



# CPUID Specification

Publication # <b>25481</b>	Revision: <b>2.26</b>
Issue Date: <b>July 2007</b>	

© 2002-2007 Advanced Micro Devices, Inc. All rights reserved.

The contents of this document are provided in connection with Advanced Micro Devices, Inc. (“AMD”) products. AMD makes no representations or warranties with respect to the accuracy or completeness of the contents of this publication and reserves the right to make changes to specifications and product descriptions at any time without notice. The information contained herein may be of a preliminary or advance nature and is subject to change without notice.. No license, whether express, implied, arising by estoppel or otherwise, to any intellectual property rights is granted by this publication. Except as set forth in AMD’s Standard Terms and Conditions of Sale, AMD assumes no liability whatsoever, and disclaims any express or implied warranty, relating to its products including, but not limited to, the implied warranty of merchantability, fitness for a particular purpose, or infringement of any intellectual property right.

AMD’s products are not designed, intended, authorized or warranted for use as components in systems intended for surgical implant into the body, or in other applications intended to support or sustain life, or in any other application in which the failure of AMD’s product could create a situation where personal injury, death, or severe property or environmental damage may occur. AMD reserves the right to discontinue or make changes to its products at any time without notice.

### **Trademarks**

AMD, the AMD Arrow logo, and combinations thereof, and 3DNow! are trademarks of Advanced Micro Devices, Inc.

MMX is a trademark and Pentium is a registered trademark of Intel Corporation.

Other product names used in this publication are for identification purposes only and may be trademarks of their respective companies.

# Table of Contents

<b>1</b>	<b>Overview</b> .....	<b>7</b>
1.1	Reference Documents .....	7
1.2	Conventions .....	7
1.3	Definitions .....	8
1.4	CPUID Function Selection .....	9
1.5	Standard, Extended, and Undefined Functions .....	9
<b>2</b>	<b>CPUID Function Specification</b> .....	<b>10</b>
	CPUID Fn0000_0000: Processor Vendor and Largest Standard Function Number .....	10
	CPUID Fn0000_0001_EAX: Family, Model, Stepping Identifiers .....	10
	CPUID Fn0000_0001_EBX: LocalApicId, LogicalProcessorCount, CLFlush .....	11
	CPUID Fn0000_0001_ECX: Feature Identifiers .....	11
	CPUID Fn0000_0001_EDX: Feature Identifiers .....	12
	CPUID Fn0000_000[4:2]: Reserved .....	13
	CPUID Fn0000_0005: MONITOR/MWAIT .....	13
	CPUID Fn8000_0000: Processor Vendor and Largest Extended Function Number .....	14
	CPUID Fn8000_0001_EAX: AMD Family, Model, Stepping .....	14
	CPUID Fn8000_0001_EBX: BrandId Identifier .....	14
	CPUID Fn8000_0001_ECX: Feature Identifiers .....	14
	CPUID Fn8000_0001_EDX: Feature Identifiers .....	15
	CPUID Fn8000_000[4:2]: Processor Name String Identifier .....	16
	CPUID Fn8000_0005: L1 Cache and TLB Identifiers .....	16
	CPUID Fn8000_0006: L2/L3 Cache and L2 TLB Identifiers .....	17
	CPUID Fn8000_0007: Advanced Power Management Information .....	19
	CPUID Fn8000_0008: Long Mode Address Size Identifiers .....	19
	CPUID Fn8000_0009: Reserved .....	20
	CPUID Fn8000_000A: SVM Revision and Feature Identification .....	20
	CPUID Fn8000_00[18:0B]: Reserved .....	21
	CPUID Fn8000_0019: TLB 1GB Page Identifiers .....	21
	CPUID Fn8000_001A: Performance Optimization Identifiers .....	21
	CPUID Fn8000_001B: Instruction Based Sampling Identifiers .....	22
<b>3</b>	<b>Multiple Core Calculation</b> .....	<b>23</b>
3.1	Legacy Method .....	23
3.2	Extended Method (Recommended) .....	24



