CPUID Specification

Publication # **25481**	Revision: **2.28**
Issue Date: **April 2008**	

Advanced Micro Devices 

Trademarks

Table of Contents

Revision History

Date	Rev	Description
April 2008	2.28	• 3.1 [Legacy Method] on page 23: Clarified. • CPUID Fn0000_0001_ECX[SSE41]: Added. • CPUID Fn0000_0001_ECX[SSSE3]: Added. • CPUID Fn8000_0001_ECX[SSE5]: Added. • CPUID Fn8000_0001_ECX[IBS]: Added. • CPUID Fn8000_0008_EAX[GuestPhysAddrSize]: Added. • CPUID Fn8000_0008_EAX[PhysAddrSize]: Updated. • CPUID Fn8000_000A_EDX[Ssse3Sse5Dis]: Added.
July 2007	2.26	• CPUID Fn0000_0001_ECX[Monitor]: Added. • CPUID Fn0000_0001_ECX[POPCNT]: Added. • CPUID Fn0000_000[4:2]: Added as reserved. • CPUID Fn0000_0005 [MONITOR/MWAIT] on page 13: Added. • CPUID Fn0000_0005_ECX[IBE, EMX]: Added. • CPUID Fn8000_0001_EBX[PkgType[3:0]]: Added. • CPUID Fn8000_0001_ECX[WDT]: Added. • CPUID Fn8000_0001_ECX[SKINIT]: Added. • CPUID Fn8000_0001_ECX[OSVW]: Added. • CPUID Fn8000_0001_ECX[3DNowPrefetch]: Added. • CPUID Fn8000_0001_ECX[MisAlignSse]: Added. • CPUID Fn8000_0001_ECX[SSE4A]: Added. • CPUID Fn8000_0001_ECX[ABM]: Added. • CPUID Fn8000_0001_ECX[ExtApicSpace]: Added. • CPUID Fn8000_0001_EDX[Page1GB]: Added. • CPUID Fn8000_0006_EDX[L3Size, L3Assoc, L3LinesPerTag, L3LineSize]: Added. • CPUID Fn8000_0006: Table 1: Added additional associativity definitions. • CPUID Fn8000_0007_EDX[HwPState]: Added Hardware P-State control. • CPUID Fn8000_0007_EDX[100MhzSteps]: Added. • CPUID Fn8000_000A_EDX[NRIPS]: Added. • CPUID Fn8000_000A_EDX[SVML]: Added. • CPUID Fn8000_000A_EDX[LbrVirt]: Added LBR virtualization. • CPUID Fn8000_000A_EDX[NP]: Added. • CPUID Fn8000_0019 [TLB 1GB Page Identifiers] on page 21: Added. • CPUID Fn8000_001A [Performance Optimization Identifiers] on page 22: Added. • CPUID Fn8000_001B [Instruction Based Sampling Identifiers] on page 22: Added.
January 2006	2.18	• Renamed CPUID Fn8000_0007_EDX[8] from TscPStateInvariant to TscInvariant. • Added CPUID Fn8000_0008_ECX[ApicIdCoreIdSize[3:0]].

Date	Rev	Description

September 2005	2.16	• Reformatted document for clarity.
		• Moved the chapter titled “Programming The Processor Name String” to the processor revision guide.
		• Added definition for HTT, CmpLegacy, and LogicalProcessorCount for multi-threading.
		  • See “Legacy Method” on page 23.
		  • See CPUID Fn8000_0001_ECX for CmpLegacy.
		  • See CPUID Fn0000_0001_EBX for LogicalProcessorCount.
		  • See CPUID Fn0000_0001_EDX for HTT.
		• Extended BrandID (BrandId[15:12]) definition to CPUID Fn8000_0001_EBX.
		• Added SVM. See CPUID Fn8000_0001_ECX.
		• Added SVM definition to CPUID Fn8000_000A.
		• Added CMPXCHG16B. See CPUID Fn0000_0001_ECX.
		• Added AltMovCr8. See CPUID Fn8000_0001_ECX.
		• Added LahfSahf. See CPUID Fn8000_0001_ECX.
		• Added RDTSCP. See CPUID Fn8000_0001_EDX.

See revision 2.15 for the change history prior to rev 2.16.		

1 Overview

This document specifies the CPUID instruction functions and return values in the EAX, EBX, ECX, and EDX registers, for all AMD processors of family 0Fh or greater. The architectural definition of the CPUID instruction is also documented in the section titled “CPUID” in the *AMD64 Architectural Programmer's Manual Volume 3: General-Purpose and System Instructions*, order #24594.

1.1 Reference Documents

The following documents provide background information:

- *AMD64 Architecture Programmer’s Manual Volume 1: Application Programming*, order# 24592.
- *AMD64 Architecture Programmer’s Manual Volume 2: System Programming*, order# 24593.
- *AMD64 Architecture Programmer’s Manual Volume 3: General Purpose and System Instructions*, order# 24594.
- *AMD64 Architecture Programmer’s Manual Volume 4: 128-Bit Media Instructions*, order# 26568.
- *AMD64 Architecture Programmer’s Manual Volume 5: 64-Bit Media and x87 Floating-Point Instructions*, order# 26569.
- *128-Bit SSE5 Instruction Set and Supplemental 64-Bit Media Instructions*, order# 43479.
- *AMD64 Architecture Programmer’s Manual Documentation Updates*, order# 33633
- BIOS and Kernel Developer’s Guide (BKDG) for the specific result value for each of the registers affected by the CPUID instruction for each function. The order number varies by processor family and sometimes by processor model.
- *AMD Processor Recognition Application Note*, order# 20734, for the definition of CPUID for processors belonging to family 0Fh or less.
- The appropriate revision guide for your target processor describes the process of programming the processor name string.

