Processor Programming Reference (PPR) for AMD Family 19h Model 11h, Revision B2 Processors Volume 1 of 6

Legal Notices

The information contained herein is for informational purposes only, and is subject to change without notice. While every precaution has been taken in the preparation of these materials, it may contain technical inaccuracies, omissions and typographical errors, and AMD is under no obligation to update or otherwise correct this information. Advanced Micro Devices, Inc. makes no representations or warranties with respect to the accuracy or completeness of the contents of these materials, and assumes no liability of any kind, including the implied warranties of noninfringement, merchantability or fitness for particular purposes, with respect to the operation or use of AMD hardware, software or other products described herein. No license, including implied or arising by estoppel, to any intellectual property rights is granted by these materials. Terms and limitations applicable to the purchase or use of AMD’s products are as set forth in a signed agreement between the parties or in AMD's Standard Terms and Conditions of Sale.

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.
Arm is a registered trademark of Arm Limited.
CXL is a trademark of Compute Express Link Consortium, Inc.
EPYC is a trademark of Advanced Micro Devices, Inc.
Infinity Fabric is a trademark of Advanced Micro Devices, Inc.
Linux is a registered trademark of Linus Torvalds.
MIPI I3C is a registered trademark of MIPI Alliance.
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.
SoundWire is a registered trademark of MIPI Alliance, Inc.
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.
docI3C	MIPI I3C® Specification, http://www.mipi.org
docJEDEC	JEDEC Standards. http://www.jedec.org.
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 10h-1Fh Processors, Publication No. 57095.
docSEV	Secure Encrypted Visualization API Specification, Publication No. 55766.
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.

Menu->Preferences:

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

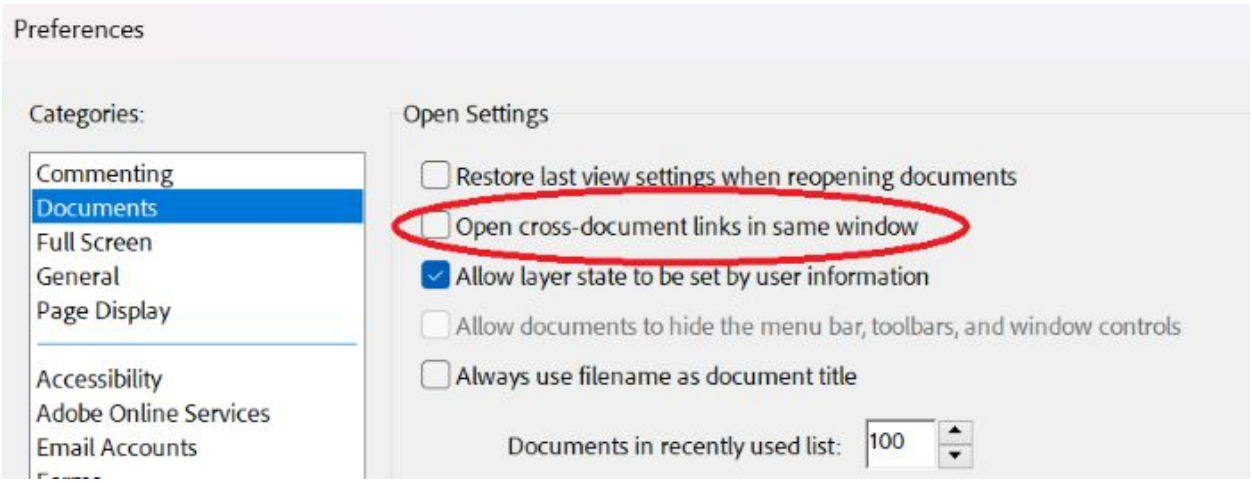


Figure 1: Adobe® Reader Hyperlink Opens New Window Configuration

The following Figure shows how when hyperlinking from one document to another, the original document is left open. The tab that is not grayed-out indicates the foreground window.

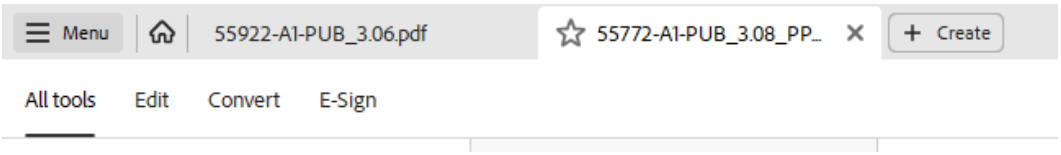


Figure 2: Adobe® Reader Select Between Opened Files

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 search across multiple PDF files can not be performed from within a zip of PDF's.
- Open search by selecting Menu->Search->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.

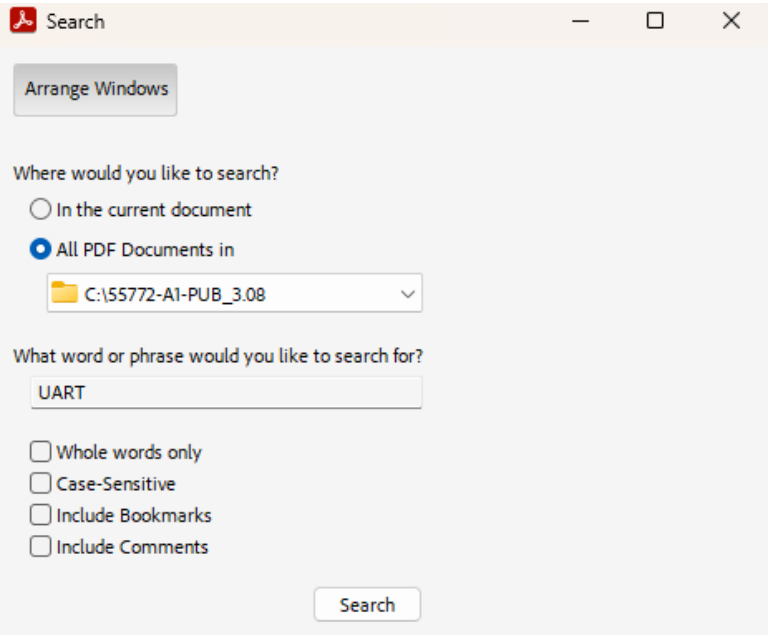


Figure 3: Adobe® Reader Searching a Multiple Volume PPR

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.
    - Menu->View->Page Navigation->Previous View
- 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 Find Current Bookmark

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

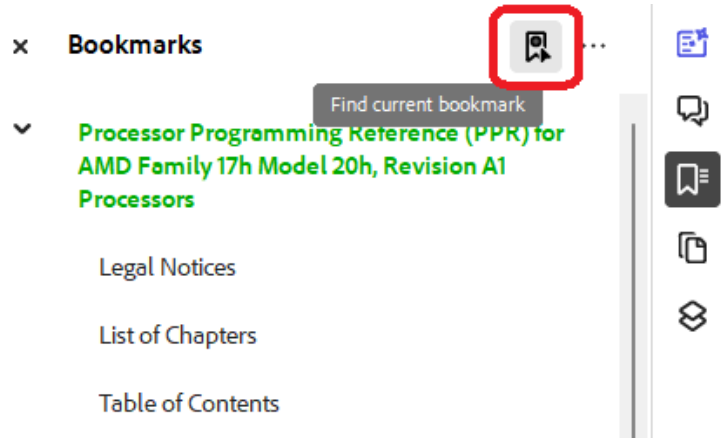


Figure 4: Adobe® Reader Find Current Bookmark Button

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 connotes 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. Bus 0 is implied, X specifies the hexadecimal device number (this may be 1 or 2 digits). Y specifies the function number. ZZZ specifies the hexadecimal byte address (this may be 2 or 3 digits). Example: D18F2x40 specifies the register at bus 0, device 18h, function 2, and address 40h. If the mnemonic starts with B, then the physical mnemonic format is BWWDXFYxZZZ where WW specifies the hexadecimal bus number (1 or 2 hex digits) or "XX" implying that the bus is relocatable. Example; BXXD00F6x40 specifies that the bus is relocatable, B0AD00F2x000 specifies that the bus is 0Ah.
BAR	The BAR, or base address register, physical register mnemonic format is of the form PREFIXxZZZ. PREFIX is an all capital letter name that connotes the BAR to which the offset is added to get the physical address of the operation. ZZZ is the offset.
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 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.

Physical Mnemonic Title Name

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 5: 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 6: 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 7: 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 8: 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 9: 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.

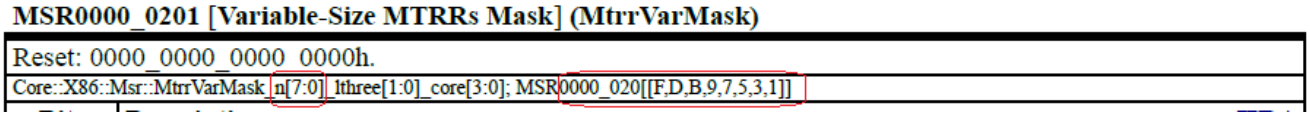


Figure 10: Register Instance Table: MSR Range Example

1.4.4.5.4 BAR Access Method

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

- BAR, which stands for "Base Address Register", is the base address for the IP block and can be a constant, a register field, or an expression that consists of one or more register fields.
- 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].
- The address of the register is NBIFEPFNCFG::BASE_ADDR_1_aliasHOST_instNBIF0_func1[BASE_ADDR] + 00000h.

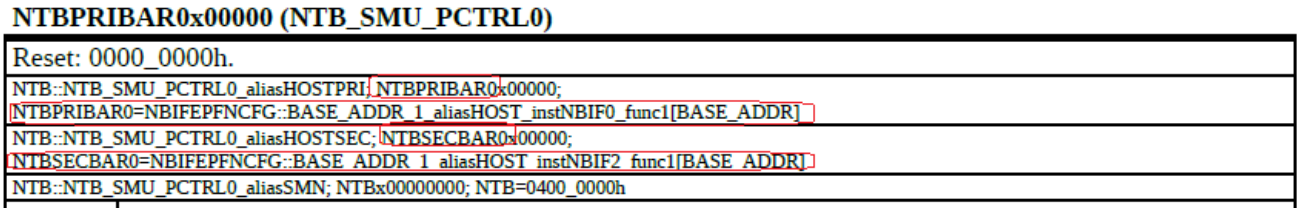


Figure 11: Register Instance Table: BAR as Register Reference

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.

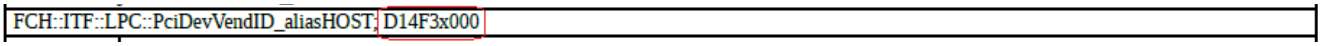


Figure 12: Register Instance Table: Bus Implied to be 00h

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.

- The registers behind a data port is also called an indirect address space.
- The implied access method is to first write the data port select and then to read/write the register at the data port address.
- There are some cases where there are 2 or more data port selects.

Example:

- The data port select value follows the "_x".
- The data port select register follows the "DataPortWrite=".
- The access method for _instPIE0_aliasHOST is:
    1. Write 0005_0001h to DF:FabricConfigAccessControl.
    2. Read/Write the PCICFG address D18F0x040.

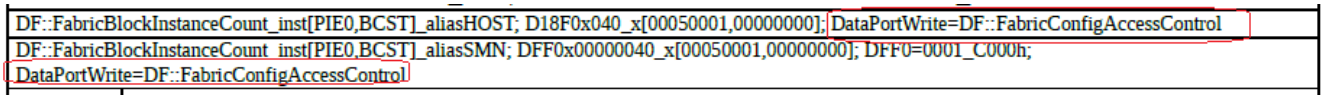


Figure 13: Register Instance Table: Data Port Select

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 do not 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 14: Simple Register Field Example

1.4.4.8 Complex Register Field Format

Content that cannot 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 15: 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 16: 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 17: 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 18: 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 "write-as-read".

- 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. Not volatile indicates that software may service a read from the results of a previous read and that a write may be dropped if it's value matches the value previously read or written.

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 read value that applies at all times.

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

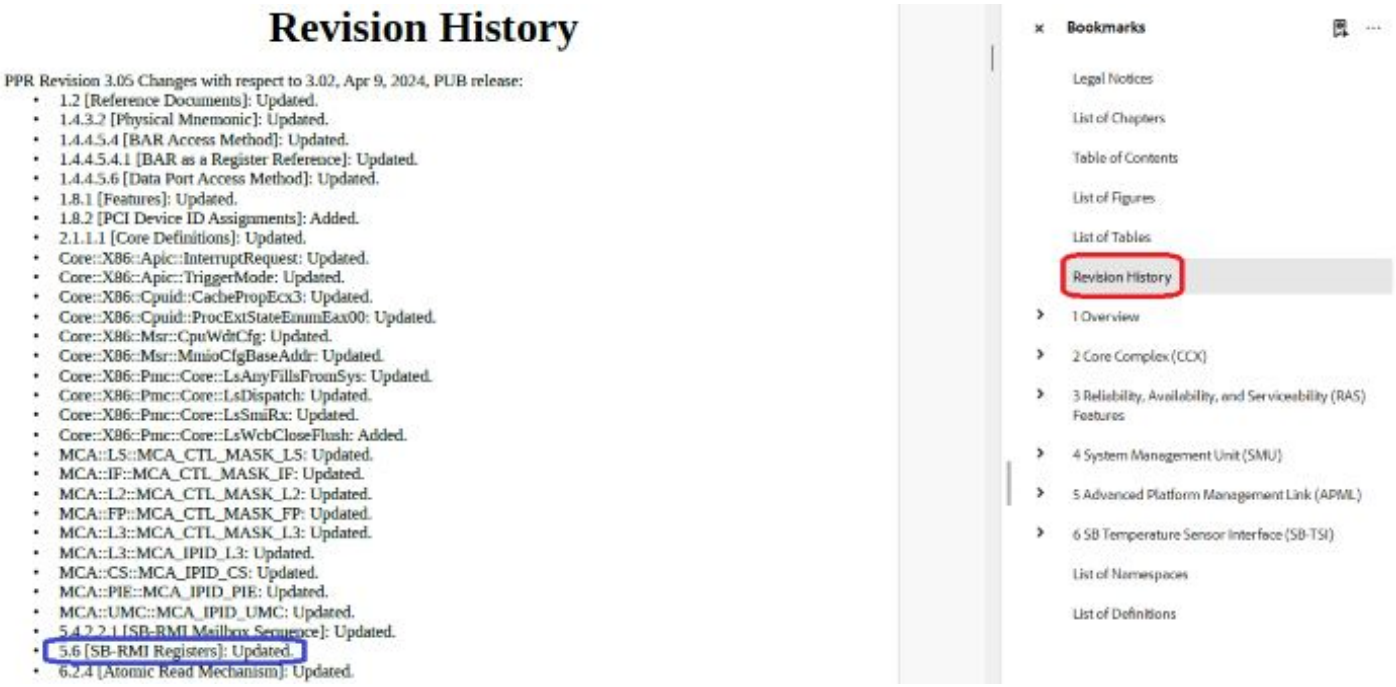
A register can optionally have either a valid values table or a valid bit table:

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

1.4.5 Revision History and Change Bar Notation

If a set of PDFs is generated to show differences with respect to a previous release, then:

- A revision history table is generated and exists before the Overview section. (highlighted in red in the following figure)
- The top line indicates what release the changes are in reference to.
- Changes in the revision history have 3 types:
    - Add. A heading, register or field is added.
    - Delete. A heading, register, or field is deleted.
    - Updated. A heading, register, or field is updated.



Revision History

PPR Revision 3.05 Changes with respect to 3.02, Apr 9, 2024, PUB release:

- 1.2 [Reference Documents]: Updated.
- 1.4.3.2 [Physical Mnemonic]: Updated.
- 1.4.4.5.4 [BAR Access Method]: Updated.
- 1.4.4.5.4.1 [BAR as a Register Reference]: Updated.
- 1.4.4.5.6 [Data Port Access Method]: Updated.
- 1.8.1 [Features]: Updated.
- 1.8.2 [PCI Device ID Assignments]: Added.
- 2.1.1.1 [Core Definitions]: Updated.
- Core::X86::Apic::InterruptRequest: Updated.
- Core::X86::Apic::TriggerMode: Updated.
- Core::X86::Cpuid::CacbePropEcx3: Updated.
- Core::X86::Cpuid::ProcExtStateEnumEax00: Updated.
- Core::X86::Msr::CpuWdtCfg: Updated.
- Core::X86::Msr::MmioCfgBaseAddr: Updated.
- Core::X86::Pmc::Core::LsAnyFillsFromSys: Updated.
- Core::X86::Pmc::Core::LsDispatch: Updated.
- Core::X86::Pmc::Core::LsSmiRx: Updated.
- Core::X86::Pmc::Core::LsWcbCloseFlush: Added.
- MCA::LS::MCA_CTL_MASK_LS: Updated.
- MCA::IF::MCA_CTL_MASK_IF: Updated.
- MCA::L2::MCA_CTL_MASK_L2: Updated.
- MCA::FP::MCA_CTL_MASK_FP: Updated.
- MCA::L3::MCA_CTL_MASK_L3: Updated.
- MCA::L3::MCA_IPID_L3: Updated.
- MCA::CS::MCA_IPID_CS: Updated.
- MCA::PIE::MCA_IPID_PIE: Updated.
- MCA::UMC::MCA_IPID_UMC: Updated.
- 5.4.2.2.1 [SB-RMI Mailbox Sequence]: Updated.
- 5.6 [SB-RMI Registers]: Updated.
- 6.2.4 [Atomic Read Mechanism]: Updated.

Bookmarks

- Legal Notices
- List of Chapters
- Table of Contents
- List of Figures
- List of Tables
- Revision History
- 1 Overview
- 2 Core Complex (CCX)
- 3 Reliability, Availability, and Serviceability (RAS) Features
- 4 System Management Unit (SMU)
- 5 Advanced Platform Management Link (APML)
- 6 SB Temperature Sensor Interface (SB-TSI)
- List of Namespaces
- List of Definitions

Figure 19: Revision History Format Example

The following diagram shows the notation for changes:

- If a change exists on a line then a thin vertical bar is rendered in the left margin, as circled in blue.
- Deleted text is indicated with amber and strike-through, as circled in red.
- Added text is indicated with amber and underlined, as circled in green.

5.6 SB-RMI Registers

Reads to unimplemented registers may return ~~0b~~ on zero value. Writes to unimplemented registers are discarded.

Figure 20: Change Notation Example

1.5 Definitions

Table 11: Definitions

Term	Description
AGESA™	AMD Generic Encapsulated Software Architecture.
AP	Applications Processor.
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; Writes 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	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.
SSC	Spread Spectrum Clocking.
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.
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
SP5	Server, multiple CCD die plus an IOD die MCM (supports single and dual sockets); up to twelve DDR5 memory channels, up to 128 + 8 PCIe® lanes, and four USB ports.

1.8 Processor Overview

1.8.1 Features

Family 19h Models 10h-1Fh is a server-class microprocessor, a System-On-a-Chip (SOC) in a multi-chip module (MCM).

It is built using a chiplet approach - multiple core/cache complex dies (CCD) plus an I/O die (IOD).

MCM

SP5 Server

The SP5 consists of multiple CCDs plus an IOD, in an SP5 multi-chip module (MCM).

A server system could be a single (1P) or a dual (2P) socket system. In a dual socket system, the sockets are interconnected using up to four xGMI ports (the actual number of xGMI ports depends on bandwidth requirements). Each xGMI port utilizes a bidirectional x16 SERDES interconnect. See section 1.9.5 [SP5 MCM Server Dual Socket].

Core/Cache Die

- Each CCD contains a multi-core Core Complex (CCX).
  - Each core may run in single-thread mode (1T) or two-thread SMT mode (2T).
  - The cache system includes L2 and L3 caches.
    - NOTE: Core::X86::Cpuid::L3CacheId (CPUID_Fn80000006_EDX) register should be examined to determine the L3 size on a given product.

I/O Die

IOD integrates:

- DDR Memory interface
  - Multiple Unified Memory Controllers (UMC), each supporting a DRAM channel
    - DDR5 RDIMM/3DS with transfer rates up-to 4800 Mbps
    - Two 40-bit DDR5 sub channels including 8-bit ECC per DRAM channel
    - Up to two DIMMs per channel

- Two instances of NorthBridge IO (NBIO), each of which supports:
  - Four 9 x16 PCIe® Gen5 controllers
  - One 4x4 PCIe® Gen3 controller. It can be used to attach a Baseband Management Controller (BMC).

- Server Controller Hub (SCH, also called FCH)
  - ACPI - Advanced Configuration and Power Interface, power management
  - CLKGEN/CGPLL - clock generation
  - GPIO (varying number depending on muxing) - general purpose IO
  - Two I2C and four I3C ports - Inter-Integrated Circuit
  - RTC - Real-Time Clock with integrated LDO
  - SMBus (2 ports) - System Management Bus
  - SPI/eSPI (2 ports) - Serial Peripheral Interface, embedded SPI
  - UART (one 4 wire port or two 2 wire ports) - Universal Asynchronous Receiver/Transmitter

- CXL™
  - Four x16 links (P0, P1, P2, P3) may be configured as CXL links
  - Each x16 link can support a single x16, two x8 or four x4 device attach
  - CXL Type 3 Devices are supported

- SATA
  - Four SATA controllers, each supporting up to 8 lanes of SATA Gen1/Gen2/Gen3

    - SGPIO - serial GPIO, can be used for activity monitors
    - U.3 and UBM support

- USB
    - Four USB 3.2 Gen2x1 ports, including support for legacy USB speeds (Server platforms support USB 3.2 Gen1x1 only)

1.8.2 PCI Device ID Assignments

PCI segment groups are supported in the Family 19h, Models 10h-1Fh processors. A multi-socket processor system can group its PCI devices into as many as two PCI segment groups. Table 13 [PCI Device ID Assignments with one PCI segment group] shows PCI devices with their PCI Vendor ID and Device ID assignments when one PCI segment group is assigned in a processor system (for multiple-socket processor only PCI devices from one processor are listed).

Table 13: PCI Device ID Assignments with one PCI segment group

Vendor ID	Device ID	Segment	Bus	Device	Function	Component
1022h	14ADh	S0	0	24	0	DF: Device 18h
1022h	14AEh	S0	0	24	1	DF: Device 18h
1022h	14AFh	S0	0	24	2	DF: Device 18h
1022h	14B0h	S0	0	24	3	DF: Device 18h
1022h	14B1h	S0	0	24	4	DF: Device 18h
1022h	14B2h	S0	0	24	5	DF: Device 18h
1022h	14B3h	S0	0	24	6	DF: Device 18h
1022h	14B4h	S0	0	24	7	DF: Device 18h
1022h	14A4h	S0	A0,A1,A2,A3	0	0	Root Complex
1022h	149Eh	S0	A0,A1,A2,A3	0	2	IOMMU
1022h	14A6h	S0	A0,A1,A2,A3	0	3	RCEC
1022h	149Fh	S0	A0,A1,A2,A3	1	0	PCIe® Dummy Host Bridge
1022h	14ABh	S0	A0,A1,A2,A3	1	1	PCIe® GPP Bridge 0
1022h	14ABh	S0	A0,A1,A2,A3	1	2	PCIe® GPP Bridge 1
1022h	14ABh	S0	A0,A1,A2,A3	1	3	PCIe® GPP Bridge 2
1022h	14ABh	S0	A0,A1,A2,A3	1	4	PCIe® GPP Bridge 3
1022h	14ABh	S0	A0,A1,A2,A3	1	5	PCIe® GPP Bridge 4
1022h	14ABh	S0	A0,A1,A2,A3	1	6	PCIe® GPP Bridge 5
1022h	14ABh	S0	A0,A1,A2,A3	1	7	PCIe® GPP Bridge 6
1022h	149Fh	S0	A0,A1,A2,A3	2	0	PCIe® Dummy Host Bridge
1022h	14ABh	S0	A0,A1,A2,A3	2	1	PCIe® GPP Bridge 7
1022h	14ABh	S0	A0,A1,A2,A3	2	2	PCIe® GPP Bridge 8
1022h	149Fh	S0	A0,A1,A2,A3	3	0	PCIe® Dummy Host Bridge
1022h	14A5h	S0	A0,A1,A2,A3	3	1	PCIe® GPP Bridge 0
1022h	14A5h	S0	A0,A1,A2,A3	3	2	PCIe® GPP Bridge 1
1022h	14A5h	S0	A0,A1,A2,A3	3	3	PCIe® GPP Bridge 2
1022h	14A5h	S0	A0,A1,A2,A3	3	4	PCIe® GPP Bridge 3
1022h	14A5h	S0	A0,A1,A2,A3	3	5	PCIe® GPP Bridge 4
1022h	14A5h	S0	A0,A1,A2,A3	3	6	PCIe® GPP Bridge 5
1022h	14A5h	S0	A0,A1,A2,A3	3	7	PCIe® GPP Bridge 6
1022h	149Fh	S0	A0,A1,A2,A3	4	0	PCIe® Dummy Host Bridge
1022h	14A5h	S0	A0,A1,A2,A3	4	1	PCIe® GPP Bridge 7

1022h	14A5h	S0	A0,A1,A2,A3	4	2	PCIe® GPP Bridge 8
1022h	149Fh	S0	A0,A2	5	0	PCIe® Dummy Host Bridge
1022h	14AAh	S0	A0,A2	5	1	PCIe® GPP Bridge 0
1022h	14AAh	S0	A0,A2	5	2	PCIe® GPP Bridge 1
1022h	14AAh	S0	A0,A2	5	3	PCIe® GPP Bridge 2
1022h	14AAh	S0	A0,A2	5	4	PCIe® GPP Bridge 3
1022h	149Fh	S0	A0,A1,A2,A3	7	0	PCIe® Dummy Host Bridge
1022h	14A7h	S0	A0	7	1	Internal PCIe® GPP Bridge 0 to Bus B0
1022h	14A7h	S0	A0	7	2	Internal PCIe® GPP Bridge 0 to Bus C0
1022h	14A7h	S0	A1	7	1	Internal PCIe® GPP Bridge 1 to Bus D1
1022h	14A7h	S0	A2	7	1	Internal PCIe® GPP Bridge 0 to Bus B2
1022h	14A7h	S0	A2	7	2	Internal PCIe® GPP Bridge 0 to Bus C2
1022h	14A7h	S0	A3	7	1	Internal PCIe® GPP Bridge 1 to Bus D3
1022h	14ACh	S0	B0,B2	0	0	PCIe® Dummy Function
1022h	14C0h	S0	B0,B2	0	2	Primary PCIe® Non-TransparentBridge (NTB)
1022h	14C1h	S0	B0,B2	0	3	Secondary vNTB
1022h	157Fh	S0	B0,B2	0	4	USB
1022h	14CAh	S0	B0,B2	0	5	PSP
1022h	14CBh	S0	B0,B2	0	6	ACP
1022h	14CCh	S0	B0,B2	0	7	HD Audio Controller (AZ)
1022h	7901h	S0	C0,C2	0	0	SATA0 (SATA AHCI controller)
1022h	7901h	S0	C0,C2	0	1	SATA1 (SATA AHCI controller)
1022h	7904h	S0	C0,C2	0	0	SATA0 (AMD AHCI driver)
1022h	7904h	S0	C0,C2	0	1	SATA1 (AMD AHCI driver)
1022h	7916h	S0	C0,C2	0	0	SATA0 (3rd party RAID driver)
1022h	7916h	S0	C0,C2	0	1	SATA1 (3rd party RAID driver)
1022h	7917h	S0	C0,C2	0	0	SATA0 (other 3rd party RAID driver)
1022h	7917h	S0	C0,C2	0	1	SATA1 (other 3rd party RAID driver)
1022h	14ACh	S0	D1,D3	0	0	PCIe® Dummy Function
1022h	14C0h	S0	D1,D3	0	2	Primary PCIe® Non-TransparentBridge (NTB)
1022h	14C1h	S0	D1,D3	0	3	Secondary vNTB
1022h	14C6h	/	F0,F2	0	0	Switch US in PCIe® (PCIESW.US)
1022h	14C4h	/	G0,G2	0	0	Switch NBIF DS (PCIE SW.DS)
1022h	14C2h	/	H0,H2	0	0	PCIe® Dummy Function
1022h	14C3h	/	H0,H2	0	1	PCIe® Non-Transparent Bridge
1022h	14C5h	/	H0,H2	0	2	NVME

Table 14 [PCI Device ID Assignments with two PCI segment groups] shows PCI devices with their PCI Vendor ID and Device ID assignments when two PCI segment groups are assigned in a processor system (for multiple-socket processor only PCI devices from one processor are listed).

Table 14: PCI Device ID Assignments with two PCI segment groups

Vendor ID	Device ID	Segment	Bus	Device	Function	Component
1022h	14ADh	S0	0	24	0	DF: Device 18h
1022h	14AEh	S0	0	24	1	DF: Device 18h
1022h	14AFh	S0	0	24	2	DF: Device 18h

1022h	14B0h	S0	0	24	3	DF: Device 18h
1022h	14B1h	S0	0	24	4	DF: Device 18h
1022h	14B2h	S0	0	24	5	DF: Device 18h
1022h	14B3h	S0	0	24	6	DF: Device 18h
1022h	14B4h	S0	0	24	7	DF: Device 18h
1022h	14A4h	S0	A0,A1	0	0	Root Complex
1022h	149Eh	S0	A0,A1	0	2	IOMMU
1022h	14A6h	S0	A0,A1	0	3	RCEC
1022h	149Fh	S0	A0,A1	1	0	PCIe® Dummy Host Bridge
1022h	14ABh	S0	A0,A1	1	1	PCIe® GPP Bridge 0
1022h	14ABh	S0	A0,A1	1	2	PCIe® GPP Bridge 1
1022h	14ABh	S0	A0,A1	1	3	PCIe® GPP Bridge 2
1022h	14ABh	S0	A0,A1	1	4	PCIe® GPP Bridge 3
1022h	14ABh	S0	A0,A1	1	5	PCIe® GPP Bridge 4
1022h	14ABh	S0	A0,A1	1	6	PCIe® GPP Bridge 5
1022h	14ABh	S0	A0,A1	1	7	PCIe® GPP Bridge 6
1022h	149Fh	S0	A0,A1	2	0	PCIe® Dummy Host Bridge
1022h	14ABh	S0	A0,A1	2	1	PCIe® GPP Bridge 7
1022h	14ABh	S0	A0,A1	2	2	PCIe® GPP Bridge 8
1022h	149Fh	S0	A0,A1	3	0	PCIe® Dummy Host Bridge
1022h	14A5h	S0	A0,A1	3	1	PCIe® GPP Bridge 0
1022h	14A5h	S0	A0,A1	3	2	PCIe® GPP Bridge 1
1022h	14A5h	S0	A0,A1	3	3	PCIe® GPP Bridge 2
1022h	14A5h	S0	A0,A1	3	4	PCIe® GPP Bridge 3
1022h	14A5h	S0	A0,A1	3	5	PCIe® GPP Bridge 4
1022h	14A5h	S0	A0,A1	3	6	PCIe® GPP Bridge 5
1022h	14A5h	S0	A0,A1	3	7	PCIe® GPP Bridge 6
1022h	149Fh	S0	A0,A1	4	0	PCIe® Dummy Host Bridge
1022h	14A5h	S0	A0,A1	4	1	PCIe® GPP Bridge 7
1022h	14A5h	S0	A0,A1	4	2	PCIe® GPP Bridge 8
1022h	149Fh	S0	A0	5	0	PCIe® Dummy Host Bridge
1022h	14AAh	S0	A0	5	1	PCIe® GPP Bridge 0
1022h	14AAh	S0	A0	5	2	PCIe® GPP Bridge 1
1022h	14AAh	S0	A0	5	3	PCIe® GPP Bridge 2
1022h	14AAh	S0	A0	5	4	PCIe® GPP Bridge 3
1022h	149Fh	S0	A0,A1	7	0	PCIe® Dummy Host Bridge
1022h	14A7h	S0	A0	7	1	Internal PCIe® GPP Bridge 0 to Bus B0
1022h	14A7h	S0	A0	7	2	Internal PCIe® GPP Bridge 0 to Bus C0
1022h	14A7h	S0	A1	7	1	Internal PCIe® GPP Bridge 1 to Bus D1
1022h	14ACh	S0	B0	0	0	PCIe® Dummy Function
1022h	14C0h	S0	B0	0	2	Primary PCIe® Non-TransparentBridge (NTB)
1022h	14C1h	S0	B0	0	3	Secondary vNTB
1022h	157Fh	S0	B0	0	4	USB
1022h	14CAh	S0	B0	0	5	PSP
1022h	14CBh	S0	B0	0	6	ACP
1022h	14CCh	S0	B0	0	7	HD Audio Controller (AZ)
1022h	7901h	S0	C0	0	0	SATA0 (SATA AHCI controller)
1022h	7901h	S0	C0	0	1	SATA1 (SATA AHCI controller)

1022h	7904h	S0	C0	0	0	SATA0 (AMD AHCI driver)
1022h	7904h	S0	C0	0	1	SATA1 (AMD AHCI driver)
1022h	7916h	S0	C0	0	0	SATA0 (3rd party RAID driver)
1022h	7916h	S0	C0	0	1	SATA1 (3rd party RAID driver)
1022h	7917h	S0	C0	0	0	SATA0 (other 3rd party RAID driver)
1022h	7917h	S0	C0	0	1	SATA1 (other 3rd party RAID driver)
1022h	14ACh	S0	D1	0	0	PCIe® Dummy Function
1022h	14C0h	S0	D1	0	2	Primary PCIe® Non-TransparentBridge (NTB)
1022h	14C1h	S0	D1	0	3	Secondary vNTB
1022h	14A4h	S1	A2,A3	0	0	Root Complex
1022h	149Eh	S1	A2,A3	0	2	IOMMU
1022h	14A6h	S1	A2,A3	0	3	RCEC
1022h	149Fh	S1	A2,A3	1	0	PCIe® Dummy Host Bridge
1022h	14ABh	S1	A2,A3	1	1	PCIe® GPP Bridge 0
1022h	14ABh	S1	A2,A3	1	2	PCIe® GPP Bridge 1
1022h	14ABh	S1	A2,A3	1	3	PCIe® GPP Bridge 2
1022h	14ABh	S1	A2,A3	1	4	PCIe® GPP Bridge 3
1022h	14ABh	S1	A2,A3	1	5	PCIe® GPP Bridge 4
1022h	14ABh	S1	A2,A3	1	6	PCIe® GPP Bridge 5
1022h	14ABh	S1	A2,A3	1	7	PCIe® GPP Bridge 6
1022h	149Fh	S1	A2,A3	2	0	PCIe® Dummy Host Bridge
1022h	14ABh	S1	A2,A3	2	1	PCIe® GPP Bridge 7
1022h	14ABh	S1	A2,A3	2	2	PCIe® GPP Bridge 8
1022h	149Fh	S1	A2,A3	3	0	PCIe® Dummy Host Bridge
1022h	14A5h	S1	A2,A3	3	1	PCIe® GPP Bridge 0
1022h	14A5h	S1	A2,A3	3	2	PCIe® GPP Bridge 1
1022h	14A5h	S1	A2,A3	3	3	PCIe® GPP Bridge 2
1022h	14A5h	S1	A2,A3	3	4	PCIe® GPP Bridge 3
1022h	14A5h	S1	A2,A3	3	5	PCIe® GPP Bridge 4
1022h	14A5h	S1	A2,A3	3	6	PCIe® GPP Bridge 5
1022h	14A5h	S1	A2,A3	3	7	PCIe® GPP Bridge 6
1022h	149Fh	S1	A2,A3	4	0	PCIe® Dummy Host Bridge
1022h	14A5h	S1	A2,A3	4	1	PCIe® GPP Bridge 7
1022h	14A5h	S1	A2,A3	4	2	PCIe® GPP Bridge 8
1022h	149Fh	S1	A2	5	0	PCIe® Dummy Host Bridge
1022h	14AAh	S1	A2	5	1	PCIe® GPP Bridge 0
1022h	14AAh	S1	A2	5	2	PCIe® GPP Bridge 1
1022h	14AAh	S1	A2	5	3	PCIe® GPP Bridge 2
1022h	14AAh	S1	A2	5	4	PCIe® GPP Bridge 3
1022h	149Fh	S1	A2,A3	7	0	PCIe® Dummy Host Bridge
1022h	14A7h	S1	A2	7	1	Internal PCIe® GPP Bridge 0 to Bus B2
1022h	14A7h	S1	A2	7	2	Internal PCIe® GPP Bridge 0 to Bus C2
1022h	14A7h	S1	A3	7	1	Internal PCIe® GPP Bridge 1 to Bus D3
1022h	14ACh	S1	B2	0	0	PCIe® Dummy Function
1022h	14C0h	S1	B2	0	2	Primary PCIe® Non-TransparentBridge (NTB)
1022h	14C1h	S1	B2	0	3	Secondary vNTB
1022h	157Fh	S1	B2	0	4	USB
1022h	14CAh	S1	B2	0	5	PSP

1022h	14CBh	S1	B2	0	6	ACP
1022h	14CCh	S1	B2	0	7	HD Audio Controller (AZ)
1022h	7901h	S1	C2	0	0	SATA0 (SATA AHCI controller)
1022h	7901h	S1	C2	0	1	SATA1 (SATA AHCI controller)
1022h	7904h	S1	C2	0	0	SATA0 (AMD AHCI driver)
1022h	7904h	S1	C2	0	1	SATA1 (AMD AHCI driver)
1022h	7916h	S1	C2	0	0	SATA0 (3rd party RAID driver)
1022h	7916h	S1	C2	0	1	SATA1 (3rd party RAID driver)
1022h	7917h	S1	C2	0	0	SATA0 (other 3rd party RAID driver)
1022h	7917h	S1	C2	0	1	SATA1 (other 3rd party RAID driver)
1022h	14ACh	S1	D3	0	0	PCIe® Dummy Function
1022h	14C0h	S1	D3	0	2	Primary PCIe® Non-TransparentBridge (NTB)
1022h	14C1h	S1	D3	0	3	Secondary vNTB
1022h	14C6h	/	F0,F2	0	0	Switch US in PCIe® (PCIESW.US)
1022h	14C4h	/	G0,G2	0	0	Switch NBIF DS (PCIE SW.DS)
1022h	14C2h	/	H0,H2	0	0	PCIe® Dummy Function
1022h	14C3h	/	H0,H2	0	1	PCIe® Non-Transparent Bridge
1022h	14C5h	/	H0,H2	0	2	NVME

In Table 13 [PCI Device ID Assignments with one PCI segment group] and Table 14 [PCI Device ID Assignments with two PCI segment groups] programmable bus numbers are labeled A, B, C, D, etc. Buses with different labels under the same PCI segment group cannot be assigned the same bus number. The boot die in a multi-die processor assigns Bus A to 0.