# Revision History

Date	Rev	Description
July 2007	2.26	<ul style="list-style-type: none"> <li>• CPUID Fn0000_0001_ECX[Monitor]: Added.</li> <li>• CPUID Fn0000_0001_ECX[POPCNT]: Added.</li> <li>• CPUID Fn0000_000[4:2]: Added as reserved.</li> <li>• CPUID Fn0000_0005 [MONITOR/MWAIT] on page 13: Added.</li> <li>• CPUID Fn0000_0005_ECX[IBE, EMX]: Added.</li> <li>• CPUID Fn8000_0001_EBX[PkgType[3:0]]: Added.</li> <li>• CPUID Fn8000_0001_ECX[WDT]: Added.</li> <li>• CPUID Fn8000_0001_ECX[SKINIT]: Added.</li> <li>• CPUID Fn8000_0001_ECX[OSVW]: Added.</li> <li>• CPUID Fn8000_0001_ECX[3DNowPrefetch]: Added.</li> <li>• CPUID Fn8000_0001_ECX[MisAlignSse]: Added.</li> <li>• CPUID Fn8000_0001_ECX[SSE4A]: Added.</li> <li>• CPUID Fn8000_0001_ECX[ABM]: Added.</li> <li>• CPUID Fn8000_0001_ECX[ExtApicSpace]: Added.</li> <li>• CPUID Fn8000_0001_EDX[Page1GB]: Added.</li> <li>• CPUID Fn8000_0006_EDX[L3Size, L3Assoc, L3LinesPerTag, L3LineSize]: Added.</li> <li>• CPUID Fn8000_0006: Table 1: Added additional associativity definitions.</li> <li>• CPUID Fn8000_0007_EDX[HwpPState]: Added Hardware P-State control.</li> <li>• CPUID Fn8000_0007_EDX[100MhzSteps]: Added.</li> <li>• CPUID Fn8000_000A_EDX[NRIPS]: Added.</li> <li>• CPUID Fn8000_000A_EDX[SVML]: Added.</li> <li>• CPUID Fn8000_000A_EDX[LbrVirt]: Added LBR virtualization.</li> <li>• CPUID Fn8000_000A_EDX[NP]: Added.</li> <li>• CPUID Fn8000_0019 [TLB 1GB Page Identifiers] on page 21: Added.</li> <li>• CPUID Fn8000_001A [Performance Optimization Identifiers] on page 21: Added.</li> <li>• CPUID Fn8000_001B [Instruction Based Sampling Identifiers] on page 22: Added.</li> </ul>
January 2006	2.18	<ul style="list-style-type: none"> <li>• Renamed CPUID Fn8000_0007_EDX[8] from TscPStateInvariant to TscInvariant.</li> <li>• Added CPUID Fn8000_0008_ECX[ApicIdCoreIdSize[3:0]].</li> </ul>

Date	Rev	Description
September 2005	2.16	<ul style="list-style-type: none"> <li>• Reformatted document for clarity.</li> <li>• Moved the chapter titled “Programming The Processor Name String” to the processor revision guide.</li> <li>• Added definition for HTT, CmpLegacy, and LogicalProcessorCount for multi-threading.               <ul style="list-style-type: none"> <li>• See “Legacy Method” on page 23.</li> <li>• See <a href="#">CPUID Fn8000_0001_ECX</a> for CmpLegacy.</li> <li>• See <a href="#">CPUID Fn0000_0001_EBX</a> for LogicalProcessorCount.</li> <li>• See <a href="#">CPUID Fn0000_0001_EDX</a> for HTT.</li> </ul> </li> <li>• Extended BrandID (BrandId[15:12]) definition to <a href="#">CPUID Fn8000_0001_EBX</a>.</li> <li>• Added SVM. See <a href="#">CPUID Fn8000_0001_ECX</a>.</li> <li>• Added SVM definition to <a href="#">CPUID Fn8000_000A</a>.</li> <li>• Added CMPXCHG16B. See <a href="#">CPUID Fn0000_0001_ECX</a>.</li> <li>• Added AltMovCr8. See <a href="#">CPUID Fn8000_0001_ECX</a>.</li> <li>• Added LahfSahf. See <a href="#">CPUID Fn8000_0001_ECX</a>.</li> <li>• Added RDTSCP. See <a href="#">CPUID Fn8000_0001_EDX</a>.</li> </ul>
See revision 2.15 for the change history prior to rev 2.16.		

