Preliminary
Processor Programming Reference (PPR) for AMD Family 19h Model 01h, Revision B1 Processors
Volume 1 of 2

Legal Notices

Trademarks:
AMD, the AMD Arrow logo, and combinations thereof are trademarks of Advanced Micro Devices, Inc.
AGESA is a trademark of Advanced Micro Devices, Inc.
AMD Virtualization is a trademark of Advanced Micro Devices, Inc.
AMD-V is a trademark of Advanced Micro Devices, Inc.
Adobe is a registered trademark of Adobe.
CXL is a trademark of Compute Express Link Consortium, Inc.
Infinity Fabric is a trademark of Advanced Micro Devices, Inc.
Linux is a registered trademark of Linus Torvalds.
Microsoft is a registered trademark of Microsoft Corporation.
PCI Express is a registered trademark of PCI-SIG Corporation.
PCIe is a registered trademark of PCI-SIG Corporation.
Windows is a registered trademark of Microsoft Corporation.
Other product names used in this publication are for identification purposes only and may be trademarks of their respective companies.

Reverse engineering or disassembly is prohibited.

USE OF THIS PRODUCT IN ANY MANNER THAT COMPLIES WITH THE MPEG ACTUAL OR DE FACTO VIDEO AND/OR AUDIO STANDARDS IS EXPRESSLY PROHIBITED WITHOUT ALL NECESSARY LICENSES UNDER APPLICABLE PATENTS. SUCH LICENSES MAY BE ACQUIRED FROM VARIOUS THIRD PARTIES INCLUDING, BUT NOT LIMITED TO, IN THE MPEG PATENT PORTFOLIO, WHICH LICENSE IS AVAILABLE FROM MPEG LA, L.L.C., 6312 S. FIDDLERS GREEN CIRCLE, SUITE 400E, GREENWOOD VILLAGE, COLORADO 80111.

List of Chapters

Table of Contents

List of Figures

List of Tables

1 Overview

1.1 Intended Audience

This document provides the processor behavioral definition and associated design notes. It is intended for platform designers and for programmers involved in the development of BIOS functions, drivers, and operating system kernel modules.

1.2 Reference Documents

Table 1: Reference Documents Listing

Term	Description
docAPM1	AMD64 Architecture Programmer's Manual Volume 1: Application Programming, Publication No. 24592.
docAPM2	AMD64 Architecture Programmer's Manual Volume 2: System Programming, Publication No. 24593.
docAPM3	AMD64 Architecture Programmer's Manual Volume 3: Instruction-Set Reference, Publication No. 24594.
docAPM4	AMD64 Architecture Programmer's Manual Volume 4: 128-Bit and 256-Bit Media Instructions, Publication No. 26568.
docAPM5	AMD64 Architecture Programmer's Manual Volume 5: 64-Bit Media and x87 Floating-Point Instructions, Publication No. 26569.
docACPI	Advanced Configuration and Power Interface (ACPI) Specification. http://www.acpi.info.
docJEDEC	JEDEC Standards. http://www.jedec.org.
docMBDG	Socket SP3 Processor Motherboard Design Guide (MBDG), Publication No. 55414.
docPCIe	PCI Express® Specification. http://www.pcisig.com.
docPCIlb	PCI Local Bus Specification. http://www.pcisig.com.
docRevG	Revision Guide for AMD Family 19h Models 00h-0Fh Processors, Publication No. 56683.
docSSP3	Socket SP3 Functional Data Sheet (FDS), Publication No. 55426.
docSMB	System Management Bus (SMBus) Specification. http://www.smbus.org.

1.2.1 Documentation Conventions

When referencing information found in external documents listed in Reference Documents, the "=>" operator is used. This notation represents the item to be searched for in the reference document. For example:

docExDoc => Header1 => Header2

is to have the reader use the search facility when opening referenced document "docExDoc" and search for "Header2". "Header2" may appear more than once in "docExDoc", therefore, referencing the one that follows "Header1". In that case, the easiest way to get to Header2 is to use the search to locate Header1, then again to locate "Header2".

1.3 Adobe® Reader

This section describes how to configure and use Adobe® Reader for the PPR PDFs.

Adobe Reader is the recommended tool for viewing PPR pdfs and can be downloaded at https://get.adobe.com/reader/.

1.3.1 Adobe® Reader Configuration

This section describes how to configure Adobe Reader for the PPR PDFs.

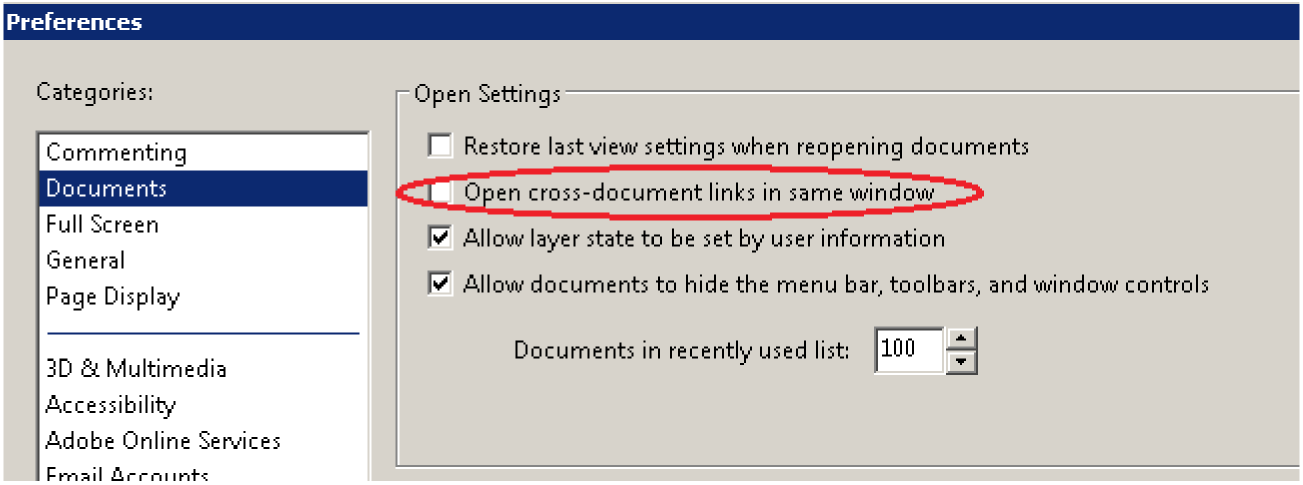
1.3.1.1 Open Hyperlink Document in New Window

The Open Hyperlink Document in New Window setting opens a new window for a hyperlink, instead of opening the hyperlink document in the same window.

- Only when deselected are previously opened files visible in the Windows® pull-down menu.

Edit->Preferences:

- Documents
  - Open Settings:
    - Deselect: Open cross-document links in same window



Preferences

Categories:

- Commenting
- Documents
- Full Screen
- General
- Page Display
- 3D & Multimedia
- Accessibility
- Adobe Online Services
- Email Accounts

Open Settings

☐ Restore last view settings when reopening documents

☐ Open cross-document links in same window

☑ Allow layer state to be set by user information

☑ Allow documents to hide the menu bar, toolbars, and window controls

Documents in recently used list: 100

Figure 1: Adobe® Reader Hyperlink Opens New Window Configuration

Figure 2 shows how when hyperlinking from volume 2 to volume 1, that volume 2 is left open. The check indicates the foreground window.

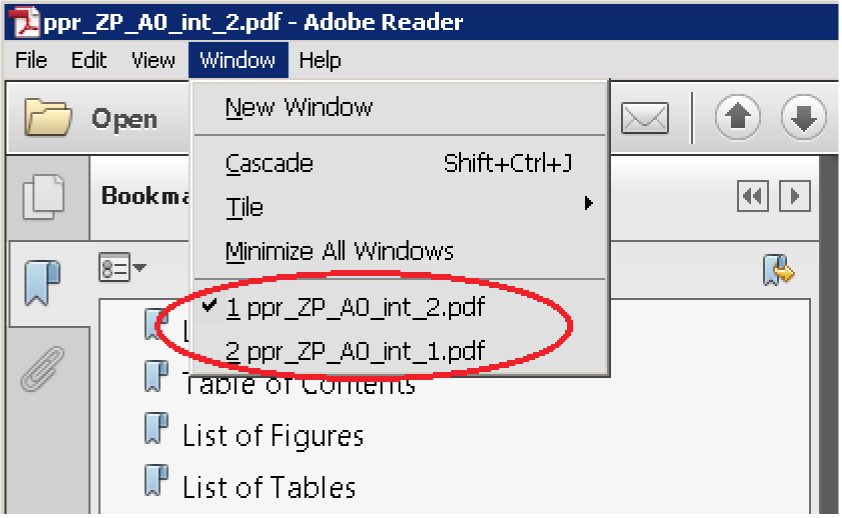


Figure 2: Adobe® Reader Select Between Opened Files

1.3.1.2 Show Toolbars

If Toolbars is not shown:

- View->Show/Hide->Toolbar Items->Show Toolbars
- The toolbar is needed to see the "Previous View" and "Next View" buttons.

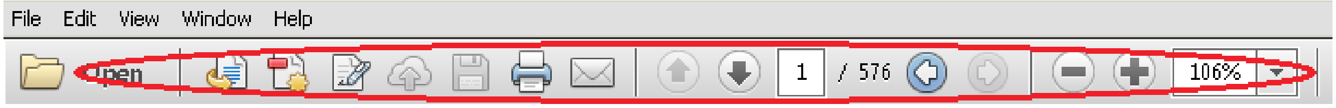


Figure 3: Adobe® Reader Show Toolbars Configuration

1.3.1.3 Show "Previous View" and "Next View" Buttons

If the "Previous View" (left arrow) and "Next View" (right arrow) buttons are not shown:

- Right click on toolbar-> Page Navigation-> select "Previous View" and "Next View" items.

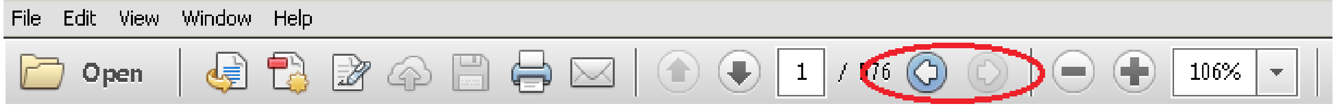


Figure 4: Adobe® Reader Prev/Next Buttons

1.3.2 Adobe® Reader Usage

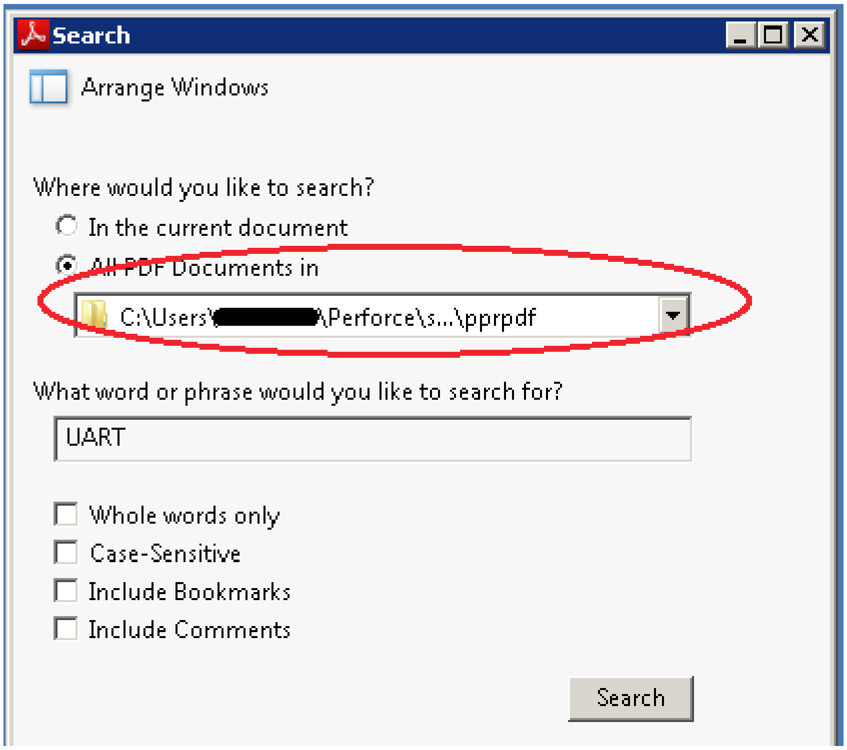
This section describes how to use Adobe Reader for the PPR PDFs.

NOTE: PDF's are distributed in zip format. In order to search and hyperlink between PDF volumes, the zip contents must be extracted to a folder.

1.3.2.1 Searching a Multiple Volume PPR

The PPR is a multiple PDF document and searching all PDFs is performed as follows:

- The zip of PDF files must be extracted to a directory where the search will be performed. A sesarch across multiple PDF files can not be performed from within a zip of PDF's.
- Open search by selecting Edit -> Advanced Search (Shift+Ctrl+F)
- Select "All PDF Documents in" and select "Browse for Location...", which opens the "Browse For Folder" window.
- In the "Browse For Folder" window, select the folder that contains the PPR PDFs that need to be searched, and select OK.



Where would you like to search?

○ In the current document

● All PDF Documents in

C:\Users\[redacted]\Perforce\s...\pprpdf

What word or phrase would you like to search for?

UART

☐ Whole words only

☐ Case-Sensitive

☐ Include Bookmarks

☐ Include Comments

Figure 5: Adobe® Reader Searching a Multiple Volume PPR

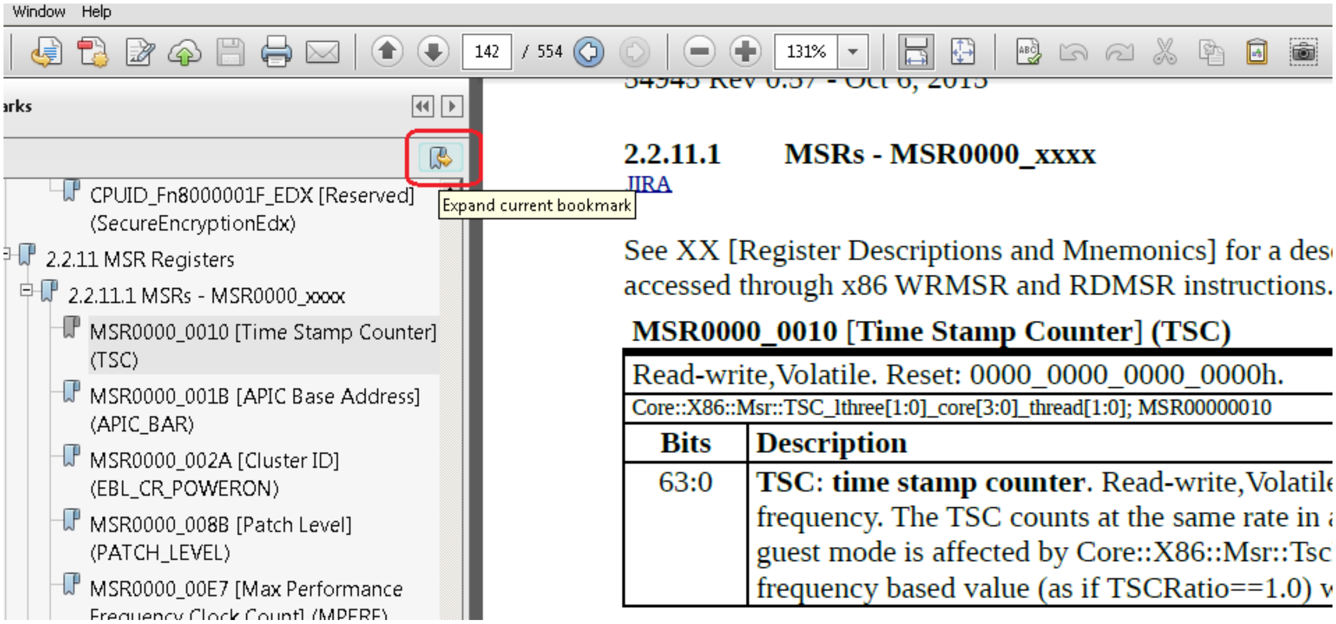
1.3.2.2 Cross-References and Hyperlinks

A cross-reference is a link to a location within the same PDF. A hyperlink is a link to a location within a different PDF.

- For cross-references, use "Previous View" to return from the current location to the previous location.
- Hyperlinks between documents leave the current location unchanged in the PDF that contained the hyperlink.
- In order for hyperlinks to work properly the zip of PDF's must be extracted to a directory. Hyperlinks will not function within a zip of PDF's.

1.3.2.3 Expand Current Bookmark

The bookmark pane can highlight the current bookmark associated with the viewer pane by selecting the "expand current bookmark" button, as shown below.



CPUID_Fn8000001F_EDX [Reserved] (SecureEncryptionEdx)
2.2.11 MSR Registers
2.2.11.1 MSRs - MSR0000_xxxx
MSR0000_0010 [Time Stamp Counter] (TSC)
MSR0000_001B [APIC Base Address] (APIC_BAR)
MSR0000_002A [Cluster ID] (EBL_CR_POWERON)
MSR0000_008B [Patch Level] (PATCH_LEVEL)
MSR0000_00E7 [Max Performance

2.2.11.1 MSRs - MSR0000_xxxx

JIRA

See XX [Register Descriptions and Mnemonics] for a des
accessed through x86 WRMSR and RDMSR instructions.

MSR0000_0010 [Time Stamp Counter] (TSC)

Read-write,Volatile. Reset: 0000_0000_0000_0000h.

Core::X86::Msr::TSC_lthree[1:0]_core[3:0]_thread[1:0]; MSR00000010

Bits	Description
63:0	**TSC: time stamp counter**. Read-write,Volatile frequency. The TSC counts at the same rate in guest mode is affected by Core::X86::Msr::Tsc frequency based value (as if TSCRatio==1.0) v

Figure 6: Adobe® Reader Expand Current Bookmark Button

1.4 Conventions

1.4.1 Numbering

- Binary numbers: Binary numbers are indicated either by appending a "b" at the end (e.g., 0110b) or by verilog syntax (e.g., 4'b0110).
- Hexadecimal numbers: Hexadecimal numbers are indicated by appending an "h" to the end (e.g., 45F8h) or by verilog syntax (e.g., 16'h45F8).
- Decimal numbers: A number is decimal if not specified to be binary or hex.
- Exception: Physical register mnemonics are implied to be hex without the h suffix.
- Underscores in numbers: Underscores are used to break up numbers to make them more readable. They do not imply any operation (e.g., 0110_1100).

1.4.2 Arithmetic And Logical Operators

In this document, formulas generally follow Verilog conventions for logic equations.

Table 2: Arithmetic and Logical Operator Definitions

Operator	Definition
{}	Concatenation. Curly brackets are used to indicate a group of bits that are concatenated together. Each set of bits is separated by a comma (e.g., {Addr[3:2], Xlate[3:0]} represents a 6-bit values; the two MSBs are Addr[3:2] and the four LSBs are Xlate[3:0]).
|	Bitwise OR (e.g., 01b | 10b == 11b).
||	Logical OR (e.g., 01b || 10b == 1b). It treats a multi-bit operand as 1 if >= 1 and produces a 1-bit result.
&	Bitwise AND (e.g., 01b & 10b == 00b).
&&	Logical AND (e.g., 01b && 10b == 1b). It treats a multi-bit operand as 1 if >= 1 and produces a 1-bit result.

^	Bitwise exclusive-OR (e.g., 01b ^ 10b == 11b). Sometimes used as "raised to the power of" as well, as indicated by the context in which it is used (e.g., 2^2 == 4).
~	Bitwise NOT (also known as one's complement). (e.g., ~10b == 01b).
!	Logical NOT (e.g., !10b == 0b). It treats a multi-bit operand as 1 if >= 1 and produces a 1-bit result.
<, <=, >, >=, ==, !=	Relational. Less than, Less than or equal, greater, greater than or equal, equal, and not equal.
+, -, *, /, %	Arithmetic. Addition, subtraction, multiplication, division, and modulus.
<<	Bitwise left shift. Shift left first operand by the number of bits specified by the 2nd operand (e.g., 01b << 01b == 10b).
>>	Bitwise right shift. Shift right first operand by the number of bits specified by the 2nd operand (e.g., 10b >> 01b == 01b).
?:	Ternary conditional (e.g., condition ? value if true : value if false).

Table 3: Function Definitions

Term	**Description**
ABS	ABS(integer expression): Remove sign from signed value.
FLOOR	FLOOR(integer expression): Rounds real number down to nearest integer.
CEIL	CEIL(real expression): Rounds real number up to nearest integer.
MIN	MIN(integer expression list): Picks minimum integer or real value of comma separated list.
MAX	MAX(integer expression list): Picks maximum integer or real value of comma separated list.
COUNT	COUNT(integer expression): Returns the number of binary 1's in the integer.
ROUND	ROUND(real expression): Rounds to the nearest integer; halfway rounds away from zero.
UNIT	UNIT(register field reference): Input operand is a register field reference that contains a valid values table that defines a value with a unit (e.g., clocks, ns, ms, etc). This function takes the value in the register field and returns the value associated with the unit (e.g., If the field had a valid value definition where 1010b was defined as 5 ns). Then if the field had the value of 1010b, then UNIT() would return the value 5.
POW	POW(base, exponent): POW(x,y) returns the value x to the power of y.

1.4.2.1 Operator Precedence and Associativity

This document follows C operator precedence and associativity. The following table lists operator precedence (highest to lowest). Their associativity indicates in what order operators of equal precedence in an expression are applied. Parentheses are also used to group subexpressions to force a different precedence; such parenthetical expressions can be nested and are evaluated from inner to outer (e.g., "X = A || !B && C" is the same as "X = A || ((!B) && C)").

Table 4: Operator Precedence and Associativity

Operator	Description	Associativity
!, ~	Logical negation/bitwise complement	right to left
*, /, %	Multiplication/division/modulus	left to right
+, -	Addition/subtraction	left to right
<<, >>	Bitwise shift left, Bitwise shift right	left to right
< , <=, >, >=, ==, !=	Relational operators	left to right
&	Bitwise AND	left to right
^	Bitwise exclusive OR	left to right
|	Bitwise inclusive OR	left to right
&&	Logical AND	left to right

||	Logical OR	left to right
?:	Ternary conditional	right to left

1.4.3 Register Mnemonics

A register mnemonic is a short name that uniquely refers to a register, either all instances of that register, some instances, or a single instance.

Every register instance can be expressed in 2 forms, logical and physical, as defined below.

Table 5: Register Mnemonic Definitions

Term	Description
logical mnemonic	The register mnemonic format that describes the register functionally, what namespace to which the register belongs, a name for the register that connotes its function, and optionally, named parameters that indicate the different function of each instance (e.g., Link::Phy::PciDevVendIDF3). See 1.4.3.1 [Logical Mnemonic].
physical mnemonic	The register mnemonic that is formed based on the physical address used to access the register (e.g., D18F3x00). See 1.4.3.2 [Physical Mnemonic].

1.4.3.1 Logical Mnemonic

The logical mnemonic format consists of a register namespace, a register name, and optionally a register instance specifier (e.g., register namespace::register name register instance specifier).

For Unb::PciDevVendIDF3:
- The register namespace is Unb, which is the UNB IP register namespace.
- The register name is PciDevVendIDF3, which reads as PCICFG device and vendor ID in Function 3.
- There is no register instance specifier because there is just a single instance of this register.

For Dct::Phy::CalMisc2_dct[1:0]_chiplet[BCST,3:0]_pad[BCST,11:0]:
- The register namespace is Dct::Phy, which is the DCT PHY register namespace.
- The register name is CalMisc2, which reads as miscellaneous calibration register 2.
- The register instance specifier is _dct[1:0]_chiplet[BCST,3:0]_pad[BCST,11:0], which indicates that there are 2 DCTPHY instances, each IP for this register has 5 chiplets (0-3 and BCST), and for each chiplet 13 pads (0-11 and BCST). This register has 130 instances. (2*5*13)

Table 6: Logical Mnemonic Definitions

Term	Description
register namespace	A namespace for which the register name must be unique. A register namespace indicates to which IP it belongs and an IP may have multiple namespaces. A namespace is a string that supports a list of "::" separated names. The convention is for the list of names to be hierarchical, with the most significant name first and the least significant name last (e.g., Link::Phy::Rx is the RX component in the Link PHY).
register name	A name that cannotes the function of the register.
register instance specifier	The register instance specifier exists when there is more than one instance for a register. The register instance specifier consists of one or more register instance parameter specifier (e.g., The register instance specifier _dct[1:0]_chiplet[BCST,3:0]_pad[BCST,11:0] consists of 3 register instance parameter specifiers, _dct[1:0], _chiplet[BCST,3:0], and _pad[BCST,11:0]).

register instance parameter specifier	A register instance parameter specifier is of the form _register parameter name[register parameter value list] (e.g., The register instance parameter specifier _dct[1:0] has a register parameter name of dct (The DCT PHY instance name) and a register parameter value list of "1:0" or 2 instances of DCT PHY).
register parameter name	A register parameter name is the name of the number of instances at some level of the logical hierarchy (e.g., The register parameter name dct specifies how many instances of the DCT PHY exist).
register parameter value list	The register parameter value list is the logical name for each instance of the register parameter name (e.g., For _dct[1:0], there are 2 DCT PHY instances, with the logical names 0 and 1, but it should be noted that the logical names 0 and 1 can correspond to physical values other than 0 and 1). It is the purpose of the AddressMappingTable to map these register parameter values to physical address values for the register.

1.4.3.2 Physical Mnemonic

The physical register mnemonic format varies by the access method. The following table describes the supported physical register mnemonic formats.

Table 7: Physical Mnemonic Definitions

Term	Description
PCICFG	The PCICFG, or PCI defined configuration space, physical register mnemonic format is of the form DXFYxZZZ.
BAR	The BAR, or base address register, physical register mnemonic format is of the form PREFIXxZZZ.
MSR	The MSR, or x86 model specific register, physical register mnemonic format is of the form MSRXXXX_XXXX, where XXXX_XXXX is the hexadecimal MSR number. This space is accessed through x86 defined RDMSR and WRMSR instructions.
PMC	The PMC, or x86 performance monitor counter, physical register mnemonic format is any of the forms {PMCxXXX, L2IPMCxXXX, NBPMCxXXX}, where XXX is the performance monitor select.
CPUID	The CPUID, or x86 processor identification state, physical register mnemonic format is of the form CPUID FnXXXX_XXXX_EiX[_xYYY], where XXXX_XXXX is the hex value in the EAX and YYY is the hex value in ECX.

1.4.4 Register Format

A register is a group of register instances that have the same field format (same bit indices and field names).

1.4.4.1 A Register is a group of Register Instances

All instances of a register:

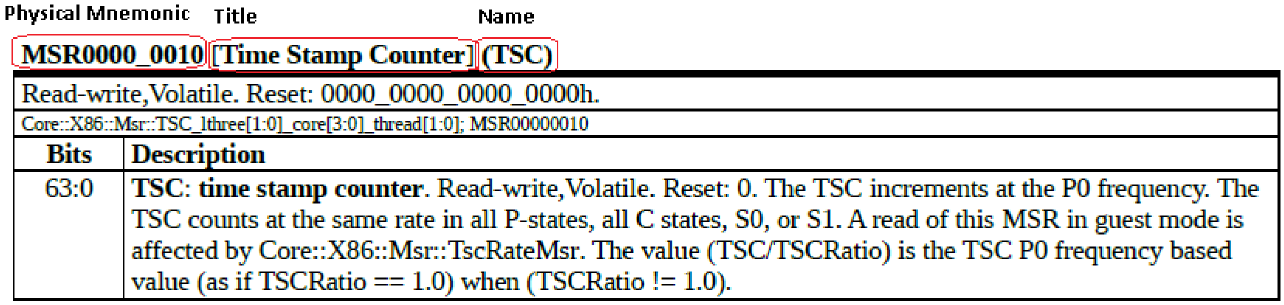
- Have the same:
    - Field bit indices and names
    - Field titles, descriptions, valid values.
    - Register title
    - Register description
- Fields may have different: (instance specific)
    - Access Type. See 1.4.4.10 [Field Access Type].
    - Reset. See 1.4.4.11 [Field Reset].

- Init. See 1.4.4.12 [Field Initialization].
- Check. See 1.4.4.13 [Field Check].

1.4.4.2 Register Physical Mnemonic, Title, and Name

A register definition is identified by a table that starts with a heavy bold line. The information above the bold line in order is:

1. The physical mnemonic of the register.
    - A register that has multiple instances, may have instances that have different access methods, each with it's own physical mnemonic format.
    - In the event that there are multiple physical mnemonic formats, the physical mnemonic format chosen is the most commonly used physical mnemonic.
    - The physical mnemonic is not intended to represent the physical mnemonics of all instances of the register. It is only a visual aid to identify a register when scanning down a list, for readers that prefer to find registers by physical mnemonic. If "..." occurs in the physical mnemonic, the range is first ... last. There is no implication as to how many instances exist between first and last. See 1.4.4.5 [Register Instance Table].
2. The register title in brackets.
3. The register name in parenthesis.



MSR0000_0010 [Time Stamp Counter] (TSC)

Read-write,Volatile. Reset: 0000_0000_0000_0000h.

Core::X86::Msr::TSC_lthree[1:0]_core[3:0]_thread[1:0]; MSR00000010

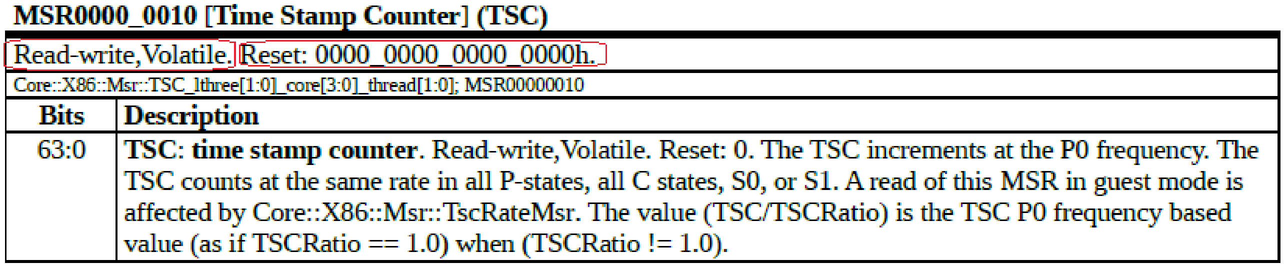
Bits	Description
63:0	**TSC**: **time stamp counter**. Read-write,Volatile. Reset: 0. The TSC increments at the P0 frequency. The TSC counts at the same rate in all P-states, all C states, S0, or S1. A read of this MSR in guest mode is affected by Core::X86::Msr::TscRateMsr. The value (TSC/TSCRatio) is the TSC P0 frequency based value (as if TSCRatio == 1.0) when (TSCRatio != 1.0).

Figure 7: Register Physical Mnemonic, Title, and Name

1.4.4.3 Full Width Register Attributes

The first line that follows the bold line contains the attributes that apply to all fields of the register. This row is rendered as a convenience to the reader and replicates content that exists in the register field.

- AccessType: If all non-reserved fields of a register have the same access type, then the access type is rendered in this row.
    - The supported access types are specified by 1.4.4.10 [Field Access Type].
    - The example figure shows that the access type "Read-write,Volatile" applies to all non-reserved fields of the register.
- Reset: If all non-reserved fields of a register have a constant reset and are all the same type (Warm, Cold, Fixed), then the full width register reset is rendered in this row. The example figure shows the reset "0000_0000_0000_0000h". See 1.4.4.11 [Field Reset].
    - The value zero (0) is assumed for display purposes for all reserved fields.
- If none of the above content is rendered, then this row of the register is not rendered.



MSR0000_0010 [Time Stamp Counter] (TSC)

Read-write,Volatile. Reset: 0000_0000_0000_0000h.

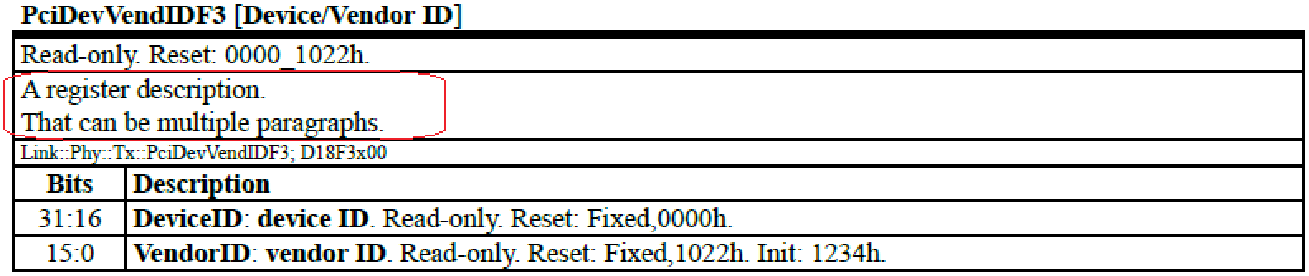
Core::X86::Msr::TSC_lthree[1:0]_core[3:0]_thread[1:0]; MSR00000010

Bits	Description
63:0	**TSC: time stamp counter**. Read-write,Volatile. Reset: 0. The TSC increments at the P0 frequency. The TSC counts at the same rate in all P-states, all C states, S0, or S1. A read of this MSR in guest mode is affected by Core::X86::Msr::TscRateMsr. The value (TSC/TSCRatio) is the TSC P0 frequency based value (as if TSCRatio == 1.0) when (TSCRatio != 1.0).

Figure 8: Full Width Register Attributes

1.4.4.4 Register Description

The register description is optional and appears after the "full width register attributes" row and before the "register instance table" rows. The register description can be one or more paragraphs.



PciDevVendIDF3 [Device/Vendor ID]

Read-only. Reset: 0000_1022h.

A register description.
That can be multiple paragraphs.

Link::Phy::Tx::PciDevVendIDF3; D18F3x00

Bits	Description
31:16	**DeviceID**: **device ID**. Read-only. Reset: Fixed,0000h.
15:0	**VendorID**: **vendor ID**. Read-only. Reset: Fixed,1022h. Init: 1234h.

Figure 9: Register Description

1.4.4.5 Register Instance Table

The zero or more rows of 8-pt font before the Bits/Description row is the register instance table.

The register instance table can generally be described as follows:

- Each row describes the access method of one or more register instances.
- If a row describes two or more instances, then the logical instance range, left to right, corresponds to the physical range, left to right.
- The absence of register instance rows indicates that the register exists for documentation purposes, and no access method is described for the register.

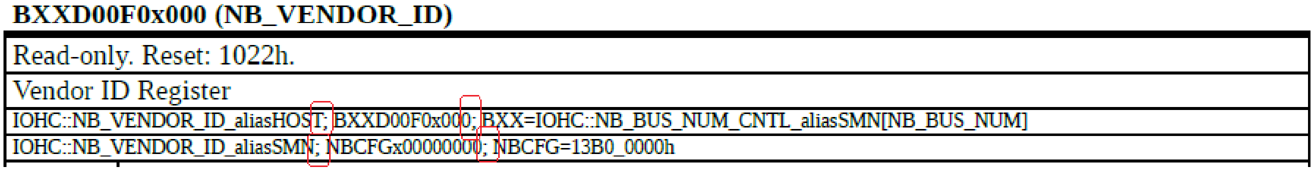
Because there are multiple access methods for all the registers, each of the following subsections describes an aspect of the register instance table in isolation.

1.4.4.5.1 Content Ordering in a Row

Content in a register instance table row is ordered as follows:

- The text up to the first semicolon is the logical mnemonic.
    - See 1.4.3.1 [Logical Mnemonic].
- The text after the first semicolon is the physical mnemonic.
    - See 1.4.3.2 [Physical Mnemonic].

- Optionally, content after the physical mnemonic provides additional information about the access method for the register instances in the row.



BXXD00F0x000 (NB_VENDOR_ID)

Read-only. Reset: 1022h.
Vendor ID Register
IOHC::NB_VENDOR_ID_aliasHOST; BXXD00F0x000; BXX=IOHC::NB_BUS_NUM_CNTL_aliasSMN[NB_BUS_NUM]
IOHC::NB_VENDOR_ID_aliasSMN; NBCFGx00000000; NBCFG=13B0_0000h

Figure 10: Register Instance Table: Content Ordering in a Row

1.4.4.5.2 Multiple Instances Per Row

Multiple instances in a row is represented by a single dimension "range" in the logical mnemonic and the physical mnemonic.

The single dimension order of instances is the same for both the logical and physical mnemonic. The first logical mnemonic is associated with the first physical mnemonic, so forth for the 2nd, up until the last.

- Brackets indicates a list, most significant to least significant.
- The ":" character indicates a continuous range between 2 values.
- The "," character separates non-contiguous values.
- There are some cases where more than one logical mnemonic maps to a single physical mnemonic.

Note that it is implied that the MSR {lthree,core,thread} parameters are not part of a range.

Example:
NAMESP::REGNAME_inst[BLOCK[5:0],BCST]_aliasHOST; FFF1x00000088_x[000[B:6]_0001,00000000]

- There are 7 instances.
- NAMESP is the namespace.
- 6 instances are represented by the sub-range 000[B:6]_0001.
- _instBCST corresponds to FFF1x00000088_x00000000.
- _inst BLOCK 0 corresponds to FFF1x00000088_x00060001.
- ...
- _inst BLOCK 5 corresponds to FFF1x00000088_x000B0001.

1.4.4.5.3 MSR Access Method

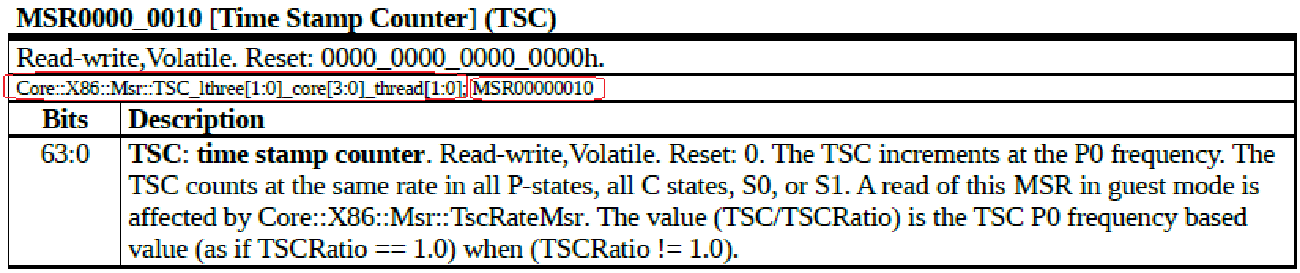
The MSR parameters {lthree,core,thread} are implied by the identity of the core on which the RDMSR/WRMSR is being executed, and therefore are not represented in the physical mnemonic.

MSRs that are:

- per-thread have the {lthree,core,thread} parameters.
- per-core do not have the thread parameter.
- per-L3 do not have the {core,thread} parameters.
- common to all L3's do not have the {lthree,core,thread} parameters.

1.4.4.5.3.1 MSR Per-Thread Example

An MSR that is per-thread has all three {lthree,core,thread} parameters and all instances have the same physical mnemonic.



MSR0000_0010 [Time Stamp Counter] (TSC)

Read-write,Volatile. Reset: 0000_0000_0000_0000h.

Core::X86::Msr::TSC_lthree[1:0]_core[3:0]_thread[1:0]; MSR00000010

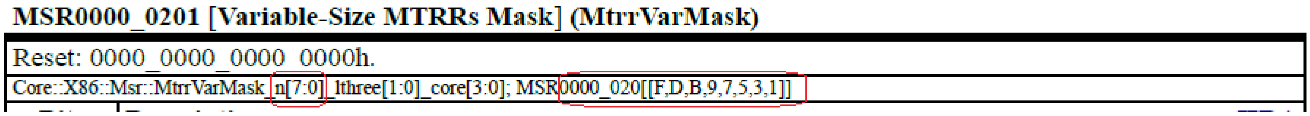
Bits	Description
63:0	**TSC: time stamp counter**. Read-write,Volatile. Reset: 0. The TSC increments at the P0 frequency. The TSC counts at the same rate in all P-states, all C states, S0, or S1. A read of this MSR in guest mode is affected by Core::X86::Msr::TscRateMsr. The value (TSC/TSCRatio) is the TSC P0 frequency based value (as if TSCRatio == 1.0) when (TSCRatio != 1.0).

Figure 11: Register Instance Table: MSR Example

1.4.4.5.3.2 MSR Range Example

An MSR can exist as a range for a parameter other than the {lthree,core,thread} parameters.

In the following example the n parameter is a range. The _n0 value corresponds to MSR0000_0201, and so on.



MSR0000_0201 [Variable-Size MTRRs Mask] (MtrrVarMask)

Reset: 0000_0000_0000_0000h.

Core::X86::Msr::MtrrVarMask_n[7:0]_lthree[1:0]_core[3:0]; MSR0000_020[[F,D,B,9,7,5,3,1]]

Figure 12: Register Instance Table: MSR Range Example

1.4.4.5.4 BAR Access Method

The BAR access method is indicated by a physical mnemonic that has the form PREFIXxNUMBER.

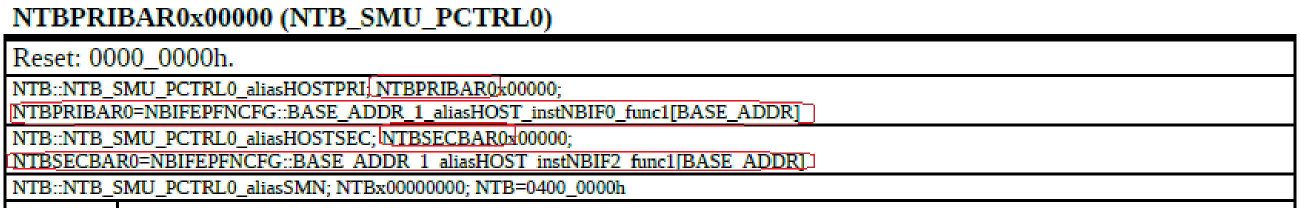
- Example: APICx0000. The BAR prefix is "APIC".

The BAR prefix represents either a constant or an expression that consists of a register reference.

1.4.4.5.4.1 BAR as a Register Reference

A relocatable BAR is when the base of an IP is not a constant.

- The prefix NTBPRIBAR0 represents the base of the IP, the value of which comes from the register NBIFEPFNCFG::BASE_ADDR_1_aliasHOST_instNBIF0_func1[BASE_ADDR].



NTBPRIBAR0x00000 (NTB_SMU_PCTRL0)

Reset: 0000_0000h.

NTB::NTB_SMU_PCTRL0_aliasHOSTPRI; NTBPRIBAR0x00000;
NTBPRIBAR0=NBIFEPFNCFG::BASE_ADDR_1_aliasHOST_instNBIF0_func1[BASE_ADDR]

NTB::NTB_SMU_PCTRL0_aliasHOSTSEC; NTBSECBAR0x00000;
NTBSECBAR0=NBIFEPFNCFG::BASE_ADDR_1_aliasHOST_instNBIF2_func1[BASE_ADDR]

NTB::NTB_SMU_PCTRL0_aliasSMN; NTBx00000000; NTB=0400_0000h

Figure 13: Register Instance Table: BAR as Register Reference

1.4.4.5.5 PCICFG Access Method

The PCICFG access method is indicated by a physical mnemonic that has the form DXXFXxNUMBER. There are 2 cases:

- Bus omitted and implied to be 00h.
- Bus represented as BXX and indicates that the bus is indicated by a register field.

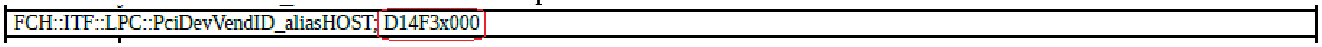
Example:

- Example: D18F0x000. (The bus, when omitted, is implied to be 00h)
- Example: BXXD0F0x000. (The bus as an expression that includes a register reference)

1.4.4.5.5.1 PCICFG Bus Implied to be 00h

Example:

- The absence of a B before the D14 implies that the bus is 0.



FCH::ITF::LPC::PciDevVendID_aliasHOST; D14F3x000

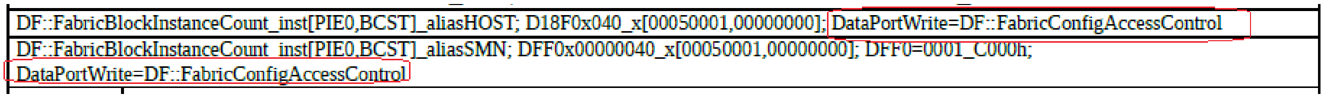
Figure 14: Register Instance Table: Bus Implied to be 00h

1.4.4.5.6 Data Port Access Method

A data port requires that the data port select be written before the register is accessed via the data port.

Example:

- The data port select value follows the "_x".
- The data port select register follows the "DataPortWrite=".



DF::FabricBlockInstanceCount_inst[PIE0,BCST]_aliasHOST; D18F0x040_x[00050001,00000000]; DataPortWrite=DF::FabricConfigAccessControl

DF::FabricBlockInstanceCount_inst[PIE0,BCST]_aliasSMN; DFF0x00000040_x[00050001,00000000]; DFF0=0001_C000h; DataPortWrite=DF::FabricConfigAccessControl

Figure 15: Register Instance Table: Data Port Select

1.4.4.6 Register Field Format

The register field definition are all rows that follow the Bits/Description row. Each field row represents the definition of a bit range, with the bit ranges ordered from most to least significant. There are 2 columns, with the left column defining the field bit range, and the right column containing the field definition.

There are 2 field definition formats, simple and complex. If the description can be described in the simple one paragraph format then the simple format is used, else the complex format is used.

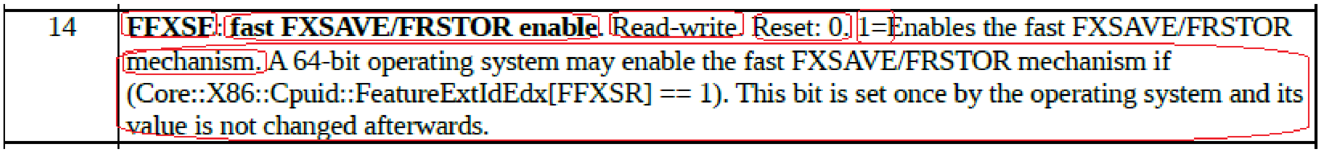
1.4.4.7 Simple Register Field Format

The simple register format compresses all content into a single paragraph with the following implied order:

1. Field name (required)
    - Allowed to be Reserved. See 1.4.4.9 [Field Name is Reserved].
    - "FFXSE" in the example figure.
2. Field title
    - "fast FXSAVE/FRSTOR enable" in the example figure.
3. Field Access Type. See 1.4.4.10 [Field Access Type].
    - In the example figure the access type is "Read-write".

4. Field Reset. See 1.4.4.11 [Field Reset].
    - In the example figure the reset is warm reset and "0".
5. Field Init. See 1.4.4.12 [Field Initialization].
6. Field Check. See 1.4.4.13 [Field Check].
7. Field Valid Values, if the valid values are single bit (e.g., 0=, 1=). See 1.4.4.14 [Field Valid Values].
    - In the example figure the 1= definition begins with "Enables" and ends with "mechanism".
    - In the example figure there is no 0= definition.
8. Field description, if it is a single paragraph.
    - In the example figure the field description begins with "This is" and ends with "afterwards".

All fields that don't exist are omitted.



14	**FFXSE**: **fast FXSAVE/FRSTOR enable**. Read-write. Reset: 0. 1=Enables the fast FXSAVE/FRSTOR mechanism. A 64-bit operating system may enable the fast FXSAVE/FRSTOR mechanism if (Core::X86::Cpuid::FeatureExtIdEdx[FFXSR] == 1). This bit is set once by the operating system and its value is not changed afterwards.

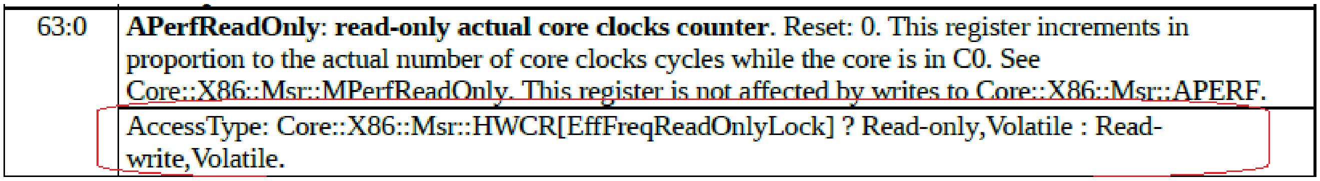
Figure 16: Simple Register Field Example

1.4.4.8 Complex Register Field Format

Content that can't be expressed in the single paragraph format is broken out to a separate sub-row (a definition column row).

Additional sub-rows are added in the following order:
1. Complex expression for {Reset,AccessType,Init,Check}.
2. Instance specific {Reset,AccessType,Init,Check} values.
3. Description, if more than 1 paragraph.
4. Valid values, if more than 0=/1=. Or a Valid bit table. (see figure)

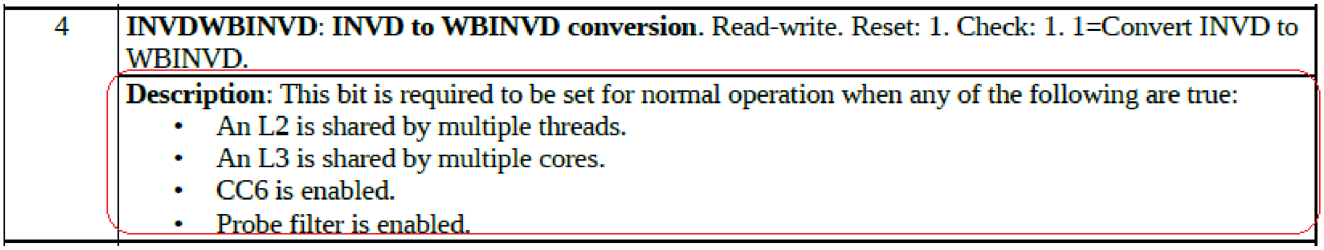
The following figure highlights a complex access type specification.



63:0	**APerfReadOnly**: **read-only actual core clocks counter**. Reset: 0. This register increments in proportion to the actual number of core clocks cycles while the core is in C0. See Core::X86::Msr::MPerfReadOnly. This register is not affected by writes to Core::X86::Msr::APERF.
	AccessType: Core::X86::Msr::HWCR[EffFreqReadOnlyLock] ? Read-only,Volatile : Read-write,Volatile.

Figure 17: Register Field Sub-Row for {Reset,AccessType,Init,Check}

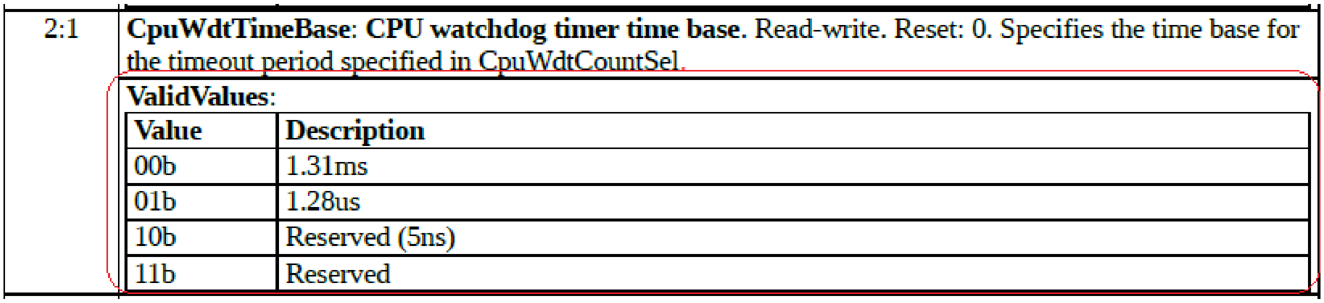
The following figure highlights a complex description specification.



4	**INVDWBINVD**: **INVD to WBINVD conversion**. Read-write. Reset: 1. Check: 1. 1=Convert INVD to WBINVD.
	Description: This bit is required to be set for normal operation when any of the following are true: • An L2 is shared by multiple threads. • An L3 is shared by multiple cores. • CC6 is enabled. • Probe filter is enabled.

Figure 18: Register Field Sub-Row for Description

The following figure highlights a complex valid value table, used either when the field is more than 1 bit or when the definition is more than a single sentence.



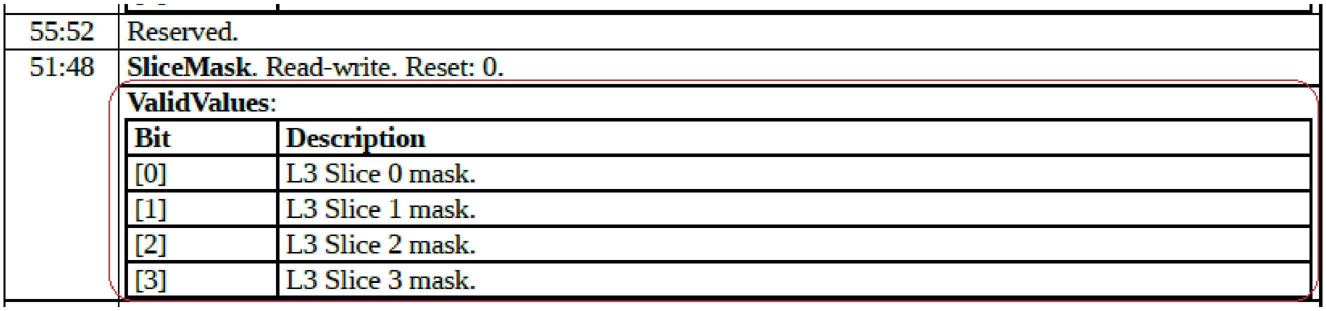
2:1	**CpuWdtTimeBase**: **CPU watchdog timer time base**. Read-write. Reset: 0. Specifies the time base for the timeout period specified in CpuWdtCountSel.

ValidValues:

Value	Description
00b	1.31ms
01b	1.28us
10b	Reserved (5ns)
11b	Reserved

Figure 19: Register Field Sub-Row for Valid Value Table

The following figure highlights a valid bit table which is used when each bit has a specific function.



55:52	Reserved.
51:48	**SliceMask**. Read-write. Reset: 0.

ValidValues:

Bit	Description
[0]	L3 Slice 0 mask.
[1]	L3 Slice 1 mask.
[2]	L3 Slice 2 mask.
[3]	L3 Slice 3 mask.

Figure 20: Register Field Sub-Row for Valid Bit Table

1.4.4.9 Field Name is Reserved

When a register field name is Reserved, and it does not explicitly specify an access type, then the implied access type is "Reserved-write-as-read".

- The Reserved-write-as-read access type is:
    - Reads must not depend on the read value.
    - Writes must only write the value that was read.

1.4.4.10 Field Access Type

The AccessType keyword is optional and specifies the access type for a register field. The access type for a field is a comma separated list of the following access types.

Table 8: AccessType Definitions

Term	Description
Read-only	Readable; writes are ignored.
Read-write	Readable and writable.
Read	Readable; must be associated with one of the following {Write-once, Write-1-only, Write-1-to-clear, Error-on-write}.
Write-once	Capable of being written once; all subsequent writes have no effect. If not associated with Read,

	then reads are undefined.
Write-only	Writable. Reads are undefined.
Write-1-only	Writing a 1 sets to a 1; Writing a 0 has no effect. If not associated with Read, then reads are undefined.
Write-1-to-clear	Writing a 1 clears to a 0; Writing a 0 has no effect. If not associated with Read, then reads are undefined.
Write-0-only	Writing a 0 clears to a 0; Writing a 1 has no effect. If not associated with Read, then reads are undefined.
Error-on-read	Error occurs on read.
Error-on-write	Error occurs on write.
Error-on-write-0	Error occurs on bitwise write of 0.
Error-on-write-1	Error occurs on bitwise write of 1.
Inaccessible	Not readable or writable (e.g., Hide ? Inaccessible : Read-Write).
Configurable	Indicates that the access type is configurable as described by the documentation.
Unpredictable	The behavior of both reads and writes is unpredictable.
Reserved-write-as-1	Reads are undefined. Must always write 1.
Reserved-write-as-0	Reads are undefined. Must always write 0.
Volatile	Indicates that a register field value may be modified by hardware, firmware, or microcode when fetching the first instruction and/or might have read or write side effects. No read may depend on the results of a previous read and no write may be omitted based on the value of a previous read or write.

1.4.4.10.1 Conditional Access Type Expression

The ternary operator can be used to express an access type that is conditional on an expression that can contain any of the following:

- A register field value
- A constant
- A definition

1.4.4.11 Field Reset

The Reset keyword is optional and specifies the value for a register field at the time that hardware exits reset, before firmware initialization initiates.

Unless preceded by one of the following prefixes, the reset value is called warm reset and the value is applied at both warm and cold reset.

Table 9: Reset Type Definitions

Type	Description
Cold	Cold reset. The value is applied only at cold reset.
Fixed	The value applies at all time.

1.4.4.12 Field Initialization

The Init keyword is optional and specifies an initialization recommendation for a register field.

If present, then there is an optional prefix that specifies the owner of the initialization. See Table 10 [Init Type Definitions].

- Example: Init: BIOS,2'b00. //A initialization recommendation for a field to be programmed by BIOS.

Table 10: Init Type Definitions

Type	Description
BIOS	Initialized by AMD provided AMD Generic Encapsulated Software Architecture (AGESA™) x86 software.
SBIOS	Initialized by OEM or IBV provided x86 software, also called Platform BIOS.
OS	Initialized by OS or Driver.

1.4.4.13 Field Check

The Check keyword is optional and specifies the value that is recommended for firmware/software to write for a register field. It is a recommendation, not a requirement, and may not under all circumstances be what software programs.

1.4.4.14 Field Valid Values

A register can optionally have either a valid values table or a valid bit table:

- A valid values table specifies the definition for specific field values.
- A valid bit table specifies the definition for specific field bits.

1.5 Definitions

Table 11: Definitions

Term	**Description**
AGESA™	AMD Generic Encapsulated Software Architecture.
APML	Advanced Platform Management Link.
BCD	Binary Coded Decimal number format.
BCS	Base Configuration Space.
BIST	Built-In Self-Test. Hardware within the processor that generates test patterns and verifies that they are stored correctly (in the case of memories) or received without error (in the case of links).
Boot VID	Boot Voltage ID. This is the VDD and VDDNB voltage level that the processor requests from the external voltage regulator during the initial phase of the cold boot sequence.
C-states	These are ACPI defined core power states. C0 is operational. All other C-states are low-power states in which the processor is not executing code. See docACPI.
CCD	Core-Complex Die.
Cold reset	PWROK is de-asserted and RESET_L is asserted.
COF	Current operating frequency of a given clock domain.
DID	Divisor Identifier. Specifies the post-PLL divisor used to reduce the COF.
Doubleword	A 32-bit value.
DW	Doubleword.
ECS	Extended Configuration Space.
FCH	The integrated platform subsystem that contains the IO interfaces and bridges them to the system BIOS. Previously included in the Southbridge.
FID	Frequency Identifier. Specifies the PLL frequency multiplier for a given clock domain.
GB	Gbyte or Gigabyte; 1,073,741,824 bytes.
GT/s	Giga-Transfers per second.

IFCM	Isochronous flow-control mode, as defined in the link specification.
IO configuration	Access to configuration space though IO ports CF8h and CFCh.
IOD	IO die.
IP	In electronic design, a semiconductor Intellectual Property, IP, or IP block is a reusable unit of logic, cell, or integrated circuit layout design that is the intellectual property of one party.
KB	Kbyte or Kilobyte; 1024 bytes.
Master abort	This is a PCI-defined term that is applied to transactions on other than PCI buses. It indicates that the transaction is terminated without affecting the intended target; reads return all 1s; write are discarded; the master abort error code is returned in the response, if applicable; master abort error bits are set if applicable.
MB	Megabyte; 1024 KB.
MMIO	Memory-Mapped Input-Output range. This is physical address space that is mapped to the IO functions such as the IO links or MMIO configuration.
MMIO configuration	Access to configuration space through memory space.
OW	Octword. An 128-bit value.
PCIe®	PCI Express.
PCS	Physical Coding Sublayer.
Processor	Die of the System on Chip (SoC) covered by this PPR. See 1.8 [Processor Overview].
QW	Quadword. A 64-bit value.
RAS	Reliability, availability and serviceability (industry term). See 3.1 [Machine Check Architecture].
REFCLK	Reference clock. Refers to the clock frequency (100 MHz) or the clock period (10 ns) depending on the context used.
RX	Receiver.
Shutdown	A state in which the affected core waits for either INIT, RESET, or NMI. When shutdown state is entered, a shutdown special cycle is sent on the IO links.
SMAF	System Management Action Field. This is the code passed from the SMC to the processors in STPCLK assertion messages.
SMC	System Management Controller. This is the platform device that communicates system management state information to the processor through an IO link, typically the system IO hub.
Speculative event	A performance monitor event counter that counts all occurrences of the event even if the event occurs during speculative code execution.
TDC	Thermal Design Current.
TDP	Thermal Design Power. A power consumption parameter that is used in conjunction with thermal specifications to design appropriate cooling solutions for the processor.
Token	A scheduler entry used in various Northbridge queues to track outstanding requests.
TOM	Top of Memory.
TOM2	Top of extended Memory.
TX	Transmitter.
UMI	Unified Media Interface. The link between the processor and the FCH.
VID	Voltage level identifier.
Warm reset	RESET_L is asserted only (while PWROK stays high).
XBAR	Cross bar; command packet switch.

1.6 Changes Between Revisions and Product Variations

1.6.1 Revision Conventions

The processor revision is specified by CPUID_Fn00000001_EAX (FamModStep) or CPUID_Fn80000001_EAX (FamModStepExt). This document uses a revision letter instead of specific model numbers. Where applicable, the processor stepping is indicated after the revision letter. All behavior marked with a revision letter apply to future revisions unless they are superseded by a change in a later revision. See the revision guide in 1.2 [Reference Documents] for additional information about revision determination.

1.7 Package

1.7.1 Package type

The following packages are supported.

Table 12: Package Definitions

Package	Description
SP3	Server, up to eight CCD die plus one IOD die MCM, single and dual socket.

1.8 Processor Overview

1.8.1 Features

Family 19h Models 00h-0Fh is a server-class microprocessor System-On-a-Chip (SOC) multi-chip module (MCM). It is built using a mixed-technology chiplet approach - up to eight core/cache complex dies (CCD) and a single I/O die (IOD).

MCM

SP3 Server

As a server product, the SP3 consists of two to eight CCDs plus one IOD, in an SP3 multi-chip module (MCM).

One or two sockets may be present (1P or 2P). If two, they are interconnected using three or four xGMI ports depending on system bandwidth requirements. Each xGMI port utilizes a bidirectional x16 SERDES interconnect.

Core/Cache Die

- Each CCD contains a single Core Complex (CCXs). A single CCX consists of:
  - Eight cores. Each core may run in single-thread mode (1T) or two-thread SMT mode (2T), for a total of up to sixteen threads (16T) per CCX.
  - The cache system with 512KB of L2 per core, for a total of 4MB L2 per CCX and 4 MB of L3 slice per core shared across all cores in the CCX, for a total 32MB L3 per CCX.
  - Thus, a CCD contains eight cores, 4MB of L2 cache and 32MB of L3 cache.
    - NOTE: Some product variants will have different L3 sizes. Core::X86::Cpuid::L3CacheId (CPUID_Fn80000006_EDX) register should be examined to determine the L3 size on a given product.

I/O Die

The IOD integrates:

- Infinity Fabric™

- DDR Memory interfaces
    - Up to eight Unified Memory Controllers (UMC), each supporting one DRAM channel.
        - 64-bit data plus 8-bit ECC
        - Up to two DIMMs per channel
        - DDR4 transfer rates from 1333MT/s to 3200MT/s

- System Management Unit (SMU)

- Four instances of NorthBridge IO (NBIO), each of which includes:
    - Two 8x16 PCIe® Gen4 controllers.
    - One instance includes a 2x2 PCIe® Gen 2 controller, which can be used to attach a Baseband Management Controller (BMC).

- Server Controller Hub
    - ACPI - Advanced Configuration and Power Interface, power management
    - CLKGEN/CGPLL - clock generation
    - GPIO (varying number depending on muxing) - general purpose IO
    - I2C (6 ports) - Inter-Integrated Circuit
    - LPC - Low Pin Count interface
    - RTC - Real-Time Clock
    - SD - Memory card interface
    - SMBus (2 ports) - System Management Bus
    - SPI/eSPI - Serial Peripheral Interface, embedded SPI
    - UART (4 ports) - Universal Asynchronous Receiver/Transmitter

- SATA
    - Four SATA controllers, each with up to 8 lanes of SATA Gen1/Gen2/Gen3
    - SGPIO - serial GPIO, can be used for activity monitors.

- USB
    - Four USB3.1 Gen 1 (USB3.0) ports, including support for legacy USB speeds.

1.9 System Overview

1.9.1 Mixed Processor Revision Supports

AMD Family 19h processors with different OPNs or different revisions cannot be mixed in a multiprocessor system. If the BIOS detects and unsupported configuration, the system will halt prior to x86 core release and signal a port 80 error code.

1.9.2 SP3 MCM Server Single-Socket

SP3-based single-socket (1P) systems target a new paradigm of high-end single socket commercial servers. These servers have a large number of memory channels, memory capacity, and I/O ports and target the following form factors:

- 1U traditional rack servers
- Blade servers
- Multi node, 1P "twins" type of shared infrastructure servers

- EATX or EATX+ servers

The characteristics and capabilities of the SP3 product in single-socket systems are shown in the following table:

Table 13: SP3 1P Capabilities

	SP3 1P Configuration
Module Type	MCM socketed LGA
Module size, pad pitch	58.5 x 75.4 @ 1.00 x 0.87mm pitch
Socket size	78.9 x 119.3 mm, heatsink actuated
Cores / module	64
Memory channels	8
Max DIMM/channel	2
DIMM Type	R/LR/NV DIMM, 3DS
Combo links/module (note1)	Eight – 16-bit links
xGMI links (16-bit)	N/A
Max PCIe®/module	128 lanes up to Gen4, plus 2 lanes up to Gen2
Max SATA/module (note2)	32
Native I/O	USB3/2, SPI/eSPI, LPC, UART, I2C, RTC, Power control, etc.

Notes:
1: Combo links can take the form of PCIe® up to Gen4 or SATA, with configuration restrictions.
2: These functions are in lieu of PCIe® on those ports (e.g., a group of 8 SATA displaces 8 PCIe® lanes).

1.9.2.1 SP3 1P Memory Support

The SP3 MCM provides eight memory channels. Each channel can accommodate two DDR4 DIMM connectors. The supported DIMM types are:

- One and two rank DDR4 RDIMM.
- Four and eight physical rank DDR4 LRDIMM.
- DDR4 3DS DIMM.
- NVDIMM-N.

1.9.2.2 SP3 1P I/O Support

The I/O interfaces for the Family 19h Models 00h-0Fh die are configurable and extremely flexible. The SP3 MCM provides eight 16-bit "combo" links, half are type A and the other half are type B.

Type A links can be configured to support:

- One 16-lane PCIe® Gen4 controller with 8 ports.
- Up to eight SATA 3 ports.

Type B links can support:

- One 16 lane PCIe® Gen4 controller with 8 ports.

Up to two additional ports of PCIe® Gen2 are supported on the two WAFL lanes in a 1P system.

This configuration capability of the links is shown in Figure 24 [Link Capabilities].

In single-socket systems, there are 8 links, 4 type A and 4 type B, all of which can be used as I/O. As an example, a one socket system that maximizes PCIe® connectivity can support up to 128 lanes of PCIe® Gen4 plus 2 lanes of PCIe® Gen2. The 16-lane links can be divided up into smaller link widths as long as there are no more than 8 ports per 16-lane group.

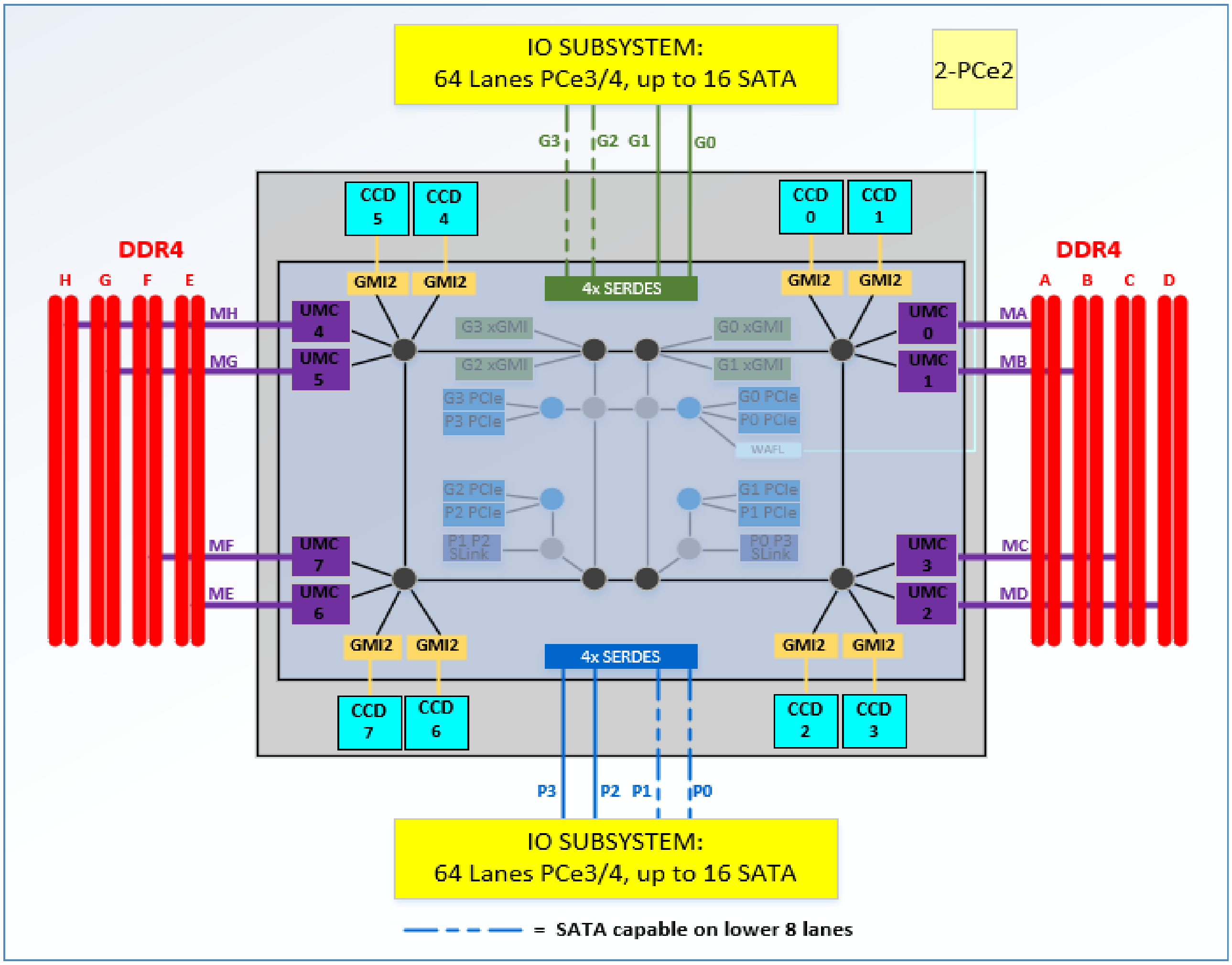


Figure 21: 1P System Block Diagram

1.9.3 SP3 MCM Server Dual-Socket

SP3-based dual-socket (2P) systems target the high end of the dual-socket commercial space. The primary form factors are:

- 2U traditional rack servers.
- Blade servers.
- "Twins" servers. These have two dual-socket systems on a single board with shared infrastructure such as power and network aggregation.

The characteristics and capabilities of the SP3 product in dual-socket systems are shown in the following table:

Table 14: SP3 2P Capabilities

	SP3 2P Configuration
Module Type	MCM socketed LGA
Module size, pad pitch	58.5 x 75.4 @ 1.00 x 0.87mm pitch
Socket size	78.9 x 119.3 mm, heatsink actuated
Cores / module	64
Memory chan/module	8
Max DIMM/channel	2
DIMM Type	R/LR/NV DIMM, 3DS
Combo links/module (note1)	8 – 16-bit links
xGMI links (16-bit)	3 or 4
Max PCIe®/module	64 lanes up to Gen4, plus 1 lane up to Gen2
Max SATA/module (note2)	16
Native I/O	USB3/2, SPI/eSPI, LPC, UART, I2C, RTC, Power control, etc.

Notes:
1: Combo links can take the form of xGMI, PCIe®, with configuration restrictions.
2: These functions are in lieu of PCIe® on those lanes (e.g., a group of 8 SATA displaces 8 PCIe® lanes).

1.9.3.1 SP3 2P Coherent Interconnect Topology

Two SP3 MCM are interconnected using four, or three, xGMI links, depending on system bandwidth requirements. Refer to Figure 22 [2P System Block Diagram, with 4 xGMI links] and Figure 23 [2P System Block Diagram, with 3 xGMI links], accordingly.