1.2 Conventions

The following conventions are used in this document:

- The convention for referring to CPUID functions is CPUID FnXXXX_XXXX, where the CPUID function is XXXX_XXXXh; e.g., CPUID Fn0000_0001.
- The convention for referring to CPUID capability fields is CPUID FnXXXX_XXXX_RRR[FieldName], where RRR is the register (EAX, EBX, ECX, EDX) and FieldName is the name of the capability field; e.g., CPUID Fn8000_0001_EDX[SVM].
- Unless otherwise specified, the 1-bit feature fields are encoded as 1 = Feature is supported by the processor; 0 = Feature is not supported by the processor.
- References to the *AMD64 Architecture Programmer’s Manual* are abbreviated as APM*n*, where *n* specifies the volume, from 1 to 5.
- The 8-bit family of a processor (Family[7:0]) is determined by CPUID Fn0000_0001_EAX[ExtendedFamily,BaseFamily].

1.2.1 Numbering

- **Binary numbers**. Binary numbers are indicated by appending a “b” at the end; e.g., 0110b.
- **Decimal numbers**. Unless specified otherwise, all numbers are decimal. This rule does not apply to the register mnemonics; register mnemonics all utilize hexadecimal numbering.
- **Hexadecimal numbers**. Hexadecimal numbers are indicated by appending an “h” to the end; e.g., 45f8h.
- **Underscores in numbers**. Underscores are used to break up numbers to make them more readable. They do not imply any operation; e.g., 0110_1100b.

1.2.2 Arithmetic and Logical Operators

- {} Curly brackets are used to indicate a group of concatenated bits. Each set of bits is separated by a comma, e.g., {10b,01b,1b}=10011b.
- | Logical OR operator.
- & Logical AND operator.

1.3 Definitions

The following definitions are used in this document:

- **APM*n***. Abbreviation for *AMD64 Architecture Programmer’s Manual: Volume* n.
- **APMU**. Abbreviation for *AMD64 Architecture Programmer’s Manual: Documentation Updates*, order# 33633.
- **BKDG**. BIOS and Kernel Developer’s Guide
- **CMP**. Chip multi-processing. Refers to processors that include multiple CPU cores.
- **CPU Core**. Executes x86 instructions and contains a set of MSRs and APIC registers.
- **DW or Doubleword**. Double word. A 32-bit value.
- **Family**. An 8-bit value that identifies one or more processors as belonging to a group that possess some common definition for software or hardware purposes. See CPUID Fn0000_0001_EAX.
- **GB or Gbyte**. Gigabyte; 1,024 Mbytes.
- **HTC**. Hardware thermal control.
- **KB or Kbyte**. Kilobyte; 1024 bytes.
- **MB or Mbyte**. Megabyte; 1024 Kbytes.
- **Model**. Model specifies one instance of a processor family. See CPUID Fn0000_0001_EAX.
- **MSR**. Model specific register. The CPU includes several MSRs for general configuration and control.
- **NB**. Northbridge. The transaction routing block of the processor.
- **Processor**. A single package that contains one or more CPU cores.
- **QW or Quadword**. Quad word. A 64-bit value.
- **OW or Octword**. Eight word. A 128-bit value.
- **RAZ**. Read as zero. Writes are ignored.
- **Reserved**. Field is reserved for future use. Software may not depend on the state of reserved fields.
- **STC**. Software thermal control.
- **SVM**. Secure virtual machine.
- **Thread**. One architectural context for instruction execution.

1.4 CPUID Function Selection

The CPUID instruction provides processor feature capabilities and configuration information. This information is accessed by (1) loading the function number into EAX, (2) executing the CPUID instruction, and (3) reading the results stored in EAX, EBX, ECX, and EDX. In the following sections, the phrase *CPUID function X* or *CPUID FnX* refers to the CPUID instruction when EAX is preloaded with X.

1.5 Standard, Extended, and Undefined Functions

The CPUID instruction supports two sets or ranges of functions, standard and extended.

- The smallest function number of the standard function range is Fn0000_0000. The largest function number of the standard function range, for a particular implementation, is returned in CPUID Fn0000_0000_EAX.
- The smallest function number of the extended function range is Fn8000_0000. The largest function number of the extended function range, for a particular implementation, is returned in CPUID Fn8000_0000_EAX.

Functions that are neither standard nor extended are undefined and should not be relied upon.

2 CPUID Function Specification

This chapter defines each of the supported CPUID functions, both standard and extended.

CPUID Fn0000_0000 Processor Vendor and Largest Standard Function Number

The values returned in EBX, ECX, and EDX for CPUID Fn0000_0000 are the same values returned in EBX, ECX, and EDX for CPUID Fn8000_0000.

Register	Bits	Description
EAX	31:0	**LFuncStd: largest standard function**. The largest CPUID standard-function input value supported by the processor implementation. See “Standard, Extended, and Undefined Functions” on page 9.
EBX, ECX, EDX	31:0	**Vendor**. The twelve 8-bit ASCII character codes that form the string “AuthenticAMD”. EBX=6874_7541h “h t u A”, ECX=444D_4163h “D M A c”, EDX=6974_6E65h “i t n e”.