# 1 Overview

This document specifies the CUID 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 CUID instruction is also documented in the section titled “CUID” 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#s 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.
- *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 CUID 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 CUID 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 CUID functions is CUID FnXXXX\_XXXX, where the CUID function is XXXX\_XXXXh; e.g., CUID Fn0000\_0001.
- The convention for referring to CUID capability fields is CUID FnXXXX\_XXXX\_RRR[FieldName], where RRR is the register (EAX, EBX, ECX, EDX) and FieldName is the name of the capability field; e.g., CUID 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 CUID Fn0000\_0001\_EAX[Extended-Family,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:

- **APMn.** 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 CPLD Function Selection

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

## 1.5 Standard, Extended, and Undefined Functions

The CPLD 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 CPLD 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 CPLD 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	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	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.

31	28 27	20 19	16 15	12 11	8 7	4 3	0
Reserved, RAZ	ExtendedFamily	ExtendedModel	Reserved, RAZ	BaseFamily	BaseModel	Stepping	

Bits	Description
31:28	Reserved.
27:20	<b>ExtendedFamily</b> : Processor extended family. See above for definition of Family[7:0].
19:16	<b>ExtendedModel</b> : Processor extended model. See above for definition of Model[7:0].
15:12	Reserved.
11:8	<b>BaseFamily</b> : Base processor family. See above for definition of Family[7:0].
7:4	<b>BaseModel</b> : Base processor model. See above for definition of Model[7:0].
3:0	<b>Stepping</b> : 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	<b>LocalApicId</b> : 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	<b>LogicalProcessorCount</b> : 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. AMD currently does not support more than one thread per CPU core. If CPUID Fn0000_0001_EDX[HTT] = 0 then LogicalProcessorCount is reserved. See “Legacy Method” on page 23.
15:8	<b>CLFlush</b> : CLFLUSH size. Specifies the size of a cache line in quadwords flushed by the CLFLUSH instruction. See “CLFLUSH” in APM3.
7:0	<b>8BitBrandId</b> : 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	<b>POPCNT</b> : POPCNT instruction. See “POPCNT” in APM3.
22:14	Reserved.
13	<b>CMPXCHG16B</b> : CMPXCHG16B instruction. See “CMPXCHG16B” in APM3.

Bits	Description
12:4	Reserved.
3	<b>MONITOR</b> : MONITOR/MWAIT instructions. See “MONITOR” and “MWAIT” in APM3.
2:1	Reserved.
0	<b>SSE3</b> : SSE3 extensions. See Appendix D “Instruction Subsets and CPIID Feature Sets” in APM3 for the list of instructions covered by the SSE3 feature bit. See APM4 for the definition of the SSE3 instructions.

### CPIID Fn0000\_0001\_EDX Feature Identifiers

This function contains the following miscellaneous feature identifiers.

