

# **Processor Programming Reference (PPR) for AMD Family 17h Model 18h, Revision B1 Processors**

# Legal Notices

© 2019-2021 Advanced Micro Devices, Inc. All rights reserved.

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 this document, 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 this document, 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 this document. 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.

DirectX is a registered trademark of Microsoft Corporation.

HDMI is a trademark of HDMI Licensing, LLC.

HyperTransport is a trademark of HyperTransport Consortium.

Linux is a registered trademark of Linus Torvalds.

Microsoft is a registered trademark of Microsoft Corporation.

OpenCL is a trademark of Apple, Inc. used by permission by Khronos Group, Inc.

OpenGL is a registered trademark of Hewlett Packard Enterprise.

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

- 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)**
- 7 Data Fabric**
- 8 Northbridge IO (NBIO)**
- 9 FCH**
- 10 MMIP – Audio**

**List of Namespaces**

**List of Definitions**

**Memory Map - MSR**

**Memory Map - Main Memory**

**Memory Map - PCICFG**

**Memory Map - SMN**

# Table of Contents

<b>1</b>	<b>Overview</b>
1.1	Intended Audience
1.2	Reference Documents
1.2.1	Documentation Conventions
1.3	Adobe® Reader
1.3.1	Adobe® Reader Configuration
1.3.1.1	Open Hyperlink Document in New Window
1.3.1.2	Show Toolbars
1.3.1.3	Show "Previous View" and "Next View" Buttons
1.3.2	Adobe® Reader Usage
1.3.2.1	Searching a Multiple Volume PPR
1.3.2.2	Cross-References and Hyperlinks
1.3.2.3	Expand Current Bookmark
1.4	Conventions
1.4.1	Numbering
1.4.2	Arithmetic And Logical Operators
1.4.2.1	Operator Precedence and Associativity
1.4.3	Register Mnemonics
1.4.3.1	Logical Mnemonic
1.4.3.2	Physical Mnemonic
1.4.4	Register Format
1.4.4.1	A Register is a group of Register Instances
1.4.4.2	Register Physical Mnemonic, Title, and Name
1.4.4.3	Full Width Register Attributes
1.4.4.4	Register Description
1.4.4.5	Register Instance Table
1.4.4.5.1	Content Ordering in a Row
1.4.4.5.2	Multiple Instances Per Row
1.4.4.5.3	MSR Access Method
1.4.4.5.3.1	MSR Per-Thread Example
1.4.4.5.3.2	MSR Range Example
1.4.4.5.4	BAR Access Method
1.4.4.5.4.1	BAR as a Register Reference
1.4.4.5.5	PCICFG Access Method
1.4.4.5.5.1	PCICFG Bus Implied to be 00h
1.4.4.5.6	Data Port Access Method
1.4.4.6	Register Field Format
1.4.4.7	Simple Register Field Format
1.4.4.8	Complex Register Field Format
1.4.4.9	Field Name is Reserved
1.4.4.10	Field Access Type
1.4.4.10.1	Conditional Access Type Expression
1.4.4.11	Field Reset
1.4.4.12	Field Initialization
1.4.4.13	Field Check
1.4.4.14	Field Valid Values
1.4.5	Revision History and Change Bar Notation
1.5	Definitions
1.6	Changes Between Revisions and Product Variations
1.6.1	Revision Conventions

- 1.7 Package
  - 1.7.1 Package type
- 1.8 Processor Overview
  - 1.8.1 Features
- 2 **Core Complex (CCX)**
  - 2.1 Processor x86 Core
    - 2.1.1 Core Definitions
    - 2.1.2 Secure Virtual Machine Mode (SVM)
      - 2.1.2.1 BIOS support for SVM Disable
        - 2.1.2.1.1 Enable AMD Virtualization™
        - 2.1.2.1.2 Disable AMD Virtualization™
        - 2.1.2.1.3 Disable AMD Virtualization™, with a user supplied key
    - 2.1.3 Memory Encryption
    - 2.1.4 Effective Frequency
    - 2.1.5 Address Space
      - 2.1.5.1 Virtual Address Space
      - 2.1.5.2 Physical Address Space
      - 2.1.5.3 System Address Map
        - 2.1.5.3.1 Memory Access to the Physical Address Space
          - 2.1.5.3.1.1 Determining Memory Type
    - 2.1.6 Configuration Space
      - 2.1.6.1 MMIO Configuration Coding Requirements
      - 2.1.6.2 MMIO Configuration Ordering
      - 2.1.6.3 Processor Configuration Space
    - 2.1.7 PCI Configuration Legacy Access
    - 2.1.8 System Software Interaction With SMT Enabled
    - 2.1.9 Register Sharing
    - 2.1.10 Timers
    - 2.1.11 Interrupts
      - 2.1.11.1 System Management Mode (SMM)
        - 2.1.11.1.1 SMM Overview
        - 2.1.11.1.2 Mode and Default Register Values
        - 2.1.11.1.3 SMI Sources And Delivery
        - 2.1.11.1.4 SMM Initial State
        - 2.1.11.1.5 SMM Save State
        - 2.1.11.1.6 System Management State
        - 2.1.11.1.7 Exceptions and Interrupts in SMM
        - 2.1.11.1.8 The Protected ASeg and TSeg Areas
        - 2.1.11.1.9 SMM Special Cycles
        - 2.1.11.1.10 Locking SMM
      - 2.1.11.2 Local APIC
        - 2.1.11.2.1 Local APIC Functional Description
          - 2.1.11.2.1.1 Detecting and Enabling
          - 2.1.11.2.1.2 APIC Register Space
          - 2.1.11.2.1.3 ApicId Enumeration Requirements
          - 2.1.11.2.1.4 Physical Destination Mode
          - 2.1.11.2.1.5 Logical Destination Mode
          - 2.1.11.2.1.6 Interrupt Delivery
          - 2.1.11.2.1.7 Vectored Interrupt Handling
          - 2.1.11.2.1.8 Interrupt Masking
          - 2.1.11.2.1.9 Spurious Interrupts
          - 2.1.11.2.1.10 Spurious Interrupts Caused by Timer Tick Interrupt
          - 2.1.11.2.1.11 Lowest-Priority Interrupt Arbitration

- 2.1.11.2.1.12 Inter-Processor Interrupts
- 2.1.11.2.1.13 APIC Timer Operation
- 2.1.11.2.1.14 Generalized Local Vector Table
- 2.1.11.2.1.15 State at Reset
- 2.1.11.2.2 Local APIC Registers
- 2.1.12 CPUID Instruction
- 2.1.12.1 CPUID Instruction Functions
- 2.1.13 MSR Registers
- 2.1.13.1 MSRs - MSR0000\_0xxx
- 2.1.13.2 MSRs - MSRC000\_0xxx
- 2.1.13.2.1 MSRs - MSRC000\_2xxx
- 2.1.13.3 MSRs - MSRC001\_0xxx
- 2.1.13.4 MSRs - MSRC001\_1xxx
- 2.1.14 Performance Monitor Counters
- 2.1.14.1 RDPMC Assignments
- 2.1.14.2 Large Increment per Cycle Events
- 2.1.14.3 Core Performance Monitor Counters
- 2.1.14.3.1 Floating Point (FP) Events
- 2.1.14.3.2 LS Events
- 2.1.14.3.3 IC and BP Events
- 2.1.14.3.4 DE Events
- 2.1.14.3.5 EX (SC) Events
- 2.1.14.3.6 L2 Cache Events
- 2.1.14.4 L3 Cache Performance Monitor Counters
- 2.1.14.4.1 L3 Cache PMC Events
- 2.1.15 Instruction Based Sampling (IBS)

### 3 Reliability, Availability, and Serviceability (RAS) Features

- 3.1 Machine Check Architecture
  - 3.1.1 Overview
    - 3.1.1.1 Legacy Machine Check Architecture
    - 3.1.1.2 Machine Check Architecture Extensions
    - 3.1.1.3 Use of MCA Information
      - 3.1.1.3.1 Error Management
      - 3.1.1.3.2 Fault Management
  - 3.1.2 Machine Check Registers
    - 3.1.2.1 Global Registers
    - 3.1.2.2 Machine Check Banks
      - 3.1.2.2.1 Legacy MCA Registers
      - 3.1.2.2.2 Legacy MCA MSRs
      - 3.1.2.2.3 MCAX Registers
      - 3.1.2.2.4 MCAX MSRs
    - 3.1.2.3 Access Permissions
  - 3.1.3 Machine Check Errors
    - 3.1.3.1 Error Severities
    - 3.1.3.2 Exceptions and Interrupts
    - 3.1.3.3 Error Codes
    - 3.1.3.4 Extended Error Codes
    - 3.1.3.5 DOER and SEER State
    - 3.1.3.6 MCA Overflow Recovery
    - 3.1.3.7 MCA Recovery
  - 3.1.4 Machine Check Features
    - 3.1.4.1 Error Thresholding
    - 3.1.4.2 Error Simulation

- 3.1.5 Software Guidelines
  - 3.1.5.1 Recognizing MCAX Support
  - 3.1.5.2 Communicating MCAX Support
  - 3.1.5.3 Machine Check Initialization
  - 3.1.5.4 Determining Bank Count
  - 3.1.5.5 Determining Bank Type
  - 3.1.5.6 Recognizing Error Type
  - 3.1.5.7 Machine Check Error Handling
- 3.2 Machine Check Architecture Implementation
  - 3.2.1 Implemented Machine Check Banks
  - 3.2.2 Implemented Machine Check Bank Registers
  - 3.2.3 Mapping of Banks to Blocks
  - 3.2.4 Decoding Error Type
  - 3.2.5 MCA Banks
    - 3.2.5.1 LS
    - 3.2.5.2 IF
    - 3.2.5.3 L2
    - 3.2.5.4 DE
    - 3.2.5.5 EX
    - 3.2.5.6 FP
    - 3.2.5.7 L3
    - 3.2.5.8 CS
    - 3.2.5.9 PIE
    - 3.2.5.10 UMC
- 4 System Management Unit (SMU)**
  - 4.1 SMU Registers
  - 4.2 Thermal (THM)
    - 4.2.1 Registers
- 5 Advanced Platform Management Link (APML)**
  - 5.1 Overview
    - 5.1.1 Definitions
  - 5.2 SBI Bus Characteristics
    - 5.2.1 SMBus Protocol Support
    - 5.2.2 I2C Support
  - 5.3 SBI Processor Information
    - 5.3.1 SBI Processor Pins
      - 5.3.1.1 Physical Layer Characteristics
    - 5.3.2 Processor States
  - 5.4 SBI Protocols
    - 5.4.1 SBI Modified Block Write-Block Read Process Call
    - 5.4.2 SBI Error Detection and Recovery
      - 5.4.2.1 Error Detection
        - 5.4.2.1.1 ACK/NAK Mechanism
        - 5.4.2.1.2 Packet Error Correction (PEC)
        - 5.4.2.1.3 Bus Timeouts
      - 5.4.2.2 Error Recovery
        - 5.4.2.2.1 SBI Bus Reset
  - 5.5 SBI Physical Interface
    - 5.5.1 SBI SMBus Address
    - 5.5.2 SBI Bus Timing
    - 5.5.3 Pass-FET Option
- 6 SB Temperature Sensor Interface (SB-TSI)**
  - 6.1 Overview

- 6.1.1 Definitions
- 6.2 SB-TSI Protocol
  - 6.2.1 SB-TSI Send/Receive Byte Protocol
    - 6.2.1.1 SB-TSI Address Pointer
  - 6.2.2 SB-TSI Read/Write Byte Protocol
  - 6.2.3 Alert Behavior
  - 6.2.4 Atomic Read Mechanism
  - 6.2.5 SB-TSI Temperature and Threshold Encodings
  - 6.2.6 SB-TSI Temperature Offset Encoding
- 6.3 SB-TSI Physical Interface
  - 6.3.1 SB-TSI SMBus Address
  - 6.3.2 SB-TSI Bus Timing
  - 6.3.3 SB-TSI Bus Electrical Parameters
  - 6.3.4 Pass-FET Option
- 6.4 SB-TSI Registers
- 7 Data Fabric**
  - 7.1 Data Fabric Instance and Fabric IDs
    - 7.1.1 System InstanceID and FabricID Assignment
- 8 Northbridge IO (NBIO)**
  - 8.1 IOMMU
    - 8.1.1 Functional Description
      - 8.1.1.1 Definitions
    - 8.1.2 Registers
      - 8.1.2.1 IOMMUL1 Registers
      - 8.1.2.2 IOMMUL2 Registers
      - 8.1.2.3 IOMMUMMIO Registers
- 9 FCH**
  - 9.1 FCH Overview
    - 9.1.1 Acronyms
    - 9.1.2 Functional
    - 9.1.3 MMIO Programming for Legacy Devices
      - 9.1.3.1 Description for FCH::IO::PCIInterrupt Map
    - 9.1.4 On-Chip Clock Generator
    - 9.1.5 EMMC
    - 9.1.6 eSPI
    - 9.1.7 LPC Bus Interface
      - 9.1.7.1 Read/Write SPI Flash through LPC
      - 9.1.7.2 Enabling SPI 100
      - 9.1.7.3 Serial Peripheral Interface (SPI) ROM
        - 9.1.7.3.1 Enable SPI ROM Protection
    - 9.1.8 GPIO
      - 9.1.8.1 Interrupt GPIO
      - 9.1.8.2 Wakeup GPIO
      - 9.1.8.3 Pure GPIO
      - 9.1.8.4 GPIO Driving Strength
      - 9.1.8.5 PCIE\_RST1 Enhancement
    - 9.1.9 I2C Slave
    - 9.1.10 FCH Register Access Information Guide
    - 9.1.11 Reset Overview
    - 9.1.12 RTC
      - 9.1.12.1 RTC Register
    - 9.1.13 Address Mapping Table
  - 9.2 Registers



- 9.2.1 Legacy Block Configuration Registers (IO)
  - 9.2.1.1 Registers
- 9.2.2 eMMC
  - 9.2.2.1 eMMC Configuration Registers
  - 9.2.2.2 eMMC Host Controller Registers
- 9.2.3 I/O Advanced Programmable Interrupt Control
  - 9.2.3.1 IOAPIC Registers
- 9.2.4 SMI Registers
- 9.2.5 High Precision Event Timer (HPET) Registers
- 9.2.6 Watchdog Timer (WDT) Registers
- 9.2.7 Wake Alarm Device (AcDcTimer) Registers
- 9.2.8 Always On Always Connected (AOAC) Registers
- 9.2.9 ISA Bridge
  - 9.2.9.1 Device 14h Function 3 (LPC Bridge) Configuration Registers
  - 9.2.9.2 SPI Registers
  - 9.2.9.3 Programming for ROM Protection register
    - 9.2.9.3.1 Index Mode (Indirect) Access
      - 9.2.9.3.1.1 24-bit Address Index Mode
    - 9.2.9.3.2 FCH::ITF::SPI Registers
  - 9.2.9.4 eSPI Registers
- 9.2.10 MISC Registers
  - 9.2.10.1 Miscellaneous (MISC) Registers
  - 9.2.10.2 Power Management (PM) Registers
  - 9.2.10.3 Power Management (PM2) Registers
  - 9.2.10.4 Standard ACPI Registers
    - 9.2.10.4.1 AcpiPmEvtBlk
    - 9.2.10.4.2 AcpiPm1CntBlk
    - 9.2.10.4.3 AcpiPmTmrBlk
    - 9.2.10.4.4 AcpiGpe0Blk
    - 9.2.10.4.5 SmiCmdBlk
- 9.2.11 GPIO Pin control registers
  - 9.2.11.1 IOMUX Registers
  - 9.2.11.2 GPIO Registers
- 9.2.12 Secure Digital (SD) Controller
  - 9.2.12.1 SD Configuration Registers (SD)
  - 9.2.12.2 SD Host Controller Memory Mapped Registers (SDHC)
- 9.2.13 USB Legacy Registers

## 10 MMIP – Audio

- 10.1 Audio Coprocessor (ACP)
  - 10.1.1 Function and Registers

# List of Figures

Figure 1:	Adobe® Reader Hyperlink Opens New Window Configuration
Figure 2:	Adobe® Reader Select Between Opened Files
Figure 3:	Adobe® Reader Show Toolbars Configuration
Figure 4:	Adobe® Reader Prev/Next Buttons
Figure 5:	Adobe® Reader Searching a Multiple Volume PPR
Figure 6:	Adobe® Reader Expand Current Bookmark Button
Figure 7:	Register Physical Mnemonic, Title, and Name
Figure 8:	Full Width Register Attributes
Figure 9:	Register Description
Figure 10:	Register Instance Table: Content Ordering in a Row
Figure 11:	Register Instance Table: MSR Example
Figure 12:	Register Instance Table: MSR Range Example
Figure 13:	Register Instance Table: BAR as Register Reference
Figure 14:	Register Instance Table: Bus Implied to be 00h
Figure 15:	Register Instance Table: Data Port Select
Figure 16:	Simple Register Field Example
Figure 17:	Register Field Sub-Row for {Reset,AccessType,Init,Check}
Figure 18:	Register Field Sub-Row for Description
Figure 19:	Register Field Sub-Row for Valid Value Table
Figure 20:	Register Field Sub-Row for Valid Bit Table
Figure 21:	Revision History Format Example
Figure 22:	Change Notation Example
Figure 23:	Register Sharing Domains
Figure 24:	Instance Parameters
Figure 25:	SBI Transmission Protocol
Figure 26:	Pass FET Implementation
Figure 27:	RTS Thermal Management Example
Figure 28:	SB-TSI Thermal Management Example
Figure 29:	Alert Assertion Diagram
Figure 30:	Pass FET Implementation

# List of Tables

Table 1:	Reference Documents Listing
Table 2:	Arithmetic and Logical Operator Definitions
Table 3:	Function Definitions
Table 4:	Operator Precedence and Associativity
Table 5:	Register Mnemonic Definitions
Table 6:	Logical Mnemonic Definitions
Table 7:	Physical Mnemonic Definitions
Table 8:	AccessType Definitions
Table 9:	Reset Type Definitions
Table 10:	Init Type Definitions
Table 11:	Definitions
Table 12:	Package Definitions
Table 13:	PCI Device ID Assignments.
Table 14:	Definitions
Table 15:	SMM Initial State
Table 16:	SMM Save State
Table 17:	ICR Valid Combinations
Table 18:	PMC_Definitions
Table 19:	Machine Check Terms and Acronyms
Table 20:	Legacy MCA MSR Layout
Table 21:	MCAX MSR Layout
Table 22:	MCAX Implementation-Specific Register Layout
Table 23:	Error Overwrite Priorities
Table 24:	Error Scope Hierarchy
Table 25:	Error Code Types
Table 26:	Error code: transaction type (TT)
Table 27:	Error codes: cache level (LL)
Table 28:	Error codes: memory transaction type (RRRR)
Table 29:	Blocks Capable of Supporting MCA Banks
Table 30:	Legacy MCA Registers
Table 31:	MCAX Registers
Table 32:	MCA Bank to Block Mapping
Table 33:	MCA_STATUS_LS[ErrorCodeExt] Decode
Table 34:	MCA_ADDR_LS Register
Table 35:	MCA_SYND_LS Register
Table 36:	MCA_STATUS_IF[ErrorCodeExt] Decode
Table 37:	MCA_ADDR_IF Register
Table 38:	MCA_SYND_IF Register
Table 39:	MCA_STATUS_L2[ErrorCodeExt] Decode
Table 40:	MCA_ADDR_L2 Register
Table 41:	MCA_SYND_L2 Register
Table 42:	MCA_STATUS_DE[ErrorCodeExt] Decode
Table 43:	MCA_ADDR_DE Register
Table 44:	MCA_SYND_DE Register
Table 45:	MCA_STATUS_EX[ErrorCodeExt] Decode
Table 46:	MCA_ADDR_EX Register
Table 47:	MCA_SYND_EX Register
Table 48:	MCA_STATUS_FP[ErrorCodeExt] Decode
Table 49:	MCA_ADDR_FP Register
Table 50:	MCA_SYND_FP Register

Table 51:	MCA_STATUS_L3[ErrorCodeExt] Decode
Table 52:	MCA_ADDR_L3 Register
Table 53:	MCA_SYND_L3 Register
Table 54:	MCA_STATUS_CS[ErrorCodeExt] Decode
Table 55:	MCA_ADDR_CS Register
Table 56:	MCA_SYND_CS Register
Table 57:	MCA_STATUS_PIE[ErrorCodeExt] Decode
Table 58:	MCA_ADDR_PIE Register
Table 59:	MCA_SYND_PIE Register
Table 60:	MCA_STATUS_UMC[ErrorCodeExt] Decode
Table 61:	MCA_ADDR_UMC Register
Table 62:	MCA_SYND_UMC Register
Table 63:	APML Definitions
Table 64:	SB-TSI Definitions
Table 65:	SB-TSI CPU Temperature and Threshold Encoding Examples
Table 66:	SB-TSI Temperature Offset Encoding Examples
Table 67:	SB-TSI Address Encodings
Table 68:	Instance and Fabric IDs
Table 69:	Link Definitions
Table 70:	List of Acronyms used in FCH
Table 71:	ValidValuesTable: PCI interrupt index list of PCI_INTR_INDEX bit[6:0]
Table 72:	Driving Strength Configuration
Table 73:	Register Access Information
Table 74:	Reset Type
Table 75:	Reset Overview
Table 76:	Address Space Mapping
Table 77:	Device D3 Control
Table 78:	SpiReadMode[2:0]
Table 79:	ACPI MMIO Space Allocation
Table 80:	IOMUX Function Table
Table 81:	I2C Pad Configuration Method
Table 82:	Reset Value for GPIO BANK0
Table 83:	Reset Value for GPIO BANK1
Table 84:	Reset Value for GPIO BANK2
Table 85:	Reset Value for GPIO BANK3
Table 86:	Debounce Timer Definition

## 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
<b>docAPM1</b>	AMD64 Architecture Programmer's Manual Volume 1: Application Programming, order# 24592.
<b>docAPM2</b>	AMD64 Architecture Programmer's Manual Volume 2: System Programming, order# 24593.
<b>docAPM3</b>	AMD64 Architecture Programmer's Manual Volume 3: Instruction-Set Reference, order# 24594.
<b>docAPM4</b>	AMD64 Architecture Programmer's Manual Volume 4: 128-Bit and 256-Bit Media Instructions, order# 26568.
<b>docAPM5</b>	AMD64 Architecture Programmer's Manual Volume 5: 64-Bit Media and x87 Floating-Point Instructions, order# 26569.
<b>docACPI</b>	Advanced Configuration and Power Interface (ACPI) Specification. <a href="http://www.acpi.info">http://www.acpi.info</a> .
<b>docASF</b>	Alert Standard Format Specification. <a href="http://dmtf.org/standards/asf">http://dmtf.org/standards/asf</a> .
<b>docATA</b>	AT Attachment with Packet Interface. <a href="http://www.t13.org">http://www.t13.org</a> .
<b>docDP</b>	VESA DisplayPort Standard. <a href="http://www.vesa.org/vesa-standards">http://www.vesa.org/vesa-standards</a> .
<b>docIOMMU</b>	AMD I/O Virtualization Technology Specification, order# 48882.
<b>docI2C</b>	I2C Bus Specification. <a href="http://www.nxp.com/documents/user_manual/UM10204.pdf">http://www.nxp.com/documents/user_manual/UM10204.pdf</a>
<b>docJEDEC</b>	JEDEC Standards. <a href="http://www.jedec.org">http://www.jedec.org</a> .
<b>docPCIe</b>	PCI Express® Specification. <a href="http://www.pcisig.org">http://www.pcisig.org</a> .
<b>docPCIb</b>	PCI Local Bus Specification. <a href="http://www.pcisig.org">http://www.pcisig.org</a> .
<b>docRevG</b>	Revision Guide for AMD Family 17h Models 10h-1Fh Processors, order# 55606.
<b>docSATA</b>	Serial ATA Specification. <a href="http://www.sata-io.org">http://www.sata-io.org</a> .
<b>docSDHC</b>	Secure Digital Host Controller Standard Specification. <a href="https://www.sdcard.org">https://www.sdcard.org</a> .
<b>docAM4</b>	Socket AM4 Processor Functional Data Sheet, order# 55509.
<b>docSFP5</b>	AMD Socket FP5 Processor Functional Data Sheet, order# 55594.
<b>docSMB</b>	System Management Bus (SMBus) Specification. <a href="http://www.smbus.org">http://www.smbus.org</a> .
<b>docUSB</b>	Universal Serial Bus Specification. <a href="http://www.usb.org">http://www.usb.org</a> .

#### 1.2.1 Documentation Conventions

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

docExDoc => Header1 => Header2

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

### 1.3 Adobe® Reader

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

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

#### 1.3.1 Adobe® Reader Configuration

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

##### 1.3.1.1 Open Hyperlink Document in New Window

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

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

Edit->Preferences:

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

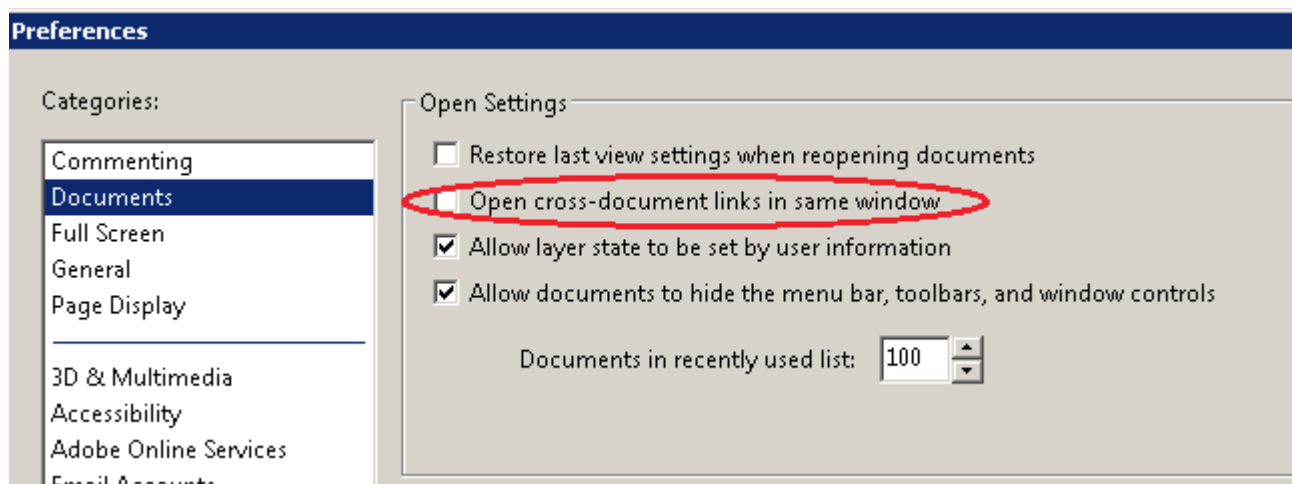


Figure 1: Adobe® Reader Hyperlink Opens New Window Configuration

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

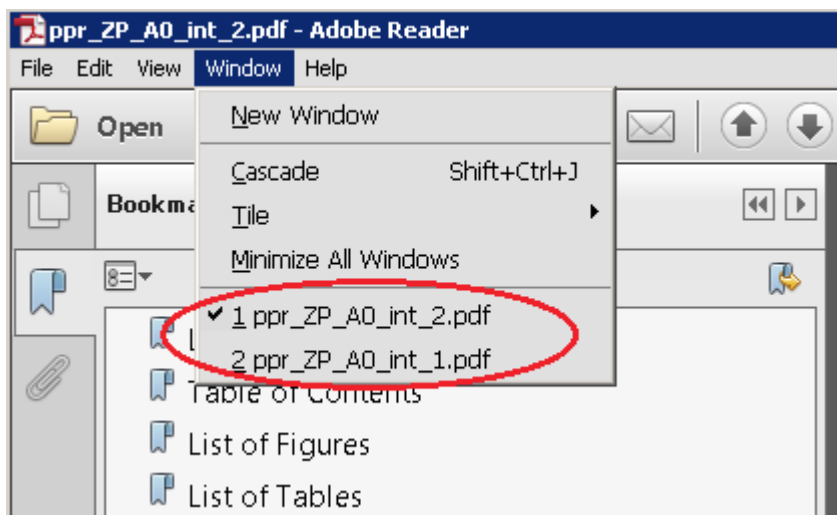


Figure 2: Adobe® Reader Select Between Opened Files

### 1.3.1.2 Show Toolbars

If Toolbars is not shown:

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



Figure 3: Adobe® Reader Show Toolbars Configuration

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

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

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



Figure 4: Adobe® Reader Prev/Next Buttons

## 1.3.2 Adobe® Reader Usage

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

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

### 1.3.2.1 Searching a Multiple Volume PPR

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

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

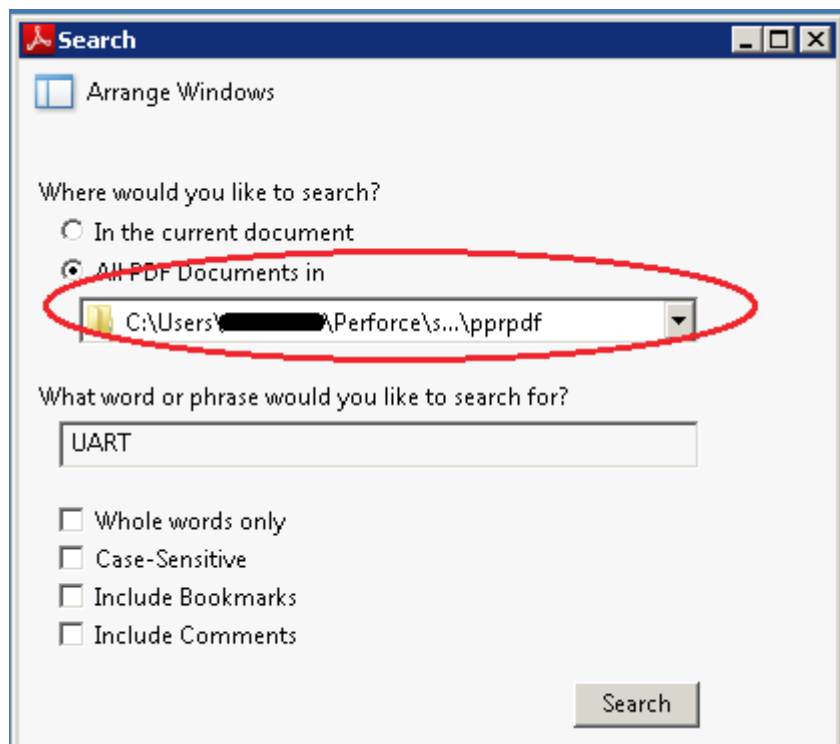


Figure 5: Adobe® Reader Searching a Multiple Volume PPR

### 1.3.2.2 Cross-References and Hyperlinks

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

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

### 1.3.2.3 Expand Current Bookmark

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



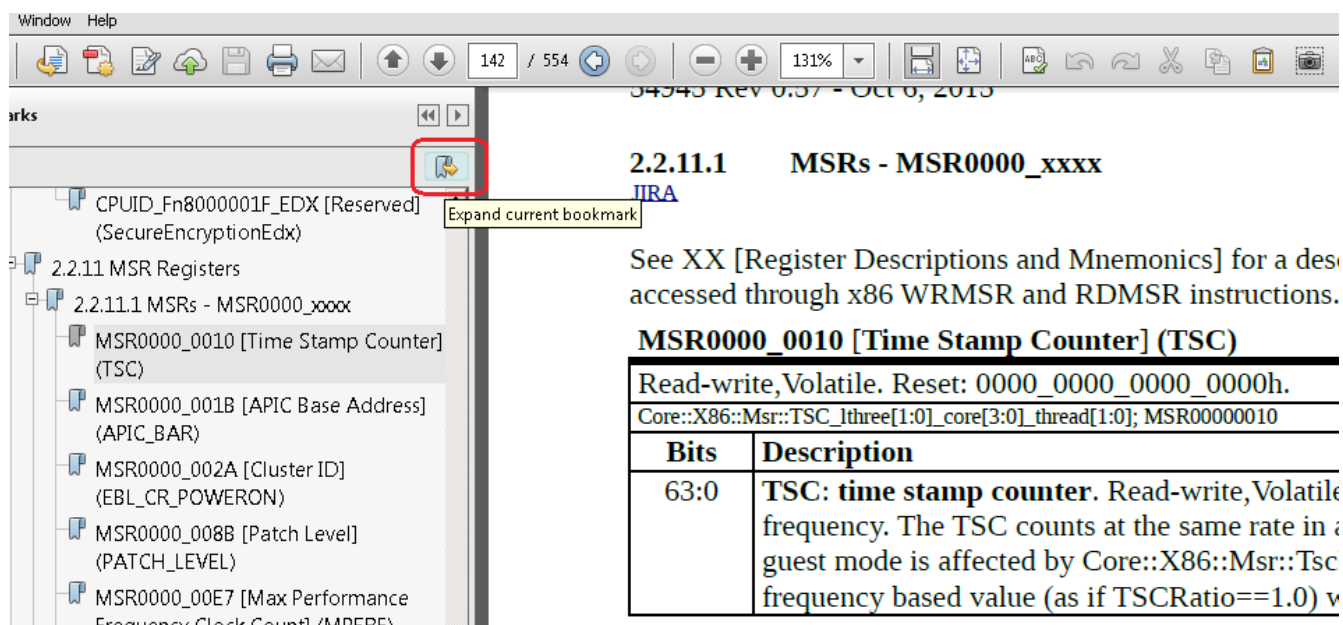


Figure 6: Adobe® Reader Expand Current Bookmark Button

## 1.4 Conventions

### 1.4.1 Numbering

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

### 1.4.2 Arithmetic And Logical Operators

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

Table 2: Arithmetic and Logical Operator Definitions

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

$\wedge$	Bitwise exclusive-OR (e.g., $01b \wedge 10b == 11b$ ). Sometimes used as "raised to the power of" as well, as indicated by the context in which it is used (e.g., $2^2 == 4$ ).
$\sim$	Bitwise NOT (also known as one's complement). (e.g., $\sim 10b == 01b$ ).
!	Logical NOT (e.g., $!10b == 0b$ ). It treats a multi-bit operand as 1 if $\geq 1$ 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., $01b \ll 01b == 10b$ ).
>>	Bitwise right shift. Shift right first operand by the number of bits specified by the 2nd operand (e.g., $10b \gg 01b == 01b$ ).
?:	Ternary conditional (e.g., condition ? value if true : value if false).

Table 3: Function Definitions

Term	Description
<b>ABS</b>	ABS(integer expression): Remove sign from signed value.
<b>FLOOR</b>	FLOOR(integer expression): Rounds real number down to nearest integer.
<b>CEIL</b>	CEIL(real expression): Rounds real number up to nearest integer.
<b>MIN</b>	MIN(integer expression list): Picks minimum integer or real value of comma separated list.
<b>MAX</b>	MAX(integer expression list): Picks maximum integer or real value of comma separated list.
<b>COUNT</b>	COUNT(integer expression): Returns the number of binary 1's in the integer.
<b>ROUND</b>	ROUND(real expression): Rounds to the nearest integer; halfway rounds away from zero.
<b>UNIT</b>	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.
<b>POW</b>	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
$\wedge$	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
<b>logical mnemonic</b>	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].
<b>physical mnemonic</b>	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
<b>register namespace</b>	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).
<b>register name</b>	A name that connotes the function of the register.
<b>register instance specifier</b>	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]).

<b>register instance parameter specifier</b>	A register instance parameter specifier is of the form <code>_register parameter name[register parameter value list]</code> (e.g., The register instance parameter specifier <code>_dct[1:0]</code> has a register parameter name of <code>dct</code> (The DCT PHY instance name) and a register parameter value list of "1:0" or 2 instances of DCT PHY).
<b>register parameter name</b>	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 <code>dct</code> specifies how many instances of the DCT PHY exist).
<b>register parameter value list</b>	The register parameter value list is the logical name for each instance of the register parameter name (e.g., For <code>_dct[1:0]</code> , 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 <code>AddressMappingTable</code> 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
<b>PCICFG</b>	The PCICFG, or PCI defined configuration space, physical register mnemonic format is of the form <code>DXFYxZZZ</code> .
<b>BAR</b>	The BAR, or base address register, physical register mnemonic format is of the form <code>PREFIXxZZZ</code> .
<b>MSR</b>	The MSR, or x86 model specific register, physical register mnemonic format is of the form <code>MSRXXXX_XXXX</code> , where <code>XXXX_XXXX</code> is the hexadecimal MSR number. This space is accessed through x86 defined <code>RDMSR</code> and <code>WRMSR</code> instructions.
<b>PMC</b>	The PMC, or x86 performance monitor counter, physical register mnemonic format is any of the forms <code>{PMCxXXX, L2IPMCxXXX, NBPMCxXXX}</code> , where <code>XXX</code> is the performance monitor select.
<b>CPUID</b>	The CPUID, or x86 processor identification state, physical register mnemonic format is of the form <code>CPUID FnXXXX_XXXX_EiX[_xYYY]</code> , where <code>XXXX_XXXX</code> is the hex value in the EAX and <code>YYY</code> is the hex value in ECX.

## 1.4.4 Register Format

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

### 1.4.4.1 A Register is a group of Register Instances

All instances of a register:

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

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

#### 1.4.4.2 Register Physical Mnemonic, Title, and Name

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

1. The physical mnemonic of the register.
  - A register that has multiple instances, may have instances that have different access methods, each with its 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
<b>MSR0000_0010</b>	<b>[Time Stamp Counter]</b>	<b>(TSC)</b>
Read-write, Volatile. Reset: 0000_0000_0000_0000h.		
Core::X86::Msr::TSC_three[1:0]_core[3:0]_thread[1:0]; MSR00000010		
Bits	Description	
63:0	TSC: <b>time stamp counter</b> . Read-write, Volatile. Reset: 0. The TSC increments at the P0 frequency. The TSC counts at the same rate in all P-states, all C states, S0, or S1. A read of this MSR in guest mode is affected by Core::X86::Msr::TscRateMsr. The value (TSC/TSCRatio) is the TSC P0 frequency based value (as if TSCRatio == 1.0) when (TSCRatio != 1.0).	

Figure 7: Register Physical Mnemonic, Title, and Name

#### 1.4.4.3 Full Width Register Attributes

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

- AccessType: If all non-reserved fields of a register have the same access type, then the access type is rendered in this row.
  - The supported access types are specified by 1.4.4.10 [Field Access Type].
  - The example figure shows that the access type "Read-write, Volatile" applies to all non-reserved fields of the register.
- Reset: If all non-reserved fields of a register have a constant reset and are all the same type (Warm, Cold, Fixed), then the full width register reset is rendered in this row. The example figure shows the reset "0000\_0000\_0000\_0000h". See 1.4.4.11 [Field Reset].
  - The value zero (0) is assumed for display purposes for all reserved fields.
- If none of the above content is rendered, then this row of the register is not rendered.

**MSR0000\_0010 [Time Stamp Counter] (TSC)**

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

Figure 8: Full Width Register Attributes

**1.4.4.4 Register Description**

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

**PciDevVendIDF3 [Device/Vendor ID]**

Read-only. Reset: 0000_1022h.	
A register description. That can be multiple paragraphs.	
Link::Phy::Tx::PciDevVendIDF3; D18F3x00	
Bits	Description
31:16	<b>DeviceID: device ID.</b> Read-only. Reset: Fixed, 0000h.
15:0	<b>VendorID: vendor ID.</b> Read-only. Reset: Fixed, 1022h. Init: 1234h.

Figure 9: Register Description

**1.4.4.5 Register Instance Table**

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

The register instance table can generally be described as follows:

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

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

**1.4.4.5.1 Content Ordering in a Row**

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

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

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

**BXXD00F0x000 (NB\_VENDOR\_ID)**

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

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

#### 1.4.4.5.2 Multiple Instances Per Row

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

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

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

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

Example:

NAMESP::REGNAME\_inst[BLOCK[5:0],BCST]\_aliasHOST; FFF1x00000088\_x[000[B:6]\_0001,00000000]

- There are 7 instances.
- NAMESP is the namespace.
- 6 instances are represented by the sub-range 000[B:6]\_0001.
- \_instBCST corresponds to FFF1x00000088\_x00000000.
- \_inst BLOCK 0 corresponds to FFF1x00000088\_x00060001.
- ...
- \_inst BLOCK 5 corresponds to FFF1x00000088\_x000B0001.

#### 1.4.4.5.3 MSR Access Method

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

MSRs that are:

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

##### 1.4.4.5.3.1 MSR Per-Thread Example

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



**MSR0000\_0010 [Time Stamp Counter] (TSC)**

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

Figure 11: Register Instance Table: MSR Example

**1.4.4.5.3.2 MSR Range Example**

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

In the following example the n parameter is a range. The \_n0 value corresponds to MSR0000\_0201, and so on.

**MSR0000\_0201 [Variable-Size MTRRs Mask] (MtrrVarMask)**

Reset: 0000_0000_0000_0000h.	
Core::X86::MtrrVarMask_n[7:0]_lthree[1:0]_core[3:0]:MSR0000_0201[[F,D,B,9,7,5,3,1]]	

Figure 12: Register Instance Table: MSR Range Example

**1.4.4.5.4 BAR Access Method**

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

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

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

**1.4.4.5.4.1 BAR as a Register Reference**

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

- The prefix NTBPRIBAR0 represents the base of the IP, the value of which comes from the register NBIFEPFNCFG::BASE\_ADDR\_1\_aliasHOST\_instNBIF0\_func1[BASE\_ADDR].

**NTBPRIBAR0x00000 (NTB\_SMU\_PCTRL0)**

Reset: 0000_0000h.	
NTB::NTB_SMU_PCTRL0_aliasHOSTPRI; NTBPRIBAR0x00000;	
NTBPRIBAR0=NBIFEPFNCFG::BASE_ADDR_1_aliasHOST_instNBIF0_func1[BASE_ADDR]	
NTB::NTB_SMU_PCTRL0_aliasHOSTSEC; NTBSECBAR0x00000;	
NTBSECBAR0=NBIFEPFNCFG::BASE_ADDR_1_aliasHOST_instNBIF2_func1[BASE_ADDR]	
NTB::NTB_SMU_PCTRL0_aliasSMN; NTBx00000000; NTB=0400_0000h	

Figure 13: Register Instance Table: BAR as Register Reference

**1.4.4.5.5 PCICFG Access Method**



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

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

Example:

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

#### 1.4.4.5.1 PCICFG Bus Implied to be 00h

Example:

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

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

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

#### 1.4.4.5.6 Data Port Access Method

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

Example:

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

DF::FabricBlockInstanceCount_inst[PIE0,BCST]_aliasHOST; D18F0x040_x[00050001,00000000]; DataPortWrite=DF::FabricConfigAccessControl
DF::FabricBlockInstanceCount_inst[PIE0,BCST]_aliasSMN; DFF0x00000040_x[00050001,00000000]; DFF0=0001_C000h;
DataPortWrite=DF::FabricConfigAccessControl

Figure 15: Register Instance Table: Data Port Select

#### 1.4.4.6 Register Field Format

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

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

#### 1.4.4.7 Simple Register Field Format

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

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

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

All fields that don't exist are omitted.

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

Figure 16: Simple Register Field Example

#### 1.4.4.8 Complex Register Field Format

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

Additional sub-rows are added in the following order:

1. Complex expression for {Reset,AccessType,Init,Check}.
2. Instance specific {Reset,AccessType,Init,Check} values.
3. Description, if more than 1 paragraph.
4. Valid values, if more than 0=/1=. Or a Valid bit table. (see figure)

The following figure highlights a complex access type specification.

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

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

The following figure highlights a complex description specification.

4	<b>INVDWBINVD: INVD to WBINVD conversion.</b> Read-write. Reset: 1. Check: 1. 1=Convert INVD to WBINVD.
	<b>Description:</b> This bit is required to be set for normal operation when any of the following are true: <ul style="list-style-type: none"> <li>• An L2 is shared by multiple threads.</li> <li>• An L3 is shared by multiple cores.</li> <li>• CC6 is enabled.</li> <li>• Probe filter is enabled.</li> </ul>

Figure 18: Register Field Sub-Row for Description

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

2:1	<b>CpuWdtTimeBase: CPU watchdog timer time base.</b> Read-write. Reset: 0. Specifies the time base for the timeout period specified in CpuWdtCountSel.										
	<b>ValidValues:</b>										
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>00b</td><td>1.31ms</td></tr> <tr> <td>01b</td><td>1.28us</td></tr> <tr> <td>10b</td><td>Reserved (5ns)</td></tr> <tr> <td>11b</td><td>Reserved</td></tr> </table>	Value	Description	00b	1.31ms	01b	1.28us	10b	Reserved (5ns)	11b	Reserved
Value	Description										
00b	1.31ms										
01b	1.28us										
10b	Reserved (5ns)										
11b	Reserved										

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

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

55:52	Reserved.										
51:48	<b>SliceMask.</b> Read-write. Reset: 0.										
	<b>ValidValues:</b>										
	<table> <tr> <th>Bit</th><th>Description</th></tr> <tr> <td>[0]</td><td>L3 Slice 0 mask.</td></tr> <tr> <td>[1]</td><td>L3 Slice 1 mask.</td></tr> <tr> <td>[2]</td><td>L3 Slice 2 mask.</td></tr> <tr> <td>[3]</td><td>L3 Slice 3 mask.</td></tr> </table>	Bit	Description	[0]	L3 Slice 0 mask.	[1]	L3 Slice 1 mask.	[2]	L3 Slice 2 mask.	[3]	L3 Slice 3 mask.
Bit	Description										
[0]	L3 Slice 0 mask.										
[1]	L3 Slice 1 mask.										
[2]	L3 Slice 2 mask.										
[3]	L3 Slice 3 mask.										

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

#### 1.4.4.9 Field Name is Reserved

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

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

#### 1.4.4.10 Field Access Type

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

Table 8: AccessType Definitions

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

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

#### 1.4.4.10.1 Conditional Access Type Expression

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

- A register field value
- A constant
- A definition

#### 1.4.4.11 Field Reset

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

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

Table 9: Reset Type Definitions

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

#### 1.4.4.12 Field Initialization

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

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

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

*Table 10: Init Type Definitions*

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

#### 1.4.4.13 Field Check

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

#### 1.4.4.14 Field Valid Values

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

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

#### 1.4.5 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.

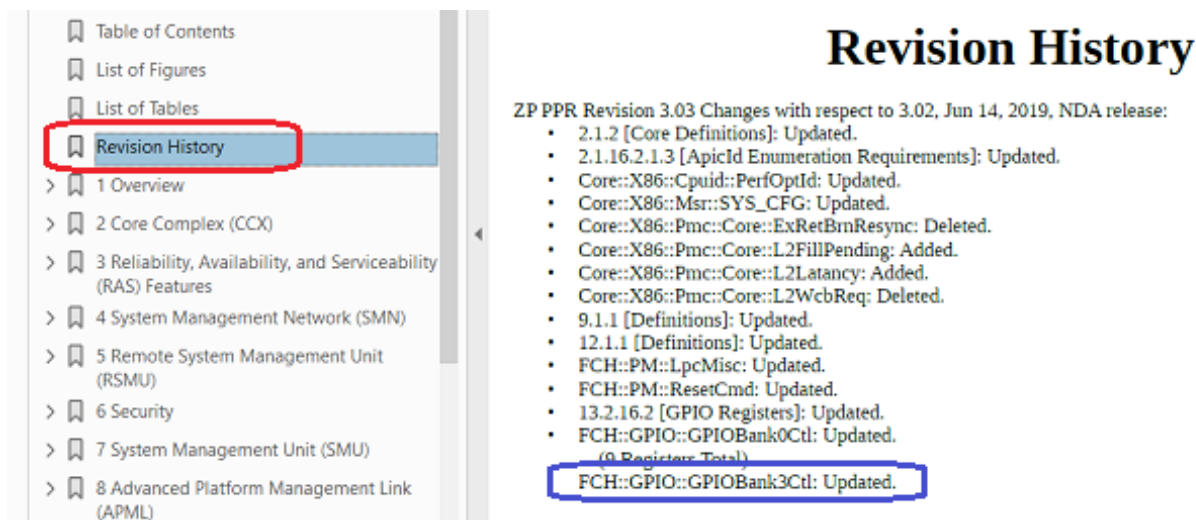


Figure 21: Revision History Format Example

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.

GPIOx003[00...FC] [GPIO Bank 3 Control Register] (FCH::GPIO::GPIOBank3Ctl)	
	Each GPIO pin is controlled by 4 bytes. These registers control GPIO bank 3 pins: GPIO[224:193]-GPIO224].
	_link[63:0]_aliasHOST;
	GPIOx003[FC,F8,F4,F0,EC,E8,E4,E0,DC,D8,D4,D0,CC,C8,C4,C0,BC,B8,B4,B0,AC,A8,A4,A0,9C,98,94,90,8C,88,84,80,7C,78,74,70,6C,68,64,60,5C,58,54,50,4C,48,44,40,3C,38,34,30,2C,28,24,20,1C,18,14,10,0C,08,04,00]; GPIO=FED8_1500h
Bits	Description
31:30	Reserved.
29	WakeSts. Read,Write-1-to-clear. Reset: X. 0=The pin did not generate a wake event. 1=The pin is a wake source. Wake <del>Sts</del> <u>Register</u> status.
28	InterruptSts. Read,Write-1-to-clear. Reset: X. 0=The pin did not generate an interrupt. 1=The pin is an interrupt

Figure 22: 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.
APU	Accelerated Processing Unit.
BatteryPower	The system is running from a limited energy or battery power source or otherwise undocked from a continuous power supply. Setting using this definition may be required to change during run time.
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.
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.
EDC	Electrical design current. Indicates the maximum current the voltage rail can demand for a short, thermally insignificant time.
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.
<b>FDS</b>	Functional Data Sheet. There is one FDS for each package type. See docAM4 or docSFP5.
<b>FID</b>	Frequency Identifier. Specifies the PLL frequency multiplier for a given clock domain.
<b>FreeRunSampleTimer</b>	An internal free running timer used by many power management features.
<b>GB</b>	Gbyte or Gigabyte; 1,073,741,824 bytes.
<b>GT/s</b>	Giga-Transfers per second.
<b>HTC</b>	Hardware Thermal Control.
<b>HTC-active state</b>	Hardware-controlled lower-power, lower performance state used to reduce temperature.
<b>IO configuration</b>	Access to configuration space through IO ports CF8h and CFCh.
<b>IP</b>	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.
<b>KB</b>	Kbyte or Kilobyte; 1024 bytes.
<b>Master abort</b>	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.
<b>MB</b>	Megabyte; 1024 KB.
<b>MMIO</b>	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.
<b>MMIO configuration</b>	Access to configuration space through memory space.
<b>Node</b>	A node, is an integrated circuit device that includes one to 4 cores (one Core Complex).
<b>OW</b>	Octword. An 128-bit value.
<b>PCIe®</b>	PCI Express.
<b>PCS</b>	Physical Coding Sublayer.
<b>Processor</b>	A package containing one or more Nodes. See Node.
<b>QW</b>	Quadword. A 64-bit value.
<b>REFCLK</b>	Reference clock. Refers to the clock frequency (100 MHz) or the clock period (10 ns) depending on the context used.
<b>RX</b>	Receiver.
<b>Shutdown</b>	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.
<b>SMAF</b>	System Management Action Field. This is the code passed from the SMC to the processors in STPCLK assertion messages.
<b>SMC</b>	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.
<b>Speculative event</b>	A performance monitor event counter that counts all occurrences of the event even if the event occurs during speculative code execution.
<b>SSC</b>	Spread Spectrum Clocking.
<b>TDC</b>	Thermal Design Current.
<b>TDP</b>	Thermal Design Power. A power consumption parameter that is used in conjunction with thermal specifications to design appropriate cooling solutions for the processor.
<b>Token</b>	A scheduler entry used in various DF queues to track outstanding requests.
<b>TOM</b>	Top of Memory.
<b>TOM2</b>	Top of extended Memory.
<b>TX</b>	Transmitter.
<b>UMI</b>	Unified Media Interface. The link between the processor and the FCH.
<b>VID</b>	Voltage level identifier.
<b>Warm reset</b>	RESET_L is asserted only (while PWROK stays high).



<b>XBAR</b>	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*

Term	Description
<b>AM4</b>	Desktop, single die, single socket. For client desktop platform (uPGA) DDR4. AM4 = (Core::X86::Cpuid::BrandId[PkgType] == 02h).
<b>FP5</b>	Notebook package for direct solder boards (BGA). FP5 = (Core::X86::Cpuid::BrandId[PkgType] == 00h).

## 1.8 Processor Overview

### 1.8.1 Features

The Family 17h Models 10h-1Fh addition to AMD's offering of Accelerated Processing Units (APU). This System-On-a-Chip (SoC) has been created to meet the needs of energy efficient, performance rich solution for desktop, laptop and tablet computing environments based on the x86 CPU architecture for 8th Generation APUs. It features the introduction of AMD's Scalable Data Fabric (SDF) into this market segment, thus maximizing bandwidth utilization across the system with minimal latencies to boost overall system performance. The SoC is a solution that includes integrated IO, graphics, multimedia, and memory interfaces where no supporting chipset is necessary, resulting in a lower Bill of Materials (BoM) cost.

- Client features:
  - AM4 Desktop class package.
  - FP5 Notebook/Tablet class package.
  - Central Processing Units (CPU):
    - Core Complex (CCX) based on the 8th Generation APU core.
      - Supports Simultaneous Multithreading over previous generation's "Clusters".
    - CCX consists of:
      - Up to 4 cores, where each core may run in single-thread mode (1T) or dual-thread SMT mode (2T) for a total of up to 8 threads for the CCX.



- All cores must be either single-threaded or all cores dual-threaded.
  - 64K, 4-way I-Cache and 32K, 8-way D-Cache per core.
  - 512KB of L2 per core for a total of 2MB L2 contained within the CCX.
  - 1MB of L3 slice per core shared across all cores within the CCX for a total 4MB L3 shared in the CCX.
    - L3 slicing allows for sections of the L3 to be clock and power gated.
    - If all cores of the cluster are power gated, the entire L3 is power gated.
    - Slices are brought up from XC6 power state depending on surpassing the eviction threshold of the L2.
- Integrated Graphics Processing Unit (GPU):
  - Heterogeneous System Architecture (HSA) 2.0 Enabled.
  - OpenGL® 4.X/2.1, OpenGL ES 3.X, EGL 1.4/1.5.
  - OpenCL™ 2.0 (Embedded and full Profile).
  - DirectX® 11/12.
  - RenderScript Graphics and Compute under Android.
  - 2D Graphics:
    - Hardware Blit.
- Scalable Data Fabric.
- System Management Unit (SMU):
  - Platform Security Processor (PSP).
  - Power Management.
  - Thermal Management (THM).
  - Voltage Control (SMUIO).
  - Clock Control.
- Memory:
  - Unified Memory Controller (UMC):
    - Supports Heterogeneous Systems Architecture (HSA), Scalable Data Fabric (SDF), and Control Plane (CP) fabric.
  - DDR4:
    - 2 channels.
- Multimedia Hub (MMHUB):
  - Video Compression Engine (VCE).
  - Universal Video Decoder (UVD).
  - Display Controller Engine (DCN):
    - Supports maximum 4 TMDP ports simultaneously.
  - Content Security:
    - Secure Frame Buffer Memory.
- FCH:
  - Supports an Integrated FCH, with no support for discrete FCH.
  - USB2: one port.
  - USB3: 2 ports at 5 Gbps.
  - USB3.1.
  - SATA up to two lanes Gen1/Gen2/Gen3 and SATA Express.
    - Also provides legacy SATA support for SATAe links.
  - SD4/eMMC5/SDIO4: Up to 1.
  - Up to 2 4-wire UART interfaces.
  - Up to 4 I2C.
  - Low Pin Count (LPC) interface.
  - 2x SMBus.
  - SPI/eSPI with 3 chip selects.
  - I2S interface.
  - Up to 2 10GbE Attachment Unit Interfaces (AUI).
  - Real-Time Clock (RTC).

- Audio:
  - Audio Co-Processor (ACP).
  - Audio DSP for low power audio playback and Soundwire<sup>SM</sup> Technology.
  - High Definition Audio.
- NBIO:
  - PCIe® generation 3 processor.
  - 1 IOHUB with .
  - One 7x16 PCIe® Controller.
    - Note that SATA Express is supported by combining a x2 PCIe® Port and two SATA ports on the same 2 lanes.
  - Non-volatile Memory Express (NVMe) support for PCI Express based interface for solid state devices.
  - Fast MMIO Reads allowing the CPU to read the GPU local memory without sacrificing write support to better enable HSA.
  - nBIF to provide PCIe® software view for the GPU.

The table, Table 13 [PCI Device ID Assignments.], shows the Family 17h, Models 10h-1Fh PCI Vendor ID and Device ID assignments.

*Table 13: PCI Device ID Assignments.*

Vendor ID	Device ID	Bus	Device	Function	Component
1022h	15D0h	0	0	0	Root Complex
1022h	15D1h	0	0	2	IOMMU
1022h	1452h	0	1	0	PCIe® Dummy Host Bridge
1022h	15D3h	0	1	1	PCIe® GPP Bridge 0
1022h	15D3h	0	1	2	PCIe® GPP Bridge 1
1022h	15D3h	0	1	3	PCIe® GPP Bridge 2
1022h	15D3h	0	1	4	PCIe® GPP Bridge 3
1022h	15D3h	0	1	5	PCIe® GPP Bridge 4
1022h	15D3h	0	1	6	PCIe® GPP Bridge 5
1022h	15D3h	0	1	7	PCIe® GPP Bridge 6
1022h	1452h	0	8	0	PCIe® Dummy Host Bridge
1022h	15DBh	0	8	1	Internal PCIe® GPP Bridge 0 to Bus A
1022h	15DCh	0	8	2	Internal PCIe® GPP Bridge 0 to Bus B
1002h	15D8h	Bus A	0	0	Internal GPU (Model 18h)
1002h	15DDh	Bus A	0	0	Internal GPU (Models 10h, 11h)
1002h	15DEh	Bus A	0	1	Display HD Audio Controller
1022h	15E0h	Bus A	0	3	USB 3.1
1022h	15E1h	Bus A	0	4	USB 3.1
1022h	15E2h	Bus A	0	5	Audio Processor
1022h	15E3h	Bus A	0	6	Audio Processor – HD Audio Controller
1022h	15E8h	0	24	0	Data Fabric: Device 18h; Function 0
1022h	15E9h	0	24	1	Data Fabric: Device 18h; Function 1
1022h	15EAh	0	24	2	Data Fabric: Device 18h; Function 2
1022h	15EBh	0	24	3	Data Fabric: Device 18h; Function 3
1022h	15ECh	0	24	4	Data Fabric: Device 18h; Function 4
1022h	15EDh	0	24	5	Data Fabric: Device 18h; Function 5
1022h	15EEh	0	24	6	Data Fabric: Device 18h; Function 6
1022h	7901h	Bus B	0	0	SATA AHCI Mode (MS Driver)

1022h	7904h	Bus B	0	0	SATA AHCI Mode with AMD driver support
1022h	7906h	0	20	6	SD Controller
1022h	790Bh	0	20	0	SMBus Controller
1022h	790Eh	0	20	3	LPC Bridge
1022h	7916h	Bus B	0	0	SATA Controller; SATA Raid/AHCI Mode
1022h	7917h	Bus B	0	0	SATA Controller: SATA Raid AHCI Mode for second vendor
1022h	1458h	Bus B	0	1	10 GbE Controller Port 0
1022h	1458h	Bus B	0	2	10 GbE Controller Port 1
1022h	145Ah	Bus B	0	0	Primary PCIe® Dummy Function

Note: In Table 13 [PCI Device ID Assignments.], programmable bus numbers are labeled A and B. Buses with different labels cannot be assigned the same bus number.

Note: Vendor ID 1002h is used for Internal GPU (15D8h or 15DDh) and Display HD Audio Controller (15DEh).

## 2 Core Complex (CCX)

### 2.1 Processor x86 Core

#### 2.1.1 Core Definitions

Table 14: Definitions

Term	Description
<b>BSC</b>	Boot strap core. Core 0 of the BSP.
<b>BSP</b>	Boot strap processor.
<b>Canonical-address</b>	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].
<b>CCX</b>	Core Complex where more than one core shares L3 resources.
<b>CMP</b>	Specifies the core number.
<b>Core</b>	The instruction execution unit of the processor when the term Core is used in a x86 core context.
<b>CoreCOF</b>	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.
<b>CPL</b>	Current Privilege Level of the running task when the term CPL is used in a x86 core context.
<b>CpuCoreNum</b>	Specifies the core number.
<b>#GP</b>	A general-protection exception.
<b>#GP(0)</b>	Notation indicating a general-protection exception (#GP) with error code of 0.
<b>IBS</b>	Instruction based sampling.
<b>IO configuration</b>	Access to configuration space through IO ports CF8h and CFCh.
<b>IORR</b>	IO range register.
<b>L1 cache</b>	The level 1 caches (instruction cache and the data cache).
<b>L2 cache</b>	The level 2 caches.
<b>L3</b>	Level 3 Cache. The L3 term is also in Addrmaps to enumerate CCX units.
<b>L3 cache</b>	Level 3 Cache.
<b>Linear (virtual) address</b>	The address generated by a core after the segment is applied.
<b>LINT</b>	Local interrupt.
<b>Logical address</b>	The address generated by a core before the segment is applied.
<b>LRU</b>	Least recently used.
<b>LVT</b>	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]).
<b>Macro-op</b>	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.
<b>Micro-op</b>	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.
<b>NBC</b>	NBC=(CPUID Fn00000001_EBX[LocalApicId[3:0]] == 0). Node Base Core. The lowest numbered core in the node.
<b>MTRR</b>	Memory-type range register. The MTRRs specify the type of memory associated with various memory ranges.

<b>NTA</b>	Non-Temporal Access.
<b>PTE</b>	Page table entry.
<b>SMI</b>	System management interrupt.
<b>SMM</b>	System Management Mode.
<b>SMT</b>	Simultaneous multithreading. See Core::X86::Cpuid::CoreId[ThreadsPerCore].
<b>Speculative event</b>	A performance monitor event counter that counts all occurrences of the event even if the event occurs during speculative code execution.
<b>SVM</b>	Secure virtual machine.
<b>Thread</b>	One architectural context for instruction execution.
<b>WDT</b>	Watchdog timer. A timer that detects activity and triggers an error if a specified period of time expires without the activity.

## 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 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.

## 2.1.5 Address Space

### 2.1.5.1 Virtual Address Space

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

### 2.1.5.2 Physical Address Space

The processor supports a 48-bit physical address space. See Core::X86::Cpuid::LongModeInfo.

The processor master aborts the following upper-address transactions (to address PhysAddr):

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

### 2.1.5.3 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.

#### 2.1.5.3.1 Memory Access to the Physical Address Space

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

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

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

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

### 2.1.5.3.1.1 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::Msrb::MTRRdefType, Core::X86::Msrb::MtrrVarBase, Core::X86::Msrb::MtrrVarMask, Core::X86::Msrb::MtrrFix\_64K and Core::X86::Msrb::MtrrFix\_16K\_0 through Core::X86::Msrb::MtrrFix\_4K\_7.
2. TSeg & ASeg SMM mechanism (See Core::X86::Msrb::SMMAddr and Core::X86::Msrb::SMMMMask).
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::Msrb::SMMMMask[AValid] && "The address falls within the ASeg region") || (Core::X86::Msrb::SMMMMask[TValid] && "The address falls within the TSeg region"))) then MemType = CD.

### 2.1.6 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 2.1.6.3 [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::Msrb::HWCR[IoCfgGpFault].
  - SMI trapping for these accesses is specified by Core::X86::Msrb::SMI\_ON\_IO\_TRAP\_CTL\_STS and Core::X86::Msrb::SMI\_ON\_IO\_TRAP.
- MMIO configuration: configuration space is a region of memory space.
  - The base address and size of this range is specified by Core::X86::Msrb::MmioCfgBaseAddr. The size is controlled by the number of configuration-space bus numbers supported by the system. Accesses to this range are converted configuration space as follows:
- Address[31:0] = {0h, bus[7:0], device[4:0], function[2:0], offset[11:0]}.

The BIOS may use either configuration space access mechanism during boot. Before booting the OS, BIOS must disable

IO access to ECS, enable MMIO configuration and build an ACPI defined MCFG table. BIOS ACPI code must use MMIO to access configuration space.

### 2.1.6.1 MMIO Configuration Coding Requirements

MMIO configuration space accesses must use the uncachable (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.

### 2.1.6.2 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.

### 2.1.6.3 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.

### 2.1.7 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 2.1.6 [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	<b>ConfigEn: configuration space enable.</b> 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	<b>ExtRegNo: extended register number.</b> 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	<b>BusNo: bus number.</b> Read-write. Reset: 00h. Specifies the bus number of the configuration cycle.
15:11	<b>Device: device number.</b> Read-write. Reset: 00h. Specifies the device number of the configuration cycle.
10:8	<b>Function.</b> Read-write. Reset: 0h. Specifies the function number of the configuration cycle.
7:2	<b>RegNo: register address.</b> 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	<b>Data.</b> Read-write. Reset: 0000_0000h. See IO::IoCfgAddr.

### 2.1.8 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.

### 2.1.9 Register Sharing

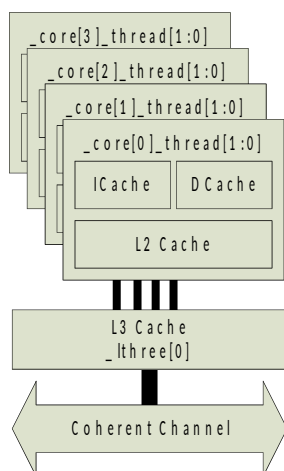


Figure 23: Register Sharing Domains

**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	<b>TSC: time stamp counter.</b> 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 24: Instance Parameters

Instances of core registers are designated as 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.

**2.1.10 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.

**2.1.11 Interrupts****2.1.11.1 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.

**2.1.11.1.1 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.

**2.1.11.1.2 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.

#### 2.1.11.1.3 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.

#### 2.1.11.1.4 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 15: 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.

### 2.1.11.1.5 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 16: 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).

#### 2.1.11.1.6 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	<b>Port: trapped IO port address.</b> Read-only, Volatile. Reset: 0000h. This provides the address of the IO instruction.
15:12	<b>BPR: IO breakpoint match.</b> Read-only, Volatile. Reset: 0h.
11	<b>TF: EFLAGS TF value.</b> Read-only, Volatile. Reset: 0.
10:7	Reserved.
6	<b>SZ32: size 32 bits.</b> Read-only, Volatile. Reset: 0. 1=Port access was 32 bits.
5	<b>SZ16: size 16 bits.</b> Read-only, Volatile. Reset: 0. 1=Port access was 16 bits.
4	<b>SZ8: size 8 bits.</b> Read-only, Volatile. Reset: 0. 1=Port access was 8 bits.
3	<b>REP: repeated port access.</b> Read-only, Volatile. Reset: 0.
2	<b>STR: string-based port access.</b> Read-only, Volatile. Reset: 0.
1	<b>V: IO trap word valid.</b> 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	<b>RW: port access type.</b> 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	<b>McRedirSts: machine check exception redirection status.</b> 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	<b>IoTrapSts: IO trap status.</b> 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	<b>RST: SMM IO Restart Byte.</b> Read-write. Reset: 00h.

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

Read-write. Reset: 00h.

Bits	Description
7:1	Reserved.
0	<b>HLT: halt restart.</b> 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	<b>NmiMask: NMI Mask.</b> 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	<b>HostEflagesIF</b> : <b>host EFLAGS IF</b> . Read-only, Volatile. Reset: 0.
2:0	<b>SvmState</b> . Read-only, Volatile. Reset: 0h.
<b>Valid Values:</b>	
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	<b>BRL</b> . Read-only. Reset: 1. 1=Base relocation supported.
16	<b>IOTrap</b> . Read-only. Reset: 1. 1=IO trap supported.
15:0	<b>Revision</b> . 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	<b>SmmBase</b> . Read-write, Volatile. Reset: 0000_0000h. See Core::X86::Msrr::SMM_BASE[SmmBase].

### 2.1.11.1.7 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.

### 2.1.11.1.8 The Protected ASeg and TSeg Areas

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



### 2.1.11.1.9 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::Msrr::HWCR[RsmSpCycDis,SmiSpCycDis].

### 2.1.11.1.10 Locking SMM

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

### 2.1.11.2 Local APIC

Family 17h, Model 11h supports the APIC interrupt controller.  
See 2.1.11.2.2 [Local APIC Registers] for the APIC registers.

#### 2.1.11.2.1 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.

#### 2.1.11.2.1.1 Detecting and Enabling

The presence of APIC is detected via Core::X86::Cpuid::FeatureIdEdx[APIC].  
The local APIC is enabled via Core::X86::Msrr::APIC\_BAR[ApicEn]. Reset forces the APIC disabled.

#### 2.1.11.2.1.2 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::Msrr::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::Msrr::APIC\_BAR[ApicEn].

### 2.1.11.2.1.3 ApicId Enumeration Requirements

Note: Family 17h 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]`.