1.9.3.2 SP3 2P Memory Support

Each SP3 MCM provides eight memory channels resulting in 16 memory channels in a dual-socket system.
Each channel can accommodate two DDR4 DIMM connectors. The supported DIMM types are:

- One and two rank DDR4 RDIMM.
- Four and eight physical rank DDR4 LRDIMM.
- DDR4 3DS DIMM.
- NVDIMM-N.

1.9.3.3 SP3 2P I/O Support

The I/O interfaces for the Family 19h Models 00h-0Fh die are configurable and extremely flexible. Each IOD die generates eight 16-bit "combo" links, one is type A and the other is type B.

Type A links can be configured to support:

- One 16-bit xGMI.
- One 16-lane PCIe® Gen4 controller with 8 ports.
- Up to eight SATA 3 ports.

Type B links can support:

- One 16-bit xGMI.
- One 16 lane PCIe® Gen4 controller with 8 ports.

An additional port of PCIe® up to Gen2 is supported on the WAFL lane of each socket that is not used for socket-to-socket connectivity.

The configuration capability of the links is shown in Figure 24 [Link Capabilities].

In dual-socket systems, three or four 16-bit links are used for xGMI interconnect between MCMs. That leaves four or five 16-bit links per MCM (two or three type A, two or three type B) remaining for non-xGMI I/O.

As an example, a dual-socket system that maximizes PCIe® connectivity can support up to 160 lanes of PCIe® Gen4 plus 2 lanes of PCIe® Gen2, 80 + 1 lanes per MCM. The 16-lane links can be divided up into smaller link widths as long as there are no more than 8 ports per 16-lane group. Each MCM is connected by three or four external GMI (xGMI) links.

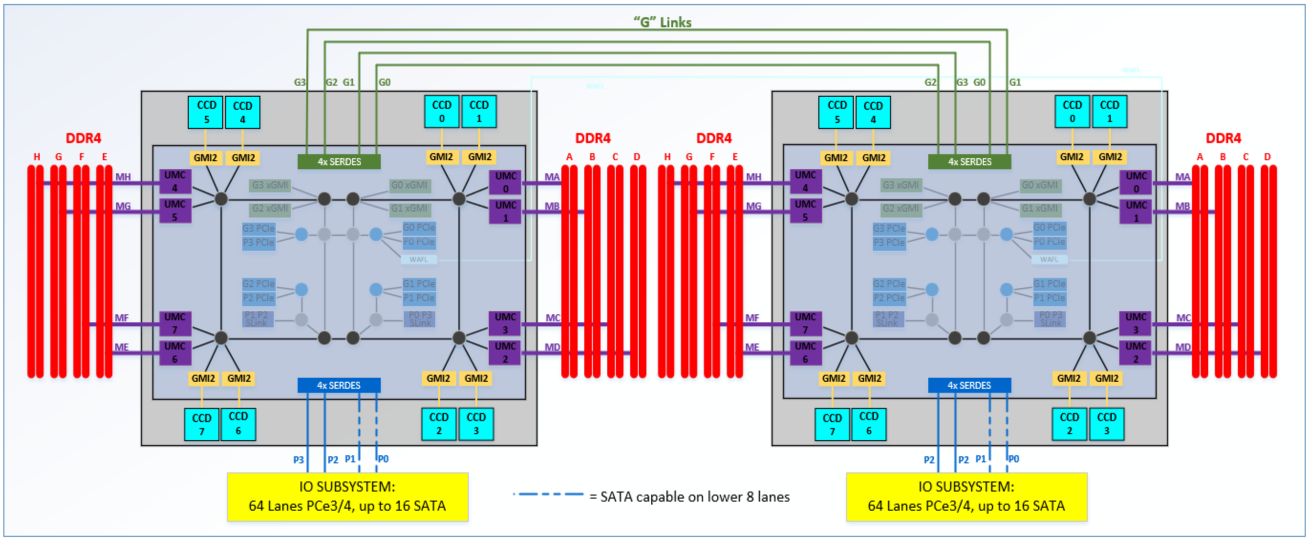


Figure 22: 2P System Block Diagram, with 4 xGMI links

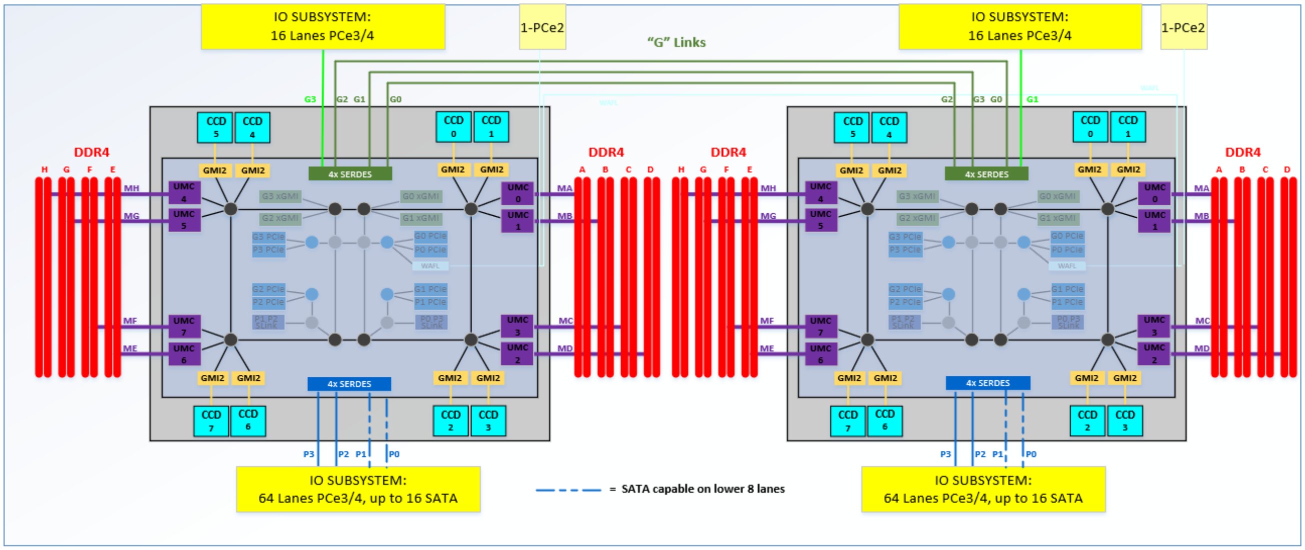


Figure 23: 2P System Block Diagram, with 3 xGMI links

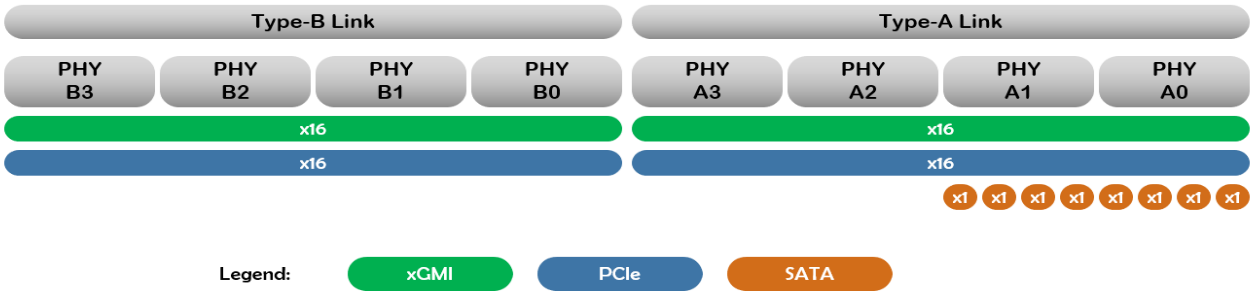


Figure 24: Link Capabilities

For additional detail and restrictions, refer to "Chapter 2.2.2 AMD Family 19h Models 00h-0Fh Processors – I/O Overview" in the AMD Socket SP3 Processor Motherboard Design Guide (MBDG) at 1.2 [Reference Documents].

2 Core Complex (CCX)

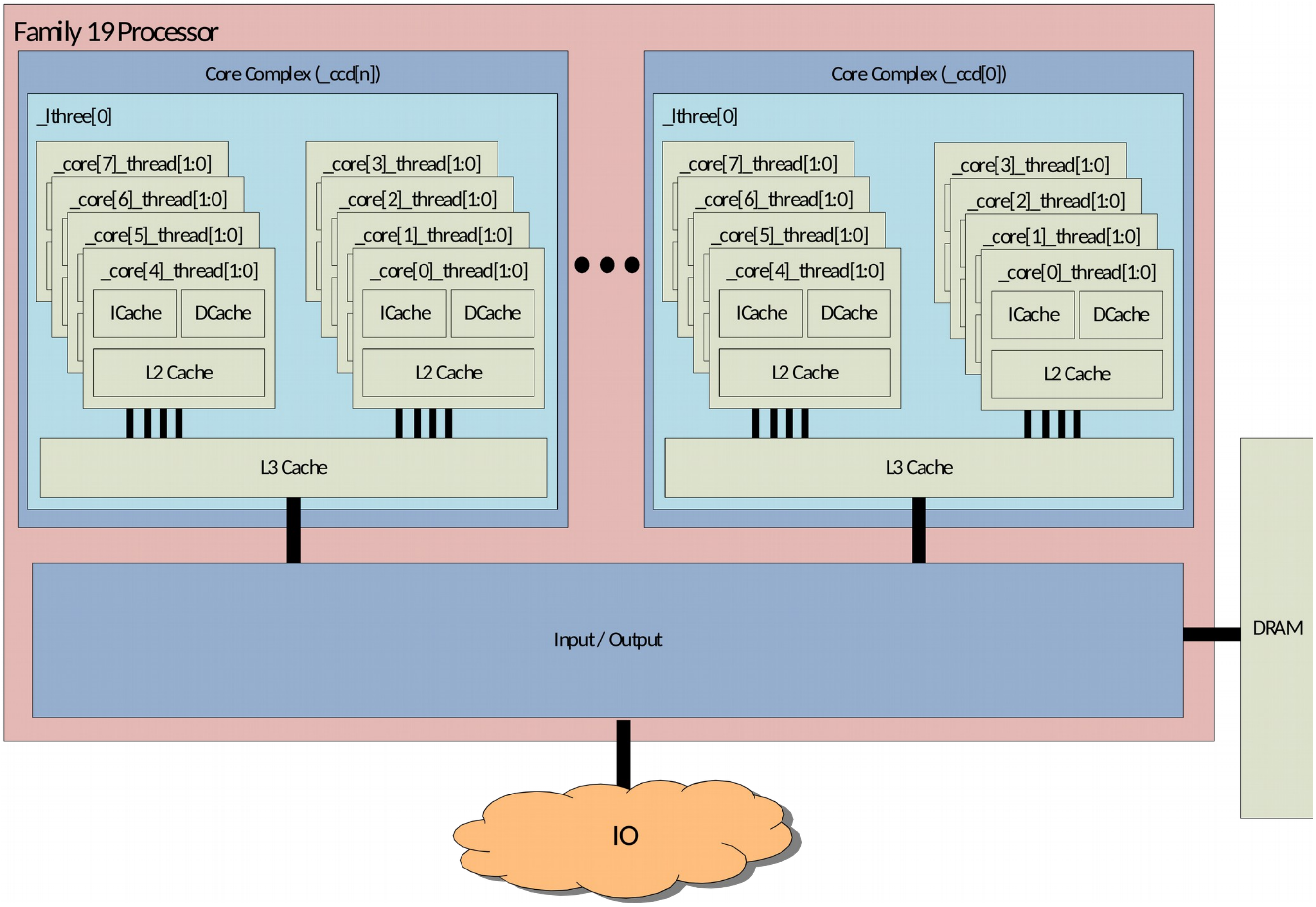


Figure 25: Overview of Family 19 Core Component Numbering

2.1 Processor x86 Core

2.1.1 Core Functional Information

2.1.2 Core Definitions

Table 15: Definitions

Term	Description
BSC	Boot strap core. Core 0 of the BSP.
BSP	Boot strap processor.
Canonical-address	An address in which the state of the most-significant implemented bit is duplicated in all the remaining higher-order bits, up to bit[63].
CCX	Core Complex where more than one core shares L3 resources.

CMP	Specifies the core number.
Core	The instruction execution unit of the processor when the term Core is used in a x86 core context.
CoreCOF	Core current operating frequency in MHz. CoreCOF = (Core::X86::Msr::PStateDef[CpuFid[7:0]]/Core::X86::Msr::PStateDef[CpuDfsId])*200. A nominal frequency reduction can occur if spread spectrum clocking is enabled.
CPL	Current Privilege Level of the running task when the term CPL is used in a x86 core context.
CpuCoreNum	Specifies the core number.
#GP	A general-protection exception.
#GP(0)	Notation indicating a general-protection exception (#GP) with error code of 0.
IBS	Instruction based sampling.
IO configuration	Access to configuration space through IO ports CF8h and CFCh.
IORR	IO range register.
L1 cache	The level 1 caches (instruction cache and the data cache).
L2 cache	The level 2 caches.
L3	Level 3 Cache. The L3 term is also in Addrmaps to enumerate CCX units.
L3 cache	Level 3 Cache.
Linear (virtual) address	The address generated by a core after the segment is applied.
LINT	Local interrupt.
Logical address	The address generated by a core before the segment is applied.
LRU	Least recently used.
LVT	Local vector table. A collection of APIC registers that define interrupts for local events (e.g., APIC[530:500] [Extended Interrupt [3:0] Local Vector Table]).
Macro-op	The front-end of the pipeline breaks instructions into macro-ops and transfers (dispatches) them to the back-end of the pipeline for scheduling and execution. See Software Optimization Guide.
Micro-op	Processor schedulers break down macro-ops into sequences of even simpler instructions called micro-ops, each of which specifies a single primitive operation. See Software Optimization Guide.
NBC	NBC = (CPUID Fn00000001_EBX[LocalApicId[3:0]]==0). Node Base Core. The lowest numbered core in the node.
MTRR	Memory-type range register. The MTRRs specify the type of memory associated with various memory ranges.
NTA	Non-Temporal Access.
PTE	Page table entry.
SMI	System management interrupt.
SMM	System Management Mode.
SMT	Simultaneous multithreading. See Core::X86::Cpuid::CoreId[ThreadsPerCore].
Speculative event	A performance monitor event counter that counts all occurrences of the event even if the event occurs during speculative code execution.
SVM	Secure virtual machine.
Thread	One architectural context for instruction execution.
WDT	Watchdog timer. A timer that detects activity and triggers an error if a specified period of time expires without the activity.
X2APICEN	x2 APIC is enabled. X2APICEN = (Core::X86::Msr::APIC_BAR[ApicEn] && Core::X86::Msr::APIC_BAR[x2ApicEn]).

2.1.3 Secure Virtual Machine Mode (SVM)

Support for SVM mode is indicated by Core::X86::Cpuid::FeatureExtIdEcx[SVM].

2.1.3.1 BIOS support for SVM Disable

The BIOS should include the following user setup options to enable and disable AMD Virtualization™ technology.

2.1.3.1.1 Enable AMD Virtualization™

- Core::X86::Msr::VM_CR[SvmeDisable] = 0.
- Core::X86::Msr::VM_CR[Lock] = 1.
- Core::X86::Msr::SvmLockKey[SvmLockKey] = 0000_0000_0000_0000h.

2.1.3.1.2 Disable AMD Virtualization™

- Core::X86::Msr::SvmLockKey[SvmLockKey] = 0000_0000_0000_0000h.
- Core::X86::Msr::VM_CR[SvmeDisable] = 1.
- Core::X86::Msr::VM_CR[Lock] = 1.

The BIOS may also include the following user setup options to disable AMD Virtualization technology.

2.1.3.1.3 Disable AMD Virtualization™, with a user supplied key

- Core::X86::Msr::VM_CR[SvmeDisable] = 1.
- Core::X86::Msr::VM_CR[Lock] = 1.
- Core::X86::Msr::SvmLockKey[SvmLockKey] programmed with value supplied by user. This value should be stored in NVRAM.

2.1.4 Memory Encryption

For details of the memory encryption, see docAPM2 section Secure Encrypted Virtualization. See docAPM2 section Enabling Memory Encryption Extensions for details about enabling memory encryption extensions.

2.1.5 Secure Nested Paging (SEV-SNP)

2.1.5.1 Overview

SEV with Secure Nested Paging (SEV-SNP) enables the isolation of virtual machines from the hypervisor. SEV-SNP enables both confidentiality and integrity protection of guest memory along with various other security features. SEV-SNP uses a structure called the Reverse Map Table (RMP) to enforce memory integrity. For more information on SEV-SNP, see docAPM2 section 15.36 Secure Nested Paging (SEV-SNP), and the SEV-SNP whitepaper on https://developer.amd.com/sev.

2.1.5.2 RMP Entry Format

Architecturally the format of RMP entries are not specified in APM. In order to assist software, the following table specifies select portions of the RMP entry format for this specific product.
The RMP entry corresponding to a page of memory at page-aligned address X can be found at:

RMP Entry Address = RMP_BASE + 0x4000 + X>>8

Where RMP_BASE is the value of Core::X86::Msr::LS_RMP_BASE[RMP_BASE]. For instance, if RMP_BASE is

0x1000_0000 then the RMP entry for address 0x2000 can be found at address 0x1000_4020.
Each RMP entry is 16B in size and is formatted as follows. Software should not rely on any field definitions not specified in this table and the format of an RMP entry may change in future processors. The definitions of the fields specified here can be found in docAPM2 section 15.36.3 Reverse Map Table.

Table 16: Reverse Map Entry

RMP Entry Bit Position	Field Name
[0]	Assigned
[1]	Page_Size
[2]	Immutable
[11:3]	Reserved
[50:12]	Guest_Physical_Address
[60:51]	ASID
[61]	VMSA
[62]	Validated
[127:63]	Reserved

2.1.5.3 CPUID Policy Enforcement

During the launch of an SEV-SNP guest, the hypervisor provides the PSP with the CPUID information it intends to report to the guest. The PSP ensures that any security critical CPUID information can be safely relied upon by the guest before allowing a guest to successfully launch (For more details, see SEV-SNP Firmware ABI, publication number 58680).

The policy depends on the EAX, ECX, and XCR0 values at the time of CPUID execution as well as the desired output values for EAX, EBX, ECX, and EDX.

The PSP policy is defined in terms of the following definitions.

- GuestVal: CPUID result value attempting to be supplied to the guest
  - This is the CPUID result value created by the hypervisor that it wants to give to the guest
- HostVal: CPUID result value as seen in host (hypervisor) mode
  - This is the actual CPUID result value specified in this PPR

For each CPUID function, the PSP performs one of the checks described in the SEV-SNP CPUID policy check table.

Table 17: SEV-SNP CPUID policy check

Check Type	Description	Check fails if
BitMask	Any bits set in the GuestVal must also be set in HostVal. This is often applied to feature fields where each bit indicates support for a feature	(GuestVal & HostVal) != GuestVal
Strict	The GuestVal must match the HostVal exactly This is often applied to either security critical fields or reserved fields to enable future expansion	GuestVal != HostVal
LessThan	The GuestVal must be less than or equal to the HostVal	GuestVal > HostVal
UnChecked	Any value is allowed	N/A

The PSP enforces the following policy:

- If the CPUID function is not in the standard range (Fn00000000 through Fn0000FFFF) or the extended range (Fn8000_0000 through Fn8000_FFFF), the function output check is UnChecked.
- If the CPUID function is in the standard or extended range and the function is not listed in SEV-SNP CPUID

Policy table, then the output check is Strict and required to be 0. Note that if the CPUID function does not depend on ECX and/or XCR0, then the PSP policy ignores those inputs, respectively.

- Otherwise, the check is defined according to the values listed in SEV-SNP CPUID Policy table.

Table 18: SEV-SNP CPUID Policy

CPUID **Function**	**Description**	**Fields**	**Check**	**Additional Notes**
Fn00000000_EAX	Processor Vendor and Largest Standard Function	All	LessThan	
Fn00000000_EBX	Processor Vendor	All	UnChecked	
Fn00000000_ECX	Processor Vendor	All	UnChecked	
Fn00000000_EDX	Processor Vendor	All	UnChecked	
Fn00000001_EAX	Family, Model, Stepping Identifiers	All	LessThan	Check applies to overall (Family, Model, Stepping) value
Fn00000001_EBX	LocalApicId, LogicalProcessorCount, CLFlush	CLFlush	Strict	
		Others	UnChecked	
Fn00000001_ECX	Feature Identifiers	OSXSAVE	UnChecked	
		Others	BitMask	
Fn00000001_EDX	Feature Identifiers	All	BitMask	
Fn00000005_EAX	Monitor/MWait	All	UnChecked	
Fn00000005_EBX	Monitor/MWait	All	UnChecked	
Fn00000005_ECX	Monitor/MWait	All	UnChecked	
Fn00000005_EDX	Monitor/MWait	All	UnChecked	
Fn00000006_EAX	Thermal and Power Management	All	BitMask	
Fn00000006_EBX	Thermal and Power Management	All	Strict	No fields
Fn00000006_ECX	Thermal and Power Management	All	BitMask	
Fn00000006_EDX	Thermal and Power Management	All	Strict	No fields
Fn00000007_EAX_x00	Structured Extended Feature Identifiers	All	LessThan	
Fn00000007_EBX_x00	Structured Extended Feature Identifiers	All	BitMask	
Fn00000007_ECX_x00	Structured Extended Feature Identifiers	OSPKE	UnChecked	
		Others	BitMask	
Fn00000007_EDX_x00	Structured Extended Feature	All	Strict	No fields

	Identifiers			
Fn0000000B_EAX_xNN	Extended Topology Enumeration	All	UnChecked	Applies to all extended functions
Fn0000000B_EBX_xNN	Extended Topology Enumeration	All	UnChecked	Applies to all extended functions
Fn0000000B_ECX_xNN	Extended Topology Enumeration	All	UnChecked	Applies to all extended functions
Fn0000000B_EDX	Extended Topology Enumeration	All	UnChecked	
Fn0000000D_EAX_x00	Processor Extended State Enumeration	[1:0]	Strict	X87 and SSE support
		Others	BitMask	
Fn0000000D_EBX_x00	Processor Extended State Enumeration	All	Strict	Strict checking takes XCR0_IN value into account
Fn0000000D_ECX_x00	Processor Extended State Enumeration	All	UnChecked	
Fn0000000D_EDX_x00	Processor Extended State Enumeration	All	Strict	
Fn0000000D_EAX_x01	Processor Extended State Enumeration	All	BitMask	
Fn0000000D_EBX_x01	Processor Extended State Enumeration	All	Strict	Strict checking takes XCR0_IN value into account
Fn0000000D_ECX_x01	Processor Extended State Enumeration	All	BitMask	
Fn0000000D_EDX_x01	Processor Extended State Enumeration	All	BitMask	
Fn0000000D_EAX_x02	Processor Extended State Enumeration	All	Strict	
Fn0000000D_EBX_x02	Processor Extended State Enumeration	All	Strict	
Fn0000000D_ECX_x02	Processor Extended State Enumeration	All	Strict	
Fn0000000D_EDX_x02	Processor Extended State Enumeration	All	Strict	No fields
Fn0000000D_EAX_x09	Processor Extended State Enumeration	All	Strict	
Fn0000000D_EAX_x09	Processor Extended State Enumeration	All	Strict	
Fn0000000D_EAX_x09	Processor Extended State Enumeration	All	Strict	
Fn0000000D_EAX_x09	Processor Extended State Enumeration	All	Strict	No fields
Fn0000000D_EAX_x0B	Processor Extended State Enumeration	All	Strict	
Fn0000000D_EBX_x0B	Processor Extended State Enumeration	All	Strict	
Fn0000000D_ECX_x0B	Processor Extended State	All	Strict	

	Enumeration			
Fn0000000D_EDX_x0B	Processor Extended State Enumeration	All	Strict	No fields
Fn0000000D_EAX_x0C	Processor Extended State Enumeration	All	Strict	
Fn0000000D_EBX_x0C	Processor Extended State Enumeration	All	Strict	
Fn0000000D_ECX_x0C	Processor Extended State Enumeration	All	Strict	
Fn0000000D_EDX_x0C	Processor Extended State Enumeration	All	Strict	No fields
Fn0000000F_EAX_x00	Resource Director Technology Monitor Capability	All	Strict	No fields
Fn0000000F_EBX_x00	Resource Director Technology Monitor Capability	All	UnChecked	
Fn0000000F_ECX_x00	Resource Director Technology Monitor Capability	All	Strict	No fields
Fn0000000F_EDX_x00	Resource Director Technology Monitor Capability	All	UnChecked	
Fn0000000F_EAX_x01	Resource Director Technology Monitor Capability	All	UnChecked	
Fn0000000F_EBX_x01	Resource Director Technology Monitor Capability	All	UnChecked	
Fn0000000F_ECX_x01	Resource Director Technology Monitor Capability	All	UnChecked	
Fn0000000F_EDX_x01	Resource Director Technology Monitor Capability	All	UnChecked	
Fn00000010_EAX_x00	Resource Director Technology Allocation Enumeration	All	Strict	No fields
Fn00000010_EBX_x00	Resource Director Technology Allocation Enumeration	All	UnChecked	
Fn00000010_ECX_x00	Resource Director Technology Allocation Enumeration	All	Strict	No fields
Fn00000010_EDX_x00	Resource Director Technology Allocation Enumeration	All	Strict	No fields
Fn00000010_EAX_x01	Resource Director Technology Allocation Enumeration	All	UnChecked	
Fn00000010_EBX_x01	Resource Director Technology Allocation Enumeration	All	UnChecked	
Fn00000010_ECX_x01	Resource Director Technology Allocation Enumeration	All	UnChecked	
Fn00000010_EDX_x01	Resource Director Technology Allocation Enumeration	All	UnChecked	
Fn80000000_EAX	Largest Extended Function Number	All	LessThan	
Fn80000000_EBX	Processor Vendor	All	UnChecked	

Fn80000000_ECX	Processor Vendor	All	UnChecked	
Fn80000000_EDX	Processor Vendor	All	UnChecked	
Fn80000001_EAX	Family, Model, Stepping Identifiers	All	LessThan	Check applies to overall (Family, Model, Stepping) value
Fn80000001_EBX	BrandId Identifier	All	UnChecked	
Fn80000001_ECX	Feature Identifiers	All	BitMask	
Fn80000001_EDX	Feature Identifiers	All	BitMask	
Fn80000002_EAX	Processor Name String Identifier	All	UnChecked	
Fn80000002_EBX	Processor Name String Identifier	All	UnChecked	
Fn80000002_ECX	Processor Name String Identifier	All	UnChecked	
Fn80000002_EDX	Processor Name String Identifier	All	UnChecked	
Fn80000003_EAX	Processor Name String Identifier	All	UnChecked	
Fn80000003_EBX	Processor Name String Identifier	All	UnChecked	
Fn80000003_ECX	Processor Name String Identifier	All	UnChecked	
Fn80000003_EDX	Processor Name String Identifier	All	UnChecked	
Fn80000004_EAX	Processor Name String Identifier	All	UnChecked	
Fn80000004_EBX	Processor Name String Identifier	All	UnChecked	
Fn80000004_ECX	Processor Name String Identifier	All	UnChecked	
Fn80000004_EDX	Processor Name String Identifier	All	UnChecked	
Fn80000005_EAX	L1 TLB 2M/4M Identifiers	All	UnChecked	
Fn80000005_EBX	L1 TLB 4K Identifiers	All	UnChecked	
Fn80000005_ECX	L1 Data Cache Identifiers	All	UnChecked	
Fn80000005_EDX	L1 Instruction Cache Identifiers	All	UnChecked	
Fn80000006_EAX	L2 TLB 2M/4M Identifiers	All	UnChecked	

Fn80000006_EBX	L2 TLB 4K Identifiers	All	UnChecked	
Fn80000006_ECX	L2 Cache Identifiers	All	UnChecked	
Fn80000006_EDX	L2 Cache Identifiers	All	UnChecked	
Fn80000007_EAX	Reserved	All	Strict	No fields
Fn80000007_EBX	RAS Capabilities	All	BitMask	
Fn80000007_ECX	Advanced Power Management Information	All	UnChecked	
Fn80000007_EDX	Advanced Power Management Information	All	BitMask	
Fn80000008_EAX	Long Mode Address Size Identifiers	All	LessThan	Check applies to each field
Fn80000008_EBX	Extended Feature Extensions ID EBX	All	BitMask	
Fn80000008_ECX	Size Identifiers	All	UnChecked	
Fn80000008_EDX	Feature Extended Size Edx	All	LessThan	Check applies to each field
Fn8000000A_EAX	SVM Revision and Feature Identification	All	UnChecked	
Fn8000000A_EBX	SVM Revision and Feature Identification	All	UnChecked	
Fn8000000A_ECX	Reserved	All	Strict	No fields
Fn8000000A_EDX	SVM Revision and Feature Identification	All	BitMask	
Fn80000019_EAX	L1 TLB 1G Identifiers	All	UnChecked	
Fn80000019_EBX	L2 TLB 1G Identifiers	All	UnChecked	
Fn80000019_ECX	Reserved	All	Strict	No fields
Fn80000019_EDX	Reserved	All	Strict	No fields
Fn8000001A_EAX	Performance Optimization Identifiers	All	UnChecked	
Fn8000001A_EBX	Reserved	All	Strict	No fields
Fn8000001A_ECX	Reserved	All	Strict	No fields
Fn8000001A_EDX	Reserved	All	Strict	No fields
Fn8000001B_EAX	Instruction Based Sampling Identifiers	All	BitMask	

Fn8000001B_EBX	Reserved	All	Strict	No fields
Fn8000001B_ECX	Reserved	All	Strict	No fields
Fn8000001B_EDX	Reserved	All	Strict	No fields
Fn8000001D_EAX_x00	Cache Properties (DC)	All	UnChecked	
Fn8000001D_EBX_x00	Cache Properties (DC)	All	UnChecked	
Fn8000001D_ECX_x00	Cache Properties (DC)	All	UnChecked	
Fn8000001D_EDX_x00	Cache Properties (DC)	All	UnChecked	
Fn8000001D_EAX_x01	Cache Properties (IC)	All	UnChecked	
Fn8000001D_EBX_x01	Cache Properties (IC)	All	UnChecked	
Fn8000001D_ECX_x01	Cache Properties (IC)	All	UnChecked	
Fn8000001D_EDX_x01	Cache Properties (IC)	All	UnChecked	
Fn8000001D_EAX_x02	Cache Properties (L2)	All	UnChecked	
Fn8000001D_EBX_x02	Cache Properties (L2)	All	UnChecked	
Fn8000001D_ECX_x02	Cache Properties (L2)	All	UnChecked	
Fn8000001D_EDX_x02	Cache Properties (L2)	All	UnChecked	
Fn8000001D_EAX_x03	Cache Properties (L3)	All	UnChecked	
Fn8000001D_EBX_x03	Cache Properties (L3)	All	UnChecked	
Fn8000001D_ECX_x03	Cache Properties (L3)	All	UnChecked	
Fn8000001D_EDX_x03	Cache Properties (L3)	All	UnChecked	
Fn8000001D_EAX_x04	Cache Properties Null	All	UnChecked	
Fn8000001D_EBX_x04	Cache Properties Null	All	Strict	No fields
Fn8000001D_ECX_x04	Cache Properties Null	All	UnChecked	
Fn8000001D_EDX_x04	Cache Properties Null	All	Strict	No fields
Fn8000001E_EAX	Extended APIC ID	All	UnChecked	
Fn8000001E_EBX	Core Identifiers	All	UnChecked	
Fn8000001E_ECX	Node Identifiers	All	UnChecked	
Fn8000001E_EDX	Reserved	All	UnChecked	
Fn8000001F_EAX	AMD Secure Encryption EAX	All	BitMask	

Fn8000001F_EBX	AMD Secure Encryption EBX	Bits[11:6]	UnChecked	Physical address width reduction
		Others	Strict	
Fn8000001F_ECX	AMD Secure Encryption ECX	All	UnChecked	
Fn8000001F_EDX	Minimum ASID	All	UnChecked	
Fn80000020_EAX_x00	Platform QoS Enforcement for Memory Bandwidth	All	Strict	No fields
Fn80000020_EBX_x00	Platform QoS Enforcement for Memory Bandwidth	All	UnChecked	
Fn80000020_ECX_x00	Platform QoS Enforcement for Memory Bandwidth	All	Strict	No fields
Fn80000020_EDX_x00	Platform QoS Enforcement for Memory Bandwidth	All	Strict	No fields
Fn80000020_EAX_x01	Platform QoS Enforcement for Memory Bandwidth	All	UnChecked	
Fn80000020_EBX_x01	Platform QoS Enforcement for Memory Bandwidth	All	Strict	No fields
Fn80000020_ECX_x01	Platform QoS Enforcement for Memory Bandwidth	All	Strict	No fields
Fn80000020_EDX_x01	Platform QoS Enforcement for Memory Bandwidth	All	UnChecked	
Fn80000021_EAX	Extended Feature 2 EAX	All	BitMask	
Fn80000021_EBX	Reserved	All	Strict	No fields
Fn80000021_ECX	Reserved	All	Strict	No fields
Fn80000021_EDX	Reserved	All	Strict	No fields

2.1.6 Effective Frequency

The effective frequency interface allows software to discern the average, or effective, frequency of a given core over a configurable window of time. This provides software a measure of actual performance rather than forcing software to assume the current frequency of the core is the frequency of the last P-state requested. Core::X86::Msr::MPERF is incremented by hardware at the P0 frequency while the core is in C0. Core::X86::Msr::APERF increments in proportion to the actual number of core clocks cycles while the core is in C0.

The following procedure calculates effective frequency using Core::X86::Msr::MPERF and Core::X86::Msr::APERF:
1. At some point in time, write 0 to both MSRs.
2. At some later point in time, read both MSRs.
3. Effective frequency = (value read from Core::X86::Msr::APERF / value read from Core::X86::Msr::MPERF) * P0 frequency.

Additional notes:
- The amount of time that elapses between steps 1 and 2 is determined by software.
- It is software's responsibility to disable interrupts or any other events that may occur in between the Write of Core::X86::Msr::MPERF and the Write of Core::X86::Msr::APERF in step 1 or between the Read of

Core::X86::Msr::MPERF and the Read of Core::X86::Msr::APERF in step 2.

- The behavior of Core::X86::Msr::MPERF and Core::X86::Msr::APERF may be modified by Core::X86::Msr::HWCR[EffFreqCntMwait].
- The effective frequency interface provides +/- 50MHz accuracy if the following constraints are met:
    - Effective frequency is read at most one time per millisecond.
    - When reading or writing Core::X86::Msr::MPERF and Core::X86::Msr::APERF software executes only MOV instructions, and no more than 3 MOV instructions, between the two RDMSR or WRMSR instructions.
    - Core::X86::Msr::MPERF and Core::X86::Msr::APERF are invalid if an overflow occurs.

2.1.7 Address Space

2.1.7.1 Virtual Address Space

The processor supports 48-bit address bits of virtual memory space (256 TB) as indicated by Core::X86::Cpuid::LongModeInfo.

2.1.7.2 Physical Address Space

The processor supports a 48-bit physical address space. See Core::X86::Cpuid::LongModeInfo.
The processor master aborts the following upper-address transactions (to address PhysAddr):

- Link or core requests with non-zero PhysAddr[63:48].

2.1.7.3 System Address Map

The processor defines a Reserved memory address region starting at FFFD_0000_0000h and extending up to FFFF_FFFF_FFFFh. System software must not map memory into this region. Downstream host accesses to the Reserved address region results in a page fault. Upstream system device accesses to the reserved address region results in an undefined operation.

2.1.7.3.1 Memory Access to the Physical Address Space

All memory accesses to the physical address space from a core are sent to its associated Data Fabric (DF). All memory accesses from a link are routed through the DF. An IO link access to physical address space indicates to the DF the cache attribute (Coherent or Non-coherent, based on bit[0] of the Sized Read and Write commands).

A core access to physical address space has two important attributes that must be determined before issuing the access to the NB: the memory type (e.g., WB, WC, UC; as described in the MTRRs) and the access destination (DRAM or MMIO).

If the memory map maps a region as DRAM that is not populated with real storage behind it, then that area of DRAM must be mapped as UC memtype.

This mechanism is managed by the BIOS and does not require any setup or changes by system software.

2.1.7.3.1.1 Determining Memory Type

The memory type for a core access is determined by the highest priority of the following ranges that the access falls in: 1=Lowest priority.

1. The memory type as determined by architectural mechanisms.
    - See the docAPM2 chapter titled "Memory System", sections "Memory-Type Range Registers" and "Page-

    Attribute Table Mechanism".
    - See the docAPM2 chapter titled "Nested Paging", section "Combining Memory Types, MTRRs".
    - See Core::X86::Msr::MTRRdefType, Core::X86::Msr::MtrrVarBase, Core::X86::Msr::MtrrVarMask, Core::X86::Msr::MtrrFix_64K and Core::X86::Msr::MtrrFix_16K_0 through Core::X86::Msr::MtrrFix_4K_7.
2. TSeg & ASeg SMM mechanism. (see Core::X86::Msr::SMMAddr and Core::X86::Msr::SMMMask).
3. CR0[CD]: If (CR0[CD] == 1) then MemType = CD.
4. MMIO configuration space, APIC space.
    - MMIO APIC space and MMIO config space must not overlap.
    - MemType = UC.
5. If ("In SMM Mode"&& ~((Core::X86::Msr::SMMMask[AValid] && "The address falls within the ASeg region") || (Core::X86::Msr::SMMMask[TValid] && "The address falls within the TSeg region"))) then MemType = CD.

2.1.8 Configuration Space

PCI-defined configuration space was originally defined to allow up to 256 bytes of register space for each function of each device; these first 256 bytes are called base configuration space (BCS). It was expanded to support up to 4096 bytes per function; bytes 256 through 4095 are called extended configuration space (ECS).

The processor includes configuration space registers located in both BCS and ECS. Processor configuration space is accessed through bus 0, devices 18h to 1Fh, where device 18h corresponds to node 0 and device 1Fh corresponds to node 7. See 2.1.8.3 [Processor Configuration Space].

Configuration space is accessed by the processor through two methods as follows:

- IO-space configuration: IO instructions to addresses CF8h and CFCh.
    - Enabled through IO::IoCfgAddr[ConfigEn], which allows access to BCS.
    - Use of IO-space configuration can be programmed to generate GP faults through Core::X86::Msr::HWCR[IoCfgGpFault].
    - SMI trapping for these accesses is specified by Core::X86::Msr::SMI_ON_IO_TRAP_CTL_STS and Core::X86::Msr::SMI_ON_IO_TRAP.
- MMIO configuration: configuration space is a region of memory space.
    - The base address and size of this range is specified by Core::X86::Msr::MmioCfgBaseAddr. The size is controlled by the number of configuration-space bus numbers supported by the system. Accesses to this range are converted configuration space as follows:
    - Address[31:0] = {0h, bus[7:0], device[4:0], function[2:0], offset[11:0]}.

The BIOS may use either configuration space access mechanism during boot. Before booting the OS, BIOS must disable IO access to ECS, enable MMIO configuration and build an ACPI defined MCFG table. BIOS ACPI code must use MMIO to access configuration space.

2.1.8.1 MMIO Configuration Coding Requirements

MMIO configuration space accesses must use the uncacheable (UC) memory type.
Instructions used to read MMIO configuration space are required to take the following form:

```
mov eax/ax/al, any_address_mode;
```

Instructions used to write MMIO configuration space are required to take the following form:

```
mov any_address_mode, eax/ax/al;
```

No other source/target registers may be used other than eax/ax/al.

In addition, all such accesses are required not to cross any naturally aligned DW boundary. Access to MMIO configuration space registers that do not meet these requirements result in undefined behavior.

2.1.8.2 MMIO Configuration Ordering

Since MMIO configuration cycles are not serializing in the way that IO configuration cycles are, their ordering rules relative to posted may result in unexpected behavior.

Therefore, processor MMIO configuration space is designed to match the following ordering relationship that exists naturally with IO-space configuration: if a core generates a configuration cycle followed by a posted write cycle, then the posted write is held in the processor until the configuration cycle completes. As a result, any unexpected behavior that might have resulted if the posted-write cycle were to pass MMIO configuration cycle is avoided.

2.1.8.3 Processor Configuration Space

Accesses to unimplemented registers of implemented functions are ignored: Writes dropped; Reads return 0. Accesses to unimplemented functions also ignored: Writes are dropped; however, Reads return all F's. The processor does not log any master abort events for accesses to unimplemented registers or functions.

Accesses to device numbers of devices not implemented in the processor are routed based on the configuration map registers. If such requests are master aborted, then the processor can log the event.

2.1.9 PCI Configuration Legacy Access

IOx0CF8 [IO-Space Configuration Address] (IO::IoCfgAddr)

Read-write. Reset: 0000_0000h.

IO::IoCfgAddr, and IO::IoCfgData are used to access system configuration space, as defined by the PCI specification. IO::IoCfgAddr provides the address register and IO::IoCfgData provides the data port. Software sets up the configuration address by writing to IO::IoCfgAddr. Then, when an access is made to IO::IoCfgData, the processor generates the corresponding configuration access to the address specified in IO::IoCfgAddr. See 2.1.8 [Configuration Space].

IO::IoCfgAddr may only be accessed through aligned, DW IO Reads and Writes; otherwise, the accesses are passed to the appropriate IO link. Accesses to IO::IoCfgAddr and IO::IoCfgData received from an IO link are treated as all other IO transactions received from an IO link. IO::IoCfgAddr and IO::IoCfgData in the processor are not accessible from an IO link.

_aliasIO; IOx0CF8; IO=0000_0000h

Bits	Description
31	**ConfigEn**: **configuration space enable**. Read-write. Reset: 0. 0=IO Read and Write accesses are passed to the appropriate IO link and no configuration access is generated. 1=IO Read and Write accesses to IO::IoCfgData are translated into configuration cycles at the configuration address specified by this register.
30:28	Reserved.
27:24	**ExtRegNo**: **extended register number**. Read-write. Reset: 0h. ExtRegNo provides bits[11:8] and RegNo provides bits[7:2] of the byte address of the configuration register.
23:16	**BusNo**: **bus number**. Read-write. Reset: 00h. Specifies the bus number of the configuration cycle.
15:11	**Device**: **device number**. Read-write. Reset: 00h. Specifies the device number of the configuration cycle.
10:8	**Function**. Read-write. Reset: 0h. Specifies the function number of the configuration cycle.
7:2	**RegNo**: **register address**. Read-write. Reset: 00h. See IO::IoCfgAddr[ExtRegNo].
1:0	Reserved.

IOx0CFC [IO-Space Configuration Data Port] (IO::IoCfgData)

Read-write. Reset: 0000_0000h.	
_aliasIO; IOx0CFC; IO=0000_0000h	
Bits	**Description**
31:0	**Data**. Read-write. Reset: 0000_0000h. See IO::IoCfgAddr.

2.1.10 System Software Interaction With SMT Enabled

If Core::X86::Cpuid::CoreId[ThreadsPerCore] > 0, then SMT is enabled in all cores in the system. When SMT is enabled, the resources of each core are dynamically balanced among the hardware threads executing on that core. The number of hardware threads (hereafter "threads") supported by a single core when SMT is enabled is reported in Core::X86::Cpuid::CoreId[ThreadsPerCore]. System software that is SMT-aware may take advantage of the knowledge that core resources are being shared among multiple threads when scheduling tasks to be run by each thread on each core. System software that is not SMT-aware sees each thread as an independent core.

2.1.11 Register Sharing

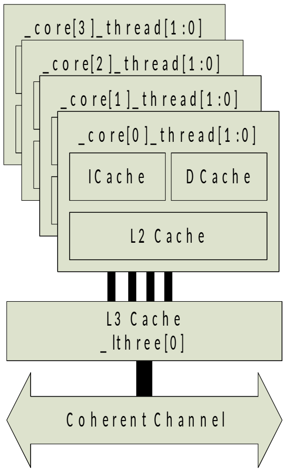
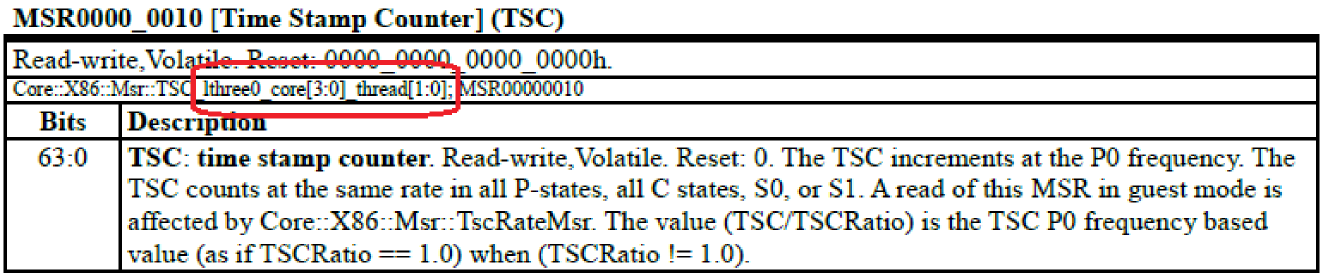


Figure 26: Register Sharing Domains



MSR0000_0010 [Time Stamp Counter] (TSC)

Read-write,Volatile. Reset: 0000_0000_0000_0000h.	
Core::X86::Msr::TSC_lthree0_core[3:0]_thread[1:0]; MSR00000010	
Bits	**Description**
63:0	**TSC**: **time stamp counter**. Read-write,Volatile. Reset: 0. The TSC increments at the P0 frequency. The TSC counts at the same rate in all P-states, all C states, S0, or S1. A read of this MSR in guest mode is affected by Core::X86::Msr::TscRateMsr. The value (TSC/TSCRatio) is the TSC P0 frequency based value (as if TSCRatio == 1.0) when (TSCRatio != 1.0).

Figure 27: Instance Parameters

Instances of core registers are designated as ccd[n:0]_lthree[n:0]_core[n:0]_thread[1:0]. Core registers may be shared at various levels of hierarchy as one register instance per node, per L3 complex, per core or per thread. The absence of the instance parameter _thread[1:0] signifies that there is not a specific instance of said register per thread and thus the register is shared between thread[1] and thread[0]. Similarly, the absence of the instance parameter _core[n:0] signifies that there is not a specific instance of said register per core and thus the register is shared by all cores in that L3 complex, and so on. The absence of instance parameters indicate there is one shared register at the node level. Software must coordinate writing to shared registers with other threads in the same sharing hierarchy level.

2.1.12 Timers

Each core includes the following timers. These timers do not vary in frequency regardless of the current P-state or C-state.

- Core::X86::Msr::TSC; the TSC increments at the rate specified by the P0 Pstate.
- The APIC timer (Core::X86::Apic::TimerInitialCount and Core::X86::Apic::TimerCurrentCount), which increments at the rate of 2xCLKIN; the APIC timer may increment in units of between 1 and 8.

2.1.13 Branch Sampling

Branch sampling is a mechanism to record taken branches after a performance monitor overflow interrupt occurs. This allows software to periodically sample a fixed number of branches after the PMC overflow is signaled. The presence of the branch sampling feature version 1 is indicated by Core::X86::Cpuid::FeatureExtIdEbx[BranchSample] == 1.

2.1.13.1 Branch Sampling Registers and Usage

A bank of MSRs exist starting at MSR C001_0300h. These MSRs are paired elements of a Core::X86::Msr::SAMP_BR_FROM register N and Core::X86::Msr::SAMP_BR_TO register N. The first sample is intended to be in register 0 and the last sample in register N-1. N can vary on future processors but for this Family/Model stepping there are MSRs for 16 branches.

Table 19: MSRs For 16 Branches

MSR **Number**	MSR **Name**
MSRC001_0300	Core::X86::Msr::SAMP_BR_FROM_n0
MSRC001_0301	Core::X86::Msr::SAMP_BR_TO_n0
MSRC001_0302	Core::X86::Msr::SAMP_BR_FROM_n1
MSRC001_0303	Core::X86::Msr::SAMP_BR_TO_n1
...	...
MSRC001_031E	Core::X86::Msr::SAMP_BR_FROM_n15
MSRC001_031F	Core::X86::Msr::SAMP_BR_TO_n15

The SAMP_BR_FROM_N set of registers hold the segment offsets (RIP) of the branch instructions. The SAMP_BR_TO_N set of registers hold the segment offsets (RIP) of the targets of the branch instructions.
The sampling mode is controlled through the following bits in Core::X86::Msr::DbgExtnCfg:

- BRSMEN is the Branch Sampling Enable which turns the recording mode on for branches.
- Reserved bits DbgExtnCfg[4:3] should be set to 1 when setting BRSMEN.
- VB indicates that there are valid branches recorded in the MSRs.
- MSROFF indicates the first sampling pair of MSRs that the processor should fill with branch records. It will proceed from that specified pair to the SAMP_BR_FROM/TO[N-1] and then will stop recording. When it stops recording by reaching the last entry the MSROFF field will be written to 0. If software disables branch sampling while recording is active, MSROFF contains the next SAMP_BR_FROM/TO MSR that would have been updated. Therefore SAMP_BR_FROM/TO_0 to MSROFF-1 should contain valid recorded branches if the VB == 1.
- PMC is the core PMC number from 0-5 that is programmed to overflow and start the branch recording.

The branch sampling mode relies on the Performance Monitor Counting Overflow Interrupt mode to initiate the start of the sampling. Any event can be used by software as the sampling event to generate the overflow. The PMC number enabled to overflow and start branch sampling should be written into the PMC field. This feature requires that the Perfmon LVT be enabled to deliver an NMI on the performance monitor overflow. The overflow will start the process of the branches being recorded and the processor will set Core::X86::Msr::DBG_CTL_MSR[LBR] to 1 to enable the legacy LBR mechanism, regardless of its current state. When the MSRs are full the hardware will clear Core::X86::Msr::DbgExtnCfg[BRSMEN] in LBR_CFG, set Core::X86::Msr::DbgExtnCfg[VB] = 1, write Core::X86::Msr::DbgExtnCfg[MSROFF] to 0, clear Core::X86::Msr::DBG_CTL_MSR[LBR] to disable the legacy LBR mechanism and attempt to enter the NMI handler. If the processor is in a state where NMI's are masked when the last branch sample is recorded, the processor will wait until the next instruction that unmasks the NMI before entering the NMI handler. Since the NMI handler could be shared by many different sources, the NMI handler should determine if the PMC dedicated for branch sampling has overflowed. If it has, it should read LBR_CFG and check Core::X86::Msr::DbgExtnCfg[VB]. If VB is set then branches have been recorded in the MSRs. If the MSROFF field is zero, all MSRs are full of branch records and the handler should collect the branches. If the MSROFF field is non-zero, the handler should collect the partial sample from 0 to MSROFF-1. If the handler wishes to take another sample it should reset the PMC to its pre-overflow value, and write the Core::X86::Msr::DbgExtnCfg register with MSROFF = 0, VB = 0, and BRSMEN = 1. Since branch sampling requires use of the LBR feature in Debug Control MSR, software should write BRSMEN = 0 before reading or writing Debug Control MSR. Since the hardware does enable the legacy LBR mechanism when the feature is recording branches, the legacy MSRs: Core::X86::Msr::BR_FROM, Core::X86::Msr::BR_TO, Core::X86::Msr::LastExcpFromIp and Core::X86::Msr::LastExcpToIp are updated as a side effect of enabling branch sampling.

Software can use the OS and USR bits in the performance counter selected to overflow to get more samples in the desired mode.

2.1.13.1.1 Special Handling for non C0 Processor States

If software requests a non C0 Processor State when branch sampling is actively recording branches in the MSRs, other incoming NMI's may be delayed. Software should avoid this problem by disabling branch sampling before requesting a non C0 State and re-enabling branch sampling after returning to the C0 state. If the processor was in the middle of recording when BRSMEN was written to zero, the processor will update the MSROFF to the next MSR location to be recorded, set VB = 1 if branches have been recorded, set the Debug Control MSR bit[0] to 0, and then attempt to enter the NMI handler.

2.1.13.1.2 Virtualization and SMM

The architecture does not maintain Guest and Host copies of these MSRs so the hypervisor is required to use MSR intercepts to allow a Guest to use this mechanism. If a VMRUN targets a secure encrypted guest and branch sampling is enabled, it will be disabled and the MSROFF field updated to the next SAMP_BR_FROM to be written. The hypervisor will have to re-enable after the VMEXIT back to the hypervisor.

Branches in SMM and transitions to and from SMM mode are never recorded. In version 1 of branch sampling, if an SMI is received in the active period where branches are being recorded in the MSRs, the processor will end branch recording by setting BRSMEN = 0, set the VB bit to 1, update the MSROFF field to the next MSR that would be written, set Debug Control MSR bit[0] to 0 and then follow the normal interrupt handling priorities.

2.1.14 Interrupts

2.1.14.1 System Management Mode (SMM)

System management mode (SMM) is typically used for system control activities such as power management. These activities are typically transparent to the operating system.

2.1.14.1.1 SMM Overview

SMM is entered by a core on the next instruction boundary after a system management interrupt (SMI) is received and recognized. A core may be programmed to broadcast a special cycle to the system, indicating that it is entering SMM mode. The core then saves its state into the SMM memory state save area and jumps to the SMI service routine (or SMI handler). The pointer to the SMI handler is specified by MSRs. The code and data for the SMI handler are stored in the SMM memory area, which may be isolated from the main memory accesses.

The core returns from SMM by executing the RSM instruction from the SMI handler. The core restores its state from the SMM state save area and resumes execution of the instruction following the point where it entered SMM. The core may be programmed to broadcast a special bus cycle to the system, indicating that it is exiting SMM mode.

2.1.14.1.2 Mode and Default Register Values

The software environment after entering SMM has the following characteristics:

- Addressing and operation is in Real mode.
    - A far jump, call or return in the SMI handler can only address the lower 1M of memory, unless the SMI handler first switches to protected mode.
    - If (Core::X86::Msr::SMM_BASE[SmmBase] >= 0010_0000h) then:
        - The value of the CS selector is undefined upon SMM entry.
        - The undefined CS selector value should not be used as the target of a far jump, call, or return.
- 4-Gbyte segment limits.
- Default 16-bit operand, address, and stack sizes (instruction prefixes can override these defaults).
- Control transfers that do not override the default operand size truncate the EIP to 16 bits.
- Far jumps or calls cannot transfer control to a segment with a base address requiring more than 20 bits, as in Real mode segment-base addressing, unless a change is made into protected mode.
- Interrupt vectors use the Real mode interrupt vector table.
- The IF flag in EFLAGS is cleared (INTR is not recognized).
- The TF flag in EFLAGS is cleared.
- The NMI and INIT interrupts are masked.
- Debug register DR7 is cleared (debug traps are disabled).

The SMM base address is specified by Core::X86::Msr::SMM_BASE[SmmBase]. Important offsets to the base address pointer are:

- Core::X86::Msr::SMM_BASE[SmmBase] + 8000h: SMI handler entry point.
- Core::X86::Msr::SMM_BASE[SmmBase] + FE00h - FFFFh: SMM state save area.

2.1.14.1.3 SMI Sources And Delivery

The processor accepts SMIs as link-defined interrupt messages only. The core/node destination of these SMIs is a function of the destination field of these messages. However, the expectation is that all such SMI messages are specified to be delivered globally (to all cores of all nodes).

There are also several local events that can trigger SMIs. However, these local events do not generate SMIs directly. Each of them triggers a programmable IO cycle that is expected to target the SMI command port in the IO hub and trigger a global SMI interrupt message back to the coherent fabric.

Local sources of SMI events that generate the IO cycle specified in Core::X86::Msr::SmiTrigIoCycle are:

- In the core, as specified by:

  - Core::X86::Msr::McExcepRedir.
  - Core::X86::Msr::SMI_ON_IO_TRAP.
- All local APIC LVT registers programmed to generate SMIs.

The status for these is stored in Core::X86::Smm::LocalSmiStatus.

2.1.14.1.4 SMM Initial State

After storing the save state, execution starts at Core::X86::Msr::SMM_BASE[SmmBase] + 08000h. The SMM initial state is specified in the following table.

Table 20: SMM Initial State

Register	SMM **Initial State**
CS	SmmBase[19:4]
DS	0000h
ES	0000h
FS	0000h
GS	0000h
SS	0000h
General-Purpose Registers	Unmodified.
EFLAGS	0000_0002h
RIP	0000_0000_0000_8000h
CR0	Bits[0,2,3,31] cleared (PE, EM, TS, and PG); remainder is unmodified.
CR4	0000_0000_0000_0000h
GDTR	Unmodified.
LDTR	Unmodified.
IDTR	Unmodified.
TR	Unmodified.
DR6	Unmodified.
DR7	0000_0000_0000_0400h
EFER	All bits are cleared except bit[12] (SVME) which is unmodified.

2.1.14.1.5 SMM Save State

In the following table, the offset field provides the offset from the SMM base address specified by Core::X86::Msr::SMM_BASE[SmmBase].

Table 21: SMM Save State

Offset	**Size**	**Contents**		**Access**
FE00h	Word	ES	Selector	Read-only
FE02h	6 Bytes		Reserved	
FE08h	Quadword		Descriptor in memory format	
FE10h	Word	CS	Selector	Read-only
FE12h	6 Bytes		Reserved	
FE18h	Quadword		Descriptor in memory format	
FE20h	Word	SS	Selector	Read-only
FE22h	6 Bytes		Reserved	

FE28h	Quadword		Descriptor in memory format	
FE30h	Word	DS	Selector	Read-only
FE32h	6 Bytes		Reserved	
FE38h	Quadword		Descriptor in memory form	
FE40h	Word	FS	Selector	Read-only
FE42h	2 Bytes		Reserved	
FE44h	Doubleword		FS Base {16'b[47], 47:32}(note 1)	
FE48h	Quadword		Descriptor in memory format	
FE50h	Word	GS	Selector	Read-only
FE52h	2 Bytes		Reserved	
FE54h	Doubleword		GS Base {16'b[47], 47:32}(note 1)	
FE58h	Quadword		Descriptor in memory format	
FE60h	4 Bytes	GDTR	Reserved	Read-only
FE64h	Word		Limit	
FE66h	2 Bytes		Reserved	
FE68h	Quadword		Descriptor in memory format	
FE70h	Word	LDTR	Selector	Read-only
FE72h	Word		Attributes	
FE74h	Doubleword		Limit	
FE78h	Quadword		Base	
FE80h	4 Bytes	IDTR	Reserved	Read-only
FE84h	Word		Limit	
FE86h	2 Bytes		Reserved	
FE88h	Quadword		Base	
FE90h	Word	TR	Selector	Read-only
FE92h	Word		Attributes	
FE94h	Doubleword		Limit	
FE98h	Quadword		Base	
FEA0h	Quadword	IO_RESTART_RIP		
FEA8h	Quadword	IO_RESTART_RCX		
FEB0h	Quadword	IO_RESTART_RSI		
FEB8h	Quadword	IO_RESTART_RDI		
FEC0h	Doubleword	Core::X86::Smm::TrapOffset [SMM IO Trap Offset]		Read-only
FEC4	Doubleword	Core::X86::Smm::LocalSmiStatus		Read-only
FEC8h	Byte	Core::X86::Smm::IoRestart		Read-write
FEC9h	Byte	Core::X86::Smm::AutoHalt		Read-write
FECAh	Byte	Core::X86::Smm::NmiMask		Read-write
FECBh	5 Bytes	Reserved		
FED0h	Quadword	EFER		Read-only
FED8h	Quadword	Core::X86::Smm::SvmState		Read-only
FEE0h	Quadword	Guest VMCB physical address		Read-only
FEE8h	Quadword	SVM Virtual Interrupt Control		Read-only
FEF0h	16 Bytes	Reserved		

FEFCh	Doubleword	Core::X86::Smm::SmmRevID	Read-only
FF00h	Doubleword	Core::X86::Smm::SmmBase	Read-write
FF04h	28 Bytes	Reserved	
FF20h	Quadword	Guest PAT	Read-only
FF28h	Quadword	Host EFER (note 2)	
FF30h	Quadword	Host CR4 (note 2)	
FF38h	Quadword	Nested CR3 (note 2)	
FF40h	Quadword	Host CR0 (note 2)	
FF48h	Quadword	CR4	
FF50h	Quadword	CR3	
FF58h	Quadword	CR0	
FF60h	Quadword	DR7	Read-only
FF68h	Quadword	DR6	
FF70h	Quadword	RFLAGS	Read-write
FF78h	Quadword	RIP	Read-write
FF80h	Quadword	R15	
FF88h	Quadword	R14	
FF90h	Quadword	R13	
FF98h	Quadword	R12	
FFA0h	Quadword	R11	
FFA8h	Quadword	R10	
FFB0h	Quadword	R9	
FFB8h	Quadword	R8	
FFC0h	Quadword	RDI	Read-write
FFC8h	Quadword	RSI	
FFD0h	Quadword	RBP	
FFD8h	Quadword	RSP	
FFE0h	Quadword	RBX	
FFE8h	Quadword	RDX	
FFF0h	Quadword	RCX	
FFF8h	Quadword	RAX	

Notes:
1. This notation specifies that bit[47] is replicated in each of the 16 MSBs of the DW (sometimes called sign extended). The 16 LSBs contain bits[47:32].
2. Only used for an SMI in guest mode with nested paging enabled.

The SMI save state includes most of the integer execution unit. Not included in the save state are: the floating point state, MSRs, and CR2. In order to be used by the SMI handler, these must be saved and restored. The save state is the same, regardless of the operating mode (32-bit or 64-bit).

2.1.14.1.6 System Management State

The following are offsets in the SMM save state area.

SMMxFEC0 [SMM IO Trap Offset] (Core::X86::Smm::TrapOffset)
Read-only,Volatile. Reset: 0000_0000h.
If the assertion of SMI is recognized on the boundary of an IO instruction, Core::X86::Smm::TrapOffset contains information about that IO instruction. For example, if an IO access targets an unavailable device, the system can assert

SMI and trap the IO instruction. Core::X86::Smm::TrapOffset then provides the SMI handler with information about the IO instruction that caused the trap. After the SMI handler takes the appropriate action, it can reconstruct and then re-execute the IO instruction from SMM. Or, more likely, it can use Core::X86::Smm::IoRestart to cause the core to re-execute the IO instruction immediately after resuming from SMM.

Bits	Description
31:16	**Port**: **trapped IO port address**. Read-only,Volatile. Reset: 0000h. This provides the address of the IO instruction.
15:12	**BPR**: **IO breakpoint match**. Read-only,Volatile. Reset: 0h.
11	**TF**: **EFLAGS TF value**. Read-only,Volatile. Reset: 0.
10:7	Reserved.
6	**SZ32**: **size 32 bits**. Read-only,Volatile. Reset: 0. 1=Port access was 32 bits.
5	**SZ16**: **size 16 bits**. Read-only,Volatile. Reset: 0. 1=Port access was 16 bits.
4	**SZ8**: **size 8 bits**. Read-only,Volatile. Reset: 0. 1=Port access was 8 bits.
3	**REP**: **repeated port access**. Read-only,Volatile. Reset: 0.
2	**STR**: **string-based port access**. Read-only,Volatile. Reset: 0.
1	**V**: **IO trap word valid**. Read-only,Volatile. Reset: 0. 0=The other fields of this offset are not valid. 1=The core entered SMM on an IO instruction boundary; all information in this offset is valid.
0	**RW**: **port access type**. Read-only,Volatile. Reset: 0. 0=IO Write (OUT instruction). 1=IO Read (IN instruction).

SMMxFEC4 [Local SMI Status] (Core::X86::Smm::LocalSmiStatus)

Read-only,Volatile. Reset: 0000_0000h.

This offset stores status bits associated with SMI sources local to the core. For each of these bits, 1=The associated mechanism generated an SMI.

Bits	Description
31:9	Reserved.
8	**MceRedirSts**: **machine check exception redirection status**. Read-only,Volatile. Reset: 0. This bit is associated with the SMI source specified in Core::X86::Msr::McExcepRedir[RedirSmiEn].
7:4	Reserved.
3:0	**IoTrapSts**: **IO trap status**. Read-only,Volatile. Reset: 0h. Each of these bits is associated with each of the respective SMI sources specified in Core::X86::Msr::SMI_ON_IO_TRAP.

SMMxFEC8 [IO Restart Byte] (Core::X86::Smm::IoRestart)

Read-write. Reset: 00h.

If the core entered SMM on an IO instruction boundary, the SMI handler may write this to FFh. This causes the core to re-execute the trapped IO instruction immediately after resuming from SMM. The SMI handler should only write to this byte if Core::X86::Smm::TrapOffset[V] == 1; otherwise, the behavior is undefined.

If a second SMI is asserted while a valid IO instruction is trapped by the first SMI handler, the core services the second SMI prior to re-executing the trapped IO instruction. Core::X86::Smm::TrapOffset[V] == 0 during the second entry into SMM, and the second SMI handler must not rewrite this byte.

If there is a simultaneous SMI IO instruction trap and debug breakpoint trap, the processor first responds to the SMI and postpones recognizing the debug exception until after resuming from SMM. If debug registers other than DR6 and DR7 are used while in SMM, they must be saved and restored by the SMI handler. If Core::X86::Smm::IoRestart is set to FFh when the RSM instruction is executed, the debug trap does not occur until after the IO instruction is re-executed.

Bits	Description
7:0	**RST**: **SMM IO Restart Byte**. Read-write. Reset: 00h.

SMMxFEC9 [Auto Halt Restart Offset] (Core::X86::Smm::AutoHalt)

Read-write. Reset: 00h.

Bits	Description

7:1	Reserved.
0	**HLT**: **halt restart**. Read-write. Reset: 0. 0=Entered SMM on a normal x86 instruction boundary. 1=Entered SMM from the Halt state. Upon SMM entry, this bit indicates whether SMM was entered from the Halt state. Before returning from SMM, this bit can be written by the SMI handler to specify whether the return from SMM should take the processor back to the Halt state or to the instruction-execution state specified by the SMM state save area (normally, the instruction after the halt). Clearing this bit the returns to the instruction specified in the SMM save state. Setting this bit returns to the halt state. If the return from SMM takes the processor back to the Halt state, the HLT instruction is not refetched and re-executed. However, the Halt special bus cycle is broadcast and the processor enters the Halt state.

SMMxFECA [NMI Mask] (Core::X86::Smm::NmiMask)

Read-write. Reset: 00h.

Bits	Description
7:1	Reserved.
0	**NmiMask**: **NMI Mask**. Read-write. Reset: 0. 0=NMI not masked. 1=NMI masked. Specifies whether NMI was masked upon entry to SMM.

SMMxFED8 [SMM SVM State] (Core::X86::Smm::SvmState)

Read-only,Volatile. Reset: 0000_0000_0000_0000h.

This offset stores the SVM state of the processor upon entry into SMM.

Bits	Description
63:4	Reserved.
3	**HostEflagesIF**: **host EFLAGS IF**. Read-only,Volatile. Reset: 0.
2:0	**SvmState**. Read-only,Volatile. Reset: 0h.
	ValidValues:

Value	Description
0h	SMM entered from a non-guest state.
1h	Reserved.
2h	SMM entered from a guest state.
5h-3h	Reserved.
6h	SMM entered from a guest state with nested paging enabled.
7h	Reserved.

SMMxFEFC [SMM Revision Identifier] (Core::X86::Smm::SmmRevID)

Read-only. Reset: 0003_0064h.

This offset stores the SVM state of the processor upon entry into SMM.

Bits	Description
31:18	Reserved.
17	**BRL**. Read-only. Reset: 1. 1=Base relocation supported.
16	**IOTrap**. Read-only. Reset: 1. 1=IO trap supported.
15:0	**Revision**. Read-only. Reset: 0064h.

SMMxFE00 [SMM Base Address] (Core::X86::Smm::SmmBase)

Read-write,Volatile. Reset: 0000_0000_0000_0000h.

This offset stores the base of the SMM-State of the processor upon entry into SMM.

Bits	Description
63:32	Reserved.
31:0	**SmmBase**. Read-write,Volatile. Reset: 0000_0000h. See Core::X86::Msr::SMM_BASE[SmmBase].

2.1.14.1.7 Exceptions and Interrupts in SMM

When SMM is entered, the core masks INTR, NMI, SMI, and INIT interrupts. The core clears the IF flag to disable INTR interrupts. To enable INTR interrupts within SMM, the SMM handler must set the IF flag to 1.

Generating an INTR interrupt can be used for unmasking NMI interrupts in SMM. The core recognizes the assertion of NMI within SMM immediately after the completion of an IRET instruction. Once NMI is recognized within SMM, NMI recognition remains enabled until SMM is exited, at which point NMI masking is restored to the state it was in before entering SMM.

While in SMM, the core responds to STPCLK interrupts, as well as to all exceptions that may be caused by the SMI handler.

2.1.14.1.8 The Protected ASeg and TSeg Areas

These ranges are controlled by Core::X86::Msr::SMMAddr and Core::X86::Msr::SMMMask; see those registers for details.

2.1.14.1.9 SMM Special Cycles

Special cycles can be initiated on entry and exit from SMM to acknowledge to the system that these transitions are occurring. These are controlled by Core::X86::Msr::HWCR[RsmSpCycDis,SmiSpCycDis].

2.1.14.1.10 Locking SMM

The SMM registers (Core::X86::Msr::SMMAddr and Core::X86::Msr::SMMMask) can be locked from being altered by setting Core::X86::Msr::HWCR[SmmLock]. SBIOS must lock the SMM registers after initialization to prevent unexpected changes to these registers.

2.1.14.1.11 SMM Page Configuration Lock

The SMM Page Configuration Lock feature allows SMM handler code to lock the paging configuration. Once locked, the paging configuration cannot be modified until RSM completes.
Core::X86::Cpuid::FeatureExt2Eax[SmmPgCfgLock] Specifies SMM page configuration locking is supported.
Core::X86::Msr::HWCR[SmmPgCfgLock] locks registers related to page configuration. If not in SMM mode, Error-on-write-1. Cleared on RSM instruction.
If Core::X86::Msr::HWCR[SmmPgCfgLock], WRMSR of Core::X86::Msr::EFER results in an error.
If Core::X86::Msr::HWCR[SmmPgCfgLock], MOV CR0, CR3 and CR4 instructions result in an error.

2.1.14.2 Local APIC

Family 19h, Model 01h supports the APIC interrupt controller and the X2APIC interrupt controllers.
See 2.1.14.2.2 [Local APIC Registers] for the APIC registers and Core::X86::Msr::APIC_ID through Core::X86::Msr::ExtendedInterruptLvtEntries for the X2APIC registers.

2.1.14.2.1 Local APIC Functional Description

The local APIC contains logic to receive interrupts from a variety of sources and to send interrupts to other local APICs, as well as registers to control its behavior and report status. Interrupts can be received from:

- IO devices including the IO hub (IO APICs)
- Other local APICs (inter-processor interrupts)

- APIC timer
- Thermal events
- Performance counters
- Legacy local interrupts from the IO hub (INTR and NMI)
- APIC internal errors

The APIC timer, thermal events, performance counters, local interrupts, and internal errors are all considered local interrupt sources, and their routing is controlled by local vector table entries. These entries assign a message type and vector to each interrupt, allow them to be masked, and track the status of the interrupt.

IO and inter-processor interrupts have their message type and vector assigned at the source and are unaltered by the local APIC. They carry a destination field and a mode bit that together determine which local APIC(s) accepts them. The destination mode (DM) bit specifies if the interrupt request packet should be handled in physical or logical destination mode.

2.1.14.2.1.1 Detecting and Enabling

The presence of APIC is detected via Core::X86::Cpuid::FeatureIdEdx[APIC], and the presence of X2APIC is detected via Core::X86::Cpuid::FeatureIdEcx[X2APIC].
The local APIC is enabled via Core::X86::Msr::APIC_BAR[ApicEn]. The X2APIC is enabled via Core::X86::Msr::APIC_BAR[x2ApicEn]. Reset forces the APIC and X2APIC disabled.

2.1.14.2.1.2 APIC Register Space

MMIO APIC space:
- Memory mapped to a 4 KB range. The memory type of this space is the UC memory type. The base address of this range is specified by {Core::X86::Msr::APIC_BAR[ApicBar[47:12]],000h}.
- The mnemonic is defined to be APICxXXX; where XXX is the byte address offset from the base address starting with APICx020 through APICx530 (Core::X86::Apic::ApicId - Core::X86::Apic::ExtendedInterruptLvtEntries).
- Treated as normal memory space when APIC is disabled, as specified by Core::X86::Msr::APIC_BAR[ApicEn].

MSR X2APIC space:
- The local APIC register space in x2APIC mode.
- MMIO APIC registers in x2APIC mode is defined by the register from MSR0000_0802 to MSR0000_08[53:50] (Core::X86::Msr::APIC_ID through Core::X86::Msr::ExtendedInterruptLvtEntries).
- If (Core::X86::Msr::APIC_BAR[x2ApicEn] == 0) then GP-read-write.
- RDMSR/WRMSR will occur in program order.

2.1.14.2.1.3 ApicId Enumeration Requirements

Note: Family 19h Model 00h-0Fh processors do not require contiguous ApicId assignments.

Operating systems are expected to use Core::X86::Cpuid::SizeId[ApicIdSize], the number of least significant bits in the Initial APIC ID that indicate core ID within a processor, in constructing per-core CPUID masks. Core::X86::Cpuid::SizeId[ApicIdSize] 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 Core::X86::Cpuid::SizeId[NC].