CPUID Fn0000_0001_EAX Family, Model, Stepping Identifiers

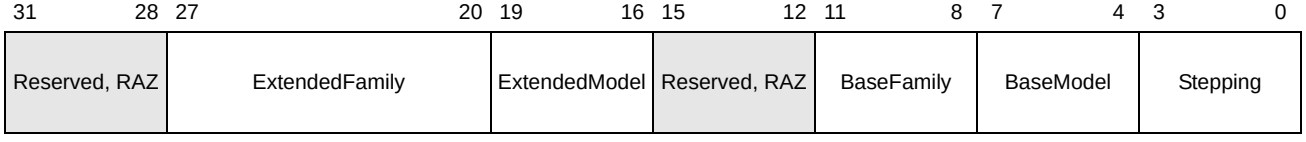
The value returned in EAX is the processor identification signature and is identical for CPUID Fn0000_0001 and CPUID Fn8000_0001. This function is an identical copy of CPUID Fn8000_0001_EAX. Reserved fields should be masked before using the value of EAX for processor identification purposes. Three values are used by software to identify a processor: Family, Model, and Stepping.

The processor *Family* identifies one or more processors as belonging to a group that possesses some common definition for software or hardware purposes. The *Model* specifies one instance of a processor family. The *Stepping* identifies a particular version of a specific model. Therefore, Family, Model and Stepping, when taken together, form a unique identification or signature for a processor.

The **Family** is an 8-bit value and is defined as: **Family[7:0] =** ({0000b,BaseFamily[3:0]} + ExtendedFamily[7:0]). For example, if BaseFamily[3:0] = 0Fh and ExtendedFamily[7:0] = 01h, then Family[7:0] = 10h. If BaseFamily[3:0] is less than 0Fh then ExtendedFamily[7:0] is reserved and Family is equal to BaseFamily[3:0].

Model is an 8-bit value and is defined as: **Model[7:0]** = {ExtendedModel[3:0],BaseModel[3:0]}. For example, if ExtendedModel[3:0] = 0Eh and BaseModel[3:0] = 08h, then Model[7:0] = E8h. If BaseFamily[3:0] is less than 0Fh then ExtendedModel[3:0] is reserved and Model is equal to BaseModel[3:0].

Stepping is analogous to a revision number.



Bits	Description
31:28	Reserved.
27:20	**ExtendedFamily: processor extended family**. See above for definition of Family[7:0].
19:16	**ExtendedModel: processor extended model**. See above for definition of Model[7:0].
15:12	Reserved.
11:8	**BaseFamily: base processor family**. See above for definition of Family[7:0].
7:4	**BaseModel: base processor model**. See above for definition of Model[7:0].
3:0	**Stepping: processor stepping**. Processor stepping (revision) for a specific model.

CPUID Fn0000_0001_EBX LocalApicId, LogicalProcessorCount, CLFlush

This function returns miscellaneous information regarding the processor brand, the number of logical threads per processor socket, the CLFLUSH instruction and APIC.

Bits	Description
31:24	**LocalApicId: Initial local APIC physical ID**. The 8-bit value assigned to the local APIC physical ID register at power-up. Some of the bits of LocalApicId represent the CPU core within a processor and other bits represent the processor ID. See the “APIC ID Register” in the processor BKDG for details.
23:16	**LogicalProcessorCount: logical processor count**. If CPUID Fn0000_0001_EDX[HTT] = 1 then LogicalProcessorCount is the number of threads per CPU core times the number of CPU cores per processor. If CPUID Fn0000_0001_EDX[HTT] = 0 then LogicalProcessorCount is reserved. See “Legacy Method” on page 23.
15:8	**CLFlush: CLFLUSH size**. Specifies the size of a cache line in quadwords flushed by the CLFLUSH instruction. See “CLFLUSH” in APM3.
7:0	**8BitBrandId: 8-bit brand ID**. This field, in conjunction with CPUID Fn8000_0001_EBX[BrandId], is used by the BIOS to generate the processor name string. See the appropriate processor revision guide for how to program the processor name string.

CPUID Fn0000_0001_ECX Feature Identifiers

This function contains the following miscellaneous feature identifiers.

Bits	Description
31	RAZ.
30:24	Reserved.
23	**POPCNT: POPCNT instruction**. See “POPCNT” in APM3.
22:20	Reserved.
19	**SSE41: SSE4.1 instruction support**. See “64-Bit Media Instructions” in the APM4.
18:14	Reserved.

Bits	Description
13	**CMPXCHG16B: CMPXCHG16B instruction**. See “CMPXCHG16B” in APM3.
12:10	Reserved.
9	**SSSE3: supplemental SSE3 extensions**.
8:4	Reserved.
3	**MONITOR: MONITOR/MWAIT instructions**. See “MONITOR” and “MWAIT” in APM3.
2:1	Reserved.
0	**SSE3: SSE3 extensions**. See Appendix D “Instruction Subsets and CPUID Feature Sets” in APM3 for the list of instructions covered by the SSE3 feature bit. See APM4 for the definition of the SSE3 instructions.

CPUID Fn0000_0001_EDX Feature Identifiers

This function contains the following miscellaneous feature identifiers.