1.9 System Overview

1.9.1 Mixed Processor Revision Supports

AMD Family 19h Models 10h-1Fh processors with different OPNs or different revisions cannot be mixed in a multiprocessor system. If the BIOS detects an unsupported configuration, the system will halt prior to X86 core release and signal an I/O port 0x80 POST error code.

1.9.2 Minimum Required Thread Count

To support all required Machine Check Architecture (MCA) registers, each AMD Family 19h Models 10h-1Fh processor requires a minimum of:

- 8 threads per processor (either 8 cores with Symmetric Multithreading (SMT) disabled, or 4 cores with SMT enabled), OR
- 1 thread per enabled Core Complex (CCX) in the processor
- whichever is greater

Systems should not be configured below this minimum required thread count. See Chapter3 [Reliability, Availability, and Serviceability (RAS) Features] for details of the MCA registers.

1.9.3 SP5 Socket

AMD Family 19h Models 10h-1Fh processors in SP5 socket target high-end server applications.

1.9.4 SP5 MCM Server Single-Socket

The SP5-based single-socket (1P) system targets commercial servers with a large number of memory channels, memory capacity, and I/O ports. It supports the following form factors:

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

The characteristics and configuration of the SP5 product in single-socket system is shown in Table 15 [SP5 Single (1P) Socket Configuration].

Table 15: SP5 Single (1P) Socket Configuration

	SP5 1P Configuration
Module Type	MCM socketed SMLGA
Module size, pad pitch	72 x 75.4 mm @ 0.94 x 0.81mm pitch
Socket size	93.4 x 120.3 mm SMLGA socket, heat sink actuated
Cores / module	Varies by product
Memory channels	Varies by product
Max DIMM/channel	2
DIMM Type	RDIMM, 3DS
Combo links/module (note1)	Eight – 16-lane links
CXL Link/Module (note1, note2)	Four – 16-lane links
xGMI links	None
PCIe®/module	Up to 128 Gen5 lanes, plus 8 Gen3 lanes
SATA/module (note2)	Up to 32 ports
Native I/O	USB, SPI/eSPI, UART, I2C/I3C, RTC, Power control, etc.

Notes:
1: Combo links can take the form of PCIe® up to Gen5, CXL 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). The same is true for CXL.

1.9.4.1 SP5 Single (1P) Socket Memory Support

The SP5 MCM provides a large number of memory channels. Each channel can accommodate two DDR5 DIMM connectors. The supported DIMM types are:

- One and two rank DDR5 RDIMM.
- Four and eight physical rank DDR5 3DS DIMM.

LRDIMM is not supported.

1.9.4.2 SP5 Single (1P) Socket I/O Support

The I/O interfaces for the Family 19h Models 10h-1Fh die are configurable. The SP5 MCM provides eight 16-lane PCIe® links: P0, P1, P2, P3, G0, G1, G2, and G3 links.

SATA is also supported on links P0 and G3, each link supporting up to 2 x8 SATA Gen1/Gen2/Gen3 ports.

Two 4-lane PCIe® Gen3 links are also available: P4 and P5. Each can optionally be used as 4 separate 1-lane links.

The one socket system maximizes PCIe® connectivity and can support up to 128 lanes of PCIe® Gen5 (eight 16-lane PCIe® links) plus 8 lanes of PCIe® Gen3. The 16-lane links can be divided up into smaller link widths as long as there are no more than 9 ports per 16-lane group. The SP5 single socket (1P) interface is summarized in Table 16 [SP5 Single Socket Interface].

Table 16: SP5 Single Socket Interface

Link	**Width**	PCIe®	**SATA**	**CXL**
G0	x16	Yes (Gen5)	No	No
G1	x16	Yes (Gen5)	No	No
G2	x16	Yes (Gen5)	No	No
G3	x16	Yes (Gen5)	Yes (2 x8) Gen3	No
P0	x16	Yes (Gen5)	Yes (2 x8) Gen3	Yes
P1	x16	Yes (Gen5)	No	Yes
P2	x16	Yes (Gen5)	No	Yes
P3	x16	Yes (Gen5)	No	Yes
P4	x4	Yes (Gen3)	No	No
P5	x4	Yes (Gen3)	No	No

The SerDes lane and function mapping is shown in Figure 21 [SerDes (P0 through P3 and G0 through G3) Link Bifurcation Options].

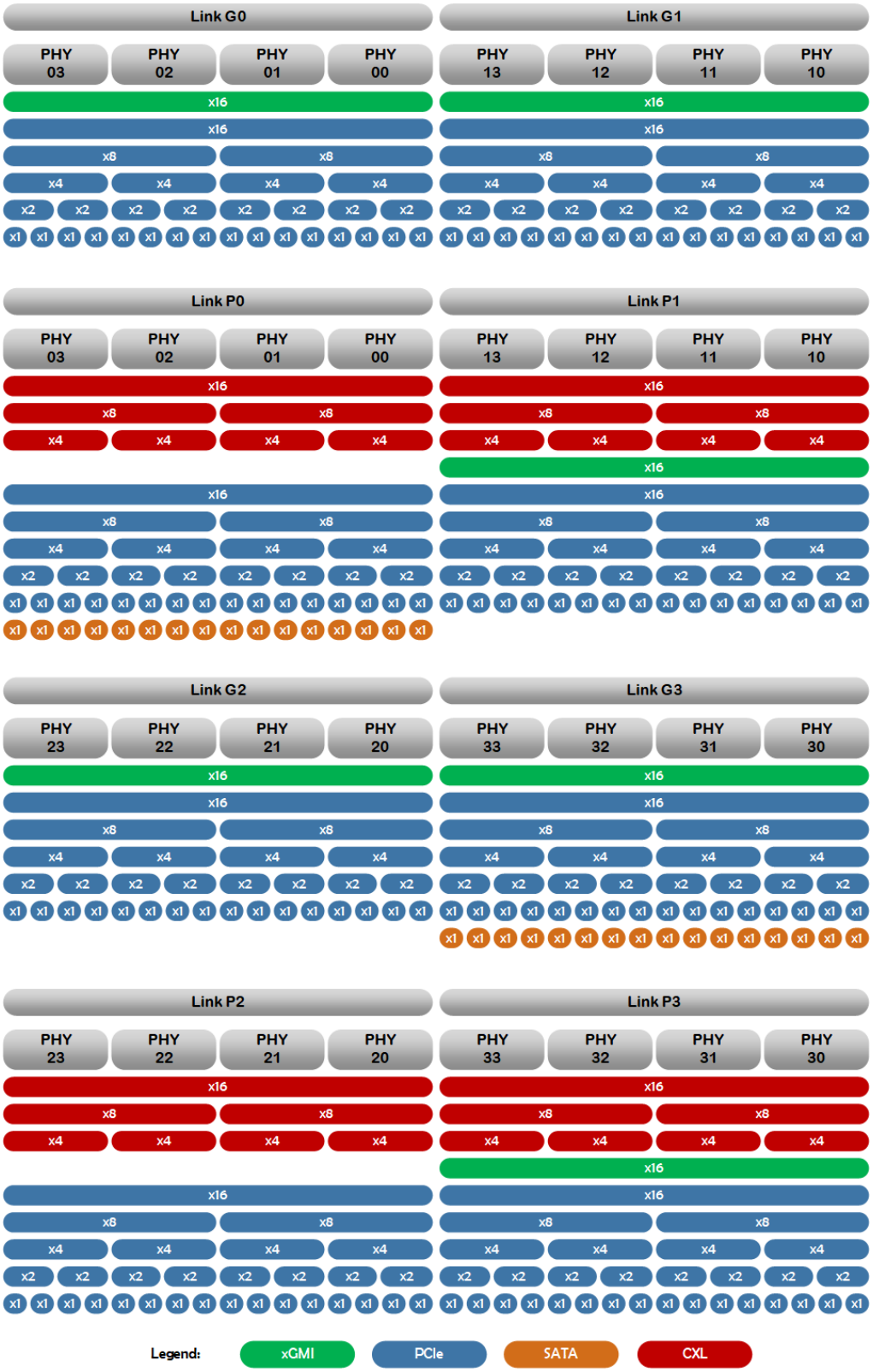


Figure 21: SerDes (P0 through P3 and G0 through G3) Link Bifurcation Options

The function and mapping of the WAFL/PCIe® P4 and P5 lanes is shown in Figure 22 [SerDes Lane/Function Mapping for WAFL/PCIe®].

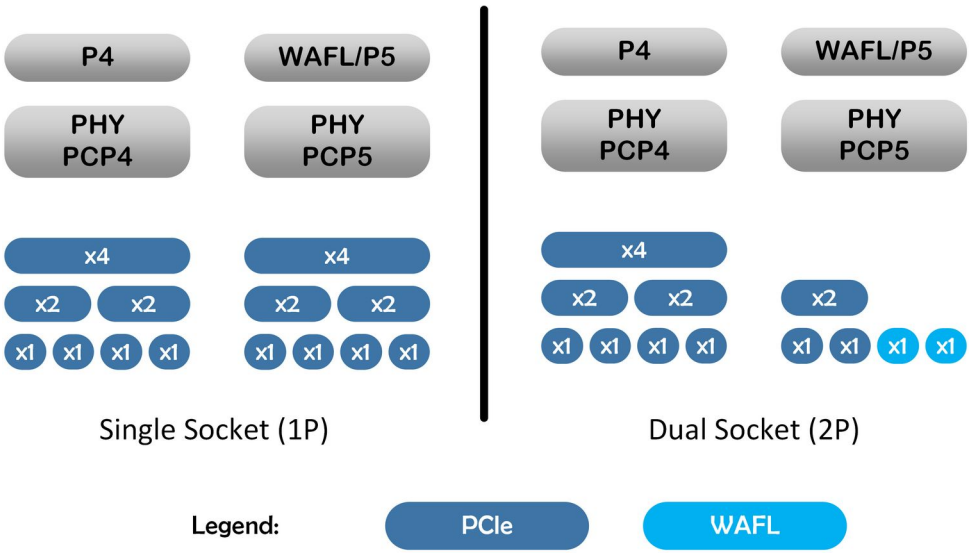


Figure 22: SerDes Lane/Function Mapping for WAFL/PCIe®

1.9.5 SP5 MCM Server Dual Socket

SP5-based Dual (2P) Socket systems target 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 configuration of the SP5 product in dual socket systems are shown in Table 17 [SP5 Dual Socket Characteristics and Configuration].

Table 17: SP5 Dual Socket Characteristics and Configuration

	SP5 2P Configuration
Module Type	MCM socketed SMLGA
Module size, pad pitch	72 x 75.4 mm @ 0.94 x 0.81mm pitch
Socket size	93.4 x 120.3 mm, heat sink actuated
Cores / module	Varies by product
Memory channels/module	Varies by product
Max DIMM/channel	2
DIMM Type	RDIMM, 3DS
Combo links/module (note1)	Up to eight 16-lane links
CXL links/module (note2)	Up to four 16-lane links
xGMI links (note2)	Up to four 16-lane links
PCIe®/module (note 3)	Up to 80 Gen5 lanes, plus 3 Gen3 lanes
SATA/module (note2)	Up to 16 ports
Native I/O	USB, SPI/eSPI, UART, I2C/I3C, RTC, Power control, etc.

Notes:
1: Combo links can take the form of xGMI, SATA, CXL, PCIe®, 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). The same is true for CXL and xGMI.
3: Up to 80 Gen5 lanes based on a three xGMI link 2P interconnect topology, up to 64 Gen5 lanes based on a four xGMI link 2P topology.

1.9.5.1 SP5 Dual (2P) Socket Memory Support

Each SP5 MCM provides a large number of memory channels in a dual socket system. Each channel can accommodate two DDR5 DIMM connectors. The supported DIMM types are:

- One and two rank DDR5 RDIMM.
- Four and eight physical rank DDR5 3DS DIMM.

LRDIMM is not supported.

1.9.5.2 SP5 Dual (2P) Socket Coherent Interconnect Topology

Two SP5 MCM are interconnected using four or three xGMI links, depending on system bandwidth requirements. See section 1.9.5.3 [SP5 Dual (2P) Socket I/O Support].

1.9.5.3 SP5 Dual (2P) Socket I/O Support

The I/O interfaces for the Family 19h Models 10h-1Fh die are configurable. The SP5 MCM provides eight 16-lane "combo" links: P0, P1, P2, P3, G0, G1, G2, and G3 links. All eight combo links support PCIe® Gen5. All links except P0 and P2 support xGMI.

SATA is supported only on links P0 and G3, each link supporting up to 2 x8 SATA Gen1/Gen2/Gen3 ports.

The SerDes lane and function mapping is shown in Figure 21 [SerDes (P0 through P3 and G0 through G3) Link Bifurcation Options].

Two 4-lane PCIe® Gen3 links are also available: P4 and P5. Link P4 supports only the PCIe® protocol. Link P5 lanes [1:0] support either a WAFL link or a PCIe® link, while link P5 lanes [3:2] support PCIe® only. The function and mapping of the x4 PCIe® Gen3 lanes is shown in Figure 22 [SerDes Lane/Function Mapping for WAFL/PCIe®].

The SP5 dual socket (2P) interface is summarized in Table 18 [SP5 Dual Socket Interface].

Table 18: SP5 Dual Socket Interface

Link	Width	PCIe®	SATA	xGMI	WAFL	CXL
G0	x16	Yes (Gen5)	No	Yes	No	No
G1	x16	Yes (Gen5)	No	Yes	No	No
G2	x16	Yes (Gen5)	No	Yes	No	No
G3	x16	Yes (Gen5)	Yes (2 x8) Gen3	Yes	No	No
P0	x16	Yes (Gen5)	Yes (2 x8) Gen3	No	No	Yes
P1	x16	Yes (Gen5)	No	Yes	No	Yes
P2	x16	Yes (Gen5)	No	No	No	Yes
P3	x16	Yes (Gen5)	No	Yes	No	Yes
P4	x4	Yes (Gen3)	No	No	No	No
P5	x4	Yes (Gen3)	No	No	Lanes[1:0]	No

In dual socket (2P) systems, three or four 16-lane links are used for xGMI interconnect between MCMs. That leaves four or five 16-lane links per MCM remaining for non-xGMI. A Dual (2P) Socket using four-link or three-link configurations is summarized below and is shown in Figure 23 [Dual Socket 3G/4G xGMI Topology]. The three-link configuration simply omits the optional xGMI link. The 16-lane links can be divided up into smaller link widths as long as there are no more than 9 ports per 16-lane group.

- G0, G1, G2, G3 to/from G2, G3, G0, G1 (four-link configuration) or
- G0, G1, G2 to/from G2, G3, G0 (three-link configuration)

Furthermore, the WAFL connection between the sockets in a dual socket system is also shown in Figure 23 [Dual Socket 3G/4G xGMI Topology].

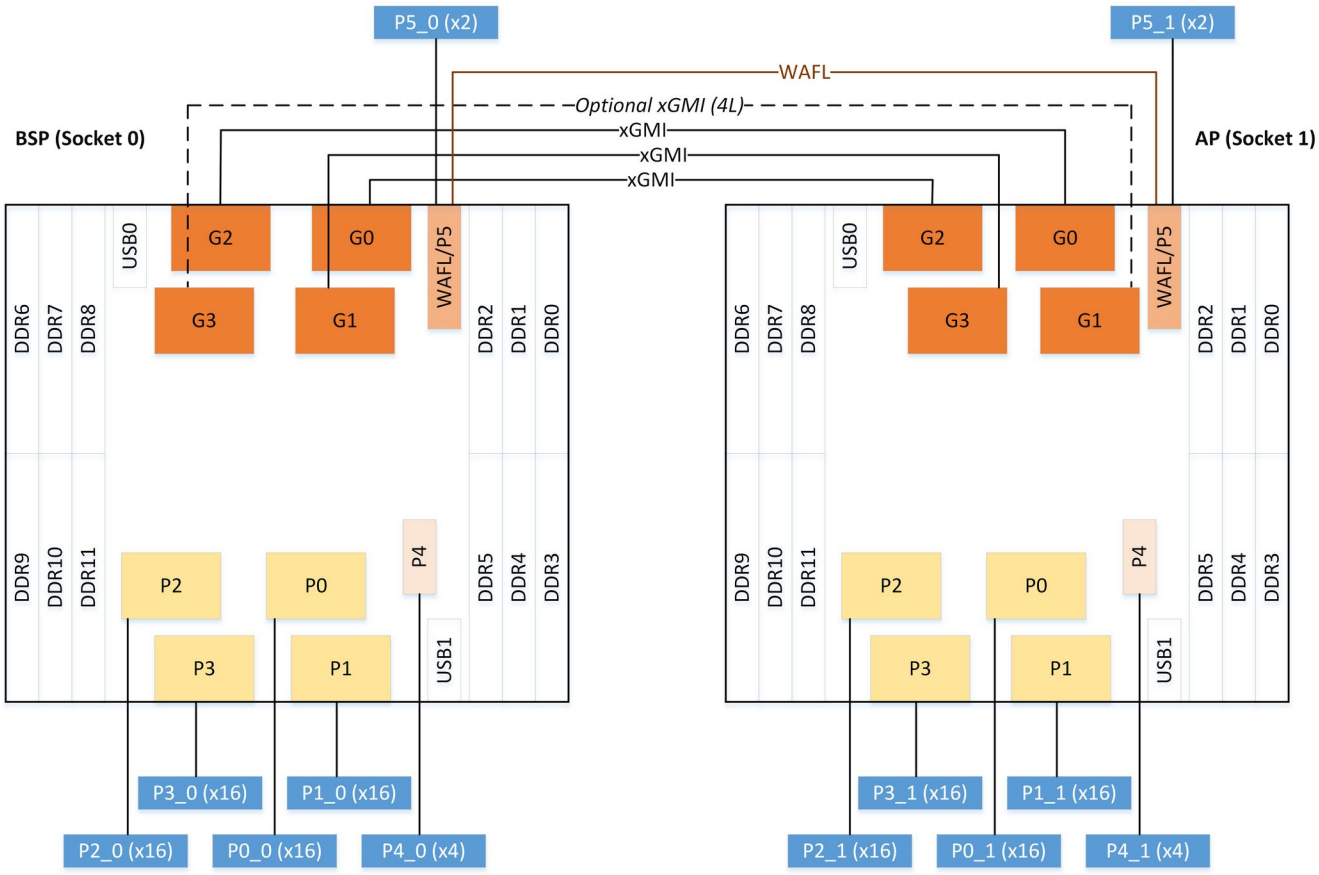


Figure 23: Dual Socket 3G/4G xGMI Topology

As an example, a dual socket (2P) system that maximizes PCIe® connectivity can support up to 160 lanes of PCIe® Gen5 plus 12 lanes of PCIe® Gen3 (80 PCIe® Gen5 plus 6 PCIe® Gen3 lanes per MCM).

A combination of P and G links could be used for socket to socket interconnection. This alternative four-link configuration using 2 G and 2 P links is summarized below and is further illustrated in Figure 24 [Dual Socket 2G + 2P xGMI Topology].

- G0, G2, P1, P3 to/from G2, G0, P3, P1

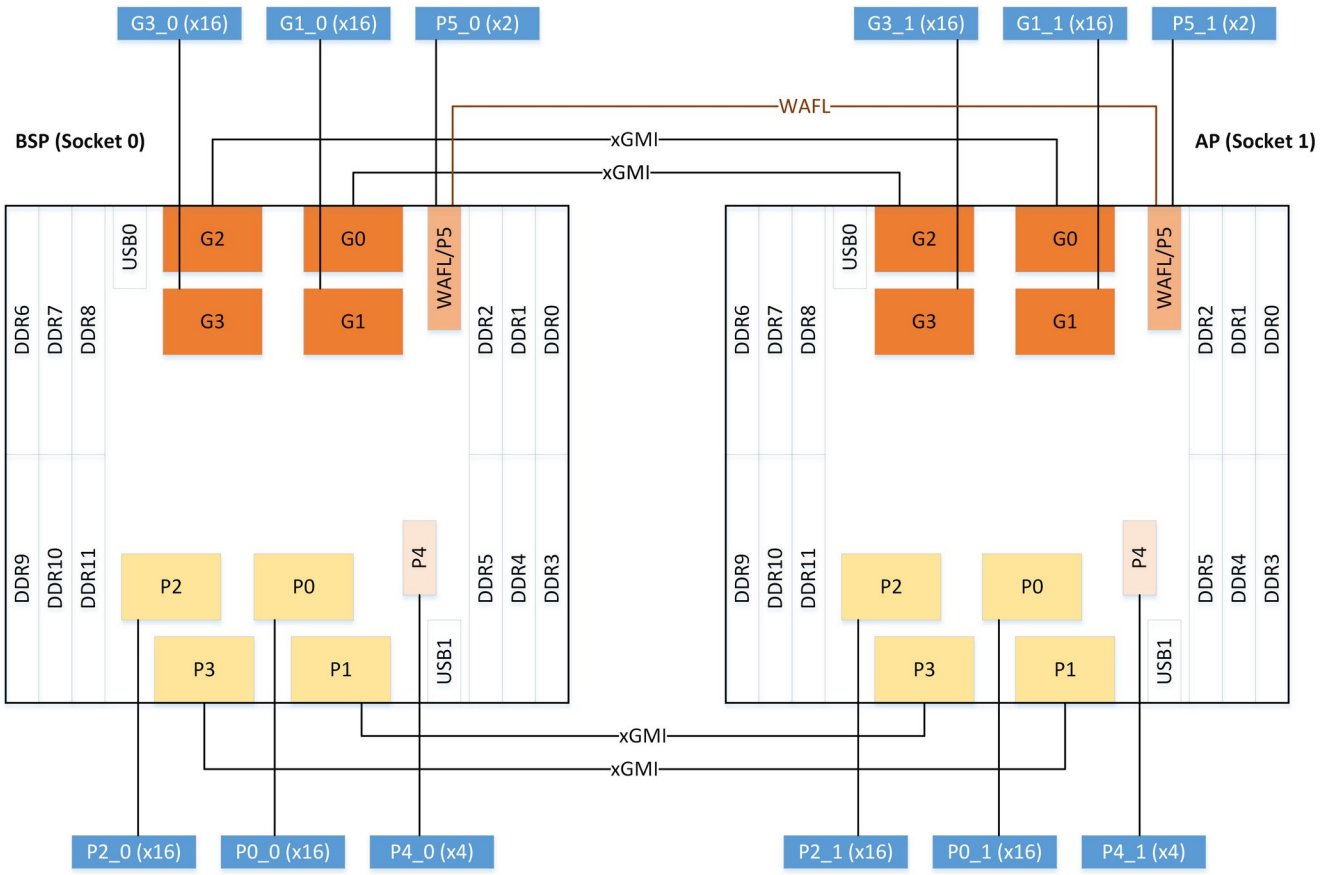


Figure 24: Dual Socket 2G + 2P xGMI Topology

Table 19: xGMI Link Mapping between two sockets

xGMI Configuration	Socket	xGMI Links			
4 links: "G"	BSP (Socket 0)	G0	G1	G2	G3
	AP (Socket 1)	G2	G3	G0	G1
3 links: "G"	BSP (Socket 0)	G0	G1	G2	-
	AP (Socket 1)	G2	G3	G0	-
4 links: 2 "G" and 2 "P"	BSP (Socket 0)	G0	G2	P1	P3
	AP (Socket 1)	G2	G0	P3	P1

2 Core Complex (CCX)

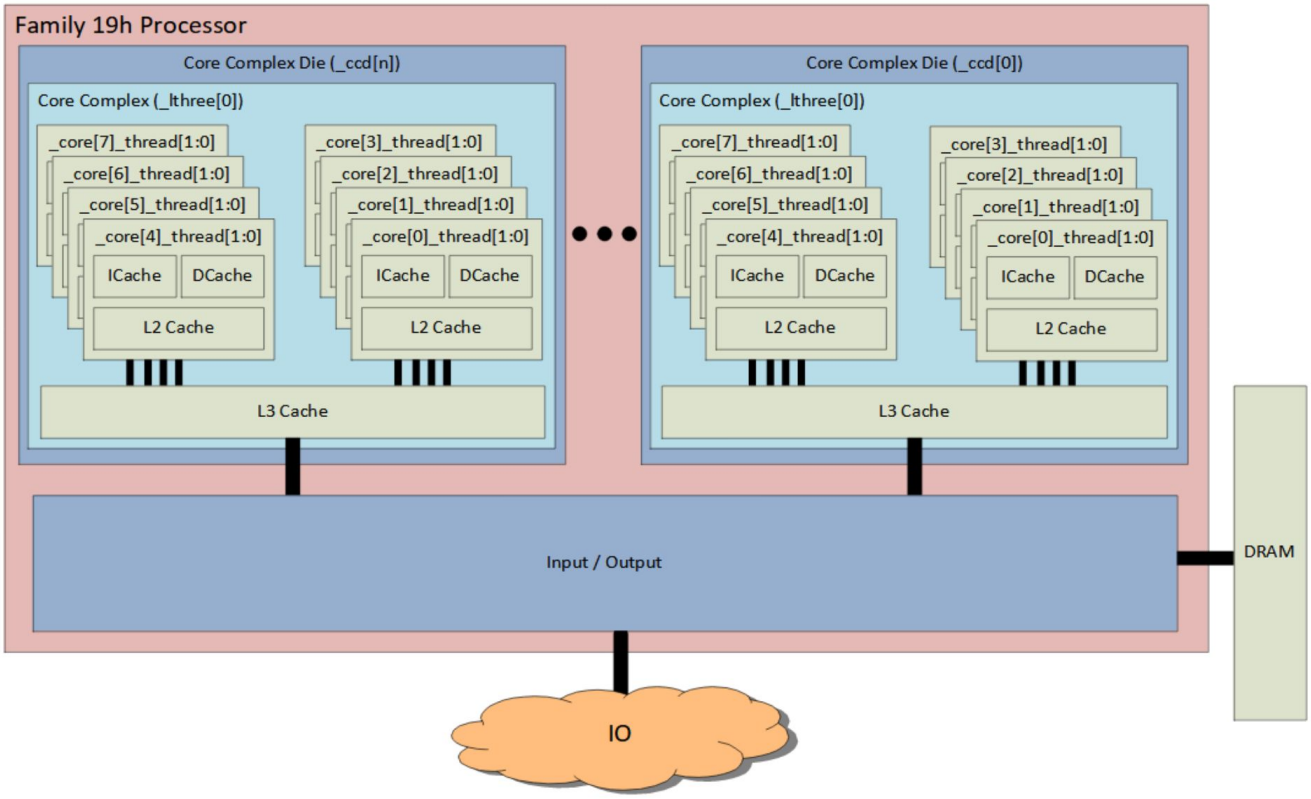


Figure 25: Overview of Family 19 Core Component Numbering

Each Core Complex contains eight (8) Family 19h Model 10h-1Fh x86 multi-thread cores and a shared 32MB L3 cache. Each core has dedicated instruction (32KB) and data (48KB) caches, along with a L2 (1MB) cache. With SMT enabled, each core has 2 threads. With SMT disabled, each core has just a single thread.

2.1 Processor x86 Core

2.1.1 Core Functional Information

2.1.1.1 Core Definitions

Table 20: 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.
HWPF	Hardware Prefetcher.
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.
MPB	Microcode patch block.
NTA	Non-Temporal Access.
PPIN	Protected Processor Inventory Number.
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.
VMPL	Virtual Machine Privilege Level.
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]).

2.1.2 Secure Virtual Machine Mode (SVM)

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

2.1.2.1 BIOS support for SVM Disable

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

2.1.2.1.1 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.

2.1.2.1.2 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.

2.1.2.1.3 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.

2.1.3 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.

2.1.4 Secure Nested Paging (SEV-SNP)

2.1.4.1 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.