2.1.14.2.1.4 Physical Destination Mode

The interrupt is only accepted by the local APIC whose Core::X86::Apic::ApicId[ApicId] matches the destination field of the interrupt. Physical mode allows up to 255 APICs to be addressed individually.

2.1.14.2.1.5 Logical Destination Mode

A local APIC accepts interrupts selected by Core::X86::Apic::LocalDestination and the destination field of the interrupt using either cluster or flat format as configured by Core::X86::Apic::DestinationFormat[Format].

If flat destinations are in use, bits[7:0] of Core::X86::Apic::LocalDestination[Destination] are checked against bits[7:0] of the arriving interrupt's destination field. If any bit position is set in both fields, the local APIC is a valid destination. Flat format allows up to 8 APICs to be addressed individually.

If cluster destinations are in use, bits[7:4] of Core::X86::Apic::LocalDestination[Destination] are checked against bits[7:4] of the arriving interrupt's destination field to identify the cluster. If all of bits[7:4] match, then bits[3:0] of Core::X86::Apic::LocalDestination[Destination] and the interrupt destination are checked for any bit positions that are set in both fields to identify processors within the cluster. If both conditions are met, the local APIC is a valid destination. Cluster format allows 15 clusters of 4 APICs each to be addressed.

2.1.14.2.1.6 Interrupt Delivery

SMI, NMI, INIT, Startup, and External interrupts are classified as non-vectored interrupts.

When an APIC accepts a non-vectored interrupt, it is handled directly by the processor instead of being queued in the APIC. When an APIC accepts a fixed or lowest-priority interrupt, it sets the bit in Core::X86::Apic::InterruptRequest corresponding to the vector in the interrupt. For local interrupt sources, this comes from the vector field in that interrupt's local vector table entry. The corresponding bit in Core::X86::Apic::TriggerMode is set if the interrupt is level-triggered and cleared if edge-triggered. If a subsequent interrupt with the same vector arrives when the corresponding bit in Core::X86::Apic::InterruptRequest[RequestBits] is already set, the two interrupts are collapsed into one. Vectors 15-0 are reserved.

2.1.14.2.1.7 Vectored Interrupt Handling

Core::X86::Apic::TaskPriority and Core::X86::Apic::ProcessorPriority each contain an 8-bit priority divided into a main priority (bits[7:4]) and a priority sub-class (bits[3:0]). The task priority is assigned by software to set a threshold priority at which the processor is interrupted.

The processor priority is calculated by comparing the main priority (bits[7:4]) of Core::X86::Apic::TaskPriority[Priority] to bits[7:4] of the 8-bit encoded value of the highest bit set in Core::X86::Apic::InService. The processor priority is the higher of the two main priorities.

The processor priority is used to determine if any accepted interrupts (indicated by Core::X86::Apic::InterruptRequest[RequestBits]) are high enough priority to be serviced by the processor. When the processor is ready to service an interrupt, the highest bit in Core::X86::Apic::InterruptRequest[RequestBits] is cleared, and the corresponding bit is set in Core::X86::Apic::InService[InServiceBits].

When the processor has completed service for an interrupt, it performs a Write to Core::X86::Apic::EndOfInterrupt, clearing the highest bit in Core::X86::Apic::InService[InServiceBits] and causing the next-highest interrupt to be serviced. If the corresponding bit in Core::X86::Apic::TriggerMode[TriggerModeBits] is set, a Write to Core::X86::Apic::EndOfInterrupt is performed on all APICs to complete service of the interrupt at the source.

2.1.14.2.1.8 Interrupt Masking

Interrupt masking is controlled by the Core::X86::Apic::ExtendedApicControl. If Core::X86::Apic::ExtendedApicControl[IerEn] is set, Core::X86::Apic::InterruptEnable are used to mask interrupts. Any bit in Core::X86::Apic::InterruptEnable[InterruptEnableBits] that is clear indicates the corresponding interrupt is masked.

A masked interrupt is not serviced and the corresponding bit in Core::X86::Apic::InterruptRequest[RequestBits] remains set.

2.1.14.2.1.9 Spurious Interrupts

In the event that the task priority is set to or above the level of the interrupt to be serviced, the local APIC delivers a spurious interrupt vector to the processor, as specified by Core::X86::Apic::SpuriousInterruptVector. Core::X86::Apic::InService is not changed and no Write to Core::X86::Apic::EndOfInterrupt occurs.

2.1.14.2.1.10 Spurious Interrupts Caused by Timer Tick Interrupt

A typical interrupt is asserted until it is serviced. An interrupt is de-asserted when software clears the interrupt status bit within the interrupt service routine. Timer tick interrupt is an exception, since it is de-asserted regardless of whether it is serviced or not.

The processor is not always able to service interrupts immediately (i.e., when interrupts are masked by clearing EFLAGS.IM).

If the processor is not able to service the timer tick interrupt for an extended period of time, the INTR caused by the first timer tick interrupt asserted during that time is delivered to the local APIC in ExtInt mode and latched, and the subsequent timer tick interrupts are lost. The following cases are possible when the processor is ready to service interrupts:

- An ExtInt interrupt is pending, and INTR is asserted. This results in timer tick interrupt servicing. This occurs 50 percent of the time.
- An ExtInt interrupt is pending, and INTR is de-asserted. The processor sends the interrupt acknowledge cycle, but when the PIC receives it, INTR is de-asserted, and the PIC sends a spurious interrupt vector. This occurs 50 percent of the time.

There is a 50 percent probability of spurious interrupts to the processor.

2.1.14.2.1.11 Lowest-Priority Interrupt Arbitration

Fixed and non-vectored interrupts are accepted by their destination APICs without arbitration.

Delivery of lowest-priority interrupts requires all APICs to arbitrate to determine which one accepts the interrupt. If Core::X86::Apic::SpuriousInterruptVector[FocusDisable] is clear, then the focus processor for an interrupt always accepts the interrupt. A processor is the focus of an interrupt if it is already servicing that interrupt (corresponding bit in Core::X86::Apic::InService[InServiceBits] is set) or if it already has a pending request for that interrupt (corresponding bit in Core::X86::Apic::InterruptRequest[RequestBits] is set). If Core::X86::Apic::ExtendedApicControl[IerEn] is set, the interrupt must also be enabled in Core::X86::Apic::InterruptEnable[InterruptEnableBits] for a processor to be the focus processor. If there is no focus processor for an interrupt, or focus processor checking is disabled, then each APIC calculates an arbitration priority value, stored in Core::X86::Apic::ArbitrationPriority, and the one with the lowest result accepts the interrupt.

The arbitration priority value is calculated by comparing Core::X86::Apic::TaskPriority[Priority] with the 8-bit encoded value of the highest bit set in Core::X86::Apic::InterruptRequest[RequestBits] (IRRVec) and the 8-bit encoded value of the highest bit set Core::X86::Apic::InService[InServiceBits] (ISRVec). If Core::X86::Apic::ExtendedApicControl[IerEn] is set the IRRVec and ISRVec are based off the highest enabled interrupt. The main priority bits[7:4] are compared as follows:

```
if ((TaskPriority[Priority[7:4]] >= InterruptRequest[IRRVec[7:4]])
&&(TaskPriority[Priority[7:4]] > InService[ISRVec[7:4]])) {
ArbitrationPriority[Priority] = TaskPriority[Priority]
```

```
} elsif { (InterruptRequest[IRRVec[7:4]] > InService[ISRVec[7:4]])
ArbitrationPriority[Priority] = {InterruptRequest[IRRVec[7:4]],0h}
} else {
ArbitrationPriority[Priority] = {InService[ISRVect[7:4]],0h}
}
```

2.1.14.2.1.12 Inter-Processor Interrupts

The Core::X86::Apic::InterruptCommandLow and Core::X86::Apic::InterruptCommandHigh provide a mechanism for generating interrupts in order to redirect an interrupt to another processor, originate an interrupt to another processor, or allow a processor to interrupt itself. A Write to register Core::X86::Apic::InterruptCommandLow causes an interrupt to be generated with the properties specified by the Core::X86::Apic::InterruptCommandLow and Core::X86::Apic::InterruptCommandHigh fields.

Message type (bits[10:8]) == 011b (Remote Read) is deprecated.

Not all combinations of ICR fields are valid. Only the following combinations are valid:
Note: x indicates a don't care.

Table 22: ICR Valid Combinations

Message Type	Trigger Mode	Level	Destination Shorthand
Fixed	Edge	x	x
	Level	Assert	x
Lowest Priority, SMI, NMI, INIT	Edge	x	Destination or all excluding self
	Level	Assert	Destination or all excluding self
Startup	x	x	Destination or all excluding self

2.1.14.2.1.13 APIC Timer Operation

The local APIC contains a 32-bit timer, controlled by Core::X86::Apic::TimerLvtEntry, Core::X86::Apic::TimerInitialCount, and Core::X86::Apic::TimerDivideConfiguration. The processor bus clock is divided by the value in Core::X86::Apic::TimerDivideConfiguration[Div[3:0]] to obtain a time base for the timer. When Core::X86::Apic::TimerInitialCount[Count] is written, the value is copied into Core::X86::Apic::TimerCurrentCount. Core::X86::Apic::TimerCurrentCount[Count] is decremented at the rate of the divided clock. When the count reaches 0, a timer interrupt is generated with the vector specified in Core::X86::Apic::TimerLvtEntry[Vector]. If Core::X86::Apic::TimerLvtEntry[Mode] specifies periodic operation, Core::X86::Apic::TimerCurrentCount[Count] is reloaded with the Core::X86::Apic::TimerInitialCount[Count] value, and it continues to decrement at the rate of the divided clock. If Core::X86::Apic::TimerLvtEntry[Mask] is set, timer interrupts are not generated.

2.1.14.2.1.14 Generalized Local Vector Table

All LVTs (Core::X86::Apic::ThermalLvtEntry to Core::X86::Apic::LVTLINT, and Core::X86::Apic::ExtendedInterruptLvtEntries) support a generalized message type as follows:

- 000b=Fixed
- 010b=SMI
- 100b=NMI
- 111b=ExtINT
- All other messages types are Reserved.

2.1.14.2.1.15 State at Reset

At power-up or reset, the APIC is hardware disabled (Core::X86::Msr::APIC_BAR[ApicEn] == 0) so only SMI, NMI, INIT, and ExtInt interrupts may be accepted.

The APIC can be software disabled through Core::X86::Apic::SpuriousInterruptVector[APICSWEn]. The software disable has no effect when the APIC is hardware disabled.

When a processor accepts an INIT interrupt, the APIC is reset as at power-up, with the exception that:
- Core::X86::Apic::ApicId is unaffected.
- Pending APIC register writes complete.

2.1.14.2.2 Local APIC Registers

APICx020 [APIC ID] (Core::X86::Apic::ApicId)

Read-only.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; APICx020; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}

Bits	Description
31:24	**ApicId**: **APIC ID**. Read-only. Reset: XXh. The reset value varies based on core number. See 2.1.14.2.1.3 [ApicId Enumeration Requirements].
23:0	Reserved.

APICx030 [APIC Version] (Core::X86::Apic::ApicVersion)

Read-only.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; APICx030; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}

Bits	Description
31	**ExtApicSpace**: **extended APIC register space present**. Read-only. Reset: 1. 1=Indicates the presence of extended APIC register space starting at Core::X86::Apic::ExtendedApicFeature.
30:25	Reserved.
24	**DirectedEoiSupport**: **directed EOI support**. Read-only. Reset: Fixed,0. 0=Directed EOI capability not supported.
23:16	**MaxLvtEntry**. Read-only. Reset: XXh. Specifies the number of entries in the local vector table minus one.
15:8	Reserved.
7:0	**Version**. Read-only. Reset: 10h. Indicates the version number of this APIC implementation.

APICx080 [Task Priority] (Core::X86::Apic::TaskPriority)

Read-write. Reset: 0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; APICx080; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}

Bits	Description
31:8	Reserved.
7:0	**Priority**. Read-write. Reset: 00h. This field is assigned by software to set a threshold priority at which the core is interrupted.

APICx090 [Arbitration Priority] (Core::X86::Apic::ArbitrationPriority)

Read-only,Volatile. Reset: 0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; APICx090; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}

Bits	Description
31:8	Reserved.
7:0	**Priority**. Read-only,Volatile. Reset: 00h. Indicates the current priority for a pending interrupt, or a task or interrupt being serviced by the core. The priority is used to arbitrate between cores to determine which accepts a lowest-priority interrupt request.

APICx0A0 [Processor Priority] (Core::X86::Apic::ProcessorPriority)

Read-only,Volatile. Reset: 0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; APICx0A0; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}

Bits	Description
31:8	Reserved.
7:0	**Priority**. Read-only,Volatile. Reset: 00h. Indicates the core's current priority servicing a task or interrupt, and is used to determine if any pending interrupts should be serviced. It is the higher value of the task priority value and the current highest in-service interrupt.

APICx0B0 [End of Interrupt] (Core::X86::Apic::EndOfInterrupt)

Write-only.

This register is written by the software interrupt handler to indicate the servicing of the current interrupt is complete.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; APICx0B0; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}

Bits	Description
31:0	Reserved.

APICx0C0 [Reserved] (Core::X86::Apic::RemoteRead)

Read-only. Reset: 0000_0000h.

Remote Read is deprecated.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; APICx0C0; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}

Bits	Description
31:0	Reserved.

APICx0D0 [Logical Destination] (Core::X86::Apic::LocalDestination)

Read-write,Volatile. Reset: 0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; APICx0D0; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}

Bits	Description
31:24	**Destination**. Read-write,Volatile. Reset: 00h. This APIC's destination identification. Used to determine which interrupts should be accepted.
23:0	Reserved.

APICx0E0 [Destination Format] (Core::X86::Apic::DestinationFormat)

Read-write. Reset: F000_0000h.

Only supported in xAPIC mode.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; APICx0E0; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}

<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>31:28</td><td>Format. Read-write. Reset: Fh. Controls which format to use when accepting interrupts with a logical destination mode.
ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>Cluster destinations are used.</td></tr>
<tr><td>Eh-1h</td><td>Reserved.</td></tr>
<tr><td>Fh</td><td>Flat destinations are used.</td></tr>
</table></td></tr>
<tr><td>27:0</td><td>Reserved.</td></tr>
</table>

APICx0F0 [Spurious-Interrupt Vector] (Core::X86::Apic::SpuriousInterruptVector)

Reset: 0000_00FFh.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; APICx0F0; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}

Bits	Description
31:10	Reserved.
9	**FocusDisable**. Read-write. Reset: 0. 1=Disable focus core checking during lowest-priority arbitrated interrupts.
8	**APICSWEn**: **APIC software enable**. Read-write,Volatile. Reset: 0. 0=SMI, NMI, INIT, LINT[1:0], and Startup

	interrupts may be accepted; pending interrupts in Core::X86::Apic::InService and Core::X86::Apic::InterruptRequest are held, but further fixed, lowest-priority, and ExtInt interrupts are not accepted. All LVT entry mask bits are set and cannot be cleared.
7:0	**Vector**. Read-write,Volatile. Reset: FFh. The vector that is sent to the core in the event of a spurious interrupt.

APICx1[0...7]0 [In-Service] (Core::X86::Apic::InService)

Read-only,Volatile. Reset: 0000_0000h.

The in-service registers provide a bit per interrupt to indicate that the corresponding interrupt is being serviced by the core. The first 16 InServiceBits of the first Core::X86::Apic::InService register are Reserved.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n0; APICx100; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n1; APICx110; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n2; APICx120; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n3; APICx130; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n4; APICx140; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n5; APICx150; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n6; APICx160; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n7; APICx170; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}

Bits	Description
31:0	**InServiceBits**. Read-only,Volatile. Reset: 0000_0000h. These bits are set when the corresponding interrupt is being serviced by the core.

APICx1[8...F]0 [Trigger Mode] (Core::X86::Apic::TriggerMode)

Read-only,Volatile. Reset: 0000_0000h.

The trigger mode registers provide a bit per interrupt to indicate the assertion mode of each interrupt. The first 16 TriggerModeBits of the each thread's APIC[1F0:180] registers are Reserved.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n0; APICx180; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n1; APICx190; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n2; APICx1A0; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n3; APICx1B0; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n4; APICx1C0; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n5; APICx1D0; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n6; APICx1E0; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n7; APICx1F0; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}

Bits	Description
31:0	**TriggerModeBits**. Read-only,Volatile. Reset: 0000_0000h. The corresponding trigger mode bit is updated when an interrupt is accepted. 1=Level-triggered interrupt. 0=Edge-triggered interrupt.

APICx2[0...7]0 [Interrupt Request] (Core::X86::Apic::InterruptRequest)

Read-only. Reset: 0000_0000h.

The interrupt request registers provide a bit per interrupt to indicate that the corresponding interrupt has been accepted by the APIC. The first 16 RequestBits of the first Core::X86::Apic::InterruptRequest register are Reserved.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n0; APICx200; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n1; APICx210; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n2; APICx220; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n3; APICx230; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n4; APICx240; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n5; APICx250; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n6; APICx260; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n7; APICx270; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}

Bits	Description
31:0	**RequestBits**. Read-only. Reset: 0000_0000h. The corresponding request bit is set when the an interrupt is accepted by the APIC.

APICx280 [Error Status] (Core::X86::Apic::ErrorStatus)

Writes to this register trigger an update of the register state. The value written by software is arbitrary. Each write causes the internal error state to be loaded into this register, clearing the internal error state. Consequently, a second write prior to the occurrence of another error causes the register to be overwritten with cleared data.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; APICx280; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}

Bits	Description
31:8	Reserved.
7	**IllegalRegAddr**: **illegal register address**. Read-write. Reset: 0. This bit indicates that an access to a nonexistent register location within this APIC was attempted. Can only be set in xAPIC mode.
6	**RcvdIllegalVector**: **received illegal vector**. Read-write. Reset: 0. This bit indicates that this APIC has received a message with an illegal vector (00h to 0Fh for fixed and lowest priority interrupts).
5	**SentIllegalVector**. Read-write. Reset: 0. This bit indicates that this APIC attempted to send a message with an illegal vector (00h to 0Fh for fixed and lowest priority interrupts).
4	Reserved.
3	**RcvAcceptError**: **receive accept error**. Read-write. Reset: 0. This bit indicates that a message received by this APIC was not accepted by this or any other APIC.
2	**SendAcceptError**. Read-write. Reset: 0. This bit indicates that a message sent by this APIC was not accepted by any APIC.
1:0	Reserved.

APICx300 [Interrupt Command Low] (Core::X86::Apic::InterruptCommandLow)

Reset: 0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; APICx300; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}

<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>31:20</td><td>Reserved.</td></tr>
<tr><td rowspan="3">19:18</td><td>DestShrthnd: destination shorthand. Read-write. Reset: 0h.</td></tr>
<tr><td>Description: Provides a quick way to specify a destination for a message.
If all including self or all excluding self is used, then destination mode is ignored and physical is automatically used.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>No shorthand (Destination field).</td></tr>
<tr><td>1h</td><td>Self.</td></tr>
<tr><td>2h</td><td>All including self.</td></tr>
<tr><td>3h</td><td>All excluding self (This sends a message with a destination encoding of all 1s, so if lowest priority is used the message could end up being reflected back to this APIC).</td></tr>
</table></td></tr>
<tr><td rowspan="2">17:16</td><td>RemoteRdStat. Read-only. Reset: 0h.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>Read was invalid.</td></tr>
<tr><td>1h</td><td>Delivery pending.</td></tr>
<tr><td>2h</td><td>Delivery complete and access was valid.</td></tr>
<tr><td>3h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>15</td><td>TM: trigger mode. Read-write. Reset: 0. 0=Edge triggered. 1=Level triggered. Indicates how this interrupt is triggered.</td></tr>
<tr><td>14</td><td>Level. Read-write. Reset: 0. 0=De-asserted. 1=Asserted.</td></tr>
<tr><td>13</td><td>Reserved.</td></tr>
<tr><td>12</td><td>DS: interrupt delivery status. Read-only. Reset: 0. 0=Idle. 1=Send pending. In xAPIC mode this bit is set to indicate that the interrupt has not yet been accepted by the destination core(s). Software may repeatedly write Core::X86::Apic::InterruptCommandLow without polling the DS bit; all requested IPIs are delivered.</td></tr>
<tr><td>11</td><td>DM: destination mode. Read-write. Reset: 0. 0=Physical. 1=Logical.</td></tr>
<tr><td rowspan="2">10:8</td><td>MsgType. Read-write. Reset: 0h. The message types are encoded as follows:</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
</table></td></tr>
</table>

	0h	Fixed
	1h	Lowest Priority.
	2h	SMI
	3h	Reserved.
	4h	NMI
	5h	INIT
	6h	Startup
	7h	External interrupt.
7:0	**Vector**. Read-write. Reset: 00h. The vector that is sent for this interrupt source.	

APICx310 [Interrupt Command High] (Core::X86::Apic::InterruptCommandHigh)

Read-write. Reset: 0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; APICx310; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}

Bits	Description
31:24	**DestinationField**. Read-write. Reset: 00h. The destination encoding used when Core::X86::Apic::InterruptCommandLow[DestShrthnd] is 00b.
23:0	Reserved.

APICx320 [LVT Timer] (Core::X86::Apic::TimerLvtEntry)

Reset: 0001_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; APICx320; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}

Bits	Description
31:18	Reserved.
17	**Mode**. Read-write. Reset: 0. 0=One-shot. 1=Periodic.
16	**Mask**. Read-write. Reset: 1. 0=Not masked. 1=Masked.
15:13	Reserved.
12	**DS**: **interrupt delivery status**. Read-only,Volatile. Reset: 0. 0=Idle. 1=Send pending. (Indicates that the interrupt has not yet been accepted by the core.)
11	Reserved.
10:8	**MsgType**: **message type**. Read-write. Reset: 0h. See2.1.14.2.1.14 [Generalized Local Vector Table].
7:0	**Vector**. Read-write. Reset: 00h. Interrupt vector number.

APICx330 [LVT Thermal Sensor] (Core::X86::Apic::ThermalLvtEntry)

Reset: 0001_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; APICx330; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}

Bits	Description
31:17	Reserved.
16	**Mask**. Read-write. Reset: 1. 0=Not masked. 1=Masked.
15:13	Reserved.
12	**DS**: **interrupt delivery status**. Read-only,Volatile. Reset: 0. 0=Idle. 1=Send pending. (Indicates that the interrupt has not yet been accepted by the core.)
11	Reserved.
10:8	**MsgType**: **message type**. Read-write. Reset: 0h. See2.1.14.2.1.14 [Generalized Local Vector Table].
7:0	**Vector**. Read-write. Reset: 00h. Interrupt vector number.

APICx340 [LVT Performance Monitor] (Core::X86::Apic::PerformanceCounterLvtEntry)

Reset: 0001_0000h.

Interrupts for this local vector table are caused by overflows of:

- Core::X86::Msr::PERF_LEGACY_CTL0..3(Performance Event Select [3:0]).
- Core::X86::Msr::PERF_CTL0..5(Performance Event Select [5:0]).

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; APICx340; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}

Bits	Description
31:17	Reserved.
16	**Mask**. Read-write. Reset: 1. 0=Not masked. 1=Masked.
15:13	Reserved.
12	**DS**: **interrupt delivery status**. Read-only,Volatile. Reset: 0. 0=Idle. 1=Send pending. (Indicates that the interrupt has not yet been accepted by the core.)
11	Reserved.
10:8	**MsgType**: **message type**. Read-write. Reset: 0h. See2.1.14.2.1.14 [Generalized Local Vector Table].
7:0	**Vector**. Read-write. Reset: 00h. Interrupt vector number.

APICx3[5...6]0 [LVT LINT[1:0]] (Core::X86::Apic::LVTLINT)

Reset: 0001_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n0; APICx350; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n1; APICx360; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}

Bits	Description
31:17	Reserved.
16	**Mask**. Read-write. Reset: 1. 0=Not masked. 1=Masked.
15	**TM**: **trigger mode**. Read-write. Reset: 0. 0=Edge. 1=Level.
14	**RmtIRR**. Read-only,Volatile. Reset: 0. If trigger mode is level, remote Core::X86::Apic::InterruptRequest is set when the interrupt has begun service. Remote Core::X86::Apic::InterruptRequest is cleared when the end of interrupt has occurred.
13	Reserved.
12	**DS**: **interrupt delivery status**. Read-only,Volatile. Reset: 0. 0=Idle. 1=Send pending. (Indicates that the interrupt has not yet been accepted by the core.)
11	Reserved.
10:8	**MsgType**: **message type**. Read-write. Reset: 0h. See2.1.14.2.1.14 [Generalized Local Vector Table].
7:0	**Vector**. Read-write. Reset: 00h. Interrupt vector number.

APICx370 [LVT Error] (Core::X86::Apic::ErrorLvtEntry)

Reset: 0001_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; APICx370; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}

Bits	Description
31:17	Reserved.
16	**Mask**. Read-write. Reset: 1. 0=Not masked. 1=Masked.
15:13	Reserved.
12	**DS**: **interrupt delivery status**. Read-only,Volatile. Reset: 0. 0=Idle. 1=Send pending. (Indicates that the interrupt has not yet been accepted by the core.)
11	Reserved.
10:8	**MsgType**: **message type**. Read-write. Reset: 0h. See 2.1.14.2.1.14 [Generalized Local Vector Table].
7:0	**Vector**. Read-write. Reset: 00h. Interrupt vector number.

APICx380 [Timer Initial Count] (Core::X86::Apic::TimerInitialCount)

Read-write,Volatile. Reset: 0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; APICx380; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}

Bits	Description
31:0	**Count**. Read-write,Volatile. Reset: 0000_0000h. The value copied into the current count register when the timer is loaded or reloaded.

APICx390 [Timer Current Count] (Core::X86::Apic::TimerCurrentCount)

Read-only,Volatile. Reset: 0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; APICx390; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}

Bits	Description

31:0	**Count**. Read-only,Volatile. Reset: 0000_0000h. The current value of the counter.

APICx3E0 [Timer Divide Configuration] (Core::X86::Apic::TimerDivideConfiguration)

Read-write. Reset: 0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; APICx3E0; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}

Bits	Description
31:4	Reserved.
3:0	**Div[3:0]**. Read-write. Reset: 0h. Div[2] is unused.

ValidValues:

Value	Description
0h	Divide by 2.
1h	Divide by 4.
2h	Divide by 8.
3h	Divide by 16.
7h-4h	Reserved.
8h	Divide by 32.
9h	Divide by 64.
Ah	Divide by 128.
Bh	Divide by 1.
Fh-Ch	Reserved.

APICx400 [Extended APIC Feature] (Core::X86::Apic::ExtendedApicFeature)

Read-only. Reset: 0004_0007h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; APICx400; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}

Bits	Description
31:24	Reserved.
23:16	**ExtLvtCount**: **extended local vector table count**. Read-only. Reset: 04h. This specifies the number of extended LVT registers (Core::X86::Apic::ExtendedInterruptLvtEntries) in the local APIC.
15:3	Reserved.
2	**ExtApicIdCap**: **extended APIC ID capable**. Read-only. Reset: 1. 1=The processor is capable of supporting an 8-bit APIC ID, as controlled by Core::X86::Apic::ExtendedApicControl[ExtApicIdEn].
1	**SeoiCap**: **specific end of interrupt capable**. Read-only. Reset: 1. 1=The Core::X86::Apic::SpecificEndOfInterrupt is present.
0	**IerCap**: **interrupt enable register capable**. Read-only. Reset: 1. This bit indicates that the Core::X86::Apic::InterruptEnable are present. See2.1.14.2.1.8 [Interrupt Masking].

APICx410 [Extended APIC Control] (Core::X86::Apic::ExtendedApicControl)

Read-write. Reset: 0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; APICx410; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}

Bits	Description
31:3	Reserved.
2	**ExtApicIdEn**: **extended APIC ID enable**. Read-write. Reset: 0. 1=Enable 8-bit APIC ID; Core::X86::Apic::ApicId[ApicId] supports an 8-bit value; an interrupt broadcast in physical destination mode requires that the IntDest[7:0] == 1111_1111b (instead of XXXX_1111b); a match in physical destination mode occurs when (IntDest[7:0] == ApicId[7:0]) instead of (IntDest[3:0] == ApicId[3:0]).
1	**SeoiEn**. Read-write. Reset: 0. 1=Enable SEOI generation when a write to Core::X86::Apic::SpecificEndOfInterrupt is received.
0	**IerEn**. Read-write. Reset: 0. 1=Enable writes to the interrupt enable registers.

APICx420 [Specific End Of Interrupt] (Core::X86::Apic::SpecificEndOfInterrupt)

Read-write. Reset: 0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; APICx420; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}

Bits	Description
31:8	Reserved.
7:0	**EoiVec**: **end of interrupt vector**. Read-write. Reset: 00h. A write to this field causes an end of interrupt cycle to be performed for the vector specified in this field. The behavior is undefined if no interrupt is pending for the specified interrupt vector.

APICx4[8...F]0 [Interrupt Enable] (Core::X86::Apic::InterruptEnable)

Read-write. Reset: FFFF_FFFFh.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n0; APICx480; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n1; APICx490; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n2; APICx4A0; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n3; APICx4B0; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n4; APICx4C0; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n5; APICx4D0; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n6; APICx4E0; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n7; APICx4F0; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}

Bits	Description
31:0	**InterruptEnableBits**. Read-write. Reset: FFFF_FFFFh. The interrupt enable bits can be used to enable each of the 256 interrupts.

APICx5[0...3]0 [Extended Interrupt Local Vector Table] (Core::X86::Apic::ExtendedInterruptLvtEntries)

Reset: 0001_0000h.

Assignments conventions:

- APIC500 provides a local vector table entry for IBS.
- APIC510 provides a local vector table entry for error thresholding. See Core::X86::Msr::McaIntrCfg[ThresholdLvtOffset].
- APIC520 provides a local vector table entry for Deferred errors. See MCi_CONFIG[DeferredIntType].

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n0; APICx500; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n1; APICx510; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n2; APICx520; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n3; APICx530; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}

Bits	Description
31:17	Reserved.
16	**Mask**. Read-write. Reset: 1. 0=Not masked. 1=Masked.
15:13	Reserved.
12	**DS**: **interrupt delivery status**. Read-only,Volatile. Reset: 0. 0=Idle. 1=Send pending. (Indicates that the interrupt has not yet been accepted by the core.)
11	Reserved.
10:8	**MsgType**: **message type**. Read-write. Reset: 0h. See2.1.14.2.1.14 [Generalized Local Vector Table].
7:0	**Vector**. Read-write. Reset: 00h. Interrupt vector number.

2.1.15 CPUID Instruction

Processor feature capabilities and configuration information are provided through the CPUID instruction. The information is accessed by (1) selecting the CPUID function setting EAX and optionally ECX for some functions, (2) executing the CPUID instruction, and (3) reading the results in the EAX, EBX, ECX, and EDX registers. The syntax CPUID FnXXXXXXXX_EiX[_xYYY] refers to the function where EAX == X, and optionally ECX == Y, and the registers specified by EiX. EiX can be any single register such as {EAX, EBX, ECX, and EDX}, or a range of registers, such as E[C,B,A]X. Undefined function numbers return 0's in all 4 registers.

Unless otherwise specified, single-bit feature fields are encoded as: 1=Feature is supported by the processor. 0=Feature is not supported by the processor. CPUID functions not listed are Reserved.

2.1.15.1 CPUID Instruction Functions

CPUID_Fn00000000_EAX [Processor Vendor and Largest Standard Function Number] (Core::X86::Cpuid::LargFuncNum)

Read-only. Reset: Fixed,0000_0010h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn00000000_EAX

Bits	Description
31:0	**LFuncStd**: **largest standard function**. Read-only. Reset: Fixed,0000_0010h. The largest CPUID standard function input value supported by the processor implementation.

CPUID_Fn00000000_EBX [Processor Vendor (ASCII Bytes [3:0])] (Core::X86::Cpuid::ProcVendEbx)

Read-only. Reset: Fixed,6874_7541h.

Core::X86::Cpuid::ProcVendEbx and Core::X86::Cpuid::ProcVendExtEbx return the same value.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn00000000_EBX

Bits	Description
31:0	**Vendor**. Read-only. Reset: Fixed,6874_7541h. ASCII Bytes [3:0] ("h t u A") of the string "AuthenticAMD".

CPUID_Fn00000000_ECX [Processor Vendor (ASCII Bytes [11:8])] (Core::X86::Cpuid::ProcVendEcx)

Read-only. Reset: Fixed,444D_4163h.

Core::X86::Cpuid::ProcVendEcx and Core::X86::Cpuid::ProcVendExtEcx return the same value.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn00000000_ECX

Bits	Description
31:0	**Vendor**. Read-only. Reset: Fixed,444D_4163h. ASCII Bytes [11:8] ("D M A c") of the string "AuthenticAMD".

CPUID_Fn00000000_EDX [Processor Vendor (ASCII Bytes [7:4])] (Core::X86::Cpuid::ProcVendEdx)

Read-only. Reset: Fixed,6974_6E65h.

Core::X86::Cpuid::ProcVendEdx and Core::X86::Cpuid::ProcVendExtEdx return the same value.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn00000000_EDX

Bits	Description
31:0	**Vendor**. Read-only. Reset: Fixed,6974_6E65h. ASCII Bytes [7:4] ("i t n e") of the string "AuthenticAMD".

CPUID_Fn00000001_EAX [Family, Model, Stepping Identifiers] (Core::X86::Cpuid::FamModStep)

Read-only.

Core::X86::Cpuid::FamModStep and Core::X86::Cpuid::FamModStepExt return the same value.

Family: Is an 8-bit value and is defined as: Family[7:0]=({0000b,BaseFamily[3:0]}+ExtendedFamily[7:0]).

- E.g., If BaseFamily[3:0] == Fh and ExtendedFamily[7:0] == 08h, then Family[7:0] = 17h.

Model: Is an 8-bit value and is defined as: Model[7:0]={ExtendedModel[3:0],BaseModel[3:0]}.

- E.g., If ExtendedModel[3:0] == 1h and BaseModel[3:0] == 8h, then Model[7:0] = 18h.
- Model numbers vary with product.

Model numbers are are assigned a letter, 0h = "A", 1h = "B", and so on. Model and Stepping form the Revision. E.g., B1.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn00000001_EAX

Bits	Description
31:28	Reserved.
27:20	**ExtFamily**: **extended family**. Read-only. Reset: 0Ah. See Family above.
19:16	**ExtModel**: **extended model**. Read-only. Reset: 0h. See Model above.
15:12	Reserved.
11:8	**BaseFamily**. Read-only. Reset: Fh. See Family description above.
7:4	**BaseModel**. Read-only. Reset: Xh. Model numbers vary with product.
3:0	**Stepping**. Read-only. Reset: 1h. Processor stepping (revision) for a specific model.

CPUID_Fn00000001_EBX [LocalApicId, LogicalProcessorCount, CLFlush] (Core::X86::Cpuid::FeatureIdEbx)

Read-only.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn00000001_EBX

Bits	Description
31:24	**LocalApicId**. Read-only. Reset: XXh. Initial local APIC physical ID.
23:16	**LogicalProcessorCount**: **logical processor count**. Read-only. Reset: Fixed,(Core::X86::Cpuid::SizeId[NC] + 1). Specifies the number of threads in the processor as Core::X86::Cpuid::SizeId[NC] + 1.
15:8	**CLFlush**. Read-only. Reset: Fixed,08h. CLFLUSH size in quadwords.
7:0	Reserved.

CPUID_Fn00000001_ECX [Feature Identifiers] (Core::X86::Cpuid::FeatureIdEcx)

Read-only.

These values can be over-written by Core::X86::Msr::CPUID_Features.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn00000001_ECX

Bits	Description
31	Reserved.
30	**RDRAND**. Read-only. Reset: Fixed,1. RDRAND instruction support.
29	**F16C**. Read-only. Reset: Fixed,1. Half-precision convert instruction support.
28	**AVX**. Read-only. Reset: Fixed,1. AVX instruction support.
27	**OSXSAVE**. Read-only. Reset: X. 1=The OS has enabled support for XGETBV/XSETBV instructions to query processor extended states. OS enabled support for XGETBV/XSETBV.
26	**XSAVE**. Read-only. Reset: Fixed,1. 1=Support provided for the XSAVE, XRSTOR, XSETBV, and XGETBV instructions and the XFEATURE_ENABLED_MASK register. XSAVE (and related) instruction support.
25	**AES**: **AES instruction support**. Read-only. Reset: X. AES instruction support.
24	Reserved.
23	**POPCNT**. Read-only. Reset: Fixed,1. POPCNT instruction.
22	**MOVBE**. Read-only. Reset: Fixed,1. MOVBE instruction support.
21	**X2APIC**. Read-only. Reset: Fixed,1. x2APIC capability.
20	**SSE42**. Read-only. Reset: Fixed,1. SSE4.2 instruction support.
19	**SSE41**. Read-only. Reset: Fixed,1. SSE4.1 instruction support.
18	Reserved.
17	**PCID**. Read-only. Reset: Fixed,1. Process context identifiers support.
16:14	Reserved.
13	**CMPXCHG16B**. Read-only. Reset: Fixed,1. CMPXCHG16B instruction.
12	**FMA**. Read-only. Reset: Fixed,1. FMA instruction support.
11:10	Reserved.
9	**SSSE3**. Read-only. Reset: Fixed,1. Supplemental SSE3 extensions.
8:4	Reserved.
3	**Monitor**. Read-only. Reset: !Core::X86::Msr::HWCR[MonMwaitDis]. Monitor/Mwait instructions.
2	Reserved.
1	**PCLMULQDQ**. Read-only. Reset: X. PCLMULQDQ instruction support.
0	**SSE3**. Read-only. Reset: Fixed,1. SSE3 extensions.

CPUID_Fn00000001_EDX [Feature Identifiers] (Core::X86::Cpuid::FeatureIdEdx)

Read-only.

These values can be over-written by Core::X86::Msr::CPUID_Features.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn00000001_EDX

Bits	Description
31:29	Reserved.
28	**HTT**. Read-only. Reset: Fixed,(Core::X86::Cpuid::SizeId[NC] != 0). 0=Single thread product

	(Core::X86::Cpuid::SizeId[NC] == 0). 1=Multi thread product (Core::X86::Cpuid::SizeId[NC] != 0). Hyper-threading technology.
27	Reserved.
26	**SSE2**. Read-only. Reset: Fixed,1. SSE2: SSE2 extensions.
25	**SSE**. Read-only. Reset: Fixed,1. SSE extensions.
24	**FXSR**. Read-only. Reset: Fixed,1. FXSAVE and FXRSTOR instructions.
23	**MMX**. Read-only. Reset: Fixed,1. MMX instructions
22:20	Reserved.
19	**CLFSH**. Read-only. Reset: Fixed,1. CLFLUSH instruction.
18	Reserved.
17	**PSE36**. Read-only. Reset: Fixed,1. Page-size extensions.
16	**PAT**. Read-only. Reset: Fixed,1. Page attribute table.
15	**CMOV**. Read-only. Reset: Fixed,1. Conditional move instructions, CMOV, FCOMI, FCMOV.
14	**MCA**. Read-only. Reset: Fixed,1. Machine check architecture, MCG_CAP.
13	**PGE**. Read-only. Reset: Fixed,1. Page global extension, CR4.PGE.
12	**MTRR**. Read-only. Reset: Fixed,1. Memory-type range registers.
11	**SysEnterSysExit**. Read-only. Reset: Fixed,1. SYSENTER and SYSEXIT instructions.
10	Reserved.
9	**APIC**: **advanced programmable interrupt controller (APIC) exists and is enabled**. Read-only. Reset: X. Core::X86::Msr::APIC_BAR[ApicEn].
8	**CMPXCHG8B**. Read-only. Reset: Fixed,1. CMPXCHG8B instruction.
7	**MCE**. Read-only. Reset: Fixed,1. Machine check exception, CR4.MCE.
6	**PAE**. Read-only. Reset: Fixed,1. Physical-address extensions (PAE).
5	**MSR**. Read-only. Reset: Fixed,1. AMD model-specific registers (MSRs), with RDMSR and WRMSR instructions.
4	**TSC**. Read-only. Reset: Fixed,1. Time Stamp Counter, RDTSC/RDTSCP instructions, CR4.TSD.
3	**PSE**. Read-only. Reset: Fixed,1. Page-size extensions (4 MB pages).
2	**DE**. Read-only. Reset: Fixed,1. Debugging extensions, IO breakpoints, CR4.DE.
1	**VME**. Read-only. Reset: Fixed,1. Virtual-mode enhancements.
0	**FPU**. Read-only. Reset: Fixed,1. x87 floating point unit on-chip.

CPUID_Fn00000005_EAX [Monitor/MWait] (Core::X86::Cpuid::MonMWaitEax)

Read-only. Reset: Fixed,0000_0040h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn00000005_EAX

Bits	Description
31:16	Reserved.
15:0	**MonLineSizeMin**. Read-only. Reset: Fixed,0040h. Smallest monitor-line size in bytes.

CPUID_Fn00000005_EBX [Monitor/MWait] (Core::X86::Cpuid::MonMWaitEbx)

Read-only. Reset: Fixed,0000_0040h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn00000005_EBX

Bits	Description
31:16	Reserved.
15:0	**MonLineSizeMax**. Read-only. Reset: Fixed,0040h. Largest monitor-line size in bytes.

CPUID_Fn00000005_ECX [Monitor/MWait] (Core::X86::Cpuid::MonMWaitEcx)

Read-only. Reset: Fixed,0000_0003h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn00000005_ECX

Bits	Description
31:2	Reserved.

1	**IBE**. Read-only. Reset: Fixed,1. Interrupt break-event.
0	**EMX**. Read-only. Reset: Fixed,1. Enumerate MONITOR/MWAIT extensions.

CPUID_Fn00000005_EDX [Monitor/MWait] (Core::X86::Cpuid::MonMWaitEdx)

Read-only. Reset: Fixed,0000_0011h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn00000005_EDX

Bits	Description
31:8	Reserved.
7:4	**MWaitC1SubStates**. Read-only. Reset: Fixed,1h. Number of C1 sub-cstates supported by MWAIT.
3:0	**MWaitC0SubStates**. Read-only. Reset: Fixed,1h. Number of C0 sub-cstates supported by MWAIT.

CPUID_Fn00000006_EAX [Thermal and Power Management] (Core::X86::Cpuid::ThermalPwrMgmtEax)

Read-only. Reset: Fixed,0000_0004h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn00000006_EAX

Bits	Description
31:3	Reserved.
2	**ARAT**: **always running APIC timer**. Read-only. Reset: Fixed,1. 1=Indicates support for APIC timer always running feature.
1:0	Reserved.

CPUID_Fn00000006_EBX [Thermal and Power Management] (Core::X86::Cpuid::ThermalPwrMgmtEbx)

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn00000006_EBX

Bits	Description
31:0	Reserved.

CPUID_Fn00000006_ECX [Thermal and Power Management] (Core::X86::Cpuid::ThermalPwrMgmtEcx)

Read-only. Reset: Fixed,0000_0001h.

These values can be over-written by Core::X86::Msr::CPUID_PWR_THERM.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn00000006_ECX

Bits	Description
31:1	Reserved.
0	**EffFreq**: **effective frequency interface**. Read-only. Reset: Fixed,1. 1=Indicates presence of Core::X86::Msr::MPERF and Core::X86::Msr::APERF.

CPUID_Fn00000006_EDX [Thermal and Power Management] (Core::X86::Cpuid::ThermalPwrMgmtEdx)

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn00000006_EDX

Bits	Description
31:0	Reserved.

CPUID_Fn00000007_EAX_x00 [Structured Extended Feature Identifiers] (Core::X86::Cpuid::StructExtFeatIdEax0)

Read-only. Reset: Fixed,0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn00000007_EAX_x00

Bits	Description
31:0	**StructExtFeatIdMax**. Read-only. Reset: Fixed,0000_0000h. The largest CPUID Fn0000_0007 sub-function supported by the processor implementation.

CPUID_Fn00000007_EBX_x00 [Structured Extended Feature Identifiers] (Core::X86::Cpuid::StructExtFeatIdEbx0)

Reset: Fixed,219C_95A9h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn00000007_EBX_x00

Bits	Description
31:30	Reserved.

29	**SHA**. Read-only. Reset: Fixed,1. 1=SHA Extensions available.
28:25	Reserved.
24	**CLWB**. Read-only. Reset: Fixed,1. Cache line write back.
23	**CLFSHOPT**. Read-only. Reset: Fixed,1. Optimized Cache Line Flush.
22:21	Reserved.
20	**SMAP**. Read-only. Reset: Fixed,1. Secure Mode Access Prevention is supported.
19	**ADX**. Read-only. Reset: Fixed,1. ADCX and ADOX are present.
18	**RDSEED**. Read-only. Reset: Fixed,1. RDSEED is present.
17:16	Reserved.
15	**PQE**. Read-only. Reset: Fixed,1. The processor supports Cache Allocation Technology.
14:13	Reserved.
12	**PQM**. Read-only. Reset: Fixed,1. Platform QoS Monitoring.
11	Reserved.
10	**INVPCID**. Read-only. Reset: Fixed,1. Invalidate processor context ID.
9	**ERMS**. Read-write. Reset: Fixed,0. Enhanced REP MOVSB/STOSB.
8	**BMI2**. Read-only. Reset: Fixed,1. Bit manipulation group 2 instruction support.
7	**SMEP**. Read-only. Reset: Fixed,1. Supervisor Mode Execution protection.
6	Reserved.
5	**AVX2**. Read-only. Reset: Fixed,1. AVX extension support.
4	Reserved.
3	**BMI1**. Read-only. Reset: Fixed,1. Bit manipulation group 1 instruction support.
2:1	Reserved.
0	**FSGSBASE**. Read-only. Reset: Fixed,1. FS and GS base read write instruction support.

CPUID_Fn00000007_ECX_x00 [Structured Extended Feature Identifier] (Core::X86::Cpuid::StructExtFeatIdEcx0)

Read-only.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn00000007_ECX_x00

Bits	Description
31:23	Reserved.
22	**RDPID**. Read-only. Reset: Fixed,1. Read Processor ID instruction support.
21:11	Reserved.
10	**VPCLMULQDQ**. Read-only. Reset: X. Vector VPCLMULQDQ instruction support
9	**VAES**. Read-only. Reset: X. Vector VAES(ENC|DEC), VAES(ENC|DEC)LAST instruction support.
8	Reserved.
7	**CET_SS**. Read-only. Reset: 1. 1=Shadow stack supported.
6:5	Reserved.
4	**OSPKE**. Read-only. Reset: X. Protection keys enabled.
3	**PKU**. Read-only. Reset: Fixed,1. Protection keys support.
2	**UMIP**. Read-only. Reset: Fixed,1. User Mode Instruction Prevention enable.
1:0	Reserved.

CPUID_Fn00000007_EDX_x00 [Structured Extended Feature Identifiers] (Core::X86::Cpuid::StructExtFeatIdEdx0)

Read-write. Reset: Fixed,0000_0000h.

Power-management feature flags.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn00000007_EDX_x00

Bits	Description
31:5	Reserved.
4	**FSRM**. Read-write. Reset: Fixed,0. Fast Short Rep Movsb supported.

3:0	Reserved.

CPUID_Fn0000000B_EAX_x00 [Extended Topology Enumeration] (Core::X86::Cpuid::ExtTopEnumEax0)

Read-only. Enable: (Core::X86::Cpuid::ExtTopEnumEbx0 > 0).

CPUID Fn0000_000B_E[D,C,B,A]X_x[2:0] specifies the hierarchy of logical cores from the SMT level through the processor socket level.
Software determines the presence of CPUID Fn0000_000B if (CPUID Fn0000_000B_EBX_x0[31:0] != 0). Software reads CPUID Fn0000_000B_E[C,B,A]X for ascending values of ECX until (CPUID Fn0000_000B_EBX[LogProcAtThisLevel] == 0).

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000B_EAX_x00

Bits	Description
31:5	Reserved.
4:0	**CoreMaskWidth**. Read-only. Number of bits to shift ExtendedApicId right to get unique topology ID of the next level type.
	Reset: SMT ? 01h : 00h.

CPUID_Fn0000000B_EBX_x00 [Extended Topology Enumeration] (Core::X86::Cpuid::ExtTopEnumEbx0)

Read-only.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000B_EBX_x00

Bits	Description
31:16	Reserved.
15:0	**LogProcAtThisLevel**. Read-only. Number of threads in a core.
	Reset: SMT ? 2 : 0001h.

CPUID_Fn0000000B_ECX_x00 [Extended Topology Enumeration] (Core::X86::Cpuid::ExtTopEnumEcx0)

Read-only. Reset: Fixed,0000_0100h. Enable: (Core::X86::Cpuid::ExtTopEnumEbx0 > 0).

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000B_ECX_x00

<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>31:16</td><td>Reserved.</td></tr>
<tr><td rowspan="2">15:8</td><td>LevelType. Read-only. Reset: Fixed,01h.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>00h</td><td>Invalid</td></tr>
<tr><td>01h</td><td>Thread</td></tr>
<tr><td>02h</td><td>Processor</td></tr>
<tr><td>FFh-03h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>7:0</td><td>EcxVal. Read-only. Reset: Fixed,00h. ECX input value.</td></tr>
</table>

CPUID_Fn0000000B_EAX_x01 [Extended Topology Enumeration] (Core::X86::Cpuid::ExtTopEnumEax1)

Read-only. Enable: (Core::X86::Cpuid::ExtTopEnumEbx1 > 0).

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000B_EAX_x01

Bits	Description
31:5	Reserved.
4:0	**CoreMaskWidth**. Read-only. Reset: XXXXXb. ExtendedApicId right shift value.

CPUID_Fn0000000B_EBX_x01 [Extended Topology Enumeration] (Core::X86::Cpuid::ExtTopEnumEbx1)

Read-only.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000B_EBX_x01

Bits	Description
31:16	Reserved.
15:0	**LogProcAtThisLevel**. Read-only. Reset: XXXXh. Number of logical cores in processor socket.

CPUID_Fn0000000B_ECX_x01 [Extended Topology Enumeration] (Core::X86::Cpuid::ExtTopEnumEcx1)

Read-only. Reset: Fixed,0000_0201h. Enable: (Core::X86::Cpuid::ExtTopEnumEbx1 > 0).

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000B_ECX_x01

<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>31:16</td><td>Reserved.</td></tr>
<tr><td rowspan="2">15:8</td><td>LevelType. Read-only. Reset: Fixed,02h.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>00h</td><td>Invalid</td></tr>
<tr><td>01h</td><td>Thread</td></tr>
<tr><td>02h</td><td>Processor</td></tr>
<tr><td>FFh-03h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>7:0</td><td>EcxVal. Read-only. Reset: Fixed,01h. ECX input value.</td></tr>
</table>

CPUID_Fn0000000B_EAX_x02 [Extended Topology Enumeration] (Core::X86::Cpuid::ExtTopEnumEax2)

Read-only. Reset: Fixed,0000_0000h. Enable: (Core::X86::Cpuid::ExtTopEnumEbx2 > 0).

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000B_EAX_x02

Bits	Description
31:5	Reserved.
4:0	**CoreMaskWidth**. Read-only. Reset: Fixed,00h. Zero indicates no more levels.

CPUID_Fn0000000B_EBX_x02 [Extended Topology Enumeration] (Core::X86::Cpuid::ExtTopEnumEbx2)

Read-only. Reset: 0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000B_EBX_x02

Bits	Description
31:16	Reserved.
15:0	**LogProcAtThisLevel**. Read-only. Reset: 0000h. Zero indicates no more levels.

CPUID_Fn0000000B_ECX_x02 [Extended Topology Enumeration] (Core::X86::Cpuid::ExtTopEnumEcx2)

Read-only. Reset: Fixed,0000_0002h. Enable: (Core::X86::Cpuid::ExtTopEnumEbx2 > 0).

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000B_ECX_x02

<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>31:16</td><td>Reserved.</td></tr>
<tr><td rowspan="2">15:8</td><td>LevelType. Read-only. Reset: Fixed,00h.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>00h</td><td>Invalid</td></tr>
<tr><td>01h</td><td>Thread</td></tr>
<tr><td>02h</td><td>Processor</td></tr>
<tr><td>FFh-03h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>7:0</td><td>EcxVal. Read-only. Reset: Fixed,02h. ECX input value.</td></tr>
</table>

CPUID_Fn0000000B_EDX [Extended Topology Enumeration] (Core::X86::Cpuid::ExtTopEnumEdx)

Read-only.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000B_EDX

Bits	Description
31:0	**ExtendedLocalApicId**: **extended APIC ID**. Read-only. Reset: XXXX_XXXXh. Extended APIC_ID.

CPUID_Fn0000000D_EAX_x00 [Processor Extended State Enumeration] (Core::X86::Cpuid::ProcExtStateEnumEax00)

Read-only. Reset: Fixed,0000_0207h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000D_EAX_x00

Bits	Description
31:0	**XFeatureSupportedMask[31:0]**. Read-only. Reset: Fixed,0000_0207h. Each set bit indicates the corresponding bit in register XCR0[31:0] is settable.

ValidValues:

Bit	Name	Description
[0]	X87	X87 Support.
[1]	SSE	128-bit SSE Support.
[2]	AVX	256-bit AVX support.
[8:3]		Reserved.
[9]	MPK	Memory Protection Keys. See Core::X86::Cpuid::StructExtFeatIdEcx0[PKU] for the availability of MPK feature support.
[31:10]		Reserved.

CPUID_Fn0000000D_EBX_x00 [Processor Extended State Enumeration] (Core::X86::Cpuid::ProcExtStateEnumEbx00)

Read-only,Volatile.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000D_EBX_x00

Bits	Description
31:0	**XFeatureEnabledSizeMax**. Read-only,Volatile. Reset: XXXX_XXXXh.

Description: Size in bytes of an uncompacted XSAVE/XRSTOR area for all features enabled in the XCR0 register.

```
IF (XCR0[MPK] == 1)
    Return EBX=0000_0988h // legacy header + X87/SSE + AVX + MPK size
ELSIF (XCR0[AVX] == 1)
    Return EBX = 0000_0340h // legacy header + X87/SSE + AVX size
ELSIF (XCR0[SSE] == 1)
    Return EBX = 0000_0240h // legacy header + X87/SSE size
ELSIF (XCR0[X87] == 1)
    Return EBX=0000_0240h
END
```

CPUID_Fn0000000D_ECX_x00 [Processor Extended State Enumeration] (Core::X86::Cpuid::ProcExtStateEnumEcx00)

Read-only. Reset: Fixed,0000_0988h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000D_ECX_x00

Bits	Description
31:0	**XFeatureSupportedSizeMax**. Read-only. Reset: Fixed,0000_0988h. Size of legacy header + X87/SSE + AVX+ Padding + MPK.

CPUID_Fn0000000D_EDX_x00 [Processor Extended State Enumeration] (Core::X86::Cpuid::ProcExtStateEnumEdx00)

Read-only. Reset: Fixed,0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000D_EDX_x00

Bits	Description
31:0	**XFeatureSupportedMask[63:32]**. Read-only. Reset: Fixed,0000_0000h. Each set bit indicates the corresponding bit in register XCR0[63:32] is settable.

CPUID_Fn0000000D_EAX_x01 [Processor Extended State Enumeration] (Core::X86::Cpuid::ProcExtStateEnumEax01)

Read-only. Reset: Fixed,0000_000Fh.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000D_EAX_x01

Bits	Description
31:4	Reserved.
3	**XSAVES**. Read-only. Reset: Fixed,1. XSAVES,XRSTORS, and XSS supported.
2	**XGETBV**. Read-only. Reset: Fixed,1. XGETBV with ECX = 1 supported.
1	**XSAVEC**. Read-only. Reset: Fixed,1. XSAVEC and compact XRSTOR supported.
0	**XSAVEOPT**. Read-only. Reset: Fixed,1. XSAVEOPT is available.

CPUID_Fn0000000D_EBX_x01 [Processor Extended State Enumeration] (Core::X86::Cpuid::ProcExtStateEnumEbx01)

Read-only,Volatile.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000D_EBX_x01

Bits	Description
31:0	**XFeatureEnabledSizeMax**. Read-only,Volatile. Reset: XXXX_XXXXh.
	Description: EBX = 0000_0240h + ((XCR0[AVX] == 1) ? 0000_0100h : 0) + ((XCR0[MPK] == 1) ? 0000_0008h : 0) + ((XSS[CET_U] == 1) ? 0000_0010h : 0) + ((XSS[CET_S] == 1) ? 0000_0018h : 0).

CPUID_Fn0000000D_ECX_x01 [Processor Extended State Enumeration] (Core::X86::Cpuid::ProcExtStateEnumEcx01)

Read-only. Reset: Fixed,0000_1800h.

Each set bit indicates the corresponding bit in register XSS[31:0] is settable.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000D_ECX_x01

Bits	Description
31:0	**MaskXss**. Read-only. Reset: Fixed,0000_1800h. Mask[31:0] of settable XSS bits.
	ValidValues:

Bit	Description
[10:0]	Reserved.
[11]	CET for user mode.
[12]	CET for supervisor mode.
[31:13]	Reserved.

CPUID_Fn0000000D_EDX_x01 [Processor Extended State Enumeration] (Core::X86::Cpuid::ProcExtStateEnumEdx01)

Read-only. Reset: Fixed,0000_0000h.

Each set bit indicates the corresponding bit in register XSS[63:32] is settable.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000D_EDX_x01

Bits	Description
31:0	**MaskXss**. Read-only. Reset: Fixed,0000_0000h. Mask[63:32] of settable XSS bits.

CPUID_Fn0000000D_EAX_x02 [Processor Extended State Enumeration] (Core::X86::Cpuid::ProcExtStateEnumEax02)

Read-only. Reset: Fixed,0000_0100h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000D_EAX_x02

Bits	Description
31:0	**YmmHiSaveStateOffset**. Read-only. Reset: Fixed,0000_0100h. YMM[31:16] save state byte size.

CPUID_Fn0000000D_EBX_x02 [Processor Extended State Enumeration] (Core::X86::Cpuid::ProcExtStateEnumEbx02)

Read-only. Reset: Fixed,0000_0240h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000D_EBX_x02

Bits	Description
31:0	**YmmHiSaveStateOffset**. Read-only. Reset: Fixed,0000_0240h. YMM[31:16] save state uncompacted byte offset.

CPUID_Fn0000000D_ECX_x02 [Processor Extended State Enumeration] (Core::X86::Cpuid::ProcExtStateEnumEcx02)

Read-only. Reset: Fixed,0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000D_ECX_x02

Bits	Description
31:2	Reserved.
1	**YmmHiAligned**. Read-only. Reset: Fixed,0. 0=YMM_hi state (YMM[31:16]) is not automatically aligned to a 64-byte boundary on compacted saves/restores. 1=YMM_hi state (YMM[31:16]) is automatically aligned to a 64-byte boundary on compacted saves/restores.
0	**XStateSupervisor**. Read-only. Reset: Fixed,0. 1=This xstate is Supervisor State.

CPUID_Fn0000000D_EDX_x02 [Processor Extended State Enumeration] (Core::X86::Cpuid::ProcExtStateEnumEdx02)

Read-only. Reset: Fixed,0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000D_EDX_x02

Bits	Description
31:0	Reserved.

CPUID_Fn0000000D_EAX_x09 [Processor Extended State Enumeration] (Core::X86::Cpuid::ProcExtStateEnumEax09)

Read-only. Reset: Fixed,0000_0008h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000D_EAX_x09

Bits	Description
31:0	**MpkSaveStateSize**. Read-only. Reset: Fixed,0000_0008h. MPK save state byte size.

CPUID_Fn0000000D_EBX_x09 [Processor Extended State Enumeration] (Core::X86::Cpuid::ProcExtStateEnumEbx09)

Read-only. Reset: Fixed,0000_0980h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000D_EBX_x09

Bits	Description
31:0	**MpKSaveStateOffset**. Read-only. Reset: Fixed,0000_0980h. MPK save state uncompacted byte offset.

CPUID_Fn0000000D_ECX_x09 [Processor Extended State Enumeration] (Core::X86::Cpuid::ProcExtStateEnumEcx09)

Read-only. Reset: Fixed,0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000D_ECX_x09

Bits	Description
31:2	Reserved.
1	**XState64BitAligned**. Read-only. Reset: Fixed,0. 1=This xstate will always be 64-byte aligned in compacted memops.
0	**XStateSupervisor**. Read-only. Reset: Fixed,0. 1=This xstate is Supervisor State.

CPUID_Fn0000000D_EDX_x09 [Processor Extended State Enumeration] (Core::X86::Cpuid::ProcExtStateEnumEdx09)

Read-only. Reset: Fixed,0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000D_EDX_x09

Bits	Description
31:0	Reserved.

CPUID_Fn0000000D_EAX_x0B [Processor Extended State Enumeration] (Core::X86::Cpuid::ProcExtStateEnumEax0B)

Read-only. Reset: Fixed,0000_0010h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000D_EAX_x0B

Bits	Description
31:0	**CetUserSize**. Read-only. Reset: Fixed,0000_0010h. : CET user size.

CPUID_Fn0000000D_EBX_x0B [Processor Extended State Enumeration] (Core::X86::Cpuid::ProcExtStateEnumEbx0B)

Read-only. Reset: Fixed,0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000D_EBX_x0B

Bits	Description
31:0	**CetUserOffset**. Read-only. Reset: Fixed,0000_0000h. CET user byte offset.

CPUID_Fn0000000D_ECX_x0B [Processor Extended State Enumeration] (Core::X86::Cpuid::ProcExtStateEnumEcx0B)

Read-only. Reset: Fixed,0000_0001h.

Processor Extended State Enumeration for CET_U.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000D_ECX_x0B

Bits	Description
31:2	Reserved.
1	**XState64BitAligned**. Read-only. Reset: Fixed,0. 1=This xstate will always be 64-byte aligned in compacted memops.
0	**XStateSupervisor**. Read-only. Reset: Fixed,1. 1=This xstate is Supervisor State.

CPUID_Fn0000000D_EDX_x0B [Processor Extended State Enumeration] (Core::X86::Cpuid::ProcExtStateEnumEdx0B)

Read-only. Reset: Fixed,0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000D_EDX_x0B

Bits	Description
31:0	Reserved.

CPUID_Fn0000000D_EAX_x0C [Processor Extended State Enumeration] (Core::X86::Cpuid::ProcExtStateEnumEax0C)

Read-only. Reset: Fixed,0000_0018h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000D_EAX_x0C

Bits	Description
31:0	**CetSprvrSize**. Read-only. Reset: Fixed,0000_0018h. CET supervisor size.

CPUID_Fn0000000D_EBX_x0C [Processor Extended State Enumeration] (Core::X86::Cpuid::ProcExtStateEnumEbx0C)

Read-only. Reset: Fixed,0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000D_EBX_x0C

Bits	Description
31:0	**CetSprvrOffset**. Read-only. Reset: Fixed,0000_0000h. CET supervisor byte offset.

CPUID_Fn0000000D_ECX_x0C [Processor Extended State Enumeration] (Core::X86::Cpuid::ProcExtStateEnumEcx0C)

Read-only. Reset: Fixed,0000_0001h.

Processor Extended State Enumeration for CET_S.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000D_ECX_x0C

Bits	Description
31:2	Reserved.
1	**XState64BitAligned**. Read-only. Reset: Fixed,0. 1=This xstate will always be 64-byte aligned in compacted memops.
0	**XStateSupervisor**. Read-only. Reset: Fixed,1. 1=This xstate is Supervisor State.

CPUID_Fn0000000D_EDX_x0C [Processor Extended State Enumeration] (Core::X86::Cpuid::ProcExtStateEnumEdx0C)

Read-only. Reset: Fixed,0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000D_EDX_x0C

Bits	Description
31:0	Reserved.

CPUID_Fn0000000F_EAX_x00 [Resource Director Technology Monitor Capability] (Core::X86::Cpuid::RsrcDirTechMonCapEax0)

Read-only. Reset: Fixed,0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000F_EAX_x00

Bits	Description
31:0	Reserved.

CPUID_Fn0000000F_EBX_x00 [Resource Director Technology Monitor Capability] (Core::X86::Cpuid::RsrcDirTechMonCapEbx0)

Read-only. Reset: Fixed,0000_00FFh.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000F_EBX_x00

Bits	Description
31:0	**RmidMaxRange**. Read-only. Reset: Fixed,0000_00FFh. RMID maximum within this processor for all types.

CPUID_Fn0000000F_ECX_x00 [Resource Director Technology Monitor Capability] (Core::X86::Cpuid::RsrcDirTechMonCapEcx0)

Read-only. Reset: Fixed,0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000F_ECX_x00

Bits	Description
31:0	Reserved.

CPUID_Fn0000000F_EDX_x00 [Resource Director Technology Monitor Capability] (Core::X86::Cpuid::RsrcDirTechMonCapEdx0)

Read-only. Reset: Fixed,0000_0002h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000F_EDX_x00

Bits	Description
31:2	Reserved.
1	**L3CacheRDT**. Read-only. Reset: Fixed,1. L3 Cache RDT Monitoring.
0	Reserved.

CPUID_Fn0000000F_EAX_x01 [Resource Director Technology L3 Monitor Capability] (Core::X86::Cpuid::RsrcDirTechMonCapEax1)

Read-only. Reset: Fixed,0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000F_EAX_x01

Bits	Description
31:9	Reserved.
8	**OverflowBit**. Read-only. Reset: Fixed,0. 1= Indicates Core::X86::Msr::QM_CTR bit 61 is an overflow bit.

7:0	**CounterSize**. Read-only. Reset: Fixed,00h. Encode counter width offset from bit 24.
	ValidValues:

Value	Description
00h	Family/Model/Stepping should be used to determine counter size; 44-bits for this product.
26h-01h	<Value>+24-bit counters.
FFh-27h	Reserved.

CPUID_Fn0000000F_EBX_x01 [Resource Director Technology L3 Monitor Capability] (Core::X86::Cpuid::RsrcDirTechMonCapEbx1)

Read-only. Reset: Fixed,0000_0040h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000F_EBX_x01

Bits	Description
31:0	**ConverFactor**. Read-only. Reset: Fixed,0000_0040h. Conversion Factor.

CPUID_Fn0000000F_ECX_x01 [Resource Director Technology L3 Monitor Capability] (Core::X86::Cpuid::RsrcDirTechMonCapEcx1)

Read-only. Reset: Fixed,0000_00FFh.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000F_ECX_x01

Bits	Description
31:0	**RmidMaxRange**. Read-only. Reset: Fixed,0000_00FFh. RMID Maximum Range of this resourse.

CPUID_Fn0000000F_EDX_x01 [Resource Director Technology L3 Monitor Capability] (Core::X86::Cpuid::RsrcDirTechMonCapEdx1)

Read-only. Reset: Fixed,0000_0007h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000F_EDX_x01

Bits	Description
31:3	Reserved.
2	**L3CacheLocalBndwdthMon**. Read-only. Reset: Fixed,1. L3 Local Bandwidth monitoring.
1	**L3CacheTotalBndwdthMon**. Read-only. Reset: Fixed,1. L3 Total Bandwidth monitoring.
0	**L3CacheOccpncyMon**. Read-only. Reset: Fixed,1. L3 occupancy monitoring.

CPUID_Fn00000010_EAX_x00 [Resource Director Technology Allocation Enumeration] (Core::X86::Cpuid::RsrcDirTechAllocEnumEax0)

Read-only. Reset: Fixed,0000_0000h. Enable: (Core::X86::Cpuid::RsrcDirTechAllocEnumEbx0 > 0).

Software determines the presence of CPUID Fn0000_0010 if (CPUID Fn0000_0010_EBX_x0[31:0] != 0). Software reads CPUID Fn0000_0010_E[D,C,B,A]X for ascending values of ECX until (CPUID Fn0000_0010_EBX[LogProcAtThisLevel] == 0).

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn00000010_EAX_x00

Bits	Description
31:0	Reserved.

CPUID_Fn00000010_EBX_x00 [Resource Director Technology Allocation Enumeration] (Core::X86::Cpuid::RsrcDirTechAllocEnumEbx0)

Read-only. Reset: 0000_0002h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn00000010_EBX_x00

Bits	Description
31:3	Reserved.
2	**L2CacheAllocTech**. Read-only. Reset: 0. L2 Cache Allocation Technology.
1	**L3CacheAllocTech**. Read-only. Reset: 1. L3 Cache Allocation Technology.
0	Reserved.

CPUID_Fn00000010_ECX_x00 [Resource Director Technology Allocation Enumeration] (Core::X86::Cpuid::RsrcDirTechAllocEnumEcx0)

Read-only. Reset: Fixed,0000_0000h. Enable: (Core::X86::Cpuid::RsrcDirTechAllocEnumEbx0 > 0).

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn00000010_ECX_x00

Bits	Description
31:0	Reserved.

CPUID_Fn00000010_EDX_x00 [Resource Director Technology Allocation Enumeration] (Core::X86::Cpuid::RsrcDirTechAllocEnumEdx0)

Read-only. Reset: Fixed,0000_0000h. Enable: (Core::X86::Cpuid::RsrcDirTechAllocEnumEbx0 > 0).

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn00000010_EDX_x00

Bits	Description
31:0	Reserved.

CPUID_Fn00000010_EAX_x01 [Resource Director Technology L3 Allocation Enumeration] (Core::X86::Cpuid::RsrcDirTechAllocEnumEax1)

Read-only. Enable: (Core::X86::Cpuid::RsrcDirTechAllocEnumEbx1 > 0).

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn00000010_EAX_x01

Bits	Description
31:5	Reserved.
4:0	**CapacityMask**. Read-only. Reset: Fixed,0Fh. Capacity bitmask length.

CPUID_Fn00000010_EBX_x01 [Resource Director Technology L3 Allocation Enumeration] (Core::X86::Cpuid::RsrcDirTechAllocEnumEbx1)

Read-only. Reset: 0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn00000010_EBX_x01

Bits	Description
31:0	**AllocUnits**. Read-only. Reset: 0000_0000h. Allocation Units.

CPUID_Fn00000010_ECX_x01 [Resource Director Technology L3 Allocation Enumeration] (Core::X86::Cpuid::RsrcDirTechAllocEnumEcx1)

Read-only. Reset: Fixed,0000_0004h. Enable: (Core::X86::Cpuid::RsrcDirTechAllocEnumEbx1 > 0).

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn00000010_ECX_x01

Bits	Description
31:3	Reserved.
2	**CDP**. Read-only. Reset: Fixed,1. Code and data prioritization.
1:0	Reserved.

CPUID_Fn00000010_EDX_x01 [Resource Director Technology L3 Allocation Enumeration] (Core::X86::Cpuid::RsrcDirTechAllocEnumEdx1)

Read-only. Reset: Fixed,0000_000Fh. Enable: (Core::X86::Cpuid::RsrcDirTechAllocEnumEbx1 > 0).

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn00000010_EDX_x01

Bits	Description
31:16	Reserved.
15:0	**HCS**. Read-only. Reset: Fixed,000Fh. Highest COS supported.

CPUID_Fn80000000_EAX [Largest Extended Function Number] (Core::X86::Cpuid::LargExtFuncNum)

Read-only. Reset: Fixed,8000_0023h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000000_EAX

Bits	Description
31:0	**LFuncExt**: **largest extended function**. Read-only. Reset: Fixed,8000_0023h. The largest CPUID extended function input value supported by the processor implementation.

CPUID_Fn80000000_EBX [Processor Vendor (ASCII Bytes [3:0])] (Core::X86::Cpuid::ProcVendExtEbx)

Read-only. Reset: Fixed,6874_7541h.

Core::X86::Cpuid::ProcVendEbx and Core::X86::Cpuid::ProcVendExtEbx return the same value.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000000_EBX

Bits	Description
31:0	**Vendor**. Read-only. Reset: Fixed,6874_7541h. ASCII Bytes [3:0] ("h t u A") of the string "AuthenticAMD".

CPUID_Fn80000000_ECX [Processor Vendor (ASCII Bytes [11:8])] (Core::X86::Cpuid::ProcVendExtEcx)

Read-only. Reset: Fixed,444D_4163h.

Core::X86::Cpuid::ProcVendEcx and Core::X86::Cpuid::ProcVendExtEcx return the same value.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000000_ECX

Bits	Description
31:0	**Vendor**. Read-only. Reset: Fixed,444D_4163h. ASCII Bytes [11:8] ("D M A c") of the string "AuthenticAMD".

CPUID_Fn80000000_EDX [Processor Vendor (ASCII Bytes [7:4])] (Core::X86::Cpuid::ProcVendExtEdx)

Read-only. Reset: Fixed,6974_6E65h.

Core::X86::Cpuid::ProcVendEdx and Core::X86::Cpuid::ProcVendExtEdx return the same value.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000000_EDX

Bits	Description
31:0	**Vendor**. Read-only. Reset: Fixed,6974_6E65h. ASCII Bytes [7:4] ("i t n e") of the string "AuthenticAMD".

CPUID_Fn80000001_EAX [Family, Model, Stepping Identifiers] (Core::X86::Cpuid::FamModStepExt)

Read-only.