### 2.1.11.2.1.4 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.

### 2.1.11.2.1.5 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.

### 2.1.11.2.1.6 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.

### 2.1.11.2.1.7 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.

#### **2.1.11.2.1.8 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.

#### **2.1.11.2.1.9 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.

#### **2.1.11.2.1.10 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.

#### **2.1.11.2.1.11 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[ISRVec[7:4]], 0h}
}
```

#### 2.1.11.2.1.12 Inter-Processor Interrupts

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

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

Not all combinations of ICR fields are valid. Only the following combinations are valid:

Note: x indicates a don't care.

Table 17: 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

#### 2.1.11.2.1.13 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.

#### 2.1.11.2.1.14 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.

#### 2.1.11.2.1.15 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.

#### 2.1.11.2.2 Local APIC Registers

APICx020 [APIC ID] (Core::X86::Apic::ApicId)	
Read-only.	
_lthree0_core[3:0]_thread[1:0]; APICx020; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}	
Bits	Description
31:24	<b>ApicId: APIC ID.</b> Read-only. Reset: XXh. The reset value varies based on core number. See 2.1.11.2.1.3 [ApicId Enumeration Requirements].
23:0	Reserved.

APICx030 [APIC Version] (Core::X86::Apic::ApicVersion)	
Read-only.	
_lthree0_core[3:0]_thread[1:0]; APICx030; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}	
Bits	Description
31	<b>ExtApicSpace: extended APIC register space present.</b> Read-only. Reset: 1. 1=Indicates the presence of extended APIC register space starting at Core::X86::Apic::ExtendedApicFeature.
30:25	Reserved.
24	<b>DirectedEoiSupport: directed EOI support.</b> Read-only. Reset: Fixed,0. 0=Directed EOI capability not supported.
23:16	<b>MaxLvtEntry.</b> Read-only. Reset: XXh. Specifies the number of entries in the local vector table minus one.
15:8	Reserved.

7:0	<b>Version.</b> Read-only. Reset: 10h. Indicates the version number of this APIC implementation.
-----	--

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

Read-write. Reset: 0000\_0000h.

\_lthree0\_core[3:0]\_thread[1:0]; APICx080; APIC={Core::X86::Msr::APIC\_BAR[ApicBar[47:12]] , 000h}

Bits	Description
31:8	Reserved.
7:0	<b>Priority.</b> 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.

\_lthree0\_core[3:0]\_thread[1:0]; APICx090; APIC={Core::X86::Msr::APIC\_BAR[ApicBar[47:12]] , 000h}

Bits	Description
31:8	Reserved.
7:0	<b>Priority.</b> 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.

\_lthree0\_core[3:0]\_thread[1:0]; APICx0A0; APIC={Core::X86::Msr::APIC\_BAR[ApicBar[47:12]] , 000h}

Bits	Description
31:8	Reserved.
7:0	<b>Priority.</b> 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.

\_lthree0\_core[3:0]\_thread[1:0]; APICx0B0; APIC={Core::X86::Msr::APIC\_BAR[ApicBar[47:12]] , 000h}

Bits	Description
31:0	Reserved.

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

Read-only. Reset: 0000\_0000h.

Remote Read is deprecated.

\_lthree0\_core[3:0]\_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.

\_lthree0\_core[3:0]\_thread[1:0]; APICx0D0; APIC={Core::X86::Msr::APIC\_BAR[ApicBar[47:12]] , 000h}

Bits	Description
31:24	<b>Destination.</b> 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.

_lthree0_core[3:0]_thread[1:0]; APICx0E0; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}									
Bits	Description								
31:28	<b>Format.</b> Read-write. Reset: Fh. Controls which format to use when accepting interrupts with a logical destination mode.								
	<b>ValidValues:</b>								
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>Cluster destinations are used.</td></tr> <tr> <td>Eh-1h</td><td>Reserved.</td></tr> <tr> <td>Fh</td><td>Flat destinations are used.</td></tr> </table>	Value	Description	0h	Cluster destinations are used.	Eh-1h	Reserved.	Fh	Flat destinations are used.
Value	Description								
0h	Cluster destinations are used.								
Eh-1h	Reserved.								
Fh	Flat destinations are used.								
27:0	Reserved.								

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

Reset: 0000\_00FFh.

\_lthree0\_core[3:0]\_thread[1:0]; APICx0F0; APIC={Core::X86::Msr::APIC\_BAR[ApicBar[47:12]] , 000h}

Bits	Description
31:10	Reserved.
9	<b>FocusDisable.</b> Read-write. Reset: 0. 1=Disable focus core checking during lowest-priority arbitrated interrupts.
8	<b>APICSWEn: APIC software enable.</b> 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	<b>Vector.</b> 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.

\_lthree0\_core[3:0]\_thread[1:0]\_n0; APICx100; APIC={Core::X86::Msr::APIC\_BAR[ApicBar[47:12]] , 000h}

\_lthree0\_core[3:0]\_thread[1:0]\_n1; APICx110; APIC={Core::X86::Msr::APIC\_BAR[ApicBar[47:12]] , 000h}

\_lthree0\_core[3:0]\_thread[1:0]\_n2; APICx120; APIC={Core::X86::Msr::APIC\_BAR[ApicBar[47:12]] , 000h}

\_lthree0\_core[3:0]\_thread[1:0]\_n3; APICx130; APIC={Core::X86::Msr::APIC\_BAR[ApicBar[47:12]] , 000h}

\_lthree0\_core[3:0]\_thread[1:0]\_n4; APICx140; APIC={Core::X86::Msr::APIC\_BAR[ApicBar[47:12]] , 000h}

\_lthree0\_core[3:0]\_thread[1:0]\_n5; APICx150; APIC={Core::X86::Msr::APIC\_BAR[ApicBar[47:12]] , 000h}

\_lthree0\_core[3:0]\_thread[1:0]\_n6; APICx160; APIC={Core::X86::Msr::APIC\_BAR[ApicBar[47:12]] , 000h}

\_lthree0\_core[3:0]\_thread[1:0]\_n7; APICx170; APIC={Core::X86::Msr::APIC\_BAR[ApicBar[47:12]] , 000h}

Bits	Description
31:0	<b>InServiceBits.</b> 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.

\_lthree0\_core[3:0]\_thread[1:0]\_n0; APICx180; APIC={Core::X86::Msr::APIC\_BAR[ApicBar[47:12]] , 000h}

\_lthree0\_core[3:0]\_thread[1:0]\_n1; APICx190; APIC={Core::X86::Msr::APIC\_BAR[ApicBar[47:12]] , 000h}

\_lthree0\_core[3:0]\_thread[1:0]\_n2; APICx1A0; APIC={Core::X86::Msr::APIC\_BAR[ApicBar[47:12]] , 000h}

\_lthree0\_core[3:0]\_thread[1:0]\_n3; APICx1B0; APIC={Core::X86::Msr::APIC\_BAR[ApicBar[47:12]] , 000h}

\_lthree0\_core[3:0]\_thread[1:0]\_n4; APICx1C0; APIC={Core::X86::Msr::APIC\_BAR[ApicBar[47:12]] , 000h}

\_lthree0\_core[3:0]\_thread[1:0]\_n5; APICx1D0; APIC={Core::X86::Msr::APIC\_BAR[ApicBar[47:12]] , 000h}

\_lthree0\_core[3:0]\_thread[1:0]\_n6; APICx1E0; APIC={Core::X86::Msr::APIC\_BAR[ApicBar[47:12]] , 000h}

\_lthree0\_core[3:0]\_thread[1:0]\_n7; APICx1F0; APIC={Core::X86::Msr::APIC\_BAR[ApicBar[47:12]] , 000h}

Bits	Description
31:0	<b>TriggerModeBits.</b> 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.
	<b>ValidValues:</b>



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.

\_lthree0\_core[3:0]\_thread[1:0]\_n0; APICx200; APIC={Core::X86::Msr::APIC\_BAR[ApicBar[47:12]] , 000h}

\_lthree0\_core[3:0]\_thread[1:0]\_n1; APICx210; APIC={Core::X86::Msr::APIC\_BAR[ApicBar[47:12]] , 000h}

\_lthree0\_core[3:0]\_thread[1:0]\_n2; APICx220; APIC={Core::X86::Msr::APIC\_BAR[ApicBar[47:12]] , 000h}

\_lthree0\_core[3:0]\_thread[1:0]\_n3; APICx230; APIC={Core::X86::Msr::APIC\_BAR[ApicBar[47:12]] , 000h}

\_lthree0\_core[3:0]\_thread[1:0]\_n4; APICx240; APIC={Core::X86::Msr::APIC\_BAR[ApicBar[47:12]] , 000h}

\_lthree0\_core[3:0]\_thread[1:0]\_n5; APICx250; APIC={Core::X86::Msr::APIC\_BAR[ApicBar[47:12]] , 000h}

\_lthree0\_core[3:0]\_thread[1:0]\_n6; APICx260; APIC={Core::X86::Msr::APIC\_BAR[ApicBar[47:12]] , 000h}

\_lthree0\_core[3:0]\_thread[1:0]\_n7; APICx270; APIC={Core::X86::Msr::APIC\_BAR[ApicBar[47:12]] , 000h}

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

\_lthree0\_core[3:0]\_thread[1:0]; APICx280; APIC={Core::X86::Msr::APIC\_BAR[ApicBar[47:12]] , 000h}

Bits	Description
31:8	Reserved.
7	<b>IllegalRegAddr: illegal register address.</b> 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	<b>RcvdIllegalVector: received illegal vector.</b> 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	<b>SentIllegalVector.</b> 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	<b>RcvAcceptError: receive accept error.</b> Read-write. Reset: 0. This bit indicates that a message received by this APIC was not accepted by this or any other APIC.

2	<b>SendAcceptError.</b> 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.

\_lthree0\_core[3:0]\_thread[1:0]; APICx300; APIC={Core::X86::Msr::APIC\_BAR[ApicBar[47:12]] , 000h}

Bits	Description																		
31:20	Reserved.																		
19:18	<b>DestShrthnd: destination shorthand.</b> Read-write. Reset: 0h. <b>Description:</b> 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. <b>ValidValues:</b>																		
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>No shorthand (Destination field).</td></tr> <tr> <td>1h</td><td>Self.</td></tr> <tr> <td>2h</td><td>All including self.</td></tr> <tr> <td>3h</td><td>All excluding self (This sends a message with a destination encoding of all 1s, so if lowest priority is used the message could end up being reflected back to this APIC).</td></tr> </table>	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).								
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	<b>RemoteRdStat.</b> Read-only. Reset: 0h. <b>ValidValues:</b>																		
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>Read was invalid.</td></tr> <tr> <td>1h</td><td>Delivery pending.</td></tr> <tr> <td>2h</td><td>Delivery complete and access was valid.</td></tr> <tr> <td>3h</td><td>Reserved.</td></tr> </table>	Value	Description	0h	Read was invalid.	1h	Delivery pending.	2h	Delivery complete and access was valid.	3h	Reserved.								
Value	Description																		
0h	Read was invalid.																		
1h	Delivery pending.																		
2h	Delivery complete and access was valid.																		
3h	Reserved.																		
15	<b>TM: trigger mode.</b> Read-write. Reset: 0. 0=Edge triggered. 1=Level triggered. Indicates how this interrupt is triggered.																		
14	<b>Level.</b> Read-write. Reset: 0. 0=De-asserted. 1=Asserted.																		
13	Reserved.																		
12	<b>DS: interrupt delivery status.</b> 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	<b>DM: destination mode.</b> Read-write. Reset: 0. 0=Physical. 1=Logical.																		
10:8	<b>MsgType.</b> Read-write. Reset: 0h. The message types are encoded as follows: <b>ValidValues:</b>																		
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>Fixed.</td></tr> <tr> <td>1h</td><td>Lowest Priority.</td></tr> <tr> <td>2h</td><td>SMI.</td></tr> <tr> <td>3h</td><td>Reserved.</td></tr> <tr> <td>4h</td><td>NMI.</td></tr> <tr> <td>5h</td><td>INIT.</td></tr> <tr> <td>6h</td><td>Startup.</td></tr> <tr> <td>7h</td><td>External interrupt.</td></tr> </table>	Value	Description	0h	Fixed.	1h	Lowest Priority.	2h	SMI.	3h	Reserved.	4h	NMI.	5h	INIT.	6h	Startup.	7h	External interrupt.
Value	Description																		
0h	Fixed.																		
1h	Lowest Priority.																		
2h	SMI.																		
3h	Reserved.																		
4h	NMI.																		
5h	INIT.																		
6h	Startup.																		
7h	External interrupt.																		
7:0	<b>Vector.</b> 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.

_lthree0_core[3:0]_thread[1:0]; APICx310; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}	
Bits	Description
31:24	<b>DestinationField.</b> 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.

\_lthree0\_core[3:0]\_thread[1:0]; APICx320; APIC={Core::X86::Msr::APIC\_BAR[ApicBar[47:12]] , 000h}

Bits	Description
31:18	Reserved.
17	<b>Mode.</b> Read-write. Reset: 0. 0=One-shot. 1=Periodic.
16	<b>Mask.</b> Read-write. Reset: 1. 0=Not masked. 1=Masked.
15:13	Reserved.
12	<b>DS: interrupt delivery status.</b> 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	<b>MsgType: message type.</b> Read-write. Reset: 0h. See 2.1.11.2.1.14 [Generalized Local Vector Table].
7:0	<b>Vector.</b> Read-write. Reset: 00h. Interrupt vector number.

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

Reset: 0001\_0000h.

\_lthree0\_core[3:0]\_thread[1:0]; APICx330; APIC={Core::X86::Msr::APIC\_BAR[ApicBar[47:12]] , 000h}

Bits	Description
31:17	Reserved.
16	<b>Mask.</b> Read-write. Reset: 1. 0=Not masked. 1=Masked.
15:13	Reserved.
12	<b>DS: interrupt delivery status.</b> 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	<b>MsgType: message type.</b> Read-write. Reset: 0h. See 2.1.11.2.1.14 [Generalized Local Vector Table].
7:0	<b>Vector.</b> 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\_CTL(Performance Event Select [3:0]).
- Core::X86::Msr::PERF\_CTL(Performance Event Select [5:0]).

\_lthree0\_core[3:0]\_thread[1:0]; APICx340; APIC={Core::X86::Msr::APIC\_BAR[ApicBar[47:12]] , 000h}

Bits	Description
31:17	Reserved.
16	<b>Mask.</b> Read-write. Reset: 1. 0=Not masked. 1=Masked.
15:13	Reserved.
12	<b>DS: interrupt delivery status.</b> 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	<b>MsgType: message type.</b> Read-write. Reset: 0h. See 2.1.11.2.1.14 [Generalized Local Vector Table].
7:0	<b>Vector.</b> Read-write. Reset: 00h. Interrupt vector number.

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

Reset: 0001\_0000h.

\_lthree0\_core[3:0]\_thread[1:0]\_n0; APICx350; APIC={Core::X86::Msr::APIC\_BAR[ApicBar[47:12]] , 000h}

_lthree0_core[3:0]_thread[1:0]_n1; APICx360; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}	
Bits	Description
31:17	Reserved.
16	<b>Mask.</b> Read-write. Reset: 1. 0=Not masked. 1=Masked.
15	<b>TM: trigger mode.</b> Read-write. Reset: 0. 0=Edge. 1=Level.
14	<b>RmtIRR.</b> 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	<b>DS: interrupt delivery status.</b> 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	<b>MsgType: message type.</b> Read-write. Reset: 0h. See 2.1.11.2.1.14 [Generalized Local Vector Table].
7:0	<b>Vector.</b> Read-write. Reset: 00h. Interrupt vector number.

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

Reset: 0001\_0000h.

\_lthree0\_core[3:0]\_thread[1:0]; APICx370; APIC={Core::X86::Msr::APIC\_BAR[ApicBar[47:12]] , 000h}

Bits	Description
31:17	Reserved.
16	<b>Mask.</b> Read-write. Reset: 1. 0=Not masked. 1=Masked.
15:13	Reserved.
12	<b>DS: interrupt delivery status.</b> 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	<b>MsgType: message type.</b> Read-write. Reset: 0h. See 2.1.11.2.1.14 [Generalized Local Vector Table].
7:0	<b>Vector.</b> Read-write. Reset: 00h. Interrupt vector number.

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

Read-write, Volatile. Reset: 0000\_0000h.

\_lthree0\_core[3:0]\_thread[1:0]; APICx380; APIC={Core::X86::Msr::APIC\_BAR[ApicBar[47:12]] , 000h}

Bits	Description
31:0	<b>Count.</b> 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.

\_lthree0\_core[3:0]\_thread[1:0]; APICx390; APIC={Core::X86::Msr::APIC\_BAR[ApicBar[47:12]] , 000h}

Bits	Description
31:0	<b>Count.</b> 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.

\_lthree0\_core[3:0]\_thread[1:0]; APICx3E0; APIC={Core::X86::Msr::APIC\_BAR[ApicBar[47:12]] , 000h}

Bits	Description
31:4	Reserved.
3:0	<b>Div[3:0].</b> Read-write. Reset: 0h. Div[2] is unused.
<b>Valid Values:</b>	
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.

\_lthree0\_core[3:0]\_thread[1:0]; APICx400; APIC={Core::X86::Msr::APIC\_BAR[ApicBar[47:12]] , 000h}

Bits	Description
31:24	Reserved.
23:16	<b>ExtLvtCount: extended local vector table count.</b> Read-only. Reset: 04h. This specifies the number of extended LVT registers (Core::X86::Apic::ExtendedInterruptLvtEntries) in the local APIC.
15:3	Reserved.
2	<b>ExtApicIdCap: extended APIC ID capable.</b> 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	<b>SeoiCap: specific end of interrupt capable.</b> Read-only. Reset: 1. 1=The Core::X86::Apic::SpecificEndOfInterrupt is present.
0	<b>IerCap: interrupt enable register capable.</b> Read-only. Reset: 1. This bit indicates that the Core::X86::Apic::InterruptEnable are present. See 2.1.11.2.1.8 [Interrupt Masking].

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

Read-write. Reset: 0000\_0000h.

\_lthree0\_core[3:0]\_thread[1:0]; APICx410; APIC={Core::X86::Msr::APIC\_BAR[ApicBar[47:12]] , 000h}

Bits	Description
31:3	Reserved.
2	<b>ExtApicIdEn: extended APIC ID enable.</b> 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	<b>SeoiEn.</b> Read-write. Reset: 0. 1=Enable SEOI generation when a Write to Core::X86::Apic::SpecificEndOfInterrupt is received.
0	<b>IerEn.</b> 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.

\_lthree0\_core[3:0]\_thread[1:0]; APICx420; APIC={Core::X86::Msr::APIC\_BAR[ApicBar[47:12]] , 000h}

Bits	Description
31:8	Reserved.
7:0	<b>EoiVec: end of interrupt vector.</b> 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.

\_lthree0\_core[3:0]\_thread[1:0]\_n0; APICx480; APIC={Core::X86::Msr::APIC\_BAR[ApicBar[47:12]] , 000h}

\_lthree0\_core[3:0]\_thread[1:0]\_n1; APICx490; APIC={Core::X86::Msr::APIC\_BAR[ApicBar[47:12]] , 000h}

\_lthree0\_core[3:0]\_thread[1:0]\_n2; APICx4A0; APIC={Core::X86::Msr::APIC\_BAR[ApicBar[47:12]] , 000h}

\_lthree0\_core[3:0]\_thread[1:0]\_n3; APICx4B0; APIC={Core::X86::Msr::APIC\_BAR[ApicBar[47:12]] , 000h}

_lthree0_core[3:0]_thread[1:0]_n4; APICx4C0; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}	
_lthree0_core[3:0]_thread[1:0]_n5; APICx4D0; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}	
_lthree0_core[3:0]_thread[1:0]_n6; APICx4E0; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}	
_lthree0_core[3:0]_thread[1:0]_n7; APICx4F0; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}	
Bits	Description
31:0	<b>InterruptEnableBits.</b> 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].

_lthree0_core[3:0]_thread[1:0]_n0; APICx500; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}	
_lthree0_core[3:0]_thread[1:0]_n1; APICx510; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}	
_lthree0_core[3:0]_thread[1:0]_n2; APICx520; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}	
_lthree0_core[3:0]_thread[1:0]_n3; APICx530; APIC={Core::X86::Msr::APIC_BAR[ApicBar[47:12]] , 000h}	
Bits	Description
31:17	Reserved.
16	<b>Mask.</b> Read-write. Reset: 1. 0=Not masked. 1=Masked.
15:13	Reserved.
12	<b>DS: interrupt delivery status.</b> 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	<b>MsgType: message type.</b> Read-write. Reset: 0h. See 2.1.11.2.1.14 [Generalized Local Vector Table].
7:0	<b>Vector.</b> Read-write. Reset: 00h. Interrupt vector number.

## 2.1.12 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.

### 2.1.12.1 CPUID Instruction Functions

#### CPUID\_Fn00000000\_EAX [Processor Vendor and Largest Standard Function Number] (Core::X86::Cpuid::LargFuncNum)

Read-only. Reset: Fixed, 0000\_000Dh.

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn00000000\_EAX

Bits	Description
31:0	<b>LFuncStd: largest standard function.</b> Read-only. Reset: Fixed, 0000_000Dh. 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.	
_lthree0_core[3:0]_thread[1:0]; CPUID_Fn00000000_EBX	
Bits	Description
31:0	<b>Vendor.</b> 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.	
_lthree0_core[3:0]_thread[1:0]; CPUID_Fn00000000_ECX	
Bits	Description
31:0	<b>Vendor.</b> 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.	
_lthree0_core[3:0]_thread[1:0]; CPUID_Fn00000000_EDX	
Bits	Description
31:0	<b>Vendor.</b> 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]).	
<ul style="list-style-type: none"> <li>E.g., If BaseFamily[3:0] == Fh and ExtendedFamily[7:0] == 08h, then Family[7:0] = 17h.</li> </ul>	
Model: Is an 8-bit value and is defined as: Model[7:0]={ExtendedModel[3:0],BaseModel[3:0]}.	
<ul style="list-style-type: none"> <li>E.g., If ExtendedModel[3:0] == 1h and BaseModel[3:0] == 8h, then Model[7:0] = 18h.</li> <li>Model numbers vary with product.</li> </ul>	
Model numbers are assigned a letter, 0h = "A", 1h = "B", and so on. Model and Stepping form the Revision. E.g., B1.	
_lthree0_core[3:0]_thread[1:0]; CPUID_Fn00000001_EAX	
Bits	Description
31:28	Reserved.
27:20	<b>ExtFamily: extended family.</b> Read-only. Reset: 08h. See Family above.
19:16	<b>ExtModel: extended model.</b> Read-only. Reset: 1h. See Model above.
15:12	Reserved.
11:8	<b>BaseFamily.</b> Read-only. Reset: Fh. See Family description above.
7:4	<b>BaseModel.</b> Read-only. Reset: Xh. Model numbers vary with product.
3:0	<b>Stepping.</b> Read-only. Reset: 1h. Processor stepping (revision) for a specific model.

#### **CPUID\_Fn00000001\_EBX [LocalApicId, LogicalProcessorCount, CLFlush] (Core::X86::Cpuid::FeatureIdEbx)**

Read-only.	
_lthree0_core[3:0]_thread[1:0]; CPUID_Fn00000001_EBX	
Bits	Description
31:24	<b>LocalApicId.</b> Read-only. Reset: XXh. Initial local APIC physical ID.
23:16	<b>LogicalProcessorCount: logical processor count.</b> Read-only. Reset: Fixed,(Core::X86::Cpuid::SizeId[NC] + 1). Specifies the number of threads in the processor as Core::X86::Cpuid::SizeId[NC] + 1.
15:8	<b>CLFlush.</b> 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::Msrr::CPUID_Features.	
_lthree0_core[3:0]_thread[1:0]; CPUID_Fn00000001_ECX	
Bits	Description
31	Reserved.
30	<b>RDRAND</b> . Read-only. Reset: Fixed,1. RDRAND instruction support.
29	<b>F16C</b> . Read-only. Reset: Fixed,1. Half-precision convert instruction support.
28	<b>AVX</b> . Read-only. Reset: Fixed,1. AVX instruction support.
27	<b>OSXSAVE</b> . 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	<b>XSAVE</b> . 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	<b>AES: AES instruction support</b> . Read-only. Reset: X. AES instruction support.
24	Reserved.
23	<b>POPCNT</b> . Read-only. Reset: Fixed,1. POPCNT instruction.
22	<b>MOVBE</b> . Read-only. Reset: Fixed,1. MOVBE instruction support.
21	<b>X2APIC</b> . Read-only. Reset: Fixed,0. x2APIC capability.
20	<b>SSE42</b> . Read-only. Reset: Fixed,1. SSE4.2 instruction support.
19	<b>SSE41</b> . Read-only. Reset: Fixed,1. SSE4.1 instruction support.
18:14	Reserved.
13	<b>CMPXCHG16B</b> . Read-only. Reset: Fixed,1. CMPXCHG16B instruction.
12	<b>FMA</b> . Read-only. Reset: Fixed,1. FMA instruction support.
11:10	Reserved.
9	<b>SSSE3</b> . Read-only. Reset: Fixed,1. Supplemental SSE3 extensions.
8:4	Reserved.
3	<b>Monitor</b> . Read-only. Reset: !Core::X86::Msrr::HWCR[MonMwaitDis]. Monitor/Mwait instructions.
2	Reserved.
1	<b>PCLMULQDQ</b> . Read-only. Reset: X. PCLMULQDQ instruction support.
0	<b>SSE3</b> . 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::Msrr::CPUID_Features.	
_lthree0_core[3:0]_thread[1:0]; CPUID_Fn00000001_EDX	
Bits	Description
31:29	Reserved.
28	<b>HTT</b> . 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	<b>SSE2</b> . Read-only. Reset: Fixed,1. SSE2: SSE2 extensions.
25	<b>SSE</b> . Read-only. Reset: Fixed,1. SSE extensions.
24	<b>FXSR</b> . Read-only. Reset: Fixed,1. FXSAVE and FXRSTOR instructions.
23	<b>MMX</b> . Read-only. Reset: Fixed,1. MMX instructions
22:20	Reserved.
19	<b>CLFSH</b> . Read-only. Reset: Fixed,1. CLFLUSH instruction.
18	Reserved.
17	<b>PSE36</b> . Read-only. Reset: Fixed,1. Page-size extensions.
16	<b>PAT</b> . Read-only. Reset: Fixed,1. Page attribute table.



15	<b>CMOV</b> . Read-only. Reset: Fixed,1. Conditional move instructions, CMOV, FCOMI, FCMOV.
14	<b>MCA</b> . Read-only. Reset: Fixed,1. Machine check architecture, MCG_CAP.
13	<b>PGE</b> . Read-only. Reset: Fixed,1. Page global extension, CR4.PGE.
12	<b>MTRR</b> . Read-only. Reset: Fixed,1. Memory-type range registers.
11	<b>SysEnterSysExit</b> . Read-only. Reset: Fixed,1. SYSENTER and SYSEXIT instructions.
10	Reserved.
9	<b>APIC: advanced programmable interrupt controller (APIC) exists and is enabled</b> . Read-only. Reset: X. Core::X86::Msrr::APIC_BAR[ApicEn].
8	<b>CMPXCHG8B</b> . Read-only. Reset: Fixed,1. CMPXCHG8B instruction.
7	<b>MCE</b> . Read-only. Reset: Fixed,1. Machine check exception, CR4.MCE.
6	<b>PAE</b> . Read-only. Reset: Fixed,1. Physical-address extensions (PAE).
5	<b>MSR</b> . Read-only. Reset: Fixed,1. AMD model-specific registers (MSRs), with RDMSR and WRMSR instructions.
4	<b>TSC</b> . Read-only. Reset: Fixed,1. Time Stamp Counter, RDTSC/RDTSCP instructions, CR4.TSD.
3	<b>PSE</b> . Read-only. Reset: Fixed,1. Page-size extensions (4 MB pages).
2	<b>DE</b> . Read-only. Reset: Fixed,1. Debugging extensions, IO breakpoints, CR4.DE.
1	<b>VME</b> . Read-only. Reset: Fixed,1. Virtual-mode enhancements.
0	<b>FPU</b> . 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.

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn00000005\_EAX

Bits	Description
31:16	Reserved.
15:0	<b>MonLineSizeMin</b> . 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.

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn00000005\_EBX

Bits	Description
31:16	Reserved.
15:0	<b>MonLineSizeMax</b> . 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.

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn00000005\_ECX

Bits	Description
31:2	Reserved.
1	<b>IBE</b> . Read-only. Reset: Fixed,1. Interrupt break-event.
0	<b>EMX</b> . Read-only. Reset: Fixed,1. Enumerate MONITOR/MWAIT extensions.

**CPUID\_Fn00000005\_EDX [Monitor/MWait] (Core::X86::Cpuid::MonMWaitEdx)**

Read-only. Reset: Fixed,0000\_0011h.

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn00000005\_EDX

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

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn00000006\_EAX

Bits	Description
31:3	Reserved.
2	<b>ARAT: always running APIC timer.</b> 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)**

\_lthree0\_core[3:0]\_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::Msrr::CPUID\_PWR\_THERM.

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn00000006\_ECX

Bits	Description
31:1	Reserved.
0	<b>EffFreq: effective frequency interface.</b> Read-only. Reset: Fixed,1. 1=Indicates presence of Core::X86::Msrr::MPERF and Core::X86::Msrr::APERF.

**CPUID\_Fn00000006\_EDX [Thermal and Power Management] (Core::X86::Cpuid::ThermalPwrMgmtEdx)**

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn00000006\_EDX

Bits	Description
31:0	Reserved.

**CPUID\_Fn00000007\_EAX\_x00 [Structured Extended Feature Identifiers] (Core::X86::Cpuid::StructExtFeatIdEax0)**

Read-only. Reset: Fixed,0000\_0000h.

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn00000007\_EAX\_x00

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

**CPUID\_Fn00000007\_EBX\_x00 [Structured Extended Feature Identifiers] (Core::X86::Cpuid::StructExtFeatIdEbx0)**

Read-only. Reset: Fixed,209C\_01A9h.

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn00000007\_EBX\_x00

Bits	Description
31:30	Reserved.
29	<b>SHA.</b> Read-only. Reset: Fixed,1. 1=SHA Extensions available.
28:24	Reserved.
23	<b>CLFSHOPT.</b> Read-only. Reset: Fixed,1. Optimized Cache Line Flush.
22:21	Reserved.
20	<b>SMAP.</b> Read-only. Reset: Fixed,1. Secure Mode Access Prevention is supported.
19	<b>ADX.</b> Read-only. Reset: Fixed,1. ADCX and ADOX are present.
18	<b>RDSEED.</b> Read-only. Reset: Fixed,1. RDSEED is present.
17:9	Reserved.
8	<b>BMI2.</b> Read-only. Reset: Fixed,1. Bit manipulation group 2 instruction support.
7	<b>SMEP.</b> Read-only. Reset: Fixed,1. Supervisor Mode Execution protection.
6	Reserved.
5	<b>AVX2.</b> Read-only. Reset: Fixed,1. AVX extension support.
4	Reserved.

3	<b>BMI1.</b> Read-only. Reset: Fixed,1. Bit manipulation group 1 instruction support.
2:1	Reserved.
0	<b>FSGSBASE.</b> Read-only. Reset: Fixed,1. FS and GS base read write instruction support.

#### **CPUID\_Fn00000007\_ECX\_x00 [Structured Extended Feature Identifier] (Core::X86::Cpuid::StructExtFeatIdEc0)**

_lthree0_core[3:0]_thread[1:0]; CPUID_Fn00000007_ECX_x00	
Bits	Description
31:0	Reserved.

#### **CPUID\_Fn00000007\_EDX\_x00 [Structured Extended Feature Identifiers] (Core::X86::Cpuid::StructExtFeatIdEd0)**

Reset: 0000_0000h.	
_lthree0_core[3:0]_thread[1:0]; CPUID_Fn00000007_EDX_x00	
Bits	Description
31:0	Reserved.

#### **CPUID\_Fn0000000B\_EAX [Extended Topology Enumeration] (Core::X86::Cpuid::ExtTopEnumEax)**

Read-only. Reset: Fixed,0000_0000h.	
_lthree0_core[3:0]_thread[1:0]; CPUID_Fn0000000B_EAX	
Bits	Description
31:0	Reserved.

#### **CPUID\_Fn0000000B\_EBX [Extended Topology Enumeration] (Core::X86::Cpuid::ExtTopEnumEbx)**

Read-only. Reset: Fixed,0000_0000h.	
_lthree0_core[3:0]_thread[1:0]; CPUID_Fn0000000B_EBX	
Bits	Description
31:0	Reserved.

#### **CPUID\_Fn0000000B\_ECX [Extended Topology Enumeration] (Core::X86::Cpuid::ExtTopEnumEc0)**

Read-only. Reset: Fixed,0000_0000h.	
_lthree0_core[3:0]_thread[1:0]; CPUID_Fn0000000B_ECX	
Bits	Description
31:0	Reserved.

#### **CPUID\_Fn0000000B\_EDX [Extended Topology Enumeration] (Core::X86::Cpuid::ExtTopEnumEd0)**

Read-only.	
_lthree0_core[3:0]_thread[1:0]; CPUID_Fn0000000B_EDX	
Bits	Description
31:8	Reserved.
7:0	<b>ApicId:</b> APIC ID. Read-only. Reset: XXh. APIC ID.

#### **CPUID\_Fn0000000D\_EAX\_x00 [Processor Extended State Enumeration] (Core::X86::Cpuid::ProcExtStateEnumEax00)**

Read-only. Reset: Fixed,0000_0007h.			
_lthree0_core[3:0]_thread[1:0]; CPUID_Fn0000000D_EAX_x00			
Bits	Description		
31:0	XFeatureSupportedMask[31:0]. Read-only. Reset: Fixed,0000_0007h. 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.
[31:3]		Reserved.

#### CPUID\_Fn0000000D\_EBX\_x00 [Processor Extended State Enumeration] (Core::X86::Cpuid::ProcExtStateEnumEbx00)

Read-only, Volatile.

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn0000000D\_EBX\_x00

Bits	Description
31:0	<b>XFeatureEnabledSizeMax.</b> Read-only, Volatile. Reset: XXXX_XXXXh. <b>Description:</b> Size in bytes of an uncompact XSAVE/XRSTOR area for all features enabled in the XCR0 register. IF (XCR0[AVX] == 1) Return EBX = 0000_0340h // legacy header + X87/SSE + AVX size ELSIF (XCR0[SSE] == 1) Return EBX=0000_0240h // legacy header + X87/SSE size ELSIF (XCR0[X87] == 1) Return EBX = 0000_0240h END

#### CPUID\_Fn0000000D\_ECX\_x00 [Processor Extended State Enumeration] (Core::X86::Cpuid::ProcExtStateEnumEcX00)

Read-only. Reset: Fixed, 0000\_0400h.

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn0000000D\_ECX\_x00

Bits	Description
31:0	<b>XFeatureSupportedSizeMax.</b> Read-only. Reset: Fixed, 0000_0400h. Size of legacy header + X87/SSE + AVX.

#### CPUID\_Fn0000000D\_EDX\_x00 [Processor Extended State Enumeration] (Core::X86::Cpuid::ProcExtStateEnumEdx00)

Read-only. Reset: Fixed, 0000\_0000h.

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn0000000D\_EDX\_x00

Bits	Description
31:0	<b>XFeatureSupportedMask[63:32].</b> 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.

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn0000000D\_EAX\_x01

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

#### CPUID\_Fn0000000D\_EBX\_x01 [Processor Extended State Enumeration] (Core::X86::Cpuid::ProcExtStateEnumEbx01)

Read-only, Volatile.

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn0000000D\_EBX\_x01

Bits	Description
------	-------------

31:0	<b>XFeatureEnabledSizeMax.</b> Read-only, Volatile. Reset: XXXX_XXXXh. Value is 512 + ((XCR0[AVX]) ? 256 : 0).	
	<b>Valid Values:</b>	
	<b>Value</b>	<b>Description</b>
	0000_0 23Fh- 0000_0 000h	Reserved.
	0000_0 240h	Legacy header + FPU/SSE size; (XCR0[AVX] == 0)
	0000_0 33Fh- 0000_0 241h	Reserved.
	0000_0 340h	Legacy header + FPU/SSE + AVX size; (XCR0[AVX] == 1)
	FFFF_F FFFh- 0000_0 341h	Reserved.

#### **CPUID\_Fn0000000D\_ECX\_x01 [Processor Extended State Enumeration] (Core::X86::Cpuid::ProcExtStateEnumEcX01)**

Read-only. Reset: Fixed, 0000\_0000h.

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn0000000D\_ECX\_x01

Bits	Description
------	-------------

31:0	Reserved.
------	-----------

#### **CPUID\_Fn0000000D\_EDX\_x01 [Processor Extended State Enumeration] (Core::X86::Cpuid::ProcExtStateEnumEdX01)**

Read-only. Reset: Fixed, 0000\_0000h.

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn0000000D\_EDX\_x01

Bits	Description
------	-------------

31:0	Reserved.
------	-----------

#### **CPUID\_Fn0000000D\_EAX\_x02 [Processor Extended State Enumeration] (Core::X86::Cpuid::ProcExtStateEnumEax02)**

Read-only. Reset: Fixed, 0000\_0100h.

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn0000000D\_EAX\_x02

Bits	Description
------	-------------

31:0	<b>YmmSaveStateSize.</b> Read-only. Reset: Fixed, 0000_0100h. YMM save state byte size.
------	---

#### **CPUID\_Fn0000000D\_EBX\_x02 [Processor Extended State Enumeration] (Core::X86::Cpuid::ProcExtStateEnumEbx02)**

Read-only. Reset: Fixed, 0000\_0240h.

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn0000000D\_EBX\_x02

Bits	Description
------	-------------

31:0	<b>YmmSaveStateOffset.</b> Read-only. Reset: Fixed, 0000_0240h. YMM save state byte offset.
------	---

**CPUID\_Fn0000000D\_ECX\_x02 [Processor Extended State Enumeration]  
(Core::X86::Cpuid::ProcExtStateEnumEcX02)**

Read-only. Reset: Fixed,0000\_0000h.

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn0000000D\_ECX\_x02

Bits	Description
------	-------------

31:0	Reserved.
------	-----------

**CPUID\_Fn0000000D\_EDX\_x02 [Processor Extended State Enumeration]  
(Core::X86::Cpuid::ProcExtStateEnumEdX02)**

Read-only. Reset: Fixed,0000\_0000h.

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn0000000D\_EDX\_x02

Bits	Description
------	-------------

31:0	Reserved.
------	-----------

**CPUID\_Fn80000000\_EAX [Largest Extended Function Number] (Core::X86::Cpuid::LargExtFuncNum)**

Read-only. Reset: Fixed,8000\_001Fh.

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn80000000\_EAX

Bits	Description
------	-------------

31:0	<b>LFuncExt: largest extended function.</b> Read-only. Reset: Fixed,8000_001Fh. 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.

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn80000000\_EBX

Bits	Description
------	-------------

31:0	<b>Vendor.</b> 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.

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn80000000\_ECX

Bits	Description
------	-------------

31:0	<b>Vendor.</b> 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.

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn80000000\_EDX

Bits	Description
------	-------------

31:0	<b>Vendor.</b> 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.

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn80000001\_EAX

Bits	Description
------	-------------

31:28	Reserved.
-------	-----------

27:20	<b>ExtFamily: extended family.</b> Read-only. Reset: 08h. See Core::X86::Cpuid::FamModStep description of Family.
-------	---

19:16	<b>ExtModel: extended model.</b> Read-only. Reset: 1h. See Core::X86::Cpuid::FamModStep description of ExtModel.
-------	--

15:12	Reserved.
-------	-----------

11:8	<b>BaseFamily.</b> Read-only. Reset: Fh. See Core::X86::Cpuid::FamModStep description of Family.
7:4	<b>BaseModel.</b> Read-only. Reset: Xh. Model numbers vary with product.
3:0	<b>Stepping.</b> Read-only. Reset: 1h. Processor stepping (revision) for a specific model.

#### CPUID\_Fn80000001\_EBX [BrandId Identifier] (Core::X86::Cpuid::BrandId)

Read-only.

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn80000001\_EBX

Bits	Description										
31:28	<b>PkgType: package type.</b> Read-only. Reset: Xh. Specifies the package type. <b>ValidValues:</b>										
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>FP5</td></tr> <tr> <td>1h</td><td>Reserved.</td></tr> <tr> <td>2h</td><td>AM4</td></tr> <tr> <td>Fh-3h</td><td>Reserved.</td></tr> </table>	Value	Description	0h	FP5	1h	Reserved.	2h	AM4	Fh-3h	Reserved.
Value	Description										
0h	FP5										
1h	Reserved.										
2h	AM4										
Fh-3h	Reserved.										
27:0	Reserved.										

#### CPUID\_Fn80000001\_ECX [Feature Identifiers] (Core::X86::Cpuid::FeatureExtIdEcX)

Read-only.

These values can be over-written by Core::X86::Msrr::CPUID\_ExtFeatures.

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn80000001\_ECX

Bits	Description
31:30	Reserved.
29	<b>MwaitExtended.</b> Read-only. Reset: !Core::X86::Msrr::HWCR[MonMwaitDis]. 1=MWAITX and MONITORX capability is supported.
28	<b>PerfCtrExtLLC: Last Level Cache performance counter extensions.</b> Read-only. Reset: Fixed,1. 1=Indicates support for Core::X86::Msrr::ChL3PmcCfg and Core::X86::Msrr::ChL3Pmc L3 performance counter extensions. L3 performance counter extensions support. See 2.1.14.4 [L3 Cache Performance Monitor Counters] and 2.1.14 [Performance Monitor Counters].
27	<b>PerfTsc.</b> Read-only. Reset: Fixed,0. Performance time-stamp counter supported.
26	<b>DataBreakpointExtension.</b> Read-only. Reset: Fixed,1. 1=Indicates data breakpoint support for Core::X86::Msrr::DR0_ADDR_MASK, Core::X86::Msrr::DR1_ADDR_MASK, Core::X86::Msrr::DR2_ADDR_MASK and Core::X86::Msrr::DR3_ADDR_MASK.
25	Reserved.
24	<b>PerfCtrExtDF: data fabric performance counter extensions support.</b> Read-only. Reset: Fixed,1. 1=Indicates support for Core::X86::Msrr::DF_PERF_CTL and Core::X86::Msrr::DF_PERF_CTR.
23	<b>PerfCtrExtCore: core performance counter extensions support.</b> Read-only. Reset: Fixed,1. 1=Indicates support for Core::X86::Msrr::PERF_CTL and Core::X86::Msrr::PERF_CTR. See 2.1.14.3 [Core Performance Monitor Counters] and 2.1.14 [Performance Monitor Counters].
22	<b>TopologyExtensions: topology extensions support.</b> Read-only. Reset: Fixed,1. 1=Indicates support for Core::X86::Cpuid::CachePropEax0 and Core::X86::Cpuid::ExtApicId.
21:18	Reserved.
17	<b>TCE.</b> Read-only. Reset: Fixed,1. Translation cache extension.
16	<b>FMA4.</b> Read-only. Reset: Fixed,0. Four-operand FMA instruction support.
15	<b>LWP.</b> Read-only. Reset: Fixed,0. Lightweight profiling support.
14	Reserved.
13	<b>WDT.</b> Read-only. Reset: Fixed,1. Watchdog timer support.
12	<b>SKINIT.</b> Read-only. Reset: Fixed,1. SKINIT and STGI support.
11	<b>XOP.</b> Read-only. Reset: Fixed,0. Extended operation support.
10	<b>IBS.</b> Read-only. Reset: 0. Instruction Based Sampling.



9	<b>OSVW.</b> Read-only. Reset: Fixed,1. OS Visible Work-around support.
8	<b>ThreeDNowPrefetch.</b> Read-only. Reset: Fixed,1. Prefetch and PrefetchW instructions.
7	<b>MisAlignSse.</b> Read-only. Reset: Fixed,1. Misaligned SSE Mode.
6	<b>SSE4A.</b> Read-only. Reset: Fixed,1. EXTRQ, INSERTQ, MOVNTSS, and MOVNTSD instruction support.
5	<b>ABM: advanced bit manipulation.</b> Read-only. Reset: Fixed,1. LZCNT instruction support.
4	<b>AltMovCr8.</b> Read-only. Reset: Fixed,1. LOCK MOV CR0 means MOV CR8.
3	<b>ExtApicSpace.</b> Read-only. Reset: Fixed,1. Extended APIC register space.
2	<b>SVM: Secure Virtual Mode feature.</b> Read-only. Reset: Fixed,1. Indicates support for: VMRUN, VMLOAD, VMSAVE, CLGI, VMMCALL, and INVLPGA.
1	<b>CmpLegacy.</b> 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	<b>LahfSahf.</b> 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.

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn80000001\_EDX

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

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn80000002\_EAX

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

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn80000002\_EBX

Bits	Description
31:24	<b>ProcNameByte7.</b> Read-only. Reset: Core::X86::Msr::ProcNameString_n0[CpuNameString7]. Processor name, byte 7.
23:16	<b>ProcNameByte6.</b> Read-only. Reset: Core::X86::Msr::ProcNameString_n0[CpuNameString6]. Processor name, byte 6.
15:8	<b>ProcNameByte5.</b> Read-only. Reset: Core::X86::Msr::ProcNameString_n0[CpuNameString5]. Processor name, byte 5.
7:0	<b>ProcNameByte4.</b> 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.

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn80000002\_ECX

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

_lthree0_core[3:0]_thread[1:0]; CPUID_Fn80000002_EDX	
Bits	Description
31:24	<b>ProcNameByte15.</b> Read-only. Reset: Core::X86::Msr::ProcNameString_n1[CpuNameString7]. Processor name, byte 15.
23:16	<b>ProcNameByte14.</b> Read-only. Reset: Core::X86::Msr::ProcNameString_n1[CpuNameString6]. Processor name, byte 14.
15:8	<b>ProcNameByte13.</b> Read-only. Reset: Core::X86::Msr::ProcNameString_n1[CpuNameString5]. Processor name, byte 13.
7:0	<b>ProcNameByte12.</b> 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.

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn80000003\_EAX

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

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn80000003\_EBX

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

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn80000003\_ECX

Bits	Description
31:24	<b>ProcNameByte27.</b> Read-only. Reset: Core::X86::Msr::ProcNameString_n3[CpuNameString3]. Processor name, byte 27.
23:16	<b>ProcNameByte26.</b> Read-only. Reset: Core::X86::Msr::ProcNameString_n3[CpuNameString2]. Processor name, byte 26.
15:8	<b>ProcNameByte25.</b> Read-only. Reset: Core::X86::Msr::ProcNameString_n3[CpuNameString1]. Processor name, byte 25.

	byte 25.
7:0	<b>ProcNameByte24.</b> 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.

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn80000003\_EDX

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

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn80000004\_EAX

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

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn80000004\_EBX

Bits	Description
31:24	<b>ProcNameByte39.</b> Read-only. Reset: Core::X86::Msr::ProcNameString_n4[CpuNameString7]. Processor name, byte 39.
23:16	<b>ProcNameByte38.</b> Read-only. Reset: Core::X86::Msr::ProcNameString_n4[CpuNameString6]. Processor name, byte 38.
15:8	<b>ProcNameByte37.</b> Read-only. Reset: Core::X86::Msr::ProcNameString_n4[CpuNameString5]. Processor name, byte 37.
7:0	<b>ProcNameByte36.</b> 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.	
_lthree0_core[3:0]_thread[1:0]; CPUID_Fn80000004_ECX	
Bits	Description
31:24	<b>ProcNameByte43.</b> Read-only. Reset: Core::X86::Msr::ProcNameString_n5[CpuNameString3]. Processor name, byte 43.
23:16	<b>ProcNameByte42.</b> Read-only. Reset: Core::X86::Msr::ProcNameString_n5[CpuNameString2]. Processor name, byte 42.
15:8	<b>ProcNameByte41.</b> Read-only. Reset: Core::X86::Msr::ProcNameString_n5[CpuNameString1]. Processor name, byte 41.
7:0	<b>ProcNameByte40.</b> 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.	
_lthree0_core[3:0]_thread[1:0]; CPUID_Fn80000004_EDX	
Bits	Description
31:24	<b>ProcNameByte47.</b> Read-only. Reset: Core::X86::Msr::ProcNameString_n5[CpuNameString7]. Processor name, byte 47.
23:16	<b>ProcNameByte46.</b> Read-only. Reset: Core::X86::Msr::ProcNameString_n5[CpuNameString6]. Processor name, byte 46.
15:8	<b>ProcNameByte45.</b> Read-only. Reset: Core::X86::Msr::ProcNameString_n5[CpuNameString5]. Processor name, byte 45.
7:0	<b>ProcNameByte44.</b> 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.	
_lthree0_core[3:0]_thread[1:0]; CPUID_Fn80000005_EAX	
Bits	Description
31:24	<b>L1DTlb2and4MAssoc: data TLB associativity for 2-MB and 4-MB pages.</b> Read-only. Reset: Fixed,FFh. See Core::X86::Cpuid::L1DcId[L1DcAssoc].
23:16	<b>L1DTlb2and4MSize: data TLB number of entries for 2-MB and 4 MB-pages.</b> Read-only. Reset: Fixed,64. The value returned is for the number of entries available for the 2-MB page size; 4-MB pages require two 2-MB entries, so the number of entries available for the 4-MB page size is one-half the returned value.
15:8	<b>L1ITlb2and4MAssoc: instruction TLB associativity for 2-MB and 4 MB-pages.</b> Read-only. Reset: Fixed,FFh. See Core::X86::Cpuid::L1DcId[L1DcAssoc].
7:0	<b>L1ITlb2and4MSize: instruction TLB number of entries for 2-MB and 4-MB pages.</b> 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.	
_lthree0_core[3:0]_thread[1:0]; CPUID_Fn80000005_EBX	
Bits	Description
31:24	<b>L1DTlb4KAssoc.</b> Read-only. Reset: Fixed,FFh. Data TLB associativity for 4-KB pages. See Core::X86::Cpuid::L1DcId[L1DcAssoc].
23:16	<b>L1DTlb4KSize.</b> Read-only. Reset: Fixed,64. Data TLB number of entries for 4-KB pages.
15:8	<b>L1ITlb4KAssoc.</b> Read-only. Reset: Fixed,FFh. Instruction TLB associativity for 4-KB pages. See

	Core::X86::Cpuid::L1DcId[L1DcAssoc].
7:0	<b>L1ITlb4KSize.</b> 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.

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn80000005\_ECX

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

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn80000005\_EDX

Bits	Description																
31:24	<b>L1IcSize.</b> Read-only. Reset: Fixed,64. L1 instruction cache size KB.																
23:16	<b>L1IcAssoc.</b> Read-only. Reset: Fixed,4. L1 instruction cache associativity.																
	<b>ValidValues:</b>																
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>00h</td><td>Reserved.</td></tr> <tr> <td>01h</td><td>1 way (direct mapped)</td></tr> <tr> <td>02h</td><td>2 way</td></tr> <tr> <td>03h</td><td>3 way</td></tr> <tr> <td>04h</td><td>4 way</td></tr> <tr> <td>FEh-05h</td><td>&lt;Value&gt; way</td></tr> <tr> <td>FFh</td><td>Fully associative.</td></tr> </table>	Value	Description	00h	Reserved.	01h	1 way (direct mapped)	02h	2 way	03h	3 way	04h	4 way	FEh-05h	<Value> way	FFh	Fully associative.
Value	Description																
00h	Reserved.																
01h	1 way (direct mapped)																
02h	2 way																
03h	3 way																
04h	4 way																
FEh-05h	<Value> way																
FFh	Fully associative.																
15:8	<b>L1IcLinesPerTag.</b> Read-only. Reset: Fixed,01h. L1 instruction cache lines per tag.																
7:0	<b>L1IcLineSize.</b> 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.

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn80000006\_EAX

Bits	Description		
31:28	<b>L2DTlb2and4MAssoc:</b> L2 data TLB associativity for 2-MB and 4-MB pages. Read-only. Reset: Xh.		
	<b>ValidValues:</b>		
	<table> <tr> <th>Value</th><th>Description</th></tr> </table>	Value	Description
Value	Description		

	1h-0h	Reserved.
	2h	2 ways
	3h	3 ways
	Fh-4h	Reserved.
27:16	<b>L2DTlb2and4MSize: L2 data TLB number of entries for 2-MB and 4-MB pages.</b> Read-only. Reset: Fixed,1536. The value returned is for the number of entries available for the 2-MB page size; 4-MB pages require two 2-MB entries, so the number of entries available for the 4-MB page size is one-half the returned value.	
15:12	<b>L2ITlb2and4MAssoc: L2 instruction TLB associativity for 2-MB and 4-MB pages.</b> Read-only. Reset: Fixed,6.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	5h-0h	Reserved.
	6h	8 ways
	Fh-7h	Reserved.
11:0	<b>L2ITlb2and4MSize: L2 instruction TLB number of entries for 2-MB and 4-MB pages.</b> Read-only. Reset: Fixed,1024. The value returned is for the number of entries available for the 2-MB page size; 4-MB pages require two 2-MB entries, so the number of entries available for the 4-MB page size is one-half the returned value.	

#### **CPUID\_Fn80000006\_EBX [L2 TLB 4K Identifiers] (Core::X86::Cpuid::L2Tlb4K)**

Read-only.

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

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn80000006\_EBX

Bits	Description
31:28	<b>L2DTlb4KAssoc.</b> Read-only. Reset: Xh. L2 data TLB associativity for 4-KB pages.
	<b>ValidValues:</b>
	<b>Value</b> <b>Description</b>
	4h-0h Reserved.
	5h 6 ways
	6h 8 ways
	Fh-7h Reserved.
27:16	<b>L2DTlb4KSize.</b> Read-only. Reset: Fixed,1536. L2 data TLB number of entries for 4-KB pages.
15:12	<b>L2ITlb4KAssoc.</b> Read-only. Reset: Fixed,6. L2 instruction TLB associativity for 4-KB pages.
	<b>ValidValues:</b>
	<b>Value</b> <b>Description</b>
	5h-0h Reserved.
	6h 8 ways
	Fh-7h Reserved.
11:0	<b>L2ITlb4KSize.</b> Read-only. Reset: Fixed,1024. 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.

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn80000006\_ECX

Bits	Description
31:16	<b>L2Size.</b> Read-only. Reset: Fixed,0200h. L2 cache size in KB.
	<b>ValidValues:</b>
	<b>Value</b> <b>Description</b>
	00FFh-0000h Reserved.
	0100h 256-KB

	01FFh-0101h	Reserved.
	0200h	512-KB
	03FFh-0201h	Reserved.
	0400h	1-MB
	07FFh-0401h	Reserved.
	0800h	2-MB
	FFFFh-0801h	Reserved.
15:12	<b>L2Assoc.</b> Read-only. Reset: Fixed,6. L2 cache associativity.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	Disabled.
	1h	1 way (direct mapped)
	2h	2 ways
	3h	Reserved.
	4h	4 ways
	5h	Reserved.
	6h	8 ways
	7h	Reserved.
	8h	16 ways
	9h	Reserved.
	Ah	32 ways
	Bh	48 ways
	Ch	64 ways
	Dh	96 ways
	Eh	128 ways
	Fh	Fully associative.
11:8	<b>L2LinesPerTag.</b> Read-only. Reset: Fixed,1h. L2 cache lines per tag.	
7:0	<b>L2LineSize.</b> Read-only. Reset: Fixed,64. L2 cache line size in bytes.	

**CPUID\_Fn80000006\_EDX [L3 Cache Identifiers] (Core::X86::Cpuid::L3CacheId)**

Read-only.

This function provides third level cache characteristics shared by all cores of a processor.

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn80000006\_EDX

Bits	Description
31:18	<b>L3Size: L3 cache size.</b> Read-only. Reset: XXXXh. The L3 cache size in 512 KB units.
	<b>ValidValues:</b>
	<b>Value</b> <b>Description</b>
	0000h Disabled.
	3FFFh-0001h (<Value> *0.5) MB
17:16	Reserved.
15:12	<b>L3Assoc.</b> Read-only. Reset: Xh. L3 cache associativity.
	<b>ValidValues:</b>
	<b>Value</b> <b>Description</b>
	7h-0h Reserved.



	8h	16 ways
	9h	Reserved.
	Ah	32 ways
	Bh	48 ways
	Ch	64 ways
	Dh	96 ways
	Eh	128 ways
	Fh	Reserved.
11:8	<b>L3LinesPerTag</b> . Read-only. Reset: Fixed,1h. L3 cache lines per tag.	
7:0	<b>L3LineSize</b> . 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.

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn80000007\_EAX

Bits	Description
31:0	Reserved.

**CPUID\_Fn80000007\_EBX [RAS Capabilities] (Core::X86::Cpuid::RasCap)**

Read-only.

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn80000007\_EBX

Bits	Description
31:4	Reserved.
3	<b>ScalableMca</b> . Read-only. Reset: Fixed,1. 0=Scalable MCA is not supported. 1=Scalable MCA is supported. See 3.1.1.2 [Machine Check Architecture Extensions] and MCA_CONFIG[McaX] for the respective bank.
2	<b>HWA</b> . Read-only. Reset: Fixed,0. Hardware assert supported.
1	<b>SUCCOR: Software uncorrectable error containment and recovery capability</b> . 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[Poisson] exist.
0	<b>McaOverflowRecov: MCA overflow recovery support</b> . Read-only. Reset: Fixed,1. 0=MCA overflow conditions require software to shutdown the system. 1=MCA overflow conditions (MCA_STATUS[Overflow] == 1) are not fatal; software may safely ignore such conditions. See 3.1 [Machine Check Architecture].

**CPUID\_Fn80000007\_ECX [Advanced Power Management Information] (Core::X86::Cpuid::ApmInfoEcX)**

Read-only. Reset: Fixed,0000\_0000h.

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn80000007\_ECX

Bits	Description
31:0	<b>CpuPwrSampleTimeRatio</b> . 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.

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn80000007\_EDX

Bits	Description
31:15	Reserved.
14	<b>RAPL</b> . Read-only. Reset: Fixed,1. Running average power limit.
13	<b>ConnectedStandby</b> . Read-only. Reset: Fixed,1. Connected Standby.
12	<b>ProcPowerReporting</b> . Read-only. Reset: Fixed,0. Core power reporting interface supported.
11	<b>ProcFeedbackInterface: processor feedback interface</b> . Read-only. Reset: Fixed,0. 1=Indicates support for processor feedback interface; Core::X86::Cpuid::ProcFeedbackCap.

10	<b>EffFreqRO: read-only effective frequency interface.</b> Read-only. Reset: Fixed,1. Indicates presence of Core::X86::Msrr::MPerfReadOnly and Core::X86::Msrr::APerfReadOnly.
9	<b>CPB: core performance boost.</b> Read-only. Reset: X. 1=Indicates presence of Core::X86::Msrr::HWCR[CpbDis] and support for core performance boost.
8	<b>TscInvariant: TSC invariant.</b> Read-only. Reset: Fixed,1. The TSC rate is invariant.
7	<b>HwPstate: hardware P-state control.</b> Read-only. Reset: Fixed,1. Core::X86::Msrr::PStateCurLim, Core::X86::Msrr::PStateCtl and Core::X86::Msrr::PStateStat exist.
6	<b>OneHundredMHzSteps.</b> Read-only. Reset: Fixed,0. 100 MHz multiplier Control.
5	Reserved.
4	<b>TM.</b> Read-only. Reset: Fixed,1. Hardware thermal control (HTC).
3	<b>TTP.</b> Read-only. Reset: Fixed,1. THERMTRIP.
2	<b>VID: Voltage ID control.</b> Read-only. Reset: Fixed,0. Function replaced by HwPstate.
1	<b>FID: Frequency ID control.</b> Read-only. Reset: Fixed,0. Function replaced by HwPstate.
0	<b>TS.</b> Read-only. Reset: Fixed,1. Temperature sensor.

#### CPUID\_Fn80000008\_EAX [Long Mode Address Size Identifiers] (Core::X86::Cpuid::LongModeInfo)

Read-only. Reset: Fixed,0000\_3030h.

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

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn80000008\_EAX

Bits	Description						
31:24	Reserved.						
23:16	<b>GuestPhysAddrSize.</b> Read-only. Reset: Fixed,00h. Maximum guest physical byte address size in bits. <b>ValidValues:</b>						
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>00h</td><td>The maximum guest physical address size defined by PhysAddrSize.</td></tr> <tr> <td>FFh-01h</td><td>The maximum guest physical address size defined by GuestPhysAddrSize.</td></tr> </table>	Value	Description	00h	The maximum guest physical address size defined by PhysAddrSize.	FFh-01h	The maximum guest physical address size defined by GuestPhysAddrSize.
Value	Description						
00h	The maximum guest physical address size defined by PhysAddrSize.						
FFh-01h	The maximum guest physical address size defined by GuestPhysAddrSize.						
15:8	<b>LinAddrSize.</b> Read-only. Reset: Fixed,30h. Maximum linear byte address size in bits.						
7:0	<b>PhysAddrSize.</b> Read-only. Reset: Fixed,30h. Maximum physical byte address size in bits.						

#### CPUID\_Fn80000008\_EBX [Extended Feature Extensions ID EBX] (Core::X86::Cpuid::FeatureExtIdEbx)

Read-only. Reset: Fixed,0000\_0007h.

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn80000008\_EBX

Bits	Description
31:3	Reserved.
2	<b>RstrFpErrPtrs.</b> 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	<b>InstRetCntMsrr: instructions retired count support.</b> Read-only. Reset: Fixed,1. 1=Core::X86::Msrr::IRPerfCount supported.
0	<b>CLZERO: Clear Zero Instruction.</b> 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.

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn80000008\_ECX

Bits	Description
31:18	Reserved.
17:16	<b>PerfTscSize: performance time-stamp counter size.</b> Read-only. Reset: Fixed,0h.
15:12	<b>ApicIdSize: APIC ID size.</b> 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	<b>NC: number of threads - 1.</b> 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\_Fn8000000A\_EAX [SVM Revision and Feature Identification] (Core::X86::Cpuid::SvmRevFeatIdEax)

Read-only. Reset: Fixed,0000\_0001h. Enable: Core::X86::Cpuid::FeatureExtIdEcxEax[SVM].

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn8000000A\_EAX

Bits	Description
31:8	Reserved.
7:0	<b>SvmRev.</b> Read-only. Reset: Fixed,01h. SVM revision.

#### CPUID\_Fn8000000A\_EBX [SVM Revision and Feature Identification] (Core::X86::Cpuid::SvmRevFeatIdEbx)

Read-only, Volatile. Reset: 0000\_8000h. Enable: Core::X86::Cpuid::FeatureExtIdEcxEbx[SVM].

This provides SVM revision and feature information.

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn8000000A\_EBX

Bits	Description
31:0	<b>NASID: number of address space identifiers (ASID).</b> Read-only, Volatile. Reset: 0000_8000h.

#### CPUID\_Fn8000000A\_EDX [SVM Revision and Feature Identification] (Core::X86::Cpuid::SvmRevFeatIdEdx)

Read-only. Reset: Fixed,0001\_B4FFh. Enable: Core::X86::Cpuid::FeatureExtIdEcxEbx[SVM].

This provides SVM feature information.

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn8000000A\_EDX

Bits	Description
31:17	Reserved.
16	<b>vGIF.</b> Read-only. Reset: Fixed,1. Virtualized GIF.
15	<b>V_VMSAVE_VMLOAD.</b> Read-only. Reset: Fixed,1. Virtualized VMLOAD and VMSAVE.
14	Reserved.
13	<b>AVIC: AMD virtual interrupt controller.</b> Read-only. Reset: Fixed,1. 1=Support indicated for SVM mode virtualized interrupt controller; Indicates support for Core::X86::Msr::AvicDoorbell.
12	<b>PauseFilterThreshold.</b> Read-only. Reset: Fixed,1. PAUSE filter threshold.
11	Reserved.
10	<b>PauseFilter.</b> Read-only. Reset: Fixed,1. Pause intercept filter.
9:8	Reserved.
7	<b>DecodeAssists.</b> Read-only. Reset: Fixed,1. Decode assists.
6	<b>FlushByAsid.</b> Read-only. Reset: Fixed,1. Flush by ASID.
5	<b>VmcblClean.</b> Read-only. Reset: Fixed,1. VMCB clean bits.
4	<b>TscRateMsr: MSR based TSC rate control.</b> Read-only. Reset: Fixed,1. 1=Indicates support for TSC ratio Core::X86::Msr::TscRateMsr.
3	<b>NRIPS.</b> Read-only. Reset: Fixed,1. NRIP Save.
2	<b>SVML.</b> Read-only. Reset: Fixed,1. SVM lock.
1	<b>LbrVirt.</b> Read-only. Reset: Fixed,1. LBR virtualization.
0	<b>NP.</b> Read-only. Reset: Fixed,1. Nested Paging.

#### CPUID\_Fn80000019\_EAX [L1 TLB 1G Identifiers] (Core::X86::Cpuid::L1Tlb1G)

Read-only.

This function provides first level TLB characteristics for 1-GB pages.

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn80000019\_EAX

Bits	Description
31:28	<b>L1DTlb1GAssoc: L1 data TLB associativity for 1-GB pages.</b> Read-only. Reset: Fixed,Fh. See Core::X86::Cpuid::L2CacheId[L2Assoc].
27:16	<b>L1DTlb1GSize.</b> Read-only. Reset: Fixed,64. L1 data TLB number of entries for 1-GB pages.

15:12	<b>L1ITlb1GAssoc.</b> Read-only. Reset: Fixed,Fh. L1 instruction TLB associativity for 1-GB pages. See Core::X86::Cpuid::L2CacheId[L2Assoc].
11:0	<b>L1ITlb1GSize.</b> 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,0000\_0000h.

This provides 1-GB paging information. The associativity fields are defined by Core::X86::Cpuid::L2Tlb2M4M, Core::X86::Cpuid::L2Tlb4K, Core::X86::Cpuid::L2CacheId and Core::X86::Cpuid::L3CacheId.

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn80000019\_EBX

Bits	Description
31:28	<b>L2DTlb1GAssoc.</b> Read-only. Reset: Fixed,0h. L2 data TLB associativity for 1-GB pages. See Core::X86::Cpuid::L2CacheId[L2Assoc].
27:16	<b>L2DTlb1GSize.</b> Read-only. Reset: Fixed,000h. L2 data TLB number of entries for 1-GB pages.
15:12	<b>L2ITlb1GAssoc.</b> Read-only. Reset: Fixed,0h. L2 instruction TLB associativity for 1-GB pages. See Core::X86::Cpuid::L2CacheId[L2Assoc].
11:0	<b>L2ITlb1GSize.</b> 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\_0003h.

This function returns performance related information.

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn8000001A\_EAX

Bits	Description
31:3	Reserved.
2	<b>FP256.</b> Read-only. Reset: Fixed,0. 256-bit AVX instructions are executed with full-width internal operations and pipelines rather than decomposing them into internal 128-bit suboperations.
1	<b>MOVU.</b> 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 MOVLDP/MOVHPD.
0	<b>FP128.</b> Read-only. Reset: Fixed,1. 128-bit SSE (multimedia) instructions are executed with full-width internal operations and pipelines rather than decomposing them into internal 64-bit suboperations.

#### CPUID\_Fn8000001B\_EAX [Instruction Based Sampling Identifiers] (Core::X86::Cpuid::IbsIdEax)

Read-only. Reset: Fixed,0000\_03FFh.

This function returns IBS feature information.

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn8000001B\_EAX

Bits	Description
31:11	Reserved.
10	<b>IbsOpData4.</b> Read-only. Reset: Fixed,0. IBS op data 4 MSR supported.
9	<b>IbsFetchCtlExtd: IBS fetch control extended MSR supported.</b> Read-only. Reset: Fixed,1. Indicates support for Core::X86::Msr::IC_IBS_EXTD_CTL.
8	<b>OpBrnFuse: fused branch op indication supported.</b> Read-only. Reset: Fixed,1. Indicates support for Core::X86::Msr::IBS_OP_DATA[IbsOpBrnFuse].
7	<b>RipInvalidChk: invalid RIP indication supported.</b> Read-only. Reset: Fixed,1. Indicates support for Core::X86::Msr::IBS_OP_DATA[IbsRipInvalid].
6	<b>OpCntExt: IbsOpCurCnt and IbsOpMaxCnt extend by 7 bits.</b> Read-only. Reset: Fixed,1. Indicates support for Core::X86::Msr::IBS_OP_CTL[IbsOpCurCnt[26:20],IbsOpMaxCnt[26:20]].
5	<b>BrnTrgt.</b> Read-only. Reset: Fixed,1. Branch target address reporting supported.
4	<b>OpCnt.</b> Read-only. Reset: Fixed,1. Op counting mode supported.
3	<b>RdWrOpCnt.</b> Read-only. Reset: Fixed,1. Read/Write of op counter supported.
2	<b>OpSam.</b> Read-only. Reset: Fixed,1. IBS execution sampling supported.
1	<b>FetchSam.</b> Read-only. Reset: Fixed,1. IBS fetch sampling supported.

0	<b>IBSFFV.</b> Read-only. Reset: Fixed,1. IBS feature flags valid.
---	--

### CPUID\_Fn8000001D\_EAX\_x00 [Cache Properties (DC)] (Core::X86::Cpuid::CachePropEax0)

Enable: Core::X86::Cpuid::FeatureExtIdEcx[TopologyExtensions].

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

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn8000001D\_EAX\_x00

Bits	Description												
31:26	Reserved.												
25:14	<b>NumSharingCache: number of logical processors sharing cache.</b> Read-only. Reset: XXXh. The number of logical processors sharing this cache is NumSharingCache + 1.												
13:10	Reserved.												
9	<b>FullyAssociative: fully associative cache.</b> Read-only. Reset: Fixed,0. 1=Cache is fully associative.												
8	<b>SelfInitialization: cache is self-initializing.</b> Read-only. Reset: Fixed,1. 1=Cache is self initializing; cache does not need software initialization.												
7:5	<b>CacheLevel: cache level.</b> Read-only. Reset: Fixed,1h. Identifies the cache level. <b>ValidValues:</b>												
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>Reserved.</td></tr> <tr> <td>1h</td><td>Level 1</td></tr> <tr> <td>2h</td><td>Level 2</td></tr> <tr> <td>3h</td><td>Level 3</td></tr> <tr> <td>7h-4h</td><td>Reserved.</td></tr> </table>	Value	Description	0h	Reserved.	1h	Level 1	2h	Level 2	3h	Level 3	7h-4h	Reserved.
Value	Description												
0h	Reserved.												
1h	Level 1												
2h	Level 2												
3h	Level 3												
7h-4h	Reserved.												
4:0	<b>CacheType: cache type.</b> Read-only. Reset: Fixed,01h. Identifies the type of cache. <b>ValidValues:</b>												
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>00h</td><td>Null; no more caches.</td></tr> <tr> <td>01h</td><td>Data cache.</td></tr> <tr> <td>02h</td><td>Instruction cache.</td></tr> <tr> <td>03h</td><td>Unified cache.</td></tr> <tr> <td>1Fh-04h</td><td>Reserved.</td></tr> </table>	Value	Description	00h	Null; no more caches.	01h	Data cache.	02h	Instruction cache.	03h	Unified cache.	1Fh-04h	Reserved.
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. Enable: Core::X86::Cpuid::FeatureExtIdEcx[TopologyExtensions].

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

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn8000001D\_EAX\_x01

Bits	Description
31:26	Reserved.
25:14	<b>NumSharingCache: number of logical processors sharing cache.</b> Read-only. Reset: XXXh. See Core::X86::Cpuid::CachePropEax0[NumSharingCache].
13:10	Reserved.
9	<b>FullyAssociative: fully associative cache.</b> Read-only. Reset: Fixed,0. See Core::X86::Cpuid::CachePropEax0[FullyAssociative].
8	<b>SelfInitialization: cache is self-initializing.</b> Read-only. Reset: Fixed,1. See Core::X86::Cpuid::CachePropEax0[SelfInitialization].
7:5	<b>CacheLevel: cache level.</b> Read-only. Reset: Fixed,1h. Identifies the cache level. See Core::X86::Cpuid::CachePropEax0[CacheLevel].
4:0	<b>CacheType: cache type.</b> Read-only. Reset: Fixed,02h. See Core::X86::Cpuid::CachePropEax0[CacheType].

### CPUID\_Fn8000001D\_EAX\_x02 [Cache Properties (L2)] (Core::X86::Cpuid::CachePropEax2)

Read-only. Enable: Core::X86::Cpuid::FeatureExtIdEcx[TopologyExtensions].

Core::X86::Cpuid::CachePropEax2 reports topology information for the L2. See Core::X86::Cpuid::CachePropEax0. _lthree0_core[3:0]_thread[1:0]; CPUID_Fn8000001D_EAX_x02	
Bits	Description
31:26	Reserved.
25:14	<b>NumSharingCache: number of logical processors sharing cache.</b> Read-only. Reset: XXXh. Core::X86::Cpuid::CachePropEax0[NumSharingCache].
13:10	Reserved.
9	<b>FullyAssociative: fully associative cache.</b> Read-only. Reset: Fixed,0. Core::X86::Cpuid::CachePropEax0[FullyAssociative].
8	<b>SelfInitialization: cache is self-initializing.</b> Read-only. Reset: Fixed,1. Core::X86::Cpuid::CachePropEax0[SelfInitialization].
7:5	<b>CacheLevel: cache level.</b> Read-only. Reset: Fixed,2h. Identifies the cache level. Core::X86::Cpuid::CachePropEax0[CacheLevel].
4:0	<b>CacheType: cache type.</b> Read-only. Reset: Fixed,03h. Core::X86::Cpuid::CachePropEax0[CacheType].

#### CPUID\_Fn8000001D\_EAX\_x03 [Cache Properties (L3)] (Core::X86::Cpuid::CachePropEax3)

Read-only. Enable: Core::X86::Cpuid::FeatureExtIdEcX[TopologyExtensions].	
Core::X86::Cpuid::CachePropEax3 reports topology information for the L3. _lthree0_core[3:0]_thread[1:0]; CPUID_Fn8000001D_EAX_x03	
Bits	Description
31:26	Reserved.
25:14	<b>NumSharingCache: number of logical processors sharing cache.</b> Read-only. Reset: XXXh. The number of logical processors sharing this cache is NumSharingCache + 1.
13:10	Reserved.
9	<b>FullyAssociative: fully associative cache.</b> Read-only. Reset: Fixed,0. Core::X86::Cpuid::CachePropEax0[FullyAssociative].
8	<b>SelfInitialization: cache is self-initializing.</b> Read-only. Reset: Fixed,1. Core::X86::Cpuid::CachePropEax0[SelfInitialization].
7:5	<b>CacheLevel: cache level.</b> Read-only. Reset: Fixed,3h. Identifies the cache level. Core::X86::Cpuid::CachePropEax0[CacheLevel].
4:0	<b>CacheType: cache type.</b> Read-only. Reset: Fixed,03h. Core::X86::Cpuid::CachePropEax0[CacheType].

#### CPUID\_Fn8000001D\_EAX\_x04 [Cache Properties Null] (Core::X86::Cpuid::CachePropEax4)

Read-only. Reset: Fixed,0000_0000h. Enable: Core::X86::Cpuid::FeatureExtIdEcX[TopologyExtensions].	
Core::X86::Cpuid::CachePropEax4 reports done/null. See Core::X86::Cpuid::CachePropEax0. _lthree0_core[3:0]_thread[1:0]; CPUID_Fn8000001D_EAX_x04	
Bits	Description
31:5	Reserved.
4:0	<b>CacheType: cache type.</b> Read-only. Reset: Fixed,00h. Core::X86::Cpuid::CachePropEax0[CacheType].

#### CPUID\_Fn8000001D\_EBX\_x00 [Cache Properties (DC)] (Core::X86::Cpuid::CachePropEbx0)

Read-only. Reset: Fixed,01C0_003Fh. Enable: Core::X86::Cpuid::FeatureExtIdEcX[TopologyExtensions].	
Core::X86::Cpuid::CachePropEbx0 reports topology information for the DC. See Core::X86::Cpuid::CachePropEax0. _lthree0_core[3:0]_thread[1:0]; CPUID_Fn8000001D_EBX_x00	
Bits	Description
31:22	<b>CacheNumWays: cache number of ways.</b> Read-only. Reset: Fixed,007h. Cache number of ways is CacheNumWays + 1.
21:12	<b>CachePhysPartitions: cache physical line partitions.</b> Read-only. Reset: Fixed,000h. Cache partitions is CachePhysPartitions + 1.
11:0	<b>CacheLineSize: cache line size in bytes.</b> 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,00C0\_003Fh. Enable: Core::X86::Cpuid::FeatureExtIdEcX[TopologyExtensions].

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

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn8000001D\_EBX\_x01

Bits	Description
31:22	<b>CacheNumWays: cache number of ways.</b> Read-only. Reset: Fixed,003h. Core::X86::Cpuid::CachePropEbx0[CacheNumWays].
21:12	<b>CachePhysPartitions: cache physical line partitions.</b> Read-only. Reset: Fixed,000h. Core::X86::Cpuid::CachePropEbx0[CachePhysPartitions].
11:0	<b>CacheLineSize: cache line size in bytes.</b> Read-only. Reset: Fixed,03Fh. Core::X86::Cpuid::CachePropEbx0[CacheLineSize].

**CPUID\_Fn8000001D\_EBX\_x02 [Cache Properties (L2)] (Core::X86::Cpuid::CachePropEbx2)**

Read-only. Reset: Fixed,01C0\_003Fh. Enable: Core::X86::Cpuid::FeatureExtIdEcX[TopologyExtensions].

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

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn8000001D\_EBX\_x02

Bits	Description
31:22	<b>CacheNumWays: cache number of ways.</b> Read-only. Reset: Fixed,007h. See Core::X86::Cpuid::CachePropEbx0[CacheNumWays].
21:12	<b>CachePhysPartitions: cache physical line partitions.</b> Read-only. Reset: Fixed,000h. See Core::X86::Cpuid::CachePropEbx0[CachePhysPartitions].
11:0	<b>CacheLineSize: cache line size in bytes.</b> Read-only. Reset: Fixed,03Fh. See Core::X86::Cpuid::CachePropEbx0[CacheLineSize].

**CPUID\_Fn8000001D\_EBX\_x03 [Cache Properties (L3)] (Core::X86::Cpuid::CachePropEbx3)**

Read-only. Reset: Fixed,03C0\_003Fh. Enable: Core::X86::Cpuid::FeatureExtIdEcX[TopologyExtensions].

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

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn8000001D\_EBX\_x03

Bits	Description
31:22	<b>CacheNumWays: cache number of ways.</b> Read-only. Reset: Fixed,00Fh. See Core::X86::Cpuid::CachePropEbx0[CacheNumWays].
21:12	<b>CachePhysPartitions: cache physical line partitions.</b> Read-only. Reset: Fixed,000h. See Core::X86::Cpuid::CachePropEbx0[CachePhysPartitions].
11:0	<b>CacheLineSize: cache line size in bytes.</b> Read-only. Reset: Fixed,03Fh. See Core::X86::Cpuid::CachePropEbx0[CacheLineSize].

**CPUID\_Fn8000001D\_EBX\_x04 [Cache Properties Null] (Core::X86::Cpuid::CachePropEbx4)**

Read-only. Reset: Fixed,0000\_0000h. Enable: Core::X86::Cpuid::FeatureExtIdEcX[TopologyExtensions].

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

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn8000001D\_EBX\_x04

Bits	Description
31:0	Reserved.

**CPUID\_Fn8000001D\_ECX\_x00 [Cache Properties (DC)] (Core::X86::Cpuid::CachePropEcX0)**

Read-only. Reset: Fixed,0000\_003Fh. Enable: Core::X86::Cpuid::FeatureExtIdEcX[TopologyExtensions].

Core::X86::Cpuid::CachePropEcX0 reports topology information for the DC. See Core::X86::Cpuid::CachePropEax0.

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn8000001D\_ECX\_x00

Bits	Description
31:0	<b>CacheNumSets: cache number of sets.</b> 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_00FFh. Enable: Core::X86::Cpuid::FeatureExtIdEcX[TopologyExtensions].	
Core::X86::Cpuid::CachePropEcX1 reports topology information for the IC. See Core::X86::Cpuid::CachePropEax0.	
_lthree0_core[3:0]_thread[1:0]; CPUID_Fn8000001D_ECX_x01	
Bits	Description
31:0	<b>CacheNumSets: cache number of sets.</b> Read-only. Reset: Fixed,0000_00FFh. See Core::X86::Cpuid::CachePropEcX0[CacheNumSets].

CPUID_Fn8000001D_ECX_x02 [Cache Properties (L2)] (Core::X86::Cpuid::CachePropEcX2)	
Read-only. Reset: Fixed,0000_03FFh. Enable: Core::X86::Cpuid::FeatureExtIdEcX[TopologyExtensions].	
Core::X86::Cpuid::CachePropEcX2 reports topology information for the L2. See Core::X86::Cpuid::CachePropEax0.	
_lthree0_core[3:0]_thread[1:0]; CPUID_Fn8000001D_ECX_x02	
Bits	Description
31:0	<b>CacheNumSets: cache number of sets.</b> Read-only. Reset: Fixed,0000_03FFh. See Core::X86::Cpuid::CachePropEcX0[CacheNumSets].

CPUID_Fn8000001D_ECX_x03 [Cache Properties (L3)] (Core::X86::Cpuid::CachePropEcX3)	
Read-only. Enable: Core::X86::Cpuid::FeatureExtIdEcX[TopologyExtensions].	
Core::X86::Cpuid::CachePropEcX3 reports topology information for the L3.	
_lthree0_core[3:0]_thread[1:0]; CPUID_Fn8000001D_ECX_x03	
Bits	Description
31:0	<b>CacheNumSets: cache number of sets.</b> Read-only. Reset: 0000_XXXXh. See Core::X86::Cpuid::CachePropEcX0[CacheNumSets].
<b>Valid Values:</b>	
Value	Description
0000_0 FFeh- 0000_0 000h	Reserved.
0000_0 FFFh	4096 L3 Cache Sets.
FFFF_F FFFh- 0000_1 000h	Reserved.

CPUID_Fn8000001D_ECX_x04 [Cache Properties Null] (Core::X86::Cpuid::CachePropEcX4)	
Read-only. Reset: Fixed,0000_0000h. Enable: Core::X86::Cpuid::FeatureExtIdEcX[TopologyExtensions].	
Core::X86::Cpuid::CachePropEax3 reports done/null. See Core::X86::Cpuid::CachePropEax0.	
_lthree0_core[3:0]_thread[1:0]; CPUID_Fn8000001D_ECX_x04	
Bits	Description
31:0	<b>CacheNumSets.</b> Read-only. Reset: Fixed,0000_0000h. Cache number of sets.

CPUID_Fn8000001D_EDX_x00 [Cache Properties (DC)] (Core::X86::Cpuid::CachePropEdx0)	
Read-only. Reset: Fixed,0000_0000h. Enable: Core::X86::Cpuid::FeatureExtIdEcX[TopologyExtensions].	
Core::X86::Cpuid::CachePropEdx0 reports topology information for the DC. See Core::X86::Cpuid::CachePropEax0.	
_lthree0_core[3:0]_thread[1:0]; CPUID_Fn8000001D_EDX_x00	
Bits	Description
31:2	Reserved.
1	<b>CacheInclusive: cache inclusive.</b> Read-only. Reset: Fixed,0. 0=Cache is not inclusive of lower cache levels. 1=Cache is inclusive of lower cache levels.
0	<b>WBINVD: Write-Back Invalidate/Invalidate.</b> Read-only. Reset: Fixed,0. 0=WBINVD/INVD invalidates all lower level caches of non-originating cores sharing this cache. 1=WBINVD/INVD not ensured to invalidate all

	lower level caches of non-originating cores sharing this cache.
--	---

**CPUID\_Fn8000001D\_EDX\_x01 [Cache Properties (IC)] (Core::X86::Cpuid::CachePropEdx1)**

Read-only. Reset: Fixed,0000\_0000h. Enable: Core::X86::Cpuid::FeatureExtIdEcx[TopologyExtensions].

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

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn8000001D\_EDX\_x01

Bits	Description
31:2	Reserved.
1	<b>CacheInclusive: cache inclusive.</b> Read-only. Reset: Fixed,0. See Core::X86::Cpuid::CachePropEdx0[CacheInclusive].
0	<b>WBINVD: Write-Back Invalidate/Invalidate.</b> Read-only. Reset: Fixed,0. 0=WBINVD/INVD invalidates all lower level caches of non-originating cores sharing this cache. 1=WBINVD/INVD may not invalidate all lower level caches of non-originating cores sharing this cache. See Core::X86::Cpuid::CachePropEdx0[WBINVD].

**CPUID\_Fn8000001D\_EDX\_x02 [Cache Properties (L2)] (Core::X86::Cpuid::CachePropEdx2)**

Read-only. Reset: Fixed,0000\_0002h. Enable: Core::X86::Cpuid::FeatureExtIdEcx[TopologyExtensions].

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

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn8000001D\_EDX\_x02

Bits	Description
31:2	Reserved.
1	<b>CacheInclusive: cache inclusive.</b> Read-only. Reset: Fixed,1. See Core::X86::Cpuid::CachePropEdx0[CacheInclusive].
0	<b>WBINVD: Write-Back Invalidate/Invalidate.</b> Read-only. Reset: Fixed,0. 0=WBINVD/INVD invalidates all lower level caches of non-originating cores sharing this cache. 1=WBINVD/INVD may not invalidate all lower level caches of non-originating cores sharing this cache.

**CPUID\_Fn8000001D\_EDX\_x03 [Cache Properties (L3)] (Core::X86::Cpuid::CachePropEdx3)**

Read-only. Reset: Fixed,0000\_0001h. Enable: Core::X86::Cpuid::FeatureExtIdEcx[TopologyExtensions].

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

Core::X86::Cpuid::CachePropEax0.

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn8000001D\_EDX\_x03

Bits	Description
31:2	Reserved.
1	<b>CacheInclusive: cache inclusive.</b> Read-only. Reset: Fixed,0. See Core::X86::Cpuid::CachePropEdx0[CacheInclusive].
0	<b>WBINVD: Write-Back Invalidate/Invalidate.</b> Read-only. Reset: Fixed,1. 0=WBINVD/INVD invalidates all lower level caches of non-originating cores sharing this cache. 1=WBINVD/INVD may not invalidate all lower level caches of non-originating cores sharing this cache.

**CPUID\_Fn8000001D\_EDX\_x04 [Cache Properties Null] (Core::X86::Cpuid::CachePropEdx4)**

Read-only. Reset: Fixed,0000\_0000h. Enable: Core::X86::Cpuid::FeatureExtIdEcx[TopologyExtensions].

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

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn8000001D\_EDX\_x04

Bits	Description
31:0	Reserved.

**CPUID\_Fn8000001E\_EAX [Extended APIC ID] (Core::X86::Cpuid::ExtApicId)**

Read-only. Enable: (Core::X86::Cpuid::FeatureExtIdEcx[TopologyExtensions] &&

Core::X86::Msr::APIC\_BAR[ApicEn]).

If Core::X86::Cpuid::FeatureExtIdEcx[TopologyExtensions] == 0 then CPUID Fn8000001E\_E[D,C,B,A]X are Reserved. If (Core::X86::Msr::APIC\_BAR[ApicEn] == 0) then Core::X86::Cpuid::ExtApicId[ExtendedApicId] is Reserved.

_lthree0_core[3:0]_thread[1:0]; CPUID_Fn8000001E_EAX	
Bits	Description
31:0	<b>ExtendedApicId: extended APIC ID.</b> Read-only. See 2.1.11.2.1.3 [ApicId Enumeration Requirements]. Reset: Core::X86::Msr::APIC_BAR[ApicEn] ? Fixed,{00_0000h , Core::X86::Apic::ApicId[ApicId]} : Fixed,0000_0000h.

#### CPUID\_Fn8000001E\_EBX [Core Identifiers] (Core::X86::Cpuid::CoreId)

Read-only. Enable: Core::X86::Cpuid::FeatureExtIdEcX[TopologyExtensions].	
See Core::X86::Cpuid::ExtApicId.	
_lthree0_core[3:0]_thread[1:0]; CPUID_Fn8000001E_EBX	
Bits	Description