2.1.4.2 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 21: 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

2.1.4.3 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 22: 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	(GuestVal & HostVal) != GuestVal

	support for a feature	
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
GreaterThan	The GuestVal must be greater than or equal to HostVal	GuestVal < HostVal

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 23: 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	Bit 31	UnChecked	
		OSXSAVE	UnChecked	
		Others	BitMask	
Fn00000001_EDX	Feature Identifiers	All	BitMask	
Fn00000002_EAX	-	All	UnChecked	
Fn00000002_EBX	-	All	UnChecked	
Fn00000002_ECX	-	All	UnChecked	
Fn00000002_EDX	-	All	UnChecked	
Fn00000004_EAX	-	All	UnChecked	
Fn00000004_EBX	-	All	UnChecked	
Fn00000004_ECX	-	All	UnChecked	
Fn00000004_EDX	-	All	UnChecked	

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 Identifiers	All	BitMask	
Fn00000007_EAX_x01	Structured Extended Feature Identifiers	All	BitMask	
Fn00000007_EBX_x01	Structured Extended Feature Identifiers	All	Strict	No fields
Fn00000007_ECX_x01	Structured Extended Feature Identifiers	All	Strict	No fields
Fn00000007_EDX_x01	Structured Extended Feature Identifiers	All	Strict	No fields
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_x05	Processor Extended State Enumeration	All	Strict	
Fn0000000D_EBX_x05	Processor Extended State Enumeration	All	Strict	
Fn0000000D_ECX_x05	Processor Extended State Enumeration	All	BitMask	
Fn0000000D_EDX_x05	Processor Extended State Enumeration	All	Strict	No fields
Fn0000000D_EAX_x06	Processor Extended State Enumeration	All	Strict	
Fn0000000D_EBX_x06	Processor Extended State Enumeration	All	Strict	
Fn0000000D_ECX_x06	Processor Extended State Enumeration	All	BitMask	
Fn0000000D_EDX_x06	Processor Extended State Enumeration	All	Strict	No fields
Fn0000000D_EAX_x07	Processor Extended State Enumeration	All	Strict	
Fn0000000D_EBX_x07	Processor Extended State Enumeration	All	Strict	
Fn0000000D_ECX_x07	Processor Extended State Enumeration	All	BitMask	
Fn0000000D_EDX_x07	Processor Extended State Enumeration	All	Strict	No fields
Fn0000000D_EAX_x09	Processor Extended State Enumeration	All	Strict	
Fn0000000D_EBX_x09	Processor Extended State Enumeration	All	Strict	
Fn0000000D_ECX_x09	Processor Extended State Enumeration	All	Strict	
Fn0000000D_EDX_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 Enumeration	All	Strict	
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	SVM Revision and Feature Identification	x2AVIC_EXT	BitMask	
		All	Strict	
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	Strict	No fields.
Fn8000001F_EAX	AMD Secure Encryption EAX	All	BitMask	
Fn8000001F_EBX	AMD Secure Encryption EBX	VmplSupported	LessThan	
		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	
Fn80000020_EAX_x02	Enforcement for Slow Memory Bandwidth	All	UnChecked	
Fn80000020_EBX_x02	Enforcement for Slow Memory Bandwidth	All	Strict	No fields
Fn80000020_ECX_x02	Enforcement for Slow Memory Bandwidth	All	Strict	No fields
Fn80000020_EDX_x02	Enforcement for Slow Memory Bandwidth	All	UnChecked	
Fn80000020_EAX_x03	Platform QoS Monitoring Bandwidth Event Configuration	All	Strict	No fields
Fn80000020_EBX_x03	Platform QoS Monitoring Bandwidth Event Configuration	All	UnChecked	

Fn80000020_ECX_x03	Platform QoS Monitoring Bandwidth Event Configuration	All	BitMask	
Fn80000020_EDX_x03	Platform QoS Monitoring Bandwidth Event Configuration	All	Strict	No fields
Fn80000021_EAX	Extended Feature 2 EAX	All	BitMask	
Fn80000021_EBX	Extended Feature 2 EBX	All	UnChecked	
Fn80000021_ECX	Reserved	All	Strict	No fields
Fn80000021_EDX	Reserved	All	Strict	No fields
Fn80000022_EAX	Extended Performance Monitoring and Debug EAX	All	BitMask	
Fn80000022_EBX	Extended Performance Monitoring and Debug EBX	LbrV2StackSz NumPerfCtrCore	LessThan	
		Others	UnChecked	
Fn80000022_ECX	Extended Performance Monitoring and Debug ECX	All	UnChecked	
Fn80000022_EDX	Extended Performance Monitoring and Debug EDX	All	Strict	No fields
Fn80000023_EAX	AMD Secure Multi-Key Encryption EAX	All	BitMask	
Fn80000023_EBX	AMD Secure Multi-Key Encryption EBX	All	UnChecked	
Fn80000023_ECX	AMD Secure Multi-Key Encryption ECX	All	Strict	No fields
Fn80000023_EDX	AMD Secure Multi-Key Encryption EDX	All	Strict	No fields
Fn80000026_EAX_x{0..3}	Extended CPU Topology	All	UnChecked	
Fn80000026_EBX_x{0..3}	Extended CPU Topology	All	UnChecked	
Fn80000026_ECX_x{0..3}	Extended CPU Topology	All	UnChecked	
Fn80000026_EDX_x{0..3}	Extended CPU Topology	All	UnChecked	

2.1.5 Microcode Patching

2.1.5.1 Microcode Patching Overview

The processor contains a small amount of RAM for implementing microcode patches. This patch RAM is loaded by a microcode routine (the Patch RAM Loader) that is part of the normal microcode contained in ROM. The patch RAM contains 64 microlines. Microlines are sets of processor microcode operations which are grouped to form a line of microcode. When the processor powers up, it uses its internal ROM microcode. If no patches are installed, the processor only executes microcode from the ROM.

The Patch RAM Loader function is called via a write to Core::X86::Msr::PATCH_LOADER. The patch loader downloads microcode from the Microcode Patch Block (MPB) stored in memory into the processor patch RAM.

2.1.5.2 Microcode Patch Block (MPB)

The MPB is 5568 bytes in length and consists of two parts. The first part is a 32-byte header. The second part contains code, including the encrypted patch data. The properties of the patch allow for multiple consistency checks after decryption. If any consistency check fails, a #GP is generated. The patch data is encrypted to prevent unauthorized use of the patch mechanism.

Table 24: MPB Definition

Field	Byte Offset	Byte Length	Description
MPB_DATE	0000h	4	A doubleword for "mmddyyyy" in BCD format
Patch Level	0004h	4	A unique patch level as defined by the Revision Guide
MPB_LOADER_ID	0008h	2	A unique ID used for checking if the update is successful after the microcode patch function is executed.
Reserved	000Ah	6	
MPB_NB_ID	0010h	4	The BIOS must match the chipset 1 device ID with this field before executing the microcode patch. A "0" means the patch is not chipset specific.
MPB_SB_ID	0014h	4	The BIOS must match the chipset 2 device ID with this field before executing the microcode patch. A "0" means the patch is not chipset specific.
MPB_REVISION	0018h	4	Microcode Patch Equivalent Processor Id. See the Revision Guide for AMD Family 19h Models 10h-1Fh Processors.
MPB_BIOS_REVISION	001Ch	1	
Load Control	001Dh	1	Bit[0]: SerializedLoad: =0 (default), =1 The patch is required to be loaded serially across all cores in the system. Bit[1]: LoadOnBothThreads: = 0 (default) The patch is required to be loaded on both threads on each core in the system. =1 The patch may be loaded on one thread on each core in the system. Bits[7:2]: Reserved.
Reserved	001Eh	2	
Patch Data	0020h	5536	Patch Data.

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 57-bit address bits of virtual memory space (128 PB) as indicated by Core::X86::Cpuid::LongModeInfo.

2.1.7.2 Physical Address Space

The processor supports a 52-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[63:52].

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 Northbridge: 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.
- Memory Mapped IO (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, segment[6:0], 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 Memory Mapped IO (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

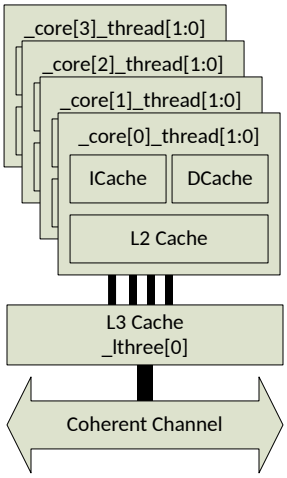


Figure 26: Register Sharing Domains

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 Interrupts

2.1.13.1 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.

2.1.13.1.1 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.

2.1.13.1.2 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.

2.1.13.1.3 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 are stored in Core::X86::Smm::LocalSmiStatus.

2.1.13.1.4 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 25: 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.

2.1.13.1.5 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 26: 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).

2.1.13.1.6 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].

2.1.13.1.7 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.

2.1.13.1.8 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.

2.1.13.1.9 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].

2.1.13.1.10 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.

2.1.13.1.11 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.

2.1.13.2 Local APIC

The processor supports the APIC interrupt controller and the X2APIC interrupt controllers.
See 2.1.13.2.2 [Local APIC Registers] for the APIC registers and Core::X86::Msr::APIC_ID through

Core::X86::Msr::ExtendedInterruptLvtEntries for the X2APIC registers.

2.1.13.2.1 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.

2.1.13.2.1.1 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.

2.1.13.2.1.2 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.

2.1.13.2.1.3 ApicId Enumeration Requirements

Note: Family 19h Model 10h-1Fh 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].

2.1.13.2.1.4 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.

2.1.13.2.1.5 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.

2.1.13.2.1.6 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.

2.1.13.2.1.7 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.

2.1.13.2.1.8 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.

2.1.13.2.1.9 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.

2.1.13.2.1.10 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.