Core::X86::Cpuid::FamModStep and Core::X86::Cpuid::FamModStepExt return the same value. See Core::X86::Cpuid::FamModStep.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000001_EAX

Bits	Description
31:28	Reserved.
27:20	**ExtFamily**: **extended family**. Read-only. Reset: 0Ah. See Core::X86::Cpuid::FamModStep description of Family.
19:16	**ExtModel**: **extended model**. Read-only. Reset: 0h. See Core::X86::Cpuid::FamModStep description of ExtModel.
15:12	Reserved.
11:8	**BaseFamily**. Read-only. Reset: Fh. See Core::X86::Cpuid::FamModStep description of Family.
7:4	**BaseModel**. Read-only. Reset: Xh. Model numbers vary with product.
3:0	**Stepping**. Read-only. Reset: 1h. Processor stepping (revision) for a specific model.

CPUID_Fn80000001_EBX [BrandId Identifier] (Core::X86::Cpuid::BrandId)

Read-only.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000001_EBX

Bits	Description
31:28	**PkgType**: **package type**. Read-only. Reset: Xh. Specifies the package type.
27:0	Reserved.

CPUID_Fn80000001_ECX [Feature Identifiers] (Core::X86::Cpuid::FeatureExtIdEcx)

Read-only.

These values can be over-written by Core::X86::Msr::CPUID_ExtFeatures.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000001_ECX

Bits	Description
31	Reserved.
30	**AdMskExtn**: **address mask extension support for instruction breakpoint**. Read-only. Reset: Fixed,1. Indicates support for address mask extension (to 32 bits and to all 4 DRs) for instruction breakpoints.

29	**MwaitExtended**. Read-only. Reset: !Core::X86::Msr::HWCR[MonMwaitDis]. 1=MWAITX and MONITORX capability is supported.
28	**PerfCtrExtLLC**: **Last Level Cache performance counter extensions**. Read-only. Reset: Fixed,1. 1=Indicates support for Core::X86::Msr::ChL3PmcCfg and Core::X86::Msr::ChL3Pmc L3 performance counter extensions. L3 performance counter extensions support. See 2.1.17.5 [L3 Cache Performance Monitor Counters] and 2.1.17 [Performance Monitor Counters].
27	**PerfTsc**. Read-only. Reset: Fixed,0. Performance time-stamp counter supported.
26	**DataBreakpointExtension**. Read-only. Reset: Fixed,1. 1=Indicates data breakpoint support for Core::X86::Msr::DR0_ADDR_MASK, Core::X86::Msr::DR1_ADDR_MASK, Core::X86::Msr::DR2_ADDR_MASK and Core::X86::Msr::DR3_ADDR_MASK.
25	Reserved.
24	**PerfCtrExtDF**: **data fabric performance counter extensions support**. Read-only. Reset: Fixed,1. 1=Indicates support for Core::X86::Msr::DF_PERF_CTL and Core::X86::Msr::DF_PERF_CTR.
23	**PerfCtrExtCore**: **core performance counter extensions support**. Read-only. Reset: Fixed,1. 1=Indicates support for Core::X86::Msr::PERF_CTL0 - 5 and Core::X86::Msr::PERF_CTR. See See 2.1.17.4 [Core Performance Monitor Counters] and 2.1.17 [Performance Monitor Counters].
22	**TopologyExtensions**: **topology extensions support**. Read-only. Reset: Fixed,1. 1=Indicates support for Core::X86::Cpuid::CachePropEax0 and Core::X86::Cpuid::ExtApicId.
21:18	Reserved.
17	**TCE**. Read-only. Reset: Fixed,1. Translation cache extension.
16	**FMA4**. Read-only. Reset: Fixed,0. Four-operand FMA instruction support.
15	**LWP**. Read-only. Reset: Fixed,0. Lightweight profiling support.
14	Reserved.
13	**WDT**. Read-only. Reset: Fixed,1. Watchdog timer support.
12	**SKINIT**. Read-only. Reset: Fixed,1. SKINIT and STGI support.
11	**XOP**. Read-only. Reset: Fixed,0. Extended operation support.
10	**IBS**. Read-only. Reset: Fixed,1. Instruction Based Sampling.
9	**OSVW**. Read-only. Reset: Fixed,1. OS Visible Work-around support.
8	**ThreeDNowPrefetch**. Read-only. Reset: Fixed,1. Prefetch and PrefetchW instructions.
7	**MisAlignSse**. Read-only. Reset: Fixed,1. Misaligned SSE Mode.
6	**SSE4A**. Read-only. Reset: Fixed,1. EXTRQ, INSERTQ, MOVNTSS, and MOVNTSD instruction support.
5	**ABM**: **advanced bit manipulation**. Read-only. Reset: Fixed,1. LZCNT instruction support.
4	**AltMovCr8**. Read-only. Reset: Fixed,1. LOCK MOV CR0 means MOV CR8.
3	**ExtApicSpace**. Read-only. Reset: Fixed,1. Extended APIC register space.
2	**SVM**: **Secure Virtual Mode feature**. Read-only. Reset: Fixed,1. Indicates support for: VMRUN, VMLOAD, VMSAVE, CLGI, VMMCALL, and INVLPGA.
1	**CmpLegacy**. Read-only. Reset: Fixed,(Core::X86::Cpuid::SizeId[NC] > 0). 0=Single core product (Core::X86::Cpuid::SizeId[NC] == 0). 1=Multi core product (Core::X86::Cpuid::SizeId[NC] !=0). Core multi-processing legacy mode.
0	**LahfSahf**. Read-only. Reset: Fixed,1. LAHF and SAHF instruction support in 64-bit mode.

CPUID_Fn80000001_EDX [Feature Identifiers] (Core::X86::Cpuid::FeatureExtIdEdx)

Read-only.

These values can be over-written by Core::X86::Msr::CPUID_ExtFeatures.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000001_EDX

Bits	Description
31	**ThreeDNow**. Read-only. Reset: Fixed,0. 3DNow! instructions.
30	**ThreeDNowExt**. Read-only. Reset: Fixed,0. AMD extensions to 3DNow! instructions.
29	**LM**. Read-only. Reset: Fixed,1. Long Mode.
28	Reserved.

27	**RDTSCP**. Read-only. Reset: Fixed,1. RDTSCP instruction.
26	**Page1GB**. Read-only. Reset: Fixed,1. 1-GB large page support.
25	**FFXSR**. Read-only. Reset: Fixed,1. FXSAVE and FXRSTOR instruction optimizations.
24	**FXSR**. Read-only. Reset: Fixed,1. FXSAVE and FXRSTOR instructions.
23	**MMX**. Read-only. Reset: Fixed,1. MMX instructions.
22	**MmxExt**. Read-only. Reset: Fixed,1. AMD extensions to MMX instructions.
21	Reserved.
20	**NX**. Read-only. Reset: Fixed,1. No-execute page protection.
19:18	Reserved.
17	**PSE36**. Read-only. Reset: Fixed,1. Page-size extensions.
16	**PAT**. Read-only. Reset: Fixed,1. Page attribute table.
15	**CMOV**. Read-only. Reset: Fixed,1. Conditional move instructions, CMOV, FCOMI, FCMOV.
14	**MCA**. Read-only. Reset: Fixed,1. Machine check architecture, MCG_CAP.
13	**PGE**. Read-only. Reset: Fixed,1. Page global extension, CR4.PGE.
12	**MTRR**. Read-only. Reset: Fixed,1. Memory-type range registers.
11	**SysCallSysRet**. Read-only. Reset: Fixed,1. SYSCALL and SYSRET instructions.
10	Reserved.
9	**APIC**: **advanced programmable interrupt controller (APIC) exists and is enabled**. Read-only. Reset: X. Reset is Core::X86::Msr::APIC_BAR[ApicEn].
8	**CMPXCHG8B**. Read-only. Reset: Fixed,1. CMPXCHG8B instruction.
7	**MCE**. Read-only. Reset: Fixed,1. Machine Check Exception, CR4.MCE.
6	**PAE**. Read-only. Reset: Fixed,1. Physical-address extensions (PAE).
5	**MSR**. Read-only. Reset: Fixed,1. Model-specific registers (MSRs), with RDMSR and WRMSR instructions.
4	**TSC**. Read-only. Reset: Fixed,1. Time stamp counter, RDTSC/RDTSCP instructions, CR4.TSD.
3	**PSE**. Read-only. Reset: Fixed,1. Page-size extensions (4 MB pages).
2	**DE**. Read-only. Reset: Fixed,1. Debugging extensions, IO breakpoints, CR4.DE.
1	**VME**. Read-only. Reset: Fixed,1. Virtual-mode enhancements.
0	**FPU**. Read-only. Reset: Fixed,1. x87 floating point unit on-chip.

CPUID_Fn80000002_EAX [Processor Name String Identifier (Bytes [3:0])] (Core::X86::Cpuid::ProcNameStr0Eax)

Read-only.

Is an alias of Core::X86::Msr::ProcNameString_n0.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000002_EAX

Bits	Description
31:24	**ProcNameByte3**. Read-only. Reset: Core::X86::Msr::ProcNameString_n0[CpuNameString3]. Processor name, byte3.
23:16	**ProcNameByte2**. Read-only. Reset: Core::X86::Msr::ProcNameString_n0[CpuNameString2]. Processor name, byte2.
15:8	**ProcNameByte1**. Read-only. Reset: Core::X86::Msr::ProcNameString_n0[CpuNameString1]. Processor name, byte1.
7:0	**ProcNameByte0**. Read-only. Reset: Core::X86::Msr::ProcNameString_n0[CpuNameString0]. Processor name, byte0.

CPUID_Fn80000002_EBX [Processor Name String Identifier (Bytes [7:4])] (Core::X86::Cpuid::ProcNameStr0Ebx)

Read-only.

Is an alias of Core::X86::Msr::ProcNameString_n0.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000002_EBX

Bits	Description

31:24	**ProcNameByte7**. Read-only. Reset: Core::X86::Msr::ProcNameString_n0[CpuNameString7]. Processor name, byte 7.
23:16	**ProcNameByte6**. Read-only. Reset: Core::X86::Msr::ProcNameString_n0[CpuNameString6]. Processor name, byte 6.
15:8	**ProcNameByte5**. Read-only. Reset: Core::X86::Msr::ProcNameString_n0[CpuNameString5]. Processor name, byte 5.
7:0	**ProcNameByte4**. Read-only. Reset: Core::X86::Msr::ProcNameString_n0[CpuNameString4]. Processor name, byte 4.

CPUID_Fn80000002_ECX [Processor Name String Identifier (Bytes [11:8])] (Core::X86::Cpuid::ProcNameStr0Ecx)

Read-only.

Is an alias of Core::X86::Msr::ProcNameString_n1.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000002_ECX

Bits	Description
31:24	**ProcNameByte11**. Read-only. Reset: Core::X86::Msr::ProcNameString_n1[CpuNameString3]. Processor name, byte 11.
23:16	**ProcNameByte10**. Read-only. Reset: Core::X86::Msr::ProcNameString_n1[CpuNameString2]. Processor name, byte 10.
15:8	**ProcNameByte9**. Read-only. Reset: Core::X86::Msr::ProcNameString_n1[CpuNameString1]. Processor name, byte 9.
7:0	**ProcNameByte8**. Read-only. Reset: Core::X86::Msr::ProcNameString_n1[CpuNameString0]. Processor name, byte 8.

CPUID_Fn80000002_EDX [Processor Name String Identifier (Bytes [15:12])] (Core::X86::Cpuid::ProcNameStr0Edx)

Read-only.

Is an alias of Core::X86::Msr::ProcNameString_n1.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000002_EDX

Bits	Description
31:24	**ProcNameByte15**. Read-only. Reset: Core::X86::Msr::ProcNameString_n1[CpuNameString7]. Processor name, byte 15.
23:16	**ProcNameByte14**. Read-only. Reset: Core::X86::Msr::ProcNameString_n1[CpuNameString6]. Processor name, byte 14.
15:8	**ProcNameByte13**. Read-only. Reset: Core::X86::Msr::ProcNameString_n1[CpuNameString5]. Processor name, byte 13.
7:0	**ProcNameByte12**. Read-only. Reset: Core::X86::Msr::ProcNameString_n1[CpuNameString4]. Processor name, byte 12.

CPUID_Fn80000003_EAX [Processor Name String Identifier (Bytes [19:16])] (Core::X86::Cpuid::ProcNameStr1Eax)

Read-only.

Is an alias of Core::X86::Msr::ProcNameString_n2.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000003_EAX

Bits	Description
31:24	**ProcNameByte19**. Read-only. Reset: Core::X86::Msr::ProcNameString_n2[CpuNameString3]. Processor name, byte 19.
23:16	**ProcNameByte18**. Read-only. Reset: Core::X86::Msr::ProcNameString_n2[CpuNameString2]. Processor name, byte 18.
15:8	**ProcNameByte17**. Read-only. Reset: Core::X86::Msr::ProcNameString_n2[CpuNameString1]. Processor name, byte 17.
7:0	**ProcNameByte16**. Read-only. Reset: Core::X86::Msr::ProcNameString_n2[CpuNameString0]. Processor name,

	byte 16.

CPUID_Fn80000003_EBX [Processor Name String Identifier (Bytes [23:20])] (Core::X86::Cpuid::ProcNameStr1Ebx)

Read-only.

Is an alias of Core::X86::Msr::ProcNameString_n2.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000003_EBX

Bits	Description
31:24	**ProcNameByte23**. Read-only. Reset: Core::X86::Msr::ProcNameString_n2[CpuNameString7]. Processor name, byte 23.
23:16	**ProcNameByte22**. Read-only. Reset: Core::X86::Msr::ProcNameString_n2[CpuNameString6]. Processor name, byte 22.
15:8	**ProcNameByte21**. Read-only. Reset: Core::X86::Msr::ProcNameString_n2[CpuNameString5]. Processor name, byte 21.
7:0	**ProcNameByte20**. Read-only. Reset: Core::X86::Msr::ProcNameString_n2[CpuNameString4]. Processor name, byte 20.

CPUID_Fn80000003_ECX [Processor Name String Identifier (Bytes [27:24])] (Core::X86::Cpuid::ProcNameStr1Ecx)

Read-only.

Is an alias of Core::X86::Msr::ProcNameString_n3.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000003_ECX

Bits	Description
31:24	**ProcNameByte27**. Read-only. Reset: Core::X86::Msr::ProcNameString_n3[CpuNameString3]. Processor name, byte 27.
23:16	**ProcNameByte26**. Read-only. Reset: Core::X86::Msr::ProcNameString_n3[CpuNameString2]. Processor name, byte 26.
15:8	**ProcNameByte25**. Read-only. Reset: Core::X86::Msr::ProcNameString_n3[CpuNameString1]. Processor name, byte 25.
7:0	**ProcNameByte24**. Read-only. Reset: Core::X86::Msr::ProcNameString_n3[CpuNameString0]. Processor name, byte 24.

CPUID_Fn80000003_EDX [Processor Name String Identifier (Bytes [31:28])] (Core::X86::Cpuid::ProcNameStr1Edx)

Read-only.

Is an alias of Core::X86::Msr::ProcNameString_n3.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000003_EDX

Bits	Description
31:24	**ProcNameByte31**. Read-only. Reset: Core::X86::Msr::ProcNameString_n3[CpuNameString7]. Processor name, byte 31.
23:16	**ProcNameByte30**. Read-only. Reset: Core::X86::Msr::ProcNameString_n3[CpuNameString6]. Processor name, byte 30.
15:8	**ProcNameByte29**. Read-only. Reset: Core::X86::Msr::ProcNameString_n3[CpuNameString5]. Processor name, byte 29.
7:0	**ProcNameByte28**. Read-only. Reset: Core::X86::Msr::ProcNameString_n3[CpuNameString4]. Processor name, byte 28.

CPUID_Fn80000004_EAX [Processor Name String Identifier (Bytes [35:32])] (Core::X86::Cpuid::ProcNameStr2Eax)

Read-only.

Is an alias of Core::X86::Msr::ProcNameString_n4.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000004_EAX

Bits	Description
31:24	**ProcNameByte35**. Read-only. Reset: Core::X86::Msr::ProcNameString_n4[CpuNameString3]. Processor name, byte 35.
23:16	**ProcNameByte34**. Read-only. Reset: Core::X86::Msr::ProcNameString_n4[CpuNameString2]. Processor name, byte 34.
15:8	**ProcNameByte33**. Read-only. Reset: Core::X86::Msr::ProcNameString_n4[CpuNameString1]. Processor name, byte 33.
7:0	**ProcNameByte32**. Read-only. Reset: Core::X86::Msr::ProcNameString_n4[CpuNameString0]. Processor name, byte 32.

CPUID_Fn80000004_EBX [Processor Name String Identifier (Bytes [39:36])] (Core::X86::Cpuid::ProcNameStr2Ebx)

Read-only.

Is an alias of Core::X86::Msr::ProcNameString_n4.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000004_EBX

Bits	Description
31:24	**ProcNameByte39**. Read-only. Reset: Core::X86::Msr::ProcNameString_n4[CpuNameString7]. Processor name, byte 39.
23:16	**ProcNameByte38**. Read-only. Reset: Core::X86::Msr::ProcNameString_n4[CpuNameString6]. Processor name, byte 38.
15:8	**ProcNameByte37**. Read-only. Reset: Core::X86::Msr::ProcNameString_n4[CpuNameString5]. Processor name, byte 37.
7:0	**ProcNameByte36**. Read-only. Reset: Core::X86::Msr::ProcNameString_n4[CpuNameString4]. Processor name, byte 36.

CPUID_Fn80000004_ECX [Processor Name String Identifier (Bytes [43:40])] (Core::X86::Cpuid::ProcNameStr2Ecx)

Read-only.

Is an alias of Core::X86::Msr::ProcNameString_n5.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000004_ECX

Bits	Description
31:24	**ProcNameByte43**. Read-only. Reset: Core::X86::Msr::ProcNameString_n5[CpuNameString3]. Processor name, byte 43.
23:16	**ProcNameByte42**. Read-only. Reset: Core::X86::Msr::ProcNameString_n5[CpuNameString2]. Processor name, byte 42.
15:8	**ProcNameByte41**. Read-only. Reset: Core::X86::Msr::ProcNameString_n5[CpuNameString1]. Processor name, byte 41.
7:0	**ProcNameByte40**. Read-only. Reset: Core::X86::Msr::ProcNameString_n5[CpuNameString0]. Processor name, byte 40.

CPUID_Fn80000004_EDX [Processor Name String Identifier (Bytes [47:44])] (Core::X86::Cpuid::ProcNameStr2Edx)

Read-only.

Is an alias of Core::X86::Msr::ProcNameString_n5.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000004_EDX

Bits	Description
31:24	**ProcNameByte47**. Read-only. Reset: Core::X86::Msr::ProcNameString_n5[CpuNameString7]. Processor name, byte 47.
23:16	**ProcNameByte46**. Read-only. Reset: Core::X86::Msr::ProcNameString_n5[CpuNameString6]. Processor name, byte 46.
15:8	**ProcNameByte45**. Read-only. Reset: Core::X86::Msr::ProcNameString_n5[CpuNameString5]. Processor name, byte 45.

7:0	**ProcNameByte44**. Read-only. Reset: Core::X86::Msr::ProcNameString_n5[CpuNameString4]. Processor name, byte 44.

CPUID_Fn80000005_EAX [L1 TLB 2M/4M Identifiers] (Core::X86::Cpuid::L1Tlb2M4M)

Read-only.

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

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000005_EAX

Bits	Description
31:24	**L1DTlb2and4MAssoc**: **data TLB associativity for 2 MB and 4 MB pages**. Read-only. Reset: Fixed,FFh. See Core::X86::Cpuid::L1DcId[L1DcAssoc].
23:16	**L1DTlb2and4MSize**: **data TLB number of entries for 2 MB and 4 MB pages**. Read-only. Reset: Fixed,64. 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.
15:8	**L1ITlb2and4MAssoc**: **instruction TLB associativity for 2 MB and 4 MB pages**. Read-only. Reset: Fixed,FFh. See Core::X86::Cpuid::L1DcId[L1DcAssoc].
7:0	**L1ITlb2and4MSize**: **instruction TLB number of entries for 2 MB and 4 MB pages**. Read-only. Reset: Fixed,64. 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.

CPUID_Fn80000005_EBX [L1 TLB 4K Identifiers] (Core::X86::Cpuid::L1Tlb4K)

Read-only.

See Core::X86::Cpuid::L1Tlb2M4M.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000005_EBX

Bits	Description
31:24	**L1DTlb4KAssoc**. Read-only. Reset: Fixed,FFh. Data TLB associativity for 4 KB pages. See Core::X86::Cpuid::L1DcId[L1DcAssoc].
23:16	**L1DTlb4KSize**. Read-only. Reset: Fixed,64. Data TLB number of entries for 4 KB pages.
15:8	**L1ITlb4KAssoc**. Read-only. Reset: Fixed,FFh. Instruction TLB associativity for 4 KB pages. See Core::X86::Cpuid::L1DcId[L1DcAssoc].
7:0	**L1ITlb4KSize**. Read-only. Reset: Fixed,64. Instruction TLB number of entries for 4 KB pages.

CPUID_Fn80000005_ECX [L1 Data Cache Identifiers] (Core::X86::Cpuid::L1DcId)

Read-only.

This function provides first level cache characteristics for each core.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000005_ECX

<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>31:24</td><td>L1DcSize. Read-only. Reset: Fixed,32. L1 data cache size in KB.</td></tr>
<tr><td rowspan="2">23:16</td><td>L1DcAssoc. Read-only. Reset: Fixed,8. L1 data cache associativity.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>00h</td><td>Reserved</td></tr>
<tr><td>01h</td><td>1 way (direct mapped)</td></tr>
<tr><td>02h</td><td>2 way</td></tr>
<tr><td>03h</td><td>3 way</td></tr>
<tr><td>FEh-04h</td><td><Value> way</td></tr>
<tr><td>FFh</td><td>Fully associative</td></tr>
</table></td></tr>
<tr><td>15:8</td><td>L1DcLinesPerTag. Read-only. Reset: Fixed,01h. L1 data cache lines per tag.</td></tr>
<tr><td>7:0</td><td>L1DcLineSize. Read-only. Reset: Fixed,64. L1 data cache line size in bytes.</td></tr>
</table>

CPUID_Fn80000005_EDX [L1 Instruction Cache Identifiers] (Core::X86::Cpuid::L1IcId)

Read-only.

This function provides first level cache characteristics for each core.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000005_EDX

Bits	Description
31:24	**L1IcSize**. Read-only. Reset: Fixed,32. L1 instruction cache size KB.
23:16	**L1IcAssoc**. Read-only. Reset: Fixed,8. L1 instruction cache associativity.

ValidValues:

Value	Description
00h	Reserved
01h	1 way (direct mapped)
02h	2 way
03h	3 way
04h	4 way
FEh-05h	<Value> way
FFh	Fully associative

Bits	Description
15:8	**L1IcLinesPerTag**. Read-only. Reset: Fixed,01h. L1 instruction cache lines per tag.
7:0	**L1IcLineSize**. Read-only. Reset: Fixed,64. L1 instruction cache line size in bytes.

CPUID_Fn80000006_EAX [L2 TLB 2M/4M Identifiers] (Core::X86::Cpuid::L2Tlb2M4M)

Read-only.

This function provides the processor's second level cache and TLB characteristics for each core.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000006_EAX

Bits	Description
31:28	**L2DTlb2and4MAssoc**: **L2 data TLB associativity for 2 MB and 4 MB pages**. Read-only. Reset: Xh.
27:16	**L2DTlb2and4MSize**: **L2 data TLB number of entries for 2 MB and 4 MB pages**. Read-only. Reset: Fixed,2048. 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.
15:12	**L2ITlb2and4MAssoc**: **L2 instruction TLB associativity for 2 MB and 4 MB pages**. Read-only. Reset: Fixed,2.

ValidValues:

Value	Description
1h-0h	Reserved.
2h	2 ways
Fh-3h	Reserved.

Bits	Description
11:0	**L2ITlb2and4MSize**: **L2 instruction TLB number of entries for 2 MB and 4 MB pages**. Read-only. Reset: Fixed,512. 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.

CPUID_Fn80000006_EBX [L2 TLB 4K Identifiers] (Core::X86::Cpuid::L2Tlb4K)

Read-only.

This function provides the processor's second level cache and TLB characteristics for each core.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000006_EBX

Bits	Description
31:28	**L2DTlb4KAssoc**. Read-only. Reset: 6h. L2 data TLB associativity for 4 KB pages.

ValidValues:

Value	Description
5h-0h	Reserved.
6h	8 ways
Fh-7h	Reserved.

Bits	Description
27:16	**L2DTlb4KSize**. Read-only. Reset: Fixed,2048. L2 data TLB number of entries for 4 KB pages.

15:12	**L2ITlb4KAssoc**. Read-only. Reset: Fixed,4. L2 instruction TLB associativity for 4 KB pages.
	ValidValues: 3h-0h: Reserved. 4h: 4 ways Fh-5h: Reserved.
11:0	**L2ITlb4KSize**. Read-only. Reset: Fixed,512. L2 instruction TLB number of entries for 4 KB pages.

CPUID_Fn80000006_ECX [L2 Cache Identifiers] (Core::X86::Cpuid::L2CacheId)

Read-only.

This function provides second level cache characteristics for each core.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000006_ECX

Bits	Description
31:16	**L2Size**. Read-only. Reset: Fixed,0200h. L2 cache size in KB.
	ValidValues:

Value	Description
00FFh-0000h	Reserved.
0100h	256 KB
01FFh-0101h	Reserved.
0200h	512 KB
03FFh-0201h	Reserved.
0400h	1 MB
07FFh-0401h	Reserved.
0800h	2 MB
FFFFh-0801h	Reserved.

Bits	Description
15:12	**L2Assoc**. Read-only. Reset: Fixed,6. L2 cache associativity.
	ValidValues:

Value	Description
0h	Disabled.
1h	1 way (direct mapped)
2h	2 ways
3h	Reserved.
4h	4 ways
5h	Reserved.
6h	8 ways
7h	Reserved.
8h	16 ways
9h	Reserved.
Ah	32 ways
Bh	48 ways
Ch	64 ways
Dh	96 ways
Eh	128 ways
Fh	Fully associative

11:8	**L2LinesPerTag**. Read-only. Reset: Fixed,1h. L2 cache lines per tag.
7:0	**L2LineSize**. Read-only. Reset: Fixed,64. L2 cache line size in bytes.

CPUID_Fn80000006_EDX [L3 Cache Identifiers] (Core::X86::Cpuid::L3CacheId)

Read-only.

This function provides third level cache characteristics shared by all cores of a processor.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000006_EDX

<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>31:18</td><td>L3Size: L3 cache size. Read-only. Reset: XXXXh. The L3 cache size in 512 KB units.
ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0000h</td><td>Disabled.</td></tr>
<tr><td>3FFFh-0001h</td><td>(<Value> *0.5) MB</td></tr>
</table></td></tr>
<tr><td>17:16</td><td>Reserved.</td></tr>
<tr><td>15:12</td><td>L3Assoc. Read-only. Reset: Fixed,9h. There are insufficient available encodings to represent all possible L3 associativities. Please refer to Core::X86::Cpuid::CachePropEbx3[CacheNumWays].
ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>8h-0h</td><td>Reserved.</td></tr>
<tr><td>9h</td><td>Invalid, not reported here.</td></tr>
<tr><td>Fh-Ah</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>11:8</td><td>L3LinesPerTag. Read-only. Reset: Fixed,1h. L3 cache lines per tag.</td></tr>
<tr><td>7:0</td><td>L3LineSize. Read-only. Reset: Fixed,64. L3 cache line size in bytes.</td></tr>
</table>

CPUID_Fn80000007_EAX [Reserved] (Core::X86::Cpuid::ProcFeedbackCap)

Read-only. Reset: Fixed,0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000007_EAX

Bits	Description
31:0	Reserved.

CPUID_Fn80000007_EBX [RAS Capabilities] (Core::X86::Cpuid::RasCap)

Read-only.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000007_EBX

Bits	Description
31:4	Reserved.
3	**ScalableMca**. Read-only. Reset: Fixed,1. 0=Scalable MCA is not supported. 1=Scalable MCA is supported. See 3.1.1.2 [Machine Check Architecture Extensions] and MCA_CONFIG[McaX] for the respective bank.
2	**HWA**. Read-only. Reset: Fixed,0. Hardware assert supported.
1	**SUCCOR**: **Software uncorrectable error containment and recovery capability**. Read-only. Reset: X. The processor supports software containment of uncorrectable errors through context synchronizing data poisoning and deferred error interrupts; MSR Core::X86::Msr::McaIntrCfg, MCA_STATUS[Deferred] and MCA_STATUS[Poison] exist.
0	**McaOverflowRecov**: **MCA overflow recovery support**. Read-only. Reset: Fixed,1. 0=MCA overflow conditions require software to shutdown the system. 1=MCA overflow conditions (MCi_STATUS[Overflow] == 1) are not fatal; software may safely ignore such conditions. See 3.1 [Machine Check Architecture].

CPUID_Fn80000007_ECX [Advanced Power Management Information] (Core::X86::Cpuid::ApmInfoEcx)

Read-only. Reset: Fixed,0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000007_ECX

Bits	Description

31:0	**CpuPwrSampleTimeRatio**. Read-only. Reset: Fixed,0000_0000h. Specifies the ratio of the compute unit power accumulator sample period to the TSC counter period.

CPUID_Fn80000007_EDX [Advanced Power Management Information] (Core::X86::Cpuid::ApmInfoEdx)

Read-only.

This function provides advanced power management feature identifiers.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000007_EDX

Bits	Description
31:15	Reserved.
14	**RAPL**. Read-only. Reset: Fixed,1. Running average power limit.
13	**ConnectedStandby**. Read-only. Reset: Fixed,1. Connected Standby.
12	**ProcPowerReporting**. Read-only. Reset: Fixed,0. Core power reporting interface supported.
11	**ProcFeedbackInterface**: **processor feedback interface**. Read-only. Reset: Fixed,0. 1=Indicates support for processor feedback interface; Core::X86::Cpuid::ProcFeedbackCap.
10	**EffFreqRO**: **read-only effective frequency interface**. Read-only. Reset: Fixed,1. Indicates presence of Core::X86::Msr::MPerfReadOnly and Core::X86::Msr::APerfReadOnly.
9	**CPB**: **core performance boost**. Read-only. Reset: X. 1=Indicates presence of Core::X86::Msr::HWCR[CpbDis] and support for core performance boost.
8	**TscInvariant**: **TSC invariant**. Read-only. Reset: Fixed,1. The TSC rate is invariant.
7	**HwPstate**: **hardware P-state control**. Read-only. Reset: Fixed,1. Core::X86::Msr::PStateCurLim, Core::X86::Msr::PStateCtl and Core::X86::Msr::PStateStat exist.
6	**OneHundredMHzSteps**. Read-only. Reset: Fixed,0. 100 MHz multiplier Control.
5	Reserved.
4	**TM**. Read-only. Reset: Fixed,1. Hardware thermal control (HTC).
3	**TTP**. Read-only. Reset: Fixed,1. THERMTRIP.
2:1	Reserved.
0	**TS**. Read-only. Reset: Fixed,1. Temperature sensor.

CPUID_Fn80000008_EAX [Long Mode Address Size Identifiers] (Core::X86::Cpuid::LongModeInfo)

Read-only. Reset: Fixed,0000_3030h.

This provides information about the maximum physical and linear address width supported by the processor.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000008_EAX

Bits	Description
31:24	Reserved.
23:16	**GuestPhysAddrSize**. Read-only. Reset: Fixed,00h. Maximum guest physical byte address size in bits.
	ValidValues:
15:8	**LinAddrSize**. Read-only. Reset: Fixed,30h. Maximum linear byte address size in bits.
7:0	**PhysAddrSize**. Read-only. Reset: Fixed,30h. Maximum physical byte address size in bits.

ValidValues for GuestPhysAddrSize (23:16):

Value	Description
00h	The maximum guest physical address size defined by PhysAddrSize.
FFh-01h	The maximum guest physical address size defined by GuestPhysAddrSize.

CPUID_Fn80000008_EBX [Extended Feature Extensions ID EBX] (Core::X86::Cpuid::FeatureExtIdEbx)

Read-only.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000008_EBX

Bits	Description
31	**BranchSample**. Read-only. Reset: Fixed,1. 1=Branch sampling feature supported. See Core::X86::Msr::SAMP_BR_FROM, Core::X86::Msr::SAMP_BR_TO and Core::X86::Msr::DbgExtnCfg.
30:29	Reserved.
28	**PSFD**. Read-only. Reset: Fixed,1. Predictive Store Forward Disable. See Core::X86::Msr::SPEC_CTRL[PSFD].

27	**CPPC**. Read-only. Reset: 0. Collaborative Processor Performance Control.
26:25	Reserved.
24	**SSBD**: **Speculative Store Bypass Disable**. Read-only. Reset: Fixed,1.
23	**PPIN**: **PPIN support**. Read-only. Reset: X. 0=PPIN capability is not supported; Core::X86::Msr::PPIN_CTL and Core::X86::Msr::PPIN are treated as RAZ. 1=Indicates that Protected Processor Inventory Number (PPIN) capability can be enabled for privileged system inventory agent to read PPIN from Core::X86::Msr::PPIN. Protected Processor Inventory Number support.
22	Reserved.
21	**TlbFlushNested**. Read-only. Reset: 1. Flushing includes all nested translations for guest. Allows setting RAX[5] for INVLPGB.
20	**EferLmsleUnsupported**. Read-only. Reset: Fixed,1. 1=Core::X86::Msr::EFER[LMSLE] is not supported, and MBZ.
19	**IbrsProvidesSameModeProtection**. Read-only. Reset: 1. IBRS provides Same Mode Protection.
18	**IbrsPreferred**. Read-only. Reset: 1. 1=IBRS is preferred over software solution.
17	**StibpAlwaysOn**. Read-only. Reset: 1. Single Thread Indirect Branch Prediction Mode has Enhanced Performance and May be left Always On.
16	Reserved.
15	**STIBP**. Read-only. Reset: 1. Single Thread Indirect Branch Prediction.
14	**IBRS**. Read-only. Reset: 1. Indirect Branch Restricted Speculation.
13	**INT_WBINVD**. Read-only. Reset: 1. Interruptible WBINVD,WBNOINVD. .
12	**IBPB**. Read-only. Reset: 1. Indirect Branch Prediction Barrier.
11:10	Reserved.
9	**WBNOINVD**. Read-only. Reset: 1. WBNOINVD writes all modified cache lines in the internal caches of the processor back to memory leaving the line valid (clean) in the internal caches.
8	**MCOMMIT**: **memory commit**. Read-only. Reset: 1. Memory commit instruction support. See Core::X86::Msr::EFER[MCOMMIT].
7	Reserved.
6	**MBE**. Read-only. Reset: Fixed,1. Memory Bandwidth Enforcement.
5	Reserved.
4	**RDPRU**: **read processor register at user level**. Read-only. Reset: Fixed,1. RDPRU instruction allows reading MPERF and APERF at user level.
3	**INVLPGB**. Read-only. Reset: Fixed,1. INVLPGB instruction broadcasts a TLB invalidate to all threads in the system.
2	**RstrFpErrPtrs**. Read-only. Reset: Fixed,1. 1=FXSAVE, XSAVE, FXSAVEOPT, XSAVEC, XSAVES always save error pointers and FXRSTOR, XRSTOR, XRSTORS always restore error pointers is supported.
1	**InstRetCntMsr**: **instructions retired count support**. Read-only. Reset: Fixed,1. 1=Core::X86::Msr::IRPerfCount supported.
0	**CLZERO**: **Clear Zero Instruction**. Read-only. Reset: Fixed,1. CLZERO instruction zero's out the 64 byte cache line specified in RAX. Note: CLZERO instruction operations are cache-line aligned and RAX[5:0] is ignored.

CPUID_Fn80000008_ECX [Size Identifiers] (Core::X86::Cpuid::SizeId)

Read-only.

This provides information about the number of threads supported by the processor.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000008_ECX

Bits	Description
31:18	Reserved.
17:16	**PerfTscSize**: **performance time-stamp counter size**. Read-only. Reset: Fixed,0h.
15:12	**ApicIdSize**: **APIC ID size**. Read-only. Reset: Xh. The number of bits in the initial Core::X86::Apic::ApicId[ApicId] value that indicate thread ID within a package.
11:8	Reserved.

7:0	**NC**: **number of threads - 1**. Read-only. Reset: XXh. The number of threads in the package is NC + 1 (e.g., if NC == 0, then there is one thread).

CPUID_Fn80000008_EDX [Feature Extended Size Edx] (Core::X86::Cpuid::FeatureExtSizeEdx)

Read-only. Reset: Fixed,0001_0007h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000008_EDX

Bits	Description
31:24	Reserved.
23:16	**RdpruMax**. Read-only. Reset: Fixed,01h. RDPRU Instruction max input supported.
15:0	**InvlpgbCountMax**. Read-only. Reset: Fixed,0007h. Maximum count for INVLPGB instruction.

CPUID_Fn8000000A_EAX [SVM Revision and Feature Identification] (Core::X86::Cpuid::SvmRevFeatIdEax)

Read-only. Reset: Fixed,0000_0001h. Enable: Core::X86::Cpuid::FeatureExtIdEcx[SVM].

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn8000000A_EAX

Bits	Description
31:8	Reserved.
7:0	**SvmRev**. Read-only. Reset: Fixed,01h. SVM revision.

CPUID_Fn8000000A_EBX [SVM Revision and Feature Identification] (Core::X86::Cpuid::SvmRevFeatIdEbx)

Read-only,Volatile. Reset: 0000_8000h. Enable: Core::X86::Cpuid::FeatureExtIdEcx[SVM].

This provides SVM revision and feature information.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn8000000A_EBX

Bits	Description
31:0	**NASID**: **number of address space identifiers (ASID)**. Read-only,Volatile. Reset: 0000_8000h.

CPUID_Fn8000000A_EDX [SVM Revision and Feature Identification] (Core::X86::Cpuid::SvmRevFeatIdEdx)

Read-only. Reset: Fixed,009B_B4FFh. Enable: Core::X86::Cpuid::FeatureExtIdEcx[SVM].

This provides SVM feature information.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn8000000A_EDX

Bits	Description
31:24	Reserved.
23	**HOST_MCE_OVERRIDE**. Read-only. Reset: Fixed,1. 1=If hCR4:MCE==1 and gCR4:MCE==0, machine check exceptions (#MC) in guest do not cause shutdown and are always intercepted.
22:21	Reserved.
20	**GuestSpecCtrl**. Read-only. Reset: Fixed,1. 1=Indicates support for Guest SPEC_CTRL.
19	**SupervisorShadowStack**. Read-only. Reset: Fixed,1. Supervisor Shadow Stack.
18	Reserved.
17	**GMET**. Read-only. Reset: Fixed,1. Guest Mode Execute Trap.
16	**vGIF**. Read-only. Reset: Fixed,1. Virtualized GIF.
15	**V_VMSAVE_VMLOAD**. Read-only. Reset: Fixed,1. Virtualized VMLOAD and VMSAVE.
14	Reserved.
13	**AVIC**: **AMD virtual interrupt controller**. Read-only. Reset: Fixed,1. 1=Support indicated for SVM mode virtualized interrupt controller; Indicates support for Core::X86::Msr::AvicDoorbell.
12	**PauseFilterThreshold**. Read-only. Reset: Fixed,1. PAUSE filter threshold.
11	Reserved.
10	**PauseFilter**. Read-only. Reset: Fixed,1. Pause intercept filter.
9:8	Reserved.
7	**DecodeAssists**. Read-only. Reset: Fixed,1. Decode assists.
6	**FlushByAsid**. Read-only. Reset: Fixed,1. Flush by ASID.
5	**VmcbClean**. Read-only. Reset: Fixed,1. VMCB clean bits.
4	**TscRateMsr**: **MSR based TSC rate control**. Read-only. Reset: Fixed,1. 1=Indicates support for TSC ratio

	Core::X86::Msr::TscRateMsr.
3	**NRIPS**. Read-only. Reset: Fixed,1. NRIP Save.
2	**SVML**. Read-only. Reset: Fixed,1. SVM lock.
1	**LbrVirt**. Read-only. Reset: Fixed,1. LBR virtualization.
0	**NP**. Read-only. Reset: Fixed,1. Nested Paging.

CPUID_Fn80000019_EAX [L1 TLB 1G Identifiers] (Core::X86::Cpuid::L1Tlb1G)

Read-only.

This function provides first level TLB characteristics for 1-GB pages.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000019_EAX

Bits	Description
31:28	**L1DTlb1GAssoc**: **L1 data TLB associativity for 1-GB pages**. Read-only. Reset: Fixed,Fh. See Core::X86::Cpuid::L2CacheId[L2Assoc].
27:16	**L1DTlb1GSize**. Read-only. Reset: Fixed,64. L1 data TLB number of entries for 1-GB pages.
15:12	**L1ITlb1GAssoc**. Read-only. Reset: Fixed,Fh. L1 instruction TLB associativity for 1-GB pages. See Core::X86::Cpuid::L2CacheId[L2Assoc].
11:0	**L1ITlb1GSize**. Read-only. Reset: Fixed,64. L1 instruction TLB number of entries for 1-GB pages.

CPUID_Fn80000019_EBX [L2 TLB 1G Identifiers] (Core::X86::Cpuid::L2Tlb1G)

Read-only. Reset: Fixed,F040_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000019_EBX

Bits	Description
31:28	**L2DTlb1GAssoc**. Read-only. Reset: Fixed,Fh. L2 data TLB associativity for 1-GB pages.
27:16	**L2DTlb1GSize**. Read-only. Reset: Fixed,040h. L2 data TLB number of entries for 1-GB pages.
15:12	**L2ITlb1GAssoc**. Read-only. Reset: Fixed,0h. L2 instruction TLB associativity for 1-GB pages.
11:0	**L2ITlb1GSize**. Read-only. Reset: Fixed,000h. L2 instruction TLB number of entries for 1-GB pages.

CPUID_Fn8000001A_EAX [Performance Optimization Identifiers] (Core::X86::Cpuid::PerfOptId)

Read-only. Reset: Fixed,0000_0006h.

This function returns performance related information.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn8000001A_EAX

Bits	Description
31:3	Reserved.
2	**FP256**. Read-only. Reset: Fixed,1. 256-bit AVX instructions are executed with full-width internal operations and pipelines rather than decomposing them into internal 128-bit suboperations.
1	**MOVU**. Read-only. Reset: Fixed,1. MOVU SSE instructions are more efficient and should be preferred to SSE MOVL/MOVH. MOVUPS is more efficient than MOVLPS/MOVHPS. MOVUPD is more efficient than MOVLPD/MOVHPD.
0	**FP128**. Read-only. Reset: Fixed,0. 128-bit SSE (multimedia) instructions are executed with full-width internal operations and pipelines rather than decomposing them into internal 64-bit suboperations.

CPUID_Fn8000001B_EAX [Instruction Based Sampling Identifiers] (Core::X86::Cpuid::IbsIdEax)

Read-only.

This function returns IBS feature information.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn8000001B_EAX

Bits	Description
31:11	Reserved.
10	**IbsOpData4**. Read-only. Reset: Fixed,0. IBS op data 4 MSR supported.
9	**IbsFetchCtlExtd**: **IBS fetch control extended MSR supported**. Read-only. Reset: Fixed,1. Indicates support for Core::X86::Msr::IC_IBS_EXTD_CTL.
8	**OpBrnFuse**: **fused branch op indication supported**. Read-only. Reset: Fixed,1. Indicates support for

	Core::X86::Msr::IBS_OP_DATA[IbsOpBrnFuse].
7	**RipInvalidChk**: **invalid RIP indication supported**. Read-only. Reset: Fixed,1. Indicates support for Core::X86::Msr::IBS_OP_DATA[IbsRipInvalid].
6	**OpCntExt**: **IbsOpCurCnt and IbsOpMaxCnt extend by 7 bits**. Read-only. Reset: Fixed,1. Indicates support for Core::X86::Msr::IBS_OP_CTL[IbsOpCurCnt[26:20],IbsOpMaxCnt[26:20]].
5	**BrnTrgt**. Read-only. Reset: Fixed,1. Branch target address reporting supported.
4	**OpCnt**. Read-only. Reset: Fixed,1. Op counting mode supported.
3	**RdWrOpCnt**. Read-only. Reset: Fixed,1. Read/Write of op counter supported.
2	**OpSam**. Read-only. Reset: Fixed,1. IBS execution sampling supported.
1	**FetchSam**. Read-only. Reset: X. IBS fetch sampling supported.
0	**IBSFFV**. Read-only. Reset: Fixed,1. IBS feature flags valid.

CPUID_Fn8000001D_EAX_x00 [Cache Properties (DC)] (Core::X86::Cpuid::CachePropEax0)

Enable: Core::X86::Cpuid::FeatureExtIdEcx[TopologyExtensions].

Core::X86::Cpuid::CachePropEax0 reports topology information for the DC.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn8000001D_EAX_x00

<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>31:26</td><td>Reserved.</td></tr>
<tr><td>25:14</td><td>NumSharingCache: number of logical processors sharing cache. Read-only. Reset: XXXh. The number of logical processors sharing this cache is NumSharingCache + 1.</td></tr>
<tr><td>13:10</td><td>Reserved.</td></tr>
<tr><td>9</td><td>FullyAssociative: fully associative cache. Read-only. Reset: Fixed,0. 1=Cache is fully associative.</td></tr>
<tr><td>8</td><td>SelfInitialization: cache is self-initializing. Read-only. Reset: Fixed,1. 1=Cache is self initializing; cache does not need software initialization.</td></tr>
<tr><td>7:5</td><td>CacheLevel: cache level. Read-only. Reset: Fixed,1h. Identifies the cache level.
ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>Reserved.</td></tr>
<tr><td>1h</td><td>Level 1</td></tr>
<tr><td>2h</td><td>Level 2</td></tr>
<tr><td>3h</td><td>Level 3</td></tr>
<tr><td>7h-4h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>4:0</td><td>CacheType: cache type. Read-only. Reset: Fixed,01h. Identifies the type of cache.
ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>00h</td><td>Null; no more caches.</td></tr>
<tr><td>01h</td><td>Data cache.</td></tr>
<tr><td>02h</td><td>Instruction cache.</td></tr>
<tr><td>03h</td><td>Unified cache.</td></tr>
<tr><td>1Fh-04h</td><td>Reserved.</td></tr>
</table></td></tr>
</table>

CPUID_Fn8000001D_EAX_x01 [Cache Properties (IC)] (Core::X86::Cpuid::CachePropEax1)

Read-only. Enable: Core::X86::Cpuid::FeatureExtIdEcx[TopologyExtensions].

Core::X86::Cpuid::CachePropEax1 reports topology information for the IC. See Core::X86::Cpuid::CachePropEax0.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn8000001D_EAX_x01

Bits	Description
31:26	Reserved.
25:14	**NumSharingCache**: **number of logical processors sharing cache**. Read-only. Reset: XXXh. See Core::X86::Cpuid::CachePropEax0[NumSharingCache].
13:10	Reserved.

9	**FullyAssociative**: **fully associative cache**. Read-only. Reset: Fixed,0. See Core::X86::Cpuid::CachePropEax0[FullyAssociative].
8	**SelfInitialization**: **cache is self-initializing**. Read-only. Reset: Fixed,1. See Core::X86::Cpuid::CachePropEax0[SelfInitialization].
7:5	**CacheLevel**: **cache level**. Read-only. Reset: Fixed,1h. Identifies the cache level. See Core::X86::Cpuid::CachePropEax0[CacheLevel].
4:0	**CacheType**: **cache type**. Read-only. Reset: Fixed,02h. See Core::X86::Cpuid::CachePropEax0[CacheType].

CPUID_Fn8000001D_EAX_x02 [Cache Properties (L2)] (Core::X86::Cpuid::CachePropEax2)

Read-only. Enable: Core::X86::Cpuid::FeatureExtIdEcx[TopologyExtensions].

Core::X86::Cpuid::CachePropEax2 reports topology information for the L2. See Core::X86::Cpuid::CachePropEax0.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn8000001D_EAX_x02

Bits	Description
31:26	Reserved.
25:14	**NumSharingCache**: **number of logical processors sharing cache**. Read-only. Reset: XXXh. Core::X86::Cpuid::CachePropEax0[NumSharingCache].
13:10	Reserved.
9	**FullyAssociative**: **fully associative cache**. Read-only. Reset: Fixed,0. Core::X86::Cpuid::CachePropEax0[FullyAssociative].
8	**SelfInitialization**: **cache is self-initializing**. Read-only. Reset: Fixed,1. Core::X86::Cpuid::CachePropEax0[SelfInitialization].
7:5	**CacheLevel**: **cache level**. Read-only. Reset: Fixed,2h. Identifies the cache level. Core::X86::Cpuid::CachePropEax0[CacheLevel].
4:0	**CacheType**: **cache type**. Read-only. Reset: Fixed,03h. Core::X86::Cpuid::CachePropEax0[CacheType].

CPUID_Fn8000001D_EAX_x03 [Cache Properties (L3)] (Core::X86::Cpuid::CachePropEax3)

Read-only. Enable: Core::X86::Cpuid::FeatureExtIdEcx[TopologyExtensions].

Core::X86::Cpuid::CachePropEax3 reports topology information for the L3.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn8000001D_EAX_x03

Bits	Description
31:26	Reserved.
25:14	**NumSharingCache**: **number of logical processors sharing cache**. Read-only. Reset: XXXh. The number of logical processors sharing this cache is NumSharingCache + 1.
13:10	Reserved.
9	**FullyAssociative**: **fully associative cache**. Read-only. Reset: Fixed,0. Core::X86::Cpuid::CachePropEax0[FullyAssociative].
8	**SelfInitialization**: **cache is self-initializing**. Read-only. Reset: Fixed,1. Core::X86::Cpuid::CachePropEax0[SelfInitialization].
7:5	**CacheLevel**: **cache level**. Read-only. Reset: Fixed,3h. Identifies the cache level. Core::X86::Cpuid::CachePropEax0[CacheLevel].
4:0	**CacheType**: **cache type**. Read-only. Reset: Fixed,03h. Core::X86::Cpuid::CachePropEax0[CacheType].

CPUID_Fn8000001D_EAX_x04 [Cache Properties Null] (Core::X86::Cpuid::CachePropEax4)

Read-only. Reset: Fixed,0000_0000h. Enable: Core::X86::Cpuid::FeatureExtIdEcx[TopologyExtensions].

Core::X86::Cpuid::CachePropEax4 reports done/null. See Core::X86::Cpuid::CachePropEax0.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn8000001D_EAX_x04

Bits	Description
31:5	Reserved.
4:0	**CacheType**: **cache type**. Read-only. Reset: Fixed,00h. Core::X86::Cpuid::CachePropEax0[CacheType].

CPUID_Fn8000001D_EBX_x00 [Cache Properties (DC)] (Core::X86::Cpuid::CachePropEbx0)

Read-only. Reset: Fixed,01C0_003Fh. Enable: Core::X86::Cpuid::FeatureExtIdEcx[TopologyExtensions].

Core::X86::Cpuid::CachePropEbx0 reports topology information for the DC. See Core::X86::Cpuid::CachePropEax0.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn8000001D_EBX_x00

Bits	Description
31:22	**CacheNumWays**: **cache number of ways**. Read-only. Reset: Fixed,007h. Cache number of ways is CacheNumWays + 1.
21:12	**CachePhysPartitions**: **cache physical line partitions**. Read-only. Reset: Fixed,000h. Cache partitions is CachePhysPartitions + 1.
11:0	**CacheLineSize**: **cache line size in bytes**. Read-only. Reset: Fixed,03Fh. Cache line size in bytes is CacheLineSize + 1.

CPUID_Fn8000001D_EBX_x01 [Cache Properties (IC)] (Core::X86::Cpuid::CachePropEbx1)

Read-only. Reset: Fixed,01C0_003Fh. Enable: Core::X86::Cpuid::FeatureExtIdEcx[TopologyExtensions].

Core::X86::Cpuid::CachePropEbx1 reports topology information for the IC. See Core::X86::Cpuid::CachePropEax0.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn8000001D_EBX_x01

Bits	Description
31:22	**CacheNumWays**: **cache number of ways**. Read-only. Reset: Fixed,007h. Core::X86::Cpuid::CachePropEbx0[CacheNumWays].
21:12	**CachePhysPartitions**: **cache physical line partitions**. Read-only. Reset: Fixed,000h. Core::X86::Cpuid::CachePropEbx0[CachePhysPartitions].
11:0	**CacheLineSize**: **cache line size in bytes**. Read-only. Reset: Fixed,03Fh. Core::X86::Cpuid::CachePropEbx0[CacheLineSize].

CPUID_Fn8000001D_EBX_x02 [Cache Properties (L2)] (Core::X86::Cpuid::CachePropEbx2)

Read-only. Reset: Fixed,01C0_003Fh. Enable: Core::X86::Cpuid::FeatureExtIdEcx[TopologyExtensions].

Core::X86::Cpuid::CachePropEbx2 reports topology information for the L2. See Core::X86::Cpuid::CachePropEax0.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn8000001D_EBX_x02

Bits	Description
31:22	**CacheNumWays**: **cache number of ways**. Read-only. Reset: Fixed,007h. See Core::X86::Cpuid::CachePropEbx0[CacheNumWays].
21:12	**CachePhysPartitions**: **cache physical line partitions**. Read-only. Reset: Fixed,000h. See Core::X86::Cpuid::CachePropEbx0[CachePhysPartitions].
11:0	**CacheLineSize**: **cache line size in bytes**. Read-only. Reset: Fixed,03Fh. See Core::X86::Cpuid::CachePropEbx0[CacheLineSize].

CPUID_Fn8000001D_EBX_x03 [Cache Properties (L3)] (Core::X86::Cpuid::CachePropEbx3)

Read-only. Reset: Fixed,03C0_003Fh. Enable: Core::X86::Cpuid::FeatureExtIdEcx[TopologyExtensions].

Core::X86::Cpuid::CachePropEbx3 reports topology information for the L3. See Core::X86::Cpuid::CachePropEax0.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn8000001D_EBX_x03

Bits	Description
31:22	**CacheNumWays**: **cache number of ways**. Read-only. Reset: Fixed,00Fh. See Core::X86::Cpuid::CachePropEbx0[CacheNumWays].
21:12	**CachePhysPartitions**: **cache physical line partitions**. Read-only. Reset: Fixed,000h. See Core::X86::Cpuid::CachePropEbx0[CachePhysPartitions].
11:0	**CacheLineSize**: **cache line size in bytes**. Read-only. Reset: Fixed,03Fh. See Core::X86::Cpuid::CachePropEbx0[CacheLineSize].

CPUID_Fn8000001D_EBX_x04 [Cache Properties Null] (Core::X86::Cpuid::CachePropEbx4)

Read-only. Reset: Fixed,0000_0000h. Enable: Core::X86::Cpuid::FeatureExtIdEcx[TopologyExtensions].

Core::X86::Cpuid::CachePropEax4 reports done/null. See Core::X86::Cpuid::CachePropEax0.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn8000001D_EBX_x04

Bits	Description

31:0	Reserved.

CPUID_Fn8000001D_ECX_x00 [Cache Properties (DC)] (Core::X86::Cpuid::CachePropEcx0)

Read-only. Reset: Fixed,0000_003Fh. Enable: Core::X86::Cpuid::FeatureExtIdEcx[TopologyExtensions].

Core::X86::Cpuid::CachePropEcx0 reports topology information for the DC. See Core::X86::Cpuid::CachePropEax0.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn8000001D_ECX_x00

Bits	Description
31:0	**CacheNumSets**: **cache number of sets**. Read-only. Reset: Fixed,0000_003Fh. Cache number of sets is CacheNumSets + 1.

CPUID_Fn8000001D_ECX_x01 [Cache Properties (IC)] (Core::X86::Cpuid::CachePropEcx1)

Read-only. Reset: Fixed,0000_003Fh. Enable: Core::X86::Cpuid::FeatureExtIdEcx[TopologyExtensions].

Core::X86::Cpuid::CachePropEcx1 reports topology information for the IC. See Core::X86::Cpuid::CachePropEax0.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn8000001D_ECX_x01

Bits	Description
31:0	**CacheNumSets**: **cache number of sets**. Read-only. Reset: Fixed,0000_003Fh. See Core::X86::Cpuid::CachePropEcx0[CacheNumSets].

CPUID_Fn8000001D_ECX_x02 [Cache Properties (L2)] (Core::X86::Cpuid::CachePropEcx2)

Read-only. Reset: Fixed,0000_03FFh. Enable: Core::X86::Cpuid::FeatureExtIdEcx[TopologyExtensions].

Core::X86::Cpuid::CachePropEcx2 reports topology information for the L2. See Core::X86::Cpuid::CachePropEax0.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn8000001D_ECX_x02

Bits	Description
31:0	**CacheNumSets**: **cache number of sets**. Read-only. Reset: Fixed,0000_03FFh. See Core::X86::Cpuid::CachePropEcx0[CacheNumSets].

CPUID_Fn8000001D_ECX_x03 [Cache Properties (L3)] (Core::X86::Cpuid::CachePropEcx3)

Read-only. Enable: Core::X86::Cpuid::FeatureExtIdEcx[TopologyExtensions].

Core::X86::Cpuid::CachePropEcx3 reports topology information for the L3.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn8000001D_ECX_x03

Bits	Description
31:0	**CacheNumSets**: **cache number of sets**. Read-only. Reset: 0000_XXXXh. See Core::X86::Cpuid::CachePropEcx0[CacheNumSets].

ValidValues:

Value	Description
0000_3FFEh-0000_0000h	Reserved.
0000_3FFFh	16384 L3 Cache Sets.
0000_7FFEh-0000_4000h	Reserved.
0000_7FFFh	32768 L3 Cache Sets.
FFFF_FFFFh-0000_8000h	Reserved.

CPUID_Fn8000001D_ECX_x04 [Cache Properties Null] (Core::X86::Cpuid::CachePropEcx4)

Read-only. Reset: Fixed,0000_0000h. Enable: Core::X86::Cpuid::FeatureExtIdEcx[TopologyExtensions].

Core::X86::Cpuid::CachePropEax3 reports done/null. See Core::X86::Cpuid::CachePropEax0.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn8000001D_ECX_x04

Bits	Description
31:0	**CacheNumSets**. Read-only. Reset: Fixed,0000_0000h. Cache number of sets.

CPUID_Fn8000001D_EDX_x00 [Cache Properties (DC)] (Core::X86::Cpuid::CachePropEdx0)

Read-only. Reset: Fixed,0000_0000h. Enable: Core::X86::Cpuid::FeatureExtIdEcx[TopologyExtensions].

Core::X86::Cpuid::CachePropEdx0 reports topology information for the DC. See Core::X86::Cpuid::CachePropEax0.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn8000001D_EDX_x00

Bits	Description
31:2	Reserved.
1	**CacheInclusive**: **cache inclusive**. Read-only. Reset: Fixed,0. 0=Cache is not inclusive of lower cache levels. 1=Cache is inclusive of lower cache levels.
0	**WBINVD**: **Write-Back Invalidate/Invalidate**. Read-only. Reset: Fixed,0. 0=WBINVD/INVD invalidates all lower level caches of non-originating cores sharing this cache. 1=WBINVD/INVD not ensured to invalidate all lower level caches of non-originating cores sharing this cache.

CPUID_Fn8000001D_EDX_x01 [Cache Properties (IC)] (Core::X86::Cpuid::CachePropEdx1)

Read-only. Reset: Fixed,0000_0000h. Enable: Core::X86::Cpuid::FeatureExtIdEcx[TopologyExtensions].

Core::X86::Cpuid::CachePropEdx1 reports topology information for the IC. See Core::X86::Cpuid::CachePropEax0.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn8000001D_EDX_x01

Bits	Description
31:2	Reserved.
1	**CacheInclusive**: **cache inclusive**. Read-only. Reset: Fixed,0. See Core::X86::Cpuid::CachePropEdx0[CacheInclusive].
0	**WBINVD**: **Write-Back Invalidate/Invalidate**. Read-only. Reset: Fixed,0. 0=WBINVD/INVD invalidates all lower level caches of non-originating cores sharing this cache. 1=WBINVD/INVD may not invalidate all lower level caches of non-originating cores sharing this cache. See Core::X86::Cpuid::CachePropEdx0[WBINVD].

CPUID_Fn8000001D_EDX_x02 [Cache Properties (L2)] (Core::X86::Cpuid::CachePropEdx2)

Read-only. Reset: Fixed,0000_0002h. Enable: Core::X86::Cpuid::FeatureExtIdEcx[TopologyExtensions].

Core::X86::Cpuid::CachePropEdx2 reports topology information for the L2. See Core::X86::Cpuid::CachePropEax0.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn8000001D_EDX_x02

Bits	Description
31:2	Reserved.
1	**CacheInclusive**: **cache inclusive**. Read-only. Reset: Fixed,1. See Core::X86::Cpuid::CachePropEdx0[CacheInclusive].
0	**WBINVD**: **Write-Back Invalidate/Invalidate**. Read-only. Reset: Fixed,0. 0=WBINVD/INVD invalidates all lower level caches of non-originating cores sharing this cache. 1=WBINVD/INVD may not invalidate all lower level caches of non-originating cores sharing this cache.

CPUID_Fn8000001D_EDX_x03 [Cache Properties (L3)] (Core::X86::Cpuid::CachePropEdx3)

Read-only. Reset: Fixed,0000_0001h. Enable: Core::X86::Cpuid::FeatureExtIdEcx[TopologyExtensions].

Core::X86::Cpuid::CachePropEdx3 reports reports topology information for the L3. See Core::X86::Cpuid::CachePropEax0.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn8000001D_EDX_x03

Bits	Description
31:2	Reserved.
1	**CacheInclusive**: **cache inclusive**. Read-only. Reset: Fixed,0. See Core::X86::Cpuid::CachePropEdx0[CacheInclusive].

0	**WBINVD**: **Write-Back Invalidate/Invalidate**. Read-only. Reset: Fixed,1. 0=WBINVD/INVD invalidates all lower level caches of non-originating cores sharing this cache. 1=WBINVD/INVD may not invalidate all lower level caches of non-originating cores sharing this cache.

CPUID_Fn8000001D_EDX_x04 [Cache Properties Null] (Core::X86::Cpuid::CachePropEdx4)

Read-only. Reset: Fixed,0000_0000h. Enable: Core::X86::Cpuid::FeatureExtIdEcx[TopologyExtensions].

Core::X86::Cpuid::CachePropEax3 reports done/null. See Core::X86::Cpuid::CachePropEax0.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn8000001D_EDX_x04

Bits	Description
31:0	Reserved.

CPUID_Fn8000001E_EAX [Extended APIC ID] (Core::X86::Cpuid::ExtApicId)

Read-only. Enable: (Core::X86::Cpuid::FeatureExtIdEcx[TopologyExtensions] && Core::X86::Msr::APIC_BAR[ApicEn]).

If Core::X86::Cpuid::FeatureExtIdEcx[TopologyExtensions] == 0 then CPUID Fn8000001E_E[D,C,B,A]X are Reserved. If (Core::X86::Msr::APIC_BAR[ApicEn] == 0) then Core::X86::Cpuid::ExtApicId[ExtendedApicId] is Reserved.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn8000001E_EAX

Bits	Description
31:0	**ExtendedApicId**: **extended APIC ID**. Read-only. See 2.1.14.2.1.3 [ApicId Enumeration Requirements].
	Reset: (Core::X86::Msr::APIC_BAR[ApicEn] && Core::X86::Msr::APIC_BAR[x2ApicEn]) ? Core::X86::Msr::APIC_ID[ApicId[31:0]] : Core::X86::Msr::APIC_BAR[ApicEn] ? {00_0000h , Core::X86::Apic::ApicId[ApicId]} : 0000_0000h.

CPUID_Fn8000001E_EBX [Core Identifiers] (Core::X86::Cpuid::CoreId)

Read-only. Enable: Core::X86::Cpuid::FeatureExtIdEcx[TopologyExtensions].

See Core::X86::Cpuid::ExtApicId.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn8000001E_EBX

Bits	Description
31:16	Reserved.
15:8	**ThreadsPerCore**: **threads per core**. Read-only. Reset: XXh. The number of threads per core is ThreadsPerCore + 1.
7:0	**CoreId**: **core ID**. Read-only. Reset: XXh. Identifies the logical core unit ID.

CPUID_Fn8000001E_ECX [Node Identifiers] (Core::X86::Cpuid::NodeId)

Read-only. Enable: Core::X86::Cpuid::FeatureExtIdEcx[TopologyExtensions].

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn8000001E_ECX

<table>
<tr><th>Bits</th><th colspan="2">Description</th></tr>
<tr><td>31:11</td><td colspan="2">Reserved.</td></tr>
<tr><td>10:8</td><td colspan="2">NodesPerProcessor: Node per processor. Read-only. Reset: XXXb.</td></tr>
<tr><td></td><td colspan="2">ValidValues:</td></tr>
<tr><td></td><td>Value</td><td>Description</td></tr>
<tr><td></td><td>0h</td><td>1 node per processor.</td></tr>
<tr><td></td><td>7h-1h</td><td>Reserved.</td></tr>
<tr><td>7:0</td><td colspan="2">NodeId: Node ID. Read-only. Reset: Fixed,XXh.</td></tr>
</table>

CPUID_Fn8000001F_EAX [AMD Secure Encryption EAX] (Core::X86::Cpuid::SecureEncryptionEax)

Read-only. Reset: Fixed,0001_FC3Fh.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn8000001F_EAX

Bits	Description
31:17	Reserved.
16	**VTE**: **Virtual Transparent Encryption for SEV**. Read-only. Reset: Fixed,1. The Virtual Transparent Encryption

	feature can be enabled to force all memory accesses within an SEV guest to be encrypted with the guest's key. When enabled the hardware pretends that the C-bits for all guest mode accesses are 1 regardless of the actual guest page tables.
15	**PreventHostIBS**. Read-only. Reset: Fixed,1. Prevent host IBS for a SEV-ES guest.
14	**DebugStateSwap**. Read-only. Reset: Fixed,1. 1=DR0-3 and DR0-3_MASK can be saved/restored on world switches.
13	**AlternateInjection**. Read-only. Reset: Fixed,1. 1=SEV-ES guests can use an encrypted vmcb field for event injection.
12	**RestrictInjection**. Read-only. Reset: Fixed,1. 1=SEV-ES guests can refuse all event-injections except #HV.
11	**Req64BitHypervisor**. Read-only. Reset: Fixed,1. Require 64Bit Hypervisor.
10	**CoherencyEnforced**. Read-only. Reset: Fixed,1. Hardware enforces cache coherency.
9:6	Reserved.
5	**VMPL**. Read-only. Reset: Fixed,1. Multiple SNP guests can share memory using differing permissions.
4	**SNP**. Read-only. Reset: Fixed,1. RMP table can be enabled to protect memory even from hypervisor.
3	**SevEs**. Read-only. Reset: Fixed,1. Secure Encrypted ES.
2	**VmPgFlush**: **VM Page Flush MSR is supported**. Read-only. Reset: Fixed,1. See Core::X86::Msr::VMPAGE_FLUSH.
1	**SEV**. Read-only. Reset: Fixed,1. Secure Encrypted Virtualization supported.
0	**SME**. Read-only. Reset: Fixed,1. Secure Memory Encryption supported.

CPUID_Fn8000001F_EBX [AMD Secure Encryption EBX] (Core::X86::Cpuid::SecureEncryptionEbx)

Read-only.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn8000001F_EBX

Bits	Description
31:16	Reserved.
15:12	**VmplSupported**. Read-only. Reset: Fixed,4. Number of VMPLs supported.
11:6	**MemEncryptPhysAddWidth**. Read-only. Reset: 000XXXb. Reduction of physical address space in bits when memory encryption is enabled (0 indicates no reduction).
	ValidValues:
	Value | **Description** 00h | Physical Address width is not reduced. 01h | Physical Address width is reduced by one. 02h | Physical Address width is reduced by two. 03h | Physical Address width is reduced by three. 04h | Physical Address width is reduced by four. 05h | Physical Address width is reduced by five. 3Fh-06h | Reserved.
5:0	**CBit**. Read-only. Reset: 33h. Page table bit number used to enable memory encryption.

CPUID_Fn8000001F_ECX [AMD Secure Encryption ECX] (Core::X86::Cpuid::SecureEncryptionEcx)

Read-only.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn8000001F_ECX

Bits	Description
31:0	**NumEncryptedGuests**. Read-only. Reset: XXXX_XXXXh. Indicates the maximum ASID value that may be used for an SEV-enabled guest.

CPUID_Fn8000001F_EDX [Minimum ASID] (Core::X86::Cpuid::SecureEncryptionEdx)

Read-only.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn8000001F_EDX

Bits	Description
31:0	**MinimumSEVASID**: **Minimum SEV enabled, SEV-ES disabled ASID**. Read-only. Reset: 0000_000Xh.

	Indicates the minimum ASID value that must be used for an SEV-enabled, SEV-ES-disabled guest.

CPUID_Fn80000020_EAX_x00 [Platform QoS Enforcement for Memory Bandwidth] (Core::X86::Cpuid::PqeBandwidthEax0)

Read-only. Reset: 0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000020_EAX_x00

Bits	Description
31:0	Reserved.

CPUID_Fn80000020_EBX_x00 [Platform QoS Enforcement for Memory Bandwidth] (Core::X86::Cpuid::PqeBandwidthEbx0)

Read-only. Reset: 0000_0002h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000020_EBX_x00

Bits	Description
31:2	Reserved.
1	**MBE**: **memory bandwidth enforcement**. Read-only. Reset: 1. Memory bandwidth enforcement.
0	Reserved.

CPUID_Fn80000020_ECX_x00 [Platform QoS Enforcement for Memory Bandwidth] (Core::X86::Cpuid::PqeBandwidthEcx0)

Read-only. Reset: 0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000020_ECX_x00

Bits	Description
31:0	Reserved.

CPUID_Fn80000020_EDX_x00 [Platform QoS Enforcement for Memory Bandwidth] (Core::X86::Cpuid::PqeBandwidthEdx0)

Read-only. Reset: 0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000020_EDX_x00

Bits	Description
31:0	Reserved.

CPUID_Fn80000020_EAX_x01 [Platform QoS Enforcement for Memory Bandwidth] (Core::X86::Cpuid::PqeBandwidthEax1)

Read-only. Reset: 0000_000Bh.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000020_EAX_x01

Bits	Description
31:0	**BW_LEN**: **QOS Memory Bandwidth Enforcement Limit Size**. Read-only. Reset: 0000_000Bh. Size of the QOS Memory Bandwidth Enforcement Limit.

CPUID_Fn80000020_EBX_x01 [Platform QoS Enforcement for Memory Bandwidth] (Core::X86::Cpuid::PqeBandwidthEbx1)

Read-only. Reset: 0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000020_EBX_x01

Bits	Description
31:0	Reserved.

CPUID_Fn80000020_ECX_x01 [Platform QoS Enforcement for Memory Bandwidth] (Core::X86::Cpuid::PqeBandwidthEcx1)

Read-only. Reset: 0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000020_ECX_x01

Bits	Description
31:0	Reserved.

CPUID_Fn80000020_EDX_x01 [**Platform QoS Enforcement for Memory Bandwidth**] **(Core::X86::Cpuid::PqeBandwidthEdx1)**

Read-only. Reset: 0000_000Fh.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000020_EDX_x01

Bits	Description
31:0	**NumClassService**. Read-only. Reset: 0000_000Fh. Number of classes of service.

CPUID_Fn80000021_EAX [**Extended Feature 2 EAX**] **(Core::X86::Cpuid::FeatureExt2Eax)**

Read-only.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000021_EAX

Bits	Description
31:14	Reserved.
13	**PrefetchCtlMsr**. Read-only. Reset: 1. 1=Prefetch control MSR supported. See Core::X86::Msr::PrefetchControl.
12:7	Reserved.
6	**NullSelectorClearsBase**. Read-only. Reset: 1. 1=Null Selector Clears Base.
5:4	Reserved.
3	**SmmPgCfgLock**. Read-only. Reset: Fixed,1. 1=SMM paging configuration lock supported.
2	**LFenceAlwaysSerializing**. Read-only. Reset: Fixed,1. LFENCE is always serializing.
1	Reserved.
0	**NoNestedDataBp**. Read-only. Reset: Fixed,1. New data-breakpoints are ignored while switching to data-breakpoint handler.

2.1.16 MSR Registers

2.1.16.1 MSRs - MSR0000_xxxx

MSR0000_0010 [**Time Stamp Counter**] **(Core::X86::Msr::TSC)**

Read-write,Volatile. Reset: 0000_0000_0000_0000h.

The TSC uses a common reference for all sockets, cores and threads.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_0010

Bits	Description
63:0	**TSC**: **time stamp counter**. Read-write,Volatile. Reset: 0000_0000_0000_0000h. The TSC increments at the P0 frequency. The TSC counts at the same rate in all P-states, all C states, S0, or S1. A read of this MSR in guest mode is affected by Core::X86::Msr::TscRateMsr. The value (TSC/TSCRatio) is the TSC P0 frequency based value (as if TSCRatio == 1.0) when (TSCRatio != 1.0).

