏移量+14 | 15-0<br>1588年的时间戳 | 这个值是由uDMA写入的。它只有在L字段被设置时才有效。 |
| 偏移量+16 | | |
| 偏移量+18-偏移量+1E | 15-0 | 保留，必须被清除。 |

## 52.6.15 客户端FIFO应用接口

FIFO接口与以太网线路完全异步，发送和接收接口可以在不同的时钟速率下运行。

所有向/从用户应用程序的传输都是独立于核心操作的，核心提供了一个基于双信号握手的简单接口给用户应用程序。

### 52.6.15.1 数据结构描述

必须遵守下表中为FIFO接口定义的数据结构，以确保在以太网线路上正确传输数据。字节0首先被发送至线路并从线路中接收。

**表52-36.FIFO接口数据结构**

| | 63 56 | 55 48 | 47 40 | 39 32 | 31 24 | 23 16 | 15 8 | 7 0 |
|---|---|---|---|---|---|---|---|---|
| 字0 | 字节7 | 字节6 | 字节5 | 字节4 | 字节3 | 字节2 | 字节1 | 字节0 |
| 字1 | 字节15 | 字节14 | 字节13 | 字节12 | 字节11 | 字节10 | 字节9 | 字节8 |
| ... | ... | | | | | | | |

FIFO接口上的帧的大小可能不是64位的模数。

用户应用程序可能不关心以太网帧格式的全部细节。它需要提供和接收一个具有以下结构的以太网帧。

- 以太网MAC目标地址
- 以太网MAC源地址
- 可选的802.1q VLAN标签（VLAN类型和信息栏）。
- 以太网长度/类型字段
- 有效载荷

FIFO接口上的帧不包含前导字段和SFD字段，它们分别由MAC在发送和接收时插入和丢弃。

- 在接收时，CRC和帧填充可以被剥离或透明地通过。
- 在传输时，填充和CRC可由用户应用提供，或由MAC为每一帧独立地自动添加。没有大小限制。

### 注意事项

在传输时，如果
ENETn_TCR[ADDINS]被设置，每个帧的第 6-11
字节可以被设置为任何值，因为 MAC 会用 ENETn_PAUR
和 ENETn_PALR 寄存器中编程的 MAC
地址覆盖这些字节。

**表52-37.FIFO接口帧格式**

| 字节号 | 场地 |
|---|---|
| 0-5 | 目的地MAC地址 |
| 6-11 | 源MAC地址 |
| 12-13 | 长度/类型字段 |
| 14-N | 有效载荷数据 |

发送和接收时都支持VLAN标记的帧，并实现附加信息（VLAN类型和信息）。

**表52-38.FIFO接口VLAN帧格式**

| 字节号 | 场地 |
|---|---|
| 0-5 | 目的地MAC地址 |
| 6-11 | 源MAC地址 |
| 12-15 | VLAN标签和信息 |
| 16-17 | 长度/类型字段 |
| 18-N | 有效载荷数据 |

### 注意事项

该标准规定，MAC地址的LSB首先被发送/接收，而对于所
有其他头字段--
换句话说，长度/类型、VLAN标签、VLAN信息和暂停量
- MSB首先被发送/接收。

## 52.6.15.2 数据结构实例

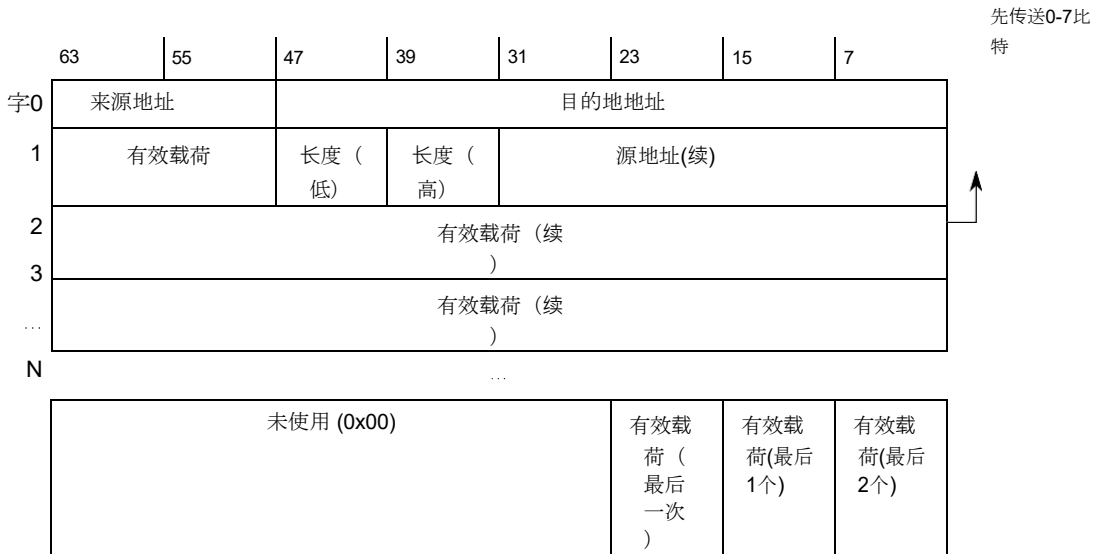



**图52-14.普通以太网帧64位映射示例**




**图52-15.VLAN标记的帧的64位映射示例**

如果启用了CRC转发（CRCFWD=0），帧的最后四个有效字节包含FCS字段。最后一个字的非显著字节可以有任何值。

## 52.6.15.3 框架状态

在接收缓冲区描述符中有一个MAC层状态字和一个加速器状态字。

每一帧的最后一个数据的状态都可以得到。

如果帧状态包含MAC层错误（例如，CRC或长度错误），RxBD[ME]也会被设置为帧的最后一个数据。

## 52.6.16 FIFO保护

### 52.6.16.1 发送FIFO下溢

在帧传输过程中，当发送FIFO达到几乎为空的阈值，且FIFO中没有存储帧结束的指示时，MAC逻辑。

- 停止从FIFO读取数据
- 发出MII错误信号（MII_TXER）（图52-16中为1），表示已经传输的片段无效。
- 断绝MII传输使能信号（MII_TXEN），终止帧传输（2）。

在下溢之后，当应用程序完成帧传输时（3），MAC发送逻辑会丢弃FIFO中的任何新数据，直到达到数据包的终点（4），并设置增强的TxBD[UE]字段。

当应用程序发送一个带有帧开始指示（5）的新帧时，MAC开始在MII接口上传输数据。

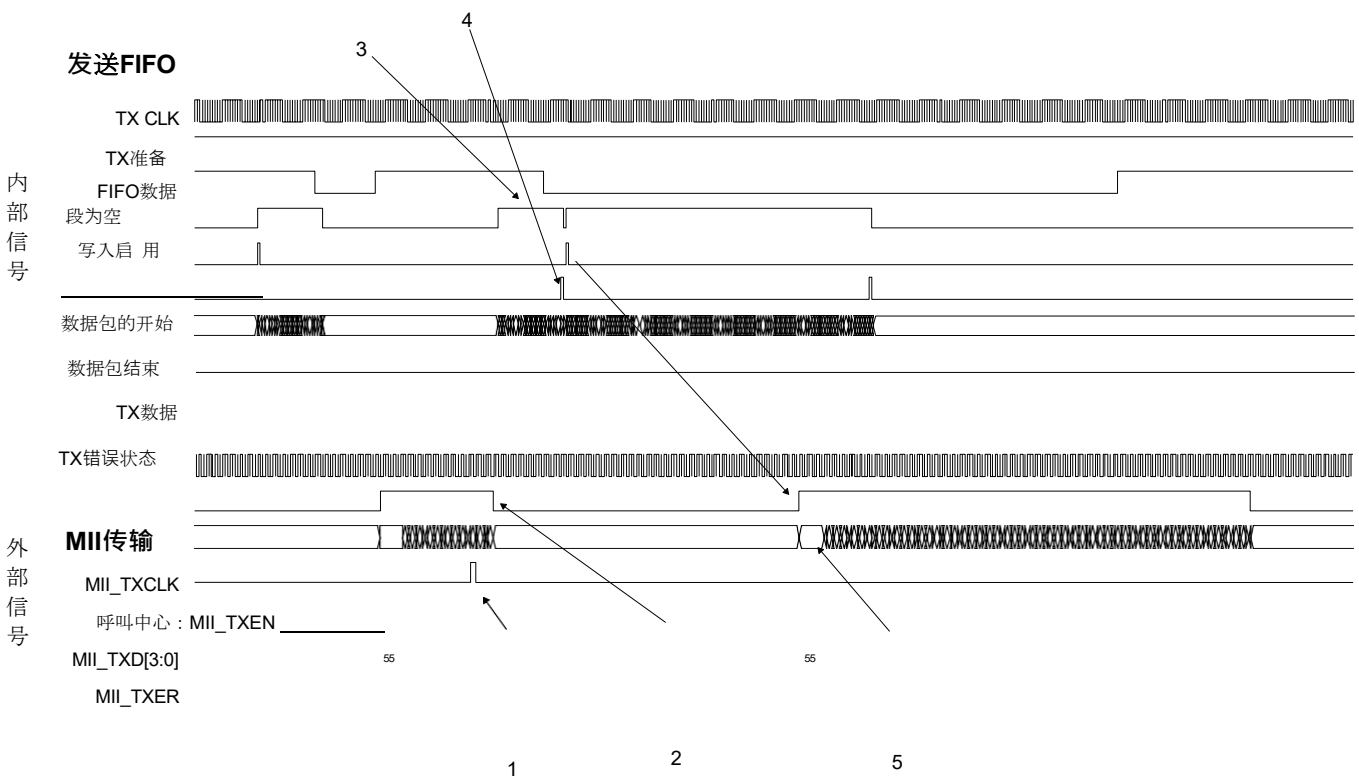



**图52-16.发送FIFO下溢保护**

### 52.6.16.2 发送FIFO溢出

在发送路径上，当FIFO达到可编程的几乎满的阈值时，内部MAC ready信号被去掉。应用程序应停止发送新数据。

然而，如果应用程序继续发送数据，发送FIFO就会溢出，破坏先前存储的内容。核心逻辑为下一个传输的帧设置增强的TxBD[OE]字段，以指示这种溢出的发生。

**注意事项**

溢出是一个致命的错误，必须通过复位核心或清除ENETn_ECR[ETHER_EN]来解决，以清除FIFO并准备再次进行正常操作。

### 52.6.16.3 接收FIFO溢出

在帧接收过程中，如果客户应用不能接收数据（1），MAC接收控制在FIFO达到可编程的几乎满的阈值时截断传入的帧，以避免溢出。

该帧随后在FIFO接口上接收，并有错误指示（增强的RxBD[ME]字段与接收结束包一起设置）(2)，并设置截断错误状态字段(3)。

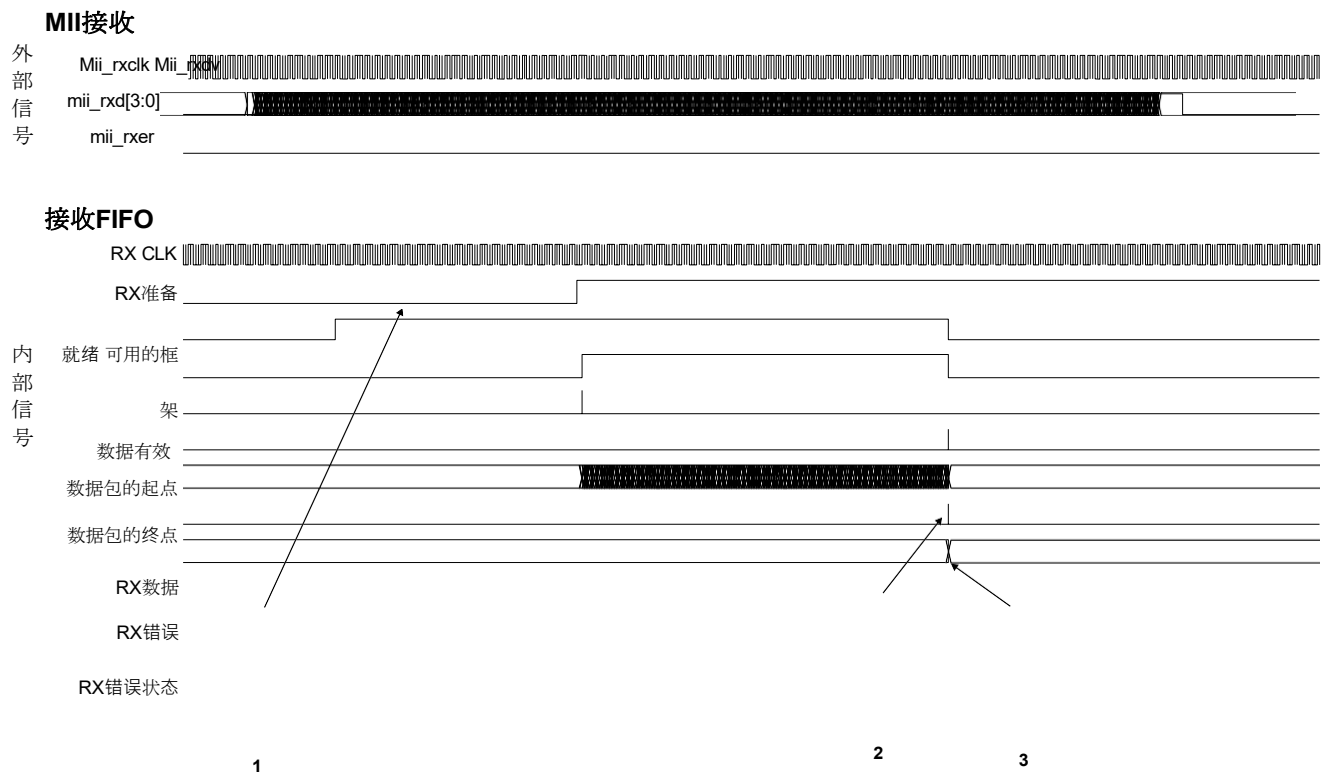



**图52-17.接收FIFO溢出保护**

## 52.6.17 参考时钟

ENET模块的输入时钟必须符合下表中的规格。

| 以太网速度模式 | 以太网总线时钟 | 最低ENET系统时钟 |
|---|---|---|
| 10 Mbit/s | 2.5 MHz | 5兆赫 |
| 100 Mbit/s | 25.0 MHz | 50兆赫 |