31:16	Reserved.
15:8	<b>ThreadsPerCore: threads per core.</b> Read-only. Reset: XXh. The number of threads per core is ThreadsPerCore + 1.
7:0	<b>CoreId: core ID.</b> Read-only. Reset: Fixed,XXh.

#### CPUID\_Fn8000001E\_ECX [Node Identifiers] (Core::X86::Cpuid::NodeId)

Read-only. Enable: Core::X86::Cpuid::FeatureExtIdEcX[TopologyExtensions].							
_lthree0_core[3:0]_thread[1:0]; CPUID_Fn8000001E_ECX							
Bits	Description						
31:11	Reserved.						
10:8	<b>NodesPerProcessor: Node per processor.</b> Read-only. Reset: XXXb. <b>ValidValues:</b>						
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>1 node per processor.</td></tr> <tr> <td>7h-1h</td><td>Reserved.</td></tr> </table>	Value	Description	0h	1 node per processor.	7h-1h	Reserved.
Value	Description						
0h	1 node per processor.						
7h-1h	Reserved.						
7:0	<b>NodeId: Node ID.</b> Read-only. Reset: Fixed,XXh.						

#### CPUID\_Fn8000001F\_EAX [AMD Secure Encryption EAX] (Core::X86::Cpuid::SecureEncryptionEax)

Read-only. Reset: Fixed,0000_000Fh.	
_lthree0_core[3:0]_thread[1:0]; CPUID_Fn8000001F_EAX	
Bits	Description
31:4	Reserved.
3	<b>SevEs.</b> Read-only. Reset: Fixed,1. Secure Encrypted ES.
2	<b>VmPgFlush: VM Page Flush MSR is supported.</b> Read-only. Reset: Fixed,1. See Core::X86::Msr::VMPAGE_FLUSH.
1	<b>SEV.</b> Read-only. Reset: Fixed,1. Secure Encrypted Virtualization supported.
0	<b>SME.</b> Read-only. Reset: Fixed,1. Secure Memory Encryption supported.

#### CPUID\_Fn8000001F\_EBX [AMD Secure Encryption EBX] (Core::X86::Cpuid::SecureEncryptionEbx)

Read-only.									
_lthree0_core[3:0]_thread[1:0]; CPUID_Fn8000001F_EBX									
Bits	Description								
31:12	Reserved.								
11:6	<b>MemEncryptPhysAddWidth.</b> Read-only. Reset: 000XXXb. Reduction of physical address space in bits when memory encryption is enabled (0 indicates no reduction). <b>ValidValues:</b>								
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>00h</td><td>Physical Address width is not reduced.</td></tr> <tr> <td>01h</td><td>Physical Address width is reduced by one.</td></tr> <tr> <td>02h</td><td>Physical Address width is reduced by two.</td></tr> </table>	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.
Value	Description								
00h	Physical Address width is not reduced.								
01h	Physical Address width is reduced by one.								
02h	Physical Address width is reduced by two.								

	03h	Physical Address width is reduced by three.
	04h	Physical Address width is reduced by four.
	05h	Physical Address width is reduced by five.
	3Fh-06h	Reserved.
5:0	<b>CBit.</b> Read-only. Reset: 2Fh. Page table bit number used to enable memory encryption.	

#### **CPUID\_Fn8000001F\_ECX [AMD Secure Encryption ECX] (Core::X86::Cpuid::SecureEncryptionEcX)**

Read-only. Reset: 0000\_000Fh.

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn8000001F\_ECX

Bits	Description
31:0	<b>NumEncryptedGuests.</b> Read-only. Reset: 0000_000Fh. 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.

\_lthree0\_core[3:0]\_thread[1:0]; CPUID\_Fn8000001F\_EDX

Bits	Description
31:0	<b>MinimumSEVASID: Minimum SEV enabled, SEV-ES disabled ASID.</b> Read-only. Reset: 0000_000Xh. Indicates the minimum ASID value that must be used for an SEV-enabled, SEV-ES-disabled guest.

## 2.1.13 MSR Registers

### 2.1.13.1 MSRs - MSR0000\_xxxx

See 1.4.3 [Register Mnemonics] for a description of the register naming convention. MSRs are accessed through x86 WRMSR and RDMSR instructions.

#### **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.

\_lthree0\_core[3:0]\_thread[1:0]; MSR0000\_0010

Bits	Description
63:0	<b>TSC: time stamp counter.</b> 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)**

\_lthree0\_core[3:0]\_thread[1:0]; MSR0000\_001B

Bits	Description
63:48	Reserved.
47:12	<b>ApicBar[47:12]: APIC base address register.</b> Read-write. Reset: 0_000F_EE00h. Specifies the base address, physical address [47:12], for the APICXX register set in xAPIC mode. See 2.1.11.2.1.2 [APIC Register Space].
11	<b>ApicEn: APIC enable.</b> Read-write. Reset: 0. 0=Disable Local Apic. 1=Local APIC is enabled in xAPIC mode. See 2.1.11.2.1.2 [APIC Register Space].
10:9	Reserved.
8	<b>BSC: boot strap core.</b> 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.

\_lthree0\_core[3:0]\_thread[1:0]; MSR0000\_002A

Bits	Description
63:18	Reserved.
17:16	<b>ClusterID</b> . Read,Error-on-write. Reset: 0h. The field does not affect hardware.
15:0	Reserved.

**MSR0000\_008B [Patch Level] (Core::X86::Msr::PATCH\_LEVEL)**

Read,Error-on-write, Volatile. Reset: 0000\_0000\_0000\_0000h.

\_lthree0\_core[3:0]; MSR0000\_008B

Bits	Description
63:32	Reserved.
31:0	<b>PatchLevel</b> . 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.

\_lthree0\_core[3:0]\_thread[1:0]; MSR0000\_00E7

Bits	Description
63:0	<b>MPERF: maximum core clocks counter</b> . 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], 2.1.4 [Effective Frequency].

**MSR0000\_00E8 [Actual Performance Frequency Clock Count] (Core::X86::Msr::APERF)**

Read-write, Volatile. Reset: 0000\_0000\_0000\_0000h.

\_lthree0\_core[3:0]\_thread[1:0]; MSR0000\_00E8

Bits	Description
63:0	<b>APERF: actual core clocks counter</b> . 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.

\_lthree0\_core[3:0]; MSR0000\_00FE

Bits	Description
63:11	Reserved.
10	<b>MtrrCapWc: write-combining memory type</b> . Read,Error-on-write. Reset: 1. 1=The write combining memory type is supported.
9	Reserved.
8	<b>MtrrCapFix: fixed range register</b> . Read,Error-on-write. Reset: 1. 1=Fixed MTRRs are supported.
7:0	<b>MtrrCapVCnt: variable range registers count</b> . Read,Error-on-write. Reset: 08h. Specifies the number of variable MTRRs supported.

**MSR0000\_0174 [SYSENTER CS] (Core::X86::Msr::SYSENTER\_CS)**

Read-write. Reset: 0000\_0000\_0000\_0000h.

\_lthree0\_core[3:0]\_thread[1:0]; MSR0000\_0174

Bits	Description
63:16	Reserved.
15:0	<b>SysEnterCS: SYSENTER target CS</b> . 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.

\_lthree0\_core[3:0]\_thread[1:0]; MSR0000\_0175

Bits	Description
63:32	Reserved.
31:0	<b>SysEnterESP: SYSENTER target SP.</b> 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.

\_lthree0\_core[3:0]\_thread[1:0]; MSR0000\_0176

Bits	Description
63:32	Reserved.
31:0	<b>SysEnterEIP: SYSENTER target IP.</b> Read-write. Reset: 0000_0000h. Holds the called procedure instruction pointer.

**MSR0000\_0179 [Global Machine Check Capabilities] (Core::X86::Msr::MCG\_CAP)**

\_lthree0\_core[3:0]\_thread[1:0]; MSR0000\_0179

Bits	Description
63:9	Reserved.
8	<b>McgCtlP: MCG_CTL register present.</b> 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	<b>Count.</b> Read-only, Error-on-write, Volatile. Reset: XXh. Indicates the number of error reporting banks visible to each 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].

\_lthree0\_core[3:0]\_thread[1:0]; MSR0000\_017A

Bits	Description
63:3	Reserved.
2	<b>MCIP: machine check in progress.</b> Read-write, Volatile. Reset: 0. 1=A machine check is in progress. Machine check progress.
1	<b>EIPV: error instruction pointer valid.</b> 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	<b>RIPV: restart instruction pointer valid.</b> 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)**

Read-write. Reset: 0000\_0000\_0000\_0000h.

This register controls enablement of the individual error reporting banks; see 3.1 [Machine Check Architecture]. 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.

\_lthree0\_core[3:0]\_thread[1:0]; MSR0000\_017B

Bits	Description
63:16	Reserved.
15:0	<b>MCnEn.</b> Read-write. Reset: 0000h. 1=The MC0 machine check register bank is enabled.

**MSR0000\_01D9 [Debug Control] (Core::X86::Msr::DBG\_CTL\_MSR)**

\_lthree0\_core[3:0]\_thread[1:0]; MSR0000\_01D9



Bits	Description
63:6	Reserved.
5:2	<b>PB: performance monitor pin control.</b> Read-write. Reset: 0h. This field does not control any hardware.
1	<b>BTF.</b> Read-write. Reset: 0. 1=Enable branch single step.
0	<b>LBR.</b> 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.

\_lthree0\_core[3:0]\_thread[1:0]; MSR0000\_01DB

Bits	Description
63:0	<b>LastBranchFromIP.</b> Read,Error-on-write,Volatile. Reset: 0000_0000_0000_0000h. Loaded with the segment offset of the branch instruction.

**MSR0000\_01DC [Last Branch To IP] (Core::X86::Msr::BR\_TO)**

Read,Error-on-write,Volatile. Reset: 0000\_0000\_0000\_0000h.

\_lthree0\_core[3:0]\_thread[1:0]; MSR0000\_01DC

Bits	Description
63:0	<b>LastBranchToIP.</b> Read,Error-on-write,Volatile. Reset: 0000_0000_0000_0000h. Holds the target RIP of the last branch that occurred before an exception or interrupt.

**MSR0000\_01DD [Last Exception From IP] (Core::X86::Msr::LastExcpFromIp)**

Read,Error-on-write,Volatile. Reset: 0000\_0000\_0000\_0000h.

\_lthree0\_core[3:0]\_thread[1:0]; MSR0000\_01DD

Bits	Description
63:0	<b>LastIntFromIP.</b> Read,Error-on-write,Volatile. Reset: 0000_0000_0000_0000h. Holds the source RIP of the last branch that occurred before the exception or interrupt.

**MSR0000\_01DE [Last Exception To IP] (Core::X86::Msr::LastExcpToIp)**

Read,Error-on-write,Volatile. Reset: 0000\_0000\_0000\_0000h.

\_lthree0\_core[3:0]\_thread[1:0]; MSR0000\_01DE

Bits	Description
63:0	<b>LastIntToIP.</b> Read,Error-on-write,Volatile. Reset: 0000_0000_0000_0000h. Holds the target RIP of the last branch that occurred before the exception or interrupt.

**MSR0000\_020[0...E] [Variable-Size MTRRs Base] (Core::X86::Msr::MtrrVarBase)**

Each MTRR (Core::X86::Msr::MtrrVarBase, Core::X86::Msr::MtrrFix\_64K through Core::X86::Msr::MtrrFix\_4K\_7, or Core::X86::Msr::MTRRdefType) specifies a physical address range and a corresponding memory type (MemType) associated with that range. Setting the memory type to an unsupported value results in a #GP.

The variable-size MTRRs come in pairs of base and mask registers (MSR0000\_0200 and MSR0000\_0201 are the first pair, etc.). Variables MTRRs are enabled through Core::X86::Msr::MTRRdefType[MtrrDefTypeEn]. A core access--with address CPUAddr--is determined to be within the address range of a variable-size MTRR if the following equation is true:

$$\text{CPUAddr}[47:12] \& \text{PhyMask}[47:12] == \text{PhyBase}[47:12] \& \text{PhyMask}[47:12].$$

For example, if the variable MTRR spans 256 KB and starts at the 1-MB address the PhyBase would be set to 0\_0010\_0000h and the PhyMask to F\_FFFC\_0000h (with zeros filling in for bits[11:0]). This results in a range from 0\_0010\_0000h to 0\_0013\_FFFFh.

\_lthree0\_core[3:0]\_n0; MSR0000\_0200

\_lthree0\_core[3:0]\_n1; MSR0000\_0202

\_lthree0\_core[3:0]\_n2; MSR0000\_0204

\_lthree0\_core[3:0]\_n3; MSR0000\_0206

\_lthree0\_core[3:0]\_n4; MSR0000\_0208

\_lthree0\_core[3:0]\_n5; MSR0000\_020A



_lthree0_core[3:0]_n6; MSR0000_020C	
_lthree0_core[3:0]_n7; MSR0000_020E	
Bits	Description
63:48	Reserved.
47:12	<b>PhyBase: base address.</b> Read-write. Reset: X_XXXX_XXXXh.
11:3	Reserved.
2:0	<b>MemType: memory type.</b> Read-write. Reset: XXXb. Address range from 00000h to 0FFFFh.
<b>ValidValues:</b>	
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::Msrr::MtrrVarMask)

_lthree0_core[3:0]_n0; MSR0000_0201	
_lthree0_core[3:0]_n1; MSR0000_0203	
_lthree0_core[3:0]_n2; MSR0000_0205	
_lthree0_core[3:0]_n3; MSR0000_0207	
_lthree0_core[3:0]_n4; MSR0000_0209	
_lthree0_core[3:0]_n5; MSR0000_020B	
_lthree0_core[3:0]_n6; MSR0000_020D	
_lthree0_core[3:0]_n7; MSR0000_020F	
Bits	Description
63:48	Reserved.
47:12	<b>PhyMask: address mask.</b> Read-write. Reset: X_XXXX_XXXXh.
11	<b>Valid: valid.</b> Read-write. Reset: X. 1=The variable-size MTRR pair is enabled.
10:0	Reserved.

#### MSR0000\_0250 [Fixed-Size MTRRs] (Core::X86::Msrr::MtrrFix\_64K)

See Core::X86::Msrr::MtrrVarBase for general MTRR information. Fixed MTRRs are enabled through Core::X86::Msrr::MTRRdefType[MtrrDefTypeFixEn,MtrrDefTypeEn]. For addresses below 1-MB, 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.

_lthree0_core[3:0]_nSIZE64K; MSR0000_0250	
Bits	Description
63:61	Reserved.
60	<b>RdDram_64K_70000: Read DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.
59	<b>WrDram_64K_70000: Write DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.
58:56	<b>MemType_64K_70000: memory type.</b> Read-write. Reset: XXXb.
<b>ValidValues:</b>	
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	<b>RdDram_64K_60000: Read DRAM.</b> 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	<b>WrDram_64K_60000: Write DRAM.</b> 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	<b>MemType_64K_60000: memory type.</b> Read-write. Reset: XXXb.		
	<b>ValidValues:</b>		
	<b>Value</b>	<b>Description</b>	
	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	<b>RdDram_64K_50000: Read DRAM.</b> 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.		
43	<b>WrDram_64K_50000: Write DRAM.</b> 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.		
42:40	<b>MemType_64K_50000: memory type.</b> Read-write. Reset: XXXb.		
	<b>ValidValues:</b>		
	<b>Value</b>	<b>Description</b>	
	0h	UC or uncacheable.	
	1h	WC or write combining.	
	3h-2h	Reserved.	
	4h	WT or write through.	
	5h	WP or write protect.	
	6h	WB or write back.	
7h	Reserved.		
39:37	Reserved.		
36	<b>RdDram_64K_40000: Read DRAM.</b> 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.		

35	<b>WrDram_64K_40000: Write DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
34:32	<b>MemType_64K_40000: memory type.</b> Read-write. Reset: XXXb. <b>ValidValues:</b> <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>	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.
Value	Description																
0h	UC or uncacheable.																
1h	WC or write combining.																
3h-2h	Reserved.																
4h	WT or write through.																
5h	WP or write protect.																
6h	WB or write back.																
7h	Reserved.																
31:29	Reserved.																
28	<b>RdDram_64K_30000: Read DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
27	<b>WrDram_64K_30000: Write DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
26:24	<b>MemType_64K_30000: memory type.</b> Read-write. Reset: XXXb. <b>ValidValues:</b> <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>	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.
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	<b>RdDram_64K_20000: Read DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
19	<b>WrDram_64K_20000: Write DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
18:16	<b>MemType_64K_20000: memory type.</b> Read-write. Reset: XXXb. <b>ValidValues:</b> <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>UC or uncacheable.</td></tr> <tr> <td>1h</td><td>WC or write combining.</td></tr> <tr> <td>3h-2h</td><td>Reserved.</td></tr> <tr> <td>4h</td><td>WT or write through.</td></tr> </table>	Value	Description	0h	UC or uncacheable.	1h	WC or write combining.	3h-2h	Reserved.	4h	WT or write through.						
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	<b>RdDram_64K_10000: Read DRAM.</b> 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	<b>WrDram_64K_10000: Write DRAM.</b> 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	<b>MemType_64K_10000: memory type.</b> Read-write. Reset: XXXb.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	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.	
4	<b>RdDram_64K_00000: Read DRAM.</b> 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.	
	AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.	
3	<b>WrDram_64K_00000: Write DRAM.</b> 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.	
	AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.	
2:0	<b>MemType_64K_00000: memory type.</b> Read-write. Reset: XXXb. Address range from 00000h to 0FFFFh.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	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\_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 1-MB, 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.

\_lthree0\_core[3:0]\_nSIZE16K0; MSR0000\_0258

Bits	Description																
63:61	Reserved.																
60	<b>RdDram_16K_9C000: Read DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
59	<b>WrDram_16K_9C000: Write DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
58:56	<b>MemType_16K_9C000: memory type.</b> Read-write. Reset: XXXb. <b>ValidValues:</b> <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>	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.
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	<b>RdDram_16K_98000: Read DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
51	<b>WrDram_16K_98000: Write DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
50:48	<b>MemType_16K_98000: memory type.</b> Read-write. Reset: XXXb. <b>ValidValues:</b> <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>	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.
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	<b>RdDram_16K_94000: Read DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
43	<b>WrDram_16K_94000: Write DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
42:40	<b>MemType_16K_94000: memory type.</b> Read-write. Reset: XXXb.																

	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	UC or uncacheable.
	1h	WC or write combining.
	3h-2h	Reserved.
	4h	WT or write through.
	5h	WP or write protect.
	6h	WB or write back.
	7h	Reserved.
39:37	Reserved.	
36	<b>RdDram_16K_90000: Read DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.	
35	<b>WrDram_16K_90000: Write DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.	
34:32	<b>MemType_16K_90000: memory type.</b> Read-write. Reset: XXXb.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	UC or uncacheable.
	1h	WC or write combining.
	3h-2h	Reserved.
	4h	WT or write through.
	5h	WP or write protect.
	6h	WB or write back.
	7h	Reserved.
31:29	Reserved.	
28	<b>RdDram_16K_8C000: Read DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.	
27	<b>WrDram_16K_8C000: Write DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.	
26:24	<b>MemType_16K_8C000: memory type.</b> Read-write. Reset: XXXb.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	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	<b>RdDram_16K_88000: Read DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
19	<b>WrDram_16K_88000: Write DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
18:16	<b>MemType_16K_88000: memory type.</b> Read-write. Reset: XXXb.																
	<b>ValidValues:</b>																
	<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>	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.
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	<b>RdDram_16K_84000: Read DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
11	<b>WrDram_16K_84000: Write DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
10:8	<b>MemType_16K_84000: memory type.</b> Read-write. Reset: XXXb.																
	<b>ValidValues:</b>																
	<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>	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.
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.																
4	<b>RdDram_16K_80000: Read DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramEn,MtrrFixDramModEn] masks Reads of the stored value.																
	AccessType: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
3	<b>WrDram_16K_80000: Write DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramEn,MtrrFixDramModEn] masks Reads of the stored value.																
	AccessType: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
2:0	<b>MemType_16K_80000: memory type.</b> Read-write. Reset: XXXb. Address range from 80000h to 83FFFh.																
	<b>ValidValues:</b>																



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\_0259 [Fixed-Size MTRRs] (Core::X86::Msrr::MtrrFix\_16K\_1)

See Core::X86::Msrr::MtrrVarBase for general MTRR information. Fixed MTRRs are enabled through Core::X86::Msrr::MTRRdefType[MtrrDefTypeFixEn,MtrrDefTypeEn]. For addresses below 1-MB, 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.

\_lthree0\_core[3:0]\_nSIZE16K1; MSR0000\_0259

Bits	Description																
63:61	Reserved.																
60	<b>RdDram_16K_BC000: Read DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
59	<b>WrDram_16K_BC000: Write DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
58:56	<b>MemType_16K_BC000: memory type.</b> Read-write. Reset: XXXb. <b>ValidValues:</b> <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>	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.
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	<b>RdDram_16K_B8000: Read DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
51	<b>WrDram_16K_B8000: Write DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
50:48	<b>MemType_16K_B8000: memory type.</b> Read-write. Reset: XXXb. <b>ValidValues:</b> <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>UC or uncacheable.</td></tr> <tr> <td>1h</td><td>WC or write combining.</td></tr> <tr> <td>3h-2h</td><td>Reserved.</td></tr> </table>	Value	Description	0h	UC or uncacheable.	1h	WC or write combining.	3h-2h	Reserved.								
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	<b>RdDram_16K_B4000: Read DRAM.</b> 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.	
43	<b>WrDram_16K_B4000: Write DRAM.</b> 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.	
42:40	<b>MemType_16K_B4000: memory type.</b> Read-write. Reset: XXXb.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	UC or uncacheable.
	1h	WC or write combining.
	3h-2h	Reserved.
	4h	WT or write through.
	5h	WP or write protect.
	6h	WB or write back.
	7h	Reserved.
39:37	Reserved.	
36	<b>RdDram_16K_B0000: Read DRAM.</b> 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.	
35	<b>WrDram_16K_B0000: Write DRAM.</b> 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.	
34:32	<b>MemType_16K_B0000: memory type.</b> Read-write. Reset: XXXb.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	UC or uncacheable.
	1h	WC or write combining.
	3h-2h	Reserved.
	4h	WT or write through.
	5h	WP or write protect.
	6h	WB or write back.
	7h	Reserved.
31:29	Reserved.	
28	<b>RdDram_16K_AC000: Read DRAM.</b> 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.	
27	<b>WrDram_16K_AC000: Write DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
26:24	<b>MemType_16K_AC000: memory type.</b> Read-write. Reset: XXXb.																
	<b>ValidValues:</b>																
	<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>	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.
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	<b>RdDram_16K_A8000: Read DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
19	<b>WrDram_16K_A8000: Write DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
18:16	<b>MemType_16K_A8000: memory type.</b> Read-write. Reset: XXXb.																
	<b>ValidValues:</b>																
	<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>	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.
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	<b>RdDram_16K_A4000: Read DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
11	<b>WrDram_16K_A4000: Write DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
10:8	<b>MemType_16K_A4000: memory type.</b> Read-write. Reset: XXXb.																
	<b>ValidValues:</b>																
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>UC or uncacheable.</td></tr> <tr> <td>1h</td><td>WC or write combining.</td></tr> <tr> <td>3h-2h</td><td>Reserved.</td></tr> <tr> <td>4h</td><td>WT or write through.</td></tr> <tr> <td>5h</td><td>WP or write protect.</td></tr> <tr> <td>6h</td><td>WB or write back.</td></tr> </table>	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.		
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.																	
4	<b>RdDram_16K_A0000: Read DRAM.</b> 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. AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																	
3	<b>WrDram_16K_A0000: Write DRAM.</b> 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. AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																	
2:0	<b>MemType_16K_A0000: memory type.</b> Read-write. Reset: XXXb. Address range from A0000h to A3FFFh. <b>ValidValues:</b> <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>		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.
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\_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 1-MB, 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.

\_lthree0\_core[3:0]\_nSIZE4K0; MSR0000\_0268

Bits	Description																
63:61	Reserved.																
60	<b>RdDram_4K_C7000: Read DRAM.</b> 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	<b>WrDram_4K_C7000: Write DRAM.</b> 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	<b>MemType_4K_C7000: memory type.</b> Read-write. Reset: XXXb. <b>ValidValues:</b> <table border="1"> <thead> <tr> <th>Value</th><th>Description</th></tr> </thead> <tbody> <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> </tbody> </table>	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.
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	<b>RdDram_4K_C6000: Read DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
51	<b>WrDram_4K_C6000: Write DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
50:48	<b>MemType_4K_C6000: memory type.</b> Read-write. Reset: XXXb.																
<b>ValidValues:</b>																	
	<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>	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.
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	<b>RdDram_4K_C5000: Read DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
43	<b>WrDram_4K_C5000: Write DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
42:40	<b>MemType_4K_C5000: memory type.</b> Read-write. Reset: XXXb.																
<b>ValidValues:</b>																	
	<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>	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.
Value	Description																
0h	UC or uncacheable.																
1h	WC or write combining.																
3h-2h	Reserved.																
4h	WT or write through.																
5h	WP or write protect.																
6h	WB or write back.																
7h	Reserved.																
39:37	Reserved.																
36	<b>RdDram_4K_C4000: Read DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
35	<b>WrDram_4K_C4000: Write DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
34:32	<b>MemType_4K_C4000: memory type.</b> Read-write. Reset: XXXb.																
<b>ValidValues:</b>																	
	<table> <tr> <th>Value</th><th>Description</th></tr> </table>	Value	Description														
Value	Description																

	0h	UC or uncacheable.
	1h	WC or write combining.
	3h-2h	Reserved.
	4h	WT or write through.
	5h	WP or write protect.
	6h	WB or write back.
	7h	Reserved.
31:29	Reserved.	
28	<b>RdDram_4K_C3000: Read DRAM.</b> 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.	
27	<b>WrDram_4K_C3000: Write DRAM.</b> 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.	
26:24	<b>MemType_4K_C3000: memory type.</b> Read-write. Reset: XXXb. <b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	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	<b>RdDram_4K_C2000: Read DRAM.</b> 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	<b>WrDram_4K_C2000: Write DRAM.</b> 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	<b>MemType_4K_C2000: memory type.</b> Read-write. Reset: XXXb. <b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	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	<b>RdDram_4K_C1000: Read DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
11	<b>WrDram_4K_C1000: Write DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
10:8	<b>MemType_4K_C1000: memory type.</b> Read-write. Reset: XXXb. <b>ValidValues:</b>																
	<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>	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.
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.																
4	<b>RdDram_4K_C0000: Read DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramEn,MtrrFixDramModEn] masks Reads of the stored value. AccessType: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
3	<b>WrDram_4K_C0000: Write DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramEn,MtrrFixDramModEn] masks Reads of the stored value. AccessType: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
2:0	<b>MemType_4K_C0000: memory type.</b> Read-write. Reset: XXXb. Address range from C0000h to C0FFFh. <b>ValidValues:</b>																
	<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>	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.
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\_0269 [Fixed-Size MTRRs] (Core::X86::Msrr::MtrrFix\_4K\_1)

See Core::X86::Msrr::MtrrVarBase for general MTRR information. Fixed MTRRs are enabled through Core::X86::Msrr::MTRRdefType[MtrrDefTypeFixEn,MtrrDefTypeEn]. For addresses below 1-MB, 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.

\_lthree0\_core[3:0]\_nSIZE4K1; MSR0000\_0269

Bits	Description
63:61	Reserved.
60	<b>RdDram_4K_CF000: Read DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.
59	<b>WrDram_4K_CF000: Write DRAM.</b> 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	<b>MemType_4K_CF000: memory type.</b> Read-write. Reset: XXXb.																
	<b>ValidValues:</b>																
	<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>	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.
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	<b>RdDram_4K_CE000: Read DRAM.</b> 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	<b>WrDram_4K_CE000: Write DRAM.</b> 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	<b>MemType_4K_CE000: memory type.</b> Read-write. Reset: XXXb.																
	<b>ValidValues:</b>																
	<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>	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.
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	<b>RdDram_4K_CD000: Read DRAM.</b> 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.																
43	<b>WrDram_4K_CD000: Write DRAM.</b> 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.																
42:40	<b>MemType_4K_CD000: memory type.</b> Read-write. Reset: XXXb.																
	<b>ValidValues:</b>																
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>UC or uncacheable.</td></tr> <tr> <td>1h</td><td>WC or write combining.</td></tr> <tr> <td>3h-2h</td><td>Reserved.</td></tr> <tr> <td>4h</td><td>WT or write through.</td></tr> <tr> <td>5h</td><td>WP or write protect.</td></tr> </table>	Value	Description	0h	UC or uncacheable.	1h	WC or write combining.	3h-2h	Reserved.	4h	WT or write through.	5h	WP or write protect.				
Value	Description																
0h	UC or uncacheable.																
1h	WC or write combining.																
3h-2h	Reserved.																
4h	WT or write through.																
5h	WP or write protect.																

	6h	WB or write back.
	7h	Reserved.
39:37	Reserved.	
36	<b>RdDram_4K_CC000: Read DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.	
35	<b>WrDram_4K_CC000: Write DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.	
34:32	<b>MemType_4K_CC000: memory type.</b> Read-write. Reset: XXXb.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	UC or uncacheable.
	1h	WC or write combining.
	3h-2h	Reserved.
	4h	WT or write through.
	5h	WP or write protect.
	6h	WB or write back.
	7h	Reserved.
31:29	Reserved.	
28	<b>RdDram_4K_CB000: Read DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.	
27	<b>WrDram_4K_CB000: Write DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.	
26:24	<b>MemType_4K_CB000: memory type.</b> Read-write. Reset: XXXb.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	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	<b>RdDram_4K_CA000: Read DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.	
19	<b>WrDram_4K_CA000: Write DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.	

18:16	<b>MemType_4K_CA000: memory type.</b> Read-write. Reset: XXXb. <b>ValidValues:</b> <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>	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.
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	<b>RdDram_4K_C9000: Read DRAM.</b> 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	<b>WrDram_4K_C9000: Write DRAM.</b> 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	<b>MemType_4K_C9000: memory type.</b> Read-write. Reset: XXXb. <b>ValidValues:</b> <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>	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.
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.																
4	<b>RdDram_4K_C8000: Read DRAM.</b> 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. AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
3	<b>WrDram_4K_C8000: Write DRAM.</b> 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. AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
2:0	<b>MemType_4K_C8000: memory type.</b> Read-write. Reset: XXXb. Address range from C8000 to C8FFF. <b>ValidValues:</b> <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>UC or uncacheable.</td></tr> <tr> <td>1h</td><td>WC or write combining.</td></tr> <tr> <td>3h-2h</td><td>Reserved.</td></tr> <tr> <td>4h</td><td>WT or write through.</td></tr> <tr> <td>5h</td><td>WP or write protect.</td></tr> <tr> <td>6h</td><td>WB or write back.</td></tr> </table>	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.		
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_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 1-MB, 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.		
_lthree0_core[3:0]_nSIZE4K2; MSR0000_026A		
Bits	Description	
63:61	Reserved.	
60	<b>RdDram_4K_D7000: Read DRAM.</b> 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	<b>WrDram_4K_D7000: Write DRAM.</b> 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	<b>MemType_4K_D7000: memory type.</b> Read-write. Reset: XXXb.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	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	<b>RdDram_4K_D6000: Read DRAM.</b> 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	<b>WrDram_4K_D6000: Write DRAM.</b> 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	<b>MemType_4K_D6000: memory type.</b> Read-write. Reset: XXXb.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	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	<b>RdDram_4K_D5000: Read DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
43	<b>WrDram_4K_D5000: Write DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
42:40	<b>MemType_4K_D5000: memory type.</b> Read-write. Reset: XXXb. <b>ValidValues:</b>																
	<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>	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.
Value	Description																
0h	UC or uncacheable.																
1h	WC or write combining.																
3h-2h	Reserved.																
4h	WT or write through.																
5h	WP or write protect.																
6h	WB or write back.																
7h	Reserved.																
39:37	Reserved.																
36	<b>RdDram_4K_D4000: Read DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
35	<b>WrDram_4K_D4000: Write DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
34:32	<b>MemType_4K_D4000: memory type.</b> Read-write. Reset: XXXb. <b>ValidValues:</b>																
	<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>	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.
Value	Description																
0h	UC or uncacheable.																
1h	WC or write combining.																
3h-2h	Reserved.																
4h	WT or write through.																
5h	WP or write protect.																
6h	WB or write back.																
7h	Reserved.																
31:29	Reserved.																
28	<b>RdDram_4K_D3000: Read DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
27	<b>WrDram_4K_D3000: Write DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
26:24	<b>MemType_4K_D3000: memory type.</b> Read-write. Reset: XXXb. <b>ValidValues:</b>																
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>UC or uncacheable.</td></tr> <tr> <td>1h</td><td>WC or write combining.</td></tr> </table>	Value	Description	0h	UC or uncacheable.	1h	WC or write combining.										
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	<b>RdDram_4K_D2000: Read DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.	
19	<b>WrDram_4K_D2000: Write DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.	
18:16	<b>MemType_4K_D2000: memory type.</b> Read-write. Reset: XXXb.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	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	<b>RdDram_4K_D1000: Read DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.	
11	<b>WrDram_4K_D1000: Write DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.	
10:8	<b>MemType_4K_D1000: memory type.</b> Read-write. Reset: XXXb.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	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.	
4	<b>RdDram_4K_D0000: Read DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramEn,MtrrFixDramModEn] masks Reads of the stored value.	
	AccessType: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.	



3	<p><b>WrDram_4K_D0000: Write DRAM.</b> 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.</p> <p>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</p>																
2:0	<p><b>MemType_4K_D0000: memory type.</b> Read-write. Reset: XXXb. Address range from D0000h to D0FFFh.</p> <p><b>ValidValues:</b></p> <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>	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.
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\_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 1-MB, 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.

\_lthree0\_core[3:0]\_nSIZE4K3; MSR0000\_026B

Bits	Description																
63:61	Reserved.																
60	<p><b>RdDram_4K_DF000: Read DRAM.</b> 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</p> <p>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</p>																
59	<p><b>WrDram_4K_DF000: Write DRAM.</b> 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</p> <p>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</p>																
58:56	<p><b>MemType_4K_DF000: memory type.</b> Read-write. Reset: XXXb.</p> <p><b>ValidValues:</b></p> <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>	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.
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	<p><b>RdDram_4K_DE000: Read DRAM.</b> 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM.</p> <p>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.</p>																
51	<p><b>WrDram_4K_DE000: Write DRAM.</b> 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM.</p> <p>AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset:</p>																



	Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
50:48	<b>MemType_4K_DE000: memory type.</b> Read-write. Reset: XXXb.																
	<b>ValidValues:</b>																
	<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>	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.
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	<b>RdDram_4K_DD000: Read DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
43	<b>WrDram_4K_DD000: Write DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
42:40	<b>MemType_4K_DD000: memory type.</b> Read-write. Reset: XXXb.																
	<b>ValidValues:</b>																
	<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>	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.
Value	Description																
0h	UC or uncacheable.																
1h	WC or write combining.																
3h-2h	Reserved.																
4h	WT or write through.																
5h	WP or write protect.																
6h	WB or write back.																
7h	Reserved.																
39:37	Reserved.																
36	<b>RdDram_4K_DC000: Read DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
35	<b>WrDram_4K_DC000: Write DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
34:32	<b>MemType_4K_DC000: memory type.</b> Read-write. Reset: XXXb.																
	<b>ValidValues:</b>																
	<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>	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.
Value	Description																
0h	UC or uncacheable.																
1h	WC or write combining.																
3h-2h	Reserved.																
4h	WT or write through.																
5h	WP or write protect.																
6h	WB or write back.																
7h	Reserved.																

31:29	Reserved.																
28	<b>RdDram_4K_DB000: Read DRAM.</b> 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.																
27	<b>WrDram_4K_DB000: Write DRAM.</b> 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.																
26:24	<b>MemType_4K_DB000: memory type.</b> Read-write. Reset: XXXb. <b>ValidValues:</b> <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>	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.
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	<b>RdDram_4K_DA000: Read DRAM.</b> 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	<b>WrDram_4K_DA000: Write DRAM.</b> 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	<b>MemType_4K_DA000: memory type.</b> Read-write. Reset: XXXb. <b>ValidValues:</b> <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>	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.
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	<b>RdDram_4K_D9000: Read DRAM.</b> 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	<b>WrDram_4K_D9000: Write DRAM.</b> 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	<b>MemType_4K_D9000: memory type.</b> Read-write. Reset: XXXb. <b>ValidValues:</b>																

	<b>Value</b>	<b>Description</b>																
	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.																	
4	<b>RdDram_4K_D8000: Read DRAM.</b> 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. AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																	
3	<b>WrDram_4K_D8000: Write DRAM.</b> 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. AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																	
2:0	<b>MemType_4K_D8000: memory type.</b> Read-write. Reset: XXXb. Address range from D8000h to D8FFFh. <b>ValidValues:</b> <table><tr><td><b>Value</b></td><td><b>Description</b></td></tr><tr><td>0h</td><td>UC or uncacheable.</td></tr><tr><td>1h</td><td>WC or write combining.</td></tr><tr><td>3h-2h</td><td>Reserved.</td></tr><tr><td>4h</td><td>WT or write through.</td></tr><tr><td>5h</td><td>WP or write protect.</td></tr><tr><td>6h</td><td>WB or write back.</td></tr><tr><td>7h</td><td>Reserved.</td></tr></table>		<b>Value</b>	<b>Description</b>	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.
<b>Value</b>	<b>Description</b>																	
0h	UC or uncacheable.																	
1h	WC or write combining.																	
3h-2h	Reserved.																	
4h	WT or write through.																	
5h	WP or write protect.																	
6h	WB or write back.																	
7h	Reserved.																	

#### MSR0000\_026C [Fixed-Size MTRRs] (Core::X86::Msr::MtrrFix\_4K\_4)

See Core::X86::Msr::MtrrVarBase for general MTRR information. Fixed MTRRs are enabled through Core::X86::Msr::MTRRdefType[MtrrDefTypeFixEn,MtrrDefTypeEn]. For addresses below 1-MB, 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.

\_lthree0\_core[3:0]\_nSIZE4K4; MSR0000\_026C

Bits	Description				
63:61	Reserved.				
60	<b>RdDram_4K_E7000: Read DRAM.</b> 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	<b>WrDram_4K_E7000: Write DRAM.</b> 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	<b>MemType_4K_E7000: memory type.</b> Read-write. Reset: XXXb. <b>ValidValues:</b> <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>UC or uncacheable.</td></tr> </table>	Value	Description	0h	UC or uncacheable.
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	<b>RdDram_4K_E6000: Read DRAM.</b> 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	<b>WrDram_4K_E6000: Write DRAM.</b> 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	<b>MemType_4K_E6000: memory type.</b> Read-write. Reset: XXXb.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	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	<b>RdDram_4K_E5000: Read DRAM.</b> 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.	
43	<b>WrDram_4K_E5000: Write DRAM.</b> 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.	
42:40	<b>MemType_4K_E5000: memory type.</b> Read-write. Reset: XXXb.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	UC or uncacheable.
	1h	WC or write combining.
	3h-2h	Reserved.
	4h	WT or write through.
	5h	WP or write protect.
	6h	WB or write back.
7h	Reserved.	
39:37	Reserved.	
36	<b>RdDram_4K_E4000: Read DRAM.</b> 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.	

35	<b>WrDram_4K_E4000: Write DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
34:32	<b>MemType_4K_E4000: memory type.</b> Read-write. Reset: XXXb. <b>ValidValues:</b> <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>	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.
Value	Description																
0h	UC or uncacheable.																
1h	WC or write combining.																
3h-2h	Reserved.																
4h	WT or write through.																
5h	WP or write protect.																
6h	WB or write back.																
7h	Reserved.																
31:29	Reserved.																
28	<b>RdDram_4K_E3000: Read DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
27	<b>WrDram_4K_E3000: Write DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
26:24	<b>MemType_4K_E3000: memory type.</b> Read-write. Reset: XXXb. <b>ValidValues:</b> <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>	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.
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	<b>RdDram_4K_E2000: Read DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
19	<b>WrDram_4K_E2000: Write DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
18:16	<b>MemType_4K_E2000: memory type.</b> Read-write. Reset: XXXb. <b>ValidValues:</b> <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>UC or uncacheable.</td></tr> <tr> <td>1h</td><td>WC or write combining.</td></tr> <tr> <td>3h-2h</td><td>Reserved.</td></tr> <tr> <td>4h</td><td>WT or write through.</td></tr> </table>	Value	Description	0h	UC or uncacheable.	1h	WC or write combining.	3h-2h	Reserved.	4h	WT or write through.						
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	<b>RdDram_4K_E1000: Read DRAM.</b> 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	<b>WrDram_4K_E1000: Write DRAM.</b> 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	<b>MemType_4K_E1000: memory type.</b> Read-write. Reset: XXXb.		
	<b>ValidValues:</b>		
	<b>Value</b>	<b>Description</b>	
	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.		
4	<b>RdDram_4K_E0000: Read DRAM.</b> 0=Read accesses to the range are marked as MMIO. 1=Read accesses to the range are marked as destined for DRAM. Address range from E0000h to E0FFFh.		
	Core::X86::Msr::SYS_CFG[MtrrFixDramEn,MtrrFixDramModEn] masks Reads of the stored value.		
	AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.		
3	<b>WrDram_4K_E0000: Write DRAM.</b> 0=Write accesses to the range are marked as MMIO. 1=Write accesses to the range are marked as destined for DRAM. Address range from E0000h to E0FFFh.		
	Core::X86::Msr::SYS_CFG[MtrrFixDramEn,MtrrFixDramModEn] masks Reads of the stored value.		
	AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.		
2:0	<b>MemType_4K_E0000: memory type.</b> Read-write. Reset: XXXb. Address range from E0000h to E0FFFh.		
	<b>ValidValues:</b>		
	<b>Value</b>	<b>Description</b>	
	0h	UC or uncacheable.	
	1h	WC or write combining.	
	3h-2h	Reserved.	
	4h	WT or write through.	
	5h	WP or write protect.	
	6h	WB or write back.	
7h	Reserved.		

#### MSR0000\_026D [Fixed-Size MTRRs] (Core::X86::Msr::MtrrFix\_4K\_5)

See Core::X86::Msr::MtrrVarBase for general MTRR information. Fixed MTRRs are enabled through Core::X86::Msr::MTRRdefType[MtrrDefTypeFixEn,MtrrDefTypeEn]. For addresses below 1-MB, 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.

\_lthree0\_core[3:0]\_nSIZE4K5; MSR0000\_026D



Bits	Description																
63:61	Reserved.																
60	<b>RdDram_4K_EF000: Read DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
59	<b>WrDram_4K_EF000: Write DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
58:56	<b>MemType_4K_EF000: memory type.</b> Read-write. Reset: XXXb. <b>ValidValues:</b> <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>	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.
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	<b>RdDram_4K_EE000: Read DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
51	<b>WrDram_4K_EE000: Write DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
50:48	<b>MemType_4K_EE000: memory type.</b> Read-write. Reset: XXXb. <b>ValidValues:</b> <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>	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.
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	<b>RdDram_4K_ED000: Read DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
43	<b>WrDram_4K_ED000: Write DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
42:40	<b>MemType_4K_ED000: memory type.</b> Read-write. Reset: XXXb.																



	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	UC or uncacheable.
	1h	WC or write combining.
	3h-2h	Reserved.
	4h	WT or write through.
	5h	WP or write protect.
	6h	WB or write back.
	7h	Reserved.
39:37	Reserved.	
36	<b>RdDram_4K_EC000: Read DRAM.</b> 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.	
35	<b>WrDram_4K_EC000: Write DRAM.</b> 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.	
34:32	<b>MemType_4K_EC000: memory type.</b> Read-write. Reset: XXXb.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	UC or uncacheable.
	1h	WC or write combining.
	3h-2h	Reserved.
	4h	WT or write through.
	5h	WP or write protect.
	6h	WB or write back.
	7h	Reserved.
31:29	Reserved.	
28	<b>RdDram_4K_EB000: Read DRAM.</b> 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.	
27	<b>WrDram_4K_EB000: Write DRAM.</b> 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.	
26:24	<b>MemType_4K_EB000: memory type.</b> Read-write. Reset: XXXb.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	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	<b>RdDram_4K_EA000: Read DRAM.</b> 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	<b>WrDram_4K_EA000: Write DRAM.</b> 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	<b>MemType_4K_EA000: memory type.</b> Read-write. Reset: XXXb.																
	<b>ValidValues:</b>																
	<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>	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.
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	<b>RdDram_4K_E9000: Read DRAM.</b> 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	<b>WrDram_4K_E9000: Write DRAM.</b> 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	<b>MemType_4K_E9000: memory type.</b> Read-write. Reset: XXXb.																
	<b>ValidValues:</b>																
	<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>	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.
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.																
4	<b>RdDram_4K_E8000: Read DRAM.</b> 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.																
	AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
3	<b>WrDram_4K_E8000: Write DRAM.</b> 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.																
	AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
2:0	<b>MemType_4K_E8000: memory type.</b> Read-write. Reset: XXXb. Address range from E8000h to E8FFFh.																
	<b>ValidValues:</b>																

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\_026E [Fixed-Size MTRRs] (Core::X86::Msrr::MtrrFix\_4K\_6)

See Core::X86::Msrr::MtrrVarBase for general MTRR information. Fixed MTRRs are enabled through Core::X86::Msrr::MTRRdefType[MtrrDefTypeFixEn,MtrrDefTypeEn]. For addresses below 1-MB, 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.

\_lthree0\_core[3:0]\_nSIZE4K6; MSR0000\_026E

Bits	Description																
63:61	Reserved.																
60	<b>RdDram_4K_F7000: Read DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
59	<b>WrDram_4K_F7000: Write DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
58:56	<b>MemType_4K_F7000: memory type.</b> Read-write. Reset: XXXb. <b>ValidValues:</b> <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>	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.
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	<b>RdDram_4K_F6000: Read DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
51	<b>WrDram_4K_F6000: Write DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
50:48	<b>MemType_4K_F6000: memory type.</b> Read-write. Reset: XXXb. <b>ValidValues:</b> <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>UC or uncacheable.</td></tr> <tr> <td>1h</td><td>WC or write combining.</td></tr> <tr> <td>3h-2h</td><td>Reserved.</td></tr> </table>	Value	Description	0h	UC or uncacheable.	1h	WC or write combining.	3h-2h	Reserved.								
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	<b>RdDram_4K_F5000: Read DRAM.</b> 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.	
43	<b>WrDram_4K_F5000: Write DRAM.</b> 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.	
42:40	<b>MemType_4K_F5000: memory type.</b> Read-write. Reset: XXXb.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	UC or uncacheable.
	1h	WC or write combining.
	3h-2h	Reserved.
	4h	WT or write through.
	5h	WP or write protect.
	6h	WB or write back.
	7h	Reserved.
39:37	Reserved.	
36	<b>RdDram_4K_F4000: Read DRAM.</b> 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.	
35	<b>WrDram_4K_F4000: Write DRAM.</b> 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.	
34:32	<b>MemType_4K_F4000: memory type.</b> Read-write. Reset: XXXb.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	UC or uncacheable.
	1h	WC or write combining.
	3h-2h	Reserved.
	4h	WT or write through.
	5h	WP or write protect.
	6h	WB or write back.
	7h	Reserved.
31:29	Reserved.	
28	<b>RdDram_4K_F3000: Read DRAM.</b> 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.	
27	<b>WrDram_4K_F3000: Write DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
26:24	<b>MemType_4K_F3000: memory type.</b> Read-write. Reset: XXXb.																
	<b>ValidValues:</b>																
	<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>	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.
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	<b>RdDram_4K_F2000: Read DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
19	<b>WrDram_4K_F2000: Write DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
18:16	<b>MemType_4K_F2000: memory type.</b> Read-write. Reset: XXXb.																
	<b>ValidValues:</b>																
	<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>	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.
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	<b>RdDram_4K_F1000: Read DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
11	<b>WrDram_4K_F1000: Write DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
10:8	<b>MemType_4K_F1000: memory type.</b> Read-write. Reset: XXXb.																
	<b>ValidValues:</b>																
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>UC or uncacheable.</td></tr> <tr> <td>1h</td><td>WC or write combining.</td></tr> <tr> <td>3h-2h</td><td>Reserved.</td></tr> <tr> <td>4h</td><td>WT or write through.</td></tr> <tr> <td>5h</td><td>WP or write protect.</td></tr> <tr> <td>6h</td><td>WB or write back.</td></tr> </table>	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.		
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.	
4	<b>RdDram_4K_F0000: Read DRAM.</b> 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.	
	AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.	
3	<b>WrDram_4K_F0000: Write DRAM.</b> 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.	
	AccessType: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.	
2:0	<b>MemType_4K_F0000: memory type.</b> Read-write. Reset: XXXb. Address range from F0000h to F0FFFh.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	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\_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 1-MB, 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.

\_lthree0\_core[3:0]\_nSIZE4K7; MSR0000\_026F

Bits	Description																
63:61	Reserved.																
60	<b>RdDram_4K_FF000: Read DRAM.</b> 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	<b>WrDram_4K_FF000: Write DRAM.</b> 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	<b>MemType_4K_FF000: memory type.</b> Read-write. Reset: XXXb. <b>ValidValues:</b> <table border="1"> <thead> <tr> <th>Value</th><th>Description</th></tr> </thead> <tbody> <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> </tbody> </table>	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.
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	<b>RdDram_4K_FE000: Read DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
51	<b>WrDram_4K_FE000: Write DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
50:48	<b>MemType_4K_FE000: memory type.</b> Read-write. Reset: XXXb. <b>ValidValues:</b> <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>	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.
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	<b>RdDram_4K_FD000: Read DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
43	<b>WrDram_4K_FD000: Write DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
42:40	<b>MemType_4K_FD000: memory type.</b> Read-write. Reset: XXXb. <b>ValidValues:</b> <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>	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.
Value	Description																
0h	UC or uncacheable.																
1h	WC or write combining.																
3h-2h	Reserved.																
4h	WT or write through.																
5h	WP or write protect.																
6h	WB or write back.																
7h	Reserved.																
39:37	Reserved.																
36	<b>RdDram_4K_FC000: Read DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
35	<b>WrDram_4K_FC000: Write DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
34:32	<b>MemType_4K_FC000: memory type.</b> Read-write. Reset: XXXb. <b>ValidValues:</b> <table> <tr> <th>Value</th><th>Description</th></tr> </table>	Value	Description														
Value	Description																



	0h	UC or uncacheable.
	1h	WC or write combining.
	3h-2h	Reserved.
	4h	WT or write through.
	5h	WP or write protect.
	6h	WB or write back.
	7h	Reserved.
31:29	Reserved.	
28	<b>RdDram_4K_FB000: Read DRAM.</b> 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.	
27	<b>WrDram_4K_FB000: Write DRAM.</b> 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.	
26:24	<b>MemType_4K_FB000: memory type.</b> Read-write. Reset: XXXb.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	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	<b>RdDram_4K_FA000: Read DRAM.</b> 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	<b>WrDram_4K_FA000: Write DRAM.</b> 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	<b>MemType_4K_FA000: memory type.</b> Read-write. Reset: XXXb.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	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	<b>RdDram_4K_F9000: Read DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
11	<b>WrDram_4K_F9000: Write DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
10:8	<b>MemType_4K_F9000: memory type.</b> Read-write. Reset: XXXb. <b>ValidValues:</b>																
	<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>	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.
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.																
4	<b>RdDram_4K_F8000: Read DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramEn,MtrrFixDramModEn] masks Reads of the stored value. AccessType: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
3	<b>WrDram_4K_F8000: Write DRAM.</b> 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::Msrr::SYS_CFG[MtrrFixDramEn,MtrrFixDramModEn] masks Reads of the stored value. AccessType: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? Read-write : Read,Error-on-write-1. Reset: Core::X86::Msrr::SYS_CFG[MtrrFixDramModEn] ? X : Fixed,0.																
2:0	<b>MemType_4K_F8000: memory type.</b> Read-write. Reset: XXXb. Address range from F8000h to F8FFFh. <b>ValidValues:</b>																
	<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>	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.
Value	Description																
0h	UC or uncacheable.																
1h	WC or write combining.																
3h-2h	Reserved.																
4h	WT or write through.																
5h	WP or write protect.																
6h	WB or write back.																
7h	Reserved.																

#### MSR0000\_0277 [Page Attribute Table] (Core::X86::Msrr::PAT)

This register specifies the memory type based on the PAT, PCD, and PWT bits in the virtual address page tables.

\_lthree0\_core[3:0]\_thread[1:0]; MSR0000\_0277

Bits	Description												
63:59	Reserved.												
58:56	<b>PA7MemType.</b> Read-write. Reset: 0h. Default UC. MemType for {PAT, PCD, PWT} = 7h. <b>ValidValues:</b>												
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>UC or uncacheable.</td></tr> <tr> <td>1h</td><td>WC or write combining.</td></tr> <tr> <td>3h-2h</td><td>Reserved.</td></tr> <tr> <td>4h</td><td>WT or write through.</td></tr> <tr> <td>5h</td><td>WP or write protect.</td></tr> </table>	Value	Description	0h	UC or uncacheable.	1h	WC or write combining.	3h-2h	Reserved.	4h	WT or write through.	5h	WP or write protect.
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.
55:51	Reserved.	
50:48	<b>PA6MemType.</b> Read-write. Reset: 7h. Default UC. MemType for {PAT, PCD, PWT} = 6h.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	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.
47:43	Reserved.	
42:40	<b>PA5MemType.</b> Read-write. Reset: 4h. Default WT. MemType for {PAT, PCD, PWT} = 5h.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	UC or uncacheable.
	1h	WC or write combining.
	3h-2h	Reserved.
	4h	WT or write through.
	5h	WP or write protect.
	6h	WB or write back.
	7h	Reserved.
39:35	Reserved.	
34:32	<b>PA4MemType.</b> Read-write. Reset: 6h. Default WB. MemType for {PAT, PCD, PWT} = 4h.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	UC or uncacheable.
	1h	WC or write combining.
	3h-2h	Reserved.
	4h	WT or write through.
	5h	WP or write protect.
	6h	WB or write back.
	7h	Reserved.
31:27	Reserved.	
26:24	<b>PA3MemType.</b> Read-write. Reset: 0h. Default UC. MemType for {PAT, PCD, PWT} = 3h.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	UC or uncacheable.
	1h	WC or write combining.
	3h-2h	Reserved.
	4h	WT or write through.
	5h	WP or write protect.
	6h	WB or write back.
	7h	Reserved.
23:19	Reserved.	
18:16	<b>PA2MemType.</b> Read-write. Reset: 7h. Default UC. MemType for {PAT, PCD, PWT} = 2h.	

	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	UC or uncacheable.
	1h	WC or write combining.
	3h-2h	Reserved.
	4h	WT or write through.
	5h	WP or write protect.
	6h	WB or write back.
	7h	Reserved.
15:11	Reserved.	
10:8	<b>PA1MemType.</b> Read-write. Reset: 4h. Default WT. MemType for {PAT, PCD, PWT} = 1h.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	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.
7:3	Reserved.	
2:0	<b>PA0MemType.</b> Read-write. Reset: 6h. MemType for {PAT, PCD, PWT} = 0h.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	UC or uncacheable.
	1h	WC or write combining.
	3h-2h	Reserved.
	4h	WT or write through.
	5h	WP or write protect.
	6h	WB or write back.
	7h	Reserved.

### MSR0000\_02FF [MTRR Default Memory Type] (Core::X86::Msr::MTRRdefType)

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

\_lthree0\_core[3:0]; MSR0000\_02FF

Bits	Description
63:12	Reserved.
11	<b>MtrrDefTypeEn: variable and fixed MTRR enable.</b> 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	<b>MtrrDefTypeFixEn: fixed MTRR enable.</b> 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	<b>MemType: memory type.</b> Read-write. Reset: 00h. <b>Description:</b> 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).
--	-----------------------------------

### 2.1.13.2 MSRs - MSRC000\_0xxx

See 1.4.3 [Register Mnemonics] for a description of the register naming convention. MSRs are accessed through x86 WRMSR and RDMSR instructions.

#### MSRC000\_0080 [Extended Feature Enable] (Core::X86::Msr::EFER)

SKINIT Execution: 0000\_0000\_0000\_0000h.

\_lthree0\_core[3:0]\_thread[1:0]; MSRC000\_0080

Bits	Description
63:16	Reserved.
15	<b>TCE: translation cache extension enable.</b> Read-write. Reset: 0. 1=Translation cache extension is enabled. PDC entries related to the linear address of the INVLPG instruction are invalidated. If <VALUE> == 0 all PDC entries are invalidated by the INVLPG instruction.
14	<b>FFXSE: fast FXSAVE/FRSTOR enable.</b> 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	<b>LMSLE: long mode segment limit enable.</b> Read-write. Reset: 0. 1=Enables the long mode segment limit check mechanism.
12	<b>SVME: secure virtual machine (SVM) enable.</b> Reset: Fixed,0. 1=SVM features are enabled. AccessType: Core::X86::Msr::VM_CR[SvmDisable] ? Read-only,Error-on-write-1 : Read-write.
11	<b>NXE: no-execute page enable.</b> Read-write. Reset: 0. 1=The no-execute page protection feature is enabled.
10	<b>LMA: long mode active.</b> 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	<b>LME: long mode enable.</b> Read-write. Reset: 0. 1=Long mode is enabled.
7:1	Reserved.
0	<b>SYSCALL: system call extension enable.</b> 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.

\_lthree0\_core[3:0]\_thread[1:0]; MSRC000\_0081

Bits	Description
63:48	<b>SysRetSel: SYSRET CS and SS.</b> Read-write. Reset: 0000h.
47:32	<b>SysCallSel: SYSCALL CS and SS.</b> Read-write. Reset: 0000h.
31:0	<b>Target: SYSCALL target address.</b> Read-write. Reset: 0000_0000h.

#### MSRC000\_0082 [Long Mode SYSCALL Target Address] (Core::X86::Msr::STAR64)

Read-write. Reset: 0000\_0000\_0000\_0000h.

\_lthree0\_core[3:0]\_thread[1:0]; MSRC000\_0082

Bits	Description
63:0	<b>LSTAR: long mode target address.</b> 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.

\_lthree0\_core[3:0]\_thread[1:0]; MSRC000\_0083

Bits	Description
63:0	<b>CSTAR: compatibility mode target address.</b> 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)**

\_lthree0\_core[3:0]\_thread[1:0]; MSRC000\_0084

Bits	Description
63:32	Reserved.
31:0	<b>Mask: SYSCALL flag mask.</b> 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.

\_lthree0\_core[3:0]\_thread[1:0]; MSRC000\_00E7

Bits	Description
63:0	<b>MPerfReadOnly: Read-only maximum core clocks counter.</b> 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], 2.1.4 [Effective Frequency]. This register is not affected by writes to Core::X86::Msr::MPERF. AccessType: Core::X86::Msr::HWCR[EffFreqReadOnlyLock] ? Read-only, Volatile : Read-write, Volatile.

**MSRC000\_00E8 [Read-Only Actual Performance Frequency Clock Count] (Core::X86::Msr::APerfReadOnly)**

Reset: 0000\_0000\_0000\_0000h.

\_lthree0\_core[3:0]\_thread[1:0]; MSRC000\_00E8

Bits	Description
63:0	<b>APerfReadOnly: Read-only actual core clocks counter.</b> 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-only, Volatile : Read-write, Volatile.

**MSRC000\_00E9 [Instructions Retired Performance Count] (Core::X86::Msr::IRPerfCount)**

Reset: 0000\_0000\_0000\_0000h.

\_lthree0\_core[3:0]\_thread[1:0]; MSRC000\_00E9

Bits	Description
63:0	<b>IRPerfCount: instructions retired counter.</b> Reset: 0000_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-only, Volatile : Read-write, Volatile.

**MSRC000\_0100 [FS Base] (Core::X86::Msr::FS\_BASE)**

Read-write. Reset: 0000\_0000\_0000\_0000h.

\_lthree0\_core[3:0]\_thread[1:0]; MSRC000\_0100

Bits	Description
63:0	<b>FSBase: expanded FS segment base.</b> 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.

\_lthree0\_core[3:0]\_thread[1:0]; MSRC000\_0101

Bits	Description
63:0	<b>GSBase: expanded GS segment base.</b> 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.

\_lthree0\_core[3:0]\_thread[1:0]; MSRC000\_0102

Bits	Description
63:0	<b>KernelGSBase: kernel data structure pointer.</b> Read-write. Reset: 0000_0000_0000_0000h. This register holds the kernel data structure pointer which can be swapped with the GS_BASE register using the SwapGS instruction. The address stored in this register must be in canonical form (if not canonical, a #GP fault occurs).

**MSRC000\_0103 [Auxiliary Time Stamp Counter] (Core::X86::Msr::TSC\_AUX)**

Read-write, Volatile. Reset: 0000\_0000\_0000\_0000h.

\_lthree0\_core[3:0]\_thread[1:0]; MSRC000\_0103

Bits	Description
63:32	Reserved.
31:0	<b>TscAux: auxiliary time stamp counter data.</b> 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 while in host mode or when virtualization is not being used or when accessed by code executed in system management mode (SMM) unless the SMM code is executed within a guest container. The TSC Ratio value does not affect the rate of the underlying TSC, MPERF, and MPerfReadOnly counters, or the value that gets written to the TSC, MPERF, and MPerfReadOnly MSRs counters on a Write by either the host or the guest. The TSC Ratio MSR contains a fixed-point number in 8.32 format, which is 8 bits of integer and 32 bits of fraction. This number is the ratio of the desired P0 frequency to the P0 frequency of the core. The reset value of the TSC Ratio MSR is 1.0, which results in a guest frequency matches the core P0 frequency.

\_lthree0\_core[3:0]\_thread[1:0]; MSRC000\_0104

Bits	Description
63:40	Reserved.
39:32	<b>TscRateMsrInt: time stamp counter rate integer.</b> Read-write. Reset: 01h. Specifies the integer part of the MSR TSC ratio value.
31:0	<b>TscRateMsrFrac: time stamp counter rate fraction.</b> Read-write. Reset: 0000_0000h. Specifies the fractional part of the MSR TSC ratio value.

**MSRC000\_0410 [MCA Interrupt Configuration] (Core::X86::Msr::McaIntrCfg)**

Read-write. Reset: 0000\_0000\_0000\_0000h.

MSRC000\_0410

Bits	Description
63:16	Reserved.
15:12	<b>ThresholdLvtOffset.</b> 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	<b>DeferredLvtOffset.</b> Read-write. Reset: 0h. <b>Description:</b> For deferred error interrupts, specifies the address of the LVT entry in the APIC registers as follows: LVT address = (LvtOffset shifted left 4 bits) + 500h (see APIC[530:500]).
3:0	Reserved.

### 2.1.13.2.1 MSRs - MSRC000\_2xxx

The MCA registers including the legacy aliases (MSR0000\_000[1:0], MSR0000\_04xx) are mapped to MSRC000\_2xxx. See 3.2.5 [MCA Banks].

### 2.1.13.3 MSRs - MSRC001\_0xxx

See 1.4.3 [Register Mnemonics] for a description of the register naming convention. MSRs are accessed through x86 WRMSR and RDMSR instructions.

#### MSRC001\_000[0...3] [Performance Event Select [3:0]] (Core::X86::Msr::PERF\_LEGACY\_CTL)

Read-write. Reset: 0000_0000_0000_0000h.									
The legacy alias of Core::X86::Msr::PERF_CTL. See Core::X86::Msr::PERF_CTL.									
_lthree0_core[3:0]_thread[1:0]_n0; MSRC001_0000									
_lthree0_core[3:0]_thread[1:0]_n1; MSRC001_0001									
_lthree0_core[3:0]_thread[1:0]_n2; MSRC001_0002									
_lthree0_core[3:0]_thread[1:0]_n3; MSRC001_0003									
Bits	Description								
63:42	Reserved.								
41:40	<b>HostGuestOnly: count only host/guest events.</b> Read-write. Reset: 0h.								
39:36	Reserved.								
35:32	<b>EventSelect[11:8]: performance event select.</b> Read-write. Reset: 0h.								
31:24	<b>CntMask: counter mask.</b> Read-write. Reset: 00h. Controls the number of events counted per clock cycle. <b>Valid Values:</b>								
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>00h</td><td>The corresponding PERF_CTR[5:0] register increments by the number of events occurring in a clock cycle. See 2.1.14.2 [Large Increment per Cycle Events] for events that can increment greater than 15 per cycle.</td></tr> <tr> <td>7Fh-01h</td><td>When Inv == 0, the corresponding PERF_CTR[5:0] register increments by 1, if the number of events occurring in a clock cycle is greater than or equal to the CntMask value. When Inv == 1, the corresponding PERF_CTR[5:0] register increments by 1, if the number of events occurring in a clock cycle is less than CntMask value.</td></tr> <tr> <td>FFh-80h</td><td>Reserved.</td></tr> </table>	Value	Description	00h	The corresponding PERF_CTR[5:0] register increments by the number of events occurring in a clock cycle. See 2.1.14.2 [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.
Value	Description								
00h	The corresponding PERF_CTR[5:0] register increments by the number of events occurring in a clock cycle. See 2.1.14.2 [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	<b>Inv: invert counter mask.</b> Read-write. Reset: 0.								
22	<b>En: enable performance counter.</b> Read-write. Reset: 0.								
21	Reserved.								
20	<b>Int: enable APIC interrupt.</b> Read-write. Reset: 0.								
19	Reserved.								
18	<b>Edge: edge detect.</b> Read-write. Reset: 0.								
17:16	<b>OsUserMode: OS and user mode.</b> Read-write. Reset: 0h.								

15:8	<b>UnitMask: event qualification.</b> Read-write. Reset: 00h. When selecting an event for which not all UnitMask bits are defined, the undefined UnitMask bits should be set to zero.
7:0	<b>EventSelect[7:0]: event select.</b> Read-write. Reset: 00h.

#### MSRC001\_000[4...7] [Performance Event Counter [3:0]] (Core::X86::Msrr::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 2.1.14.2 [Large Increment per Cycle Events].

The legacy alias of Core::X86::Msrr::PERF\_CTR. See Core::X86::Msrr::PERF\_CTR.

\_lthree0\_core[3:0]\_thread[1:0]\_n0; MSRC001\_0004

\_lthree0\_core[3:0]\_thread[1:0]\_n1; MSRC001\_0005

\_lthree0\_core[3:0]\_thread[1:0]\_n2; MSRC001\_0006

\_lthree0\_core[3:0]\_thread[1:0]\_n3; MSRC001\_0007

Bits	Description
63:48	Reserved.
47:0	<b>CTR: performance counter value.</b> Read-write, Volatile. Reset: 0000_0000_0000h. In special cases (see 2.1.14.2 [Large Increment per Cycle Events]) CTR can appear as a 64-bit counter.

#### MSRC001\_0010 [System Configuration] (Core::X86::Msrr::SYS\_CFG)

Reset: 0000\_0000\_0000\_0000h.

\_lthree0\_core[3:0]; MSRC001\_0010

Bits	Description
63:24	Reserved.
23	<b>SMEE: secure memory encryption enable.</b> Reset: 0. 0=Memory encryption features are disabled. 1=Memory encryption features are enabled. Once Core::X86::Msrr::HWCR[SmmLock] is set, this bit cannot be cleared until a reset. For enabling secure memory encryption see 2.1.3 [Memory Encryption]. AccessType: Core::X86::Msrr::HWCR_thread0[SmmLock] ? Read,Write-1-only : Read-write.
22	<b>Tom2ForceMemTypeWB: top of memory 2 memory type write back.</b> Read-write. Reset: 0. 1=The default memory type of memory between 4-GB and Core::X86::Msrr::TOM2 is Write-back instead of the memory type defined by Core::X86::Msrr::MTRRdefType[MemType]. For this bit to have any effect, Core::X86::Msrr::MTRRdefType[MtrrDefTypeEn] must be 1. MTRRs and PAT can be used to override this memory type.
21	<b>MtrrTom2En: MTRR top of memory 2 enable.</b> Read-write. Reset: 0. 0=Core::X86::Msrr::TOM2 is disabled. 1=Core::X86::Msrr::TOM2 is enabled.