MSR0000_001B [**APIC Base Address**] **(Core::X86::Msr::APIC_BAR)**

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_001B

Bits	Description
63:48	Reserved.
47:12	**ApicBar[47:12]**: **APIC base address register**. Read-write. Reset: 0_000F_EE00h. Specifies the base address, physical address [47:12], for the APICXX register set in xAPIC mode. See 2.1.14.2.1.2 [APIC Register Space].
11	**ApicEn**: **APIC enable**. Read-write. Reset: 0. 0=Disable Local Apic. 1=Local APIC is enabled in xAPIC mode. See 2.1.14.2.1.2 [APIC Register Space].
10	**x2ApicEn**: **Extended APIC enable**. Read-write. Reset: 0. 0=Disable Extended Local Apic. 1=Extended Local APIC is enabled in x2APIC mode.
9	Reserved.
8	**BSC**: **boot strap core**. Read-write,Volatile. Reset: X. 0=The core is not the boot core of the BSP. 1=The core is the boot core of the BSP.

7:0	Reserved.

MSR0000_002A [Cluster ID] (Core::X86::Msr::EBL_CR_POWERON)

Writes to this register result in a GP fault with error code 0.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_002A

Bits	Description
63:18	Reserved.
17:16	**ClusterID**. Read,Error-on-write. Reset: 0h. The field does not affect hardware.
15:0	Reserved.

MSR0000_0048 [Speculative Control] (Core::X86::Msr::SPEC_CTRL)

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_0048

Bits	Description
63:8	Reserved.
7	**PSFD**: **Predictive Store Forwarding Disable**. Read-write. Reset: 0. 1= Disable predictive store forwarding.
6:3	Reserved.
2	**SSBD**. Read-write. Reset: 0. Speculative Store Bypass Disable.
1	**STIBP**. Read-write. Reset: 0. Single thread indirect branch predictor.
0	**IBRS**. Read-write. Reset: 0. Indirect branch restriction speculation.

MSR0000_0049 [Prediction Command] (Core::X86::Msr::PRED_CMD)

_ccd[7:0]_lthree0_core[7:0]; MSR0000_0049

Bits	Description
63:1	Reserved.
0	**IBPB**: **indirect branch prediction barrier**. Write-only,Error-on-read. Reset: 0. Supported if Core::X86::Cpuid::FeatureExtIdEbx[IBPB]==1.

MSR0000_008B [Patch Level] (Core::X86::Msr::PATCH_LEVEL)

Read,Error-on-write,Volatile. Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]; MSR0000_008B

Bits	Description
63:32	Reserved.
31:0	**PatchLevel**. Read,Error-on-write,Volatile. Reset: 0000_0000h. This returns an identification number for the microcode patch that has been loaded. If no patch has been loaded, this returns 0.

MSR0000_00E7 [Max Performance Frequency Clock Count] (Core::X86::Msr::MPERF)

Read-write,Volatile. Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_00E7

Bits	Description
63:0	**MPERF**: **maximum core clocks counter**. Read-write,Volatile. Reset: 0000_0000_0000_0000h. Incremented by hardware at the P0 frequency while the core is in C0. This register does not increment when the core is in the stop-grant state. In combination with Core::X86::Msr::APERF, this is used to determine the effective frequency of the core. A read of this MSR in guest mode is affected by Core::X86::Msr::TscRateMsr. This field uses software P-state numbering. See Core::X86::Msr::HWCR[EffFreqCntMwait], 2.1.6 [Effective Frequency]

MSR0000_00E8 [Actual Performance Frequency Clock Count] (Core::X86::Msr::APERF)

Read-write,Volatile. Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_00E8

Bits	Description
63:0	**APERF**: **actual core clocks counter**. Read-write,Volatile. Reset: 0000_0000_0000_0000h. This register increments in proportion to the actual number of core clocks cycles while the core is in C0. The register does not increment when the core is in the stop-grant state. See Core::X86::Msr::MPERF.

MSR0000_00FE [MTRR Capabilities] (Core::X86::Msr::MTRRcap)

Read,Error-on-write. Reset: 0000_0000_0000_0508h.

_ccd[7:0]_lthree0_core[7:0]; MSR0000_00FE

Bits	Description
63:11	Reserved.
10	**MtrrCapWc**: **write-combining memory type**. Read,Error-on-write. Reset: 1. 1=The write combining memory type is supported.
9	Reserved.
8	**MtrrCapFix**: **fixed range register**. Read,Error-on-write. Reset: 1. 1=Fixed MTRRs are supported.
7:0	**MtrrCapVCnt**: **variable range registers count**. Read,Error-on-write. Reset: 08h. Specifies the number of variable MTRRs supported.

MSR0000_0174 [SYSENTER CS] (Core::X86::Msr::SYSENTER_CS)

Read-write. Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_0174

Bits	Description
63:16	Reserved.
15:0	**SysEnterCS**: **SYSENTER target CS**. Read-write. Reset: 0000h. Holds the called procedure code segment.

MSR0000_0175 [SYSENTER ESP] (Core::X86::Msr::SYSENTER_ESP)

Read-write. Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_0175

Bits	Description
63:32	Reserved.
31:0	**SysEnterESP**: **SYSENTER target SP**. Read-write. Reset: 0000_0000h. Holds the called procedure stack pointer.

MSR0000_0176 [SYSENTER EIP] (Core::X86::Msr::SYSENTER_EIP)

Read-write. Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_0176

Bits	Description
63:32	Reserved.
31:0	**SysEnterEIP**: **SYSENTER target IP**. Read-write. Reset: 0000_0000h. Holds the called procedure instruction pointer.

MSR0000_0179 [Global Machine Check Capabilities] (Core::X86::Msr::MCG_CAP)

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_0179

Bits	Description
63:9	Reserved.
8	**McgCtlP**: **MCG_CTL register present**. Read-only,Error-on-write. Reset: Fixed,1. 1=The machine check control registers (MCi_CTL) are present. See 3.1 [Machine Check Architecture].
7:0	**Count**. Read-only,Error-on-write,Volatile. Reset: XXh. Indicates the number of error reporting banks visible to the core. This value may differ from core to core.

MSR0000_017A [Global Machine Check Status] (Core::X86::Msr::MCG_STAT)

Read-write,Volatile. Reset: 0000_0000_0000_0000h.

See 3.1 [Machine Check Architecture].

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_017A

Bits	Description
63:3	Reserved.
2	**MCIP**. Read-write,Volatile. Reset: 0. 1=A machine check is in progress. Machine check in progress.
1	**EIPV**: **error instruction pointer valid**. Read-write,Volatile. Reset: 0. 1=The instruction pointer that was pushed

	onto the stack by the machine check mechanism references the instruction that caused the machine check error.
0	**RIPV**: **restart instruction pointer valid**. Read-write,Volatile. Reset: 0. 0=The interrupt was not precise and/or the process (task) context may be corrupt; continued operation of this process may not be possible without intervention, however system processing or other processes may be able to continue with appropriate software clean up. 1=Program execution can be reliably restarted at the EIP address on the stack.

MSR0000_017B [Global Machine Check Exception Reporting Control] (Core::X86::Msr::MCG_CTL)

Reset: 0000_0000_0000_0000h.

This register controls enablement of the individual error reporting banks; see 3.1 [Machine Check Architecture] and 3.1.2.1 [Global Registers]. When a machine check register bank is not enabled in MCG_CTL, errors for that bank are not logged or reported, and actions enabled through the MCA are not taken; each MCi_CTL register identifies which errors are still corrected when MCG_CTL[i] is disabled.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_017B

Bits	Description
63:7	**MCnEn**. Configurable. Reset: 000_0000_0000_0000h.
	Description: 1=The MC machine check register bank is enabled. Width of this field is SOC implementation and configuration specific. See 3.1.2.1 [Global Registers].
6:0	**MCnEnCore**. Read-write. Reset: 00h. 1=The MC machine check register bank is enabled.
	ValidValues:

Bit	Description
[0]	Enable MCA for LSDC
[1]	Enable MCA for ICBP
[2]	Enable MCA for L2
[3]	Enable MCA for DE
[4]	Reserved
[5]	Enable MCA for SCEX
[6]	Enable MCA for FP

MSR0000_01D9 [Debug Control] (Core::X86::Msr::DBG_CTL_MSR)

Read-write. Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_01D9

Bits	Description
63:6	Reserved.
5:2	**PB**: **performance monitor pin control**. Read-write. Reset: 0h. This field does not control any hardware.
1	**BTF**. Read-write. Reset: 0. 1=Enable branch single step.
0	**LBR**. Read-write. Reset: 0. 1=Enable last branch record.

MSR0000_01DB [Last Branch From IP] (Core::X86::Msr::BR_FROM)

Read,Error-on-write,Volatile. Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_01DB

Bits	Description
63:0	**LastBranchFromIP**. Read,Error-on-write,Volatile. Reset: 0000_0000_0000_0000h. Loaded with the segment offset of the branch instruction.

MSR0000_01DC [Last Branch To IP] (Core::X86::Msr::BR_TO)

Read,Error-on-write,Volatile. Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_01DC

Bits	Description
63:61	Reserved.
60:0	**LastBranchToIP**. Read,Error-on-write,Volatile. Reset: 0000_0000_0000_0000h. Holds the target RIP of the last branch that occurred before an exception or interrupt.

MSR0000_01DD [Last Exception From IP] (Core::X86::Msr::LastExcpFromIp)

Read,Error-on-write,Volatile. Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_01DD

Bits	Description
63:0	**LastIntFromIP**. Read,Error-on-write,Volatile. Reset: 0000_0000_0000_0000h. Holds the source RIP of the last branch that occurred before the exception or interrupt.

MSR0000_01DE [Last Exception To IP] (Core::X86::Msr::LastExcpToIp)

Read,Error-on-write,Volatile. Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_01DE

Bits	Description
63:61	Reserved.
60:0	**LastIntToIP**. Read,Error-on-write,Volatile. Reset: 0000_0000_0000_0000h. Holds the target RIP of the last branch that occurred before the exception or interrupt.

MSR0000_020[0...E] [Variable-Size MTRRs Base] (Core::X86::Msr::MtrrVarBase)

Each MTRR (Core::X86::Msr::MtrrVarBase, Core::X86::Msr::MtrrFix_64K through Core::X86::Msr::MtrrFix_4K_7, or Core::X86::Msr::MTRRdefType) specifies a physical address range and a corresponding memory type (MemType) associated with that range. Setting the memory type to an unsupported value results in a #GP.

The variable-size MTRRs come in pairs of base and mask registers (MSR0000_0200 and MSR0000_0201 are the first pair, etc.). Variables MTRRs are enabled through Core::X86::Msr::MTRRdefType[MtrrDefTypeEn]. A core access--with address CPUAddr--is determined to be within the address range of a variable-size MTRR if the following equation is true:
CPUAddr[47:12] & PhyMask[47:12] == PhyBase[47:12] & PhyMask[47:12].

For example, if the variable MTRR spans 256 KB and starts at the 1 MB address the PhyBase would be set to 0_0010_0000h and the PhyMask to F_FFFC_0000h (with zeros filling in for bits[11:0]). This results in a range from 0_0010_0000h to 0_0013_FFFFh.

_ccd[7:0]_lthree0_core[7:0]_n0; MSR0000_0200
_ccd[7:0]_lthree0_core[7:0]_n1; MSR0000_0202
_ccd[7:0]_lthree0_core[7:0]_n2; MSR0000_0204
_ccd[7:0]_lthree0_core[7:0]_n3; MSR0000_0206
_ccd[7:0]_lthree0_core[7:0]_n4; MSR0000_0208
_ccd[7:0]_lthree0_core[7:0]_n5; MSR0000_020A
_ccd[7:0]_lthree0_core[7:0]_n6; MSR0000_020C
_ccd[7:0]_lthree0_core[7:0]_n7; MSR0000_020E

Bits	Description
63:48	Reserved.
47:12	**PhyBase**: **base address**. Read-write. Reset: X_XXXX_XXXXh. Physical base address.
11:3	Reserved.
2:0	**MemType**: **memory type**. Read-write. Reset: XXXb. Address range from 00000h to 0FFFFh.
	ValidValues:

Value	Description
0h	UC or uncacheable.
1h	WC or write combining.
3h-2h	Reserved.
4h	WT or write through.
5h	WP or write protect.
6h	WB or write back.
7h	Reserved.

MSR0000_020[1...F] [Variable-Size MTRRs Mask] (Core::X86::Msr::MtrrVarMask)

_ccd[7:0]_lthree0_core[7:0]_n0; MSR0000_0201
_ccd[7:0]_lthree0_core[7:0]_n1; MSR0000_0203
_ccd[7:0]_lthree0_core[7:0]_n2; MSR0000_0205
_ccd[7:0]_lthree0_core[7:0]_n3; MSR0000_0207
_ccd[7:0]_lthree0_core[7:0]_n4; MSR0000_0209
_ccd[7:0]_lthree0_core[7:0]_n5; MSR0000_020B
_ccd[7:0]_lthree0_core[7:0]_n6; MSR0000_020D
_ccd[7:0]_lthree0_core[7:0]_n7; MSR0000_020F

Bits	Description
63:48	Reserved.
47:12	**PhyMask**: **address mask**. Read-write. Reset: X_XXXX_XXXXh. Physical address mask.
11	**Valid**: **valid**. Read-write. Reset: X. 1=The variable-size MTRR pair is enabled.
10:0	Reserved.

MSR0000_0250 [Fixed-Size MTRRs] (Core::X86::Msr::MtrrFix_64K)

See Core::X86::Msr::MtrrVarBase for general MTRR information. Fixed MTRRs are enabled through Core::X86::Msr::MTRRdefType[MtrrDefTypeFixEn,MtrrDefTypeEn]. For addresses below 1MB, the appropriate Fixed MTRRs override the default access destination. Each fixed MTRR includes two bits, RdDram and WrDram, that determine the destination based on the access type. Writing reserved MemType values causes an error-on-write.

_ccd[7:0]_lthree0_core[7:0]_nSIZE64K; MSR0000_0250

<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>63:61</td><td>Reserved.</td></tr>
<tr><td rowspan="2">60</td><td>RdDram_64K_70000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">59</td><td>WrDram_64K_70000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">58:56</td><td>MemType_64K_70000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>55:53</td><td>Reserved.</td></tr>
<tr><td rowspan="2">52</td><td>RdDram_64K_60000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">51</td><td>WrDram_64K_60000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">50:48</td><td>MemType_64K_60000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
</table></td></tr>
</table>

<table>
<tr><td></td><td><table>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>47:45</td><td>Reserved.</td></tr>
<tr><td rowspan="2">44</td><td>RdDram_64K_50000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">43</td><td>WrDram_64K_50000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">42:40</td><td>MemType_64K_50000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>39:37</td><td>Reserved.</td></tr>
<tr><td rowspan="2">36</td><td>RdDram_64K_40000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">35</td><td>WrDram_64K_40000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">34:32</td><td>MemType_64K_40000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>31:29</td><td>Reserved.</td></tr>
<tr><td rowspan="2">28</td><td>RdDram_64K_30000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset:</td></tr>
</table>

<table>
<tr><td></td><td>Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">27</td><td>WrDram_64K_30000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">26:24</td><td>MemType_64K_30000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>23:21</td><td>Reserved.</td></tr>
<tr><td rowspan="2">20</td><td>RdDram_64K_20000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">19</td><td>WrDram_64K_20000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">18:16</td><td>MemType_64K_20000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>15:13</td><td>Reserved.</td></tr>
<tr><td rowspan="2">12</td><td>RdDram_64K_10000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">11</td><td>WrDram_64K_10000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">10:8</td><td>MemType_64K_10000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
</table></td></tr>
</table>

<table>
<tr><td></td><td><table>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>7:5</td><td>Reserved.</td></tr>
<tr><td rowspan="2">4</td><td>RdDram_64K_00000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM. Address range from 00000h to 0FFFFh.
Core::X86::Msr::SYS_CFG[MtrrFixDramEn,MtrrFixDramModEn] masks reads of the stored value.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">3</td><td>WrDram_64K_00000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM. Address range from 00000h to 0FFFFh.
Core::X86::Msr::SYS_CFG[MtrrFixDramEn,MtrrFixDramModEn] masks reads of the stored value.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">2:0</td><td>MemType_64K_00000: memory type. Read-write. Reset: XXXb. Address range from 00000h to 0FFFFh.</td></tr>
<tr><td>ValidValues:<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
</table>

MSR0000_0258 [Fixed-Size MTRRs] (Core::X86::Msr::MtrrFix_16K_0)

See Core::X86::Msr::MtrrVarBase for general MTRR information. Fixed MTRRs are enabled through Core::X86::Msr::MTRRdefType[MtrrDefTypeFixEn,MtrrDefTypeEn]. For addresses below 1MB, the appropriate Fixed MTRRs override the default access destination. Each fixed MTRR includes two bits, RdDram and WrDram, that determine the destination based on the access type. Writing reserved MemType values causes an error-on-write.

_ccd[7:0]_lthree0_core[7:0]_nSIZE16K0; MSR0000_0258

<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>63:61</td><td>Reserved.</td></tr>
<tr><td rowspan="2">60</td><td>RdDram_16K_9C000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">59</td><td>WrDram_16K_9C000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">58:56</td><td>MemType_16K_9C000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
</table></td></tr>
</table>

<table>
<tr><td></td><td><table><tr><td>6h</td><td>WB or write back.</td></tr><tr><td>7h</td><td>Reserved.</td></tr></table></td></tr>
<tr><td>55:53</td><td>Reserved.</td></tr>
<tr><td rowspan="2">52</td><td>RdDram_16K_98000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">51</td><td>WrDram_16K_98000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">50:48</td><td>MemType_16K_98000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:<table><tr><th>Value</th><th>Description</th></tr><tr><td>0h</td><td>UC or uncacheable.</td></tr><tr><td>1h</td><td>WC or write combining.</td></tr><tr><td>3h-2h</td><td>Reserved.</td></tr><tr><td>4h</td><td>WT or write through.</td></tr><tr><td>5h</td><td>WP or write protect.</td></tr><tr><td>6h</td><td>WB or write back.</td></tr><tr><td>7h</td><td>Reserved.</td></tr></table></td></tr>
<tr><td>47:45</td><td>Reserved.</td></tr>
<tr><td rowspan="2">44</td><td>RdDram_16K_94000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">43</td><td>WrDram_16K_94000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">42:40</td><td>MemType_16K_94000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:<table><tr><th>Value</th><th>Description</th></tr><tr><td>0h</td><td>UC or uncacheable.</td></tr><tr><td>1h</td><td>WC or write combining.</td></tr><tr><td>3h-2h</td><td>Reserved.</td></tr><tr><td>4h</td><td>WT or write through.</td></tr><tr><td>5h</td><td>WP or write protect.</td></tr><tr><td>6h</td><td>WB or write back.</td></tr><tr><td>7h</td><td>Reserved.</td></tr></table></td></tr>
<tr><td>39:37</td><td>Reserved.</td></tr>
<tr><td rowspan="2">36</td><td>RdDram_16K_90000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">35</td><td>WrDram_16K_90000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
</table>

<table>
<tr><td rowspan="2">34:32</td><td>MemType_16K_90000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>31:29</td><td>Reserved.</td></tr>
<tr><td rowspan="2">28</td><td>RdDram_16K_8C000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">27</td><td>WrDram_16K_8C000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">26:24</td><td>MemType_16K_8C000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>23:21</td><td>Reserved.</td></tr>
<tr><td rowspan="2">20</td><td>RdDram_16K_88000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">19</td><td>WrDram_16K_88000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">18:16</td><td>MemType_16K_88000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>15:13</td><td>Reserved.</td></tr>
</table>

<table>
<tr><td rowspan="2">12</td><td>RdDram_16K_84000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">11</td><td>WrDram_16K_84000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">10:8</td><td>MemType_16K_84000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>7:5</td><td>Reserved.</td></tr>
<tr><td rowspan="2">4</td><td>RdDram_16K_80000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM. Address range from 80000h to 83FFFh. Core::X86::Msr::SYS_CFG[MtrrFixDramEn,MtrrFixDramModEn] masks reads of the stored value.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">3</td><td>WrDram_16K_80000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM. Address range from 80000h to 83FFFh. Core::X86::Msr::SYS_CFG[MtrrFixDramEn,MtrrFixDramModEn] masks reads of the stored value.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">2:0</td><td>MemType_16K_80000: memory type. Read-write. Reset: XXXb. Address range from 80000h to 83FFFh.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
</table>

MSR0000_0259 [Fixed-Size MTRRs] (Core::X86::Msr::MtrrFix_16K_1)

See Core::X86::Msr::MtrrVarBase for general MTRR information. Fixed MTRRs are enabled through Core::X86::Msr::MTRRdefType[MtrrDefTypeFixEn,MtrrDefTypeEn]. For addresses below 1MB, the appropriate Fixed MTRRs override the default access destination. Each fixed MTRR includes two bits, RdDram and WrDram, that determine the destination based on the access type. Writing reserved MemType values causes an error-on-write.

_ccd[7:0]_lthree0_core[7:0]_nSIZE16K1; MSR0000_0259

Bits	Description
63:61	Reserved.
60	**RdDram_16K_BC000**: **read DRAM**. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.

<table>
<tr><td></td><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td>59</td><td>WrDram_16K_BC000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td></td><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td>58:56</td><td>MemType_16K_BC000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td></td><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>55:53</td><td>Reserved.</td></tr>
<tr><td>52</td><td>RdDram_16K_B8000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td></td><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td>51</td><td>WrDram_16K_B8000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td></td><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td>50:48</td><td>MemType_16K_B8000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td></td><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>47:45</td><td>Reserved.</td></tr>
<tr><td>44</td><td>RdDram_16K_B4000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td></td><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td>43</td><td>WrDram_16K_B4000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td></td><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td>42:40</td><td>MemType_16K_B4000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td></td><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
</table></td></tr>
</table>

<table>
<tr><td></td><td><table>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>39:37</td><td>Reserved.</td></tr>
<tr><td>36</td><td>RdDram_16K_B0000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td></td><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td>35</td><td>WrDram_16K_B0000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td></td><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td>34:32</td><td>MemType_16K_B0000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td></td><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>31:29</td><td>Reserved.</td></tr>
<tr><td>28</td><td>RdDram_16K_AC000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td></td><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td>27</td><td>WrDram_16K_AC000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td></td><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td>26:24</td><td>MemType_16K_AC000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td></td><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>23:21</td><td>Reserved.</td></tr>
<tr><td>20</td><td>RdDram_16K_A8000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td></td><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td>19</td><td>WrDram_16K_A8000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to</td></tr>
</table>

<table>
<tr><td rowspan="2"></td><td>the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">18:16</td><td>MemType_16K_A8000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>15:13</td><td>Reserved.</td></tr>
<tr><td rowspan="2">12</td><td>RdDram_16K_A4000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">11</td><td>WrDram_16K_A4000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">10:8</td><td>MemType_16K_A4000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>7:5</td><td>Reserved.</td></tr>
<tr><td rowspan="2">4</td><td>RdDram_16K_A0000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM. Address range from A0000h to A3FFFh. Core::X86::Msr::SYS_CFG[MtrrFixDramEn,MtrrFixDramModEn] masks reads of the stored value.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">3</td><td>WrDram_16K_A0000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM. Address range from A0000h to A3FFFh. Core::X86::Msr::SYS_CFG[MtrrFixDramEn,MtrrFixDramModEn] masks reads of the stored value.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">2:0</td><td>MemType_16K_A0000: memory type. Read-write. Reset: XXXb. Address range from A0000h to A3FFFh.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
</table></td></tr>
</table>

4h	WT or write through.
5h	WP or write protect.
6h	WB or write back.
7h	Reserved.

MSR0000_0268 [Fixed-Size MTRRs] (Core::X86::Msr::MtrrFix_4K_0)

See Core::X86::Msr::MtrrVarBase for general MTRR information. Fixed MTRRs are enabled through Core::X86::Msr::MTRRdefType[MtrrDefTypeFixEn,MtrrDefTypeEn]. For addresses below 1MB, the appropriate Fixed MTRRs override the default access destination. Each fixed MTRR includes two bits, RdDram and WrDram, that determine the destination based on the access type. Writing reserved MemType values causes an error-on-write.

_ccd[7:0]_lthree0_core[7:0]_nSIZE4K0; MSR0000_0268

<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>63:61</td><td>Reserved.</td></tr>
<tr><td rowspan="2">60</td><td>RdDram_4K_C7000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">59</td><td>WrDram_4K_C7000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">58:56</td><td>MemType_4K_C7000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>55:53</td><td>Reserved.</td></tr>
<tr><td rowspan="2">52</td><td>RdDram_4K_C6000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">51</td><td>WrDram_4K_C6000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">50:48</td><td>MemType_4K_C6000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
</table>

<table>
<tr><td>47:45</td><td>Reserved.</td></tr>
<tr><td rowspan="2">44</td><td>RdDram_4K_C5000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">43</td><td>WrDram_4K_C5000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">42:40</td><td>MemType_4K_C5000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>39:37</td><td>Reserved.</td></tr>
<tr><td rowspan="2">36</td><td>RdDram_4K_C4000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">35</td><td>WrDram_4K_C4000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">34:32</td><td>MemType_4K_C4000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>31:29</td><td>Reserved.</td></tr>
<tr><td rowspan="2">28</td><td>RdDram_4K_C3000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">27</td><td>WrDram_4K_C3000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">26:24</td><td>MemType_4K_C3000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:</td></tr>
</table>

<table>
<tr><td></td><td><table><tr><th>Value</th><th>Description</th></tr><tr><td>0h</td><td>UC or uncacheable.</td></tr><tr><td>1h</td><td>WC or write combining.</td></tr><tr><td>3h-2h</td><td>Reserved.</td></tr><tr><td>4h</td><td>WT or write through.</td></tr><tr><td>5h</td><td>WP or write protect.</td></tr><tr><td>6h</td><td>WB or write back.</td></tr><tr><td>7h</td><td>Reserved.</td></tr></table></td></tr>
<tr><td>23:21</td><td>Reserved.</td></tr>
<tr><td rowspan="2">20</td><td>RdDram_4K_C2000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">19</td><td>WrDram_4K_C2000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">18:16</td><td>MemType_4K_C2000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:<table><tr><th>Value</th><th>Description</th></tr><tr><td>0h</td><td>UC or uncacheable.</td></tr><tr><td>1h</td><td>WC or write combining.</td></tr><tr><td>3h-2h</td><td>Reserved.</td></tr><tr><td>4h</td><td>WT or write through.</td></tr><tr><td>5h</td><td>WP or write protect.</td></tr><tr><td>6h</td><td>WB or write back.</td></tr><tr><td>7h</td><td>Reserved.</td></tr></table></td></tr>
<tr><td>15:13</td><td>Reserved.</td></tr>
<tr><td rowspan="2">12</td><td>RdDram_4K_C1000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">11</td><td>WrDram_4K_C1000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">10:8</td><td>MemType_4K_C1000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:<table><tr><th>Value</th><th>Description</th></tr><tr><td>0h</td><td>UC or uncacheable.</td></tr><tr><td>1h</td><td>WC or write combining.</td></tr><tr><td>3h-2h</td><td>Reserved.</td></tr><tr><td>4h</td><td>WT or write through.</td></tr><tr><td>5h</td><td>WP or write protect.</td></tr><tr><td>6h</td><td>WB or write back.</td></tr><tr><td>7h</td><td>Reserved.</td></tr></table></td></tr>
<tr><td>7:5</td><td>Reserved.</td></tr>
<tr><td>4</td><td>RdDram_4K_C0000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM. Address range from C0000h to C0FFFh.</td></tr>
</table>

<table>
<tr><td></td><td>Core::X86::Msr::SYS_CFG[MtrrFixDramEn,MtrrFixDramModEn] masks reads of the stored value.</td></tr>
<tr><td></td><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td>3</td><td>WrDram_4K_C0000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM. Address range from C0000h to C0FFFh. Core::X86::Msr::SYS_CFG[MtrrFixDramEn,MtrrFixDramModEn] masks reads of the stored value.</td></tr>
<tr><td></td><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td>2:0</td><td>MemType_4K_C0000: memory type. Read-write. Reset: XXXb. Address range from C0000h to C0FFFh.</td></tr>
<tr><td></td><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
</table>

MSR0000_0269 [Fixed-Size MTRRs] (Core::X86::Msr::MtrrFix_4K_1)

See Core::X86::Msr::MtrrVarBase for general MTRR information. Fixed MTRRs are enabled through Core::X86::Msr::MTRRdefType[MtrrDefTypeFixEn,MtrrDefTypeEn]. For addresses below 1MB, the appropriate Fixed MTRRs override the default access destination. Each fixed MTRR includes two bits, RdDram and WrDram, that determine the destination based on the access type. Writing reserved MemType values causes an error-on-write.

_ccd[7:0]_lthree0_core[7:0]_nSIZE4K1; MSR0000_0269

<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>63:61</td><td>Reserved.</td></tr>
<tr><td>60</td><td>RdDram_4K_CF000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td></td><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td>59</td><td>WrDram_4K_CF000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td></td><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td>58:56</td><td>MemType_4K_CF000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td></td><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>55:53</td><td>Reserved.</td></tr>
<tr><td>52</td><td>RdDram_4K_CE000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td></td><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
</table>

<table>
<tr><td rowspan="2">51</td><td>WrDram_4K_CE000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">50:48</td><td>MemType_4K_CE000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>47:45</td><td>Reserved.</td></tr>
<tr><td rowspan="2">44</td><td>RdDram_4K_CD000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">43</td><td>WrDram_4K_CD000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">42:40</td><td>MemType_4K_CD000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>39:37</td><td>Reserved.</td></tr>
<tr><td rowspan="2">36</td><td>RdDram_4K_CC000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">35</td><td>WrDram_4K_CC000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">34:32</td><td>MemType_4K_CC000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
</table></td></tr>
</table>

<table>
<tr><td></td><td><table><tr><td>5h</td><td>WP or write protect.</td></tr><tr><td>6h</td><td>WB or write back.</td></tr><tr><td>7h</td><td>Reserved.</td></tr></table></td></tr>
<tr><td>31:29</td><td>Reserved.</td></tr>
<tr><td rowspan="2">28</td><td>RdDram_4K_CB000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">27</td><td>WrDram_4K_CB000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">26:24</td><td>MemType_4K_CB000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:<table><tr><th>Value</th><th>Description</th></tr><tr><td>0h</td><td>UC or uncacheable.</td></tr><tr><td>1h</td><td>WC or write combining.</td></tr><tr><td>3h-2h</td><td>Reserved.</td></tr><tr><td>4h</td><td>WT or write through.</td></tr><tr><td>5h</td><td>WP or write protect.</td></tr><tr><td>6h</td><td>WB or write back.</td></tr><tr><td>7h</td><td>Reserved.</td></tr></table></td></tr>
<tr><td>23:21</td><td>Reserved.</td></tr>
<tr><td rowspan="2">20</td><td>RdDram_4K_CA000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">19</td><td>WrDram_4K_CA000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">18:16</td><td>MemType_4K_CA000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:<table><tr><th>Value</th><th>Description</th></tr><tr><td>0h</td><td>UC or uncacheable.</td></tr><tr><td>1h</td><td>WC or write combining.</td></tr><tr><td>3h-2h</td><td>Reserved.</td></tr><tr><td>4h</td><td>WT or write through.</td></tr><tr><td>5h</td><td>WP or write protect.</td></tr><tr><td>6h</td><td>WB or write back.</td></tr><tr><td>7h</td><td>Reserved.</td></tr></table></td></tr>
<tr><td>15:13</td><td>Reserved.</td></tr>
<tr><td rowspan="2">12</td><td>RdDram_4K_C9000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">11</td><td>WrDram_4K_C9000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset:</td></tr>
</table>

<table>
<tr><td></td><td>Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">10:8</td><td>MemType_4K_C9000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>7:5</td><td>Reserved.</td></tr>
<tr><td rowspan="2">4</td><td>RdDram_4K_C8000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM. Address range from C8000 to C8FFF. Core::X86::Msr::SYS_CFG[MtrrFixDramEn,MtrrFixDramModEn] masks reads of the stored value.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">3</td><td>WrDram_4K_C8000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM. Address range from C8000 to C8FFF. Core::X86::Msr::SYS_CFG[MtrrFixDramEn,MtrrFixDramModEn] masks reads of the stored value.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">2:0</td><td>MemType_4K_C8000: memory type. Read-write. Reset: XXXb. Address range from C8000 to C8FFF.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
</table>

MSR0000_026A [Fixed-Size MTRRs] (Core::X86::Msr::MtrrFix_4K_2)

See Core::X86::Msr::MtrrVarBase for general MTRR information. Fixed MTRRs are enabled through Core::X86::Msr::MTRRdefType[MtrrDefTypeFixEn,MtrrDefTypeEn]. For addresses below 1MB, the appropriate Fixed MTRRs override the default access destination. Each fixed MTRR includes two bits, RdDram and WrDram, that determine the destination based on the access type. Writing reserved MemType values causes an error-on-write.

_ccd[7:0]_lthree0_core[7:0]_nSIZE4K2; MSR0000_026A

Bits	Description
63:61	Reserved.
60	**RdDram_4K_D7000**: **read DRAM**. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.
	AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.
59	**WrDram_4K_D7000**: **write DRAM**. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.
	AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.
58:56	**MemType_4K_D7000**: **memory type**. Read-write. Reset: XXXb.

<table>
<tr><td></td><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>55:53</td><td>Reserved.</td></tr>
<tr><td rowspan="2">52</td><td>RdDram_4K_D6000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">51</td><td>WrDram_4K_D6000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">50:48</td><td>MemType_4K_D6000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>47:45</td><td>Reserved.</td></tr>
<tr><td rowspan="2">44</td><td>RdDram_4K_D5000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">43</td><td>WrDram_4K_D5000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">42:40</td><td>MemType_4K_D5000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>39:37</td><td>Reserved.</td></tr>
<tr><td>36</td><td>RdDram_4K_D4000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the</td></tr>
</table>

<table>
<tr><td rowspan="2"></td><td>range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">35</td><td>WrDram_4K_D4000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">34:32</td><td>MemType_4K_D4000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>31:29</td><td>Reserved.</td></tr>
<tr><td rowspan="2">28</td><td>RdDram_4K_D3000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">27</td><td>WrDram_4K_D3000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">26:24</td><td>MemType_4K_D3000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>23:21</td><td>Reserved.</td></tr>
<tr><td rowspan="2">20</td><td>RdDram_4K_D2000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">19</td><td>WrDram_4K_D2000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">18:16</td><td>MemType_4K_D2000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
</table></td></tr>
</table>

<table>
<tr><td></td><td><table>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>15:13</td><td>Reserved.</td></tr>
<tr><td rowspan="2">12</td><td>RdDram_4K_D1000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">11</td><td>WrDram_4K_D1000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">10:8</td><td>MemType_4K_D1000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>7:5</td><td>Reserved.</td></tr>
<tr><td rowspan="2">4</td><td>RdDram_4K_D0000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM. Address range from D0000h to D0FFFh. Core::X86::Msr::SYS_CFG[MtrrFixDramEn,MtrrFixDramModEn] masks reads of the stored value.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">3</td><td>WrDram_4K_D0000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM. Address range from D0000h to D0FFFh. Core::X86::Msr::SYS_CFG[MtrrFixDramEn,MtrrFixDramModEn] masks reads of the stored value.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">2:0</td><td>MemType_4K_D0000: memory type. Read-write. Reset: XXXb. Address range from D0000h to D0FFFh.</td></tr>
<tr><td>ValidValues:<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
</table>

MSR0000_026B [Fixed-Size MTRRs] (Core::X86::Msr::MtrrFix_4K_3)

See Core::X86::Msr::MtrrVarBase for general MTRR information. Fixed MTRRs are enabled through Core::X86::Msr::MTRRdefType[MtrrDefTypeFixEn,MtrrDefTypeEn]. For addresses below 1MB, the appropriate Fixed

<table>
<tr><td colspan="2">MTRRs override the default access destination. Each fixed MTRR includes two bits, RdDram and WrDram, that determine the destination based on the access type. Writing reserved MemType values causes an error-on-write.</td></tr>
<tr><td colspan="2">_ccd[7:0]_lthree0_core[7:0]_nSIZE4K3; MSR0000_026B</td></tr>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>63:61</td><td>Reserved.</td></tr>
<tr><td rowspan="2">60</td><td>RdDram_4K_DF000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">59</td><td>WrDram_4K_DF000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">58:56</td><td>MemType_4K_DF000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>55:53</td><td>Reserved.</td></tr>
<tr><td rowspan="2">52</td><td>RdDram_4K_DE000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">51</td><td>WrDram_4K_DE000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">50:48</td><td>MemType_4K_DE000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>47:45</td><td>Reserved.</td></tr>
<tr><td rowspan="2">44</td><td>RdDram_4K_DD000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td>43</td><td>WrDram_4K_DD000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
</table>

<table>
<tr><td></td><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">42:40</td><td>MemType_4K_DD000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>39:37</td><td>Reserved.</td></tr>
<tr><td rowspan="2">36</td><td>RdDram_4K_DC000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">35</td><td>WrDram_4K_DC000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">34:32</td><td>MemType_4K_DC000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>31:29</td><td>Reserved.</td></tr>
<tr><td rowspan="2">28</td><td>RdDram_4K_DB000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">27</td><td>WrDram_4K_DB000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">26:24</td><td>MemType_4K_DB000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
</table></td></tr>
</table>

<table>
<tr><td></td><td>7h</td><td>Reserved.</td></tr>
<tr><td>23:21</td><td colspan="2">Reserved.</td></tr>
<tr><td rowspan="2">20</td><td colspan="2">RdDram_4K_DA000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td colspan="2">AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">19</td><td colspan="2">WrDram_4K_DA000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td colspan="2">AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="10">18:16</td><td colspan="2">MemType_4K_DA000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td colspan="2">ValidValues:</td></tr>
<tr><td>Value</td><td>Description</td></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
<tr><td>15:13</td><td colspan="2">Reserved.</td></tr>
<tr><td rowspan="2">12</td><td colspan="2">RdDram_4K_D9000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td colspan="2">AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">11</td><td colspan="2">WrDram_4K_D9000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td colspan="2">AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="10">10:8</td><td colspan="2">MemType_4K_D9000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td colspan="2">ValidValues:</td></tr>
<tr><td>Value</td><td>Description</td></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
<tr><td>7:5</td><td colspan="2">Reserved.</td></tr>
<tr><td rowspan="2">4</td><td colspan="2">RdDram_4K_D8000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM. Address range from D8000h to D8FFFh. Core::X86::Msr::SYS_CFG[MtrrFixDramEn,MtrrFixDramModEn] masks reads of the stored value.</td></tr>
<tr><td colspan="2">AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">3</td><td colspan="2">WrDram_4K_D8000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM. Address range from D8000h to D8FFFh. Core::X86::Msr::SYS_CFG[MtrrFixDramEn,MtrrFixDramModEn] masks reads of the stored value.</td></tr>
<tr><td colspan="2">AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset:</td></tr>
</table>

	Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.
2:0	**MemType_4K_D8000**: **memory type**. Read-write. Reset: XXXb. Address range from D8000h to D8FFFh.
	ValidValues:

Value	Description
0h	UC or uncacheable.
1h	WC or write combining.
3h-2h	Reserved.
4h	WT or write through.
5h	WP or write protect.
6h	WB or write back.
7h	Reserved.

MSR0000_026C [Fixed-Size MTRRs] (Core::X86::Msr::MtrrFix_4K_4)

See Core::X86::Msr::MtrrVarBase for general MTRR information. Fixed MTRRs are enabled through Core::X86::Msr::MTRRdefType[MtrrDefTypeFixEn,MtrrDefTypeEn]. For addresses below 1MB, the appropriate Fixed MTRRs override the default access destination. Each fixed MTRR includes two bits, RdDram and WrDram, that determine the destination based on the access type. Writing reserved MemType values causes an error-on-write.

_ccd[7:0]_lthree0_core[7:0]_nSIZE4K4; MSR0000_026C

Bits	Description
63:61	Reserved.
60	**RdDram_4K_E7000**: **read DRAM**. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.
	AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.
59	**WrDram_4K_E7000**: **write DRAM**. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.
	AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.
58:56	**MemType_4K_E7000**: **memory type**. Read-write. Reset: XXXb.
	ValidValues:

Value	Description
0h	UC or uncacheable.
1h	WC or write combining.
3h-2h	Reserved.
4h	WT or write through.
5h	WP or write protect.
6h	WB or write back.
7h	Reserved.

Bits	Description
55:53	Reserved.
52	**RdDram_4K_E6000**: **read DRAM**. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.
	AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.
51	**WrDram_4K_E6000**: **write DRAM**. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.
	AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.
50:48	**MemType_4K_E6000**: **memory type**. Read-write. Reset: XXXb.
	ValidValues:

Value	Description

<table>
<tr><td></td><td><table>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>47:45</td><td>Reserved.</td></tr>
<tr><td>44</td><td>RdDram_4K_E5000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td></td><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td>43</td><td>WrDram_4K_E5000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td></td><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td>42:40</td><td>MemType_4K_E5000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td></td><td>ValidValues:<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>39:37</td><td>Reserved.</td></tr>
<tr><td>36</td><td>RdDram_4K_E4000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td></td><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td>35</td><td>WrDram_4K_E4000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td></td><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td>34:32</td><td>MemType_4K_E4000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td></td><td>ValidValues:<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>31:29</td><td>Reserved.</td></tr>
<tr><td>28</td><td>RdDram_4K_E3000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td></td><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset:</td></tr>
</table>

<table>
<tr><td></td><td>Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">27</td><td>WrDram_4K_E3000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">26:24</td><td>MemType_4K_E3000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>23:21</td><td>Reserved.</td></tr>
<tr><td rowspan="2">20</td><td>RdDram_4K_E2000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">19</td><td>WrDram_4K_E2000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">18:16</td><td>MemType_4K_E2000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>15:13</td><td>Reserved.</td></tr>
<tr><td rowspan="2">12</td><td>RdDram_4K_E1000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">11</td><td>WrDram_4K_E1000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">10:8</td><td>MemType_4K_E1000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
</table></td></tr>
</table>

4h	WT or write through.
5h	WP or write protect.
6h	WB or write back.
7h	Reserved.

7:5	Reserved.
4	**RdDram_4K_E0000**: **read DRAM**. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM. Address range from E0000h to E0FFFh. Core::X86::Msr::SYS_CFG[MtrrFixDramEn,MtrrFixDramModEn] masks reads of the stored value.
	AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.
3	**WrDram_4K_E0000**: **write DRAM**. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM. Address range from E0000h to E0FFFh. Core::X86::Msr::SYS_CFG[MtrrFixDramEn,MtrrFixDramModEn] masks reads of the stored value.
	AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.
2:0	**MemType_4K_E0000**: **memory type**. Read-write. Reset: XXXb. Address range from E0000h to E0FFFh.
	ValidValues:

Value	Description
0h	UC or uncacheable.
1h	WC or write combining.
3h-2h	Reserved.
4h	WT or write through.
5h	WP or write protect.
6h	WB or write back.
7h	Reserved.

MSR0000_026D [Fixed-Size MTRRs] (Core::X86::Msr::MtrrFix_4K_5)

See Core::X86::Msr::MtrrVarBase for general MTRR information. Fixed MTRRs are enabled through Core::X86::Msr::MTRRdefType[MtrrDefTypeFixEn,MtrrDefTypeEn]. For addresses below 1MB, the appropriate Fixed MTRRs override the default access destination. Each fixed MTRR includes two bits, RdDram and WrDram, that determine the destination based on the access type. Writing reserved MemType values causes an error-on-write.

_ccd[7:0]_lthree0_core[7:0]_nSIZE4K5; MSR0000_026D

Bits	Description
63:61	Reserved.
60	**RdDram_4K_EF000**: **read DRAM**. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.
	AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.
59	**WrDram_4K_EF000**: **write DRAM**. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.
	AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.
58:56	**MemType_4K_EF000**: **memory type**. Read-write. Reset: XXXb.
	ValidValues:

Value	Description
0h	UC or uncacheable.
1h	WC or write combining.
3h-2h	Reserved.
4h	WT or write through.
5h	WP or write protect.

<table>
<tr><td></td><td><table><tr><td>6h</td><td>WB or write back.</td></tr><tr><td>7h</td><td>Reserved.</td></tr></table></td></tr>
<tr><td>55:53</td><td>Reserved.</td></tr>
<tr><td>52</td><td>RdDram_4K_EE000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td></td><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td>51</td><td>WrDram_4K_EE000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td></td><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td>50:48</td><td>MemType_4K_EE000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td></td><td>ValidValues:<table><tr><th>Value</th><th>Description</th></tr><tr><td>0h</td><td>UC or uncacheable.</td></tr><tr><td>1h</td><td>WC or write combining.</td></tr><tr><td>3h-2h</td><td>Reserved.</td></tr><tr><td>4h</td><td>WT or write through.</td></tr><tr><td>5h</td><td>WP or write protect.</td></tr><tr><td>6h</td><td>WB or write back.</td></tr><tr><td>7h</td><td>Reserved.</td></tr></table></td></tr>
<tr><td>47:45</td><td>Reserved.</td></tr>
<tr><td>44</td><td>RdDram_4K_ED000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td></td><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td>43</td><td>WrDram_4K_ED000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td></td><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td>42:40</td><td>MemType_4K_ED000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td></td><td>ValidValues:<table><tr><th>Value</th><th>Description</th></tr><tr><td>0h</td><td>UC or uncacheable.</td></tr><tr><td>1h</td><td>WC or write combining.</td></tr><tr><td>3h-2h</td><td>Reserved.</td></tr><tr><td>4h</td><td>WT or write through.</td></tr><tr><td>5h</td><td>WP or write protect.</td></tr><tr><td>6h</td><td>WB or write back.</td></tr><tr><td>7h</td><td>Reserved.</td></tr></table></td></tr>
<tr><td>39:37</td><td>Reserved.</td></tr>
<tr><td>36</td><td>RdDram_4K_EC000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td></td><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td>35</td><td>WrDram_4K_EC000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td></td><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
</table>

<table>
<tr><td rowspan="2">34:32</td><td>MemType_4K_EC000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:<table><tr><th>Value</th><th>Description</th></tr><tr><td>0h</td><td>UC or uncacheable.</td></tr><tr><td>1h</td><td>WC or write combining.</td></tr><tr><td>3h-2h</td><td>Reserved.</td></tr><tr><td>4h</td><td>WT or write through.</td></tr><tr><td>5h</td><td>WP or write protect.</td></tr><tr><td>6h</td><td>WB or write back.</td></tr><tr><td>7h</td><td>Reserved.</td></tr></table></td></tr>
<tr><td>31:29</td><td>Reserved.</td></tr>
<tr><td rowspan="2">28</td><td>RdDram_4K_EB000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">27</td><td>WrDram_4K_EB000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">26:24</td><td>MemType_4K_EB000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:<table><tr><th>Value</th><th>Description</th></tr><tr><td>0h</td><td>UC or uncacheable.</td></tr><tr><td>1h</td><td>WC or write combining.</td></tr><tr><td>3h-2h</td><td>Reserved.</td></tr><tr><td>4h</td><td>WT or write through.</td></tr><tr><td>5h</td><td>WP or write protect.</td></tr><tr><td>6h</td><td>WB or write back.</td></tr><tr><td>7h</td><td>Reserved.</td></tr></table></td></tr>
<tr><td>23:21</td><td>Reserved.</td></tr>
<tr><td rowspan="2">20</td><td>RdDram_4K_EA000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">19</td><td>WrDram_4K_EA000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">18:16</td><td>MemType_4K_EA000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:<table><tr><th>Value</th><th>Description</th></tr><tr><td>0h</td><td>UC or uncacheable.</td></tr><tr><td>1h</td><td>WC or write combining.</td></tr><tr><td>3h-2h</td><td>Reserved.</td></tr><tr><td>4h</td><td>WT or write through.</td></tr><tr><td>5h</td><td>WP or write protect.</td></tr><tr><td>6h</td><td>WB or write back.</td></tr><tr><td>7h</td><td>Reserved.</td></tr></table></td></tr>
<tr><td>15:13</td><td>Reserved.</td></tr>
</table>

<table>
<tr><td rowspan="2">12</td><td>RdDram_4K_E9000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">11</td><td>WrDram_4K_E9000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">10:8</td><td>MemType_4K_E9000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>7:5</td><td>Reserved.</td></tr>
<tr><td rowspan="2">4</td><td>RdDram_4K_E8000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM. Address range from E8000h to E8FFFh. Core::X86::Msr::SYS_CFG[MtrrFixDramEn,MtrrFixDramModEn] masks reads of the stored value.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">3</td><td>WrDram_4K_E8000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM. Address range from E8000h to E8FFFh. Core::X86::Msr::SYS_CFG[MtrrFixDramEn,MtrrFixDramModEn] masks reads of the stored value.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">2:0</td><td>MemType_4K_E8000: memory type. Read-write. Reset: XXXb. Address range from E8000h to E8FFFh.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
</table>

MSR0000_026E [Fixed-Size MTRRs] (Core::X86::Msr::MtrrFix_4K_6)

See Core::X86::Msr::MtrrVarBase for general MTRR information. Fixed MTRRs are enabled through Core::X86::Msr::MTRRdefType[MtrrDefTypeFixEn,MtrrDefTypeEn]. For addresses below 1MB, the appropriate Fixed MTRRs override the default access destination. Each fixed MTRR includes two bits, RdDram and WrDram, that determine the destination based on the access type. Writing reserved MemType values causes an error-on-write.

_ccd[7:0]_lthree0_core[7:0]_nSIZE4K6; MSR0000_026E

Bits	Description
63:61	Reserved.
60	**RdDram_4K_F7000**: **read DRAM**. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.

<table>
<tr><td></td><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">59</td><td>WrDram_4K_F7000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">58:56</td><td>MemType_4K_F7000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>55:53</td><td>Reserved.</td></tr>
<tr><td rowspan="2">52</td><td>RdDram_4K_F6000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">51</td><td>WrDram_4K_F6000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">50:48</td><td>MemType_4K_F6000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>47:45</td><td>Reserved.</td></tr>
<tr><td rowspan="2">44</td><td>RdDram_4K_F5000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">43</td><td>WrDram_4K_F5000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">42:40</td><td>MemType_4K_F5000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
</table></td></tr>
</table>

<table>
<tr><td></td><td><table>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>39:37</td><td>Reserved.</td></tr>
<tr><td>36</td><td>RdDram_4K_F4000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td></td><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td>35</td><td>WrDram_4K_F4000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td></td><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td>34:32</td><td>MemType_4K_F4000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td></td><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>31:29</td><td>Reserved.</td></tr>
<tr><td>28</td><td>RdDram_4K_F3000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td></td><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td>27</td><td>WrDram_4K_F3000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td></td><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td>26:24</td><td>MemType_4K_F3000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td></td><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>23:21</td><td>Reserved.</td></tr>
<tr><td>20</td><td>RdDram_4K_F2000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td></td><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td>19</td><td>WrDram_4K_F2000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to</td></tr>
</table>

<table>
<tr><td rowspan="2"></td><td>the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">18:16</td><td>MemType_4K_F2000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>15:13</td><td>Reserved.</td></tr>
<tr><td rowspan="2">12</td><td>RdDram_4K_F1000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">11</td><td>WrDram_4K_F1000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">10:8</td><td>MemType_4K_F1000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>7:5</td><td>Reserved.</td></tr>
<tr><td rowspan="2">4</td><td>RdDram_4K_F0000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM. Address range from F0000h to F0FFF. Core::X86::Msr::SYS_CFG[MtrrFixDramEn,MtrrFixDramModEn] masks reads of the stored value.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">3</td><td>WrDram_4K_F0000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM. Address range from F0000h to F0FFF. Core::X86::Msr::SYS_CFG[MtrrFixDramEn,MtrrFixDramModEn] masks reads of the stored value.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">2:0</td><td>MemType_4K_F0000: memory type. Read-write. Reset: XXXb. Address range from F0000h to F0FFFh.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
</table></td></tr>
</table>

4h	WT or write through.
5h	WP or write protect.
6h	WB or write back.
7h	Reserved.

MSR0000_026F [Fixed-Size MTRRs] (Core::X86::Msr::MtrrFix_4K_7)

See Core::X86::Msr::MtrrVarBase for general MTRR information. Fixed MTRRs are enabled through Core::X86::Msr::MTRRdefType[MtrrDefTypeFixEn,MtrrDefTypeEn]. For addresses below 1MB, the appropriate Fixed MTRRs override the default access destination. Each fixed MTRR includes two bits, RdDram and WrDram, that determine the destination based on the access type. Writing reserved MemType values causes an error-on-write.

_ccd[7:0]_lthree0_core[7:0]_nSIZE4K7; MSR0000_026F

Bits	Description
63:61	Reserved.
60	**RdDram_4K_FF000**: **read DRAM**. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.
	AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.
59	**WrDram_4K_FF000**: **write DRAM**. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.
	AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.
58:56	**MemType_4K_FF000**: **memory type**. Read-write. Reset: XXXb.
	ValidValues:

Value	Description
0h	UC or uncacheable.
1h	WC or write combining.
3h-2h	Reserved.
4h	WT or write through.
5h	WP or write protect.
6h	WB or write back.
7h	Reserved.

Bits	Description
55:53	Reserved.
52	**RdDram_4K_FE000**: **read DRAM**. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.
	AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.
51	**WrDram_4K_FE000**: **write DRAM**. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.
	AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.
50:48	**MemType_4K_FE000**: **memory type**. Read-write. Reset: XXXb.
	ValidValues:

Value	Description
0h	UC or uncacheable.
1h	WC or write combining.
3h-2h	Reserved.
4h	WT or write through.
5h	WP or write protect.
6h	WB or write back.
7h	Reserved.

<table>
<tr><td>47:45</td><td>Reserved.</td></tr>
<tr><td rowspan="2">44</td><td>RdDram_4K_FD000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">43</td><td>WrDram_4K_FD000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">42:40</td><td>MemType_4K_FD000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>39:37</td><td>Reserved.</td></tr>
<tr><td rowspan="2">36</td><td>RdDram_4K_FC000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">35</td><td>WrDram_4K_FC000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">34:32</td><td>MemType_4K_FC000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>31:29</td><td>Reserved.</td></tr>
<tr><td rowspan="2">28</td><td>RdDram_4K_FB000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">27</td><td>WrDram_4K_FB000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">26:24</td><td>MemType_4K_FB000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:</td></tr>
</table>

<table>
<tr><td></td><td><table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>23:21</td><td>Reserved.</td></tr>
<tr><td rowspan="2">20</td><td>RdDram_4K_FA000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">19</td><td>WrDram_4K_FA000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">18:16</td><td>MemType_4K_FA000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>15:13</td><td>Reserved.</td></tr>
<tr><td rowspan="2">12</td><td>RdDram_4K_F9000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">11</td><td>WrDram_4K_F9000: write DRAM. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</td></tr>
<tr><td>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td rowspan="2">10:8</td><td>MemType_4K_F9000: memory type. Read-write. Reset: XXXb.</td></tr>
<tr><td>ValidValues:<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>UC or uncacheable.</td></tr>
<tr><td>1h</td><td>WC or write combining.</td></tr>
<tr><td>3h-2h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>WT or write through.</td></tr>
<tr><td>5h</td><td>WP or write protect.</td></tr>
<tr><td>6h</td><td>WB or write back.</td></tr>
<tr><td>7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>7:5</td><td>Reserved.</td></tr>
<tr><td>4</td><td>RdDram_4K_F8000: read DRAM. 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM. Address range from F8000h to F8FFFh.</td></tr>
</table>

	Core::X86::Msr::SYS_CFG[MtrrFixDramEn,MtrrFixDramModEn] masks reads of the stored value.
	AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.
3	**WrDram_4K_F8000**: **write DRAM**. 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM. Address range from F8000h to F8FFFh. Core::X86::Msr::SYS_CFG[MtrrFixDramEn,MtrrFixDramModEn] masks reads of the stored value.
	AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.
2:0	**MemType_4K_F8000**: **memory type**. Read-write. Reset: XXXb. Address range from F8000h to F8FFFh.
	ValidValues:

Value	Description
0h	UC or uncacheable.
1h	WC or write combining.
3h-2h	Reserved.
4h	WT or write through.
5h	WP or write protect.
6h	WB or write back.
7h	Reserved.

MSR0000_0277 [Page Attribute Table] (Core::X86::Msr::PAT)

This register specifies the memory type based on the PAT, PCD, and PWT bits in the virtual address page tables.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_0277

Bits	Description
63:59	Reserved.
58:56	**PA7MemType**. Read-write. Reset: 0h. Default UC. MemType for {PAT, PCD, PWT} = 7h.
	ValidValues:

Value	Description
0h	UC or uncacheable.
1h	WC or write combining.
3h-2h	Reserved.
4h	WT or write through.
5h	WP or write protect.
6h	WB or write back.
7h	Reserved.

Bits	Description
55:51	Reserved.
50:48	**PA6MemType**. Read-write. Reset: 7h. Default UC. MemType for {PAT, PCD, PWT} = 6h.
	ValidValues:

Value	Description
0h	UC or uncacheable.
1h	WC or write combining.
3h-2h	Reserved.
4h	WT or write through.
5h	WP or write protect.
6h	WB or write back.
7h	Reserved.

Bits	Description
47:43	Reserved.
42:40	**PA5MemType**. Read-write. Reset: 4h. Default WT. MemType for {PAT, PCD, PWT} = 5h.
	ValidValues:

Value	Description

0h	UC or uncacheable.
1h	WC or write combining.
3h-2h	Reserved.
4h	WT or write through.
5h	WP or write protect.
6h	WB or write back.
7h	Reserved.

39:35 Reserved.

34:32 **PA4MemType**. Read-write. Reset: 6h. Default WB. MemType for {PAT, PCD, PWT} = 4h.

ValidValues:

Value	Description
0h	UC or uncacheable.
1h	WC or write combining.
3h-2h	Reserved.
4h	WT or write through.
5h	WP or write protect.
6h	WB or write back.
7h	Reserved.

31:27 Reserved.

26:24 **PA3MemType**. Read-write. Reset: 0h. Default UC. MemType for {PAT, PCD, PWT} = 3h.

ValidValues:

Value	Description
0h	UC or uncacheable.
1h	WC or write combining.
3h-2h	Reserved.
4h	WT or write through.
5h	WP or write protect.
6h	WB or write back.
7h	Reserved.

23:19 Reserved.

18:16 **PA2MemType**. Read-write. Reset: 7h. Default UC. MemType for {PAT, PCD, PWT} = 2h.

ValidValues:

Value	Description
0h	UC or uncacheable.
1h	WC or write combining.
3h-2h	Reserved.
4h	WT or write through.
5h	WP or write protect.
6h	WB or write back.
7h	Reserved.

15:11 Reserved.

10:8 **PA1MemType**. Read-write. Reset: 4h. Default WT. MemType for {PAT, PCD, PWT} = 1h.

ValidValues:

Value	Description
0h	UC or uncacheable.
1h	WC or write combining.
3h-2h	Reserved.
4h	WT or write through.

Bits	Description
	5h: WP or write protect. 6h: WB or write back. 7h: Reserved.
7:3	Reserved.
2:0	**PA0MemType**. Read-write. Reset: 6h. MemType for {PAT, PCD, PWT} = 0h.
	ValidValues:

Value	Description
0h	UC or uncacheable.
1h	WC or write combining.
3h-2h	Reserved.
4h	WT or write through.
5h	WP or write protect.
6h	WB or write back.
7h	Reserved.

MSR0000_02FF [MTRR Default Memory Type] (Core::X86::Msr::MTRRdefType)

See Core::X86::Msr::MtrrVarBase for general MTRR information.

_ccd[7:0]_lthree0_core[7:0]; MSR0000_02FF

Bits	Description
63:12	Reserved.
11	**MtrrDefTypeEn**: **variable and fixed MTRR enable**. Read-write. Reset: 0. 0=Fixed and variable MTRRs are not enabled. 1=Core::X86::Msr::MtrrVarBase, and Core::X86::Msr::MtrrFix_64K through Core::X86::Msr::MtrrFix_4K_7 are enabled.
10	**MtrrDefTypeFixEn**: **fixed MTRR enable**. Read-write. Reset: 0. 0=Core::X86::Msr::MtrrFix_64K through Core::X86::Msr::MtrrFix_4K_7 are not enabled. 1=Core::X86::Msr::MtrrFix_64K through Core::X86::Msr::MtrrFix_4K_7 are enabled. This field is ignored (and the fixed MTRRs are not enabled) if Core::X86::Msr::MTRRdefType[MtrrDefTypeEn] == 0.
9:8	Reserved.
7:0	**MemType**: **memory type**. Read-write. Reset: 00h.
	Description: If MtrrDefTypeEn == 1 then MemType specifies the memory type for memory space that is not specified by either the fixed or variable range MTRRs. If MtrrDefTypeEn == 0 then the default memory type for all of memory is UC. Valid encodings are {00000b, Core::X86::Msr::MtrrFix_64K through Core::X86::Msr::MtrrFix_4K_7[2:0]}. Other write values cause a GP(0).

MSR0000_06A0 [User CET] (Core::X86::Msr::U_CET)

Read-write. Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_06A0

Bits	Description
63:2	Reserved.
1	**WRSHSTKEN**. Read-write. Reset: 0. Enables the WRSS instruction in User Mode.
0	**SHSTKEN**. Read-write. Reset: 0. When Set Shadow stack is enabled in User mode.

MSR0000_06A2 [Supervisor CET] (Core::X86::Msr::S_CET)

Read-write. Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_06A2

Bits	Description
63:2	Reserved.
1	**WRSHSTKEN**. Read-write. Reset: 0. Enables the WRSS instruction in Supervisor Mode.

0	**SHSTKEN**. Read-write. Reset: 0. When Set Shadow stack is enabled in Supervisor mode.

MSR0000_06A4 [PL0 Shadow Stack Pointer] (Core::X86::Msr::PL0Ssp)

Read-write. Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_06A4

Bits	Description
63:2	**UserLinAddress**: **PL0 user top of SSP**. Read-write. Reset: 0000_0000_0000_0000h. UserLinAddress[63:32] must be zero in 32bit mode.
1:0	Reserved.

MSR0000_06A5 [PL1 Shadow Stack Pointer] (Core::X86::Msr::PL1Ssp)

Read-write. Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_06A5

Bits	Description
63:2	**UserLinAddress**: **PL1 user top of SSP**. Read-write. Reset: 0000_0000_0000_0000h. UserLinAddress[63:32] must be zero in 32bit mode.
1:0	Reserved.

MSR0000_06A6 [PL2 Shadow Stack Pointer] (Core::X86::Msr::PL2Ssp)

Read-write. Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_06A6

Bits	Description
63:2	**UserLinAddress**: **PL2 user top of SSP**. Read-write. Reset: 0000_0000_0000_0000h. UserLinAddress[63:32] must be zero in 32bit mode.
1:0	Reserved.

MSR0000_06A7 [PL3 Shadow Stack Pointer] (Core::X86::Msr::PL3Ssp)

Read-write. Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_06A7

Bits	Description
63:2	**UserLinAddress**: **PL3 user top of SSP**. Read-write. Reset: 0000_0000_0000_0000h. UserLinAddress[63:32] must be zero in 32bit mode.
1:0	Reserved.

MSR0000_06A8 [Interrupt SSP Table Address] (Core::X86::Msr::IstSspAddr)

Read-write. Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_06A8

Bits	Description
63:0	**IntrLinTableAddress**. Read-write. Reset: 0000_0000_0000_0000h. Shadow Stack Pointer interrupt table.

MSR0000_0802 [APIC ID] (Core::X86::Msr::APIC_ID)

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_0802

Bits	Description
63:32	Reserved.
31:0	**ApicId[31:0]**: **APIC ID[31:0]**. Reset: XXXX_XXXXh. Local x2APIC ID register.
	AccessType: X2APICEN ? Read-only,Error-on-write : Error-on-read,Error-on-write.

MSR0000_0803 [APIC Version] (Core::X86::Msr::ApicVersion)

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_0803

Bits	Description
63:32	Reserved.
31	**ExtApicSpace**: **extended APIC register space present**. Reset: 1. 1=Indicates the presence of extended APIC register space starting at Core::X86::Msr::ExtendedApicFeature.
	AccessType: X2APICEN ? Read-only,Error-on-write : Error-on-read,Error-on-write.
30:25	Reserved.
24	**DirectedEoiSupport**: **directed EOI support**. Reset: 0. 0=Directed EOI capability not supported. 1=Directed EOI capability supported.
	AccessType: X2APICEN ? Read-only,Error-on-write : Error-on-read,Error-on-write.
23:16	**MaxLvtEntry**. Reset: XXh. Specifies the number of entries in the local vector table minus one.
	AccessType: X2APICEN ? Read-only,Error-on-write : Error-on-read,Error-on-write.
15:8	Reserved.
7:0	**Version**. Reset: 10h. Indicates the version number of this APIC implementation.
	AccessType: X2APICEN ? Read-only,Error-on-write : Error-on-read,Error-on-write.

MSR0000_0808 [Task Priority] (Core::X86::Msr::TPR)

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_0808

Bits	Description
63:8	Reserved.
7:0	**Priority**. Reset: 00h. This field is assigned by software to set a threshold priority at which the core is interrupted.
	AccessType: X2APICEN ? Read-write,Volatile : Error-on-read,Error-on-write.

MSR0000_0809 [Arbitration Priority] (Core::X86::Msr::ArbitrationPriority)

Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_0809

Bits	Description
63:8	Reserved.
7:0	**Priority**. Reset: 00h. Indicates the current priority for a pending interrupt, or a task or interrupt being serviced by the core. The priority is used to arbitrate between cores to determine which accepts a lowest-priority interrupt request.
	AccessType: X2APICEN ? Read-only,Error-on-write,Volatile : Error-on-read,Error-on-write.

MSR0000_080A [Processor Priority] (Core::X86::Msr::ProcessorPriority)

Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_080A

Bits	Description
63:8	Reserved.
7:0	**Priority**. Reset: 00h. Indicates the core's current priority servicing a task or interrupt, and is used to determine if any pending interrupts should be serviced. It is the higher value of the task priority value and the current highest in-service interrupt.
	AccessType: X2APICEN ? Read-only,Error-on-write,Volatile : Error-on-read,Error-on-write.

MSR0000_080B [End Of Interrupt] (Core::X86::Msr::EOI)

Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_080B

Bits	Description
63:0	**EOI**. Reset: 0000_0000_0000_0000h. A write zero to this field indicates the end of interrupt processing the currently in service interrupt.
	AccessType: X2APICEN ? Write-0-only,Error-on-read,Error-on-write-1 : Error-on-read,Error-on-write.

MSR0000_080D [Logical Destination Register] (Core::X86::Msr::LDR)

Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_080D

Bits	Description
63:32	Reserved.
31:16	**ClusterDestination**. Reset: 0000h. Specifies cluster's destination identification.
	AccessType: X2APICEN ? Read-only,Error-on-write : Error-on-read,Error-on-write.
15:0	**LogicalDestination**. Reset: 0000h. Specifies one of up to sixteen x2APICs within the cluster specified by ClusterDestination.
	AccessType: X2APICEN ? Read-only,Error-on-write : Error-on-read,Error-on-write.
	ValidValues:

Bit	Description
[0]	x2APIC 0
[1]	x2APIC 1
[2]	x2APIC 2
[3]	x2APIC 3
[4]	x2APIC 4
[5]	x2APIC 5
[6]	x2APIC 6
[7]	x2APIC 7
[8]	x2APIC 8
[9]	x2APIC 9
[10]	x2APIC 10
[11]	x2APIC 11
[12]	x2APIC 12
[13]	x2APIC 13
[14]	x2APIC 14
[15]	x2APIC 15

MSR0000_080F [Spurious Interrupt Vector] (Core::X86::Msr::SVR)

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_080F

Bits	Description
63:10	Reserved.
9	**FocusDisable**. Reset: 0. 1=Disable focus core checking during lowest-priority arbitrated interrupts.
	AccessType: X2APICEN ? Read-write : Error-on-read,Error-on-write.
8	**APICSWEn**: **APIC software enable**. Reset: 0. All LVT entry mask bits are set and cannot be cleared.
	AccessType: X2APICEN ? Read-write : Error-on-read,Error-on-write.
7:0	**Vector**. Reset: FFh. The vector that is sent to the core in the event of a spurious interrupt.
	AccessType: X2APICEN ? Read-write : Error-on-read,Error-on-write.

MSR0000_081[0...7] [In Service Register] (Core::X86::Msr::ISR)

Reset: 0000_0000_0000_0000h.

Interrupt In Service status bits [255:0] accessible through 8 ISR registers.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_nISR0_aliasMSR; MSR0000_0810
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_nISR1_aliasMSR; MSR0000_0811
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_nISR2_aliasMSR; MSR0000_0812
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_nISR3_aliasMSR; MSR0000_0813
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_nISR4_aliasMSR; MSR0000_0814
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_nISR5_aliasMSR; MSR0000_0815
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_nISR6_aliasMSR; MSR0000_0816
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_nISR7_aliasMSR; MSR0000_0817

Bits	Description

63:32	Reserved.
31:0	**InServiceBits**. Reset: 0000_0000h. These bits are set when the corresponding interrupt is being serviced by the core.
	AccessType: X2APICEN ? Read-only,Error-on-write,Volatile : Error-on-read,Error-on-write.

MSR0000_081[8...F] [**Trigger Mode Register**] **(Core::X86::Msr::TMR)**

Reset: 0000_0000_0000_0000h.

Trigger Mode status bits [255:0] accessible through 8 TMR registers.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_nTMR0_aliasMSR; MSR0000_0818
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_nTMR1_aliasMSR; MSR0000_0819
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_nTMR2_aliasMSR; MSR0000_081A
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_nTMR3_aliasMSR; MSR0000_081B
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_nTMR4_aliasMSR; MSR0000_081C
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_nTMR5_aliasMSR; MSR0000_081D
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_nTMR6_aliasMSR; MSR0000_081E
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_nTMR7_aliasMSR; MSR0000_081F

<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>63:32</td><td>Reserved.</td></tr>
<tr><td>31:0</td><td>TriggerModeBits. Reset: 0000_0000h. The corresponding trigger mode bit is updated when an interrupt is accepted.</td></tr>
<tr><td></td><td>AccessType: X2APICEN ? Read-only,Error-on-write,Volatile : Error-on-read,Error-on-write.</td></tr>
<tr><td></td><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0</td><td>Edge-triggered interrupt</td></tr>
<tr><td>1</td><td>Level-triggered interrupt</td></tr>
</table></td></tr>
</table>

MSR0000_082[0...7] [**Interrupt Request Register**] **(Core::X86::Msr::IRR)**

Reset: 0000_0000_0000_0000h.

Interrupt Request status bits [255:0] accessible through 8 IRR registers.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_nIRR0_aliasMSR; MSR0000_0820
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_nIRR1_aliasMSR; MSR0000_0821
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_nIRR2_aliasMSR; MSR0000_0822
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_nIRR3_aliasMSR; MSR0000_0823
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_nIRR4_aliasMSR; MSR0000_0824
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_nIRR5_aliasMSR; MSR0000_0825
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_nIRR6_aliasMSR; MSR0000_0826
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_nIRR7_aliasMSR; MSR0000_0827

Bits	Description
63:32	Reserved.
31:0	**RequestBits**. Reset: 0000_0000h. The corresponding request bit is set when the an interrupt is accepted by the x2APIC.
	AccessType: X2APICEN ? Read-only,Error-on-write,Volatile : Error-on-read,Error-on-write.

MSR0000_0828 [**Error Status Register**] **(Core::X86::Msr::ESR)**

Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_0828

Bits	Description
63:8	Reserved.
7	**IllegalRegAddr**: **illegal register address**. Reset: 0. This bit indicates that an access to a nonexistent register location within this APIC was attempted. Can only be set in xAPIC mode.
	AccessType: X2APICEN ? Read,Write-0-only,Error-on-write-1,Volatile : Error-on-read,Error-on-write.
6	**RcvdIllegalVector**: **received illegal vector**. Reset: 0. This bit indicates that this APIC has received a message with an illegal vector (00h to 0Fh for fixed and lowest priority interrupts).
	AccessType: X2APICEN ? Read,Write-0-only,Error-on-write-1,Volatile : Error-on-read,Error-on-write.

5	**SentIllegalVector**. Reset: 0. This bit indicates that this x2APIC attempted to send a message with an illegal vector (00h to 0Fh for fixed and lowest priority interrupts).
	AccessType: X2APICEN ? Read,Write-0-only,Error-on-write-1,Volatile : Error-on-read,Error-on-write.
4	Reserved.
3	**RcvAcceptError**: **receive accept error**. Reset: 0. This bit indicates that a message received by this APIC was not accepted by this or any other x2APIC.
	AccessType: X2APICEN ? Read,Write-0-only,Error-on-write-1,Volatile : Error-on-read,Error-on-write.
2	**SendAcceptError**. Reset: 0. This bit indicates that a message sent by this APIC was not accepted by any x2APIC.
	AccessType: X2APICEN ? Read,Write-0-only,Error-on-write-1,Volatile : Error-on-read,Error-on-write.
1:0	Reserved.