## 52.6.18 PHY管理接口

MDIO接口是一个两线制的管理接口。MDIO管理接口实现了一种标准化的方法来访问PHY设备管理寄存器。

该核心实现了一个主MDIO接口，可以连接到多达32个PHY设备。

### 52.6.18.1 MDIO第22条帧格式

核心MDIO主控制器使用下表中定义的帧与从机（PHY设备）进行通信。

一个完整的帧的长度为64位，由可选的32位序言、14位命令、2位总线方向改变和16位数据组成。每个比特都是在MDIO时钟（MDC信号）的上升沿传输的。MDIO数据信号在帧之间是三态的。

核心PHY管理接口支持标准MDIO规范（IEEE 802.3第22条）。

**表52-39.MDIO第22条帧结构**

| ĂĂĂ | 职务 | PHYADR | ĂĂĂ | 淘宝网 | 数据 |
|---|---|---|---|---|---|

**表52-40.MDIO帧字段描述**

| 场地 | 描述 |
|---|---|
| ĂĂĂ<br>(2比特) | 开始指示字段，用ENETn_MMFR[ST]编程，对于标准MDIO（条款22）等于01。 |
| 职务<br>(2比特) | 操作码定义了操作的类型。用ENETn_MMFR[OP]编程。01 写操作<br>10 读取操作 |
| PHYADR<br>(5位) | 五位PHY设备地址，用ENETn_MMFR[PA]进行编程。最多可以寻址32个设备。 |

*表继续在下一页...*

表52-40.MDIO帧字段描述（续）

| 场地 | 描述 |
| --- | --- |
| ÃÃÃ<br>(5位) | 五位寄存器地址，用ENETn_MMFR[RA]进行编程。每个PHY最多可以实现32个寄存器。 |
| 淘宝网<br>(2比特) | 周转时间，用ENETn_MMFR[TA]编程。为读操作保留两个比特时间，以便将数据总线从写切换到读。PHY设备在数据阶段呈现其寄存器内容，并从周转阶段的第二个位驱动总线。 |
| 数据<br>(16位) | 数据，由ENETn_MMFR[DATA]设置。写入或读出PHY |

## 52.6.18.2 MDIO第45条帧格式

IEEE
802.3第45条中定义的扩展MDIO帧结构引入了间接寻址。首先，对地址寄存器进行写操作，然后是写或读操作，分别将16位数据放入寄存器或检索寄存器的内容。在每个交易之前，都要发送一个32位的逻辑1的前奏。MDIO数据信号在帧之间是三态的。

扩展的MDIO定义了四个事务，由两位操作码字段决定。

表52-41.MDIO条款45帧结构

| ÃÃÃ | 职务 | PRTAD | 残疾人 | 淘宝网 | 地址/数据 |
| --- | --- | --- | --- | --- | --- |

所有比特从左到右传输（序言比特在前），所有字段的最重要比特都先发送（上表中最左边）。完整的帧有64位的长度（32位前言，14位命令，2位总线方向改变，16位数据）。
每个位都是随着MDIO时钟（MDC）的上升沿传输的。这些字段

和事务在下表中进行了总结。

表52-42.MDIO条款45帧字段说明

| 场地 | 描述 |
| --- | --- |
| ÃÃÃ | 开始指示。表示序言的结束和帧的开始。对于扩展的MDIO（条款45）帧，该值为00。 |

职能描述

| | |
|---|---|
| 职务 | 操作码定义了是否进行读或写操作，并通过ENETn_MMFR[OP]进行编程。更多信息见表52-43。<br>00 地址写入<br>01 写操作<br>10 读取包括操作 |

*表继续在下一页...*

表52-42.MDIO条款45帧字段描述（续）

| 场地 | 说明 |
|---|---|
| | 11 读取操作 |
| PRTAD | 端口地址指定的是一个MDIO端口。每个端口最多可以有32个设备，每个设备可以有一套独立的寄存器。 |
| 残疾人 | 设备地址。最多可以寻址32个设备（在一个端口内）。 |
| 淘宝网 | 周转时间，用ENETn_MMFR[TA]编程。为读操作保留两个比特时间，以便将数据总线从写切换到读。PHY设备在数据阶段呈现其寄存器内容，并从周转阶段的第二个位驱动总线。 |
| 地址/数据 | 16位地址（用于地址写入）或数据，由ENETn_MMFR[DATA]设置，写入PHY或从PHY读取。 |

表52-43.MDIO条款45交易

| 交易类型 | 描述 |
|---|---|
| 地址 | 对设备/端口的内部地址寄存器的写交易。帧的数据部分包含要存储在设备内部地址 "指针 "寄存器中的值，用于进一步的交易。 |
| 撰写 | 向寄存器写入数据。16位数据将被写入由设备内部地址确定的寄存器中。 |
| 阅读 | 从设备内部地址识别的寄存器中读取数据。 |
| 阅读完毕。 | 读取地址后增量。读取由设备内部地址识别的寄存器。在这之后，设备内部的地址被递增。如果地址寄存器都是'1'(0xFFFF)，则不进行递增（即递增不环绕）。 |

### 52.6.18.3 MDIO时钟生成

MDC时钟是由内部总线时钟（即IPS总线时钟）除以ENETn_MSCR[MII_SPEED]中的编程值产生。

### 52.6.18.4 MDIO操作

为了执行MDIO访问，根据MII管理帧寄存器（ENETn_MMFR）中提供的描述，设置MDIO命令寄存器（ENETn_MMFR）。

要检查编程访问何时完成，请读取ENETn_EIR[MII]字段。

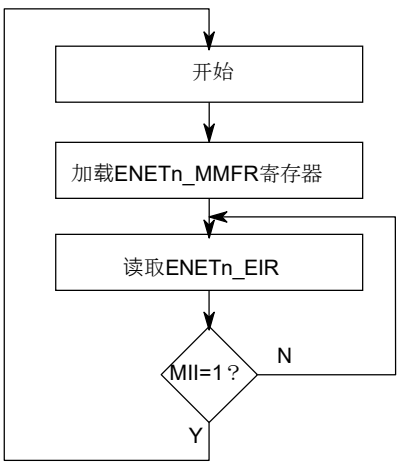



**图52-18.MDIO访问概述**

## 52.6.19 以太网接口

实现了以下以太网接口。

- 快速以太网MII（媒体独立接口）
- 使用接口转换器/垫圈的RMII 10/100

下表显示了如何配置ENET寄存器来选择每个接口。

| 模式 | RCR[RMII_10T] | RCR[RMII_MODE] |
|---|---|---|
| MII - 10 Mbit/s | - | 0 |
| MII - 100 Mbit/s | - | 0 |
| RMII - 10 Mbit/s | 1 | 1 |
| RMII - 100 Mbit/s | 0 | 1 |

### 52.6.19.1 RMII接口

在RMII接收模式下，对于CRS_DV断言后的正常接收，RXD[1:0]为00，直到接收器确定接收事件有一个适当的流开始分隔符（SSD）。

序言出现（RXD[1:0]=01），MAC在检测到SFD后开始抓取数据。

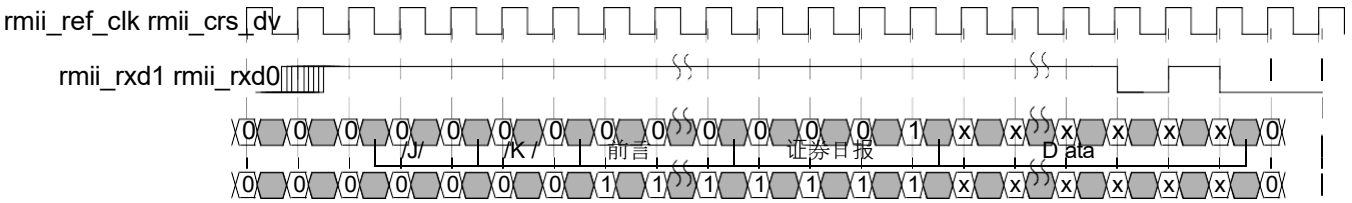



**图52-19.RMII接收操作**

如果检测到一个错误的载波（坏的SSD），那么RXD[1:0]为10，直到接收事件的结束。这是一个独特的模式，因为假载波只可能发生在一个数据包的开头，而这个数据包的序言被解码（RXD[1:0] = 01）。

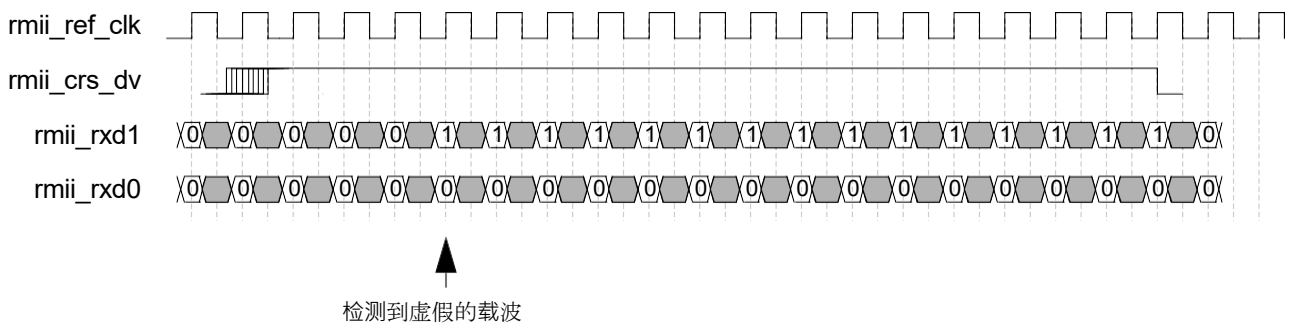



**图52-20.带有假载波的RMII接收操作**

在RMII传输模式下，TXD[1:0]为每个REF_CLK周期提供有效数据，同时TXEN被断言。

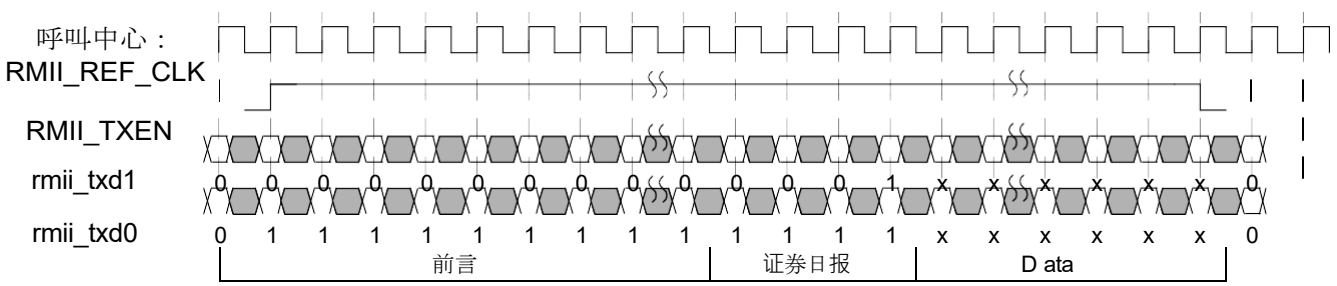



**图52-21.RMII发送操作**

### 52.6.19.2 MII接口 - 发送

在发送时，所有的数据传输都与MII_TXCLK的上升沿同步。MII数据使能信号MII_TXEN被断言以指示新帧的开始，并保持断言，直到MII_TXD[3:0]总线上出现该帧的最后一个字节。