2.1.13.2.1.11 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}
}
```

2.1.13.2.1.12 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.

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

Table 27: 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

2.1.13.2.1.13 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.

2.1.13.2.1.14 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.

2.1.13.2.1.15 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.

2.1.13.2.2 Local APIC Registers

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

Read-only.

_ccd[11:0]_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 2.1.13.2.1.3 [ApicId Enumeration Requirements].
23:0	Reserved.

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

Read-only.

_ccd[11:0]_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,1. 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[11:0]_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[11:0]_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[11:0]_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[11:0]_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 not supported.

_ccd[11:0]_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[11:0]_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[11:0]_lthree0_core[7:0]_thread[1:0]; APICx0E0; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}

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

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

Reset: 0000_00FFh.

_ccd[11:0]_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[11:0]_lthree0_core[7:0]_thread[1:0]_n0; APICx100; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n1; APICx110; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n2; APICx120; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n3; APICx130; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n4; APICx140; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n5; APICx150; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n6; APICx160; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}

_ccd[11:0]_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[11:0]_lthree0_core[7:0]_thread[1:0]_n0; APICx180; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n1; APICx190; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n2; APICx1A0; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n3; APICx1B0; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n4; APICx1C0; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n5; APICx1D0; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n6; APICx1E0; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[11:0]_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.
	ValidValues: See: Core::X86::Apic::TriggerMode[TriggerModeBits] Valid Values.

Core::X86::Apic::TriggerMode[TriggerModeBits] Valid Values

Bit	Description
[0]	0=Edge-triggered interrupt. 1=Level-triggered interrupt.
[1]	0=Edge-triggered interrupt. 1=Level-triggered interrupt.
[2]	0=Edge-triggered interrupt. 1=Level-triggered interrupt.
[3]	0=Edge-triggered interrupt. 1=Level-triggered interrupt.
[4]	0=Edge-triggered interrupt. 1=Level-triggered interrupt.
[5]	0=Edge-triggered interrupt. 1=Level-triggered interrupt.
[6]	0=Edge-triggered interrupt. 1=Level-triggered interrupt.
[7]	0=Edge-triggered interrupt. 1=Level-triggered interrupt.
[8]	0=Edge-triggered interrupt. 1=Level-triggered interrupt.
[9]	0=Edge-triggered interrupt. 1=Level-triggered interrupt.
[10]	0=Edge-triggered interrupt. 1=Level-triggered interrupt.
[11]	0=Edge-triggered interrupt. 1=Level-triggered interrupt.
[12]	0=Edge-triggered interrupt. 1=Level-triggered interrupt.
[13]	0=Edge-triggered interrupt. 1=Level-triggered interrupt.
[14]	0=Edge-triggered interrupt. 1=Level-triggered interrupt.
[15]	0=Edge-triggered interrupt. 1=Level-triggered interrupt.
[16]	0=Edge-triggered interrupt. 1=Level-triggered interrupt.
[17]	0=Edge-triggered interrupt. 1=Level-triggered interrupt.
[18]	0=Edge-triggered interrupt. 1=Level-triggered interrupt.
[19]	0=Edge-triggered interrupt. 1=Level-triggered interrupt.
[20]	0=Edge-triggered interrupt. 1=Level-triggered interrupt.
[21]	0=Edge-triggered interrupt. 1=Level-triggered interrupt.
[22]	0=Edge-triggered interrupt. 1=Level-triggered interrupt.
[23]	0=Edge-triggered interrupt. 1=Level-triggered interrupt.
[24]	0=Edge-triggered interrupt. 1=Level-triggered interrupt.
[25]	0=Edge-triggered interrupt. 1=Level-triggered interrupt.
[26]	0=Edge-triggered interrupt. 1=Level-triggered interrupt.
[27]	0=Edge-triggered interrupt. 1=Level-triggered interrupt.
[28]	0=Edge-triggered interrupt. 1=Level-triggered interrupt.
[29]	0=Edge-triggered interrupt. 1=Level-triggered interrupt.
[30]	0=Edge-triggered interrupt. 1=Level-triggered interrupt.

[31]	0=Edge-triggered interrupt. 1=Level-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[11:0]_lthree0_core[7:0]_thread[1:0]_n0; APICx200; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n1; APICx210; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n2; APICx220; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n3; APICx230; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n4; APICx240; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n5; APICx250; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n6; APICx260; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[11:0]_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.
	ValidValues: See: Core::X86::Apic::InterruptRequest[RequestBits] Valid Values.

Core::X86::Apic::InterruptRequest[RequestBits] Valid Values

Bit	Description
[0]	0=Request bit not set. 1=Request bit set.
[1]	0=Request bit not set. 1=Request bit set.
[2]	0=Request bit not set. 1=Request bit set.
[3]	0=Request bit not set. 1=Request bit set.
[4]	0=Request bit not set. 1=Request bit set.
[5]	0=Request bit not set. 1=Request bit set.
[6]	0=Request bit not set. 1=Request bit set.
[7]	0=Request bit not set. 1=Request bit set.
[8]	0=Request bit not set. 1=Request bit set.
[9]	0=Request bit not set. 1=Request bit set.
[10]	0=Request bit not set. 1=Request bit set.
[11]	0=Request bit not set. 1=Request bit set.
[12]	0=Request bit not set. 1=Request bit set.
[13]	0=Request bit not set. 1=Request bit set.
[14]	0=Request bit not set. 1=Request bit set.
[15]	0=Request bit not set. 1=Request bit set.
[16]	0=Request bit not set. 1=Request bit set.
[17]	0=Request bit not set. 1=Request bit set.
[18]	0=Request bit not set. 1=Request bit set.
[19]	0=Request bit not set. 1=Request bit set.
[20]	0=Request bit not set. 1=Request bit set.
[21]	0=Request bit not set. 1=Request bit set.
[22]	0=Request bit not set. 1=Request bit set.
[23]	0=Request bit not set. 1=Request bit set.
[24]	0=Request bit not set. 1=Request bit set.
[25]	0=Request bit not set. 1=Request bit set.
[26]	0=Request bit not set. 1=Request bit set.
[27]	0=Request bit not set. 1=Request bit set.
[28]	0=Request bit not set. 1=Request bit set.
[29]	0=Request bit not set. 1=Request bit set.

[30]	0=Request bit not set. 1=Request bit set.
[31]	0=Request bit not set. 1=Request bit set.

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[11:0]_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[11:0]_lthree0_core[7:0]_thread[1:0]; APICx300; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}

Bits	Description
31:20	Reserved.
19:18	**DestShrthnd**: **destination shorthand**. Read-write. Reset: 0h.
	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.
	ValidValues:

Value	Description
0h	No shorthand (Destination field).
1h	Self.
2h	All including self.
3h	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).

Bits	Description
17:16	**RemoteRdStat**. Read-only. Reset: 0h.
	ValidValues:

Value	Description
0h	Read was invalid.
1h	Delivery pending.
2h	Delivery complete and access was valid.
3h	Reserved.

Bits	Description
15	**TM**: **trigger mode**. Read-write. Reset: 0. 0=Edge triggered. 1=Level triggered. Indicates how this interrupt is triggered.
14	**Level**. Read-write. Reset: 0. 0=De-asserted. 1=Asserted.
13	Reserved.
12	**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.
11	**DM**: **destination mode**. Read-write. Reset: 0. 0=Physical. 1=Logical.
10:8	**MsgType**. Read-write. Reset: 0h. The message types are encoded as follows:
	ValidValues:

Value	Description
0h	Fixed.
1h	Lowest Priority.
2h	SMI.
3h	Reserved.
4h	NMI.
5h	INIT.
6h	Startup.
7h	External interrupt.

Bits	Description
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[11:0]_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[11:0]_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. See2.1.13.2.1.14 [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[11:0]_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. See2.1.13.2.1.14 [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[11:0]_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. See2.1.13.2.1.14 [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[11:0]_lthree0_core[7:0]_thread[1:0]_n0; APICx350; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}

_ccd[11:0]_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. See2.1.13.2.1.14 [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[11:0]_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 2.1.13.2.1.14 [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[11:0]_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[11:0]_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[11:0]_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[11:0]_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. See2.1.13.2.1.8 [Interrupt Masking].

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

Read-write. Reset: 0000_0000h.

_ccd[11:0]_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[11:0]_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[11:0]_lthree0_core[7:0]_thread[1:0]_n0; APICx480; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n1; APICx490; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n2; APICx4A0; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n3; APICx4B0; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n4; APICx4C0; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n5; APICx4D0; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n6; APICx4E0; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[11:0]_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[11:0]_lthree0_core[7:0]_thread[1:0]_n0; APICx500; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n1; APICx510; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n2; APICx520; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}
_ccd[11:0]_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. See2.1.13.2.1.14 [Generalized Local Vector Table].
7:0	**Vector**. Read-write. Reset: 00h. Interrupt vector number.

2.1.14 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.

2.1.14.1 CPUID Instruction Functions

The following provides processor specific details about CPUID.

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

Read-only. Reset: Fixed,0000_0010h.

_ccd[11:0]_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[11:0]_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[11:0]_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[11:0]_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.

_ccd[11:0]_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: 1h. 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: Xh. Processor stepping (revision) for a specific model.

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

Read-only.

_ccd[11:0]_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**. Read-only. Reset: Fixed,XXh. Specifies the number of threads in the processor as 0FFh&(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[11:0]_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: X. 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[11:0]_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[11:0]_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[11:0]_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[11:0]_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[11:0]_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[11:0]_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[11:0]_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[11:0]_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[11:0]_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_0001h.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn00000007_EAX_x00

Bits	Description
31:0	**StructExtFeatIdMax**. Read-only. Reset: Fixed,0000_0001h. The largest CPUID Fn0000_0007 sub-function supported by the processor implementation.

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

Read-only. Reset: Fixed,F1BF_97A9h.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn00000007_EBX_x00

Bits	Description
31	**AVX512VL**. Read-only. Reset: Fixed,1. 1=Extends AVX512 instructions to 128 and 256 bits.
30	**AVX512BW**. Read-only. Reset: Fixed,1. 1=AVX512BW packed integer instructions.
29	**SHA**. Read-only. Reset: Fixed,1. 1=SHA Extensions available.
28	**AVX512CD**. Read-only. Reset: Fixed,1. 1=AVX512 Conflict Detection for vectorizing loops supported.
27:25	Reserved.
24	**CLWB**. Read-only. Reset: Fixed,1. 1=CLWB (Cache Line Write Back) instruction supported.
23	**CLFSHOPT**. Read-only. Reset: Fixed,1. 1=CLFSHOPT (Optimized Cache Line Flush) instruction supported.
22	Reserved.
21	**AVX512_IFMA**. Read-only. Reset: Fixed,1. 1=AVX512 integer fused mul-add instructions supported.
20	**SMAP**. Read-only. Reset: Fixed,1. 1=Secure Mode Access Prevention is supported.
19	**ADX**. Read-only. Reset: Fixed,1. 1=ADCX and ADOX instructions are supported.
18	**RDSEED**. Read-only. Reset: Fixed,1. 1=RDSEED instruction is supported.
17	**AVX512DQ**. Read-only. Reset: Fixed,1. 1=AVX512DQ packed integer instructions supported.
16	**AVX512F**. Read-only. Reset: Fixed,1. 1=AVX512 Foundation supported.
15	**PQE**. Read-only. Reset: Fixed,1. 1=Cache Allocation Technology is supported.
14:13	Reserved.
12	**PQM**. Read-only. Reset: Fixed,1. 1=Platform QoS Monitoring is supported.
11	Reserved.
10	**INVPCID**. Read-only. Reset: Fixed,1. 1=INVPCID instruction is supported.
9	**ERMS**. Read-only. Reset: Fixed,1. 1=Enhanced REP MOVSB/STOSB is supported.
8	**BMI2**. Read-only. Reset: Fixed,1. 1=Bit manipulation group 2 instructions are supported.
7	**SMEP**. Read-only. Reset: Fixed,1. 1=Supervisor Mode Execution protection is supported.
6	Reserved.
5	**AVX2**. Read-only. Reset: Fixed,1. 1=AVX extension support is supported.
4	Reserved.
3	**BMI1**. Read-only. Reset: Fixed,1. 1=Bit manipulation group 1 instructions are supported.
2:1	Reserved.
0	**FSGSBASE**. Read-only. Reset: Fixed,1. 1=FS and GS base read write instruction are supported.

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

Read-only.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn00000007_ECX_x00

Bits	Description
31:23	Reserved.
22	**RDPID**. Read-only. Reset: Fixed,1. 1=Read Processor ID instruction is supported.
21:17	Reserved.
16	**LA57**. Read-only. Reset: Fixed,1. 1=57-bit virtual address translation is supported.
15	Reserved.
14	**AVX512_VPOPCNTDQ**. Read-only. Reset: Fixed,1. 1=AVX-512 VPOPCNTD/Q instructions supported.
13	Reserved.
12	**AVX512_BITALG**. Read-only. Reset: Fixed,1. 1=AVX-512 bit algorithm instructions VPSHUFBITQMB and VPOPCNTB/W supported.
11	**AVX512_VNNI**. Read-only. Reset: Fixed,1. 1=AVX512 vector neural network instructions supported.
10	**VPCLMULQDQ**. Read-only. Reset: X. 1=Vector VPCLMULQDQ instruction supported.
9	**VAES**. Read-only. Reset: X. 1=Vector VAES(ENC|DEC), VAES(ENC|DEC)LAST instruction supported.
8	**GFNI**. Read-only. Reset: Fixed,1. 1=Galois Field New Instructions supported.
7	**CET_SS**. Read-only. Reset: 1. 1=Shadow stack supported.
6	**AVX512_VBMI2**. Read-only. Reset: Fixed,1. 1=AVX512 vector byte permutation instruction 2 supported.
5:4	Reserved.
3	**PKU**. Read-only. Reset: Fixed,1. 1=Protection Keys are supported.
2	**UMIP**. Read-only. Reset: Fixed,1. 1=User Mode Instruction Prevention is supported.
1	**AVX512_VBMI**. Read-only. Reset: Fixed,1. 1=AVX512 vector byte permutation instruction are supported.
0	Reserved.

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

Read-only. Reset: Fixed,1000_0010h.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn00000007_EDX_x00

Bits	Description
31:29	Reserved.
28	**L1D_FLUSH**. Read-only. Reset: Fixed,1. 1=L1D_FLUSH supported in FLUSH_CMD MSR is supported.
27:5	Reserved.
4	**FSRM**. Read-only. Reset: Fixed,1. 1=Fast Short REP MOVSB is supported.
3:0	Reserved.

CPUID_Fn00000007_EAX_x01 [Structured Extended Feature Identifiers] (Core::X86::Cpuid::StructExtFeatIdEax1)

Read-only. Reset: Fixed,0000_0020h.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn00000007_EAX_x01

Bits	Description
31:6	Reserved.
5	**AVX512_BF16**. Read-only. Reset: Fixed,1. 1=BFLOAT16 instructions are supported.
4:0	Reserved.

CPUID_Fn00000007_EBX_x01 [Structured Extended Feature Identifiers] (Core::X86::Cpuid::StructExtFeatIdEbx1)

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn00000007_EBX_x01

Bits	Description
31:0	Reserved.

CPUID_Fn00000007_ECX_x01 [**Structured Extended Feature Identifier**] **(Core::X86::Cpuid::StructExtFeatIdEcx1)**

Reset: 0000_0000h.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn00000007_ECX_x01

Bits	Description
31:0	Reserved.

CPUID_Fn00000007_EDX_x01 [**Structured Extended Feature Identifiers**] **(Core::X86::Cpuid::StructExtFeatIdEdx1)**

Reset: 0000_0000h.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn00000007_EDX_x01

Bits	Description
31:0	Reserved.

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

Read-only.

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[11:0]_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 instance of the current level type.
	Reset: SMT ? 01h : 00h.

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

Read-only.

_ccd[11:0]_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.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000B_ECX_x00

Bits	Description
31:16	Reserved.
15:8	**LevelType**. Read-only. Reset: Fixed,01h.
	ValidValues:

Value	Description
00h	Invalid
01h	Thread
02h	Processor
FFh-03h	Reserved.

7:0	**EcxVal**. Read-only. Reset: Fixed,00h. ECX input value.

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

Read-only.

_ccd[11:0]_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[11:0]_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.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000B_ECX_x01

Bits	Description
31:16	Reserved.
15:8	**LevelType**. Read-only. Reset: Fixed,02h. **ValidValues**: Value / Description: 00h Invalid; 01h Thread; 02h Processor; FFh-03h Reserved.
7:0	**EcxVal**. Read-only. Reset: Fixed,01h. ECX input value.

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

Read-only. Reset: Fixed,0000_0000h.

_ccd[11:0]_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[11:0]_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.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000B_ECX_x02

Bits	Description
31:16	Reserved.
15:8	**LevelType**. Read-only. Reset: Fixed,00h. **ValidValues**: Value / Description 00h: Invalid 01h: Thread 02h: Processor FFh-03h: Reserved.
7:0	**EcxVal**. Read-only. Reset: Fixed,02h. ECX input value.

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

Read-only.

_ccd[11:0]_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_02E7h.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000D_EAX_x00

Bits	Description
31:0	**XFeatureSupportedMask[31:0]**. Read-only. Reset: Fixed,0000_02E7h. 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. [4:3] / / Reserved. [5] / KREGS / KREGS [6] / ZMMHI / ZMMHI [7] / HIZMM / HIZMM [8] / / 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[11:0]_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
	ELSIF (XCR0[HIZMM] == 1)
	  return EBX=0000_0980h
	ELSIF (XCR0[ZMMHI] == 1)
	  return EBX=0000_0580h
	ELSIF (XCR0[KREGS] == 1)
	  return EBX=0000_0380h
	ELSIF (XCR0[AVX] == 1)
	  return EBX=0000_0340h
	ELSIF (XCR0[SSE] == 1) or (XCR0[X87] == 1)
	  return EBX=0000_0240h // legacy header + X87/SSE size
	END

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

Read-only. Reset: Fixed,0000_0988h.

_ccd[11:0]_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 + KREGS + ZMMHI + HIZMM + MPK.

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

Read-only. Reset: Fixed,0000_0000h.

_ccd[11:0]_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[11:0]_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[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000D_EBX_x01

Bits	Description
31:0	**XFeatureEnabledSizeMax**. Read-only,Volatile. Reset: XXXX_XXXXh.
	Description: Reset is EBX = 0000_0240h + ((XCR0[AVX] == 1) ? 0000_0100h : 0) + ((XCR0[KREGS] == 1) ? 0000_0040h : 0) + ((XCR0[ZMMHI] == 1) ? 0000_0200h : 0) + ((XCR0[HIZMM] == 1) ? 0000_0400h : 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[11:0]_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[11:0]_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[11:0]_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[11:0]_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[11:0]_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[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000D_EDX_x02

Bits	Description
31:0	Reserved.

CPUID_Fn0000000D_EAX_x05 [Processor Extended State Enumeration] (Core::X86::Cpuid::ProcExtStateEnumEax05)

Read-only. Reset: Fixed,0000_0040h.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000D_EAX_x05

Bits	Description
31:0	**KREGSSaveStateSize**. Read-only. Reset: Fixed,0000_0040h. KREGS save state byte size.

CPUID_Fn0000000D_EBX_x05 [Processor Extended State Enumeration] (Core::X86::Cpuid::ProcExtStateEnumEbx05)

Read-only. Reset: Fixed,0000_0340h.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000D_EBX_x05

Bits	Description
31:0	**KREGSSaveStateOffset**. Read-only. Reset: Fixed,0000_0340h. KREGS save state uncompacted byte offset.

CPUID_Fn0000000D_ECX_x05 [Processor Extended State Enumeration] (Core::X86::Cpuid::ProcExtStateEnumEcx05)

Read-only. Reset: Fixed,0000_0000h.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000D_ECX_x05

Bits	Description
31:2	Reserved.
1	**KREGSAligned**. Read-only. Reset: Fixed,0. 0=KREGS state is not automatically aligned to a 64-byte boundary on compacted saves/restores. 1=KREGS state 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_x05 [Processor Extended State Enumeration] (Core::X86::Cpuid::ProcExtStateEnumEdx05)

Read-only. Reset: Fixed,0000_0000h.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000D_EDX_x05

Bits	Description
31:0	Reserved.

CPUID_Fn0000000D_EAX_x06 [Processor Extended State Enumeration] (Core::X86::Cpuid::ProcExtStateEnumEax06)

Read-only. Reset: Fixed,0000_0200h.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000D_EAX_x06

Bits	Description
31:0	**ZMMHISaveStateSize**. Read-only. Reset: Fixed,0000_0200h. ZMMHI save state byte size.

CPUID_Fn0000000D_EBX_x06 [Processor Extended State Enumeration] (Core::X86::Cpuid::ProcExtStateEnumEbx06)

Read-only. Reset: Fixed,0000_0380h.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000D_EBX_x06

Bits	Description
31:0	**ZMMHISaveStateOffset**. Read-only. Reset: Fixed,0000_0380h. ZMMHI save state uncompacted byte offset.

CPUID_Fn0000000D_ECX_x06 [Processor Extended State Enumeration] (Core::X86::Cpuid::ProcExtStateEnumEcx06)

Read-only. Reset: Fixed,0000_0000h.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000D_ECX_x06

Bits	Description
31:2	Reserved.
1	**ZMMHIAligned**. Read-only. Reset: Fixed,0. 0=ZMMHI state is not automatically aligned to a 64-byte boundary on compacted saves/restores. 1=ZMMHI 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_x06 [Processor Extended State Enumeration] (Core::X86::Cpuid::ProcExtStateEnumEdx06)

Read-only. Reset: Fixed,0000_0000h.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000D_EDX_x06

Bits	Description
31:0	Reserved.

CPUID_Fn0000000D_EAX_x07 [Processor Extended State Enumeration] (Core::X86::Cpuid::ProcExtStateEnumEax07)

Read-only. Reset: Fixed,0000_0400h.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000D_EAX_x07

Bits	Description
31:0	**HIZMMSaveStateSize**. Read-only. Reset: Fixed,0000_0400h. HIZMM save state byte size.

CPUID_Fn0000000D_EBX_x07 [Processor Extended State Enumeration] (Core::X86::Cpuid::ProcExtStateEnumEbx07)

Read-only. Reset: Fixed,0000_0580h.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000D_EBX_x07

Bits	Description
31:0	**HIZMMSaveStateOffset**. Read-only. Reset: Fixed,0000_0580h. HIZMM save state uncompacted byte offset.

CPUID_Fn0000000D_ECX_x07 [Processor Extended State Enumeration] (Core::X86::Cpuid::ProcExtStateEnumEcx07)

Read-only. Reset: Fixed,0000_0000h.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000D_ECX_x07

Bits	Description
31:2	Reserved.
1	**HIZMMAligned**. Read-only. Reset: Fixed,0. 0=HIZMM state is not automatically aligned to a 64-byte boundary on compacted saves/restores. 1=HIZMM state 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_x07 [Processor Extended State Enumeration] (Core::X86::Cpuid::ProcExtStateEnumEdx07)

Read-only. Reset: Fixed,0000_0000h.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000D_EDX_x07

Bits	Description
31:0	Reserved.

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

Read-only. Reset: Fixed,0000_0008h.

_ccd[11:0]_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[11:0]_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[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000D_ECX_x09

Bits	Description
31:2	Reserved.
1	**XState64ByteAligned**. 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[11:0]_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[11:0]_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[11:0]_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[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000D_ECX_x0B

Bits	Description
31:2	Reserved.
1	**XState64ByteAligned**. 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[11:0]_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[11:0]_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[11:0]_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[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn0000000D_ECX_x0C

Bits	Description
31:2	Reserved.
1	**XState64ByteAligned**. 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[11:0]_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[11:0]_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[11:0]_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[11:0]_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[11:0]_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.

_ccd[11:0]_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,20. Encode counter width offset from bit 24.
	ValidValues:

Value	Description
00h	Family/Model/Stepping should be used to determine counter size.
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[11:0]_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[11:0]_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 resource.

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

Read-only. Reset: Fixed,0000_0007h.

_ccd[11:0]_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.

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[11:0]_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[11:0]_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.

_ccd[11:0]_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.

_ccd[11:0]_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.

_ccd[11:0]_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[11:0]_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.

_ccd[11:0]_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.

_ccd[11:0]_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 Class Of Service (COS) supported.

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

Read-only. Reset: Fixed,8000_0028h.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000000_EAX

Bits	Description
31:0	**LFuncExt**: **largest extended function**. Read-only. Reset: Fixed,8000_0028h. 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[11:0]_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[11:0]_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[11:0]_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[11:0]_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: 1h. 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: Xh. Processor stepping (revision) for a specific model.

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

Read-only.

_ccd[11:0]_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. **ValidValues**: Value / Description 3h-0h: Reserved. 4h: SP5 Fh-5h: Reserved.
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[11:0]_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 L3 performance counter extensions.
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.
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[11:0]_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[11:0]_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[11:0]_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[11:0]_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[11:0]_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[11:0]_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[11:0]_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[11:0]_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[11:0]_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[11:0]_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[11:0]_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[11:0]_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[11:0]_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[11:0]_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,72. 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[11:0]_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,72. 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[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000005_ECX

Bits	Description
31:24	**L1DcSize**. Read-only. Reset: Fixed,32. L1 data cache size in KB.
23:16	**L1DcAssoc**. Read-only. Reset: Fixed,08h. L1 data cache associativity.
	ValidValues:

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

Bits	Description
15:8	**L1DcLinesPerTag**. Read-only. Reset: Fixed,01h. L1 data cache lines per tag.
7:0	**L1DcLineSize**. Read-only. Reset: Fixed,64. L1 data cache line size in bytes.

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

Read-only.

This function provides first level cache characteristics for each core.

_ccd[11:0]_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: 00h – Description: Reserved. Value: 01h – Description: 1 way (direct mapped) Value: 02h – Description: 2 way Value: 03h – Description: 3 way Value: 04h – Description: 4 way Value: FEh-05h – Description: <Value> way Value: FFh – Description: Fully associative
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[11:0]_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: Fixed,5h. **ValidValues**: Value: 4h-0h – Description: Reserved. Value: 5h – Description: 6 – 7 ways Value: Fh-6h – Description: Reserved.
27:16	**L2DTlb2and4MSize**: **L2 data TLB number of entries for 2 MB and 4 MB pages**. Read-only. Reset: Fixed,3072. 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 number of 2 MB entries.
15:12	**L2ITlb2and4MAssoc**: **L2 instruction TLB associativity for 2 MB and 4 MB pages**. Read-only. Reset: Fixed,2. **ValidValues**: Value: 1h-0h – Description: Reserved. Value: 2h – Description: 2 ways Value: Fh-3h – Description: Reserved.
11:0	**L2ITlb2and4MSize**: **L2 instruction TLB number of entries for 2 MB and 4 MB pages**. Read-only. Reset: Fixed,512. 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 number of 2 MB entries.

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[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000006_EBX

Bits	Description
31:28	**L2DTlb4KAssoc**. Read-only. Reset: Fixed,6h. L2 data TLB associativity for 4 KB pages.
	ValidValues:

Value	Description
5h-0h	Reserved.
6h	8 – 15 ways
Fh-7h	Reserved.

Bits	Description
27:16	**L2DTlb4KSize**. Read-only. Reset: Fixed,3072. The 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:

Value	Description
3h-0h	Reserved.
4h	4 – 5 ways
Fh-5h	Reserved.

Bits	Description
11:0	**L2ITlb4KSize**. Read-only. Reset: Fixed,512. The 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[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000006_ECX

<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>31:16</td><td>L2Size. Read-only. Reset: Fixed,0400h. L2 cache size in KB.
ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>00FFh-0000h</td><td>Reserved.</td></tr>
<tr><td>0100h</td><td>256 KB</td></tr>
<tr><td>01FFh-0101h</td><td>Reserved.</td></tr>
<tr><td>0200h</td><td>512 KB</td></tr>
<tr><td>03FFh-0201h</td><td>Reserved.</td></tr>
<tr><td>0400h</td><td>1 MB</td></tr>
<tr><td>07FFh-0401h</td><td>Reserved.</td></tr>
<tr><td>0800h</td><td>2 MB</td></tr>
<tr><td>FFFFh-0801h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>15:12</td><td>L2Assoc. Read-only. Reset: Fixed,6. L2 cache associativity.
ValidValues:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>5h-0h</td><td>Reserved.</td></tr>
<tr><td>6h</td><td>8 – 15 ways</td></tr>
<tr><td>Fh-7h</td><td>Reserved.</td></tr>
</table></td></tr>
<tr><td>11:8</td><td>L2LinesPerTag. Read-only. Reset: Fixed,1h. L2 cache lines per tag.</td></tr>
<tr><td>7:0</td><td>L2LineSize. Read-only. Reset: Fixed,64. L2 cache line size in bytes.</td></tr>
</table>

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[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000006_EDX

Bits	Description
31:18	**L3Size**: **L3 cache size**. Read-only. Reset: XXXXh. The L3 cache size in 512 KB units. **ValidValues**: Value: 0000h — Disabled. Value: 3FFFh-0001h — (<Value> * 0.5) MB
17:16	Reserved.
15:12	**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**: Value: 8h-0h — Reserved. Value: 9h — Invalid, not reported here. Value: Fh-Ah — Reserved.
11:8	**L3LinesPerTag**. Read-only. Reset: Fixed,1h. L3 cache lines per tag.
7:0	**L3LineSize**. Read-only. Reset: Fixed,64. L3 cache line size in bytes.

CPUID_Fn80000007_EAX [Reserved] (Core::X86::Cpuid::ProcFeedbackCap)

Read-only. Reset: Fixed,0000_0000h.

_ccd[11:0]_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[11:0]_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.
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.

CPUID_Fn80000007_ECX [Advanced Power Management Information] (Core::X86::Cpuid::ApmInfoEcx)

Read-only. Reset: Fixed,0000_0000h.

_ccd[11:0]_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[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000007_EDX

Bits	Description
31:16	Reserved.
15	**FastCPPC**. Read-only. Reset: X. When set, indicates that Fast CPPC is supported.
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_3934h.

This provides information about the maximum physical and linear address width supported by the processor.

_ccd[11:0]_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**: Value: 00h — The maximum guest physical address size defined by PhysAddrSize. Value: FFh-01h — The maximum guest physical address size defined by GuestPhysAddrSize.
15:8	**LinAddrSize**. Read-only. Reset: Fixed,39h. Maximum linear byte address size in bits.
7:0	**PhysAddrSize**. Read-only. Reset: Fixed,34h. Maximum physical byte address size in bits.

CPUID_Fn80000008_EBX [Extended Feature Extensions ID EBX] (Core::X86::Cpuid::FeatureExtIdEbx)

Read-only.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000008_EBX

Bits	Description
31	**BranchSample**. Read-only. Reset: Fixed,0. 1=Branch sampling feature supported. Branch sampling has been replaced with LBRv2.
30	**IBPB_RET**. Read-only. Reset: Fixed,1. When set, the processor clears the return address predictor on Core::X86::Msr::PRED_CMD[IBPB].
29	**BTC_NO**. Read-only. Reset: Fixed,1. Processor is not vulnerable to Branch Type Confusion.
28	**PSFD**. Read-only. Reset: Fixed,1. Predictive Store Forward Disable. See Core::X86::Msr::SPEC_CTRL[PSFD].
27	**CPPC**. Read-only. Reset: 1. 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 the nested translations for guest translations. 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: 0. Memory commit instruction support.
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[11:0]_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[11:0]_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.

_ccd[11:0]_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.

This provides SVM revision and feature information.

_ccd[11:0]_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.

This provides SVM feature information.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn8000000A_EDX

Bits	Description
31:29	Reserved.
28	**VmcbAddrChkChg**. Read-only. Reset: Fixed,1. VMCB address check change
27	**ExtLvtOffsetFaultChg**. Read-only. Reset: Fixed,1. 1=Read/Write fault behavior for the extended LVT offsets (APIC addresses 0x500-0x530) changed to Read Allowed, Write #MVEXIT (trap).
26	**IbsVirt**. Read-only. Reset: Fixed,1. IBS Virtualization
25	**NmiVirt**. Read-only. Reset: Fixed,1. Guest NMI Virtualization
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	Reserved.
21	**AllowNonWriteAbleGPT**. Read-only. Reset: Fixed,1. 1=Indicates support for Non-Writeable Guest Pages for NPT.
20	**GuestSpecCtrl**. Read-only. Reset: Fixed,1. 1=Indicates support for Guest Spec_ctl.
19	**SupervisorShadowStack**. Read-only. Reset: Fixed,1.
18	**X2AVIC**: **virtualized X2APIC**. Read-only. Reset: 1. 1=Virtualized X2APIC is supported.
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: 1. 1=Support indicated for SVM mode virtualized interrupt controller; Indicates support for Core::X86::Msr::AvicDoorbell.
12	**PauseFilterThreshold**. Read-only. Reset: Fixed,1.
11	Reserved.
10	**PauseFilter**. Read-only. Reset: Fixed,1. Pause intercept filter.
9:8	Reserved.
7	**DecodeAssists**. Read-only. Reset: Fixed,1.
6	**FlushByAsid**. Read-only. Reset: Fixed,1.
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. MSR based TSC rate control.
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 1GB pages.

_ccd[11:0]_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,72. 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[11:0]_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[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn8000001A_EAX

Bits	Description
31:3	Reserved.
2	**FP256**. Read-only. Reset: Fixed,1. When set, the internal FP/SIMD execution datapath is 256-bits wide.
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. When set, the internal FP/SIMD execution datapath is 128-bits wide.

CPUID_Fn8000001B_EAX [Instruction Based Sampling Identifiers] (Core::X86::Cpuid::IbsIdEax)

Read-only. Reset: Fixed,0000_0BFFh.

This function returns IBS feature information.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn8000001B_EAX

Bits	Description
31:12	Reserved.
11	**Zen4IbsExtensions**. Read-only. Reset: Fixed,1. Fetch and Op IBS support IBS extensions added with Zen4: Core::X86::Msr::IBS_FETCH_CTL[IbsFetchL3Miss], Core::X86::Msr::IBS_FETCH_CTL[IbsFetchOcMiss], Core::X86::Msr::IBS_FETCH_CTL[IbsL3MissOnly] and Core::X86::Msr::IBS_OP_DATA2 DataSrc extension.
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	**OpFuse**: **fused branch op indication supported**. Read-only. Reset: Fixed,1. Indicates support for Core::X86::Msr::IBS_OP_DATA[IbsOpFuse].
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: Fixed,1. 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)

Core::X86::Cpuid::CachePropEax0 reports topology information for the DC.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn8000001D_EAX_x00

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. 1=Cache is fully associative.
8	**SelfInitialization**: **cache is self-initializing**. Read-only. Reset: Fixed,1. 1=Cache is self initializing; cache does not need software initialization.
7:5	**CacheLevel**: **cache level**. Read-only. Reset: Fixed,1h. Identifies the cache level.
	ValidValues:

Value	Description
0h	Reserved.
1h	Level 1
2h	Level 2
3h	Level 3
7h-4h	Reserved.

4:0	**CacheType**: **cache type**. Read-only. Reset: Fixed,01h. Identifies the type of cache.
	ValidValues:

Value	Description
00h	Null; no more caches.
01h	Data cache.
02h	Instruction cache.
03h	Unified cache.
1Fh-04h	Reserved.

CPUID_Fn8000001D_EAX_x01 [Cache Properties (IC)] (Core::X86::Cpuid::CachePropEax1)

Read-only.

Core::X86::Cpuid::CachePropEax1 reports topology information for the IC. See Core::X86::Cpuid::CachePropEax0.

_ccd[11:0]_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.

Core::X86::Cpuid::CachePropEax2 reports topology information for the L2. See Core::X86::Cpuid::CachePropEax0.

_ccd[11:0]_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.

Core::X86::Cpuid::CachePropEax3 reports topology information for the L3.

_ccd[11:0]_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.

Core::X86::Cpuid::CachePropEax4 reports done/null. See Core::X86::Cpuid::CachePropEax0.

_ccd[11:0]_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.

Core::X86::Cpuid::CachePropEbx0 reports topology information for the DC. See Core::X86::Cpuid::CachePropEax0.

_ccd[11:0]_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.

Core::X86::Cpuid::CachePropEbx1 reports topology information for the IC. See Core::X86::Cpuid::CachePropEax0.

_ccd[11:0]_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.