MSR0000_0830 [Interrupt Command] (Core::X86::Msr::InterruptCommand)

Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_0830

<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>63:32</td><td>DestinationField. Reset: 0000_0000h. The destination encoding used when Core::X86::Msr::InterruptCommand[DestShrthnd] is 00b.</td></tr>
<tr><td></td><td>AccessType: X2APICEN ? Read-write : Error-on-read,Error-on-write.</td></tr>
<tr><td>31:20</td><td>Reserved.</td></tr>
<tr><td>19:18</td><td>DestShrthnd: destination shorthand. Reset: 0h. Provides a quick way to specify a destination for a message. If all including self or all excluding self is used, then destination mode is ignored and physical is automatically used.</td></tr>
<tr><td></td><td>AccessType: X2APICEN ? Read-write : Error-on-read,Error-on-write.</td></tr>
<tr><td></td><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>No shorthand (Destination field).</td></tr>
<tr><td>1h</td><td>Self.</td></tr>
<tr><td>2h</td><td>All including self.</td></tr>
<tr><td>3h</td><td>All excluding self. (This sends a message with a destination encoding of all 1s, so if lowest priority is used the message could end up being reflected back to this APIC.)</td></tr>
</table></td></tr>
<tr><td>17:16</td><td>Reserved.</td></tr>
<tr><td>15</td><td>TM: trigger mode. Reset: 0. 0=Edge triggered. 1=Level triggered. Indicates how this interrupt is triggered.</td></tr>
<tr><td></td><td>AccessType: X2APICEN ? Read-write : Error-on-read,Error-on-write.</td></tr>
<tr><td>14</td><td>Level. Reset: 0. 0=Deasserted. 1=Asserted.</td></tr>
<tr><td></td><td>AccessType: X2APICEN ? Read-write : Error-on-read,Error-on-write.</td></tr>
<tr><td>13:12</td><td>Reserved.</td></tr>
<tr><td>11</td><td>DM: destination mode. Reset: 0. 0=Physical. 1=Logical.</td></tr>
<tr><td></td><td>AccessType: X2APICEN ? Read-write : Error-on-read,Error-on-write.</td></tr>
<tr><td>10:8</td><td>MsgType. Reset: 0h. The message types are encoded as follows:</td></tr>
<tr><td></td><td>AccessType: X2APICEN ? Read-write : Error-on-read,Error-on-write.</td></tr>
<tr><td></td><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>Fixed</td></tr>
<tr><td>1h</td><td>Lowest Priority.</td></tr>
<tr><td>2h</td><td>SMI</td></tr>
<tr><td>3h</td><td>Reserved.</td></tr>
<tr><td>4h</td><td>NMI</td></tr>
<tr><td>5h</td><td>INIT</td></tr>
<tr><td>6h</td><td>Startup</td></tr>
</table></td></tr>
</table>

	7h External interrupt.
7:0	**Vector**. Reset: 00h. The vector that is sent for this interrupt source.
	AccessType: X2APICEN ? Read-write : Error-on-read,Error-on-write.

MSR0000_0832 [LVT Timer] (Core::X86::Msr::TimerLvtEntry)

Reset: 0000_0000_0001_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_0832

Bits	Description
63:18	Reserved.
17	**Mode**. Reset: 0. 0=One-shot. 1=Periodic.
	AccessType: X2APICEN ? Read-write : Error-on-read,Error-on-write.
16	**Mask**. Reset: 1. 0=Not masked. 1=Masked.
	AccessType: X2APICEN ? Read-write : Error-on-read,Error-on-write.
15:13	Reserved.
12	**DS**: **interrupt delivery status**. Reset: 0. 0=Idle. 1=Send pending. (Indicates that the interrupt has not yet been accepted by the core.)
	AccessType: X2APICEN ? Read-only,Volatile : Error-on-read,Error-on-write.
11:8	Reserved.
7:0	**Vector**. Reset: 00h. Interrupt vector number.
	AccessType: X2APICEN ? Read-write : Error-on-read,Error-on-write.

MSR0000_0833 [LVT Thermal Sensor] (Core::X86::Msr::ThermalLvtEntry)

Reset: 0000_0000_0001_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_0833

Bits	Description
63:17	Reserved.
16	**Mask**. Reset: 1. 0=Not masked. 1=Masked.
	AccessType: X2APICEN ? Read-write : Error-on-read,Error-on-write.
15:13	Reserved.
12	**DS**: **interrupt delivery status**. Reset: 0. 0=Idle. 1=Send pending. (Indicates that the interrupt has not yet been accepted by the core.)
	AccessType: X2APICEN ? Read-only,Volatile : Error-on-read,Error-on-write.
11	Reserved.
10:8	**MsgType**: **message type**. Reset: 0h. See2.1.14.2.1.14 [Generalized Local Vector Table].
	AccessType: X2APICEN ? Read-write : Error-on-read,Error-on-write.
7:0	**Vector**. Reset: 00h. Interrupt vector number.
	AccessType: X2APICEN ? Read-write : Error-on-read,Error-on-write.

MSR0000_0834 [LVT Performance Monitor] (Core::X86::Msr::PerformanceCounterLvtEntry)

Reset: 0000_0000_0001_0000h.

Interrupts for this local vector table are caused by overflows of:

- Core::X86::Msr::PERF_LEGACY_CTL0..3(Performance Event Select [3:0])].
- Core::X86::Msr::PERF_CTL0..5(Performance Event Select [5:0])].

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_0834

Bits	Description
63:17	Reserved.
16	**Mask**. Reset: 1. 0=Not masked. 1=Masked.
	AccessType: X2APICEN ? Read-write : Error-on-read,Error-on-write.
15:13	Reserved.

12	**DS**: **interrupt delivery status**. Reset: 0. 0=Idle. 1=Send pending. Indicates that the interrupt has not yet been accepted by the core.
	AccessType: X2APICEN ? Read-only,Volatile : Error-on-read,Error-on-write.
11	Reserved.
10:8	**MsgType**: **message type**. Reset: 0h. See2.1.14.2.1.14 [Generalized Local Vector Table].
	AccessType: X2APICEN ? Read-write : Error-on-read,Error-on-write.
7:0	**Vector**. Reset: 00h. Interrupt vector number.
	AccessType: X2APICEN ? Read-write : Error-on-read,Error-on-write.

MSR0000_083[5...6] [LVT LINT[1:0]] (Core::X86::Msr::LVTLINT)

Reset: 0000_0000_0001_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_nLVTLINT0_aliasMSR; MSR0000_0835

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_nLVTLINT1_aliasMSR; MSR0000_0836

Bits	Description
63:17	Reserved.
16	**Mask**. Reset: 1. 0=Not masked. 1=Masked.
	AccessType: X2APICEN ? Read-write : Error-on-read,Error-on-write.
15	**TM**: **trigger mode**. Reset: 0. 0=Edge. 1=Level.
	AccessType: X2APICEN ? Read-write : Error-on-read,Error-on-write.
14	**RmtIRR**. Reset: 0. If trigger mode is level, remote Core::X86::Msr::IRR is set when the interrupt has begun service. Remote Core::X86::Msr::IRR is cleared when the end of interrupt has occurred.
	AccessType: X2APICEN ? Read-only,Volatile : Error-on-read,Error-on-write.
13	Reserved.
12	**DS**: **interrupt delivery status**. Reset: 0. 0=Idle. 1=Send pending. (Indicates that the interrupt has not yet been accepted by the core.)
	AccessType: X2APICEN ? Read-only,Volatile : Error-on-read,Error-on-write.
11	Reserved.
10:8	**MsgType**: **message type**. Reset: 0h. See2.1.14.2.1.14 [Generalized Local Vector Table].
	AccessType: X2APICEN ? Read-write : Error-on-read,Error-on-write.
7:0	**Vector**. Reset: 00h. Interrupt vector number.
	AccessType: X2APICEN ? Read-write : Error-on-read,Error-on-write.

MSR0000_0837 [LVT Error] (Core::X86::Msr::ErrorLvtEntry)

Reset: 0000_0000_0001_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_0837

Bits	Description
63:17	Reserved.
16	**Mask**. Reset: 1. 0=Not masked. 1=Masked.
	AccessType: X2APICEN ? Read-write : Error-on-read,Error-on-write.
15:13	Reserved.
12	**DS**: **interrupt delivery status**. Reset: 0. 0=Idle. 1=Send pending. (Indicates that the interrupt has not yet been accepted by the core.)
	AccessType: X2APICEN ? Read-only,Volatile : Error-on-read,Error-on-write.
11	Reserved.
10:8	**MsgType**: **message type**. Reset: 0h. See 2.1.14.2.1.14 [Generalized Local Vector Table].
	AccessType: X2APICEN ? Read-write : Error-on-read,Error-on-write.
7:0	**Vector**. Reset: 00h. Interrupt vector number.
	AccessType: X2APICEN ? Read-write : Error-on-read,Error-on-write.

MSR0000_0838 [Timer Initial Count] (Core::X86::Msr::TimerInitialCount)

Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_0838

Bits	Description
63:32	Reserved.
31:0	**Count**. Reset: 0000_0000h. The value copied into the current count register when the timer is loaded or reloaded.
	AccessType: X2APICEN ? Read-write : Error-on-read,Error-on-write.

MSR0000_0839 [Timer Current Count] (Core::X86::Msr::TimerCurrentCount)

Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_0839

Bits	Description
63:32	Reserved.
31:0	**Count**. Reset: 0000_0000h. The current value of the counter.
	AccessType: X2APICEN ? Read,Error-on-write,Volatile : Error-on-read,Error-on-write.

MSR0000_083E [Timer Divide Configuration] (Core::X86::Msr::TimerDivideConfiguration)

Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_083E

Bits	Description
63:4	Reserved.
3:0	**Div[3:0]**. Reset: 0h. Div[2] is unused.
	AccessType: X2APICEN ? Read-write : Error-on-read,Error-on-write.
	ValidValues:

Value	Description
0h	Divide by 2.
1h	Divide by 4.
2h	Divide by 8.
3h	Divide by 16.
7h-4h	Reserved.
8h	Divide by 32.
9h	Divide by 64.
Ah	Divide by 128.
Bh	Divide by 1.
Fh-Ch	Reserved.

MSR0000_083F [Self IPI] (Core::X86::Msr::SelfIPI)

Reset: 0000_0000_0000_0000h.

The self IPI register provides a perforamnce optimized path for sending self IPI's. A self IPI is semantically identical to an inter-processor interrupt sent via the ICR, with a Destination Shorthand of Self, Trigger Mode equal to Edge, and a Delivery Mode equal to Fixed.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_083F

Bits	Description
63:8	Reserved.
7:0	**Vector**. Reset: 00h. Interrupt vector number.
	AccessType: X2APICEN ? Write-only,Error-on-read : Error-on-read,Error-on-write.

MSR0000_0840 [Extended APIC Feature] (Core::X86::Msr::ExtendedApicFeature)

Reset: 0000_0000_0004_0007h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_0840

Bits	Description
63:24	Reserved.
23:16	**ExtLvtCount**: **extended local vector table count**. Reset: 04h. This specifies the number of extended LVT

	registers (Core::X86::Msr::ExtendedInterruptLvtEntries) in the local APIC.
	AccessType: X2APICEN ? Read-only,Error-on-write : Error-on-read,Error-on-write.
15:3	Reserved.
2	**ExtApicIdCap**: **extended APIC ID capable**. Reset: 1. 1=The processor is capable of supporting an 8-bit APIC ID, as controlled by Core::X86::Msr::ExtendedApicControl[ExtApicIdEn].
	AccessType: X2APICEN ? Read-only,Error-on-write : Error-on-read,Error-on-write.
1	**SeoiCap**: **specific end of interrupt capable**. Reset: 1. 1=The Core::X86::Msr::SpecificEndOfInterrupt is present.
	AccessType: X2APICEN ? Read-only,Error-on-write : Error-on-read,Error-on-write.
0	**IerCap**: **interrupt enable register capable**. Reset: 1. This bit indicates that the Core::X86::Msr::InterruptEnable0 - 7 are present. See 2.1.14.2.1.8 [Interrupt Masking].
	AccessType: X2APICEN ? Read-only,Error-on-write : Error-on-read,Error-on-write.

MSR0000_0841 [Extended APIC Control] (Core::X86::Msr::ExtendedApicControl)

Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_0841

Bits	Description
63:3	Reserved.
2	**ExtApicIdEn**: **extended APIC ID enable**. Reset: 0. 1=Enable 8-bit APIC ID; Core::X86::Msr::APIC_ID[ApicId[31:0]] supports an 8-bit value; an interrupt broadcast in physical destination mode requires that the IntDest[7:0]=1111_1111b (instead of xxxx_1111b); a match in physical destination mode occurs when (IntDest[7:0] == ApicId[7:0]) instead of (IntDest[3:0] == ApicId[3:0]).
	AccessType: X2APICEN ? Read-write : Error-on-read,Error-on-write.
1	**SeoiEn**. Reset: 0. 1=Enable SEOI generation when a write to Core::X86::Msr::SpecificEndOfInterrupt is received.
	AccessType: X2APICEN ? Read-write : Error-on-read,Error-on-write.
0	**IerEn**. Reset: 0. 1=Enable writes to the interrupt enable registers.
	AccessType: X2APICEN ? Read-write : Error-on-read,Error-on-write.

MSR0000_0842 [Specific End Of Interrupt] (Core::X86::Msr::SpecificEndOfInterrupt)

Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_0842

Bits	Description
63:8	Reserved.
7:0	**EoiVec**: **end of interrupt vector**. Reset: 00h. A write to this field causes an end of interrupt cycle to be performed for the vector specified in this field. The behavior is undefined if no interrupt is pending for the specified interrupt vector.
	AccessType: X2APICEN ? Read-write : Error-on-read,Error-on-write.

MSR0000_0848 [Interrupt Enable 0] (Core::X86::Msr::InterruptEnable0)

Reset: 0000_0000_FFFF_FFFFh.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n0_aliasMSR; MSR0000_0848

Bits	Description
63:32	Reserved.
31:16	**InterruptEnableBits**. Reset: FFFFh. The interrupt enable bits can be used to enable each of the 256 interrupts.
	AccessType: X2APICEN ? Read-write : Error-on-read,Error-on-write.
15:0	Reserved.

MSR0000_084[9...F] [Interrupt Enable 7..1] (Core::X86::Msr::InterruptEnable71)

Reset: 0000_0000_FFFF_FFFFh.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n1_aliasMSR; MSR0000_0849
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n2_aliasMSR; MSR0000_084A
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n3_aliasMSR; MSR0000_084B
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n4_aliasMSR; MSR0000_084C
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n5_aliasMSR; MSR0000_084D
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n6_aliasMSR; MSR0000_084E
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n7_aliasMSR; MSR0000_084F

Bits	Description
63:32	Reserved.
31:0	**InterruptEnableBits**. Reset: FFFF_FFFFh. The interrupt enable bits can be used to enable each of the 256 interrupts.
	AccessType: X2APICEN ? Read-write : Error-on-read,Error-on-write.

MSR0000_085[0...3] [Extended Interrupt Local Vector Table] (Core::X86::Msr::ExtendedInterruptLvtEntries)

Reset: 0000_0000_0001_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n0_aliasMSR; MSR0000_0850
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n1_aliasMSR; MSR0000_0851
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n2_aliasMSR; MSR0000_0852
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n3_aliasMSR; MSR0000_0853

Bits	Description
63:17	Reserved.
16	**Mask**. Reset: 1. 0=Not masked. 1=Masked.
	AccessType: X2APICEN ? Read-write : Error-on-read,Error-on-write.
15:13	Reserved.
12	**DS**: **interrupt delivery status**. Reset: 0. 0=Idle. 1=Send pending. Indicates that the interrupt has not yet been accepted by the core.
	AccessType: X2APICEN ? Read-write,Volatile : Error-on-read,Error-on-write.
11	Reserved.
10:8	**MsgType**: **message type**. Reset: 0h. See2.1.14.2.1.14 [Generalized Local Vector Table].
	AccessType: X2APICEN ? Read-write : Error-on-read,Error-on-write.
7:0	**Vector**. Reset: 00h. Interrupt vector number.
	AccessType: X2APICEN ? Read-write : Error-on-read,Error-on-write.

MSR0000_0C81 [L3 QoS Configuration] (Core::X86::Msr::L3QosCfg1)

_ccd[7:0]_lthree0; MSR0000_0C81

Bits	Description
63:1	Reserved.
0	**CDP**. Read-write. Reset: 0. Code and Data Prioritization Technology enable.

MSR0000_0C8D [Monitoring Event Select] (Core::X86::Msr::QM_EVTSEL)

_ccd[7:0]_lthree0; MSR0000_0C8D

Bits	Description
63:40	Reserved.
39:32	**RMID**. Read-write. Reset: 00h. Resource Monitoring Identifier.
31:8	Reserved.
7:0	**EventId**. Read-write. Reset: 00h. Monitored Event ID.

MSR0000_0C8E [QOS L3 Counter] (Core::X86::Msr::QM_CTR)

Read,Error-on-write. Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0; MSR0000_0C8E

Bits	Description
63	**Error**. Read,Error-on-write. Reset: 0. Unsupported RMID or event type was written to Core::X86::Msr::QM_EVTSEL.
62	**Unavailable**. Read,Error-on-write. Reset: 0. Data for this RMID is not available or not monitored for this

	resource or RMID.
61:0	**RmData**. Read,Error-on-write. Reset: 0000_0000_0000_0000h. Resource Monitored Data.

MSR0000_0C8F [Resource Association] (Core::X86::Msr::PQR_ASSOC)

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_0C8F

Bits	Description
63:36	Reserved.
35:32	**CLOS**. Read-write. Reset: 0h. Class of Service.
31:8	Reserved.
7:0	**RMID**. Read-write. Reset: 00h. Resource Monitoring Identifier.

MSR0000_0C9[0...F] [L3 QOS Allocation Mask] (Core::X86::Msr::L3QosAllocMask)

_ccd[7:0]_lthree0_n0; MSR0000_0C90
_ccd[7:0]_lthree0_n1; MSR0000_0C91
_ccd[7:0]_lthree0_n2; MSR0000_0C92
_ccd[7:0]_lthree0_n3; MSR0000_0C93
_ccd[7:0]_lthree0_n4; MSR0000_0C94
_ccd[7:0]_lthree0_n5; MSR0000_0C95
_ccd[7:0]_lthree0_n6; MSR0000_0C96
_ccd[7:0]_lthree0_n7; MSR0000_0C97
_ccd[7:0]_lthree0_n8; MSR0000_0C98
_ccd[7:0]_lthree0_n9; MSR0000_0C99
_ccd[7:0]_lthree0_n10; MSR0000_0C9A
_ccd[7:0]_lthree0_n11; MSR0000_0C9B
_ccd[7:0]_lthree0_n12; MSR0000_0C9C
_ccd[7:0]_lthree0_n13; MSR0000_0C9D
_ccd[7:0]_lthree0_n14; MSR0000_0C9E
_ccd[7:0]_lthree0_n15; MSR0000_0C9F

Bits	Description
63:16	Reserved.
15:0	**WayMask**. Read-write. Reset: FFFFh. L3 way mask used for allocation control.

MSR0000_0DA0 [Extended Supervisor State] (Core::X86::Msr::XSS)

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_0DA0

Bits	Description
63:13	Reserved.
12	**CET_S**. Read-write. Reset: 0. System Control-flow Enforcement Technology.
11	**CET_U**. Read-write. Reset: 0. User Control-flow Enforcement Technology.
10:0	Reserved.

2.1.16.2 MSRs - MSRC000_xxxx

MSRC000_0080 [Extended Feature Enable] (Core::X86::Msr::EFER)

SKINIT Execution: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC000_0080

Bits	Description
63:19	Reserved.
18	**IntWbinvdEn**. Read-write. Reset: 0. Interruptible wbinvd, wbnoinvd enable.
17	**MCOMMIT**: **enable memory commit instruction**. Read-write. Reset: 0. 0=The MCOMMIT opcode is treated as an undefined opcode. 1=The MCOMMIT instruction is enabled. Enable MCOMMIT instruction. See Core::X86::Cpuid::FeatureExtIdEbx[MCOMMIT].
16	Reserved.

15	**TCE**: **translation cache extension enable**. Read-write. Reset: 0. 1=Translation cache extension is enabled. PDC entries related to the linear address of the INVLPG instruction are invalidated. If this bit is 0 all PDC entries are invalidated by the INVLPG instruction.
14	**FFXSE**: **fast FXSAVE/FRSTOR enable**. Read-write. Reset: 0. 1=Enables the fast FXSAVE/FRSTOR mechanism. A 64-bit operating system may enable the fast FXSAVE/FRSTOR mechanism if (Core::X86::Cpuid::FeatureExtIdEdx[FFXSR] == 1). This bit is set once by the operating system and its value is not changed afterwards.
13	**LMSLE**: **long mode segment limit enable**. Read-only,Error-on-write-1. Reset: Fixed,0. 1=Enables the long mode segment limit check mechanism.
12	**SVME**: **secure virtual machine (SVM) enable**. Reset: Fixed,0. 1=SVM features are enabled.
	AccessType: Core::X86::Msr::VM_CR[SvmeDisable] ? Read-only,Error-on-write-1 : Read-write.
11	**NXE**: **no-execute page enable**. Read-write. Reset: 0. 1=The no-execute page protection feature is enabled.
10	**LMA**: **long mode active**. Read-only. Reset: 0. 1=Indicates that long mode is active. When writing the EFER register the value of this bit must be preserved. Software must read the EFER register to determine the value of LMA, change any other bits as required and then write the EFER register. An attempt to write a value that differs from the state determined by hardware results in a #GP fault.
9	Reserved.
8	**LME**: **long mode enable**. Read-write. Reset: 0. 1=Long mode is enabled.
7:1	Reserved.
0	**SYSCALL**: **system call extension enable**. Read-write. Reset: 0. 1=SYSCALL and SYSRET instructions are enabled. This adds the SYSCALL and SYSRET instructions which can be used in flat addressed operating systems as low latency system calls and returns.

MSRC000_0081 [SYSCALL Target Address] (Core::X86::Msr::STAR)

Read-write. Reset: 0000_0000_0000_0000h.

This register holds the target address used by the SYSCALL instruction and the code and stack segment selector bases used by the SYSCALL and SYSRET instructions.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC000_0081

Bits	Description
63:48	**SysRetSel**. Read-write. Reset: 0000h. SYSRET CS and SS.
47:32	**SysCallSel**. Read-write. Reset: 0000h. SYSCALL CS and SS.
31:0	**Target**. Read-write. Reset: 0000_0000h. SYSCALL target address.

MSRC000_0082 [Long Mode SYSCALL Target Address] (Core::X86::Msr::STAR64)

Read-write. Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC000_0082

Bits	Description
63:0	**LSTAR**: **long mode target address**. Read-write. Reset: 0000_0000_0000_0000h. Target address for 64-bit mode calling programs. The address stored in this register must be in canonical form (if not canonical, a #GP fault occurs).

MSRC000_0083 [Compatibility Mode SYSCALL Target Address] (Core::X86::Msr::STARCOMPAT)

Read-write. Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC000_0083

Bits	Description
63:0	**CSTAR**: **compatibility mode target address**. Read-write. Reset: 0000_0000_0000_0000h. Target address for compatibility mode. The address stored in this register must be in canonical form (if not canonical, a #GP fault occurs).

MSRC000_0084 [SYSCALL Flag Mask] (Core::X86::Msr::SYSCALL_FLAG_MASK)

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC000_0084

Bits	Description

63:32	Reserved.
31:0	**Mask**: **SYSCALL flag mask**. Read-write. Reset: 0000_0000h. This register holds the EFLAGS mask used by the SYSCALL instruction. 1=Clear the corresponding EFLAGS bit when executing the SYSCALL instruction.

MSRC000_00E7 [Read-Only Max Performance Frequency Clock Count] (Core::X86::Msr::MPerfReadOnly)

Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC000_00E7

Bits	Description
63:0	**MPerfReadOnly**: **read-only maximum core clocks counter**. Reset: 0000_0000_0000_0000h. Incremented by hardware at the P0 frequency while the core is in C0. In combination with Core::X86::Msr::APerfReadOnly, this is used to determine the effective frequency of the core. A read of this MSR in guest mode is affected by Core::X86::Msr::TscRateMsr. This field uses software P-state numbering. See Core::X86::Msr::HWCR[EffFreqCntMwait], 2.1.6 [Effective Frequency]. This register is not affected by writes to Core::X86::Msr::MPERF.
	AccessType: Core::X86::Msr::HWCR[EffFreqReadOnlyLock] ? Read,Error-on-write,Volatile : Read-write,Volatile.

MSRC000_00E8 [Read-Only Actual Performance Frequency Clock Count] (Core::X86::Msr::APerfReadOnly)

Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC000_00E8

Bits	Description
63:0	**APerfReadOnly**: **read-only actual core clocks counter**. Reset: 0000_0000_0000_0000h. This register increments in proportion to the actual number of core clocks cycles while the core is in C0. See Core::X86::Msr::MPerfReadOnly. This register is not affected by writes to Core::X86::Msr::APERF.
	AccessType: Core::X86::Msr::HWCR[EffFreqReadOnlyLock] ? Read,Error-on-write,Volatile : Read-write,Volatile.

MSRC000_00E9 [Instructions Retired Performance Count] (Core::X86::Msr::IRPerfCount)

Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC000_00E9

Bits	Description
63:48	Reserved.
47:0	**IRPerfCount**: **instructions retired counter**. Reset: 0000_0000_0000h. Dedicated Instructions Retired register increments on once for every instruction retired. See Core::X86::Msr::HWCR[IRPerfEn].
	AccessType: Core::X86::Msr::HWCR[EffFreqReadOnlyLock] ? Read,Error-on-write,Volatile : Read-write,Volatile.

MSRC000_0100 [FS Base] (Core::X86::Msr::FS_BASE)

Read-write. Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC000_0100

Bits	Description
63:0	**FSBase**: **expanded FS segment base**. Read-write. Reset: 0000_0000_0000_0000h. This register provides access to the expanded 64-bit FS segment base. The address stored in this register must be in canonical form (if not canonical, a #GP fault fill occurs).

MSRC000_0101 [GS Base] (Core::X86::Msr::GS_BASE)

Read-write. Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC000_0101

Bits	Description
63:0	**GSBase**: **expanded GS segment base**. Read-write. Reset: 0000_0000_0000_0000h. This register provides access to the expanded 64-bit GS segment base. The address stored in this register must be in canonical form (if not canonical, a #GP fault fill occurs).

MSRC000_0102 [Kernel GS Base] (Core::X86::Msr::KernelGSbase)

Read-write. Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC000_0102

Bits	Description
63:0	**KernelGSBase**: **kernel data structure pointer**. Read-write. Reset: 0000_0000_0000_0000h. This register holds the kernel data structure pointer which can be swapped with the GS_BASE register using the SwapGS instruction. The address stored in this register must be in canonical form (if not canonical, a #GP fault occurs).

MSRC000_0103 [Auxiliary Time Stamp Counter] (Core::X86::Msr::TSC_AUX)

Read-write,Volatile. Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC000_0103

Bits	Description
63:32	Reserved.
31:0	**TscAux**: **auxiliary time stamp counter data**. Read-write,Volatile. Reset: 0000_0000h. It is expected that this is initialized by privileged software to a meaningful value, such as a processor ID. This value is returned in the RDTSCP instruction.

MSRC000_0104 [Time Stamp Counter Ratio] (Core::X86::Msr::TscRateMsr)

Core::X86::Msr::TscRateMsr allows the hypervisor to control the guest's view of the Time Stamp Counter. It provides a multiplier that scales the value returned when Core::X86::Msr::TSC[TSC], Core::X86::Msr::MPERF[MPERF], and Core::X86::Msr::MPerfReadOnly[MPerfReadOnly] are read by a guest running under virtualization. This allows the hypervisor to provide a consistent TSC, MPERF, and MPerfReadOnly rate for a guest process when moving that process between cores that have a differing P0 rate. The TSC Ratio MSR does not affect the value read from the TSC, MPERF, and MPerfReadOnly MSRs when read when in host mode or when virtualization is not being used or when accessed by code executed in system management mode (SMM) unless the SMM code is executed within a guest container. The TSC Ratio value does not affect the rate of the underlying TSC, MPERF, and MPerfReadOnly counters, or the value that gets written to the TSC, MPERF, and MPerfReadOnly MSRs counters on a write by either the host or the guest. The TSC Ratio MSR contains a fixed-point number in 8.32 format, which is 8 bits of integer and 32 bits of fraction. This number is the ratio of the desired P0 frequency to the P0 frequency of the core. The reset value of the TSC Ratio MSR is 1.0, which results in a guest frequency matches the core P0 frequency.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC000_0104

Bits	Description
63:40	Reserved.
39:32	**TscRateMsrInt**: **time stamp counter rate integer**. Read-write. Reset: 01h. Specifies the integer part of the MSR TSC ratio value.
31:0	**TscRateMsrFrac**: **time stamp counter rate fraction**. Read-write. Reset: 0000_0000h. Specifies the fractional part of the MSR TSC ratio value.

MSRC000_0108 [Prefetch Control] (Core::X86::Msr::PrefetchControl)

Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]; MSRC000_0108

Bits	Description
63:6	Reserved.
5	**UpDown**. Read-write. Reset: 0. Disable prefetcher that uses memory access history to determine whether to fetch the next or previous line into L2 cache for all memory accesses.
4	Reserved.
3	**L2Stream**. Read-write. Reset: 0. Disable prefetcher that uses history of memory access patterns to fetch additional sequential lines into L2 cache.
2	**L1Region**. Read-write. Reset: 0. Disable prefetcher that uses memory access history to fetch additional lines into L1 cache when the data access for a given instruction tends to be followed by a consistent pattern of other accesses within a localized region.

1	**L1Stride**. Read-write. Reset: 0. Disable stride prefetcher that uses memory access history of individual instructions to fetch additional lines into L1 cache when each access is a constant distance from the previous.
0	**L1Stream**. Read-write. Reset: 0. Disable stream prefetcher that uses history of memory access patterns to fetch additional sequential lines into L1 cache.

MSRC000_010F [Debug Extension Configuration] (Core::X86::Msr::DbgExtnCfg)

Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC000_010F

Bits	Description
63:27	Reserved.
26:24	**PMC**. Read-write. Reset: 0h. PMC number that is programmed to overflow. See Core::X86::Msr::PERF_CTL0..5.
23:20	Reserved.
19:16	**MSROFF**. Read-write. Reset: 0h. Offset in the MSR map for the next recorded branch. Indicates the first sampling pair of MSRs that the processor should fill with branch records. It will proceed from that specified pair to the Core::X86::Msr::SAMP_BR_FROM/TO[N-1] and then will stop recording. When it stops recording by reaching the last entry the MSROFF field will be written to 0. If software disables branch sampling while recording is active, MSROFF contains the next Core::X86::Msr::SAMP_BR_FROM/TO MSR that would have been updated. Therefore Core::X86::Msr::SAMP_BR_FROM/TO_0 to MSROFF-1 should contain valid recorded branches if the VB =1.
15:6	Reserved.
5	**VB**. Read-write. Reset: 0. Valid Branches are recorded in the MSRs. See Core::X86::Msr::SAMP_BR_FROM and Core::X86::Msr::SAMP_BR_TO.
4:3	**RSVD4_3**. Read-write. Reset: 0h. Write 11b when setting BRSMEN.
2	**BRSMEN**. Read-write. Reset: 0. Branch Sampling Enable which turns the recording mode on for branches.
1:0	Reserved.

MSRC000_020[0...F] [L3 QOS Bandwidth Control] (Core::X86::Msr::L3QosBwControl)

_ccd[7:0]_lthree0_n0; MSRC000_0200
_ccd[7:0]_lthree0_n1; MSRC000_0201
_ccd[7:0]_lthree0_n2; MSRC000_0202
_ccd[7:0]_lthree0_n3; MSRC000_0203
_ccd[7:0]_lthree0_n4; MSRC000_0204
_ccd[7:0]_lthree0_n5; MSRC000_0205
_ccd[7:0]_lthree0_n6; MSRC000_0206
_ccd[7:0]_lthree0_n7; MSRC000_0207
_ccd[7:0]_lthree0_n8; MSRC000_0208
_ccd[7:0]_lthree0_n9; MSRC000_0209
_ccd[7:0]_lthree0_n10; MSRC000_020A
_ccd[7:0]_lthree0_n11; MSRC000_020B
_ccd[7:0]_lthree0_n12; MSRC000_020C
_ccd[7:0]_lthree0_n13; MSRC000_020D
_ccd[7:0]_lthree0_n14; MSRC000_020E
_ccd[7:0]_lthree0_n15; MSRC000_020F

Bits	Description
63:12	Reserved.
11:0	**Ceiling**. Read-write. Reset: 800h. QOS BW Control BW ceiling value.

MSRC000_0410 [MCA Interrupt Configuration] (Core::X86::Msr::McaIntrCfg)

Read-write. Reset: 0000_0000_0000_0000h.

MSRC000_0410

Bits	Description
63:16	Reserved.
15:12	**ThresholdLvtOffset**. Read-write. Reset: 0h. For error thresholding interrupts, specifies the address of the LVT entry in the APIC registers as follows: LVT address = (LvtOffset shifted left 4 bits) + 500h (see

	Core::X86::Apic::ExtendedInterruptLvtEntries).
11:8	Reserved.
7:4	**DeferredLvtOffset**. Read-write. Reset: 0h. **Description**: For deferred error interrupts, specifies the address of the LVT entry in the APIC registers as follows: LVT address = (LvtOffset shifted left 4 bits) + 500h (see APIC[530:500]).
3:0	Reserved.

2.1.16.3 MSRs - MSRC001_0xxx

<table>
<tr><th colspan="2">MSRC001_0000 [Performance Event Select 0] (Core::X86::Msr::PERF_LEGACY_CTL0)</th></tr>
<tr><td colspan="2">Read-write. Reset: 0000_0000_0000_0000h.</td></tr>
<tr><td colspan="2">The legacy alias of Core::X86::Msr::PERF_CTL0. See Core::X86::Msr::PERF_CTL0.</td></tr>
<tr><td colspan="2">_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_0000</td></tr>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>63:42</td><td>Reserved.</td></tr>
<tr><td>41:40</td><td>HostGuestOnly: count only host/guest events. Read-write. Reset: 0h.
ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>Count all events, irrespective of guest/host.</td></tr>
<tr><td>1h</td><td>Count guest events if [SVME] == 1.</td></tr>
<tr><td>2h</td><td>Count host events if [SVME] == 1.</td></tr>
<tr><td>3h</td><td>Count all guest and host events if [SVME] == 1.</td></tr>
</table></td></tr>
<tr><td>39:36</td><td>Reserved.</td></tr>
<tr><td>35:32</td><td>EventSelect[11:8]. Read-write. Reset: 0h. Performance event select[11:8].</td></tr>
<tr><td>31:24</td><td>CntMask: counter mask. Read-write. Reset: 00h. Controls the number of events counted per clock cycle.
ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>00h</td><td>The corresponding PERF_CTR[5:0] register increments by the number of events occurring in a clock cycle. See 2.1.17.3 [Large Increment per Cycle Events] for events that can increment greater than 15 per cycle.</td></tr>
<tr><td>7Fh-01h</td><td>When Inv == 0, the corresponding PERF_CTR[5:0] register increments by 1, if the number of events occurring in a clock cycle is greater than or equal to the CntMask value. When Inv == 1, the corresponding PERF_CTR[5:0] register increments by 1, if the number of events occurring in a clock cycle is less than CntMask value.</td></tr>
<tr><td>FFh-80h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>23</td><td>Inv: invert counter mask. Read-write. Reset: 0. See CntMask.</td></tr>
<tr><td>22</td><td>En: enable performance counter. Read-write. Reset: 0. 1=Performance event counter is enabled.</td></tr>
<tr><td>21</td><td>Reserved.</td></tr>
<tr><td>20</td><td>Int: enable APIC interrupt. Read-write. Reset: 0. 1=APIC performance counter LVT interrupt is enabled to generate an interrupt via Core::X86::Apic::PerformanceCounterLvtEntry when the performance counter overflows.</td></tr>
<tr><td>19</td><td>Reserved.</td></tr>
<tr><td>18</td><td>Edge: edge detect. Read-write. Reset: 0. 0=Level detect. 1=Zero-to-one Edge detect. Read-write. The edge count mode increments the counter when a transition happens on the monitored event. If the event selected is changed without disabling the counter, an extra edge is falsely detected when the first event is a static 0 and the second event is a static one. To avoid this false edge detection, disable the counter when changing the event and then enable the counter with a second MSR write.</td></tr>
</table>

<table>
<tr><td>17:16</td><td>OsUserMode: OS and user mode. Read-write. Reset: 0h.
ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>Count no events.</td></tr>
<tr><td>1h</td><td>Count user events (CPL>0).</td></tr>
<tr><td>2h</td><td>Count OS events (CPL=0).</td></tr>
<tr><td>3h</td><td>Count all events, irrespective of the CPL.</td></tr>
</table></td></tr>
<tr><td>15:8</td><td>UnitMask: event qualification. Read-write. Reset: 00h. Each UnitMask bit further specifies or qualifies the event specified by EventSelect. All events selected by UnitMask are simultaneously monitored. Unless otherwise stated, the UnitMask values shown may be combined (logically ORed) to select any desired combination of the sub-events for a given event. In some cases, certain combinations can result in misleading counts, or the UnitMask value is an ordinal rather than a bit mask. These situations are described where applicable, or should be obvious from the event descriptions. For events where no UnitMask table is shown, the UnitMask is Unused.</td></tr>
<tr><td>7:0</td><td>EventSelect[7:0]: event select. Read-write. Reset: 00h. EventSelect[11:0] = {EventSelect[11:8], EventSelect[7:0]}. EventSelect specifies the event or event duration in a processor unit to be counted by the corresponding PERF_CTR[5:0] register. The events are specified in 2.1.17.4 [Core Performance Monitor Counters]. Some events are reserved; when a reserved event is selected, the results are undefined.</td></tr>
</table>

MSRC001_0001 [Performance Event Select 1] (Core::X86::Msr::PERF_LEGACY_CTL1)

<table>
<tr><td colspan="2">Read-write. Reset: 0000_0000_0000_0000h.</td></tr>
<tr><td colspan="2">The legacy alias of Core::X86::Msr::PERF_CTL1. See Core::X86::Msr::PERF_CTL1.</td></tr>
<tr><td colspan="2">_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_0001</td></tr>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>63:42</td><td>Reserved.</td></tr>
<tr><td>41:40</td><td>HostGuestOnly: count only host/guest events. Read-write. Reset: 0h.
ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>Count all events, irrespective of guest/host.</td></tr>
<tr><td>1h</td><td>Count guest events if [SVME] == 1.</td></tr>
<tr><td>2h</td><td>Count host events if [SVME] == 1.</td></tr>
<tr><td>3h</td><td>Count all guest and host events if [SVME] == 1.</td></tr>
</table></td></tr>
<tr><td>39:36</td><td>Reserved.</td></tr>
<tr><td>35:32</td><td>EventSelect[11:8]. Read-write. Reset: 0h. Performance event select[11:8].</td></tr>
<tr><td>31:24</td><td>CntMask: counter mask. Read-write. Reset: 00h. Controls the number of events counted per clock cycle.
ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>00h</td><td>The corresponding PERF_CTR[5:0] register increments by the number of events occurring in a clock cycle. See 2.1.17.3 [Large Increment per Cycle Events] for events that can increment greater than 15 per cycle.</td></tr>
<tr><td>7Fh-01h</td><td>When Inv == 0, the corresponding PERF_CTR[5:0] register increments by 1, if the number of events occurring in a clock cycle is greater than or equal to the CntMask value. When Inv == 1, the corresponding PERF_CTR[5:0] register increments by 1, if the number of events occurring in a clock cycle is less than CntMask value.</td></tr>
<tr><td>FFh-80h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>23</td><td>Inv: invert counter mask. Read-write. Reset: 0. See CntMask.</td></tr>
<tr><td>22</td><td>En: enable performance counter. Read-write. Reset: 0. 1=Performance event counter is enabled.</td></tr>
<tr><td>21</td><td>Reserved.</td></tr>
<tr><td>20</td><td>Int: enable APIC interrupt. Read-write. Reset: 0. 1=APIC performance counter LVT interrupt is enabled to generate an interrupt via Core::X86::Apic::PerformanceCounterLvtEntry when the performance counter</td></tr>
</table>

<table>
<tr><td></td><td>overflows.</td></tr>
<tr><td>19</td><td>Reserved.</td></tr>
<tr><td>18</td><td>Edge: edge detect. Read-write. Reset: 0. 0=Level detect. 1=Zero-to-one Edge detect. Read-write. The edge count mode increments the counter when a transition happens on the monitored event. If the event selected is changed without disabling the counter, an extra edge is falsely detected when the first event is a static 0 and the second event is a static one. To avoid this false edge detection, disable the counter when changing the event and then enable the counter with a second MSR write.</td></tr>
<tr><td rowspan="2">17:16</td><td>OsUserMode: OS and user mode. Read-write. Reset: 0h.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>Count no events.</td></tr>
<tr><td>1h</td><td>Count user events (CPL>0).</td></tr>
<tr><td>2h</td><td>Count OS events (CPL=0).</td></tr>
<tr><td>3h</td><td>Count all events, irrespective of the CPL.</td></tr>
</table></td></tr>
<tr><td>15:8</td><td>UnitMask: event qualification. Read-write. Reset: 00h. Each UnitMask bit further specifies or qualifies the event specified by EventSelect. All events selected by UnitMask are simultaneously monitored. Unless otherwise stated, the UnitMask values shown may be combined (logically ORed) to select any desired combination of the sub-events for a given event. In some cases, certain combinations can result in misleading counts, or the UnitMask value is an ordinal rather than a bit mask. These situations are described where applicable, or should be obvious from the event descriptions. For events where no UnitMask table is shown, the UnitMask is Unused.</td></tr>
<tr><td>7:0</td><td>EventSelect[7:0]: event select. Read-write. Reset: 00h. EventSelect[11:0] = {EventSelect[11:8], EventSelect[7:0]}. EventSelect specifies the event or event duration in a processor unit to be counted by the corresponding PERF_CTR[5:0] register. The events are specified in 2.1.17.4 [Core Performance Monitor Counters]. Some events are reserved; when a reserved event is selected, the results are undefined.</td></tr>
</table>

MSRC001_0002 [Performance Event Select 2] (Core::X86::Msr::PERF_LEGACY_CTL2)

<table>
<tr><td colspan="2">Read-write. Reset: 0000_0000_0000_0000h.</td></tr>
<tr><td colspan="2">The legacy alias of Core::X86::Msr::PERF_CTL2. See Core::X86::Msr::PERF_CTL2.</td></tr>
<tr><td colspan="2">_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_0002</td></tr>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>63:42</td><td>Reserved.</td></tr>
<tr><td rowspan="2">41:40</td><td>HostGuestOnly: count only host/guest events. Read-write. Reset: 0h.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>Count all events, irrespective of guest/host.</td></tr>
<tr><td>1h</td><td>Count guest events if [SVME] == 1.</td></tr>
<tr><td>2h</td><td>Count host events if [SVME] == 1.</td></tr>
<tr><td>3h</td><td>Count all guest and host events if [SVME] == 1.</td></tr>
</table></td></tr>
<tr><td>39:36</td><td>Reserved.</td></tr>
<tr><td>35:32</td><td>EventSelect[11:8]. Read-write. Reset: 0h. Performance event select[11:8].</td></tr>
<tr><td rowspan="2">31:24</td><td>CntMask: counter mask. Read-write. Reset: 00h. Controls the number of events counted per clock cycle.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>00h</td><td>The corresponding PERF_CTR[5:0] register increments by the number of events occurring in a clock cycle. See 2.1.17.3 [Large Increment per Cycle Events] for events that can increment greater than 15 per cycle.</td></tr>
<tr><td>7Fh-01h</td><td>When Inv == 0, the corresponding PERF_CTR[5:0] register increments by 1, if the number of events occurring in a clock cycle is greater than or equal to the CntMask value. When Inv == 1, the corresponding PERF_CTR[5:0] register increments by 1, if the number of events occurring in a clock cycle is less than CntMask value.</td></tr>
</table></td></tr>
</table>

FFh-80h	Reserved.

23	**Inv**: **invert counter mask**. Read-write. Reset: 0. See CntMask.
22	**En**: **enable performance counter**. Read-write. Reset: 0. 1=Performance event counter is enabled.
21	Reserved.
20	**Int**: **enable APIC interrupt**. Read-write. Reset: 0. 1=APIC performance counter LVT interrupt is enabled to generate an interrupt via Core::X86::Apic::PerformanceCounterLvtEntry when the performance counter overflows.
19	Reserved.
18	**Edge**: **edge detect**. Read-write. Reset: 0. 0=Level detect. 1=Zero-to-one Edge detect. Read-write. The edge count mode increments the counter when a transition happens on the monitored event. If the event selected is changed without disabling the counter, an extra edge is falsely detected when the first event is a static 0 and the second event is a static one. To avoid this false edge detection, disable the counter when changing the event and then enable the counter with a second MSR write.
17:16	**OsUserMode**: **OS and user mode**. Read-write. Reset: 0h.
	ValidValues:

Value	Description
0h	Count no events.
1h	Count user events (CPL>0).
2h	Count OS events (CPL=0).
3h	Count all events, irrespective of the CPL.

15:8	**UnitMask**: **event qualification**. Read-write. Reset: 00h. Each UnitMask bit further specifies or qualifies the event specified by EventSelect. All events selected by UnitMask are simultaneously monitored. Unless otherwise stated, the UnitMask values shown may be combined (logically ORed) to select any desired combination of the sub-events for a given event. In some cases, certain combinations can result in misleading counts, or the UnitMask value is an ordinal rather than a bit mask. These situations are described where applicable, or should be obvious from the event descriptions. For events where no UnitMask table is shown, the UnitMask is Unused.
7:0	**EventSelect[7:0]**: **event select**. Read-write. Reset: 00h. EventSelect[11:0] = {EventSelect[11:8], EventSelect[7:0]}. EventSelect specifies the event or event duration in a processor unit to be counted by the corresponding PERF_CTR[5:0] register. The events are specified in 2.1.17.4 [Core Performance Monitor Counters]. Some events are reserved; when a reserved event is selected, the results are undefined.

MSRC001_0003 [Performance Event Select 3] (Core::X86::Msr::PERF_LEGACY_CTL3)

Read-write. Reset: 0000_0000_0000_0000h.

The legacy alias of Core::X86::Msr::PERF_CTL3. See Core::X86::Msr::PERF_CTL3.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_0003

Bits	Description
63:42	Reserved.
41:40	**HostGuestOnly**: **count only host/guest events**. Read-write. Reset: 0h.
	ValidValues:

Value	Description
0h	Count all events, irrespective of guest/host.
1h	Count guest events if [SVME] == 1.
2h	Count host events if [SVME] == 1.
3h	Count all guest and host events if [SVME] == 1.

39:36	Reserved.
35:32	**EventSelect[11:8]**. Read-write. Reset: 0h. Performance event select[11:8].
31:24	**CntMask**: **counter mask**. Read-write. Reset: 00h. Controls the number of events counted per clock cycle.
	ValidValues:

Value	Description

00h	The corresponding PERF_CTR[5:0] register increments by the number of events occurring in a clock cycle. See 2.1.17.3 [Large Increment per Cycle Events] for events that can increment greater than 15 per cycle.
7Fh-01h	When Inv == 0, the corresponding PERF_CTR[5:0] register increments by 1, if the number of events occurring in a clock cycle is greater than or equal to the CntMask value. When Inv == 1, the corresponding PERF_CTR[5:0] register increments by 1, if the number of events occurring in a clock cycle is less than CntMask value.
FFh-80h	Reserved.

23	**Inv**: **invert counter mask**. Read-write. Reset: 0. See CntMask.
22	**En**: **enable performance counter**. Read-write. Reset: 0. 1=Performance event counter is enabled.
21	Reserved.
20	**Int**: **enable APIC interrupt**. Read-write. Reset: 0. 1=APIC performance counter LVT interrupt is enabled to generate an interrupt via Core::X86::Apic::PerformanceCounterLvtEntry when the performance counter overflows.
19	Reserved.
18	**Edge**: **edge detect**. Read-write. Reset: 0. 0=Level detect. 1=Zero-to-one Edge detect. Read-write. The edge count mode increments the counter when a transition happens on the monitored event. If the event selected is changed without disabling the counter, an extra edge is falsely detected when the first event is a static 0 and the second event is a static one. To avoid this false edge detection, disable the counter when changing the event and then enable the counter with a second MSR write.
17:16	**OsUserMode**: **OS and user mode**. Read-write. Reset: 0h.
	ValidValues:

Value	Description
0h	Count no events.
1h	Count user events (CPL>0).
2h	Count OS events (CPL=0).
3h	Count all events, irrespective of the CPL.

15:8	**UnitMask**: **event qualification**. Read-write. Reset: 00h. Each UnitMask bit further specifies or qualifies the event specified by EventSelect. All events selected by UnitMask are simultaneously monitored. Unless otherwise stated, the UnitMask values shown may be combined (logically ORed) to select any desired combination of the sub-events for a given event. In some cases, certain combinations can result in misleading counts, or the UnitMask value is an ordinal rather than a bit mask. These situations are described where applicable, or should be obvious from the event descriptions. For events where no UnitMask table is shown, the UnitMask is Unused.
7:0	**EventSelect[7:0]**: **event select**. Read-write. Reset: 00h. EventSelect[11:0] = {EventSelect[11:8], EventSelect[7:0]}. EventSelect specifies the event or event duration in a processor unit to be counted by the corresponding PERF_CTR[5:0] register. The events are specified in 2.1.17.4 [Core Performance Monitor Counters]. Some events are reserved; when a reserved event is selected, the results are undefined.

MSRC001_000[4...7] [Performance Event Counter [3:0]] (Core::X86::Msr::PERF_LEGACY_CTR)

Read-write,Volatile. Reset: 0000_0000_0000_0000h.

Note: When counting events that capable of counting greater than 15 events per cycle (MergeEvent) the even and the corresponding odd PERF_LEGACY_CTR must be paired to appear as a single 64 bit counter. See 2.1.17.3 [Large Increment per Cycle Events].
The legacy alias of Core::X86::Msr::PERF_CTR. See Core::X86::Msr::PERF_CTR.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n0; MSRC001_0004
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n1; MSRC001_0005
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n2; MSRC001_0006
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n3; MSRC001_0007

Bits	Description
63:48	Reserved.
47:0	**CTR**. Read-write,Volatile. Reset: 0000_0000_0000h. Performance counter value.

MSRC001_0010 [**System Configuration**] **(Core::X86::Msr::SYS_CFG)**

Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]; MSRC001_0010

Bits	Description
63:26	Reserved.
25	**VmplEn**. Reset: 0. VM permission levels enable.
	AccessType: Core::X86::Msr::SYS_CFG[SecureNestedPagingEn] ? Read-only : Read-write.
24	**SecureNestedPagingEn**. Read,Write-1-only. Reset: 0. Enable Secure Nested Paging (SNP).
23	**SMEE**: **secure memory encryption enable**. Read,Write-1-only. Reset: 0. 0=Memory encryption features are disabled. 1=Memory encryption features are enabled. For enabling secure memory encryption see 2.1.4 [Memory Encryption]
22	**Tom2ForceMemTypeWB**: **top of memory 2 memory type write back**. Read-write. Reset: 0. 1=The default memory type of memory between 4-GB and Core::X86::Msr::TOM2 is write back instead of the memory type defined by Core::X86::Msr::MTRRdefType[MemType]. For this bit to have any effect, Core::X86::Msr::MTRRdefType[MtrrDefTypeEn] must be 1. MTRRs and PAT can be used to override this memory type.
21	**MtrrTom2En**: **MTRR top of memory 2 enable**. Read-write. Reset: 0. 0=Core::X86::Msr::TOM2 is disabled. 1= Core::X86::Msr::TOM2 is enabled.
20	**MtrrVarDramEn**: **MTRR variable DRAM enable**. Read-write. Reset: 0. Init: BIOS,1. 0=Core::X86::Msr::TOP_MEM and IORRs are disabled. 1=These registers are enabled.
19	**MtrrFixDramModEn**: **MTRR fixed RdDram and WrDram modification enable**. Read-write. Reset: 0. 0=Core::X86::Msr::MtrrFix_64K through Core::X86::Msr::MtrrFix_4K_7 [RdDram,WrDram] read values is masked 00b; writing does not change the hidden value. 1=Core::X86::Msr::MtrrFix_64K through Core::X86::Msr::MtrrFix_4K_7 [RdDram,WrDram] access type is Read-write. Not shared between threads. Controls access to Core::X86::Msr::MtrrFix_64K through Core::X86::Msr::MtrrFix_4K_7 [RdDram ,WrDram]. This bit should be set to 1 during BIOS initialization of the fixed MTRRs, then cleared to 0 for operation.
18	**MtrrFixDramEn**: **MTRR fixed RdDram and WrDram attributes enable**. Read-write. Reset: 0. Init: BIOS,1. 1=Enables the RdDram and WrDram attributes in Core::X86::Msr::MtrrFix_64K through Core::X86::Msr::MtrrFix_4K_7.
17:0	Reserved.

MSRC001_0015 [**Hardware Configuration**] **(Core::X86::Msr::HWCR)**

Reset: 0000_0000_0100_0010h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_0015

Bits	Description
63:34	Reserved.
33	**SmmPgCfgLock**. Read-write. Reset: 0. 1=SMM page config locked. Error-on-write-1 if not in SMM mode. RSM unconditionally clears Core::X86::Msr::HWCR[SmmPgCfgLock].
32:31	Reserved.
30	**IRPerfEn**: **enable instructions retired counter**. Read-write. Reset: 0. 1=Enable Core::X86::Msr::IRPerfCount.
29:28	Reserved.
27	**EffFreqReadOnlyLock**: **read-only effective frequency counter lock**. Write-1-only. Reset: 0. Init: BIOS,1. 1=Core::X86::Msr::MPerfReadOnly, Core::X86::Msr::APerfReadOnly and Core::X86::Msr::IRPerfCount are Read-only.
26	**EffFreqCntMwait**: **effective frequency counting during mwait**. Read-write. Reset: 0. 0=The registers do not increment. 1=The registers increment. Specifies whether Core::X86::Msr::MPERF and Core::X86::Msr::APERF increment while the core is in the monitor event pending state. See 2.1.6 [Effective Frequency].
25	**CpbDis**: **core performance boost disable**. Read-write. Reset: 0. 0=CPB is requested to be enabled. 1=CPB is disabled. Specifies whether core performance boost is requested to be enabled or disabled. If core performance boost is disabled while a core is in a boosted P-state, the core automatically transitions to the highest performance non-boosted P-state.

24	**TscFreqSel**: **TSC frequency select**. Read-only. Reset: 1. 1=The TSC increments at the P0 frequency.
23:22	Reserved.
21	**LockTscToCurrentP0**: **lock the TSC to the current P0 frequency**. Read-write. Reset: 0. 0=The TSC will count at the P0 frequency. 1=The TSC frequency is locked to the current P0 frequency at the time this bit is set and remains fixed regardless of future changes to the P0 frequency.
20	**IoCfgGpFault**: **IO-space configuration causes a GP fault**. Read-write. Reset: 0. 1=IO-space accesses to configuration space cause a GP fault. The fault is triggered if any part of the IO Read/Rrite address range is between CF8h and CFFh, inclusive. These faults only result from single IO instructions, not to string and REP IO instructions. This fault takes priority over the IO trap mechanism described by Core::X86::Msr::SMI_ON_IO_TRAP_CTL_STS.
19	Reserved.
18	**McStatusWrEn**: **machine check status write enable**. Read-write. Reset: 0. 0=MCA_STATUS registers are Readable; Writing a non-zero pattern to these registers causes a general protection fault. 1=MCA_STATUS registers are Read-write, including Reserved fields; do not cause general protection faults; such Writes update all implemented bits in these registers; All fields of all threshold registers are Read-write when accessed from MSR space, including Locked, except BlkPtr which is always Read-only; McStatusWrEn does not change the access type for the thresholding registers accessed via configuration space.
	Description: McStatusWrEn can be used to debug machine check exception and interrupt handlers. Independent of the value of this bit, the processor may enforce Write-Ignored behavior on MCA_STATUS registers depending on platform settings. See 3.1 [Machine Check Architecture].
17	**Wrap32Dis**: **32-bit address wrap disable**. Read-write. Reset: 0. 1=Disable 32-bit address wrapping. Software can use Wrap32Dis to access physical memory above 4 Gbytes without switching into 64-bit mode. To do so, software should Write a greater-than 4-Gbyte address to Core::X86::Msr::FS_BASE and Core::X86::Msr::GS_BASE. Then it would address ±2 Gbytes from one of those bases using normal memory reference instructions with a FS or GS override prefix. However, the INVLPG, FST, and SSE store instructions generate 32-bit addresses in legacy mode, regardless of the state of Wrap32Dis.
16:15	Reserved.
14	**RsmSpCycDis**: **RSM special bus cycle disable**. Reset: 0. 0=A link special bus cycle, SMIACK, is generated on a resume from SMI.
	AccessType: Core::X86::Msr::HWCR[SmmLock] ? Read-only : Read-write.
13	**SmiSpCycDis**: **SMI special bus cycle disable**. Reset: 0. 0=A link special bus cycle, SMIACK, is generated when an SMI interrupt is taken.
	AccessType: Core::X86::Msr::HWCR[SmmLock] ? Read-only : Read-write.
12:11	Reserved.
10	**MonMwaitUserEn**: **MONITOR/MWAIT user mode enable**. Read-write. Reset: 0. 0=The MONITOR and MWAIT instructions are supported only in privilege level 0; these instructions in privilege levels 1 to 3 cause a #UD exception. 1=The MONITOR and MWAIT instructions are supported in all privilege levels. The state of this bit is ignored if MonMwaitDis is set.
9	**MonMwaitDis**: **MONITOR and MWAIT disable**. Read-write. Reset: 0. 1=The MONITOR, MWAIT, MONITORX, and MWAITX opcodes become invalid. This affects what is reported back through Core::X86::Cpuid::FeatureIdEcx[Monitor] and Core::X86::Cpuid::FeatureExtIdEcx[MwaitExtended].
8	**IgnneEm**: **IGNNE port emulation enable**. Read-write. Reset: 0. 1=Enable emulation of IGNNE port.
7	**AllowFerrOnNe**: **allow FERR on NE**. Read-write. Reset: 0. 0=Disable legacy FERR signaling and generate FERR exception directly. 1=Legacy FERR signaling.
6:5	Reserved.
4	**INVDWBINVD**: **INVD to WBINVD conversion**. Read-write. Reset: 1. Check: 1. 1=Convert INVD to WBINVD.
	Description: This bit is required to be set for normal operation when any of the following are true: • An L2 is shared by multiple threads. • An L3 is shared by multiple cores.

	• CC6 is enabled. • Probe filter is enabled.
3	**TlbCacheDis**: **cacheable memory disable**. Read-write. Reset: 0. 1=Disable performance improvement that assumes that the PML4, PDP, PDE and PTE entries are in cacheable WB DRAM.
	Description: Operating systems that maintain page tables in any other memory type must set the TlbCacheDis bit to insure proper operation. • TlbCacheDis does not override the memory type specified by the SMM ASeg and TSeg memory regions controlled by Core::X86::Msr::SMMAddr Core::X86::Msr::SMMMask.
2:1	Reserved.
0	**SmmLock**: **SMM code lock**. Read,Write-1-only. Reset: 0. Init: BIOS,1. 1=SMM code in the ASeg and TSeg range and the SMM registers are Read-only and SMI interrupts are not intercepted in SVM. See 2.1.14.1.10 [Locking SMM].

MSRC001_001[6...8] [IO Range Base] (Core::X86::Msr::IORR_BASE)

Read-write.

Core::X86::Msr::IORR_BASE and Core::X86::Msr::IORR_MASK combine to specify the two sets of base and mask pairs for two IORR ranges. A core access, with address CPUAddr, is determined to be within IORR address range if the following equation is true:
CPUAddr[47:12] & PhyMask[47:12] == PhyBase[47:12] & PhyMask[47:12].
BIOS can use the IORRs to create an IO hole within a range of addresses that would normally be mapped to DRAM. It can also use the IORRs to re-assert a DRAM destination for a range of addresses that fall within a bigger IO hole that overlays DRAM.

_ccd[7:0]_lthree0_core[7:0]_n0; MSRC001_0016
_ccd[7:0]_lthree0_core[7:0]_n1; MSRC001_0018

Bits	Description
63:48	Reserved.
47:12	**PhyBase**. Read-write. Reset: X_XXXX_XXXXh. Physical base address for IO range.
11:5	Reserved.
4	**RdMem**: **read from memory**. Read-write. Reset: X. 0=Read accesses to the range are directed to IO. 1=Read accesses to the range are directed to system memory.
3	**WrMem**: **write to memory**. Read-write. Reset: X. 0=Write accesses to the range are directed to IO. 1=Write accesses to the range are directed to system memory.
2:0	Reserved.

MSRC001_001[7...9] [IO Range Mask] (Core::X86::Msr::IORR_MASK)

Read-write. Reset: 0000_0000_0000_0000h.

See Core::X86::Msr::IORR_BASE.

_ccd[7:0]_lthree0_core[7:0]_n0; MSRC001_0017
_ccd[7:0]_lthree0_core[7:0]_n1; MSRC001_0019

Bits	Description
63:48	Reserved.
47:12	**PhyMask**. Read-write. Reset: 0_0000_0000h. Physical address mask for IO range.
11	**Valid**. Read-write. Reset: 0. 1=The pair of registers that specifies an IORR range is valid.
10:0	Reserved.

MSRC001_001A [Top Of Memory] (Core::X86::Msr::TOP_MEM)

Read-write.

_ccd[7:0]_lthree0_core[7:0]; MSRC001_001A

Bits	Description
63:48	Reserved.
47:23	**TOM[47:23]**: **top of memory**. Read-write. Reset: XXX_XXXXh. Specifies the address that divides between MMIO and DRAM. This value is normally placed below 4-GB. From TOM to 4-GB is MMIO; below TOM is

	DRAM. See 2.1.7.3 [System Address Map].
22:0	Reserved.

MSRC001_001D [Top Of Memory 2] (Core::X86::Msr::TOM2)

Read-write.

_ccd[7:0]_lthree0_core[7:0]; MSRC001_001D

Bits	Description
63:48	Reserved.
47:23	**TOM2[47:23]**: **second top of memory**. Read-write. Reset: XXX_XXXXh. Specifies the address divides between MMIO and DRAM. This value is normally placed above 4-GB. From 4-GB to (TOM2 - 1) is DRAM; TOM2 and above is MMIO. See 2.1.7.3 [System Address Map]. This register is enabled by Core::X86::Msr::SYS_CFG[MtrrTom2En].
22:0	Reserved.

MSRC001_0022 [Machine Check Exception Redirection] (Core::X86::Msr::McExcepRedir)

Read-write. Reset: 0000_0000_0000_0000h.

This register can be used to redirect machine check exceptions (MCEs) to SMIs or vectored interrupts. If both RedirSmiEn and RedirVecEn are set, then undefined behavior results.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_0022

Bits	Description
63:10	Reserved.
9	**RedirSmiEn**. Read-write. Reset: 0. 1=Redirect MCEs (that are directed to this core) to generate an SMI-trigger IO cycle via Core::X86::Msr::SmiTrigIoCycle. The status is stored in Core::X86::Smm::LocalSmiStatus[MceRedirSts].
8	**RedirVecEn**. Read-write. Reset: 0. 1=Redirect MCEs (that are directed to this core) to generate a vectored interrupt, using the interrupt vector specified in RedirVector.
7:0	**RedirVector**. Read-write. Reset: 00h. See RedirVecEn.

MSRC001_003[0...5] [Processor Name String] (Core::X86::Msr::ProcNameString)

Read-write.

These 6 registers hold the CPUID name string in ASCII. The state of these registers are returned by CPUID instructions, Core::X86::Cpuid::ProcNameStr0Eax through Core::X86::Cpuid::ProcNameStr2Edx. BIOS should set these registers to the product name for the processor as provided by AMD. Each register contains a block of 8 ASCII characters; the least byte corresponds to the first ASCII character of the block; the most-significant byte corresponds to the last character of the block. MSRC001_0030 contains the first block of the name string; MSRC001_0035 contains the last block of the name string.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n0; MSRC001_0030
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n1; MSRC001_0031
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n2; MSRC001_0032
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n3; MSRC001_0033
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n4; MSRC001_0034
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n5; MSRC001_0035

Bits	Description
63:56	**CpuNameString7**. Read-write. Reset: XXh. CPUID name string in ASCII.
55:48	**CpuNameString6**. Read-write. Reset: XXh. CPUID name string in ASCII.
47:40	**CpuNameString5**. Read-write. Reset: XXh. CPUID name string in ASCII.
39:32	**CpuNameString4**. Read-write. Reset: XXh. CPUID name string in ASCII.
31:24	**CpuNameString3**. Read-write. Reset: XXh. CPUID name string in ASCII.
23:16	**CpuNameString2**. Read-write. Reset: XXh. CPUID name string in ASCII.
15:8	**CpuNameString1**. Read-write. Reset: XXh. CPUID name string in ASCII.
7:0	**CpuNameString0**. Read-write. Reset: XXh. CPUID name string in ASCII.

MSRC001_005[0...3] [IO Trap] (Core::X86::Msr::SMI_ON_IO_TRAP)

Read-write. Reset: 0000_0000_0000_0000h.	
Core::X86::Msr::SMI_ON_IO_TRAP and Core::X86::Msr::SMI_ON_IO_TRAP_CTL_STS provide a mechanism for executing the SMI handler if a an access to one of the specified addresses is detected. Access address and access type checking is performed before IO instruction execution. If the access address and access type match one of the specified IO address and access types, then: (1) the IO instruction is not executed; (2) any breakpoint, other than the single-step breakpoint, set on the IO instruction is not taken (the single-step breakpoint is taken after resuming from SMM); and (3) issue the SMI-trigger IO cycle specified by Core::X86::Msr::SmiTrigIoCycle if enabled. The status is stored in Core::X86::Smm::LocalSmiStatus[IoTrapSts]. IO-space configuration accesses are special IO accesses. An IO access is defined as an IO-space configuration access when IO instruction address bits[31:0] are CFCh, CFDh, CFEh, or CFFh when IO-space configuration is enabled (IO::IoCfgAddr[ConfigEn]). The access address for a configuration space access is the current value of IO::IoCfgAddr[BusNo,Device,Function,RegNo]. The access address for an IO access that is not a configuration access is equivalent to the IO instruction address, bits[31:0]. The access address is compared with SmiAddr, and the instruction access type is compared with the enabled access types defined by ConfigSMI, SmiOnRdEn, and SmiOnWrEn. Access address bits[23:0] can be masked with SmiMask. IO and configuration space trapping to SMI applies only to single IO instructions; it does not apply to string and REP IO instructions. The conditional GP fault described by Core::X86::Msr::HWCR[IoCfgGpFault] takes priority over this trap.	
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n0; MSRC001_0050	
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n1; MSRC001_0051	
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n2; MSRC001_0052	
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n3; MSRC001_0053	

Bits	Description
63	**SmiOnRdEn**: **enable SMI on IO read**. Read-write. Reset: 0. 1=Enables SMI generation on a read access.
62	**SmiOnWrEn**: **enable SMI on IO write**. Read-write. Reset: 0. 1=Enables SMI generation on a write access.
61	**ConfigSmi**: **configuration space SMI**. Read-write. Reset: 0. 0=IO access (that is not an IO-space configuration access). 1=Configuration access.
60:56	Reserved.
55:32	**SmiMask[23:0]**. Read-write. Reset: 00_0000h. 1=Do not mask address bit. 0=Mask address bit. SMI IO trap mask.
31:0	**SmiAddr[31:0]**. Read-write. Reset: 0000_0000h. SMI IO trap address.

MSRC001_0054 [IO Trap Control] (Core::X86::Msr::SMI_ON_IO_TRAP_CTL_STS)

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_0054

Bits	Description
63:16	Reserved.
15	**IoTrapEn**: **IO trap enable**. Read-write. Reset: 0. 1=Enable IO and configuration space trapping specified by Core::X86::Msr::SMI_ON_IO_TRAP and Core::X86::Msr::SMI_ON_IO_TRAP_CTL_STS.
14:8	Reserved.
7	**SmiEn3**. Read-write. Reset: 0. 1=The trap Core::X86::Msr::SMI_ON_IO_TRAP_n[3] is enabled.
6	Reserved.
5	**SmiEn2**. Read-write. Reset: 0. 1=The trap Core::X86::Msr::SMI_ON_IO_TRAP_n[2] is enabled.
4	Reserved.
3	**SmiEn1**. Read-write. Reset: 0. 1=The trap Core::X86::Msr::SMI_ON_IO_TRAP_n[1] is enabled.
2	Reserved.
1	**SmiEn0**. Read-write. Reset: 0. 1=The trap Core::X86::Msr::SMI_ON_IO_TRAP_n[0] is enabled.
0	Reserved.

MSRC001_0055 [Reserved.] (Core::X86::Msr::IntPend)

Read-only. Reset: Fixed,0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]; MSRC001_0055

Bits	Description
63:0	Reserved.

MSRC001_0056 [SMI Trigger IO Cycle] (Core::X86::Msr::SmiTrigIoCycle)

Read-write. Reset: 0000_0000_0000_0000h.

See 2.1.14.1.3 [SMI Sources And Delivery]. This register specifies an IO cycle that may be generated when a local SMI trigger event occurs. If IoCycleEn is set and there is a local SMI trigger event, then the IO cycle generated is a byte read or write, based on IoRd, to address IoPortAddress. If the cycle is a write, then IoData contains the data written. If the cycle is a read, the value read is discarded. If IoCycleEn is clear and a local SMI trigger event occurs, then undefined behavior results.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_0056

Bits	Description
63:27	Reserved.
26	**IoRd**: **IO Read**. Read-write. Reset: 0. 0=IO write. 1=IO read.
25	**IoCycleEn**: **IO cycle enable**. Read-write. Reset: 0. 1=The SMI trigger IO cycle is enabled to be generated.
24	Reserved.
23:16	**IoData**. Read-write. Reset: 00h. See 2.1.14.1.3 [SMI Sources And Delivery].
15:0	**IoPortAddress**. Read-write. Reset: 0000h. See 2.1.14.1.3 [SMI Sources And Delivery].

MSRC001_0058 [MMIO Configuration Base Address] (Core::X86::Msr::MmioCfgBaseAddr)

See 2.1.8 [Configuration Space] for a description of MMIO configuration space.

_ccd[7:0]_lthree0_core[7:0]; MSRC001_0058

<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>63:48</td><td>Reserved.</td></tr>
<tr><td>47:20</td><td>MmioCfgBaseAddr[47:20]: MMIO configuration base address bits[47:20]. Read-write. Reset: XXX_XXXXh. Specifies the base address of the MMIO configuration range.</td></tr>
<tr><td>19:6</td><td>Reserved.</td></tr>
<tr><td rowspan="2">5:2</td><td>BusRange: bus range identifier. Read-write. Reset: 0h. Specifies the number of buses in the MMIO configuration space range. The size of the MMIO configuration space is 1 MB times the number of buses.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>1</td></tr>
<tr><td>1h</td><td>2</td></tr>
<tr><td>2h</td><td>4</td></tr>
<tr><td>3h</td><td>8</td></tr>
<tr><td>4h</td><td>16</td></tr>
<tr><td>5h</td><td>32</td></tr>
<tr><td>6h</td><td>64</td></tr>
<tr><td>7h</td><td>128</td></tr>
<tr><td>8h</td><td>256</td></tr>
<tr><td>Fh-9h</td><td>Reserved</td></tr>
</table></td></tr>
<tr><td>1</td><td>Reserved.</td></tr>
<tr><td>0</td><td>Enable. Read-write. Reset: 0. 1=MMIO configuration space is enabled.</td></tr>
</table>

MSRC001_0061 [P-state Current Limit] (Core::X86::Msr::PStateCurLim)

_ccd[7:0]_lthree0_core[7:0]; MSRC001_0061

Bits	Description
63:7	Reserved.
6:4	**PstateMaxVal**: **P-state maximum value**. Read,Error-on-write,Volatile. Reset: XXXb. Specifies the lowest-performance non-boosted P-state (highest non-boosted value) allowed. Attempts to change Core::X86::Msr::PStateCtl[PstateCmd] to a lower-performance P-state (higher value) are clipped to the value of this field.

3	Reserved.
2:0	**CurPstateLimit**: **current P-state limit**. Read,Error-on-write,Volatile. Reset: XXXb. Specifies the highest-performance P-state (lowest value) allowed. CurPstateLimit is always bounded by Core::X86::Msr::PStateCurLim[PstateMaxVal]. Attempts to change the CurPstateLimit to a value greater (lower performance) than Core::X86::Msr::PStateCurLim[PstateMaxVal] leaves CurPstateLimit unchanged.

MSRC001_0062 [P-state Control] (Core::X86::Msr::PStateCtl)

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_0062

Bits	Description
63:3	Reserved.
2:0	**PstateCmd**: **P-state change command**. Read-write. Reset: XXXb. Cold reset value varies by product; after a warm reset, value initializes to the P-state the core was in prior to the reset. Writes to this field cause the core to change to the indicated non-boosted P-state number, specified by Core::X86::Msr::PStateDef. 0=P0, 1=P1, etc. P-state limits are applied to any P-state requests made through this register. Reads from this field return the last written value, regardless of whether any limits are applied.

MSRC001_0063 [P-state Status] (Core::X86::Msr::PStateStat)

Read,Error-on-write,Volatile.