在各帧之间，MII_TXEN保持去干扰状态。

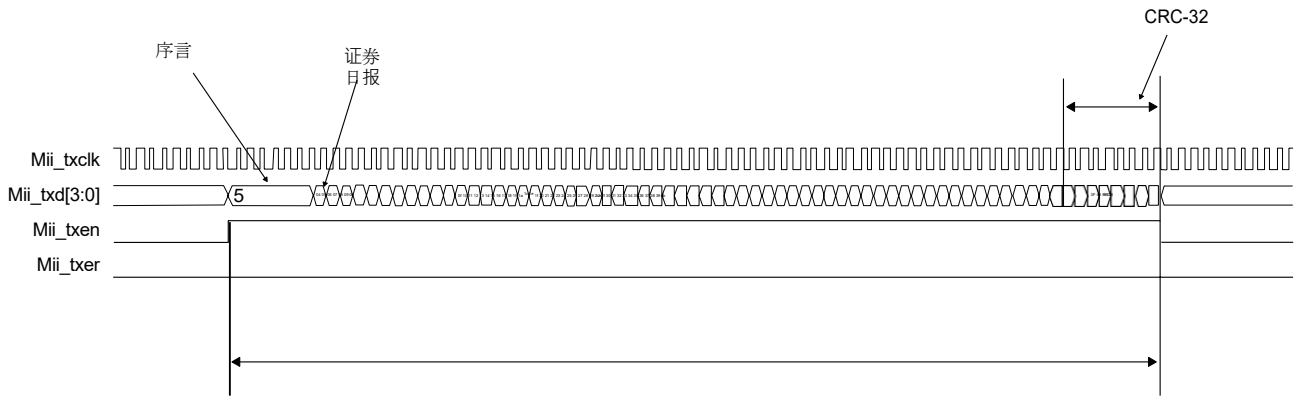



**图52-22.MII发送操作**

如果在FIFO接口上收到的帧有错误（例如，RxBD[ME]设置），该帧随后在数据包传输过程中的任何时候都会以MII_TXER错误信号传输一个时钟周期。

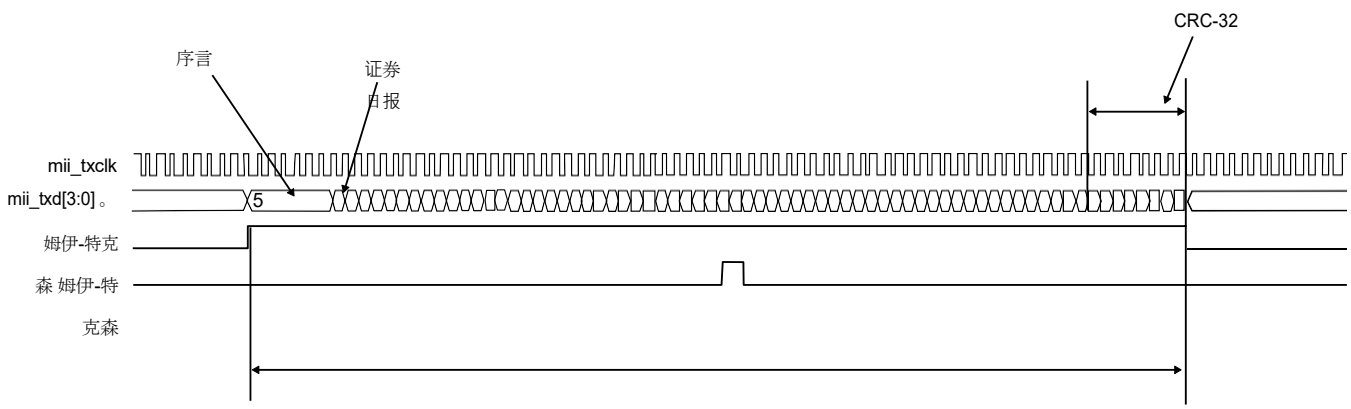



**图52-23.MII发送操作 - 误码帧**

### 52.6.19.2.1 带碰撞的传输 - 半双工

当在帧传输过程中检测到碰撞时（MII_COL断言），MAC停止当前的传输，发送一个32位干扰模式，并重新传输当前的帧。

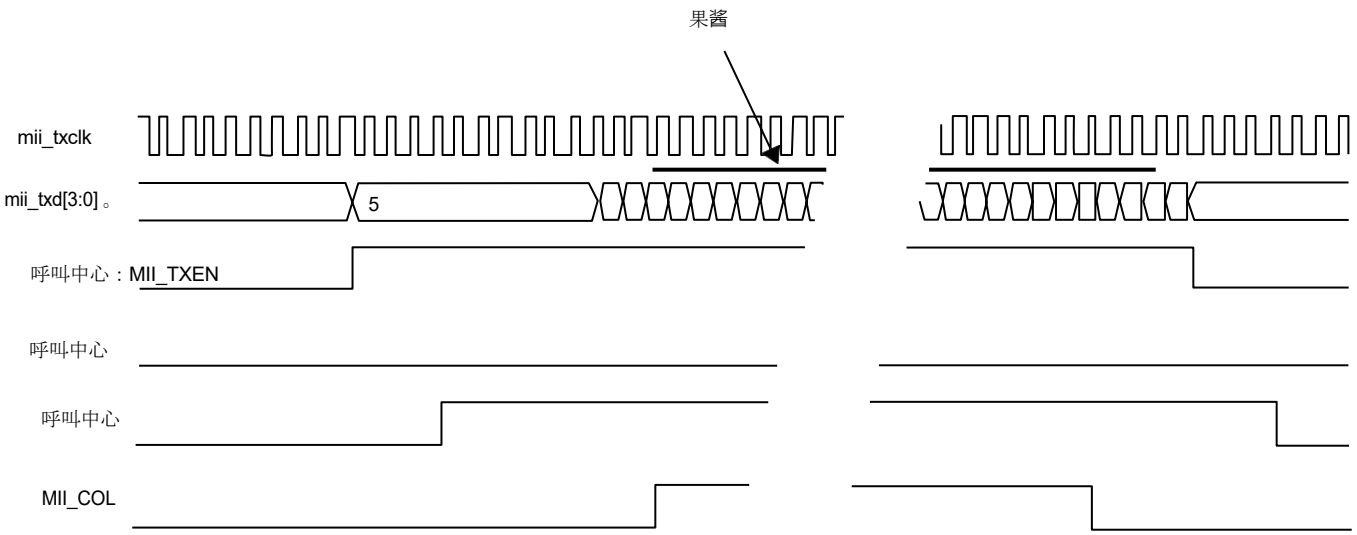



**图52-24.MII传输操作--带碰撞的传输**

### 52.6.19.3 MII接口 - 接收

在接收时，所有的信号都在MII_RXCLK的上升沿被采样。MII数据使能信号，MII_RXDV，由PHY发出，表示一个新帧的开始，并保持发出，直到MII_RXD[3:0]总线上出现该帧的最后一个字节。

在帧之间，MII_RXDV保持脱引状态。

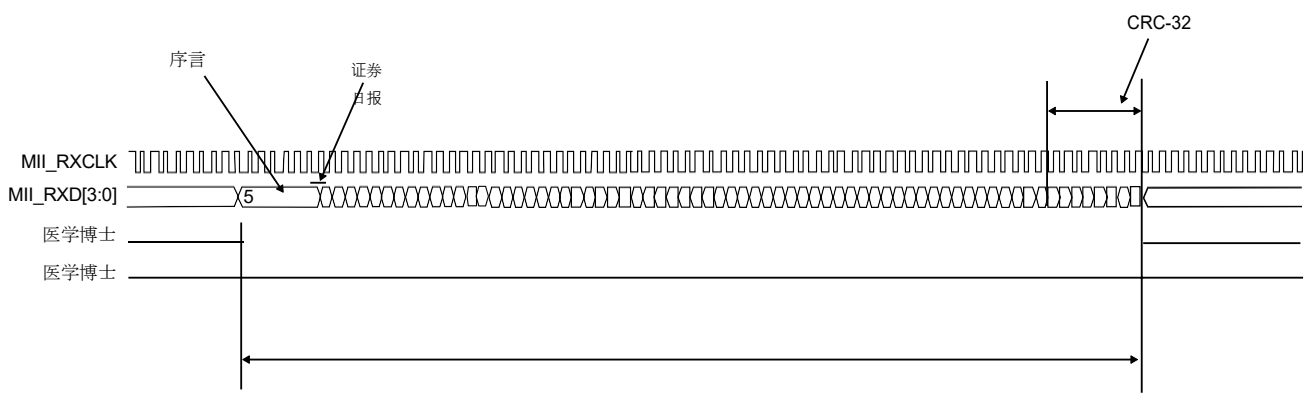



**图52-25.MII接收操作**

如果PHY检测到从线路上收到的帧有错误，PHY在数据包传输过程中的任何时候都会断言MII错误信号MII_RXER，至少有一个时钟周期。

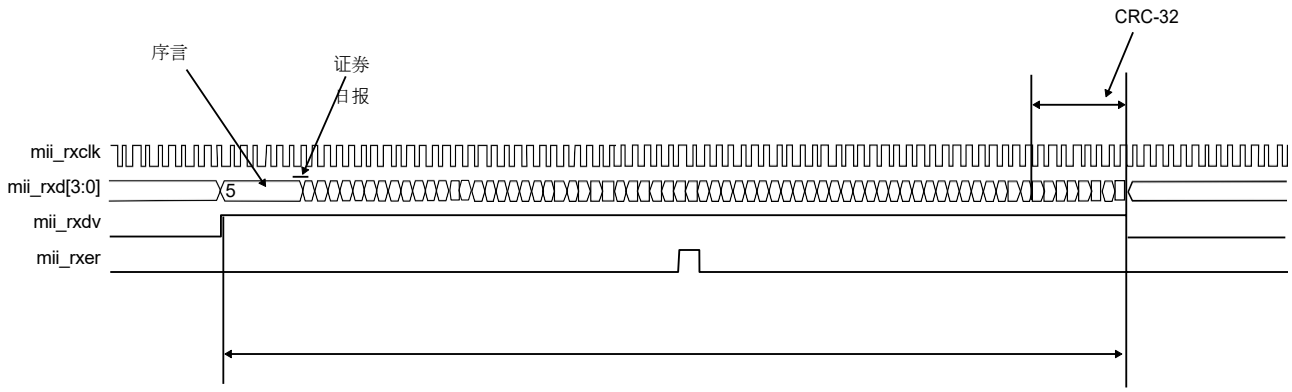



**图52-26.MII接收操作 - 误码帧**

在MII接口上收到的带有PHY错误指示的帧，随后在FIFO接口上传输，RxBD[ME]被设置。

# 第53章 调试

## 53.1 简介

该器件的调试功能基于 ARM CoreSight™
架构，并在每个器件上进行配置，以提供引脚布局和其他可用资源限制所允许的最大灵活性。

**表53-1.支持的调试接口**

| 芯片 | 支持的调试接口 | | | |
|---|---|---|---|---|
| | IEE 1149.1 JTAG | 串行线调试(SWD) | ÅÅÅ | 4针并行跟踪端口1 |
| S32K142 | 是 | 是 | 是 | 没有 |
| S32K146 | 是 | 是 | 是 | 没有 |
| S32K144 | 是 | 是 | 是 | 没有 |
| S32K148 | 是 | 是 | 是 | 是 |

1.覆盖TPIU的支持端口大小寄存器的值（在位置0x000）。

基本的Cortex-
M4调试架构是非常灵活的。下图显示了核心调试架构的拓扑结构和它的组件。

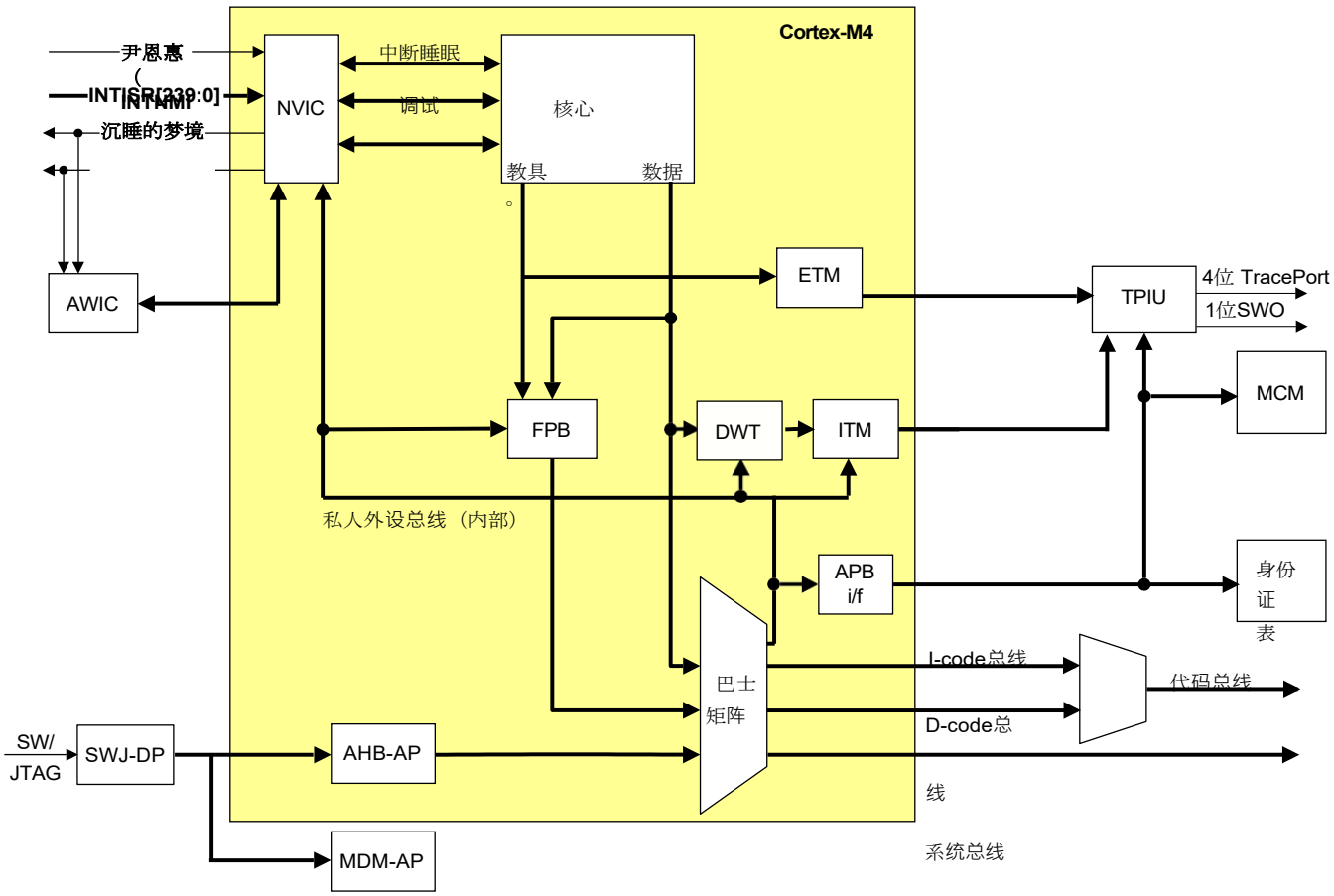



**图53-1.Cortex-M4调试拓扑结构**

下表是对每个调试组件的简要描述。

**表53-2.调试组件描述**

| 模块 | 描述 |
|---|---|
| SWJ-DP（串行线和JTAG - 调试端口）。 | 修改后的调试端口，支持SWD、JTAG |
| AHB-AP（AMBA高性能总线-访问端口）。 | 从JTAG到调试模块和SOC系统内存图的AHB主接口 |
| MDM-AP（杂项调试模块-接入端口）。 | 为外部调试器提供集中的控制和状态寄存器，以控制设备。 |
| ROM表 | 确定哪个调试IP是可用的。 |
| 核心调试 | 单步，寄存器访问，运行，核心状态 |
| ITM（指令跟踪宏单元）。 | S/W仪表信息传递+简单数据跟踪信息传递+观察点信息传递 |
| ETM（嵌入式跟踪宏基）。 | 用于指令追踪 |
| DWT（数据和地址观察点） | 4个数据和地址观察点 |
| FPB（Flash补丁和断点）。 | FPB实现了硬件断点，并将代码和数据从代码空间修补到系统空间。<br>FPB单元包含两个字面比较器，用于与代码空间的字面负载相匹配，并重新映射到系统空间的相应区域。 |

表继续在下一页...