Bits	Description
31:29	Reserved.
28	<b>HTT</b> : Hyper-Threading Technology. Indicates either that there is more than one thread per CPU core or more than one CPU core per processor. AMD currently does not support more than one thread per CPU core. See “Legacy Method” on page 23.
27	Reserved.
26	<b>SSE2</b> : SSE2 extensions. See Appendix D “CPIID Feature Sets” in APM3.
25	<b>SSE</b> : SSE extensions. See Appendix D “CPIID Feature Sets” in APM3 appendix and “64-Bit Media Programming” in APM1.
24	<b>FXSR</b> : FXSAVE and FXRSTOR instructions. See “FXSAVE” and “FXRSTOR” in APM4.
23	<b>MMX</b> : MMX™ instructions. See Appendix D “CPIID Feature Sets” in APM3 and “128-Bit Media and Scientific Programming” in APM1.
22:20	Reserved.
19	<b>CLFSH</b> : CLFLUSH instruction. See “CLFLUSH” in APM3.
18	Reserved.
17	<b>PSE36</b> : Page-size extensions. The PDE[20:13] supplies physical address [39:32]. See “Page Translation and Protection” in APM2.
16	<b>PAT</b> : Page attribute table. PCD, PWT, and PATi are used to alter memory type. See “Page-Attribute Table Mechanism” in APM2.
15	<b>CMOV</b> : Conditional move instructions, CMOV, FCMOV. See “CMOV”, “FCMOV” in APM3.
14	<b>MCA</b> : Machine check architecture, MCG_CAP. See “Machine Check Mechanism” in APM2.
13	<b>PGE</b> : Page global extension, CR4.PGE. See “Page Translation and Protection” in APM2.
12	<b>MTRR</b> : Memory-type range registers. MTRRcap supported. See “Page Translation and Protection” in APM2.
11	<b>SysEnterSysExit</b> : SYSENTER and SYSEXIT instructions. See “SYSENTER”, “SYSEXIT” in APM3.
10	Reserved.

Bits	Description
9	<b>APIC</b> : Advanced programmable interrupt controller (APIC) exists and is enabled. See “Exceptions and Interrupts” in APM2.
8	<b>CMPXCHG8B</b> : CMPXCHG8B instruction. See “CMPXCHG8B” in APM3.
7	<b>MCE</b> : Machine check exception, CR4.MCE. See “Machine Check Mechanism” in APM2.
6	<b>PAE</b> : Physical-address extensions (PAE), support for physical addresses $\geq 32$ b. Number of physical address bits above 32b is implementation specific. See “Page Translation and Protection” in APM2.
5	<b>MSR</b> : AMD model-specific registers (MSRs), with RDMSR and WRMSR instructions. See “Model Specific Registers” in APM2.
4	<b>TSC</b> : Time stamp counter. RDTSC and RDTSCP instruction support. See “Debug and Performance Resources” in APM2.
3	<b>PSE</b> : Page-size extensions (4 MB pages). See “Page Translation and Protection” in APM2.
2	<b>DE</b> : Debugging extensions, I/O breakpoints, CR4.DE. See “Debug and Performance Resources” in APM2.
1	<b>VME</b> : 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	<b>FPU</b> : 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	<b>MonLineSizeMin</b> : Smallest monitor-line size in bytes.
EBX	31:16	Reserved.
EBX	15:0	<b>MonLineSizeMax</b> : Largest monitor-line size in bytes.
ECX	31:2	Reserved.
ECX	1	<b>IBE</b> : 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	<b>EMX</b> : 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	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	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	<b>PkgType:</b> 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	<b>BrandId:</b> Brand ID. This field, in conjunction with <a href="#">CPUID Fn0000_0001_EBX[8BitBrandId]</a> , 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	<b>WDT:</b> Watchdog timer support.
12	<b>SKINIT:</b> SKINIT, STGI, and DEV support.
11:10	Reserved.
9	<b>OSVW:</b> OS visible workaround. Indicates OS-visible workaround support. See “OS Visible Workaround (OSVW) Information” in APM2.
8	<b>3DNowPrefetch:</b> PREFETCH and PREFETCHW instruction support. See “PREFETCH” and “PREFETCHW” in APM3.