Bits	Description
31:29	Reserved.
28	**HTT: hyper-threading technology**. Indicates either that there is more than one thread per CPU core or more than one CPU core per processor. See “Legacy Method” on page 23.
27	Reserved.
26	**SSE2: SSE2 extensions**. See Appendix D “CPUID Feature Sets” in APM3.
25	**SSE: SSE extensions**. See Appendix D “CPUID Feature Sets” in APM3 appendix and “64-Bit Media Programming” in APM1.
24	**FXSR: FXSAVE and FXRSTOR instructions**. See “FXSAVE” and “FXRSTOR” in APM4.
23	**MMX: MMX™ instructions**. See Appendix D “CPUID Feature Sets” in APM3 and “128-Bit Media and Scientific Programming” in APM1.
22:20	Reserved.
19	**CLFSH: CLFLUSH instruction**. See “CLFLUSH” in APM3.
18	Reserved.
17	**PSE36: page-size extensions**. The PDE[20:13] supplies physical address [39:32]. See “Page Translation and Protection” in APM2.
16	**PAT: page attribute table**. See “Page-Attribute Table Mechanism” in APM2.
15	**CMOV: conditional move instructions**. See “CMOV”, “FCMOV” in APM3.
14	**MCA: machine check architecture**. See “Machine Check Mechanism” in APM2.
13	**PGE: page global extension**. See “Page Translation and Protection” in APM2.
12	**MTRR: memory-type range registers**. See “Page Translation and Protection” in APM2.
11	**SysEnterSysExit: SYSENTER and SYSEXIT instructions**. See “SYSENTER”, “SYSEXIT“ in APM3.
10	Reserved.

Bits	Description
9	**APIC: advanced programmable interrupt controller**. Indicates APIC exists and is enabled. See "Exceptions and Interrupts" in APM2.
8	**CMPXCHG8B: CMPXCHG8B instruction**. See "CMPXCHG8B" in APM3.
7	**MCE: Machine check exception**. See "Machine Check Mechanism" in APM2.
6	**PAE: physical-address extensions**. Indicates support for physical addresses ≥ 32b. Number of physical address bits above 32b is implementation specific. See "Page Translation and Protection" in APM2.
5	**MSR: AMD model-specific registers**. Indicates support for AMD model-specific registers (MSRs), with RDMSR and WRMSR instructions. See "Model Specific Registers" in APM2.
4	**TSC: time stamp counter**. RDTSC and RDTSCP instruction support. See "Debug and Performance Resources" in APM2.
3	**PSE: page-size extensions**. See "Page Translation and Protection" in APM2.
2	**DE: debugging extensions**. See "Debug and Performance Resources" in APM2.
1	**VME: virtual-mode enhancements**. CR4.VME, CR4.PVI, software interrupt indirection, expansion of the TSS with the software, indirection bitmap, EFLAGS.VIF, EFLAGS.VIP. See "System Resources" in APM2.
0	**FPU: x87 floating point unit on-chip**. See "x87 Floating Point Programming" in APM1.

CPUID Fn0000_000[4:2] Reserved

The three functions from CPUID Fn0000_0002 to CPUID Fn0000_0004 are reserved.

CPUID Fn0000_0005 MONITOR/MWAIT

This function contains the feature identifiers for the MONITOR and MWAIT instructions. See "MONITOR" and "MWAIT" in APM3.

Register	Bits	Description
EAX	31:16	Reserved.
EAX	15:0	**MonLineSizeMin: smallest monitor-line size in bytes**.
EBX	31:16	Reserved.
EBX	15:0	**MonLineSizeMax: largest monitor-line size in bytes**.
ECX	31:2	Reserved.
ECX	1	**IBE**. Indicates MWAIT can use ECX bit 0 to allow interrupts to cause an exit from the monitor event pending state, even if EFLAGS.IF=0.
ECX	0	**EMX**: Indicates enumeration MONITOR/MWAIT extensions are supported.
EDX	31:0	Reserved.

CPUID Fn8000_0000 Processor Vendor and Largest Extended Function Number

The values returned in EBX, ECX, and EDX for CPUID Fn8000_0000 are the same values returned in EBX, ECX, and EDX for CPUID Fn0000_0000.

Register	Bits	Description
EAX	31:0	**LFuncExt: largest extended function**. The largest CPUID extended-function input value supported by the processor implementation. See “Standard, Extended, and Undefined Functions” on page 9.
EBX, ECX, EDX	31:0	**Vendor**. The twelve 8-bit ASCII character codes to create the string “AuthenticAMD”. EBX=6874_7541h “h t u A”, ECX=444D_4163h “D M A c”, EDX=6974_6E65h “i t n e”.

CPUID Fn8000_0001_EAX AMD Family, Model, Stepping

Same as CPUID Fn0000_0001_EAX.

CPUID Fn8000_0001_EBX BrandId Identifier

This function returns the extended brand ID field.

Bits	Description
31:28	**PkgType: package type**. If (Family[7:0] >= 10h) then the definition of PkgType is contained in the processor BKDG. If (Family[7:0]<10h) then the definition of PkgType is reserved.
27:16	Reserved.
15:0	**BrandId: brand ID**. This field, in conjunction with CPUID Fn0000_0001_EBX[8BitBrandId], is used by BIOS to generate the processor name string. See your processor revision guide for how to program the processor name string.

CPUID Fn8000_0001_ECX Feature Identifiers

This function contains the following miscellaneous feature identifiers.

Bits	Description
31:14	Reserved.
13	**WDT: watchdog timer support**.
12	**SKINIT: SKINIT and STGI are supported, independent of the value of MSRC000_0080[SVME]**.
11	**SSE5: SSE5 Instruction support**. See the *128-Bit SSE5 Instruction Set and Supplemental 64-Bit Media Instructions*, order# 43479.
10	**IBS: instruction based sampling**. See “Instruction Based Sampling (IBS)” in APMU.