表53-2.调试组件描述（续）

| 模块 | 描述 |
| --- | --- |
| | FPB还包含六个指令比较器，用于与从代码空间获取的指令进行匹配，并重新映射到系统空间的相应区域。另外，这六个指令比较器可以单独配置比较器，以便在匹配时向处理器内核返回断点指令（BKPT），提供硬件断点能力。 |
| TPIU（跟踪端口接口单元） | 异步模式（1针）=TRACE_SWO（在JTAG_TDO上可用）。<br>支持4针跟踪输出和一个单针SWO |

## 53.2 CM4 ROM表

ROM表是用来保存调试组件的信息的。

CM4 ROM表位于CM4 AHB AP上，有CM4调试组件的条目。

表53-3.ROM表中的位分配

| 咬牙切齿 | 命名 | 描述 |
| --- | --- | --- |
| [31:12] | 地址偏移 | 组件的基本地址，相对于ROM地址。负值允许使用二进制补码。<br>ComponentAddress = ROMAddress + （AddressOffset SHL 12）。 |
| [11:2] | - | 保留SBZ。 |
| [1] | 格式 | 1 = 32位格式。在DAP Debug ROM中，这被设置为1。<br>0 = 8位格式。 |
| [0] | 目前进入 | 设为高电平，表示有一个条目存在。 |

表53-4.CM4 ROM表

| 组成部分 | 地址 | 价值 | 笔记 |
| --- | --- | --- | --- |
| NVIC | 0xE00FF000 | 0xFFF0F003 | 存在于S32K14x中<br>基准地址 = ROM表的基准地址 + 0xFFF0F000 |
| DWT | 0xE00FF004 | 0xFFF02003 | 存在于S32K14x中<br>基准地址=ROM表的基准地址+0xFFF02000 |
| FPB | 0xE00FF008 | 0xFFF03003 | 存在于S32K14x中<br>基准地址=ROM表的基准地址+0xFFF03000 |
| ITM | 0xE00FF00C | 0xFFF01003 | 存在于S32K14x中<br>基准地址=ROM表的基准地址+0xFFF01000 |
| TPIU | 0xE00FF010 | 0xFFF41003 | 存在于S32K14x中<br>基准地址=ROM表的基准地址+0xFFF41000 |

*表继续在下一页...*

表53-4.CM4 ROM表（续）

| 组成部分 | 地址 | 价值 | 笔记 |
|---|---|---|---|
| ETM | 0xE00FF014 | 0xFFF42002<br>0xFFF42003 | 存在于148个<br>在142、144和S32K146中不存在。 |
| ETB | 0xE00FF018 | 0xFFF43002 | 不存在于S32K14x中 |
| 芬尼爾 | 0xE00FF01C | 0xFFF44002 | 不存在于S32K14x中 |

## 53.3 调试端口

JTAG控制器和调试端口的配置，如下图所示。

**注意事项**

该芯片不包括CJTAG模块。

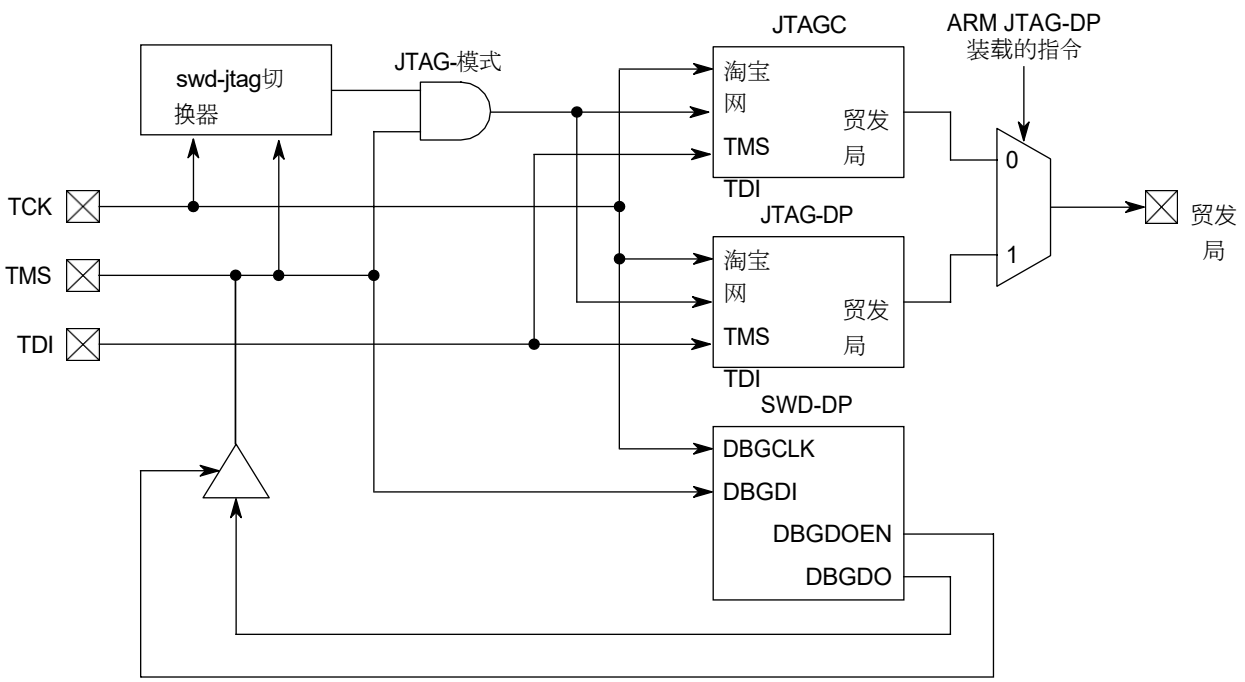



图53-2.修改后的调试端口

调试端口在标准的JTAG模式下复位，并通过以下序列切换到SWD模式。一旦模式被改变，未使用的调试引脚可以被重新分配到它们的任何备选的混合功能中。

### 53.3.1 JTAG到SWD的变化顺序

1. 在TMS（SWDIO）=1的情况下发送超过50个TCK周期
2. 在TMS上发送16位序列（SWDIO）=0111_1001_1110_0111（MSB先传送）。
3. 在TMS（SWDIO）=1的情况下发送超过50个TCK周期

**注意事项**

有关限制，请参见CoreSight DAP Lite的ARM文档。

## 53.4 调试端口引脚描述

在POR之后，所有的调试端口引脚默认为其JTAG功能，以后可以重新分配为其备用功能。在SWD模式下，JTAG_TDI可以被配置为备用的GPIO功能。

**表53-5.调试端口引脚**

| 引脚名称 | JTAG调试端口 | | SWD调试端口 | | 内部上拉/下拉 |
|---|---|---|---|---|---|
| | 类型 | 描述 | 类型 | 描述 | |
| jtag_tms/ swd_dio | 输入/输出 | JTAG测试模式选择 | 输入/输出 | 串行线数据 | 拉升 |
| jtag_tclk/ swd_clk | I | JTAG测试时钟 | I | 串行线时钟 | 下拉式 |
| JTAG_TDI | I | JTAG测试数据输入 | - | - | 拉升 |
| jtag_tdo/ trace_swo | O | JTAG测试数据输出 | O | 在单个引脚上跟踪输出 | N/C |

## 53.5 系统TAP连接

系统JTAG控制器与ARM测试访问端口（TAP）控制器并行连接。系统JTAG控制器指令寄存器（IR）代码与ARM
JTAG控制器的IR代码重叠，没有冲突。有关可用的IR代码的列表，请参考IR代码表。TAP的输出（TDO）是根据所选择的IR代码进行混频的。该设计完全符合JTAG要求，在JTAG链上显示为一个单一的TAP。上电复位时，ARM的IDCODE（IR=

1110b）被选中。

### 53.5.1 IR代码

表53-6.JTAG指令

| 指示 | 代码[3:0] | 指示摘要 |
|---|---|---|
| IDCODE | 0000 | 选择设备识别寄存器进行移位 |
| 采样/预装填 | 0010 | 在不干扰功能操作的情况下，选择边界扫描寄存器进行移位、采样和预加载 |
| 样品 | 0011 | 选择边界扫描寄存器进行移位和采样，不干扰功能操作 |
| 延伸 | 0100 | 选择边界扫描寄存器，同时将预加载的值应用于输出引脚并主张功能复位 |
| 攎璈璀 | 1001 | 选择旁路寄存器，同时对所有输出引脚进行三态，并提出功能复位要求 |
| CLAMP | 1100 | 选择旁路寄存器，同时将预载值应用于输出引脚并断言功能复位 |
| ARM_IDCODE | 1110 | ARM JTAG-DP指令 |
| 旁路 | 1111 | 为数据操作选择旁路寄存器 |
| 保留工厂调试功能 | 0101, 0110, 0111, 1101 | 仅用于工厂调试 |
| ARM JTAG-DP 保留 | 1000, 1010, 1011, 1110 | 这些指令将去ARM JTAG-DP控制器。关于这些指令的更多信息，请参见ARM JTAG-DP文档。 |
| 保留 [1] | 所有其他操作码 | 解码后选择旁路寄存器 |

1.制造商保留在未来改变保留指令代码的解码的权利

## 53.6 JTAG状态和控制寄存器

通过ARM调试访问端口（DAP），调试器可以访问状态和控制元素，如图53-3所示，在DAP总线上实现为寄存器。
这些寄存器为低功耗模式的恢复和典型的运行控制情况提供额外的控制和状态。状态寄存器位还允许调试器获得内核的最新状态，而不需要在横梁开关上启动总线交易，因此在调试会话中保持较低的干扰性。

这些DAP控制和状态寄存器在系统内存图中没有内存映射，只能通过DAP使用JTAG或SWD访问。MDM-AP可作为调试访问端口1访问，其可用寄存器如下表所示。

表53-7.MDM-AP寄存器摘要

| 地址 | 注册 | 描述 |
|---|---|---|
| 0x0100_0000 | 状况 | 见MDM-AP状态寄存器 |

| | | |
|---|---|---|
| 0x0100_0004 | 控制 | 见MDM-AP控制寄存器 |
| 0x0100_00FC | 身份证 | 只读的识别寄存器，总是读作0x001C_0000 |

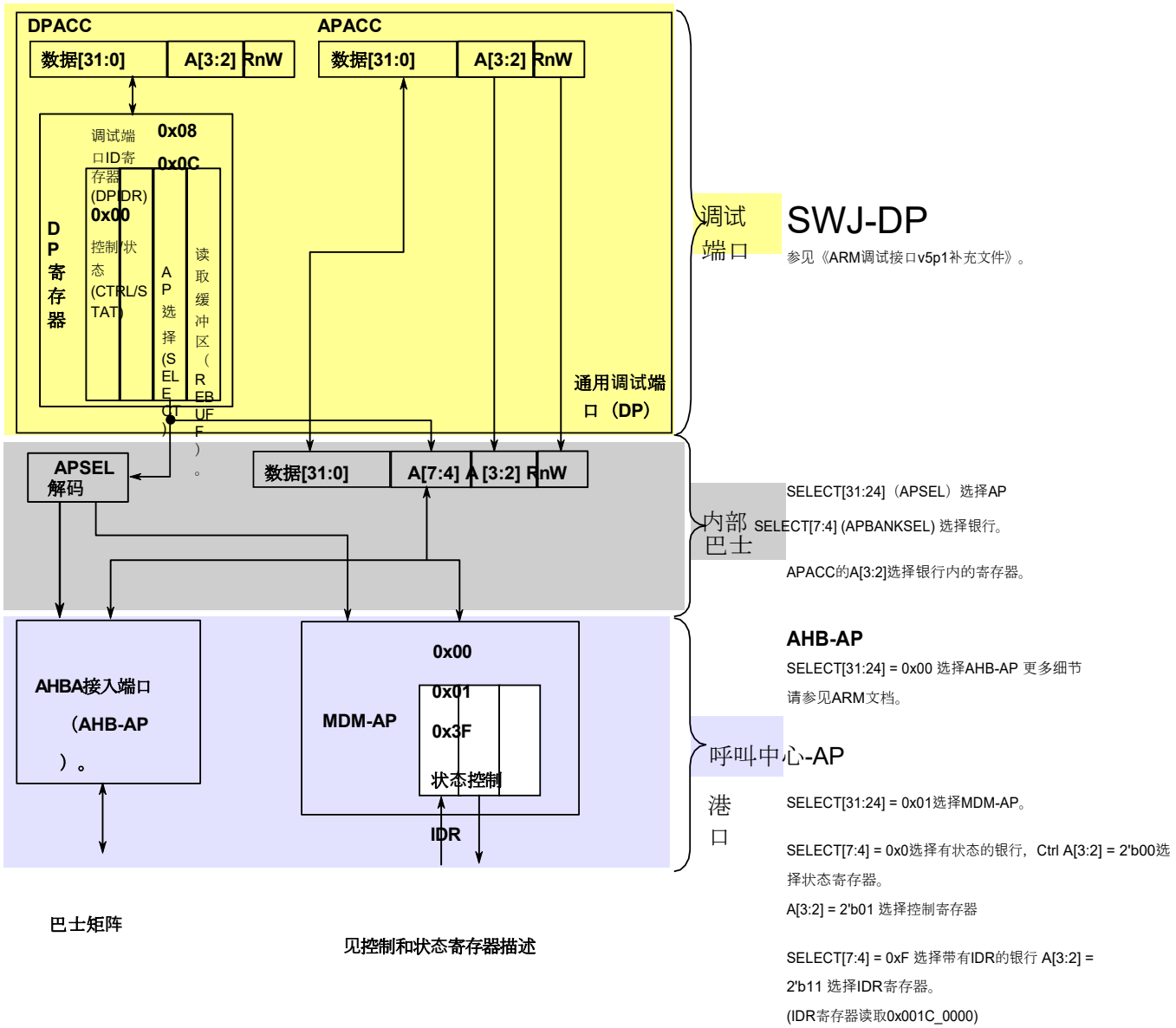



**图53-3.MDM AP寻址**

## 53.6.1 MDM-AP控制寄存器

**表53-8.MDM-AP控制寄存器的分配**

| 位 | 命名 | 命令在安全模式下可用吗？ | 描述 |
|---|---|---|---|

| | | | |
|---|---|---|---|
| 0 | 闪存大量擦除正在进行中 | 是 | 设置为导致大规模擦除。在大规模擦除操作完成后由硬件清除。<br>当大规模擦除被禁用时（通过MEEN和SEC设置），擦除请求不会发生，闪存大规模擦除进行中的位继续断言，直到下一次系统复位。 |

*表继续在下一页...*

表53-8.MDM-AP控制寄存器的分配（续）

| 位 | 命名 | 命令在安全模式下可用吗？ | 描述 |
|---|---|---|---|
| 1 | 禁用调试 | 没有 | 设置为禁用调试。清除则允许调试操作。当设置时，它将覆盖DHCSR中的C_DEBUGEN位，并强制禁用调试逻辑。 |
| 2 | 调试请求 | 没有 | 设置为强制内核停止。<br>如果内核处于停止模式，这个位可以用来唤醒内核并过渡到停止状态。 |
| 3 | 系统复位请求 | 没有 | 设为强制系统复位。系统保持在复位状态，直到该位被清除。 |
| 4 | 磁芯保持复位 | 没有 | 配置位，在系统复位排序结束时控制核心操作。<br>0：正常操作 - 在系统复位顺序结束时，将核心与系统的其他部分一起从复位中释放出来。<br>1: 暂停操作 - 在复位序列结束时将内核保持在复位状态。在系统进入这个暂停状态后，清除这个控制位会立即将内核从复位中释放出来，CPU开始运行。 |
| 5 - 7 | 保留 | 没有 | |
| 8 | 禁用时间戳 | 没有 | 在调试停止模式中，当内核停止时，设置该位以禁用48位全局跟踪时间戳计数器。<br>0: 假设跟踪功能被启用，时间戳计数器继续计数。(默认)<br>1：当内核停止时，时间戳计数器会冻结（调试停止模式）。 |
| 9 - 31 | 保留给未来使用 | 没有 | |