Core::X86::Cpuid::CachePropEbx2 reports topology information for the L2. See Core::X86::Cpuid::CachePropEax0.

_ccd[11:0]_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.

Core::X86::Cpuid::CachePropEbx3 reports topology information for the L3. See Core::X86::Cpuid::CachePropEax0.

_ccd[11:0]_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.

Core::X86::Cpuid::CachePropEax4 reports done/null. See Core::X86::Cpuid::CachePropEax0.

_ccd[11:0]_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.

Core::X86::Cpuid::CachePropEcx0 reports topology information for the DC. See Core::X86::Cpuid::CachePropEax0.

_ccd[11:0]_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.

Core::X86::Cpuid::CachePropEcx1 reports topology information for the IC. See Core::X86::Cpuid::CachePropEax0.

_ccd[11:0]_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_07FFh.

Core::X86::Cpuid::CachePropEcx2 reports topology information for the L2. See Core::X86::Cpuid::CachePropEax0.

_ccd[11:0]_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_07FFh. See Core::X86::Cpuid::CachePropEcx0[CacheNumSets].

CPUID_Fn8000001D_ECX_x03 [Cache Properties (L3)] (Core::X86::Cpuid::CachePropEcx3)

Read-only.

Core::X86::Cpuid::CachePropEcx3 reports topology information for the L3.

_ccd[11:0]_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.

Core::X86::Cpuid::CachePropEax3 reports done/null. See Core::X86::Cpuid::CachePropEax0.

_ccd[11:0]_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.

Core::X86::Cpuid::CachePropEdx0 reports topology information for the DC. See Core::X86::Cpuid::CachePropEax0.

_ccd[11:0]_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.

Core::X86::Cpuid::CachePropEdx1 reports topology information for the IC. See Core::X86::Cpuid::CachePropEax0.

_ccd[11:0]_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.

Core::X86::Cpuid::CachePropEdx2 reports topology information for the L2. See Core::X86::Cpuid::CachePropEax0.

_ccd[11:0]_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.

Core::X86::Cpuid::CachePropEdx3 reports reports topology information for the L3. See Core::X86::Cpuid::CachePropEax0.

_ccd[11:0]_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.

Core::X86::Cpuid::CachePropEax3 reports done/null. See Core::X86::Cpuid::CachePropEax0.

_ccd[11:0]_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.

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[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn8000001E_EAX

Bits	Description
31:0	**ExtendedApicId**: **extended APIC ID**. Read-only. Reserved.
	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.

See Core::X86::Cpuid::ExtApicId.

_ccd[11:0]_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 unique per-socket logical core unit ID.

CPUID_Fn8000001E_ECX [Node Identifiers] (Core::X86::Cpuid::NodeId)

Read-only.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn8000001E_ECX

Bits	Description
31:11	Reserved.
10:8	**NodesPerProcessor**. Read-only. Reset: XXXb. Nodes per processor.
	ValidValues:

Value	Description
0h	1 node per processor.
7h-1h	Reserved.

7:0	**NodeId**. Read-only. Reset: Fixed,XXh. Node ID.

CPUID_Fn8000001F_EAX [AMD Secure Encryption EAX] (Core::X86::Cpuid::SecureEncryptionEax)

Read-only.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn8000001F_EAX

Bits	Description
31:26	Reserved.
25	**SmtProt**. Read-only. Reset: Fixed,1. SMT Protection for SEV-SNP is supported.
24	**VmsaRegProt**. Read-only. Reset: Fixed,1. VMSA Register Protection is supported.
23:20	Reserved.
19	**IbsVirtualization**. Read-only. Reset: Fixed,1. IBS state virtualization is supported for SEV-ES guests via opt-in control in SEV_FEATURES.
18	**VirtualTomMsr**. Read-only. Reset: Fixed,1. Virtual TOM MSR is supported.
17	**VmgexitParameter**. Read-only. Reset: Fixed,1. VmgexitParameter is supported in SEV_FEATURES.
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 64-Bit Hypervisor.
10	**CoherencyEnforced**. Read-only. Reset: Fixed,1. Hardware enforces cache coherency.
9	**TscAuxVirtualization**. Read-only. Reset: Fixed,1. Hardware virtualizes TSC_AUX.
8	**SecureTsc**. Read-only. Reset: Fixed,1. Support for Secure TSC.
7	**VmplSSS**. Read-only. Reset: 1. The Supervisor Shadow Stack bit is supported in the VMPL permission set.
6	**RMPQUERY**. Read-only. Reset: 1. The RMPQUERY instruction is supported.
5	**VMPL**. Read-only. Reset: Fixed,1. VMPL VM Permission Levels supported.
4	**SNP**. Read-only. Reset: Fixed,X. 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 support**. Read-only. Reset: Fixed,0. VMPAGE_FLUSH is no longer supported.
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[11:0]_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. 06h: Physical Address width is reduced by six. 3Fh-07h: 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[11:0]_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[11:0]_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 Extended Feature Identifiers] (Core::X86::Cpuid::AmdQosExtEax0)

Read-only. Reset: 0000_0000h.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000020_EAX_x00

Bits	Description
31:0	Reserved.

CPUID_Fn80000020_EBX_x00 [Platform QoS Extended Feature Identifiers] (Core::X86::Cpuid::AmdQosExtEbx0)

Read-only. Reset: 0000_001Eh.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000020_EBX_x00

Bits	Description
31:5	Reserved.
4	**L3RR**. Read-only. Reset: 1. AMD L3 Range Reservation.
3	**EVT_CFG**. Read-only. Reset: 1. AMD Bandwidth Monitoring Event Configuration.
2	**SMBE**. Read-only. Reset: 1. Slow Memory Bandwidth Enforcement. See Core::X86::Cpuid::PqeBandwidthEax2 - Core::X86::Cpuid::PqeBandwidthEdx2.
1	**MBE**. Read-only. Reset: 1. Memory Bandwidth Enforcement.
0	Reserved.

CPUID_Fn80000020_ECX_x00 [Platform QoS Extended Feature Identifiers] (Core::X86::Cpuid::AmdQosExtEcx0)

Read-only. Reset: 0000_0000h.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000020_ECX_x00

Bits	Description
31:0	Reserved.

CPUID_Fn80000020_EDX_x00 [Platform QoS Extended Feature Identifiers] (Core::X86::Cpuid::AmdQosExtEdx0)

Read-only. Reset: 0000_0000h.

_ccd[11:0]_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[11:0]_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[11:0]_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[11:0]_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[11:0]_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_Fn80000020_EAX_x02 [Enforcement for Slow Memory Bandwidth] (Core::X86::Cpuid::PqeBandwidthEax2)

Read-only. Reset: 0000_000Bh.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000020_EAX_x02

Bits	Description
31:0	**BW_LEN**. Read-only. Reset: 0000_000Bh. Slow Memory Bandwidth Enforcement Bit Range Length.

CPUID_Fn80000020_EBX_x02 [Enforcement for Slow Memory Bandwidth] (Core::X86::Cpuid::PqeBandwidthEbx2)

Read-only. Reset: 0000_0000h.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000020_EBX_x02

Bits	Description
31:0	Reserved.

CPUID_Fn80000020_ECX_x02 [Enforcement for Slow Memory Bandwidth] (Core::X86::Cpuid::PqeBandwidthEcx2)

Read-only. Reset: 0000_0000h.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000020_ECX_x02

Bits	Description
31:0	Reserved.

CPUID_Fn80000020_EDX_x02 [Enforcement for Slow Memory Bandwidth] (Core::X86::Cpuid::PqeBandwidthEdx2)

Read-only. Reset: 0000_000Fh.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000020_EDX_x02

Bits	Description
31:0	**COS_MAX**. Read-only. Reset: 0000_000Fh. Maximum Class Of Service (COS) number for Memory Bandwidth Enforcement.

CPUID_Fn80000020_EBX_x03 [Platform QoS Monitoring Bandwidth Event Configuration] (Core::X86::Cpuid::PqeBandwidthEvtCfgEbx3)

Read-only. Reset: 0000_0002h.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000020_EBX_x03

Bits	Description
31:8	Reserved.
7:0	**EVT_NUM**. Read-only. Reset: 02h. Reports the number of bandwidth events that can be configured.

CPUID_Fn80000020_ECX_x03 [Platform QoS Monitoring Bandwidth Event Configuration] (Core::X86::Cpuid::PqeBandwidthEvtCfgEcx3)

Read-only. Reset: 0000_007Fh.

Identifies the bandwidth sources that can be tracked.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000020_ECX_x03

Bits	Description
31:7	Reserved.
6	**L3CacheBwVicMon**. Read-only. Reset: 1. Dirty victims to all types of memory.
5	**L3CacheRmtSlowBwFillMon**. Read-only. Reset: 1. Reads from remote memory the system identifies as slow memory.
4	**L3CacheLclSlowBwFillMon**. Read-only. Reset: 1. Reads from local memory the system identifies as slow memory.
3	**L3CacheRmtBwNtWrMon**. Read-only. Reset: 1. Non-temporal writes to remote memory.
2	**L3CacheLclBwNtWrMon**. Read-only. Reset: 1. Non-temporal writes to local memory.
1	**L3CacheRmtBwFillMon**. Read-only. Reset: 1. Reads from remote memory.
0	**L3CacheLclBwFillMon**. Read-only. Reset: 1. Reads from local memory.

CPUID_Fn80000021_EAX [Extended Feature 2 EAX] (Core::X86::Cpuid::FeatureExt2Eax)

Read-only.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000021_EAX

Bits	Description
31:19	Reserved.
18	**EPSF**. Read-only. Reset: 1. Enhanced Predictive Store Forwarding supported.
17	**GpOnUserCpuid**. Read-only. Reset: 1. Indicates support for #GP when executing CPUID at CPL > 0.
16:14	Reserved.
13	**PrefetchCtlMsr**. Read-only. Reset: 1. 1=Prefetch control MSR supported. See Core::X86::Msr::PrefetchControl.
12	Reserved.
11	**FSRC**. Read-only. Reset: Fixed,1. Fast Short Repe Cmpsb supported.
10	**FSRS**. Read-only. Reset: Fixed,1. Fast Short Rep Stosb supported.
9	**NoSmmCtlMSR**. Read-only. Reset: Fixed,1. Indicates that MSRC001_0116 (SMM_CTL) is not present.
8	**AutomaticIBRS**. Read-only. Reset: Fixed,1. Indicates that Core::X86::Msr::EFER[AutomaticIBRSEn] can be set to automatically toggle IBRS protection when CPL changes.
7	**UpperAddressIgnore**. Read-only. Reset: Fixed,1. Indicates that Core::X86::Msr::EFER[UAIE] bit controls canonical check for upper address bits for certain types of accesses.
6	**NullSelectorClearsBase**. Read-only. Reset: 1. 1=Null Selector Clears Base. When this bit is set, a null segment load clears the segment 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	**FsGsKernelGsBaseNonSerializing**. Read-only. Reset: Fixed,1. WRMSR to Core::X86::Msr::FS_BASE, Core::X86::Msr::GS_BASE, and Core::X86::Msr::KernelGSbase are non-serializing.
0	**NoNestedDataBp**. Read-only. Reset: Fixed,1. New data-breakpoints are ignored while switching to data-breakpoint handler.

CPUID_Fn80000021_EBX [Extended Feature 2 EBX] (Core::X86::Cpuid::FeatureExt2Ebx)

Read-only.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000021_EBX

Bits	Description
31:12	Reserved.
11:0	**MicrocodePatchSize**. Read-only. Reset: XXXh. Reports the size of the Microcode patch in 16-byte multiples. If 0, the size of the patch is at most 5568 (15C0h) bytes.

CPUID_Fn80000022_EAX [Extended Performance Monitoring and Debug EAX] (Core::X86::Cpuid::ExtPerfMonAndDbgEax)

Read-only. Reset: Fixed,0000_0003h.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000022_EAX

Bits	Description
31:3	Reserved.
2	**LbrAndPmcFreeze**. Read-only. Reset: Fixed,0. 1=PMC and LBR freeze is supported in Core::X86::Msr::DBG_CTL_MSR.
1	**LbrExtV2**. Read-only. Reset: Fixed,1. 1=LBR extension Version 2 supported.
0	**PerfMonV2**. Read-only. Reset: Fixed,1. 1=Performance Monitoring Version 2 supported.

CPUID_Fn80000022_EBX [Extended Performance Monitoring and Debug EBX] (Core::X86::Cpuid::ExtPerfMonAndDbgEbx)

Read-only.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000022_EBX

Bits	Description
31:22	Reserved.
21:16	**NumPerfCtrUmc**. Read-only. Reset: Fixed,XXXXXXb. Number of available UMC PMCs.
15:10	**NumPerfCtrNB**. Read-only. Reset: Fixed,10h. Number of available Data Fabric (Northbridge) Performance Monitor Counters.
9:4	**LbrV2StackSz**. Read-only. Reset: Fixed,10h. Number of available LBR stack entries.
3:0	**NumPerfCtrCore**. Read-only. Reset: Fixed,6h. Number of Core Performance Counters.

CPUID_Fn80000022_ECX [Extended Performance Monitoring and Debug ECX] (Core::X86::Cpuid::ExtPerfMonAndDbgEcx)

Read-only.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000022_ECX

Bits	Description
31:0	**ActiveUmcMask**. Read-only. Reset: Fixed,XXXX_XXXXh. Bitmask representing active UMCs. Calculate the number of PMCs per UMC as Core::X86::Cpuid::ExtPerfMonAndDbgEbx[NumPerfCtrUmc] / POPCNT(CPUID_8000_0022_ECX[31:0]).

CPUID_Fn80000023_EAX [AMD Secure Multi-Key Encryption EAX] (Core::X86::Cpuid::SecureMultiKeyEncryptionEax)

Read-only. Reset: Fixed,0000_0001h.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000023_EAX

Bits	Description
31:1	Reserved.
0	**MemHmkSupport**. Read-only. Reset: Fixed,1. 1=MEM-HMK mode is supported.

CPUID_Fn80000023_EBX [AMD Secure Multi-Key Encryption EBX] (Core::X86::Cpuid::SecureMultiKeyEncryptionEbx)

Read-only.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000023_EBX

Bits	Description
31:16	Reserved.
15:0	**MaxMemHmkEncrKeyID**. Read-only. Reset: Fixed,63. Total number of available encryption keys in MEM-HMK mode.

CPUID_Fn80000023_ECX [AMD Secure Multi-Key Encryption ECX] (Core::X86::Cpuid::SecureMultiKeyEncryptionEcx)

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000023_ECX

Bits	Description
31:0	Reserved.

CPUID_Fn80000023_EDX [AMD Secure Multi-Key Encryption EDX] (Core::X86::Cpuid::SecureMultiKeyEncryptionEdx)

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000023_EDX

Bits	Description
31:0	Reserved.

CPUID_Fn80000026_EAX_x0[0...3] [Extended CPU Topology] (Core::X86::Cpuid::ExCpuTopologyEax)

Read-only.

CPUID Fn8000_0026_E[D,C,B,A]X_x[3:0] specifies the hierarchy of logical cores from the SMT level through the processor socket level. Software reads CPUID Fn8000_0026_E[C,B,A]X for ascending values of ECX until (CPUID Fn8000_0026_EBX[LogProcAtThisLevel] == 0).
Note: While CPUID Fn8000_0026 is a preferred superset to CPUID_Fn0000000B, CPUID_Fn0000000B information is valid for software for the supported levels on AMD.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n0; CPUID_Fn80000026_EAX_x00
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n1; CPUID_Fn80000026_EAX_x01
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n2; CPUID_Fn80000026_EAX_x02
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n3; CPUID_Fn80000026_EAX_x03

Bits	Description
31	**AsymmetricCores**. Read-only. Reset: Fixed,X. When set, not all cores in the system report the same value in Core::X86::Cpuid::ExtCpuTopologyEbx[LogProcThisLevel] for Core::X86::Cpuid::ExtCpuTopologyEcx[LevelType]=0x1. When cleared, all cores in the system report the same value in Core::X86::Cpuid::ExtCpuTopologyEbx[LogProcThisLevel] for Core::X86::Cpuid::ExtCpuTopologyEcx[LevelType]=0x1.
30	**HeterogeneousCoreTopology**. Read-only. Reset: Fixed,0. When set, not all instances at the current hierarchy level have the same Core Type topology. The core type is reported in Core::X86::Cpuid::ExtCpuTopologyEbx[CoreType].
29	**EfficiencyRankingAvailable**. Read-only. Reset: Fixed,0. When set, Core::X86::Cpuid::ExtCpuTopologyEbx[ProcessorPowerEfficiencyRanking] is valid and varies between individual cores.
28:5	Reserved.
4:0	**CoreMaskWidth**. Read-only. Reset: Fixed,XXh. Number of bits to shift Core::X86::Cpuid::ExtCpuTopologyEdx[ExtendedLocalApicId] right to get unique topology ID of the next instance of the current level type. All logical processors with the same next level ID share current level.

CPUID_Fn80000026_EBX_x0[0...3] [Extended CPU Topology] (Core::X86::Cpuid::ExtCpuTopologyEbx)

Read-only.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n0; CPUID_Fn80000026_EBX_x00
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n1; CPUID_Fn80000026_EBX_x01
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n2; CPUID_Fn80000026_EBX_x02
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n3; CPUID_Fn80000026_EBX_x03

Bits	Description
31:28	**CoreType**. Read-only. Reset: Fixed,Xh. Defines per-core architectural feature differentiation (microarchitectural resources, etc.) that may lead to a different performance, core clock boost, and power characteristic. Only valid while LevelType=Core. **ValidValues**: Value: 0h — Performance Core. Value: 1h — Efficiency Core. Value: Fh-2h — Reserved.
27:24	**NativeModelId**. Read-only. Reset: Fixed,0h. Context sensitive to CPUID_Fn00000001_EAX and CPUID_Fn80000001_EAX [Family, Model, Stepping Identifiers] (Core::X86::Cpuid::FamModStep). Only valid while Level Type=Core. This value is used by software to further differentiate implementation specific software visible properties between the cores. **ValidValues**: Value: 0h — Zen4 Core. Value: Fh-1h — Reserved.
23:16	**ProcessorPowerEfficiencyRanking**. Read-only. Reset: Fixed,XXh. Identifies a static efficiency ranking between each of the cores of a specific CoreType, where a core with a lower value has intrinsically better power, but potentially lower performance potential vs cores with a higher value.
15:0	**LogProcThisLevel**. Read-only. Reset: Fixed,XXXXh. Number of logical processors at this level type.

CPUID_Fn80000026_ECX_x0[0...3] [Extended CPU Topology] (Core::X86::Cpuid::ExtCpuTopologyEcx)

Read-only.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n0; CPUID_Fn80000026_ECX_x00
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n1; CPUID_Fn80000026_ECX_x01
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n2; CPUID_Fn80000026_ECX_x02
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n3; CPUID_Fn80000026_ECX_x03

Bits	Description
31:16	Reserved.
15:8	**LevelType**. Read-only. Reset: Fixed,XXh. **ValidValues**: Value: 00h — Reserved. Value: 01h — Core Value: 02h — Complex Value: 03h — CCD (Die) Value: 04h — Socket Value: FFh-05h — Reserved.
7:0	**EcxVal**. Read-only. Reset: Fixed,XXh. ECX input value.

CPUID_Fn80000026_EDX [Extended CPU Topology] (Core::X86::Cpuid::ExtCpuTopologyEdx)

Read-only.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; CPUID_Fn80000026_EDX

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

2.1.15 MSR Registers

2.1.15.1 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[11:0]_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[11:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_001B

Bits	Description
63:52	Reserved.
51:12	**ApicBar[51:12]**: **APIC base address register**. Read-write. Reset: 00_000F_EE00h. Specifies the base address, physical address [51:12], for the APICXX register set in xAPIC mode. See 2.1.13.2.1.2 [APIC Register Space].
11	**ApicEn**: **APIC enable**. Read-write. Reset: 0. 0=Disable Local APIC. 1=Local APIC is enabled in xAPIC mode. See 2.1.13.2.1.2 [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. Clearing this bit after it has been set requires ApicEn to be cleared as well.
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[11:0]_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[11:0]_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)**

Write-only,Error-on-read. Reset: 0000_0000_0000_0000h.

_ccd[11:0]_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[11:0]_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[11:0]_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[11:0]_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[11:0]_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_010B [Flush Command] (Core::X86::Msr::FLUSH_CMD)

Writes to this register do not execute until all prior instructions finished execution and have completed locally. Later instructions do not execute until the Write completes.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_010B

Bits	Description
63:1	Reserved.
0	**L1D_FLUSH**. Write-only,Volatile. Reset: 0. When written to 1, performs a write-back and invalidate of the L1 data cache.

MSR0000_0174 [SYSENTER CS] (Core::X86::Msr::SYSENTER_CS)

Read-write. Reset: 0000_0000_0000_0000h.

_ccd[11:0]_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[11:0]_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[11:0]_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[11:0]_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[11:0]_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[11:0]_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[11:0]_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. Performance Monitor Pins are not supported on this processor.
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[11:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_01DB

Bits	Description
63:58	Reserved.
57:0	**LastBranchFromIP**. Read,Error-on-write,Volatile. Reset: 000_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[11:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_01DC

Bits	Description
63:58	Reserved.
57:0	**LastBranchToIP**. Read,Error-on-write,Volatile. Reset: 000_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[11:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_01DD

Bits	Description
63:58	Reserved.
57:0	**LastIntFromIP**. Read,Error-on-write,Volatile. Reset: 000_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[11:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_01DE

Bits	Description
63:58	Reserved.
57:0	**LastIntToIP**. Read,Error-on-write,Volatile. Reset: 000_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 will result 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[51:12] & PhyMask[51:12] == PhyBase[51:12] & PhyMask[51:12].

For example, if the variable MTRR spans 256 KB and starts at the 1 MB address the PhyBase would be set to 00_0010_0000h and the PhyMask to FF_FFFC_0000h (with zeros filling in for bits[11:0]). This results in a range from 00_0010_0000h to 00_0013_FFFFh.

_ccd[11:0]_lthree0_core[7:0]_n0; MSR0000_0200
_ccd[11:0]_lthree0_core[7:0]_n1; MSR0000_0202
_ccd[11:0]_lthree0_core[7:0]_n2; MSR0000_0204
_ccd[11:0]_lthree0_core[7:0]_n3; MSR0000_0206
_ccd[11:0]_lthree0_core[7:0]_n4; MSR0000_0208
_ccd[11:0]_lthree0_core[7:0]_n5; MSR0000_020A
_ccd[11:0]_lthree0_core[7:0]_n6; MSR0000_020C
_ccd[11:0]_lthree0_core[7:0]_n7; MSR0000_020E

Bits	Description
63:52	Reserved.
51:12	**PhyBase**. Read-write. Reset: XX_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[11:0]_lthree0_core[7:0]_n0; MSR0000_0201
_ccd[11:0]_lthree0_core[7:0]_n1; MSR0000_0203
_ccd[11:0]_lthree0_core[7:0]_n2; MSR0000_0205
_ccd[11:0]_lthree0_core[7:0]_n3; MSR0000_0207
_ccd[11:0]_lthree0_core[7:0]_n4; MSR0000_0209
_ccd[11:0]_lthree0_core[7:0]_n5; MSR0000_020B
_ccd[11:0]_lthree0_core[7:0]_n6; MSR0000_020D
_ccd[11:0]_lthree0_core[7:0]_n7; MSR0000_020F

Bits	Description
63:52	Reserved.
51:12	**PhyMask**. Read-write. Reset: XX_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. If Core::X86::Msr::SYS_CFG[SecureNestedPagingEn] is set to 1, this register will be read-only and attempts to write to this register will result in #GP(0).

_ccd[11:0]_lthree0_core[7:0]_nSIZE64K; MSR0000_0250

Bits	Description
63:61	Reserved.
60	**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.
	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_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.
	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_64K_70000**: **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.

55:53	Reserved.
52	**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.
	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_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.
	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_64K_60000**: **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.

47:45	Reserved.
44	**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.

<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>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></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_64K_50000: 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_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></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_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></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_64K_40000: 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_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></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_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></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_64K_30000: memory type. Read-write. Reset: XXXb.</td></tr>
</table>

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.

23:21 Reserved.

20 **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.

AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.

19 **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.

AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.

18:16 **MemType_64K_20000**: **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.

15:13 Reserved.

12 **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.

AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.

11 **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.

AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.

10:8 **MemType_64K_10000**: **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.

7:5 Reserved.

<table>
<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. If Core::X86::Msr::SYS_CFG[SecureNestedPagingEn] is set to 1, this register will be read-only and attempts to write to this register will result in #GP(0).

_ccd[11:0]_lthree0_core[7:0]_nSIZE16K0; MSR0000_0258

Bits	Description
63:61	Reserved.
60	**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.
	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_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.
	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_16K_9C000**: **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.

55:53	Reserved.
52	**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.
	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_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.
	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_16K_98000**: **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.

47:45	Reserved.
44	**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.

<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>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></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_94000: 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_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></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_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></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_90000: 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_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></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_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></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_8C000: memory type. Read-write. Reset: XXXb.</td></tr>
</table>

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.

23:21 Reserved.

20 **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.

AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.

19 **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.

AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.

18:16 **MemType_16K_88000**: **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.

15:13 Reserved.

12 **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.

AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.

11 **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.

AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.

10:8 **MemType_16K_84000**: **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.

7:5 Reserved.

<table>
<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. If Core::X86::Msr::SYS_CFG[SecureNestedPagingEn] is set to 1, this register will be read-only and attempts to write to this register will result in #GP(0).

_ccd[11:0]_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.
	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_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.
	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_16K_BC000**: **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.

55:53	Reserved.
52	**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.
	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_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.
	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_16K_B8000**: **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.

47:45	Reserved.
44	**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.

<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>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>
<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>
</table>

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.

23:21 Reserved.

20 **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.

AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.

19 **WrDram_16K_A8000**: **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.

18:16 **MemType_16K_A8000**: **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.

15:13 Reserved.

12 **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.

AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.

11 **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.

AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.

10:8 **MemType_16K_A4000**: **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.

7:5 Reserved.

<table>
<tr><td>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></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_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></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_16K_A0000: memory type. Read-write. Reset: XXXb. Address range from A0000h to A3FFFh.</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_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. If Core::X86::Msr::SYS_CFG[SecureNestedPagingEn] is set to 1, this register will be read-only and attempts to write to this register will result in #GP(0).

_ccd[11:0]_lthree0_core[7:0]_nSIZE4K0; MSR0000_0268

Bits	Description
63:61	Reserved.
60	**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.
	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_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.
	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_C7000**: **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.

55:53	Reserved.
52	**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.
	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_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.
	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_C6000**: **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.

47:45	Reserved.
44	**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.

<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>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></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_C5000: 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_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></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_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></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_C4000: 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_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></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_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></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_C3000: memory type. Read-write. Reset: XXXb.</td></tr>
</table>

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.

23:21 Reserved.

20 **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.

AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.

19 **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.

AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.

18:16 **MemType_4K_C2000**: **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.

15:13 Reserved.

12 **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.

AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.

11 **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.

AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.

10:8 **MemType_4K_C1000**: **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.

7:5 Reserved.

<table>
<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. 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. If Core::X86::Msr::SYS_CFG[SecureNestedPagingEn] is set to 1, this register will be read-only and attempts to write to this register will result in #GP(0).

_ccd[11:0]_lthree0_core[7:0]_nSIZE4K1; MSR0000_0269

Bits	Description
63:61	Reserved.
60	**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.
	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_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.
	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_CF000**: **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.

55:53	Reserved.
52	**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.
	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_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.
	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_CE000**: **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.

47:45	Reserved.
44	**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.

<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>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></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_CD000: 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_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></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_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></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_CC000: 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_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></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_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></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_CB000: memory type. Read-write. Reset: XXXb.</td></tr>
</table>

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.

23:21 Reserved.

20 **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.

AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.

19 **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.

AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.

18:16 **MemType_4K_CA000**: **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.

15:13 Reserved.

12 **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.

AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.

11 **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.

AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.

10:8 **MemType_4K_C9000**: **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.

7:5 Reserved.

<table>
<tr><td>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></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_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></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_C8000: memory type. Read-write. Reset: XXXb. Address range from C8000 to C8FFF.</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_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. If Core::X86::Msr::SYS_CFG[SecureNestedPagingEn] is set to 1, this register will be read-only and attempts to write to this register will result in #GP(0).

_ccd[11:0]_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.
	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.

55:53	Reserved.
52	**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.
	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_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.
	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_D6000**: **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.

47:45	Reserved.
44	**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.

<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>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></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_D5000: 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_D4000: 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_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></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_D4000: 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_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></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_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></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_D3000: memory type. Read-write. Reset: XXXb.</td></tr>
</table>

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.

23:21 Reserved.

20 **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.

AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.

19 **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.

AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.

18:16 **MemType_4K_D2000**: **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.

15:13 Reserved.

12 **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.

AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.

11 **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.

AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.

10:8 **MemType_4K_D1000**: **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.

7:5 Reserved.

<table>
<tr><td>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></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_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></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_D0000: memory type. Read-write. Reset: XXXb. Address range from D0000h to D0FFFh.</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_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 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. If Core::X86::Msr::SYS_CFG[SecureNestedPagingEn] is set to 1, this register will be read-only and attempts to write to this register will result in #GP(0).

_ccd[11:0]_lthree0_core[7:0]_nSIZE4K3; MSR0000_026B

Bits	Description
63:61	Reserved.
60	**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.
	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_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.
	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_DF000**: **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.

55:53	Reserved.
52	**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.
	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_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.
	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_DE000**: **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.

47:45	Reserved.
44	**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.

<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">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>
<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_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>26:24</td><td>MemType_4K_DB000: memory type. Read-write. Reset: XXXb.</td></tr>
</table>

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.

23:21 Reserved.

20 **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.

AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.

19 **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.

AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.

18:16 **MemType_4K_DA000**: **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.

15:13 Reserved.

12 **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.

AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.

11 **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.

AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.

10:8 **MemType_4K_D9000**: **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.

7:5 Reserved.

<table>
<tr><td>4</td><td>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></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_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></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_D8000: memory type. Read-write. Reset: XXXb. Address range from D8000h to D8FFFh.</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_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. If Core::X86::Msr::SYS_CFG[SecureNestedPagingEn] is set to 1, this register will be read-only and attempts to write to this register will result in #GP(0).

_ccd[11:0]_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.

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
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.

47:45	Reserved.
44	**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.

<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>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: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</td></tr>
<tr><td>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></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_E3000: memory type. Read-write. Reset: XXXb.</td></tr>
</table>

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.

23:21 Reserved.

20 **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.

AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.

19 **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.

AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.

18:16 **MemType_4K_E2000**: **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.

15:13 Reserved.

12 **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.

AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.

11 **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.

AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.