Bits	Description
7	<b>MisAlignSse</b> : Misaligned SSE mode. See “Misaligned Access Support Added for SSE Instructions” in APM1.
6	<b>SSE4A</b> : EXTRQ, INSERTQ, MOVNTSS, and MOVNTSD instruction support. See “EXTRQ”, “INSERTQ”, “MOVNTSS”, and “MOVNTSD” in APM4.
5	<b>ABM</b> : Advanced bit manipulation. LZCNT instruction support. See “LZCNT” in APM3.
4	<b>AltMovCr8</b> : LOCK MOV CR0 means MOV CR8. See “MOV(CRn)” in APM3.
3	<b>ExtApicSpace</b> : 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	<b>SVM</b> : Secure virtual machine feature. See “Secure Virtual Machine” in APM2.
1	<b>CmpLegacy</b> : Core multi-processing legacy mode. See “Legacy Method” on page 23.
0	<b>LahfSahf</b> : 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	<b>3DNow</b> : 3DNow!™ instructions. See Appendix D “Instruction Subsets and CPUID Feature Sets” in APM3.
30	<b>3DNowExt</b> : AMD extensions to 3DNow! instructions. See Appendix D “Instruction Subsets and CPUID Feature Sets” in APM3.
29	<b>LM</b> : Long mode. See “Processor Initialization and Long-Mode Activation” in APM2.
28	Reserved.
27	<b>RDTSCP</b> : RDTSCP instruction. See “RDTSCP” in APM3.
26	<b>Page1GB</b> : 1-GB large page support. See “1-GB Paging Support” in APM2.
25	<b>FXSR</b> : FXSAVE and FXRSTOR instruction optimizations. See “FXSAVE” and “FXRSTOR” in APM4.
24	<b>FXSR</b> : FXSAVE and FXRSTOR instructions. Same as <a href="#">CPUID Fn0000_0001_EDX[FXSR]</a> .
23	<b>MMX</b> : MMX™ instructions. Same as <a href="#">CPUID Fn0000_0001_EDX[MMX]</a> .
22	<b>MmxExt</b> : 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	<b>NX</b> : No-execute page protection. See “Page Translation and Protection” in APM2.
19:18	Reserved.
17	<b>PSE36</b> : Page-size extensions. Same as <a href="#">CPUID Fn0000_0001_EDX[PSE36]</a> .
16	<b>PAT</b> : Page attribute table. Same as <a href="#">CPUID Fn0000_0001_EDX[PAT]</a> .
15	<b>CMOV</b> : Conditional move instructions. Same as <a href="#">CPUID Fn0000_0001_EDX[CMOV]</a> .

Bits	Description
14	<b>MCA:</b> Machine check architecture. Same as CPUID Fn0000_0001_EDX[MCA].
13	<b>PGE:</b> Page global extension. Same as CPUID Fn0000_0001_EDX[PGE].
12	<b>MTRR:</b> Memory-type range registers. Same as CPUID Fn0000_0001_EDX[MTRR].
11	<b>SysCallSysRet:</b> SYSCALL and SYSRET instructions. See “SYSCALL” and “SYSRET” in APM3.
10	Reserved.
9	<b>APIC:</b> Advanced programmable interrupt controller. Same as CPUID Fn0000_0001_EDX[APIC].
8	<b>CMPXCHG8B:</b> CMPXCHG8B instruction. Same as CPUID Fn0000_0001_EDX[CMPXCHG8B].
7	<b>MCE:</b> Machine check exception. Same as CPUID Fn0000_0001_EDX[MCE].
6	<b>PAE:</b> Physical-address extensions. Same as CPUID Fn0000_0001_EDX[PAE].
5	<b>MSR:</b> AMD model-specific registers. Same as CPUID Fn0000_0001_EDX[MSR].
4	<b>TSC:</b> Time stamp counter. Same as CPUID Fn0000_0001_EDX[TSC].
3	<b>PSE:</b> Page-size extensions. Same as CPUID Fn0000_0001_EDX[PSE].
2	<b>DE:</b> Debugging extensions. Same as CPUID Fn0000_0001_EDX[DE].
1	<b>VME:</b> Virtual-mode enhancements. Same as CPUID Fn0000_0001_EDX[VME].
0	<b>FPU:</b> 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	<b>L1DTlb2and4MAssoc.</b> Data TLB associativity for 2-MB and 4-MB pages.
EAX	23:16	<b>L1DTlb2and4MSize.</b> 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	<b>L1ITlb2and4MAssoc.</b> Instruction TLB associativity for 2-MB and 4-MB pages.
EAX	7:0	<b>L1ITlb2and4MSize.</b> 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	<b>L1DTlb4KAssoc.</b> Data TLB associativity for 4 KB pages.
EBX	23:16	<b>L1DTlb4KSize.</b> Data TLB number of entries for 4 KB pages.
EBX	15:8	<b>L1ITlb4KAssoc.</b> Instruction TLB associativity for 4 KB pages.
EBX	7:0	<b>L1ITlb4KSize.</b> Instruction TLB number of entries for 4 KB pages.
ECX	31:24	<b>L1DcSize.</b> L1 data cache size in KB.
ECX	23:16	<b>L1DcAssoc.</b> L1 data cache associativity.
ECX	15:8	<b>L1DcLinesPerTag.</b> L1 data cache lines per tag.
ECX	7:0	<b>L1DcLineSize.</b> L1 data cache line size in bytes.
EDX	31:24	<b>L1IcSize.</b> L1 instruction cache size KB.
EDX	23:16	<b>L1IcAssoc.</b> L1 instruction cache associativity.
EDX	15:8	<b>L1IcLinesPerTag.</b> L1 instruction cache lines per tag.
EDX	7:0	<b>L1IcLineSize.</b> L1 instruction cache line size in bytes.