Bits	Description
9	**OSVW: OS visible workaround**. Indicates OS-visible workaround support. See “OS Visible Workaround (OSVW) Information” in APM2.
8	**3DNowPrefetch: PREFETCH and PREFETCHW instruction support**. See “PREFETCH” and “PREFETCHW” in APM3.
7	**MisAlignSse: misaligned SSE mode**. See “Misaligned Access Support Added for SSE Instructions” in APM1.
6	**SSE4A: EXTRQ, INSERTQ, MOVNTSS, and MOVNTSD instruction support**. See “EXTRQ”, “INSERTQ”, “MOVNTSS”, and “MOVNTSD” in APM4.
5	**ABM: advanced bit manipulation**. LZCNT instruction support. See “LZCNT” in APM3.
4	**AltMovCr8: LOCK MOV CR0 means MOV CR8**. See “MOV(CRn)” in APM3.
3	**ExtApicSpace: extended APIC space**. This bit indicates the presence of extended APIC register space starting at offset 400h from the “APIC Base Address Register,” as specified in the BKDG..
2	**SVM: secure virtual machine**. See “Secure Virtual Machine” in APM2.
1	**CmpLegacy: core multi-processing legacy mode**. See “Legacy Method” on page 23.
0	**LahfSahf: LAHF and SAHF instruction support in 64-bit mode**. See “LAHF” and “SAHF” in APM3.

CPUID Fn8000_0001_EDX Feature Identifiers

This function contains the following miscellaneous feature identifiers.

Bits	Description
31	**3DNow: 3DNow!™ instructions**. See Appendix D “Instruction Subsets and CPUID Feature Sets” in APM3.
30	**3DNowExt: AMD extensions to 3DNow! instructions**. See Appendix D “Instruction Subsets and CPUID Feature Sets” in APM3.
29	**LM: long mode**. See “Processor Initialization and Long-Mode Activation” in APM2.
28	Reserved.
27	**RDTSCP: RDTSCP instruction**. See “RDTSCP” in APM3.
26	**Page1GB: 1-GB large page support**. See “1-GB Paging Support” in APM2.
25	**FFXSR: FXSAVE and FXRSTOR instruction optimizations**. See “FXSAVE” and “FXRSTOR” in APM4.
24	**FXSR: FXSAVE and FXRSTOR instructions**. Same as CPUID Fn0000_0001_EDX[FXSR].
23	**MMX: MMX™ instructions**. Same as CPUID Fn0000_0001_EDX[MMX].
22	**MmxExt: AMD extensions to MMX instructions**. See Appendix D “Instruction Subsets and CPUID Feature Sets” in APM3 and “128-Bit Media and Scientific Programming” in APM1.
21	Reserved.
20	**NX: no-execute page protection**. See “Page Translation and Protection” in APM2.
19:18	Reserved.

Bits	Description
17	**PSE36: page-size extensions**. Same as CPUID Fn0000_0001_EDX[PSE36].
16	**PAT: page attribute table**. Same as CPUID Fn0000_0001_EDX[PAT].
15	**CMOV: conditional move instructions**. Same as CPUID Fn0000_0001_EDX[CMOV].
14	**MCA: machine check architecture**. Same as CPUID Fn0000_0001_EDX[MCA].
13	**PGE: page global extension**. Same as CPUID Fn0000_0001_EDX[PGE].
12	**MTRR: memory-type range registers**. Same as CPUID Fn0000_0001_EDX[MTRR].
11	**SysCallSysRet: SYSCALL and SYSRET instructions**. See “SYSCALL” and “SYSRET” in APM3.
10	Reserved.
9	**APIC: advanced programmable interrupt controller**. Same as CPUID Fn0000_0001_EDX[APIC].
8	**CMPXCHG8B: CMPXCHG8B instruction**. Same as CPUID Fn0000_0001_EDX[CMPXCHG8B].
7	**MCE: machine check exception**. Same as CPUID Fn0000_0001_EDX[MCE].
6	**PAE: physical-address extensions**. Same as CPUID Fn0000_0001_EDX[PAE].
5	**MSR: AMD model-specific registers**. Same as CPUID Fn0000_0001_EDX[MSR].
4	**TSC: time stamp counter**. Same as CPUID Fn0000_0001_EDX[TSC].
3	**PSE: page-size extensions**. Same as CPUID Fn0000_0001_EDX[PSE].
2	**DE: debugging extensions**. Same as CPUID Fn0000_0001_EDX[DE].
1	**VME: virtual-mode enhancements**. Same as CPUID Fn0000_0001_EDX[VME].
0	**FPU: x87 floating-point unit on-chip**. Same as CPUID Fn0000_0001_EDX[FPU].

CPUID Fn8000_000[4:2] Processor Name String Identifier

The three extended functions from Fn8000_0002 to Fn8000_0004 are initialized to and return a null terminated ASCII string up to 48 characters in length corresponding to the processor name. (The 48 character maximum includes the null character.) The 48 character sequence is ordered first to last as follows:

Fn8000_0002[EAX[7:0],..., EAX[31:24], EBX[7:0],..., EBX[31:24], ECX[7:0],..., ECX[31:24], EDX[7:0],..., EDX[31:24]],
Fn8000_0003[EAX[7:0],..., EAX[31:24], EBX[7:0],..., EBX[31:24], ECX[7:0],..., ECX[31:24], EDX[7:0],..., EDX[31:24]],
Fn8000_0004[EAX[7:0],..., EAX[31:24], EBX[7:0],..., EBX[31:24], ECX[7:0],..., ECX[31:24], EDX[7:0],..., EDX[31:24]].

The processor name string must be programmed by the BIOS during system initialization. See your processor revision guide for information about how to program and display the processor name string.

CPUID Fn8000_0005 L1 Cache and TLB Identifiers

This function contains the processor's first level cache and TLB characteristics for each CPU core.

The *associativity* fields are encoded as follows:

00h: Reserved.
01h: Direct mapped.
02h-FEh: Associativity. (e.g., 04h = 4-way associative.)
FFh: Fully associative.