## 53.6.2 MDM-AP状态寄存器

表53-9.MDM-AP状态寄存器的分配

| 位 | 命名 | 描述 |
|---|---|---|
| 0 | 闪存大量擦除确认 | 这个位在调试复位时被清空。由于在MDM AP控制寄存器中写入了Mass Erase in Progress位，在启动大规模擦除命令时，该位也被清除。该位在闪存控制逻辑启动大规模擦除操作后被设置。<br>当大规模擦除被禁用时（通过MEEN和SEC设置），由于设置该位而产生的擦除请求不会被确认。 |
| 1 | 闪存准备就绪 | 表示闪存已经被初始化，即使系统继续通过调试器保持在复位状态，也可以对调试器进行配置。 |

| | | |
|---|---|---|
| 2 | 系统安全 | 表示安全状态。当安全时，调试器不能访问系统总线或任何内存映射的外围设备。该位表示当部件被锁定时，不可能有系统总线访问。 |

*表继续在下一页...*

表53-9.MDM-AP状态寄存器的分配（续）

| 位 | 命名 | 描述 |
|---|---|---|
| 3 | 系统重置 | 表明系统复位状态。0: 系统处于<br>复位状态<br>1：系统未处于复位状态 |
| 4 | 保留 | |
| 5 | 启用大规模擦除功能 | 表明MCU是否可以被大规模擦除 0: 大规模擦除被<br>禁用<br>1: 启用了大规模擦除功能 |
| 6 | 启用后门访问密钥 | 表示MCU是否启用了后门访问密钥。更多信息见FTFE_FSEC[KEYEN]位。<br>0: 禁用<br>1: 已启用 |
| 7 | 启用LP | 对LPLLSM控制位进行解码，表明VLPS是下一次ARM核进入深度睡眠时选择的电源模式。<br>0：未启用低功率停止模式 1：启用低功<br>率停止模式<br>使用方法是用于尝试运行到VLPS的调试操作。根据调试定义，系统实际上进入了停止状态。调试器应该将深度睡眠指示（SLEEPDEEP和SLEEPING断言）与该位断言一起解释为调试器-VLPS状态指示。 |
| 8 | 非常低的功率模式 | 表示当前的电源模式是VLPx。该位不是 "粘性 "的，应始终代表VLPx是否被启用。<br>该位被调试器用来节制JTAG TCK频率的上升/下降。 |
| 9 - 10 | 保留 | 总是读作0。 |
| 11 - 15 | 保留给未来使用 | 总是读作0。 |
| 16 | 核心提示：停止了 | 表示内核已进入调试停止模式 |
| 17 | 核心的SLEEPDEEP | 表示内核已进入低功率模式<br>SLEEPING==1和SLEEPDEEP==1表示停止或VLPS模式。 |
| 18 | 核心提示： 睡眠 | |
| 19 - 31 | 保留给未来使用 | 总是读作0。 |

# 53.7 调试复位

调试系统接收以下复位源。

- 调试复位（SWJ-DP

CTRL/STAT寄存器内的CDBGRSTREQ位）在TCLK域，允许调试器复位调试逻辑。

- 系统POR复位

反之，调试系统能够使用以下机制产生系统复位。

- DAP控制寄存器中的系统复位，允许调试器将系统保持在复位状态。
- NVIC应用中断和复位控制寄存器中的SYSRESETREQ位

## 53.8 AHB-AP

AHB-
AP为调试器提供对系统中所有内存和寄存器的访问，包括通过NVIC的处理器寄存器。系统访问与处理器的状态无关。AHB-
AP在总线上不执行背对背的事务，因此所有的事务都是非连续的。AHB-
AP可以执行非对齐和位带交易。AHB-
AP事务绕过FPB，所以FPB不能重新映射AHB-AP事务。SWJ/SW-
DP发起的交易中止会驱动一个AHB-
AP支持的边带信号，称为HABORT。这个信号被驱动到总线矩阵，它重置了总线矩阵的状态，这样AHB-
AP就可以访问私有外设总线进行最后的调试，如读/停/重设内核。AHB-
AP的交易是小字头的。

MPU包括区域描述符0（RGD0）的默认设置和保护，这样调试器总是可以访问整个地址空间，这些权利不能被内核或任何其他总线主站改变。

在系统复位事件开始的短时间内，系统安全状态正在被确定，调试器对所有AHB-
AP事务的访问被阻止。MDM-
AP状态寄存器是可以访问的，并且可以被监测以确定这个初始时期何时完成。在这个初始阶段之后，如果通过RESET_b引脚的断言保持系统复位，调试器可以通过总线矩阵访问私有外设总线来配置调试IP，即使在系统复位断言的情况下。在系统复位时，对其他内存和寄存器资源的访问被阻断，这些资源通过横梁开关访问。

## 53.9 ITM

Instrumentation Trace Macrocell (ITM)
是一个应用程序驱动的跟踪源，支持printf风格的调试，以跟踪操作系统和应用程序事件，并发射诊断系统信息。ITM以数据包的形式发射跟踪信息。有四个源可以产生数据包。如果多个源同时产生数据包，ITM将仲裁数据包的输出顺序。这四个源按优先级递减的顺序是：。

1. 软件跟踪。软件可以直接写到ITM刺激寄存器。这就发出了数据包。

2. 硬件跟踪。DWT产生这些数据包，而ITM则发射这些数据包。

3. 时间戳。时间戳是相对于数据包而发出的。ITM包含一个21位的计数器来生成时间戳。该计数器由Cortex-M4时钟或由串行线查看器（SWV）输出的位时钟速率来控制。
4. 全局系统时间戳。时间戳可以选择使用全系统48位计数值生成。

## 53.10 核心跟踪连接

下表提供了跨S32K14x设备的核心跟踪连接。

**表53-10.S32K14x内核跟踪连接**

| 芯片 | 特点 |
|---|---|
| S32K42, S32K144, S32K146 | ITM将其数据传送到TPIU |
| S32K148 | ETM、ITM和DWT都通过TPIU将TRACE数据送出。 |

## 53.11 TPIU

TPIU是一个跟踪数据引流器，作为片上跟踪数据与*跟踪端口分析仪*（TPA）所捕获的数据流之间的桥梁。

**表53-11.S32K14x TPIU**

| 芯片 | 特点 |
|---|---|
| S32K142, S32K144, S32K146 | 器件中的TPIU支持完整TPIU功能的有限子集（只有SWO模式），以尽量减少简单调试方案的门数量。 |
| S32K148 | 对于需要高追踪带宽的应用，可以使用4针追踪接口。<br>SWO模式只能在SWD模式下使用，因为它与JTAG TDO使用相同的引脚。 |

SWO的最大跟踪时钟为20MHz。4针追踪接口的最大追踪时钟映射到。

- 快速垫是80兆赫
- 慢速垫是20MHz

这可以通过系统时钟分频器寄存器4（CLKDIV4）进行配置。

## 53.12 DWT

DWT提供调试功能。它包含四个比较器，每个都可以配置为硬件观察点、PC采样器事件触发器或数据地址采样器事件触发器。第一个比较器，DWT_COMP0，也可以与时钟周期计数器CYCCNT比较。第二个比较器DWT_COMP1也可以作为数据比较器使用。DWT还包含以下的计数器。

- 时钟周期 (CYCCNT)
- 折叠的说明
- 负载存储单元（LSU）操作
- 睡眠周期
- CPI（除第一个周期外的所有指令周期）
- 中断开销

每次计数器溢出时都会发出一个事件。

DWT可以被配置为在定义的时间间隔内发射PC样本，并发射中断事件信息。

## 53.13 低功耗模式下的调试

在低功耗模式下，调试模块保持静态或断电，调试器在低功耗模式期间不能收集任何调试数据。在调试器保持静态的情况下，一旦低功耗模式退出，调试端口就会恢复全部功能，系统就会恢复到活动调试的状态。在调试器逻辑断电的情况下，调试器在恢复时被重置，必须在退出低功率模式后重新配置。

来自调试端口的调试上电和系统上电信号被电源模式进入逻辑监测，作为调试器被激活的指示。这些信号可以在RUN和VLPR中改变。如果调试信号是激活的，而系统试图进入停止或VLPS，FCLK继续运行以支持核心寄存器访问。在这些FCLK保持激活的模式下，调试模块可以访问核心寄存器，但不能访问通过横梁开关访问的系统内存资源。

启用调试后，不允许从运行直接过渡到VLPS，而是导致系统进入停止模式。MDM-
AP状态寄存器中的状态位可以被评估以确定这种伪VLPS状态。请注意，在调试启用的情况下，从运行→VLPR→VLPS的转换仍然是可能的，但也会导致系统进入停止模式。

### 53.13.1 低功耗模式下的调试模块状态

下表显示了调试模块在低功率模式下的状态。使用了这些术语。

- FF =
  全功能。在VLPR中，系统频率是有限的，但如果一个模块的功能没有限制，它仍然被列为FF。
- 静态 = 保留模块寄存器状态和相关内存。
- OFF = 模块已断电；模块在唤醒时处于复位状态。

**表53-12.低功耗模式下的调试模块状态**

| 模块 | 停止[1] | VLPR | VLPS |
|---|---|---|---|
| 调试端口 | FF | FF | 关闭 |
| AHB-AP | FF | FF | 关闭 |
| ITM | FF | FF | 关闭 |
| TPIU | FF | FF | 关闭 |
| DWT | FF | FF | 关闭 |
| ETM [2] | FF | FF | 关闭 |

1. 调试器在STOP1和STOP2中的状态与在STOP中相同。
2. 仅存在于S32K 148中

## 53.14 调试和安全

当安全状态被启用（FSEC[SEC]
!=10）时，调试端口的能力被限制，以防止安全数据被利用。在安全状态下，调试器仍然可以访问MDM-
AP状态寄存器，并可以确定设备的当前安全状态。在安全设备的情况下，调试器也有能力通过写入MDM-
AP控制寄存器来执行大规模擦除操作。如果安全设备的大规模擦除功能被禁用（FSEC[MEEN] = 10），通过调试接口进行大规模擦除的尝试会被阻止。

# 第54章
# JTAG控制器（JTAGC）

## 54.1 芯片特定的JTAGC信息

在JTAGC的设备识别寄存器中，S32K14x系列的不同产品的零件识别码（PIN）字段的值是不同的。下表提供了每个产品的PIN值。

**表54-1.JTAG ID寄存器的PIN值**

| 芯片 | 密码值 |
|---|---|
| S32K116 | TBD |
| S32K118 | TBD |
| S32K142 | 0b0100111000 |
| S32K144 | 0b0100111101 |
| S32K146 | 0b0101000100 |
| S32K148 | 0b0101010000 |

## 54.2 简介

JTAGC块提供了测试芯片功能和连接性的方法，同时在不处于测试模式时对系统逻辑保持透明。测试是通过边界扫描技术进行的，如IEEE 1149.1-2001标准所定义。所有输入和输出到JTAGC模块的数据都是以串行格式进行通信。

### 54.2.1 框图

下面是JTAG控制器（JTAGC）块的简化框图。关于JTAGC寄存器的更多信息，请参考芯片的具体配置信息以及寄存器描述。

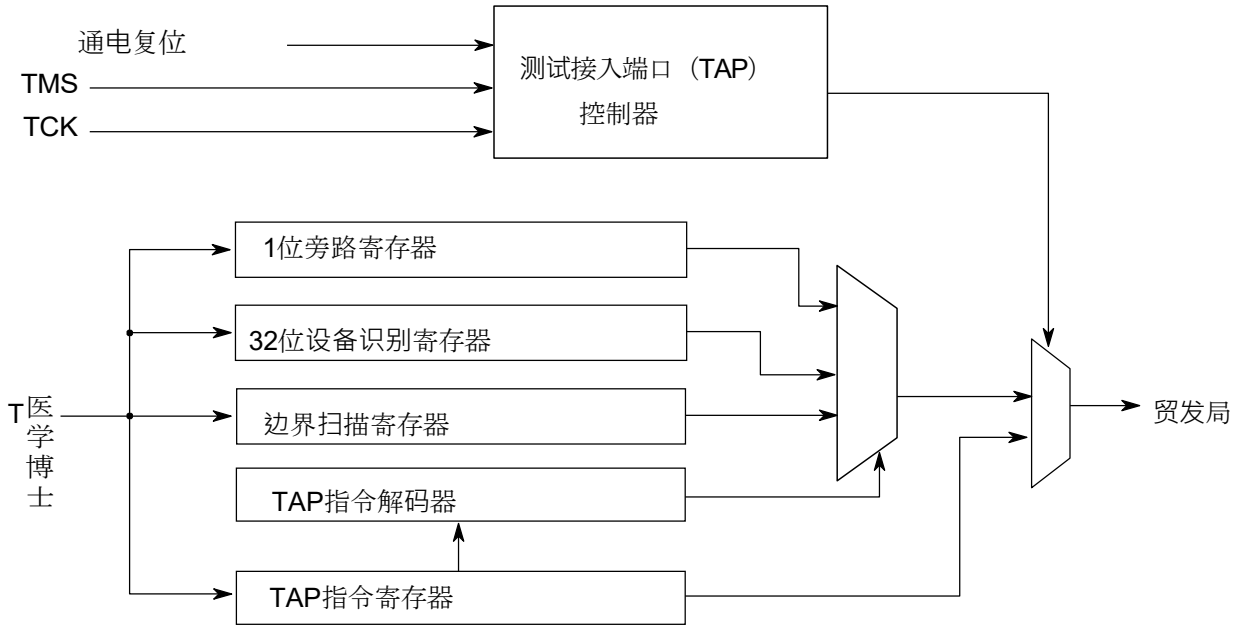



**图54-1.JTAG（IEEE 1149.1）框图**

## 54.2.2 特点

JTAGC块符合IEEE 1149.1-2001标准，并支持以下功能。

- IEEE 1149.1-2001测试接入端口（TAP）接口
  - 4个引脚（TDI、TMS、TCK和TDO）。
- 指令寄存器，支持几个IEEE 1149.1-2001定义的指令，以及几个公共和私人设备特定的指令。关于支持的指令列表，请参考JTAGC块指令。
- 旁路寄存器，边界扫描寄存器，和设备识别寄存器。
- TAP控制器状态机，控制数据寄存器、指令寄存器和相关电路的运行。

## 54.2.3 操作模式

JTAGC模块使用上电复位指示作为其主要复位信号。支持几个IEEE 1149.1-2001定义的测试模式，以及一个旁路模式。

#### 54.2.3.1 复位

当上电复位时，或TMS输入被保持在高电平，有足够多的TCK连续上升沿，使TAP控制器状态机进入测试-逻辑-
复位状态时，JTAGC块被置于复位状态。无论当前TAP控制器的状态如何，保持TMS高电平连续5个TCK上升沿可保证进入测试-逻辑-
复位状态。断电复位会导致异步进入复位状态。在复位状态下，会发生以下动作。