_ccd[7:0]_lthree0_core[7:0]; MSRC001_0063

Bits	Description
63:3	Reserved.
2:0	**CurPstate**: **current P-state**. Read,Error-on-write,Volatile. Reset: XXXb. This field provides the frequency component of the current non-boosted P-state of the core (regardless of the source of the P-state change, including Core::X86::Msr::PStateCtl[PstateCmd]. 0=P0, 1=P1, etc. The value of this field is updated when the COF transitions to a new value associated with a P-state.

MSRC001_006[4...B] [P-state [7:0]] (Core::X86::Msr::PStateDef)

Read-write.

Each of these registers specify the frequency and voltage associated with each of the core P-states.
The CpuVid field in these registers is required to be programmed to the same value in all cores of a processor, but are allowed to be different between processors in a multi-processor system. All other fields in these registers are required to be programmed to the same value in each core of the coherent fabric.

_n0_aliasMSR; MSRC001_0064
_n1_aliasMSR; MSRC001_0065
_n2_aliasMSR; MSRC001_0066
_n3_aliasMSR; MSRC001_0067
_n4_aliasMSR; MSRC001_0068
_n5_aliasMSR; MSRC001_0069
_n6_aliasMSR; MSRC001_006A
_n7_aliasMSR; MSRC001_006B

Bits	Description
63	**PstateEn**. Read-write. Reset: X. 0=The P-state specified by this MSR is not valid. 1=The P-state specified by this MSR is valid. The purpose of this register is to indicate if the rest of the P-state information in the register is valid after a reset; it controls no hardware.
62:32	Reserved.
31:30	**IddDiv**: **current divisor**. Read-write. Reset: XXb. See IddValue.
29:22	**IddValue**: **current value**. Read-write. Reset: XXXXXXXXb. After a reset, IddDiv and IddValue combine to specify the expected maximum current dissipation of a single core that is in the P-state corresponding to the MSR number. These values are intended to be used to create ACPI-defined _PSS objects. The values are expressed in amps; they are not intended to convey final product power levels; they may not match the power levels specified in the Power and Thermal Datasheets.
21:14	**CpuVid[7:0]**: **core VID**. Read-write. Reset: XXXXXXXXb.
13:8	**CpuDfsId**: **core divisor ID**. Read-write. Reset: XXXXXXb. Specifies the core frequency divisor; see CpuFid. For values [1Ah:08h], 1/8th integer divide steps supported down to VCO/3.25 (Note, L3/L2 FIFO logic related to

4-cycle data heads-up requires core to be 1/3 of L3 frequency or higher). For values [30h:1Ch], 1/4th integer divide steps supported down to VCO/6 (DID[0] should zero if DID[5:0] > 1Ah). (Note, core and L3 frequencies below 400MHz are not supported by the architecture). Core supports DID up to 30h, but L3 must be 2Ch (VCO/5.5) or less.

ValidValues:

Value	Description
00h	Off
07h-01h	Reserved.
08h	VCO/1
09h	VCO/1.125
1Ah-0Ah	VCO/<Value/8>
1Bh	Reserved.
1Ch	VCO/<Value/8>
1Dh	Reserved.
1Eh	VCO/<Value/8>
1Fh	Reserved.
20h	VCO/<Value/8>
21h	Reserved.
22h	VCO/<Value/8>
23h	Reserved.
24h	VCO/<Value/8>
25h	Reserved.
26h	VCO/<Value/8>
27h	Reserved.
28h	VCO/<Value/8>
29h	Reserved.
2Ah	VCO/<Value/8>
2Bh	Reserved.
2Ch	VCO/<Value/8>
3Fh-2Dh	Reserved.

7:0 **CpuFid[7:0]**: **core frequency ID**. Read-write. Reset: XXh. Specifies the core frequency multiplier. The core COF is a function of CpuFid and CpuDid, and defined by CoreCOF.

ValidValues:

Value	Description
0Fh-00h	Reserved.
FFh-10h	<Value>*25

MSRC001_0073 [C-state Base Address] (Core::X86::Msr::CStateBaseAddr)

Read-write. Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_0073

Bits	Description
63:16	Reserved.
15:0	**CstateAddr**: **C-state address**. Read-write. Reset: 0000h. Specifies the IO addresses trapped by the core for C-state entry requests. A value of 0 in this field specifies that the core does not trap any IO addresses for C-state entry. Writing values greater than FFF8h into this field result in undefined behavior. All other values cause the core to trap IO addresses CstateAddr through CstateAddr + 7.

MSRC001_0074 [CPU Watchdog Timer] (Core::X86::Msr::CpuWdtCfg)

Read-write. Reset: 0000_0000_0000_0280h.

_ccd[7:0]_lthree0_core[7:0]; MSRC001_0074

<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>63:10</td><td>Reserved.</td></tr>
<tr><td rowspan="2">9:7</td><td>CpuWdTmrCfgSeverity. Read-write. Reset: 5h. Specifies the CPU Watch Dog Timer severity.</td></tr>
<tr><td>ValidValues:<table><tr><th>Value</th><th>Description</th></tr><tr><td>4h-0h</td><td>Reserved.</td></tr><tr><td>5h</td><td>MCA_EXSC_ERROR_SEVERITY_FATAL</td></tr><tr><td>7h-6h</td><td>Reserved.</td></tr></table></td></tr>
<tr><td>6:3</td><td>Reserved.</td></tr>
<tr><td rowspan="2">2:1</td><td>CpuWdTmrTimebaseSel: CPU watchdog timer time base. Read-write. Reset: 0h. Specifies the time base for the timeout period specified in CpuWdtCountSel.</td></tr>
<tr><td>ValidValues:<table><tr><th>Value</th><th>Description</th></tr><tr><td>0h</td><td>1.31ms</td></tr><tr><td>1h</td><td>1.28us</td></tr><tr><td>3h-2h</td><td>Reserved</td></tr></table></td></tr>
<tr><td>0</td><td>CpuWdTmrCfgEn: CPU watchdog timer enable. Read-write. Reset: 0. Init: BIOS,1. 1=The WDT is enabled.</td></tr>
</table>

MSRC001_0111 [SMM Base Address] (Core::X86::Msr::SMM_BASE)

Reset: 0000_0000_0003_0000h.

This holds the base of the SMM memory region. The value of this register is stored in the save state on entry into SMM (see 2.1.14.1.5 [SMM Save State]) and it is restored on returning from SMM. The 16-bit CS (code segment) selector is loaded with SmmBase[19:4] on entering SMM. SmmBase[3:0] is required to be 0. The SMM base address can be changed in two ways:

- The SMM base address, at offset FF00h in the SMM state save area, may be changed by the SMI handler. The RSM instruction updates SmmBase with the new value.
- Normal WRMSR access to this register.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_0111

Bits	Description
63:32	Reserved.
31:0	**SmmBase**. Reset: 0003_0000h. Base address of the SMM memory region.
	AccessType: Core::X86::Msr::HWCR[SmmLock] ? Read-only : Read-write.

MSRC001_0112 [SMM TSeg Base Address] (Core::X86::Msr::SMMAddr)

Configurable. Reset: 0000_0000_0000_0000h.

See 2.1.14.1 [System Management Mode (SMM)] and 2.1.7.3.1 [Memory Access to the Physical Address Space]. See Core::X86::Msr::SMMMask for more information about the ASeg and TSeg address ranges.

Each CPU access, directed at CPUAddr, is determined to be in the TSeg range if the following is true:

CPUAddr[47:17] & TSegMask[47:17] == TSegBase[47:17] & TSegMask[47:17].

For example, if TSeg spans 256 KB and starts at the 1-MB address. The Core::X86::Msr::SMMAddr[TSegBase[47:17]] would be set to 0010_0000h and the Core::X86::Msr::SMMMask[TSegMask[47:17]] to FFFC_0000h (with zeros filling in for bits[16:0]). This results in a TSeg range from 0010_0000 to 0013_FFFFh.

_ccd[7:0]_lthree0_core[7:0]; MSRC001_0112

Bits	Description
63:48	Reserved.

47:17	**TSegBase[47:17]**: **TSeg address range base**. Configurable. Reset: 0000_0000h. AccessType: (Core::X86::Msr::HWCR[SmmLock]) ? Read-only : Read-write.
16:0	Reserved.

MSRC001_0113 [SMM TSeg Mask] (Core::X86::Msr::SMMMask)

Configurable. Reset: 0000_0000_0000_0000h.

See 2.1.14.1 [System Management Mode (SMM)].
The ASeg address range is located at a fixed address from A0000h–BFFFFh. The TSeg range is located at a variable base (specified by Core::X86::Msr::SMMAddr[TSegBase[47:17]]) with a variable size (specified by Core::X86::Msr::SMMMask[TSegMask[47:17]]). These ranges provide a safe location for SMM code and data that is not readily accessible by non-SMM applications. The SMI handler can be located in one of these two ranges, or it can be located outside these ranges. These ranges must never overlap each other.

This register specifies how accesses to the ASeg and TSeg address ranges are controlled as follows:

- If [A,T]Valid == 1, then:
    - If in SMM, then:
        - If [A, T]Close == 0, then the accesses are directed to DRAM with memory type as specified in [A, T]MTypeDram.
        - If [A, T]Close == 1, then instruction accesses are directed to DRAM with memory type as specified in [A, T]MTypeDram and data accesses are directed at MMIO space and with attributes based on [A, T]MTypeIoWc.
    - If not in SMM, then the accesses are directed at MMIO space with attributes based on [A,T]MTypeIoWc.
- See 2.1.7.3.1.1 [Determining Memory Type].

_ccd[7:0]_lthree0_core[7:0]; MSRC001_0113

Bits	Description
63:48	Reserved.
47:17	**TSegMask[47:17]**: **TSeg address range mask**. Configurable. Reset: 0000_0000h. See Core::X86::Msr::SMMAddr. AccessType: (Core::X86::Msr::HWCR[SmmLock]) ? Read-only : Read-write.
16:15	Reserved.
14:12	**TMTypeDram**: **TSeg address range memory type**. Configurable. Reset: 0h. Specifies the memory type for SMM accesses to the TSeg range that are directed to DRAM. AccessType: (Core::X86::Msr::HWCR[SmmLock]) ? Read-only : Read-write.

ValidValues:

Value	Description
0h	UC or uncacheable.
1h	WC or write combining.
3h-2h	Reserved.
4h	WT or write through.
5h	WP or write protect.
6h	WB or write back.
7h	Reserved.

Bits	Description
11	Reserved.
10:8	**AMTypeDram**: **ASeg Range Memory Type**. Configurable. Reset: 0h. Specifies the memory type for SMM accesses to the ASeg range that are directed to DRAM. AccessType: (Core::X86::Msr::HWCR[SmmLock]) ? Read-only : Read-write.

ValidValues:

Value	Description
0h	UC or uncacheable.
1h	WC or write combining.

3h-2h	Reserved.
4h	WT or write through.
5h	WP or write protect.
6h	WB or write back.
7h	Reserved.

7:6	Reserved.
5	**TMTypeIoWc**: **non-SMM TSeg address range memory type**. Configurable. Reset: 0. 0=UC (uncacheable). 1=WC (write combining). Specifies the attribute of TSeg accesses that are directed to MMIO space. AccessType: (Core::X86::Msr::HWCR[SmmLock]) ? Read-only : Read-write.
4	**AMTypeIoWc**: **non-SMM ASeg address range memory type**. Configurable. Reset: 0. 0=UC (uncacheable). 1=WC (write combining). Specifies the attribute of ASeg accesses that are directed to MMIO space. AccessType: (Core::X86::Msr::HWCR[SmmLock]) ? Read-only : Read-write.
3	**TClose**: **send TSeg address range data accesses to MMIO**. Configurable. Reset: 0. 1=When in SMM, direct data accesses in the TSeg address range to MMIO space. See AClose. AccessType: (Core::X86::Msr::HWCR[SmmLock]) ? Read-only : Read-write.
2	**AClose**: **send ASeg address range data accesses to MMIO**. Configurable. Reset: 0. 1=When in SMM, direct data accesses in the ASeg address range to MMIO space. [A,T]Close allows the SMI handler to access the MMIO space located in the same address region as the [A,T]Seg. When the SMI handler is finished accessing the MMIO space, it must clear the bit. Failure to do so before resuming from SMM causes the CPU to erroneously read the save state from MMIO space. AccessType: (Core::X86::Msr::HWCR[SmmLock]) ? Read-only : Read-write.
1	**TValid**: **enable TSeg SMM address range**. Configurable. Reset: 0. 1=The TSeg address range SMM enabled. AccessType: (Core::X86::Msr::HWCR[SmmLock]) ? Read-only : Read-write.
0	**AValid**: **enable ASeg SMM address range**. Configurable. Reset: 0. 1=The ASeg address range SMM enabled. AccessType: (Core::X86::Msr::HWCR[SmmLock]) ? Read-only : Read-write.

MSRC001_0114 [Virtual Machine Control] (Core::X86::Msr::VM_CR)

Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_0114

Bits	Description
63:5	Reserved.
4	**SvmeDisable**: **SVME disable**. Configurable. Reset: 0. 0=Core::X86::Msr::EFER[SVME] is read-write. 1=Core::X86::Msr::EFER[SVME] is Read-only,Error-on-write-1. See Lock for the access type of this field. Attempting to set this field when (Core::X86::Msr::EFER[SVME]==1) causes a #GP fault, regardless of the state of Lock. See the docAPM2 section titled "Enabling SVM" for software use of this field.
3	**Lock**: **SVM lock**. Read-only,Volatile. Reset: 0. 0=SvmeDisable is read-write. 1=SvmeDisable is read-only. See Core::X86::Msr::SvmLockKey[SvmLockKey] for the condition that causes hardware to clear this field.
2	Reserved.
1	**InterceptInit**: **intercept INIT**. Read-write,Volatile. Reset: 0. 0=INIT delivered normally. 1=INIT translated into a SX interrupt. This bit controls how INIT is delivered in host mode. This bit is set by hardware when the SKINIT instruction is executed.
0	Reserved.

MSRC001_0115 [IGNNE] (Core::X86::Msr::IGNNE)

Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_0115

Bits	Description
63:1	Reserved.
0	**IGNNE**: **current IGNNE state**. Read-write. Reset: 0. This bit controls the current state of the processor internal IGNNE signal.

MSRC001_0116 [SMM Control] (Core::X86::Msr::SMM_CTL)

Reset: 0000_0000_0000_0000h.

The bits in this register are processed in the order of: SmmEnter, SmiCycle, SmmDismiss, RsmCycle and SmmExit. However, only the following combination of bits may be set in a single write (all other combinations result in undefined behavior):

- SmmEnter and SmiCycle.
- SmmEnter and SmmDismiss.
- SmmEnter, SmiCycle and SmmDismiss.
- SmmExit and RsmCycle.

Software is responsible for ensuring that SmmEnter and SmmExit operations are properly matched and are not nested.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_0116

Bits	Description
63:5	Reserved.
4	**RsmCycle**: **send RSM special cycle**. Reset: 0. 1=Send a RSM special cycle.
	AccessType: Core::X86::Msr::HWCR[SmmLock] ? Error-on-read,Error-on-write : Write-only,Error-on-read.
3	**SmmExit**: **exit SMM**. Reset: 0. 1=Exit SMM.
	AccessType: Core::X86::Msr::HWCR[SmmLock] ? Error-on-read,Error-on-write : Write-only,Error-on-read.
2	**SmiCycle**: **send SMI special cycle**. Reset: 0. 1=Send a SMI special cycle.
	AccessType: Core::X86::Msr::HWCR[SmmLock] ? Error-on-read,Error-on-write : Write-only,Error-on-read.
1	**SmmEnter**: **enter SMM**. Reset: 0. 1=Enter SMM.
	AccessType: Core::X86::Msr::HWCR[SmmLock] ? Error-on-read,Error-on-write : Write-only,Error-on-read.
0	**SmmDismiss**: **clear SMI**. Reset: 0. 1=Clear the SMI pending flag.
	AccessType: Core::X86::Msr::HWCR[SmmLock] ? Error-on-read,Error-on-write : Write-only,Error-on-read.

MSRC001_0117 [Virtual Machine Host Save Physical Address] (Core::X86::Msr::VM_HSAVE_PA)

Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_0117

Bits	Description
63:48	Reserved.
47:12	**VM_HSAVE_PA**: **physical address of host save area**. Read-write. Reset: 0_0000_0000h. This register contains the physical address of a 4-KB region where VMRUN saves host state and where vm-exit restores host state from. Writing this register causes a #GP if (FFFF_FFFF_Fh >= VM_HSAVE_PA >= FFFD_0000_0h) or if either the TSEG or ASEG regions overlap with the range defined by this register.
11:0	Reserved.

MSRC001_0118 [SVM Lock Key] (Core::X86::Msr::SvmLockKey)

Read-write. Reset: Fixed,0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_0118

Bits	Description
63:0	**SvmLockKey**: **SVM lock key**. Read-write. Reset: Fixed,0000_0000_0000_0000h. Writes to this register when (Core::X86::Msr::VM_CR[Lock] == 0) modify SvmLockKey. If ((Core::X86::Msr::VM_CR[Lock] == 1) && (SvmLockKey!=0) && (The write value == The value stored in SvmLockKey)) for a write to this register then hardware updates Core::X86::Msr::VM_CR[Lock]=0.

MSRC001_011A [Local SMI Status] (Core::X86::Msr::LocalSmiStatus)

Read-write. Reset: 0000_0000_0000_0000h.

This register returns the same information that is returned in Core::X86::Smm::LocalSmiStatus portion of the SMM save state. The information in this register is only updated when Core::X86::Msr::SMM_CTL[SmmDismiss] is set by software.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_011A

Bits	Description
63:32	Reserved.
31:0	**LocalSmiStatus**. Read-write. Reset: 0000_0000h. See Core::X86::Smm::LocalSmiStatus.

MSRC001_011B [AVIC Doorbell] (Core::X86::Msr::AvicDoorbell)

Write-only,Error-on-read. Reset: 0000_0000_0000_0000h.

The ApicId is a physical APIC Id; not valid for logical APIC ID.
Enable: (Core::X86::Cpuid::SvmRevFeatIdEdx[AVIC]==1).

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_011B

Bits	Description
63:32	Reserved.
31:0	**ApicId**: **APIC ID [31:0]**. Write-only,Error-on-read. Reset: 0000_0000h. The value written must be a valid physical APID_ID.

MSRC001_011E [VM Page Flush] (Core::X86::Msr::VMPAGE_FLUSH)

Write-only,Error-on-read.

Writes to this MSR causes 4 KBs of encrypted, guest-tagged data to be flushed from caches if present. This MSR is write-only, and can only be written from ASID=0 code and only if Core::X86::Msr::SYS_CFG[SMEE]=1.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_011E

Bits	Description
63:12	**GuestPhysicalAddr**. Write-only,Error-on-read. Reset: X_XXXX_XXXX_XXXXh. Guest physical address of page to flush.
11:0	**ASID**. Write-only,Error-on-read. Reset: XXXh. ASID to use for flush. This value must be within the legal ASID range indicated by CPUID_Fn8000001F_ECX (Core::X86::Cpuid::SecureEncryptionEcx), and cannot be zero.

MSRC001_0130 [Guest Host Communication Block] (Core::X86::Msr::GHCB)

Read-write. Reset: 0000_0000_0000_0000h.

If Core::X86::Msr::GHCB is accessed in hypervisor mode, #GP is generated.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_0130

Bits	Description
63:0	**GHCBPA**. Read-write. Reset: 0000_0000_0000_0000h. Guest physical address of GHCB.

MSRC001_0131 [SEV Status] (Core::X86::Msr::SEV_Status)

Read,Error-on-write. Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_0131

Bits	Description
63:10	Reserved.
9	**SNPBTBIsolation**. Read,Error-on-write. Reset: 0. 1=BTB predictor isolation is enabled for this guest.
8	Reserved.
7	**DebugSwapSupport**. Read,Error-on-write. Reset: 0. 1=Extra debug registers are swapped for this guest.
6	**AlternateInjection**. Read,Error-on-write. Reset: 0. 1=Alternate Injection feature is enabled for this guest (encrypted VMSA fields used to provide injection information).
5	**RestrictInjection**. Read,Error-on-write. Reset: 0. 1=Restrict Injection feature is enabled for this guest (only #HV can be injected).
4	**ReflectVC**. Read,Error-on-write. Reset: 0. 1= #VC exceptions are turned into an AE VMEXIT for this guest.
3	**VirtualTOM**. Read,Error-on-write. Reset: 0. 1=Virtual TOM feature is enabled for this guest.
2	**SNPActive**. Read,Error-on-write. Reset: 0. 1=Secure Nested Paging is active for this guest.
1	**SevEsEnabled**. Read,Error-on-write. Reset: 0. 1=The guest was launched with the Sev-ES feature enabled in VMCB offset 90h.
0	**SevEnabled**. Read,Error-on-write. Reset: 0. 1=The guest was launched with SEV feature enabled in VMCB offset 90h.

MSRC001_0132 [RMP Base] (Core::X86::Msr::LS_RMP_BASE)

Secure Nested Paging Reverse Map Table Base.

_ccd[7:0]_lthree0_core[7:0]; MSRC001_0132

Bits	Description

63:52	Reserved.
51:13	**RMP_BASE**. Reset: 00_0000_0000h. Secure Nested Paging Reverse Map Table Base.
	AccessType: Core::X86::Msr::SYS_CFG[SecureNestedPagingEn] ? Read,Error-on-write : Read-write.
12:0	Reserved.

MSRC001_0133 [RMP End] (Core::X86::Msr::LS_RMP_END)

Secure Nested Paging Reverse Map Table Limit.

_ccd[7:0]_lthree0_core[7:0]; MSRC001_0133

Bits	Description
63:52	Reserved.
51:13	**RMP_END**. Reset: 00_0000_0000h. Secure Nested Paging Reverse Map Table Limit.
	AccessType: Core::X86::Msr::SYS_CFG[SecureNestedPagingEn] ? Read,Error-on-write : Read-write.
12:0	Reserved.

MSRC001_0140 [OS Visible Work-around Length] (Core::X86::Msr::OSVW_ID_Length)

Read-write. Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_0140

Bits	Description
63:16	Reserved.
15:0	**OSVWIdLength**: **OS visible work-around ID length**. Read-write. Reset: 0000h. See the Revision Guide for the definition of this field; see 1.2 [Reference Documents].

MSRC001_0141 [OS Visible Work-around Status] (Core::X86::Msr::OSVW_Status)

Read-write. Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_0141

Bits	Description
63:0	**OsvwStatusBits**: **OS visible work-around status bits**. Read-write. Reset: 0000_0000_0000_0000h. See the Revision Guide for the definition of this field; see 1.2 [Reference Documents].

MSRC001_0200 [Performance Event Select 0] (Core::X86::Msr::PERF_CTL0)

Read-write. Reset: 0000_0000_0000_0000h.

See 2.1.17 [Performance Monitor Counters]. Core::X86::Msr::PERF_LEGACY_CTL0 is an alias of this register.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_0200

Bits	Description
63:42	Reserved.
41:40	**HostGuestOnly**: **count only host/guest events**. Read-write. Reset: 0h.
	ValidValues:

Value	Description
0h	Count all events, irrespective of guest/host.
1h	Count guest events if [SVME] == 1.
2h	Count host events if [SVME] == 1.
3h	Count all guest and host events if [SVME] == 1.

Bits	Description
39:36	Reserved.
35:32	**EventSelect[11:8]**. Read-write. Reset: 0h. Performance event select[11:8].
31:24	**CntMask**: **counter mask**. Read-write. Reset: 00h. Controls the number of events counted per clock cycle.
	ValidValues:

Value	Description
00h	The corresponding PERF_CTR[5:0] register increments by the number of events occurring in a clock cycle. See 2.1.17.3 [Large Increment per Cycle Events] for events that can increment greater than 15 per cycle.
7Fh-01h	When Inv == 0, the corresponding PERF_CTR[5:0] register increments by 1, if the number of events

	occurring in a clock cycle is greater than or equal to the CntMask value. When Inv == 1, the corresponding PERF_CTR[5:0] register increments by 1, if the number of events occurring in a clock cycle is less than CntMask value.
FFh-80h	Reserved.

23	**Inv**: **invert counter mask**. Read-write. Reset: 0. See CntMask.
22	**En**: **enable performance counter**. Read-write. Reset: 0. 1=Performance event counter is enabled.
21	Reserved.
20	**Int**: **enable APIC interrupt**. Read-write. Reset: 0. 1=APIC performance counter LVT interrupt is enabled to generate an interrupt via Core::X86::Apic::PerformanceCounterLvtEntry when the performance counter overflows.
19	Reserved.
18	**Edge**: **edge detect**. Read-write. Reset: 0. 0=Level detect. 1=Zero-to-one Edge detect. Read-write. The edge count mode increments the counter when a transition happens on the monitored event. If the event selected is changed without disabling the counter, an extra edge is falsely detected when the first event is a static 0 and the second event is a static one. To avoid this false edge detection, disable the counter when changing the event and then enable the counter with a second MSR write.
17:16	**OsUserMode**: **OS and user mode**. Read-write. Reset: 0h.
	ValidValues:

Value	Description
0h	Count no events.
1h	Count user events (CPL>0).
2h	Count OS events (CPL=0).
3h	Count all events, irrespective of the CPL.

15:8	**UnitMask**: **event qualification**. Read-write. Reset: 00h. Each UnitMask bit further specifies or qualifies the event specified by EventSelect. All events selected by UnitMask are simultaneously monitored. Unless otherwise stated, the UnitMask values shown may be combined (logically ORed) to select any desired combination of the sub-events for a given event. In some cases, certain combinations can result in misleading counts, or the UnitMask value is an ordinal rather than a bit mask. These situations are described where applicable, or should be obvious from the event descriptions. For events where no UnitMask table is shown, the UnitMask is Unused.
7:0	**EventSelect[7:0]**: **event select**. Read-write. Reset: 00h. EventSelect[11:0] = {EventSelect[11:8], EventSelect[7:0]}. EventSelect specifies the event or event duration in a processor unit to be counted by the corresponding PERF_CTR[5:0] register. The events are specified in 2.1.17.4 [Core Performance Monitor Counters]. Some events are reserved; when a reserved event is selected, the results are undefined.

MSRC001_020[1...B] [Performance Event Counter [5:0]] (Core::X86::Msr::PERF_CTR)

Note: When counting events that capable of counting greater than 15 events per cycle (MergeEvent) the even and the corresponding odd PERF_CTR must be paired to appear as a single 64 bit counter. See 2.1.17.3 [Large Increment per Cycle Events].
See Core::X86::Msr::PERF_CTL0..5. Core::X86::Msr::PERF_LEGACY_CTR is an alias of MSRC001_020[7,5,3,1].
Also can be Read via x86 instructions RDPMC ECX = [05:00].

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n0; MSRC001_0201
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n1; MSRC001_0203
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n2; MSRC001_0205
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n3; MSRC001_0207
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n4; MSRC001_0209
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n5; MSRC001_020B

Bits	Description
63:48	Reserved.
47:0	**CTR**. Read-write,Volatile. Reset: 0000_0000_0000h. Performance counter value.

MSRC001_0202 [Performance Event Select 1] (Core::X86::Msr::PERF_CTL1)

<table>
<tr><td colspan="2">Read-write. Reset: 0000_0000_0000_0000h.</td></tr>
<tr><td colspan="2">See 2.1.17 [Performance Monitor Counters]. Core::X86::Msr::PERF_LEGACY_CTL1 is an alias of this register.</td></tr>
<tr><td colspan="2">_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_0202</td></tr>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>63:42</td><td>Reserved.</td></tr>
<tr><td rowspan="2">41:40</td><td>HostGuestOnly: count only host/guest events. Read-write. Reset: 0h.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>Count all events, irrespective of guest/host.</td></tr>
<tr><td>1h</td><td>Count guest events if [SVME] == 1.</td></tr>
<tr><td>2h</td><td>Count host events if [SVME] == 1.</td></tr>
<tr><td>3h</td><td>Count all guest and host events if [SVME] == 1.</td></tr>
</table></td></tr>
<tr><td>39:36</td><td>Reserved.</td></tr>
<tr><td>35:32</td><td>EventSelect[11:8]. Read-write. Reset: 0h. Performance event select[11:8].</td></tr>
<tr><td rowspan="2">31:24</td><td>CntMask: counter mask. Read-write. Reset: 00h. Controls the number of events counted per clock cycle.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>00h</td><td>The corresponding PERF_CTR[5:0] register increments by the number of events occurring in a clock cycle. See 2.1.17.3 [Large Increment per Cycle Events] for events that can increment greater than 15 per cycle.</td></tr>
<tr><td>7Fh-01h</td><td>When Inv == 0, the corresponding PERF_CTR[5:0] register increments by 1, if the number of events occurring in a clock cycle is greater than or equal to the CntMask value. When Inv == 1, the corresponding PERF_CTR[5:0] register increments by 1, if the number of events occurring in a clock cycle is less than CntMask value.</td></tr>
<tr><td>FFh-80h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>23</td><td>Inv: invert counter mask. Read-write. Reset: 0. See CntMask.</td></tr>
<tr><td>22</td><td>En: enable performance counter. Read-write. Reset: 0. 1=Performance event counter is enabled.</td></tr>
<tr><td>21</td><td>Reserved.</td></tr>
<tr><td>20</td><td>Int: enable APIC interrupt. Read-write. Reset: 0. 1=APIC performance counter LVT interrupt is enabled to generate an interrupt via Core::X86::Apic::PerformanceCounterLvtEntry when the performance counter overflows.</td></tr>
<tr><td>19</td><td>Reserved.</td></tr>
<tr><td>18</td><td>Edge: edge detect. Read-write. Reset: 0. 0=Level detect. 1=Zero-to-one Edge detect. Read-write. The edge count mode increments the counter when a transition happens on the monitored event. If the event selected is changed without disabling the counter, an extra edge is falsely detected when the first event is a static 0 and the second event is a static one. To avoid this false edge detection, disable the counter when changing the event and then enable the counter with a second MSR write.</td></tr>
<tr><td rowspan="2">17:16</td><td>OsUserMode: OS and user mode. Read-write. Reset: 0h.</td></tr>
<tr><td>ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>Count no events.</td></tr>
<tr><td>1h</td><td>Count user events (CPL>0).</td></tr>
<tr><td>2h</td><td>Count OS events (CPL=0).</td></tr>
<tr><td>3h</td><td>Count all events, irrespective of the CPL.</td></tr>
</table></td></tr>
<tr><td>15:8</td><td>UnitMask: event qualification. Read-write. Reset: 00h. Each UnitMask bit further specifies or qualifies the event specified by EventSelect. All events selected by UnitMask are simultaneously monitored. Unless otherwise stated, the UnitMask values shown may be combined (logically ORed) to select any desired combination of the sub-events for a given event. In some cases, certain combinations can result in misleading counts, or the UnitMask value is an ordinal rather than a bit mask. These situations are described where applicable, or should be</td></tr>
</table>

	obvious from the event descriptions. For events where no UnitMask table is shown, the UnitMask is Unused.
7:0	**EventSelect[7:0]**: **event select**. Read-write. Reset: 00h. EventSelect[11:0] = {EventSelect[11:8], EventSelect[7:0]}. EventSelect specifies the event or event duration in a processor unit to be counted by the corresponding PERF_CTR[5:0] register. The events are specified in 2.1.17.4 [Core Performance Monitor Counters]. Some events are reserved; when a reserved event is selected, the results are undefined.

MSRC001_0204 [Performance Event Select 2] (Core::X86::Msr::PERF_CTL2)

<table>
<tr><td colspan="2">Read-write. Reset: 0000_0000_0000_0000h.</td></tr>
<tr><td colspan="2">See 2.1.17 [Performance Monitor Counters]. Core::X86::Msr::PERF_LEGACY_CTL2 is an alias of this register.</td></tr>
<tr><td colspan="2">_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_0204</td></tr>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>63:42</td><td>Reserved.</td></tr>
<tr><td>41:40</td><td>HostGuestOnly: count only host/guest events. Read-write. Reset: 0h.
ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>Count all events, irrespective of guest/host.</td></tr>
<tr><td>1h</td><td>Count guest events if [SVME] == 1.</td></tr>
<tr><td>2h</td><td>Count host events if [SVME] == 1.</td></tr>
<tr><td>3h</td><td>Count all guest and host events if [SVME] == 1.</td></tr>
</table></td></tr>
<tr><td>39:36</td><td>Reserved.</td></tr>
<tr><td>35:32</td><td>EventSelect[11:8]. Read-write. Reset: 0h. Performance event select[11:8].</td></tr>
<tr><td>31:24</td><td>CntMask: counter mask. Read-write. Reset: 00h. Controls the number of events counted per clock cycle.
ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>00h</td><td>The corresponding PERF_CTR[5:0] register increments by the number of events occurring in a clock cycle. See 2.1.17.3 [Large Increment per Cycle Events] for events that can increment greater than 15 per cycle.</td></tr>
<tr><td>7Fh-01h</td><td>When Inv == 0, the corresponding PERF_CTR[5:0] register increments by 1, if the number of events occurring in a clock cycle is greater than or equal to the CntMask value. When Inv == 1, the corresponding PERF_CTR[5:0] register increments by 1, if the number of events occurring in a clock cycle is less than CntMask value.</td></tr>
<tr><td>FFh-80h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>23</td><td>Inv: invert counter mask. Read-write. Reset: 0. See CntMask.</td></tr>
<tr><td>22</td><td>En: enable performance counter. Read-write. Reset: 0. 1=Performance event counter is enabled.</td></tr>
<tr><td>21</td><td>Reserved.</td></tr>
<tr><td>20</td><td>Int: enable APIC interrupt. Read-write. Reset: 0. 1=APIC performance counter LVT interrupt is enabled to generate an interrupt via Core::X86::Apic::PerformanceCounterLvtEntry when the performance counter overflows.</td></tr>
<tr><td>19</td><td>Reserved.</td></tr>
<tr><td>18</td><td>Edge: edge detect. Read-write. Reset: 0. 0=Level detect. 1=Zero-to-one Edge detect. Read-write. The edge count mode increments the counter when a transition happens on the monitored event. If the event selected is changed without disabling the counter, an extra edge is falsely detected when the first event is a static 0 and the second event is a static one. To avoid this false edge detection, disable the counter when changing the event and then enable the counter with a second MSR write.</td></tr>
<tr><td>17:16</td><td>OsUserMode: OS and user mode. Read-write. Reset: 0h.
ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>Count no events.</td></tr>
<tr><td>1h</td><td>Count user events (CPL>0).</td></tr>
</table></td></tr>
</table>

<table>
<tr><td></td><td><table><tr><td>2h</td><td>Count OS events (CPL=0).</td></tr><tr><td>3h</td><td>Count all events, irrespective of the CPL.</td></tr></table></td></tr>
<tr><td>15:8</td><td>UnitMask: event qualification. Read-write. Reset: 00h. Each UnitMask bit further specifies or qualifies the event specified by EventSelect. All events selected by UnitMask are simultaneously monitored. Unless otherwise stated, the UnitMask values shown may be combined (logically ORed) to select any desired combination of the sub-events for a given event. In some cases, certain combinations can result in misleading counts, or the UnitMask value is an ordinal rather than a bit mask. These situations are described where applicable, or should be obvious from the event descriptions. For events where no UnitMask table is shown, the UnitMask is Unused.</td></tr>
<tr><td>7:0</td><td>EventSelect[7:0]: event select. Read-write. Reset: 00h. EventSelect[11:0] = {EventSelect[11:8], EventSelect[7:0]}. EventSelect specifies the event or event duration in a processor unit to be counted by the corresponding PERF_CTR[5:0] register. The events are specified in 2.1.17.4 [Core Performance Monitor Counters]. Some events are reserved; when a reserved event is selected, the results are undefined.</td></tr>
</table>

MSRC001_0206 [Performance Event Select 3] (Core::X86::Msr::PERF_CTL3)

<table>
<tr><td colspan="2">Read-write. Reset: 0000_0000_0000_0000h.</td></tr>
<tr><td colspan="2">See 2.1.17 [Performance Monitor Counters]. Core::X86::Msr::PERF_LEGACY_CTL3 is an alias of this register.</td></tr>
<tr><td colspan="2">_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_0206</td></tr>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>63:42</td><td>Reserved.</td></tr>
<tr><td>41:40</td><td>HostGuestOnly: count only host/guest events. Read-write. Reset: 0h.
ValidValues:<table><tr><th>Value</th><th>Description</th></tr><tr><td>0h</td><td>Count all events, irrespective of guest/host.</td></tr><tr><td>1h</td><td>Count guest events if [SVME] == 1.</td></tr><tr><td>2h</td><td>Count host events if [SVME] == 1.</td></tr><tr><td>3h</td><td>Count all guest and host events if [SVME] == 1.</td></tr></table></td></tr>
<tr><td>39:36</td><td>Reserved.</td></tr>
<tr><td>35:32</td><td>EventSelect[11:8]. Read-write. Reset: 0h. Performance event select[11:8].</td></tr>
<tr><td>31:24</td><td>CntMask: counter mask. Read-write. Reset: 00h. Controls the number of events counted per clock cycle.
ValidValues:<table><tr><th>Value</th><th>Description</th></tr><tr><td>00h</td><td>The corresponding PERF_CTR[5:0] register increments by the number of events occurring in a clock cycle. See 2.1.17.3 [Large Increment per Cycle Events] for events that can increment greater than 15 per cycle.</td></tr><tr><td>7Fh-01h</td><td>When Inv == 0, the corresponding PERF_CTR[5:0] register increments by 1, if the number of events occurring in a clock cycle is greater than or equal to the CntMask value. When Inv == 1, the corresponding PERF_CTR[5:0] register increments by 1, if the number of events occurring in a clock cycle is less than CntMask value.</td></tr><tr><td>FFh-80h</td><td>Reserved.</td></tr></table></td></tr>
<tr><td>23</td><td>Inv: invert counter mask. Read-write. Reset: 0. See CntMask.</td></tr>
<tr><td>22</td><td>En: enable performance counter. Read-write. Reset: 0. 1=Performance event counter is enabled.</td></tr>
<tr><td>21</td><td>Reserved.</td></tr>
<tr><td>20</td><td>Int: enable APIC interrupt. Read-write. Reset: 0. 1=APIC performance counter LVT interrupt is enabled to generate an interrupt via Core::X86::Apic::PerformanceCounterLvtEntry when the performance counter overflows.</td></tr>
<tr><td>19</td><td>Reserved.</td></tr>
<tr><td>18</td><td>Edge: edge detect. Read-write. Reset: 0. 0=Level detect. 1=Zero-to-one Edge detect. Read-write. The edge count mode increments the counter when a transition happens on the monitored event. If the event selected is changed without disabling the counter, an extra edge is falsely detected when the first event is a static 0 and the second</td></tr>
</table>

<table>
<tr><td></td><td>event is a static one. To avoid this false edge detection, disable the counter when changing the event and then enable the counter with a second MSR write.</td></tr>
<tr><td>17:16</td><td>OsUserMode: OS and user mode. Read-write. Reset: 0h.
ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>Count no events.</td></tr>
<tr><td>1h</td><td>Count user events (CPL>0).</td></tr>
<tr><td>2h</td><td>Count OS events (CPL=0).</td></tr>
<tr><td>3h</td><td>Count all events, irrespective of the CPL.</td></tr>
</table></td></tr>
<tr><td>15:8</td><td>UnitMask: event qualification. Read-write. Reset: 00h. Each UnitMask bit further specifies or qualifies the event specified by EventSelect. All events selected by UnitMask are simultaneously monitored. Unless otherwise stated, the UnitMask values shown may be combined (logically ORed) to select any desired combination of the sub-events for a given event. In some cases, certain combinations can result in misleading counts, or the UnitMask value is an ordinal rather than a bit mask. These situations are described where applicable, or should be obvious from the event descriptions. For events where no UnitMask table is shown, the UnitMask is Unused.</td></tr>
<tr><td>7:0</td><td>EventSelect[7:0]: event select. Read-write. Reset: 00h. EventSelect[11:0] = {EventSelect[11:8], EventSelect[7:0]}. EventSelect specifies the event or event duration in a processor unit to be counted by the corresponding PERF_CTR[5:0] register. The events are specified in 2.1.17.4 [Core Performance Monitor Counters]. Some events are reserved; when a reserved event is selected, the results are undefined.</td></tr>
</table>

MSRC001_0208 [Performance Event Select 4] (Core::X86::Msr::PERF_CTL4)

<table>
<tr><td colspan="2">Read-write. Reset: 0000_0000_0000_0000h.</td></tr>
<tr><td colspan="2">See 2.1.17 [Performance Monitor Counters].</td></tr>
<tr><td colspan="2">_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_0208</td></tr>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>63:42</td><td>Reserved.</td></tr>
<tr><td>41:40</td><td>HostGuestOnly: count only host/guest events. Read-write. Reset: 0h.
ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>Count all events, irrespective of guest/host.</td></tr>
<tr><td>1h</td><td>Count guest events if [SVME] == 1.</td></tr>
<tr><td>2h</td><td>Count host events if [SVME] == 1.</td></tr>
<tr><td>3h</td><td>Count all guest and host events if [SVME] == 1.</td></tr>
</table></td></tr>
<tr><td>39:36</td><td>Reserved.</td></tr>
<tr><td>35:32</td><td>EventSelect[11:8]. Read-write. Reset: 0h. Performance event select[11:8].</td></tr>
<tr><td>31:24</td><td>CntMask: counter mask. Read-write. Reset: 00h. Controls the number of events counted per clock cycle.
ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>00h</td><td>The corresponding PERF_CTR[5:0] register increments by the number of events occurring in a clock cycle. See 2.1.17.3 [Large Increment per Cycle Events] for events that can increment greater than 15 per cycle.</td></tr>
<tr><td>7Fh-01h</td><td>When Inv == 0, the corresponding PERF_CTR[5:0] register increments by 1, if the number of events occurring in a clock cycle is greater than or equal to the CntMask value. When Inv == 1, the corresponding PERF_CTR[5:0] register increments by 1, if the number of events occurring in a clock cycle is less than CntMask value.</td></tr>
<tr><td>FFh-80h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>23</td><td>Inv: invert counter mask. Read-write. Reset: 0. See CntMask.</td></tr>
<tr><td>22</td><td>En: enable performance counter. Read-write. Reset: 0. 1=Performance event counter is enabled.</td></tr>
<tr><td>21</td><td>Reserved.</td></tr>
</table>

<table>
<tr><td>20</td><td>Int: enable APIC interrupt. Read-write. Reset: 0. 1=APIC performance counter LVT interrupt is enabled to generate an interrupt via Core::X86::Apic::PerformanceCounterLvtEntry when the performance counter overflows.</td></tr>
<tr><td>19</td><td>Reserved.</td></tr>
<tr><td>18</td><td>Edge: edge detect. Read-write. Reset: 0. 0=Level detect. 1=Zero-to-one Edge detect. Read-write. The edge count mode increments the counter when a transition happens on the monitored event. If the event selected is changed without disabling the counter, an extra edge is falsely detected when the first event is a static 0 and the second event is a static one. To avoid this false edge detection, disable the counter when changing the event and then enable the counter with a second MSR write.</td></tr>
<tr><td>17:16</td><td>OsUserMode: OS and user mode. Read-write. Reset: 0h.
ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>Count no events.</td></tr>
<tr><td>1h</td><td>Count user events (CPL>0).</td></tr>
<tr><td>2h</td><td>Count OS events (CPL=0).</td></tr>
<tr><td>3h</td><td>Count all events, irrespective of the CPL.</td></tr>
</table></td></tr>
<tr><td>15:8</td><td>UnitMask: event qualification. Read-write. Reset: 00h. Each UnitMask bit further specifies or qualifies the event specified by EventSelect. All events selected by UnitMask are simultaneously monitored. Unless otherwise stated, the UnitMask values shown may be combined (logically ORed) to select any desired combination of the sub-events for a given event. In some cases, certain combinations can result in misleading counts, or the UnitMask value is an ordinal rather than a bit mask. These situations are described where applicable, or should be obvious from the event descriptions. For events where no UnitMask table is shown, the UnitMask is Unused.</td></tr>
<tr><td>7:0</td><td>EventSelect[7:0]: event select. Read-write. Reset: 00h. EventSelect[11:0] = {EventSelect[11:8], EventSelect[7:0]}. EventSelect specifies the event or event duration in a processor unit to be counted by the corresponding PERF_CTR[5:0] register. The events are specified in 2.1.17.4 [Core Performance Monitor Counters]. Some events are reserved; when a reserved event is selected, the results are undefined.</td></tr>
</table>

MSRC001_020A [Performance Event Select 5] (Core::X86::Msr::PERF_CTL5)

<table>
<tr><td colspan="2">Read-write. Reset: 0000_0000_0000_0000h.</td></tr>
<tr><td colspan="2">See 2.1.17 [Performance Monitor Counters].</td></tr>
<tr><td colspan="2">_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_020A</td></tr>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>63:42</td><td>Reserved.</td></tr>
<tr><td>41:40</td><td>HostGuestOnly: count only host/guest events. Read-write. Reset: 0h.
ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>Count all events, irrespective of guest/host.</td></tr>
<tr><td>1h</td><td>Count guest events if [SVME] == 1.</td></tr>
<tr><td>2h</td><td>Count host events if [SVME] == 1.</td></tr>
<tr><td>3h</td><td>Count all guest and host events if [SVME] == 1.</td></tr>
</table></td></tr>
<tr><td>39:36</td><td>Reserved.</td></tr>
<tr><td>35:32</td><td>EventSelect[11:8]. Read-write. Reset: 0h. Performance event select[11:8].</td></tr>
<tr><td>31:24</td><td>CntMask: counter mask. Read-write. Reset: 00h. Controls the number of events counted per clock cycle.
ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>00h</td><td>The corresponding PERF_CTR[5:0] register increments by the number of events occurring in a clock cycle. See 2.1.17.3 [Large Increment per Cycle Events] for events that can increment greater than 15 per cycle.</td></tr>
<tr><td>7Fh-01h</td><td>When Inv == 0, the corresponding PERF_CTR[5:0] register increments by 1, if the number of events occurring in a clock cycle is greater than or equal to the CntMask value. When Inv == 1, the</td></tr>
</table></td></tr>
</table>

<table>
<tr><td></td><td><table>
<tr><td></td><td>corresponding PERF_CTR[5:0] register increments by 1, if the number of events occurring in a clock cycle is less than CntMask value.</td></tr>
<tr><td>FFh-80h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>23</td><td>Inv: invert counter mask. Read-write. Reset: 0. See CntMask.</td></tr>
<tr><td>22</td><td>En: enable performance counter. Read-write. Reset: 0. 1=Performance event counter is enabled.</td></tr>
<tr><td>21</td><td>Reserved.</td></tr>
<tr><td>20</td><td>Int: enable APIC interrupt. Read-write. Reset: 0. 1=APIC performance counter LVT interrupt is enabled to generate an interrupt via Core::X86::Apic::PerformanceCounterLvtEntry when the performance counter overflows.</td></tr>
<tr><td>19</td><td>Reserved.</td></tr>
<tr><td>18</td><td>Edge: edge detect. Read-write. Reset: 0. 0=Level detect. 1=Zero-to-one Edge detect. Read-write. The edge count mode increments the counter when a transition happens on the monitored event. If the event selected is changed without disabling the counter, an extra edge is falsely detected when the first event is a static 0 and the second event is a static one. To avoid this false edge detection, disable the counter when changing the event and then enable the counter with a second MSR write.</td></tr>
<tr><td>17:16</td><td>OsUserMode: OS and user mode. Read-write. Reset: 0h.</td></tr>
<tr><td></td><td>ValidValues:<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>Count no events.</td></tr>
<tr><td>1h</td><td>Count user events (CPL>0).</td></tr>
<tr><td>2h</td><td>Count OS events (CPL=0).</td></tr>
<tr><td>3h</td><td>Count all events, irrespective of the CPL.</td></tr>
</table></td></tr>
<tr><td>15:8</td><td>UnitMask: event qualification. Read-write. Reset: 00h. Each UnitMask bit further specifies or qualifies the event specified by EventSelect. All events selected by UnitMask are simultaneously monitored. Unless otherwise stated, the UnitMask values shown may be combined (logically ORed) to select any desired combination of the sub-events for a given event. In some cases, certain combinations can result in misleading counts, or the UnitMask value is an ordinal rather than a bit mask. These situations are described where applicable, or should be obvious from the event descriptions. For events where no UnitMask table is shown, the UnitMask is Unused.</td></tr>
<tr><td>7:0</td><td>EventSelect[7:0]: event select. Read-write. Reset: 00h. EventSelect[11:0] = {EventSelect[11:8], EventSelect[7:0]}. EventSelect specifies the event or event duration in a processor unit to be counted by the corresponding PERF_CTR[5:0] register. The events are specified in 2.1.17.4 [Core Performance Monitor Counters]. Some events are reserved; when a reserved event is selected, the results are undefined.</td></tr>
</table>

MSRC001_023[0...A] [L3 Performance Event Select [5:0]] (Core::X86::Msr::ChL3PmcCfg)

<table>
<tr><td colspan="2">Read-write. Reset: 0000_0000_0000_0000h.</td></tr>
<tr><td colspan="2">See 2.1.17.5 [L3 Cache Performance Monitor Counters]</td></tr>
<tr><td colspan="2">_ccd[7:0]_lthree0_n0; MSRC001_0230</td></tr>
<tr><td colspan="2">_ccd[7:0]_lthree0_n1; MSRC001_0232</td></tr>
<tr><td colspan="2">_ccd[7:0]_lthree0_n2; MSRC001_0234</td></tr>
<tr><td colspan="2">_ccd[7:0]_lthree0_n3; MSRC001_0236</td></tr>
<tr><td colspan="2">_ccd[7:0]_lthree0_n4; MSRC001_0238</td></tr>
<tr><td colspan="2">_ccd[7:0]_lthree0_n5; MSRC001_023A</td></tr>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>63:60</td><td>Reserved.</td></tr>
<tr><td>59:56</td><td>ThreadMask. Read-write. Reset: 0h. Controls which of the 2 threads in the selected core are being counted. In non-SMT mode, thread 0 must be selected. One or more threads must be selected unless otherwise specified by the specific L3PMC event.</td></tr>
<tr><td></td><td>ValidValues:<table>
<tr><th>Bit</th><th>Description</th></tr>
<tr><td>[0]</td><td>Thread 0.</td></tr>
<tr><td>[1]</td><td>Thread 1.</td></tr>
</table></td></tr>
</table>

<table>
<tr><td></td><td><table><tr><td>[3:2]</td><td>Reserved.</td></tr></table></td></tr>
<tr><td>55:51</td><td>Reserved.</td></tr>
<tr><td>50:48</td><td>SliceId. Read-write. Reset: 0h. Controls the L3 slice for which events are counted. Unless otherwise noted by the specific L3PMC event, use Core::X86::Msr::ChL3PmcCfg[SliceId] to select an individual slice or Core::X86::Msr::ChL3PmcCfg[EnAllSlices] to select all slices.
ValidValues:<table><tr><th>Value</th><th>Description</th></tr><tr><td>7h-0h</td><td><Value> Slice.</td></tr></table></td></tr>
<tr><td>47</td><td>EnAllCores. Read-write. Reset: 0. 1=Enable counting L3 events for all cores simultaneously.</td></tr>
<tr><td>46</td><td>EnAllSlices. Read-write. Reset: 0. 1=Enable counting L3 events for all 8 L3 slices simultaneously.</td></tr>
<tr><td>45</td><td>Reserved.</td></tr>
<tr><td>44:42</td><td>CoreId. Read-write. Reset: 0h. Controls core for which events are to be counted. See Core::X86::Msr::ChL3PmcCfg[EnAllCores] to count all cores simultaneously.
ValidValues:<table><tr><th>Value</th><th>Description</th></tr><tr><td>7h-0h</td><td><Value> CoreId.</td></tr></table></td></tr>
<tr><td>41:23</td><td>Reserved.</td></tr>
<tr><td>22</td><td>Enable: Enable L3 performance counter. Read-write. Reset: 0. 1=Enable.</td></tr>
<tr><td>21:16</td><td>Reserved.</td></tr>
<tr><td>15:8</td><td>UnitMask: event qualification. Read-write. Reset: 00h. Each UnitMask bit further specifies or qualifies the event specified by EventSelect. All events selected by UnitMask are simultaneously monitored. Unless otherwise stated, the UnitMask values shown may be combined (logically ORed) to select any desired combination of the sub-events for a given event. In some cases, certain combinations can result in misleading counts, or the UnitMask value is an ordinal rather than a bit mask. These situations are described where applicable, or should be obvious from the event descriptions. For events where no UnitMask table is shown, the UnitMask is Unused. When selecting an event for which not all UnitMask bits are defined, the undefined UnitMask bits should be set to zero.</td></tr>
<tr><td>7:0</td><td>EventSel. Read-write. Reset: 00h. L3 Event select.</td></tr>
</table>

MSRC001_023[1...B] [L3 Performance Event Counter [5:0]] (Core::X86::Msr::ChL3Pmc)

Reset: 0000_0000_0000_0000h.

Also can be read via x86 instructions RDPMC ECX=[0F:0A].

_ccd[7:0]_lthree0_n0; MSRC001_0231
_ccd[7:0]_lthree0_n1; MSRC001_0233
_ccd[7:0]_lthree0_n2; MSRC001_0235
_ccd[7:0]_lthree0_n3; MSRC001_0237
_ccd[7:0]_lthree0_n4; MSRC001_0239
_ccd[7:0]_lthree0_n5; MSRC001_023B

Bits	Description
63:49	Reserved.
48	**Overflow**. Read-write. Reset: 0. Count overflow bit.
47:32	**CountHi**. Read-write,Volatile. Reset: 0000h. Bits 47:32 of the count.
31:0	**CountLo**. Read-write,Volatile. Reset: 0000_0000h. Bits 31:0 of the count.

MSRC001_024[0...6] [Data Fabric Performance Event Select [3:0]] (Core::X86::Msr::DF_PERF_CTL)

Read-write. Reset: 0000_0000_0000_0000h.

See 2.1.17 [Performance Monitor Counters].
The DF Performance Monitors are shared by all cores/threads in the node. See 2.1.11 [Register Sharing].

_n0; MSRC001_0240
_n1; MSRC001_0242
_n2; MSRC001_0244
_n3; MSRC001_0246

Bits	Description
63:61	Reserved.
60:59	**EventSelect[13:12]**. Read-write. Reset: 0h. Performance event select [13:12].
58:36	Reserved.
35:32	**EventSelect[11:8]**: **performance event select**. Read-write. Reset: 0h. Performance event select [11:0].
31:23	Reserved.
22	**En**: **enable performance counter**. Read-write. Reset: 0. 1=Performance event counter is enabled.
21:16	Reserved.
15:8	**UnitMask**: **event qualification**. Read-write. Reset: 00h. Each UnitMask bit further specifies or qualifies the event specified by EventSelect. All events selected by UnitMask are simultaneously monitored.
7:0	**EventSelect[7:0]**: **event select**. Read-write. Reset: 00h. This field, along with EventSelect[13:12] and EventSelect[11:8] above, combine to form the 14-bit event select field, EventSelect[13:0]. EventSelect specifies the event or event duration in a processor unit to be counted by the corresponding DF_PERF_CTR[3:0] register. Some events are reserved; when a reserved event is selected, the results are undefined.

Table 23: DF_PERF_CTL EventSelect[13:6] - Infinity Fabric™ Component

EventSelect[13:6]	Component
0x00	DRAM Channel Controller 0
0x01	DRAM Channel Controller 1
0x02	DRAM Channel Controller 2
0x03	DRAM Channel Controller 3
0x04	DRAM Channel Controller 4
0x05	DRAM Channel Controller 5
0x06	DRAM Channel Controller 6
0x07	DRAM Channel Controller 7
0x1f	Remote Link Controller 0
0x20	Remote Link Controller 1
0x21	Remote Link Controller 2
0x22	Remote Link Controller 3

Table 24: DF_PERF_CTL EventSelect[5:0] - Component Performance Event

EventSelect[13:6]	EventSelect[5:0]	UnitMask	Description
0x00 to 0x07	0x07	0x38	DRAM Channel Controller Request Types: Requests with Data (64B)
0x1f to 0x22	0x07	0x2	Remote Link Controller Outbound Packet Typesi: Data (32B)

MSRC001_024[1...7] [Data Fabric Performance Event Counter [3:0]] (Core::X86::Msr::DF_PERF_CTR)

See Core::X86::Msr::DF_PERF_CTL. Also can be Read via x86 instructions RDPMC ECX = [09:06].
The DF Performance Monitors are shared by all cores/threads in the node. See 2.1.11 [Register Sharing].

_n0; MSRC001_0241
_n1; MSRC001_0243
_n2; MSRC001_0245
_n3; MSRC001_0247

Bits	Description
63:48	Reserved.
47:0	**CTR[47:0]**: **performance counter value[47:0]**. Read-write,Volatile. Reset: 0000_0000_0000h. The current value of the event counter.

MSRC001_0299 [RAPL Power Unit] (Core::X86::Msr::RAPL_PWR_UNIT)

Read-only,Volatile. Reset: 0000_0000_000A_1003h.

_ccd[7:0]_lthree0; MSRC001_0299

<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>63:20</td><td>Reserved.</td></tr>
<tr><td>19:16</td><td>TU: Time Units in seconds. Read-only,Volatile. Reset: Ah. Time information (in Seconds) is based on the multiplier, 1/ 2^TU; where TU is an unsigned integer. Default value is 1010b, indicating time unit is in 976 microseconds increment.
ValidValues:<table><tr><th>Value</th><th>Description</th></tr><tr><td>Fh-0h</td><td>1/2^<Value> Seconds</td></tr></table></td></tr>
<tr><td>15:13</td><td>Reserved.</td></tr>
<tr><td>12:8</td><td>ESU: Energy Status Units. Read-only,Volatile. Reset: 10h. Energy information (in Joules) is based on the multiplier, 1/2^ESU; where ESU is an unsigned integer. Default value is 10000b, indicating energy status unit is in 15.3 micro-Joules increment.
ValidValues:<table><tr><th>Value</th><th>Description</th></tr><tr><td>1Fh-00h</td><td>1/2^<Value> Joules</td></tr></table></td></tr>
<tr><td>7:4</td><td>Reserved.</td></tr>
<tr><td>3:0</td><td>PU: Power Units. Read-only,Volatile. Reset: 3h. Power information (in Watts) is based on the multiplier, 1/ 2^PU; where PU is an unsigned integer. Default value is 0011b, indicating power unit is in 1/8 Watts increment.
ValidValues:<table><tr><th>Value</th><th>Description</th></tr><tr><td>Fh-0h</td><td>1/2^<Value> Watts</td></tr></table></td></tr>
</table>

MSRC001_029A [Core Energy Status] (Core::X86::Msr::CORE_ENERGY_STAT)

Read-only,Volatile. Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]; MSRC001_029A

Bits	Description
63:32	Reserved.
31:0	**TotalEnergyConsumed**. Read-only,Volatile. Reset: 0000_0000h.

MSRC001_029B [Package Energy Status] (Core::X86::Msr::PKG_ENERGY_STAT)

Read-only,Volatile. Reset: 0000_0000_0000_0000h.

_ccd[7:0]; MSRC001_029B

Bits	Description
63:32	Reserved.
31:0	**TotalEnergyConsumed**. Read-only,Volatile. Reset: 0000_0000h.

MSRC001_02F0 [Protected Processor Inventory Number Control] (Core::X86::Msr::PPIN_CTL)

Unpredictable.

MSRC001_02F0

Bits	Description
63:2	Reserved.
1	**PPIN_EN**. Unpredictable. Reset: X. 0=Reading Core::X86::Msr::PPIN will cause a #GP. 1=Core::X86::Msr::PPIN is accessible using RDMSR. Once set, attempting to write 1 to Core::X86::Msr::PPIN_CTL[Lockout] will cause a #GP.
0	**Lockout**. Unpredictable. Reset: X. 0=Writes to Core::X86::Msr::PPIN_CTL are permitted if PPIN_EN=0.

	1=Further writes to Core::X86::Msr::PPIN_CTL are ignored.
	Description: Writing 1 to Core::X86::Msr::PPIN_CTL[Lockout] is permitted only if Core::X86::Msr::PPIN_CTL[PPIN_EN]==0. BIOS should provide an opt-in menu to enable the user to turn on Core::X86::Msr::PPIN_CTL[PPIN_EN] for privileged inventory initialization agent to access Core::X86::Msr::PPIN. After reading Core::X86::Msr::PPIN, the privileged inventory initialization agent should write 00b followed by 01b to Core::X86::Msr::PPIN_CTL to disable further access to MSR_PPIN and prevent unauthorized modification to MSR_PPIN_CTL. Once this bit is written with 1, subsequent writes to this register are ignored, and a reset (warm or cold) is required in order to clear it, which gives BIOS the opportunity to set it again at the next boot.

MSRC001_02F1 [Protected Processor Inventory Number] (Core::X86::Msr::PPIN)

MSRC001_02F1

Bits	Description
63:0	**PPIN**. Reset: Fixed,XXXX_XXXX_XXXX_XXXXh. Protected Processor Inventory Number.
	AccessType: ({Core::X86::Msr::PPIN_CTL[PPIN_EN] , Core::X86::Msr::PPIN_CTL[Lockout]} == 2h) ? Read,Error-on-write : Error-on-read,Error-on-write.

MSRC001_03[00...1E] [Sample Branch From] (Core::X86::Msr::SAMP_BR_FROM)

Read-write,Volatile.

These MSRs are paired elements of a SAMP_BR_FROM register N and SAMP_BR_TO register N. The first sample is intended to be in register 0 and the last sample in register N-1.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n0; MSRC001_0300
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n1; MSRC001_0302
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n2; MSRC001_0304
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n3; MSRC001_0306
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n4; MSRC001_0308
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n5; MSRC001_030A
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n6; MSRC001_030C
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n7; MSRC001_030E
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n8; MSRC001_0310
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n9; MSRC001_0312
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n10; MSRC001_0314
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n11; MSRC001_0316
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n12; MSRC001_0318
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n13; MSRC001_031A
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n14; MSRC001_031C
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n15; MSRC001_031E

Bits	Description
63:0	**BranchFrmOffSet**. Read-write,Volatile. Reset: XXXX_XXXX_XXXX_XXXXh. The 64 bit Segment offset (RIP) of the Branch Instruction

MSRC001_03[01...1F] [Sample Branch To] (Core::X86::Msr::SAMP_BR_TO)

These MSRs are paired elements of a SAMP_BR_FROM register N and SAMP_BR_TO register N. The first sample is intended to be in register 0 and the last sample in register N-1.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n0; MSRC001_0301
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n1; MSRC001_0303
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n2; MSRC001_0305
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n3; MSRC001_0307
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n4; MSRC001_0309
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n5; MSRC001_030B
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n6; MSRC001_030D
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n7; MSRC001_030F
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n8; MSRC001_0311
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n9; MSRC001_0313
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n10; MSRC001_0315
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n11; MSRC001_0317
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n12; MSRC001_0319

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n13; MSRC001_031B	
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n14; MSRC001_031D	
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_n15; MSRC001_031F	
Bits	**Description**
63:61	Reserved.
60:0	**BranchToOffSet**. Read-write,Volatile. Reset: XXXX_XXXX_XXXX_XXXXh. The 64 bit Segment offset (RIP) of the Target of the Branch Instruction.

2.1.16.4 MSRs - MSRC001_1xxx

MSRC001_1002 [CPUID Features for CPUID Fn00000007_E[A,B]X] (Core::X86::Msr::CPUID_7_Features)	
Read-write.	
Core::X86::Msr::CPUID_7_Features[63:32] provides control over values read from Core::X86::Cpuid::StructExtFeatIdEax0; Core::X86::Msr::CPUID_7_Features[31:0] provides control over values read from Core::X86::Cpuid::StructExtFeatIdEbx0.	
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_1002	
Bits	**Description**
63:30	Reserved.
29	**SHA**. Read-write. Reset: Core::X86::Cpuid::StructExtFeatIdEbx0[SHA]. See Core::X86::Cpuid::StructExtFeatIdEbx0[SHA].
28:25	Reserved.
24	**CLWB**: **cache line write back**. Read-write. Reset: Core::X86::Cpuid::StructExtFeatIdEbx0[CLWB]. See Core::X86::Cpuid::StructExtFeatIdEbx0[CLWB].
23	**CLFSHOPT**. Read-write. Reset: Core::X86::Cpuid::StructExtFeatIdEbx0[CLFSHOPT]. See Core::X86::Cpuid::StructExtFeatIdEbx0[CLFSHOPT].
22:21	Reserved.
20	**SMAP**. Read-write. Reset: Core::X86::Cpuid::StructExtFeatIdEbx0[SMAP]. See Core::X86::Cpuid::StructExtFeatIdEbx0[SMAP].
19	**ADX**. Read-write. Reset: Core::X86::Cpuid::StructExtFeatIdEbx0[ADX]. See Core::X86::Cpuid::StructExtFeatIdEbx0[ADX].
18	**RDSEED**. Read-write. Reset: Core::X86::Cpuid::StructExtFeatIdEbx0[RDSEED]. See Core::X86::Cpuid::StructExtFeatIdEbx0[RDSEED].
17:16	Reserved.
15	**PQE**. Read-write. Reset: Core::X86::Cpuid::StructExtFeatIdEbx0[PQE]. See Core::X86::Cpuid::StructExtFeatIdEbx0[PQE].
14:13	Reserved.
12	**PQM**. Read-write. Reset: Core::X86::Cpuid::StructExtFeatIdEbx0[PQM]. See Core::X86::Cpuid::StructExtFeatIdEbx0[PQM].
11	Reserved.
10	**INVPCID**. Read-write. Reset: Core::X86::Cpuid::StructExtFeatIdEbx0[INVPCID]. See Core::X86::Cpuid::StructExtFeatIdEbx0[INVPCID].
9	Reserved.
8	**BMI2**. Read-write. Reset: Core::X86::Cpuid::StructExtFeatIdEbx0[BMI2]. See Core::X86::Cpuid::StructExtFeatIdEbx0[BMI2].
7	**SMEP**. Read-write. Reset: Core::X86::Cpuid::StructExtFeatIdEbx0[SMEP]. See Core::X86::Cpuid::StructExtFeatIdEbx0[SMEP].
6	Reserved.
5	**AVX2**. Read-write. Reset: Core::X86::Cpuid::StructExtFeatIdEbx0[AVX2]. See Core::X86::Cpuid::StructExtFeatIdEbx0[AVX2].
4	Reserved.

3	**BMI1**. Read-write. Reset: Core::X86::Cpuid::StructExtFeatIdEbx0[BMI1]. See Core::X86::Cpuid::StructExtFeatIdEbx0[BMI1].
2:1	Reserved.
0	**FSGSBASE**. Read-write. Reset: Core::X86::Cpuid::StructExtFeatIdEbx0[FSGSBASE]. See Core::X86::Cpuid::StructExtFeatIdEbx0[FSGSBASE].

MSRC001_1003 [Thermal and Power Management CPUID Features] (Core::X86::Msr::CPUID_PWR_THERM)

Read-write.

Core::X86::Msr::CPUID_PWR_THERM provides control over values read from Core::X86::Cpuid::ThermalPwrMgmtEcx.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_1003

Bits	Description
63:1	Reserved.
0	**EffFreq**. Read-write. Reset: Core::X86::Cpuid::ThermalPwrMgmtEcx[EffFreq]. See Core::X86::Cpuid::ThermalPwrMgmtEcx[EffFreq].

MSRC001_1004 [CPUID Features for CPUID Fn00000001_E[C,D]X] (Core::X86::Msr::CPUID_Features)

Read-write.

Core::X86::Msr::CPUID_Features[63:32] provides control over values read from Core::X86::Cpuid::FeatureIdEcx; Core::X86::Msr::CPUID_Features[31:0] provides control over values read from Core::X86::Cpuid::FeatureIdEdx.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_1004

Bits	Description
63	Reserved.
62	**RDRAND**. Read-write. Reset: Core::X86::Cpuid::FeatureIdEcx[RDRAND]. See Core::X86::Cpuid::FeatureIdEcx[RDRAND].
61	**F16C**. Read-write. Reset: Core::X86::Cpuid::FeatureIdEcx[F16C]. See Core::X86::Cpuid::FeatureIdEcx[F16C].
60	**AVX**. Read-write. Reset: Core::X86::Cpuid::FeatureIdEcx[AVX]. See Core::X86::Cpuid::FeatureIdEcx[AVX].
59	**OSXSAVE**. Read-write. Reset: Core::X86::Cpuid::FeatureIdEcx[OSXSAVE]. Modifies Core::X86::Cpuid::FeatureIdEcx[OSXSAVE] only if CR4[OSXSAVE].
58	**XSAVE**. Read-write. Reset: Core::X86::Cpuid::FeatureIdEcx[XSAVE]. See Core::X86::Cpuid::FeatureIdEcx[XSAVE].
57	**AES**. Read-write. Reset: Core::X86::Cpuid::FeatureIdEcx[AES]. Modifies Core::X86::Cpuid::FeatureIdEcx[AES] only if the reset value is 1.
56	Reserved.
55	**POPCNT**. Read-write. Reset: Core::X86::Cpuid::FeatureIdEcx[POPCNT]. See Core::X86::Cpuid::FeatureIdEcx[POPCNT].
54	**MOVBE**. Read-write. Reset: Core::X86::Cpuid::FeatureIdEcx[MOVBE]. See Core::X86::Cpuid::FeatureIdEcx[MOVBE].
53	**X2APIC**. Read-write. Reset: Core::X86::Cpuid::FeatureIdEcx[X2APIC]. See Core::X86::Cpuid::FeatureIdEcx[X2APIC].
52	**SSE42**. Read-write. Reset: Core::X86::Cpuid::FeatureIdEcx[SSE42]. See Core::X86::Cpuid::FeatureIdEcx[SSE42].
51	**SSE41**. Read-write. Reset: Core::X86::Cpuid::FeatureIdEcx[SSE41]. See Core::X86::Cpuid::FeatureIdEcx[SSE41].
50	Reserved.
49	**PCID**. Read-write. Reset: Core::X86::Cpuid::FeatureIdEcx[PCID]. See Core::X86::Cpuid::FeatureIdEcx[PCID].
48:46	Reserved.
45	**CMPXCHG16B**. Read-write. Reset: Core::X86::Cpuid::FeatureIdEcx[CMPXCHG16B]. See Core::X86::Cpuid::FeatureIdEcx[CMPXCHG16B].
44	**FMA**. Read-write. Reset: Core::X86::Cpuid::FeatureIdEcx[FMA]. See Core::X86::Cpuid::FeatureIdEcx[FMA].
43:42	Reserved.

41	**SSSE3**. Read-write. Reset: Core::X86::Cpuid::FeatureIdEcx[SSSE3]. See Core::X86::Cpuid::FeatureIdEcx[SSSE3].
40:36	Reserved.
35	**Monitor**. Read-write. Reset: Core::X86::Cpuid::FeatureIdEcx[Monitor]. Modifies Core::X86::Cpuid::FeatureIdEcx[Monitor] only if ~Core::X86::Msr::HWCR[MonMwaitDis].
34	Reserved.
33	**PCLMULQDQ**. Read-write. Reset: Core::X86::Cpuid::FeatureIdEcx[PCLMULQDQ]. Modifies Core::X86::Cpuid::FeatureIdEcx[PCLMULQDQ] only if the reset value is 1.
32	**SSE3**. Read-write. Reset: Core::X86::Cpuid::FeatureIdEcx[SSE3]. See Core::X86::Cpuid::FeatureIdEcx[SSE3].
31:29	Reserved.
28	**HTT**. Read-write. Reset: Core::X86::Cpuid::FeatureIdEdx[HTT]. See Core::X86::Cpuid::FeatureIdEdx[HTT].
27	Reserved.
26	**SSE2**. Read-write. Reset: Core::X86::Cpuid::FeatureIdEdx[SSE2]. See Core::X86::Cpuid::FeatureIdEdx[SSE2].
25	**SSE**. Read-write. Reset: Core::X86::Cpuid::FeatureIdEdx[SSE]. See Core::X86::Cpuid::FeatureIdEdx[SSE].
24	**FXSR**. Read-write. Reset: Core::X86::Cpuid::FeatureIdEdx[FXSR]. See Core::X86::Cpuid::FeatureIdEdx[FXSR].
23	**MMX**: **MMX instructions**. Read-write. Reset: Core::X86::Cpuid::FeatureIdEdx[MMX]. See Core::X86::Cpuid::FeatureIdEdx[MMX].
22:20	Reserved.
19	**CLFSH**. Read-write. Reset: Core::X86::Cpuid::FeatureIdEdx[CLFSH]. See Core::X86::Cpuid::FeatureIdEdx[CLFSH].
18	Reserved.
17	**PSE36**. Read-write. Reset: Core::X86::Cpuid::FeatureIdEdx[PSE36]. See Core::X86::Cpuid::FeatureIdEdx[PSE36].
16	**PAT**. Read-write. Reset: Core::X86::Cpuid::FeatureIdEdx[PAT]. See Core::X86::Cpuid::FeatureIdEdx[PAT].
15	**CMOV**. Read-write. Reset: Core::X86::Cpuid::FeatureIdEdx[CMOV]. See Core::X86::Cpuid::FeatureIdEdx[CMOV].
14	**MCA**. Read-write. Reset: Core::X86::Cpuid::FeatureIdEdx[MCA]. See Core::X86::Cpuid::FeatureIdEdx[MCA].
13	**PGE**. Read-write. Reset: Core::X86::Cpuid::FeatureIdEdx[PGE]. See Core::X86::Cpuid::FeatureIdEdx[PGE].
12	**MTRR**. Read-write. Reset: Core::X86::Cpuid::FeatureIdEdx[MTRR]. See Core::X86::Cpuid::FeatureIdEdx[MTRR].
11	**SysEnterSysExit**. Read-write. Reset: Core::X86::Cpuid::FeatureIdEdx[SysEnterSysExit]. See Core::X86::Cpuid::FeatureIdEdx[SysEnterSysExit].
10	Reserved.
9	**APIC**. Read-write. Reset: Core::X86::Cpuid::FeatureIdEdx[APIC]. Modifies Core::X86::Cpuid::FeatureIdEdx[APIC] only if Core::X86::Msr::APIC_BAR[ApicEn].
8	**CMPXCHG8B**. Read-write. Reset: Core::X86::Cpuid::FeatureIdEdx[CMPXCHG8B]. See Core::X86::Cpuid::FeatureIdEdx[CMPXCHG8B].
7	**MCE**. Read-write. Reset: Core::X86::Cpuid::FeatureIdEdx[MCE]. See Core::X86::Cpuid::FeatureIdEdx[MCE].
6	**PAE**. Read-write. Reset: Core::X86::Cpuid::FeatureIdEdx[PAE]. See Core::X86::Cpuid::FeatureIdEdx[PAE].
5	**MSR**. Read-write. Reset: Core::X86::Cpuid::FeatureIdEdx[MSR]. See Core::X86::Cpuid::FeatureIdEdx[MSR].
4	**TSC**. Read-write. Reset: Core::X86::Cpuid::FeatureIdEdx[TSC]. See Core::X86::Cpuid::FeatureIdEdx[TSC].
3	**PSE**. Read-write. Reset: Core::X86::Cpuid::FeatureIdEdx[PSE]. See Core::X86::Cpuid::FeatureIdEdx[PSE].
2	**DE**. Read-write. Reset: Core::X86::Cpuid::FeatureIdEdx[DE]. See Core::X86::Cpuid::FeatureIdEdx[DE].
1	**VME**. Read-write. Reset: Core::X86::Cpuid::FeatureIdEdx[VME]. See Core::X86::Cpuid::FeatureIdEdx[VME].
0	**FPU**. Read-write. Reset: Core::X86::Cpuid::FeatureIdEdx[FPU]. See Core::X86::Cpuid::FeatureIdEdx[FPU].