Register	Bits	Description
EAX	31:24	**L1DTlb2and4MAssoc**. Data TLB associativity for 2-MB and 4-MB pages.
EAX	23:16	**L1DTlb2and4MSize**. Data TLB number of entries for 2-MB and 4-MB pages. The value returned is for the number of entries available for the 2-MB page size; 4-MB pages require two 2-MB entries, so the number of entries available for the 4-MB page size is one-half the returned value.
EAX	15:8	**L1ITlb2and4MAssoc**. Instruction TLB associativity for 2-MB and 4-MB pages.
EAX	7:0	**L1ITlb2and4MSize**. Instruction TLB number of entries for 2-MB and 4-MB pages. The value returned is for the number of entries available for the 2-MB page size; 4-MB pages require two 2-MB entries, so the number of entries available for the 4-MB page size is one-half the returned value.
EBX	31:24	**L1DTlb4KAssoc**. Data TLB associativity for 4 KB pages.
EBX	23:16	**L1DTlb4KSize**. Data TLB number of entries for 4 KB pages.
EBX	15:8	**L1ITlb4KAssoc**. Instruction TLB associativity for 4 KB pages.
EBX	7:0	**L1ITlb4KSize**. Instruction TLB number of entries for 4 KB pages.
ECX	31:24	**L1DcSize**. L1 data cache size in KB.
ECX	23:16	**L1DcAssoc**. L1 data cache associativity.
ECX	15:8	**L1DcLinesPerTag**. L1 data cache lines per tag.
ECX	7:0	**L1DcLineSize**. L1 data cache line size in bytes.
EDX	31:24	**L1IcSize**. L1 instruction cache size KB.
EDX	23:16	**L1IcAssoc**. L1 instruction cache associativity.
EDX	15:8	**L1IcLinesPerTag**. L1 instruction cache lines per tag.
EDX	7:0	**L1IcLineSize**. L1 instruction cache line size in bytes.

CPUID Fn8000_0006 L2/L3 Cache and L2 TLB Identifiers

This function contains the processor's second level cache and TLB characteristics for each CPU core. The EDX register contains the processor's third level cache characteristics that are shared by all CPU cores of the processor.

The *associativity* fields are encoded as follows:

Table 1: L2/L3 Cache and TLB Associativity Field Definition

Associativity [3:0]	Definition
0h	L2/L3 cache or TLB is disabled.
1h	Direct mapped.
2h	2-way associative.
4h	4-way associative.
6h	8-way associative.
8h	16-way associative.
Ah	32-way associative.
Bh	48-way associative.
Ch	64-way associative.
Dh	96-way associative.
Eh	128-way associative.
Fh	Fully associative.
All other encodings are reserved.	

Register	Bits	Description
EAX	31:28	**L2DTlb2and4MAssoc**. L2 data TLB associativity for 2-MB and 4-MB pages. (See Table 1)
EAX	27:16	**L2DTlb2and4MSize**. L2 data TLB number of entries for 2-MB and 4-MB pages. The value returned is for the number of entries available for the 2-MB page size; 4-MB pages require two 2-MB entries, so the number of entries available for the 4-MB page size is one-half the returned value.
EAX	15:12	**L2ITlb2and4MAssoc**. L2 instruction TLB associativity for 2-MB and 4-MB pages. (See Table 1.)
EAX	11:0	**L2ITlb2and4MSize**. L2 instruction TLB number of entries for 2-MB and 4-MB pages. The value returned is for the number of entries available for the 2-MB page size; 4-MB pages require two 2-MB entries, so the number of entries available for the 4-MB page size is one-half the returned value.
EBX	31:28	**L2DTlb4KAssoc**. L2 data TLB associativity for 4-KB pages. (See Table 1.)
EBX	27:16	**L2DTlb4KSize**. L2 data TLB number of entries for 4-KB pages.
EBX	15:12	**L2ITlb4KAssoc**. L2 instruction TLB associativity for 4-KB pages. (See Table 1.)
EBX	11:0	**L2ITlb4KSize**. L2 instruction TLB number of entries for 4-KB pages.
ECX	31:16	**L2Size**. L2 cache size in KB.
ECX	15:12	**L2Assoc**. L2 cache associativity. (See Table 1.)
ECX	11:8	**L2LinesPerTag**. L2 cache lines per tag.
ECX	7:0	**L2LineSize**. L2 cache line size in bytes.

Register	Bits	Description
EDX	31:18	**L3Size**. Specifies the L3 cache size is within the following range: (L3Size[31:18] * 512KB) ≤ L3 cache size < ((L3Size[31:18]+1) * 512KB).
EDX	17:16	Reserved.
EDX	15:12	**L3Assoc**. L3 cache associativity. (See Table 1)
EDX	11:8	**L3LinesPerTag**. L3 cache lines per tag.
EDX	7:0	**L3LineSize**. L3 cache line size in bytes.

CPUID Fn8000_0007 Advanced Power Management Information

This function provides advanced power management feature identifiers. Refer to the processor BKDG for a detailed description of the definition of each power management feature and whether that feature is supported.

Register	Bits	Description
EAX	31:0	Reserved.
EBX	31:0	Reserved.
ECX	31:0	Reserved.
EDX	31:9	Reserved.
EDX	8	**TscInvariant**. 1 = The TSC rate is ensured to be invariant across all P-States, C-States, and stop grant transitions (such as STPCLK Throttling); therefore the TSC is suitable for use as a source of time. 0 = No such guarantee is made and software should avoid attempting to use the TSC as a source of time.
EDX	7	**HwPState**. Hardware P-State control. [The P-State Current Limit Register] MSRC001_0061, [The P-State Control Register] MSRC001_0062, and [The P-State Status Register] MSRC001_0063 exist.
EDX	6	**100MhzSteps: 100 Mhz multiplier control**.
EDX	5	**STC: software thermal control**.
EDX	4	**TM: hardware thermal control**.
EDX	3	**TTP: THERMTRIP**.
EDX	2	**VID: Voltage ID control**.
EDX	1	**FID: Frequency ID control**.
EDX	0	**TS: Temperature sensor**.