## CPIID 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 presence of a unified L2 TLB is indicated by a value of 0000h in the upper 16 bits of the EAX and EBX registers. The unified L2 TLB information is contained in the lower 16 bits of these registers.

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	<b>L2DTlb2and4MAssoc.</b> L2 data TLB associativity for 2-MB and 4-MB pages. (See Table 1)
EAX	27:16	<b>L2DTlb2and4MSize.</b> 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	<b>L2ITlb2and4MAssoc.</b> L2 instruction TLB associativity for 2-MB and 4-MB pages. (See Table 1.)
EAX	11:0	<b>L2ITlb2and4MSize.</b> 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	<b>L2DTlb4KAssoc.</b> L2 data TLB associativity for 4-KB pages. (See Table 1.)
EBX	27:16	<b>L2DTlb4KSize.</b> L2 data TLB number of entries for 4-KB pages.
EBX	15:12	<b>L2ITlb4KAssoc.</b> L2 instruction TLB associativity for 4-KB pages. (See Table 1.)
EBX	11:0	<b>L2ITlb4KSize.</b> L2 instruction TLB number of entries for 4-KB pages.
ECX	31:16	<b>L2Size.</b> L2 cache size in KB.
ECX	15:12	<b>L2Assoc.</b> L2 cache associativity. (See Table 1.)
ECX	11:8	<b>L2LinesPerTag.</b> L2 cache lines per tag.
ECX	7:0	<b>L2LineSize.</b> L2 cache line size in bytes.

Register	Bits	Description
EDX	31:18	<b>L3Size.</b> 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	<b>L3Assoc.</b> L3 cache associativity. (See Table 1)
EDX	11:8	<b>L3LinesPerTag.</b> L3 cache lines per tag.
EDX	7:0	<b>L3LineSize.</b> 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	<b>TscInvariant:</b> 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	<b>HwpPState:</b> 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	<b>100MhzSteps:</b> 100 Mhz multiplier control.
EDX	5	<b>STC:</b> Software thermal control is supported.
EDX	4	<b>TM:</b> Hardware thermal control is supported.
EDX	3	<b>TTP:</b> THERMTRIP is supported.
EDX	2	<b>VID:</b> Voltage ID control is supported.
EDX	1	<b>FID:</b> Frequency ID control.
EDX	0	<b>TS:</b> Temperature sensor.

### CPUID Fn8000\_0008 Long Mode Address Size Identifiers

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	31:16	Reserved.
EAX	15:8	<b>LinAddrSize</b> : Maximum linear byte address size in bits.
EAX	7:0	<b>PhysAddrSize</b> : Maximum physical byte address size in bits.
EBX	31:0	Reserved.
ECX	31:16	Reserved.
ECX	15:12	<b>ApicIdCoreIdSize[3:0]</b> . 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 <a href="#">CPUID Fn8000_0008_ECX[NC]</a> .  <pre> 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]); } </pre>
ECX	11:8	Reserved.
ECX	7:0	<b>NC</b> : 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	<b>SvmRev</b> : SVM revision.
EBX	31:0	<b>NASID</b> : Number of address space identifiers (ASID).
ECX	31:0	Reserved.
EDX	31:4	Reserved.
EDX	3	<b>NRIPS</b> : NRIP Save. Indicates support for NRIP save on #VMEXIT. See “Save Next Sequential Instruction Pointer on #VMEXIT” in APM2.