- TAP控制器被强制进入测试-逻辑-
  复位状态，从而使测试逻辑失效，并允许片上系统逻辑的正常运行不受阻碍地继续。
- 指令寄存器被加载IDCODE指令

#### 54.2.3.2 IEEE 1149.1-2001定义的测试模式

JTAGC块支持几种IEEE 1149.1-
2001定义的测试模式。在JTAGC启用时，通过向指令寄存器加载适当的指令来选择测试模式。支持的测试指令包括EXTEST、HIGHZ、CLAMP、SAMPLE和SAMPLE/PRELOAD。每条指令都定义了一组数据寄存器，当指令当前时，这些寄存器可以操作并与片上系统逻辑互动。每条指令只有一个测试数据寄存器路径被启用，在TDI和TDO之间转移数据。

当EXTEST、SAMPLE或SAMPLE/PRELOAD指令激活时，边界扫描寄存器被启用，用于TDI和TDO之间的串行访问。当下列指令激活时，单比特旁路寄存器移位阶段被启用，用于TDI和TDO之间的串行访问。

- 旁路
- 攒璈瑽
- CLAMP

每个测试模式的功能在JTAGC块指令中有更详细的解释。

#### 54.2.3.3 旁路模式

当不需要测试操作时，可以加载BYPASS指令，使JTAGC块进入旁路模式。在旁路模式下，单位旁路移位寄存器用于提供最小长度的串行路径，在TDI和TDO之间移位数据。

## 54.3 外部信号描述

JTAGC由一组信号组成，连接到片外开发工具并允许访问测试支持功能。下表概述了JTAGC的信号，并在以下章节进行了描述。

**表54-2.JTAG信号属性**

| 命名 | 输入/输出 | 职能 | 重置状态 |
|---|---|---|---|
| 淘宝网 | 输入 | 测试时钟 | 疲软的下拉 |
| TDI | 输入 | 测试数据输入 | 弱拉升 |
| 贸发局 | 输出 | 测试数据输出 | 高Z1 |
| TMS | 输入 | 测试模式选择 | 弱拉升 |

1.当JTAGC不处于Shift-IR或Shift-DR状态时，TDO输出缓冲器使能被否定。当JTAGC不工作时，可以在TDO焊盘上实现一个弱的拉力。

### 54.3.1 TCK-测试时钟输入

测试时钟输入（TCK）是一个输入引脚，用于同步测试逻辑和控制寄存器通过TAP的访问。

### 54.3.2 TDI-测试数据输入

测试数据输入（TDI）是一个接收串行测试指令和数据的输入引脚。TDI在TCK的上升沿被采样。

### 54.3.3 TDO-测试数据输出

测试数据输出（TDO）是一个输出引脚，用于传输测试指令和数据的串行输出。TDO是三态的，只在TAP控制器状态机的Shift-IR和Shift-DR状态下被主动驱动，这在TAP控制器状态机中有描述。

### 54.3.4 TMS-测试模式选择

测试模式选择（TMS）是一个输入引脚，用于对IEEE 1149.1-2001测试控制状态机进行排序。TMS在TCK的上升沿被采样。

## 54.4 注册说明

本节详细描述了可通过TAP接口访问的JTAGC块寄存器，包括数据寄存器和指令寄存器。包括每个寄存器的个别位级描述和复位状态。这些寄存器没有内存映射，只能通过TAP访问。

### 54.4.1 指令寄存器

JTAGC块使用一个4位指令寄存器，如下图所示。指令寄存器允许将指令加载到该块中，以选择要执行的测试或要访问的测试数据寄存器，或两者兼有。当TAP控制器处于移位-IR状态时，指令通过TDI移入，并在更新-IR状态下的TCK下降沿锁定。锁定的指令值只能在Update-IR和Test-Logic-Reset TAP控制器状态下改变。同步进入Test-Logic-Reset状态的结果是IDCODE指令在TCK的下降沿被加载。异步进入Test-Logic-Reset状态的结果是IDCODE指令的异步加载。在Capture-IR TAP控制器状态下，指令移位寄存器被加载值0b1，当TAP控制器被排序到Shift-IR状态时，这个值成为寄存器的读取值。

| | 3 | 2 | 1 | 0 |
|---|---|---|---|---|
| R | 0 | 0 | 0 | 1 |
| W | 指示代码 | | | |
| 重置。 | 0 | 0 | 0 | 1 |

**图54-2.指令寄存器**

### 54.4.2 旁路寄存器

旁路寄存器是一个单位移位寄存器路径，当下列指令激活时，在TDI和TDO之间选择串行数据传输。

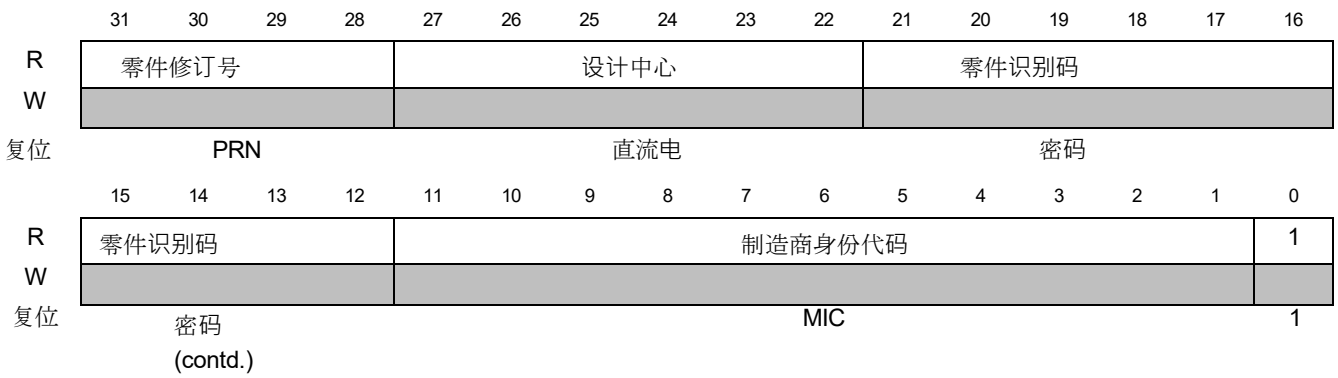
- 旁路

- 攎璈璇
- CLAMP

进入捕获-
DR状态后，单比特移位寄存器被设置为逻辑0。因此，选择旁路寄存器后移出的第一个位总是逻辑0。

## 54.4.3 设备识别寄存器

设备识别（JTAG
ID）寄存器，如下图所示，允许通过TAP确定修订号、零件号、制造商和负责设计该零件的设计中心。当IDCODE指令激活时，设备识别寄存器被选择用于TDI和TDO之间的串行数据传输。当设备识别寄存器被选中时进入捕获-
DR状态，将IDCODE加载到移位寄存器中，在移位-
DR状态下被移出TDO。在Update-DR状态下不发生任何动作。




下表描述了设备识别寄存器的功能。

**表54-3.设备识别寄存器字段描述**

| 场地 | 描述 |
|---|---|
| 报道 | 零件修订号。包含零件的修订号。值为0x0。 |
| 直流电 | 设计中心。表示设计中心。值为0x26。 |
| 密码 | 零件识别号。包含设备的部件号。值为0b0101010000。 |
| 医学博士 | 制造商身份代码。包含减少的联合电子器件工程委员会（JEDEC）ID。值为0x00E。 |
| IDCODE ID | IDCODE 寄存器ID。确定该寄存器为设备识别寄存器，而不是旁路寄存器。总是设置为1。 |

### 54.4.4 边界扫描寄存器

当EXTEST、SAMPLE或SAMPLE/PRELO指令激活时，边界扫描寄存器被连接在TDI和TDO之间。它被用来捕获输入引脚数据，强制输出引脚的固定值，并为双向引脚选择一个逻辑值和方向。边界扫描寄存器的每一位都代表一个独立的边界扫描寄存器单元，如IEEE 1149.1-
2001标准中所述，并在边界扫描中讨论。边界扫描寄存器的大小和位排序与器件有关，可以在器件BSDL文件中找到。

## 54.5 职能描述

本节解释JTAGC的功能描述。

### 54.5.1 JTAGC复位配置

在复位时，TAP控制器被强制进入测试-逻辑-
复位状态，从而使测试逻辑失效，允许片上系统逻辑的正常运行。此外，指令寄存器被加载IDCODE指令。

### 54.5.2 IEEE 1149.1-2001（JTAG）测试接入端口

JTAGC块使用IEEE 1149.1-2001
TAP来访问寄存器。这个端口可以与MCU上的其他TAP控制器共享。该端口的所有权由当前加载指令的值决定。

数据在TDI和TDO之间通过选定的寄存器从最小有效位开始移位，如下图所示。这适用于指令寄存器、测试数据寄存器和旁路寄存器。

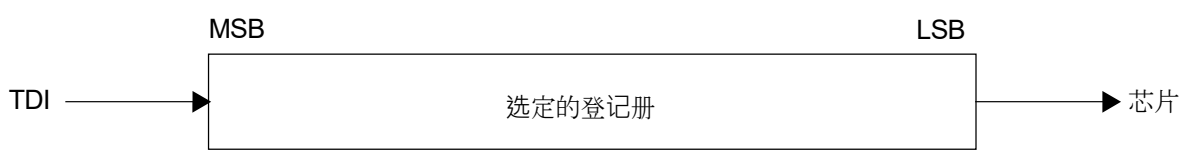
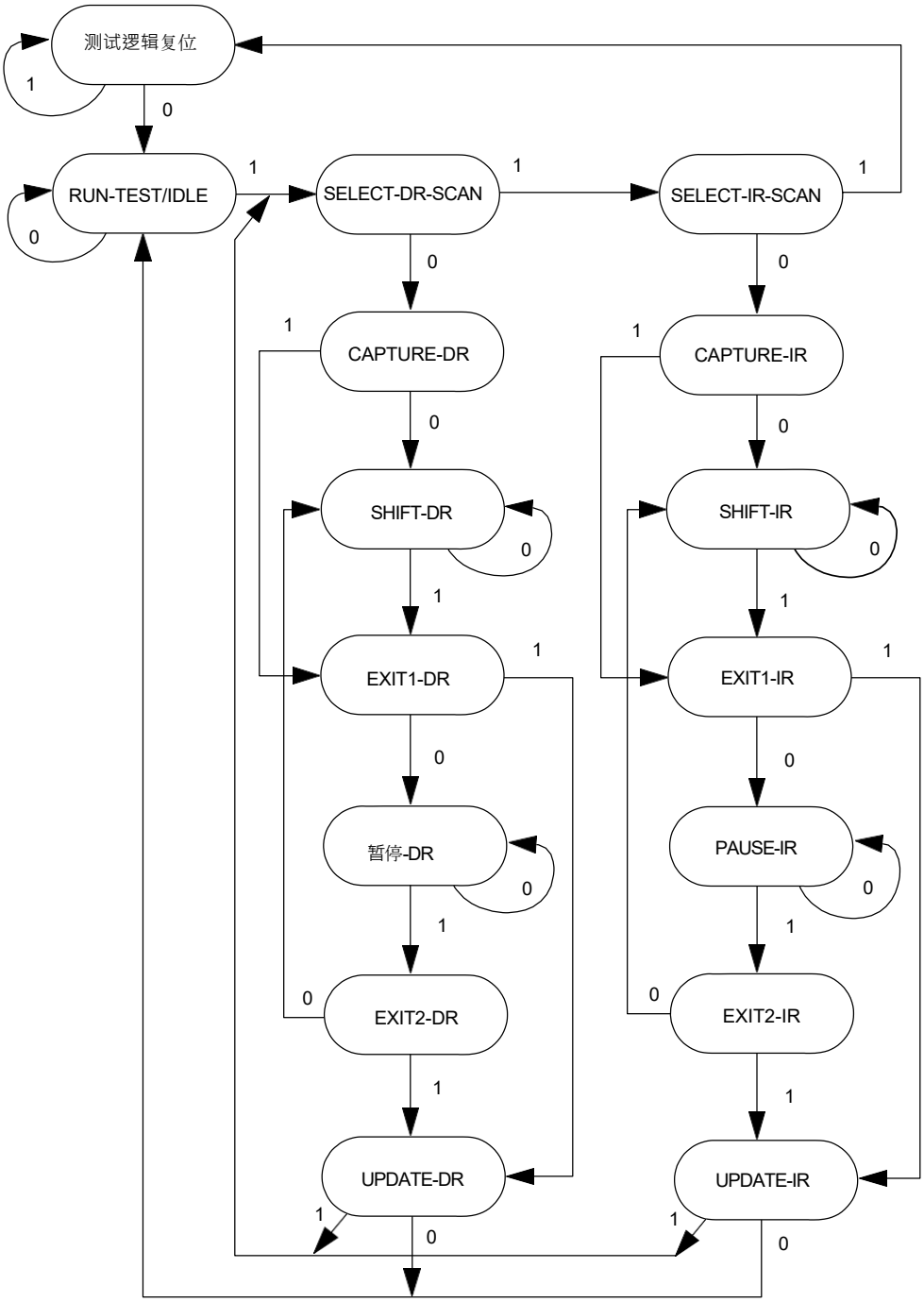



**图54-3.通过寄存器移位数据**

### 54.5.3 TAP控制器状态机

TAP控制器是一个同步状态机，解释TMS引脚上的逻辑值序列。下图显示了该机器的状态。每个状态旁边显示的值是在TCK信号的上升沿采样的TMS信号的值。如下图所示，将TMS保持在逻辑1，同时通过足够多的上升沿对TCK进行计时，也会使状态机进入测试-逻辑-复位状态。




本图中每个状态转换旁显示的数值代表TCK上升沿时TMS的数值。

**图54-4.IEEE 1149.1-2001 TAP控制器有限状态机**

### 54.5.3.1 启用TAP控制器

JTAGC TAP控制器通过设置JTAGC enable为逻辑1值而被启用。

#### 54.5.3.2 选择一个IEEE 1149.1-2001寄存器

对JTAGC数据寄存器的访问是通过在JTAGC启用时用任何JTAGC块指令加载指令寄存器实现的。指令通过Select-IR-Scan路径移入，在Update-IR状态下加载。这时，所有的数据寄存器访问都是通过Select-DR-Scan路径进行的。

选择-DR-扫描路径用于在Shift-DR状态下通过移位数据（LSB优先）来读或写寄存器数据。当读一个寄存器时，在捕获-DR状态下，寄存器的值被加载到IEEE 1149.1-2001移位器。当写一个寄存器时，在Update-DR状态下，数值从IEEE 1149.1-2001移位器加载到寄存器中。当读一个寄存器时，没有要求移出整个寄存器的内容。一旦获得了所需的位数，移位就可以终止了。