CPUID Fn8000_0008 Address Size And Physical Core Count Information

This function returns information about the number of CPU cores and the maximum physical and linear address width (in bits) supported by the processor. The width reported is the maximum supported in any mode. For long mode capable processors, the size reported is independent of whether long mode is enabled. See “Processor Initialization and Long-Mode Activation” in APM2.

Register	Bits	Description
EAX	31:24	Reserved.
EAX	23:16	**GuestPhysAddrSize: maximum guest physical byte address size in bits**. This number applies only to guests using nested paging. When this field is zero, refer to the PhysAddrSize field for the maximum guest physical address size. See “Secure Virtual Machine” in APM2.
EAX	15:8	**LinAddrSize: maximum linear byte address size in bits**.
EAX	7:0	**PhysAddrSize: maximum physical byte address size in bits**. When GuestPhysAddrSize is zero, this field also indicates the maximum guest physical address size.
EBX	31:0	Reserved.
ECX	31:16	Reserved.

ECX	15:12	**ApicIdCoreIdSize[3:0]**. Indicates the number of least significant bits in the Initial APIC ID that indicate CPU core ID within a processor. A zero value indicates that legacy methods must be used to derive the maximum number of cores. The size of this field determines the maximum number of cores (MNC) that the processor could theoretically support, not the actual number of cores that are actually implemented or enabled on the processor, as indicated by CPUID Fn8000_0008_ECX[NC].
		
		if (ApicIdCoreIdSize[3:0] == 0){
		// Used by legacy dual-core/single-core processors
		MNC = CPUID Fn8000_0008_ECX[NC] + 1;
		} else {
		// use ApicIdCoreIdSize[3:0] field
		MNC = (2 ^ ApicIdCoreIdSize[3:0]);
		}

ECX	11:8	Reserved.
ECX	7:0	**NC: Number of CPU cores - 1**. The number of CPU cores per processor is NC+1. See “Legacy Method” on page 23.
EDX	31:0	Reserved.

CPUID Fn8000_0009 Reserved

This function is reserved.

CPUID Fn8000_000A SVM Revision and Feature Identification

This function returns SVM revision and feature information. See “Secure Virtual Machine” in APM2. If CPUID Fn8000_0001_ECX[SVM] = 0 then CPUID Fn8000_000A is reserved.

Register	Bits	Description
EAX	31:8	Reserved.
EAX	7:0	**SvmRev**: SVM revision.

Register	Bits	Description
EBX	31:0	**NASID**: Number of address space identifiers (ASID).
ECX	31:0	Reserved.
EDX	31:10	Reserved.
EDX	9	**Ssse3Sse5Dis: SSSE3 and SSE5 opcode set disable**. Indicates support for MSR C001_1024[Ssse3Sse5Dis] to disable the SSSE3 and SSE5 instructions. See “Secure Virtual Machine” in APM2.
EDX	3	**NRIPS: NRIP save**. Indicates support for NRIP save on #VMEXIT. See “Save Next Sequential Instruction Pointer on #VMEXIT” in APM2.
EDX	2	**SVML: SVM lock**. Indicates support for SVM-Lock. See “Locking the SVM enable bit” in APM2.
EDX	1	**LbrVirt: LBR virtualization**. Support is provided for VMRUN to save and for VMEXIT to restore the following five MSRs: MSR0000_01D9 Debug Control Register (DBG_CTL_MSR), MSR0000_01DB Last Branch From IP Register (BR_FROM), MSR0000_01DC Last Branch To IP Register (BR_TO), MSR0000_01DD Last Exception From IP Register, MSR0000_01DE Last Exception To IP Register. See “Secure Virtual Machine” in APM2.
EDX	0	**NP: nested paging**. See “Secure Virtual Machine” in APM2.

CPUID Fn8000_00[18:0B] Reserved

These functions are reserved.

CPUID Fn8000_0019 TLB 1GB Page Identifiers

This function returns 1 GB paging information. The *associativity* fields are defined by the CPUID Fn8000_0006 register section.

Register	Bits	Description
EAX	31:28	**L1DTlb1GAssoc**. L1 data TLB associativity for 1 GB pages. (See Table 1 on page 18)
EAX	27:16	**L1DTlb1GSize**. L1 data TLB number of entries for 1 GB pages.
EAX	15:12	**L1ITlb1GAssoc**. L1 instruction TLB associativity for 1 GB pages. (See Table 1 on page 18)
EAX	11:0	**L1ITlb1GSize**. L1 instruction TLB number of entries for 1 GB pages.
EBX	31:28	**L2DTlb1GAssoc**. L2 data TLB associativity for 1 GB pages. (See Table 1 on page 18)
EBX	27:16	**L2DTlb1GSize**. L2 data TLB number of entries for 1-GB pages.
EBX	15:12	**L2ITlb1GAssoc**. L2 instruction TLB associativity for 1-GB pages. (See Table 1 on page 18)
EBX	11:0	**L2ITlb1GSize**. L2 instruction TLB number of entries for 1-GB pages.
ECX	31:0	Reserved.
EDX	31:0	Reserved.