20	<b>MtrrVarDramEn: MTRR variable DRAM enable.</b> Read-write. Reset: 0. Init: BIOS,1. 0=Core::X86::Msrr::TOP_MEM and IORRs are disabled. 1=These registers are enabled.
19	<b>MtrrFixDramModEn: MTRR fixed RdDram and WrDram modification enable.</b> Read-write. Reset: 0. 0=Core::X86::Msrr::MtrrFix_64K through Core::X86::Msrr::MtrrFix_4K_7 [RdDram,WrDram] read values is masked 00b; writing does not change the hidden value. 1=Core::X86::Msrr::MtrrFix_64K through Core::X86::Msrr::MtrrFix_4K_7 [RdDram,WrDram] access type is Read-write. Not shared between threads. Controls access to Core::X86::Msrr::MtrrFix_64K through Core::X86::Msrr::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	<b>MtrrFixDramEn: MTRR fixed RdDram and WrDram attributes enable.</b> Read-write. Reset: 0. Init: BIOS,1. 1=Enables the RdDram and WrDram attributes in Core::X86::Msrr::MtrrFix_64K through Core::X86::Msrr::MtrrFix_4K_7.
17:0	Reserved.

#### MSRC001\_0015 [Hardware Configuration] (Core::X86::Msrr::HWCR)

Reset: 0000\_0000\_0100\_0010h.

\_lthree0\_core[3:0]\_thread[1:0]; MSRC001\_0015

Bits	Description
63:31	Reserved.

30	<b>IRPerfEn: enable instructions retired counter.</b> Read-write. Reset: 0. 1=Enable Core::X86::Msrr::IRPerfCount.
29:28	Reserved.
27	<b>EffFreqReadOnlyLock: read-only effective frequency counter lock.</b> Write-1-only. Reset: 0. Init: BIOS, 1. 1=Core::X86::Msrr::MPerfReadOnly, Core::X86::Msrr::APerfReadOnly and Core::X86::Msrr::IRPerfCount are Read-only.
26	<b>EffFreqCntMwait: effective frequency counting during mwait.</b> Read-write. Reset: 0. 0=The registers do not increment. 1=The registers increment. Specifies whether Core::X86::Msrr::MPERF and Core::X86::Msrr::APERF increment while the core is in the monitor event pending state. See 2.1.4 [Effective Frequency].
25	<b>CpbDis: core performance boost disable.</b> 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	<b>TscFreqSel: TSC frequency select.</b> Read-only. Reset: 1. 1=The TSC increments at the P0 frequency.
23:22	Reserved.
21	<b>LockTscToCurrentP0: lock the TSC to the current P0 frequency.</b> 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	<b>IoCfgGpFault: IO-space configuration causes a GP fault.</b> 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::Msrr::SMI_ON_IO_TRAP_CTL_STS.
19	Reserved.
18	<b>McStatusWrEn: machine check status write enable.</b> Read-write. Reset: 0. 0=MCi_STATUS registers are Readable; Writing a non-zero pattern to these registers causes a general protection fault. 1=MCi_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. <b>Description:</b> McStatusWrEn can be used to debug machine check exception and interrupt handlers. See 3.1 [Machine Check Architecture].
17	<b>Wrap32Dis: 32-bit address wrap disable.</b> 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::Msrr::FS_BASE and Core::X86::Msrr::GS_BASE. Then it would address $\pm 2$ 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	<b>RsmSpCycDis: RSM special bus cycle disable.</b> Reset: 0. 0=A link special bus cycle, SMIACK, is generated on a resume from SMI. AccessType: Core::X86::Msrr::HWCR[SmmLock] ? Read-only : Read-write.
13	<b>SmiSpCycDis: SMI special bus cycle disable.</b> Reset: 0. 0=A link special bus cycle, SMIACK, is generated when an SMI interrupt is taken. AccessType: Core::X86::Msrr::HWCR[SmmLock] ? Read-only : Read-write.
12:11	Reserved.
10	<b>MonMwaitUserEn: MONITOR/MWAIT user mode enable.</b> 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	<b>MonMwaitDis: MONITOR and MWAIT disable.</b> Read-write. Reset: 0. 1=The MONITOR and MWAIT opcodes become invalid. This affects what is reported back through Core::X86::Cpuid::FeatureIdEcX[Monitor].

8	<b>IgnneEm: IGNNE port emulation enable.</b> Read-write. Reset: 0. 1=Enable emulation of IGNNE port.
7	<b>AllowFerrOnNe: allow FERR on NE.</b> Read-write. Reset: 0. 0=Disable legacy FERR signaling and generate FERR exception directly. 1=Legacy FERR signaling.
6:5	Reserved.
4	<b>INVDWBINVD: INVD to WBINVD conversion.</b> Read-write. Reset: 1. 1=Convert INVD to WBINVD. <b>Description:</b> This bit is required to be set for normal operation when any of the following are true: <ul style="list-style-type: none"> <li>• An L2 is shared by multiple threads.</li> <li>• An L3 is shared by multiple cores.</li> <li>• CC6 is enabled.</li> <li>• Probe filter is enabled.</li> </ul>
3	<b>TlbCacheDis: cacheable memory disable.</b> Read-write. Reset: 0. 1=Disable performance improvement that assumes that the PML4, PDP, PDE and PTE entries are in cacheable WB DRAM. <b>Description:</b> Operating systems that maintain page tables in any other memory type must set the TlbCacheDis bit to insure proper operation. <ul style="list-style-type: none"> <li>• TlbCacheDis does not override the memory type specified by the SMM ASeg and TSeg memory regions controlled by Core::X86::Msrr::SMMAddr Core::X86::Msrr::SMMMask.</li> </ul>
2:1	Reserved.
0	<b>SmmLock: SMM code lock.</b> 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 2.1.11.1.10 [Locking SMM].

#### MSRC001\_001[6...8] [IO Range Base] (Core::X86::Msrr::IIRR\_BASE)

Read-write.

Core::X86::Msrr::IIRR\_BASE and Core::X86::Msrr::IIRR\_MASK combine to specify the two sets of base and mask pairs for two IIRR ranges. A core access, with address CPUAddr, is determined to be within IIRR address range if the following equation is true:

$CPUAddr[47:12] \& PhyMask[47:12] == PhyBase[47:12] \& PhyMask[47:12]$ .

BIOS can use the IIRRs to create an IO hole within a range of addresses that would normally be mapped to DRAM. It can also use the IIRRs to re-assert a DRAM destination for a range of addresses that fall within a bigger IO hole that overlays DRAM.

\_lthree0\_core[3:0]\_n0; MSRC001\_0016

\_lthree0\_core[3:0]\_n1; MSRC001\_0018

Bits	Description
63:48	Reserved.
47:12	<b>PhyBase: physical base address.</b> Read-write. Reset: X_XXXX_XXXXh.
11:5	Reserved.
4	<b>RdMem: read from memory.</b> 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	<b>WrMem: write to memory.</b> 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::Msrr::IIRR\_MASK)

Read-write. Reset: 0000\_0000\_0000\_0000h.

See Core::X86::Msrr::IIRR\_BASE.

\_lthree0\_core[3:0]\_n0; MSRC001\_0017

\_lthree0\_core[3:0]\_n1; MSRC001\_0019

Bits	Description
63:48	Reserved.
47:12	<b>PhyMask: physical address mask.</b> Read-write. Reset: 0_0000_0000h.
11	<b>Valid.</b> Read-write. Reset: 0. 1=The pair of registers that specifies an IIRR range is valid.
10:0	Reserved.

**MSRC001\_001A [Top Of Memory] (Core::X86::Msr::TOP\_MEM)**

Read-write.

\_lthree0\_core[3:0]; MSRC001\_001A

Bits	Description
63:48	Reserved.
47:23	<b>TOM[47:23]: top of memory.</b> Read-write. Reset: XXX_XXXXh. Specifies the address that divides between MMIO and DRAM. This value is normally placed below 4-GB. From TOM to (4-GB - 1) is MMIO; below TOM is DRAM. See 2.1.5.3 [System Address Map].
22:0	Reserved.

**MSRC001\_001D [Top Of Memory 2] (Core::X86::Msr::TOM2)**

Read-write.

\_lthree0\_core[3:0]; MSRC001\_001D

Bits	Description
63:48	Reserved.
47:23	<b>TOM2[47:23]: second top of memory.</b> Read-write. Reset: XXX_XXXXh. Specifies the address divides between MMIO and DRAM. This value is normally placed above 4 GBs. From 4-GB to (TOM2 - 1) is DRAM; TOM2 and above is MMIO. See 2.1.5.3 [System Address Map]. This register is enabled by Core::X86::Msr::SYS_CFG[MtrrTom2En].
22:0	Reserved.

**MSRC001\_0022 [Machine Check Exception Redirection] (Core::X86::Msr::McExcepRedir)**

Read-write. Reset: 0000\_0000\_0000\_0000h.

This register can be used to redirect machine check exceptions (MCEs) to SMIs or vectored interrupts. If both RedirSmiEn and RedirVecEn are set, then undefined behavior results.

\_lthree0\_core[3:0]\_thread[1:0]; MSRC001\_0022

Bits	Description
63:10	Reserved.
9	<b>RedirSmiEn.</b> 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	<b>RedirVecEn.</b> 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	<b>RedirVector.</b> Read-write. Reset: 00h. See RedirVecEn.

**MSRC001\_003[0...5] [Processor Name String] (Core::X86::Msr::ProcNameString)**

Read-write.

These 6 registers hold the CUID name string in ASCII. The state of these registers are returned by CUID 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.

\_lthree0\_core[3:0]\_thread[1:0]\_n0; MSRC001\_0030

\_lthree0\_core[3:0]\_thread[1:0]\_n1; MSRC001\_0031

\_lthree0\_core[3:0]\_thread[1:0]\_n2; MSRC001\_0032

\_lthree0\_core[3:0]\_thread[1:0]\_n3; MSRC001\_0033

\_lthree0\_core[3:0]\_thread[1:0]\_n4; MSRC001\_0034

\_lthree0\_core[3:0]\_thread[1:0]\_n5; MSRC001\_0035

Bits	Description
63:56	<b>CpuNameString7.</b> Read-write. Reset: XXh.
55:48	<b>CpuNameString6.</b> Read-write. Reset: XXh.
47:40	<b>CpuNameString5.</b> Read-write. Reset: XXh.



39:32	<b>CpuNameString4</b> . Read-write. Reset: XXh.
31:24	<b>CpuNameString3</b> . Read-write. Reset: XXh.
23:16	<b>CpuNameString2</b> . Read-write. Reset: XXh.
15:8	<b>CpuNameString1</b> . Read-write. Reset: XXh.
7:0	<b>CpuNameString0</b> . Read-write. Reset: XXh.

#### MSRC001\_005[0...3] [IO Trap] (Core::X86::Msr::SMI\_ON\_IO\_TRAP)

Read-write. Reset: 0000\_0000\_0000\_0000h.

Core::X86::Msr::SMI\_ON\_IO\_TRAP and Core::X86::Msr::SMI\_ON\_IO\_TRAP\_CTL\_STS provide a mechanism for executing the SMI handler if a an access to one of the specified addresses is detected. Access address and access type checking is performed before IO instruction execution. If the access address and access type match one of the specified IO address and access types, then: (1) the IO instruction is not executed; (2) any breakpoint, other than the single-step breakpoint, set on the IO instruction is not taken (the single-step breakpoint is taken after resuming from SMM); and (3) issue the SMI-trigger IO cycle specified by Core::X86::Msr::SmiTrigIoCycle if enabled. The status is stored in Core::X86::Smm::LocalSmiStatus[IoTrapSts].

IO-space configuration accesses are special IO accesses. An IO access is defined as an IO-space configuration access when IO instruction address bits[31:0] are CFCh, CFDh, CFEh, or CFFh when IO-space configuration is enabled (IO::IoCfgAddr[ConfigEn]). The access address for a configuration space access is the current value of IO::IoCfgAddr[BusNo,Device,Function,RegNo]. The access address for an IO access that is not a configuration access is equivalent to the IO instruction address, bits[31:0].

The access address is compared with SmiAddr, and the instruction access type is compared with the enabled access types defined by ConfigSmi, SmiOnRdEn, and SmiOnWrEn. Access address bits[23:0] can be masked with SmiMask. IO and configuration space trapping to SMI applies only to single IO instructions; it does not apply to string and REP IO instructions. The conditional GP fault described by Core::X86::Msr::HWCR[IoCfgGpFault] takes priority over this trap.

\_lthree0\_core[3:0]\_thread[1:0]\_n0; MSRC001\_0050

\_lthree0\_core[3:0]\_thread[1:0]\_n1; MSRC001\_0051

\_lthree0\_core[3:0]\_thread[1:0]\_n2; MSRC001\_0052

\_lthree0\_core[3:0]\_thread[1:0]\_n3; MSRC001\_0053

Bits	Description
63	<b>SmiOnRdEn: enable SMI on IO read</b> . Read-write. Reset: 0. 1=Enables SMI generation on a Read access.
62	<b>SmiOnWrEn: enable SMI on IO write</b> . Read-write. Reset: 0. 1=Enables SMI generation on a Write access.
61	<b>ConfigSmi: configuration space SMI</b> . Read-write. Reset: 0. 0=IO access (that is not an IO-space configuration access). 1=Configuration access.
60:56	Reserved.
55:32	<b>SmiMask[23:0]</b> . Read-write. Reset: 00_0000h. 1=Do not mask address bit. 0=Mask address bit. SMI IO trap mask.
31:0	<b>SmiAddr[31:0]</b> . Read-write. Reset: 0000_0000h. SMI IO trap address.

#### MSRC001\_0054 [IO Trap Control] (Core::X86::Msr::SMI\_ON\_IO\_TRAP\_CTL\_STS)

\_lthree0\_core[3:0]\_thread[1:0]; MSRC001\_0054

Bits	Description
63:16	Reserved.
15	<b>IoTrapEn: IO trap enable</b> . 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	<b>SmiEn3</b> . Read-write. Reset: 0. 1=The trap Core::X86::Msr::SMI_ON_IO_TRAP_n[3] is enabled.
6	Reserved.
5	<b>SmiEn2</b> . Read-write. Reset: 0. 1=The trap Core::X86::Msr::SMI_ON_IO_TRAP_n[2] is enabled.
4	Reserved.
3	<b>SmiEn1</b> . Read-write. Reset: 0. 1=The trap Core::X86::Msr::SMI_ON_IO_TRAP_n[1] is enabled.
2	Reserved.
1	<b>SmiEn0</b> . 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.

\_lthree0\_core[3:0]; 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 2.1.11.1.3 [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.

\_lthree0\_core[3:0]; thread[1:0]; MSRC001\_0056

Bits	Description
63:27	Reserved.
26	<b>IoRd: IO Read.</b> Read-write. Reset: 0. 0=IO Write. 1=IO Read.
25	<b>IoCycleEn: IO cycle enable.</b> Read-write. Reset: 0. 1=The SMI trigger IO cycle is enabled to be generated.
24	Reserved.
23:16	<b>IoData.</b> Read-write. Reset: 00h.
15:0	<b>IoPortAddress.</b> Read-write. Reset: 0000h.

**MSRC001\_0058 [MMIO Configuration Base Address] (Core::X86::Msr::MmioCfgBaseAddr)**

See 2.1.6 [Configuration Space] for a description of MMIO configuration space.

\_lthree0\_core[3:0]; MSRC001\_0058

Bits	Description																						
63:48	Reserved.																						
47:20	<b>MmioCfgBaseAddr[47:20]: MMIO configuration base address bits[47:20].</b> Read-write. Reset: XXX_XXXXh. Specifies the base address of the MMIO configuration range.																						
19:6	Reserved.																						
5:2	<b>BusRange: bus range identifier.</b> 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. <b>Valid Values:</b> <table border="1"> <thead> <tr> <th>Value</th><th>Description</th></tr> </thead> <tbody> <tr><td>0h</td><td>1</td></tr> <tr><td>1h</td><td>2</td></tr> <tr><td>2h</td><td>4</td></tr> <tr><td>3h</td><td>8</td></tr> <tr><td>4h</td><td>16</td></tr> <tr><td>5h</td><td>32</td></tr> <tr><td>6h</td><td>64</td></tr> <tr><td>7h</td><td>128</td></tr> <tr><td>8h</td><td>256</td></tr> <tr><td>Fh-9h</td><td>Reserved</td></tr> </tbody> </table>	Value	Description	0h	1	1h	2	2h	4	3h	8	4h	16	5h	32	6h	64	7h	128	8h	256	Fh-9h	Reserved
Value	Description																						
0h	1																						
1h	2																						
2h	4																						
3h	8																						
4h	16																						
5h	32																						
6h	64																						
7h	128																						
8h	256																						
Fh-9h	Reserved																						
1	Reserved.																						
0	<b>Enable.</b> Read-write. Reset: 0. 1=MMIO configuration space is enabled.																						

**MSRC001\_0061 [P-state Current Limit] (Core::X86::Msr::PStateCurLim)**



_lthree0_core[3:0]; MSRC001_0061	
Bits	Description
63:7	Reserved.
6:4	<b>PstateMaxVal: P-state maximum value.</b> 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::Msrr::PStateCtl[PstateCmd] to a lower-performance P-state (higher value) are clipped to the value of this field.
3	Reserved.
2:0	<b>CurPstateLimit: current P-state limit.</b> Read,Error-on-write, Volatile. Reset: XXXb. Specifies the highest-performance P-state (lowest value) allowed. CurPstateLimit is always bounded by Core::X86::Msrr::PStateCurLim[PstateMaxVal]. Attempts to change the CurPstateLimit to a value greater (lower performance) than Core::X86::Msrr::PStateCurLim[PstateMaxVal] leaves CurPstateLimit unchanged.

#### MSRC001\_0062 [P-state Control] (Core::X86::Msrr::PStateCtl)

_lthree0_core[3:0]_thread[1:0]; MSRC001_0062	
Bits	Description
63:3	Reserved.
2:0	<b>PstateCmd: P-state change command.</b> 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::Msrr::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::Msrr::PStateStat)

Read,Error-on-write, Volatile.	
_lthree0_core[3:0]; MSRC001_0063	
Bits	Description
63:3	Reserved.
2:0	<b>CurPstate: current P-state.</b> 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::Msrr::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::Msrr::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; MSRC001_0064	
_n1; MSRC001_0065	
_n2; MSRC001_0066	
_n3; MSRC001_0067	
_n4; MSRC001_0068	
_n5; MSRC001_0069	
_n6; MSRC001_006A	
_n7; MSRC001_006B	
Bits	Description
63	<b>PstateEn.</b> Read-write. Reset: X. 0=The P-state specified by this MSR is not valid. 1=The P-state specified by this MSR is valid. The purpose of this register is to indicate if the rest of the P-state information in the register is valid after a reset; it controls no hardware.
62:32	Reserved.
31:30	<b>IddDiv: current divisor.</b> Read-write. Reset: XXb. See IddValue.
29:22	<b>IddValue: current value.</b> Read-write. Reset: XXXXXXXXXb. 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	<b>CpuVid[7:0]: core VID.</b> Read-write. Reset: XXXXXXXXb.																																																		
13:8	<p><b>CpuDfsId: core divisor ID.</b> Read-write. Reset: XXXXXXXb. 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] &gt; 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.</p> <p><b>ValidValues:</b></p> <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>00h</td><td>Off</td></tr> <tr> <td>07h-01h</td><td>Reserved.</td></tr> <tr> <td>08h</td><td>VCO/1</td></tr> <tr> <td>09h</td><td>VCO/1.125</td></tr> <tr> <td>1Ah-0Ah</td><td>VCO/&lt;Value/8&gt;</td></tr> <tr> <td>1Bh</td><td>Reserved.</td></tr> <tr> <td>1Ch</td><td>VCO/&lt;Value/8&gt;</td></tr> <tr> <td>1Dh</td><td>Reserved.</td></tr> <tr> <td>1Eh</td><td>VCO/&lt;Value/8&gt;</td></tr> <tr> <td>1Fh</td><td>Reserved.</td></tr> <tr> <td>20h</td><td>VCO/&lt;Value/8&gt;</td></tr> <tr> <td>21h</td><td>Reserved.</td></tr> <tr> <td>22h</td><td>VCO/&lt;Value/8&gt;</td></tr> <tr> <td>23h</td><td>Reserved.</td></tr> <tr> <td>24h</td><td>VCO/&lt;Value/8&gt;</td></tr> <tr> <td>25h</td><td>Reserved.</td></tr> <tr> <td>26h</td><td>VCO/&lt;Value/8&gt;</td></tr> <tr> <td>27h</td><td>Reserved.</td></tr> <tr> <td>28h</td><td>VCO/&lt;Value/8&gt;</td></tr> <tr> <td>29h</td><td>Reserved.</td></tr> <tr> <td>2Ah</td><td>VCO/&lt;Value/8&gt;</td></tr> <tr> <td>2Bh</td><td>Reserved.</td></tr> <tr> <td>2Ch</td><td>VCO/&lt;Value/8&gt;</td></tr> <tr> <td>3Fh-2Dh</td><td>Reserved.</td></tr> </table>	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.
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	<p><b>CpuFid[7:0]: core frequency ID.</b> Read-write. Reset: XXh. Specifies the core frequency multiplier. The core COF is a function of CpuFid and CpuDid, and defined by CoreCOF.</p> <p><b>ValidValues:</b></p> <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0Fh-00h</td><td>Reserved.</td></tr> <tr> <td>FFh-10h</td><td>&lt;Value&gt;*25</td></tr> </table>	Value	Description	0Fh-00h	Reserved.	FFh-10h	<Value>*25																																												
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.

\_lthree0\_core[3:0]\_thread[1:0]; MSRC001\_0073

Bits	Description
63:16	Reserved.
15:0	<b>CstateAddr: C-state address.</b> Read-write. Reset: 0000h. Specifies the IO addresses trapped by the core for C-state entry requests. A value of 0 in this field specifies that the core does not trap any IO addresses for C-state entry. Writing values greater than FFF8h into this field result in undefined behavior. All other values cause the core to trap IO addresses CstateAddr through CstateAddr + 7.

**MSRC001\_0074 [CPU Watchdog Timer] (Core::X86::Msr::CpuWdtCfg)**

Read-write. Reset: 0000\_0000\_0000\_0000h.

\_lthree0\_core[3:0]; MSRC001\_0074

Bits	Description																								
63:7	Reserved.																								
6:3	<p><b>CpuWdtCountSel: CPU watchdog timer count select.</b> Read-write. Reset: 0h. CpuWdtCountSel and CpuWdtTimeBase together specify the time period required for the WDT to expire. The time period is ((the multiplier specified by CpuWdtCountSel) * (the time base specified by CpuWdtTimeBase)). The actual timeout period may be anywhere from zero to one increment less than the values specified, due to non-deterministic behavior.</p> <p><b>ValidValues:</b></p> <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>4095</td></tr> <tr> <td>1h</td><td>2047</td></tr> <tr> <td>2h</td><td>1023</td></tr> <tr> <td>3h</td><td>511</td></tr> <tr> <td>4h</td><td>255</td></tr> <tr> <td>5h</td><td>127</td></tr> <tr> <td>6h</td><td>63</td></tr> <tr> <td>7h</td><td>31</td></tr> <tr> <td>8h</td><td>8191</td></tr> <tr> <td>9h</td><td>16383</td></tr> <tr> <td>Fh-Ah</td><td>Reserved</td></tr> </table>	Value	Description	0h	4095	1h	2047	2h	1023	3h	511	4h	255	5h	127	6h	63	7h	31	8h	8191	9h	16383	Fh-Ah	Reserved
Value	Description																								
0h	4095																								
1h	2047																								
2h	1023																								
3h	511																								
4h	255																								
5h	127																								
6h	63																								
7h	31																								
8h	8191																								
9h	16383																								
Fh-Ah	Reserved																								
2:1	<p><b>CpuWdtTimeBase: CPU watchdog timer time base.</b> Read-write. Reset: 0h. Specifies the time base for the timeout period specified in CpuWdtCountSel.</p> <p><b>ValidValues:</b></p> <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>1.31ms</td></tr> <tr> <td>1h</td><td>1.28us</td></tr> <tr> <td>3h-2h</td><td>Reserved</td></tr> </table>	Value	Description	0h	1.31ms	1h	1.28us	3h-2h	Reserved																
Value	Description																								
0h	1.31ms																								
1h	1.28us																								
3h-2h	Reserved																								
0	<b>CpuWdtEn: CPU watchdog timer enable.</b> Read-write. Reset: 0. Init: BIOS,1. 1=The WDT is enabled.																								

**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 2.1.11.1.5 [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.

\_lthree0\_core[3:0]\_thread[1:0]; MSRC001\_0111

Bits	Description
63:32	Reserved.

31:0	<b>SmmBase.</b> Reset: 0003_0000h.
	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 2.1.11.1 [System Management Mode (SMM)] and 2.1.5.3.1 [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:

$\text{CPUAddr}[47:17] \& \text{TSegMask}[47:17] == \text{TSegBase}[47:17] \& \text{TSegMask}[47:17]$ .

For example, if TSeg spans 256 KBs and starts at the 1-MB address. The Core::X86::Msr::SMMAddr[TSegBase[47:17]] would be set to 0010\_0000h and the Core::X86::Msr::SMMMask[TSegMask[47:17]] to FFFC\_0000h (with zeros filling in for bits[16:0]). This results in a TSeg range from 0010\_0000 to 0013\_FFFFh.

\_lthree0\_core[3:0]; MSRC001\_0112

Bits	Description
63:48	Reserved.
47:17	<b>TSegBase[47:17]: TSeg address range base.</b> Configurable. Reset: 0000_0000h. AccessType: (Core::X86::Msr::HWCR[SmmLock]) ? Read-only : Read-write.
16:0	Reserved.

**MSRC001\_0113 [SMM TSeg Mask] (Core::X86::Msr::SMMMask)**

Configurable. Reset: 0000\_0000\_0000\_0000h.

See 2.1.11.1 [System Management Mode (SMM)].

The ASeg address range is located at a fixed address from A0000h–BFFFFh. The TSeg range is located at a variable base (specified by Core::X86::Msr::SMMAddr[TSegBase[47:17]]) with a variable size (specified by Core::X86::Msr::SMMMask[TSegMask[47:17]]). These ranges provide a safe location for SMM code and data that is not readily accessible by non-SMM applications. The SMI handler can be located in one of these two ranges, or it can be located outside these ranges. These ranges must never overlap each other.

This register specifies how accesses to the ASeg and TSeg address ranges are controlled as follows:

- If [A,T]Valid == 1, then:
  - If in SMM, then:
    - If [A, T]Close == 0, then the accesses are directed to DRAM with memory type as specified in [A, T]MTypeDram.
    - If [A, T]Close == 1, then instruction accesses are directed to DRAM with memory type as specified in [A, T]MTypeDram and data accesses are directed at MMIO space and with attributes based on [A, T]MTypeIoWc.
  - If not in SMM, then the accesses are directed at MMIO space with attributes based on [A,T]MTypeIoWc.
- See 2.1.5.3.1.1 [Determining Memory Type].

\_lthree0\_core[3:0]; MSRC001\_0113

Bits	Description
63:48	Reserved.
47:17	<b>TSegMask[47:17]: TSeg address range mask.</b> Configurable. Reset: 0000_0000h. See Core::X86::Msr::SMMAddr. AccessType: (Core::X86::Msr::HWCR[SmmLock]) ? Read-only : Read-write.
16:15	Reserved.
14:12	<b>TMTypeDram: TSeg address range memory type.</b> 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.

	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	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	<b>AMTypeDram: ASeg Range Memory Type.</b> 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.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	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	<b>TMTypeIoWc: non-SMM TSeg address range memory type.</b> 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	<b>AMTypeIoWc: non-SMM ASeg address range memory type.</b> 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	<b>TClose: send TSeg address range data accesses to MMIO.</b> 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	<b>AClose: send ASeg address range data accesses to MMIO.</b> 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	<b>TValid: enable TSeg SMM address range.</b> Configurable. Reset: 0. 1=The TSeg address range SMM enabled. AccessType: (Core::X86::Msr::HWCR[SmmLock]) ? Read-only : Read-write.	
0	<b>AValid: enable ASeg SMM address range.</b> 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.

\_lthree0\_core[3:0]\_thread[1:0]; MSRC001\_0114

Bits	Description
63:5	Reserved.
4	<b>SvmeDisable: SVM disable.</b> 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	<b>Lock: SVM lock.</b> 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	<b>InterceptInit: intercept INIT.</b> 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.

\_lthree0\_core[3:0]\_thread[1:0]; MSRC001\_0115

Bits	Description
63:1	Reserved.
0	<b>IGNNE: current IGNNE state.</b> Read-write. Reset: 0. This bit controls the current state of the processor internal IGNNE signal.

**MSRC001\_0116 [SMM Control] (Core::X86::Msr::SMM\_CTL)**

Reset: 0000\_0000\_0000\_0000h.

The bits in this register are processed in the order of: SmmEnter, SmiCycle, SmmDismiss, RsmCycle and SmmExit. However, only the following combination of bits may be set in a single Write (all other combinations result in undefined behavior):

- SmmEnter and SmiCycle.
- SmmEnter and SmmDismiss.
- SmmEnter, SmiCycle and SmmDismiss.
- SmmExit and RsmCycle.

Software is responsible for ensuring that SmmEnter and SmmExit operations are properly matched and are not nested.

\_lthree0\_core[3:0]\_thread[1:0]; MSRC001\_0116

Bits	Description
63:5	Reserved.
4	<b>RsmCycle: send RSM special cycle.</b> Reset: 0. 1=Send a RSM special cycle. AccessType: Core::X86::Msr::HWCR[SmmLock] ? Error-on-read, Error-on-write : Write-only, Error-on-read.
3	<b>SmmExit: exit SMM.</b> Reset: 0. 1=Exit SMM. AccessType: Core::X86::Msr::HWCR[SmmLock] ? Error-on-read, Error-on-write : Write-only, Error-on-read.
2	<b>SmiCycle: send SMI special cycle.</b> Reset: 0. 1=Send a SMI special cycle. AccessType: Core::X86::Msr::HWCR[SmmLock] ? Error-on-read, Error-on-write : Write-only, Error-on-read.
1	<b>SmmEnter: enter SMM.</b> Reset: 0. 1=Enter SMM. AccessType: Core::X86::Msr::HWCR[SmmLock] ? Error-on-read, Error-on-write : Write-only, Error-on-read.
0	<b>SmmDismiss: clear SMI.</b> Reset: 0. 1=Clear the SMI pending flag. AccessType: Core::X86::Msr::HWCR[SmmLock] ? Error-on-read, Error-on-write : Write-only, Error-on-read.

**MSRC001\_0117 [Virtual Machine Host Save Physical Address] (Core::X86::Msr::VM\_HSAVE\_PA)**

Reset: 0000\_0000\_0000\_0000h.

\_lthree0\_core[3:0]\_thread[1:0]; MSRC001\_0117

Bits	Description
63:48	Reserved.
47:12	<b>VM_HSAVE_PA: physical address of host save area.</b> Read-write. Reset: 0_0000_0000h. This register contains the physical address of a 4-KB region where VMRUN saves host state and where vm-exit restores host state from. Writing this register causes a #GP if (FFFF_FFFF_Fh >= VM_HSAVE_PA >= FFFD_0000_0h) or if either the TSEG or ASEG regions overlap with the range defined by this register.
11:0	Reserved.



**MSRC001\_0118 [SVM Lock Key] (Core::X86::Msrb::SvmLockKey)**

Read-write. Reset: Fixed,0000\_0000\_0000\_0000h.

\_lthree0\_core[3:0]\_thread[1:0]; MSRC001\_0118

Bits	Description
63:0	<b>SvmLockKey: SVM lock key.</b> Read-write. Reset: Fixed,0000_0000_0000_0000h. Writes to this register when (Core::X86::Msrb::VM_CR[Lock] == 0) modify SvmLockKey. If ((Core::X86::Msrb::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::Msrb::VM_CR[Lock] = 0.

**MSRC001\_011A [Local SMI Status] (Core::X86::Msrb::LocalSmiStatus)**

Read-write. Reset: 0000\_0000\_0000\_0000h.

This register returns the same information that is returned in Core::X86::Smm::LocalSmiStatus portion of the SMM save state. The information in this register is only updated when Core::X86::Msrb::SMM\_CTL[SmmDismiss] is set by software.

\_lthree0\_core[3:0]\_thread[1:0]; MSRC001\_011A

Bits	Description
63:32	Reserved.
31:0	<b>LocalSmiStatus.</b> Read-write. Reset: 0000_0000h. See Core::X86::Smm::LocalSmiStatus.

**MSRC001\_011B [AVIC Doorbell] (Core::X86::Msrb::AvicDoorbell)**

Reset: 0000\_0000\_0000\_0000h.

The ApicId is a physical APIC Id; not valid for logical APIC ID.

See Core::X86::Cpuid::SvmRevFeatIdEdx[AVIC].

\_lthree0\_core[3:0]\_thread[1:0]; MSRC001\_011B

Bits	Description
63:8	Reserved.
7:0	<b>ApicId: APIC ID [7:0].</b> Write-only,Error-on-read. Reset: 00h.

**MSRC001\_011E [VM Page Flush] (Core::X86::Msrb::VMPAGE\_FLUSH)**

Writes to this MSR cause 4 KBs of encrypted, guest-tagged data to be flushed from caches if present.

\_lthree0\_core[3:0]\_thread[1:0]; MSRC001\_011E

Bits	Description
63:12	<b>VirtualAddr.</b> Reset: X_XXXX_XXXX_XXXXh. Guest physical address of page to flush. AccessType: Core::X86::Msrb::SYS_CFG[SME] ? Write-only,Error-on-read : Error-on-read,Error-on-write.
11:0	<b>ASID.</b> Reset: XXXh. ASID to use for flush. Writing reserved values generates #GP. AccessType: Core::X86::Msrb::SYS_CFG[SME] ? Write-only,Error-on-read : Error-on-read,Error-on-write.

**MSRC001\_0130 [Guest Host Communication Block] (Core::X86::Msrb::GHCB)**

Read-write. Reset: 0000\_0000\_0000\_0000h.

If Core::X86::Msrb::GHCB is accessed in hypervisor mode, #GP is generated.

\_lthree0\_core[3:0]\_thread[1:0]; MSRC001\_0130

Bits	Description
63:0	<b>GHCBPA.</b> Read-write. Reset: 0000_0000_0000_0000h. Guest physical address of GHCB.

**MSRC001\_0131 [SEV Status] (Core::X86::Msrb::SEV\_Status)**

Read,Error-on-write. Reset: 0000\_0000\_0000\_0000h.

\_lthree0\_core[3:0]\_thread[1:0]; MSRC001\_0131

Bits	Description
63:2	Reserved.
1	<b>SevEsEnabled.</b> Read,Error-on-write. Reset: 0. 1=The guest was launched with the Sev-ES feature enabled in VMCB offset 90h.
0	<b>SevEnabled.</b> Read,Error-on-write. Reset: 0. 1=The guest was launched with SEV feature enabled in VMCB



	offset 90h.	
<b>MSRC001_0140 [OS Visible Work-around Length] (Core::X86::Msr::OSVW_ID_Length)</b>		
Read-write. Reset: 0000_0000_0000_0000h.		
_lthree0_core[3:0]_thread[1:0]; MSRC001_0140		
<b>Bits</b>	<b>Description</b>	
63:16	Reserved.	
15:0	<b>OSVWIDLength: OS visible work-around ID length.</b> Read-write. Reset: 0000h. See the Revision Guide for the definition of this field; see 1.2 [Reference Documents].	
<b>MSRC001_0141 [OS Visible Work-around Status] (Core::X86::Msr::OSVW_Status)</b>		
Read-write. Reset: 0000_0000_0000_0000h.		
_lthree0_core[3:0]_thread[1:0]; MSRC001_0141		
<b>Bits</b>	<b>Description</b>	
63:0	<b>OsvwStatusBits: OS visible work-around status bits.</b> Read-write. Reset: 0000_0000_0000_0000h. See the Revision Guide for the definition of this field; see 1.2 [Reference Documents].	
<b>MSRC001_020[0...A] [Performance Event Select [5:0]] (Core::X86::Msr::PERF_CTL)</b>		
Read-write. Reset: 0000_0000_0000_0000h.		
See 2.1.14 [Performance Monitor Counters]. Core::X86::Msr::PERF_LEGACY_CTL is an alias of MSRC001_020[6,4,2,0].		
_lthree0_core[3:0]_thread[1:0]_n0; MSRC001_0200		
_lthree0_core[3:0]_thread[1:0]_n1; MSRC001_0202		
_lthree0_core[3:0]_thread[1:0]_n2; MSRC001_0204		
_lthree0_core[3:0]_thread[1:0]_n3; MSRC001_0206		
_lthree0_core[3:0]_thread[1:0]_n4; MSRC001_0208		
_lthree0_core[3:0]_thread[1:0]_n5; MSRC001_020A		
<b>Bits</b>	<b>Description</b>	
63:42	Reserved.	
41:40	<b>HostGuestOnly: count only host/guest events.</b> Read-write. Reset: 0h.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	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	<b>EventSelect[11:8]: performance event select.</b> Read-write. Reset: 0h.	
31:24	<b>CntMask: counter mask.</b> Read-write. Reset: 00h. Controls the number of events counted per clock cycle.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	00h	The corresponding PERF_CTR[5:0] register increments by the number of events occurring in a clock cycle. See 2.1.14.2 [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	<b>Inv: invert counter mask.</b> Read-write. Reset: 0. See CntMask.	
22	<b>En: enable performance counter.</b> Read-write. Reset: 0. 1=Performance event counter is enabled.	

21	Reserved.
20	<b>Int: enable APIC interrupt.</b> Read-write. Reset: 0. 1=APIC performance counter LVT interrupt is enabled to generate an interrupt via Core::X86::Apic::PerformanceCounterLvtEntry when the performance counter overflows.
19	Reserved.
18	<b>Edge: edge detect.</b> Read-write. Reset: 0. 0=Level detect. 1=Zero-to-one Edge detect. Read-write. The edge count mode increments the counter when a transition happens on the monitored event. If the event selected is changed without disabling the counter, an extra edge is falsely detected when the first event is a static 0 and the second event is a static one. To avoid this false edge detection, disable the counter when changing the event and then enable the counter with a second MSR write.
17:16	<b>OsUserMode: OS and user mode.</b> Read-write. Reset: 0h.
<b>ValidValues:</b>	
<b>Value</b>	<b>Description</b>
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	<b>UnitMask: event qualification.</b> 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	<b>EventSelect[7:0]: event select.</b> 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 2.1.14.3 [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 2.1.14.2 [Large Increment per Cycle Events].

Core::X86::Msr::PERF\_CTL. Core::X86::Msr::PERF\_LEGACY\_CTR is an alias of MSRC001\_020[7,5,3,1]. Also can be read via x86 instructions RDPMSR ECX = [05:00].

\_lthree0\_core[3:0]\_thread[1:0]\_n0; MSRC001\_0201  
\_lthree0\_core[3:0]\_thread[1:0]\_n1; MSRC001\_0203  
\_lthree0\_core[3:0]\_thread[1:0]\_n2; MSRC001\_0205  
\_lthree0\_core[3:0]\_thread[1:0]\_n3; MSRC001\_0207  
\_lthree0\_core[3:0]\_thread[1:0]\_n4; MSRC001\_0209  
\_lthree0\_core[3:0]\_thread[1:0]\_n5; MSRC001\_020B

Bits	Description
63:48	Reserved.
47:0	<b>CTR: performance counter value.</b> Read-write, Volatile. Reset: 0000_0000_0000h.

#### MSRC001\_023[0...A] [L3 Performance Event Select [5:0]] (Core::X86::Msr::ChL3PmcCfg)

Read-write. Reset: 0000\_0000\_0000\_0000h.

See 2.1.14.4 [L3 Cache Performance Monitor Counters].

\_lthree0\_n0; MSRC001\_0230  
\_lthree0\_n1; MSRC001\_0232  
\_lthree0\_n2; MSRC001\_0234  
\_lthree0\_n3; MSRC001\_0236  
\_lthree0\_n4; MSRC001\_0238

_lthree0_n5; MSRC001_023A	
Bits	Description
63:56	<b>ThreadMask.</b> Read-write. Reset: 00h. Controls which of the up to 8 threads in the complex are being counted (Dependent upon number of cores). In non-SMT mode, thread 0 must be selected. One or more threads must be selected unless otherwise specified by the specific L3PMC event.
<b>ValidValues:</b>	
Bit	Description
[0]	Core 0 Thread 0 mask.
[1]	Core 0 Thread 1 mask.
[2]	Core 1 Thread 0 mask.
[3]	Core 1 Thread 1 mask.
[4]	Core 2 Thread 0 mask.
[5]	Core 2 Thread 1 mask.
[6]	Core 3 Thread 0 mask.
[7]	Core 3 Thread 1 mask.
55:52	Reserved.
51:48	<b>SliceMask.</b> Read-write. Reset: 0h. Controls which L3 slices are counting this event. One or more Slices must be selected unless otherwise specified by the specific L3PMC event.
<b>ValidValues:</b>	
Bit	Description
[0]	L3 Slice 0 mask.
[1]	L3 Slice 1 mask.
[2]	L3 Slice 2 mask.
[3]	L3 Slice 3 mask.
47:23	Reserved.
22	<b>Enable: Enable L3 performance counter.</b> Read-write. Reset: 0. 1=Enable.
21:16	Reserved.
15:8	<b>UnitMask: event qualification.</b> 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	<b>EventSel: event select.</b> Read-write. Reset: 00h.

**MSRC001\_023[1...B] [L3 Performance Event Counter [5:0]] (Core::X86::Msr::ChL3Pmc)**

Reset: 0000\_0000\_0000\_0000h.

Also can be read via x86 instructions RDPMC ECX = [0F:0A].

\_lthree0\_n0; MSRC001\_0231

\_lthree0\_n1; MSRC001\_0233

\_lthree0\_n2; MSRC001\_0235

\_lthree0\_n3; MSRC001\_0237

\_lthree0\_n4; MSRC001\_0239

\_lthree0\_n5; MSRC001\_023B

Bits	Description
63:49	Reserved.
48	<b>Overflow.</b> Read-write. Reset: 0.
47:32	<b>CountHi.</b> Read-write, Volatile. Reset: 0000h.
31:0	<b>CountLo.</b> Read-write, Volatile. Reset: 0000_0000h.

**MSRC001\_024[0...6] [Data Fabric Performance Event Select [3:0]] (Core::X86::Msr::DF\_PERF\_CTL)**

Read-write. Reset: 0000\_0000\_0000\_0000h.

See 2.1.14 [Performance Monitor Counters].

\_n0; MSRC001\_0240

\_n1; MSRC001\_0242

\_n2; MSRC001\_0244

\_n3; MSRC001\_0246

Bits	Description
63:61	Reserved.
60:59	<b>EventSelect[13:12]: performance event select.</b> Read-write. Reset: 0h.
58:36	Reserved.
35:32	<b>EventSelect[11:8]: performance event select.</b> Read-write. Reset: 0h. See EventSelect[7:0].
31:23	Reserved.
22	<b>En: enable performance counter.</b> Read-write. Reset: 0. 1=Performance event counter is enabled.
21:16	Reserved.
15:8	<b>UnitMask: event qualification.</b> 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	<b>EventSelect[7:0]: event select.</b> Read-write. Reset: 00h. This field, along with EventSelect[13:12] and EventSelect[11:8] above, combine to form the 14-bit event select field, EventSelect[13:0]. EventSelect specifies the event or event duration in a processor unit to be counted by the corresponding DF_PERF_CTR[3:0] register. Some events are reserved; when a reserved event is selected, the results are undefined.

**MSRC001\_024[1...7] [Data Fabric Performance Event Counter [3:0]] (Core::X86::Msr::DF\_PERF\_CTR)**

See Core::X86::Msr::DF\_PERF\_CTL. Also can be read via x86 instructions RDPMC ECX = [09:06].

\_n0; MSRC001\_0241

\_n1; MSRC001\_0243

\_n2; MSRC001\_0245

\_n3; MSRC001\_0247

Bits	Description
63:48	Reserved.
47:0	<b>CTR[47:0]: performance counter value[47:0].</b> Read-write, Volatile. Reset: 0000_0000_0000h. The current value of the event counter.

**MSRC001\_0299 [RAPL Power Unit] (Core::X86::Msr::RAPL\_PWR\_UNIT)**

Read-only, Volatile. Reset: 0000\_0000\_000A\_1003h.

\_lthree0; MSRC001\_0299

Bits	Description				
63:20	Reserved.				
19:16	<b>TU: Time Units in seconds.</b> Read-only, Volatile. Reset: Ah. Time information (in Seconds) is based on the multiplier, $1/2^{\text{TU}}$ ; where TU is an unsigned integer. Default value is 1010b, indicating time unit is in 976 microseconds increment. <b>ValidValues:</b>				
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>Fh-0h</td><td><math>1/2^{\text{&lt;Value&gt;}}</math> Seconds</td></tr> </table>	Value	Description	Fh-0h	$1/2^{\text{<Value>}}$ Seconds
Value	Description				
Fh-0h	$1/2^{\text{<Value>}}$ Seconds				
15:13	Reserved.				
12:8	<b>ESU: Energy Status Units.</b> Read-only, Volatile. Reset: 10h. Energy information (in Joules) is based on the multiplier, $1/2^{\text{ESU}}$ ; where ESU is an unsigned integer. Default value is 10000b, indicating energy status unit is in 15.3 micro-Joules increment. <b>ValidValues:</b>				
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>1Fh-00h</td><td><math>1/2^{\text{&lt;Value&gt;}}</math> Joules</td></tr> </table>	Value	Description	1Fh-00h	$1/2^{\text{<Value>}}$ Joules
Value	Description				
1Fh-00h	$1/2^{\text{<Value>}}$ Joules				
7:4	Reserved.				

3:0	<b>PU: Power Units.</b> Read-only, Volatile. Reset: 3h. Power information (in Watts) is based on the multiplier, $1/2^{\text{PU}}$ ; where PU is an unsigned integer. Default value is 0011b, indicating power unit is in 1/8 Watts increment.	
	<b>Valid Values:</b>	
	<b>Value</b>	<b>Description</b>
	Fh-0h	$1/2^{\text{Value}}$ Watts

**MSRC001\_029A [Core Energy Status] (Core::X86::Msr::CORE\_ENERGY\_STAT)**

Read-only, Volatile. Reset: 0000\_0000\_0000\_0000h.

\_lthree0\_core[3:0]; MSRC001\_029A

Bits	Description
63:32	Reserved.
31:0	<b>TotalEnergyConsumed.</b> Read-only, Volatile. Reset: 0000_0000h.

**MSRC001\_029B [Package Energy Status] (Core::X86::Msr::PKG\_ENERGY\_STAT)**

Read-only, Volatile. Reset: 0000\_0000\_0000\_0000h.

MSRC001\_029B

Bits	Description
63:32	Reserved.
31:0	<b>TotalEnergyConsumed.</b> Read-only, Volatile. Reset: 0000_0000h.

**2.1.13.4 MSRs - MSRC001\_1xxx**

See 1.4.3 [Register Mnemonics] for a description of the register naming convention. MSRs are accessed through x86 WRMSR and RDMSR instructions.

**MSRC001\_1002 [CPUID Features for CPUID Fn00000007\_E[A,B]X] (Core::X86::Msr::CPUID\_7\_Features)**

Read-write.

Core::X86::Msr::CPUID\_7\_Features[63:32] provides control over values read from Core::X86::Cpuid::StructExtFeatIdEax0; Core::X86::Msr::CPUID\_7\_Features[31:0] provides control over values read from Core::X86::Cpuid::StructExtFeatIdEbx0.

\_lthree0\_core[3:0]\_thread[1:0]; MSRC001\_1002

Bits	Description
63:30	Reserved.
29	<b>SHA.</b> Read-write. Reset: Core::X86::Cpuid::StructExtFeatIdEbx0[SHA].
28:24	Reserved.
23	<b>CLFSHOPT.</b> Read-write. Reset: Core::X86::Cpuid::StructExtFeatIdEbx0[CLFSHOPT].
22:21	Reserved.
20	<b>SMAP.</b> Read-write. Reset: Core::X86::Cpuid::StructExtFeatIdEbx0[SMAP].
19	<b>ADX.</b> Read-write. Reset: Core::X86::Cpuid::StructExtFeatIdEbx0[ADX].
18	<b>RDSEED.</b> Read-write. Reset: Core::X86::Cpuid::StructExtFeatIdEbx0[RDSEED].
17:9	Reserved.
8	<b>BMI2.</b> Read-write. Reset: Core::X86::Cpuid::StructExtFeatIdEbx0[BMI2].
7	<b>SMEP.</b> Read-write. Reset: Core::X86::Cpuid::StructExtFeatIdEbx0[SMEP].
6	Reserved.
5	<b>AVX2.</b> Read-write. Reset: Core::X86::Cpuid::StructExtFeatIdEbx0[AVX2].
4	Reserved.
3	<b>BMI1.</b> Read-write. Reset: Core::X86::Cpuid::StructExtFeatIdEbx0[BMI1].
2:1	Reserved.

0	<b>FSGSBASE.</b> Read-write. Reset: Core::X86::Cpuid::StructExtFeatIdEbx0[FSGSBASE].
---	--

**MSRC001\_1003 [Thermal and Power Management CPUID Features] (Core::X86::Msr::CPUID\_PWR\_THERM)**

Read-write.

Core::X86::Msr::CPUID\_PWR\_THERM provides control over values read from

Core::X86::Cpuid::ThermalPwrMgmtEcX.

\_lthree0\_core[3:0]\_thread[1:0]; MSRC001\_1003

Bits	Description
63:1	Reserved.
0	<b>EffFreq.</b> Read-write. Reset: Core::X86::Cpuid::ThermalPwrMgmtEcX[EffFreq].

**MSRC001\_1004 [CUID 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.

\_lthree0\_core[3:0]\_thread[1:0]; MSRC001\_1004

Bits	Description
63	Reserved.
62	<b>RDRAND.</b> Read-write. Reset: Core::X86::Cpuid::FeatureIdEcX[RDRAND].
61	<b>F16C.</b> Read-write. Reset: Core::X86::Cpuid::FeatureIdEcX[F16C].
60	<b>AVX.</b> Read-write. Reset: Core::X86::Cpuid::FeatureIdEcX[AVX].
59	<b>OSXSAVE.</b> Read-write. Reset: Core::X86::Cpuid::FeatureIdEcX[OSXSAVE]. Modifies Core::X86::Cpuid::FeatureIdEcX[OSXSAVE] only if CR4[OSXSAVE].
58	<b>XSAVE.</b> Read-write. Reset: Core::X86::Cpuid::FeatureIdEcX[XSAVE].
57	<b>AES.</b> Read-write. Reset: Core::X86::Cpuid::FeatureIdEcX[AES]. Modifies Core::X86::Cpuid::FeatureIdEcX[AES] only if the reset value is 1.
56	Reserved.
55	<b>POPCNT.</b> Read-write. Reset: Core::X86::Cpuid::FeatureIdEcX[POPCNT].
54	<b>MOVBE.</b> Read-write. Reset: Core::X86::Cpuid::FeatureIdEcX[MOVBE].
53	Reserved.
52	<b>SSE42.</b> Read-write. Reset: Core::X86::Cpuid::FeatureIdEcX[SSE42].
51	<b>SSE41.</b> Read-write. Reset: Core::X86::Cpuid::FeatureIdEcX[SSE41].
50:46	Reserved.
45	<b>CMPXCHG16B.</b> Read-write. Reset: Core::X86::Cpuid::FeatureIdEcX[CMPXCHG16B].
44	<b>FMA.</b> Read-write. Reset: Core::X86::Cpuid::FeatureIdEcX[FMA].
43:42	Reserved.
41	<b>SSSE3.</b> Read-write. Reset: Core::X86::Cpuid::FeatureIdEcX[SSSE3].
40:36	Reserved.
35	<b>Monitor.</b> Read-write. Reset: Core::X86::Cpuid::FeatureIdEcX[Monitor]. Modifies Core::X86::Cpuid::FeatureIdEcX[Monitor] only if ~Core::X86::Msr::HWCR[MonMwaitDis].
34	Reserved.
33	<b>PCLMULQDQ.</b> Read-write. Reset: Core::X86::Cpuid::FeatureIdEcX[PCLMULQDQ]. Modifies Core::X86::Cpuid::FeatureIdEcX[PCLMULQDQ] only if the reset value is 1.
32	<b>SSE3.</b> Read-write. Reset: Core::X86::Cpuid::FeatureIdEcX[SSE3].
31:29	Reserved.
28	<b>HTT.</b> Read-write. Reset: Core::X86::Cpuid::FeatureIdEdX[HTT].
27	Reserved.
26	<b>SSE2.</b> Read-write. Reset: Core::X86::Cpuid::FeatureIdEdX[SSE2].
25	<b>SSE.</b> Read-write. Reset: Core::X86::Cpuid::FeatureIdEdX[SSE].
24	<b>FXSR.</b> Read-write. Reset: Core::X86::Cpuid::FeatureIdEdX[FXSR].



23	<b>MMX: MMX instructions.</b> Read-write. Reset: Core::X86::Cpuid::FeatureIdEdx[MMX].
22:20	Reserved.
19	<b>CLFSH.</b> Read-write. Reset: Core::X86::Cpuid::FeatureIdEdx[CLFSH].
18	Reserved.
17	<b>PSE36.</b> Read-write. Reset: Core::X86::Cpuid::FeatureIdEdx[PSE36].
16	<b>PAT.</b> Read-write. Reset: Core::X86::Cpuid::FeatureIdEdx[PAT].
15	<b>CMOV.</b> Read-write. Reset: Core::X86::Cpuid::FeatureIdEdx[CMOV].
14	<b>MCA.</b> Read-write. Reset: Core::X86::Cpuid::FeatureIdEdx[MCA].
13	<b>PGE.</b> Read-write. Reset: Core::X86::Cpuid::FeatureIdEdx[PGE].
12	<b>MTRR.</b> Read-write. Reset: Core::X86::Cpuid::FeatureIdEdx[MTRR].
11	<b>SysEnterSysExit.</b> Read-write. Reset: Core::X86::Cpuid::FeatureIdEdx[SysEnterSysExit].
10	Reserved.
9	<b>APIC.</b> Read-write. Reset: Core::X86::Cpuid::FeatureIdEdx[APIC]. Modifies Core::X86::Cpuid::FeatureIdEdx[APIC] only if Core::X86::Msrr::APIC_BAR[ApicEn].
8	<b>CMPXCHG8B.</b> Read-write. Reset: Core::X86::Cpuid::FeatureIdEdx[CMXCHG8B].
7	<b>MCE.</b> Read-write. Reset: Core::X86::Cpuid::FeatureIdEdx[MCE].
6	<b>PAE.</b> Read-write. Reset: Core::X86::Cpuid::FeatureIdEdx[PAE].
5	<b>MSR.</b> Read-write. Reset: Core::X86::Cpuid::FeatureIdEdx[MSR].
4	<b>TSC.</b> Read-write. Reset: Core::X86::Cpuid::FeatureIdEdx[TSC].
3	<b>PSE.</b> Read-write. Reset: Core::X86::Cpuid::FeatureIdEdx[PSE].
2	<b>DE.</b> Read-write. Reset: Core::X86::Cpuid::FeatureIdEdx[DE].
1	<b>VME.</b> Read-write. Reset: Core::X86::Cpuid::FeatureIdEdx[VME].
0	<b>FPU.</b> Read-write. Reset: Core::X86::Cpuid::FeatureIdEdx[FPU].

#### MSRC001\_1005 [CPUID Features for CPUID Fn80000001\_E[C,D]X] (Core::X86::Msrr::CPUID\_ExtFeatures)

Read-write.

Core::X86::Msrr::CPUID\_ExtFeatures[63:32] provides control over values read from

Core::X86::Cpuid::FeatureExtIdEcX; Core::X86::Msrr::CPUID\_ExtFeatures[31:0] provides control over values read from Core::X86::Cpuid::FeatureExtIdEdx.

\_lthree0\_core[3:0]\_thread[1:0]; MSRC001\_1005

Bits	Description
63:62	Reserved.
61	<b>MwaitExtended.</b> Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEcX[MwaitExtended].
60	<b>PerfCtrExtLLC.</b> Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEcX[PerfCtrExtLLC].
59	<b>PerfTsc.</b> Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEcX[PerfTsc].
58	<b>DataBreakpointExtension.</b> Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEcX[DataBreakpointExtension].
57	Reserved.
56	<b>PerfCtrExtDF.</b> Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEcX[PerfCtrExtDF].
55	<b>PerfCtrExtCore.</b> Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEcX[PerfCtrExtCore].
54	<b>TopologyExtensions.</b> Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEcX[TopologyExtensions].
53:50	Reserved.
49	<b>TCE.</b> Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEcX[TCE].
48	<b>FMA4.</b> Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEcX[FMA4]. Init: 0.
47	<b>LWP.</b> Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEcX[LWP].
46	Reserved.
45	<b>WDT.</b> Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEcX[WDT].
44	<b>SKINIT.</b> Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEcX[SKINIT].
43	<b>XOP.</b> Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEcX[XOP].
42	<b>IBS.</b> Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEcX[IBS]. Init: BIOS,0. To enable the IBS feature, use



	BIOS setup option.
41	<b>OSVW</b> . Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEcxC[OSVW].
40	<b>ThreeDNowPrefetch</b> . Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEcxC[ThreeDNowPrefetch].
39	<b>MisAlignSse</b> . Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEcxC[MisAlignSse].
38	<b>SSE4A</b> . Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEcxC[SSE4A].
37	<b>ABM</b> . Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEcxC[ABM].
36	<b>AltMovCr8</b> . Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEcxC[AltMovCr8].
35	<b>ExtApicSpace</b> . Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEcxC[ExtApicSpace].
34	<b>SVM</b> . Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEcxC[SVM].
33	<b>CmpLegacy</b> . Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEcxC[CmpLegacy].
32	<b>LahfSahf</b> . Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEcxC[LahfSahf].
31	<b>ThreeDNow: 3DNow! instructions</b> . Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEdxC[ThreeDNow].
30	<b>ThreeDNowExt</b> . Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEdxC[ThreeDNowExt].
29	<b>LM</b> . Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEdxC[LM].
28	Reserved.
27	<b>RDTSCP</b> . Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEdxC[RDTSCP].
26	<b>Page1GB</b> . Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEdxC[Page1GB].
25	<b>FFXSR</b> . Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEdxC[FFXSR].
24	<b>FXSR</b> . Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEdxC[FXSR].
23	<b>MMX: MMX instructions</b> . Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEdxC[MMX].
22	<b>MmxExt</b> . Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEdxC[MmxExt].
21	Reserved.
20	<b>NX</b> . Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEdxC[NX].
19:18	Reserved.
17	<b>PSE36</b> . Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEdxC[PSE36].
16	<b>PAT</b> . Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEdxC[PAT].
15	<b>CMOV</b> . Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEdxC[CMOV].
14	<b>MCA</b> . Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEdxC[MCA].
13	<b>PGE</b> . Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEdxC[PGE].
12	<b>MTRR</b> . Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEdxC[MTRR].
11	<b>SysCallSysRet</b> . Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEdxC[SysCallSysRet].
10	Reserved.
9	<b>APIC</b> . Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEdxC[APIC].
8	<b>CMPXCHG8B</b> . Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEdxC[CMXCHG8B].
7	<b>MCE</b> . Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEdxC[MCE].
6	<b>PAE</b> . Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEdxC[PAE].
5	<b>MSR</b> . Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEdxC[MSR].
4	<b>TSC</b> . Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEdxC[TSC].
3	<b>PSE</b> . Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEdxC[PSE].
2	<b>DE</b> . Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEdxC[DE].
1	<b>VME</b> . Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEdxC[VME].
0	<b>FPU</b> . Read-write. Reset: Core::X86::Cpuid::FeatureExtIdEdxC[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::FeatureExtIdEcxC[DataBreakpointExtension].

\_lthree0\_core[3:0]\_thread[1:0]; MSRC001\_1019

Bits	Description
63:32	Reserved.

31:0	<b>AddrMask: mask for DR linear address data breakpoint DR1.</b> 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::FeatureExtIdEc[DataBreakpointExtension].

\_lthree0\_core[3:0]\_thread[1:0]; MSRC001\_101A

Bits	Description
63:32	Reserved.
31:0	<b>AddrMask: mask for DR linear address data breakpoint DR2.</b> 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::FeatureExtIdEc[DataBreakpointExtension].

\_lthree0\_core[3:0]\_thread[1:0]; MSRC001\_101B

Bits	Description
63:32	Reserved.
31:0	<b>AddrMask: mask for DR linear address data breakpoint DR3.</b> 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\_1023 [Table Walker Configuration] (Core::X86::Msr::TW\_CFG)**

Read-write. Reset: 0000\_0000\_0000\_0000h.

\_lthree0\_core[3:0]; MSRC001\_1023

Bits	Description
63:50	Reserved.
49	<b>TwCfgCombineCr0Cd: combine CR0_CD for both threads of a core.</b> Read-write. Reset: 0. Init: BIOS, 1. 1=The host Cr0_Cd values from the two threads are OR'd together and used by both threads.
48:0	Reserved.

**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::FeatureExtIdEc[DataBreakpointExtension]. See Core::X86::Msr::DR1\_ADDR\_MASK.

\_lthree0\_core[3:0]\_thread[1:0]; MSRC001\_1027

Bits	Description
63:32	Reserved.
31:0	<b>DR0: mask for DR0 linear address data breakpoint.</b> 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 2.1.15 [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::Msrr::IBS\_FETCH\_LINADDR and Core::X86::Msrr::IBS\_FETCH\_PHYSADDR).
- An interrupt is generated as specified by Core::X86::Msrr::IBS\_CTL. The interrupt service routine associated with this interrupt is responsible for saving the performance information stored in IBS execution registers.

\_lthree0\_core[3:0]\_thread[1:0]; MSRC001\_1030

Bits	Description										
63:59	Reserved.										
58	<b>IbsFetchL2Miss: L2 cache miss for the sampled fetch.</b> Read-only, Volatile. Reset: 0. 1=The instruction fetch missed in the L2 Cache. Qualified by (IbsFetchComp == 1).										
57	<b>IbsRandEn: random instruction fetch tagging enable.</b> 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	<b>IbsL2TlbMiss: instruction cache L2TLB miss.</b> Read-only, Volatile. Reset: 0. 1=The instruction fetch missed in the L2 TLB.										
55	<b>IbsL1TlbMiss: instruction cache L1TLB miss.</b> Read-only, Volatile. Reset: 0. 1=The instruction fetch missed in the L1 TLB.										
54:53	<b>IbsL1TlbPgSz: instruction cache L1TLB page size.</b> Read-only, Volatile. Reset: 0h. Indicates the page size of the translation in the L1 TLB. This field is only valid if IbsPhyAddrValid == 1.										
<b>Valid Values:</b>											
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>4 KB</td></tr> <tr> <td>1h</td><td>2 MB</td></tr> <tr> <td>2h</td><td>1 GB</td></tr> <tr> <td>3h</td><td>Reserved.</td></tr> </table>	Value	Description	0h	4 KB	1h	2 MB	2h	1 GB	3h	Reserved.
Value	Description										
0h	4 KB										
1h	2 MB										
2h	1 GB										
3h	Reserved.										
52	<b>IbsPhyAddrValid: instruction fetch physical address valid.</b> Read-only, Volatile. Reset: 0. 1=The physical address in Core::X86::Msrr::IBS_FETCH_PHYSADDR and the IbsL1TlbPgSz field are valid for the instruction fetch.										
51	<b>IbsIcMiss: instruction cache miss.</b> Read-only, Volatile. Reset: 0. 1=The instruction fetch missed in the instruction cache.										
50	<b>IbsFetchComp: instruction fetch complete.</b> Read-only, Volatile. Reset: 0. 1=The instruction fetch completed and the data is available for use by the instruction decoder.										
49	<b>IbsFetchVal: instruction fetch valid.</b> Read-only, Volatile. Reset: 0. 1=New instruction fetch data available. When this bit is set, the fetch counter stops counting and an interrupt is generated as specified by Core::X86::Msrr::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	<b>IbsFetchEn: instruction fetch enable.</b> Read-write. Reset: 0. 1=Instruction fetch sampling is enabled.										
47:32	<b>IbsFetchLat: instruction fetch latency.</b> Read-only, Volatile. Reset: 0000h. Indicates the number of clock cycles from when the instruction fetch was initiated to when the data was delivered to the core. If the instruction fetch is abandoned before the fetch completes, this field returns the number of clock cycles from when the instruction										

	fetch was initiated to when the fetch was abandoned.
31:16	<b>IbsFetchCnt[19:4]</b> . 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	<b>IbsFetchMaxCnt[19:4]</b> . 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.

\_lthree0\_core[3:0]\_thread[1:0]; MSRC001\_1031

Bits	Description
63:0	<b>IbsFetchLinAd: instruction fetch linear address</b> . 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)**

\_lthree0\_core[3:0]\_thread[1:0]; MSRC001\_1032

Bits	Description
63:48	Reserved.
47:0	<b>IbsFetchPhysAd: instruction fetch physical address</b> . Read-write, Volatile. Reset: 0000_0000_0000_0000h. Provides the physical address for the tagged instruction fetch. The lower 12 bits are not modified by address translation, so they are always the same as the linear address. This field contains valid data only if Core::X86::Msr::IBS_FETCH_CTL[IbsPhyAddrValid] is asserted.

**MSRC001\_1033 [IBS Execution Control] (Core::X86::Msr::IBS\_OP\_CTL)**

Reset: 0000\_0000\_0000\_0000h.

See 2.1.15 [Instruction Based Sampling (IBS)].

The IBS execution sampling engine is described as follows for IbsOpCntCtl == 1. If IbsOpCntCtl == 1n then references to "periodic op counter" mean "periodic cycle counter".

- The periodic op counter is an internal 27-bit counter:
  - It is set to IbsOpCurCnt[26:0] when IbsOpEn is changed from 0 to 1.
  - It increments every dispatched 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 micro-op counter [26:0].
- When the periodic micro-op counter reaches IbsOpMaxCnt:
  - The next dispatched micro-op is tagged if IbsOpCntCtl == 1. A valid op in the next dispatched line is tagged if IbsOpCntCtl == 0. See IbsOpCntCtl.
  - The periodic micro-op counter [26:7] = 0; [6:0] is randomized by hardware.
- The periodic micro-op counter is not modified when a tagged micro-op is flushed.
- When a tagged micro-op is retired:
  - IbsOpVal is set to 1.
    - Drivers can't assume that IbsOpCurCnt == 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.

\_lthree0\_core[3:0]\_thread[1:0]; MSRC001\_1033

Bits	Description
63:59	Reserved.
58:32	<b>IbsOpCurCnt[26:0]: periodic op counter current count</b> . Read-write, Volatile. Reset: 000_0000h. Returns the

	current value of the periodic op counter.						
31:27	Reserved.						
26:20	<b>IbsOpMaxCnt[26:20]: periodic op counter maximum count.</b> Read-write. Reset: 00h. See IbsOpMaxCnt[19:4].						
19	<b>IbsOpCntCtl: periodic op counter count control.</b> 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 Micro-Ops; when a roll-over occurs, the counter is preloaded with a pseudorandom 7-bit value between 1 and 127.						
18	<b>IbsOpVal: micro-op sample valid.</b> 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	<b>IbsOpEn: micro-op sampling enable.</b> Read-write. Reset: 0. 1=Instruction execution sampling enabled.						
16	Reserved.						
15:0	<b>IbsOpMaxCnt[19:4]: periodic op counter maximum count.</b> 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. <b>Valid Values:</b>						
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0008h-0000h</td><td>Reserved.</td></tr> <tr> <td>FFFFh-0009h</td><td>&lt;Value&gt; *16 Ops.</td></tr> </table>	Value	Description	0008h-0000h	Reserved.	FFFFh-0009h	<Value> *16 Ops.
Value	Description						
0008h-0000h	Reserved.						
FFFFh-0009h	<Value> *16 Ops.						

**MSRC001\_1034 [IBS Op Logical Address] (Core::X86::Msr::IBS\_OP\_RIP)**

Read-write, Volatile. Reset: 0000\_0000\_0000\_0000h.

\_lthree0\_core[3:0]\_thread[1:0]; MSRC001\_1034

Bits	Description
63:0	<b>IbsOpRip: micro-op linear address.</b> Read-write, Volatile. Reset: 0000_0000_0000_0000h. Linear address in canonical form for the instruction that contains the tagged micro-op.

**MSRC001\_1035 [IBS Op Data] (Core::X86::Msr::IBS\_OP\_DATA)**

Read-write, Volatile. Reset: 0000\_0000\_0000\_0000h.

\_lthree0\_core[3:0]\_thread[1:0]; MSRC001\_1035

Bits	Description
63:41	Reserved.
40	<b>IbsOpMicrocode.</b> Read-write, Volatile. Reset: 0. 1=Tagged operation from microcode.
39	<b>IbsOpBrnFuse: fused branch micro-op.</b> Read-write, Volatile. Reset: 0. 1=Tagged operation was a fused branch micro-op. Support indicated by Core::X86::Cpuid::IbsIdEax[OpBrnFuse].
38	<b>IbsRipInvalid: RIP is invalid.</b> Read-write, Volatile. Reset: 0. 1=Tagged operation RIP is invalid. Support indicated by Core::X86::Cpuid::IbsIdEax[RipInvalidChk].
37	<b>IbsOpBrnRet: branch micro-op retired.</b> Read-write, Volatile. Reset: 0. 1=Tagged operation was a branch micro-op that retired.
36	<b>IbsOpBrnMisp: mispredicted branch micro-op.</b> Read-write, Volatile. Reset: 0. 1=Tagged operation was a branch micro-op that was mispredicted. Qualified by IbsOpBrnRet == 1.
35	<b>IbsOpBrnTaken: taken branch micro-op.</b> Read-write, Volatile. Reset: 0. 1=Tagged operation was a branch micro-op that was taken. Qualified by IbsOpBrnRet == 1.
34	<b>IbsOpReturn: return micro-op.</b> Read-write, Volatile. Reset: 0. 1=Tagged operation was return micro-op. Qualified by (IbsOpBrnRet == 1).
33:32	Reserved.
31:16	<b>IbsTagToRetCtr: micro-op tag to retire count.</b> Read-write, Volatile. Reset: 0000h. This field returns the number of cycles from when the micro-op was tagged to when the micro-op was retired. This field is equal to IbsCompToRetCtr when the tagged micro-op is a NOP.



15:0	<b>IbsCompToRetCtr: micro-op completion to retire count.</b> Read-write, Volatile. Reset: 0000h. This field returns the number of cycles from when the micro-op was completed to when the micro-op was retired.
------	---

#### MSRC001\_1036 [IBS Op Data 2] (Core::X86::Msr::IBS\_OP\_DATA2)

Reset: 0000\_0000\_0000\_0000h.

Data is only valid for load operations that miss both the L1 data cache and the L2 cache. If a load operation crosses a cache line boundary, the data returned in this register is the data for the access to the lower cache line.

\_lthree0\_core[3:0]\_thread[1:0]; MSRC001\_1036

Bits	Description
63:6	Reserved.
5	<b>CacheHitSt: IBS cache hit state.</b> Read-write, Volatile. Reset: 0. 0=M State. 1=O State. Valid when the data source type is Cache(2h).
4	<b>RmtNode: IBS request destination node.</b> Read-write, Volatile. Reset: 0. 0=The request is serviced by the NB in the same node as the core. 1=The request is serviced by the NB in a different node than the core. Valid when NbIbsReqSrc is non-zero.
3	Reserved.
2:0	<b>DataSrc: northbridge IBS request data source.</b> Read-write. Reset: 0h.
<b>ValidValues:</b>	
Value	Description
0h	No valid status.
1h	Reserved.
2h	Cache: data returned from another cores cache.
3h	DRAM: data returned from DRAM.
4h	Reserved for remote cache.
6h-5h	Reserved.
7h	Other: data returned from MMIO/Config/PCI/APIC.

#### MSRC001\_1037 [IBS Op Data 3] (Core::X86::Msr::IBS\_OP\_DATA3)

Read-write, Volatile. Reset: 0000\_0000\_0000\_0000h.

If a load or store operation crosses a 256-bit boundary, the data returned in this register is the data for the access to the data below the 256-bit boundary.

\_lthree0\_core[3:0]\_thread[1:0]; MSRC001\_1037

Bits	Description
63:48	<b>IbsTlbRefillLat: L1 DTLB refill latency.</b> Read-write, Volatile. Reset: 0000h. The number of cycles from when a L1 DTLB refill is triggered by a tagged op to when the L1 DTLB fill has been completed.
47:32	<b>IbsDcMissLat: data cache miss latency.</b> Read-write, Volatile. Reset: 0000h. Indicates the number of clock cycles from when a miss is detected in the data cache to when the data was delivered to the core. The value returned by this counter is not valid for data cache writes or prefetch instructions.
31:26	<b>IbsOpDcMissOpenMemReqs: outstanding memory requests on DC fill.</b> 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	<b>IbsOpMemWidth: load/store size in bytes.</b> Read-write, Volatile. Reset: 0h. Report the number of bytes the load or store is attempting to access.
<b>ValidValues:</b>	
Value	Description
0h	No information provided.
1h	Byte.
2h	Word.
3h	DW.
4h	QW.
5h	OW.

	Fh-6h	Reserved.
21	<b>IbsSwPf: software prefetch.</b> Read-write, Volatile. Reset: 0. 1=The op is a software prefetch.	
20	<b>IbsL2Miss: L2 cache miss for the sampled operation.</b> Read-write, Volatile. Reset: 0. 1=The operation missed in the L2, regardless of whether the op initiated the request to the L2.	
19	<b>IbsDcL2TlbHit1G: data cache L2TLB hit in 1G page.</b> 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	<b>IbsDcPhyAddrValid: data cache physical address valid.</b> Read-write, Volatile. Reset: 0. 1=The physical address in Core::X86::Msrr::IBS_DC_PHYSADDR is valid for the load or store operation.	
17	<b>IbsDcLinAddrValid: data cache linear address valid.</b> Read-write, Volatile. Reset: 0. 1=The linear address in Core::X86::Msrr::IBS_DC_LINADDR is valid for the load or store operation.	
16	<b>DcMissNoMabAlloc: DC miss with no MAB allocated.</b> Read-write, Volatile. Reset: 0. 1=The tagged load or store operation hit on an already allocated MAB.	
15	<b>IbsDcLockedOp: locked operation.</b> Read-write, Volatile. Reset: 0. 1=Tagged load or store operation is a locked operation.	
14	<b>IbsDcUcMemAcc: UC memory access.</b> Read-write, Volatile. Reset: 0. 1=Tagged load or store operation accessed uncacheable memory.	
13	<b>IbsDcWcMemAcc: WC memory access.</b> Read-write, Volatile. Reset: 0. 1=Tagged load or store operation accessed write combining memory.	
12:9	Reserved.	
8	<b>IbsDcMisAcc: misaligned access.</b> Read-write, Volatile. Reset: 0. 1=The tagged load or store operation crosses a 256-bit address boundary.	
7	<b>IbsDcMiss: data cache miss.</b> Read-write, Volatile. Reset: 0. 1=The cache line used by the tagged load or store was not present in the data cache.	
6	<b>IbsDcL2tlbHit2M: data cache L2TLB hit in 2M page.</b> 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	<b>IbsDcL1TlbHit1G: data cache L1TLB hit in 1G page.</b> 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	<b>IbsDcL1TlbHit2M: data cache L1TLB hit in 2M page.</b> 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	<b>IbsDcL2TlbMiss: data cache L2TLB miss.</b> 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	<b>IbsDcL1tlbMiss: data cache L1TLB miss.</b> 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	<b>IbsStOp: store op.</b> Read-write, Volatile. Reset: 0. 1=Tagged operation is a store operation.	
0	<b>IbsLdOp: load op.</b> Read-write, Volatile. Reset: 0. 1=Tagged operation is a load operation.	

**MSRC001\_1038 [IBS DC Linear Address] (Core::X86::Msrr::IBS\_DC\_LINADDR)**

Read-write, Volatile. Reset: 0000\_0000\_0000\_0000h.

\_lthree0\_core[3:0]\_thread[1:0]; MSRC001\_1038

Bits	Description
63:0	<b>IbsDcLinAd.</b> 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::Msrr::IBS_OP_DATA3[IbsDcLinAddrValid] is asserted.

**MSRC001\_1039 [IBS DC Physical Address] (Core::X86::Msrr::IBS\_DC\_PHYSADDR)**

Read-write, Volatile. Reset: 0000\_0000\_0000\_0000h.

\_lthree0\_core[3:0]\_thread[1:0]; MSRC001\_1039

Bits	Description
63:48	Reserved.
47:0	<b>IbsDcPhysAd: load or store physical address.</b> Read-write, Volatile. Reset: 0000_0000_0000_0000h. Provides the physical address for the tagged load or store operation. The lower 12 bits are not modified by address translation,



so they are always the same as the linear address. This field contains valid data only if Core::X86::Msr::IBS\_OP\_DATA3[IbsDcPhyAddrValid] is asserted.

#### MSRC001\_103A [IBS Control] (Core::X86::Msr::IBS\_CTL)

Read,Error-on-write.

\_lthree0\_core[3:0]\_thread[1:0]; MSRC001\_103A

Bits	Description
63:9	Reserved.
8	<b>LvtOffsetVal: local vector table offset valid.</b> Read,Error-on-write. Reset: X.
7:4	Reserved.
3:0	<b>LvtOffset: local vector table offset.</b> 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].

\_lthree0\_core[3:0]\_thread[1:0]; MSRC001\_103B

Bits	Description
63:0	<b>IbsBrTarget.</b> Read-write, Volatile. Reset: 0000_0000_0000_0000h. The logical address in canonical form for the branch target. Contains a valid target if != 0. Qualified by Core::X86::Msr::IBS_OP_DATA[IbsOpBrnRet] == 1.

#### MSRC001\_103C [IBS Fetch Control Extended] (Core::X86::Msr::IC\_IBS\_EXTD\_CTL)

Read-only, Volatile. Reset: 0000\_0000\_0000\_0000h.

Support for this register indicated by Core::X86::Cpuid::IbsIdEax[IbsFetchCtlExtd].

\_lthree0\_core[3:0]\_thread[1:0]; MSRC001\_103C

Bits	Description
63:16	Reserved.
15:0	<b>IbsItlbRefillLat: ITLB Refill Latency for the sampled fetch, if there is a reload.</b> Read-only, Volatile. Reset: 0000h. The number of cycles when the fetch engine is stalled for an ITLB reload for the sampled fetch. If there is no reload, the latency == 0.

## 2.1.14 Performance Monitor Counters

### 2.1.14.1 RDPMC Assignments

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

- The RDPMC[5:0] instruction accesses core events. See 2.1.14.3 [Core Performance Monitor Counters].
- The RDPMC[9:6] instruction accesses data fabric events.
- The RDPMC[F:A] instruction accesses L3 cache events. See 2.1.14.4 [L3 Cache Performance Monitor Counters].

### 2.1.14.2 Large Increment per Cycle Events

Table 18: PMC\_Definitions

Term	Description
<b>MergeEvent</b>	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::Msrr::PERF\_CTL is programmed with the event Core::X86::Pmc::Core::Merge (PMCxFFFF) with the enable bit (En) turned on and with the remaining bits off. Then the corresponding even numbered Core::X86::Msrr::PERF\_CTL is programmed with the desired PMC event. The performance monitor combines the count value to an 8-bit increment event and extends the counter to a 64-bit counter.

Software wanting to preload a value to a merged counter pair writes the high-order 16-bit value to the low-order 16 bits of the odd counter and then writes the low-order 48-bit value to the even counter. Reading the even counter of the merged counter pair returns the full 64-bit value.

If an even performance monitor is programmed with the event Core::X86::Pmc::Core::Merge the Read results are undetermined. If an even performance monitor is programmed with a non-merge-able event (i.e., a 4-bit event) while the corresponding odd performance monitor is programmed as Merge, the Read results are undetermined. When discontinuing use of a merged counter pair, clear the Merge event from the odd performance monitor.

PMCxFFFF [Merge] (Core::X86::Pmc::Core::Merge)	
See 2.1.14.2 [Large Increment per Cycle Events].	
PMCxFFFF	
Bits	Description
7:0	Reserved.

### 2.1.14.3 Core Performance Monitor Counters

This section provides the core performance counter events that may be selected through Core::X86::Msrr::PERF\_CTL[EventSelect[11:8],EventSelect[7:0],UnitMask]. See Core::X86::Msrr::PERF\_CTR. See Core::X86::Msrr::PERF\_LEGACY\_CTL and Core::X86::Msrr::PERF\_LEGACY\_CTR.

#### 2.1.14.3.1 Floating Point (FP) Events

PMCx000 [FPU Pipe Assignment] (Core::X86::Pmc::Core::FpuPipeAssignment)	
Read-only. Reset: 00h.	
The number of operations (uOps) and dual-pipeuOps dispatched to each of the 4 FPU execution pipelines. This event reflects how busy the FPU pipelines are and may be used for workload characterization. This includes all operations performed by x87, MMX, and SSE instructions, including moves. Each increment represents a one-cycle dispatch event. This event is a speculative event. (See Core::X86::Pmc::Core::ExRetMmxFpInstr). Since this event includes non-numeric operations it is not suitable for measuring MFLOPs.	
PMCx000	
Bits	Description
7	<b>Dual3: Total number multi-pipe uOps assigned to Pipe 3.</b> Read-only. Reset: 0.
6	<b>Dual2: Total number multi-pipe uOps assigned to Pipe 2.</b> Read-only. Reset: 0.
5	<b>Dual1: Total number multi-pipe uOps assigned to Pipe 1.</b> Read-only. Reset: 0.
4	<b>Dual0: Total number multi-pipe uOps assigned to Pipe 0.</b> Read-only. Reset: 0.
3	<b>Total3: Total number uOps assigned to Pipe 3.</b> Read-only. Reset: 0.
2	<b>Total2: Total number uOps assigned to Pipe 2.</b> Read-only. Reset: 0.
1	<b>Total1: Total number uOps assigned to Pipe 1.</b> Read-only. Reset: 0.
0	<b>Total0: Total number uOps assigned to Pipe 0.</b> Read-only. Reset: 0.

PMCx001 [FP Scheduler Empty] (Core::X86::Pmc::Core::FpSchedEmpty)	
This is a speculative event. The number of cycles in which the FPU scheduler is empty. Note that some Ops like FP loads bypass the scheduler. Invert this (Core::X86::Msrr::PERF_CTL[Inv] == 1) to count cycles in which at least one FPU	

operation is present in the FPU.

PMCx001

Bits	Description
7:0	Reserved.

#### PMCx002 [Retired x87 Floating Point Operations] (Core::X86::Pmc::Core::FpRetx87FpOps)

Read-write. Reset: 00h.

The number of x87 floating-point Ops that have retired. The number of events logged per cycle can vary from 0 to 8.

PMCx002

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

#### PMCx003 [Retired SSE/AVX Operations] (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 is a MergeEvent since it can count above 15 events per cycle. See 2.1.14.2 [Large Increment per Cycle Events].

PMCx003

Bits	Description
7	<b>DpMultAddFLOPs: Double precision multiply-add FLOPs.</b> Read-write. Reset: 0. Multiply-add counts as 2 FLOPs.
6	<b>DpDivFLOPs: Double precision divide/square root FLOPs.</b> Read-write. Reset: 0.
5	<b>DpMultFLOPs: Double precision multiply FLOPs.</b> Read-write. Reset: 0.
4	<b>DpAddSubFLOPs: Double precision add/subtract FLOPs.</b> Read-write. Reset: 0.
3	<b>SpMultAddFLOPs: Single precision multiply-add FLOPs.</b> Read-write. Reset: 0. Multiply-add counts as 2 FLOPs.
2	<b>SpDivFLOPs: Single-precision divide/square root FLOPs.</b> Read-write. Reset: 0.
1	<b>SpMultFLOPs: Single-precision multiply FLOPs.</b> Read-write. Reset: 0.
0	<b>SpAddSubFLOPs: Single-precision add/subtract FLOPs.</b> Read-write. Reset: 0.

#### PMCx004 [Number of Move Elimination and Scalar Op Optimization] (Core::X86::Pmc::Core::FpNumMovElimScalOp)

Read-write. Reset: 00h.

This is a dispatch based speculative event, and is useful for measuring the effectiveness of the Move elimination and Scalar code optimization schemes.

PMCx004

Bits	Description
7:4	Reserved.
3	<b>Optimized: Number of Scalar Ops optimized.</b> Read-write. Reset: 0.
2	<b>OptPotential: Number of Ops that are candidates for optimization (have Z-bit either set or pass).</b> Read-write. Reset: 0.
1	<b>SseMovOpsElim: Number of SSE Move Ops eliminated.</b> Read-write. Reset: 0.
0	<b>SseMovOps: Number of SSE Move Ops.</b> 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	<b>X87CtrlRet: x87 control word mispredict traps due to mispredictions in RC or PC, or changes in mask bits.</b> Read-write. Reset: 0.
2	<b>X87BotRet: x87 bottom-executing uOps retired.</b> Read-write. Reset: 0.
1	<b>SseCtrlRet: SSE control word mispredict traps due to mispredictions in RC, FTZ or DAZ, or changes in mask bits.</b> Read-write. Reset: 0.
0	<b>SseBotRet: SSE bottom-executing uOps retired.</b> Read-write. Reset: 0.

### 2.1.14.3.2 LS Events

#### PMCx024 [Bad Status 2] (Core::X86::Pmc::Core::LsBadStatus2)

Read-write. Reset: 00h.

PMCx024

Bits	Description
7:2	Reserved.
1	<b>StliOther.</b> Read-write. Reset: 0.
0	Reserved.

#### PMCx025 [Retired Lock Instructions] (Core::X86::Pmc::Core::LsLocks)

Read-write. Reset: 00h.

Unit Mask 0x0E represents cacheable locks.

PMCx025

Bits	Description
7:1	Reserved.
0	<b>BusLock.</b> Read-write. Reset: 0. Comparable to legacy bus lock.

#### PMCx026 [Retired CLFLUSH Instructions] (Core::X86::Pmc::Core::LsRetClFlush)

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 ADDED.

PMCx029

Bits	Description
7:3	Reserved.
2	<b>LdStDispatch: Load-op-Store Dispatch.</b> Read-write. Reset: 0. Dispatch of a single op that performs a load from and store to the same memory address.
1	<b>StoreDispatch.</b> Read-write. Reset: 0. Dispatch of a single op that performs a memory store.
0	<b>LdDispatch.</b> Read-write. Reset: 0. Dispatch of a single op that performs a memory load.

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

Counts the number of SMIs received.

PMCx02B	
Bits	Description
7:0	Reserved.

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

Reset: 00h.

Counts the number of interrupts taken.

PMCx02C	
Bits	Description
7:0	Reserved.

**PMCx035 [Store to Load Forward] (Core::X86::Pmc::Core::LsSTLF)**

Number of STLF hits.

PMCx035	
Bits	Description
7:0	Reserved.

**PMCx037 [Store Commit Cancels 2] (Core::X86::Pmc::Core::LsStCommitCancel2)**

Read-write. Reset: 00h.

PMCx037	
Bits	Description
7:1	Reserved.
0	<b>StCommitCancelWcbFull</b> . Read-write. Reset: 0. A non-cacheable store and the non-cacheable commit buffer is full.