10:8 **MemType_4K_E1000**: **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.

7:5 Reserved.

<table>
<tr><td>4</td><td>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.</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_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.</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_E0000: memory type. Read-write. Reset: XXXb. Address range from E0000h to E0FFFh.</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_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. If Core::X86::Msr::SYS_CFG[SecureNestedPagingEn] is set to 1, this register will be read-only and attempts to write to this register will result in #GP(0).

_ccd[11:0]_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.
6h	WB or write back.
7h	Reserved.

55:53	Reserved.
52	**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.
	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_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.
	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_EE000**: **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.

47:45	Reserved.
44	**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.

<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>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>
<tr><td>34:32</td><td>MemType_4K_EC000: 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_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></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_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></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_EB000: memory type. Read-write. Reset: XXXb.</td></tr>
</table>

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.

23:21 Reserved.

20 **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.

AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.

19 **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.

AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.

18:16 **MemType_4K_EA000**: **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.

15:13 Reserved.

12 **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.

AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.

11 **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.

AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.

10:8 **MemType_4K_E9000**: **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.

7:5 Reserved.

<table>
<tr><td>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></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_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></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_E8000: memory type. Read-write. Reset: XXXb. Address range from E8000h to E8FFFh.</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_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. If Core::X86::Msr::SYS_CFG[SecureNestedPagingEn] is set to 1, this register will be read-only and attempts to write to this register will result in #GP(0).

_ccd[11:0]_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.
	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_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.
	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_F7000**: **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.

55:53	Reserved.
52	**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.
	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_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.
	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_F6000**: **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.

47:45	Reserved.
44	**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.

<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>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></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_F5000: 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_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>
</table>

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.

23:21 Reserved.

20 **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.

AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.

19 **WrDram_4K_F2000**: **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.

18:16 **MemType_4K_F2000**: **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.

15:13 Reserved.

12 **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.

AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.

11 **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.

AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.

10:8 **MemType_4K_F1000**: **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.

7:5 Reserved.

<table>
<tr><td>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></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_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></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_F0000: memory type. Read-write. Reset: XXXb. Address range from F0000h to F0FFFh.</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_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. If Core::X86::Msr::SYS_CFG[SecureNestedPagingEn] is set to 1, this register will be read-only and attempts to write to this register will result in #GP(0).

_ccd[11:0]_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.

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.

47:45	Reserved.
44	**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.

<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>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></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_FD000: 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_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></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_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></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_FC000: 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_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></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_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></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_FB000: memory type. Read-write. Reset: XXXb.</td></tr>
</table>

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.

23:21 Reserved.

20 **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.

AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.

19 **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.

AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.

18:16 **MemType_4K_FA000**: **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.

15:13 Reserved.

12 **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.

AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.

11 **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.

AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.

10:8 **MemType_4K_F9000**: **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.

7:5 Reserved.

<table>
<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. 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_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.</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_F8000: memory type. Read-write. Reset: XXXb. Address range from F8000h to F8FFFh.</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_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[11:0]_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	UC minus.

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.

Bits	Description
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.

<table>
<tr><td>31:27</td><td>Reserved.</td></tr>
<tr><td>26:24</td><td>PA3MemType. Read-write. Reset: 0h. Default UC. MemType for {PAT, PCD, PWT} = 3h.
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>23:19</td><td>Reserved.</td></tr>
<tr><td>18:16</td><td>PA2MemType. Read-write. Reset: 7h. Default UC. MemType for {PAT, PCD, PWT} = 2h.
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>UC minus.</td></tr></table></td></tr>
<tr><td>15:11</td><td>Reserved.</td></tr>
<tr><td>10:8</td><td>PA1MemType. Read-write. Reset: 4h. Default WT. MemType for {PAT, PCD, PWT} = 1h.
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>7:3</td><td>Reserved.</td></tr>
<tr><td>2:0</td><td>PA0MemType. Read-write. Reset: 6h. MemType for {PAT, PCD, PWT} = 0h.
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_02FF [MTRR Default Memory Type] (Core::X86::Msr::MTRRdefType)

See Core::X86::Msr::MtrrVarBase for general MTRR information.

_ccd[11:0]_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[11:0]_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[11:0]_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[11:0]_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 32-bit mode.
1:0	Reserved.

MSR0000_06A5 [PL1 Shadow Stack Pointer] (Core::X86::Msr::PL1Ssp)

Read-write. Reset: 0000_0000_0000_0000h.

_ccd[11:0]_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 32-bit mode.
1:0	Reserved.

MSR0000_06A6 [PL2 Shadow Stack Pointer] (Core::X86::Msr::PL2Ssp)

Read-write. Reset: 0000_0000_0000_0000h.

_ccd[11:0]_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 32-bit mode.
1:0	Reserved.

MSR0000_06A7 [PL3 Shadow Stack Pointer] (Core::X86::Msr::PL3Ssp)

Read-write. Reset: 0000_0000_0000_0000h.

_ccd[11:0]_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 32-bit mode.
1:0	Reserved.

MSR0000_06A8 [Interrupt SSP Table Address] (Core::X86::Msr::IstSspAddr)

Read-write. Reset: 0000_0000_0000_0000h.

_ccd[11:0]_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[11:0]_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[11:0]_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: 1. 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[11:0]_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[11:0]_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[11:0]_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[11:0]_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[11:0]_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[11:0]_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[11:0]_lthree0_core[7:0]_thread[1:0]_nISR0_aliasMSR; MSR0000_0810
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_nISR1_aliasMSR; MSR0000_0811
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_nISR2_aliasMSR; MSR0000_0812
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_nISR3_aliasMSR; MSR0000_0813
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_nISR4_aliasMSR; MSR0000_0814
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_nISR5_aliasMSR; MSR0000_0815
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_nISR6_aliasMSR; MSR0000_0816
_ccd[11:0]_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[11:0]_lthree0_core[7:0]_thread[1:0]_nTMR0_aliasMSR; MSR0000_0818
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_nTMR1_aliasMSR; MSR0000_0819
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_nTMR2_aliasMSR; MSR0000_081A
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_nTMR3_aliasMSR; MSR0000_081B
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_nTMR4_aliasMSR; MSR0000_081C
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_nTMR5_aliasMSR; MSR0000_081D
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_nTMR6_aliasMSR; MSR0000_081E
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_nTMR7_aliasMSR; MSR0000_081F

Bits	Description
63:32	Reserved.
31:0	**TriggerModeBits**. Reset: 0000_0000h. The corresponding trigger mode bit is updated when an interrupt is accepted.
	AccessType: X2APICEN ? Read-only,Error-on-write,Volatile : Error-on-read,Error-on-write.
	ValidValues:

Value	Description
0	Edge-triggered interrupt
1	Level-triggered interrupt

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[11:0]_lthree0_core[7:0]_thread[1:0]_nIRR0_aliasMSR; MSR0000_0820
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_nIRR1_aliasMSR; MSR0000_0821
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_nIRR2_aliasMSR; MSR0000_0822
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_nIRR3_aliasMSR; MSR0000_0823
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_nIRR4_aliasMSR; MSR0000_0824
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_nIRR5_aliasMSR; MSR0000_0825
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_nIRR6_aliasMSR; MSR0000_0826
_ccd[11:0]_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[11:0]_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[11:0]_lthree0_core[7:0]_thread[1:0]; MSR0000_0830

Bits	Description
63:32	**DestinationField**. Reset: 0000_0000h. The destination encoding used when Core::X86::Msr::InterruptCommand[DestShrthnd] is 00b.
	AccessType: X2APICEN ? Read-write : Error-on-read,Error-on-write.
31:20	Reserved.
19:18	**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.
	AccessType: X2APICEN ? Read-write : Error-on-read,Error-on-write.
	ValidValues:

Value	Description
0h	No shorthand (Destination field).
1h	Self.
2h	All including self.
3h	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.)

17:16	Reserved.
15	**TM**: **trigger mode**. Reset: 0. 0=Edge triggered. 1=Level triggered. Indicates how this interrupt is triggered.
	AccessType: X2APICEN ? Read-write : Error-on-read,Error-on-write.
14	**Level**. Reset: 0. 0=Deasserted. 1=Asserted.
	AccessType: X2APICEN ? Read-write : Error-on-read,Error-on-write.
13:12	Reserved.
11	**DM**: **destination mode**. Reset: 0. 0=Physical. 1=Logical.
	AccessType: X2APICEN ? Read-write : Error-on-read,Error-on-write.
10:8	**MsgType**. Reset: 0h. The message types are encoded as follows:
	AccessType: X2APICEN ? Read-write : Error-on-read,Error-on-write.
	ValidValues:

Value	Description
0h	Fixed
1h	Lowest Priority.
2h	SMI
3h	Reserved.
4h	NMI
5h	INIT
6h	Startup
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[11:0]_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[11:0]_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. See2.1.13.2.1.14 [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[11:0]_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. See2.1.13.2.1.14 [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[11:0]_lthree0_core[7:0]_thread[1:0]_nLVTLINT0_aliasMSR; MSR0000_0835

_ccd[11:0]_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. See2.1.13.2.1.14 [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[11:0]_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 2.1.13.2.1.14 [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[11:0]_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[11:0]_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[11:0]_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 performance 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[11:0]_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[11:0]_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 2.1.13.2.1.8 [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[11:0]_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[11:0]_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[11:0]_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[11:0]_lthree0_core[7:0]_thread[1:0]_n1_aliasMSR; MSR0000_0849
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n2_aliasMSR; MSR0000_084A
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n3_aliasMSR; MSR0000_084B
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n4_aliasMSR; MSR0000_084C
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n5_aliasMSR; MSR0000_084D
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n6_aliasMSR; MSR0000_084E
_ccd[11:0]_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[11:0]_lthree0_core[7:0]_thread[1:0]_n0_aliasMSR; MSR0000_0850

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n1_aliasMSR; MSR0000_0851

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n2_aliasMSR; MSR0000_0852

_ccd[11:0]_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. Interrupt Mask.
	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. See2.1.13.2.1.14 [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_0C8D [Monitoring Event Select] (Core::X86::Msr::QM_EVTSEL)

_ccd[11:0]_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: 8000_0000_0000_0000h.

_ccd[11:0]_lthree0; MSR0000_0C8E

Bits	Description
63	**Error**. Read,Error-on-write. Reset: 1. 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 (Core::X86::Msr::PQR_ASSOC)

Reset: 0000_0000_0000_0000h.

QOS L2 RMID. The behavior of this register is defined in the AMD64 Technology Platform Quality of Service Extensions specification.

_ccd[11:0]_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 Monitor Identifier.

MSR0000_0DA0 [Extended Supervisor State] (Core::X86::Msr::XSS)

_ccd[11:0]_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.

2.1.15.2 MSRs – MSRC000_xxxx

MSRC000_0080 [Extended Feature Enable] (Core::X86::Msr::EFER)

SKINIT Execution: 0000_0000_0000_0000h.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; MSRC000_0080

Bits	Description
63:22	Reserved.
21	**AutomaticIBRSEn**: **Automatic IBRS Enable**. Read-write. Reset: 0. 0=IBRS protection is not enabled unless (SPEC_CTRL[IBRS] == 1). 1=IBRS protection is enabled for any process running at (CPL < 3) or ((ASID == 0) && SEV-SNP).
20	**UAIE**: **Upper Address Ignore Enable**. Read-write. Reset: 0. Upper Address Ignore suppresses canonical faults for most data access virtual addresses, which allows software to use the upper bits of a virtual address as tags.
19	Reserved.
18	**IntWbinvdEn**. Read-write. Reset: 0. Interruptible wbinvd, wbnoinvd enable.
17:16	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[11:0]_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[11:0]_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[11:0]_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[11:0]_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[11:0]_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[11:0]_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[11:0]_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[11:0]_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[11:0]_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[11:0]_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 fill occurs).

MSRC000_0103 [Auxiliary Time Stamp Counter] (Core::X86::Msr::TSC_AUX)

Read-write,Volatile. Reset: 0000_0000_0000_0000h.

_ccd[11:0]_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 when a guest frequency matches the core P0 frequency.

_ccd[11:0]_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[11:0]_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_010E [LBR V2 Branch Select] (Core::X86::Msr::LbrSelect)

Read-write. Reset: 0000_0000_0000_0000h.

This MSR allows LBR V2 recording to be suppressed based on branch type and privilege level.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; MSRC000_010E

Bits	Description
63:9	Reserved.
8	**FarBranch**. Read-write. Reset: 0. When set far branches are not recorded.
7	**JmpNearRel**. Read-write. Reset: 0. When set, near relative jumps, excluding near relative calls, are not recorded.
6	**JmpNearInd**. Read-write. Reset: 0. When set, near indirect jumps, excluding near indirect calls and near returns, are not recorded.
5	**RetNear**. Read-write. Reset: 0. When set, near returns are not recorded.
4	**CallNearInd**. Read-write. Reset: 0. When set, near indirect calls are not recorded.
3	**CallNearRel**. Read-write. Reset: 0. When set, near relative calls are not recorded.
2	**Jcc**. Read-write. Reset: 0. When set conditional branches are not recorded.
1	**CplGe0**. Read-write. Reset: 0. When set, no branches ending in CPL > 0 are recorded.
0	**CplEq0**. Read-write. Reset: 0. When set, no branches ending in CPL = 0 are recorded.

MSRC000_010F [Debug Extension Configuration] (Core::X86::Msr::DbgExtnCfg)

Reset: 0000_0000_0000_0000h.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; MSRC000_010F

Bits	Description
63:7	Reserved.
6	**LBRV2EN**. Read-write. Reset: 0. When enabled the last 16 branch targets and from addresses are recorded in LBR_TO_V2_[0..15] and LBR_FROM_V2_[0..15]. Core::X86::Msr::DbgExtnCfg[LBRV2EN] is cleared upon #DB entry.
5:0	Reserved.

MSRC000_0300 [Performance Counter Global Status] (Core::X86::Msr::PerfCntrGlobalStatus)

Read-only. Reset: 0000_0000_0000_0000h.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; MSRC000_0300

Bits	Description
63:6	Reserved.
5:0	**PerfCntrOvfl**: **Performance Counter Overflow Bits**. Read-only. Reset: 00h. For each available core Performance Counter there is one overflow bit starting at bit position 0 for counter 0 (PerfCntr0). The bit is set when the corresponding counter overflows and remains set until cleared by software via Core::X86::Msr::PerfCntrGlobalStatusClr. The bits can also be set directly by software via Core::X86::Msr::PerfCntrGlobalStatusSet. Note that the overflow bit is set even when the Performance Counter was not configured to signal an interrupt.

MSRC000_0301 [Performance Counter Global Control] (Core::X86::Msr::PerfCntrGlobalCtl)

Read-write. Reset: 0000_0000_0000_003Fh.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; MSRC000_0301

Bits	Description
63:6	Reserved.
5:0	**PerfCntrEn**: **Global Performance Counter Enable**. Read-write. Reset: 3Fh. For each available performance counter there is one enable bit in this field. Bit position 0 corresponds to counter 0 (PerfCntr0), bit position 1 corresponds to counter 1 (PerfCtntr1) and so forth. A Performance counter is enabled to count when both its Core::X86::Msr::PerfCntrGlobalCtl[PerfCntrEn] bit and its Core::X86::Msr::PERF_CTL0..5[En] bit are set.

MSRC000_0302 [**Performance Counter Global Status Clear**] **(Core::X86::Msr::PerfCntrGlobalStatusClr)**

Write-only. Reset: 0000_0000_0000_0000h.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; MSRC000_0302

Bits	Description
63:6	Reserved.
5:0	**PerfCntrOvflClr**: **Performance Counter Overflow Bits Clear**. Write-only. Reset: 00h.
	Description: These bits allow software to clear PerfCntrOvfl bits in Core::X86::Msr::PerfCntrGlobalStatus. To clear a bit software needs to write a 1 to the corresponding bit. Software should clear the Performance Counter Overflow bits when: - Handling a performance counter overflow interrupt - Disabling a performance counter - Resetting a performance counter

MSRC000_0303 [**Performance Counter Global Status Set**] **(Core::X86::Msr::PerfCntrGlobalStatusSet)**

Write-only. Reset: 0000_0000_0000_0000h.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; MSRC000_0303

Bits	Description
63:6	Reserved.
5:0	**PerfCntrOvflSet**: **Performance Counter Overflow Bits Set**. Write-only. Reset: 00h. These bits allow software to set PerfCntrOvfl bits PerfCntrGlobalStatus. Setting an overflow bit in PerfCntrGlobalStatus does not result in the generation of an interrupt, freeze of performance counters, freeze of LBR or other actions that may be taken when a performance counter overflows.

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. 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 (seeAPIC[530:500]).
3:0	Reserved.

2.1.15.3 MSRs - MSRC001_0xxx

MSRC001_0000 [Performance Event Select 0] (Core::X86::Msr::PERF_LEGACY_CTL0)

Read-write. Reset: 0000_0000_0000_0000h.

The legacy alias of Core::X86::Msr::PERF_CTL0. See Core::X86::Msr::PERF_CTL0.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_0000

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 2.1.16.4 [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. Performance counter enable.
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. APIC Interrupt enable.
19	Reserved.
18	**Edge**: **edge detect**. Read-write. Reset: 0. 0=Level detect. 1=Zero-to-one Edge detect. 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. Event counter for OS and user mode. **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 2.1.16.5 [Core Performance Monitor Counters]. Some events are Reserved; when a Reserved event is selected, the results are undefined.

MSRC001_0001 [Performance Event Select 1] (Core::X86::Msr::PERF_LEGACY_CTL1)

Read-write. Reset: 0000_0000_0000_0000h.

The legacy alias of Core::X86::Msr::PERF_CTL1. See Core::X86::Msr::PERF_CTL1.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_0001

Bits	Description
63:42	Reserved.
41:40	**HostGuestOnly**: **count only host/guest events**. Read-write. Reset: 0h. Host/Guest event counter. **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 2.1.16.4 [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. Performance counter enable.
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. APIC interrupt enable.
19	Reserved.
18	**Edge**: **edge detect**. Read-write. Reset: 0. 0=Level detect. 1=Zero-to-one Edge detect. 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. OS and user mode counter events. **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 2.1.16.5 [Core Performance Monitor Counters]. Some events are Reserved; when a Reserved event is selected, the results are undefined.

MSRC001_0002 [Performance Event Select 2] (Core::X86::Msr::PERF_LEGACY_CTL2)

Read-write. Reset: 0000_0000_0000_0000h.

The legacy alias of Core::X86::Msr::PERF_CTL2. See Core::X86::Msr::PERF_CTL2.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_0002

Bits	Description
63:42	Reserved.
41:40	**HostGuestOnly**: **count only host/guest events**. Read-write. Reset: 0h. Host/Guest event counter. **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 2.1.16.4 [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. Performance counter enable.
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. APIC interrupt enable.
19	Reserved.
18	**Edge**: **edge detect**. Read-write. Reset: 0. 0=Level detect. 1=Zero-to-one Edge detect. 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. OS and user mode. **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 2.1.16.5 [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[11:0]_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. Host/guest event counter. **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 2.1.16.4 [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. Performance counter enable.
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. APIC interrupt enable.
19	Reserved.
18	**Edge**: **edge detect**. Read-write. Reset: 0. 0=Level detect. 1=Zero-to-one Edge detect. 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. OS and user mode counter events. **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 2.1.16.5 [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 2.1.16.4 [Large Increment per Cycle Events].
The legacy alias of Core::X86::Msr::PERF_CTR. See Core::X86::Msr::PERF_CTR.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n0; MSRC001_0004
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n1; MSRC001_0005
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n2; MSRC001_0006
_ccd[11:0]_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.

If Core::X86::Msr::SYS_CFG[SecureNestedPagingEn] is set, writes to this register are ignored.

_ccd[11:0]_lthree0_core[7:0]; MSRC001_0010

Bits	Description
63:27	Reserved.
26	**HMKEE**: **Host Multi-Key Encryption Enable**. Read,Write-1-only. Reset: 0. Used with SYS_CFG[SMEE] to select secure memory encryption mode. See SYS_CFG[SMEE] for a table listing the available memory encryption modes.
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.
	Description: Used with SYS_CFG[HMKEE] to select secure memory encryption mode. See the table below for the available memory encryption modes.

HMKEE	SMEE	Description
0	0	No encryption.
0	1	Enables SME and SEV memory encryption.
1	0	Enables SME-HMK memory encryption.
1	1	Not supported. Results in #GP.

22	**Tom2ForceMemTypeWB**: **top of memory 2 memory type write back**. Read-write. Reset: 0. 1=The default memory type of memory between 4GB 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[11:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_0015

Bits	Description
63:36	Reserved.
35	**CpuidFltEn**. Read-write. Reset: 0. 1=Executing CPUID outside of SMM and with CPL > 0 results in #GP.
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/Write 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. Init: BIOS,1. 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. Init: BIOS,1. 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 FERR signalling when generating an x87 floating point exception (when CR0[NE] is set). 1=FERR is signaled on any x87 floating point exception, regardless of CR0[NE].
6:5	Reserved.
4	**INVDWBINVD**: **INVD to WBINVD conversion**. Read,Error-on-write-0. Reset: 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. Operating system should do a full TLB flush before and after any changes to this bit value. • 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 2.1.13.1.10 [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[51:12] & PhyMask[51:12] == PhyBase[51:12] & PhyMask[51:12].
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.
If Core::X86::Msr::SYS_CFG[SecureNestedPagingEn] is set to 1, this register will be read-only and attempts to write to this register will result in #GP(0).

_ccd[11:0]_lthree0_core[7:0]_n0; MSRC001_0016

_ccd[11:0]_lthree0_core[7:0]_n1; MSRC001_0018

Bits	Description
63:52	Reserved.
51:12	**PhyBase**. Read-write. Reset: XX_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.
If Core::X86::Msr::SYS_CFG[SecureNestedPagingEn] is set to 1, this register will be read-only and attempts to write to this register will result in #GP(0).

_ccd[11:0]_lthree0_core[7:0]_n0; MSRC001_0017

_ccd[11:0]_lthree0_core[7:0]_n1; MSRC001_0019

Bits	Description
63:52	Reserved.
51:12	**PhyMask**. Read-write. Reset: 00_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.

If Core::X86::Msr::SYS_CFG[SecureNestedPagingEn] is set to 1, this register will be read-only and attempts to write to this register will result in #GP(0).

_ccd[11:0]_lthree0_core[7:0]; MSRC001_001A

Bits	Description
63:52	Reserved.
51:23	**TOM[51:23]**: **top of memory**. Read-write. Reset: XXXX_XXXXh. Specifies the address that divides between MMIO and DRAM. This value is normally placed below 4G. From TOM to 4G 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.

If Core::X86::Msr::SYS_CFG[SecureNestedPagingEn] is set to 1, this register will be read-only and attempts to write to this register will result in #GP(0).

_ccd[11:0]_lthree0_core[7:0]; MSRC001_001D

Bits	Description
63:52	Reserved.
51:23	**TOM2[51:23]**: **second top of memory**. Read-write. Reset: XXXX_XXXXh. Specifies the address divides between MMIO and DRAM. This value is normally placed above 4G. From 4G 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_0020 [**Patch Loader**] **(Core::X86::Msr::PATCH_LOADER)**

Write-only,Error-on-read.

_ccd[11:0]_lthree0_core[7:0]; MSRC001_0020

Bits	Description
63:0	**PatchBase**. Write-only,Error-on-read. Reset: XXXX_XXXX_XXXX_XXXXh. Linear address of the Microcode Patch Block. PatchBase[63:32] is ignored when the core is not operating in 64-bit mode.

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[11:0]_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[11:0]_lthree0_core[7:0]_thread[1:0]_n0; MSRC001_0030
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n1; MSRC001_0031
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n2; MSRC001_0032
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n3; MSRC001_0033
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n4; MSRC001_0034
_ccd[11:0]_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 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[11:0]_lthree0_core[7:0]_thread[1:0]_n0; MSRC001_0050
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n1; MSRC001_0051
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n2; MSRC001_0052
_ccd[11:0]_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. SMI IO trap mask. **ValidValues**: Value 0: Mask address bit Value 1: Do not mask address bit
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[11:0]_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[11:0]_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 2.1.13.1.3 [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[11:0]_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 2.1.13.1.3 [SMI Sources And Delivery].
15:0	**IoPortAddress**. Read-write. Reset: 0000h. See 2.1.13.1.3 [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[11:0]_lthree0_core[7:0]; MSRC001_0058

Bits	Description
63:52	Reserved.
51:20	**MmioCfgBaseAddr[51:20]**: **MMIO configuration base address bits[51:20]**. Read-write. Reset: XXXX_XXXXh. Specifies the base address of the MMIO configuration range.
19:6	Reserved.
5:2	**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. **ValidValues**:

Value	Description
0h	1 bus, 1 segment
1h	2 buses, 1 segment
2h	4 buses, 1 segment
3h	8 buses, 1 segment
4h	16 buses, 1 segment
5h	32 buses, 1 segment
6h	64 buses, 1 segment
7h	128 buses, 1 segment
8h	256 buses, 1 segment
9h	2 segments, 256 buses per segment
Ah	4 segments, 256 buses per segment
Bh	8 segments, 256 buses per segment
Ch	16 segments, 256 buses per segment
Dh	32 segments, 256 buses per segment
Eh	64 segments, 256 buses per segment
Fh	128 segments, 256 buses per segment

1	Reserved.
0	**Enable**. Read-write. Reset: 0. 1=MMIO configuration space is enabled.

MSRC001_0061 [P-state Current Limit] (Core::X86::Msr::PStateCurLim)

_ccd[11:0]_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[11:0]_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[11:0]_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.
62:33	Reserved.
32	**CpuVid[8]**: **core VID[8]**. Read-write. Reset: X.
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[7:0]**. 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 defined as CpuFid * 5MHz.

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[11:0]_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_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 2.1.13.1.5 [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[11:0]_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 2.1.13.1 [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[51:17] & TSegMask[51:17] == TSegBase[51:17] & TSegMask[51:17].

For example, if TSeg spans 256 KB and starts at the 1 MB address. The Core::X86::Msr::SMMAddr[TSegBase[51:17]] would be set to 0_0010_0000h and the Core::X86::Msr::SMMMask[TSegMask[51:17]] to 0_FFFC_0000h (with zeros filling in for bits[16:0]). This results in a TSeg range from 0_0010_0000 to 0_0013_FFFFh.

_ccd[11:0]_lthree0_core[7:0]; MSRC001_0112

Bits	Description
63:52	Reserved.
51:17	**TSegBase[51:17]**: **TSeg address range base**. Configurable. Reset: 0_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 2.1.13.1 [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[51:17]]) with a variable size (specified by Core::X86::Msr::SMMMask[TSegMask[51:17]]). 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[11:0]_lthree0_core[7:0]; MSRC001_0113

Bits	Description
63:52	Reserved.
51:17	**TSegMask[51:17]**: **TSeg address range mask**. Configurable. Reset: 0_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.

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[11:0]_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[11:0]_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_0117 [Virtual Machine Host Save Physical Address] (Core::X86::Msr::VM_HSAVE_PA)

Reset: 0000_0000_0000_0000h.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_0117

Bits	Description
63:52	Reserved.
51:12	**VM_HSAVE_PA**: **physical address of host save area**. Read-write. Reset: 00_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[11:0]_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. Reads of this register always return zero.

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[11:0]_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 cause 4KB 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[11:0]_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[11:0]_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[11:0]_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[11:0]_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[11:0]_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_0135 [Virtual TOM] (Core::X86::Msr::VIRTUAL_TOM)

Configurable. Reset: 0000_0000_0000_0000h.

Access of Core::X86::Msr::VIRTUAL_TOM in hypervisor mode causes #GP.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_0135

Bits	Description
63:52	Reserved.
51:21	**VIRTUAL_TOM**. Configurable. Reset: 0000_0000h. Guest physical addresses below VIRTUAL_TOM are considered private (C=1) when VIRTUAL_TOM is enabled. Access is AccessType: (SEV_FEATURES[VirtualTom] AND SEV_FEATURES[SNPActive]) ? Read-write : Error-on-read, Error-on-write.
20:0	Reserved.

MSRC001_0140 [OS Visible Work-around Length] (Core::X86::Msr::OSVW_ID_Length)

Read-write. Reset: 0000_0000_0000_0000h.

_ccd[11:0]_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. Init: BIOS,0005h. 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[11:0]_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 2.1.16 [Performance Monitor Counters]. Core::X86::Msr::PERF_LEGACY_CTL0 is an alias of this register.

_ccd[11:0]_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. Host/Guest event counter. **ValidValues**: Value: 0h — Count all events, irrespective of guest/host. Value: 1h — Count guest events if [SVME] == 1. Value: 2h — Count host events if [SVME] == 1. Value: 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: 00h — The corresponding PERF_CTR[5:0] register increments by the number of events occurring in a clock cycle. See 2.1.16.4 [Large Increment per Cycle Events] for events that can increment greater than 15 per cycle. Value: 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. 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. Performance counter enable.
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. APIC interrupt enable.
19	Reserved.
18	**Edge**: **edge detect**. Read-write. Reset: 0. 0=Level detect. 1=Zero-to-one Edge detect. 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. OS and user mode counter events. **ValidValues**: Value: 0h — Count no events. Value: 1h — Count user events (CPL>0). Value: 2h — Count OS events (CPL=0). Value: 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 2.1.16.5 [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 2.1.16.4 [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[11:0]_lthree0_core[7:0]_thread[1:0]_n0; MSRC001_0201
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n1; MSRC001_0203
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n2; MSRC001_0205
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n3; MSRC001_0207
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n4; MSRC001_0209
_ccd[11:0]_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)

Read-write. Reset: 0000_0000_0000_0000h.

See 2.1.16 [Performance Monitor Counters]. Core::X86::Msr::PERF_LEGACY_CTL1 is an alias of this register.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_0202

Bits	Description
63:42	Reserved.
41:40	**HostGuestOnly**: **count only host/guest events**. Read-write. Reset: 0h. Host/Guest event counter. **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 2.1.16.4 [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. Performance counter enable.
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. APIC interrupt enable.
19	Reserved.
18	**Edge**: **edge detect**. Read-write. Reset: 0. 0=Level detect. 1=Zero-to-one Edge detect. 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. OS and user mode counter events. **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 2.1.16.5 [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)

Read-write. Reset: 0000_0000_0000_0000h.

See 2.1.16 [Performance Monitor Counters]. Core::X86::Msr::PERF_LEGACY_CTL2 is an alias of this register.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_0204

Bits	Description
63:42	Reserved.
41:40	**HostGuestOnly**: **count only host/guest events**. Read-write. Reset: 0h. Host/Guest event counter. **ValidValues**: Value: 0h — Count all events, irrespective of guest/host. Value: 1h — Count guest events if [SVME] == 1. Value: 2h — Count host events if [SVME] == 1. Value: 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: 00h — The corresponding PERF_CTR[5:0] register increments by the number of events occurring in a clock cycle. See 2.1.16.4 [Large Increment per Cycle Events] for events that can increment greater than 15 per cycle. Value: 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. 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. Performance counter enable.
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. APIC interrupt enable.
19	Reserved.
18	**Edge**: **edge detect**. Read-write. Reset: 0. 0=Level detect. 1=Zero-to-one Edge detect. 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. OS and user mode counter events. **ValidValues**: Value: 0h — Count no events. Value: 1h — Count user events (CPL>0). Value: 2h — Count OS events (CPL=0). Value: 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 2.1.16.5 [Core Performance Monitor Counters]. Some events are Reserved; when a Reserved event is selected, the results are undefined.

MSRC001_0206 [Performance Event Select 3] (Core::X86::Msr::PERF_CTL3)

Read-write. Reset: 0000_0000_0000_0000h.

See 2.1.16 [Performance Monitor Counters]. Core::X86::Msr::PERF_LEGACY_CTL3 is an alias of this register.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_0206

Bits	Description
63:42	Reserved.