MSRC001_1005 [CPUID Features for CPUID Fn80000001_E[C,D]X] (Core::X86::Msr::CPUID_ExtFeatures)

Read-write.

Core::X86::Msr::CPUID_ExtFeatures[63:32] provides control over values read from

Core::X86::Cpuid::FeatureExtIdEcx; Core::X86::Msr::CPUID_ExtFeatures[31:0] provides control over values read from Core::X86::Cpuid::FeatureExtIdEdx.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_1005

Bits	Description
63	Reserved.
62	**AdMskExtn**. Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEcx[AdMskExtn]. See Core::X86::Cpuid::FeatureExtIdEcx[AdMskExtn].
61	**MwaitExtended**. Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEcx[MwaitExtended]. See Core::X86::Cpuid::FeatureExtIdEcx[MwaitExtended].
60	**PerfCtrExtLLC**. Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEcx[PerfCtrExtLLC]. See Core::X86::Cpuid::FeatureExtIdEcx[PerfCtrExtLLC].
59	**PerfTsc**. Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEcx[PerfTsc]. See Core::X86::Cpuid::FeatureExtIdEcx[PerfTsc].
58	**DataBreakpointExtension**. Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEcx[DataBreakpointExtension]. See Core::X86::Cpuid::FeatureExtIdEcx[DataBreakpointExtension].
57	Reserved.
56	**PerfCtrExtDF**. Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEcx[PerfCtrExtDF]. See Core::X86::Cpuid::FeatureExtIdEcx[PerfCtrExtDF].
55	**PerfCtrExtCore**. Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEcx[PerfCtrExtCore]. See Core::X86::Cpuid::FeatureExtIdEcx[PerfCtrExtCore].
54	**TopologyExtensions**. Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEcx[TopologyExtensions]. See Core::X86::Cpuid::FeatureExtIdEcx[TopologyExtensions].
53:50	Reserved.
49	**TCE**. Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEcx[TCE]. See Core::X86::Cpuid::FeatureExtIdEcx[TCE].
48	**FMA4**. Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEcx[FMA4]. See Core::X86::Cpuid::FeatureExtIdEcx[FMA4].
47	**LWP**. Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEcx[LWP]. See Core::X86::Cpuid::FeatureExtIdEcx[LWP].
46	Reserved.
45	**WDT**. Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEcx[WDT]. See Core::X86::Cpuid::FeatureExtIdEcx[WDT].
44	**SKINIT**. Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEcx[SKINIT]. See Core::X86::Cpuid::FeatureExtIdEcx[SKINIT].
43	**XOP**. Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEcx[XOP]. See Core::X86::Cpuid::FeatureExtIdEcx[XOP].
42	**IBS**. Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEcx[IBS]. See Core::X86::Cpuid::FeatureExtIdEcx[IBS].
41	**OSVW**. Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEcx[OSVW]. See Core::X86::Cpuid::FeatureExtIdEcx[OSVW].
40	**ThreeDNowPrefetch**. Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEcx[ThreeDNowPrefetch]. See Core::X86::Cpuid::FeatureExtIdEcx[ThreeDNowPrefetch].
39	**MisAlignSse**. Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEcx[MisAlignSse]. See Core::X86::Cpuid::FeatureExtIdEcx[MisAlignSse].
38	**SSE4A**. Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEcx[SSE4A]. See Core::X86::Cpuid::FeatureExtIdEcx[SSE4A].
37	**ABM**. Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEcx[ABM]. See Core::X86::Cpuid::FeatureExtIdEcx[ABM].
36	**AltMovCr8**. Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEcx[AltMovCr8]. See Core::X86::Cpuid::FeatureExtIdEcx[AltMovCr8].
35	**ExtApicSpace**. Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEcx[ExtApicSpace]. See

	Core::X86::Cpuid::FeatureExtIdEcx[ExtApicSpace].
34	**SVM**. Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEcx[SVM]. See Core::X86::Cpuid::FeatureExtIdEcx[SVM].
33	**CmpLegacy**. Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEcx[CmpLegacy]. See Core::X86::Cpuid::FeatureExtIdEcx[CmpLegacy].
32	**LahfSahf**. Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEcx[LahfSahf]. See Core::X86::Cpuid::FeatureExtIdEcx[LahfSahf].
31	**ThreeDNow**: **3DNow! instructions**. Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEdx[ThreeDNow]. See Core::X86::Cpuid::FeatureExtIdEdx[ThreeDNow].
30	**ThreeDNowExt**. Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEdx[ThreeDNowExt]. See Core::X86::Cpuid::FeatureExtIdEdx[ThreeDNowExt].
29	**LM**. Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEdx[LM]. See Core::X86::Cpuid::FeatureExtIdEdx[LM].
28	Reserved.
27	**RDTSCP**. Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEdx[RDTSCP]. Core::X86::Cpuid::FeatureExtIdEdx[RDTSCP].
26	**Page1GB**. Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEdx[Page1GB]. Core::X86::Cpuid::FeatureExtIdEdx[Page1GB].
25	**FFXSR**. Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEdx[FFXSR]. See Core::X86::Cpuid::FeatureExtIdEdx[FFXSR].
24	**FXSR**. Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEdx[FXSR]. See Core::X86::Cpuid::FeatureExtIdEdx[FXSR].
23	**MMX**: **MMX instructions**. Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEdx[MMX]. See Core::X86::Cpuid::FeatureExtIdEdx[MMX].
22	**MmxExt**. Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEdx[MmxExt]. See Core::X86::Cpuid::FeatureExtIdEdx[MmxExt].
21	Reserved.
20	**NX**. Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEdx[NX]. See Core::X86::Cpuid::FeatureExtIdEdx[NX].
19:18	Reserved.
17	**PSE36**. Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEdx[PSE36]. See Core::X86::Cpuid::FeatureExtIdEdx[PSE36].
16	**PAT**. Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEdx[PAT]. See Core::X86::Cpuid::FeatureExtIdEdx[PAT].
15	**CMOV**. Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEdx[CMOV]. See Core::X86::Cpuid::FeatureExtIdEdx[CMOV].
14	**MCA**. Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEdx[MCA]. See Core::X86::Cpuid::FeatureExtIdEdx[MCA].
13	**PGE**. Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEdx[PGE]. See Core::X86::Cpuid::FeatureExtIdEdx[PGE].
12	**MTRR**. Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEdx[MTRR]. See Core::X86::Cpuid::FeatureExtIdEdx[MTRR].
11	**SysCallSysRet**. Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEdx[SysCallSysRet]. See Core::X86::Cpuid::FeatureExtIdEdx[SysCallSysRet].
10	Reserved.
9	**APIC**. Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEdx[APIC]. See Core::X86::Cpuid::FeatureExtIdEdx[APIC].
8	**CMPXCHG8B**. Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEdx[CMPXCHG8B]. See Core::X86::Cpuid::FeatureExtIdEdx[CMPXCHG8B].
7	**MCE**. Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEdx[MCE]. See Core::X86::Cpuid::FeatureExtIdEdx[MCE].

6	**PAE**. Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEdx[PAE]. See Core::X86::Cpuid::FeatureExtIdEdx[PAE].
5	**MSR**. Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEdx[MSR]. See Core::X86::Cpuid::FeatureExtIdEdx[MSR].
4	**TSC**. Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEdx[TSC]. See Core::X86::Cpuid::FeatureExtIdEdx[TSC].
3	**PSE**. Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEdx[PSE]. See Core::X86::Cpuid::FeatureExtIdEdx[PSE].
2	**DE**. Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEdx[DE]. See Core::X86::Cpuid::FeatureExtIdEdx[DE].
1	**VME**. Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEdx[VME]. See Core::X86::Cpuid::FeatureExtIdEdx[VME].
0	**FPU**. Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEdx[FPU]. See Core::X86::Cpuid::FeatureExtIdEdx[FPU].

MSRC001_1019 [Address Mask For DR1 Breakpoint] (Core::X86::Msr::DR1_ADDR_MASK)

Read-write. Reset: 0000_0000_0000_0000h.

Support indicated by Core::X86::Cpuid::FeatureExtIdEcx[DataBreakpointExtension].

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_1019

Bits	Description
63:32	Reserved.
31:0	**AddrMask**: **mask for DR linear address data breakpoint DR1**. Read-write. Reset: 0000_0000h. 1=Exclude bit into address compare. 0=Include bit into address compare. See Core::X86::Msr::DR1_ADDR_MASK. AddrMask[11:0] qualifies the DR1 linear address instruction breakpoint, allowing the DR1 instruction breakpoint on a range of addresses in memory.

MSRC001_101A [Address Mask For DR2 Breakpoint] (Core::X86::Msr::DR2_ADDR_MASK)

Read-write. Reset: 0000_0000_0000_0000h.

Support indicated by Core::X86::Cpuid::FeatureExtIdEcx[DataBreakpointExtension].

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_101A

Bits	Description
63:32	Reserved.
31:0	**AddrMask**: **mask for DR linear address data breakpoint DR2**. Read-write. Reset: 0000_0000h. 1=Exclude bit into address compare. 0=Include bit into address compare. See Core::X86::Msr::DR0_ADDR_MASK. AddrMask[11:0] qualifies the DR2 linear address instruction breakpoint, allowing the DR2 instruction breakpoint on a range of addresses in memory.

MSRC001_101B [Address Mask For DR3 Breakpoint] (Core::X86::Msr::DR3_ADDR_MASK)

Read-write. Reset: 0000_0000_0000_0000h.

Support indicated by Core::X86::Cpuid::FeatureExtIdEcx[DataBreakpointExtension].

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_101B

Bits	Description
63:32	Reserved.
31:0	**AddrMask**: **mask for DR linear address data breakpoint DR3**. Read-write. Reset: 0000_0000h. 1=Exclude bit into address compare. 0=Include bit into address compare. See Core::X86::Msr::DR0_ADDR_MASK. AddrMask[11:0] qualifies the DR3 linear address instruction breakpoint, allowing the DR3 instruction breakpoint on a range of addresses in memory.

MSRC001_1027 [Address Mask For DR0 Breakpoints] (Core::X86::Msr::DR0_ADDR_MASK)

Read-write. Reset: 0000_0000_0000_0000h.

Support for DR0[31:12] is indicated by Core::X86::Cpuid::FeatureExtIdEcx[DataBreakpointExtension]. See Core::X86::Msr::DR1_ADDR_MASK.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_1027

Bits	Description
63:32	Reserved.
31:0	**DR0**: **mask for DR0 linear address data breakpoint**. Read-write. Reset: 0000_0000h. 1=Exclude bit into address compare. 0=Include bit into address compare. See Core::X86::Msr::DR1_ADDR_MASK. This field qualifies the DR0 linear address data breakpoint, allowing the DR0 data breakpoint on a range of addresses in memory. AddrMask[11:0] qualifies the DR0 linear address instruction breakpoint, allowing the DR0 instruction breakpoint on a range of addresses in memory. DR0[31:12] is only valid for data breakpoints. The legacy DR0 breakpoint function is provided by DR0[31:0] == 0000_0000h). The mask bits are active high. DR0 is always used, and it can be used in conjunction with any debug function that uses DR0.

MSRC001_1030 [IBS Fetch Control] (Core::X86::Msr::IBS_FETCH_CTL)

Reset: 0000_0000_0000_0000h.

See 2.1.18 [Instruction Based Sampling (IBS)].
The IBS fetch sampling engine is described as follows:

- The periodic fetch counter is an internal 20-bit counter:
  - The periodic fetch counter [19:4] is set to IbsFetchCnt[19:4] and the periodic fetch counter [3:0] is set according to IbsRandEn when IbsFetchEn is changed from 0 to 1.
  - It increments for every fetch cycle that completes when IbsFetchEn == 1 and IbsFetchVal == 0.
    - The periodic fetch counter is undefined when IbsFetchEn == 0 or IbsFetchVal == 1.
  - When IbsFetchCnt[19:4] is read it returns the current value of the periodic fetch counter [19:4].
- When the periodic fetch counter reaches {IbsFetchMaxCnt[19:4],0h} and the selected instruction fetch completes or is aborted:
  - IbsFetchVal is set to 1.
    - Drivers can't assume that IbsFetchCnt[19:4] is 0 when IbsFetchVal == 1.
- The status of the operation is written to the IBS fetch registers (this register, Core::X86::Msr::IBS_FETCH_LINADDR and Core::X86::Msr::IBS_FETCH_PHYSADDR).
- An interrupt is generated as specified by Core::X86::Msr::IBS_CTL. The interrupt service routine associated with this interrupt is responsible for saving the performance information stored in IBS execution registers.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_1030

Bits	Description
63:59	Reserved.
58	**IbsFetchL2Miss**: **L2 cache miss for the sampled fetch**. Read-only,Volatile. Reset: 0. 1=The instruction fetch missed in the L2 Cache. Qualified by (IbsFetchComp == 1).
57	**IbsRandEn**: **random instruction fetch tagging enable**. Read-write. Reset: 0. 0=Bits[3:0] of the fetch counter are set to 0h when IbsFetchEn is set to start the fetch counter. 1=Bits[3:0] of the fetch counter are randomized when IbsFetchEn is set to start the fetch counter.
56	**IbsL2TlbMiss**: **instruction cache L2TLB miss**. Read-only,Volatile. Reset: 0. 1=The instruction fetch missed in the L2 TLB.
55	**IbsL1TlbMiss**: **instruction cache L1TLB miss**. Read-only,Volatile. Reset: 0. 1=The instruction fetch missed in the L1 TLB.
54:53	**IbsL1TlbPgSz**: **instruction cache L1TLB page size**. Read-only,Volatile. Reset: 0h. Indicates the page size of the translation in the L1 TLB. This field is only valid if IbsPhyAddrValid == 1. **ValidValues**: Value | Description 0h | 4 KB 1h | 2 MB 2h | 1 GB 3h | Reserved
52	**IbsPhyAddrValid**: **instruction fetch physical address valid**. Read-only,Volatile. Reset: 0. 1=The physical address in Core::X86::Msr::IBS_FETCH_PHYSADDR and the IbsL1TlbPgSz field are valid for the instruction fetch.

51	**IbsIcMiss**: **instruction cache miss**. Read-only,Volatile. Reset: 0. 1=The instruction fetch missed in the instruction cache.
50	**IbsFetchComp**: **instruction fetch complete**. Read-only,Volatile. Reset: 0. 1=The instruction fetch completed and the data is available for use by the instruction decoder.
49	**IbsFetchVal**: **instruction fetch valid**. Read,Write-0-only,Volatile. Reset: 0. 1=New instruction fetch data available. When this bit is set, the fetch counter stops counting and an interrupt is generated as specified by Core::X86::Msr::IBS_CTL. This bit must be cleared for the fetch counter to start counting. When clearing this bit, software can write 0000h to IbsFetchCnt[19:4] to start the fetch counter at IbsFetchMaxCnt[19:4].
48	**IbsFetchEn**: **instruction fetch enable**. Read-write. Reset: 0. 1=Instruction fetch sampling is enabled.
47:32	**IbsFetchLat**: **instruction fetch latency**. Read-only,Volatile. Reset: 0000h. Indicates the number of clock cycles from when the instruction fetch was initiated to when the data was delivered to the core. If the instruction fetch is abandoned before the fetch completes, this field returns the number of clock cycles from when the instruction fetch was initiated to when the fetch was abandoned.
31:16	**IbsFetchCnt[19:4]**. Read-write,Volatile. Reset: 0000h. Provides read/write access to bits[19:4] of the periodic fetch counter. Programming this field to a value greater than or equal to IbsFetchMaxCnt[19:4] results in undefined behavior.
15:0	**IbsFetchMaxCnt[19:4]**. Read-write. Reset: 0000h. Specifies bits[19:4] of the maximum count value of the periodic fetch counter. Programming this field to 0000h and setting IbsFetchEn results in undefined behavior. Bits[3:0] of the maximum count are always 0000b.

MSRC001_1031 [IBS Fetch Linear Address] (Core::X86::Msr::IBS_FETCH_LINADDR)

Read-write,Volatile. Reset: 0000_0000_0000_0000h.

Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_1031

Bits	Description
63:0	**IbsFetchLinAd**: **instruction fetch linear address**. Read-write,Volatile. Reset: 0000_0000_0000_0000h. Provides the linear address in canonical form for the tagged instruction fetch.

MSRC001_1032 [IBS Fetch Physical Address] (Core::X86::Msr::IBS_FETCH_PHYSADDR)

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_1032

Bits	Description
63:48	Reserved.
47:0	**IbsFetchPhysAd**: **instruction fetch physical address**. Read-write,Volatile. Reset: 0000_0000_0000h. Provides the physical address for the tagged instruction fetch. The lower 12 bits are not modified by address translation, so they are always the same as the linear address. This field contains valid data only if Core::X86::Msr::IBS_FETCH_CTL[IbsPhyAddrValid] is asserted.

MSRC001_1033 [IBS Execution Control] (Core::X86::Msr::IBS_OP_CTL)

Reset: 0000_0000_0000_0000h.

See 2.1.18 [Instruction Based Sampling (IBS)].
The IBS execution sampling engine is described as follows for IbsOpCntCtl == 1. If IbsOpCntCtl == 1n then references to "periodic op counter" mean "periodic cycle counter".

- The periodic op counter is an internal 27-bit counter:
  - It is set to IbsOpCurCnt[26:0] when IbsOpEn is changed from 0 to 1.
  - It increments every dispatched macro-op when IbsOpEn == 1 and IbsOpVal == 0.
    - The periodic op counter is undefined when IbsOpEn == 0 or IbsOpVal == 1.
  - When IbsOpCurCnt[26:0] is read then it returns the current value of the periodic op counter [26:0].
- When the periodic op counter reaches IbsOpMaxCnt:
  - The next dispatched op is tagged if IbsOpCntCtl == 1. A valid op in the next dispatched line is tagged if IbsOpCntCtl == 0. See IbsOpCntCtl.
  - The periodic op counter [26:7]=0; [6:0] is randomized by hardware.
- The periodic op counter is not modified when a tagged op is flushed.

- When a tagged op is retired:
  - IbsOpVal is set to 1.
    - Drivers can't assume that IbsOpCurCnt is 0 when IbsOpVal == 1.
- The status of the operation is written to the IBS execution registers (this register, Core::X86::Msr::IBS_OP_RIP, Core::X86::Msr::IBS_OP_DATA, Core::X86::Msr::IBS_OP_DATA2, Core::X86::Msr::IBS_OP_DATA3, Core::X86::Msr::IBS_DC_LINADDR and Core::X86::Msr::IBS_DC_PHYSADDR).
- An interrupt is generated as specified by Core::X86::Msr::IBS_CTL. The interrupt service routine associated with this interrupt is responsible for saving the performance information stored in IBS execution registers.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_1033

Bits	Description
63:59	Reserved.
58:32	**IbsOpCurCnt[26:0]**: **periodic op counter current count**. Read-write,Volatile. Reset: 000_0000h. Returns the current value of the periodic op counter.
31:27	Reserved.
26:20	**IbsOpMaxCnt[26:20]**: **periodic op counter maximum count**. Read-write. Reset: 00h. See IbsOpMaxCnt[19:4].
19	**IbsOpCntCtl**: **periodic op counter count control**. Read-write. Reset: 0. 0=Count clock cycles; a 1-of-4 round-robin counter selects an op in the next dispatch line; if the op pointed to by the round-robin counter is invalid, then the next younger valid op is selected. 1=Count dispatched ops; when a roll-over occurs, the counter is preloaded with a pseudorandom 7 bit value between 1 and 127.
18	**IbsOpVal**: **op sample valid**. Read-write,Volatile. Reset: 0. 1=New instruction execution data available; the periodic op counter is disabled from counting. An interrupt may be generated when this bit is set as specified by Core::X86::Msr::IBS_CTL[LvtOffset].
17	**IbsOpEn**: **op sampling enable**. Read-write. Reset: 0. 1=Instruction execution sampling enabled.
16	Reserved.
15:0	**IbsOpMaxCnt[19:4]**: **periodic op counter maximum count**. Read-write. Reset: 0000h. IbsOpMaxCnt[26:0] = {IbsOpMaxCnt[26:20], IbsOpMaxCnt[19:4], 0000b}. Specifies maximum count value of the periodic op counter. Bits [3:0] of the maximum count are always 0000b.
	ValidValues:

Value	Description
0008h-0000h	Reserved.
FFFFh-0009h	<Value> *16 Ops.

MSRC001_1034 [IBS Op RIP] (Core::X86::Msr::IBS_OP_RIP)

Read-write,Volatile. Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_1034

Bits	Description
63:0	**IbsOpRip**. Read-write,Volatile. Reset: 0000_0000_0000_0000h. 64 bit Segment offset (RIP) of the instruction that contains the tagged op.

MSRC001_1035 [IBS Op Data] (Core::X86::Msr::IBS_OP_DATA)

Read-write,Volatile. Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_1035

Bits	Description
63:41	Reserved.
40	**IbsOpMicrocode**. Read-write,Volatile. Reset: 0. 1=Tagged operation from microcode.
39	**IbsOpBrnFuse**: **fused branch op**. Read-write,Volatile. Reset: 0. 1=Tagged operation was a fused branch op. Support indicated by Core::X86::Cpuid::IbsIdEax[OpBrnFuse].
38	**IbsRipInvalid**: **RIP is invalid**. Read-write,Volatile. Reset: 0. 1=Tagged operation RIP is invalid. Support indicated by Core::X86::Cpuid::IbsIdEax[RipInvalidChk].
37	**IbsOpBrnRet**: **branch op retired**. Read-write,Volatile. Reset: 0. 1=Tagged operation was a branch op that

	retired.
36	**IbsOpBrnMisp**: **mispredicted branch op**. Read-write,Volatile. Reset: 0. 1=Tagged operation was a branch op that was mispredicted. Qualified by IbsOpBrnRet == 1.
35	**IbsOpBrnTaken**: **taken branch op**. Read-write,Volatile. Reset: 0. 1=Tagged operation was a branch op that was taken. Qualified by IbsOpBrnRet == 1.
34	**IbsOpReturn**: **return op**. Read-write,Volatile. Reset: 0. 1=Tagged operation was return op. Qualified by (IbsOpBrnRet == 1).
33:32	Reserved.
31:16	**IbsTagToRetCtr**: **op tag to retire count**. Read-write,Volatile. Reset: 0000h. This field returns the number of cycles from when the op was tagged to when the op was retired. This field is equal to IbsCompToRetCtr when the tagged op is a NOP.
15:0	**IbsCompToRetCtr**: **op completion to retire count**. Read-write,Volatile. Reset: 0000h. This field returns the number of cycles from when the op was completed to when the op was retired.

MSRC001_1036 [IBS Op Data 2] (Core::X86::Msr::IBS_OP_DATA2)

Reset: 0000_0000_0000_0000h.

Data is only valid for load operations that miss both the L1 data cache and the L2 cache. If a load operation crosses a cache line boundary, the data returned in this register is the data for the access to the lower cache line.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_1036

Bits	Description
63:5	Reserved.
4	**RmtNode**: **IBS request destination node**. Read-write,Volatile. Reset: 0. 0=The request is serviced by the NB in the same node as the core. 1=The request is serviced by the NB in a different node than the core. Valid when NbIbsReqSrc is non-zero.
3	Reserved.
2:0	**DataSrc**: **northbridge IBS request data source**. Read-write. Reset: 0h.
	ValidValues:

Value	Description
0h	No valid status.
1h	Reserved.
2h	Cache: data returned from shared L3, other L2 on same CCX or other core's cache trough same node.
3h	DRAM: data returned from DRAM.
4h	Cache: other core's cache through remote node.
6h-5h	Reserved.
7h	Other; data returned from MMIO/Config/PCI/APIC.

MSRC001_1037 [IBS Op Data 3] (Core::X86::Msr::IBS_OP_DATA3)

Read-write,Volatile. Reset: 0000_0000_0000_0000h.

If a load or store operation crosses a 256-bit boundary, the data returned in this register is the data for the access to the data below the 256-bit boundary.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_1037

Bits	Description
63:48	**IbsTlbRefillLat**: **L1 DTLB refill latency**. Read-write,Volatile. Reset: 0000h. The number of cycles from when a L1 DTLB refill is triggered by a tagged op to when the L1 DTLB fill has been completed.
47:32	**IbsDcMissLat**: **data cache miss latency**. Read-write,Volatile. Reset: 0000h. Indicates the number of clock cycles from when a miss is detected in the data cache to when the data was delivered to the core. The value returned by this counter is not valid for data cache writes or prefetch instructions.
31:26	**IbsOpDcMissOpenMemReqs**: **outstanding memory requests on DC fill**. Read-write,Volatile. Reset: 00h. The number of allocated, valid DC MABs when the MAB corresponding to a tagged DC miss op is deallocated. Includes the MAB allocated by the sampled op. 00000b=No information provided.
25:22	**IbsOpMemWidth**: **load/store size in bytes**. Read-write,Volatile. Reset: 0h. Report the number of bytes the load

	or store is attempting to access. **ValidValues**: **Value** / **Description** 0h: No information provided. 1h: Byte. 2h: Word. 3h: DW. 4h: QW. 5h: OW. Fh-6h: Reserved.
21	**IbsSwPf**: **software prefetch**. Read-write,Volatile. Reset: 0. 1=The op is a software prefetch.
20	**IbsL2Miss**: **L2 cache miss for the sampled operation**. Read-write,Volatile. Reset: 0. 1=The operation missed in the L2, regardless of whether the op initiated the request to the L2.
19	**IbsDcL2TlbHit1G**: **data cache L2TLB hit in 1G page**. Read-write,Volatile. Reset: 0. 1=The physical address for the tagged load or store operation was present in a 1G page table entry in the data cache L2TLB.
18	**IbsDcPhyAddrValid**: **data cache physical address valid**. Read-write,Volatile. Reset: 0. 1=The physical address in Core::X86::Msr::IBS_DC_PHYSADDR is valid for the load or store operation.
17	**IbsDcLinAddrValid**: **data cache linear address valid**. Read-write,Volatile. Reset: 0. 1=The linear address in Core::X86::Msr::IBS_DC_LINADDR is valid for the load or store operation.
16	**DcMissNoMabAlloc**: **DC miss with no MAB allocated**. Read-write,Volatile. Reset: 0. 1=The tagged load or store operation hit on an already allocated MAB.
15	**IbsDcLockedOp**: **locked operation**. Read-write,Volatile. Reset: 0. 1=Tagged load or store operation is a locked operation.
14	**IbsDcUcMemAcc**: **UC memory access**. Read-write,Volatile. Reset: 0. 1=Tagged load or store operation accessed uncacheable memory.
13	**IbsDcWcMemAcc**: **WC memory access**. Read-write,Volatile. Reset: 0. 1=Tagged load or store operation accessed write combining memory.
12:9	Reserved.
8	**IbsDcMisAcc**: **misaligned access**. Read-write,Volatile. Reset: 0. 1=The tagged load or store operation crosses a 256 bit address boundary.
7	**IbsDcMiss**: **data cache miss**. Read-write,Volatile. Reset: 0. 1=The cache line used by the tagged load or store was not present in the data cache.
6	**IbsDcL2tlbHit2M**: **data cache L2TLB hit in 2M page**. Read-write,Volatile. Reset: 0. 1=The physical address for the tagged load or store operation was present in a 2M page table entry in the data cache L2TLB.
5	**IbsDcL1TlbHit1G**: **data cache L1TLB hit in 1G page**. Read-write,Volatile. Reset: 0. 1=The physical address for the tagged load or store operation was present in a 1G page table entry in the data cache L1TLB.
4	**IbsDcL1TlbHit2M**: **data cache L1TLB hit in 2M page**. Read-write,Volatile. Reset: 0. 1=The physical address for the tagged load or store operation was present in a 2M page table entry in the data cache L1TLB.
3	**IbsDcL2TlbMiss**: **data cache L2TLB miss**. Read-write,Volatile. Reset: 0. 1=The physical address for the tagged load or store operation was not present in the data cache L2TLB.
2	**IbsDcL1tlbMiss**: **data cache L1TLB miss**. Read-write,Volatile. Reset: 0. 1=The physical address for the tagged load or store operation was not present in the data cache L1TLB.
1	**IbsStOp**: **store op**. Read-write,Volatile. Reset: 0. 1=Tagged operation is a store operation.
0	**IbsLdOp**: **load op**. Read-write,Volatile. Reset: 0. 1=Tagged operation is a load operation.

MSRC001_1038 [IBS DC Linear Address] (Core::X86::Msr::IBS_DC_LINADDR)

Read-write,Volatile. Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_1038

Bits	Description
63:0	**IbsDcLinAd**. Read-write,Volatile. Reset: 0000_0000_0000_0000h. Provides the linear address in canonical form

	for the tagged load or store operation. This field contains valid data only if Core::X86::Msr::IBS_OP_DATA3[IbsDcLinAddrValid] is asserted.

MSRC001_1039 [IBS DC Physical Address] (Core::X86::Msr::IBS_DC_PHYSADDR)

Read-write,Volatile. Reset: 0000_0000_0000_0000h.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_1039

Bits	Description
63:48	Reserved.
47:0	**IbsDcPhysAd**: **load or store physical address**. Read-write,Volatile. Reset: 0000_0000_0000h. Provides the physical address for the tagged load or store operation. The lower 12 bits are not modified by address translation, so they are always the same as the linear address. This field contains valid data only if Core::X86::Msr::IBS_OP_DATA3[IbsDcPhyAddrValid] is asserted.

MSRC001_103A [IBS Control] (Core::X86::Msr::IBS_CTL)

Read,Error-on-write.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_103A

Bits	Description
63:9	Reserved.
8	**LvtOffsetVal**: **local vector table offset valid**. Read,Error-on-write. Reset: X.
7:4	Reserved.
3:0	**LvtOffset**: **local vector table offset**. Read,Error-on-write. Reset: Xh.

MSRC001_103B [IBS Branch Target Address] (Core::X86::Msr::BP_IBSTGT_RIP)

Read-write,Volatile. Reset: 0000_0000_0000_0000h.

Support for this register indicated by Core::X86::Cpuid::IbsIdEax[BrnTrgt].

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_103B

Bits	Description
63:0	**IbsBrTarget**. Read-write,Volatile. Reset: 0000_0000_0000_0000h. The logical address in canonical form for the branch target. Contains a valid target if non-0. Qualified by Core::X86::Msr::IBS_OP_DATA[IbsOpBrnRet] == 1.

MSRC001_103C [IBS Fetch Control Extended] (Core::X86::Msr::IC_IBS_EXTD_CTL)

Read-only,Volatile. Reset: 0000_0000_0000_0000h.

Support for this register indicated by Core::X86::Cpuid::IbsIdEax[IbsFetchCtlExtd].

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_103C

Bits	Description
63:16	Reserved.
15:0	**IbsItlbRefillLat**: **ITLB Refill Latency for the sampled fetch, if there is a reload**. Read-only,Volatile. Reset: 0000h. The number of cycles when the fetch engine is stalled for an ITLB reload for the sampled fetch. If there is no reload, the latency is 0.

2.1.17 Performance Monitor Counters

2.1.17.1 RDPMC Assignments

There are six core performance event counters per thread, six performance events counters per L3 complex and four Data Fabric performance events counters mapped to the RDPMC instruction as follows:

- The RDPMC[5:0] instruction accesses core events. See 2.1.17.4 [Core Performance Monitor Counters].
- The RDPMC[9:6] instruction accesses data fabric events.
- The RDPMC[F:A] instruction accesses L3 cache events. See 2.1.17.5 [L3 Cache Performance Monitor Counters].

2.1.17.2 Performance Measurement

This section contains AMD's recommended method for collecting microarchitecture performance common to software optimization. This may require combining multiple performance event selections. Table 25 [Guidance for Common Performance Statistics with Complex Event Selects] lists formulas for collecting common performance statistics.

- The term Event is the full value written to Core::X86::Msr::PERF_CTL0..5.
    - Core PMC select bits [63:36,31:16] are at the user's discretion, (i.e. they are not part of the event selection).
- The term L3Event is the full value written to Core::X86::Msr::ChL3PmcCfg.
- The term DFEvent is the full value written to Core::X86::Msr::DF_PERF_CTL.

Some UnitMask fields are not disclosed, but may be used by 2.1.17.2 [Performance Measurement].

Table 25: Guidance for Common Performance Statistics with Complex Event Selects

Description	**Equation**
Execution-Time Branch Misprediction Ratio (Non-Speculative).	Event[0x4300C3] / Event[0x4300C2]
All Data Cache Accesses	Event[0x430729]
All L2 Cache Accesses	Event[0x43F960] + Event[0x43FF70] + Event[0x43FF71] + Event[0x43FF72]
L2 Cache Access from L1 Instruction Cache Miss (including prefetch)	Event[0x431060]
L2 Cache Access from L1 Data Cache Miss (including Prefetch)	Event[0x43E860]
L2 Cache Access from L2 Cache HWPF	Event[0x43FF70] + Event[0x43FF71] + Event[0x43FF72]
All L2 Cache Misses	Event[0x430964] + Event[0x431F71] + Event[0x431f72]
L2 Cache Miss from L1 Instruction Cache Miss	Event[0x430164]
L2 Cache Miss from L1 Data Cache Miss	Event[0x430864]
L2 Cache Miss from L2 Cache HWPF	Event[0x43FF71] + Event[0x43FF72]
All L2 Cache Hits	Event[0x43f664] + Event[0x43ff70]
L2 Cache Hit from L1 Instruction Cache Miss	Event[0x430664]
L2 Cache Hit from L1 Data Cache Miss	Event[0x43F064]
L2 Cache Hit from L2 Cache HWPF	Event[0x43FF70]
L3 Cache Accesses	L3Event[0x0300C0000040FF04]
L3 Miss (includes cacheline state change requests)	L3Event[0x0300C00000400104]
Average L3 Cache Read Miss Latency (in core clocks)	L3Event[0x0300C00000400090] * 16 / L3Event[0x0300C00000401F9a]
Op Cache (64B) Fetch Miss Ratio	Event[0x20043048F] / Event[0x20043078F]
Instruction Cache (32B) Fetch Miss Ratio"	Event[0x10043188E] / Event[0x100431F8E]
L1 Data Cache Fills: From Memory	Event[0x434844]
L1 Data Cache Fills: From Remote Node	Event[0x435044]
L1 Data Cache Fills: From within Same CCX	Event[0x430344]
L1 Data Cache Fills: From External CCX Cache	Event[0x431444]
L1 Data Cache Fills: All	Event[0x43FF44]
L1 ITLB Misses	Event[0x430084] + Event[0x430785]
L2 ITLB Misses & Instruction page walk	Event[0x430785]
L1 DTLB Misses	Event[0x43FF45]
L2 DTLB Misses & Data page walk	Event[0x43F045]
All TLBs Flushed	Event[0x43FF78]

Macro-ops Dispatched	Event[0x4388AB]+Event[0x4384AB]
Mixed SSE/AVX Stalls	Event[0x430E0E]
Macro-ops Retired	Event[0x4300C1]
Approximate: Outbound data bytes for all Remote Links for a node (die)	(DfEvent[0x00000007004002C7] + DfEvent[0x0000000800400207] + DfEvent[0x0000000800400247] + DfEvent[0x0000000800400287]) * 32B
Approximate: Combined DRAM bytes of all channels on a NPS1 node (die)	(DfEvent[0x0000000000403807]+ DfEvent[0x0000000000403847]+ DfEvent[0x0000000000403887]+ DfEvent[0x00000000004038C7]+ DfEvent[0x0000000100403807]+ DfEvent[0x0000000100403847]+ DfEvent[0x0000000100403887]+ DfEvent[0x00000001004038C7]) * 64B

2.1.17.3 Large Increment per Cycle Events

Table 26: PMC_Definitions

Term	Description
MergeEvent	A PMC event that is capable of counter increments greater than 15, thus requiring merging a pair of even/odd performance monitors.

The maximum increment for a regular performance event is 15 (i.e., a 4-bit event). However some event types can have a larger increments every cycle (example: Core::X86::Pmc::Core::FpRetSseAvxOps).

An option is provided for merging a pair of even/odd performance monitors to acquire an accurate count. First the odd numbered Core::X86::Msr::PERF_CTL0..5 is programmed with the event Core::X86::Pmc::Core::Merge (PMCxFFF) with the enable bit (En) turned on and with the remaining bits off. Then the corresponding even numbered Core::X86::Msr::PERF_CTL0..5 is programmed with the desired PMC event. The performance monitor combines the count value to an 8-bit increment event and extends the counter to a 64-bit counter.

Software wanting to preload a value to a merged counter pair writes the high-order 16-bit value to the low-order 16 bits of the odd counter and then writes the low-order 48-bit value to the even counter. Reading the even counter of the merged counter pair returns the full 64-bit value.

If an even performance monitor is programmed with the event Core::X86::Pmc::Core::Merge the read results are undetermined. If an even performance monitor is programmed with a non-merge-able event (i.e., a 4-bit event) while the corresponding odd performance monitor is programmed as Merge, the Read results are undetermined. When discontinuing use of a merged counter pair, clear the Merge event from the odd performance monitor.

PMCxFFF [**Merge**] **(Core::X86::Pmc::Core::Merge)**

See 2.1.17.3 [Large Increment per Cycle Events].

PMCxFFF

Bits	Description
7:0	Reserved.

2.1.17.4 Core Performance Monitor Counters

This section provides the core performance counter events that may be selected through Core::X86::Msr::PERF_CTL0[EventSelect[11:8],EventSelect[7:0],UnitMask]. See Core::X86::Msr::PERF_CTR. See

Core::X86::Msr::PERF_LEGACY_CTL0..3 and Core::X86::Msr::PERF_LEGACY_CTR.

2.1.17.4.1 Floating Point (FP) Events

PMCx003 [Retired SSE/AVX FLOPs] (Core::X86::Pmc::Core::FpRetSseAvxOps)
Read-write. Reset: 00h.
This is a retire-based event. The number of retired SSE/AVX FLOPs. The number of events logged per cycle can vary from 0 to 64. This event requires the use of the MergeEvent since it can count above 15 events per cycle. See 2.1.17.3 [Large Increment per Cycle Events]. It does not provide a useful count without the use of the MergeEvent.
PMCx003

Bits	Description
7:4	Reserved.
3	**MacFLOPs**: **Multiply-Accumulate FLOPs**. Read-write. Reset: 0. Each MAC operation is counted as 2 FLOPS.
2	**DivFLOPs**: **Divide/square root FLOPs**. Read-write. Reset: 0.
1	**MultFLOPs**: **Multiply FLOPs**. Read-write. Reset: 0.
0	**AddSubFLOPs**: **Add/subtract FLOPs**. Read-write. Reset: 0.

PMCx005 [Retired Serializing Ops] (Core::X86::Pmc::Core::FpRetiredSerOps)
Read-write. Reset: 00h.
The number of serializing Ops retired.
PMCx005

Bits	Description
7:4	Reserved.
3	**SseBotRet**. Read-write. Reset: 0. SSE/AVX bottom-executing ops retired.
2	**SseCtrlRet**. Read-write. Reset: 0. SSE/AVX control word mispredict traps.
1	**X87BotRet**. Read-write. Reset: 0. x87 bottom-executing ops retired.
0	**X87CtrlRet**. Read-write. Reset: 0. x87 control word mispredict traps due to mispredictions in RC or PC, or changes in Exception Mask bits.

PMCx00E [FP Dispatch Faults] (Core::X86::Pmc::Core::FpDispFaults)
Read-write. Reset: 00h.
Floating Point Dispatch Faults.
PMCx00E

Bits	Description
7:4	Reserved.
3	**YmmSpillFault**: **YMM Spill fault**. Read-write. Reset: 0.
2	**YmmFillFault**: **YMM Fill fault**. Read-write. Reset: 0.
1	**XmmFillFault**: **XMM Fill fault**. Read-write. Reset: 0.
0	**x87FillFault**: **x87 Fill fault**. Read-write. Reset: 0.

2.1.17.4.2 LS Events

PMCx024 [Bad Status 2] (Core::X86::Pmc::Core::LsBadStatus2)
Read-write. Reset: 00h.
PMCx024

Bits	Description
7:2	Reserved.
1	**StliOther**. Read-write. Reset: 0. Store-to-load conflicts: A load was unable to complete due to a non-forwardable conflict with an older store. Most commonly, a load's address range partially but not completely overlaps with an

	uncompleted older store. Software can avoid this problem by using same-size and same-alignment loads and stores when accessing the same data. Vector/SIMD code is particularly susceptible to this problem; software should construct wide vector stores by manipulating vector elements in registers using shuffle/blend/swap instructions prior to storing to memory, instead of using narrow element-by-element stores.
0	Reserved.

PMCx025 [Retired Lock Instructions] (Core::X86::Pmc::Core::LsLocks)

Read-write. Reset: 00h.

PMCx025

Bits	Description
7:1	Reserved.
0	**BusLock**. Read-write. Reset: 0. Comparable to legacy bus lock.

PMCx026 [Retired CLFLUSH Instructions] (Core::X86::Pmc::Core::LsRetClClush)

The number of retired CLFLUSH instructions. This is a non-speculative event.

PMCx026

Bits	Description
7:0	Reserved.

PMCx027 [Retired CPUID Instructions] (Core::X86::Pmc::Core::LsRetCpuid)

The number of CPUID instructions retired.

PMCx027

Bits	Description
7:0	Reserved.

PMCx029 [LS Dispatch] (Core::X86::Pmc::Core::LsDispatch)

Read-write. Reset: 00h.

Counts the number of operations dispatched to the LS unit. Unit Masks events are ADDed.

PMCx029

Bits	Description
7:3	Reserved.
2	**LdStDispatch**: **Load-op-Store Dispatch**. Read-write. Reset: 0. Dispatch of a single op that performs a load from and store to the same memory address.
1	**StoreDispatch**. Read-write. Reset: 0. Dispatch of a single op that performs a memory store.
0	**LdDispatch**. Read-write. Reset: 0. Dispatch of a single op that performs a memory load.

PMCx02B [SMIs Received] (Core::X86::Pmc::Core::LsSmiRx)

Reset: 00h.

Counts the number of SMIs received.

PMCx02B

Bits	Description
7:0	Reserved.

PMCx02C [Interrupts Taken] (Core::X86::Pmc::Core::LsIntTaken)

Reset: 00h.

Counts the number of interrupts taken.

PMCx02C

Bits	Description
7:0	Reserved.

PMCx035 [Store to Load Forward] (Core::X86::Pmc::Core::LsSTLF)

Number of STLF hits.

PMCx035

Bits	Description
7:0	Reserved.

PMCx037 [Store Commit Cancels 2] (Core::X86::Pmc::Core::LsStCommitCancel2)

Read-write. Reset: 00h.

PMCx037

Bits	Description
7:1	Reserved.
0	**StCommitCancelWcbFull**. Read-write. Reset: 0. A non-cacheable store and the non-cacheable commit buffer is full.

PMCx041 [LS MAB Allocates by Type] (Core::X86::Pmc::Core::LsMabAlloc)

Read-write. Reset: 00h.

Counts when a LS pipe allocates a MAB entry.

PMCx041

Bits	Description
7	Reserved.
6:0	**LsMabAllocation**. Read-write. Reset: 00h. **ValidValues**:

Value	Description
3Eh-00h	Reserved.
3Fh	Load Store Allocations.
40h	Hardware Prefetcher Allocations.
7Eh-41h	Reserved.
7Fh	All Allocations.

PMCx043 [Demand Data Cache Fills by Data Source] (Core::X86::Pmc::Core::LsDmndFillsFromSys)

Read-write. Reset: 00h.

Demand Data Cache Fills by Data Source.

PMCx043

Bits	Description
7	Reserved.
6	**MemIoRemote**. Read-write. Reset: 0. From DRAM or IO connected in different Node.
5	Reserved.
4	**ExtCacheRemote**. Read-write. Reset: 0. From CCX Cache in different Node.
3	**MemIoLocal**. Read-write. Reset: 0. From DRAM or IO connected in same node.
2	**ExtCacheLocal**. Read-write. Reset: 0. From cache of different CCX in same node.
1	**IntCache**. Read-write. Reset: 0. From L3 or different L2 in same CCX.
0	**LclL2**. Read-write. Reset: 0. From Local L2 to the core.

PMCx044 [Any Data Cache Fills by Data Source] (Core::X86::Pmc::Core::LsAnyFillsFromSys)

Read-write. Reset: 00h.

Any Data Cache Fills by Data Source.

PMCx044

Bits	Description
7	Reserved.
6	**MemIoRemote**. Read-write. Reset: 0. From DRAM or IO connected in different Node.
5	Reserved.
4	**ExtCacheRemote**. Read-write. Reset: 0. From CCX Cache in different Node.

3	**MemIoLocal**. Read-write. Reset: 0. From DRAM or IO connected in same node.
2	**ExtCacheLocal**. Read-write. Reset: 0. From cache of different CCX in same node.
1	**IntCache**. Read-write. Reset: 0. From L3 or different L2 in same CCX.
0	**LclL2**. Read-write. Reset: 0. From Local L2 to the core.

PMCx045 [L1 DTLB Misses] (Core::X86::Pmc::Core::LsL1DTlbMiss)

Read-write. Reset: 00h.

PMCx045

Bits	Description
7	**TlbReload1GL2Miss**. Read-write. Reset: 0. DTLB reload to a 1G page that also missed in the L2 TLB.
6	**TlbReload2ML2Miss**. Read-write. Reset: 0. DTLB reload to a 2M page that also missed in the L2 TLB.
5	**TlbReloadCoalescedPageMiss**. Read-write. Reset: 0. DTLB reload to a coalesced page that also missed in the L2 TLB.
4	**TlbReload4KL2Miss**. Read-write. Reset: 0. DTLB reload to a 4K page that missed the L2 TLB.
3	**TlbReload1GL2Hit**. Read-write. Reset: 0. DTLB reload to a 1G page that hit in the L2 TLB.
2	**TlbReload2ML2Hit**. Read-write. Reset: 0. DTLB reload to a 2M page that hit in the L2 TLB.
1	**TlbReloadCoalescedPageHit**. Read-write. Reset: 0. DTLB reload to a coalesced page that hit in the L2 TLB.
0	**TlbReload4KL2Hit**. Read-write. Reset: 0. DTLB reload to a 4K page that hit in the L2 TLB.

PMCx047 [Misaligned loads] (Core::X86::Pmc::Core::LsMisalLoads)

Read-write. Reset: 00h.

PMCx047

Bits	Description
7:2	Reserved.
1	**MA4K**. Read-write. Reset: 0. The number of 4KB misaligned (i.e., page crossing) loads.
0	**MA64**. Read-write. Reset: 0. The number of 64B misaligned (i.e., cacheline crossing) loads.

PMCx04B [Prefetch Instructions Dispatched] (Core::X86::Pmc::Core::LsPrefInstrDisp)

Read-write. Reset: 00h.

Software Prefetch Instructions Dispatched (Speculative).

PMCx04B

Bits	Description
7:3	Reserved.
2	**PREFETCHNTA**. Read-write. Reset: 0. PrefetchNTA instruction. See docAPM3 PREFETCHlevel.
1	**PREFETCHW**. Read-write. Reset: 0. PrefetchW instruction. See docAPM3 PREFETCHW.
0	**PREFETCH**. Read-write. Reset: 0. PrefetchT0, T1 and T2 instructions. See docAPM3 PREFETCHlevel.

PMCx052 [Ineffective Software Prefetches] (Core::X86::Pmc::Core::LsInefSwPref)

Read-write. Reset: 00h.

The number of software prefetches that did not fetch data outside of the processor core.

PMCx052

Bits	Description
7:2	Reserved.
1	**MabMchCnt**. Read-write. Reset: 0. Software PREFETCH instruction saw a match on an already-allocated miss request buffer.
0	**DataPipeSwPfDcHit**. Read-write. Reset: 0. Software PREFETCH instruction saw a DC hit.

PMCx059 [Software Prefetch Data Cache Fills] (Core::X86::Pmc::Core::LsSwPfDcFills)

Read-write. Reset: 00h.

Software Prefetch Data Cache Fills by Data Source.

PMCx059

Bits	Description
7	Reserved.
6	**MemIoRemote**. Read-write. Reset: 0. From DRAM or IO connected in different Node.
5	Reserved.
4	**ExtCacheRemote**. Read-write. Reset: 0. From CCX Cache in different Node.
3	**MemIoLocal**. Read-write. Reset: 0. From DRAM or IO connected in same node.
2	**ExtCacheLocal**. Read-write. Reset: 0. From cache of different CCX in same node.
1	**IntCache**. Read-write. Reset: 0. From L3 or different L2 in same CCX.
0	**LclL2**. Read-write. Reset: 0. From Local L2 to the core.

PMCx05A [Hardware Prefetch Data Cache Fills] (Core::X86::Pmc::Core::LsHwPfDcFills)

Read-write. Reset: 00h.

Hardware Prefetch Data Cache Fills by Data Source.

PMCx05A

Bits	Description
7	Reserved.
6	**MemIoRemote**. Read-write. Reset: 0. From DRAM or IO connected in different Node.
5	Reserved.
4	**ExtCacheRemote**. Read-write. Reset: 0. From CCX Cache in different Node.
3	**MemIoLocal**. Read-write. Reset: 0. From DRAM or IO connected in same node.
2	**ExtCacheLocal**. Read-write. Reset: 0. From cache of different CCX in same node.
1	**IntCache**. Read-write. Reset: 0. From L3 or different L2 in same CCX.
0	**LclL2**. Read-write. Reset: 0. From Local L2 to the core.

PMCx05F [Count of Allocated Mabs] (Core::X86::Pmc::Core::LsAllocMabCount)

This event counts the in-flight L1 data cache misses (allocated Miss Address Buffers) divided by 4 and rounded down each cycle unless used with the MergeEvent functionality. If the MergeEvent is used, it counts the exact number of outstanding L1 data cache misses. See 2.1.17.3 [Large Increment per Cycle Events].

PMCx05F

Bits	Description
7:0	Reserved.

PMCx076 [Cycles not in Halt] (Core::X86::Pmc::Core::LsNotHaltedCyc)

PMCx076

Bits	Description
7:0	Reserved.

PMCx078 [All TLB Flushes] (Core::X86::Pmc::Core::LsTlbFlush)

Read-write. Reset: 00h.

Requires unit mask 0xFF to engage event for counting.

PMCx078

Bits	Description
7:0	**All**. Read-write. Reset: 00h.
	ValidValues:

Value	Description
FEh-00h	Reserved.
FFh	All TLB Flushes.

2.1.17.4.3 IC and BP Events

Note: All instruction cache events are speculative events unless specified otherwise.

PMCx082 [Instruction Cache Refills from L2] (Core::X86::Pmc::Core::IcCacheFillL2)

The number of 64-byte instruction cache line was fulfilled from the L2 cache.

PMCx082

Bits	Description
7:0	Reserved.

PMCx083 [Instruction Cache Refills from System] (Core::X86::Pmc::Core::IcCacheFillSys)

The number of 64-byte instruction cache line fulfilled from system memory or another cache.

PMCx083

Bits	Description
7:0	Reserved.

PMCx084 [L1 ITLB Miss, L2 ITLB Hit] (Core::X86::Pmc::Core::BpL1TlbMissL2TlbHit)

The number of instruction fetches that miss in the L1 ITLB but hit in the L2 ITLB.

PMCx084

Bits	Description
7:0	Reserved.

PMCx085 [ITLB Reload from Page-Table walk] (Core::X86::Pmc::Core::BpL1TlbMissL2TlbMiss)

Read-write. Reset: 00h.

The number of valid fills into the ITLB originating from the LS Page-Table Walker. Tablewalk requests are issued for L1-ITLB and L2-ITLB misses.

PMCx085

Bits	Description
7:4	Reserved.
3	**Coalesced4K**. Read-write. Reset: 0. Walk for >4K Coalesced page.
2	**IF1G**. Read-write. Reset: 0. Walk for 1G page.
1	**IF2M**. Read-write. Reset: 0. Walk for 2M page.
0	**IF4K**. Read-write. Reset: 0. Walk to 4K page.

PMCx08B [L2 Branch Prediction Overrides Existing Prediction (speculative)] (Core::X86::Pmc::Core::BpL2BTBCorrect)

PMCx08B

Bits	Description
7:0	Reserved.

PMCx08E [Dynamic Indirect Predictions] (Core::X86::Pmc::Core::BpDynIndPred)

The number of times a branch used the indirect predictor to make a prediction.

PMCx08E

Bits	Description
7:0	Reserved.

PMCx091 [Decode Redirects] (Core::X86::Pmc::Core::BpDeReDirect)

Reset: 00h.

The number of times the instruction decoder overrides the predicted target.

PMCx091

Bits	Description
7:0	Reserved.

PMCx094 [L1 TLB Hits for Instruction Fetch] (Core::X86::Pmc::Core::BpL1TlbFetchHit)

Read-write. Reset: 00h.

The number of instruction fetches that hit in the L1 ITLB.

PMCx094

Bits	Description
7:3	Reserved.
2	**IF1G**. Read-write. Reset: 0. L1 Instruction TLB hit (1G page size).
1	**IF2M**. Read-write. Reset: 0. L1 Instruction TLB hit (2M page size).
0	**IF4K**. Read-write. Reset: 0. L1 Instruction TLB hit (4K or 16K page size).

PMCx18E [IC Tag Hit/Miss Events] (Core::X86::Pmc::Core::IcTagHitMiss)

Read-write. Reset: 00h.

Counts various IC tag related hit and miss events.

PMCx18E

Bits	Description
7:5	Reserved.
4:0	**IcAccessTypes**. Read-write. Reset: 00h. Instruction Cache accesses.

ValidValues:

Value	Description
06h-00h	Reserved.
07h	Instruction Cache Hit.
17h-08h	Reserved.
18h	Instruction Cache Miss.
1Eh-19h	Reserved.
1Fh	All Instruction Cache Accesses.

PMCx28F [Op Cache Hit/Miss] (Core::X86::Pmc::Core::OpCacheHitMiss)

Read-write. Reset: 00h.

Counts Op Cache micro-tag hit/miss events

PMCx28F

Bits	Description
7:3	Reserved.
2:0	**OpCacheAccesses**. Read-write. Reset: 0h.

ValidValues:

Value	Description
2h-0h	Reserved.
3h	Op Cache Hit.
4h	Op Cache Miss.
6h-5h	Reserved.
7h	All Op Cache accesses.

2.1.17.4.4 DE Events

PMCx0AA [Source of Op Dispatched From Decoder] (Core::X86::Pmc::Core::DeSrcOpDisp)

Read-write. Reset: 00h.

Counts the number of ops dispatched from the decoder classified by op source.
See docRevG erratum #1287.

PMCx0AA

Bits	Description

7:2	Reserved.
1	**OpCache**. Read-write. Reset: 0. Count of ops fetched from Op Cache and dispatched.
0	**x86Decoder**. Read-write. Reset: 0. Count of ops fetched from Instruction Cache and dispatched.

PMCx0AB [Types of Oops Dispatched From Decoder] (Core::X86::Pmc::Core::DeDisCopsFromDecoder)

Read-write. Reset: 00h.

Counts the number of ops dispatched from the decoder classified by op type. The UnitMask value encodes which types of ops are counted.

PMCx0AB

<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>7</td><td>OpCountingMode. Read-write. Reset: 0. 0= count aligns with IBS count; 1= count aligns with retire count (PMCx0C1). .</td></tr>
<tr><td>6:5</td><td>Reserved.</td></tr>
<tr><td>4:0</td><td>DispOpType. Read-write. Reset: 00h.
ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>03h-00h</td><td>Reserved.</td></tr>
<tr><td>04h</td><td>Any FP dispatch</td></tr>
<tr><td>07h-05h</td><td>Reserved.</td></tr>
<tr><td>08h</td><td>Any Integer dispatch</td></tr>
<tr><td>1Fh-09h</td><td>Reserved.</td></tr>
</table></td></tr>
</table>

PMCx0AE [Dispatch Resource Stall Cycles 1] (Core::X86::Pmc::Core::DeDisDispatchTokenStalls1)

Read-write. Reset: 00h.

Cycles where a dispatch group is valid but does not get dispatched due to a Token Stall. Also counts cycles when the thread is not selected to dispatch but would have been stalled due to a Token Stall.

PMCx0AE

Bits	Description
7	**FpFlushRecoveryStall**. Read-write. Reset: 0. FP Flush recovery stall.
6	**FPSchRsrcStall**: **FP scheduler resource stall**. Read-write. Reset: 0. Applies to ops that use the FP scheduler.
5	**FpRegFileRsrcStall**: **floating point register file resource stall**. Read-write. Reset: 0. Applies to all FP ops that have a destination register.
4	**TakenBrnchBufferRsrc**: **taken branch buffer resource stall**. Read-write. Reset: 0.
3	Reserved.
2	**StoreQueueRsrcStall**: **Store Queue resource stall**. Read-write. Reset: 0. Applies to all ops with store semantics.
1	**LoadQueueRsrcStall**: **Load Queue resource stall**. Read-write. Reset: 0. Applies to all ops with load semantics.
0	**IntPhyRegFileRsrcStall**: **Integer Physical Register File resource stall**. Read-write. Reset: 0. Integer Physical Register File, applies to all ops that have an integer destination register.

PMCx0AF [Dynamic Tokens Dispatch Stall Cycles 2] (Core::X86::Pmc::Core::DeDisDispatchTokenStalls2)

Read-write. Reset: 00h.

Cycles where a dispatch group is valid but does not get dispatched due to a token stall.

PMCx0AF

Bits	Description
7:6	Reserved.
5	**RetireTokenStall**. Read-write. Reset: 0. Insufficient Retire Queue tokens available.
4	Reserved.
3	**IntSch3TokenStall**. Read-write. Reset: 0. No tokens for Integer Scheduler Queue 3 available.
2	**IntSch2TokenStall**. Read-write. Reset: 0. No tokens for Integer Scheduler Queue 2 available.
1	**IntSch1TokenStall**. Read-write. Reset: 0. No tokens for Integer Scheduler Queue 1 available.
0	**IntSch0TokenStall**. Read-write. Reset: 0. No tokens for Integer Scheduler Queue 0 available.

2.1.17.4.5 EX (SC) Events

PMCx0C0 [Retired Instructions] (Core::X86::Pmc::Core::ExRetInstr)

The number of instructions retired.

PMCx0C0

Bits	Description
7:0	Reserved.

PMCx0C1 [Retired Ops] (Core::X86::Pmc::Core::ExRetOps)

The number of macro-ops retired.

PMCx0C1

Bits	Description
7:0	Reserved.

PMCx0C2 [Retired Branch Instructions] (Core::X86::Pmc::Core::ExRetBrn)

The number of branch instructions retired. This includes all types of architectural control flow changes, including exceptions and interrupts.

PMCx0C2

Bits	Description
7:0	Reserved.

PMCx0C3 [Retired Branch Instructions Mispredicted] (Core::X86::Pmc::Core::ExRetBrnMisp)

The number of retired branch instructions, that were mispredicted.

PMCx0C3

Bits	Description
7:0	Reserved.

PMCx0C4 [Retired Taken Branch Instructions] (Core::X86::Pmc::Core::ExRetBrnTkn)

The number of taken branches that were retired. This includes all types of architectural control flow changes, including exceptions and interrupts.

PMCx0C4

Bits	Description
7:0	Reserved.

PMCx0C5 [Retired Taken Branch Instructions Mispredicted] (Core::X86::Pmc::Core::ExRetBrnTknMisp)

The number of retired taken branch instructions that were mispredicted.

PMCx0C5

Bits	Description
7:0	Reserved.

PMCx0C6 [Retired Far Control Transfers] (Core::X86::Pmc::Core::ExRetBrnFar)

The number of far control transfers retired including far call/jump/return, IRET, SYSCALL and SYSRET, plus exceptions and interrupts. Far control transfers are not subject to branch prediction.

PMCx0C6

Bits	Description
7:0	Reserved.

PMCx0C8 [Retired Near Returns] (Core::X86::Pmc::Core::ExRetNearRet)

The number of near return instructions (RET or RET Iw) retired.

PMCx0C8

Bits	Description
7:0	Reserved.

PMCx0C9 [Retired Near Returns Mispredicted] (Core::X86::Pmc::Core::ExRetNearRetMispred)

The number of near returns retired that were not correctly predicted by the return address predictor. Each such mispredict incurs the same penalty as a mispredicted conditional branch instruction.

PMCx0C9

Bits	Description
7:0	Reserved.

PMCx0CA [Retired Indirect Branch Instructions Mispredicted] (Core::X86::Pmc::Core::ExRetBrnIndMisp)

The number of indirect branches retired that were not correctly predicted. Each such mispredict incurs the same penalty as a mispredicted conditional branch instruction. Note that only EX mispredicts are counted.

PMCx0CA

Bits	Description
7:0	Reserved.

PMCx0CB [Retired MMX/FP Instructions] (Core::X86::Pmc::Core::ExRetMmxFpInstr)

Read-write. Reset: 00h.

The number of MMX, SSE or x87 instructions retired. The UnitMask allows the selection of the individual classes of instructions as given in the table. Each increment represents one complete instruction. Since this event includes non-numeric instructions it is not suitable for measuring MFLOPs.

PMCx0CB

Bits	Description
7:3	Reserved.
2	**SseInstr**. Read-write. Reset: 0. SSE instructions (SSE, SSE2, SSE3, SSSE3, SSE4A, SSE41, SSE42, AVX).
1	**MmxInstr**. Read-write. Reset: 0. MMX instructions.
0	**X87Instr**: **x87 instructions**. Read-write. Reset: 0.

PMCx0CC [Retired Indirect Branch Instructions] (Core::X86::Pmc::Core::ExRetIndBrchInstr)

The number of indirect branches retired.

PMCx0CC

Bits	Description
7:0	Reserved.

PMCx0D1 [Retired Conditional Branch Instructions] (Core::X86::Pmc::Core::ExRetCond)

PMCx0D1

Bits	Description
7:0	Reserved.

PMCx0D3 [Div Cycles Busy count] (Core::X86::Pmc::Core::ExDivBusy)

PMCx0D3

Bits	Description
7:0	Reserved.

PMCx0D4 [Div Op Count] (Core::X86::Pmc::Core::ExDivCount)

PMCx0D4

Bits	Description
7:0	Reserved.

PMCx1C7 [Retired Mispredicted Branch Instructions due to Direction Mismatch] (Core::X86::Pmc::Core::ExRetMsprdBrnchInstrDirMsmtch)

The number of retired conditional branch instructions that were not correctly predicted because of a branch direction mismatch.

PMCx1C7

Bits	Description

7:0	Reserved.

PMCx1CF [Tagged IBS Ops] (Core::X86::Pmc::Core::ExTaggedIbsOps)

Read-write. Reset: 00h.

Counts Op IBS related events.

PMCx1CF

Bits	Description
7:3	Reserved.
2	**IbsCountRollover**. Read-write. Reset: 0. Number of times an op could not be tagged by IBS because of a previous tagged op that has not retired.
1	**IbsTaggedOpsRet**. Read-write. Reset: 0. Number of Ops tagged by IBS that retired.
0	**IbsTaggedOps**. Read-write. Reset: 0. Number of Ops tagged by IBS.

PMCx1D0 [Retired Fused Instructions] (Core::X86::Pmc::Core::ExRetFusedInstr)

Reset: 00h.

Counts retired fused instructions.

PMCx1D0

Bits	Description
7:0	Reserved.

2.1.17.4.6 L2 Cache Events

PMCx060 [Requests to L2 Group1] (Core::X86::Pmc::Core::L2RequestG1)

Read-write. Reset: 00h.

All L2 Cache Requests (Breakdown 1 - Common).

PMCx060

Bits	Description
7	**RdBlkL**. Read-write. Reset: 0. Data Cache Reads (including hardware and software prefetch).
6	**RdBlkX**. Read-write. Reset: 0. Data Cache Stores.
5	**LsRdBlkC_S**. Read-write. Reset: 0. Data Cache Shared Reads.
4	**CacheableIcRead**. Read-write. Reset: 0. Instruction Cache Reads.
3	**ChangeToX**: **Data Cache State Change Requests**. Read-write. Reset: 0. Request change to writable, check L2 for current state.
2	**PrefetchL2Cmd**. Read-write. Reset: 0.
1	**L2HwPf**: **L2 Prefetcher**. Read-write. Reset: 0. All prefetches accepted by L2 pipeline, hit or miss. Types of PF and L2 hit/miss broken out in a separate perfmon event.
0	Reserved.

PMCx064 [Core to L2 Cacheable Request Access Status] (Core::X86::Pmc::Core::L2CacheReqStat)

Read-write. Reset: 00h.

L2 Cache Request Outcomes (not including L2 Prefetch).

PMCx064

Bits	Description
7	**LsRdBlkCS**: **Data Cache Shared Read Hit in L2**. Read-write. Reset: 0.
6	**LsRdBlkLHitX**: **Data Cache Read Hit in L2**. Read-write. Reset: 0. Modifiable.
5	**LsRdBlkLHitS**: **Data Cache Read Hit Non-Modifiable Line in L2**. Read-write. Reset: 0.
4	**LsRdBlkX**: **Data Cache Store or State Change Hit in L2**. Read-write. Reset: 0.
3	**LsRdBlkC**: **Data Cache Req Miss in L2 (all types)**. Read-write. Reset: 0.
2	**IcFillHitX**: **Instruction Cache Hit Modifiable Line in L2**. Read-write. Reset: 0.
1	**IcFillHitS**: **Instruction Cache Hit Non-Modifiable Line in L2**. Read-write. Reset: 0.

0	**IcFillMiss**: **Instruction Cache Req Miss in L2**. Read-write. Reset: 0.

PMCx070 [**L2 Prefetch Hit in L2**] **(Core::X86::Pmc::Core::L2PfHitL2)**

Reset: 00h.

PMCx070

Bits	Description
7:0	Reserved.

PMCx071 [**L2 Prefetcher Hits in L3**] **(Core::X86::Pmc::Core::L2PfMissL2HitL3)**

Reset: 00h.

Counts all L2 prefetches accepted by the L2 pipeline which miss the L2 cache and hit the L3.

PMCx071

Bits	Description
7:0	Reserved.

PMCx072 [**L2 Prefetcher Misses in L3**] **(Core::X86::Pmc::Core::L2PfMissL2L3)**

Reset: 00h.

Counts all L2 prefetches accepted by the L2 pipeline which miss the L2 and the L3 caches.

PMCx072

Bits	Description
7:0	Reserved.

2.1.17.5 L3 Cache Performance Monitor Counters

The L3 cache is organized as eight "slices" of L3 shared by eight cores.
This section provides the core performance counter events that may be selected through Core::X86::Msr::ChL3PmcCfg.

- Unless otherwise noted, Family 19h L3 Perfmon events utilize Core::X86::Msr::ChL3PmcCfg[SliceId] to select an individual slice or Core::X86::Msr::ChL3PmcCfg[EnAllSlices] to select all slices.
- Family 19h L3 Perfmon events utilize Core::X86::Msr::ChL3PmcCfg[CoreId] to select an individual core or Core::X86::Msr::ChL3PmcCfg[EnAllCores] to select all cores.
- Unless otherwise noted, L3 PMC's require Core::X86::Msr::ChL3PmcCfg[CoreId] to be set or the PMC count will be zero.
- When in non-SMT mode, thread 0 must be selected for events that don't ignore ThreadMask.

2.1.17.5.1 L3 Cache PMC Events

L3PMCx04 [**All L3 Cache Requests**] **(Core::X86::Pmc::L3::L3LookupState)**

Read-write. Reset: 00h.

L3PMCx04

Bits	Description
7:0	**AllL3ReqTyps**: **All L3 Request Types**. Read-write. Reset: 00h.

L3PMCx90 [**L3 Cache Miss Latency**] **(Core::X86::Pmc::L3::XiSysFillLatency)**

Ignores SliceID, EnAllSlices, CoreID, EnAllCores and ThreadMask.

L3PMCx90

Bits	Description
7:0	Reserved.

L3PMCx9A [**L3 Misses by Request Type**] **(Core::X86::Pmc::L3::XiCcxSdpReq1)**

Reset: 00h.

Ignores SliceID, EnAllSlices, CoreID, EnAllCores and ThreadMask.
Requires unit mask 0xFF to engage event for counting.

L3PMCx9A

Bits	Description
7:0	Reserved.

2.1.18 Instruction Based Sampling (IBS)

IBS is a code profiling mechanism that enables the processor to select a random instruction fetch or macro-op after a programmed time interval has expired and record specific performance information about the operation. An interrupt is generated when the operation is complete as specified by Core::X86::Msr::IBS_CTL. An interrupt handler can then read the performance information that was logged for the operation.

The IBS mechanism is split into two parts: instruction fetch performance controlled by Core::X86::Msr::IBS_FETCH_CTL; and instruction execution performance controlled by Core::X86::Msr::IBS_OP_CTL. Instruction fetch sampling provides information about instruction TLB and instruction cache behavior for fetched instructions. Instruction execution sampling provides information about op execution behavior. The data collected for instruction fetch performance is independent from the data collected for instruction execution performance. Support for the IBS feature is indicated by the Core::X86::Cpuid::FeatureExtIdEcx[IBS].

Instruction fetch performance is profiled by recording the following performance information for the tagged instruction fetch:

- If the instruction fetch completed or was aborted. See Core::X86::Msr::IBS_FETCH_CTL.
- The number of clock cycles spent on the instruction fetch. See Core::X86::Msr::IBS_FETCH_CTL.
- If the instruction fetch hit or missed the IC, hit/missed in the L1 and L2 TLBs, and page size. See Core::X86::Msr::IBS_FETCH_CTL.
- The linear address, physical address associated with the fetch. See Core::X86::Msr::IBS_FETCH_LINADDR, Core::X86::Msr::IBS_FETCH_PHYSADDR.

Instruction execution performance is profiled by tagging one macro-op associated with an instruction. Instructions that decode to more than one macro-op return different performance data depending upon which macro-op associated with the instruction is tagged. These macro-ops are associated with the RIP of the next instruction to retire. The following performance information is returned for the tagged op:

- Branch and execution status. See Core::X86::Msr::IBS_OP_DATA.
- Branch target address for branch ops. See Core::X86::Msr::BP_IBSTGT_RIP.
- The logical address associated with the op. See Core::X86::Msr::IBS_OP_RIP.
- The linear and physical address associated with a load or store op. See Core::X86::Msr::IBS_DC_LINADDR, Core::X86::Msr::IBS_DC_PHYSADDR.
- The data cache access status associated with the op: DC hit/miss, DC miss latency, TLB hit/miss, TLB page size. See Core::X86::Msr::IBS_OP_DATA3.
- The number clocks from when the op was tagged until the op retires. See Core::X86::Msr::IBS_OP_DATA.
- The number clocks from when the op completes execution until the op retires. See Core::X86::Msr::IBS_OP_DATA.
- Source information for DRAM and MMIO. See Core::X86::Msr::IBS_OP_DATA2.