**PMCx040 [Data Cache Accesses] (Core::X86::Pmc::Core::LsDcAccesses)**

The number of accesses to the data cache for load and store references. This may include certain microcode scratchpad accesses, although these are generally rare. Each increment represents an eight-byte access, although the instruction may only be accessing a portion of that. This event is a speculative event.

PMCx040	
Bits	Description
7:0	Reserved.

**PMCx041 [LS MAB Allocates by Type] (Core::X86::Pmc::Core::LsMabAlloc)**

Read-write. Reset: 00h.

PMCx041	
Bits	Description
7:4	Reserved.
3	<b>DcPrefetcher</b> . Read-write. Reset: 0.
2	Reserved.
1	<b>Stores</b> . Read-write. Reset: 0.
0	<b>Loads</b> . Read-write. Reset: 0.

**PMCx043 [Data Cache Refills from System] (Core::X86::Pmc::Core::LsRefillsFromSys)**

Read-write. Reset: 00h.

Demand Data Cache Fills by Data Source.

PMCx043	
Bits	Description
7	Reserved.
6	<b>LS_MABRESP_RMT_DRAM</b> . Read-write. Reset: 0. From DRAM (home node remote).
5	Reserved.
4	<b>LS_MABRESP_RMT_CACHE</b> . Read-write. Reset: 0. From another cache (home node remote).

3	<b>LS_MABRESP_LCL_DRAM.</b> Read-write. Reset: 0. From DRAM (home node local).
2	Reserved.
1	<b>LS_MABRESP_LCL_CACHE.</b> Read-write. Reset: 0. From another cache (home node local).
0	<b>MABRESP_LCL_L2.</b> Read-write. Reset: 0. From local L2 hit.

**PMCx045 [L1 DTLB Misses] (Core::X86::Pmc::Core::LsL1DTlbMiss)**

Read-write. Reset: 00h.

PMCx045

Bits	Description
7	<b>TlbReload1GL2Miss.</b> Read-write. Reset: 0. DTLB reload to a 1G page that also miss in the L2 TLB.
6	<b>TlbReload2ML2Miss.</b> Read-write. Reset: 0. DTLB reload to a 2M page that also miss in the L2 TLB.
5	<b>TlbReload32KL2Miss.</b> Read-write. Reset: 0. DTLB reload to a 32K page that miss in the L2 TLB.
4	<b>TlbReload4KL2Miss.</b> Read-write. Reset: 0. DTLB reload to a 4K page that miss the L2 TLB.
3	<b>TlbReload1GL2Hit.</b> Read-write. Reset: 0. DTLB reload to a 1G page that hit in the L2 TLB.
2	<b>TlbReload2ML2Hit.</b> Read-write. Reset: 0. DTLB reload to a 2M page that hit in the L2 TLB.
1	<b>TlbReload32KL2Hit.</b> Read-write. Reset: 0. DTLB reload to a 32K page that hit in the L2 TLB.
0	<b>TlbReload4KL2Hit.</b> Read-write. Reset: 0. DTLB reload to a 4K page that hit in the L2 TLB.

**PMCx046 [Total Page Table Walks] (Core::X86::Pmc::Core::LsTablewalker)**

Read-write. Reset: 00h.

PMCx046

Bits	Description
7:4	Reserved.
3	<b>ICType1.</b> Read-write. Reset: 0. Tablewalker1 table walk initiated by IC.
2	<b>ICType0.</b> Read-write. Reset: 0. Tablewalker0 table walk initiated by IC.
1	<b>DCType1.</b> Read-write. Reset: 0. Tablewalker1 table walk initiated by LS/DC.
0	<b>DCType0.</b> Read-write. Reset: 0. Tablewalker0 table walk initiated by LS/DC.

**PMCx047 [Misaligned loads] (Core::X86::Pmc::Core::LsMisalLoads)**

PMCx047

Bits	Description
7:0	Reserved.

**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	<b>PrefetchNTA.</b> Read-write. Reset: 0. PrefetchNTA instruction. See docAPM3 PREFETCHlevel.
1	<b>PrefetchW.</b> Read-write. Reset: 0. PrefetchW instruction. See docAPM3 PREFETCHW.
0	<b>Prefetch: Prefetch_T0_T1_T2.</b> Read-write. Reset: 0. PrefetchT0, T1 and T2 instructions. See docAPM3 PREFETCHlevel.

**PMCx052 [Ineffective Software Prefetches] (Core::X86::Pmc::Core::LsInefSwPref)**

Read-write. Reset: 00h.

The number of software prefetches that did not fetch data outside of the processor core.

PMCx052

Bits	Description
7:2	Reserved.

1	<b>MabMchCnt.</b> Read-write. Reset: 0. Software PREFETCH instruction saw a match on an already-allocated miss request buffer.
0	<b>DataPipeSwPfdCtHit.</b> Read-write. Reset: 0. Software PREFETCH instruction saw a DC hit.

**PMCx059 [Software Prefetch Data Cache Fills] (Core::X86::Pmc::Core::LsSwPfdCtFills)**

Read-write. Reset: 00h.

Software Prefetch Data Cache Fills by Data Source.

PMCx059

Bits	Description
7	Reserved.
6	<b>LS_MABRESP_RMT_DRAM.</b> Read-write. Reset: 0. From DRAM (home node remote).
5	Reserved.
4	<b>LS_MABRESP_RMT_CACHE.</b> Read-write. Reset: 0. From another cache (home node remote).
3	<b>LS_MABRESP_LCL_DRAM.</b> Read-write. Reset: 0. From DRAM (home node local).
2	Reserved.
1	<b>LS_MABRESP_LCL_CACHE.</b> Read-write. Reset: 0. From another cache (home node local).
0	<b>MABRESP_LCL_L2.</b> Read-write. Reset: 0. From local L2 hit.

**PMCx05A [Hardware Prefetch Data Cache Fills] (Core::X86::Pmc::Core::LsHwPfdCtFills)**

Read-write. Reset: 00h.

Hardware Prefetch Data Cache Fills by Data Source.

PMCx05A

Bits	Description
7	Reserved.
6	<b>LS_MABRESP_RMT_DRAM.</b> Read-write. Reset: 0. From DRAM (home node remote).
5	Reserved.
4	<b>LS_MABRESP_RMT_CACHE.</b> Read-write. Reset: 0. From another cache (home node remote).
3	<b>LS_MABRESP_LCL_DRAM.</b> Read-write. Reset: 0. From DRAM (home node local).
2	Reserved.
1	<b>LS_MABRESP_LCL_CACHE.</b> Read-write. Reset: 0. From another cache (home node local).
0	<b>MABRESP_LCL_L2.</b> Read-write. Reset: 0. From local L2 hit.

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

This event counts the in-flight L1 data cache misses each cycle. This event is a MergeEvent since it can count above 15 events per cycle. See 2.1.14.2 [Large Increment per Cycle Events].

PMCx05F

Bits	Description
7:0	Reserved.

**PMCx076 [Cycles not in Halt] (Core::X86::Pmc::Core::LsNotHaltedCyc)**

PMCx076

Bits	Description
7:0	Reserved.



### 2.1.14.3.3 IC and BP Events

Note: All instruction cache events are speculative events unless specified otherwise.

#### PMCx080 [Total Number of 32B Instruction Fetches] (Core::X86::Pmc::Core::IcFw32)

The number of 32B instruction cache fetches issued to the instruction decoder.

PMCx080

Bits	Description
7:0	Reserved.

#### PMCx081 [Number of 32B Instruction Fetches that Miss in Instruction Cache] (Core::X86::Pmc::Core::IcFw32Miss)

The number of 32B fetches that tried to read the L1 instruction cache and missed.

PMCx081

Bits	Description
7:0	Reserved.

#### PMCx082 [Instruction Cache Refills from L2] (Core::X86::Pmc::Core::IcCacheFillL2)

The number of 64-byte instruction cache line was fulfilled from the L2 cache.

PMCx082

Bits	Description
7:0	Reserved.

#### PMCx083 [Instruction Cache Refills from System] (Core::X86::Pmc::Core::IcCacheFillSys)

The number of 64-byte instruction cache line fulfilled from system memory or another cache.

PMCx083

Bits	Description
7:0	Reserved.

#### PMCx084 [L1 ITLB Miss, L2 ITLB Hit] (Core::X86::Pmc::Core::BpL1TlbMissL2TlbHit)

The number of instruction fetches that miss in the L1 ITLB but hit in the L2 ITLB.

PMCx084

Bits	Description
7:0	Reserved.

#### PMCx085 [L1 ITLB Miss, L2 ITLB Miss] (Core::X86::Pmc::Core::BpL1TlbMissL2TlbMiss)

The number of instruction fetches that miss in both the L1 and L2 TLBs.

PMCx085

Bits	Description
7:0	Reserved.

#### PMCx08A [L1 Branch Prediction Overrides Existing Prediction (speculative)] (Core::X86::Pmc::Core::BpL1BTBCorrect)

PMCx08A

Bits	Description
7:0	Reserved.

#### PMCx08B [L2 Branch Prediction Overrides Existing Prediction (speculative)] (Core::X86::Pmc::Core::BpL2BTBCorrect)

PMCx08B

Bits	Description
7:0	Reserved.

#### PMCx08C [Instruction Cache Lines Invalidated] (Core::X86::Pmc::Core::IcCacheInval)

Read-write. Reset: 00h.	
The number of instruction cache lines invalidated. A non-SMC event is CMC (cross modifying code), either from the other thread of the core or another core.	
PMCx08C	
Bits	Description
7:2	Reserved.
1	<b>L2InvalidatingProbe.</b> Read-write. Reset: 0. IC line invalidated due to L2 invalidating probe (external or LS).
0	<b>FillInvalidated.</b> Read-write. Reset: 0. IC line invalidated due to overwriting fill response.

#### PMCx08E [Dynamic Indirect Predictions] (Core::X86::Pmc::Core::BpDynIndPred)

Indirect Branch Prediction for potential multi-target branch (speculative).	
PMCx08E	
Bits	Description
7:0	Reserved.

#### PMCx091 [Decoder Overrides Existing Branch Prediction (speculative)] (Core::X86::Pmc::Core::BpDeReDirect)

PMCx091	
Bits	Description
7:0	Reserved.

#### PMCx099 [L1 ITLB Reloads] (Core::X86::Pmc::Core::BpTlbRel)

The number of ITLB reload requests.	
PMCx099	
Bits	Description
7:0	Reserved.

#### PMCx28A [Switches between instruction fetch and op cache fetch modes] (Core::X86::Pmc::Core::IcOcModeSwitch)

Read-write. Reset: 00h.	
See Core::X86::Pmc::Core::DeDisUopsFromDecoder (PMCx0AA) for number of ops dispatched from OC.	
PMCx28A	
Bits	Description
7:2	Reserved.
1	<b>OcIcModeSwitch.</b> Read-write. Reset: 0. OC to IC mode switch.
0	<b>IcOcModeSwitch.</b> Read-write. Reset: 0. IC to OC mode switch.

### 2.1.14.3.4 DE Events

#### PMCx0AA [UOps Dispatched From Decoder] (Core::X86::Pmc::Core::DeDisUopsFromDecoder)

Read-write. Reset: 00h.	
Ops dispatched from either the decoders, OpCache or both.	
PMCx0AA	
Bits	Description
7:2	Reserved.
1	<b>OpCacheDispatched: Count of dispatched Ops from OpCache.</b> Read-write. Reset: 0.
0	<b>DecoderDispatched: Count of dispatched Ops from Decoder.</b> Read-write. Reset: 0.

#### 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.	
PMCx0AE	

Bits	Description
7	<b>FPMiscRsrcStall: FP Miscellaneous resource unavailable.</b> Read-write. Reset: 0. Applies to the recovery of mispredicts with FP ops.
6	<b>FPSchRsrcStall: FP scheduler resource stall.</b> Read-write. Reset: 0. Applies to ops that use the FP scheduler.
5	<b>FpRegFileRsrcStall: floating point register file resource stall.</b> Read-write. Reset: 0. Applies to all FP ops that have a destination register.
4	<b>TakenBrnchBufferRsrc: taken branch buffer resource stall.</b> Read-write. Reset: 0.
3	<b>IntSchedulerMiscRsrcStall: Integer Scheduler miscellaneous resource stall.</b> Read-write. Reset: 0.
2	<b>StoreQueueRsrcStall: Store Queue resource stall.</b> Read-write. Reset: 0. Applies to all ops with store semantics.
1	<b>LoadQueueRsrcStall: Load Queue resource stall.</b> Read-write. Reset: 0. Applies to all ops with load semantics.
0	<b>IntPhyRegFileRsrcStall: Integer Physical Register File resource stall.</b> Read-write. Reset: 0. Integer Physical Register File, applies to all ops that have an integer destination register.

#### PMCx0AF [Dispatch Resource Stall Cycles 0] (Core::X86::Pmc::Core::DeDisDispatchTokenStalls0)

Read-write. Reset: 00h.

Cycles where a dispatch group is valid but does not get dispatched due to a token stall.

PMCx0AF

Bits	Description
7	Reserved.
6	<b>RetireRsrcStall: RETIRE Resource unavailable.</b> Read-write. Reset: 0.
5	<b>AGSQRsrcStall: AGSQ Resources unavailable.</b> Read-write. Reset: 0.
4	<b>ALURsrcStall: ALU Resources total unavailable.</b> Read-write. Reset: 0.
3	<b>ALSQ3_0_RsrcStall.</b> Read-write. Reset: 0.
2	<b>ALSQ3RsrcStall: ALSQ 3 Resources unavailable.</b> Read-write. Reset: 0.
1	<b>ALSQ2RsrcStall: ALSQ 2 Resources unavailable.</b> Read-write. Reset: 0.
0	<b>ALSQ1RsrcStall: ALSQ 1 Resources unavailable.</b> Read-write. Reset: 0.

### 2.1.14.3.5 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. This count includes all processor activity (instructions, exceptions, interrupts, microcode assists, etc.). The number of events logged per cycle can vary from 0 to 8.

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 branch instructions retired, of any type, that were not correctly predicted. This includes those for which prediction is not attempted (far control transfers, exceptions and interrupts).

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	<b>SseInstr.</b> Read-write. Reset: 0. SSE instructions (SSE, SSE2, SSE3, SSSE3, SSE4A, SSE41, SSE42, AVX).
1	<b>MmxInstr.</b> Read-write. Reset: 0. MMX instructions.
0	<b>X87Instr: x87 instructions.</b> Read-write. Reset: 0.

**PMCx0D1 [Retired Conditional Branch Instructions] (Core::X86::Pmc::Core::ExRetCond)**

PMCx0D1

Bits	Description
7:0	Reserved.

**PMCx0D3 [Div Cycles Busy count] (Core::X86::Pmc::Core::ExDivBusy)**

PMCx0D3

Bits	Description
7:0	Reserved.

**PMCx0D4 [Div Op Count] (Core::X86::Pmc::Core::ExDivCount)**

PMCx0D4

Bits	Description
7:0	Reserved.

**PMCx1C7 [Retired Mispredicted Branch Instructions due to Direction Mismatch] (Core::X86::Pmc::Core::ExRetMsprdBrnchInstrDirMsmth)**

The number of retired conditional branch instructions that were not correctly predicted because of a branch direction mismatch.

PMCx1C7

Bits	Description
7:0	Reserved.

**PMCx1CF [Tagged IBS Ops] (Core::X86::Pmc::Core::ExTaggedIbsOps)**

Read-write. Reset: 00h.

Counts Op IBS related events.

PMCx1CF

Bits	Description
7:3	Reserved.
2	<b>IbsCountRollover.</b> 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	<b>IbsTaggedOpsRet.</b> Read-write. Reset: 0. Number of Ops tagged by IBS that retired.
0	<b>IbsTaggedOps: Number of Ops tagged by IBS.</b> Read-write. Reset: 0.

**PMCx1D0 [Retired Fused Instructions] (Core::X86::Pmc::Core::ExRetFusBrnchInst)**

The number of fuse-branch instructions retired per cycle. The number of events logged per cycle can vary from 0-8.

PMCx1D0

Bits	Description
7:0	Reserved.

**2.1.14.3.6 L2 Cache Events****PMCx060 [Requests to L2 Group1] (Core::X86::Pmc::Core::L2RequestG1)**

Read-write. Reset: 00h.

All L2 Cache Requests (Breakdown 1 - Common).

PMCx060

Bits	Description
------	-------------

7	<b>RdBlkL</b> . Read-write. Reset: 0. Data Cache Reads (including hardware and software prefetch).
6	<b>RdBlkX</b> . Read-write. Reset: 0. Data Cache Stores.
5	<b>LsRdBlkC_S</b> . Read-write. Reset: 0. Data Cache Shared Reads.
4	<b>CacheableIcRead</b> . Read-write. Reset: 0. Instruction Cache Reads.
3	<b>ChangeToX: Data Cache State Change Requests</b> . Read-write. Reset: 0. Request change to writable, check L2 for current state.
2	<b>PrefetchL2: Software Prefetch</b> . Read-write. Reset: 0. Assume core should also count these and allow the breakdown between hardware versus software and data versus instruction.
1	<b>L2HwPf: L2 Prefetcher</b> . Read-write. Reset: 0. All prefetches accepted by L2 pipeline, hit or miss. Types of PF and L2 hit/miss broken out in a separate perfmon event.
0	<b>Group2</b> . Read-write. Reset: 0. Miscellaneous events covered in more detail by Core::X86::Pmc::Core::L2RequestG2 (PMCx061).

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

Read-write. Reset: 00h.

All L2 Cache Requests (Breakdown 2 - Rare).

PMCx061

Bits	Description
7	<b>Group1</b> . Read-write. Reset: 0. Miscellaneous events covered in more detail by Core::X86::Pmc::Core::L2RequestG1 (PMCx060).
6	<b>LsRdSized</b> . Read-write. Reset: 0. Data cache Read sized.
5	<b>LsRdSizedNC</b> . Read-write. Reset: 0. Data cache Read sized non-cacheable.
4	<b>IcRdSized</b> . Read-write. Reset: 0. Instruction cache Read sized.
3	<b>IcRdSizedNC</b> . Read-write. Reset: 0. Instruction cache Read sized non-cacheable.
2	<b>SmcInval</b> . Read-write. Reset: 0. Self-modifying code invalidates.
1	<b>BusLocksOriginator: Bus locks</b> . Read-write. Reset: 0.
0	<b>BusLocksResponses</b> . Read-write. Reset: 0. Bus Lock Response.

#### PMCx062 [L2 Latency] (Core::X86::Pmc::Core::L2Latency)

Read-write. Reset: 00h.

Total cycles spent waiting for L2 fills to complete from L3 or memory, divided by four. This may be used to calculate average latency by multiplying this count by four and then dividing by the total number of L2 fills (unit mask Core::X86::Pmc::Core::L2RequestG1 == FEh). Event counts are for both threads. To calculate average latency, the number of fills from both threads must be used.

PMCx062

Bits	Description
7:1	Reserved.
0	<b>L2CyclesWaitingOnFills</b> . Read-write. Reset: 0.

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

Read-write. Reset: 00h.

L2 Cache Request Outcomes (not including L2 Prefetch).

PMCx064

Bits	Description
7	<b>LsRdBlkCS: Data Cache Shared Read Hit in L2</b> . Read-write. Reset: 0.
6	<b>LsRdBlkLHitX: Data Cache Read Hit in L2</b> . Read-write. Reset: 0.
5	<b>LsRdBlkLHitS: Data Cache Read Hit Non-Modifiable Line in L2</b> . Read-write. Reset: 0.
4	<b>LsRdBlkX: Data Cache Store or State Change Hit in L2</b> . Read-write. Reset: 0.
3	<b>LsRdBlkC: Data Cache Req Miss in L2 (all types)</b> . Read-write. Reset: 0.
2	<b>IcFillHitX: Instruction Cache Hit Modifiable Line in L2</b> . Read-write. Reset: 0.
1	<b>IcFillHitS: Instruction Cache Hit Non-Modifiable Line in L2</b> . Read-write. Reset: 0.

0	<b>ICFillMiss: Instruction Cache Req Miss in L2.</b> Read-write. Reset: 0.
---	--

**PMCx06D [Cycles with fill pending from L2] (Core::X86::Pmc::Core::L2FillPending)**

Read-write. Reset: 00h.

Total cycles spent with one or more fill requests in flight from L2.

PMCx06D

Bits	Description
7:1	Reserved.
0	<b>L2FillBusy.</b> Read-write. Reset: 0.

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

Reset: 00h.

Requires unit mask 0xFF to engage event for counting.

PMCx070

Bits	Description
7:0	Reserved.

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

Reset: 00h.

Requires unit mask 0xFF to engage event for counting.

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

PMCx071

Bits	Description
7:0	Reserved.

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

Reset: 00h.

Requires unit mask 0xFF to engage event for counting.

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

PMCx072

Bits	Description
7:0	Reserved.

#### 2.1.14.4 L3 Cache Performance Monitor Counters

This section provides the core performance counter events that may be selected through Core::X86::Msr::ChL3PmcCfg.

- Unless otherwise noted, L3 Perfmon events require the Core::X86::Msr::ChL3PmcCfg[SliceMask] field to be set or the PMC count will be zero.
- Unless otherwise noted, L3 PMC's require Core::X86::Msr::ChL3PmcCfg[ThreadMask] 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.

##### 2.1.14.4.1 L3 Cache PMC Events

**L3PMCx01 [L3 Cache Accesses] (Core::X86::Pmc::L3::L3RequestG1)**

Read-write. Reset: 00h.

L3PMCx01

Bits	Description
7	<b>Caching: L3 cache accesses.</b> Read-write. Reset: 0.
6:0	Reserved.



**L3PMCx04 [All L3 Cache Requests] (Core::X86::Pmc::L3::L3LookupState)**

Read-write. Reset: 00h.	
L3PMCx04	
Bits	Description
7:0	<b>AllL3ReqTypes: All L3 Request Types.</b> Read-write. Reset: 00h.

**L3PMCx06 [L3 Miss] (Core::X86::Pmc::L3::L3CombClstrState)**

Read-write. Reset: 00h.	
L3 Cache Request Outcomes.	
L3PMCx06	
Bits	Description
7:1	<b>OtherL3MissTypes: Other L3 Miss Request Types.</b> Read-write. Reset: 00h.
0	<b>RequestMiss: L3 cache miss.</b> Read-write. Reset: 0.

**L3PMCx90 [L3 Cache Miss Latency] (Core::X86::Pmc::L3::XiSysFillLatency)**

Ignores SliceMask and ThreadMask	
Total cycles for all transactions divided by 16.	
L3PMCx90	
Bits	Description
7:0	Reserved.

**L3PMCx9A [L3 Misses by Request Type] (Core::X86::Pmc::L3::XiCcxSdpReq1)**

Reset: 00h.	
Ignores SliceMask and ThreadMask.	
Requires unit mask 0xFF to engage event for counting.	
L3PMCx9A	
Bits	Description
7:0	Reserved.

**2.1.15 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.

### 3 Reliability, Availability, and Serviceability (RAS) Features

A full implementation of RAS involves capabilities and support from the processor design, board hardware design, BIOS, firmware, and software.

#### 3.1 Machine Check Architecture

*Table 19: Machine Check Terms and Acronyms*

Term	Description
<b>MCA</b>	Machine Check Architecture.
<b>MCAX</b>	Machine Check Architecture eXtensions.
<b>WRIG</b>	Writes Ignored.

##### 3.1.1 Overview

The processor contains logic and registers to detect, log, and correct errors in the data or control paths. The Machine Check Architecture (MCA) defines facilities by which processor and system hardware errors are logged and reported to system software. This allows system software to perform a strategic role in recovery from and diagnosis of hardware errors.

##### 3.1.1.1 Legacy Machine Check Architecture

The legacy x86 Machine Check Architecture (MCA) refers to the standard x86 facilities for error logging and reporting. Refer to the AMD64 Architecture Programmer's Manual for an architectural overview of the Machine Check Architecture.

Support for the MCA is indicated by Core::X86::Cpuid::FeatureIdEdx[MCA] or Core::X86::Cpuid::FeatureExtIdEdx[MCA].

##### 3.1.1.2 Machine Check Architecture Extensions

Machine Check Architecture Extensions (MCAX) is AMD's x86-64 extension to the Machine Check Architecture.

Goals of MCAX include:

- Accommodate a variety of implementations, where each implementation may have a different assignment of MCA bank to block.
  - For example, one implementation may have 1 memory channel with an MCA bank, and another otherwise identical implementation may have 2 memory channels, each with their own MCA bank. Therefore, MCA bank allocation will appear different between these two implementations. MCAX is designed to require no assumptions about which MCA banks access which blocks.
  - Provide granular information for error logging, to improve error handling and diagnosability.
  - Preserve compatibility with system software which is not MCAX-aware.

Features of the MCA Extensions include:

- Increased MCA Bank Count: Features to support an expansion of the number of MCA banks supported by AMD processors.
- MCA Extension Registers: Expanded information logged in MCA banks to allow for improved error handling, better diagnosability, and future scalability.
- MCA DOER/SEER Roles: Separation of MCA information to take advantage of emerging software roles, namely

Error Management (Dynamic Operational Error Handling, or DOER) for managing running programs, and Fault Management (Symptom Elaboration of Errors, or SEER) for hardware diagnosability and reconfiguration. This clearer separation is accompanied by the assurances of architectural state (vs. implementation dependent state), so that operating systems can rely on the state and exploit new functionality.

Support for Machine Check Architecture Extensions (MCAX) is indicated by Core::X86::Cpuid::RasCap[ScalableMca].

### 3.1.1.3 Use of MCA Information

The MCA registers contain information that can be used for multiple purposes. Some of this information is architecturally specified, and remains consistent from generation to generation, enabling portable, stable code. Some of this information is implementation specific; it is vital for diagnosis and other software functions, but may change with new implementations. It is important to understand how this information is categorized, and how it should be used. This section describes a framework for that.

There are two fundamental roles to be carried out after an error occurs; Error Management and Fault Management. All information required for Error Management is architectural and stable; some information required for Fault Management is also architectural.

#### 3.1.1.3.1 Error Management

Error Management describes actions necessary by operational software (e.g., the operating system or the hypervisor) to manage running programs that are affected by the error. The list of possible actions for operational error management is generally fairly short: take no action; terminate a single affected process, program, or virtual machine; terminate system operation. The Error Management role is defined as the DOER role (Dynamic Operational Error Handling). The name is intended to indicate an active role in managing running programs. Information used by the DOER is fairly limited and straightforward. It includes only those status fields needed to make decisions about the scope and severity of the error, and to determine what immediate action is to be taken.

#### 3.1.1.3.2 Fault Management

Fault Management describes optional actions for purposes of diagnosis, repair, and reconfiguration of the underlying hardware. The Fault Management role is described as SEER (Symptom Elaboration of Errors) because it peers further into hardware behavior and may try to influence future behavior via Predictive Fault Analysis, reconfiguration, service actions, etc. Because the SEER depends on understanding specifics of hardware configuration, it necessarily requires implementation specific knowledge and may not be portable across implementations.

Fields that are not explicitly specified as DOER are SEER. By separating error handling software into DOER and SEER roles, programmers can create both simpler and more functional code. The terms DOER and SEER appear in other sections of this document as an aid to reasoning about error handling and understanding actions to be taken.

### 3.1.2 Machine Check Registers

Host software references MCA registers via MSRs. MSRs are accessed through x86 WRMSR and RDMSR instructions. MSR addresses are private to a logical core; a given MSR referenced by two different cores results in references to two different MCA registers.

#### 3.1.2.1 Global Registers

Core::X86::Cpuid::FeatureIdEdx[MCA] or Core::X86::Cpuid::FeatureExtIdEdx[MCA] indicates the presence of the following machine check registers:

- Core::X86::Msr::MCG\_CAP
  - Reports how many machine check register banks are supported. This value reflects the number of MCA banks visible to that logical core. Some banks may be RAZ/WRIG either due to the bank being reserved or unused on this processor or because the block's MCA bank is controlled by another logical core.
- Core::X86::Msr::MCG\_STAT
  - Provides basic information about processor state after the occurrence of a machine check error.
- Core::X86::Msr::MCG\_CTL
  - Used by software to enable or disable the logging and reporting of machine check errors in the error reporting banks. Some bits may be RAZ/WRIG either due to the bank being reserved or unused on this processor or because the block's MCA bank is controlled by another logical core.
- Core::X86::Msr::McaIntrCfg
  - Used by software to configure certain machine check interrupts.

### 3.1.2.2 Machine Check Banks

A processor contains multiple blocks, and some of them have banks of machine check architecture registers (MCA banks). An MCA bank logs and reports errors to software.

The legacy MCA supports up to 32 MCA banks per logical core. MCAX supports up to 64 MCA banks per logical core.

The processor ensures that non-zero error status in an MCA bank is visible to exactly one logical core in a system, and that error notifications are directed to that logical core. Hardware also makes MCA bank configuration and control registers available to exactly one logical core. Banks associated with a CPU core are controlled by that logical core. Banks associated with other blocks are controlled by an implementation-specific logical core.

#### 3.1.2.2.1 Legacy MCA Registers

Each legacy MCA bank allocates address space for 4 legacy MCA registers.

The legacy MCA registers include:

- MCA\_CTL
  - Enables error reporting via machine check exception.
- MCA\_STATUS
  - Logs information associated with errors.
- MCA\_ADDR
  - Logs address information associated with errors.
- MCA\_MISC0
  - Logs miscellaneous information associated with errors.

#### 3.1.2.2.2 Legacy MCA MSRs

The legacy MCA MSRs are MSR0000\_04[7F:00]. The legacy MCA MSR space contains 32 banks of 4 registers per bank. The layout of the legacy MCA MSR space is given in Table 20 [Legacy MCA MSR Layout].

Table 20: Legacy MCA MSR Layout

MCA bank (decimal)	MCA_CTL (MSR0000_0xxx)	MCA_STATUS	MCA_ADDR	MCA_MISC0
0	400	401	402	403
1	404	405	406	407
2	408	409	40A	40B
3	40C	40D	40E	40F

4	410	411	412	413
5	414	415	416	417
6	418	419	41A	41B
...				
31	47C	47D	47E	47F

Features and registers associated with the MCA Extensions are not available in this legacy MSR address range. AMD recommends that operating systems use the MCAX MSR address range, rather than rely on the legacy MCA MSR address range.

All unimplemented or unused registers in the legacy MCA MSR address range are RAZ/WRIG. MC4 registers (MSR0000\_0410:0000\_0413) are RAZ/WRIG.

MSR0000\_0000 is aliased to the MCAX MSR address for MC0\_ADDR, and MSR0000\_0001 is aliased to the MCAX MSR address of MC0\_STATUS.

### 3.1.2.2.3 MCAX Registers

Each MCAX bank allocates address space for 16 MCA registers. All unimplemented registers in the MCA MSR space are RAZ/WRIG. MCAX bank registers include the legacy MCA registers as well as registers associated with the MCA Extensions.

The MCA Extension registers include:

- MCA\_CONFIG
  - Provide configuration capabilities for this MCA bank.
- MCA\_IPID
  - Provides information on the block associated with this MCA bank.
- MCA\_SYND
  - Logs physical location information associated with a logged error.
- MCA\_DESTSTATUS
  - Logs status information associated with a deferred error.
- MCA\_DEADDR
  - Logs address information associated with a deferred error.
- MCA\_MISC[1:4]
  - Provides additional threshold counters within an MCA bank.

### 3.1.2.2.4 MCAX MSRs

MCAX MSRs are present at MSRC000\_2[3FF:000]. This MSR address range contains space for 64 banks of 16 registers each. MSRC000\_2[FFF:400] are Reserved for future use. The MCAX MSR address range allows access to both legacy MCA registers and MCAX registers in each MCA bank.

The x86 MCAX MSR address format is SSSS\_SBBR (hex). S=MCA register space (i.e., MSRC000\_2XXX). B=MCA bank. R=Register offset within MCA bank. The layout of the MCAX MSR space is given in Table 21 [MCAX MSR Layout].

Access to unused MCAX MSRs is RAZ/WRIG. MCA Bank 4 is always Read-as-zero (RAZ/WRIG).

Table 21: MCAX MSR Layout

MCA bank	MCAX MSR (MSRC000_2xxx)	
	Legacy MCA Bank registers	MCAX Bank registers

	CTL	STATUS	ADDR	MISC0	CONFIG	IPID	SYND	Reserved	DESTAT	DEADDR	MISC[4:1]
0	000	001	002	003	004	005	006	007	008	009	00D:00A
1	010	011	012	013	014	015	016	017	018	019	01D:01A
2	020	021	022	023	024	025	026	027	028	029	02D:02A
...											
63	3F0	3F1	3F2	3F3	3F4	3F5	3F6	3F7	3F8	3F9	3FD:3FA

All processors maintain the same mapping of MSR to MCA bank number (MSRC000\_2000 for the beginning of MCA Bank 0, MSRC000\_2010 for the beginning of MCA Bank 1, etc.), regardless of what block the bank represents (see 3.1.5.5 [Determining Bank Type]).

MCA\_CTL\_MASK MSRs are present at MSRC001\_04[3F:00]. MSRC001\_04[FF:40] are Reserved for future use. The layout of these registers is given in Table 22 [MCAX Implementation-Specific Register Layout].

*Table 22: MCAX Implementation-Specific Register Layout*

MCA bank	MCA_CTL_MASK (MSRC001_04xx)
0	00
1	01
2	02
...	
63	3F

### 3.1.2.3 Access Permissions

When McStatusWrEn == 0, a Write to an implemented MCA\_STATUS register causes a General Protection Fault (#GP) unless the value being written is zero. When McStatusWrEn == 1, a Write to an implemented MCA\_STATUS register does not cause a #GP regardless of data value.

Access to legacy MCA\_CTL\_MASK (MSRC001\_00xx) causes a General Protection Fault (#GP).

Access to legacy MC4\_MISC1-8 (MSRC000\_0408:C000\_040F) is RAZ/WRIG.

### 3.1.3 Machine Check Errors

#### 3.1.3.1 Error Severities

The classes of machine check errors are, in priority order from highest to lowest:

- Uncorrected
- Deferred
- Corrected

Uncorrected errors cannot be corrected by hardware. Uncorrected errors update the status and address registers if not masked from logging in MCA\_CTL\_MASK. Information in the status and address registers from a previously logged lower priority error is overwritten. Previously logged errors of the same priority are not overwritten. Uncorrected errors that are enabled for reporting in MCA\_CTL result in reporting to software via machine check exceptions. If an uncorrected error is masked from logging, the error is ignored by hardware (exceptions are noted in the register definitions). If an uncorrected error is disabled from reporting, containment of the error and logging/reporting of subsequent errors may be affected. Therefore, enable reporting of unmasked uncorrected errors for normal operation. Disable reporting of uncorrected errors only for debug purposes.



Deferred errors are errors that cannot be corrected by hardware, but do not cause an immediate interruption in program flow, loss of data integrity, or corruption of processor state. These errors indicate that data has been corrupted but not consumed; no exception is generated because the data has not been referenced by a core or an IO link. Hardware writes information to the status and address registers in the corresponding bank that identifies the source of the error if deferred errors are enabled for logging. If there is information in the status and address registers from a previously logged lower priority error, it is overwritten. Previously logged errors of the same or higher priority are not overwritten. Deferred errors are not reported via machine check exceptions; they can optionally be reported via LVT or SMI.

Corrected errors are those which have been corrected by hardware and cause no loss of data or corruption of processor state. Hardware writes the status and address registers in the corresponding register bank with information that identifies the source of the error if they are enabled for logging. Corrected errors are not reported via machine check exceptions. Some corrected errors may optionally be reported to software via LVT or SMI if the number of errors exceeds a configurable threshold.

An error to be logged when the status register contains valid data can result in an overflow condition. During error overflow conditions, the new error may not be logged or an error which has already been logged in the status register may be overwritten.

Table 23 [Error Overwrite Priorities] indicates which errors are overwritten in the error status registers.

*Table 23: Error Overwrite Priorities*

		Older Error		
		Uncorrected	Deferred	Corrected
Newer Error	Uncorrected	-	Overwrite	Overwrite
	Deferred	-	-	Overwrite
	Corrected	-	-	-

Table 24 [Error Scope Hierarchy] provides a hierarchy of error scopes that determine the potential ability to recover the system based on fields in MCA\_STATUS when MCA\_STATUS[Val] == 1.

*Table 24: Error Scope Hierarchy*

PCC	UC	TCC	Deferred	Comments
1	X	X	X	Uncorrected system fatal error. Action required. A hardware-uncorrected error has corrupted system state. The error is fatal to the system and the system processing must be terminated.
0	1	1	X	Uncorrected thread fatal error. Action required. A hardware-uncorrected error has corrupted state for the process thread executing on the interrupted logical core. State for other process threads is unaffected.
0	1	0	X	Uncorrected recoverable error. Action required. A hardware-uncorrected error has not corrupted state of the process thread. Recovery of the process thread is possible if the uncorrected error is corrected by software.
0	0	0	1	Deferred error. Action optional. A hardware-uncorrected error has been discovered but not yet consumed. Error handling software may attempt to correct this error, or prevent access by processes which map the data, or make the physical resource containing the data inaccessible.

0	0	0	0	Corrected error. Action optional. A hardware-corrected error has been corrected. No action is required by error handling software.
---	---	---	---	--

### 3.1.3.2 Exceptions and Interrupts

Some or all errors logged in the MCA may require an interrupt or exception to be signaled.

The processor supports the following x86 interrupt/exception types to be communicated to the x86 core in response to an error:

- Machine Check Exception (MCE)
- System Management Interrupt (SMI)
- APIC based interrupt (LVT)

MCEs can be architecturally precise, context-synchronous, or asynchronous. An MCE that sets Core::X86::Msr::MCG\_STAT[RIPV] = 1 and Core::X86::Msr::MCG\_STAT[EIPV] = 1 is precise and the program can be restarted reliably. Other interrupts are architecturally asynchronous.

The ability of hardware to generate a machine check exception upon an error is indicated by Core::X86::Cpuid::FeatureIdEdx[MCE] or Core::X86::Cpuid::FeatureExtIdEdx[MCE].

### 3.1.3.3 Error Codes

The MCA\_STATUS[ErrorCode] field contains information used to identify the logged error. This section identifies how to decode the ErrorCode field.

Table 25: Error Code Types

Error Code	Error Code Type	Description
0000 0000 0001 TTLL	TLB	TT = Transaction Type LL = Cache Level
0000 0001 RRRR TTLL	Memory	RRRR = Memory Transaction Type TT = Transaction Type LL = Cache Level
0000 1XXT RRRR XXLL	Bus	XX = Reserved T = Timeout RRRR = Memory Transaction Type LL = Cache Level
0000 01UU 0000 0000	Internal Unclassified	UU = Internal Error Type

Table 26: Error code: transaction type (TT)

TT	Transaction Type
00	Instruction
01	Data
10	Generic
11	Reserved

Table 27: Error codes: cache level (LL)

LL	Cache Level
00	L0: Core
01	L1: Level 1

10	L2: Level 2
11	LG: Generic

Table 28: Error codes: memory transaction type (RRRR)

RRRR	Memory Transaction Type
0000	Generic
0001	Generic Read
0010	Generic Write
0011	Data Read
0100	Data Write
0101	Instruction Fetch
0110	Prefetch
0111	Evict
1000	Snoop (Probe)

Errors can also be identified by the MCA\_STATUS[ErrorCodeExt] field. MCA\_STATUS[ErrorCodeExt] indicates which bit position in the corresponding MCA\_CTL register enables error reporting for the logged error. For instance, MCA\_STATUS[ErrorCodeExt] == 0x9 means that the logged error is enabled by MCA\_CTL[9], and the description of MCA\_CTL[9] contains information on decoding the error log. Specific ErrorCodeExt values are implementation dependent, and should not be used by architectural or portable code.

### 3.1.3.4 Extended Error Codes

The MCA\_STATUS[ErrorCodeExt] field contains additional information used to identify the logged error. Error positions in MCA\_CTL and MCA\_CTL\_MASK and Extended Error Codes are fixed within a given bank type. That is, for an MCA bank with a given MCA\_IPID[HwId, McaType] value, the processor ensures that the same error is reported in a given bit position of MCA\_CTL regardless of the product in which that bank appears. Similarly, for an MCA bank with a given MCA\_IPID[HwId, McaType] value, hardware ensures that the mapping of errors to Extended Error Codes is consistent across products.

### 3.1.3.5 DOER and SEER State

The DOER fields are:

- MCG\_STAT
  - Count
  - MCIP
  - RIPV
  - EIPV
- MCA\_STATUS
  - Val
  - PCC
  - TCC
  - UC
  - MiscV
  - AddrV

The MCA\_STATUS[Deferred] bit is used for SEER functionality but is architectural.

### 3.1.3.6 MCA Overflow Recovery

MCA Overflow Recovery is a feature allowing recovery of the system when the overflow bit is set. MCA Overflow Recovery is supported when `Core::X86::Cpuid::RasCap[McaOverflowRecov] == 1`.

When MCA Overflow Recovery is supported, software may rely on `MCA_STATUS[PCC] == 1` to indicate all system-fatal conditions. When MCA Overflow Recovery is not supported, an uncorrected error logged with `MCA_STATUS[Overflow] = 1` may indicate the system-fatal condition that an error requiring software intervention was not logged. Therefore, software must terminate system processing whenever an uncorrected error is logged with `MCA_STATUS[Overflow] = 1`.

### 3.1.3.7 MCA Recovery

MCA Recovery is a feature allowing recovery of the system when the hardware cannot correct an error. MCA Recovery is supported when `Core::X86::Cpuid::RasCap[SUCCOR] == 1`.

When MCA Recovery is supported and an uncorrected error has been detected that the hardware can contain to the task or process to which the machine check has been delivered, it logs a context-synchronous uncorrectable error (`MCA_STATUS[UC] = 1`, `MCA_STATUS[PCC] = 0`). The rest of the system is unaffected and may continue running if supervisory software can terminate only the affected process or VM.

## 3.1.4 Machine Check Features

### 3.1.4.1 Error Thresholding

For some types of errors, the hardware maintains counts of the number of errors. When the counter reaches a programmable threshold, an event may optionally be triggered to signal system software. This is known as error thresholding. The primary purpose of error thresholding is to help software recognize an excessive rate of errors, which may indicate marginal or failing hardware. This information can be used to make decisions about deconfiguring hardware or scheduling service actions. The error count is incremented for corrected, deferred, and uncorrected errors.

The `MCA_MISCx` registers contain the architectural interface for error thresholding. The registers contain a 12-bit error counter that can be initialized to any value except `FFFh`, with the option to interrupt when the counter reaches `FFFh`.

`MCA_MISCx[ThresholdIntType]` determines the type of interrupt to be generated for threshold overflow errors in that counter. This can be set to `None`, `LVT`, or `SMI`. If this is set to `LVT`, `Core::X86::Msr::McaIntrCfg[ThresholdLvtOffset]` specifies the LVT offset that is used. Only one LVT offset is used per socket and the interrupt is routed to the APIC of the logical core from which the MCA bank is visible.

### 3.1.4.2 Error Simulation

Error simulation involves creating the appearance to software that an error occurred, and can be used to debug machine check interrupt handlers. See `Core::X86::Msr::HWCR[McStatusWrEn]` for making MCA registers writable for non-zero values. When `McStatusWrEn` is set, privileged software can write non-zero values to the specified registers without generating exceptions, and then simulate a machine check using the `INT18` instruction (`INTn` instruction with an operand of 18). Setting a reserved bit in these registers does not generate an exception when this mode is enabled. However, setting a reserved bit may result in undefined behavior.

### 3.1.5 Software Guidelines

#### 3.1.5.1 Recognizing MCAX Support

Software which reads the MCA registers must recognize whether an implementation uses the legacy format or the MCAX format. This is accomplished by starting with CPUID Fn8000\_0007\_EBX[ScalableMca]. If ScalableMca == 1, then the implementation supports the MCAX indicator (MCA\_CONFIG[Mcax]). An MCA bank is an MCAX bank if MCA\_CONFIG[Mcax] == 1 in that bank.

#### 3.1.5.2 Communicating MCAX Support

Software which supports MCAX must set MCA\_CONFIG[McaxEn] = 1 in each MCA bank.

Software that supports MCAX should use the MCAX MSRs to access both legacy and MCAX registers.

#### 3.1.5.3 Machine Check Initialization

The following initialization sequence must be followed:

- Platform firmware must initialize the MCA\_CTL\_MASK registers prior to the initialization of the MCA\_CTL registers and Core::X86::Msr::MCG\_CTL. Platform firmware and the operating system must not clear MCA\_CTL\_MASK bits that are set to 1. MCA\_CTL\_MASK registers must be set the same across all cores.
- The operating system must initialize the MCA\_CONFIG registers prior to initialization of the MCA\_CTL registers.
- The MCA\_CTL registers must be initialized prior to enabling the error reporting banks in MCG\_CTL.
- The Core::X86::Msr::MCG\_CTL register must be programmed identically for all cores in a processor, although the Read-write bits may differ per core.
- CR4.MCE must be set to enable machine check exceptions.

The operating system should configure the MCA\_CONFIG registers as follows:

- MCA\_CONFIG[McaxEn] = 1 if the operating system has been updated to use the MCA Extension MSR addresses. Otherwise, the operating system should preserve the platform firmware-programmed value of this field.
- MCA\_CONFIG[LogDeferredInMcaStat] and MCA\_CONFIG[DeferredIntType] to appropriate values based on OS support for deferred errors.

MCA\_STATUS MSRs are cleared by hardware after a cold reset. If initializing after a warm reset, then platform firmware should check for valid MCA errors and if present save the status for later diagnostic use.

Platform firmware may initialize the MCA without setting CR4.MCE; this results in a shutdown on any machine check which would have caused a machine check exception (followed by a reboot if configured). Alternatively, platform firmware that wishes to ensure continued operation in the event that a machine check occurs during boot may write MCG\_CTL with all ones and write zeros into each MCA\_CTL register. With these settings, a machine check error results in MCA\_STATUS being written without generating a machine check exception or a shutdown. Platform firmware may then poll MCA\_STATUS registers during critical sections of boot to ensure system integrity. Note that the system may be operating with corrupt data before polling MCA\_STATUS registers. Before passing control to the operating system, platform firmware should restore the values of those registers to what the operating system is expecting.

After MCA initialization, system software should check the Val bit on each MCA\_STATUS register. It is possible that valid error status information has already been logged in the MCA\_STATUS registers at the time software is attempting to initialize them. The status can reflect errors logged prior to a warm reset or errors recorded during the system power-up and boot process. Before clearing the MCA\_STATUS registers, software should examine their contents and log any errors

found.

### 3.1.5.4 Determining Bank Count

System software should Read Core::X86::Msr::MCG\_CAP[Count] to determine the number of machine check banks visible to a logical core. The banks are numbered from 0 to one less than the value found in Core::X86::Msr::MCG\_CAP[Count]. For example, if the Count field indicates five banks are supported, they are numbered MC0 through MC4.

### 3.1.5.5 Determining Bank Type

To determine which type of block is mapped to an MCA bank, software can query the MCA\_IPID register within that bank. This register exists when MCA\_CONFIG[McaX] == 1 in a given bank.

MCA\_IPID[HardwareID] provides the block type for the block that contains this MCA bank. For blocks that contain multiple MCA bank types (e.g., CPU cores), MCA\_IPID[McaType] provides an identifier for the type of MCA bank. MCA\_IPID[McaType] values are specific to a given MCA\_IPID[HardwareID]. Therefore, an MCA bank type can be identified by the value of {MCA\_IPID[Hwid], MCA\_IPID[McaType]}. For instance, the CPU core's LS bank is identified by MCA::LS::MCA\_IPID\_LS[HardwareID] == 176 and MCA::LS::MCA\_IPID\_LS[McaType] == 0. An MCA\_IPID[HardwareID] value of 0 indicates an unpopulated MCA bank that is ensured to be RAZ/WRIG.

MCA\_IPID[InstanceId] provides a unique instance number to allow software to differentiate blocks with multiple identical instances within a processor. MCA\_IPID[InstanceId] values are processor-specific and are not ensured to be stable across different processor generations.

### 3.1.5.6 Recognizing Error Type

Software can use the combination of MCA\_IPID[Hwid, McaType] and MCA\_STATUS[ErrorCodeExt] to recognize a specific error type.

### 3.1.5.7 Machine Check Error Handling

A machine check handler is invoked to handle an exception for a particular thread. The information needed by the machine check handler is not shared with other threads, so no cross-thread coordination or special handling is required. Specifically, all MCA banks are only visible from a single thread, so software on a single thread can access each bank through MSR space without contention from other threads.

At a minimum, the machine check handler must be capable of logging error information for later examination. The handler should log as much information as is needed to diagnose the error. More thorough exception handler implementations can analyze errors to determine if each error is recoverable by software. If a recoverable error is identified, the exception handler can attempt to correct the error and restart the interrupted program. An error may not be recoverable for the process or virtual machine it directly affects, but may be containable, so that other processes or virtual machines in the system are unaffected and system operation is recovered.

Machine check exception handlers that attempt to recover must be thorough in their analysis and the corrective actions they take. The following guidelines should be used when writing such a handler:

- Data collection:
  - Read Core::X86::Msr::MCG\_CAP[Count] to determine the number of status registers visible to the logical core.
  - All status registers in all error reporting banks must be examined to identify the cause of the machine check exception.

- Check the valid bit in each status register (MCA\_STATUS[Val]). The remainder of the status register should be examined only when its valid bit is set.
- When identifying the error condition and determining how to handle the error, portable exception handlers should examine only DOER fields in machine check registers.
- Error handlers should collect all available MCA information, but should only interrogate details to the level which affects their actions. Lower level details may be useful for diagnosis and root cause analysis, but not for error handling.
- Error handlers should save the values in MCA\_ADDR, MCA\_MISC0, and MCA\_SYND even if MCA\_STATUS[AddrV], MCA\_STATUS[MiscV], and MCA\_STATUS[SyndV] are zero. Error handlers should save the values in MCA\_MISC[4:1] if the registers exist.
- DOER Error Management:
  - Check MCA\_STATUS[PCC].
    - If PCC is set, error recovery is not possible. The handler should log the error information and terminate the system. If PCC is clear, the handler may continue with the following recovery steps.
  - Check MCA\_STATUS[UC].
    - If UC is set, the processor did not correct the error. Continue with the following recovery steps.
      - If MCA Overflow Recovery is not supported, and MCA\_STATUS[Overflow] == 1, error recovery is not possible; follow the steps for PCC = 1. See 3.1.3.6 [MCA Overflow Recovery].
      - If MCA Recovery is not supported, error recovery is not possible; follow the steps for PCC = 1. See 3.1.3.7 [MCA Recovery].
      - If MCA Recovery is supported:
        - Check MCA\_STATUS[TCC].
          - If TCC is set, the context of the process thread executing on the interrupted logical core may be corrupt and the thread cannot be recovered. The rest of the system is unaffected; it is possible to terminate only the affected process thread.
          - If TCC is clear, the context of the process thread executing on the interrupted logical core is not corrupt. Recovery of the process thread may be possible, but only if the uncorrected error condition is first corrected by software; otherwise, the interrupted process thread must be terminated.
          - Legacy exception handlers can check Core::X86::Msr::MCG\_STAT[RIPV] and Core::X86::Msr::MCG\_STAT[EIPV] in place of MCA\_STATUS[TCC]. If RIPV == EIPV == 1, the interrupted program can be restarted reliably. Otherwise, the program cannot be restarted reliably.
    - If UC is clear, the processor either corrected or deferred the error and no software action is needed. The handler can log the error information and continue process execution.
  - Exit:
    - When an exception handler is able to successfully log an error condition, clear the MCA\_STATUS registers prior to exiting the machine check handler.
    - Prior to exiting the machine check handler, clear Core::X86::Msr::MCG\_STAT[MCIP]. MCIP indicates that a machine check exception is in progress. If this bit is set when another machine check exception occurs, the processor enters the shutdown state.

## 3.2 Machine Check Architecture Implementation

### 3.2.1 Implemented Machine Check Banks

Blocks capable of supporting MCA banks are given in Table 29 [Blocks Capable of Supporting MCA Banks]. Whether a



particular block implements an MCA bank is processor-specific and can be determined by querying MCA\_IPID for each MCA bank. The block type can be uniquely identified by the MCA\_IPID[HardwareId] and MCA\_IPID[McaType] values in each MCA bank.

**Table 29: Blocks Capable of Supporting MCA Banks**

Block	Block Function	MCA_IPID[HardwareId]	MCA_IPID[McaType]
LS	Load-Store Unit	0xB0	0x0
IF	Instruction Fetch Unit	0xB0	0x1
L2	L2 Cache Unit	0xB0	0x2
DE	Decode Unit	0xB0	0x3
EX	Execution Unit	0xB0	0x5
FP	Floating Point Unit	0xB0	0x6
L3	L3 Cache Unit	0xB0	0x7
UMC	Unified Memory Controller	0x96	0x0
SMU	System Management Unit	0x01	0x0
PSP	Platform Security Processor	0xFF	0x0
PB	Parameter Block	0x5	0x0
CS	Coherent Slave	0x2E	0x0
PIE	Power Management, Interrupts, Etc.	0x2E	0x1

### 3.2.2 Implemented Machine Check Bank Registers

Table 30 [Legacy MCA Registers] provides links to the description of each block's Legacy MCA registers. Table 31 [MCAX Registers] provides links to the description of each block's MCA Extension Registers.

**Table 30: Legacy MCA Registers**

Block	MCA Register				
	CTL	STATUS	ADDR	MISC	CTL_MASK
LS	MCA::LS::MCA_CTL_LS	MCA::LS::MCA_STATUS_LS	MCA::LS::MCA_ADDR_LS	MCA::LS::MCA_MISC0_LS	MCA::LS::MCA_CTL_MASK_LS
IF	MCA::IF::MCA_CTL_IF	MCA::IF::MCA_STATUS_IF	MCA::IF::MCA_ADDR_IF	MCA::IF::MCA_MISC0_IF	MCA::IF::MCA_CTL_MASK_IF
L2	MCA::L2::MCA_CTL_L2	MCA::L2::MCA_STATUS_L2	MCA::L2::MCA_ADDR_L2	MCA::L2::MCA_MISC0_L2	MCA::L2::MCA_CTL_MASK_L2
DE	MCA::DE::MCA_CTL_DE	MCA::DE::MCA_STATUS_DE	MCA::DE::MCA_ADDR_DE	MCA::DE::MCA_MISC0_DE	MCA::DE::MCA_CTL_MASK_DE
EX	MCA::EX::MCA_CTL_EX	MCA::EX::MCA_STATUS_EX	MCA::EX::MCA_ADDR_EX	MCA::EX::MCA_MISC0_EX	MCA::EX::MCA_CTL_MASK_EX
FP	MCA::FP::MCA_CTL_FP	MCA::FP::MCA_STATUS_FP	MCA::FP::MCA_ADDR_FP	MCA::FP::MCA_MISC0_FP	MCA::FP::MCA_CTL_MASK_FP
L3	MCA::L3::MCA_CTL_L3	MCA::L3::MCA_STATUS_L3	MCA::L3::MCA_ADDR_L3	MCA::L3::MCA_MISC0_L3	MCA::L3::MCA_CTL_MASK_L3
PIE	MCA::PIE::MCA_CTL_PIE	MCA::PIE::MCA_STATUS_PIE	MCA::PIE::MCA_ADDR_PIE	MCA::PIE::MCA_MISC0_PIE	MCA::PIE::MCA_CTL_MASK_PIE
CS	MCA::CS::MCA_CTL_CS	MCA::CS::MCA_STATUS_CS	MCA::CS::MCA_ADDR_CS	MCA::CS::MCA_MISC0_CS	MCA::CS::MCA_CTL_MASK_CS
UMC	MCA::UMC::MCA_CTL_UMC	MCA::UMC::MCA_STATUS_UMC	MCA::UMC::MCA_ADDR_UMC	MCA::UMC::MCA_MISC0_UMC MCA::UMC::MCA_MISC1_UMC	MCA::UMC::MCA_CTL_MASK_UMC

**Table 31: MCAX Registers**

Block	MCA Register				
	CONFIG	IPID	SYND	DESTAT	DEADDR
LS	MCA::LS::MCA_CONFIG_LS	MCA::LS::MCA_IPID_LS	MCA::LS::MCA_SYND_LS	MCA::LS::MCA_DESTAT_LS	MCA::LS::MCA_DEADDR_LS

IF	MCA::IF::MCA_CONFIG_IF	MCA::IF::MCA_IPID_IF	MCA::IF::MCA_SYND_IF	--	--
L2	MCA::L2::MCA_CONFIG_L2	MCA::L2::MCA_IPID_L2	MCA::L2::MCA_SYND_L2	MCA::L2::MCA_DESTAT_L2	MCA::L2::MCA_DEADDR_L2
DE	MCA::DE::MCA_CONFIG_DE	MCA::DE::MCA_IPID_DE	MCA::DE::MCA_SYND_DE	--	--
EX	MCA::EX::MCA_CONFIG_EX	MCA::EX::MCA_IPID_EX	MCA::EX::MCA_SYND_EX	--	--
FP	MCA::FP::MCA_CONFIG_FP	MCA::FP::MCA_IPID_FP	MCA::FP::MCA_SYND_FP	--	--
L3	MCA::L3::MCA_CONFIG_L3	MCA::L3::MCA_IPID_L3	MCA::L3::MCA_SYND_L3	MCA::L3::MCA_DESTAT_L3	MCA::L3::MCA_DEADDR_L3
PIE	MCA::PIE::MCA_CONFIG_PIE	MCA::PIE::MCA_IPID_PIE	MCA::PIE::MCA_SYND_PIE	MCA::PIE::MCA_DESTAT_PIE	MCA::PIE::MCA_DEADDR_PIE
CS	MCA::CS::MCA_CONFIG_CS	MCA::CS::MCA_IPID_CS	MCA::CS::MCA_SYND_CS	MCA::CS::MCA_DESTAT_CS	MCA::CS::MCA_DEADDR_CS
UMC	MCA::UMC::MCA_CONFIG_UMC	MCA::UMC::MCA_IPID_UMC	MCA::UMC::MCA_SYND_UMC	MCA::UMC::MCA_DESTAT_UMC	MCA::UMC::MCA_DEADDR_UMC

### 3.2.3 Mapping of Banks to Blocks

Table 32: MCA Bank to Block Mapping

Bank	Block
0	LS
1	IF
2	L2
3	DE
4	RAZ
5	EX
6	FP
7	L3
8	L3
9	L3
10	L3
11	L3
12	L3
13	L3
14	L3
15	UMC
16	UMC
17	SMU
18	PSP
19	PB
20	CS
21	CS
22	PIE

### 3.2.4 Decoding Error Type

If a valid error is logged in MCA\_STATUS or MCA\_DESTAT of an MCA bank:

1. Read the values of this bank's MCA\_IPID and MCA\_STATUS registers.
2. Use Table 29 [Blocks Capable of Supporting MCA Banks] to look up the block associated with the values of

MCA\_IPID[HwId] and MCA\_IPID[McaType].

3. In 3.2.5 [MCA Banks], find the sub-section associated with the block in error.
4. In this sub-section, find the MCA\_STATUS table.
5. In the table, look up the row associated with the MCA\_STATUS[ErrorCodeExt] value.
6. The error type in this row is the logged error. The MCA\_STATUS, MCA\_ADDR and MCA\_SYND tables contain information associated with this error.
7. If there is an error in both MCA\_STATUS and MCA\_DESTAT, the registers contain the same error if MCA\_STATUS[Deferred] is set. If MCA\_STATUS[Deferred] is not set, MCA\_DESTAT contains information for a different error than MCA\_STATUS. MCA\_DESTAT does not contain an ErrorCodeExt field, so in this case it is not possible to determine the type of error logged in MCA\_DESTAT.

### 3.2.5 MCA Banks

#### 3.2.5.1 LS

##### MSR0000\_0400...MSRC000\_2000 [LS Machine Check Control] (MCA::LS::MCA\_CTL\_LS)

Read-write. Reset: 0000\_0000\_0000\_0000h.

0=Disables error reporting for the corresponding error. 1=Enables error reporting via machine check exception for the corresponding error. The MCA::LS::MCA\_CTL\_LS register must be enabled by the corresponding enable bit in Core::X86::Msrr::MCG\_CTL. Does not affect error detection, correction, or logging.

\_lthree0\_core[3:0]\_inst0\_aliasMSRLEGACY; MSR0000\_0400

\_lthree0\_core[3:0]\_inst0\_aliasMSR; MSRC000\_2000

Bits	Description
63:21	Reserved.
20	<b>L2DataErr</b> . Read-write. Reset: 0. L2 Fill Data error.
19	<b>DcTagErr5</b> . Read-write. Reset: 0. DC Tag error type 5.
18	<b>DcTagErr3</b> . Read-write. Reset: 0. DC Tag error type 3.
17	<b>PDC</b> . Read-write. Reset: 0. PDC parity error. MCA_ADDR_LS logs a virtual address.
16	<b>L2DTLB</b> . Read-write. Reset: 0. Level 2 TLB parity error. MCA_ADDR_LS logs a virtual address.
15	<b>DcTagErr4</b> . Read-write. Reset: 0. DC Tag error type 4.
14	<b>DcDataErr3</b> . Read-write. Reset: 0. DC Data error type 3.
13	<b>DcDataErr2</b> . Read-write. Reset: 0. DC Data error type 2.
12	<b>DcDataErr1</b> . Read-write. Reset: 0. DC Data error type 1 and poison consumption. MCA_STATUS[Poison] is set on poison consumption from L2/L3.
11	<b>DcTagErr2</b> . Read-write. Reset: 0. DC Tag error type 2.
10	<b>SystemReadDataErrorT1</b> . Read-write. Reset: 0. System Read Data Error Thread 1. An error in a read of a line from the data fabric. Possible reasons include master abort and target abort.
9	<b>SystemReadDataErrorT0</b> . Read-write. Reset: 0. System Read Data Error Thread 0. An error in a read of a line from the data fabric. Possible reasons include master abort and target abort.
8	<b>IntErrTyp2</b> . Read-write. Reset: 0. Internal error type 2.
7	<b>IntErrTyp1</b> . Read-write. Reset: 0. Internal error type 1.
6	<b>DcTagErr1</b> . Read-write. Reset: 0. DC Tag error type 1.
5	<b>DcTagErr6</b> . Read-write. Reset: 0. DC Tag error type 6.
4	Reserved.
3	<b>L1DTLB</b> . Read-write. Reset: 0. Level 1 TLB parity error.
2	<b>MAB</b> . Read-write. Reset: 0. Miss address buffer payload parity error.
1	<b>STQ</b> . Read-write. Reset: 0. Store queue parity error.
0	<b>LDQ</b> . Read-write. Reset: 0. Load queue parity error.

**MSR0000\_0001...MSRC000\_2001 [LS Machine Check Status Thread 0] (MCA::LS::MCA\_STATUS\_LS)**

Reset: Cold,0000\_0000\_0000\_0000h.

Logs information associated with errors.

\_lthree0\_core[3:0]\_thread[1:0]\_inst0\_aliasMSRSLLEGACY; MSR0000\_0001

\_lthree0\_core[3:0]\_thread[1:0]\_inst0\_aliasMSRLEGACY; MSR0000\_0401

\_lthree0\_core[3:0]\_thread[1:0]\_inst0\_aliasMSR; MSRC000\_2001

Bits	Description
63	<b>Val.</b> Reset: Cold,0. 1=A valid error has been detected. This bit should be cleared by software after the register has been read. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
62	<b>Overflow.</b> Reset: Cold,0. 1=An error was detected while the valid bit (Val) was set; at least one error was not logged. Overflow is set independently of whether the existing error is overwritten. See 3.1.3 [Machine Check Errors]. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
61	<b>UC.</b> Reset: Cold,0. 1=The error was not corrected by hardware. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
60	<b>En.</b> Reset: Cold,0. 1=MCA error reporting is enabled for this error, as indicated by the corresponding bit in MCA::LS::MCA_CTL_LS. This bit is a copy of bit in MCA::LS::MCA_CTL_LS for this error. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
59	<b>MiscV.</b> Reset: Cold,0. 1=Valid thresholding in MCA::LS::MCA_MISC0_LS. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
58	<b>AddrV.</b> Reset: Cold,0. 1=MCA::LS::MCA_ADDR_LS contains address information associated with the error. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
57	<b>PCC.</b> Reset: Cold,0. 1=Hardware context held by the processor may have been corrupted. Continued operation of the system may have unpredictable results. The error is not recoverable or survivable, and the system should be reinitialized. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
56	<b>ErrCoreIdVal.</b> Reset: Cold,0. 1=The ErrCoreId field is valid. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
55	<b>TCC.</b> Reset: Cold,0. 0=The thread which consumed the error can be restarted reliably at the instruction pointer address pushed onto the exception handler stack if any uncorrected error has been corrected by software. 1=The thread which consumed the error is not restartable and must be terminated. Only meaningful when MCA::LS::MCA_STATUS_LS[PCC]=0. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
54	Reserved.
53	<b>SyndV.</b> Reset: Cold,0. 1=This error logged information in MCA::LS::MCA_SYND_LS. If MCA::LS::MCA_SYND_LS[ErrorPriority] is the same as the priority of the error in MCA::LS::MCA_STATUS_LS, then the information in MCA::LS::MCA_SYND_LS is associated with the error in MCA::LS::MCA_STATUS_LS. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
52:47	Reserved.
46	<b>CECC.</b> Reset: Cold,0. 1=The error was a correctable ECC error according to the restrictions of the ECC algorithm. UC indicates whether the error was actually corrected by the processor. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
45	<b>UECC.</b> Reset: Cold,0. 1=The error was an uncorrectable ECC error according to the restrictions of the ECC algorithm. UC indicates whether the error was actually corrected by the processor. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
44	<b>Deferred.</b> Reset: Cold,0. 1=A deferred error was created. A deferred error is the result of an uncorrectable data error which did not immediately cause a processor exception; an exception is deferred until the erroneous data is consumed.

	AccessType: Core::X86::Msrr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
43	<b>Poison.</b> Reset: Cold,0. 1=The error was the result of attempting to consume poisoned data.
	AccessType: Core::X86::Msrr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
42:38	Reserved.
37:32	<b>ErrCoreId.</b> Reset: Cold,00h. When ErrCoreIdVal=1 this field indicates which core within the processor is associated with the error; Otherwise this field is reserved.
	AccessType: Core::X86::Msrr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
31:22	Reserved.
21:16	<b>ErrorCodeExt.</b> Reset: Cold,00h. Extended Error Code. This field is used to identify the error type for root cause analysis. This field indicates which bit position in MCA::LS::MCA_CTL_LS enables error reporting for the logged error.
	AccessType: Core::X86::Msrr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
15:0	<b>ErrorCode.</b> Reset: Cold,0000h. Error code for this error. See 3.1.3.3 [Error Codes] for details on decoding this field.
	AccessType: Core::X86::Msrr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.

Table 33: MCA\_STATUS\_LS[ErrorCodeExt] Decode

Error Type	Bits	Description
LDQ	[6:0]	0x0
STQ	[6:0]	0x1
MAB	[6:0]	0x2
L1DTLB	[6:0]	0x3
DcTagErr5	[6:0]	0x4
DcTagErr6	[6:0]	0x5
DcTagErr1	[6:0]	0x6
IntErrTyp1	[6:0]	0x7
IntErrTyp2	[6:0]	0x8
SystemReadDataErrorT0	[6:0]	0x9
SystemReadDataErrorT1	[6:0]	0xa
DcTagErr2	[6:0]	0xb
DcDataErr1	[6:0]	0xc
DcDataErr2	[6:0]	0xd
DcDataErr3	[6:0]	0xe
DcTagErr4	[6:0]	0xf
L2DTLB	[6:0]	0x10
PDC	[6:0]	0x11
DcTagErr3	[6:0]	0x12
DcTagErr5	[6:0]	0x13
L2DataErr	[6:0]	0x14

**MSR0000\_0000...MSRC000\_2002 [LS Machine Check Address Thread 0] (MCA::LS::MCA\_ADDR\_LS)**

Reset: Cold,0000_0000_0000_0000h.	
MCA::LS::MCA_ADDR_LS stores an address and other information associated with the error in MCA::LS::MCA_STATUS_LS. The register is only meaningful if MCA::LS::MCA_STATUS_LS[Val]=1 and MCA::LS::MCA_STATUS_LS[AddrV]=1.	
_lthree0_core[3:0]_thread[1:0]_inst0_aliasMSRSLLEGACY; MSR0000_0000	
_lthree0_core[3:0]_thread[1:0]_inst0_aliasMSRLEGACY; MSR0000_0402	
_lthree0_core[3:0]_thread[1:0]_inst0_aliasMSR; MSRC000_2002	
Bits	Description
63:62	Reserved.

61:56	<b>LSB.</b> Read-write, Volatile. Reset: Cold, 00h. Specifies the least significant valid bit of the address contained in MCA::LS::MCA_ADDR_LS[ErrorAddr]. A value of 0 indicates that MCA::LS::MCA_ADDR_LS[55:0] contains a valid byte address. A value of 6 indicates that MCA::LS::MCA_ADDR_LS[55:6] contains a valid cache line address and that MCA::LS::MCA_ADDR_LS[5:0] are not part of the address and should be ignored by error handling software. A value of 12 indicates that MCA::LS::MCA_ADDR_LS[55:12] contain a valid 4KB memory page and that MCA::LS::MCA_ADDR_LS[11:0] should be ignored by error handling software.
55:0	<b>ErrorAddr.</b> Read-write, Volatile. Reset: Cold, 00_0000_0000_0000h. Unless otherwise specified by an error, contains the address associated with the error logged in MCA::LS::MCA_STATUS_LS. For physical addresses, the most significant bit is given by Core::X86::CpuId::LongModeInfo[PhysAddrSize].

Table 34: MCA\_ADDR\_LS Register

Error Type	Bits	Description
LDQ	[55:0]	Reserved
STQ	[55:0]	Reserved
MAB	[55:0]	Reserved
L1DTLB	[55:48] [47:12] [11:0]	Reserved Virtual Address Reserved
DcTagErr5	[55:0]	Reserved
DcTagErr6	[55:0]	Reserved
DcTagErr1	[55:0]	Reserved
IntErrTyp1	[55:0]	Reserved
IntErrTyp2	[55:0]	Reserved
SystemReadDataErrorT0	[55:48] [47:6]	Reserved Physical Address
SystemReadDataErrorT1	[55:48] [47:6]	Reserved Physical Address
DcTagErr2	[55:48] [47:6]	Reserved Physical Address
DcDataErr1	[55:48] [47:6]	Reserved Physical Address
DcDataErr2	[55:48] [47:1]	Reserved Physical Address
DcDataErr3	[55:48] [47:1]	Reserved Physical Address
DcTagErr4	[55:0]	Reserved
L2DTLB	[55:48] [47:12] [11:0]	Reserved Virtual Address Reserved
PDC	[55:48] [47:12] [11:0]	Reserved Virtual Address Reserved
DcTagErr3	[55:0]	Reserved
DcTagErr5	[55:0]	Reserved
L2DataErr	[55:48] [47:6]	Reserved Physical Address

**MSR0000\_0403...MSRC000\_2003 [LS Machine Check Miscellaneous 0 Thread 0] (MCA::LS::MCA\_MISC0\_LS)**

Log miscellaneous information associated with errors.	
_lthree0_core[3:0]_thread[1:0]_inst0_aliasMSRLEGACY; MSR0000_0403	
_lthree0_core[3:0]_thread[1:0]_inst0_aliasMSR; MSRC000_2003	
Bits	Description
63	<b>Valid.</b> Reset: 1. 1=A valid CntP field is present in this register. AccessType: Core::X86::Msrr::HWCR[McStatusWrEn] ? Read-write : Read-only.
62	<b>CntP.</b> Reset: 1. 1=A valid threshold counter is present. AccessType: Core::X86::Msrr::HWCR[McStatusWrEn] ? Read-write : Read-only.
61	<b>Locked.</b> Reset: 0. 1=Writes to this register are ignored. This bit is set by BIOS to indicate that this register is not available for OS use. BIOS should set this bit if ThresholdIntType is set to SMI. AccessType: Core::X86::Msrr::HWCR[McStatusWrEn] ? Read-write : Read-only.
60	<b>IntP.</b> Reset: 1. 1=ThresholdIntType can be used to generate interrupts. 0=ThresholdIntType and interrupt generation are not supported. AccessType: (Core::X86::Msrr::HWCR[McStatusWrEn]   !MCA::LS::MCA_MISC0_LS[Locked]) ? Read-write : Read-only.
59:56	Reserved.
55:52	<b>LvtOffset.</b> Reset: 0h. One per die. 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). AccessType: (Core::X86::Msrr::HWCR[McStatusWrEn]   !MCA::LS::MCA_MISC0_LS[Locked]) ? Read-write : Read-only.
51	<b>CntEn.</b> Reset: 0. 1=Count thresholding errors. AccessType: (Core::X86::Msrr::HWCR[McStatusWrEn]   !MCA::LS::MCA_MISC0_LS[Locked]) ? Read-write : Read-only.
50:49	<b>ThresholdIntType.</b> Reset: Cold,0h. Specifies the type of interrupt signaled when Ovrflw is set and IntP==1. 00b = No interrupt. 01b = APIC based interrupt (see Core::X86::Msrr::McaIntrCfg[ThresholdLvtOffset]). 10b = SMI trigger event. 11b = Reserved. AccessType: (Core::X86::Msrr::HWCR[McStatusWrEn]   !MCA::LS::MCA_MISC0_LS[Locked]) ? Read-write : Read-only.
48	<b>Ovrflw.</b> Reset: Cold,0. Set by hardware when ErrCnt transitions from FFEh to FFFh. When this field is set, ErrCnt no longer increments. When this bit is set, the interrupt selected by the ThresholdIntType field is generated. AccessType: (Core::X86::Msrr::HWCR[McStatusWrEn]   !MCA::LS::MCA_MISC0_LS[Locked]) ? Read-write : Read-only.
47:44	Reserved.
43:32	<b>ErrCnt.</b> Reset: Cold,000h. This is written by software to set the starting value of the error counter. This is incremented by hardware when errors are logged. When this counter overflows, it stays at FFFh (no rollover). The threshold value, written by software, is (FFFh - the desired error count (the number of errors necessary in order for an interrupt to be taken)); the desired error count of 0 (a write value of FFFh) is not supported. AccessType: (Core::X86::Msrr::HWCR[McStatusWrEn]   !MCA::LS::MCA_MISC0_LS[Locked]) ? Read-write : Read-only.
31:24	<b>BlkPtr.</b> Read-write. Reset: 01h. 00h=Extended MISC MSR block is not valid. 01h=Extended MSR block is valid.
23:0	Reserved.

#### MSRC000\_2004 [LS Machine Check Configuration] (MCA::LS::MCA\_CONFIG\_LS)

Reset: 0000\_0002\_0000\_0025h.

Controls configuration of the associated machine check bank.

\_lthree0\_core[3:0]\_inst0\_aliasMSR; MSRC000\_2004

Bits	Description
63:39	Reserved.
38:37	<b>DeferredIntType.</b> Read-write. Reset: 0h. Specifies the type of interrupt signaled when a deferred error is logged.



	00b = No interrupt. 01b = APIC based interrupt (see Core::X86::Msr::McaIntrCfg[DeferredLvtOffset]). 10b = SMI trigger event. 11b = Reserved.
36:35	Reserved.
34	<b>LogDeferredInMcaStat.</b> Read-write. Reset: 0. Init: BIOS,1. 1=Log deferred errors in MCA::LS::MCA_STATUS_LS and MCA::LS::MCA_ADDR_LS in addition to MCA::LS::MCA_DESTAT_LS and MCA::LS::MCA_DEADDR_LS. 0=Only log deferred errors in MCA::LS::MCA_DESTAT_LS and MCA::LS::MCA_DEADDR_LS. This bit does not affect logging of deferred errors in MCA::LS::MCA_SYND_LS, MCA::LS::MCA_MISC0_LS.
33	Reserved.
32	<b>McaXEnable.</b> Read-write. Reset: 0. Init: BIOS,1. Check: 1. 1=Software has acknowledged support for the MCAX feature set. 0=Software has not acknowledged support for the MCAX feature set. All uncorrected and fatal errors will cause an ErrorEvent packet to be generated. Deferred error interrupts are configured via Core::X86::Msr::McaIntrCfg.
31:6	Reserved.
5	<b>DeferredIntTypeSupported.</b> Read-only. Reset: 1. 1=MCA::LS::MCA_CONFIG_LS[DeferredIntType] controls the type of interrupt generated on a deferred error. Deferred errors are supported in this bank only if MCA::LS::MCA_CONFIG_LS[DeferredErrorLoggingSupported]=1.
4:3	Reserved.
2	<b>DeferredErrorLoggingSupported.</b> Read-only. Reset: 1. 1=Deferred errors are supported in this MCA bank, and MCA::LS::MCA_CONFIG_LS[LogDeferredInMcaStat] controls the logging behavior of these errors. MCA::LS::MCA_DESTAT_LS and MCA::LS::MCA_DEADDR_LS are supported in this MCA bank. 0=Deferred errors are not supported in this bank.
1	Reserved.
0	<b>McaX.</b> Read-only. Reset: 1. 1=This bank provides Machine Check Architecture Extensions. Up to 4 additional MISC registers (MISC1-MISC4) are supported. MCA::LS::MCA_MISC0_LS[BlkPtr] indicates the presence of the additional MISC registers, but is not used to determine their MSR numbers. Deferred error interrupt type is specifiable by MCA bank. MCA::LS::MCA_STATUS_LS[TCC] is present.

#### MSRC000\_2005 [LS IP Identification] (MCA::LS::MCA\_IPID\_LS)

Reset: 0000\_00B0\_0000\_0000h.

The MCA::LS::MCA\_IPID\_LS register is used by software to determine what IP type and revision is associated with the MCA bank.

\_lthree0\_core[3:0]\_inst0\_aliasMSR; MSRC000\_2005

Bits	Description
63:48	<b>McaType.</b> Read-only. Reset: 0000h. The McaType of the MCA bank within this IP.
47:44	Reserved.
43:32	<b>HardwareID.</b> Read-only. Reset: 0B0h. The Hardware ID of the IP associated with this MCA bank.
31:0	<b>InstanceId.</b> Read-write. Reset: 0000_0000h. The instance ID of this IP. This is initialized to a unique ID per instance of this register.

#### MSRC000\_2006 [LS Machine Check Syndrome Thread 0] (MCA::LS::MCA\_SYND\_LS)

Read-write, Volatile. Reset: Cold, 0000\_0000\_0000\_0000h.

Logs physical location information associated with error in MCA::LS::MCA\_STATUS\_LS Thread 0

\_lthree0\_core[3:0]\_thread[1:0]\_inst0\_aliasMSR; MSRC000\_2006

Bits	Description
63:39	Reserved.
38:32	<b>Syndrom.</b> Read-write, Volatile. Reset: Cold, 00h. Contains the syndrome, if any, associated with the error logged in MCA::LS::MCA_STATUS_LS. The low-order bit of the syndrome is stored in bit 0, and the syndrome has a length specified by MCA::LS::MCA_SYND_LS[Length]. The Syndrome field is only valid when MCA::LS::MCA_SYND_LS[Length] is not 0.
31:27	Reserved.

26:24	<b>ErrorPriority.</b> Read-write, Volatile. Reset: Cold, 0h. Encodes the priority of the error logged in MCA::LS::MCA_SYND_LS. 3'b000 = No error; 3'b001 = Reserved; 3'b010 = Corrected Error; 3'b011 = Deferred Error; 3'b100 = Uncorrected Error; 3'b101 = Fatal Error; all others reserved.
23:18	<b>Length.</b> Read-write, Volatile. Reset: Cold, 00h. Specifies the length in bits of the syndrome contained in MCA::LS::MCA_SYND_LS[Syndrome]. A value of 0 indicates that there is no valid syndrome in MCA::LS::MCA_SYND_LS. For example, a syndrome length of 9 means that MCA::LS::MCA_SYND_LS[Syndrome] bits [8:0] contains a valid syndrome.
17:0	<b>ErrorInformation.</b> Read-write, Volatile. Reset: Cold, 0_0000h. Contains error-specific information about the location of the error. Decoding is available in Table 35 [MCA_SYND_LS Register].

Table 35: MCA\_SYND\_LS Register

Error Type	Bits	Description
LDQ	[17:0]	Reserved
STQ	[17:0]	Reserved
MAB	[17:0]	Reserved
L1DTLB	[17:0]	Reserved
DcTagErr5	[17:16] [15:8] [7:0]	Reserved Index Way
DcTagErr6	[17:16] [15:8] [7:0]	Reserved Index Way
DcTagErr1	[17:16] [15:8] [7:0]	Reserved Index Way
IntErrTyp1	[17] [16] [15:0]	Reserved Thread ID Reserved
IntErrTyp2	[17] [16] [15:0]	Reserved Thread ID Reserved
SystemReadDataErrorT0	[17:2] [1:0]	Reserved 2'b00 = Master Abort; 2'b01 = Target Abort; 2'b10 = Transaction Error; 2'b11 = Protection Violation
SystemReadDataErrorT1	[17:2] [1:0]	Reserved 2'b00 = Master Abort; 2'b01 = Target Abort; 2'b10 = Transaction Error; 2'b11 = Protection Violation
DcTagErr2	[17:16] [15:8] [7:0]	Reserved Index Way
DcDataErr1	[17:16] [15:8] [7:0]	Reserved Index Way
DcDataErr2	[17:16] [15:8] [7:0]	Reserved Index Way
DcDataErr3	[17:16] [15:8]	Reserved Index

	[7:0]	Way
DcTagErr4	[17:16] [15:8] [7:0]	Reserved Index Way
L2DTLB	[17:15] [14:8] [7:4] [3:0]	Reserved Reserved Reserved Reserved
PDC	[17:0]	Reserved
DcTagErr3	[17:16] [15:8] [7:0]	Reserved Index Way
DcTagErr5	[17:16] [15:8] [7:0]	Reserved Index Way
L2DataErr	[17:0]	Reserved

**MSRC000\_2008 [LS Machine Check Deferred Error Status Thread 0] (MCA::LS::MCA\_DESTAT\_LS)**

Read-write, Volatile. Reset: Cold, 0000\_0000\_0000\_0000h.

Holds status information for the first deferred error seen in this bank.

\_lthree0\_core[3:0]\_thread[1:0]\_inst0\_aliasMSR; MSRC000\_2008

Bits	Description
63	<b>Val.</b> Read-write, Volatile. Reset: Cold, 0. 1=A valid error has been detected (whether it is enabled or not).
62	<b>Overflow.</b> Read-write, Volatile. Reset: Cold, 0. 1=An error was detected while the valid bit (Val) was set; at least one error was not logged. Overflow is set independently of whether the existing error is overwritten. (See the section on overwrite priorities.)
61:59	Reserved.
58	<b>AddrV.</b> Read-write, Volatile. Reset: Cold, 0. 1=MCA::LS::MCA_DEADDR_LS contains address information associated with the error.
57:54	Reserved.
53	<b>SyndV.</b> Read-write, Volatile. Reset: Cold, 0. 1=This error logged information in MCA::LS::MCA_SYND_LS. If MCA::LS::MCA_SYND_LS[ErrorPriority] is the same as the priority of the error in MCA::LS::MCA_STATUS_LS, then the information in MCA::LS::MCA_SYND_LS is associated with the error in MCA::LS::MCA_DESTAT_LS.
52:45	Reserved.
44	<b>Deferred.</b> Read-write, Volatile. Reset: Cold, 0. 1=A deferred error was created. A deferred error is the result of an uncorrectable data error which did not immediately cause a processor exception; poison is created and an exception is deferred until the poison data is consumed.