Register	Bits	Description
EDX	2	<b>SVML</b> : SVM Lock. Indicates support for SVM-Lock. See “Locking the SVM enable bit” in APM2.
EDX	1	<b>LbrVirt</b> : 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	<b>NP</b> : 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	<b>L1DTlb1GAssoc</b> . L1 data TLB associativity for 1 GB pages. (See Table 1 on page 18)
EAX	27:16	<b>L1DTlb1GSize</b> . L1 data TLB number of entries for 1 GB pages.
EAX	15:12	<b>L1ITlb1GAssoc</b> . L1 instruction TLB associativity for 1 GB pages. (See Table 1 on page 18)
EAX	11:0	<b>L1ITlb1GSize</b> . L1 instruction TLB number of entries for 1 GB pages.
EBX	31:28	<b>L2DTlb1GAssoc</b> . L2 data TLB associativity for 1 GB pages. (See Table 1 on page 18)
EBX	27:16	<b>L2DTlb1GSize</b> . L2 data TLB number of entries for 1-GB pages.
EBX	15:12	<b>L2ITlb1GAssoc</b> . L2 instruction TLB associativity for 1-GB pages. (See Table 1 on page 18)
EBX	11:0	<b>L2ITlb1GSize</b> . 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	<b>MOVU</b> : MOVU SSE (multimedia) instructions are more efficient and should be preferred to SSE (multimedia) MOVL/MOVH. MOVUPS is more efficient than MOV-LPS/MOVHPS. MOVUPD is more efficient than MOVLPD/MOVHPD.
EAX	0	<b>FP128</b> : 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 CPLUID identification of total number of CPU cores per processor (c) and threads per processor (t) is derived from information returned by the following fields:

- CPLUID Fn0000\_0001\_EBX[LogicalProcessorCount]
- CPLUID Fn0000\_0001\_EDX[HTT] (Hyper-Threading Technology)
- CPLUID Fn8000\_0001\_ECX[CmpLegacy]
- CPLUID 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) and the number of threads per CPU core (t).

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

When HTT = 1 and CmpLegacy = 1, LogicalProcessorCount represents the number of CPU cores per processor (c), where each CPU core is single-threaded.

When HTT = 1 and CmpLegacy = 0, LogicalProcessorCount represents the number of total threads for the processor, which is the product of  $c * t$ . NC always represents the number of CPU cores per processor - 1.

**Table 2: LogicalProcessorCount, CmpLegacy, HTT, and NC**

CPU Cores per Processor (c)	Threads per CPU Core (t)	CmpLegacy	HTT	LogicalProcessorCount	NC
1	1	0	0	Reserved	0
2 or more	1	1	1	c	c-1
1	2 or more	0	1	$c*t$	0
2 or more	2 or more	0	1	$c*t$	c-1

AMD currently does not support multiple threads per CPU core. The number of threads per CPU core is always 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:

$$\text{MNC} = \text{CPUID Fn8000_0008\_ECX[NC]} + 1.$$

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

$$\text{MNC} = (2 \wedge \text{ApicIdCoreIdSize[3:0]})$$

### 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:

$$\text{LocalApicId[proc=i, core=j]} = (\text{OFFSET\_IDX} + i) * \text{MNC} + j$$

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):

$$\begin{aligned} \text{LocalApicId[proc=0,core=0]} &= (2+0)*4 + 0 = 0x08 \\ \text{LocalApicId[proc=0,core=1]} &= (2+0)*4 + 1 = 0x09 \end{aligned}$$

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  $N = (\text{Number\_Of\_Processors} * \text{MNC})$  and  $M = \text{Number\_Of\_IOAPICs}$ :

- 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  $(N+M) \geq 16$ , 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  $M-1$ .