### 54.5.4 JTAGC块指令

JTAGC块实现了下表中所列的IEEE 1149.1-2001定义的指令。本节对每条指令进行了概述，更多细节请参考IEEE 1149.1-2001标准。所有未定义的操作码都被保留。

**表54-4.4位JTAG指令**

| 指示 | 代码[3:0] | 指示摘要 |
|---|---|---|
| IDCODE | 0000 | 选择设备识别寄存器进行移位 |
| 采样/预装填 | 0010 | 在不影响功能操作的情况下，选择边界扫描寄存器进行移位、采样和预加载 |
| 样品 | 0011 | 选择边界扫描寄存器进行移位和采样，不干扰功能操作 |
| 延伸 | 0100 | 选择边界扫描寄存器并将预装的值应用于输出引脚。<br>**注意：**执行这条指令会断言功能复位。 |
| 保留工厂调试功能 | 0101 | 仅用于工厂调试 |
| 保留工厂调试功能 | 0110 | 仅用于工厂调试 |
| 保留工厂调试功能 | 0111 | 仅用于工厂调试 |
| ARM JTAG-DP 保留 | 1000 | 这条指令进入ARM JTAG-DP控制器。更多信息请参见ARM JTAG-DP文档。 |
| 攇璈璀 | 1001 | 选择旁路寄存器和所有输出引脚的三态。 |

| | | |
|---|---|---|
| | | **注意：**执行这条指令会断言功能复位。 |
| ARM JTAG-DP 保留 | 1010 | 这条指令进入ARM JTAG-DP控制器。更多信息请参见ARM JTAG-DP文档。 |

*表继续在下一页...*

表54-4.4位JTAG指令（续）

| 指示 | 代码[3:0] | 指示摘要 |
|---|---|---|
| ARM JTAG-DP 保留 | 1011 | 这条指令进入ARM JTAG-DP控制器。更多信息请参见ARM JTAG-DP文档。 |
| CLAMP | 1100 | 选择旁路寄存器并将预装的值应用于输出引脚。<br><br>**注意：**执行这条指令会断言功能复位。 |
| ARM JTAG-DP 保留 | 1110 | 这条指令进入ARM JTAG-DP控制器。更多信息请参见ARM JTAG-DP文档。 |
| 旁路 | 1111 | 为数据操作选择旁路寄存器 |

#### 54.5.4.1 IDCODE指令

IDCODE选择32位设备识别寄存器作为TDI和TDO之间的转移路径。该指令允许对MCU进行询问，以确定其版本号和其他部件识别数据。IDCODE是当JTAGC块被复位时放入指令寄存器的指令。

#### 54.5.4.2 SAMPLE/PRELOAD指令

SAMPLE/PRELOAD指令有两个功能。

- 指令的SAMPLE部分获得系统数据和控制信号的样本，这些信号存在于MCU的输入引脚和输出引脚的边界扫描寄存器单元之前。当SAMPLE/PRELO指令激活时，该采样发生在捕获-DR状态下TCK的上升沿。采样的数据在移位-DR状态下，通过边界扫描寄存器移位到TDO输出来查看。数据捕获和移位操作对系统操作都是透明的。
- 在选择EXTEST或CLAMP指令进行边界扫描测试之前，指令的PRELOAD部分初始化了边界扫描寄存器单元。这是通过在Shift-DR状态下将初始化数据移入边界扫描寄存器来实现的。在Update-DR状态下，初始化数据在TCK的下降沿被传输到边界扫描寄存器单元的并行输出。该数据通过EXTEST或CLAMP指令应用于外部输出引脚。系统操作不受影响。

#### 54.5.4.3 示例指令

SAMPLE指令获取MCU输入引脚和输出引脚的边界扫描寄存器单元之前的系统数据和控制信号的采样。这个采样发生在SAMPLE指令激活时的捕获-DR状态下的TCK上升沿。采样的数据在移位-DR状态下通过边界扫描寄存器移位到TDO输出来查看。在Update-DR状态下，没有确定的操作。数据采集和移位操作对系统操作都是透明的。

#### 54.5.4.4 EXTEST 外部测试指令

EXTEST选择边界扫描寄存器作为TDI和TDO之间的转移路径。它允许通过驱动边界扫描寄存器中包含的预装数据到系统输出引脚来测试片外电路和板级互连。通常情况下，在选择EXTEST之前，使用SAMPLE/PRELOAD指令将预装数据加载到边界扫描寄存器。EXTEST为MCU发出内部系统复位信号，在执行外部边界扫描操作时，强制保持可预测的内部状态。

#### 54.5.4.5 ENABLE_SOC_DATA1指令

ENABLE_SOC_DATA1指令捕获SoC数据并选择SOC_DATA寄存器作为TDI和TDO之间的转移路径进行连接。

#### 54.5.4.6 HIGHZ指令

HIGHZ选择旁路寄存器作为TDI和TDO之间的转移路径。当HIGHZ处于激活状态时，所有的输出驱动器都处于非激活驱动状态（例如，高阻抗）。HIGHZ也为MCU发出内部系统复位信号，以迫使其进入可预测的内部状态。

#### 54.5.4.7 CLAMP指令

CLAMP允许从MCU引脚驱动的信号的状态从边界扫描寄存器确定，同时旁路寄存器被选为TDI和TDO之间的串行路径。CLAMP增强了测试效率，因为它将整个移位路径减少到了

在通过边界扫描寄存器进行EXTEST类型的指令时，单一位（旁路寄存器）。CLAMP还为MCU发出内部系统复位，以迫使其进入可预测的内部状态。

#### 54.5.4.8 绕过指令

BYPASS选择旁路寄存器，在TDI和TDO之间建立一个单比特移位寄存器路径。当不需要对MCU进行测试操作时，BYPASS通过减少整个移位路径来提高测试效率。这使得测试数据可以更快速地在电路板上需要执行测试功能的其他部件之间移动。当BYPASS指令处于激活状态时，系统逻辑运行正常。

### 54.5.5 边界扫描

边界扫描技术允许通过与每个焊盘相关的移位寄存器级来控制和观察元件边界上的信号。每个阶段都是更大的边界扫描寄存器单元的一部分，每个焊盘的单元都是串行连接的，形成一个围绕设计边界的移位寄存器链。边界扫描寄存器由这个移位寄存器链组成，当EXTEST、SAMPLE或SAMPLE/PRELOAD指令被加载时，它连接在TDI和TDO之间。移位寄存器链包含一个串行输入和串行输出，以及时钟和控制信号。

## 54.6 初始化/应用信息

测试逻辑是一种静态逻辑设计，TCK可以在高或低的状态下停止，而不会丢失数据。然而，系统时钟在内部并不与TCK同步。任何同时使用测试逻辑和系统功能逻辑的混合操作都需要外部同步。

为了初始化JTAGC块并启用对寄存器的访问，需要按以下顺序进行。

1. 通过TMS控制的TAP控制器状态机转换，将JTAGC置于复位状态。
2. 为要执行的测试或行动加载适当的指令

# 附录A
# 第4版的发行说明

## A.1 一般性变化

- 在整个RM中。增加了S32K11x系列设备的内容。
- AIPS章节。
  - 在寄存器MPRA中，增加了位域。MTR3, MTW3, 和MPL3

## A.2 关于本手册的变化

- 更新了观众，以反映S32K1xx的信息。

## A.3 介绍变化

- 图中S32K1xx产品系列比较：删除了 "系统 "类别下的实时时钟行，并更新了图例。

- 更新了S32K11x系列的章节；以反映S32K1xx系列的内容。
- 更新的S32K14x产品系列框图为S32K14x系列
- 在应用中
  - 更新的一般用途的应用
  - 增加了对S32K14x安全手册和应用说明 "使用S32K1xx EEPROM功能 "的参考。

## A.4 记忆图谱的变化

- 将SRAM内存图部分拆分为两个子部分 "S32K14x:SRAM内存地图 "和 "S32K11x:SRAM内存地图"
- 更新了 "读后写顺序和所需的内存操作序列化 "一节中的说明，以表明它只适用于S32K14x器件。
- 在 "私有外设总线（PPB）内存图 "部分增加了CM0+内核的PPB内存图。
- 将 "异化位带区域 "一节的标题改为 "CM4内核的异化位带区域"，以表明其仅适用于S32K14x器件。

## A.10 SIM卡的变化
## A.5 信号复用变化

- 在IO信号表部分
- 更新的SSS选项
- 增加了 "如果振荡器被启用，则启用..... "的说明。

## A.6 安全概述的变化

- 更新了该章以反映S32K1xx的信息。

## A.7 安全概述的变化

- 更新了该章以反映S32K1xx的信息。

## A.8 核心概述的变化

- 没有实质性的内容变化

## A.9 MCM的变化

### A.9.1 芯片特有的MCM信息变化

- 在表8-1中增加了S32K11x的寄存器复位值

### A.9.2 MCM的变化

- 保留/删除寄存器中的位域CF1 本地内存描述符寄存器（MCM_LMDRn）。

# A.10 SIM卡的变化

## A.10.1 芯片特定的SIM卡信息变化

- 更新了SIM寄存器位域实现部分，使之成为通用的。

## A.10.2 SIM卡的变化

- 删除了 PLATCGC[CGCDMA] 和 PLATCGC[CGMMPU] 的注释。

# A.11 港口变化

## A.11.1 芯片特定的端口信息变化

- 在PCR数量中增加了S32K11x系列的PCR数量
- 添加主题 数字输入滤波器配置顺序

## A.11.2 端口控制和中断（PORT）的变化

- 添加了 "全局中断控制 "部分。添加了 "全局中断控制低寄存器（PORT_GICLR）"和 "全局中断控制高寄存器（PORT_GICHR）"部分。

# A.12 GPIO的变化

## A.12.1 芯片特定的GPIO信息变化

- 添加了GPIO端口内存图部分的说明

## A.12.2 GPIO模块的变化

| |
|---|
| 没有实质性的内容变化。 |

# A.10 SIM卡的变化
# A.13 AXBS-Lite的变化

## A.13.1 特定芯片的AXBS-Lite信息变化

- 更新了横梁开关的主控分配，增加了S32K11x的信息
- 更新了横梁开关从属分配，增加了S32K11x的信息

## A.13.2 AXBS模块的变化

- 没有实质性的内容变化

# A.14 MPU的变化

## A.14.1 芯片特定的MPU信息变化

- 更新了区域描述符和从属端口配置以反映S32K1xx信息

## A.14.2 MPU模块变化

没有实质性的内容变化。

# A.15 AIPS-Lite的变化

## A.15.1 芯片特有的AIPS信息变化

- 更新了寄存器重置值以反映S32K1xx的信息。

## A.15.2 AIPS-Lite模块的变化

| |
|---|
| 没有实质性的内容变化。 |

# A.16 DMAMUX的变化

## A.16.1 芯片特定的DMAMUX信息变化

| |
|---|
| • 添加章节 通道数量 |

## A.16.2 DMAMUX模块的变化

| |
|---|
| 没有实质性的内容变化。 |

# A.17 eDMA的变化

## A.17.1 芯片特定的eDMA信息变化

| |
|---|
| • 增加了通道数以反映S32K1xx的信息。 |

## A.17.2 eDMA2模块的变化

| |
|---|
| • TCD控制和状态（TCD0_CSR - TCD15_CSR）。将TCD_CSR[ACTIVE]的访问从RW改为RO。 |
| • TCD 控制和状态（TCD0_CSR - TCD15_CSR）。将TCDn_CSR[DONE]的访问从RW改为W0C。 |
| • 分组的通道n优先级寄存器部分通道n优先级寄存器（DCHPRI3 - DCHPRI12）。<br>• 在 "章节下的寄存器复位值 "表中增加了从偏移量100h到10FH的寄存器的复位值。通道n优先级寄存器(DCHPRI3 - DCHPRI12) |

# A.18 TRGMUX的变化

## A.18.1 芯片特有的TRGMUX信息变化

- 从表中删除了SAI、ENET、QSPI的行和FTM6的重复行 芯片专用TRGMUX寄存器

## A.18.2 TRGMUX模块的变化

| |
|---|
| 没有实质性的内容变化。 |

# A.19 EWM的变化

## A.19.1 芯片特有的EWM信息变化

| |
|---|
| • 更新了特定芯片的EWM信息以反映S32K1xx信息。 |

## A.19.2 EWM模块的变化

| |
|---|
| 没有实质性的内容变化。 |

# A.20 EIM的变化

## A.20.1 芯片特有的EIM信息变化

| |
|---|
| • 更新了特定芯片的EIM信息以反映S32K1xx的信息。 |

## A.20.2 EIM的变化

| |
|---|
| 没有实质性的内容变化 |

# A.21 企业风险管理的变化

## A.21.1 特定芯片的ERM信息变化

| |
|---|
| • 更新了特定芯片的ERM信息以反映S32K1xx信息。 |

### A.21.2 企业风险管理的变化

- 没有实质性的内容变化

## A.22 WDOG的变化

### A.22.1 芯片特有的WDOG信息变化

- 没有实质性的内容变化

### A.22.2 WDOG模块的变化

增加了复位后禁用看门狗一节和代码示例

## A.23 CRC模块的变化

没有实质性的内容变化

## A.24 重置和启动变化

- 将表23-2中的NMI_DIS更名为NMI_PIN_CFG

## A.25 区域协调机制的变化

### A.25.1 特定芯片的RCM信息变化

- 没有实质性的内容变化

## A.25.2 区域协调机制的变化

- 在 "系统复位中断使能寄存器（RCM_SRIE）"部分，增加了这样的说明："这个寄存器只在芯片POR上复位，它不受其他复位类型的影响。"

# A.26 时钟分布变化

- 更新后的图25-1。
- 在表25-1中，更新了SIRCDIV1_CLK和SIRCDIV2_CLK的描述，以包括VLPS模式
- 在 "内部时钟要求 "和 "VLPR模式时钟："部分，更新了注释 "所有异步时钟包括VLPS模式"。
- 在模块时钟中，更新了 "外设模块时钟 "表中的QuadSPI和ENET的图表。

# A.27 变化

## A.27.1 芯片特有的SCG信息变化

- 更新了特定芯片的SCG信息：
  - 添加了 "SPLL不可用 "的说明。
  - 在 "振荡器限制 "一节中增加了 "IO信号表 "一节的参考。
  - 增加了 "系统时钟切换 "部分