43:0	Reserved.

**MSRC000\_2009 [LS Deferred Error Address Thread 0] (MCA::LS::MCA\_DEADDR\_LS)**

Reset: Cold, 0000\_0000\_0000\_0000h.

The MCA::LS::MCA\_DEADDR\_LS register stores the address associated with the error in MCA::LS::MCA\_DESTAT\_LS. The register is only meaningful if MCA::LS::MCA\_DESTAT\_LS[Val]=1 and MCA::LS::MCA\_DESTAT\_LS[AddrV]=1. The lowest valid bit of the address is defined by MCA::LS::MCA\_DEADDR\_LS[LSB].

\_lthree0\_core[3:0]\_thread[1:0]\_inst0\_aliasMSR; MSRC000\_2009

Bits	Description
63:62	Reserved.
61:56	<b>LSB.</b> Read-write, Volatile. Reset: Cold, 00h. Specifies the least significant valid bit of the address contained in

	MCA::LS::MCA_DEADDR_LS[ErrorAddr]. A value of 0 indicates that MCA::LS::MCA_DEADDR_LS[55:0] contains a valid byte address. A value of 6 indicates that MCA::LS::MCA_DEADDR_LS[55:6] contains a valid cache line address and that MCA::LS::MCA_DEADDR_LS[5:0] are not part of the address and should be ignored by error handling software. A value of 12 indicates that MCA::LS::MCA_DEADDR_LS[55:12] contain a valid 4KB memory page and that MCA::LS::MCA_DEADDR_LS[11:0] should be ignored by error handling software.
55:0	<b>ErrorAddr</b> . Read-write, Volatile. Reset: Cold, 00_0000_0000_0000h. Contains the address, if any, associated with the error logged in MCA::LS::MCA_DESTAT_LS. The lowest-order valid bit of the address is specified in MCA::LS::MCA_DEADDR_LS[LSB].

#### MSRC001\_0400 [LS Machine Check Control Mask] (MCA::LS::MCA\_CTL\_MASK\_LS)

Read-write. Reset: 0000\_0000\_0000\_0000h.

Inhibit detection of an error source.

\_lthree0\_core[3:0]\_inst0\_aliasMSR; MSRC001\_0400

Bits	Description
63:21	Reserved.
20	<b>L2DataErr</b> . Read-write. Reset: 0. L2 Fill Data error.
19	<b>DcTagErr5</b> . Read-write. Reset: 0. DC Tag error type 5.
18	<b>DcTagErr3</b> . Read-write. Reset: 0. DC Tag error type 3.
17	<b>PDC</b> . Read-write. Reset: 0. PDC parity error. MCA_ADDR_LS logs a virtual address.
16	<b>L2DTLB</b> . Read-write. Reset: 0. Level 2 TLB parity error. MCA_ADDR_LS logs a virtual address.
15	<b>DcTagErr4</b> . Read-write. Reset: 0. DC Tag error type 4.
14	<b>DcDataErr3</b> . Read-write. Reset: 0. DC Data error type 3.
13	<b>DcDataErr2</b> . Read-write. Reset: 0. DC Data error type 2.
12	<b>DcDataErr1</b> . Read-write. Reset: 0. DC Data error type 1 and poison consumption. MCA_STATUS[Poison] is set on poison consumption from L2/L3.
11	<b>DcTagErr2</b> . Read-write. Reset: 0. DC Tag error type 2.
10	<b>SystemReadDataErrorT1</b> . Read-write. Reset: 0. Init: BIOS, 1. System Read Data Error Thread 1. An error in a read of a line from the data fabric. Possible reasons include master abort and target abort.
9	<b>SystemReadDataErrorT0</b> . Read-write. Reset: 0. Init: BIOS, 1. System Read Data Error Thread 0. An error in a read of a line from the data fabric. Possible reasons include master abort and target abort.
8	<b>IntErrTyp2</b> . Read-write. Reset: 0. Internal error type 2.
7	<b>IntErrTyp1</b> . Read-write. Reset: 0. Internal error type 1.
6	<b>DcTagErr1</b> . Read-write. Reset: 0. DC Tag error type 1.
5	<b>DcTagErr6</b> . Read-write. Reset: 0. DC Tag error type 6.
4	Reserved.
3	<b>L1DTLB</b> . Read-write. Reset: 0. Level 1 TLB parity error.
2	<b>MAB</b> . Read-write. Reset: 0. Miss address buffer payload parity error.
1	<b>STQ</b> . Read-write. Reset: 0. Store queue parity error.
0	<b>LDQ</b> . Read-write. Reset: 0. Load queue parity error.

### 3.2.5.2 IF

#### MSR0000\_0404...MSRC000\_2010 [IF Machine Check Control] (MCA::IF::MCA\_CTL\_IF)

Read-write. Reset: 0000\_0000\_0000\_0000h.

0=Disables error reporting for the corresponding error. 1=Enables error reporting via machine check exception for the corresponding error. The MCA::IF::MCA\_CTL\_IF register must be enabled by the corresponding enable bit in Core::X86::Msr::MCG\_CTL. Does not affect error detection, correction, or logging.

\_lthree0\_core[3:0]\_inst1\_aliasMSRLEGACY; MSR0000\_0404

\_lthree0\_core[3:0]\_inst1\_aliasMSR; MSRC000\_2010

Bits	Description
63:14	Reserved.
13	<b>SystemReadDataError.</b> Read-write. Reset: 0. System Read Data Error. An error in a demand fetch of a line. Possible reasons include master abort and target abort.
12	<b>L2RespPoison.</b> Read-write. Reset: 0. L2 Cache Response Poison Error. Error is the result of consuming poison data.
11	<b>L2BtbMultiHit.</b> Read-write. Reset: 0. L2 BTB Multi-Match Error.
10	<b>L1BtbMultiHit.</b> Read-write. Reset: 0. L1 BTB Multi-Match Error.
9	<b>BpqSnpParT1.</b> Read-write. Reset: 0. BPQ Thread 1 Snoop Parity Error.
8	<b>BpqSnpParT0.</b> Read-write. Reset: 0. BPQ Thread 0 Snoop Parity Error.
7	<b>L2ItlbParity.</b> Read-write. Reset: 0. L2 ITLB Parity Error.
6	<b>L1ItlbParity.</b> Read-write. Reset: 0. L1 ITLB Parity Error.
5	<b>L0ItlbParity.</b> Read-write. Reset: 0. L0 ITLB Parity Error.
4	<b>DqParity.</b> Read-write. Reset: 0. Decoupling Queue PhysAddr Parity Error.
3	<b>DataParity.</b> Read-write. Reset: 0. IC Data Array Parity Error.
2	<b>TagParity.</b> Read-write. Reset: 0. IC Full Tag Parity Error.
1	<b>TagMultiHit.</b> Read-write. Reset: 0. IC Microtag or Full Tag Multi-hit Error.
0	<b>OcUtagParity.</b> Read-write. Reset: 0. Op Cache Microtag Probe Port Parity Error.

**MSR0000\_0405...MSRC000\_2011 [IF Machine Check Status Thread 0] (MCA::IF::MCA\_STATUS\_IF)**

Reset: Cold,0000\_0000\_0000\_0000h.

Logs information associated with errors.

\_lthree0\_core[3:0]\_thread[1:0]\_inst1\_aliasMSRLEGACY; MSR0000\_0405

\_lthree0\_core[3:0]\_thread[1:0]\_inst1\_aliasMSR; MSRC000\_2011

Bits	Description
63	<b>Val.</b> Reset: Cold,0. 1=A valid error has been detected. This bit should be cleared by software after the register has been read. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
62	<b>Overflow.</b> Reset: Cold,0. 1=An error was detected while the valid bit (Val) was set; at least one error was not logged. Overflow is set independently of whether the existing error is overwritten. See 3.1.3 [Machine Check Errors]. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
61	<b>UC.</b> Reset: Cold,0. 1=The error was not corrected by hardware. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
60	<b>En.</b> Reset: Cold,0. 1=MCA error reporting is enabled for this error, as indicated by the corresponding bit in MCA::IF::MCA_CTL_IF. This bit is a copy of bit in MCA::IF::MCA_CTL_IF for this error. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
59	<b>MiscV.</b> Reset: Cold,0. 1=Valid thresholding in MCA::IF::MCA_MISC0_IF. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
58	<b>AddrV.</b> Reset: Cold,0. 1=MCA::IF::MCA_ADDR_IF contains address information associated with the error. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
57	<b>PCC.</b> Reset: Cold,0. 1=Hardware context held by the processor may have been corrupted. Continued operation of the system may have unpredictable results. The error is not recoverable or survivable, and the system should be reinitialized. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
56	<b>ErrCoreIdVal.</b> Reset: Cold,0. 1=The ErrCoreId field is valid. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
55	<b>TCC.</b> Reset: Cold,0. 0=The thread which consumed the error can be restarted reliably at the instruction pointer address pushed onto the exception handler stack if any uncorrected error has been corrected by software. 1=The thread which consumed the error is not restartable and must be terminated. Only meaningful when

	MCA::IF::MCA_STATUS_IF[PCC]=0.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
54	Reserved.
53	<b>SyndV.</b> Reset: Cold,0. 1=This error logged information in MCA::IF::MCA_SYND_IF. If MCA::IF::MCA_SYND_IF[ErrorPriority] is the same as the priority of the error in MCA::IF::MCA_STATUS_IF, then the information in MCA::IF::MCA_SYND_IF is associated with the error in MCA::IF::MCA_STATUS_IF.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
52:47	Reserved.
46	<b>CECC.</b> Reset: Cold,0. 1=The error was a correctable ECC error according to the restrictions of the ECC algorithm. UC indicates whether the error was actually corrected by the processor.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
45	<b>UECC.</b> Reset: Cold,0. 1=The error was an uncorrectable ECC error according to the restrictions of the ECC algorithm. UC indicates whether the error was actually corrected by the processor.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
44	<b>Deferred.</b> Reset: Cold,0. 1=A deferred error was created. A deferred error is the result of an uncorrectable data error which did not immediately cause a processor exception; an exception is deferred until the erroneous data is consumed.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
43	<b>Poison.</b> Reset: Cold,0. 1=The error was the result of attempting to consume poisoned data.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
42:38	Reserved.
37:32	<b>ErrCoreId.</b> Reset: Cold,00h. When ErrCoreIdVal=1 this field indicates which core within the processor is associated with the error; Otherwise this field is reserved.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
31:22	Reserved.
21:16	<b>ErrorCodeExt.</b> Reset: Cold,00h. Extended Error Code. This field is used to identify the error type for root cause analysis. This field indicates which bit position in MCA::IF::MCA_CTL_IF enables error reporting for the logged error.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
15:0	<b>ErrorCode.</b> Reset: Cold,0000h. Error code for this error. See 3.1.3.3 [Error Codes] for details on decoding this field.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.

Table 36: MCA\_STATUS\_IF[ErrorCodeExt] Decode

Error Type	Bits	Description
OcUtagParity	[6:0]	0x0
TagMultiHit	[6:0]	0x1
TagParity	[6:0]	0x2
DataParity	[6:0]	0x3
DqParity	[6:0]	0x4
L0ItlbParity	[6:0]	0x5
L1ItlbParity	[6:0]	0x6
L2ItlbParity	[6:0]	0x7
BpqSnpParT0	[6:0]	0x8
BpqSnpParT1	[6:0]	0x9
L1BtbMultiHit	[6:0]	0xa
L2BtbMultiHit	[6:0]	0xb
L2RespPoison	[6:0]	0xc



SystemReadDataError	[6:0]	0xd
<b>MSR0000_0406...MSRC000_2012 [IF Machine Check Address Thread 0] (MCA::IF::MCA_ADDR_IF)</b>		
Reset: Cold,0000_0000_0000_0000h.		
MCA::IF::MCA_ADDR_IF stores an address and other information associated with the error in MCA::IF::MCA_STATUS_IF. The register is only meaningful if MCA::IF::MCA_STATUS_IF[Val]=1 and MCA::IF::MCA_STATUS_IF[AddrV]=1.		
_lthree0_core[3:0]_thread[1:0]_inst1_aliasMSRLEGACY; MSR0000_0406		
_lthree0_core[3:0]_thread[1:0]_inst1_aliasMSR; MSRC000_2012		
Bits	Description	
63:62	Reserved.	
61:56	<b>LSB.</b> Read-write, Volatile. Reset: Cold,00h. Specifies the least significant valid bit of the address contained in MCA::IF::MCA_ADDR_IF[ErrorAddr]. A value of 0 indicates that MCA::IF::MCA_ADDR_IF[55:0] contains a valid byte address. A value of 6 indicates that MCA::IF::MCA_ADDR_IF[55:6] contains a valid cache line address and that MCA::IF::MCA_ADDR_IF[5:0] are not part of the address and should be ignored by error handling software. A value of 12 indicates that MCA::IF::MCA_ADDR_IF[55:12] contain a valid 4KB memory page and that MCA::IF::MCA_ADDR_IF[11:0] should be ignored by error handling software.	
55:0	<b>ErrorAddr.</b> Read-write, Volatile. Reset: Cold,00_0000_0000_0000h. Unless otherwise specified by an error, contains the address associated with the error logged in MCA::IF::MCA_STATUS_IF. For physical addresses, the most significant bit is given by Core::X86::Cpuid::LongModeInfo[PhysAddrSize].	

Table 37: MCA\_ADDR\_IF Register

Error Type	Bits	Description
OcUtagParity	[55:0]	Reserved
TagMultiHit	[55:48] [47:0]	Reserved Physical Address
TagParity	[55:48] [47:0]	Reserved Physical Address
DataParity	[55:48] [47:0]	Reserved Physical Address
DqParity	[55:48] [47:0]	Reserved Physical Address
L0ItlbParity	[55:48] [47:12] [11:0]	Reserved Linear Address Reserved
L1ItlbParity	[55:48] [47:12] [11:0]	Reserved Linear Address Reserved
L2ItlbParity	[55:48] [47:12] [11:0]	Reserved Linear Address Reserved
BpqSnpParT0	[55:0]	Reserved
BpqSnpParT1	[55:0]	Reserved
L1BtbMultiHit	[55:0]	Reserved
L2BtbMultiHit	[55:0]	Reserved
L2RespPoison	[55:48] [47:5] [4:0]	Reserved Physical Address Reserved
SystemReadDataError	[55:48]	Reserved



	[47:5]	Physical Address
	[4:0]	Reserved

**MSR0000\_0407...MSRC000\_2013 [IF Machine Check Miscellaneous 0 Thread 0] (MCA::IF::MCA\_MISC0\_IF)**

Log miscellaneous information associated with errors.

\_lthree0\_core[3:0]\_thread[1:0]\_inst1\_aliasMSRLEGACY; MSR0000\_0407

\_lthree0\_core[3:0]\_thread[1:0]\_inst1\_aliasMSR; MSRC000\_2013

Bits	Description
63	<b>Valid.</b> Reset: 1. 1=A valid CntP field is present in this register. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read-only.
62	<b>CntP.</b> Reset: 1. 1=A valid threshold counter is present. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read-only.
61	<b>Locked.</b> Reset: 0. 1=Writes to this register are ignored. This bit is set by BIOS to indicate that this register is not available for OS use. BIOS should set this bit if ThresholdIntType is set to SMI. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read-only.
60	<b>IntP.</b> Reset: 1. 1=ThresholdIntType can be used to generate interrupts. 0=ThresholdIntType and interrupt generation are not supported. AccessType: (Core::X86::Msr::HWCR[McStatusWrEn]   !MCA::IF::MCA_MISC0_IF[Locked]) ? Read-write : Read-only.
59:56	Reserved.
55:52	<b>LvtOffset.</b> Reset: 0h. One per die. 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). AccessType: (Core::X86::Msr::HWCR[McStatusWrEn]   !MCA::IF::MCA_MISC0_IF[Locked]) ? Read-write : Read-only.
51	<b>CntEn.</b> Reset: 0. 1=Count thresholding errors. AccessType: (Core::X86::Msr::HWCR[McStatusWrEn]   !MCA::IF::MCA_MISC0_IF[Locked]) ? Read-write : Read-only.
50:49	<b>ThresholdIntType.</b> Reset: Cold,0h. Specifies the type of interrupt signaled when Ovrlw is set and IntP==1. 00b = No interrupt. 01b = APIC based interrupt (see Core::X86::Msr::McaIntrCfg[ThresholdLvtOffset]). 10b = SMI trigger event. 11b = Reserved. AccessType: (Core::X86::Msr::HWCR[McStatusWrEn]   !MCA::IF::MCA_MISC0_IF[Locked]) ? Read-write : Read-only.
48	<b>Ovrlw.</b> Reset: Cold,0. Set by hardware when ErrCnt transitions from FFEh to FFFh. When this field is set, ErrCnt no longer increments. When this bit is set, the interrupt selected by the ThresholdIntType field is generated. AccessType: (Core::X86::Msr::HWCR[McStatusWrEn]   !MCA::IF::MCA_MISC0_IF[Locked]) ? Read-write : Read-only.
47:44	Reserved.
43:32	<b>ErrCnt.</b> Reset: Cold,000h. This is written by software to set the starting value of the error counter. This is incremented by hardware when errors are logged. When this counter overflows, it stays at FFFh (no rollover). The threshold value, written by software, is (FFFh - the desired error count (the number of errors necessary in order for an interrupt to be taken)); the desired error count of 0 (a write value of FFFh) is not supported. AccessType: (Core::X86::Msr::HWCR[McStatusWrEn]   !MCA::IF::MCA_MISC0_IF[Locked]) ? Read-write : Read-only.
31:24	<b>BlkPtr.</b> Read-write. Reset: 01h. 00h=Extended MISC MSR block is not valid. 01h=Extended MSR block is valid.
23:0	Reserved.

**MSRC000\_2014 [IF Machine Check Configuration] (MCA::IF::MCA\_CONFIG\_IF)**

Reset: 0000\_0002\_0000\_0021h.

Controls configuration of the associated machine check bank.

_lthree0_core[3:0]_inst1_aliasMSR; MSRC000_2014	
Bits	Description
63:39	Reserved.
38:37	<b>DeferredIntType</b> . Read-write. Reset: 0h. Specifies the type of interrupt signaled when a deferred error is logged. 00b = No interrupt. 01b = APIC based interrupt (see Core::X86::Msr::McaIntrCfg[DeferredLvtOffset]). 10b = SMI trigger event. 11b = Reserved.
36:33	Reserved.
32	<b>McaXEnable</b> . Read-write. Reset: 0. Init: BIOS,1. Check: 1. 1=Software has acknowledged support for the MCAX feature set. 0=Software has not acknowledged support for the MCAX feature set. All uncorrected and fatal errors will cause an ErrorEvent packet to be generated. Deferred error interrupts are configured via Core::X86::Msr::McaIntrCfg.
31:6	Reserved.
5	<b>DeferredIntTypeSupported</b> . Read-only. Reset: 1. 1=MCA::IF::MCA_CONFIG_IF[DeferredIntType] controls the type of interrupt generated on a deferred error. Deferred errors are supported in this bank only if MCA::IF::MCA_CONFIG_IF[DeferredErrorLoggingSupported]=1.
4:3	Reserved.
2	<b>DeferredErrorLoggingSupported</b> . Read-only. Reset: 0. 1=Deferred errors are supported in this MCA bank, and the LogDeferredInMcaStat field in this register controls the logging behavior of these errors. MCA_DESTAT and MCA_DEADDR are supported in this MCA bank. 0=Deferred errors are not supported in this bank.
1	Reserved.
0	<b>McaX</b> . Read-only. Reset: 1. 1=This bank provides Machine Check Architecture Extensions. Up to 4 additional MISC registers (MISC1-MISC4) are supported. MCA::IF::MCA_MISC0_IF[BlkPtr] indicates the presence of the additional MISC registers, but is not used to determine their MSR numbers. Deferred error interrupt type is specifiable by MCA bank. MCA::IF::MCA_STATUS_IF[TCC] is present.

#### MSRC000\_2015 [IF IP Identification] (MCA::IF::MCA\_IPID\_IF)

Reset: 0001\_00B0\_0000\_0000h.

The MCA::IF::MCA\_IPID\_IF register is used by software to determine what IP type and revision is associated with the MCA bank.

_lthree0_core[3:0]_inst1_aliasMSR; MSRC000_2015	
Bits	Description
63:48	<b>McaType</b> . Read-only. Reset: 0001h. The McaType of the MCA bank within this IP.
47:44	Reserved.
43:32	<b>HardwareID</b> . Read-only. Reset: 0B0h. The Hardware ID of the IP associated with this MCA bank.
31:0	<b>InstanceId</b> . Read-write. Reset: 0000_0000h. The instance ID of this IP. This is initialized to a unique ID per instance of this register.

#### MSRC000\_2016 [IF Machine Check Syndrome Thread 0] (MCA::IF::MCA\_SYND\_IF)

Read-write, Volatile. Reset: Cold, 0000\_0000\_0000\_0000h.

Logs physical location information associated with error in MCA::IF::MCA\_STATUS\_IF Thread 0

_lthree0_core[3:0]_thread[1:0]_inst1_aliasMSR; MSRC000_2016	
Bits	Description
63:33	Reserved.
32	<b>Syndrom</b> . Read-write, Volatile. Reset: Cold, 0. Contains the syndrome, if any, associated with the error logged in MCA::IF::MCA_STATUS_IF. The low-order bit of the syndrome is stored in bit 0, and the syndrome has a length specified by MCA::IF::MCA_SYND_IF[Length]. The Syndrome field is only valid when MCA::IF::MCA_SYND_IF[Length] is not 0.
31:27	Reserved.
26:24	<b>ErrorPriority</b> . Read-write, Volatile. Reset: Cold, 0h. Encodes the priority of the error logged in MCA::IF::MCA_SYND_IF. 3'b000 = No error; 3'b001 = Reserved; 3'b010 = Corrected Error; 3'b011 = Deferred Error; 3'b100 = Uncorrected Error; 3'b101 = Fatal Error; all others reserved.

23:18	<b>Length.</b> Read-write, Volatile. Reset: Cold, 00h. Specifies the length in bits of the syndrome contained in MCA::IF::MCA_SYND_IF[Syndrome]. A value of 0 indicates that there is no valid syndrome in MCA::IF::MCA_SYND_IF. For example, a syndrome length of 9 means that MCA::IF::MCA_SYND_IF[Syndrome] bits [8:0] contains a valid syndrome.
17:0	<b>ErrorInformation.</b> Read-write, Volatile. Reset: Cold, 0_0000h. Contains error-specific information about the location of the error. Decoding is available in Table 38 [MCA_SYND_IF Register].

Table 38: MCA\_SYND\_IF Register

Error Type	Bits	Description
OcUtagParity	[17:5] [4:0]	Reserved Index
TagMultiHit	[17:16] [15:8] [8:0]	Reserved Subcache Reserved
TagParity	[17:8] [7:0]	Reserved Way
DataParity	[17:16] [15:8] [8:0]	Reserved Subcache Way
DqParity	[17:0]	Reserved
L0ItlbParity	[17:4] [3:0]	Reserved Reserved
L1ItlbParity	[17:6] [5:0]	Reserved Reserved
L2ItlbParity	[17:8] [7:0]	Reserved Reserved
BpqSnpParT0	[17:0]	Reserved
BpqSnpParT1	[17:0]	Reserved
L1BtbMultiHit	[17:0]	Reserved
L2BtbMultiHit	[17:0]	Reserved
L2RespPoison	[17:0]	Reserved
SystemReadDataError	[17:2] [1:0]	Reserved 2'b00 = Master Abort; 2'b01 = Target Abort; 2'b10 = Transaction Error; 2'b11 = Protection Violation

**MSRC001\_0401 [IF Machine Check Control Mask] (MCA::IF::MCA\_CTL\_MASK\_IF)**

Read-write. Reset: 0000\_0000\_0000\_0000h.

Inhibit detection of an error source.

\_lthree0\_core[3:0]\_inst1\_aliasMSR; MSRC001\_0401

Bits	Description
63:14	Reserved.
13	<b>SystemReadDataError.</b> Read-write. Reset: 0. Init: BIOS, 1. System Read Data Error. An error in a demand fetch of a line. Possible reasons include master abort and target abort.
12	<b>L2RespPoison.</b> Read-write. Reset: 0. L2 Cache Response Poison Error. Error is the result of consuming poison data.
11	<b>L2BtbMultiHit.</b> Read-write. Reset: 0. L2 BTB Multi-Match Error.
10	<b>L1BtbMultiHit.</b> Read-write. Reset: 0. L1 BTB Multi-Match Error.
9	<b>BpqSnpParT1.</b> Read-write. Reset: 0. BPQ Thread 1 Snoop Parity Error.
8	<b>BpqSnpParT0.</b> Read-write. Reset: 0. BPQ Thread 0 Snoop Parity Error.

7	<b>L2ItlbParity.</b> Read-write. Reset: 0. L2 ITLB Parity Error.
6	<b>L1ItlbParity.</b> Read-write. Reset: 0. L1 ITLB Parity Error.
5	<b>L0ItlbParity.</b> Read-write. Reset: 0. L0 ITLB Parity Error.
4	<b>DqParity.</b> Read-write. Reset: 0. Decoupling Queue PhysAddr Parity Error.
3	<b>DataParity.</b> Read-write. Reset: 0. IC Data Array Parity Error.
2	<b>TagParity.</b> Read-write. Reset: 0. IC Full Tag Parity Error.
1	<b>TagMultiHit.</b> Read-write. Reset: 0. IC Microtag or Full Tag Multi-hit Error.
0	<b>OcUtagParity.</b> Read-write. Reset: 0. Op Cache Microtag Probe Port Parity Error.

### 3.2.5.3 L2

#### MSR0000\_0408...MSRC000\_2020 [L2 Machine Check Control] (MCA::L2::MCA\_CTL\_L2)

Read-write. Reset: 0000\_0000\_0000\_0000h.

0=Disables error reporting for the corresponding error. 1=Enables error reporting via machine check exception for the corresponding error. The MCA::L2::MCA\_CTL\_L2 register must be enabled by the corresponding enable bit in Core::X86::Msr::MCG\_CTL. Does not affect error detection, correction, or logging.

\_lthree0\_core[3:0]\_inst2\_aliasMSRLEGACY; MSR0000\_0408

\_lthree0\_core[3:0]\_inst2\_aliasMSR; MSRC000\_2020

Bits	Description
63:4	Reserved.
3	<b>Hwa.</b> Read-write. Reset: 0. Hardware Assert Error.
2	<b>Data.</b> Read-write. Reset: 0. L2M Data Array ECC Error.
1	<b>Tag.</b> Read-write. Reset: 0. L2M Tag or State Array ECC Error.
0	<b>MultiHit.</b> Read-write. Reset: 0. L2M Tag Multiple-Way-Hit error.

#### MSR0000\_0409...MSRC000\_2021 [L2 Machine Check Status Thread 0] (MCA::L2::MCA\_STATUS\_L2)

Reset: Cold,0000\_0000\_0000\_0000h.

Logs information associated with errors.

\_lthree0\_core[3:0]\_thread[1:0]\_inst2\_aliasMSRLEGACY; MSR0000\_0409

\_lthree0\_core[3:0]\_thread[1:0]\_inst2\_aliasMSR; MSRC000\_2021

Bits	Description
63	<b>Val.</b> Reset: Cold,0. 1=A valid error has been detected. This bit should be cleared by software after the register has been read. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
62	<b>Overflow.</b> Reset: Cold,0. 1=An error was detected while the valid bit (Val) was set; at least one error was not logged. Overflow is set independently of whether the existing error is overwritten. See 3.1.3 [Machine Check Errors]. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
61	<b>UC.</b> Reset: Cold,0. 1=The error was not corrected by hardware. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
60	<b>En.</b> Reset: Cold,0. 1=MCA error reporting is enabled for this error, as indicated by the corresponding bit in MCA::L2::MCA_CTL_L2. This bit is a copy of bit in MCA::L2::MCA_CTL_L2 for this error. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
59	<b>MiscV.</b> Reset: Cold,0. 1=Valid thresholding in MCA::L2::MCA_MISC0_L2. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
58	<b>AddrV.</b> Reset: Cold,0. 1=MCA::L2::MCA_ADDR_L2 contains address information associated with the error. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
57	<b>PCC.</b> Reset: Cold,0. 1=Hardware context held by the processor may have been corrupted. Continued operation of the system may have unpredictable results. The error is not recoverable or survivable, and the system should be reinitialized.

	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
56	<b>ErrCoreIdVal.</b> Reset: Cold,0. 1=The ErrCoreId field is valid.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
55	<b>TCC.</b> Reset: Cold,0. 0=The thread which consumed the error can be restarted reliably at the instruction pointer address pushed onto the exception handler stack if any uncorrected error has been corrected by software. 1=The thread which consumed the error is not restartable and must be terminated. Only meaningful when MCA::L2::MCA_STATUS_L2[PCC]=0.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
54	Reserved.
53	<b>SyndV.</b> Reset: Cold,0. 1=This error logged information in MCA::L2::MCA_SYND_L2. If MCA::L2::MCA_SYND_L2[ErrorPriority] is the same as the priority of the error in MCA::L2::MCA_STATUS_L2, then the information in MCA::L2::MCA_SYND_L2 is associated with the error in MCA::L2::MCA_STATUS_L2.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
52:47	Reserved.
46	<b>CECC.</b> Reset: Cold,0. 1=The error was a correctable ECC error according to the restrictions of the ECC algorithm. UC indicates whether the error was actually corrected by the processor.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
45	<b>UECC.</b> Reset: Cold,0. 1=The error was an uncorrectable ECC error according to the restrictions of the ECC algorithm. UC indicates whether the error was actually corrected by the processor.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
44	<b>Deferred.</b> Reset: Cold,0. 1=A deferred error was created. A deferred error is the result of an uncorrectable data error which did not immediately cause a processor exception; an exception is deferred until the erroneous data is consumed.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
43	<b>Poison.</b> Reset: Cold,0. 1=The error was the result of attempting to consume poisoned data.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
42:38	Reserved.
37:32	<b>ErrCoreId.</b> Reset: Cold,00h. When ErrCoreIdVal=1 this field indicates which core within the processor is associated with the error; Otherwise this field is reserved.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
31:22	Reserved.
21:16	<b>ErrorCodeExt.</b> Reset: Cold,00h. Extended Error Code. This field is used to identify the error type for root cause analysis. This field indicates which bit position in MCA::L2::MCA_CTL_L2 enables error reporting for the logged error.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
15:0	<b>ErrorCode.</b> Reset: Cold,0000h. Error code for this error. See 3.1.3.3 [Error Codes] for details on decoding this field.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.

Table 39: MCA\_STATUS\_L2[ErrorCodeExt] Decode

Error Type	Bits	Description
MultiHit	[6:0]	0x0
Tag	[6:0]	0x1
Data	[6:0]	0x2
Hwa	[6:0]	0x3

**MSR0000\_040A...MSRC000\_2022 [L2 Machine Check Address Thread 0] (MCA::L2::MCA\_ADDR\_L2)**

Reset: Cold,0000\_0000\_0000\_0000h.

MCA::L2::MCA\_ADDR\_L2 stores an address and other information associated with the error in

MCA::L2::MCA_STATUS_L2. The register is only meaningful if MCA::L2::MCA_STATUS_L2[Val]=1 and MCA::L2::MCA_STATUS_L2[AddrV]=1.	
_lthree0_core[3:0]_thread[1:0]_inst2_aliasMSRLEGACY; MSR0000_040A	
_lthree0_core[3:0]_thread[1:0]_inst2_aliasMSR; MSRC000_2022	
Bits	Description
63:62	Reserved.
61:56	<b>LSB.</b> Read-write, Volatile. Reset: Cold, 00h. Specifies the least significant valid bit of the address contained in MCA::L2::MCA_ADDR_L2[ErrorAddr]. A value of 0 indicates that MCA::L2::MCA_ADDR_L2[55:0] contains a valid byte address. A value of 6 indicates that MCA::L2::MCA_ADDR_L2[55:6] contains a valid cache line address and that MCA::L2::MCA_ADDR_L2[5:0] are not part of the address and should be ignored by error handling software. A value of 12 indicates that MCA::L2::MCA_ADDR_L2[55:12] contain a valid 4KB memory page and that MCA::L2::MCA_ADDR_L2[11:0] should be ignored by error handling software.
55:0	<b>ErrorAddr.</b> Read-write, Volatile. Reset: Cold, 00_0000_0000_0000h. Unless otherwise specified by an error, contains the address associated with the error logged in MCA::L2::MCA_STATUS_L2. For physical addresses, the most significant bit is given by Core::X86::CpuId::LongModeInfo[PhysAddrSize].

Table 40: MCA\_ADDR\_L2 Register

Error Type	Bits	Description
MultiHit	[55:48]	Reserved
	[47:6]	Physical Address
	[5:0]	Reserved
Tag	[55:48]	Reserved
	[47:6]	Physical Address
	[5:0]	Reserved
Data	[55:48]	Reserved
	[47:6]	Physical Address
	[5:0]	Reserved
Hwa	[31:0]	Reserved

**MSR0000\_040B...MSRC000\_2023 [L2 Machine Check Miscellaneous 0 Thread 0] (MCA::L2::MCA\_MISC0\_L2)**

Log miscellaneous information associated with errors.	
_lthree0_core[3:0]_thread[1:0]_inst2_aliasMSRLEGACY; MSR0000_040B	
_lthree0_core[3:0]_thread[1:0]_inst2_aliasMSR; MSRC000_2023	
Bits	Description
63	<b>Valid.</b> Reset: 1. 1=A valid CntP field is present in this register. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read-only.
62	<b>CntP.</b> Reset: 1. 1=A valid threshold counter is present. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read-only.
61	<b>Locked.</b> Reset: 0. 1=Writes to this register are ignored. This bit is set by BIOS to indicate that this register is not available for OS use. BIOS should set this bit if ThresholdIntType is set to SMI. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read-only.
60	<b>IntP.</b> Reset: 1. 1=ThresholdIntType can be used to generate interrupts. 0=ThresholdIntType and interrupt generation are not supported. AccessType: (Core::X86::Msr::HWCR[McStatusWrEn]   !MCA::L2::MCA_MISC0_L2[Locked]) ? Read-write : Read-only.
59:56	Reserved.
55:52	<b>LvtOffset.</b> Reset: 0h. One per die. 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). AccessType: (Core::X86::Msr::HWCR[McStatusWrEn]   !MCA::L2::MCA_MISC0_L2[Locked]) ? Read-write :



	Read-only.
51	<b>CntEn.</b> Reset: 0. 1=Count thresholding errors. AccessType: (Core::X86::Msr::HWCR[McStatusWrEn]   !MCA::L2::MCA_MISC0_L2[Locked]) ? Read-write : Read-only.
50:49	<b>ThresholdIntType.</b> Reset: Cold,0h. Specifies the type of interrupt signaled when Ovrflw is set and IntP==1. 00b = No interrupt. 01b = APIC based interrupt (see Core::X86::Msr::McaIntrCfg[ThresholdLvtOffset]). 10b = SMI trigger event. 11b = Reserved. AccessType: (Core::X86::Msr::HWCR[McStatusWrEn]   !MCA::L2::MCA_MISC0_L2[Locked]) ? Read-write : Read-only.
48	<b>Ovrflw.</b> Reset: Cold,0. Set by hardware when ErrCnt transitions from FFEh to FFFh. When this field is set, ErrCnt no longer increments. When this bit is set, the interrupt selected by the ThresholdIntType field is generated. AccessType: (Core::X86::Msr::HWCR[McStatusWrEn]   !MCA::L2::MCA_MISC0_L2[Locked]) ? Read-write : Read-only.
47:44	Reserved.
43:32	<b>ErrCnt.</b> Reset: Cold,000h. This is written by software to set the starting value of the error counter. This is incremented by hardware when errors are logged. When this counter overflows, it stays at FFFh (no rollover). The threshold value, written by software, is (FFFh - the desired error count (the number of errors necessary in order for an interrupt to be taken)); the desired error count of 0 (a write value of FFFh) is not supported. AccessType: (Core::X86::Msr::HWCR[McStatusWrEn]   !MCA::L2::MCA_MISC0_L2[Locked]) ? Read-write : Read-only.
31:24	<b>BlkPtr.</b> Read-write. Reset: 01h. 00h=Extended MISC MSR block is not valid. 01h=Extended MSR block is valid.
23:0	Reserved.

#### MSRC000\_2024 [L2 Machine Check Configuration] (MCA::L2::MCA\_CONFIG\_L2)

Reset: 0000\_0000\_0000\_0025h.

Controls configuration of the associated machine check bank.

\_lthree0\_core[3:0]\_inst2\_aliasMSR; MSRC000\_2024

Bits	Description
63:39	Reserved.
38:37	<b>DeferredIntType.</b> Read-write. Reset: 0h. Specifies the type of interrupt signaled when a deferred error is logged. 00b = No interrupt. 01b = APIC based interrupt (see Core::X86::Msr::McaIntrCfg[DeferredLvtOffset]). 10b = SMI trigger event. 11b = Reserved.
36:35	Reserved.
34	<b>LogDeferredInMcaStat.</b> Read-write. Reset: 0. Init: BIOS,1. 1=Log deferred errors in MCA::L2::MCA_STATUS_L2 and MCA::L2::MCA_ADDR_L2 in addition to MCA::L2::MCA_DESTAT_L2 and MCA::L2::MCA_DEADDR_L2. 0=Only log deferred errors in MCA::L2::MCA_DESTAT_L2 and MCA::L2::MCA_DEADDR_L2. This bit does not affect logging of deferred errors in MCA::L2::MCA_SYND_L2, MCA::L2::MCA_MISC0_L2.
33	Reserved.
32	<b>McaXEnable.</b> Read-write. Reset: 0. Init: BIOS,1. Check: 1. 1=Software has acknowledged support for the MCAX feature set. 0=Software has not acknowledged support for the MCAX feature set. All uncorrected and fatal errors will cause an ErrorEvent packet to be generated. Deferred error interrupts are configured via Core::X86::Msr::McaIntrCfg.
31:6	Reserved.
5	<b>DeferredIntTypeSupported.</b> Read-only. Reset: 1. 1=MCA::L2::MCA_CONFIG_L2[DeferredIntType] controls the type of interrupt generated on a deferred error. Deferred errors are supported in this bank only if MCA::L2::MCA_CONFIG_L2[DeferredErrorLoggingSupported]=1.
4:3	Reserved.
2	<b>DeferredErrorLoggingSupported.</b> Read-only. Reset: 1. 1=Deferred errors are supported in this MCA bank, and MCA::L2::MCA_CONFIG_L2[LogDeferredInMcaStat] controls the logging behavior of these errors.



	MCA::L2::MCA_DESTAT_L2 and MCA::L2::MCA_DEADDR_L2 are supported in this MCA bank. 0=Deferred errors are not supported in this bank.
1	Reserved.
0	<b>McaX.</b> Read-only. Reset: 1. 1=This bank provides Machine Check Architecture Extensions. Up to 4 additional MISC registers (MISC1-MISC4) are supported. MCA::L2::MCA_MISC0_L2[BlkPtr] indicates the presence of the additional MISC registers, but is not used to determine their MSR numbers. Deferred error interrupt type is specifiable by MCA bank. MCA::L2::MCA_STATUS_L2[TCC] is present.

**MSRC000\_2025 [L2 IP Identification] (MCA::L2::MCA\_IPID\_L2)**

Reset: 0002\_00B0\_0000\_0000h.

The MCA::L2::MCA\_IPID\_L2 register is used by software to determine what IP type and revision is associated with the MCA bank.

\_lthree0\_core[3:0]\_inst2\_aliasMSR; MSRC000\_2025

Bits	Description
63:48	<b>McaType.</b> Read-only. Reset: 0002h. The McaType of the MCA bank within this IP.
47:44	Reserved.
43:32	<b>HardwareID.</b> Read-only. Reset: 0B0h. The Hardware ID of the IP associated with this MCA bank.
31:0	<b>InstanceId.</b> Read-write. Reset: 0000_0000h. The instance ID of this IP. This is initialized to a unique ID per instance of this register.

**MSRC000\_2026 [L2 Machine Check Syndrome Thread 0] (MCA::L2::MCA\_SYND\_L2)**

Read-write, Volatile. Reset: Cold, 0000\_0000\_0000\_0000h.

Logs physical location information associated with error in MCA::L2::MCA\_STATUS\_L2 Thread 0

\_lthree0\_core[3:0]\_thread[1:0]\_inst2\_aliasMSR; MSRC000\_2026

Bits	Description
63:49	Reserved.
48:32	<b>Syndrom.</b> Read-write, Volatile. Reset: Cold, 0_0000h. Contains the syndrome, if any, associated with the error logged in MCA::L2::MCA_STATUS_L2. The low-order bit of the syndrome is stored in bit 0, and the syndrome has a length specified by MCA::L2::MCA_SYND_L2[Length]. The Syndrome field is only valid when MCA::L2::MCA_SYND_L2[Length] is not 0.
31:27	Reserved.
26:24	<b>ErrorPriority.</b> Read-write, Volatile. Reset: Cold, 0h. Encodes the priority of the error logged in MCA::L2::MCA_SYND_L2. 3'b000 = No error; 3'b001 = Reserved; 3'b010 = Corrected Error; 3'b011 = Deferred Error; 3'b100 = Uncorrected Error; 3'b101 = Fatal Error; all others reserved.
23:18	<b>Length.</b> Read-write, Volatile. Reset: Cold, 00h. Specifies the length in bits of the syndrome contained in MCA::L2::MCA_SYND_L2[Syndrome]. A value of 0 indicates that there is no valid syndrome in MCA::L2::MCA_SYND_L2. For example, a syndrome length of 9 means that MCA::L2::MCA_SYND_L2[Syndrome] bits [8:0] contains a valid syndrome.
17:0	<b>ErrorInformation.</b> Read-write, Volatile. Reset: Cold, 0_0000h. Contains error-specific information about the location of the error. Decoding is available in Table 41 [MCA_SYND_L2 Register].

Table 41: MCA\_SYND\_L2 Register

Error Type	Bits	Description
MultiHit	[17:8] [7:0]	Index One-hot way vector
Tag	[17:13] [12:3] [2:0]	Reserved Index Way
Data	[17:15] [14:5]	Reserved Index

	[4:3]	Quarter-line
	[2:0]	Way
Hwa	[17:0]	Reserved

**MSRC000\_2028 [L2 Machine Check Deferred Error Status Thread 0] (MCA::L2::MCA\_DESTAT\_L2)**

Read-write, Volatile. Reset: Cold, 0000\_0000\_0000\_0000h.

Holds status information for the first deferred error seen in this bank.

\_lthree0\_core[3:0]\_thread[1:0]\_inst2\_aliasMSR; MSRC000\_2028

Bits	Description
63	<b>Val.</b> Read-write, Volatile. Reset: Cold, 0. 1=A valid error has been detected (whether it is enabled or not).
62	<b>Overflow.</b> Read-write, Volatile. Reset: Cold, 0. 1=An error was detected while the valid bit (Val) was set; at least one error was not logged. Overflow is set independently of whether the existing error is overwritten. (See the section on overwrite priorities.)
61:59	Reserved.
58	<b>AddrV.</b> Read-write, Volatile. Reset: Cold, 0. 1=MCA::L2::MCA_DEADDR_L2 contains address information associated with the error.
57:54	Reserved.
53	<b>SyndV.</b> Read-write, Volatile. Reset: Cold, 0. 1=This error logged information in MCA::L2::MCA_SYND_L2. If MCA::L2::MCA_SYND_L2[ErrorPriority] is the same as the priority of the error in MCA::L2::MCA_STATUS_L2, then the information in MCA::L2::MCA_SYND_L2 is associated with the error in MCA::L2::MCA_DESTAT_L2.
52:45	Reserved.
44	<b>Deferred.</b> Read-write, Volatile. Reset: Cold, 0. 1=A deferred error was created. A deferred error is the result of an uncorrectable data error which did not immediately cause a processor exception; poison is created and an exception is deferred until the poison data is consumed.
43:0	Reserved.

**MSRC000\_2029 [L2 Deferred Error Address Thread 0] (MCA::L2::MCA\_DEADDR\_L2)**

Reset: Cold, 0000\_0000\_0000\_0000h.

The MCA::L2::MCA\_DEADDR\_L2 register stores the address associated with the error in MCA::L2::MCA\_DESTAT\_L2. The register is only meaningful if MCA::L2::MCA\_DESTAT\_L2[Val]=1 and MCA::L2::MCA\_DESTAT\_L2[AddrV]=1. The lowest valid bit of the address is defined by MCA::L2::MCA\_DEADDR\_L2[LSB].

\_lthree0\_core[3:0]\_thread[1:0]\_inst2\_aliasMSR; MSRC000\_2029

Bits	Description
63:62	Reserved.
61:56	<b>LSB.</b> Read-write, Volatile. Reset: Cold, 00h. Specifies the least significant valid bit of the address contained in MCA::L2::MCA_DEADDR_L2[ErrorAddr]. A value of 0 indicates that MCA::L2::MCA_DEADDR_L2[55:0] contains a valid byte address. A value of 6 indicates that MCA::L2::MCA_DEADDR_L2[55:6] contains a valid cache line address and that MCA::L2::MCA_DEADDR_L2[5:0] are not part of the address and should be ignored by error handling software. A value of 12 indicates that MCA::L2::MCA_DEADDR_L2[55:12] contain a valid 4KB memory page and that MCA::L2::MCA_DEADDR_L2[11:0] should be ignored by error handling software.
55:0	<b>ErrorAddr.</b> Read-write, Volatile. Reset: Cold, 00_0000_0000_0000h. Contains the address, if any, associated with the error logged in MCA::L2::MCA_DESTAT_L2. The lowest-order valid bit of the address is specified in MCA::L2::MCA_DEADDR_L2[LSB].

**MSRC001\_0402 [L2 Machine Check Control Mask] (MCA::L2::MCA\_CTL\_MASK\_L2)**

Read-write. Reset: 0000\_0000\_0000\_0000h.

Inhibit detection of an error source.

\_lthree0\_core[3:0]\_inst2\_aliasMSR; MSRC001\_0402

Bits	Description
63:4	Reserved.

3	<b>Hwa.</b> Read-write. Reset: 0. Init: BIOS,1. Hardware Assert Error.
2	<b>Data.</b> Read-write. Reset: 0. L2M Data Array ECC Error.
1	<b>Tag.</b> Read-write. Reset: 0. L2M Tag or State Array ECC Error.
0	<b>MultiHit.</b> Read-write. Reset: 0. L2M Tag Multiple-Way-Hit error.

### 3.2.5.4 DE

#### MSR0000\_040C...MSRC000\_2030 [DE Machine Check Control] (MCA::DE::MCA\_CTL\_DE)

Read-write. Reset: 0000\_0000\_0000\_0000h.

0=Disables error reporting for the corresponding error. 1=Enables error reporting via machine check exception for the corresponding error. The MCA::DE::MCA\_CTL\_DE register must be enabled by the corresponding enable bit in Core::X86::Msr::MCG\_CTL. Does not affect error detection, correction, or logging.

\_lthree0\_core[3:0]\_inst3\_aliasMSRLEGACY; MSR0000\_040C

\_lthree0\_core[3:0]\_inst3\_aliasMSR; MSRC000\_2030

Bits	Description
63:9	Reserved.
8	<b>OCBQ.</b> Read-write. Reset: 0. Micro-op buffer parity error.
7	<b>UcSeq.</b> Read-write. Reset: 0. Patch RAM sequencer parity error.
6	<b>UcDat.</b> Read-write. Reset: 0. Patch RAM data parity error.
5	<b>Faq.</b> Read-write. Reset: 0. Fetch address FIFO parity error.
4	<b>Idq.</b> Read-write. Reset: 0. Instruction dispatch queue parity error.
3	<b>UopQ.</b> Read-write. Reset: 0. Micro-op queue parity error.
2	<b>Ibq.</b> Read-write. Reset: 0. Instruction buffer parity error.
1	<b>OcDat.</b> Read-write. Reset: 0. Micro-op cache data parity error.
0	<b>OcTag.</b> Read-write. Reset: 0. Micro-op cache tag parity error.

#### MSR0000\_040D...MSRC000\_2031 [DE Machine Check Status Thread 0] (MCA::DE::MCA\_STATUS\_DE)

Reset: Cold,0000\_0000\_0000\_0000h.

Logs information associated with errors.

\_lthree0\_core[3:0]\_thread[1:0]\_inst3\_aliasMSRLEGACY; MSR0000\_040D

\_lthree0\_core[3:0]\_thread[1:0]\_inst3\_aliasMSR; MSRC000\_2031

Bits	Description
63	<b>Val.</b> Reset: Cold,0. 1=A valid error has been detected. This bit should be cleared by software after the register has been read. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
62	<b>Overflow.</b> Reset: Cold,0. 1=An error was detected while the valid bit (Val) was set; at least one error was not logged. Overflow is set independently of whether the existing error is overwritten. See 3.1.3 [Machine Check Errors]. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
61	<b>UC.</b> Reset: Cold,0. 1=The error was not corrected by hardware. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
60	<b>En.</b> Reset: Cold,0. 1=MCA error reporting is enabled for this error, as indicated by the corresponding bit in MCA::DE::MCA_CTL_DE. This bit is a copy of bit in MCA::DE::MCA_CTL_DE for this error. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
59	<b>MiscV.</b> Reset: Cold,0. 1=Valid thresholding in MCA::DE::MCA_MISC0_DE. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
58	<b>AddrV.</b> Reset: Cold,0. 1=MCA::DE::MCA_ADDR_DE contains address information associated with the error. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
57	<b>PCC.</b> Reset: Cold,0. 1=Hardware context held by the processor may have been corrupted. Continued operation of the system may have unpredictable results. The error is not recoverable or survivable, and the system should be

	reinitialized.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
56	<b>ErrCoreIdVal.</b> Reset: Cold,0. 1=The ErrCoreId field is valid.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
55	<b>TCC.</b> Reset: Cold,0. 0=The thread which consumed the error can be restarted reliably at the instruction pointer address pushed onto the exception handler stack if any uncorrected error has been corrected by software. 1=The thread which consumed the error is not restartable and must be terminated. Only meaningful when MCA::DE::MCA_STATUS_DE[PCC]=0.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
54	Reserved.
53	<b>SyndV.</b> Reset: Cold,0. 1=This error logged information in MCA::DE::MCA_SYND_DE. If MCA::DE::MCA_SYND_DE[ErrorPriority] is the same as the priority of the error in MCA::DE::MCA_STATUS_DE, then the information in MCA::DE::MCA_SYND_DE is associated with the error in MCA::DE::MCA_STATUS_DE.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
52:47	Reserved.
46	<b>CECC.</b> Reset: Cold,0. 1=The error was a correctable ECC error according to the restrictions of the ECC algorithm. UC indicates whether the error was actually corrected by the processor.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
45	<b>UECC.</b> Reset: Cold,0. 1=The error was an uncorrectable ECC error according to the restrictions of the ECC algorithm. UC indicates whether the error was actually corrected by the processor.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
44	<b>Deferred.</b> Reset: Cold,0. 1=A deferred error was created. A deferred error is the result of an uncorrectable data error which did not immediately cause a processor exception; an exception is deferred until the erroneous data is consumed.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
43	<b>Poison.</b> Reset: Cold,0. 1=The error was the result of attempting to consume poisoned data.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
42:38	Reserved.
37:32	<b>ErrCoreId.</b> Reset: Cold,00h. When ErrCoreIdVal=1 this field indicates which core within the processor is associated with the error; Otherwise this field is reserved.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
31:22	Reserved.
21:16	<b>ErrorCodeExt.</b> Reset: Cold,00h. Extended Error Code. This field is used to identify the error type for root cause analysis. This field indicates which bit position in MCA::DE::MCA_CTL_DE enables error reporting for the logged error.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
15:0	<b>ErrorCode.</b> Reset: Cold,0000h. Error code for this error. See 3.1.3.3 [Error Codes] for details on decoding this field.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.

Table 42: MCA\_STATUS\_DE[ErrorCodeExt] Decode

Error Type	Bits	Description
OcTag	[6:0]	0x0
OcDat	[6:0]	0x1
Ibq	[6:0]	0x2
UopQ	[6:0]	0x3
Idq	[6:0]	0x4
Faq	[6:0]	0x5

UcDat	[6:0]	0x6
UcSeq	[6:0]	0x7
OCBQ	[6:0]	0x8

**MSR0000\_040E...MSRC000\_2032 [DE Machine Check Address Thread 0] (MCA::DE::MCA\_ADDR\_DE)**

Reset: Cold,0000\_0000\_0000\_0000h.

MCA::DE::MCA\_ADDR\_DE stores an address and other information associated with the error in MCA::DE::MCA\_STATUS\_DE. The register is only meaningful if MCA::DE::MCA\_STATUS\_DE[Val]=1 and MCA::DE::MCA\_STATUS\_DE[AddrV]=1.

\_lthree0\_core[3:0]\_thread[1:0]\_inst3\_aliasMSRLEGACY; MSR0000\_040E

\_lthree0\_core[3:0]\_thread[1:0]\_inst3\_aliasMSR; MSRC000\_2032

Bits	Description
63:62	Reserved.
61:56	<b>LSB.</b> Read-write, Volatile. Reset: Cold,00h. Specifies the least significant valid bit of the address contained in MCA::DE::MCA_ADDR_DE[ErrorAddr]. A value of 0 indicates that MCA::DE::MCA_ADDR_DE[55:0] contains a valid byte address. A value of 6 indicates that MCA::DE::MCA_ADDR_DE[55:6] contains a valid cache line address and that MCA::DE::MCA_ADDR_DE[5:0] are not part of the address and should be ignored by error handling software. A value of 12 indicates that MCA::DE::MCA_ADDR_DE[55:12] contain a valid 4KB memory page and that MCA::DE::MCA_ADDR_DE[11:0] should be ignored by error handling software.
55:0	<b>ErrorAddr.</b> Read-write, Volatile. Reset: Cold,00_0000_0000_0000h. Unless otherwise specified by an error, contains the address associated with the error logged in MCA::DE::MCA_STATUS_DE. For physical addresses, the most significant bit is given by Core::X86::Cpuid::LongModeInfo[PhysAddrSize].

Table 43: MCA\_ADDR\_DE Register

Error Type	Bits	Description
OcTag	[55:0]	Reserved
OcDat	[55:0]	Reserved
Ibq	[55:0]	Reserved
UopQ	[55:0]	Reserved
Idq	[55:0]	Reserved
Faq	[55:0]	Reserved
UcDat	[55:0]	Reserved
UcSeq	[55:0]	Reserved
OCBQ	[55:0]	Reserved

**MSR0000\_040F...MSRC000\_2033 [DE Machine Check Miscellaneous 0 Thread 0] (MCA::DE::MCA\_MISC0\_DE)**

Log miscellaneous information associated with errors.

\_lthree0\_core[3:0]\_thread[1:0]\_inst3\_aliasMSRLEGACY; MSR0000\_040F

\_lthree0\_core[3:0]\_thread[1:0]\_inst3\_aliasMSR; MSRC000\_2033

Bits	Description
63	<b>Valid.</b> Reset: 1. 1=A valid CntP field is present in this register. AccessType: Core::X86::Msrr::HWCR[McStatusWrEn] ? Read-write : Read-only.
62	<b>CntP.</b> Reset: 1. 1=A valid threshold counter is present. AccessType: Core::X86::Msrr::HWCR[McStatusWrEn] ? Read-write : Read-only.
61	<b>Locked.</b> Reset: 0. 1=Writes to this register are ignored. This bit is set by BIOS to indicate that this register is not available for OS use. BIOS should set this bit if ThresholdIntType is set to SMI. AccessType: Core::X86::Msrr::HWCR[McStatusWrEn] ? Read-write : Read-only.
60	<b>IntP.</b> Reset: 1. 1=ThresholdIntType can be used to generate interrupts. 0=ThresholdIntType and interrupt generation are not supported. AccessType: (Core::X86::Msrr::HWCR[McStatusWrEn]   !MCA::DE::MCA_MISC0_DE[Locked]) ? Read-write :

	Read-only.
59:56	Reserved.
55:52	<b>LvtOffset.</b> Reset: 0h. One per die. 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). AccessType: (Core::X86::Msr::HWCR[McStatusWrEn]   !MCA::DE::MCA_MISC0_DE[Locked]) ? Read-write : Read-only.
51	<b>CntEn.</b> Reset: 0. 1=Count thresholding errors. AccessType: (Core::X86::Msr::HWCR[McStatusWrEn]   !MCA::DE::MCA_MISC0_DE[Locked]) ? Read-write : Read-only.
50:49	<b>ThresholdIntType.</b> Reset: Cold,0h. Specifies the type of interrupt signaled when Ovrflw is set and IntP==1. 00b = No interrupt. 01b = APIC based interrupt (see Core::X86::Msr::McaIntrCfg[ThresholdLvtOffset]). 10b = SMI trigger event. 11b = Reserved. AccessType: (Core::X86::Msr::HWCR[McStatusWrEn]   !MCA::DE::MCA_MISC0_DE[Locked]) ? Read-write : Read-only.
48	<b>Ovrflw.</b> Reset: Cold,0. Set by hardware when ErrCnt transitions from FFEh to FFFh. When this field is set, ErrCnt no longer increments. When this bit is set, the interrupt selected by the ThresholdIntType field is generated. AccessType: (Core::X86::Msr::HWCR[McStatusWrEn]   !MCA::DE::MCA_MISC0_DE[Locked]) ? Read-write : Read-only.
47:44	Reserved.
43:32	<b>ErrCnt.</b> Reset: Cold,000h. This is written by software to set the starting value of the error counter. This is incremented by hardware when errors are logged. When this counter overflows, it stays at FFFh (no rollover). The threshold value, written by software, is (FFFh - the desired error count (the number of errors necessary in order for an interrupt to be taken)); the desired error count of 0 (a write value of FFFh) is not supported. AccessType: (Core::X86::Msr::HWCR[McStatusWrEn]   !MCA::DE::MCA_MISC0_DE[Locked]) ? Read-write : Read-only.
31:24	<b>BlkPtr.</b> Read-write. Reset: 01h. 00h=Extended MISC MSR block is not valid. 01h=Extended MSR block is valid.
23:0	Reserved.

**MSRC000\_2034 [DE Machine Check Configuration] (MCA::DE::MCA\_CONFIG\_DE)**

Reset: 0000\_0002\_0000\_0021h.

Controls configuration of the associated machine check bank.

\_lthree0\_core[3:0]\_inst3\_aliasMSR; MSRC000\_2034

Bits	Description
63:39	Reserved.
38:37	<b>DeferredIntType.</b> Read-write. Reset: 0h. Specifies the type of interrupt signaled when a deferred error is logged. 00b = No interrupt. 01b = APIC based interrupt (see Core::X86::Msr::McaIntrCfg[DeferredLvtOffset]). 10b = SMI trigger event. 11b = Reserved.
36:33	Reserved.
32	<b>McaXEnable.</b> Read-write. Reset: 0. Init: BIOS,1. Check: 1. 1=Software has acknowledged support for the MCAX feature set. 0=Software has not acknowledged support for the MCAX feature set. All uncorrected and fatal errors will cause an ErrorEvent packet to be generated. Deferred error interrupts are configured via Core::X86::Msr::McaIntrCfg.
31:6	Reserved.
5	<b>DeferredIntTypeSupported.</b> Read-only. Reset: 1. 1=MCA::DE::MCA_CONFIG_DE[DeferredIntType] controls the type of interrupt generated on a deferred error. Deferred errors are supported in this bank only if MCA::DE::MCA_CONFIG_DE[DeferredErrorLoggingSupported]=1.
4:3	Reserved.
2	<b>DeferredErrorLoggingSupported.</b> Read-only. Reset: 0. 1=Deferred errors are supported in this MCA bank, and the LogDeferredInMcaStat field in this register controls the logging behavior of these errors. MCA_DESTAT and



	MCA_DEADDR are supported in this MCA bank. 0=Deferred errors are not supported in this bank.
1	Reserved.
0	<b>McaX.</b> Read-only. Reset: 1. 1=This bank provides Machine Check Architecture Extensions. Up to 4 additional MISC registers (MISC1-MISC4) are supported. MCA::DE::MCA_MISC0_DE[BlkPtr] indicates the presence of the additional MISC registers, but is not used to determine their MSR numbers. Deferred error interrupt type is specifiable by MCA bank. MCA::DE::MCA_STATUS_DE[TCC] is present.

**MSRC000\_2035 [DE IP Identification] (MCA::DE::MCA\_IPID\_DE)**

Reset: 0003\_00B0\_0000\_0000h.

The MCA::DE::MCA\_IPID\_DE register is used by software to determine what IP type and revision is associated with the MCA bank.

\_lthree0\_core[3:0]\_inst3\_aliasMSR; MSRC000\_2035

Bits	Description
63:48	<b>McaType.</b> Read-only. Reset: 0003h. The McaType of the MCA bank within this IP.
47:44	Reserved.
43:32	<b>HardwareID.</b> Read-only. Reset: 0B0h. The Hardware ID of the IP associated with this MCA bank.
31:0	<b>InstanceId.</b> Read-write. Reset: 0000_0000h. The instance ID of this IP. This is initialized to a unique ID per instance of this register.

**MSRC000\_2036 [DE Machine Check Syndrome Thread 0] (MCA::DE::MCA\_SYND\_DE)**

Read-write, Volatile. Reset: Cold, 0000\_0000\_0000\_0000h.

Logs physical location information associated with error in MCA::DE::MCA\_STATUS\_DE Thread 0

\_lthree0\_core[3:0]\_thread[1:0]\_inst3\_aliasMSR; MSRC000\_2036

Bits	Description
63:33	Reserved.
32	<b>Syndrom.</b> Read-write, Volatile. Reset: Cold, 0. Contains the syndrome, if any, associated with the error logged in MCA::DE::MCA_STATUS_DE. The low-order bit of the syndrome is stored in bit 0, and the syndrome has a length specified by MCA::DE::MCA_SYND_DE[Length]. The Syndrome field is only valid when MCA::DE::MCA_SYND_DE[Length] is not 0.
31:27	Reserved.
26:24	<b>ErrorPriority.</b> Read-write, Volatile. Reset: Cold, 0h. Encodes the priority of the error logged in MCA::DE::MCA_SYND_DE. 3'b000 = No error; 3'b001 = Reserved; 3'b010 = Corrected Error; 3'b011 = Deferred Error; 3'b100 = Uncorrected Error; 3'b101 = Fatal Error; all others reserved.
23:18	<b>Length.</b> Read-write, Volatile. Reset: Cold, 00h. Specifies the length in bits of the syndrome contained in MCA::DE::MCA_SYND_DE[Syndrome]. A value of 0 indicates that there is no valid syndrome in MCA::DE::MCA_SYND_DE. For example, a syndrome length of 9 means that MCA::DE::MCA_SYND_DE[Syndrome] bits [8:0] contains a valid syndrome.
17:0	<b>ErrorInformation.</b> Read-write, Volatile. Reset: Cold, 0_0000h. Contains error-specific information about the location of the error. Decoding is available in Table 44 [MCA_SYND_DE Register].

Table 44: MCA\_SYND\_DE Register

Error Type	Bits	Description
OcTag	[17:16] [15:8] [7:0]	Reserved Index Way
OcDat	[17:16] [15:8] [7:0]	Reserved Index Way
Ibq	[17:0]	Reserved
UopQ	[17:0]	Reserved



Idq	[17:0]	Reserved
Faq	[17:0]	Reserved
UcDat	[17:0]	Reserved
UcSeq	[17:0]	Reserved
OCBQ	[17:0]	Reserved

#### MSRC001\_0403 [DE Machine Check Control Mask] (MCA::DE::MCA\_CTL\_MASK\_DE)

Read-write. Reset: 0000\_0000\_0000\_0000h.

Inhibit detection of an error source.

\_lthree0\_core[3:0]\_inst3\_aliasMSR; MSRC001\_0403

Bits	Description
63:9	Reserved.
8	<b>OCBQ</b> . Read-write. Reset: 0. Micro-op buffer parity error.
7	<b>UcSeq</b> . Read-write. Reset: 0. Patch RAM sequencer parity error.
6	<b>UcDat</b> . Read-write. Reset: 0. Patch RAM data parity error.
5	<b>Faq</b> . Read-write. Reset: 0. Fetch address FIFO parity error.
4	<b>Idq</b> . Read-write. Reset: 0. Instruction dispatch queue parity error.
3	<b>UopQ</b> . Read-write. Reset: 0. Micro-op queue parity error.
2	<b>Ibq</b> . Read-write. Reset: 0. Instruction buffer parity error.
1	<b>OcDat</b> . Read-write. Reset: 0. Micro-op cache data parity error.
0	<b>OcTag</b> . Read-write. Reset: 0. Micro-op cache tag parity error.

### 3.2.5.5 EX

#### MSR0000\_0414...MSRC000\_2050 [EX Machine Check Control] (MCA::EX::MCA\_CTL\_EX)

Read-write. Reset: 0000\_0000\_0000\_0000h.

0=Disables error reporting for the corresponding error. 1=Enables error reporting via machine check exception for the corresponding error. The MCA::EX::MCA\_CTL\_EX register must be enabled by the corresponding enable bit in Core::X86::Msr::MCG\_CTL. Does not affect error detection, correction, or logging.

\_lthree0\_core[3:0]\_inst5\_aliasMSRLEGACY; MSR0000\_0414

\_lthree0\_core[3:0]\_inst5\_aliasMSR; MSRC000\_2050

Bits	Description
63:11	Reserved.
10	<b>BBQ</b> . Read-write. Reset: 0. Branch buffer queue parity error.
9	<b>SQ</b> . Read-write. Reset: 0. Scheduling queue parity error.
8	<b>STATQ</b> . Read-write. Reset: 0. Retire status queue parity error.
7	<b>RETDISP</b> . Read-write. Reset: 0. Retire dispatch queue parity error.
6	<b>CHKPTQ</b> . Read-write. Reset: 0. CHKPTQ. Checkpoint queue parity error.
5	<b>PLDAL</b> . Read-write. Reset: 0. EX payload parity error.
4	<b>PLDAG</b> . Read-write. Reset: 0. Address generator payload parity error.
3	<b>IDRF</b> . Read-write. Reset: 0. Immediate displacement register file parity error.
2	<b>FRF</b> . Read-write. Reset: 0. Flag register file parity error.
1	<b>PRF</b> . Read-write. Reset: 0. Physical register file parity error.
0	<b>WDT</b> . Read-write. Reset: 0. Watchdog Timeout error.

#### MSR0000\_0415...MSRC000\_2051 [EX Machine Check Status Thread 0] (MCA::EX::MCA\_STATUS\_EX)

Reset: Cold,0000\_0000\_0000\_0000h.

Logs information associated with errors.

\_lthree0\_core[3:0]\_thread[1:0]\_inst5\_aliasMSRLEGACY; MSR0000\_0415

\_lthree0\_core[3:0]\_thread[1:0]\_inst5\_aliasMSR; MSRC000\_2051

Bits	Description
63	<b>Val.</b> Reset: Cold,0. 1=A valid error has been detected. This bit should be cleared by software after the register has been read. AccessType: Core::X86::Msrr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
62	<b>Overflow.</b> Reset: Cold,0. 1=An error was detected while the valid bit (Val) was set; at least one error was not logged. Overflow is set independently of whether the existing error is overwritten. See 3.1.3 [Machine Check Errors]. AccessType: Core::X86::Msrr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
61	<b>UC.</b> Reset: Cold,0. 1=The error was not corrected by hardware. AccessType: Core::X86::Msrr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
60	<b>En.</b> Reset: Cold,0. 1=MCA error reporting is enabled for this error, as indicated by the corresponding bit in MCA::EX::MCA_CTL_EX. This bit is a copy of bit in MCA::EX::MCA_CTL_EX for this error. AccessType: Core::X86::Msrr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
59	<b>MiscV.</b> Reset: Cold,0. 1=Valid thresholding in MCA::EX::MCA_MISC0_EX. AccessType: Core::X86::Msrr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
58	<b>AddrV.</b> Reset: Cold,0. 1=MCA::EX::MCA_ADDR_EX contains address information associated with the error. AccessType: Core::X86::Msrr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
57	<b>PCC.</b> Reset: Cold,0. 1=Hardware context held by the processor may have been corrupted. Continued operation of the system may have unpredictable results. The error is not recoverable or survivable, and the system should be reinitialized. AccessType: Core::X86::Msrr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
56	<b>ErrCoreIdVal.</b> Reset: Cold,0. 1=The ErrCoreId field is valid. AccessType: Core::X86::Msrr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
55	<b>TCC.</b> Reset: Cold,0. 0=The thread which consumed the error can be restarted reliably at the instruction pointer address pushed onto the exception handler stack if any uncorrected error has been corrected by software. 1=The thread which consumed the error is not restartable and must be terminated. Only meaningful when MCA::EX::MCA_STATUS_EX[PCC]=0. AccessType: Core::X86::Msrr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
54	Reserved.
53	<b>SyndV.</b> Reset: Cold,0. 1=This error logged information in MCA::EX::MCA_SYND_EX. If MCA::EX::MCA_SYND_EX[ErrorPriority] is the same as the priority of the error in MCA::EX::MCA_STATUS_EX, then the information in MCA::EX::MCA_SYND_EX is associated with the error in MCA::EX::MCA_STATUS_EX. AccessType: Core::X86::Msrr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
52:47	Reserved.
46	<b>CECC.</b> Reset: Cold,0. 1=The error was a correctable ECC error according to the restrictions of the ECC algorithm. UC indicates whether the error was actually corrected by the processor. AccessType: Core::X86::Msrr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
45	<b>UECC.</b> Reset: Cold,0. 1=The error was an uncorrectable ECC error according to the restrictions of the ECC algorithm. UC indicates whether the error was actually corrected by the processor. AccessType: Core::X86::Msrr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
44	<b>Deferred.</b> Reset: Cold,0. 1=A deferred error was created. A deferred error is the result of an uncorrectable data error which did not immediately cause a processor exception; an exception is deferred until the erroneous data is consumed. AccessType: Core::X86::Msrr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
43	<b>Poison.</b> Reset: Cold,0. 1=The error was the result of attempting to consume poisoned data. AccessType: Core::X86::Msrr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
42:38	Reserved.
37:32	<b>ErrCoreId.</b> Reset: Cold,00h. When ErrCoreIdVal=1 this field indicates which core within the processor is associated with the error; Otherwise this field is reserved.

	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
31:22	Reserved.
21:16	<b>ErrorCodeExt.</b> Reset: Cold,00h. Extended Error Code. This field is used to identify the error type for root cause analysis. This field indicates which bit position in MCA::EX::MCA_CTL_EX enables error reporting for the logged error.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
15:0	<b>ErrorCode.</b> Reset: Cold,0000h. Error code for this error. See 3.1.3.3 [Error Codes] for details on decoding this field.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.

Table 45: MCA\_STATUS\_EX[ErrorCodeExt] Decode

Error Type	Bits	Description
WDT	[6:0]	0x0
PRF	[6:0]	0x1
FRF	[6:0]	0x2
IDRF	[6:0]	0x3
PLDAG	[6:0]	0x4
PLDAL	[6:0]	0x5
CHKPTQ	[6:0]	0x6
RETDISP	[6:0]	0x7
STATQ	[6:0]	0x8
SQ	[6:0]	0x9
BBQ	[6:0]	0xa

**MSR0000\_0416...MSRC000\_2052 [EX Machine Check Address Thread 0] (MCA::EX::MCA\_ADDR\_EX)**

Read-only. Reset: Cold,0000_0000_0000_0000h.	
MCA::EX::MCA_ADDR_EX stores an address and other information associated with the error in MCA::EX::MCA_STATUS_EX. The register is only meaningful if MCA::EX::MCA_STATUS_EX[Val]=1 and MCA::EX::MCA_STATUS_EX[AddrV]=1.	
_lthree0_core[3:0]_thread[1:0]_inst5_aliasMSRLEGACY; MSR0000_0416	
_lthree0_core[3:0]_thread[1:0]_inst5_aliasMSR; MSRC000_2052	
Bits	Description
63:62	Reserved.
61:56	<b>LSB.</b> Read-only. Reset: Cold,00h. Specifies the least significant valid bit of the address contained in MCA::EX::MCA_ADDR_EX[ErrorAddr]. A value of 0 indicates that MCA::EX::MCA_ADDR_EX[55:0] contains a valid byte address. A value of 6 indicates that MCA::EX::MCA_ADDR_EX[55:6] contains a valid cache line address and that MCA::EX::MCA_ADDR_EX[5:0] are not part of the address and should be ignored by error handling software. A value of 12 indicates that MCA::EX::MCA_ADDR_EX[55:12] contain a valid 4KB memory page and that MCA::EX::MCA_ADDR_EX[11:0] should be ignored by error handling software.
55:0	<b>ErrorAddr.</b> Read-only. Reset: Cold,00_0000_0000_0000h. Contains the address, if any, associated with the error logged in MCA::EX::MCA_STATUS_EX.

Table 46: MCA\_ADDR\_EX Register

Error Type	Bits	Description
WDT	55:49] [48:0]	Reserved RIP of thread triggering the watchdog timeout
PRF	[55:0]	Reserved
FRF	[55:0]	Reserved
IDRF	[55:0]	Reserved

PLDAG	[55:0]	Reserved
PLDAL	[55:0]	Reserved
CHKPTQ	[55:0]	Reserved
RETDISP	[55:0]	Reserved
STATQ	[55:0]	Reserved
SQ	[55:0]	Reserved
BBQ	[55:0]	Reserved

#### MSR0000\_0417...MSRC000\_2053 [EX Machine Check Miscellaneous 0 Thread 0] (MCA::EX::MCA\_MISC0\_EX)

Log miscellaneous information associated with errors.

\_lthree0\_core[3:0]\_thread[1:0]\_inst5\_aliasMSRLEGACY; MSR0000\_0417

\_lthree0\_core[3:0]\_thread[1:0]\_inst5\_aliasMSR; MSRC000\_2053

Bits	Description
63	<b>Valid.</b> Reset: 1. 1=A valid CntP field is present in this register. AccessType: Core::X86::Msrr::HWCR[McStatusWrEn] ? Read-write : Read-only.
62	<b>CntP.</b> Reset: 1. 1=A valid threshold counter is present. AccessType: Core::X86::Msrr::HWCR[McStatusWrEn] ? Read-write : Read-only.
61	<b>Locked.</b> Reset: 0. 1=Writes to this register are ignored. This bit is set by BIOS to indicate that this register is not available for OS use. BIOS should set this bit if ThresholdIntType is set to SMI. AccessType: Core::X86::Msrr::HWCR[McStatusWrEn] ? Read-write : Read-only.
60	<b>IntP.</b> Reset: 1. 1=ThresholdIntType can be used to generate interrupts. 0=ThresholdIntType and interrupt generation are not supported. AccessType: (Core::X86::Msrr::HWCR[McStatusWrEn]   !MCA::EX::MCA_MISC0_EX[Locked]) ? Read-write : Read-only.
59:56	Reserved.
55:52	<b>LvtOffset.</b> Reset: 0h. One per die. 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). AccessType: (Core::X86::Msrr::HWCR[McStatusWrEn]   !MCA::EX::MCA_MISC0_EX[Locked]) ? Read-write : Read-only.
51	<b>CntEn.</b> Reset: 0. 1=Count thresholding errors. AccessType: (Core::X86::Msrr::HWCR[McStatusWrEn]   !MCA::EX::MCA_MISC0_EX[Locked]) ? Read-write : Read-only.
50:49	<b>ThresholdIntType.</b> Reset: Cold,0h. Specifies the type of interrupt signaled when Ovrflw is set and IntP==1. 00b = No interrupt. 01b = APIC based interrupt (see Core::X86::Msrr::McaIntrCfg[ThresholdLvtOffset]). 10b = SMI trigger event. 11b = Reserved. AccessType: (Core::X86::Msrr::HWCR[McStatusWrEn]   !MCA::EX::MCA_MISC0_EX[Locked]) ? Read-write : Read-only.
48	<b>Ovrflw.</b> Reset: Cold,0. Set by hardware when ErrCnt transitions from FFEh to FFFh. When this field is set, ErrCnt no longer increments. When this bit is set, the interrupt selected by the ThresholdIntType field is generated. AccessType: (Core::X86::Msrr::HWCR[McStatusWrEn]   !MCA::EX::MCA_MISC0_EX[Locked]) ? Read-write : Read-only.
47:44	Reserved.
43:32	<b>ErrCnt.</b> Reset: Cold,000h. This is written by software to set the starting value of the error counter. This is incremented by hardware when errors are logged. When this counter overflows, it stays at FFFh (no rollover). The threshold value, written by software, is (FFFh - the desired error count (the number of errors necessary in order for an interrupt to be taken)); the desired error count of 0 (a write value of FFFh) is not supported. AccessType: (Core::X86::Msrr::HWCR[McStatusWrEn]   !MCA::EX::MCA_MISC0_EX[Locked]) ? Read-write : Read-only.

31:24	<b>BlkPtr.</b> Read-write. Reset: 01h. 00h=Extended MISC MSR block is not valid. 01h=Extended MSR block is valid.
23:0	Reserved.

**MSRC000\_2054 [EX Machine Check Configuration] (MCA::EX::MCA\_CONFIG\_EX)**

Reset: 0000\_0002\_0000\_0021h.

Controls configuration of the associated machine check bank.

\_lthree0\_core[3:0]\_inst5\_aliasMSR; MSRC000\_2054

Bits	Description
63:39	Reserved.
38:37	<b>DeferredIntType.</b> Read-write. Reset: 0h. Specifies the type of interrupt signaled when a deferred error is logged. 00b = No interrupt. 01b = APIC based interrupt (see Core::X86::Msr::McaIntrCfg[DeferredLvtOffset]). 10b = SMI trigger event. 11b = Reserved.
36:33	Reserved.
32	<b>McaXEnable.</b> Read-write. Reset: 0. Init: BIOS,1. Check: 1. 1=Software has acknowledged support for the MCAX feature set. 0=Software has not acknowledged support for the MCAX feature set. All uncorrected and fatal errors will cause an ErrorEvent packet to be generated. Deferred error interrupts are configured via Core::X86::Msr::McaIntrCfg.
31:6	Reserved.
5	<b>DeferredIntTypeSupported.</b> Read-only. Reset: 1. 1=MCA::EX::MCA_CONFIG_EX[DeferredIntType] controls the type of interrupt generated on a deferred error. Deferred errors are supported in this bank only if MCA::EX::MCA_CONFIG_EX[DeferredErrorLoggingSupported]=1.
4:3	Reserved.
2	<b>DeferredErrorLoggingSupported.</b> Read-only. Reset: 0. 1=Deferred errors are supported in this MCA bank, and the LogDeferredInMcaStat field in this register controls the logging behavior of these errors. MCA_DESTAT and MCA_DEADDR are supported in this MCA bank. 0=Deferred errors are not supported in this bank.
1	Reserved.
0	<b>McaX.</b> Read-only. Reset: 1. 1=This bank provides Machine Check Architecture Extensions. Up to 4 additional MISC registers (MISC1-MISC4) are supported. MCA::EX::MCA_MISC0_EX[BlkPtr] indicates the presence of the additional MISC registers, but is not used to determine their MSR numbers. Deferred error interrupt type is specifiable by MCA bank. MCA::EX::MCA_STATUS_EX[TCC] is present.

**MSRC000\_2055 [EX IP Identification] (MCA::EX::MCA\_IPID\_EX)**

Reset: 0005\_00B0\_0000\_0000h.

The MCA::EX::MCA\_IPID\_EX register is used by software to determine what IP type and revision is associated with the MCA bank.

\_lthree0\_core[3:0]\_inst5\_aliasMSR; MSRC000\_2055

Bits	Description
63:48	<b>McaType.</b> Read-only. Reset: 0005h. The McaType of the MCA bank within this IP.
47:44	Reserved.
43:32	<b>HardwareID.</b> Read-only. Reset: 0B0h. The Hardware ID of the IP associated with this MCA bank.
31:0	<b>InstanceId.</b> Read-write. Reset: 0000_0000h. The instance ID of this IP. This is initialized to a unique ID per instance of this register.

**MSRC000\_2056 [EX Machine Check Syndrome Thread 0] (MCA::EX::MCA\_SYND\_EX)**

Read-write, Volatile. Reset: Cold, 0000\_0000\_0000\_0000h.

Logs physical location information associated with error in MCA::EX::MCA\_STATUS\_EX Thread 0

\_lthree0\_core[3:0]\_thread[1:0]\_inst5\_aliasMSR; MSRC000\_2056

Bits	Description
63:33	Reserved.
32	<b>Syndrone.</b> Read-write, Volatile. Reset: Cold, 0. Contains the syndrome, if any, associated with the error logged in MCA::EX::MCA_STATUS_EX. The low-order bit of the syndrome is stored in bit 0, and the syndrome has a length specified by MCA::EX::MCA_SYND_EX[Length]. The Syndrome field is only valid when

	MCA::EX::MCA_SYND_EX[Length] is not 0.
31:27	Reserved.
26:24	<b>ErrorPriority.</b> Read-write, Volatile. Reset: Cold, 0h. Encodes the priority of the error logged in MCA::EX::MCA_SYND_EX. 3'b000 = No error; 3'b001 = Reserved; 3'b010 = Corrected Error; 3'b011 = Deferred Error; 3'b100 = Uncorrected Error; 3'b101 = Fatal Error; all others reserved.
23:18	<b>Length.</b> Read-write, Volatile. Reset: Cold, 00h. Specifies the length in bits of the syndrome contained in MCA::EX::MCA_SYND_EX[Syndrome]. A value of 0 indicates that there is no valid syndrome in MCA::EX::MCA_SYND_EX. For example, a syndrome length of 9 means that MCA::EX::MCA_SYND_EX[Syndrome] bits [8:0] contains a valid syndrome.
17:0	<b>ErrorInformation.</b> Read-write, Volatile. Reset: Cold, 0_0000h. Contains error-specific information about the location of the error. Decoding is available in Table 47 [MCA_SYND_EX Register].

Table 47: MCA\_SYND\_EX Register

Error Type	Bits	Description
WDT	[17:0]	Reserved
PRF	[17:0]	Reserved
FRF	[17:4] [3:0]	Reserved Reserved
IDRF	[17:6] [5:4] [3:0]	Reserved Reserved Reserved
PLDAG	[17:2] [1:0]	Reserved Reserved
PLDAL	[17:4] [3:0]	Reserved Reserved
CHKPTQ	[17:4] [3:2] [1:0]	Reserved Reserved Reserved
RETDISP	[17:2] [1:0]	Reserved Reserved
STATQ	[17:0]	Reserved
SQ	[17:6] [5:0]	Reserved Reserved
BBQ	[17:6] [5:0]	Reserved Reserved

**MSRC001\_0405 [EX Machine Check Control Mask] (MCA::EX::MCA\_CTL\_MASK\_EX)**

Read-write. Reset: 0000\_0000\_0000\_0000h.

Inhibit detection of an error source.

\_lthree0\_core[3:0]\_inst5\_aliasMSR; MSRC001\_0405

Bits	Description
63:11	Reserved.
10	<b>BBQ.</b> Read-write. Reset: 0. Branch buffer queue parity error.
9	<b>SQ.</b> Read-write. Reset: 0. Scheduling queue parity error.
8	<b>STATQ.</b> Read-write. Reset: 0. Retire status queue parity error.
7	<b>RETDISP.</b> Read-write. Reset: 0. Retire dispatch queue parity error.
6	<b>CHKPTQ.</b> Read-write. Reset: 0. CHKPTQ. Checkpoint queue parity error.
5	<b>PLDAL.</b> Read-write. Reset: 0. EX payload parity error.
4	<b>PLDAG.</b> Read-write. Reset: 0. Address generator payload parity error.



3	<b>IDRF.</b> Read-write. Reset: 0. Immediate displacement register file parity error.
2	<b>FRF.</b> Read-write. Reset: 0. Flag register file parity error.
1	<b>PRF.</b> Read-write. Reset: 0. Physical register file parity error.
0	<b>WDT.</b> Read-write. Reset: 0. Watchdog Timeout error.

### 3.2.5.6 FP

#### MSR0000\_0418...MSRC000\_2060 [FP Machine Check Control] (MCA::FP::MCA\_CTL\_FP)

Read-write. Reset: 0000\_0000\_0000\_0000h.

0=Disables error reporting for the corresponding error. 1=Enables error reporting via machine check exception for the corresponding error. The MCA::FP::MCA\_CTL\_FP register must be enabled by the corresponding enable bit in Core::X86::Msr::MCG\_CTL. Does not affect error detection, correction, or logging.

\_lthree0\_core[3:0]\_inst6\_aliasMSRLEGACY; MSR0000\_0418

\_lthree0\_core[3:0]\_inst6\_aliasMSR; MSRC000\_2060

Bits	Description
63:7	Reserved.
6	<b>HWA.</b> Read-write. Reset: 0. Hardware assertion.
5	<b>SRF.</b> Read-write. Reset: 0. Status register file (SRF) parity error.
4	<b>RQ.</b> Read-write. Reset: 0. Retire queue (RQ) parity error.
3	<b>NSQ.</b> Read-write. Reset: 0. NSQ parity error.
2	<b>SCH.</b> Read-write. Reset: 0. Schedule queue parity error.
1	<b>FL.</b> Read-write. Reset: 0. Freelist (FL) parity error.
0	<b>PRF.</b> Read-write. Reset: 0. Physical register file (PRF) parity error.

#### MSR0000\_0419...MSRC000\_2061 [FP Machine Check Status Thread 0] (MCA::FP::MCA\_STATUS\_FP)

Reset: Cold,0000\_0000\_0000\_0000h.

Logs information associated with errors.

\_lthree0\_core[3:0]\_thread[1:0]\_inst6\_aliasMSRLEGACY; MSR0000\_0419

\_lthree0\_core[3:0]\_thread[1:0]\_inst6\_aliasMSR; MSRC000\_2061

Bits	Description
63	<b>Val.</b> Reset: Cold,0. 1=A valid error has been detected. This bit should be cleared by software after the register has been read. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
62	<b>Overflow.</b> Reset: Cold,0. 1=An error was detected while the valid bit (Val) was set; at least one error was not logged. Overflow is set independently of whether the existing error is overwritten. See 3.1.3 [Machine Check Errors]. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
61	<b>UC.</b> Reset: Cold,0. 1=The error was not corrected by hardware. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
60	<b>En.</b> Reset: Cold,0. 1=MCA error reporting is enabled for this error, as indicated by the corresponding bit in MCA::FP::MCA_CTL_FP. This bit is a copy of bit in MCA::FP::MCA_CTL_FP for this error. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
59	<b>MiscV.</b> Reset: Cold,0. 1=Valid thresholding in MCA::FP::MCA_MISC0_FP. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
58	<b>AddrV.</b> Reset: Cold,0. 1=MCA::FP::MCA_ADDR_FP contains address information associated with the error. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
57	<b>PCC.</b> Reset: Cold,0. 1=Hardware context held by the processor may have been corrupted. Continued operation of the system may have unpredictable results. The error is not recoverable or survivable, and the system should be reinitialized. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.



56	<b>ErrCoreIdVal.</b> Reset: Cold,0. 1=The ErrCoreId field is valid. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
55	<b>TCC.</b> Reset: Cold,0. 0=The thread which consumed the error can be restarted reliably at the instruction pointer address pushed onto the exception handler stack if any uncorrected error has been corrected by software. 1=The thread which consumed the error is not restartable and must be terminated. Only meaningful when MCA::FP::MCA_STATUS_FP[PCC]=0. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
54	Reserved.
53	<b>SyndV.</b> Reset: Cold,0. 1=This error logged information in MCA::FP::MCA_SYND_FP. If MCA::FP::MCA_SYND_FP[ErrorPriority] is the same as the priority of the error in MCA::FP::MCA_STATUS_FP, then the information in MCA::FP::MCA_SYND_FP is associated with the error in MCA::FP::MCA_STATUS_FP. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