41:40	**HostGuestOnly**: **count only host/guest events**. Read-write. Reset: 0h. Host/Guest event counter. **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 2.1.16.4 [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. Performance counter enable.
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. APIC interrupt enable.
19	Reserved.
18	**Edge**: **edge detect**. Read-write. Reset: 0. 0=Level detect. 1=Zero-to-one Edge detect. 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. OS and user mode counter events. **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 2.1.16.5 [Core Performance Monitor Counters]. Some events are Reserved; when a Reserved event is selected, the results are undefined.

MSRC001_0208 [Performance Event Select 4] (Core::X86::Msr::PERF_CTL4)

Read-write. Reset: 0000_0000_0000_0000h.

See 2.1.16 [Performance Monitor Counters].

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_0208

Bits	Description
63:42	Reserved.
41:40	**HostGuestOnly**: **count only host/guest events**. Read-write. Reset: 0h. Host/Guest event counters. **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 2.1.16.4 [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. Performance counter enable.
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. APIC interrupt enable.
19	Reserved.
18	**Edge**: **edge detect**. Read-write. Reset: 0. 0=Level detect. 1=Zero-to-one Edge detect. 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. OS and user mode counter events. **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 2.1.16.5 [Core Performance Monitor Counters]. Some events are Reserved; when a Reserved event is selected, the results are undefined.

MSRC001_020A [Performance Event Select 5] (Core::X86::Msr::PERF_CTL5)

Read-write. Reset: 0000_0000_0000_0000h.

See 2.1.16 [Performance Monitor Counters].

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_020A

Bits	Description
63:42	Reserved.
41:40	**HostGuestOnly**: **count only host/guest events**. Read-write. Reset: 0h. Host/Guest event counter. **ValidValues**: Value: 0h — Count all events, irrespective of guest/host. Value: 1h — Count guest events if [SVME] == 1. Value: 2h — Count host events if [SVME] == 1. Value: 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: 00h — The corresponding PERF_CTR[5:0] register increments by the number of events occurring in a clock cycle. See 2.1.16.4 [Large Increment per Cycle Events] for events that can increment greater than 15 per cycle. Value: 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. 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. Performance counter enable.
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. APIC interrupt enable.
19	Reserved.
18	**Edge**: **edge detect**. Read-write. Reset: 0. 0=Level detect. 1=Zero-to-one Edge detect. 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. OS and user mode counter events. **ValidValues**: Value: 0h — Count no events. Value: 1h — Count user events (CPL>0). Value: 2h — Count OS events (CPL=0). Value: 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 2.1.16.5 [Core Performance Monitor Counters]. Some events are Reserved; when a Reserved event is selected, the results are undefined.

MSRC001_023[0...A] [L3 Performance Event Select [5:0]] (Core::X86::Msr::ChL3PmcCfg)

Read-write. Reset: 0000_0000_0000_0000h.

See 2.1.16.6 [L3 Cache Performance Monitor Counters]

_ccd[11:0]_lthree0_n0; MSRC001_0230
_ccd[11:0]_lthree0_n1; MSRC001_0232
_ccd[11:0]_lthree0_n2; MSRC001_0234
_ccd[11:0]_lthree0_n3; MSRC001_0236
_ccd[11:0]_lthree0_n4; MSRC001_0238
_ccd[11:0]_lthree0_n5; MSRC001_023A

Bits	Description
63:60	Reserved.
59:56	**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. **ValidValues**: Bit: [0] — Thread 0. Bit: [1] — Thread 1. Bit: [3:2] — Reserved.
55:51	Reserved.
50:48	**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**: Value: 7h-0h — <Value> Slice.
47	**EnAllCores**. Read-write. Reset: 0. 1=Enable counting L3 events for all cores simultaneously.
46	**EnAllSlices**. Read-write. Reset: 0. 1=Enable counting L3 events for all 8 L3 slices simultaneously.
45	Reserved.
44:42	**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**: Value: 7h-0h — <Value> CoreId.
41:23	Reserved.
22	**Enable**: **Enable L3 performance counter**. Read-write. Reset: 0. 1=Enable.
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. 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.
7:0	**EventSel**. Read-write. Reset: 00h. L3 Event select.

MSRC001_02[40...5E] [Data Fabric Performance Event Select [3:0]] (Core::X86::Msr::DF_PERF_CTL)

Read-write. Reset: 0000_0000_0000_0000h.

See 2.1.16 [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
_n4; MSRC001_0248
_n5; MSRC001_024A
_n6; MSRC001_024C
_n7; MSRC001_024E
_n8; MSRC001_0250
_n9; MSRC001_0252
_n10; MSRC001_0254
_n11; MSRC001_0256
_n12; MSRC001_0258
_n13; MSRC001_025A
_n14; MSRC001_025C
_n15; MSRC001_025E

Bits	Description
63:38	Reserved.
37:32	**EventSelect[13:8]**: **performance event select**. Read-write. Reset: 00h. Performance event select [13:0]. See EventSelect[7:0].
31:28	Reserved.
27:24	**UnitMask[11:8]**: **event qualification**. Read-write. Reset: 0h. Uppper 4 bits of UnitMask. See UnitMask[7:0].
23	Reserved.
22	**En**: **enable performance counter**. Read-write. Reset: 0. 1=Performance event counter is enabled.
21:16	Reserved.
15:8	**UnitMask[7:0]**: **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[13:8] 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.

MSRC001_02[41...5F] [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 socket.

_n0; MSRC001_0241
_n1; MSRC001_0243
_n2; MSRC001_0245
_n3; MSRC001_0247
_n4; MSRC001_0249
_n5; MSRC001_024B
_n6; MSRC001_024D
_n7; MSRC001_024F
_n8; MSRC001_0251
_n9; MSRC001_0253
_n10; MSRC001_0255
_n11; MSRC001_0257
_n12; MSRC001_0259
_n13; MSRC001_025B
_n14; MSRC001_025D
_n15; MSRC001_025F

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_029A [Core Energy Status] (Core::X86::Msr::CORE_ENERGY_STAT)

Read-only,Volatile. Reset: 0000_0000_0000_0000h.

_ccd[11:0]_lthree0_core[7:0]; MSRC001_029A

Bits	Description
63:0	**TotalEnergyConsumed**. Read-only,Volatile. Reset: 0000_0000_0000_0000h.

MSRC001_02B0 [CPPC Capability 1] (Core::X86::Msr::CppcCapability1)

Collaborative Processor Performance Control Capability 1.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_02B0

Bits	Description
63:32	Reserved.
31:24	**HighestPerf**: **Highest Performance**. Read-only,Error-on-write,Volatile. Reset: 00h. Value for the maximum non-ensured performance level.
23:16	**NominalPerf**: **Nominal Performance**. Read-only,Error-on-write,Volatile. Reset: 00h. Value for the maximum sustained performance level assuming ideal operating conditions.
15:8	**LowNonLinPerf**: **Lowest Nonlinear Performance**. Read-only,Error-on-write,Volatile. Reset: 00h. Value for the lowest nonlinear performance level.
7:0	**LowestPerf**: **Lowest Performance**. Read-only,Error-on-write,Volatile. Reset: 00h. Value for the lowest performance level that software can program to MSR_CPPC_REQUEST.

MSRC001_02B1 [CPPC Enable] (Core::X86::Msr::CppcEnable)

Collaborative Processor Performance Control Enable.

MSRC001_02B1

Bits	Description
63:1	Reserved.
0	**CppcEnable**. Read,Write-1-only. Reset: 0. CPPC Enable.

MSRC001_02B2 [CPPC Capability 2] (Core::X86::Msr::CppcCapability2)

Collaborative Processor Performance Control Capability 2.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_02B2

Bits	Description
63:8	Reserved.
7:0	**MaxPerf**: **constrained maximum performance**. Read-only,Error-on-write,Volatile. Reset: 00h. Value for the current maximum performance level taking into account all known external constraints (i.e., power limits, thermal limits, AC/DC power source, etc.).

MSRC001_02B3 [CPPC Request] (Core::X86::Msr::CppcRequest)

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_02B3

Bits	Description
63:32	Reserved.
31:24	**EnergyPerfPref**. Read-write. Reset: 00h. Energy Performance Preference.
23:16	**DesPerf**. Read-write. Reset: 00h. Desired Performance.
15:8	**MinPerf**. Read-write. Reset: 00h. Minimum Performance.
7:0	**MaxPerf**. Read-write. Reset: 00h. Maximum Performance.

MSRC001_02B4 [CPPC Status] (Core::X86::Msr::CppcStatus)

Reset: 0000_0000_0000_0000h.

Collaborative Processor Performance Control Status.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_02B4

Bits	Description
63:2	Reserved.
1	**MINEXCURSION**. Read-write,Volatile. Reset: 0. 1=Minimum Excursion has occurred.
0	Reserved.

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] [LBR V2 Branch From] (Core::X86::Msr::LBR_FROM_V2)

Reset: 0000_0000_0000_0000h.

These MSRs capture the branch from IP when LBR V2 is enabled (see Core::X86::Msr::DbgExtnCfg[LBRV2EN]). The youngest branch is in LBR_FROM_V2_00 and the oldest branch is in LBR_FROM_V2_15.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n0; MSRC001_0300
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n1; MSRC001_0302
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n2; MSRC001_0304
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n3; MSRC001_0306
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n4; MSRC001_0308
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n5; MSRC001_030A
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n6; MSRC001_030C
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n7; MSRC001_030E
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n8; MSRC001_0310
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n9; MSRC001_0312
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n10; MSRC001_0314
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n11; MSRC001_0316
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n12; MSRC001_0318
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n13; MSRC001_031A
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n14; MSRC001_031C
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n15; MSRC001_031E

Bits	Description
63	**Mispredict**. Read-write. Reset: 0. Indicates if the recorded branch mispredicted.
62:58	**BranchFromIpSignExt**. Read-write,Volatile. Reset: 00h. Sign extension of BranchFromIp.
57:0	**BranchFromIp**. Read-write,Volatile. Reset: 000_0000_0000_0000h. Either the lower 63b of the branch segment offset or the segment offset of an instruction preceding the branch. If the segment offset recorded is not the start of the branch instruction, it is the start of a non-branch instruction up to 16 bytes before the end of the branch instruction. The next branch in the sequential path after this instruction is the branch for this LBR Stack entry. Not recording the segment offset of the branch is the result of instruction fusion. If it is desired that BranchFromIp always records the address of branches Instruction Fusion needs to be disabled by also enabling Legacy LBR via Core::X86::Msr::DBG_CTL_MSR[LBR].

MSRC001_03[01...1F] [LBR V2 Branch To] (Core::X86::Msr::LBR_TO_V2)

Reset: 0000_0000_0000_0000h.

These MSRs capture the branch to IP and additional information when LBR V2 is enabled (see Core::X86::Msr::DbgExtnCfg[LBRV2EN]). The youngest branch is in LBR_TO_V2_00 and the oldest branch is in LBR_TO_V2_15.

The following table shows the types of branch records based on the Valid and Spec bits:

Valid	Spec	Description
1	0	Normal recorded branch.
1	1	Branch was recorded when speculative performance feature was active.
0	1	Branch was recorded but speculative feature was not successful.
0	0	No branch has yet been logged in this entry since software last cleared this MSR by writing 0.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n0; MSRC001_0301
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n1; MSRC001_0303
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n2; MSRC001_0305
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n3; MSRC001_0307
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n4; MSRC001_0309
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n5; MSRC001_030B
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n6; MSRC001_030D
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n7; MSRC001_030F
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n8; MSRC001_0311
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n9; MSRC001_0313
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n10; MSRC001_0315
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n11; MSRC001_0317
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n12; MSRC001_0319
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n13; MSRC001_031B
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n14; MSRC001_031D
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]_n15; MSRC001_031F

Bits	Description
63	**Valid**. Read-write. Reset: 0. The valid bit is cleared when the branch was speculatively recorded by the hardware but eventually discarded. Valid LastBranchFrom/To entries have this bit set.
62	**Spec**: **Speculative**. Read-write. Reset: 0. When set, the entry was written speculatively .
61	Reserved.
60:58	**BranchToIpSignExt**. Read-write,Volatile. Reset: 0h. Sign extension of BranchToIp.
57:0	**BranchToIp**. Read-write,Volatile. Reset: 000_0000_0000_0000h. Lower 58b of the branch target code segment offset.

MSRC001_08[00...7E] [UMC Performance Monitor Control] (Core::X86::Msr::UMC_PerfMonCtl)
Read-write. Reset: 0000_0000_0000_0000h.
These MSRs provide access to the UMC Performance Monitor Control registers. Refer to Core::X86::Cpuid::ExtPerfMonAndDbgEbx[NumPerfCtrUmc] and Core::X86::Cpuid::ExtPerfMonAndDbgEcx[ActiveUmcMask] for the number of available UMC PMCs and the number of PMCs per UMC. MSRs belonging to non-existent UMC Performance Monitors return zero when read and ignore writes. Reserved bits must be set to 0 for reliable results.
_n0; MSRC001_0800
_n1; MSRC001_0802
_n2; MSRC001_0804
_n3; MSRC001_0806
_n4; MSRC001_0808
_n5; MSRC001_080A
_n6; MSRC001_080C
_n7; MSRC001_080E
_n8; MSRC001_0810
_n9; MSRC001_0812
_n10; MSRC001_0814
_n11; MSRC001_0816
_n12; MSRC001_0818
_n13; MSRC001_081A
_n14; MSRC001_081C
_n15; MSRC001_081E
_n16; MSRC001_0820
_n17; MSRC001_0822
_n18; MSRC001_0824
_n19; MSRC001_0826
_n20; MSRC001_0828
_n21; MSRC001_082A
_n22; MSRC001_082C
_n23; MSRC001_082E
_n24; MSRC001_0830
_n25; MSRC001_0832
_n26; MSRC001_0834
_n27; MSRC001_0836
_n28; MSRC001_0838
_n29; MSRC001_083A
_n30; MSRC001_083C
_n31; MSRC001_083E
_n32; MSRC001_0840
_n33; MSRC001_0842
_n34; MSRC001_0844
_n35; MSRC001_0846
_n36; MSRC001_0848
_n37; MSRC001_084A
_n38; MSRC001_084C
_n39; MSRC001_084E
_n40; MSRC001_0850
_n41; MSRC001_0852
_n42; MSRC001_0854
_n43; MSRC001_0856
_n44; MSRC001_0858
_n45; MSRC001_085A
_n46; MSRC001_085C
_n47; MSRC001_085E
_n48; MSRC001_0860
_n49; MSRC001_0862
_n50; MSRC001_0864
_n51; MSRC001_0866
_n52; MSRC001_0868
_n53; MSRC001_086A
_n54; MSRC001_086C
_n55; MSRC001_086E

_n56; MSRC001_0870	
_n57; MSRC001_0872	
_n58; MSRC001_0874	
_n59; MSRC001_0876	
_n60; MSRC001_0878	
_n61; MSRC001_087A	
_n62; MSRC001_087C	
_n63; MSRC001_087E	
Bits	**Description**
63:32	Reserved.
31	**Enable**. Read-write. Reset: 0. 0=Disable. 1=Enable. Counter enable.
30:10	Reserved.
9:8	**RdWrMask**. Read-write. Reset: 0h. Mask transactions based on read or write.
	ValidValues:

Value	Description
0h	No masking, includes reads and writes.
1h	Mask writes.
2h	Mask reads.
3h	Reserved.

7:0	**EventSelect**. Read-write. Reset: 00h. Select the performance monitor event.

MSRC001_08[01...7F] [UMC Performance Monitor Counter] (Core::X86::Msr::UMC_PerfMonCntr)
Read-write. Reset: 0000_0000_0000_0000h.
These MSRs provide access to the UMC Performance Monitor Counter registers. Refer to Core::X86::Cpuid::ExtPerfMonAndDbgEbx[NumPerfCtrUmc] and Core::X86::Cpuid::ExtPerfMonAndDbgEcx[ActiveUmcMask] for the number of available UMC PMCs and the number of PMCs per UMC. MSRs belonging to non-existent UMC Performance Monitors return zero when read and ignore writes.
_n0; MSRC001_0801
_n1; MSRC001_0803
_n2; MSRC001_0805
_n3; MSRC001_0807
_n4; MSRC001_0809
_n5; MSRC001_080B
_n6; MSRC001_080D
_n7; MSRC001_080F
_n8; MSRC001_0811
_n9; MSRC001_0813
_n10; MSRC001_0815
_n11; MSRC001_0817
_n12; MSRC001_0819
_n13; MSRC001_081B
_n14; MSRC001_081D
_n15; MSRC001_081F
_n16; MSRC001_0821
_n17; MSRC001_0823
_n18; MSRC001_0825
_n19; MSRC001_0827
_n20; MSRC001_0829
_n21; MSRC001_082B
_n22; MSRC001_082D
_n23; MSRC001_082F
_n24; MSRC001_0831
_n25; MSRC001_0833
_n26; MSRC001_0835
_n27; MSRC001_0837
_n28; MSRC001_0839
_n29; MSRC001_083B
_n30; MSRC001_083D
_n31; MSRC001_083F
_n32; MSRC001_0841
_n33; MSRC001_0843
_n34; MSRC001_0845
_n35; MSRC001_0847
_n36; MSRC001_0849
_n37; MSRC001_084B
_n38; MSRC001_084D
_n39; MSRC001_084F
_n40; MSRC001_0851
_n41; MSRC001_0853
_n42; MSRC001_0855
_n43; MSRC001_0857
_n44; MSRC001_0859
_n45; MSRC001_085B
_n46; MSRC001_085D
_n47; MSRC001_085F
_n48; MSRC001_0861
_n49; MSRC001_0863
_n50; MSRC001_0865
_n51; MSRC001_0867
_n52; MSRC001_0869
_n53; MSRC001_086B
_n54; MSRC001_086D
_n55; MSRC001_086F

_n56; MSRC001_0871	
_n57; MSRC001_0873	
_n58; MSRC001_0875	
_n59; MSRC001_0877	
_n60; MSRC001_0879	
_n61; MSRC001_087B	
_n62; MSRC001_087D	
_n63; MSRC001_087F	
Bits	**Description**
63:49	Reserved.
48	**Overflow**. Read-write. Reset: 0. Performance Counter Overflow bit. Write-0-to-clear.
47:0	**Data**. Read-write. Reset: 0000_0000_0000h. Performance Counter Value.

2.1.15.4 MSRs - MSRC001_1xxx

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[11:0]_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[11:0]_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[11:0]_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[11:0]_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[11:0]_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[11:0]_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[11:0]_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 2.1.17 [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[11:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_1030

Bits	Description
63:62	Reserved.
61	**IbsFetchL3Miss**: **L3 cache miss for the sampled fetch**. Read-write,Volatile. Reset: 0. 1=The instruction fetch missed in the L3 Cache on the same CCX.
60	**IbsFetchOcMiss**: **Op cache miss for the sampled fetch**. Read-write,Volatile. Reset: 0. 1=The Op Cache was not able to supply all bytes for the tagged fetch.
59	**IbsL3MissOnly**: **Only L3 miss samples are reported**. Read-write,Volatile. Reset: 0. 1=An IBS interrupt is only created for fetch samples that had an L3 miss; Fetch samples without an L3 miss are discarded and the internal periodic fetch counter is reset to a pseudo random value between 0 and 15. Allows for filtering of Fetch IBS samples based on their L3Miss status.
58	**IbsFetchL2Miss**: **L2 cache miss for the sampled fetch**. Read-write,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-write,Volatile. Reset: 0. 1=The instruction fetch missed in the L2 TLB.
55	**IbsL1TlbMiss**: **instruction cache L1TLB miss**. Read-write,Volatile. Reset: 0. 1=The instruction fetch missed in the L1 TLB.
54:53	**IbsL1TlbPgSz**: **instruction cache L1TLB page size**. Read-write,Volatile. Reset: 0h. Indicates the page size of the translation in the L1 TLB. This field is only valid if IbsPhyAddrValid == 1. **ValidValues**: Value: 0h — 4 KB Value: 1h — 2 MB Value: 2h — 1 GB Value: 3h — 16 KB (Coalesced 4 KB pages)
52	**IbsPhyAddrValid**: **instruction fetch physical address valid**. Read-write,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-write,Volatile. Reset: 0. 1=The instruction fetch missed in either the instruction cache or the Op Cache.

50	**IbsFetchComp**: **instruction fetch complete**. Read-write,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,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-write,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[11:0]_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)**

Read-write,Volatile. Reset: 0000_0000_0000_0000h.	
_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_1032	
Bits	**Description**
63:52	Reserved.
51:0	**IbsFetchPhysAd**: **instruction fetch physical address**. Read-write,Volatile. Reset: 0_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. When nested paging is active, the reported physical address is the system physical address. This register reads zero for a guest with active IBS Virtualization.

MSRC001_1033 [IBS Execution Control] (Core::X86::Msr::IBS_OP_CTL)

Reset: 0000_0000_0000_0000h.

See 2.1.17 [Instruction Based Sampling (IBS)].
The IBS execution sampling engine is described as follows for IbsOpCntCtl == 1. If IbsOpCntCtl == 0 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 and all sample data has been collected:
  - 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[11:0]_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	**IbsOpL3MissOnly**. Read-write. Reset: 0. 1=An IBS interrupt is only created for op samples that had an L3 miss. L3 miss in this context means the data came outside this core's CCX. A hit in another CCX's L3 is considered an L3 miss. Op samples that do not have an L3 miss are dropped and IbsOpCurCnt is reset to a pseudo random value between 0 and 127.
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[11:0]_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[11:0]_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	**IbsOpFuse**: **fused instruction op**. Read-write,Volatile. Reset: 0. 1=Tagged operation was part of a fused instruction pair; IBS_OP_RIP reports the rIP of the older instruction within that pair. Support indicated by Core::X86::Cpuid::IbsIdEax[OpFuse].
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 has zero-cycle latency.
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 64B boundary, the data returned in this register may be for either one of the two cache lines that are accessed.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_1036

Bits	Description
63:8	Reserved.
7:6	**DataSrc[4:3]**: **Northbridge IBS request data source high bits**. Read-write,Volatile. Reset: 0h. High order bits for the DataSrc field. See DataSrcLo for a description of the valid values.
5	**NearFarCache_NearFar**: **Requests that return from any cache**. Read-write,Volatile. Reset: 0. 0=M State. 1=O State. Valid when the data source type is a cache.
4	**FarAny_NearFar**: **Requests that return from another NUMA node**. Read-write,Volatile. Reset: 0. 0=The request is serviced by the Northbridge in the same node as the requesting core. 1=The request is serviced by the Northbridge in a different NUMA node than the requesting core. Valid when DataSrc is non-zero.
3	Reserved.
2:0	**DataSrc[2:0]**: **Northbridge IBS request data source low bits**. Read-write. Reset: 0h.

Description: Valid Values for {DataSrc[4:3],DataSrc[2:0]: }

Values	Description
0h	No valid status.
1h	LocalCcx. Local L3 or different L2 in the same CCX.
2h	NearFarCache_Near. Requests that target the same NUMA node and return from another CCX's cache.
3h	DramIO_Near. Requests that target the same NUMA node and return from DRAM or MMIO.
4h	Reserved.
5h	NearFarCache_Far. Requests that target another NUMA node and return from another CCX's cache.
6h	LongLat_NearFar. Long-latency DIMM.
7h	DramIO_Far. Requests that target another NUMA node and return from DRAM or MMIO.
8h	Ext_NearFar. Extension Memory.
9h-Bh	Reserved.
Ch	Peer_NearFar. Coherent Memory of a different processor type.
Dh-1Fh	Reserved.

MSRC001_1037 [IBS Op Data 3] (Core::X86::Msr::IBS_OP_DATA3)

Read-write,Volatile. Reset: 0000_0000_0000_0000h.

Data in this register is only valid when either Core::X86::Msr::IBS_OP_DATA3[IbsStOp] or Core::X86::Msr::IBS_OP_DATA3[IbsLdOp] are set.
If a load or store operation crosses a 64B boundary, the data returned in this register may be for either one of the two cache lines that are accessed.

_ccd[11:0]_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. This field is only valid when Core::X86::Msr::IBS_OP_DATA3[IbsDcPhyAddrValid] is set.
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 non-load ops (Core::X86::Msr::IBS_OP_DATA3[IbsLdOp]=0) or software prefetch ops (Core::X86::Msr::IBS_OP_DATA3[IbsSwPf]=1).
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 (4 Bytes). 4h: QW (8 Bytes). 5h: OW (16 Bytes). 6h: 32 Bytes. 7h: 64 Bytes. Fh-8h: 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. This is not expected to be set for store operations to non-cacheable memory.
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 64 byte 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. This is not expected to be set for store operations to non-cacheable memory.
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[11:0]_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[11:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_1039

Bits	Description
63:52	Reserved.
51:0	**IbsDcPhysAd**: **load or store physical address**. Read-write,Volatile. Reset: 0_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. For memory accesses, that cross a 64B boundary (Core::X86::Msr::IBS_OP_DATA3[IbsDcMisAcc] is set), this field always points to the first cache line for the access. Cache miss related information in Core::X86::Msr::IBS_OP_DATA3 or Core::X86::Msr::IBS_OP_DATA2 will be for one of the two cache lines accessed by the load or store operation. This field contains valid data only if Core::X86::Msr::IBS_OP_DATA3[IbsDcPhyAddrValid]=1 and Core::X86::Msr::IBS_OP_DATA3[IbsDcLinAddrValid]=1. When nested paging is active, the reported physical address is the system physical address. This register reads zero for a guest with active IBS Virtualization.

MSRC001_103A [IBS Control] (Core::X86::Msr::IBS_CTL)

Read,Error-on-write.

_ccd[11:0]_lthree0_core[7:0]_thread[1:0]; MSRC001_103A

Bits	Description
63:9	Reserved.
8	**LvtOffsetVal**: **IBS Fetch and Op local vector table offset valid**. Read,Error-on-write. Reset: X.
7:4	Reserved.
3:0	**LvtOffset**: **IBS Fetch and Op 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[11:0]_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. It only contains a valid value for branch instructions (Core::X86::Msr::IBS_OP_DATA[IbsOpBrnRet] == 1).

MSRC001_103C [IBS Fetch Control Extended] (Core::X86::Msr::IC_IBS_EXTD_CTL)	
Read-write,Volatile. Reset: 0000_0000_0000_0000h.	
Support for this register indicated by Core::X86::Cpuid::IbsIdEax[IbsFetchCtlExtd].	
_ccd[11:0]_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-write,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.

2.1.15.5 L3 MSRs - MSRC001_1xxx

See 2.2.1 [L3 MSR Registers].

2.1.16 Performance Monitor Counters

2.1.16.1 RDPMC Assignments

There are six core performance event counters per thread, six performance events counters per L3 complex and sixteen Data Fabric performance events counters mapped to the RDPMC instruction as follows:

- The RDPMC[5:0] instruction accesses core events. See 2.1.16.5 [Core Performance Monitor Counters].
- The RDPMC[9:6, 1B:10] instruction accesses data fabric events.
- The RDPMC[F:A] instruction accesses L3 cache events. See 2.1.16.6 [L3 Cache Performance Monitor Counters].

2.1.16.2 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 28 [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 2.1.16.2 [Performance Measurement].

Table 28: Guidance for Common Performance Statistics with Complex Event Selects

Description	Equation

Branch Prediction	
Execution-Time Branch Misprediction Ratio (Non-Speculative).	Event[0x4300C3] / Event[0x4300C2]
Basic Caching	
All Data Cache Accesses	Event[0x430729]
All L2 Cache Accesses	Event[0x43F960] + Event[0x431F70] + Event[0x431F71] + Event[0x431F72]
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[0x431F70] + Event[0x431F71] + Event[0x431F72]
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[0x431F71] + Event[0x431F72]
All L2 Cache Hits	Event[0x43f664] + Event[0x431f70]
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[0x431F70]
L3 Cache Accesses	L3Event[0x0300C0000040FF04]
L3 Miss (includes cacheline state change requests)	L3Event[0x0300C00000400104]
Average L3 Cache Read Miss Latency (in nanoseconds)	L3Event[0x0303C00000403FAC]*10/ L3Event[0x0303C00000403FAD]
Op Cache (64B) Fetch Miss Ratio	Event[0x20043048F] / Event[0x20043078F]
Instruction Cache (32B) Fetch Miss Ratio	Event[0x10043188E] / Event[0x100431F8E]
Advanced Caching	
L1 Data Cache Fills from DRAM or IO in any NUMA node	Event[0x434844]
L1 Data Cache Fills from a different NUMA node	Event[0x435044]
L1 Data Cache Fills from within the same CCX	Event[0x430344]
L1 Data Cache Fills from another CCX cache in any NUMA node	Event[0x431444]
L1 Data Cache Fills All	Event[0x435F44]
Demand L1 Data Cache Fills from local L2	Event[0x430143]
Demand L1 Data Cache Fills from local L3 or different L2 in same CCX	Event[0x430243]
Demand L1 Data Cache Fills from another CCX cache in the same NUMA node	Event[0x430443]
Demand L1 Data Cache Fills from DRAM or MMIO in the same NUMA node	Event[0x430843]
Demand L1 Data Cache Fills from another CCX cache in a different NUMA node	Event[0x431043]
Demand L1 Data Cache Fills from Remote Memory or IO	Event[0x434043]
64B lines written per WCB close	Event[0x430150] / Event[0x432063]
TLBs	
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]
Stalls	
Macro-ops Dispatched	Event[0x4307AA]
Mixed SSE/AVX Stalls	Event[0x430E0E]
Macro-ops Retired	Event[0x4300C1]
Data Fabric	
DRAM read data bytes for local processor	(DfEvent[0x000000000740FE1F] + DfEvent[0x000000000740FE5F] + DfEvent[0x000000000740FE9F] + DfEvent[0x000000000740FEDF] + DfEvent[0x000000010740FE1F] + DfEvent[0x000000010740FE5F] + DfEvent[0x000000010740FE9F] + DfEvent[0x000000010740FEDF] + DfEvent[0x000000020740FE1F] + DfEvent[0x000000020740FE5F] + DfEvent[0x000000020740FE9F] + DfEvent[0x000000020740FEDF]) * 64B
DRAM write data bytes for local processor	(DfEvent[0x000000000740FF1F] + DfEvent[0x000000000740FF5F] + DfEvent[0x000000000740FF9F] + DfEvent[0x000000000740FFDF] + DfEvent[0x000000010740FF1F] + DfEvent[0x000000010740FF5F] + DfEvent[0x000000010740FF9F] + DfEvent[0x000000010740FFDF] + DfEvent[0x000000020740FF1F] + DfEvent[0x000000020740FF5F] + DfEvent[0x000000020740FF9F] + DfEvent[0x000000020740FFDF]) * 64B
DRAM read data bytes for remote processor	(DfEvent[0x000000000B40FE1F] + DfEvent[0x000000000B40FE5F] + DfEvent[0x000000000B40FE9F] + DfEvent[0x000000000B40FEDF] + DfEvent[0x000000010B40FE1F] + DfEvent[0x000000010B40FE5F] + DfEvent[0x000000010B40FE9F] + DfEvent[0x000000010B40FEDF] + DfEvent[0x000000020B40FE1F] + DfEvent[0x000000020B40FE5F] + DfEvent[0x000000020B40FE9F] + DfEvent[0x000000020B40FEDF]) * 64B
DRAM write data bytes for remote processor	(DfEvent[0x000000000B40FF1F] + DfEvent[0x000000000B40FF5F] + DfEvent[0x000000000B40FF9F] + DfEvent[0x000000000B40FFDF] + DfEvent[0x000000010B40FF1F] + DfEvent[0x000000010B40FF5F] + DfEvent[0x000000010B40FF9F] +

	DfEvent[0x000000010B40FFDF] + DfEvent[0x000000020B40FF1F] + DfEvent[0x000000020B40FF5F] + DfEvent[0x000000020B40FF9F] + DfEvent[0x000000020B40FFDF]) * 64B
Local upstream DMA read data bytes	(DfEvent[0x000000080740FE1F] + DfEvent[0x000000080740FE5F] + DfEvent[0x000000080740FE9F] + DfEvent[0x000000080740FEDF]) * 64B
Local upstream DMA write data bytes	(DfEvent[0x000000080740FF1F] + DfEvent[0x000000080740FF5F] + DfEvent[0x000000080740FF9F] + DfEvent[0x000000080740FFDF]) * 64B
Remote socket upstream DMA read data bytes	(DfEvent[0x000000080B40FE1F] + DfEvent[0x000000080B40FE5F] + DfEvent[0x000000080B40FE9F] + DfEvent[0x000000080B40FEDF]) * 64B
Remote socket upstream DMA write data bytes	(DfEvent[0x000000080B40FF1F] + DfEvent[0x000000080B40FF5F] + DfEvent[0x000000080B40FF9F] + DfEvent[0x000000080B40FFDF]) * 64B
Local inbound data bytes to the CPU (e.g. read data)	(DfEvent[0x000000040740FE1E] + DfEvent[0x000000040740FE1F] + DfEvent[0x000000040740FE5E] + DfEvent[0x000000040740FE5F] + DfEvent[0x000000040740FE9E] + DfEvent[0x000000040740FE9F] + DfEvent[0x000000040740FEDE] + DfEvent[0x000000040740FEDF] + DfEvent[0x000000050740FE1E] + DfEvent[0x000000050740FE1F] + DfEvent[0x000000050740FE5E] + DfEvent[0x000000050740FE5F] + DfEvent[0x000000050740FE9E] + DfEvent[0x000000050740FE9F] + DfEvent[0x000000050740FEDE] + DfEvent[0x000000050740FEDF]) * 32B
Local outbound data bytes from the CPU (e.g. write data)	(DfEvent[0x000000040740FF1E] + DfEvent[0x000000040740FF1F] + DfEvent[0x000000040740FF5E] + DfEvent[0x000000040740FF5F] + DfEvent[0x000000040740FF9E] + DfEvent[0x000000040740FF9F] + DfEvent[0x000000040740FFDE] + DfEvent[0x000000040740FFDF] + DfEvent[0x000000050740FF1E] + DfEvent[0x000000050740FF1F] + DfEvent[0x000000050740FF5E] + DfEvent[0x000000050740FF5F] + DfEvent[0x000000050740FF9E] + DfEvent[0x000000050740FF9F] + DfEvent[0x000000050740FFDE] +

	DfEvent[0x000000050740FFDF]) * 64
Remote socket inbound data bytes to the CPU (e.g. read data)	(DfEvent[0x000000040B40FE1E] + DfEvent[0x000000040B40FE1F] + DfEvent[0x000000040B40FE5E] + DfEvent[0x000000040B40FE5F] + DfEvent[0x000000040B40FE9E] + DfEvent[0x000000040B40FE9F] + DfEvent[0x000000040B40FEDE] + DfEvent[0x000000040B40FEDF] + DfEvent[0x000000050B40FE1E] + DfEvent[0x000000050B40FE1F] + DfEvent[0x000000050B40FE5E] + DfEvent[0x000000050B40FE5F] + DfEvent[0x000000050B40FE9E] + DfEvent[0x000000050B40FE9F] + DfEvent[0x000000050B40FEDE] + DfEvent[0x000000050B40FEDF]) * 32B
Remote socket outbound data bytes from the CPU (e.g. write data)	(DfEvent[0x000000040B40FF1E] + DfEvent[0x000000040B40FF1F] + DfEvent[0x000000040B40FF5E] + DfEvent[0x000000040B40FF5F] + DfEvent[0x000000040B40FF9E] + DfEvent[0x000000040B40FF9F] + DfEvent[0x000000040B40FFDE] + DfEvent[0x000000040B40FFDF] + DfEvent[0x000000050B40FF1E] + DfEvent[0x000000050B40FF1F] + DfEvent[0x000000050B40FF5E] + DfEvent[0x000000050B40FF5F] + DfEvent[0x000000050B40FF9E] + DfEvent[0x000000050B40FF9F] + DfEvent[0x000000050B40FFDE] + DfEvent[0x000000050B40FFDF]) * 64B
Outbound data bytes from all links (local socket)	(DfEvent[0x0000000B0F403E5F] + DfEvent[0x0000000B0F403E9F] + DfEvent[0x0000000B0F403EDF] + DfEvent[0x0000000C0F403E1F] + DfEvent[0x0000000C0F403E5F] + DfEvent[0x0000000C0F403E9F] + DfEvent[0x0000000C0F403EDF] + DfEvent[0x0000000D0F403E1F]) * 64B

2.1.16.3 Pipeline Utilization Analysis

Table 29: Guidance for Pipeline Utilization Analysis Statistics

Name	Description	Equation
	Level 1	
Total Dispatch Slots	Up to 6 instructions can be dispatched in one cycle.	6 * Event[430076]
Frontend Bound	Fraction of dispatch slots that remained unused because the frontend did not supply enough	Event[1004301A0] / Total Dispatch Slots

	instructions/ops.	
Bad Speculation	Fraction of dispatched ops that did not retire.	(Event[4307AA] – Event[4300C1]) / Total Dispatch Slots
Backend Bound	Fraction of dispatch slots that remained unused because of backend stalls.	Event[100431EA0] / Total Dispatch Slots
SMT contention	Fraction of unused dispatch slots because the other thread was selected.	Event[1004360A0] / Total Dispatch Slots
Retiring	Fraction of dispatch slots used by ops that retired.	Event[4300C1] / Total Dispatch Slots
	Level 2	
Frontend Bound - Latency	Fraction of dispatch slots that remained unused because of a latency bottleneck in the frontend, such as Instruction Cache or ITLB misses.	6 * Event[1064301A0] / Total Dispatch Slots
Frontend Bound - BW	Fraction of dispatch slots that remained unused because of a bandwidth bottleneck in the frontend, such as decode bandwidth or Op Cache fetch bandwidth.	Event[1004301A0] – (6 * Event[1064301A0]) / Total Dispatch Slots
Bad Speculation – Mispredicts	Fraction of dispatched ops that were flushed due to branch mispredicts.	Bad Speculation * Event[4300C3] / (Event[4300C3] + Event[430096])
Bad Speculation - Pipeline Restarts	Fraction of dispatched ops that were flushed due to pipeline restarts (resyncs).	Bad Speculation * Event[430096]) / (Event[4300C3] + Event[430096]))
Backend Bound - Memory	Fraction of dispatched slots that remained unused because of stalls due to the memory subsystem.	Backend Bound * (Event[43A2D6] / Event[4302D6])
Backend Bound – CPU	Fraction of dispatched slots that remained unused because of stalls not related to the memory subsystem.	Backend Bound * (1 - (Event[43A2D6] / Event[4302D6]))
Retiring - Fastpath	Fraction of dispatch slots used by fastpath ops that retired.	Retiring * (Event[4300C1] – Event[1004300C2]) / Event[4300C1]
Retiring - Microcode	Fraction of dispatch slots used by microcode ops that retired.	Retiring * Event[1004300C2] / Event[4300C1]

2.1.16.4 Large Increment per Cycle Events

Table 30: 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. Both the odd and even numbered counter need to be enabled in Core::X86::Msr::PerfCntrGlobalCtl for the merged counter to count. 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 2.1.16.4 [Large Increment per Cycle Events].

PMCxFFF

Bits	Description
7:0	Reserved.

2.1.16.5 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.

2.1.16.5.1 Floating-Point (FP) Events

PMCx002 [Retired x87 FP Ops] (Core::X86::Pmc::Core::FpRetx87FpOps)

Read-write. Reset: 00h.

The number of x87 floating-point Ops that have retired.

PMCx002

Bits	Description
7:3	Reserved.
2	**DivSqrROps**: **Divide and square root Ops**. Read-write. Reset: 0.
1	**MulOps**: **Multiply Ops**. Read-write. Reset: 0.
0	**AddSubOps**: **Add/subtract Ops**. Read-write. Reset: 0.

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 2.1.16.4 [Large Increment per Cycle Events]. It does not provide a useful count without the use of the MergeEvent.

PMCx003

Bits	Description
7:5	Reserved.
4	**BfloatMacFLOPs**: **bfloat Multiply-Accumulate FLOPs**. Read-write. Reset: 0. Each bfloat MAC operation is counted as 2 FLOPS.
3	**MacFLOPs**: **Multiply-Accumulate FLOPs**. Read-write. Reset: 0. Each MAC operation is counted as 2 FLOPS. This event does not include bfloat MAC operations.
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.

PMCx008 [Retired FP Ops By Width] (Core::X86::Pmc::Core::FpOpsRetiredByWidth)

Read-write. Reset: 00h.

PMCx008

Bits	Description
7:6	Reserved.
5	**Pack512uOpsRetired**. Read-write. Reset: 0. Number of packed 512-bit ops retired.
4	**Pack256uOpsRetired**. Read-write. Reset: 0. Number of packed 256-bit ops retired.
3	**Pack128uOpsRetired**. Read-write. Reset: 0. Number of packed 128-bit ops retired.
2	**ScalaruOpsRetired**. Read-write. Reset: 0. Number of scalar ops retired.
1	**MMXuOpsRetired**. Read-write. Reset: 0. Number of MMX ops retired.
0	**x87uOpsRetired**. Read-write. Reset: 0. Number of x87 ops retired.

PMCx00A [Retired FP Ops By Type] (Core::X86::Pmc::Core::FpOpsRetiredByType)

Read-write. Reset: 00h.

Note: Shuffle op counts may count for instructions that are not necessarily thought of as including shuffles. For example, Horizontal Add, Dot Product, and certain MOV instructions may include or use only shuffle type ops.

PMCx00A

Bits	Description
7:4	**VectorFpOpType**. Read-write. Reset: 0h. **ValidValues**:

Value	Description
0h	None
1h	Add
2h	Sub
3h	Mul
4h	Mac
5h	Div
6h	Sqrt
7h	Cmp
8h	Cvt
9h	Blend
Ah	Reserved.
Bh	Shuffle
Ch	Reserved.
Dh	Logical
Eh	Other
Fh	All

3:0 **ScalarFpOpType**. Read-write. Reset: 0h.

ValidValues:

Value	Description
0h	None
1h	Add
2h	Sub
3h	Mul
4h	Mac
5h	Div
6h	Sqrt
7h	Cmp
8h	Cvt
9h	Blend
Dh-Ah	Reserved.
Eh	Other
Fh	All

PMCx00B [INT Ops Retired] (Core::X86::Pmc::Core::SseAvxOpsRetired)

Read-write. Reset: 00h.

Note: Shuffle op counts may count for instructions that are not necessarily thought of as including shuffles. For example, Horizontal Add, Dot Product, and certain MOV instructions may include or use only shuffle type ops.

PMCx00B

Bits	Description
7:4	**SseAvxOpType**. Read-write. Reset: 0h. **ValidValues**:

Value	Description
0h	None
1h	Add
2h	Sub
3h	Mul
4h	Mac
5h	AES
6h	SHA
7h	Cmp
8h	CLM
9h	Shift
Ah	Mov
Bh	Shuffle
Ch	Pack
Dh	Logical
Eh	Other
Fh	All

Bits	Description
3:0	**MmxOpType**. Read-write. Reset: 0h. **ValidValues**:

Value	Description
0h	None
1h	Add
2h	Sub
3h	Mul
4h	Mac
6h-5h	Reserved.
7h	Cmp
8h	Reserved.
9h	Shift
Ah	Mov
Bh	Shuffle
Ch	Pack
Dh	Logical
Eh	Other
Fh	All

PMCx00C [Packed FP Ops Retired] (Core::X86::Pmc::Core::FpPackOpsRetired)

Read-write. Reset: 00h.

Note: Shuffle op counts may count for instructions that are not necessarily thought of as including shuffles. For example, Horizontal Add, Dot Product, and certain MOV instructions may include or use only shuffle type ops.

PMCx00C

Bits	Description
7:4	**Fp256OpType**. Read-write. Reset: 0h. **ValidValues**:

Value	Description
0h	None
1h	Add
2h	Sub
3h	Mul
4h	Mac
5h	Div
6h	Sqrt
7h	Cmp
8h	Cvt
9h	Blend
Ah	Reserved.
Bh	Shuffle
Ch	Reserved.
Dh	Logical
Eh	Other
Fh	All

3:0 **Fp128OpType**. Read-write. Reset: 0h.

ValidValues:

Value	Description
0h	None
1h	Add
2h	Sub
3h	Mul
4h	Mac
5h	Div
6h	Sqrt
7h	Cmp
8h	Cvt
9h	Blend
Ah	Reserved.
Bh	Shuffle
Ch	Reserved.
Dh	Logical
Eh	Other
Fh	All

PMCx00D [Packed INT Ops Retired] (Core::X86::Pmc::Core::PackedIntOpType)

Read-write. Reset: 00h.

Note: Shuffle op counts may count for instructions that are not necessarily thought of as including shuffles. For example, Horizontal Add, Dot Product, and certain MOV instructions may include or use only shuffle type ops.

PMCx00D

Bits	Description
7:4	**Int256OpType**. Read-write. Reset: 0h. This event also counts FP data type packed and scalar MOV and shuffle operations. **ValidValues**:

Value	Description
0h	None
1h	Add
2h	Sub
3h	Mul
4h	Mac
6h-5h	Reserved.
7h	Cmp
8h	Reserved.
9h	Shift
Ah	Mov
Bh	Shuffle
Ch	Pack
Dh	Logical
Eh	Other
Fh	All

Bits	Description
3:0	**Int128OpType**. Read-write. Reset: 0h. This event also counts FP data type packed and scalar MOV and shuffle operations. **ValidValues**:

Value	Description
0h	None
1h	Add
2h	Sub
3h	Mul
4h	Mac
5h	AES
6h	SHA
7h	Cmp
8h	CLM
9h	Shift
Ah	Mov
Bh	Shuffle
Ch	Pack
Dh	Logical
Eh	Other
Fh	All

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.