## A.27.2 变化

- SOSC和SIRC在HSRUN模式下不可用。
- 对RCCR和CSR寄存器中的DIVCORE位域的有效复位值的信息进行了轻微的重新表述。

- 增加了DIVCORE位域的信息。如果选择SPLL作为系统时钟源，最大DIVCORE值为除以4。
- 删除了面向客户的版本中的非客户信息。

# A.28 PCC的变化

## A.28.1 芯片特定的PCC信息变化

- 删除了表格。S32K14x PCC寄存器；为S32K1xx系列增加了寄存器的可用性/不可用性的通用信息。

## A.28.2 PCC模块的变化

没有实质性的内容变化。

# A.29 存储器和存储接口的变化

- 在SRAM尺寸部分
  - 更新说明'没有ECC或访问错误是.....'
  - 增加了S32K11x的SRAM尺寸

# A.30 LMEM的变化

## A.30.1 特定芯片的LMEM信息变化

- 更新了特定芯片的LMEM信息以反映S32K11x的信息

## A.30.2 LMEM模块的变化

没有实质性的内容变化。

# A.31 MSCM的变化

## A.31.1 芯片特定的MSCM信息变化

- 在表31-1中
  - 更新以反映S32K1xx的信息
  - 更新了S32K146的寄存器复位值。

## A.31.2 MSCM的变化

- 更新了OCMDRn寄存器
- 寄存器CPxTYPE。修改了位域PERSONALITY的描述
- 寄存器CPxCFG0：修改了位域ICSZ的描述
- 寄存器CPxCFG1：修改了位域L2SZ的描述
- 寄存器CPxCFG2。修改了位域TMLSZ和TMUSZ的描述
- 寄存器CP0CFG0：修改了位域ICSZ和DCSZ的描述。
- 寄存器CP0CFG1：修改了位域L2SZ的描述。
- 寄存器CP0CFG2。修改了位域TMLSZ和TMUSZ的描述

# A.32 FMC的变化

## A.32.1 特定芯片的FMC变化

- 没有实质性的内容变化

## A.32.2 FMC的变化

- 没有实质性的内容变化

# A.33 FTFC的变化

## A.33.1 芯片特有的FTFC信息变化

- 更新了闪存类型和PFLASH读取分区的同时操作以反映S32K1xx的信息

## A.33.2 FTFC的变化

- 贯穿始终的小规模编辑更新
- 在 "介绍 "一节中，增加了 "配置特定信息"。
- 在方框图部分，增加了P/D-闪存映射（S32K144/512kB）；P/D-闪存映射（S32K146/1MB）；P/D-闪存映射（S32K148/2M）；P/D-闪存映射（S32K118/256kB）
- 在寄存器Program Flash Protection Registers (FPROT3 - FPROT0)的描述中，增加了'如果配置PFLash ... 2MB作为region_size。'
- 更新的表格计划部分指令FCCOB要求
- 在节中。程序节命令中，更新了'起始地址必须是不受保护的.....'
- 在EEPROM实现概述部分增加了 "中断模拟EEPROM... "的注意事项
- 在Plash program and erase部分增加了 "中断程序或擦除..... "的警告。
- 加密服务引擎（CSEc）部分增加了 "中断CSEc密钥更新... "的注意事项

# A.34 QSPI的变化

## A.34.1 芯片特定的QSPI信息变化

- 更新概述以反映S32K1xx的信息
- 在QuadSPI寄存器重置值中增加了芯片特定的寄存器重置值
- 更新了QuadSPI_SOCCR[SOCCFG]实现中SOCCFG[28]的位描述

## A.34.2 QuadSPI的变化

- 在QuadSPI_BUF0IND中，将寄存器描述改为："寄存器值应设置为所需的字节数减去8。例如，将BUF0IND设置为0可以得到8字节，1可以得到16字节，等等。缓冲区0的大小是BUF0IND+8和0之间的差。"改为 "寄存器的值应设置为所需的字节数。例如，将BUF0IND[31:3]设置为0可以得到0字节，1可以得到8字节等。缓冲区0的大小是BUF0IND和0的差值"。

- 在QuadSPI_RSER中，将第31位的访问从 "RU "更新为 "RW"。
- 在 QuadSPI_RSER 中，将第 30:29 位的访问从 "RU "更新为 "RW"。

- 在QuadSPI_MCR中，增加了SCLKCFG仅在禁用模式下可写。

# A.35 电源管理的变化

- 在电源模式描述部分，更新了 "芯片电源模式 "表，增加了 "HSRUN模式不可用 "的说明。
- 在可用的低功率模式下的模块操作中，更新了 "可用的低功率模式下的模块操作 "表。

# A.36 SMC的变化

- 没有实质性的内容变化

# A.37 PMC的变化

## A.37.1 针对芯片的PMC变化

- 没有实质性的内容变化

## A.37.2 PMC模块的变化

在位域LVDSC1[LVDRE]中添加了 "不受复位影响 "的注释。

# A.38 ADC配置变化

- 更新了实例化信息以反映S2K1xx的信息
- 在章节中 ADC通道的数量
  - 增加了表格。ADCx_SC1n[ADCH]字段配置
  - 增加了 "在某一特定的.....中无法使用的频道 "一句。
- 在ADC连接/通道分配部分
  - 删除 "此外，ADC0通道... "一句。
  - 增加了 "内部渠道可用性 "表

# A.39 ADC的变化

## A.39.1 芯片特定的ADC信息变化

- 没有实质性的内容变化

## A.39.2 ADC模块的变化

- ADC状态和控制寄存器1（SC1A - aSC1P）ADC状态和控制寄存器1（SC1Q - SC1AF）：
  - 将ADCH字段值011010b从 "温度传感器 "改为 "保留"。

- 特点：
  - 去掉了温度传感器的子弹。
- 框图：
  - 删除了温度传感器的输入。
- 更新部分。模拟接地（VSSA），电压基准选择，时钟选择和分频控制，转换控制，以及校准功能

- ADC偏移校正寄存器（OFS），ADC增益寄存器（G）：
  - 将字段的重置值改为 "未定义"。
  - 增加了一个脚注 "复位值从IFR中加载"。
- ADC状态和控制寄存器1（SC1A - aSC1P）ADC状态和控制寄存器1（SC1Q - SC1AF）：
  - 将 "1111（模块禁用）"改为 "1111（模块禁用）"。
  - 将 "在SC1A主动控制转换时写入SC1A中止当前转换 "改为 "在SC1A主动控制转换时向SC1A ADCH字段写入1111111中止当前转换"。
  - 在ADCH字段描述中。
    - 将 "外部输出通道0被选中 "改为 "外部通道0被选为输入"。
    - 将所有 "Exernal "的实例改为 "External"。
    - 将 "01001 "的实例改为 "001001"。
    - 将值 "011111 "从 "模块已禁用 "改为 "保留"。
    - 将 "内部通道16被选为输入 "改为 "外部通道16被选为输入"。
    - 将 "11XXXX - 模块已禁用 "改为 "11XXXX - 保留（字段值111111除外）"和 "1111111 - 模块已禁用"。
  - 将寄存器重置值从0x0000_001F改为0x0000_003F。

# A.40 CMP的变化

## A.40.1 针对芯片的CMP变化

- 没有实质性的内容变化

## A.40.2 CMP的变化

- 没有实质性的内容变化

# A.41 PDB的变化

## A.41.1 特定芯片的PDB信息变化

- 更新了实例化信息部分以反映S32K1xx的信息
- 更新后的表 41-3
- 更新了背对背确认连接，以显示每个PDB的所有通道上的PDB背对背操作支持。
- 在图中'例子。PDB1 CH0背对背链'。将CH0重命名为CH1

## A.41.2 PDB的变化

- 没有实质性的内容变化

# A.42 FTM的变化

## A.42.1 芯片特有的FTM信息变化

- 更新了实例化信息以反映S32K1xx信息

## A.42.2 FTM模块的变化

- 更新了通道输入滤波器和故障输入滤波器所增加的延迟。

# A.43 LPIT的变化

## A.43.1 特定芯片的LPIT信息变化

- 更新了实例化信息以反映S32K1xx信息

## A.43.2 LPIT模块的变化

| |
|---|
| • 在 "定时器模式 "部分，在现有的注释中增加了一个项目："基于外部触发器的定时器变化在实际的外部触发器断言后的4个外围时钟后生效（由于时钟同步、上升沿检测和定时器更新）。" |
| • 在MCR[SW_RST]软件复位位描述中，增加了 "软件必须等待4个外设时钟（用于时钟同步和复位传播），然后才能清除这个位 "的说明。 |

# A.44 LPTMR的变化

## A.44.1 芯片特有的LPTMR信息变化

| |
|---|
| • 更新了实例化信息以反映S32K1xx信息 |

## A.44.2 LPTMR模块的变化

| |
|---|
| 没有实质性的内容变化。 |

# A.45 RTC的变化

## A.45.1 芯片特有的RTC变化

| |
|---|
| • 没有实质性的内容变化 |

## A.45.2 RTC模块的变化

| |
|---|
| 没有实质性的内容变化。 |

# A.46 LPSPI的变化

## A.46.1 芯片特有的LPSPI信息变化

- 更新了实例化信息以反映S32K1xx信息

## A.46.2 LPSPI模块的变化

添加的部分。时钟环回

# A.47 LPI2C的变化

## A.47.1 芯片特有的LPI2C信息变化

- 更新了实例化信息以反映S32K1xx信息

## A.47.2 LPI2C模块的变化

没有实质性的内容变化。

# A.48 LPUART的变化

## A.48.1 芯片特有的LPUART信息变化

- 更新了实例化信息以反映S32K1xx信息

## A.48.2 LPUART模块的变化

| |
|---|
| 没有实质性的内容变化。 |

# A.49 FlexIO的变化

## A.49.1 特定芯片的FlexIO变化

- 没有实质性的内容变化

## A.49.2 FLEXIO模块变化

没有实质性的内容变化。

# A.50 FlexCAN的变化

## A.50.1 特定芯片的FlexCAN信息变化

- 更新实例化信息
  - 为了反映S32K1xx的信息
  - 更新了S32K146的MBs计数和CTRL2重置值
  - 增加了 "该功能仅适用于..... "的说明。

## A.50.2 FlexCAN3_FD模块的变化

- 在CAN FD帧中，将 "最大可配置比特率..... "一段中的8个时间量子和5 Mbit/s改为5个时间量子和8 Mbit/s。

- 增加了表50-3。
- 在DMA操作下的Rx FIFO中，增加了关于DMA的FD和非FD的区别的说明和链接。
- 在假装联网模式中，增加了关于FD和非FD假装联网的区别的说明和链接。
- 在Rx FIFO中，增加了关于FD和非FD的RX FIFO的区别的说明和链接。
- 在模块配置寄存器（MCR）中，更新了PNET_EN的位域描述。
- 在控制2寄存器（CTRL2）中，更新了位域描述TIMER_SRC。

# A.51 SAI的变化

## A.51.1 针对芯片的SAI变化

- 更新了SAI配置以反映S32K1xx信息

## A.51.2 SAI模块的变化

- 更新的SAI发送配置寄存器3 TCR3[TCE]（发送通道启用），TCR3[CFR]（通道FIFO复位）。
- 更新的SAI接收配置寄存器3 RCR3[RCE]（接收通道启用），RCR3[CFR]（通道FIFO复位）。

# A.52 ENETchanges

## A.52.1 特定芯片的ENET变化

- 在芯片特定的ENET信息部分增加了 "启用驱动力... "的说明

## A.52.2 ENET模块的变化

- 没有实质性的内容变化

# A.53 调试变化

- 没有实质性的内容变化

# A.54 JTAGC的变化

## A.54.1 芯片特定的JTAGC信息变化

- 没有实质性的内容变化

## A.54.2 JTAGC模块的变化

- 在IEEE 1149.1-2001定义的测试模式中，更新并重写了 "单比特旁路寄存器移位阶段被启用，用于... "的句子。
- 在旁路寄存器中，更新并重写了以 "旁路寄存器是一个单比特移位寄存器... "开头的句子。
-




*如何联系我们。*

**主页。**
nxp.com

**网络支持。**
nxp.com/support



本文件中的信息仅用于使系统和软件实施者能够使用恩智浦的产品。本文件没有明示或暗示授予版权许可，以根据本文件的信息设计或制造任何集成电路。恩智浦保留对本文中任何产品进行修改的权利，恕不另行通知。

恩智浦对其产品在任何特定用途上的适用性不作任何保证、陈述或担保，也不承担因应用或使用任何产品或电路而产生的任何责任，并特别声明不承担任何和所有责任，包括但不限于间接或附带的损害。恩智浦数据表和/或规格中可能提供的 "典型 "参数在不同的应用中可能而且确实存在差异，而且实际性能可能随时间变化。所有操作参数，包括 "典型 "参数，必须由客户的技术专家针对每个客户的应用进行验证。恩智浦不传达其专利权或其他权利下的任何许可。恩智浦根据标准的销售条款和条件销售产品，可在以下地址找到：nxp.com/SalesTermsandConditions。

恩智浦，恩智浦标志，恩智浦SECURE CONNECTIONS FOR A SMARTER WORLD, COOLFLUX, EMBRACE, GREENCHIP, HITAG, I2C BUS, ICODE, JCOP, LIFE VIBES, MIFARE, MIFARE CLASSIC,MIFARE DESFire, MIFARE PLUS, MIFARE FLEX, MANTIS, MIFARE ULTRALIGHT, MIFARE4MOBILE, MIGLO, NTAG, ROADLINK, SMARTLX, SMARTMX, STARPLUG, TOPFET, TRENCHMOS、UCODE、飞思卡尔、飞思卡尔标志、AltiVec、C-5、CodeTest、CodeWarrior、ColdFire、ColdFire+、C-Ware、节能解决方案标志、Kinetis、Layerscape、MagniV、mobileGT、PEG、PowerQUICC、处理器专家、QorIQ。QorIQ Qonverge, Ready Play, SafeAssure, the SafeAssure logo, StarCore, Symphony, VortiQa, Vybrid, Airfast, BeeKit, BeeStack, CoreNet, Flexis, MXC, Platform in a Package, QUICC Engine, SMARTMOS, Tower, TurboLink, 和UMEMS是恩智浦B.V.所有其他产品或服务名称是其各自所有者的财产。ARM、AMBA、ARM Powered、Artisan、Cortex、Jazelle、Keil、SecurCore、Thumb、TrustZone和μVision是ARM Limited（或其子公司）在欧盟和/或其他地区的注册商标。ARM7、ARM9、ARM11、big.LITTLE、CoreLink、CoreSight、DesignStart、Mali、mbed、NEON、POP、Sensinode、Socrates、ULINK和Versatile是ARM Limited（或其子公司）在欧盟和/或其他地区的商标。保留所有权利。Oracle和Java是Oracle和/或其附属公司的注册商标。Power Architecture和Power.org字样以及Power和Power.org标识和相关标识是Power.org授权的商标和服务标识。

© 2015-2017 NXP B.V.



文件编号S32K1XXRM
第4次修订，2017年6月