52:47	Reserved.
46	<b>CECC.</b> Reset: Cold,0. 1=The error was a correctable ECC error according to the restrictions of the ECC algorithm. UC indicates whether the error was actually corrected by the processor. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
45	<b>UECC.</b> Reset: Cold,0. 1=The error was an uncorrectable ECC error according to the restrictions of the ECC algorithm. UC indicates whether the error was actually corrected by the processor. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
44	<b>Deferred.</b> Reset: Cold,0. 1=A deferred error was created. A deferred error is the result of an uncorrectable data error which did not immediately cause a processor exception; an exception is deferred until the erroneous data is consumed. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
43	<b>Poison.</b> Reset: Cold,0. 1=The error was the result of attempting to consume poisoned data. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
42:38	Reserved.
37:32	<b>ErrCoreId.</b> Reset: Cold,00h. When ErrCoreIdVal=1 this field indicates which core within the processor is associated with the error; Otherwise this field is reserved. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
31:22	Reserved.
21:16	<b>ErrorCodeExt.</b> Reset: Cold,00h. Extended Error Code. This field is used to identify the error type for root cause analysis. This field indicates which bit position in MCA::FP::MCA_CTL_FP enables error reporting for the logged error. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
15:0	<b>ErrorCode.</b> Reset: Cold,0000h. Error code for this error. See 3.1.3.3 [Error Codes] for details on decoding this field. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.

Table 48: MCA\_STATUS\_FP[ErrorCodeExt] Decode

Error Type	Bits	Description
PRF	[6:0]	0x0
FL	[6:0]	0x1
SCH	[6:0]	0x2
NSQ	[6:0]	0x3
RQ	[6:0]	0x4
SRF	[6:0]	0x5
HWA	[6:0]	0x6

MSR0000\_041A...MSRC000\_2062 [FP Machine Check Address Thread 0] (MCA::FP::MCA\_ADDR\_FP)

Read-only. Reset: Cold,0000_0000_0000_0000h.	
MCA::FP::MCA_ADDR_FP stores an address and other information associated with the error in MCA::FP::MCA_STATUS_FP. The register is only meaningful if MCA::FP::MCA_STATUS_FP[Val]=1 and MCA::FP::MCA_STATUS_FP[AddrV]=1.	
_lthree0_core[3:0]_thread[1:0]_inst6_aliasMSRLEGACY; MSR0000_041A	
_lthree0_core[3:0]_thread[1:0]_inst6_aliasMSR; MSRC000_2062	
Bits	Description
63:62	Reserved.
61:56	<b>LSB.</b> Read-only. Reset: Cold,00h. Specifies the least significant valid bit of the address contained in MCA::FP::MCA_ADDR_FP[ErrorAddr]. A value of 0 indicates that MCA::FP::MCA_ADDR_FP[55:0] contains a valid byte address. A value of 6 indicates that MCA::FP::MCA_ADDR_FP[55:6] contains a valid cache line address and that MCA::FP::MCA_ADDR_FP[5:0] are not part of the address and should be ignored by error handling software. A value of 12 indicates that MCA::FP::MCA_ADDR_FP[55:12] contain a valid 4KB memory page and that MCA::FP::MCA_ADDR_FP[11:0] should be ignored by error handling software.
55:0	<b>ErrorAddr.</b> Read-only. Reset: Cold,00_0000_0000_0000h. Contains the address, if any, associated with the error logged in MCA::FP::MCA_STATUS_FP.

Table 49: MCA\_ADDR\_FP Register

Error Type	Bits	Description
PRF	[55:0]	Reserved
FL	[55:0]	Reserved
SCH	[55:0]	Reserved
NSQ	[55:0]	Reserved
RQ	[55:0]	Reserved
SRF	[55:0]	Reserved
HWA	[55:0]	Reserved

**MSR0000\_041B...MSRC000\_2063 [FP Machine Check Miscellaneous 0 Thread 0] (MCA::FP::MCA\_MISC0\_FP)**

Log miscellaneous information associated with errors.	
_lthree0_core[3:0]_thread[1:0]_inst6_aliasMSRLEGACY; MSR0000_041B	
_lthree0_core[3:0]_thread[1:0]_inst6_aliasMSR; MSRC000_2063	
Bits	Description
63	<b>Valid.</b> Reset: 1. 1=A valid CntP field is present in this register. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read-only.
62	<b>CntP.</b> Reset: 1. 1=A valid threshold counter is present. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read-only.
61	<b>Locked.</b> Reset: 0. 1=Writes to this register are ignored. This bit is set by BIOS to indicate that this register is not available for OS use. BIOS should set this bit if ThresholdIntType is set to SMI. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read-only.
60	<b>IntP.</b> Reset: 1. 1=ThresholdIntType can be used to generate interrupts. 0=ThresholdIntType and interrupt generation are not supported. AccessType: (Core::X86::Msr::HWCR[McStatusWrEn]   !MCA::FP::MCA_MISC0_FP[Locked]) ? Read-write : Read-only.
59:56	Reserved.
55:52	<b>LvtOffset.</b> Reset: 0h. One per die. 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). AccessType: (Core::X86::Msr::HWCR[McStatusWrEn]   !MCA::FP::MCA_MISC0_FP[Locked]) ? Read-write : Read-only.
51	<b>CntEn.</b> Reset: 0. 1=Count thresholding errors.

	AccessType: (Core::X86::Msr::HWCR[McStatusWrEn]   !MCA::FP::MCA_MISC0_FP[Locked]) ? Read-write : Read-only.
50:49	<b>ThresholdIntType.</b> Reset: Cold,0h. Specifies the type of interrupt signaled when Ovrflw is set and IntP==1. 00b = No interrupt. 01b = APIC based interrupt (see Core::X86::Msr::McaIntrCfg[ThresholdLvtOffset]). 10b = SMI trigger event. 11b = Reserved.
	AccessType: (Core::X86::Msr::HWCR[McStatusWrEn]   !MCA::FP::MCA_MISC0_FP[Locked]) ? Read-write : Read-only.
48	<b>Ovrflw.</b> Reset: Cold,0. Set by hardware when ErrCnt transitions from FFEh to FFFh. When this field is set, ErrCnt no longer increments. When this bit is set, the interrupt selected by the ThresholdIntType field is generated.
	AccessType: (Core::X86::Msr::HWCR[McStatusWrEn]   !MCA::FP::MCA_MISC0_FP[Locked]) ? Read-write : Read-only.
47:44	Reserved.
43:32	<b>ErrCnt.</b> Reset: Cold,000h. This is written by software to set the starting value of the error counter. This is incremented by hardware when errors are logged. When this counter overflows, it stays at FFFh (no rollover). The threshold value, written by software, is (FFFh - the desired error count (the number of errors necessary in order for an interrupt to be taken)); the desired error count of 0 (a write value of FFFh) is not supported.
	AccessType: (Core::X86::Msr::HWCR[McStatusWrEn]   !MCA::FP::MCA_MISC0_FP[Locked]) ? Read-write : Read-only.
31:24	<b>BlkPtr.</b> Read-write. Reset: 01h. 00h=Extended MISC MSR block is not valid. 01h=Extended MSR block is valid.
23:0	Reserved.

#### MSRC000\_2064 [FP Machine Check Configuration] (MCA::FP::MCA\_CONFIG\_FP)

Reset: 0000\_0002\_0000\_0021h.

Controls configuration of the associated machine check bank.

\_lthree0\_core[3:0]\_inst6\_aliasMSR; MSRC000\_2064

Bits	Description
63:39	Reserved.
38:37	<b>DeferredIntType.</b> Read-write. Reset: 0h. Specifies the type of interrupt signaled when a deferred error is logged. 00b = No interrupt. 01b = APIC based interrupt (see Core::X86::Msr::McaIntrCfg[DeferredLvtOffset]). 10b = SMI trigger event. 11b = Reserved.
36:33	Reserved.
32	<b>McaXEnable.</b> Read-write. Reset: 0. Init: BIOS,1. Check: 1. 1=Software has acknowledged support for the MCAX feature set. 0=Software has not acknowledged support for the MCAX feature set. All uncorrected and fatal errors will cause an ErrorEvent packet to be generated. Deferred error interrupts are configured via Core::X86::Msr::McaIntrCfg.
31:6	Reserved.
5	<b>DeferredIntTypeSupported.</b> Read-only. Reset: 1. 1=MCA::FP::MCA_CONFIG_FP[DeferredIntType] controls the type of interrupt generated on a deferred error. Deferred errors are supported in this bank only if MCA::FP::MCA_CONFIG_FP[DeferredErrorLoggingSupported]=1.
4:3	Reserved.
2	<b>DeferredErrorLoggingSupported.</b> Read-only. Reset: 0. 1=Deferred errors are supported in this MCA bank, and the LogDeferredInMcaStat field in this register controls the logging behavior of these errors. MCA_DESTAT and MCA_DEADDR are supported in this MCA bank. 0=Deferred errors are not supported in this bank.
1	Reserved.
0	<b>McaX.</b> Read-only. Reset: 1. 1=This bank provides Machine Check Architecture Extensions. Up to 4 additional MISC registers (MISC1-MISC4) are supported. MCA::FP::MCA_MISC0_FP[BlkPtr] indicates the presence of the additional MISC registers, but is not used to determine their MSR numbers. Deferred error interrupt type is specifiable by MCA bank. MCA::FP::MCA_STATUS_FP[TCC] is present.

#### MSRC000\_2065 [FP IP Identification] (MCA::FP::MCA\_IPID\_FP)

Reset: 0006_00B0_0000_0000h.	
The MCA::FP::MCA_IPID_FP register is used by software to determine what IP type and revision is associated with the MCA bank.	
_lthree0_core[3:0]_inst6_aliasMSR; MSRC000_2065	
Bits	Description
63:48	<b>McaType</b> . Read-only. Reset: 0006h. The McaType of the MCA bank within this IP.
47:44	Reserved.
43:32	<b>HardwareID</b> . Read-only. Reset: 0B0h. The Hardware ID of the IP associated with this MCA bank.
31:0	<b>InstanceId</b> . Read-write. Reset: 0000_0000h. The instance ID of this IP. This is initialized to a unique ID per instance of this register.

#### MSRC000\_2066 [FP Machine Check Syndrome Thread 0] (MCA::FP::MCA\_SYND\_FP)

Read-write, Volatile. Reset: Cold, 0000_0000_0000_0000h.	
Logs physical location information associated with error in MCA::FP::MCA_STATUS_FP Thread 0	
_lthree0_core[3:0]_thread[1:0]_inst6_aliasMSR; MSRC000_2066	
Bits	Description
63:33	Reserved.
32	<b>Syndrom</b> . Read-write, Volatile. Reset: Cold, 0. Contains the syndrome, if any, associated with the error logged in MCA::FP::MCA_STATUS_FP. The low-order bit of the syndrome is stored in bit 0, and the syndrome has a length specified by MCA::FP::MCA_SYND_FP[Length]. The Syndrome field is only valid when MCA::FP::MCA_SYND_FP[Length] is not 0.
31:27	Reserved.
26:24	<b>ErrorPriority</b> . Read-write, Volatile. Reset: Cold, 0h. Encodes the priority of the error logged in MCA::FP::MCA_SYND_FP. 3'b000 = No error; 3'b001 = Reserved; 3'b010 = Corrected Error; 3'b011 = Deferred Error; 3'b100 = Uncorrected Error; 3'b101 = Fatal Error; all others reserved.
23:18	<b>Length</b> . Read-write, Volatile. Reset: Cold, 00h. Specifies the length in bits of the syndrome contained in MCA::FP::MCA_SYND_FP[Syndrome]. A value of 0 indicates that there is no valid syndrome in MCA::FP::MCA_SYND_FP. For example, a syndrome length of 9 means that MCA::FP::MCA_SYND_FP[Syndrome] bits [8:0] contains a valid syndrome.
17:0	<b>ErrorInformation</b> . Read-write, Volatile. Reset: Cold, 0_0000h. Contains error-specific information about the location of the error. Decoding is available in Table 50 [MCA_SYND_FP Register].

Table 50: MCA\_SYND\_FP Register

Error Type	Bits	Description
PRF	[17:0]	Reserved
FL	[17:0]	Reserved
SCH	[17:0]	Reserved
NSQ	[17:0]	Reserved
RQ	[17:0]	Reserved
SRF	[17:0]	Reserved
HWA	[17:0]	Reserved

#### MSRC001\_0406 [FP Machine Check Control Mask] (MCA::FP::MCA\_CTL\_MASK\_FP)

Read-write. Reset: 0000_0000_0000_0000h.	
Inhibit detection of an error source.	
_lthree0_core[3:0]_inst6_aliasMSR; MSRC001_0406	
Bits	Description
63:7	Reserved.
6	<b>HWA</b> . Read-write. Reset: 0. Init: BIOS, 1. Hardware assertion.
5	<b>SRF</b> . Read-write. Reset: 0. Status register file (SRF) parity error.

4	<b>RQ.</b> Read-write. Reset: 0. Retire queue (RQ) parity error.
3	<b>NSQ.</b> Read-write. Reset: 0. NSQ parity error.
2	<b>SCH.</b> Read-write. Reset: 0. Schedule queue parity error.
1	<b>FL.</b> Read-write. Reset: 0. Freelist (FL) parity error.
0	<b>PRF.</b> Read-write. Reset: 0. Physical register file (PRF) parity error.

### 3.2.5.7 L3

#### MSR0000\_041C...MSRC000\_20A0 [L3 Machine Check Control] (MCA::L3::MCA\_CTL\_L3)

Read-write. Reset: 0000\_0000\_0000\_0000h.

0=Disables error reporting for the corresponding error. 1=Enables error reporting via machine check exception for the corresponding error. The MCA::L3::MCA\_CTL\_L3 register must be enabled by the corresponding enable bit in Core::X86::Msr::MCG\_CTL. Does not affect error detection, correction, or logging.

\_ccx0\_inst7\_aliasMSRLEGACY; MSR0000\_041C

\_ccx0\_inst8\_aliasMSRLEGACY; MSR0000\_0420

\_ccx0\_inst9\_aliasMSRLEGACY; MSR0000\_0424

\_ccx0\_inst10\_aliasMSRLEGACY; MSR0000\_0428

\_ccx0\_inst7\_aliasMSR; MSRC000\_2070

\_ccx0\_inst8\_aliasMSR; MSRC000\_2080

\_ccx0\_inst9\_aliasMSR; MSRC000\_2090

\_ccx0\_inst10\_aliasMSR; MSRC000\_20A0

Bits	Description
63:8	Reserved.
7	<b>Hwa.</b> Read-write. Reset: 0. L3 Hardware Assertion.
6	<b>XiVictimQueue.</b> Read-write. Reset: 0. L3 Victim Queue Parity Error.
5	<b>SdpParity.</b> Read-write. Reset: 0. SDP Parity Error from XI.
4	<b>DataArray.</b> Read-write. Reset: 0. L3M Data ECC Error.
3	<b>MultiHitTag.</b> Read-write. Reset: 0. L3M Tag Multi-way-hit Error.
2	<b>Tag.</b> Read-write. Reset: 0. L3M Tag ECC Error.
1	<b>MultiHitShadowTag.</b> Read-write. Reset: 0. Shadow Tag Macro Multi-way-hit Error.
0	<b>ShadowTag.</b> Read-write. Reset: 0. Shadow Tag Macro ECC Error.

#### MSR0000\_041D...MSRC000\_20A1 [L3 Machine Check Status] (MCA::L3::MCA\_STATUS\_L3)

Reset: Cold,0000\_0000\_0000\_0000h.

Logs information associated with errors.

\_ccx0\_inst7\_aliasMSRLEGACY; MSR0000\_041D

\_ccx0\_inst8\_aliasMSRLEGACY; MSR0000\_0421

\_ccx0\_inst9\_aliasMSRLEGACY; MSR0000\_0425

\_ccx0\_inst10\_aliasMSRLEGACY; MSR0000\_0429

\_ccx0\_inst7\_aliasMSR; MSRC000\_2071

\_ccx0\_inst8\_aliasMSR; MSRC000\_2081

\_ccx0\_inst9\_aliasMSR; MSRC000\_2091

\_ccx0\_inst10\_aliasMSR; MSRC000\_20A1

Bits	Description
63	<b>Val.</b> Reset: Cold,0. 1=A valid error has been detected. This bit should be cleared by software after the register has been read. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
62	<b>Overflow.</b> Reset: Cold,0. 1=An error was detected while the valid bit (Val) was set; at least one error was not logged. Overflow is set independently of whether the existing error is overwritten. See 3.1.3 [Machine Check Errors]. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
61	<b>UC.</b> Reset: Cold,0. 1=The error was not corrected by hardware.



	AccessType: Core::X86::Msrr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
60	<b>En.</b> Reset: Cold,0. 1=MCA error reporting is enabled for this error, as indicated by the corresponding bit in MCA::L3::MCA_CTL_L3. This bit is a copy of bit in MCA::L3::MCA_CTL_L3 for this error.
	AccessType: Core::X86::Msrr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
59	<b>MiscV.</b> Reset: Cold,0. 1=Valid thresholding in MCA::L3::MCA_MISC0_L3.
	AccessType: Core::X86::Msrr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
58	<b>AddrV.</b> Reset: Cold,0. 1=MCA::L3::MCA_ADDR_L3 contains address information associated with the error.
	AccessType: Core::X86::Msrr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
57	<b>PCC.</b> Reset: Cold,0. 1=Hardware context held by the processor may have been corrupted. Continued operation of the system may have unpredictable results. The error is not recoverable or survivable, and the system should be reinitialized.
	AccessType: Core::X86::Msrr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
56	<b>ErrCoreIdVal.</b> Reset: Cold,0. 1=The ErrCoreId field is valid.
	AccessType: Core::X86::Msrr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
55	<b>TCC.</b> Reset: Cold,0. 0=The thread which consumed the error can be restarted reliably at the instruction pointer address pushed onto the exception handler stack if any uncorrected error has been corrected by software. 1=The thread which consumed the error is not restartable and must be terminated. Only meaningful when MCA::L3::MCA_STATUS_L3[PCC]=0.
	AccessType: Core::X86::Msrr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
54	Reserved.
53	<b>SyndV.</b> Reset: Cold,0. 1=This error logged information in MCA::L3::MCA_SYND_L3. If MCA::L3::MCA_SYND_L3[ErrorPriority] is the same as the priority of the error in MCA::L3::MCA_STATUS_L3, then the information in MCA::L3::MCA_SYND_L3 is associated with the error in MCA::L3::MCA_STATUS_L3.
	AccessType: Core::X86::Msrr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
52:47	Reserved.
46	<b>CECC.</b> Reset: Cold,0. 1=The error was a correctable ECC error according to the restrictions of the ECC algorithm. UC indicates whether the error was actually corrected by the processor.
	AccessType: Core::X86::Msrr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
45	<b>UECC.</b> Reset: Cold,0. 1=The error was an uncorrectable ECC error according to the restrictions of the ECC algorithm. UC indicates whether the error was actually corrected by the processor.
	AccessType: Core::X86::Msrr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
44	<b>Deferred.</b> Reset: Cold,0. 1=A deferred error was created. A deferred error is the result of an uncorrectable data error which did not immediately cause a processor exception; an exception is deferred until the erroneous data is consumed.
	AccessType: Core::X86::Msrr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
43	<b>Poison.</b> Reset: Cold,0. 1=The error was the result of attempting to consume poisoned data.
	AccessType: Core::X86::Msrr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
42:38	Reserved.
37:32	<b>ErrCoreId.</b> Reset: Cold,00h. When ErrCoreIdVal=1 this field indicates which core within the processor is associated with the error; Otherwise this field is reserved.
	AccessType: Core::X86::Msrr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
31:22	Reserved.
21:16	<b>ErrorCodeExt.</b> Reset: Cold,00h. Extended Error Code. This field is used to identify the error type for root cause analysis. This field indicates which bit position in MCA::L3::MCA_CTL_L3 enables error reporting for the logged error.
	AccessType: Core::X86::Msrr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
15:0	<b>ErrorCode.</b> Reset: Cold,0000h. Error code for this error. See 3.1.3.3 [Error Codes] for details on decoding this field.
	AccessType: Core::X86::Msrr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.

Table 51: MCA\_STATUS\_L3[ErrorCodeExt] Decode

Error Type	Bits	Description
ShadowTag	[6:0]	0x0
MultiHitShadowTag	[6:0]	0x1
Tag	[6:0]	0x2
MultiHitTag	[6:0]	0x3
DataArray	[6:0]	0x4
SdpParity	[6:0]	0x5
XiVictimQueue	[6:0]	0x6
Hwa	[6:0]	0x7

**MSR0000\_041E...MSRC000\_20A2 [L3 Machine Check Address] (MCA::L3::MCA\_ADDR\_L3)**

Reset: Cold,0000_0000_0000_0000h.	
MCA::L3::MCA_ADDR_L3 stores an address and other information associated with the error in MCA::L3::MCA_STATUS_L3. The register is only meaningful if MCA::L3::MCA_STATUS_L3[Val]=1 and MCA::L3::MCA_STATUS_L3[AddrV]=1.	
_ccx0_inst7_aliasMSRLEGACY; MSR0000_041E	
_ccx0_inst8_aliasMSRLEGACY; MSR0000_0422	
_ccx0_inst9_aliasMSRLEGACY; MSR0000_0426	
_ccx0_inst10_aliasMSRLEGACY; MSR0000_042A	
_ccx0_inst7_aliasMSR; MSRC000_2072	
_ccx0_inst8_aliasMSR; MSRC000_2082	
_ccx0_inst9_aliasMSR; MSRC000_2092	
_ccx0_inst10_aliasMSR; MSRC000_20A2	
Bits	Description
63:62	Reserved.
61:56	<b>LSB.</b> Read-write, Volatile. Reset: Cold,00h. Specifies the least significant valid bit of the address contained in MCA::L3::MCA_ADDR_L3[ErrorAddr]. A value of 0 indicates that MCA::L3::MCA_ADDR_L3[55:0] contains a valid byte address. A value of 6 indicates that MCA::L3::MCA_ADDR_L3[55:6] contains a valid cache line address and that MCA::L3::MCA_ADDR_L3[5:0] are not part of the address and should be ignored by error handling software. A value of 12 indicates that MCA::L3::MCA_ADDR_L3[55:12] contain a valid 4KB memory page and that MCA::L3::MCA_ADDR_L3[11:0] should be ignored by error handling software.
55:0	<b>ErrorAddr.</b> Read-write, Volatile. Reset: Cold,00_0000_0000_0000h. Unless otherwise specified by an error, contains the address associated with the error logged in MCA::L3::MCA_STATUS_L3. For physical addresses, the most significant bit is given by Core::X86::CpuId::LongModeInfo[PhysAddrSize].

Table 52: MCA\_ADDR\_L3 Register

Error Type	Bits	Description
ShadowTag	[55:16] [15:0]	Reserved 16'b{8'b{Index}, 2'b{Slice}, 6'b{0}}
MultiHitShadowTag	[55:16] [15:0]	Reserved 16'b{8'b{Index}, 2'b{Slice}, 6'b{0}}
Tag	[55:19] [18:0]	Reserved 19'b{1'b{Bank[3]}, 7'b{Index}, 3'b{Bank[2:0]}, 2'b{slice}, 6'b{0}}
MultiHitTag	[55:19] [18:0]	Reserved 19'b{1'b{Bank[3]}, 7'b{Index}, 3'b{Bank[2:0]}, 2'b{slice}, 6'b{0}}
DataArray	[55:48] [47:0]	Reserved Physical Address
SdpParity	[55:48]	Reserved



	[47:0]	Physical Address
XiVictimQueue	[55:48] [47:0]	Reserved Physical Address
Hwa	[55:34] [33:0]	Reserved Reserved

**MSR0000\_041F...MSRC000\_20A3 [L3 Machine Check Miscellaneous 0] (MCA::L3::MCA\_MISC0\_L3)**

Log miscellaneous information associated with errors.

\_ccx0\_inst7\_aliasMSRLEGACY; MSR0000\_041F

\_ccx0\_inst8\_aliasMSRLEGACY; MSR0000\_0423

\_ccx0\_inst9\_aliasMSRLEGACY; MSR0000\_0427

\_ccx0\_inst10\_aliasMSRLEGACY; MSR0000\_042B

\_ccx0\_inst7\_aliasMSR; MSRC000\_2073

\_ccx0\_inst8\_aliasMSR; MSRC000\_2083

\_ccx0\_inst9\_aliasMSR; MSRC000\_2093

\_ccx0\_inst10\_aliasMSR; MSRC000\_20A3

Bits	Description
63	<b>Valid.</b> Reset: 1. 1=A valid CntP field is present in this register. AccessType: Core::X86::Msrr::HWCR[McStatusWrEn] ? Read-write : Read-only.
62	<b>CntP.</b> Reset: 1. 1=A valid threshold counter is present. AccessType: Core::X86::Msrr::HWCR[McStatusWrEn] ? Read-write : Read-only.
61	<b>Locked.</b> Reset: 0. 1=Writes to this register are ignored. This bit is set by BIOS to indicate that this register is not available for OS use. BIOS should set this bit if ThresholdIntType is set to SMI. AccessType: Core::X86::Msrr::HWCR[McStatusWrEn] ? Read-write : Read-only.
60	<b>IntP.</b> Reset: 1. 1=ThresholdIntType can be used to generate interrupts. 0=ThresholdIntType and interrupt generation are not supported. AccessType: (Core::X86::Msrr::HWCR[McStatusWrEn]   !MCA::L3::MCA_MISC0_L3[Locked]) ? Read-write : Read-only.
59:56	Reserved.
55:52	<b>LvtOffset.</b> Reset: 0h. One per die. 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). AccessType: (Core::X86::Msrr::HWCR[McStatusWrEn]   !MCA::L3::MCA_MISC0_L3[Locked]) ? Read-write : Read-only.
51	<b>CntEn.</b> Reset: 0. 1=Count thresholding errors. AccessType: (Core::X86::Msrr::HWCR[McStatusWrEn]   !MCA::L3::MCA_MISC0_L3[Locked]) ? Read-write : Read-only.
50:49	<b>ThresholdIntType.</b> Reset: Cold,0h. Specifies the type of interrupt signaled when Ovrflw is set and IntP==1. 00b = No interrupt. 01b = APIC based interrupt (see Core::X86::Msrr::McaIntrCfg[ThresholdLvtOffset]). 10b = SMI trigger event. 11b = Reserved. AccessType: (Core::X86::Msrr::HWCR[McStatusWrEn]   !MCA::L3::MCA_MISC0_L3[Locked]) ? Read-write : Read-only.
48	<b>Ovrflw.</b> Reset: Cold,0. Set by hardware when ErrCnt transitions from FFEh to FFFh. When this field is set, ErrCnt no longer increments. When this bit is set, the interrupt selected by the ThresholdIntType field is generated. AccessType: (Core::X86::Msrr::HWCR[McStatusWrEn]   !MCA::L3::MCA_MISC0_L3[Locked]) ? Read-write : Read-only.
47:44	Reserved.
43:32	<b>ErrCnt.</b> Reset: Cold,000h. This is written by software to set the starting value of the error counter. This is incremented by hardware when errors are logged. When this counter overflows, it stays at FFFh (no rollover). The threshold value, written by software, is (FFFh - the desired error count (the number of errors necessary in order for an interrupt to be taken)); the desired error count of 0 (a write value of FFFh) is not supported.

	AccessType: (Core::X86::Msr::HWCR[McStatusWrEn]   !MCA::L3::MCA_MISC0_L3[Locked]) ? Read-write : Read-only.
31:24	<b>BlkPtr.</b> Read-write. Reset: 01h. 00h=Extended MISC MSR block is not valid. 01h=Extended MSR block is valid.
23:0	Reserved.

#### MSRC000\_20[7...A]4 [L3 Machine Check Configuration] (MCA::L3::MCA\_CONFIG\_L3)

Reset: 0000\_0000\_0000\_0025h.

Controls configuration of the associated machine check bank.

\_ccx0\_inst7\_aliasMSR; MSRC000\_2074

\_ccx0\_inst8\_aliasMSR; MSRC000\_2084

\_ccx0\_inst9\_aliasMSR; MSRC000\_2094

\_ccx0\_inst10\_aliasMSR; MSRC000\_20A4

Bits	Description
63:39	Reserved.
38:37	<b>DeferredIntType.</b> Read-write. Reset: 0h. Specifies the type of interrupt signaled when a deferred error is logged. 00b = No interrupt. 01b = APIC based interrupt (see Core::X86::Msr::McaIntrCfg[DeferredLvtOffset]). 10b = SMI trigger event. 11b = Reserved.
36:35	Reserved.
34	<b>LogDeferredInMcaStat.</b> Read-write. Reset: 0. Init: BIOS,1. 1=Log deferred errors in MCA::L3::MCA_STATUS_L3 and MCA::L3::MCA_ADDR_L3 in addition to MCA::L3::MCA_DESTAT_L3 and MCA::L3::MCA_DEADDR_L3. 0=Only log deferred errors in MCA::L3::MCA_DESTAT_L3 and MCA::L3::MCA_DEADDR_L3. This bit does not affect logging of deferred errors in MCA::L3::MCA_SYND_L3, MCA::L3::MCA_MISC0_L3.
33	Reserved.
32	<b>McaXEnable.</b> Read-write. Reset: 0. Init: BIOS,1. Check: 1. 1=Software has acknowledged support for the MCAX feature set. 0=Software has not acknowledged support for the MCAX feature set. All uncorrected and fatal errors will cause an ErrorEvent packet to be generated. Deferred error interrupts are configured via Core::X86::Msr::McaIntrCfg.
31:6	Reserved.
5	<b>DeferredIntTypeSupported.</b> Read-only. Reset: 1. 1=MCA::L3::MCA_CONFIG_L3[DeferredIntType] controls the type of interrupt generated on a deferred error. Deferred errors are supported in this bank only if MCA::L3::MCA_CONFIG_L3[DeferredErrorLoggingSupported]=1.
4:3	Reserved.
2	<b>DeferredErrorLoggingSupported.</b> Read-only. Reset: 1. 1=Deferred errors are supported in this MCA bank, and MCA::L3::MCA_CONFIG_L3[LogDeferredInMcaStat] controls the logging behavior of these errors. MCA::L3::MCA_DESTAT_L3 and MCA::L3::MCA_DEADDR_L3 are supported in this MCA bank. 0=Deferred errors are not supported in this bank.
1	Reserved.
0	<b>McaX.</b> Read-only. Reset: 1. 1=This bank provides Machine Check Architecture Extensions. Up to 4 additional MISC registers (MISC1-MISC4) are supported. MCA::L3::MCA_MISC0_L3[BlkPtr] indicates the presence of the additional MISC registers, but is not used to determine their MSR numbers. Deferred error interrupt type is specifiable by MCA bank. MCA::L3::MCA_STATUS_L3[TCC] is present.

#### MSRC000\_20[7...A]5 [L3 IP Identification] (MCA::L3::MCA\_IPID\_L3)

Reset: 0007\_00B0\_0000\_0000h.

The MCA::L3::MCA\_IPID\_L3 register is used by software to determine what IP type and revision is associated with the MCA bank.

\_ccx0\_inst7\_aliasMSR; MSRC000\_2075

\_ccx0\_inst8\_aliasMSR; MSRC000\_2085

\_ccx0\_inst9\_aliasMSR; MSRC000\_2095

\_ccx0\_inst10\_aliasMSR; MSRC000\_20A5

Bits	Description
63:48	<b>McaType.</b> Read-only. Reset: 0007h. The McaType of the MCA bank within this IP.

47:44	Reserved.
43:32	<b>HardwareID.</b> Read-only. Reset: 0B0h. The Hardware ID of the IP associated with this MCA bank.
31:0	<b>InstanceId.</b> Read-write. Reset: 0000_0000h. The instance ID of this IP. This is initialized to a unique ID per instance of this register.

**MSRC000\_20[7...A]6 [L3 Machine Check Syndrome] (MCA::L3::MCA\_SYND\_L3)**

Read-write, Volatile. Reset: Cold, 0000_0000_0000_0000h.	
Logs physical location information associated with error in MCA::L3::MCA_STATUS_L3 Thread 0	
_ccx0_inst7_aliasMSR; MSRC000_2076	
_ccx0_inst8_aliasMSR; MSRC000_2086	
_ccx0_inst9_aliasMSR; MSRC000_2096	
_ccx0_inst10_aliasMSR; MSRC000_20A6	
Bits	Description
63:49	Reserved.
48:32	<b>Syndrone.</b> Read-write, Volatile. Reset: Cold, 0_0000h. Contains the syndrome, if any, associated with the error logged in MCA::L3::MCA_STATUS_L3. The low-order bit of the syndrome is stored in bit 0, and the syndrome has a length specified by MCA::L3::MCA_SYND_L3[Length]. The Syndrome field is only valid when MCA::L3::MCA_SYND_L3[Length] is not 0.
31:27	Reserved.
26:24	<b>ErrorPriority.</b> Read-write, Volatile. Reset: Cold, 0h. Encodes the priority of the error logged in MCA::L3::MCA_SYND_L3. 3'b000 = No error; 3'b001 = Reserved; 3'b010 = Corrected Error; 3'b011 = Deferred Error; 3'b100 = Uncorrected Error; 3'b101 = Fatal Error; all others reserved.
23:18	<b>Length.</b> Read-write, Volatile. Reset: Cold, 00h. Specifies the length in bits of the syndrome contained in MCA::L3::MCA_SYND_L3[Syndrome]. A value of 0 indicates that there is no valid syndrome in MCA::L3::MCA_SYND_L3. For example, a syndrome length of 9 means that MCA::L3::MCA_SYND_L3[Syndrome] bits [8:0] contains a valid syndrome.
17:0	<b>ErrorInformation.</b> Read-write, Volatile. Reset: Cold, 0_0000h. Contains error-specific information about the location of the error. Decoding is available in Table 53 [MCA_SYND_L3 Register].

Table 53: MCA\_SYND\_L3 Register

Error Type	Bits	Description
ShadowTag	[17:12]	Reserved
	[11:8]	Pack
	[7:3]	Reserved
	[2:0]	Way
MultiHitShadowTag	[17:12]	Reserved
	[11:8]	Pack
	[7:0]	Reserved
Tag	[17:12]	Reserved
	[11:8]	Bank.
	[7:0]	Way
MultiHitTag	[17:0]	Reserved
DataArray	[17:12]	Reserved
	[11:8]	Bank[2:0]
	[7:3]	Reserved
	[2:0]	Way
SdpParity	[17:0]	Reserved
XiVictimQueue	[17:0]	Reserved
Hwa	[17:0]	Reserved

**MSRC000\_20[7...A]8 [L3 Machine Check Deferred Error Status] (MCA::L3::MCA\_DESTAT\_L3)**

Read-write, Volatile. Reset: Cold, 0000\_0000\_0000\_0000h.

Holds status information for the first deferred error seen in this bank.

\_ccx0\_inst7\_aliasMSR; MSRC000\_2078

\_ccx0\_inst8\_aliasMSR; MSRC000\_2088

\_ccx0\_inst9\_aliasMSR; MSRC000\_2098

\_ccx0\_inst10\_aliasMSR; MSRC000\_20A8

Bits	Description
63	<b>Val.</b> Read-write, Volatile. Reset: Cold, 0. 1=A valid error has been detected (whether it is enabled or not).
62	<b>Overflow.</b> Read-write, Volatile. Reset: Cold, 0. 1=An error was detected while the valid bit (Val) was set; at least one error was not logged. Overflow is set independently of whether the existing error is overwritten. (See the section on overwrite priorities.)
61:59	Reserved.
58	<b>AddrV.</b> Read-write, Volatile. Reset: Cold, 0. 1=MCA::L3::MCA_DEADDR_L3 contains address information associated with the error.
57:54	Reserved.
53	<b>SyndV.</b> Read-write, Volatile. Reset: Cold, 0. 1=This error logged information in MCA::L3::MCA_SYND_L3. If MCA::L3::MCA_SYND_L3[ErrorPriority] is the same as the priority of the error in MCA::L3::MCA_STATUS_L3, then the information in MCA::L3::MCA_SYND_L3 is associated with the error in MCA::L3::MCA_DESTAT_L3.
52:45	Reserved.
44	<b>Deferred.</b> Read-write, Volatile. Reset: Cold, 0. 1=A deferred error was created. A deferred error is the result of an uncorrectable data error which did not immediately cause a processor exception; poison is created and an exception is deferred until the poison data is consumed.
43:0	Reserved.

**MSRC000\_20[7...A]9 [L3 Deferred Error Address] (MCA::L3::MCA\_DEADDR\_L3)**

Reset: Cold, 0000\_0000\_0000\_0000h.

The MCA::L3::MCA\_DEADDR\_L3 register stores the address associated with the error in MCA::L3::MCA\_DESTAT\_L3. The register is only meaningful if MCA::L3::MCA\_DESTAT\_L3[Val]=1 and MCA::L3::MCA\_DESTAT\_L3[AddrV]=1. The lowest valid bit of the address is defined by MCA::L3::MCA\_DEADDR\_L3[LSB].

\_ccx0\_inst7\_aliasMSR; MSRC000\_2079

\_ccx0\_inst8\_aliasMSR; MSRC000\_2089

\_ccx0\_inst9\_aliasMSR; MSRC000\_2099

\_ccx0\_inst10\_aliasMSR; MSRC000\_20A9

Bits	Description
63:62	Reserved.
61:56	<b>LSB.</b> Read-write, Volatile. Reset: Cold, 00h. Specifies the least significant valid bit of the address contained in MCA::L3::MCA_DEADDR_L3[ErrorAddr]. A value of 0 indicates that MCA::L3::MCA_DEADDR_L3[55:0] contains a valid byte address. A value of 6 indicates that MCA::L3::MCA_DEADDR_L3[55:6] contains a valid cache line address and that MCA::L3::MCA_DEADDR_L3[5:0] are not part of the address and should be ignored by error handling software. A value of 12 indicates that MCA::L3::MCA_DEADDR_L3[55:12] contain a valid 4KB memory page and that MCA::L3::MCA_DEADDR_L3[11:0] should be ignored by error handling software.
55:0	<b>ErrorAddr.</b> Read-write, Volatile. Reset: Cold, 00_0000_0000_0000h. Contains the address, if any, associated with the error logged in MCA::L3::MCA_DESTAT_L3. The lowest-order valid bit of the address is specified in MCA::L3::MCA_DEADDR_L3[LSB].

**MSRC001\_040[7...A] [L3 Machine Check Control Mask] (MCA::L3::MCA\_CTL\_MASK\_L3)**

Read-write. Reset: 0000\_0000\_0000\_0000h.

Inhibit detection of an error source.

\_ccx0\_inst7\_aliasMSR; MSRC001\_0407

\_ccx0\_inst8\_aliasMSR; MSRC001\_0408

_ccx0_inst9_aliasMSR; MSRC001_0409	
_ccx0_inst10_aliasMSR; MSRC001_040A	
Bits	Description
63:8	Reserved.
7	<b>Hwa</b> . Read-write. Reset: 0. Init: BIOS,1. L3 Hardware Assertion.
6	<b>XiVictimQueue</b> . Read-write. Reset: 0. L3 Victim Queue Parity Error.
5	<b>SdpParity</b> . Read-write. Reset: 0. SDP Parity Error from XI.
4	<b>DataArray</b> . Read-write. Reset: 0. L3M Data ECC Error.
3	<b>MultiHitTag</b> . Read-write. Reset: 0. L3M Tag Multi-way-hit Error.
2	<b>Tag</b> . Read-write. Reset: 0. L3M Tag ECC Error.
1	<b>MultiHitShadowTag</b> . Read-write. Reset: 0. Shadow Tag Macro Multi-way-hit Error.
0	<b>ShadowTag</b> . Read-write. Reset: 0. Shadow Tag Macro ECC Error.

### 3.2.5.8 CS

MSR0000_0450...MSRC000_2150 [CS Machine Check Control] (MCA::CS::MCA_CTL_CS)	
Read-write. Reset: 0000_0000_0000_0000h.	
0=Disables error reporting for the corresponding error. 1=Enables error reporting via machine check exception for the corresponding error. The MCA::CS::MCA_CTL_CS register must be enabled by the corresponding enable bit in Core::X86::Msr::MCG_CTL. Does not affect error detection, correction, or logging.	
_instCS0_aliasMSRLEGACY; MSR0000_0450	
_instCS1_aliasMSRLEGACY; MSR0000_0454	
_instCS0_aliasMSR; MSRC000_2140	
_instCS1_aliasMSR; MSRC000_2150	
Bits	Description
63:9	Reserved.
8	<b>SPF_ECC_ERR</b> . Read-write. Reset: 0. Probe Filter ECC Error: An ECC error occurred on a probe filter access.
7	<b>ATM_PAR_ERR</b> . Read-write. Reset: 0. Atomic Request Parity Error: Parity error on read of an atomic transaction.
6	<b>SDP_PAR_ERR</b> . Read-write. Reset: 0. Read Response Parity Error: Parity error on incoming read response data.
5	<b>FTI_PAR_ERR</b> . Read-write. Reset: 0. Request or Probe Parity Error: Parity error on incoming request or probe response data.
4	<b>FTI_RSP_NO_MTCH</b> . Read-write. Reset: 0. Unexpected Response: A response was received from the transport layer which does not match any request.
3	<b>FTI_ILL_RSP</b> . Read-write. Reset: 0. Illegal Response: An illegal response was received from the transport layer.
2	<b>FTI_SEC_VIOL</b> . Read-write. Reset: 0. Security Violation: A security violation was received from the transport layer.
1	<b>FTI_ADDR_VIOL</b> . Read-write. Reset: 0. Address Violation: An address violation was received from the transport layer.
0	<b>FTI_ILL_REQ</b> . Read-write. Reset: 0. Illegal Request: An illegal request was received from the transport layer.
MSR0000_0451...MSRC000_2151 [CS Machine Check Status] (MCA::CS::MCA_STATUS_CS)	
Reset: Cold,0000_0000_0000_0000h.	
Logs information associated with errors.	
_instCS0_aliasMSRLEGACY; MSR0000_0451	
_instCS1_aliasMSRLEGACY; MSR0000_0455	
_instCS0_aliasMSR; MSRC000_2141	
_instCS1_aliasMSR; MSRC000_2151	
Bits	Description
63	<b>Val</b> . Reset: Cold,0. 1=A valid error has been detected. This bit should be cleared by software after the register has been read.

	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
62	<b>Overflow.</b> Reset: Cold,0. 1=An error was detected while the valid bit (Val) was set; at least one error was not logged. Overflow is set independently of whether the existing error is overwritten. See 3.1.3 [Machine Check Errors].
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
61	<b>UC.</b> Reset: Cold,0. 1=The error was not corrected by hardware.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
60	<b>En.</b> Reset: Cold,0. 1=MCA error reporting is enabled for this error, as indicated by the corresponding bit in MCA::CS::MCA_CTL_CS. This bit is a copy of bit in MCA::CS::MCA_CTL_CS for this error.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
59	<b>MiscV.</b> Reset: Cold,0. 1=Valid thresholding in MCA::CS::MCA_MISC0_CS.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
58	<b>AddrV.</b> Reset: Cold,0. 1=MCA::CS::MCA_ADDR_CS contains address information associated with the error.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
57	<b>PCC.</b> Reset: Cold,0. 1=Hardware context held by the processor may have been corrupted. Continued operation of the system may have unpredictable results. The error is not recoverable or survivable, and the system should be reinitialized.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
56	<b>ErrCoreIdVal.</b> Reset: Cold,0. 1=The ErrCoreId field is valid.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
55	<b>TCC.</b> Reset: Cold,0. 0=The thread which consumed the error can be restarted reliably at the instruction pointer address pushed onto the exception handler stack if any uncorrected error has been corrected by software. 1=The thread which consumed the error is not restartable and must be terminated. Only meaningful when MCA::CS::MCA_STATUS_CS[PCC]=0.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
54	Reserved.
53	<b>SyndV.</b> Reset: Cold,0. 1=This error logged information in MCA::CS::MCA_SYND_CS. If MCA::CS::MCA_SYND_CS[ErrorPriority] is the same as the priority of the error in MCA::CS::MCA_STATUS_CS, then the information in MCA::CS::MCA_SYND_CS is associated with the error in MCA::CS::MCA_STATUS_CS.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
52:47	Reserved.
46	<b>CECC.</b> Reset: Cold,0. 1=The error was a correctable ECC error according to the restrictions of the ECC algorithm. UC indicates whether the error was actually corrected by the processor.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
45	<b>UECC.</b> Reset: Cold,0. 1=The error was an uncorrectable ECC error according to the restrictions of the ECC algorithm. UC indicates whether the error was actually corrected by the processor.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
44	<b>Deferred.</b> Reset: Cold,0. 1=A deferred error was created. A deferred error is the result of an uncorrectable data error which did not immediately cause a processor exception; an exception is deferred until the erroneous data is consumed.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
43	<b>Poison.</b> Reset: Cold,0. 1=The error was the result of attempting to consume poisoned data.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
42:38	Reserved.
37:32	<b>ErrCoreId.</b> Reset: Cold,00h. When ErrCoreIdVal=1 this field indicates which core within the processor is associated with the error; Otherwise this field is reserved.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
31:22	Reserved.
21:16	<b>ErrorCodeExt.</b> Reset: Cold,00h. Extended Error Code. This field is used to identify the error type for root cause



	analysis. This field indicates which bit position in MCA::CS::MCA_CTL_CS enables error reporting for the logged error.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
15:0	<b>ErrorCode.</b> Reset: Cold,0000h. Error code for this error. See 3.1.3.3 [Error Codes] for details on decoding this field.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.

Table 54: MCA\_STATUS\_CS[ErrorCodeExt] Decode

Error Type	Bits	Description
FTI_ILL_REQ	[6:0]	0x0
FTI_ADDR_VIOL	[6:0]	0x1
FTI_SEC_VIOL	[6:0]	0x2
FTI_ILL_RSP	[6:0]	0x3
FTI_RSP_NO_MTCH	[6:0]	0x4
FTI_PAR_ERR	[6:0]	0x5
SDP_PAR_ERR	[6:0]	0x6
ATM_PAR_ERR	[6:0]	0x7
SPF_ECC_ERR	[6:0]	0x8

**MSR0000\_0452...MSRC000\_2152 [CS Machine Check Address] (MCA::CS::MCA\_ADDR\_CS)**

Reset: Cold,0000_0000_0000_0000h.	
MCA::CS::MCA_ADDR_CS stores an address and other information associated with the error in MCA::CS::MCA_STATUS_CS. The register is only meaningful if MCA::CS::MCA_STATUS_CS[Val]=1 and MCA::CS::MCA_STATUS_CS[AddrV]=1.	
_instCS0_aliasMSRLEGACY; MSR0000_0452	
_instCS1_aliasMSRLEGACY; MSR0000_0456	
_instCS0_aliasMSR; MSRC000_2142	
_instCS1_aliasMSR; MSRC000_2152	
Bits	Description
63:62	Reserved.
61:56	<b>LSB.</b> Read-write, Volatile. Reset: Cold,00h. Specifies the least significant valid bit of the address contained in MCA::CS::MCA_ADDR_CS[ErrorAddr]. A value of 0 indicates that MCA::CS::MCA_ADDR_CS[55:0] contains a valid byte address. A value of 6 indicates that MCA::CS::MCA_ADDR_CS[55:6] contains a valid cache line address and that MCA::CS::MCA_ADDR_CS[5:0] are not part of the address and should be ignored by error handling software. A value of 12 indicates that MCA::CS::MCA_ADDR_CS[55:12] contain a valid 4KB memory page and that MCA::CS::MCA_ADDR_CS[11:0] should be ignored by error handling software.
55:0	<b>ErrorAddr.</b> Read-write, Volatile. Reset: Cold,00_0000_0000_0000h. Unless otherwise specified by an error, contains the address associated with the error logged in MCA::CS::MCA_STATUS_CS. For physical addresses, the most significant bit is given by Core::X86::Cpuid::LongModeInfo[PhysAddrSize].

Table 55: MCA\_ADDR\_CS Register

Error Type	Bits	Description
FTI_ILL_REQ	[55:0]	Reserved
FTI_ADDR_VIOL	[55:0]	Reserved
FTI_SEC_VIOL	[55:0]	Reserved
FTI_ILL_RSP	[55:0]	Reserved
FTI_RSP_NO_MTCH	[55:0]	Reserved
FTI_PAR_ERR	[55:0]	Reserved
SDP_PAR_ERR	[55:0]	Reserved
ATM_PAR_ERR	[55:0]	Reserved



SPF_ECC_ERR	[55:0]	Reserved
-------------	--------	----------

**MSR0000\_0453...MSRC000\_2153 [CS Machine Check Miscellaneous 0] (MCA::CS::MCA\_MISC0\_CS)**

Log miscellaneous information associated with errors.

\_instCS0\_aliasMSRLEGACY; MSR0000\_0453

\_instCS1\_aliasMSRLEGACY; MSR0000\_0457

\_instCS0\_aliasMSR; MSRC000\_2143

\_instCS1\_aliasMSR; MSRC000\_2153

Bits	Description
63	<b>Valid.</b> Reset: 1. 1=A valid CntP field is present in this register. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read-only.
62	<b>CntP.</b> Reset: 1. 1=A valid threshold counter is present. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read-only.
61	<b>Locked.</b> Reset: 0. 1=Writes to this register are ignored. This bit is set by BIOS to indicate that this register is not available for OS use. BIOS should set this bit if ThresholdIntType is set to SMI. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read-only.
60	<b>IntP.</b> Reset: 1. 1=ThresholdIntType can be used to generate interrupts. 0=ThresholdIntType and interrupt generation are not supported. AccessType: (Core::X86::Msr::HWCR[McStatusWrEn]   !MCA::CS::MCA_MISC0_CS[Locked]) ? Read-write : Read-only.
59:56	Reserved.
55:52	<b>LvtOffset.</b> Reset: 0h. One per die. 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). AccessType: (Core::X86::Msr::HWCR[McStatusWrEn]   !MCA::CS::MCA_MISC0_CS[Locked]) ? Read-write : Read-only.
51	<b>CntEn.</b> Reset: 0. 1=Count thresholding errors. AccessType: (Core::X86::Msr::HWCR[McStatusWrEn]   !MCA::CS::MCA_MISC0_CS[Locked]) ? Read-write : Read-only.
50:49	<b>ThresholdIntType.</b> Reset: Cold,0h. Specifies the type of interrupt signaled when Ovrlw is set and IntP==1. 00b = No interrupt. 01b = APIC based interrupt (see Core::X86::Msr::McaIntrCfg[ThresholdLvtOffset]). 10b = SMI trigger event. 11b = Reserved. AccessType: (Core::X86::Msr::HWCR[McStatusWrEn]   !MCA::CS::MCA_MISC0_CS[Locked]) ? Read-write : Read-only.
48	<b>Ovrlw.</b> Reset: Cold,0. Set by hardware when ErrCnt transitions from FFEh to FFFh. When this field is set, ErrCnt no longer increments. When this bit is set, the interrupt selected by the ThresholdIntType field is generated. AccessType: (Core::X86::Msr::HWCR[McStatusWrEn]   !MCA::CS::MCA_MISC0_CS[Locked]) ? Read-write : Read-only.
47:44	Reserved.
43:32	<b>ErrCnt.</b> Reset: Cold,000h. This is written by software to set the starting value of the error counter. This is incremented by hardware when errors are logged. When this counter overflows, it stays at FFFh (no rollover). The threshold value, written by software, is (FFFh - the desired error count (the number of errors necessary in order for an interrupt to be taken)); the desired error count of 0 (a write value of FFFh) is not supported. AccessType: (Core::X86::Msr::HWCR[McStatusWrEn]   !MCA::CS::MCA_MISC0_CS[Locked]) ? Read-write : Read-only.
31:24	<b>BlkPtr.</b> Read-write. Reset: 01h. 00h=Extended MISC MSR block is not valid. 01h=Extended MSR block is valid.
23:0	Reserved.

**MSRC000\_21[4...5]4 [CS Machine Check Configuration] (MCA::CS::MCA\_CONFIG\_CS)**

Reset: 0000\_0000\_0000\_0025h.

Controls configuration of the associated machine check bank.

_instCS0_aliasMSR; MSRC000_2144	
_instCS1_aliasMSR; MSRC000_2154	
Bits	Description
63:39	Reserved.
38:37	<b>DeferredIntType.</b> Read-write. Reset: 0h. Specifies the type of interrupt signaled when a deferred error is logged. 00b = No interrupt. 01b = APIC based interrupt (see Core::X86::Msr::McaIntrCfg[DeferredLvtOffset]). 10b = SMI trigger event. 11b = Reserved.
36:35	Reserved.
34	<b>LogDeferredInMcaStat.</b> Read-write. Reset: 0. Init: BIOS,1. 1=Log deferred errors in MCA::CS::MCA_STATUS_CS and MCA::CS::MCA_ADDR_CS in addition to MCA::CS::MCA_DESTAT_CS and MCA::CS::MCA_DEADDR_CS. 0=Only log deferred errors in MCA::CS::MCA_DESTAT_CS and MCA::CS::MCA_DEADDR_CS. This bit does not affect logging of deferred errors in MCA::CS::MCA_SYND_CS, MCA::CS::MCA_MISC0_CS.
33	Reserved.
32	<b>McaXEnable.</b> Read-write. Reset: 0. Init: BIOS,1. Check: 1. 1=Software has acknowledged support for the MCAX feature set. 0=Software has not acknowledged support for the MCAX feature set. All uncorrected and fatal errors will cause an ErrorEvent packet to be generated. Deferred error interrupts are configured via Core::X86::Msr::McaIntrCfg.
31:6	Reserved.
5	<b>DeferredIntTypeSupported.</b> Read-only. Reset: 1. 1=MCA::CS::MCA_CONFIG_CS[DeferredIntType] controls the type of interrupt generated on a deferred error. Deferred errors are supported in this bank only if MCA::CS::MCA_CONFIG_CS[DeferredErrorLoggingSupported]=1.
4:3	Reserved.
2	<b>DeferredErrorLoggingSupported.</b> Read-only. Reset: 1. 1=Deferred errors are supported in this MCA bank, and MCA::CS::MCA_CONFIG_CS[LogDeferredInMcaStat] controls the logging behavior of these errors. MCA::CS::MCA_DESTAT_CS and MCA::CS::MCA_DEADDR_CS are supported in this MCA bank. 0=Deferred errors are not supported in this bank.
1	Reserved.
0	<b>McaX.</b> Read-only. Reset: 1. 1=This bank provides Machine Check Architecture Extensions. Up to 4 additional MISC registers (MISC1-MISC4) are supported. MCA::CS::MCA_MISC0_CS[BlkPtr] indicates the presence of the additional MISC registers, but is not used to determine their MSR numbers. Deferred error interrupt type is specifiable by MCA bank. MCA::CS::MCA_STATUS_CS[TCC] is present.

#### MSRC000\_21[4...5]5 [CS IP Identification] (MCA::CS::MCA\_IPID\_CS)

Reset: 0000\_002E\_0000\_0000h.

The MCA::CS::MCA\_IPID\_CS register is used by software to determine what IP type and revision is associated with the MCA bank.

\_instCS0\_aliasMSR; MSRC000\_2145

\_instCS1\_aliasMSR; MSRC000\_2155

Bits	Description
63:48	<b>McaType.</b> Read-only. Reset: 0000h. The McaType of the MCA bank within this IP.
47:44	Reserved.
43:32	<b>HardwareID.</b> Read-only. Reset: 02Eh. The Hardware ID of the IP associated with this MCA bank.
31:0	<b>InstanceId.</b> Read-write. Reset: 0000_0000h. The instance ID of this IP. This is initialized to a unique ID per instance of this register.

#### MSRC000\_21[4...5]6 [CS Machine Check Syndrome] (MCA::CS::MCA\_SYND\_CS)

Read-write, Volatile. Reset: Cold, 0000\_0000\_0000\_0000h.

Logs physical location information associated with error in MCA::CS::MCA\_STATUS\_CS Thread 0

\_instCS0\_aliasMSR; MSRC000\_2146

\_instCS1\_aliasMSR; MSRC000\_2156

Bits	Description
------	-------------

63:48	Reserved.
47:32	<b>Syndrome.</b> Read-write, Volatile. Reset: Cold, 0000h. Contains the syndrome, if any, associated with the error logged in MCA::CS::MCA_STATUS_CS. The low-order bit of the syndrome is stored in bit 0, and the syndrome has a length specified by MCA::CS::MCA_SYND_CS[Length]. The Syndrome field is only valid when MCA::CS::MCA_SYND_CS[Length] is not 0.
31:27	Reserved.
26:24	<b>ErrorPriority.</b> Read-write, Volatile. Reset: Cold, 0h. Encodes the priority of the error logged in MCA::CS::MCA_SYND_CS. 3'b000 = No error; 3'b001 = Reserved; 3'b010 = Corrected Error; 3'b011 = Deferred Error; 3'b100 = Uncorrected Error; 3'b101 = Fatal Error; all others reserved.
23:18	<b>Length.</b> Read-write, Volatile. Reset: Cold, 00h. Specifies the length in bits of the syndrome contained in MCA::CS::MCA_SYND_CS[Syndrome]. A value of 0 indicates that there is no valid syndrome in MCA::CS::MCA_SYND_CS. For example, a syndrome length of 9 means that MCA::CS::MCA_SYND_CS[Syndrome] bits [8:0] contains a valid syndrome.
17:0	<b>ErrorInformation.</b> Read-write, Volatile. Reset: Cold, 0_0000h. Contains error-specific information about the location of the error. Decoding is available in Table 56 [MCA_SYND_CS Register].

Table 56: MCA\_SYND\_CS Register

Error Type	Bits	Description
FTI_ILL_REQ	[8:0]	
FTI_ADDR_VIOL	[8:0]	
FTI_SEC_VIOL	[8:0]	
FTI_ILL_RSP	[2:0]	
FTI_RSP_NO_MTCH	[2:0]	
FTI_PAR_ERR	[7:0]	
SDP_PAR_ERR	[7:0]	
ATM_PAR_ERR	[7:0]	
SPF_ECC_ERR	[17:0]	

**MSRC000\_21[4...5]8 [CS Machine Check Deferred Error Status] (MCA::CS::MCA\_DESTAT\_CS)**

Read-write, Volatile. Reset: Cold, 0000\_0000\_0000\_0000h.

Holds status information for the first deferred error seen in this bank.

\_instCS0\_aliasMSR; MSRC000\_2148

\_instCS1\_aliasMSR; MSRC000\_2158

Bits	Description
63	<b>Val.</b> Read-write, Volatile. Reset: Cold, 0. 1=A valid error has been detected (whether it is enabled or not).
62	<b>Overflow.</b> Read-write, Volatile. Reset: Cold, 0. 1=An error was detected while the valid bit (Val) was set; at least one error was not logged. Overflow is set independently of whether the existing error is overwritten. (See the section on overwrite priorities.)
61:59	Reserved.
58	<b>AddrV.</b> Read-write, Volatile. Reset: Cold, 0. 1=MCA::CS::MCA_DEADDR_CS contains address information associated with the error.
57:54	Reserved.
53	<b>SyndV.</b> Read-write, Volatile. Reset: Cold, 0. 1=This error logged information in MCA::CS::MCA_SYND_CS. If MCA::CS::MCA_SYND_CS[ErrorPriority] is the same as the priority of the error in MCA::CS::MCA_STATUS_CS, then the information in MCA::CS::MCA_SYND_CS is associated with the error in MCA::CS::MCA_DESTAT_CS.
52:45	Reserved.
44	<b>Deferred.</b> Read-write, Volatile. Reset: Cold, 0. 1=A deferred error was created. A deferred error is the result of an uncorrectable data error which did not immediately cause a processor exception; poison is created and an exception is deferred until the poison data is consumed.

43:0	Reserved.
------	-----------

**MSRC000\_21[4...5]9 [CS Deferred Error Address] (MCA::CS::MCA\_DEADDR\_CS)**

Reset: Cold,0000\_0000\_0000\_0000h.

The MCA::CS::MCA\_DEADDR\_CS register stores the address associated with the error in MCA::CS::MCA\_DESTAT\_CS. The register is only meaningful if MCA::CS::MCA\_DESTAT\_CS[Val]=1 and MCA::CS::MCA\_DESTAT\_CS[AddrV]=1. The lowest valid bit of the address is defined by MCA::CS::MCA\_DEADDR\_CS[LSB].

\_instCS0\_aliasMSR; MSRC000\_2149

\_instCS1\_aliasMSR; MSRC000\_2159

Bits	Description
63:62	Reserved.
61:56	<b>LSB.</b> Read-write, Volatile. Reset: Cold,00h. Specifies the least significant valid bit of the address contained in MCA::CS::MCA_DEADDR_CS[ErrorAddr]. A value of 0 indicates that MCA::CS::MCA_DEADDR_CS[55:0] contains a valid byte address. A value of 6 indicates that MCA::CS::MCA_DEADDR_CS[55:6] contains a valid cache line address and that MCA::CS::MCA_DEADDR_CS[5:0] are not part of the address and should be ignored by error handling software. A value of 12 indicates that MCA::CS::MCA_DEADDR_CS[55:12] contain a valid 4KB memory page and that MCA::CS::MCA_DEADDR_CS[11:0] should be ignored by error handling software.
55:0	<b>ErrorAddr.</b> Read-write, Volatile. Reset: Cold,00_0000_0000_0000h. Contains the address, if any, associated with the error logged in MCA::CS::MCA_DESTAT_CS. The lowest-order valid bit of the address is specified in MCA::CS::MCA_DEADDR_CS[LSB].

**MSRC001\_041[4...5] [CS Machine Check Control Mask] (MCA::CS::MCA\_CTL\_MASK\_CS)**

Read-write. Reset: 0000\_0000\_0000\_0000h.

Inhibit detection of an error source.

\_instCS0\_aliasMSR; MSRC001\_0414

\_instCS1\_aliasMSR; MSRC001\_0415

Bits	Description
63:9	Reserved.
8	<b>SPF_ECC_ERR.</b> Read-write. Reset: 0. Probe Filter ECC Error: An ECC error occurred on a probe filter access.
7	<b>ATM_PAR_ERR.</b> Read-write. Reset: 0. Atomic Request Parity Error: Parity error on read of an atomic transaction.
6	<b>SDP_PAR_ERR.</b> Read-write. Reset: 0. Read Response Parity Error: Parity error on incoming read response data.
5	<b>FTI_PAR_ERR.</b> Read-write. Reset: 0. Request or Probe Parity Error: Parity error on incoming request or probe response data.
4	<b>FTI_RSP_NO_MTCH.</b> Read-write. Reset: 0. Unexpected Response: A response was received from the transport layer which does not match any request.
3	<b>FTI_ILL_RSP.</b> Read-write. Reset: 0. Illegal Response: An illegal response was received from the transport layer.
2	<b>FTI_SEC_VIOL.</b> Read-write. Reset: 0. Security Violation: A security violation was received from the transport layer.
1	<b>FTI_ADDR_VIOL.</b> Read-write. Reset: 0. Address Violation: An address violation was received from the transport layer.
0	<b>FTI_ILL_REQ.</b> Read-write. Reset: 0. Illegal Request: An illegal request was received from the transport layer.

### 3.2.5.9 PIE

**MSR0000\_0458...MSRC000\_2160 [PIE Machine Check Control] (MCA::PIE::MCA\_CTL\_PIE)**

Read-write. Reset: 0000\_0000\_0000\_0000h.

0=Disables error reporting for the corresponding error. 1=Enables error reporting via machine check exception for the corresponding error. The MCA::PIE::MCA\_CTL\_PIE register must be enabled by the corresponding enable bit in

Core::X86::Msr::MCG_CTL. Does not affect error detection, correction, or logging.	
_instPIE_aliasMSRLEGACY; MSR0000_0458	
_instPIE_aliasMSR; MSRC000_2160	
Bits	Description
63:4	Reserved.
3	<b>FTI_DAT_STAT</b> . Read-write. Reset: 0. Poison data consumption: Poison data was written to an internal PIE register.
2	<b>GMI</b> . Read-write. Reset: 0. Link Error: An error occurred on a GMI or xGMI link.
1	<b>CSW</b> . Read-write. Reset: 0. Register security violation: A security violation was detected on an access to an internal PIE register.
0	<b>HW_ASSERT</b> . Read-write. Reset: 0. Hardware Assert: A hardware assert was detected.

#### MSR0000\_0459...MSRC000\_2161 [PIE Machine Check Status] (MCA::PIE::MCA\_STATUS\_PIE)

Reset: Cold,0000_0000_0000_0000h.	
Logs information associated with errors.	
_instPIE_aliasMSRLEGACY; MSR0000_0459	
_instPIE_aliasMSR; MSRC000_2161	
Bits	Description
63	<b>Val</b> . Reset: Cold,0. 1=A valid error has been detected. This bit should be cleared by software after the register has been read. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
62	<b>Overflow</b> . Reset: Cold,0. 1=An error was detected while the valid bit (Val) was set; at least one error was not logged. Overflow is set independently of whether the existing error is overwritten. See 3.1.3 [Machine Check Errors]. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
61	<b>UC</b> . Reset: Cold,0. 1=The error was not corrected by hardware. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
60	<b>En</b> . Reset: Cold,0. 1=MCA error reporting is enabled for this error, as indicated by the corresponding bit in MCA::PIE::MCA_CTL_PIE. This bit is a copy of bit in MCA::PIE::MCA_CTL_PIE for this error. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
59	<b>MiscV</b> . Reset: Cold,0. 1=Valid thresholding in MCA::PIE::MCA_MISC0_PIE. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
58	<b>AddrV</b> . Reset: Cold,0. 1=MCA::PIE::MCA_ADDR_PIE contains address information associated with the error. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
57	<b>PCC</b> . Reset: Cold,0. 1=Hardware context held by the processor may have been corrupted. Continued operation of the system may have unpredictable results. The error is not recoverable or survivable, and the system should be reinitialized. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
56	<b>ErrCoreIdVal</b> . Reset: Cold,0. 1=The ErrCoreId field is valid. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
55	<b>TCC</b> . Reset: Cold,0. 0=The thread which consumed the error can be restarted reliably at the instruction pointer address pushed onto the exception handler stack if any uncorrected error has been corrected by software. 1=The thread which consumed the error is not restartable and must be terminated. Only meaningful when MCA::PIE::MCA_STATUS_PIE[PCC]=0. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
54	Reserved.
53	<b>SyndV</b> . Reset: Cold,0. 1=This error logged information in MCA::PIE::MCA_SYND_PIE. If MCA::PIE::MCA_SYND_PIE[ErrorPriority] is the same as the priority of the error in MCA::PIE::MCA_STATUS_PIE, then the information in MCA::PIE::MCA_SYND_PIE is associated with the error in MCA::PIE::MCA_STATUS_PIE. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.



52:47	Reserved.
46	<b>CECC.</b> Reset: Cold,0. 1=The error was a correctable ECC error according to the restrictions of the ECC algorithm. UC indicates whether the error was actually corrected by the processor. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
45	<b>UECC.</b> Reset: Cold,0. 1=The error was an uncorrectable ECC error according to the restrictions of the ECC algorithm. UC indicates whether the error was actually corrected by the processor. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
44	<b>Deferred.</b> Reset: Cold,0. 1=A deferred error was created. A deferred error is the result of an uncorrectable data error which did not immediately cause a processor exception; an exception is deferred until the erroneous data is consumed. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
43	<b>Poison.</b> Reset: Cold,0. 1=The error was the result of attempting to consume poisoned data. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
42:38	Reserved.
37:32	<b>ErrCoreId.</b> Reset: Cold,00h. When ErrCoreIdVal=1 this field indicates which core within the processor is associated with the error; Otherwise this field is reserved. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
31:22	Reserved.
21:16	<b>ErrorCodeExt.</b> Reset: Cold,00h. Extended Error Code. This field is used to identify the error type for root cause analysis. This field indicates which bit position in MCA::PIE::MCA_CTL_PIE enables error reporting for the logged error. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
15:0	<b>ErrorCode.</b> Reset: Cold,0000h. Error code for this error. See 3.1.3.3 [Error Codes] for details on decoding this field. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.

Table 57: MCA\_STATUS\_PIE[ErrorCodeExt] Decode

Error Type	Bits	Description
HW_ASSERT	[6:0]	0x0
CSW	[6:0]	0x1
GMI	[6:0]	0x2
FTI_DAT_STAT	[6:0]	0x3

**MSR0000\_045A...MSRC000\_2162 [PIE Machine Check Address] (MCA::PIE::MCA\_ADDR\_PIE)**

Read-only. Reset: Cold,0000_0000_0000_0000h.	
MCA::PIE::MCA_ADDR_PIE stores an address and other information associated with the error in MCA::PIE::MCA_STATUS_PIE. The register is only meaningful if MCA::PIE::MCA_STATUS_PIE[Val]=1 and MCA::PIE::MCA_STATUS_PIE[AddrV]=1.	
_instPIE_aliasMSRLEGACY; MSR0000_045A	
_instPIE_aliasMSR; MSRC000_2162	
Bits	Description
63:62	Reserved.
61:56	<b>LSB.</b> Read-only. Reset: Cold,00h. Specifies the least significant valid bit of the address contained in MCA::PIE::MCA_ADDR_PIE[ErrorAddr]. A value of 0 indicates that MCA::PIE::MCA_ADDR_PIE[55:0] contains a valid byte address. A value of 6 indicates that MCA::PIE::MCA_ADDR_PIE[55:6] contains a valid cache line address and that MCA::PIE::MCA_ADDR_PIE[5:0] are not part of the address and should be ignored by error handling software. A value of 12 indicates that MCA::PIE::MCA_ADDR_PIE[55:12] contain a valid 4KB memory page and that MCA::PIE::MCA_ADDR_PIE[11:0] should be ignored by error handling software.
55:0	<b>ErrorAddr.</b> Read-only. Reset: Cold,00_0000_0000_0000h. Contains the address, if any, associated with the error logged in MCA::PIE::MCA_STATUS_PIE.

Table 58: MCA\_ADDR\_PIE Register

Error Type	Bits	Description
HW_ASSERT	[55:0]	Reserved
CSW	[55:0]	Reserved
GMI	[55:0]	Reserved
FTI_DAT_STAT	[55:0]	Reserved

**MSR0000\_045B...MSRC000\_2163 [PIE Machine Check Miscellaneous 0] (MCA::PIE::MCA\_MISC0\_PIE)**

Log miscellaneous information associated with errors.

\_instPIE\_aliasMSRLEGACY; MSR0000\_045B

\_instPIE\_aliasMSR; MSRC000\_2163

Bits	Description
63	<b>Valid.</b> Reset: 1. 1=A valid CntP field is present in this register. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read-only.
62	<b>CntP.</b> Reset: 1. 1=A valid threshold counter is present. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read-only.
61	<b>Locked.</b> Reset: 0. 1=Writes to this register are ignored. This bit is set by BIOS to indicate that this register is not available for OS use. BIOS should set this bit if ThresholdIntType is set to SMI. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read-only.
60	<b>IntP.</b> Reset: 1. 1=ThresholdIntType can be used to generate interrupts. 0=ThresholdIntType and interrupt generation are not supported. AccessType: (Core::X86::Msr::HWCR[McStatusWrEn]   !MCA::PIE::MCA_MISC0_PIE[Locked]) ? Read-write : Read-only.
59:56	Reserved.
55:52	<b>LvtOffset.</b> Reset: 0h. One per die. 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). AccessType: (Core::X86::Msr::HWCR[McStatusWrEn]   !MCA::PIE::MCA_MISC0_PIE[Locked]) ? Read-write : Read-only.
51	<b>CntEn.</b> Reset: 0. 1=Count thresholding errors. AccessType: (Core::X86::Msr::HWCR[McStatusWrEn]   !MCA::PIE::MCA_MISC0_PIE[Locked]) ? Read-write : Read-only.
50:49	<b>ThresholdIntType.</b> Reset: Cold,0h. Specifies the type of interrupt signaled when Ovrflw is set and IntP==1. 00b = No interrupt. 01b = APIC based interrupt (see Core::X86::Msr::McaIntrCfg[ThresholdLvtOffset]). 10b = SMI trigger event. 11b = Reserved. AccessType: (Core::X86::Msr::HWCR[McStatusWrEn]   !MCA::PIE::MCA_MISC0_PIE[Locked]) ? Read-write : Read-only.
48	<b>Ovrflw.</b> Reset: Cold,0. Set by hardware when ErrCnt transitions from FFEh to FFFh. When this field is set, ErrCnt no longer increments. When this bit is set, the interrupt selected by the ThresholdIntType field is generated. AccessType: (Core::X86::Msr::HWCR[McStatusWrEn]   !MCA::PIE::MCA_MISC0_PIE[Locked]) ? Read-write : Read-only.
47:44	Reserved.
43:32	<b>ErrCnt.</b> Reset: Cold,000h. This is written by software to set the starting value of the error counter. This is incremented by hardware when errors are logged. When this counter overflows, it stays at FFFh (no rollover). The threshold value, written by software, is (FFFh - the desired error count (the number of errors necessary in order for an interrupt to be taken)); the desired error count of 0 (a write value of FFFh) is not supported. AccessType: (Core::X86::Msr::HWCR[McStatusWrEn]   !MCA::PIE::MCA_MISC0_PIE[Locked]) ? Read-write : Read-only.
31:24	<b>BlkPtr.</b> Read-write. Reset: 01h. 00h=Extended MISC MSR block is not valid. 01h=Extended MSR block is valid.
23:0	Reserved.



**MSRC000\_2164 [PIE Machine Check Configuration] (MCA::PIE::MCA\_CONFIG\_PIE)**

Reset: 0000_0002_0000_0025h.	
Controls configuration of the associated machine check bank.	
_instPIE_aliasMSR; MSRC000_2164	
Bits	Description
63:39	Reserved.
38:37	<b>DeferredIntType.</b> Read-write. Reset: 0h. Specifies the type of interrupt signaled when a deferred error is logged. 00b = No interrupt. 01b = APIC based interrupt (see Core::X86::Msr::McaIntrCfg[DeferredLvtOffset]). 10b = SMI trigger event. 11b = Reserved.
36:35	Reserved.
34	<b>LogDeferredInMcaStat.</b> Read-write. Reset: 0. Init: BIOS,1. 1=Log deferred errors in MCA::PIE::MCA_STATUS_PIE and MCA::PIE::MCA_ADDR_PIE in addition to MCA::PIE::MCA_DESTAT_PIE and MCA::PIE::MCA_DEADDR_PIE. 0=Only log deferred errors in MCA::PIE::MCA_DESTAT_PIE and MCA::PIE::MCA_DEADDR_PIE. This bit does not affect logging of deferred errors in MCA::PIE::MCA_SYND_PIE, MCA::PIE::MCA_MISC0_PIE.
33	Reserved.
32	<b>McaXEnable.</b> Read-write. Reset: 0. Init: BIOS,1. Check: 1. 1=Software has acknowledged support for the MCAX feature set. 0=Software has not acknowledged support for the MCAX feature set. All uncorrected and fatal errors will cause an ErrorEvent packet to be generated. Deferred error interrupts are configured via Core::X86::Msr::McaIntrCfg.
31:6	Reserved.
5	<b>DeferredIntTypeSupported.</b> Read-only. Reset: 1. 1=MCA::PIE::MCA_CONFIG_PIE[DeferredIntType] controls the type of interrupt generated on a deferred error. Deferred errors are supported in this bank only if MCA::PIE::MCA_CONFIG_PIE[DeferredErrorLoggingSupported]=1.
4:3	Reserved.
2	<b>DeferredErrorLoggingSupported.</b> Read-only. Reset: 1. 1=Deferred errors are supported in this MCA bank, and MCA::PIE::MCA_CONFIG_PIE[LogDeferredInMcaStat] controls the logging behavior of these errors. MCA::PIE::MCA_DESTAT_PIE and MCA::PIE::MCA_DEADDR_PIE are supported in this MCA bank. 0=Deferred errors are not supported in this bank.
1	Reserved.
0	<b>McaX.</b> Read-only. Reset: 1. 1=This bank provides Machine Check Architecture Extensions. Up to 4 additional MISC registers (MISC1-MISC4) are supported. MCA::PIE::MCA_MISC0_PIE[BlkPtr] indicates the presence of the additional MISC registers, but is not used to determine their MSR numbers. Deferred error interrupt type is specifiable by MCA bank. MCA::PIE::MCA_STATUS_PIE[TCC] is present.

**MSRC000\_2165 [PIE IP Identification] (MCA::PIE::MCA\_IPID\_PIE)**

Reset: 0001_002E_0000_0000h.	
The MCA::PIE::MCA_IPID_PIE register is used by software to determine what IP type and revision is associated with the MCA bank.	
_instPIE_aliasMSR; MSRC000_2165	
Bits	Description
63:48	<b>McaType.</b> Read-only. Reset: 0001h. The McaType of the MCA bank within this IP.
47:44	Reserved.
43:32	<b>HardwareID.</b> Read-only. Reset: 02Eh. The Hardware ID of the IP associated with this MCA bank.
31:0	<b>InstanceId.</b> Read-write. Reset: 0000_0000h. The instance ID of this IP. This is initialized to a unique ID per instance of this register.

**MSRC000\_2166 [PIE Machine Check Syndrome] (MCA::PIE::MCA\_SYND\_PIE)**

Read-write, Volatile. Reset: Cold, 0000_0000_0000_0000h.	
Logs physical location information associated with error in MCA::PIE::MCA_STATUS_PIE Thread 0	
_instPIE_aliasMSR; MSRC000_2166	

Bits	Description
63:33	Reserved.
32	<b>Syndrome.</b> Read-write, Volatile. Reset: Cold, 0. Contains the syndrome, if any, associated with the error logged in MCA::PIE::MCA_STATUS_PIE. The low-order bit of the syndrome is stored in bit 0, and the syndrome has a length specified by MCA::PIE::MCA_SYND_PIE[Length]. The Syndrome field is only valid when MCA::PIE::MCA_SYND_PIE[Length] is not 0.
31:27	Reserved.
26:24	<b>ErrorPriority.</b> Read-write, Volatile. Reset: Cold, 0h. Encodes the priority of the error logged in MCA::PIE::MCA_SYND_PIE. 3'b000 = No error; 3'b001 = Reserved; 3'b010 = Corrected Error; 3'b011 = Deferred Error; 3'b100 = Uncorrected Error; 3'b101 = Fatal Error; all others reserved.
23:18	<b>Length.</b> Read-write, Volatile. Reset: Cold, 00h. Specifies the length in bits of the syndrome contained in MCA::PIE::MCA_SYND_PIE[Syndrome]. A value of 0 indicates that there is no valid syndrome in MCA::PIE::MCA_SYND_PIE. For example, a syndrome length of 9 means that MCA::PIE::MCA_SYND_PIE[Syndrome] bits [8:0] contains a valid syndrome.
17:0	<b>ErrorInformation.</b> Read-write, Volatile. Reset: Cold, 0_0000h. Contains error-specific information about the location of the error. Decoding is available in Table 59 [MCA_SYND_PIE Register].

Table 59: MCA\_SYND\_PIE Register

Error Type	Bits	Description
HW_ASSERT	[17:0]	Reserved
CSW	[17:0]	Reserved
GMI	[4:0]	
FTI_DAT_STAT	[3:0]	

**MSRC000\_2168 [PIE Machine Check Deferred Error Status] (MCA::PIE::MCA\_DESTAT\_PIE)**

Read-write, Volatile. Reset: Cold, 0000_0000_0000_0000h.	
Holds status information for the first deferred error seen in this bank.	
_instPIE_aliasMSR; MSRC000_2168	
Bits	Description
63	<b>Val.</b> Read-write, Volatile. Reset: Cold, 0. 1=A valid error has been detected (whether it is enabled or not).
62	<b>Overflow.</b> Read-write, Volatile. Reset: Cold, 0. 1=An error was detected while the valid bit (Val) was set; at least one error was not logged. Overflow is set independently of whether the existing error is overwritten. (See the section on overwrite priorities.)
61:59	Reserved.
58	<b>AddrV.</b> Read-write, Volatile. Reset: Cold, 0. 1=MCA::PIE::MCA_DEADDR_PIE contains address information associated with the error.
57:54	Reserved.
53	<b>SyndV.</b> Read-write, Volatile. Reset: Cold, 0. 1=This error logged information in MCA::PIE::MCA_SYND_PIE. If MCA::PIE::MCA_SYND_PIE[ErrorPriority] is the same as the priority of the error in MCA::PIE::MCA_STATUS_PIE, then the information in MCA::PIE::MCA_SYND_PIE is associated with the error in MCA::PIE::MCA_DESTAT_PIE.
52:45	Reserved.
44	<b>Deferred.</b> Read-write, Volatile. Reset: Cold, 0. 1=A deferred error was created. A deferred error is the result of an uncorrectable data error which did not immediately cause a processor exception; poison is created and an exception is deferred until the poison data is consumed.
43:0	Reserved.

**MSRC000\_2169 [PIE Deferred Error Address] (MCA::PIE::MCA\_DEADDR\_PIE)**

Reset: Cold, 0000_0000_0000_0000h.	
The MCA::PIE::MCA_DEADDR_PIE register stores the address associated with the error in	

MCA::PIE::MCA_DESTAT_PIE. The register is only meaningful if MCA::PIE::MCA_DESTAT_PIE[Val]=1 and MCA::PIE::MCA_DESTAT_PIE[AddrV]=1. The lowest valid bit of the address is defined by MCA::PIE::MCA_DEADDR_PIE[LSB].	
_instPIE_aliasMSR; MSRC000_2169	
Bits	Description
63:62	Reserved.
61:56	<b>LSB.</b> Read-write, Volatile. Reset: Cold, 00h. Specifies the least significant valid bit of the address contained in MCA::PIE::MCA_DEADDR_PIE[ErrorAddr]. A value of 0 indicates that MCA::PIE::MCA_DEADDR_PIE[55:0] contains a valid byte address. A value of 6 indicates that MCA::PIE::MCA_DEADDR_PIE[55:6] contains a valid cache line address and that MCA::PIE::MCA_DEADDR_PIE[5:0] are not part of the address and should be ignored by error handling software. A value of 12 indicates that MCA::PIE::MCA_DEADDR_PIE[55:12] contain a valid 4KB memory page and that MCA::PIE::MCA_DEADDR_PIE[11:0] should be ignored by error handling software.
55:0	<b>ErrorAddr.</b> Read-write, Volatile. Reset: Cold, 00_0000_0000_0000h. Contains the address, if any, associated with the error logged in MCA::PIE::MCA_DESTAT_PIE. The lowest-order valid bit of the address is specified in MCA::PIE::MCA_DEADDR_PIE[LSB].

#### MSRC001\_0416 [PIE Machine Check Control Mask] (MCA::PIE::MCA\_CTL\_MASK\_PIE)

Read-write. Reset: 0000_0000_0000_0000h.	
Inhibit detection of an error source.	
_instPIE_aliasMSR; MSRC001_0416	
Bits	Description
63:4	Reserved.
3	<b>FTI_DAT_STAT.</b> Read-write. Reset: 0. Poison data consumption: Poison data was written to an internal PIE register.
2	<b>GMI.</b> Read-write. Reset: 0. Link Error: An error occurred on a GMI or xGMI link.
1	<b>CSW.</b> Read-write. Reset: 0. Register security violation: A security violation was detected on an access to an internal PIE register.
0	<b>HW_ASSERT.</b> Read-write. Reset: 0. Hardware Assert: A hardware assert was detected.

### 3.2.5.10 UMC

#### MSR0000\_043C...MSRC000\_2100 [UMC Machine Check Control] (MCA::UMC::MCA\_CTL\_UMC)

Read-write. Reset: 0000_0000_0000_0000h.	
0=Disables error reporting for the corresponding error. 1=Enables error reporting via machine check exception for the corresponding error. The MCA::UMC::MCA_CTL_UMC register must be enabled by the corresponding enable bit in Core::X86::Msr::MCG_CTL. Does not affect error detection, correction, or logging.	
_instUMC_umc0_aliasMSRLEGACY; MSR0000_043C	
_instUMC_umc1_aliasMSRLEGACY; MSR0000_0440	
_instUMC_umc0_aliasMSR; MSRC000_20F0	
_instUMC_umc1_aliasMSR; MSRC000_2100	
Bits	Description
63:6	Reserved.
5	<b>WriteDataCrcErr.</b> Read-write. Reset: 0. Write data CRC error. A write data CRC error on the DRAM data bus.
4	<b>AddressCommandParityErr.</b> Read-write. Reset: 0. Address/command parity error. A parity error on the DRAM address/command bus.
3	<b>ApbErr.</b> Read-write. Reset: 0. Advanced peripheral bus error. An error on the advanced peripheral bus.
2	<b>SdpParityErr.</b> Read-write. Reset: 0. SDP parity error. A parity error on write data from the data fabric.
1	<b>WriteDataPoisonErr.</b> Read-write. Reset: 0. Data poison error.
0	<b>DramEccErr.</b> Read-write. Reset: 0. DRAM ECC error. An ECC error on a DRAM read.

**MSR0000\_043D...MSRC000\_2101 [UMC Machine Check Status] (MCA::UMC::MCA\_STATUS\_UMC)**

Reset: Cold,0000\_0000\_0000\_0000h.

Logs information associated with errors.

\_instUMC\_umc0\_aliasMSRLEGACY; MSR0000\_043D

\_instUMC\_umc1\_aliasMSRLEGACY; MSR0000\_0441

\_instUMC\_umc0\_aliasMSR; MSRC000\_20F1

\_instUMC\_umc1\_aliasMSR; MSRC000\_2101

Bits	Description
63	<b>Val.</b> Reset: Cold,0. 1=A valid error has been detected. This bit should be cleared by software after the register has been read. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
62	<b>Overflow.</b> Reset: Cold,0. 1=An error was detected while the valid bit (Val) was set; at least one error was not logged. Overflow is set independently of whether the existing error is overwritten. See 3.1.3 [Machine Check Errors]. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
61	<b>UC.</b> Reset: Cold,0. 1=The error was not corrected by hardware. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
60	<b>En.</b> Reset: Cold,0. 1=MCA error reporting is enabled for this error, as indicated by the corresponding bit in MCA::UMC::MCA_CTL_UMC. This bit is a copy of bit in MCA::UMC::MCA_CTL_UMC for this error. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
59	<b>MiscV.</b> Reset: Cold,0. 1=Valid thresholding in MCA::UMC::MCA_MISC0_UMC. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
58	<b>AddrV.</b> Reset: Cold,0. 1=MCA::UMC::MCA_ADDR_UMC contains address information associated with the error. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
57	<b>PCC.</b> Reset: Cold,0. 1=Hardware context held by the processor may have been corrupted. Continued operation of the system may have unpredictable results. The error is not recoverable or survivable, and the system should be reinitialized. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
56	<b>ErrCoreIdVal.</b> Reset: Cold,0. 1=The ErrCoreId field is valid. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
55	<b>TCC.</b> Reset: Cold,0. 0=The thread which consumed the error can be restarted reliably at the instruction pointer address pushed onto the exception handler stack if any uncorrected error has been corrected by software. 1=The thread which consumed the error is not restartable and must be terminated. Only meaningful when MCA::UMC::MCA_STATUS_UMC[PCC]=0. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
54	Reserved.
53	<b>SyndV.</b> Reset: Cold,0. 1=This error logged information in MCA::UMC::MCA_SYND_UMC. If MCA::UMC::MCA_SYND_UMC[ErrorPriority] is the same as the priority of the error in MCA::UMC::MCA_STATUS_UMC, then the information in MCA::UMC::MCA_SYND_UMC is associated with the error in MCA::UMC::MCA_STATUS_UMC. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
52:47	Reserved.
46	<b>CECC.</b> Reset: Cold,0. 1=The error was a correctable ECC error according to the restrictions of the ECC algorithm. UC indicates whether the error was actually corrected by the processor. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
45	<b>UECC.</b> Reset: Cold,0. 1=The error was an uncorrectable ECC error according to the restrictions of the ECC algorithm. UC indicates whether the error was actually corrected by the processor. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
44	<b>Deferred.</b> Reset: Cold,0. 1=A deferred error was created. A deferred error is the result of an uncorrectable data error which did not immediately cause a processor exception; an exception is deferred until the erroneous data is

	consumed.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
43	<b>Poison.</b> Reset: Cold,0. 1=The error was the result of attempting to consume poisoned data.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
42:38	Reserved.