2.1.16.5.2 Load/Store (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 an op that performs a load from and store to the same memory address.
1	**StoreDispatch**. Read-write. Reset: 0. Dispatch of an op that performs a memory store.
0	**LdDispatch**. Read-write. Reset: 0. Dispatch of an op that performs a memory load.

PMCx02B [SMIs Received] (Core::X86::Pmc::Core::LsSmiRx)

Read-only. Reset: 00h.

Counts the number of SMIs received.

PMCx02B

Bits	Description
7:0	Reserved.

PMCx02C [Interrupts Taken] (Core::X86::Pmc::Core::LsIntTaken)

Read-write. Reset: 00h.

Counts the number of interrupts taken.

PMCx02C

Bits	Description
7:1	Reserved.
0	**IntTaken**. Read-write. Reset: 0. Number of Interrupts taken. This event is also counted when UnitMask[7:0]=0.

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	**AlternateMemories_NearFar**. Read-write. Reset: 0. Requests that return from Extension Memory.
6	**DramIO_Far**. Read-write. Reset: 0. Requests that target another NUMA node and return from DRAM or MMIO.
5	Reserved.
4	**NearFarCache_Far**. Read-write. Reset: 0. Requests that target another NUMA node and return from another CCX's cache.
3	**DramIO_Near**. Read-write. Reset: 0. Requests that target the same NUMA node and return from DRAM or MMIO.
2	**NearFarCache_Near**. Read-write. Reset: 0. Requests that target the same NUMA node and return from another CCX's cache.
1	**LocalCcx**. Read-write. Reset: 0. Data returned from L3 or different L2 in the same CCX.
0	**LocalL2**. Read-write. Reset: 0. Data returned from local L2.

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	**AlternateMemories_NearFar**. Read-write. Reset: 0. Requests that return from Extension Memory.
6	**DramIO_Far**. Read-write. Reset: 0. Requests that target another NUMA node and return from DRAM.
5	Reserved.
4	**NearFarCache_Far**. Read-write. Reset: 0. Requests that target another NUMA node and return from another CCX's cache.
3	**DramIO_Near**. Read-write. Reset: 0. Requests that target the same NUMA node and return from DRAM or MMIO.
2	**NearFarCache_Near**. Read-write. Reset: 0. Requests that target the same NUMA node and return from another CCX's cache.
1	**LocalCcx**. Read-write. Reset: 0. Data returned from L3 or different L2 in the same CCX.
0	**LocalL2**. Read-write. Reset: 0. Data returned from local L2.

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 1-G page that also missed in the L2 TLB.
6	**TlbReload2ML2Miss**. Read-write. Reset: 0. DTLB reload to a 2-M 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 4-K page that missed the L2 TLB.
3	**TlbReload1GL2Hit**. Read-write. Reset: 0. DTLB reload to a 1-G page that hit in the L2 TLB.
2	**TlbReload2ML2Hit**. Read-write. Reset: 0. DTLB reload to a 2-M 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 4-K 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 4-KB misaligned (i.e., page crossing) loads.
0	**MA64**. Read-write. Reset: 0. The number of 64-B 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.

PMCx050 [Write Combine Buffer Close Flush] (Core::X86::Pmc::Core::LsWcbCloseFlush)

Read-write. Reset: 00h.

UnitMask events ADDed.
Multible WCB can report events at the same time.

PMCx050

Bits	Description
7:1	Reserved.
0	**FullLine64B**. Read-write. Reset: 0. All 64 bytes of the WCB entry have been written.

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	**AlternateMemories_NearFar**. Read-write. Reset: 0. Requests that return from Extension Memory.
6	**DramIO_Far**. Read-write. Reset: 0. Requests that target another NUMA node and return from DRAM or MMIO.
5	Reserved.
4	**NearFarCache_Far**. Read-write. Reset: 0. Requests that target another NUMA node and return from another CCX's cache.
3	**DramIO_Near**. Read-write. Reset: 0. Requests that target the same NUMA node and return from DRAM or MMIO.
2	**NearFarCache_Near**. Read-write. Reset: 0. Requests that target the same NUMA node and return from another CCX's cache.
1	**LocalCcx**. Read-write. Reset: 0. Data returned from L3 or different L2 in the same CCX.
0	**LocalL2**. Read-write. Reset: 0. Data returned from local L2.

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	**AlternateMemories_NearFar**. Read-write. Reset: 0. Requests that return from Extension Memory.
6	**DramIO_Far**. Read-write. Reset: 0. Requests that target another NUMA node and return from DRAM or MMIO.
5	Reserved.
4	**NearFarCache_Far**. Read-write. Reset: 0. Requests that target another NUMA node and return from another CCX's cache.
3	**DramIO_Near**. Read-write. Reset: 0. Requests that target the same NUMA node and return from DRAM or MMIO.
2	**NearFarCache_Near**. Read-write. Reset: 0. Requests that target the same NUMA node and return from another CCX's cache.
1	**LocalCcx**. Read-write. Reset: 0. Data returned from L3 or different L2 in the same CCX.
0	**LocalL2**. Read-write. Reset: 0. Data returned from local L2.

PMCx05F [Count of Allocated Mabs] (Core::X86::Pmc::Core::LsAllocMabCount)

This event counts the in-flight L1 data cache misses (allocated Miss Address Buffers) each cycle.

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.

PMCx078

Bits	Description
7:0	**All**. Read-write. Reset: 00h. **ValidValues**: Value: FEh-00h — Reserved. Value: FFh — All TLB Flushes.

PMCx120 [P0 Freq Cycles not in Halt] (Core::X86::Pmc::Core::LsNotHaltedP0Cyc)

Read-write. Reset: 00h.

PMCx120

Bits	Description
7:1	Reserved.
0	**P0FreqCyc**. Read-write. Reset: 0. Counts at the P0 frequency (same as Core::X86::Msr::MPERF) when not in Halt.

2.1.16.5.3 Instruction Cache (IC) and Branch Prediction (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 lines 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 >4-K Coalesced page.
2	**IF1G**. Read-write. Reset: 0. Walk for 1-G page.
1	**IF2M**. Read-write. Reset: 0. Walk for 2-M page.
0	**IF4K**. Read-write. Reset: 0. Walk to 4-K 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 (1-G page size).
1	**IF2M**. Read-write. Reset: 0. L1 Instruction TLB hit (2-M page size).
0	**IF4K**. Read-write. Reset: 0. L1 Instruction TLB hit (4-K or 16-K page size).

PMCx096 [Resyncs] (Core::X86::Pmc::Core::ResyncsOrNcRedirects)

Counts the number of HW resyncs (pipeline restarts) or NC redirects. NC redirects occur when the front-end transitions to fetching from UC (un-cacheable) memory.

PMCx096

Bits	Description
7:0	Reserved.

PMCx188 [Fetch IBS events] (Core::X86::Pmc::Core::IcFetchIbs)

Read-write. Reset: 00h.

Counts Fetch IBS related events.

PMCx188

Bits	Description
7:5	Reserved.
4	**SampleVal**. Read-write. Reset: 0. Counts collected IBS samples. Each collected IBS sample signals an interrupt.
3	**SampleFiltered**. Read-write. Reset: 0. Counts the number of tagged fetches that were discarded due to IBS filtering (Core::X86::Msr::IBS_FETCH_CTL[IbsL3MissOnly] set). When a tagged fetch is discarded the Fetch IBS facility will automatically tag a new fetch.
2	**SampleDiscarded**. Read-write. Reset: 0. Counts when the Fetch IBS facility discards an IBS tagged fetch for reasons other than IBS filtering. When a tagged fetch is discarded the Fetch IBS facility will automatically tag a new fetch.
1	**FetchTagged**. Read-write. Reset: 0. Counts the number of fetches tagged for Fetch IBS. Not all tagged fetches create an IBS interrupt and valid fetch sample.
0	Reserved.

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.

2.1.16.5.4 DE Events

PMCx0A9 [Op Queue Empty] (Core::X86::Pmc::Core::DeOpQueueEmpty)

Reset: 00h.

Cycles where the Op Queue is empty.

PMCx0A9

Bits	Description
7:0	Reserved.

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.

PMCx0AA

Bits	Description
7:3	Reserved.
2	**LoopBuffer**. Read-write. Reset: 0. Count of ops dispatched from Loop Buffer.
1	**OpCache**. Read-write. Reset: 0. Count of ops fetched from Op Cache and dispatched.
0	**Decoder**. Read-write. Reset: 0. Count of ops fetched from Instruction Cache and dispatched.

PMCx0AB [Types of Ops Dispatched From Decoder] (Core::X86::Pmc::Core::DeDisOpsFromDecoder)

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

Bits	Description
7:5	Reserved.
4:0	**DispOpType**. Read-write. Reset: 00h.

ValidValues:

Value	Description
03h-00h	Reserved.
04h	Any FP dispatch.
07h-05h	Reserved.
08h	Any Integer dispatch.
1Fh-09h	Reserved.

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. UnitMask bits select the stall types included in the count.

PMCx0AE

Bits	Description
7	**FpFlushRecoveryStall**. Read-write. Reset: 0. Counts FP Flush Recovery stall cycles.
6	**FPSchRsrcStall**: **FP scheduler resource stall**. Read-write. Reset: 0. Counts FP Scheduler token stall cycles. .
5	**FpRegFileRsrcStall**: **floating-point register file resource stall**. Read-write. Reset: 0. Counts FP Register File token stall cycles. This applies to all ops that have an FP or SIMD destination register. .
4	**TakenBrnchBufferRsrc**: **taken branch buffer resource stall**. Read-write. Reset: 0. Counts Taken Branch Buffer token stall cycles.
3	Reserved.
2	**StoreQueueRsrcStall**: **Store Queue resource stall**. Read-write. Reset: 0. Counts Store Queue token stall cycles.
1	**LoadQueueRsrcStall**: **Load Queue resource stall**. Read-write. Reset: 0. Counts Load Queue token stall cycles.
0	**IntPhyRegFileRsrcStall**: **Integer Physical Register File resource stall**. Read-write. Reset: 0. Counts Integer PRF token stall cycles. This 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. UnitMask bits select the stall types included in the count.

PMCx0AF

Bits	Description
7:6	Reserved.
5	**RetireTokenStall**. Read-write. Reset: 0. Counts Retire Queue token stall cycles.
4	Reserved.
3	**IntSch3TokenStall**. Read-write. Reset: 0. Counts Integer Scheduler Queue 3 token stall cycles.
2	**IntSch2TokenStall**. Read-write. Reset: 0. Counts Integer Scheduler Queue 2 token stall cycles.
1	**IntSch1TokenStall**. Read-write. Reset: 0. Counts Integer Scheduler Queue 1 token stall cycles.
0	**IntSch0TokenStall**. Read-write. Reset: 0. Counts Integer Scheduler Queue 0 token stall cycles.

PMCx1A0 [Dispatch Stalls Per Slot] (Core::X86::Pmc::Core::DeNoDispatchPerSlot)

Read-write. Reset: 00h.

Counts the number of dispatch slots (each cycle) that remained unused for reasons selected by StallReason.

PMCx1A0

Bits	Description
7:0	**StallReason**. Read-write. Reset: 00h. **ValidValues**:

Value	Description
00h	Reserved.
01h	Counts dispatch slots left empty because the front-end did not supply ops.
1Dh-02h	Reserved.
1Eh	Counts ops unable to dispatch due to back-end stalls.
5Fh-1Fh	Reserved.
60h	Counts ops unable to dispatch because the dispatch cycle was granted to the other SMT thread.
FFh-61h	Reserved.

PMCx1A2 [Dispatch Additional Resource Stalls] (Core::X86::Pmc::Core::DeAdditionalResourceStalls)

Read-write. Reset: 00h.

This PMC event counts additional resource stalls that are not captured by Core::X86::Pmc::Core::DeDisDispatchTokenStalls1 or Core::X86::Pmc::Core::DeDisDispatchTokenStalls2.

PMCx1A2

Bits	Description
7:0	**Stall**. Read-write. Reset: 00h. **ValidValues**: Value: 2Fh-00h — Reserved. Value: 30h — Counts additional cycles dispatch is stalled due to the lack of dispatch resources Value: FFh-31h — Reserved.

2.1.16.5.5 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.

PMCx0D6 [Cycles With No Retire] (Core::X86::Pmc::Core::ExNoRetire)

Read-write. Reset: 00h.

This event counts cycles when the hardware thread does not retire any ops for reasons selected by UnitMask[4:0]. UnitMask events [4:0] are mutually exclusive. If multiple reasons apply for a given cycle, the lowest numbered UnitMask event is counted.

PMCx0D6

Bits	Description
7:5	**CompletionFilter**. Read-write. Reset: 0h. **ValidValues**: Value / Description 0h: Load and ALU completion is considered for UnitMask[1]:NotComplete events. 4h-1h: Reserved. 5h: Only missing load completion is considered for UnitMask[1]:NotComplete events. 7h-6h: Reserved.
4	**ThreadNotSelected**. Read-write. Reset: 0. The number cycles where ops could have retired (i.e. did not fall into UnitMask events [0]...[3]). but did not retire because thread arbitration did not select the thread for retire.
3	**Other**. Read-write. Reset: 0. The number of cycles where ops could have retired (self and older ops are complete), but were stopped from retirement for other reasons: retire breaks, traps, faults, etc.
2	Reserved.
1	**NotComplete**. Read-write. Reset: 0. The number of cycles where the oldest retire slot did not have its completion bits set.
0	**Empty**. Read-write. Reset: 0. The number of cycles when there were no valid ops in the retire queue. This may be caused by front-end bottlenecks or pipeline redirects.

PMCx1C1 [Retired Microcoded Instructions] (Core::X86::Pmc::Core::ExRetUcodeInstr)

Retired Microcoded Instructions.

PMCx1C1

Bits	Description
7:0	Reserved.

PMCx1C2 [Retired Microcode Ops] (Core::X86::Pmc::Core::ExRetUcodeOps)

The number of microcode ops that have retired.

PMCx1C2

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.

PMCx1C8 [Retired Unconditional Indirect Branch Instructions Mispredicted] (Core::X86::Pmc::Core::ExRetUncondBrnchInstrMispred)

The number of retired unconditional indirect branch instructions that were mispredicted.

PMCx1C8

Bits	Description
7:0	Reserved.

PMCx1C9 [Retired Unconditional Branch Instructions] (Core::X86::Pmc::Core::ExRetUncondBrnchInstr)

The number of retired unconditional branch instructions.

PMCx1C9

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)

Counts retired fused instructions.

PMCx1D0

Bits	Description
7:0	Reserved.

2.1.16.5.6 L2 Cache Events

PMCx060 [Requests to L2 Group1] (Core::X86::Pmc::L2::L2RequestG1)

Read-write.

All L2 Cache Requests (Breakdown 1 - Common)

PMCx060

Bits	Description
7	**RdBlkL**. Read-write. Data Cache Reads (including hardware and software prefetch).
6	**RdBlkX**. Read-write. Data Cache Stores
5	**LsRdBlkC_S**. Read-write. Data Cache Shared Reads
4	**CacheableIcRead**. Read-write. Instruction Cache Reads.
3	**ChangeToX**: **Data Cache State Change Requests**. Read-write. Request change to writable, check L2 for current state.
2	**PrefetchL2Cmd**. Read-write.
1	**L2HwPf**: **L2 Prefetcher**. Read-write. All prefetches accepted by L2 pipeline, hit or miss. Types of PF and L2 hit/miss broken out in a separate perfmon event
0	**Group2**. Read-write. MiscRequests. Read-write. Various Noncacheable requests. Non-cached Data Reads, Non-cached Instruction Reads, Self-modifying code checks.

PMCx061 [Requests to L2 Group2] (Core::X86::Pmc::L2::L2RequestG2)

Read-write.

All L2 Cache Requests (Breakdown 2 - Rare).

PMCx061

Bits	Description
7	Reserved.
6	**LsRdSized**. Read-write. LS sized read, coherent non-cacheable.
5	**LsRdSizedNC**. Read-write. LS sized read, non-coherent, non-cacheable.
4:0	Reserved.

PMCx063 [Write Combining Buffer Requests] (Core::X86::Pmc::L2::L2WcbReq)

Read-write.

Write Combining Buffer operations. For information on Write Combining see docAPM2 sections: Memory System, Memory Types, Buffering and Combining Memory Writes.

PMCx063

Bits	Description
7	Reserved.
6	**WcbWrite**. Read-write. Write Combining Buffer write
5	**WcbClose**. Read-write. Write Combining Buffer close
4:0	Reserved.

PMCx064 [Core to L2 Cacheable Request Access Status] (Core::X86::Pmc::L2::L2CacheReqStat)

Read-write.

L2 Cache Request Outcomes (not including L2 Prefetch).

PMCx064

Bits	Description
7	**LsRdBlkCS**: **Data Cache Shared Read Hit in L2**. Read-write. LsRdBlkCS
6	**LsRdBlkLHitX**: **Data Cache Read Hit in L2**. Read-write. Modifiable
5	**LsRdBlkLHitS**: **Data Cache Read Hit Non-Modifiable Line in L2**. Read-write.
4	**LsRdBlkX**: **Data Cache Store or State Change Hit in L2**. Read-write.
3	**LsRdBlkC**: **Data Cache Req Miss in L2**. Read-write.
2	**IcFillHitX**: **Instruction Cache Hit Modifiable Line in L2**. Read-write. IcFillHitX
1	**IcFillHitS**: **Instruction Cache Hit Non-Modifiable Line in L2.**. Read-write.
0	**IcFillMiss**: **Instruction Cache Req Miss in L2**. Read-write. IcFillMiss

PMCx070 [L2 Prefetch Hit in L2] (Core::X86::Pmc::L2::L2PfHitL2)

Read-write.

Counts all L2 prefetches accepted by L2 pipeline which hit in the L2 cache.

PMCx070

Bits	Description
7:0	**Prefetches**. Read-write. **ValidValues**:

Value	Description
1Eh-00h	Reserved.
1Fh	Counts requests generated from L2 Hardware Prefetchers.
DFh-20h	Reserved.
E0h	Counts requests generated from L1 DC Hardware Prefetchers.
FEh-E1h	Reserved.
FFh	Counts requests generated from L1 DC and L2 Hardware Prefetchers.

PMCx071 [L2 Prefetcher Hits in L3] (Core::X86::Pmc::L2::L2PfMissL2HitL3)

Read-write.

Counts all L2 prefetches accepted by the L2 pipeline which miss the L2 cache and hit the L3.

PMCx071

Bits	Description
7:0	**Prefetches**. Read-write. L2Stream **ValidValues**:

Value	Description
1Eh-00h	Reserved.
1Fh	Counts requests generated from L2 Hardware Prefetchers.
DFh-20h	Reserved.
E0h	Counts requests generated from L1 DC Hardware Prefetchers.
FEh-E1h	Reserved.
FFh	Counts requests generated from L1 DC and L2 Hardware Prefetchers.

PMCx072 [L2 Prefetcher Misses in L3] (Core::X86::Pmc::L2::L2PfMissL2L3)

Read-write.

Counts all L2 prefetches accepted by the L2 pipeline which miss the L2 and the L3 caches

PMCx072

Bits	Description
7:0	**Prefetches**. Read-write. L2Stream **ValidValues**:

Value	Description
1Eh-00h	Reserved.
1Fh	Counts requests generated from L2 Hardware Prefetchers.
DFh-20h	Reserved.
E0h	Counts requests generated from L1 DC Hardware Prefetchers.
FEh-E1h	Reserved.
FFh	Counts requests generated from L1 DC and L2 Hardware Prefetchers.

PMCx165 [L2 Fill Response Source] (Core::X86::Pmc::L2::L2FillRspSrc)

Read-write.

Counts fill responses based on their source. Selecting an event mask of 0xfe will count all L3 responses.
This will count all L3 responses to fill requests.
This event is similar to LS PMC 0x44

PMCx165

Bits	Description
7	**AlternateMemories_NearFar**. Read-write. "Requests that return from Extension Memory"
6	**DramIO_Far**. Read-write. Requests that target another NUMA node and return from either DRAM or MMIO from another NUMA node, either from the same or different NUMA node.
5	Reserved.
4	**NearFarCache_Far**. Read-write. Requests that target another NUMA node and return from another CCX's cache.
3	**DramIO_Near**. Read-write. Requests that target the same NUMA node and return from either DRAM or MMIO from the same NUMA node.
2	**NearFarCache_Near**. Read-write. Requests that target the same NUMA node and return from another CCX's cache.
1	**LocalCcx**. Read-write. Data returned from L3 or different L2 in the same CCX.
0	Reserved.

2.1.16.6 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.

2.1.16.6.1 L3 Cache PMC Events

L3PMCx04 [L3 tag lookup state] (Core::X86::Pmc::L3::L3LookupState)

Read-write.

All L3 Requests.

L3PMCx04

Bits	Description
7:0	**L3LookupMask**. Read-write. L3 Request Types **ValidValues**: **Value** / **Description** 00h: Reserved. 01h: L3 Miss FDh-02h: Reserved. FEh: L3 Hit FFh: All coherent accesses to L3

L3PMCxAC [L3 XiSampledLatency] (Core::X86::Pmc::L3::L3_XiSampledLatency)

Read-write.

When used in conjunction with L3_XiSampledLatencyRequests, this PMC Event will measure the average memory latency (excluding MMIO) observed by this CCX.
Configure two PMCs with the L3_XiSampledLatency and L3_XiSampledLatencyRequests events and use the following equation to identify the observed latency.

if (L3Size-per-CCX >= 32MB)
L3LatScalingFactor=10
else
L3LatScalingFactor=30
end

Average Sampled Latency = L3_XiSampledLatency/L3_XiSampledLatencyRequests * L3LatScalingFactor ns

Some ChL3PmcCfg fields must be programmed as follows to ensure that these events accurately measure latency: ChL3PmcCfg[EnAllSlices]=0x1 and ChL3PmcCfg[SliceId]=0x3.

Other ChL3PmcCfg fields can be used to filter the measured latency based on originating thread (EnAllCores, CoreID) and Data Source (UnitMask).
To measure average latency from all threads to all Data Sources, use the following configuration: ChL3PmcCfg[EnAllCores]=0x1, ChL3PmcCfg[ThreadMask]=0x3, and ChL3PmcCfg[UnitMask]=0xFF.

L3PMCxAC

Bits	Description
7:6	Reserved.
5	**Ext_Far**. Read-write. Requests that target another NUMA node and return from Extension Memory (CXL™)
4	**Ext_Near**. Read-write. Requests that target the same NUMA node and return from Extension Memory (CXL)
3	**NearCache_FarCache_Far**. Read-write. Requests that target another NUMA node and return from another CCX's cache.
2	**NearCache_FarCache_Near**. Read-write. Requests that target the same NUMA node and return from another CCX's cache.
1	**Dram_Far**. Read-write. Requests that target another NUMA node and return from DRAM
0	**Dram_Near**. Read-write. Requests that target the same NUMA node and return from DRAM

L3PMCxAD [L3 XiSampledLatencyRequests] (Core::X86::Pmc::L3::L3_XiSampledLatencyRequests)

Read-write.

When used in conjunction with L3_XiSampledLatency, this PMC Event will measure the average memory latency (excluding MMIO) observed by this CCX.
Configure two PMCs with the L3_XiSampledLatency and L3_XiSampledLatencyRequests events and use the following equation to identify the observed latency.

if (L3Size-per-CCX >= 32MB)
L3LatScalingFactor=10
else
L3LatScalingFactor=30
end

Average Sampled Latency = L3_XiSampledLatency/L3_XiSampledLatencyRequests * L3LatScalingFactor ns

Some ChL3PmcCfg fields must be programmed as follows to ensure that these events accurately measure latency: ChL3PmcCfg[EnAllSlices]=0x1 and ChL3PmcCfg[SliceId]=0x3.

Other ChL3PmcCfg fields can be used to filter the measured latency based on originating thread (EnAllCores, CoreID) and Data Source (UnitMask).
To measure average latency from all threads to all Data Sources, use the following configuration: ChL3PmcCfg[EnAllCores]=0x1, ChL3PmcCfg[ThreadMask]=0x3, and ChL3PmcCfg[UnitMask]=0xFF.

L3PMCxAD

Bits	Description
7:6	Reserved.
5	**Ext_Far**. Read-write. Requests that target another NUMA node and return from Extension Memory (CXL)
4	**Ext_Near**. Read-write. Requests that target the same NUMA node and return from Extension Memory (CXL)
3	**NearCache_FarCache_Far**. Read-write. Requests that target another NUMA node and return from another CCX's cache.
2	**NearCache_FarCache_Near**. Read-write. Requests that target the same NUMA node and return from another CCX's cache.
1	**Dram_Far**. Read-write. Requests that target another NUMA node and return from DRAM
0	**Dram_Near**. Read-write. Requests that target the same NUMA node and return from DRAM

2.1.17 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.

2.2 L3 Cache

The Level-3 cache (L3) forms the third level of cache in the CPU caching hierarchy. The L3 is a shared, unified cache inside a core complex.

2.2.1 L3 MSR Registers

MSR0000_0C81 [L3 QoS Configuration] (L3::L3CRB::L3QosCfg1)

Reset: 0000_0000_0000_0000h.

QOS L3 Cache Allocation CDP mode enable (I vs. D). Contents are copied to ChL2QosCfg1 and ChL3QosCfg1_0.

_ccd[11:0]_lthree0; MSR0000_0C81

Bits	Description
63:1	Reserved.
0	**CDP**. Read-write. Reset: 0. Code and Data Prioritization Technology enable

MSR0000_0C90 [L3 QoS Allocation Mask] (L3::L3CRB::L3QosAllocMask0)

Reset: 0000_0000_0000_FFFFh.

QOS L3 Allocation Mask for CLOS0

_ccd[11:0]_lthree0; MSR0000_0C90

Bits	Description
63:16	Reserved.
15:0	**WayMask**. Read-write. Reset: FFFFh. L3 way mask used for allocation control.

MSR0000_0C91 [L3 QoS Allocation Mask] (L3::L3CRB::L3QosAllocMask1)

Reset: 0000_0000_0000_FFFFh.

QOS L3 Allocation Mask for CLOS1

_ccd[11:0]_lthree0; MSR0000_0C91

Bits	Description
63:16	Reserved.
15:0	**WayMask**. Read-write. Reset: FFFFh. L3 way mask used for allocation control.

MSR0000_0C92 [L3 QoS Allocation Mask] (L3::L3CRB::L3QosAllocMask2)

Reset: 0000_0000_0000_FFFFh.

QOS L3 Allocation Mask for CLOS2

_ccd[11:0]_lthree0; MSR0000_0C92

Bits	Description
63:16	Reserved.
15:0	**WayMask**. Read-write. Reset: FFFFh. L3 way mask used for allocation control.

MSR0000_0C93 [L3 QoS Allocation Mask] (L3::L3CRB::L3QosAllocMask3)

Reset: 0000_0000_0000_FFFFh.

QOS L3 Allocation Mask for CLOS3

_ccd[11:0]_lthree0; MSR0000_0C93

Bits	Description
63:16	Reserved.
15:0	**WayMask**. Read-write. Reset: FFFFh. L3 way mask used for allocation control.

MSR0000_0C94 [L3 QoS Allocation Mask] (L3::L3CRB::L3QosAllocMask4)

Reset: 0000_0000_0000_FFFFh.

QOS L3 Allocation Mask for CLOS4

_ccd[11:0]_lthree0; MSR0000_0C94

Bits	Description
63:16	Reserved.
15:0	**WayMask**. Read-write. Reset: FFFFh. L3 way mask used for allocation control.

MSR0000_0C95 [L3 QoS Allocation Mask] (L3::L3CRB::L3QosAllocMask5)

Reset: 0000_0000_0000_FFFFh.

QOS L3 Allocation Mask for CLOS5

_ccd[11:0]_lthree0; MSR0000_0C95

Bits	Description
63:16	Reserved.
15:0	**WayMask**. Read-write. Reset: FFFFh. L3 way mask used for allocation control.

MSR0000_0C96 [L3 QoS Allocation Mask] (L3::L3CRB::L3QosAllocMask6)

Reset: 0000_0000_0000_FFFFh.

QOS L3 Allocation Mask for CLOS6

_ccd[11:0]_lthree0; MSR0000_0C96

Bits	Description
63:16	Reserved.
15:0	**WayMask**. Read-write. Reset: FFFFh. L3 way mask used for allocation control.

MSR0000_0C97 [L3 QoS Allocation Mask] (L3::L3CRB::L3QosAllocMask7)

Reset: 0000_0000_0000_FFFFh.

QOS L3 Allocation Mask for CLOS7

_ccd[11:0]_lthree0; MSR0000_0C97

Bits	Description
63:16	Reserved.
15:0	**WayMask**. Read-write. Reset: FFFFh. L3 way mask used for allocation control.

MSR0000_0C98 [L3 QoS Allocation Mask] (L3::L3CRB::L3QosAllocMask8)

Reset: 0000_0000_0000_FFFFh.

QOS L3 Allocation Mask for CLOS8

_ccd[11:0]_lthree0; MSR0000_0C98

Bits	Description
63:16	Reserved.
15:0	**WayMask**. Read-write. Reset: FFFFh. L3 way mask used for allocation control.

MSR0000_0C99 [L3 QoS Allocation Mask] (L3::L3CRB::L3QosAllocMask9)

Reset: 0000_0000_0000_FFFFh.

QOS L3 Allocation Mask for CLOS9

_ccd[11:0]_lthree0; MSR0000_0C99

Bits	Description
63:16	Reserved.
15:0	**WayMask**. Read-write. Reset: FFFFh. L3 way mask used for allocation control.

MSR0000_0C9A [L3 QoS Allocation Mask] (L3::L3CRB::L3QosAllocMask10)

Reset: 0000_0000_0000_FFFFh.

QOS L3 Allocation Mask for CLOS10

_ccd[11:0]_lthree0; MSR0000_0C9A

Bits	Description
63:16	Reserved.
15:0	**WayMask**. Read-write. Reset: FFFFh. L3 way mask used for allocation control.

MSR0000_0C9B [L3 QoS Allocation Mask] (L3::L3CRB::L3QosAllocMask11)

Reset: 0000_0000_0000_FFFFh.

QOS L3 Allocation Mask for CLOS11

_ccd[11:0]_lthree0; MSR0000_0C9B

Bits	Description
63:16	Reserved.
15:0	**WayMask**. Read-write. Reset: FFFFh. L3 way mask used for allocation control.

MSR0000_0C9C [L3 QoS Allocation Mask] (L3::L3CRB::L3QosAllocMask12)

Reset: 0000_0000_0000_FFFFh.

QOS L3 Allocation Mask for CLOS12

_ccd[11:0]_lthree0; MSR0000_0C9C

Bits	Description
63:16	Reserved.
15:0	**WayMask**. Read-write. Reset: FFFFh. L3 way mask used for allocation control.

MSR0000_0C9D [L3 QoS Allocation Mask] (L3::L3CRB::L3QosAllocMask13)

Reset: 0000_0000_0000_FFFFh.

QOS L3 Allocation Mask for CLOS13

_ccd[11:0]_lthree0; MSR0000_0C9D

Bits	Description
63:16	Reserved.
15:0	**WayMask**. Read-write. Reset: FFFFh. L3 way mask used for allocation control.

MSR0000_0C9E [L3 QoS Allocation Mask] (L3::L3CRB::L3QosAllocMask14)

Reset: 0000_0000_0000_FFFFh.

QOS L3 Allocation Mask for CLOS14

_ccd[11:0]_lthree0; MSR0000_0C9E

Bits	Description
63:16	Reserved.
15:0	**WayMask**. Read-write. Reset: FFFFh. L3 way mask used for allocation control.

MSR0000_0C9F [L3 QoS Allocation Mask] (L3::L3CRB::L3QosAllocMask15)

Reset: 0000_0000_0000_FFFFh.

QOS L3 Allocation Mask for CLOS15

_ccd[11:0]_lthree0; MSR0000_0C9F

Bits	Description
63:16	Reserved.
15:0	**WayMask**. Read-write. Reset: FFFFh. L3 way mask used for allocation control.

MSRC000_0200 [L3 QoS Bandwidth Control] (L3::L3CRB::L3QosBwControl0)

Reset: 0000_0000_0000_0800h.

QOS BW Control0

_ccd[11:0]_lthree0; MSRC000_0200

Bits	Description
63:12	Reserved.
11:0	**Ceiling**. Read-write. Reset: 800h. QOS BW Control BW ceiling value

MSRC000_0201 [L3 QoS Bandwidth Control] (L3::L3CRB::L3QosBwControl1)

Reset: 0000_0000_0000_0800h.

QOS BW Control1

_ccd[11:0]_lthree0; MSRC000_0201

Bits	Description
63:12	Reserved.
11:0	**Ceiling**. Read-write. Reset: 800h. QOS BW Control BW ceiling value

MSRC000_0202 [L3 QoS Bandwidth Control] (L3::L3CRB::L3QosBwControl2)

Reset: 0000_0000_0000_0800h.

QOS BW Control2

_ccd[11:0]_lthree0; MSRC000_0202

Bits	Description
63:12	Reserved.
11:0	**Ceiling**. Read-write. Reset: 800h. QOS BW Control BW ceiling value

MSRC000_0203 [L3 QoS Bandwidth Control] (L3::L3CRB::L3QosBwControl3)

Reset: 0000_0000_0000_0800h.

QOS BW Control3

_ccd[11:0]_lthree0; MSRC000_0203

Bits	Description
63:12	Reserved.
11:0	**Ceiling**. Read-write. Reset: 800h. QOS BW Control BW ceiling value

MSRC000_0204 [L3 QoS Bandwidth Control] (L3::L3CRB::L3QosBwControl4)

Reset: 0000_0000_0000_0800h.

QOS BW Control4

_ccd[11:0]_lthree0; MSRC000_0204

Bits	Description
63:12	Reserved.
11:0	**Ceiling**. Read-write. Reset: 800h. QOS BW Control BW ceiling value

MSRC000_0205 [L3 QoS Bandwidth Control] (L3::L3CRB::L3QosBwControl5)

Reset: 0000_0000_0000_0800h.

QOS BW Control5

_ccd[11:0]_lthree0; MSRC000_0205

Bits	Description
63:12	Reserved.
11:0	**Ceiling**. Read-write. Reset: 800h. QOS BW Control BW ceiling value

MSRC000_0206 [L3 QoS Bandwidth Control] (L3::L3CRB::L3QosBwControl6)

Reset: 0000_0000_0000_0800h.

QOS BW Control6

_ccd[11:0]_lthree0; MSRC000_0206

Bits	Description
63:12	Reserved.
11:0	**Ceiling**. Read-write. Reset: 800h. QOS BW Control BW ceiling value

MSRC000_0207 [L3 QoS Bandwidth Control] (L3::L3CRB::L3QosBwControl7)

Reset: 0000_0000_0000_0800h.

QOS BW Control7

_ccd[11:0]_lthree0; MSRC000_0207

Bits	Description
63:12	Reserved.
11:0	**Ceiling**. Read-write. Reset: 800h. QOS BW Control BW ceiling value

MSRC000_0208 [L3 QoS Bandwidth Control] (L3::L3CRB::L3QosBwControl8)

Reset: 0000_0000_0000_0800h.

QOS BW Control8

_ccd[11:0]_lthree0; MSRC000_0208

Bits	Description
63:12	Reserved.
11:0	**Ceiling**. Read-write. Reset: 800h. QOS BW Control BW ceiling value

MSRC000_0209 [L3 QoS Bandwidth Control] (L3::L3CRB::L3QosBwControl9)

Reset: 0000_0000_0000_0800h.

QOS BW Control9

_ccd[11:0]_lthree0; MSRC000_0209

Bits	Description
63:12	Reserved.
11:0	**Ceiling**. Read-write. Reset: 800h. QOS BW Control BW ceiling value

MSRC000_020A [L3 QoS Bandwidth Control] (L3::L3CRB::L3QosBwControl10)

Reset: 0000_0000_0000_0800h.

QOS BW Control10

_ccd[11:0]_lthree0; MSRC000_020A

Bits	Description
63:12	Reserved.
11:0	**Ceiling**. Read-write. Reset: 800h. QOS BW Control BW ceiling value

MSRC000_020B [L3 QoS Bandwidth Control] (L3::L3CRB::L3QosBwControl11)

Reset: 0000_0000_0000_0800h.

QOS BW Control11

_ccd[11:0]_lthree0; MSRC000_020B

Bits	Description
63:12	Reserved.
11:0	**Ceiling**. Read-write. Reset: 800h. QOS BW Control BW ceiling value

MSRC000_020C [L3 QoS Bandwidth Control] (L3::L3CRB::L3QosBwControl12)

Reset: 0000_0000_0000_0800h.

QOS BW Control12

_ccd[11:0]_lthree0; MSRC000_020C

Bits	Description
63:12	Reserved.
11:0	**Ceiling**. Read-write. Reset: 800h. QOS BW Control BW ceiling value

MSRC000_020D [L3 QoS Bandwidth Control] (L3::L3CRB::L3QosBwControl13)

Reset: 0000_0000_0000_0800h.

QOS BW Control13

_ccd[11:0]_lthree0; MSRC000_020D

Bits	Description
63:12	Reserved.
11:0	**Ceiling**. Read-write. Reset: 800h. QOS BW Control BW ceiling value

MSRC000_020E [L3 QoS Bandwidth Control] (L3::L3CRB::L3QosBwControl14)

Reset: 0000_0000_0000_0800h.

QOS BW Control14

_ccd[11:0]_lthree0; MSRC000_020E

Bits	Description
63:12	Reserved.
11:0	**Ceiling**. Read-write. Reset: 800h. QOS BW Control BW ceiling value

MSRC000_020F [L3 QoS Bandwidth Control] (L3::L3CRB::L3QosBwControl15)

Reset: 0000_0000_0000_0800h.

QOS BW Control15

_ccd[11:0]_lthree0; MSRC000_020F

Bits	Description
63:12	Reserved.
11:0	**Ceiling**. Read-write. Reset: 800h. QOS BW Control BW ceiling value

MSRC000_0280 [L3 QoS Slow Memory Bandwidth Control] (L3::L3CRB::L3QOS_SLOWBW_CONTROL_0)

Reset: 0000_0000_0000_0800h.

QOS SM BW Control0

_ccd[11:0]_lthree0; MSRC000_0280

Bits	Description
63:12	Reserved.
11:0	**Ceiling**. Read-write. Reset: 800h. QOS BW Control BW ceiling value

MSRC000_0281 [L3 QoS Slow Memory Bandwidth Control] (L3::L3CRB::L3QOS_SLOWBW_CONTROL_1)

Reset: 0000_0000_0000_0800h.

QOS SM BW Control1

_ccd[11:0]_lthree0; MSRC000_0281

Bits	Description
63:12	Reserved.
11:0	**Ceiling**. Read-write. Reset: 800h. QOS BW Control BW ceiling value

MSRC000_0282 [L3 QoS Slow Memory Bandwidth Control] (L3::L3CRB::L3QOS_SLOWBW_CONTROL_2)

Reset: 0000_0000_0000_0800h.

QOS SM BW Control2

_ccd[11:0]_lthree0; MSRC000_0282

Bits	Description
63:12	Reserved.
11:0	**Ceiling**. Read-write. Reset: 800h. QOS BW Control BW ceiling value

MSRC000_0283 [L3 QoS Slow Memory Bandwidth Control] (L3::L3CRB::L3QOS_SLOWBW_CONTROL_3)

Reset: 0000_0000_0000_0800h.

QOS SM BW Control3

_ccd[11:0]_lthree0; MSRC000_0283

Bits	Description
63:12	Reserved.
11:0	**Ceiling**. Read-write. Reset: 800h. QOS BW Control BW ceiling value

MSRC000_0284 [L3 QoS Slow Memory Bandwidth Control] (L3::L3CRB::L3QOS_SLOWBW_CONTROL_4)

Reset: 0000_0000_0000_0800h.

QOS SM BW Control4

_ccd[11:0]_lthree0; MSRC000_0284

Bits	Description
63:12	Reserved.
11:0	**Ceiling**. Read-write. Reset: 800h. QOS BW Control BW ceiling value

MSRC000_0285 [L3 QoS Slow Memory Bandwidth Control] (L3::L3CRB::L3QOS_SLOWBW_CONTROL_5)

Reset: 0000_0000_0000_0800h.

QOS SM BW Control5

_ccd[11:0]_lthree0; MSRC000_0285

Bits	Description
63:12	Reserved.
11:0	**Ceiling**. Read-write. Reset: 800h. QOS BW Control BW ceiling value

MSRC000_0286 [L3 QoS Slow Memory Bandwidth Control] (L3::L3CRB::L3QOS_SLOWBW_CONTROL_6)

Reset: 0000_0000_0000_0800h.

QOS SM BW Control6

_ccd[11:0]_lthree0; MSRC000_0286

Bits	Description
63:12	Reserved.
11:0	**Ceiling**. Read-write. Reset: 800h. QOS BW Control BW ceiling value

MSRC000_0287 [L3 QoS Slow Memory Bandwidth Control] (L3::L3CRB::L3QOS_SLOWBW_CONTROL_7)

Reset: 0000_0000_0000_0800h.

QOS SM BW Control7

_ccd[11:0]_lthree0; MSRC000_0287

Bits	Description
63:12	Reserved.
11:0	**Ceiling**. Read-write. Reset: 800h. QOS BW Control BW ceiling value

MSRC000_0288 [L3 QoS Slow Memory Bandwidth Control] (L3::L3CRB::L3QOS_SLOWBW_CONTROL_8)

Reset: 0000_0000_0000_0800h.

QOS SM BW Control8

_ccd[11:0]_lthree0; MSRC000_0288

Bits	Description
63:12	Reserved.
11:0	**Ceiling**. Read-write. Reset: 800h. QOS BW Control BW ceiling value

MSRC000_0289 [L3 QoS Slow Memory Bandwidth Control] (L3::L3CRB::L3QOS_SLOWBW_CONTROL_9)

Reset: 0000_0000_0000_0800h.

QOS SM BW Control9

_ccd[11:0]_lthree0; MSRC000_0289

Bits	Description
63:12	Reserved.
11:0	**Ceiling**. Read-write. Reset: 800h. QOS BW Control BW ceiling value

MSRC000_028A [L3 QoS Slow Memory Bandwidth Control] (L3::L3CRB::L3QOS_SLOWBW_CONTROL_10)

Reset: 0000_0000_0000_0800h.

QOS SM BW Control10

_ccd[11:0]_lthree0; MSRC000_028A

Bits	Description
63:12	Reserved.
11:0	**Ceiling**. Read-write. Reset: 800h. QOS BW Control BW ceiling value

MSRC000_028B [L3 QoS Slow Memory Bandwidth Control] (L3::L3CRB::L3QOS_SLOWBW_CONTROL_11)

Reset: 0000_0000_0000_0800h.

QOS SM BW Control11

_ccd[11:0]_lthree0; MSRC000_028B

Bits	Description
63:12	Reserved.
11:0	**Ceiling**. Read-write. Reset: 800h. QOS BW Control BW ceiling value

MSRC000_028C [L3 QoS Slow Memory Bandwidth Control] (L3::L3CRB::L3QOS_SLOWBW_CONTROL_12)

Reset: 0000_0000_0000_0800h.

QOS SM BW Control12

_ccd[11:0]_lthree0; MSRC000_028C

Bits	Description
63:12	Reserved.
11:0	**Ceiling**. Read-write. Reset: 800h. QOS BW Control BW ceiling value

MSRC000_028D [L3 QoS Slow Memory Bandwidth Control] (L3::L3CRB::L3QOS_SLOWBW_CONTROL_13)

Reset: 0000_0000_0000_0800h.

QOS SM BW Control13

_ccd[11:0]_lthree0; MSRC000_028D

Bits	Description
63:12	Reserved.
11:0	**Ceiling**. Read-write. Reset: 800h. QOS BW Control BW ceiling value

MSRC000_028E [L3 QoS Slow Memory Bandwidth Control] (L3::L3CRB::L3QOS_SLOWBW_CONTROL_14)

Reset: 0000_0000_0000_0800h.

QOS SM BW Control14

_ccd[11:0]_lthree0; MSRC000_028E

Bits	Description
63:12	Reserved.
11:0	**Ceiling**. Read-write. Reset: 800h. QOS BW Control BW ceiling value

MSRC000_028F [L3 QoS Slow Memory Bandwidth Control] (L3::L3CRB::L3QOS_SLOWBW_CONTROL_15)

Reset: 0000_0000_0000_0800h.

QOS SM BW Control15

_ccd[11:0]_lthree0; MSRC000_028F

Bits	Description
63:12	Reserved.
11:0	**Ceiling**. Read-write. Reset: 800h. QOS BW Control BW ceiling value

MSRC000_0400 [L3 QoS Event Configuration] (L3::L3CRB::QOS_EVT_CFG_0)

Reset: 0000_0000_0000_007Fh.

Identifies the Bandwidth Sources to include in Bandwidth Monitoring Event 0

_ccd[11:0]_lthree0; MSRC000_0400

Bits	Description
63:7	Reserved.
6	**L3CacheVicBwMon**. Read-write. Reset: 1. Dirty Victims from the QOS domain to all types of memory
5	**L3CacheRmtSlowBwFillMon**. Read-write. Reset: 1. Reads to slow memory in the non-local NUMA domain
4	**L3CacheLclSlowBwFillMon**. Read-write. Reset: 1. Reads to slow memory in the local NUMA domain
3	**L3CacheRmtBwNtWrMon**. Read-write. Reset: 1. Non-temporal writes to non-local NUMA domain
2	**L3CacheLclBwNtWrMon**. Read-write. Reset: 1. Non-temporal writes to local NUMA domain
1	**L3CacheRmtBwFillMon**. Read-write. Reset: 1. Reads to memory in the non-local NUMA domain
0	**L3CacheLclBwFillMon**. Read-write. Reset: 1. Reads to memory in the local NUMA domain

MSRC000_0401 [L3 QoS Event Configuration] (L3::L3CRB::QOS_EVT_CFG_1)

Reset: 0000_0000_0000_0015h.

Identifies the Bandwidth Sources to include in Bandwidth Monitoring Event 1

_ccd[11:0]_lthree0; MSRC000_0401

Bits	Description
63:7	Reserved.
6	**L3CacheVicBwMon**. Read-write. Reset: 0. Dirty Victims from the QOS domain to all types of memory
5	**L3CacheRmtSlowBwFillMon**. Read-write. Reset: 0. Reads to slow memory in the non-local NUMA domain
4	**L3CacheLclSlowBwFillMon**. Read-write. Reset: 1. Reads to slow memory in the local NUMA domain
3	**L3CacheRmtBwNtWrMon**. Read-write. Reset: 0. Non-temporal writes to non-local NUMA domain
2	**L3CacheLclBwNtWrMon**. Read-write. Reset: 1. Non-temporal writes to local NUMA domain
1	**L3CacheRmtBwFillMon**. Read-write. Reset: 0. Reads to memory in the non-local NUMA domain
0	**L3CacheLclBwFillMon**. Read-write. Reset: 1. Reads to memory in the local NUMA domain

MSRC001_023[1...B] (L3::L3CRB::ChL3Pmc)

Read-write,Volatile. Reset: 0000_0000_0000_0000h.

_ccd[11:0]_lthree0_n0; MSRC001_0231
_ccd[11:0]_lthree0_n1; MSRC001_0233
_ccd[11:0]_lthree0_n2; MSRC001_0235
_ccd[11:0]_lthree0_n3; MSRC001_0237
_ccd[11:0]_lthree0_n4; MSRC001_0239
_ccd[11:0]_lthree0_n5; MSRC001_023B
_ccd[11:0]_lthree0_n0_aliasSMN; L3L3CRBx2[5B,53,4B,43,3B,33,2B,23,1B,13,0B,03]5_08A0; L3L3CRB=0000_0000h
_ccd[11:0]_lthree0_n1_aliasSMN; L3L3CRBx2[5B,53,4B,43,3B,33,2B,23,1B,13,0B,03]5_08A8; L3L3CRB=0000_0000h
_ccd[11:0]_lthree0_n2_aliasSMN; L3L3CRBx2[5B,53,4B,43,3B,33,2B,23,1B,13,0B,03]5_08B0; L3L3CRB=0000_0000h
_ccd[11:0]_lthree0_n3_aliasSMN; L3L3CRBx2[5B,53,4B,43,3B,33,2B,23,1B,13,0B,03]5_08B8; L3L3CRB=0000_0000h
_ccd[11:0]_lthree0_n4_aliasSMN; L3L3CRBx2[5B,53,4B,43,3B,33,2B,23,1B,13,0B,03]5_08C0; L3L3CRB=0000_0000h
_ccd[11:0]_lthree0_n5_aliasSMN; L3L3CRBx2[5B,53,4B,43,3B,33,2B,23,1B,13,0B,03]5_08C8; L3L3CRB=0000_0000h

Bits	Description
63:49	Reserved.
48	**Overflow**. Read-write,Volatile. Reset: 0. Counter 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_1095 [L3 Cache Range Reserve Base Address] (L3::L3CRB::L3RangeReserveBaseAddr)

Reset: 0000_0000_0000_0000h.

_ccd[11:0]_lthree0; MSRC001_1095

Bits	Description
63:52	Reserved.
51:12	**Addr**. Read-write. Reset: 00_0000_0000h. Base physical address bits 51:12 for the locked range.
11:0	Reserved.

MSRC001_1096 [L3 Cache Range Reserve Maximum Address] (L3::L3CRB::L3RangeReserveMaxAddr)

Reset: 0000_0000_0000_0000h.

_ccd[11:0]_lthree0; MSRC001_1096

Bits	Description
63:52	Reserved.
51:12	**Addr**. Read-write. Reset: 00_0000_0000h. Max physical address bits 51:12 for the locked range.
11:1	Reserved.
0	**En**. Read-write. Reset: 0. 0=Disable L3 Range Reservation. 1=Enable L3 Range Reservation. Enables the L3 range reservation when set.

MSRC001_109A [L3 Cache Range Reservation Way Mask] (L3::L3CRB::L3RangeReserveWayMask)

Reset: 0000_0000_0000_0000h.

Way mask used to specify which L3 cache ways are used for range reservation

_ccd[11:0]_lthree0; MSRC001_109A

Bits	Description
63:16	Reserved.
15:0	**Mask**. Read-write. Reset: 0000h. L3 ways used for range reservation.

2.2.2 L3 Clocks and Test (CT) MSR Registers.

MSRC001_0299 (L3::L3CT::L3RAPLPowerUnit0)

Read-only. Reset: 0000_0000_0000_0000h.

L3 RAPL Power Unit 0

_ccd[11:0]_lthree0; MSRC001_0299

Bits	Description
63:20	Reserved.
19:16	**TimeUnits**. Read-only. Reset: 0h. Time information (in Seconds) is based on the multiplier.1/2^TU where TU is unsigned integer Default value is 1010b, indicating time unit is in 976 microseconds increment
15:13	Reserved.
12:8	**EnergyStatusUnits**. Read-only. Reset: 00h. Energy information (in Joules) is based on the multiplier,1/2^ESU where ESU is unsigned integer.Default value is 10000b, indicating energy status unit is in 15.3 micro-Joules increment
7:4	Reserved.
3:0	**PowerUnits**. Read-only. Reset: 0h. 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

MSRC001_029B (L3::L3CT::L3PackageEnergyStatus)

Read-only. Reset: 0000_0000_0000_0000h.

L3 Package Energy Status 0

_ccd[11:0]_lthree0; MSRC001_029B

Bits	Description
63:0	**TotalEnergyConsumed**. Read-only. Reset: 0000_0000_0000_0000h.
	Description: Total Energy consumed since the last time the register is cleared. It reports the actual energy use for respective power domain.This MSR is updated every ~1ms. Energy status is free running.Users calculate power for a given domain by calculating dEnergy/dTime for that domain. Users must ensure successive reads contain atleast one,but preferably many energy status updates by hardware. Readable/writable field for use by SMU.