CPUID Fn8000_001A Performance Optimization Identifiers

This function returns performance related information. For more details on how to use these bits to optimize software, see the optimization guide for your processor implementation.

Register	Bits	Description
EAX	31:2	Reserved.
EAX	1	**MOVU**: MOVU SSE (multimedia) instructions are more efficient and should be preferred to SSE (multimedia) MOVL/MOVH. MOVUPS is more efficient than MOVLPS/MOVHPS. MOVUPD is more efficient than MOVLPD/MOVHPD.
EAX	0	**FP128**: 128-bit SSE (multimedia) instructions are executed with full-width internal operations and pipelines rather than decomposing them into internal 64-bit suboperations. This may impact how software performs instruction selection and scheduling.
EBX	31:0	Reserved.
ECX	31:0	Reserved.
EDX	31:0	Reserved.

3 Multiple Core Calculation

Operating systems use one of two possible methods to calculate the number of cores per processor (NC), and the maximum number of cores per processor (MNC). The extended method is recommended, but a legacy method is also available for existing operating systems.

3.1 Legacy Method

The CPUID identification of total number of CPU cores per processor (c) is derived from information returned by the following fields:

- CPUID Fn0000_0001_EBX[LogicalProcessorCount]
- CPUID Fn0000_0001_EDX[HTT] (Hyper-Threading Technology)
- CPUID Fn8000_0001_ECX[CmpLegacy]
- CPUID Fn8000_0008_ECX[NC] (number of CPU cores - 1)

Table 2 defines LogicalProcessorCount, HTT, CmpLegacy, and NC as a function of the number of CPU cores per processor (c).

When HTT = 0, LogicalProcessorCount is reserved and the processor contains one CPU core.

When HTT = 1 and CmpLegacy = 1, LogicalProcessorCount represents the number of CPU cores per processor (c).

Table 2: LogicalProcessorCount, CmpLegacy, HTT, and NC

CPU Cores per Processor (c)	CmpLegacy	HTT	LogicalProcessorCount	NC
1	0	0	Reserved	0
2 or more	1	1	c	c-1

The use of CmpLegacy and LogicalProcessorCount for the determination of the number of CPU cores is deprecated. Instead, use NC to determine the number of CPU cores.

3.2 Extended Method (Recommended)

The CPUID identification of total number of CPU cores per processor is derived from information returned by the CPUID Fn8000_0008_ECX[ApicIdCoreIdSize[3:0]]. This field indicates the number of least significant bits in the CPUID Fn0000_0001_EBX[LocalApicId] that indicates CPU core ID within the processor. The size of this field determines the maximum number of cores (MNC) that the processor could theoretically support, not the actual number of cores that are actually implemented or enabled on the processor, as indicated by CPUID Fn8000_0008_ECX[NC].

A value of zero for ApicIdCoreIdSize[3:0] indicates that the legacy method (section 2.1) should be used to derive the maximum number of cores:

MNC = CPUID Fn8000_0008_ECX[NC] + 1.

For non-zero values of ApicIdCoreIdSize[3:0],

MNC = (2 ^ ApicIdCoreIdSize[3:0])

3.2.1 APIC Enumeration Requirements

System hardware and BIOS must ensure that the maximum number of cores per processor (MNC) exposed to the operating system across all cores and processors in the system is identical.

Local ApicId MNC rule: The ApicId of core j on processor node i must be enumerated/assigned as:

LocalApicId[proc=i, core=j] = (OFFSET_IDX + i) * MNC + j

Where "OFFSET_IDX" is an integer offset (0 to N) used to shift up the CPU LocalApicId values to allow room for IOAPIC devices. This assignment allows software to use a simple bitmask in addressing all the cores of a single processor. (The assignment also has the effect of reserving some IDs from use to ensure alignment of the ID of core 0 on each processor.)

For example, consider a 3-processor system where:

processor 0 has 4 cores
processor 1 has 1 core
processor 2 has 2 cores
there are 8 IOAPIC devices
cpuid.core_id_bits =2 for all cases, so MNC=4

The LocalApicId and IOAPIC ID spaces cannot be disjointed and must be enumerated in the same ID space in order to support legacy operating systems. Each CPU core can support an 8-bit ApicId. But if each IOAPIC device supports only a 4-bit IOAPIC ID, then the problem can be solved by shifting the LocalApicId space to start at some integer multiple of MNC, such as offset 8 (MNC = 4; OFFSET_IDX=2):

LocalApicId[proc=0,core=0] = (2+0)*4 + 0 = 0x08
LocalApicId[proc=0,core=1] = (2+0)*4 + 1 = 0x09
LocalApicId[proc=0,core=2] = (2+0)*4 + 2 = 0x0A
LocalApicId[proc=0,core=3] = (2+0)*4 + 3 = 0x0B

LocalApicId[proc=1,core=0] = (2+1)*4 + 0 = 0x0C
LocalApicId 0xD to 0xF are reserved

LocalApicId[proc=2,core=0] = (2+2)*4 + 0 = 0x10
LocalApicId[proc=2,core=1] = (2+2)*4 + 1 = 0x11

LocalApicId 0x12 and 0x13 are reserved

It is recommended that BIOS use the following LocalApicId assignments for the broadest operating system support. Given N = (Number_Of_Processors * MNC) and M = Number_Of_IOAPICs:

- If (N+M) <16, assign the LocalApicIds for the cores first from 0 to N-1, and the IOAPIC IDs from N to N+(M-1).
- If (N+M) > =16, assign the IOAPIC IDs first from 0 to M-1, and the LocalApicIds for the cores from K to K+(N-1), where K is an integer multiple of MNC greater than M-1.