37:32	<b>ErrCoreId.</b> Reset: Cold,00h. When ErrCoreIdVal=1 this field indicates which core within the processor is associated with the error; Otherwise this field is reserved.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
31:22	Reserved.
21:16	<b>ErrorCodeExt.</b> Reset: Cold,00h. Extended Error Code. This field is used to identify the error type for root cause analysis. This field indicates which bit position in MCA::UMC::MCA_CTL_UMC enables error reporting for the logged error.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
15:0	<b>ErrorCode.</b> Reset: Cold,0000h. Error code for this error. See 3.1.3.3 [Error Codes] for details on decoding this field.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.

Table 60: MCA\_STATUS\_UMC[ErrorCodeExt] Decode

Error Type	Bits	Description
DramEccErr	[6:0]	0x0
WriteDataPoisonErr	[6:0]	0x1
SdpParityErr	[6:0]	0x2
ApbErr	[6:0]	0x3
AddressCommandParityErr	[6:0]	0x4
WriteDataCrcErr	[6:0]	0x5

**MSR0000\_043E...MSRC000\_2102 [UMC Machine Check Address] (MCA::UMC::MCA\_ADDR\_UMC)**

Reset: Cold,0000\_0000\_0000\_0000h.

MCA::UMC::MCA\_ADDR\_UMC stores an address and other information associated with the error in MCA::UMC::MCA\_STATUS\_UMC. The register is only meaningful if MCA::UMC::MCA\_STATUS\_UMC[Val]=1 and MCA::UMC::MCA\_STATUS\_UMC[AddrV]=1.

\_instUMC\_umc0\_aliasMSRLEGACY; MSR0000\_043E\_instUMC\_umc1\_aliasMSRLEGACY; MSR0000\_0442\_instUMC\_umc0\_aliasMSR; MSRC000\_20F2\_instUMC\_umc1\_aliasMSR; MSRC000\_2102

Bits	Description
63:62	Reserved.
61:56	<b>LSB.</b> Read-write, Volatile. Reset: Cold,00h. Specifies the least significant valid bit of the address contained in MCA::UMC::MCA_ADDR_UMC[ErrorAddr]. A value of 0 indicates that MCA::UMC::MCA_ADDR_UMC[55:0] contains a valid byte address. A value of 6 indicates that MCA::UMC::MCA_ADDR_UMC[55:6] contains a valid cache line address and that MCA::UMC::MCA_ADDR_UMC[5:0] are not part of the address and should be ignored by error handling software. A value of 12 indicates that MCA::UMC::MCA_ADDR_UMC[55:12] contain a valid 4KB memory page and that MCA::UMC::MCA_ADDR_UMC[11:0] should be ignored by error handling software.
55:0	<b>ErrorAddr.</b> Read-write, Volatile. Reset: Cold,00_0000_0000_0000h. Unless otherwise specified by an error, contains the address associated with the error logged in MCA::UMC::MCA_STATUS_UMC. For physical addresses, the most significant bit is given by Core::X86::Cpuid::LongModeInfo[PhysAddrSize].

Table 61: MCA\_ADDR\_UMC Register

Error Type	Bits	Description
------------	------	-------------

DramEccErr	[55:39] [39:4]	Reserved Reserved
WriteDataPoisonErr	[55:0]	Reserved
SdpParityErr	[55:0]	Reserved
ApbErr	[55:30] [29:0]	Reserved Reserved
AddressCommandParityErr	[55:38] [37:36] [35:32] [31:0]	Reserved Reserved Chip Select Reserved
WriteDataCrcErr	[55:38] [37:36] [35:32] [31:0]	Reserved Reserved Chip Select Reserved

#### MSR0000\_043F...MSRC000\_2103 [UMC Machine Check Miscellaneous 0] (MCA::UMC::MCA\_MISC0\_UMC)

Log miscellaneous information associated with errors.

\_instUMC\_umc0\_aliasMSRLEGACY; MSR0000\_043F

\_instUMC\_umc1\_aliasMSRLEGACY; MSR0000\_0443

\_instUMC\_umc0\_aliasMSR; MSRC000\_20F3

\_instUMC\_umc1\_aliasMSR; MSRC000\_2103

Bits	Description
63	<b>Valid.</b> Reset: 1. 1=A valid CntP field is present in this register. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read-only.
62	<b>CntP.</b> Reset: 1. 1=A valid threshold counter is present. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read-only.
61	<b>Locked.</b> Reset: 0. 1=Writes to this register are ignored. This bit is set by BIOS to indicate that this register is not available for OS use. BIOS should set this bit if ThresholdIntType is set to SMI. AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read-only.
60	<b>IntP.</b> Reset: 1. 1=ThresholdIntType can be used to generate interrupts. 0=ThresholdIntType and interrupt generation are not supported. AccessType: (Core::X86::Msr::HWCR[McStatusWrEn]   !MCA::UMC::MCA_MISC0_UMC[Locked]) ? Read-write : Read-only.
59:56	Reserved.
55:52	<b>LvtOffset.</b> Reset: 0h. One per die. 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). AccessType: (Core::X86::Msr::HWCR[McStatusWrEn]   !MCA::UMC::MCA_MISC0_UMC[Locked]) ? Read-write : Read-only.
51	<b>CntEn.</b> Reset: 0. 1=Count thresholding errors. AccessType: (Core::X86::Msr::HWCR[McStatusWrEn]   !MCA::UMC::MCA_MISC0_UMC[Locked]) ? Read-write : Read-only.
50:49	<b>ThresholdIntType.</b> Reset: Cold,0h. Specifies the type of interrupt signaled when Ovrflw is set and IntP==1. 00b = No interrupt. 01b = APIC based interrupt (see Core::X86::Msr::McaIntrCfg[ThresholdLvtOffset]). 10b = SMI trigger event. 11b = Reserved. AccessType: (Core::X86::Msr::HWCR[McStatusWrEn]   !MCA::UMC::MCA_MISC0_UMC[Locked]) ? Read-write : Read-only.
48	<b>Ovrflw.</b> Reset: Cold,0. Set by hardware when ErrCnt transitions from FFEh to FFFh. When this field is set, ErrCnt no longer increments. When this bit is set, the interrupt selected by the ThresholdIntType field is generated.



	AccessType: (Core::X86::Msrr::HWCR[McStatusWrEn]   !MCA::UMC::MCA_MISC0_UMC[Locked]) ? Read-write : Read-only.
47:44	Reserved.
43:32	<b>ErrCnt.</b> Reset: Cold,000h. This is written by software to set the starting value of the error counter. This is incremented by hardware when errors are logged. When this counter overflows, it stays at FFFh (no rollover). The threshold value, written by software, is (FFFh - the desired error count (the number of errors necessary in order for an interrupt to be taken)); the desired error count of 0 (a write value of FFFh) is not supported.
	AccessType: (Core::X86::Msrr::HWCR[McStatusWrEn]   !MCA::UMC::MCA_MISC0_UMC[Locked]) ? Read-write : Read-only.
31:24	<b>BlkPtr.</b> Read-write. Reset: 01h. 00h=Extended MISC MSR block is not valid. 01h=Extended MSR block is valid.
23:0	Reserved.

#### MSRC000\_2[0F...10]4 [UMC Machine Check Configuration] (MCA::UMC::MCA\_CONFIG\_UMC)

Reset: 0000\_0002\_0000\_0025h.

Controls configuration of the associated machine check bank.

\_instUMC\_umc0\_aliasMSR; MSRC000\_20F4

\_instUMC\_umc1\_aliasMSR; MSRC000\_2104

Bits	Description
63:39	Reserved.
38:37	<b>DeferredIntType.</b> Read-write. Reset: 0h. Specifies the type of interrupt signaled when a deferred error is logged. 00b = No interrupt. 01b = APIC based interrupt (see Core::X86::Msrr::McaIntrCfg[DeferredLvtOffset]). 10b = SMI trigger event. 11b = Reserved.
36:35	Reserved.
34	<b>LogDeferredInMcaStat.</b> Read-write. Reset: 0. Init: BIOS,1. 1=Log deferred errors in MCA::UMC::MCA_STATUS_UMC and MCA::UMC::MCA_ADDR_UMC in addition to MCA::UMC::MCA_DESTAT_UMC and MCA::UMC::MCA_DEADDR_UMC. 0=Only log deferred errors in MCA::UMC::MCA_DESTAT_UMC and MCA::UMC::MCA_DEADDR_UMC. This bit does not affect logging of deferred errors in MCA::UMC::MCA_SYND_UMC, MCA::UMC::MCA_MISC0_UMC.
33	Reserved.
32	<b>McaXEnable.</b> Read-write. Reset: 0. Init: BIOS,1. Check: 1. 1=Software has acknowledged support for the MCAX feature set. 0=Software has not acknowledged support for the MCAX feature set. All uncorrected and fatal errors will cause an ErrorEvent packet to be generated. Deferred error interrupts are configured via Core::X86::Msrr::McaIntrCfg.
31:6	Reserved.
5	<b>DeferredIntTypeSupported.</b> Read-only. Reset: 1. 1=MCA::UMC::MCA_CONFIG_UMC[DeferredIntType] controls the type of interrupt generated on a deferred error. Deferred errors are supported in this bank only if MCA::UMC::MCA_CONFIG_UMC[DeferredErrorLoggingSupported]=1.
4:3	Reserved.
2	<b>DeferredErrorLoggingSupported.</b> Read-only. Reset: 1. 1=Deferred errors are supported in this MCA bank, and MCA::UMC::MCA_CONFIG_UMC[LogDeferredInMcaStat] controls the logging behavior of these errors. MCA::UMC::MCA_DESTAT_UMC and MCA::UMC::MCA_DEADDR_UMC are supported in this MCA bank. 0=Deferred errors are not supported in this bank.
1	Reserved.
0	<b>McaX.</b> Read-only. Reset: 1. 1=This bank provides Machine Check Architecture Extensions. Up to 4 additional MISC registers (MISC1-MISC4) are supported. MCA::UMC::MCA_MISC0_UMC[BlkPtr] indicates the presence of the additional MISC registers, but is not used to determine their MSR numbers. Deferred error interrupt type is specifiable by MCA bank. MCA::UMC::MCA_STATUS_UMC[TCC] is present.

#### MSRC000\_2[0F...10]5 [UMC IP Identification] (MCA::UMC::MCA\_IPID\_UMC)

Reset: 0000\_0096\_0000\_0000h.

The MCA::UMC::MCA\_IPID\_UMC register is used by software to determine what IP type and revision is associated with the MCA bank.



_instUMC_umc0_aliasMSR; MSRC000_20F5	
_instUMC_umc1_aliasMSR; MSRC000_2105	
Bits	Description
63:48	<b>McaType.</b> Read-only. Reset: 0000h. The McaType of the MCA bank within this IP.
47:44	Reserved.
43:32	<b>HardwareID.</b> Read-only. Reset: 096h. The Hardware ID of the IP associated with this MCA bank.
31:0	<b>InstanceId.</b> Read-write. Reset: 0000_0000h. The instance ID of this IP. This is initialized to a unique ID per instance of this register.

**MSRC000\_2[0F...10]6 [UMC Machine Check Syndrome] (MCA::UMC::MCA\_SYND\_UMC)**

Read-write, Volatile. Reset: Cold, 0000_0000_0000_0000h.	
Logs physical location information associated with error in MCA::UMC::MCA_STATUS_UMC Thread 0	
_instUMC_umc0_aliasMSR; MSRC000_20F6	
_instUMC_umc1_aliasMSR; MSRC000_2106	
Bits	Description
63:48	Reserved.
47:32	<b>Syndrom.</b> Read-write, Volatile. Reset: Cold, 0000h. Contains the syndrome, if any, associated with the error logged in MCA::UMC::MCA_STATUS_UMC. The low-order bit of the syndrome is stored in bit 0, and the syndrome has a length specified by MCA::UMC::MCA_SYND_UMC[Length]. The Syndrome field is only valid when MCA::UMC::MCA_SYND_UMC[Length] is not 0.
31:27	Reserved.
26:24	<b>ErrorPriority.</b> Read-write, Volatile. Reset: Cold, 0h. Encodes the priority of the error logged in MCA::UMC::MCA_SYND_UMC. 3'b000 = No error; 3'b001 = Reserved; 3'b010 = Corrected Error; 3'b011 = Deferred Error; 3'b100 = Uncorrected Error; 3'b101 = Fatal Error; all others reserved.
23:18	<b>Length.</b> Read-write, Volatile. Reset: Cold, 00h. Specifies the length in bits of the syndrome contained in MCA::UMC::MCA_SYND_UMC[Syndrome]. A value of 0 indicates that there is no valid syndrome in MCA::UMC::MCA_SYND_UMC. For example, a syndrome length of 9 means that MCA::UMC::MCA_SYND_UMC[Syndrome] bits [8:0] contains a valid syndrome.
17:0	<b>ErrorInformation.</b> Read-write, Volatile. Reset: Cold, 0_0000h. Contains error-specific information about the location of the error. Decoding is available in Table 62 [MCA_SYND_UMC Register].

Table 62: MCA\_SYND\_UMC Register

Error Type	Bits	Description
DramEccErr	[17:16]	Reserved
	[15]	Software-Managed Bad Symbol ID Error
	[14]	Reserved
	[13:8]	Symbol. Only contains valid information if ErrorPriority indicates a corrected error.
	[7]	Reserved
	[6:4]	Cid. Specifies the rank multiply ID for supported DIMMs.
	[3]	Reserved
	[2:0]	Chip Select
WriteDataPoisonErr	[17:0]	Reserved
SdpParityErr	[17:0]	Reserved
ApbErr	[17:0]	Reserved
AddressCommandParityErr	[17:0]	Reserved
WriteDataCrcErr	[17:0]	Reserved

**MSRC000\_2[0F...10]8 [UMC Machine Check Deferred Error Status] (MCA::UMC::MCA\_DESTAT\_UMC)**

Read-write, Volatile. Reset: Cold, 0000_0000_0000_0000h.	
--	--

Holds status information for the first deferred error seen in this bank.	
_instUMC_umc0_aliasMSR; MSRC000_20F8	
_instUMC_umc1_aliasMSR; MSRC000_2108	
Bits	Description
63	<b>Val.</b> Read-write, Volatile. Reset: Cold, 0. 1=A valid error has been detected (whether it is enabled or not).
62	<b>Overflow.</b> Read-write, Volatile. Reset: Cold, 0. 1=An error was detected while the valid bit (Val) was set; at least one error was not logged. Overflow is set independently of whether the existing error is overwritten. (See the section on overwrite priorities.)
61:59	Reserved.
58	<b>AddrV.</b> Read-write, Volatile. Reset: Cold, 0. 1=MCA::UMC::MCA_DEADDR_UMC contains address information associated with the error.
57:54	Reserved.
53	<b>SyndV.</b> Read-write, Volatile. Reset: Cold, 0. 1=This error logged information in MCA::UMC::MCA_SYND_UMC. If MCA::UMC::MCA_SYND_UMC[ErrorPriority] is the same as the priority of the error in MCA::UMC::MCA_STATUS_UMC, then the information in MCA::UMC::MCA_SYND_UMC is associated with the error in MCA::UMC::MCA_DESTAT_UMC.
52:45	Reserved.
44	<b>Deferred.</b> Read-write, Volatile. Reset: Cold, 0. 1=A deferred error was created. A deferred error is the result of an uncorrectable data error which did not immediately cause a processor exception; poison is created and an exception is deferred until the poison data is consumed.
43:0	Reserved.

#### MSRC000\_2[0F...10]9 [UMC Deferred Error Address] (MCA::UMC::MCA\_DEADDR\_UMC)

Reset: Cold, 0000\_0000\_0000\_0000h.

The MCA::UMC::MCA\_DEADDR\_UMC register stores the address associated with the error in MCA::UMC::MCA\_DESTAT\_UMC. The register is only meaningful if MCA::UMC::MCA\_DESTAT\_UMC[Val]=1 and MCA::UMC::MCA\_DESTAT\_UMC[AddrV]=1. The lowest valid bit of the address is defined by MCA::UMC::MCA\_DEADDR\_UMC[LSB].

\_instUMC\_umc0\_aliasMSR; MSRC000\_20F9

\_instUMC\_umc1\_aliasMSR; MSRC000\_2109

Bits	Description
63:62	Reserved.
61:56	<b>LSB.</b> Read-write, Volatile. Reset: Cold, 00h. Specifies the least significant valid bit of the address contained in MCA::UMC::MCA_DEADDR_UMC[ErrorAddr]. A value of 0 indicates that MCA::UMC::MCA_DEADDR_UMC[55:0] contains a valid byte address. A value of 6 indicates that MCA::UMC::MCA_DEADDR_UMC[55:6] contains a valid cache line address and that MCA::UMC::MCA_DEADDR_UMC[5:0] are not part of the address and should be ignored by error handling software. A value of 12 indicates that MCA::UMC::MCA_DEADDR_UMC[55:12] contain a valid 4KB memory page and that MCA::UMC::MCA_DEADDR_UMC[11:0] should be ignored by error handling software.
55:0	<b>ErrorAddr.</b> Read-write, Volatile. Reset: Cold, 00_0000_0000_0000h. Contains the address, if any, associated with the error logged in MCA::UMC::MCA_DESTAT_UMC. The lowest-order valid bit of the address is specified in MCA::UMC::MCA_DEADDR_UMC[LSB].

#### MSRC000\_2[0F...10]A [UMC Machine Check Miscellaneous 1] (MCA::UMC::MCA\_MISC1\_UMC)

Read-write.

Log miscellaneous information associated with errors, as defined by each error type.

\_instUMC\_umc0\_aliasMSR; MSRC000\_20FA

\_instUMC\_umc1\_aliasMSR; MSRC000\_210A

Bits	Description
63	<b>Valid.</b> Read-write. Reset: 1. 1=A valid CntP field is present in this register.
62	<b>CntP.</b> Read-write. Reset: 1. 1=A valid threshold counter is present.
61	<b>Locked.</b> Read-write. Reset: 0. 1=Writes to this register are ignored. This bit is set by BIOS to indicate that this

	register is not available for OS use. BIOS should set this bit if ThresholdIntType is set to SMI.
60	<b>IntP.</b> Read-write. Reset: 1. 1=ThresholdIntType can be used to generate interrupts. 0=ThresholdIntType and interrupt generation are not supported.
59:52	Reserved.
51	<b>CntEn.</b> Read-write. Reset: 0. 1=Count thresholding errors.
50:49	<b>ThresholdIntType.</b> Read-write. Reset: Cold,0h. Specifies the type of interrupt signaled when Ovrflw is set. 00b = No interrupt. 01b = APIC based interrupt (see Core::X86::Msr::McaIntrCfg[ThresholdLvtOffset]) to all cores. 10b = SMI trigger event. 11b = Reserved.
48	<b>Ovrflw.</b> Read-write. Reset: Cold,0. Set by hardware when ErrCnt transitions from FFEh to FFFh; also set by hardware if ErrCnt is initialized to FFFh and transitions from FFFh to 000h. When this field is set, ErrCnt no longer increments. When this bit is set, the interrupt selected by the ThresholdIntType field is generated.
47:44	Reserved.
43:32	<b>ErrCnt.</b> Read-write. Reset: Cold,000h. This is written by software to set the starting value of the error counter. This is incremented by hardware when errors are logged. When this counter overflows, it stays at FFFh (no rollover). The threshold value, written by software, is (FFFh - the desired error count (the number of errors necessary in order for an interrupt to be taken)); the desired error count of 0 (a write value of FFFh) is not supported.
31:24	<b>BlkPtr.</b> Read-write. Reset: 01h. 00h=Extended MISC MSR block is not valid. 01h=Extended MSR block is valid.
23:0	Reserved.

#### MSRC001\_04[0F...10] [UMC Machine Check Control Mask] (MCA::UMC::MCA\_CTL\_MASK\_UMC)

Read-write. Reset: 0000_0000_0000_0000h.	
Inhibit detection of an error source.	
<u>_instUMC_umc0_aliasMSR; MSRC001_040F</u>	
<u>_instUMC_umc1_aliasMSR; MSRC001_0410</u>	
Bits	Description
63:6	Reserved.
5	<b>WriteDataCrcErr.</b> Read-write. Reset: 0. Write data CRC error. A write data CRC error on the DRAM data bus.
4	<b>AddressCommandParityErr.</b> Read-write. Reset: 0. Address/command parity error. A parity error on the DRAM address/command bus.
3	<b>ApbErr.</b> Read-write. Reset: 0. Advanced peripheral bus error. An error on the advanced peripheral bus.
2	<b>SdpParityErr.</b> Read-write. Reset: 0. SDP parity error. A parity error on write data from the data fabric.
1	<b>WriteDataPoisonErr.</b> Read-write. Reset: 0. Data poison error.
0	<b>DramEccErr.</b> Read-write. Reset: 0. DRAM ECC error. An ECC error on a DRAM read.

## 4 System Management Unit (SMU)

### 4.1 SMU Registers

The system management unit (SMU) is a subcomponent of the processor that is responsible for a variety of system and power management tasks during boot and runtime.

### 4.2 Thermal (THM)

The thermal block contains all the features related to temperature sensing, control, and reporting. It includes:

- Temperature collection and calculation logic.
- Fan speed control for off-chip fans.
- Temperature reporting through the APML interface.

#### 4.2.1 Registers

SMUTHMx00000000 (SMU::THM::THM_TCON_CUR_TMP)	
Reset: 0000_0000h.	
_aliasSMN; SMUTHMx00000000; SMUTHM=0005_9800h	
_aliasHOSTGPU; GPUF0REGx59800; GPUF0REG=0000_0000h	
Bits	Description
31:21	<b>CUR_TEMP</b> . Reset: 000h. Provides current control temperature. AccessType: _aliasHOSTGPU: Read-write AccessType: _aliasSMN: (InitiatorTrust > 2) ? Read-only : Read-write
20	Reserved.
19	<b>CUR_TEMP_RANGE_SEL</b> . Reset: 0. 0=Report on 0C to 225C scale range. 1=Report on -49C to 206C scale range. AccessType: _aliasHOSTGPU: Read-write AccessType: _aliasSMN: (InitiatorTrust > 2) ? Read-only : Read-write
18:0	Reserved.

## 5 Advanced Platform Management Link (APML)

### 5.1 Overview

The Advanced Platform Management Link (APML) is a SMBus v2.0 compatible 2-wire processor slave interface. APML is also referred as the sideband interface (SBI).

APML is used to communicate with the Remote Management Interface (see SBI Remote Management Interface (SB-RMI) and SBI Temperature Sensor Interface (SB-TSI). For related specifications, see 1.2 [Reference Documents].

#### 5.1.1 Definitions

Table 63: APML Definitions

Term	Description
<b>ARA</b>	Alert response address.
<b>ARP</b>	Address Resolution Protocol
<b>EC</b>	Embedded Controller.
<b>KBC</b>	Keyboard Controller.
<b>Master or SMBus Master</b>	The device that initiates and terminates all communication and drives the clock, SCL.
<b>PEC</b>	Packet error code.
<b>POR</b>	Power on reset.
<b>RTS</b>	Remote temperature sensor, typical examples are ADM1032, LM99, MAX6657, EMC1002.
<b>SB-RMI</b>	Remote Management interface.
<b>Slave or SMBus slave</b>	The slave cannot initiate SMBus communication and cannot drive the clock but can drive the data signal SDA and the alert signal ALERT_L.
<b>TSI</b>	Temperature sensor interface.

### 5.2 SBI Bus Characteristics

The SBI largely follows SMBus v2.0. This section describes the exceptions.

#### 5.2.1 SMBus Protocol Support

The SBI follows SMBus protocol except:

- The processor does not implement SMBus master functionality.
- The SBI implements the Send Byte/Receive Byte, Read Byte/Write Byte, Block Read/Block Write and Block Write-Block Read Process Call SMBus protocols. The Send Byte/Receive Byte SMBus protocol is only supported by SB-TSI.
- Packet error checking (PEC) is not supported by SB-TSI.
- Address Resolution Protocol (ARP) is not implemented.
- Cumulative clock extensions are not enforced.

#### 5.2.2 I2C Support

The processor supports higher I2C-defined speeds as specified in the Physical Layer Characteristics section. The

processor supports the I2C master code transmission in order to reach the high-speed bus mode. Multiple SBI commands may be sent within a single high-speed mode session. Ten-bit addressing is not supported.

## 5.3 SBI Processor Information

### 5.3.1 SBI Processor Pins

Up to six processor pins are used for SBI support: two for data transfer, three for address determination and one for an interrupt output. Of the three address pins, one bit is `socket_id` used to determine which package is addressed. These pins do not have changeable pinstrap. The Serial Interface Clock (SIC) and Serial Interface Data (SID) pins function as the SMBus clock and data pins respectively. The SMBus alert pin (`ALERT_L`) is used to signal interrupts to the SMBus master.

#### 5.3.1.1 Physical Layer Characteristics

The SIC and SID pins differ from the SMBus specification with regard to voltage. System board voltage translators are necessary to convert the SIC and SID pin voltage levels to that of the SMBus specification. SBI supports frequencies of 100 KHz, 400 KHz over SIC.

### 5.3.2 Processor States

SBI responds to SMBus traffic except when `PWROK` is de-asserted (and for a brief period after it is de-asserted). Access to internal processor state using SB-RMI is not supported under the following conditions:

- During cold and warm resets.
- During the APIC spin loop.

## 5.4 SBI Protocols

### 5.4.1 SBI Modified Block Write-Block Read Process Call

SBI uses a modified SMBus PEC-optional Block Write-Block Read Process Call protocol. The change from the SMBus protocol is support for an optional intermediate PEC byte and ACK after the ACK for Data Byte M. The PEC byte after Data Byte N covers all previous bytes excluding the first PEC byte. Figure below shows the transmission protocol. Each byte in the protocol is sent with the most significant bit first (bit[7]). The master may reset the bus by holding the clock low for 25ms as specified by the SMBus Specification.



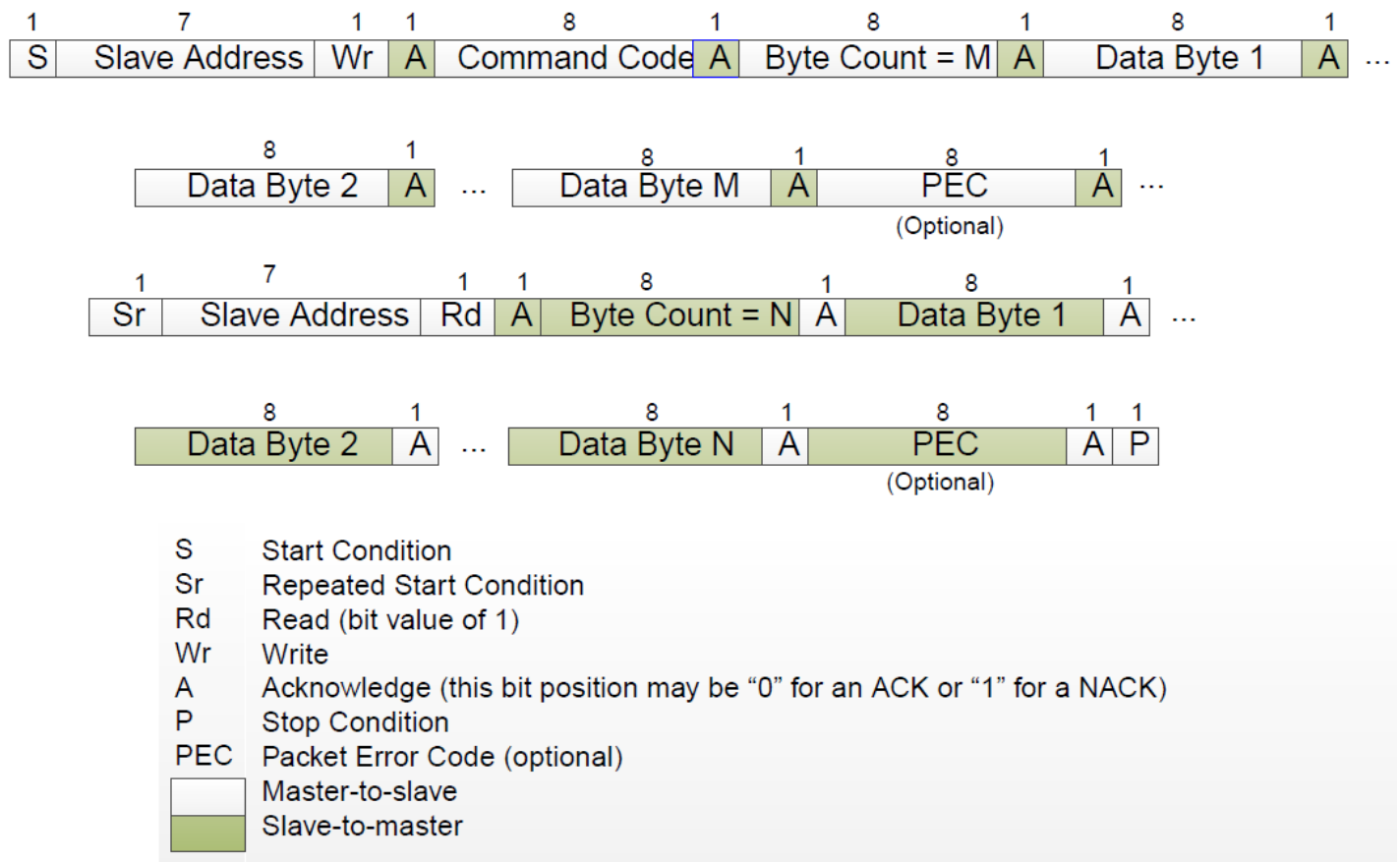


Figure 25: SBI Transmission Protocol

## 5.4.2 SBI Error Detection and Recovery

This section describes the various error detection and recovery methods that can be used on the SBI bus. The important item in providing a high reliability SBI connection is the ability to detect when an error occurs and to gracefully recover from that error. When the SBI connections are noisy, messages can become garbled which, in turn, may cause undefined behavior on the SBI bus. The most common noise sources are cross-talk and clock skew. Cross-talk results when the SBI connections are routed too close to other signal carrying lines. Clock skew is usually a result of higher than expected capacitance, between the SBI signals (clock and / or data) and ground, which causes the master and slave devices to disagree on when data should be stable and when it is allowed to be changing.

### 5.4.2.1 Error Detection

SBI provides several methods of error detection: protocol ACK/NAK, packet error correction (PEC) fields, and timeouts. The ACK/NAK mechanism is always active in SBI, but the PEC and timeouts are optional.

#### 5.4.2.1.1 ACK/NAK Mechanism

After each byte of an SBI message, the device receiving that byte must either acknowledge (ACK) that it received the byte correctly, or deny (NAK) that the byte was correctly received. This is most easily seen in the case of the address bytes which follow a START (or REPEATED START) sequence, but can be used anywhere in the message. In the case of an address byte, if a slave device recognizes the address, it will respond with an ACK and await the rest of the message. If a slave device does not recognize the message, it will respond with a NAK and ignore the rest of the message.

#### 5.4.2.1.2 Packet Error Correction (PEC)

The RMI protocols allow for PEC bytes to be appended to messages. The sending side calculates the PEC, based on the data it intends to transmit, and appends it to the transmitted data. The receiving side calculates the PEC based on the data it actually receives and compares that to the PEC it receives. If the two PECs do not match, an error has occurred and the message should be discarded. When a device detects a PEC mismatch, it should send a NAK in response to the PEC. No special programming is needed to enable the PEC on AMD devices. If the PEC is present on an incoming message, the device will verify the PEC and ACK or NAK as appropriate. The PEC is always calculated on outgoing messages. It is up to the bus master to request the PEC by sending clocks for that byte before sending either a NAK or a STOP sequence.

#### 5.4.2.1.3 Bus Timeouts

Bus timeouts should be enabled to prevent a device waiting indefinitely on a message that may not be coming. Some timeouts are used to prevent the SBI bus from waiting for a response from a CPU that is in a power-saving idle mode. Other timeouts are used to allow the slave device to recognize that the bus master is attempting to reset all of the devices on the SBI bus. Either way, when a device recognizes a timeout, it should abort its current message transfer.

#### 5.4.2.2 Error Recovery

The simplest form of error recovery is a retry. When the bus master detects an unexpected NAK, it should abort the current transfer and retry the message sequence. In some cases, however, a message can be so garbled that a simple retry is insufficient. This can occur, if there are multiple devices on the bus and a garbled address byte has caused the wrong slave device to be selected. That slave device may even continue to transmit during the retry. In those cases, it will be necessary to force a reset of all devices on the SBI bus, before retrying the message transfer.

##### 5.4.2.2.1 SBI Bus Reset

The bus master can hold the clock low for a period longer the standard timeout in order to force slave devices off the bus (see docSMB section 3.1.1.3 of the System Management Bus (SMBus) Specification, version 2.0). All SBI slave devices are required to reset their communications if another device holds the clock line low for longer than TTimeout, min (25 milliseconds). The devices are required to complete their reset within TTimeout, max (35 milliseconds). SBI bus masters should use the extended timeout to force a reset of all slave devices if a simple retry does not remove an error condition.

### 5.5 SBI Physical Interface

#### 5.5.1 SBI SMBus Address

The SMBus address is really 7 bits. Some vendors and the SMBus specification show the address as 8 bits: bits[7:1] as the left-justified address, and bit[0] as the Read/Write flag, where 0 indicates a Write and 1 indicates a Read. Some vendors use only the 7 bits to describe the address.

#### 5.5.2 SBI Bus Timing

SBI supports 100KHz standard-mode and 400 KHz fast-mode I2C operation. Refer to the standard-mode and fast-mode timing parameters in the I2C specification.

#### 5.5.3 Pass-FET Option

There is a possibility that a device with a standard SMBus interface will not be able to directly interface to SBI. Therefore,

pass FETs must be used to create two SMBus segments, see the following figure.

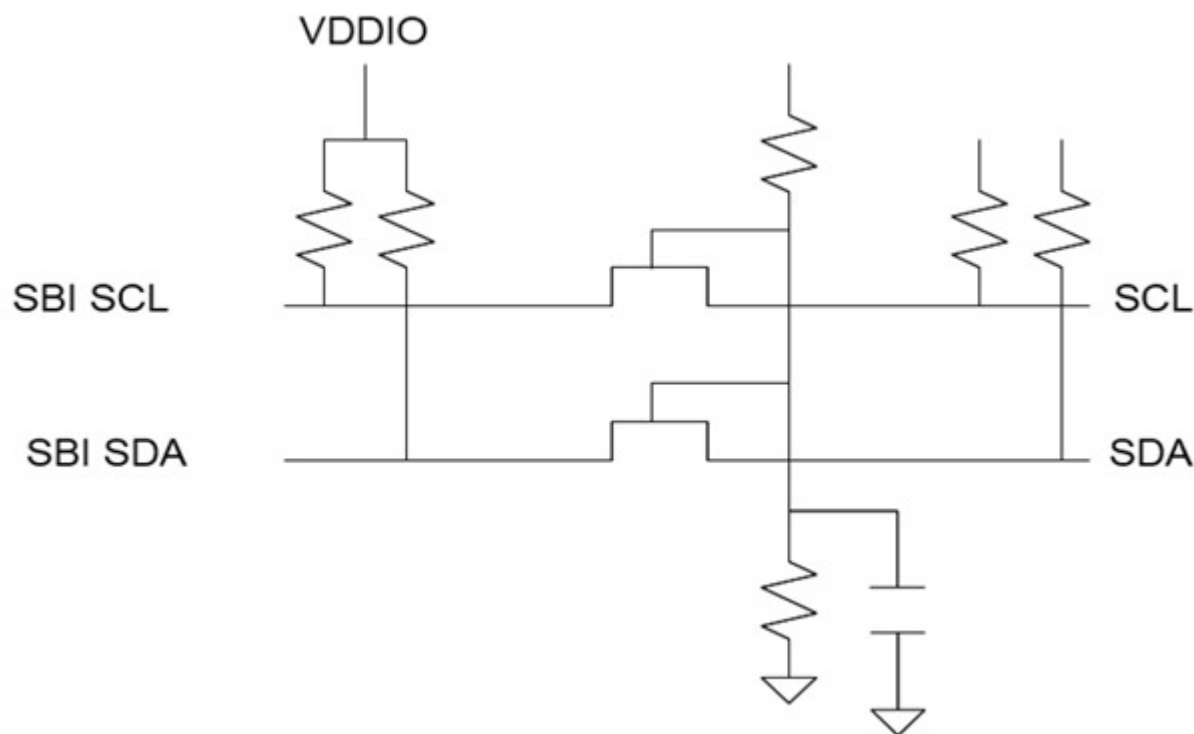


Figure 26: Pass FET Implementation

Notes:

- SCL and SDA pull-up resistors are the normal pull-up resistors for a SMBus segment, and are not part of the translation circuit. They are shown for completeness.
- The gates of the FETs are tied to a voltage approximately  $V_{gs}$  above the lower rail voltage. A resistive divider is shown, but a convenient power rail will also work.
- Care must be taken to install the FETs so that any body diode does not conduct.
- The key requirement is the high side drive low enough to register as low on the low side (High side  $V_{ol} < V_{il}$  on low side)

## 6 SB Temperature Sensor Interface (SB-TSI)

### 6.1 Overview

The SBI temperature sensor interface (SB-TSI) is an emulation of the software and physical interface of a typical 8-pin remote temperature sensor (RTS), see Figure 27 [RTS Thermal Management Example]. The goal is to resemble a typical RTS so that KBC or BMC firmware requires minimal changes for future AMD products, see Figure 28 [SB-TSI Thermal Management Example]. SB-TSI supports the SMBus protocols that typical RTS supports.

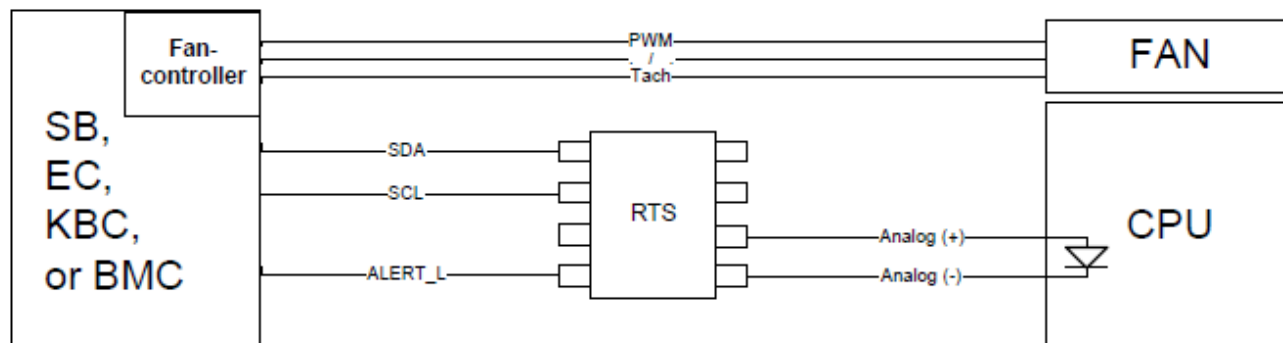


Figure 27: RTS Thermal Management Example

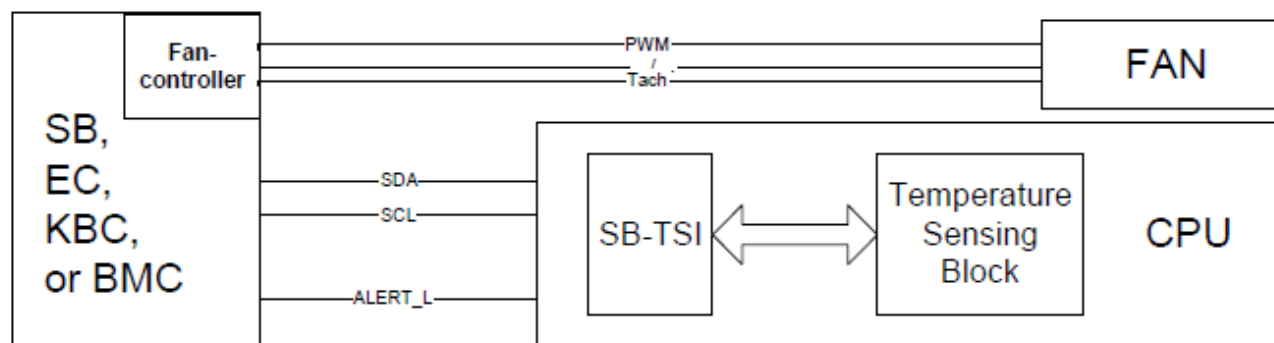


Figure 28: SB-TSI Thermal Management Example

Refer to the following external sources for additional information.

- System Management Bus (SMBus) specification. See docSMB.
- I2C-bus Specification and User Manual, Revision 03. See docI2C.

#### 6.1.1 Definitions

Table 64: SB-TSI Definitions

Term	Description
<b>BMC</b>	Base management controller.
<b>TCC</b>	Temperature calculation circuit.
<b>Tctl</b>	Processor temperature control value.

<b>TSM</b>	Temperature sensor macro.
<b>SB-TSI</b>	Sideband Internal Temperature Sensor Interface. See APMML.

## 6.2 SB-TSI Protocol

The SB-TSI largely follows SMBus v2.0 specification except:

- The combined-format repeated start sequence is not supported in standard-mode and fast-mode. The response of the processor's SB-TSI to the sequence is undefined.
- Only 7-bit SMBus addresses are supported.
- SB-TSI implements the Send/Receive Byte and Read/Write Byte protocols.
- SB-TSI registers can only be written by using a write byte command.
- Address Resolution Protocol (ARP) is not supported.
- Packet Error Checking (PEC) is not supported.
- The usage of unsupported protocols may lead to an undefined bus condition.
- To release the bus from an undefined condition and to reset the SB-TSI slave, the bus master must hold the clock low for a duration of time that is longer than Timeout.max, as specified for SMBus. The time-out needs to be enabled by SBTISI::TimeoutConfig[TimeoutEn] = 1.

### 6.2.1 SB-TSI Send/Receive Byte Protocol

A SMBus master can Read SB-TSI registers by issuing a send byte command with the address of the register to be read as the data byte followed by a receive byte command.

#### 6.2.1.1 SB-TSI Address Pointer

The SB-TSI controller has an internal address pointer that is updated when a register is accessed using a Read or Write byte command or when a send byte command is received. This address pointer is used to determine the address of the register being read when a receive byte command is processed by the controller.

### 6.2.2 SB-TSI Read/Write Byte Protocol

An SMBus master can Read or Write SB-TSI registers by issuing a Read or a Write byte command with the address of the register to be read or written in the command code field.

### 6.2.3 Alert Behavior

The ALERT\_L pin is asserted if (SBTISI::Status[TempHighAlert] || SBTISI::Status[TempLowAlert]) && ~SBTISI::Config[AlertMask] as shown in Figure 3. The following registers also affect temperature alert behavior.

- SBTISI::Config[AraDis]: Disables ARA response.
- SBTISI::UpdateRate[UpRate]: Specifies rate at which temperature thresholds are checked.
- {SBTISI::HiTempInt[HiTempInt], SBTISI::HiTempDec[HiTempDec]}: Sets high temperature threshold.
- {SBTISI::LoTempInt[LoTempInt], SBTISI::LoTempDec[LoTempDec]}: Sets low temperature threshold.
- SBTISI::AlertThreshold[AlertThr]: Specifies number of consecutive temperature samples to assert an alert.
- SBTISI::AlertConfig[AlertCompEn]: Specifies ALERT\_L pin to be in latched or comparator mode. Affects ARA.

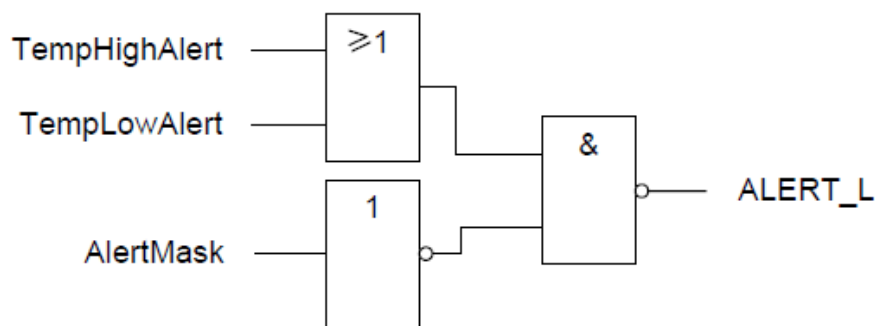


Figure 29: Alert Assertion Diagram

#### 6.2.4 Atomic Read Mechanism

To ensure that the two required Reads (integer and decimal) for reading the CPU temperature are always originated from one temperature value, atomic reading procedures are required. SB-TSI offers functions to maintain atomicity between the temperature integer and decimal bytes.

[The SB-TSI Configuration Register] SBTSI::Config[ReadOrder] specifies the order for reading integer and decimal part of the CPU temperature value for atomic CPU temperature Reads. If SBTSI::Config[ReadOrder] is 0, then a Read of the integer part (SBTSI::CpuTempInt) of the CPU temperature triggers a latch of the decimal part (SBTSI::CpuTempDec) until the next Read of the integer part. This latch syncs the decimal part with the integer part. The integer part is continuously updated.

If SBTSI::Config[ReadOrder] is 1, then the Read order to ensure atomicity is Reversed, i.e., decimal part = first, integer part = second.

If it is not possible to ensure a dedicated Read order as described above, the Run/Stop bit ([The SB-TSI Configuration Register] SBTSI::Config[RunStop]) may be used to provide atomicity of reading the CPU temperature. If this bit is 0, the CPU temperature registers are updated continuously. If it is 1, they get frozen and always deliver their last value on Read requests.

- Set SBTSI::Config[RunStop].
- Read the integer (SBTSI::CpuTempInt) or the decimal (SBTSI::CpuTempDec) part of the CPU temperature.
- Read the remaining part of the CPU temperature.
- Clear SBTSI::Config[RunStop].

#### 6.2.5 SB-TSI Temperature and Threshold Encodings

SB-TSI CPU temperature readings and limit registers encode the temperature in increments of 0.125 from 0 to 255.875. The high byte represents the integer portion of the temperature from 0 to 255. One increment in the high byte is equivalent to a step of one. The upper three bits of the low byte represent the decimal portion of the temperature. One increment of these bits is equivalent to a step of 0.125.

Table 65: SB-TSI CPU Temperature and Threshold Encoding Examples

Temperature	Temperature High Byte SBTSI::CpuTempInt[CpuTempInt] SBTSI::HiTempInt[HiTempInt]	Temperature Low Byte SBTSI::CpuTempDec[CpuTempDec] SBTSI::HiTempDec[HiTempDec]
-------------	---	--

	SBTSI::LoTempInt[LoTempInt]	SBTSI::LoTempDec[LoTempDec]
0.000 °C	0000_0000b	0000_0000b
1.000 °C	0000_0001b	0000_0000b
25.125 °C	0001_1001b	0010_0000b
50.875 °C	0011_0010b	1110_0000b
90.000 °C	0101_1010b	0000_0000b

## 6.2.6 SB-TSI Temperature Offset Encoding

By default, SBTISI::CpuTempInt and SBTISI::CpuTempDec provide Tctl from the processor. The temperature offset registers allow the system to adjust the SB-TSI temperature from Tctl.

The SB-TSI temperature offset registers use a different encoding in order to provide negative temperature values. SBTISI::CpuTempOffInt[CpuTempOffInt] and SBTISI::CpuTempOffDec[CpuTempOffDec] form an 11-bit, 2's complement value representing the temperature offset. The high byte encodes the integer portion of the temperature and the upper three bits of the low byte represent the fractional portion of the temperature offset. One increment of these bits is equivalent to a step of 0.125 °C. After reset the offset is always set to 0 °C. Software needs to adjust the offset to the appropriate level.

*Table 66: SB-TSI Temperature Offset Encoding Examples*

Temperature	Temperature High Byte SBTSI::CpuTempOffInt[CpuTempOffInt]	Temperature Low Byte SBTSI::CpuTempOffDec[CpuTempOffDec]
-10.375 °C	1111_0101b	1010_0000b
-0.250 °C	1111_1111b	1100_0000b
0.000 °C	0000_0000b	0000_0000b
0.875 °C	0000_0000b	1110_0000b
10.000 °C	0000_1010b	0000_0000b

## 6.3 SB-TSI Physical Interface

This chapter describes the physical interface of the SB-TSI.

### 6.3.1 SB-TSI SMBus Address

The SMBus address is really 7 bits. Some vendors and the SMBus specification show the address as 8 bits: bits[7:1] as the left-justified address, and bit[0] as the Read/Write flag, where 0 indicates a Write and 1 indicates a Read. Some vendors use only the 7 bits to describe the address. The addresses can vary with address select pins.

*Table 67: SB-TSI Address Encodings*

Socket ID	SB-TSI Address
0b	98h for 8-bit or 4Ch for 7-bit.
1b	90h for 8-bit or 48h for 7-bit.

### 6.3.2 SB-TSI Bus Timing

SB-TSI supports standard-mode (100 kHz) and fast-mode (400 kHz) according to the I2C-bus Specification and User Manual.



### 6.3.3 SB-TSI Bus Electrical Parameters

SB-TSI conforms to most of the I2C fast-mode electrical parameters. See the Electrical Data Sheet for the processor family for electrical parameters.

### 6.3.4 Pass-FET Option

The KBC may not have the capability to directly interface to SB-TSI. Pass FETs may be used to create two SMBus segments, see Figure 4.

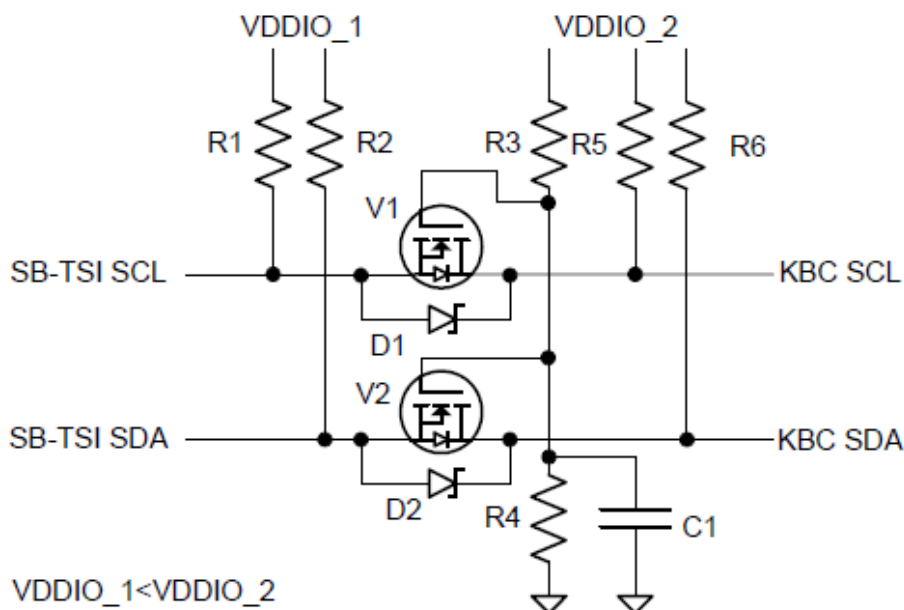


Figure 30: Pass FET Implementation

Notes:

- SCL and SDA pull-up resistors (R5 and R6, respectively) are the normal pull-up resistors for an SMBus segment and are not part of the translation circuit. They are shown for completeness.
- The gates of the FETs are tied to a voltage approximately  $V_{gs}$  above the lower rail voltage. A resistive divider is shown, but a convenient power rail would do nicely.
- Care must be taken to install the FETs so that any body diode does not conduct.
- The key requirement is that the high side drive low enough to register as a low on the low side. (High side  $V_{ol} < V_{il}$  on low side)

## 6.4 SB-TSI Registers

Reads to unimplemented registers return 00h. Writes to unimplemented registers are discarded.

### SBTSIx01 [CPU Integer Temperature] (SBTSI::CpuTempInt)

Read-only.

The CPU temperature is calculated by adding the CPU temperature offset (SBTSI::CpuTempOffInt, SBTSI::CpuTempOffDec) to the processor control temperature (Tctl). SBTSI::CpuTempInt and SBTSI::CpuTempDec combine to return the CPU temperature. For the temperature encoding, see 6.2.5 [SB-TSI Temperature and Threshold Encodings]

Bits	Description
7:0	<b>CpuTempInt: integer CPU temperature value.</b> Read-only. Reset: Cold,XXh. This field returns the integer

portion of the CPU temperature.

#### SBTSIx02 [SB-TSI Status] (SBTSI::Status)

Read-only, Volatile.

If SBTSI::AlertConfig[AlertCompEn] == 0, the temperature alert is latched high until the alert is Read. If SBTSI::AlertConfig[AlertCompEn] == 1, the alert is cleared when the temperature does not meet the threshold conditions for temperature and number of samples. See 6.2.3 [Alert Behavior].

Bits	Description
7:5	Reserved.
4	<b>TempHighAlert: temperature high alert.</b> Read-only, Volatile. Reset: Cold, X. 1=Indicates that the CPU temperature is greater than or equal to the high temperature threshold (SBTSI::HiTempInt, SBTSI::HiTempDec) for SBTSI::AlertThreshold[AlertThr] consecutive samples. 0=Indicates that the CPU temperature is less than the high temperature threshold (SBTSI::HiTempInt, SBTSI::HiTempDec) for SBTSI::AlertThreshold[AlertThr] samples and SBTSI::AlertConfig[AlertCompEn] == 1. Hardware will clear this bit when Read if SBTSI::AlertConfig[AlertCompEn] == 0.
3	<b>TempLowAlert: temperature low alert.</b> Read-only, Volatile. Reset: Cold, X. 1=Indicates that the CPU temperature is less than or equal to the low temperature threshold (SBTSI::LoTempInt, SBTSI::LoTempDec) for SBTSI::AlertThreshold[AlertThr] consecutive samples. 0=Indicates the CPU temperature is greater than the low temperature threshold (SBTSI::LoTempInt, SBTSI::LoTempDec) for SBTSI::AlertThreshold[AlertThr] samples and SBTSI::AlertConfig[AlertCompEn] == 1. Hardware will clear this bit when Read if SBTSI::AlertConfig[AlertCompEn] == 0.
2:0	Reserved.

#### SBTSIx03 [SB-TSI Configuration] (SBTSI::Config)

Reset: Cold, 00h.

The bits in this register are Read-only and can be written by Writing to the corresponding bits in SBTSI::ConfigWr. See 6.2.3 [Alert Behavior] and 6.2.4 [Atomic Read Mechanism].

Bits	Description
7	<b>AlertMask: alert mask.</b> Read-only, Volatile. Reset: Cold, 0. 0=ALERT_L pin enabled. 1=ALERT_L pin disabled and does not assert. IF (SBTSI::Config[AraDis] == 0) THEN Read-only; set-by-hardware. ELSE Read-only ENDIF. Hardware sets this bit if SBTSI::Config[AraDis] == 0, either SBTSI::Status[TempHighAlert] == 1 or SBTSI::Status[TempLowAlert] == 1, and a successful ARA is sent.
6	<b>RunStop: run stop.</b> Read-only. Reset: Cold, 0. 0=Updates to SBTSI::CpuTempInt and SBTSI::CpuTempDec and the alert comparisons are enabled; Alert history counters (specified by SBTSI::AlertThreshold[AlertThr]) and the corresponding timer (specified by SBTSI::UpdateRate[UpRate]) continue to update. 1=Updates to SBTSI::CpuTempInt and SBTSI::CpuTempDec and the alert comparisons are disabled; Alert history counters (specified by SBTSI::AlertThreshold[AlertThr]) and the corresponding timer (specified by SBTSI::UpdateRate[UpRate]) are stopped. See 6.2.4 [Atomic Read Mechanism] for further details.
5	<b>ReadOrder: atomic read order.</b> Read-only. Reset: Cold, 0. 0=Reading SBTSI::CpuTempInt causes the state of SBTSI::CpuTempDec to be latched. 1=Reading SBTSI::CpuTempDec causes the state of SBTSI::CpuTempInt to be latched. See 6.2.4 [Atomic Read Mechanism] for further details.
4:2	Reserved.
1	<b>AraDis: ARA disable.</b> Read-only. Reset: Cold, 0. Read-only. 1=ARA response disabled.
0	Reserved.

#### SBTSIx04 [Update Rate] (SBTSI::UpdateRate)

Read-write. Reset: Cold, 08h.

Bits	Description
7:0	<b>UpRate: update rate.</b> Read-write. Reset: Cold, 08h. This field specifies the rate at which CPU temperature is compared against the temperature thresholds to determine if an alert event has occurred. Write access causes a reset of the alert history counters (specified by SBTSI::AlertThreshold[AlertThr]) and the corresponding timer (specified by SBTSI::UpdateRate[UpRate]).

Valid Values:		
Value	Description	
00h	0.0625 Hz	
01h	0.125 Hz	
02h	0.25 Hz	
03h	0.5 Hz	
04h	1 Hz	
05h	2 Hz	
06h	4 Hz	
07h	8 Hz	
08h	16 Hz	
09h	32 Hz	
0Ah	64 Hz	
FFh-0Bh	Reserved.	

#### SBTSIx07 [High Temperature Integer Threshold] (SBTSI::HiTempInt)

Read-write. Reset: Cold,46h.

The high temperature threshold specifies the CPU temperature that causes ALERT\_L to assert if the CPU temperature is greater than or equal to the threshold. SBTSI::HiTempInt and SBTSI::HiTempDec combine to specify the high temperature threshold. See 6.2.5 [SB-TSI Temperature and Threshold Encodings]. Reset value equals 70 °C. Write access causes a reset of the alert history counters (specified by SBTSI::AlertThreshold[AlertThr]) and the corresponding timer (specified by SBTSI::UpdateRate[UpRate]). See 6.2.3 [Alert Behavior].

Bits	Description
7:0	<b>HiTempInt: high temperature integer threshold.</b> Read-write. Reset: Cold,46h. This field specifies the integer portion of the high temperature threshold.

#### SBTSIx08 [Low Temperature Integer Threshold] (SBTSI::LoTempInt)

Read-write. Reset: Cold,00h.

The low temperature threshold specifies the CPU temperature that causes ALERT\_L to assert if the CPU temperature is less than or equal to the threshold. SBTSI::LoTempInt and SBTSI::LoTempDec combine to specify the low temperature threshold. See 6.2.5 [SB-TSI Temperature and Threshold Encodings]. Write access causes a reset of the alert history counters (specified by SBTSI::AlertThreshold[AlertThr]) and the corresponding timer (specified by SBTSI::UpdateRate[UpRate]). See 6.2.3 [Alert Behavior].

Bits	Description
7:0	<b>LoTempInt: low temperature integer threshold.</b> Read-write. Reset: Cold,00h. This field specifies the integer portion of the low temperature threshold.

#### SBTSIx09 [SB-TSI Configuration Write] (SBTSI::ConfigWr)

Read-write. Reset: Cold,00h.

This register provides write access to SBTSI::Config.

Bits	Description
7	<b>AlertMask: alert mask.</b> Read-write. Reset: Cold,0. See SBTSI::Config[AlertMask].
6	<b>RunStop: run stop.</b> Read-write. Reset: Cold,0. See SBTSI::Config[RunStop].
5	<b>ReadOrder: atomic read order.</b> Read-write. Reset: Cold,0. See SBTSI::Config[ReadOrder].
4:2	Reserved.
1	<b>AraDis: ARA disable.</b> Read-write. Reset: Cold,0. See SBTSI::Config[AraDis].
0	Reserved.

#### SBTSIx10 [CPU Decimal Temperature] (SBTSI::CpuTempDec)

Read-only.

See SBTSI::CpuTempInt.	
Bits	Description
7:5	<b>CpuTempDec: decimal CPU temperature value.</b> Read-only. Reset: Cold,XXXb. Read-only. This field returns the decimal portion of the CPU temperature.
4:0	Reserved.

**SBTSIx11 [CPU Temperature Offset High Byte] (SBTSI::CpuTempOffInt)**

Read-write. Reset: Cold,00h.

SBTSI::CpuTempOffInt and SBTSI::CpuTempOffDec combine to specify the CPU temperature offset. See 6.2.6 [SB-TSI Temperature Offset Encoding] for encoding details.

Bits	Description
7:0	<b>CpuTempOffInt: CPU temperature integer offset.</b> Read-write. Reset: Cold,00h. This field specifies the integer portion of the CPU temperature offset added to Tctl to calculate the CPU temperature. Write access causes a reset of the alert history counters (specified by SBTSI::AlertThreshold[AlertThr]) and the corresponding timer (specified by SBTSI::UpdateRate[UpRate]).

**SBTSIx12 [CPU Temperature Decimal Offset] (SBTSI::CpuTempOffDec)**

Read-write. Reset: Cold,00h.

See SBTSI::CpuTempOffInt.

Bits	Description
7:5	<b>CpuTempOffDec: CPU temperature decimal offset.</b> Read-write. Reset: Cold,0h. This field specifies the decimal/fractional portion of the CPU temperature offset added to Tctl to calculate the CPU temperature. Write access causes a reset of the alert history counters (specified by SBTSI::AlertThreshold[AlertThr]) and the corresponding timer (specified by SBTSI::UpdateRate[UpRate]).
4:0	Reserved.

**SBTSIx13 [High Temperature Decimal Threshold] (SBTSI::HiTempDec)**

Read-write. Reset: Cold,00h.

See SBTSI::HiTempInt.

Bits	Description
7:5	<b>HiTempDec: high temperature decimal threshold.</b> Read-write. Reset: Cold,0h. This field specifies the decimal portion of the high temperature threshold.
4:0	Reserved.

**SBTSIx14 [Low Temperature Decimal Threshold] (SBTSI::LoTempDec)**

Read-write. Reset: Cold,00h.

See SBTSI::LoTempInt.

Bits	Description
7:5	<b>LoTempDec: low temperature decimal threshold.</b> Read-write. Reset: Cold,0h. This field specifies the decimal portion of the low temperature threshold.
4:0	Reserved.

**SBTSIx22 [Timeout Configuration] (SBTSI::TimeoutConfig)**

Read-write. Reset: Cold,80h.

Bits	Description
7	<b>TimeoutEn: SMBus timeout enable.</b> Read-write. Reset: Cold,1. 0=SMBus defined timeout support disabled. 1=SMBus defined timeout support enabled. SMBus timeout enable.
6:0	Reserved.

**SBTSIx32 [Alert Threshold Register] (SBTSI::AlertThreshold)**

Read-write. Reset: Cold,00h.

See 6.2.3 [Alert Behavior].

Bits	Description
7:3	Reserved.
2:0	<b>AlertThr: alert threshold.</b> Read-write. Reset: Cold,0h. Specifies the number of consecutive CPU temperature samples for which a temperature alert condition needs to remain valid before the corresponding alert bit is set. For SBTSI::AlertConfig[AlertCompEn] == 1, it specifies the number of consecutive CPU temperature samples for which a temperature alert condition need to remain not valid before the corresponding alert bit gets cleared. Write access resets the alert history counters (specified by SBTSI::AlertThreshold[AlertThr]) and the corresponding timer (specified by SBTSI::UpdateRate[UpRate]). Details in SBTSI::Status.
<b>ValidValues:</b>	
Value	Description
0h	1 Sample
6h-1h	<Value+1> Samples
7h	8 Samples

#### SBTSIxBF [Alert Configuration] (SBTSI::AlertConfig)

Read-write.

Bits	Description
7:1	Reserved.
0	<b>AlertCompEn: alert comparator mode enable.</b> Read-write. Reset: Cold,X. 0=SBTSI::Status[TempHighAlert] and SBTSI::Status[TempLowAlert] are Read to clear. 1=SBTSI::Status[TempHighAlert] and SBTSI::Status[TempLowAlert] are Read-only; ARA response disabled. Write access does not change the alert history counters (specified by SBTSI::AlertThreshold[AlertThr]) or the corresponding timer (specified by SBTSI::UpdateRate[UpRate]). See SBTSI::Status.

#### SBTSIxFE [Manufacture ID] (SBTSI::ManId)

Read-only. Reset: Cold,00h.

Bits	Description
7:1	Reserved.
0	<b>ManId: Manufacture ID.</b> Read-only. Reset: Cold,0. Returns the AMD manufacture ID.

#### SBTSIxFF [Revision] (SBTSI::Revision)

Read-only. Reset: Cold,04h.

Bits	Description
7:0	<b>Revision: SB-TSI revision.</b> Read-only. Reset: Cold,04h. Specifies the SBI temperature sensor interface revision.

## 7 Data Fabric

The Scalable Data Fabric (SDF) is a coherent fabric connecting CPU, GPU, multimedia clients, I/O, and DRAM.

### 7.1 Data Fabric Instance and Fabric IDs

Every data fabric component is identified by a unique instance identifier ("InstanceID") used to access component registers.

#### 7.1.1 System InstanceID and FabricID Assignment

The table below displays the DF Component identifiers.

*Table 68: Instance and Fabric IDs*

Component	InstanceID	FabricID[4:0]
Coherent Slave 0 (CS0)	0	0
Coherent Slave 1 (CS1)	1	1
Coherent Core Master 0 (CCM0)	2	2
Graphics Core Master 0 (GCM0)	3	3
Graphics Core Master 1 (GCM1)	4	4
MultiMedia Hub 0 (MMHUB0)	5	5
Display Controller 0 (DCN0)	6	6
Address Translation Hub 0 (ATHUB0)	7	7
Cache Resident Self-Test 0 (CReST0)	8	8
I/O master and slave 0 (IOMS0)	9	9
PIE (PIE0)	10	10

## 8 Northbridge IO (NBIO)

### 8.1 IOMMU

The I/O Memory Management Unit (IOMMU) extends the AMD64 system architecture by adding support for address translation and system memory access protection on DMA transfers from peripheral devices.

#### 8.1.1 Functional Description

##### 8.1.1.1 Definitions

Table 69: Link Definitions

Term	Description
<b>IOMMU</b>	IO Memory Management Unit
<b>SDXI</b>	Smart Data Accelerator Interface

#### 8.1.2 Registers

##### 8.1.2.1 IOMMUL1 Registers

IOMMUL1INT[0...1]x0000001C (IOMMUL1::L1_MISC_CNTRL_1)	
Read-write. Reset: 0000_0000h.	
_inst[IOAGR,PCIE0]_aliasSMN; IOMMUL1INT[1:0]x0000001C; IOMMUL1INT[1:0]=14[8:7]0_0000h	
Bits	Description
31:18	Reserved.
17	<b>REG_enable_nw</b> . Read-write. Reset: 0. 0=Don't force. 1=Force NW low for ATS. Force NW low.
16:7	Reserved.
6	<b>REG_force_invalid_filter</b> . Read-write. Reset: 0. 0=Don't force. 1=Force invalidate_filter_on in L1. Force filter invalidation.
5:1	Reserved.
0	<b>REG_force_OrderStreamID_func</b> . Read-write. Reset: 0. 0=Force the Function bit in StreamID/ReqID to '0' for Ordering in L1. 1=Use the full StreamID/ReqID bus. Force StreamID/ReqID for ordering.
IOMMUL1INT[0...1]x00000024 (IOMMUL1::L1_SB_LOCATION)	
Read-write. Reset: 0000_0000h.	
_inst[IOAGR,PCIE0]_aliasSMN; IOMMUL1INT[1:0]x00000024; IOMMUL1INT[1:0]=14[8:7]0_0000h	
Bits	Description
31:16	<b>SBlocated_Core</b> . Read-write. Reset: 0000h. Core location of SB.
15:0	<b>SBlocated_Port</b> . Read-write. Reset: 0000h. Port location of SB.
IOMMUL1INT[0...1]x00000030 (IOMMUL1::L1_CNTRL_0)	
Read-write. Reset: 0000_0000h.	
_inst[IOAGR,PCIE0]_aliasSMN; IOMMUL1INT[1:0]x00000030; IOMMUL1INT[1:0]=14[8:7]0_0000h	
Bits	Description



31:1	Reserved.
0	<b>Unfilter_dis.</b> Read-write. Reset: 0. 0=Enable. 1=Disable. Disable unfiltering in the L1 WQ of aborted L2 requests.

**IOMMUL1INT[0...1]x00000038 (IOMMUL1::L1\_CNTRL\_2)**

Read-write. Reset: 1200\_0000h.

\_inst[IOAGR,PCIE0]\_aliasSMN; IOMMUL1INT[1:0]x00000038; IOMMUL1INT[1:0]=14[8:7]0\_0000h

Bits	Description												
31:29	Reserved.												
28	<b>L1ConsumedDataErrorSignalEn.</b> Read-write. Reset: 1. 0=Disable. 1=Enable. Enables asserting Data Error on egress SDP request when parity check fails on IOMMU consumed data.												
27	<b>L1NonConsumedDataErrorSignalEn.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. Enables asserting Data Error on egress SDP request when parity check fails on IOMMU non-consumed data.												
26:24	<b>CPD_RESP_MODE.</b> Read-write. Reset: 2h. Consumption of poisoned data response mode. <b>Valid Values:</b>												
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>No response taken when IOMMU consumes poisoned data.</td></tr> <tr> <td>1h</td><td>ErrEvent issued when IOMMU consumes poisoned data.</td></tr> <tr> <td>2h</td><td>Target Abort issued when IOMMU consumes poisoned data.</td></tr> <tr> <td>3h</td><td>Master Abort issued when IOMMU consumes poisoned data.</td></tr> <tr> <td>7h-4h</td><td>Reserved.</td></tr> </table>	Value	Description	0h	No response taken when IOMMU consumes poisoned data.	1h	ErrEvent issued when IOMMU consumes poisoned data.	2h	Target Abort issued when IOMMU consumes poisoned data.	3h	Master Abort issued when IOMMU consumes poisoned data.	7h-4h	Reserved.
Value	Description												
0h	No response taken when IOMMU consumes poisoned data.												
1h	ErrEvent issued when IOMMU consumes poisoned data.												
2h	Target Abort issued when IOMMU consumes poisoned data.												
3h	Master Abort issued when IOMMU consumes poisoned data.												
7h-4h	Reserved.												
23:0	Reserved.												

**IOMMUL1INT[0...1]x0000009C (IOMMUL1::L1\_FEATURE\_CNTRL)**

Read-write. Reset: 0000\_0000h.

\_inst[IOAGR,PCIE0]\_aliasSMN; IOMMUL1INT[1:0]x0000009C; IOMMUL1INT[1:0]=14[8:7]0\_0000h

Bits	Description
31:3	Reserved.
2	<b>EXE_lock_bit.</b> Read-write. Reset: 0. When programmed to 1 will remain 1 until reset. Sticky lock bits for the EXE permission checks.
1	<b>PMR_lock_bit.</b> Read-write. Reset: 0. When programmed to 1 will remain 1 until reset. Sticky lock bits for the PMR permission checks.
0	Reserved.

**IOMMUL1INT[0...1]x000000A8 (IOMMUL1::L1\_PGMEM\_CTRL\_5)**

Read-write. Reset: 0000\_0001h.

\_inst[IOAGR,PCIE0]\_aliasSMN; IOMMUL1INT[1:0]x000000A8; IOMMUL1INT[1:0]=14[8:7]0\_0000h

Bits	Description
31:0	<b>L1_LS_Req_Maintain_Cnt.</b> Read-write. Reset: 0000_0001h. LS request maintain time.

**IOMMUL1INT[0...1]x000000AC (IOMMUL1::L1\_PGMEM\_CTRL\_6)**

Read-write. Reset: 0000\_0001h.

\_inst[IOAGR,PCIE0]\_aliasSMN; IOMMUL1INT[1:0]x000000AC; IOMMUL1INT[1:0]=14[8:7]0\_0000h

Bits	Description
31:0	<b>L1_LS_Exit_Maintain_Cnt.</b> Read-write. Reset: 0000_0001h. LS request exit maintain time.

**IOMMUL1INT[0...1]x000000B0 (IOMMUL1::L1\_PGMEM\_CTRL\_7)**

Read-write. Reset: 0000\_0001h.

\_inst[IOAGR,PCIE0]\_aliasSMN; IOMMUL1INT[1:0]x000000B0; IOMMUL1INT[1:0]=14[8:7]0\_0000h

Bits	Description
31:0	<b>L1_DS_Req_Maintain_Cnt.</b> Read-write. Reset: 0000_0001h. DS request maintain time.

**IOMMUL1INT[0...1]x000000B4 (IOMMUL1::L1\_PGMEM\_CTRL\_8)**

Read-write. Reset: 0000\_0006h.

\_inst[IOAGR,PCIE0]\_aliasSMN; IOMMUL1INT[1:0]x000000B4; IOMMUL1INT[1:0]=14[8:7]0\_0000h

Bits	Description
31:0	<b>L1_DS_Exit_Maintain_Cnt.</b> Read-write. Reset: 0000_0006h. DS request exit maintain time.

**IOMMUL1INT[0...1]x000000B8 (IOMMUL1::L1\_PGMEM\_CTRL\_9)**

Read-write. Reset: 0000\_0001h.

\_inst[IOAGR,PCIE0]\_aliasSMN; IOMMUL1INT[1:0]x000000B8; IOMMUL1INT[1:0]=14[8:7]0\_0000h

Bits	Description
31:0	<b>L1_SD_Req_Maintain_Cnt.</b> Read-write. Reset: 0000_0001h. SD request maintain time.

**IOMMUL1INT[0...1]x000000BC (IOMMUL1::L1\_PGMEM\_CTRL\_10)**

Read-write. Reset: 0000\_0006h.

\_inst[IOAGR,PCIE0]\_aliasSMN; IOMMUL1INT[1:0]x000000BC; IOMMUL1INT[1:0]=14[8:7]0\_0000h

Bits	Description
31:0	<b>L1_SD_Exit_Maintain_Cnt.</b> Read-write. Reset: 0000_0006h. SD request exit maintain time.

**IOMMUL1INT[0...1]x000000C8 (IOMMUL1::L1\_CNTRL\_4)**

Read-write. Reset: 0000\_0000h.

\_inst[IOAGR,PCIE0]\_aliasSMN; IOMMUL1INT[1:0]x000000C8; IOMMUL1INT[1:0]=14[8:7]0\_0000h

Bits	Description
31:3	Reserved.
2	<b>Timeout_pulse_ext_En.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. Use external pulse for Invalidation Timeout.
1	Reserved.
0	<b>ATS_multiple_resp_en.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. Allow multiple ATS responses for a large sized ATS request.

**IOMMUL1INT[0...1]x000000CC (IOMMUL1::L1\_CLKCNTRL\_0)**

Read-write. Reset: 0002\_0000h.

\_inst[IOAGR,PCIE0]\_aliasSMN; IOMMUL1INT[1:0]x000000CC; IOMMUL1INT[1:0]=14[8:7]0\_0000h

Bits	Description
31	<b>L1_L2_CLKGATE_EN.</b> Read-write. Reset: 0. 0=L1 does not wait for L2toL1ClkGrant before sending request. 1=L1 needs to receive L2toL1ClkGrant before sending request. This register must be set if L2 clock gating is enabled.
30:22	Reserved.
21:14	<b>L1_CLKGATE_HYSTERESIS.</b> Read-write. Reset: 08h. IOMMU L1 Clock branch hysteresis used in all the branches. Resets to 16 clycles.
13	<b>L1_HOSTRSP_CLKGATE_EN.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. Enables clock gating for internal Host Response path blocks.
12	<b>L1_DMARSP_CLKGATE_EN.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. Enables clock gating for internal DMA Response path blocks.
11	<b>L1_HOSTREQ_CLKGATE_EN.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. Enables clock gating for internal Host path blocks.
10	<b>L1_REG_CLKGATE_EN.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. Enables clock gating for internal register blocks.
9	<b>L1_MEMORY_CLKGATE_EN.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. Enables clock gating for internal memory blocks.
8	<b>L1_PERF_CLKGATE_EN.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. Enables clock gating for internal Performance blocks.
7	Reserved.
6	<b>L1_CPSLV_CLKGATE_EN.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. Enables clock gating for internal

	Command Process Slave blocks.
5	<b>L1_CACHE_CLKGATE_EN</b> . Read-write. Reset: 0. 0=Disable. 1=Enable. Enables clock gating for internal Cache blocks.
4	<b>L1_DMA_CLKGATE_EN</b> . Read-write. Reset: 0. 0=Disable. 1=Enable. Enables clock gating for internal DMA path blocks.
3:0	Reserved.

**IOMMUL1INT[0...1]x000000D4 (IOMMUL1::L1\_SDP\_CLKREQ\_CNTRL)**

Read-write. Reset: 0000\_0000h.

\_inst[IOAGR,PCIE0]\_aliasSMN; IOMMUL1INT[1:0]x000000D4; IOMMUL1INT[1:0]=14[8:7]0\_0000h

Bits	Description
31:2	Reserved.
1	<b>HW_PG_WAKEUP_EN_HOST</b> . Read-write. Reset: 0. 0=Disable. 1=Enable. Connect the ingress OrigClkReq signal on the IOHC SDP interface to the OrigClkReq on the L1_CLIENT interface.
0	<b>HW_PG_WAKEUP_EN_DMA</b> . Read-write. Reset: 0. 0=Disable. 1=Enable. Connect the ingress OrigClkReq signal on the client SDP interface to the OrigClkReq on the L1_IOHC interface, and the L1_L2_req_clkreq to the L2.

**IOMMUL1INT[0...1]x000000F0 (IOMMUL1::L1\_PGMEM\_CTRL\_4)**

Read-write. Reset: 0000\_0000h.

\_inst[IOAGR,PCIE0]\_aliasSMN; IOMMUL1INT[1:0]x000000F0; IOMMUL1INT[1:0]=14[8:7]0\_0000h

Bits	Description
31:0	<b>L1_SD_thres</b> . Read-write. Reset: 0000_0000h.

**IOMMUL1INT[0...1]x000000F4 (IOMMUL1::IOMMU\_PGSLV\_CONTROL)**

Reset: 0000\_000Bh.

Register used in L1 ECO.

\_inst[IOAGR,PCIE0]\_aliasSMN; IOMMUL1INT[1:0]x000000F4; IOMMUL1INT[1:0]=14[8:7]0\_0000h

Bits	Description
31:6	Reserved.
5	<b>L1_PG_STATUS</b> . Read-only. Reset: 0. 0=Not in isolation. 1=In isolation (power-gated). Software readable register for finding the isolation status of the L1.
4:0	<b>CFG_IDLE_HYSTERESIS</b> . Read-write. Reset: 0Bh. Controls the number of consecutive clocks when IP is idle before asserting PGMS_PGSLV_pwrGate_ready. Large enough to account for L2->L1 ClkReq/ClkAck path with repeaters. Cannot be smaller than 0xB.

**8.1.2.2 IOMMUL2 Registers****D00F2x000 (IOMMUL2::IOMMU\_VENDOR\_ID)**

Read-only. Reset: 1022h.

\_aliasHOST; D00F2x000

\_aliasSMN; IOMMUCFGx00000000; IOMMUCFG=13F0\_0000h

Bits	Description
15:0	<b>VENDOR_ID</b> . Read-only. Reset: 1022h. Vendor Identifier. This 16-bit field identifies the manufacturer of the device: Advanced Micro Devices Inc.

**D00F2x002 (IOMMUL2::IOMMU\_DEVICE\_ID)**

Read-only. Reset: 15D1h.

\_aliasHOST; D00F2x002

\_aliasSMN; IOMMUCFGx00000002; IOMMUCFG=13F0\_0000h

Bits	Description
------	-------------

15:0	<b>DEVICE_ID.</b> Read-only. Reset: 15D1h. Device Identifier. This 16-bit field is assigned by the device manufacturer and identifies the type of device.
------	---

**D00F2x004 (IOMMUL2::IOMMU\_COMMAND)**

_aliasHOST; D00F2x004	
_aliasSMN; IOMMUCFGx00000004; IOMMUCFG=13F0_0000h	
Bits	Description
15:11	Reserved.
10	<b>INTERRUPT_DIS.</b> Read-write. Reset: 0. 0=This function is allowed to generate legacy INTx interrupts. 1=This function is not allowed to generate legacy INTx interrupts. <b>Description:</b> Interrupt Disable. 1=This function is not allowed to generate legacy INTx interrupts. 0=This function is allowed to generate legacy INTx interrupts.
9	Reserved.
8	<b>SERR_EN.</b> Read-only. Reset: Fixed,0. 0=Disable. 1=Enable. <b>Description:</b> System Error Enable. This function does not set IOMMUL2::IOMMU_STATUS[SIGNALLED_SYSTEM_ERROR].
7	Reserved.
6	<b>PARITY_ERROR_EN.</b> Read-write. Reset: 0. 0=This function is not allowed to set IOMMUL2::IOMMU_STATUS[PARITY_ERROR_DETECTED]. 1=This function is allowed to set IOMMUL2::IOMMU_STATUS[PARITY_ERROR_DETECTED]. Parity Error Enable.
5:3	Reserved.
2	<b>BUS_MASTER_EN.</b> Read-write. Reset: 0. Init: BIOS,1. 0=This function is not allowed to generate DMA requests. 1=This function is allowed to generate DMA requests. Bus Master Enable.
1	<b>MEM_ACCESS_EN.</b> Read-only. Reset: Fixed,0. 0=Disable. 1=Enable. <b>Description:</b> Memory Access Enable. This register controls no hardware. This function does not declare any standard memory BARs.
0	<b>IO_ACCESS_EN.</b> Read-only. Reset: Fixed,0. 0=Disable. 1=Enable. <b>Description:</b> I/O Access Enable. This register controls no hardware. This function does not declare any standard I/O BARs.

**D00F2x006 (IOMMUL2::IOMMU\_STATUS)**

_aliasHOST; D00F2x006	
_aliasSMN; IOMMUCFGx00000006; IOMMUCFG=13F0_0000h	
Bits	Description
15	<b>PARITY_ERROR_DETECTED.</b> Read,Write-1-to-clear. Reset: 0. <b>Description:</b> Parity Error Detected. This bit is set by hardware whenever this function receives a Read response with the data error attribute set.
14	<b>SIGNALLED_SYSTEM_ERROR.</b> Read-only. Reset: Fixed,0. <b>Description:</b> Signaled System Error. This bit is always set to 0 because this function does not generate signaled system error status.
13	<b>RECEIVED_MASTER_ABORT.</b> Read,Write-1-to-clear. Reset: 0. 0=Inactive. 1=Active. <b>Description:</b> Received Target Abort. This bit is set by hardware whenever this function receives a Read response with the master abort status.
12	<b>RECEIVED_TARGET_ABORT.</b> Read,Write-1-to-clear. Reset: 0. 0=Inactive. 1=Active. <b>Description:</b> Received Target Abort. This bit is set by hardware whenever this function receives a Read response with the target abort status.
11	<b>SIGNAL_TARGET_ABORT.</b> Read-only. Reset: Fixed,0. 0=No Abort. 1=Target Abort asserted. <b>Description:</b> Signal Target Abort. This bit is always set to 0 because this function does not terminate host requests with target abort.
10:9	Reserved.

8	<b>MASTER_DATA_ERROR.</b> Read,Write-1-to-clear. Reset: 0. 0=Disable. 1=Enable. <b>Description:</b> Master Data Error. This bit is set by hardware whenever IOMMUL2::IOMMU_COMMAND[PARITY_ERROR_EN] is set and this function receives a Read response with the data error attribute set.
7:5	Reserved.
4	<b>CAP_LIST.</b> Read-only. Reset: Fixed,1. 0=Disable. 1=Enable. <b>Description:</b> Capabilities List. This bit is set to indicate that this function's configuration space supports a capabilities list.
3	<b>INT_Status.</b> Read-only. Reset: 0. 0=This function does not have any outstanding interrupts. 1=This function has an outstanding interrupt. Interrupt Status.
2:0	Reserved.

**D00F2x008 (IOMMUL2::IOMMU\_REVISION\_ID)**

Read-only. Reset: 00h.

\_aliasHOST; D00F2x008

\_aliasSMN; IOMMUCFGx00000008; IOMMUCFG=13F0\_0000h

Bits	Description
7:4	<b>MAJOR_REV_ID.</b> Read-only. Reset: 0h. Identifies the revision number of the function.
3:0	<b>MINOR_REV_ID.</b> Read-only. Reset: 0h. Identifies the stepping number of the function.

**D00F2x009 (IOMMUL2::IOMMU\_REGPROG\_INF)**

Read-only. Reset: Fixed,00h.

\_aliasHOST; D00F2x009

\_aliasSMN; IOMMUCFGx00000009; IOMMUCFG=13F0\_0000h

Bits	Description
7:0	<b>REG_LEVEL_PROG_INF.</b> Read-only. Reset: Fixed,00h. <b>Description:</b> Register Programming Interface. IOMMU.

**D00F2x00A (IOMMUL2::IOMMU\_SUB\_CLASS)**

Read-only. Reset: Fixed,06h.

\_aliasHOST; D00F2x00A

\_aliasSMN; IOMMUCFGx0000000A; IOMMUCFG=13F0\_0000h

Bits	Description
7:0	<b>SUB_CLASS_INF.</b> Read-only. Reset: Fixed,06h. <b>Description:</b> Sub-Class Code. IOMMU.

**D00F2x00B (IOMMUL2::IOMMU\_BASE\_CODE)**

Read-only. Reset: Fixed,08h.

\_aliasHOST; D00F2x00B

\_aliasSMN; IOMMUCFGx0000000B; IOMMUCFG=13F0\_0000h

Bits	Description
7:0	<b>BASE_CLASS_CODE.</b> Read-only. Reset: Fixed,08h. <b>Description:</b> Sub-Class Code. System Base Peripheral.

**D00F2x00E (IOMMUL2::IOMMU\_HEADER)**

Read-only. Reset: Fixed,80h.

\_aliasHOST; D00F2x00E

\_aliasSMN; IOMMUCFGx0000000E; IOMMUCFG=13F0\_0000h

Bits	Description
7:0	<b>HEADER_TYPE.</b> Read-only. Reset: Fixed,80h.

	<b>Description:</b> Header Type. Indicates a multi-function device with a type 0 configuration space header.
--	---

**D00F2x02C (IOMMUL2::IOMMU\_ADAPTER\_ID)**

Read-only. Reset: 1451\_1022h.

\_aliasHOST; D00F2x02C

\_aliasSMN; IOMMUCFGx0000002C; IOMMUCFG=13F0\_0000h

Bits	Description
31:16	<b>SUBSYSTEM_ID.</b> Read-only. Reset: 1451h. Subsystem ID.
15:0	<b>SUBSYSTEM_VENDOR_ID.</b> Read-only. Reset: 1022h. Subsystem vendor ID.

**D00F2x034 (IOMMUL2::IOMMU\_CAPABILITIES\_PTR)**

Read-only. Reset: Fixed,0000\_0040h.

\_aliasHOST; D00F2x034

\_aliasSMN; IOMMUCFGx00000034; IOMMUCFG=13F0\_0000h

Bits	Description
31:8	Reserved.
7:0	<b>CAP_PTR.</b> Read-only. Reset: Fixed,40h. <b>Description:</b> Capabilities Pointer. Points to the start of the capabilities list.

**D00F2x03C (IOMMUL2::IOMMU\_INTERRUPT\_LINE)**

Read-write. Reset: 00h.

\_aliasHOST; D00F2x03C

\_aliasSMN; IOMMUCFGx0000003C; IOMMUCFG=13F0\_0000h

Bits	Description
7:0	<b>INTERRUPT_LINE.</b> Read-write. Reset: 00h. Init: BIOS,FFh. This register is Read/Write for software compatibility. It controls no hardware.

**D00F2x03D (IOMMUL2::IOMMU\_INTERRUPT\_PIN)**

Read-only. Reset: 01h.

\_aliasHOST; D00F2x03D

\_aliasSMN; IOMMUCFGx0000003D; IOMMUCFG=13F0\_0000h

Bits	Description														
7:0	<b>INTERRUPT_PIN.</b> Read-only. Reset: 01h. This field indicates the INTx line used to generate legacy interrupts. All other encodings are not supported. <b>Valid Values:</b>														
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>00h</td><td>Reserved.</td></tr> <tr> <td>01h</td><td>INTA</td></tr> <tr> <td>02h</td><td>INTB</td></tr> <tr> <td>03h</td><td>INTC</td></tr> <tr> <td>04h</td><td>INTD</td></tr> <tr> <td>FFh-05h</td><td>Reserved.</td></tr> </table>	Value	Description	00h	Reserved.	01h	INTA	02h	INTB	03h	INTC	04h	INTD	FFh-05h	Reserved.
Value	Description														
00h	Reserved.														
01h	INTA														
02h	INTB														
03h	INTC														
04h	INTD														
FFh-05h	Reserved.														

**D00F2x040 (IOMMUL2::IOMMU\_CAP\_HEADER)**

Read-only.

\_aliasHOST; D00F2x040

\_aliasSMN; IOMMUCFGx00000040; IOMMUCFG=13F0\_0000h

Bits	Description
31:29	Reserved.
28	<b>IOMMU_CAP_EXT.</b> Read-only. Reset: 1. 0=IOMMU Miscellaneous Information Register 1 [Capability Offset

	14h] not supported. 1=Indicates support for IOMMU Miscellaneous Information Register 1 [Capability Offset 14h]. IOMMU Miscellaneous Information Register 1 support (IOMMUMMIO::IOMMU_MMIO_EVENT_BASE_1).
27	<b>IOMMU_EFR_SUP.</b> Read-only. Reset: 1. 0=Extended Feature Register [MMIO Offset 0030h] is Reserved. 1=Indicates IOMMU Extended Feature Register [MMIO Offset 0030h] is supported. Extended Feature Register support (IOMMUMMIO::IOMMU_MMIO_EFR_0).
26	<b>IOMMU_NP_CACHE.</b> Read-only. Reset: Fixed,0. 0=Indicates the IOMMU cache's only page table entries that are marked as present. When IOMMU_NP_CACHE is clear, software must issue an invalidate after any change to a PDE or PTE marked present before the change. 1=Indicates the IOMMU cache's page table entries that are marked as not present. When this bit is set, software must issue an invalidate after any change to a PDE or PTE. Indication of the IOMMU caches page table entries.
25	<b>IOMMU_HT_TUNNEL_SUP.</b> Read-only. Reset: Fixed,0. Hypertransport tunnel translation support. This register is always set to 0 to indicate that the device does not contain a hypertransport tunnel supporting address translation.
24	<b>IOMMU_IO_TLBSUP.</b> Read-only. Reset: 1. Indicates support for remote IOTLBs.
23:19	<b>IOMMU_CAP_REV.</b> Read-only. Reset: Fixed,01h. Specifies the IOMMU specification revision.
18:16	<b>IOMMU_CAP_TYPE.</b> Read-only. Reset: Fixed,3h. Specifies the layout of the Capability Block as an IOMMU capability block.
15:8	<b>IOMMU_CAP_PTR.</b> Read-only. Reset: Fixed,64h. Indicates the location of the next capability block if one is present.
7:0	<b>IOMMU_CAP_ID.</b> Read-only. Reset: Fixed,0Fh. Indicates a Secure Device capability block.

**D00F2x044 (IOMMUL2::IOMMU\_CAP\_BASE\_LO)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; D00F2x044

\_aliasSMN; IOMMUCFGx00000044; IOMMUCFG=13F0\_0000h

Bits	Description
31:19	<b>IOMMU_BASE_ADDR_LO.</b> Read-write. Reset: 0000h. Specifies address bits[31:19] of the 512KB-aligned base address of the IOMMU memory-mapped control registers.
18:1	Reserved.
0	<b>IOMMU_ENABLE.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. IOMMU accepts memory accesses to the address specified in the Base Address Register. Upon writing IOMMU_ENABLE = 1, all IOMMU Read/Write capability registers in PCI configuration space are locked until the next system reset.

**D00F2x048 (IOMMUL2::IOMMU\_CAP\_BASE\_HI)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; D00F2x048

\_aliasSMN; IOMMUCFGx00000048; IOMMUCFG=13F0\_0000h

Bits	Description
31:0	<b>IOMMU_BASE_ADDR_HI.</b> Read-write. Reset: 0000_0000h. Specifies address bits[63:32] of the 512KB-aligned base address of the IOMMU memory-mapped control registers.

**D00F2x04C (IOMMUL2::IOMMU\_CAP\_RANGE)**

Read-only.

\_aliasHOST; D00F2x04C

\_aliasSMN; IOMMUCFGx0000004C; IOMMUCFG=13F0\_0000h

Bits	Description
31:24	<b>IOMMU_LAST_DEVICE.</b> Read-only. Reset: 00h. Indicates device and function number of the last integrated device associated with the IOMMU.
23:16	<b>IOMMU_FIRST_DEVICE.</b> Read-only. Reset: 00h. Indicates device and function number of the first integrated device associated with the IOMMU.



15:8	<b>IOMMU_BUS_NUMBER.</b> Read-only. Reset: 00h. Indicates the bus number that the IOMMU_FIRST_DEVICE and IOMMU_LAST_DEVICE reside on.
7	<b>IOMMU_RNG_VALID.</b> Read-only. Reset: 0. 0=Software must use the I/O topology information. 1=The IOMMU_BUS_NUMBER, IOMMU_FIRST_DEVICE and IOMMU_LAST_DEVICE fields are valid. Although the register contents are valid, software is encouraged to use the I/O topology information. IOMMU_BUS_NUMBER, IOMMU_FIRST_DEVICE and IOMMU_LAST_DEVICE validity status.
6:5	Reserved.
4:0	<b>IOMMU_UNIT_ID.</b> Read-only. Reset: Fixed,00h. This field returns the HyperTransport™ UnitID used by the IOMMU.

**D00F2x050 (IOMMUL2::IOMMU\_CAP\_MISC)**

\_aliasHOST; D00F2x050

\_aliasSMN; IOMMUCFGx00000050; IOMMUCFG=13F0\_0000h

Bits	Description																				
31:27	<b>IOMMU_MSI_NUM_PPR.</b> Read-only. Reset: 00h. This field must indicate which MSI vector is used for the interrupt message generated by the IOMMU for the peripheral page service request log when IOMMUMMIO::IOMMU_MMIO_EFR_0[PPR_SUP] == 1. IOMMU_MSI_NUM_PPR must be 0 when PPR_SUP == 0. For MSI, there can be only one IOMMU so this field must be 0. This interrupt is not remapped by the IOMMU.																				
26:23	Reserved.																				
22	<b>IOMMU_HT_ATS_RESV.</b> Read-write. Reset: 0. 0=The Address Translation address range can be translated by the IOMMU. 1=The HyperTransport Address Translation address range for ATS responses are Reserved and cannot be translated by the IOMMU. HyperTransport Address Translation address range for ATS responses.																				
21:15	<b>IOMMU_VA_SIZE.</b> Read-only. Reset: Fixed,40h. This field must indicate the size of the maximum virtual address processed by the IOMMU. The value is the (unsigned) binary log of the maximum address size. Allowed values are 32, 40, 48, and 64 bits. All other values are Reserved. <b>ValidValues:</b>																				
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>1Fh-00h</td><td>Reserved.</td></tr> <tr> <td>20h</td><td>32 bits.</td></tr> <tr> <td>27h-21h</td><td>Reserved.</td></tr> <tr> <td>28h</td><td>40 bits.</td></tr> <tr> <td>2Fh-29h</td><td>Reserved.</td></tr> <tr> <td>30h</td><td>48 bits.</td></tr> <tr> <td>3Fh-31h</td><td>Reserved.</td></tr> <tr> <td>40h</td><td>64 bits.</td></tr> <tr> <td>7Fh-41h</td><td>Reserved.</td></tr> </table>	Value	Description	1Fh-00h	Reserved.	20h	32 bits.	27h-21h	Reserved.	28h	40 bits.	2Fh-29h	Reserved.	30h	48 bits.	3Fh-31h	Reserved.	40h	64 bits.	7Fh-41h	Reserved.
Value	Description																				
1Fh-00h	Reserved.																				
20h	32 bits.																				
27h-21h	Reserved.																				
28h	40 bits.																				
2Fh-29h	Reserved.																				
30h	48 bits.																				
3Fh-31h	Reserved.																				
40h	64 bits.																				
7Fh-41h	Reserved.																				
14:8	<b>IOMMU_PA_SIZE.</b> Read-only. Reset: 30h. This field must indicate the size of the maximum physical address generated by the IOMMU. The value is the (unsigned) binary log of the maximum address size. Allowed values are 40, 48 and 52 bits. All other values are Reserved. <b>ValidValues:</b>																				
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>27h-00h</td><td>Reserved.</td></tr> <tr> <td>28h</td><td>40 bits.</td></tr> <tr> <td>2Fh-29h</td><td>Reserved.</td></tr> <tr> <td>30h</td><td>48 bits.</td></tr> <tr> <td>33h-31h</td><td>Reserved.</td></tr> <tr> <td>34h</td><td>52 bits.</td></tr> <tr> <td>7Fh-35h</td><td>Reserved.</td></tr> </table>	Value	Description	27h-00h	Reserved.	28h	40 bits.	2Fh-29h	Reserved.	30h	48 bits.	33h-31h	Reserved.	34h	52 bits.	7Fh-35h	Reserved.				
Value	Description																				
27h-00h	Reserved.																				
28h	40 bits.																				
2Fh-29h	Reserved.																				
30h	48 bits.																				
33h-31h	Reserved.																				
34h	52 bits.																				
7Fh-35h	Reserved.																				
7:5	<b>IOMMU_GVA_SIZE.</b> Read-only. Reset: Fixed,2h. This field indicates the width of the maximum guest virtual address processed by IOMMU.																				

	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	1h-0h	Reserved.
	2h	48 Bits.
	7h-3h	Reserved.
4:0	<b>IOMMU_MSI_NUM.</b> Read-only. Reset: 00h. Indicates the MSI vector used for interrupt messages generated by the IOMMU.	

**D00F2x054 (IOMMUL2::IOMMU\_CAP\_MISC\_1)**

Reset: 0000\_0000h.

\_aliasHOST; D00F2x054

\_aliasSMN; IOMMUCFGx00000054; IOMMUCFG=13F0\_0000h

Bits	Description
31:6	Reserved.
5	<b>IOMMU_ARCH_MODE.</b> Read-write. Reset: 0. 0=x86 mode. 1=ARM v1 mode. Indicates the working architecture mode of the IOMMU.
4:0	<b>IOMMU_MSI_NUM_GA.</b> Read-only. Reset: 00h. MSI message number. Message number for MSI or MSI-X interrupt associated with the guest vAPIC virtual interrupt request log.

**D00F2x064 (IOMMUL2::IOMMU\_MSI\_CAP)**

\_aliasHOST; D00F2x064

\_aliasSMN; IOMMUCFGx00000064; IOMMUCFG=13F0\_0000h

Bits	Description
31:24	Reserved.
23	<b>MSI_64_EN.</b> Read-only. Reset: Fixed,1. 0=64-bit MSI address not supported. 1=64-bit MSI address supported. Set to indicate that a 64-bit MSI address is supported.
22:20	<b>MSI_MULT_MESS_EN.</b> Read-write. Reset: 0h. Sets the number of MSI messages assigned to this function.
19:17	<b>MSI_MULT_MESS_CAP.</b> Read-only. Reset: 2h. Indicates the number of MSI messages requested by this function.
16	<b>MSI_EN.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. Enables MSI for this function and causes legacy interrupts to be disabled.
15:8	<b>MSI_CAP_PTR.</b> Read-only. Reset: Fixed,74h. Pointer to the next configuration space capability.
7:0	<b>MSI_CAP_ID.</b> Read-only. Reset: Fixed,05h. Indicates the MSI capability.

**D00F2x068 (IOMMUL2::IOMMU\_MSI\_ADDR\_LO)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; D00F2x068

\_aliasSMN; IOMMUCFGx00000068; IOMMUCFG=13F0\_0000h

Bits	Description
31:2	<b>MSI_ADDR_LO.</b> Read-write. Reset: 0000_0000h. Sets address bits[31:2] used to issue MSI messages.
1:0	Reserved.

**D00F2x06C (IOMMUL2::IOMMU\_MSI\_ADDR\_HI)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; D00F2x06C

\_aliasSMN; IOMMUCFGx0000006C; IOMMUCFG=13F0\_0000h

Bits	Description
31:0	<b>MSI_ADDR_HI.</b> Read-write. Reset: 0000_0000h. Sets address bits[63:32] used to issue MSI messages.

**D00F2x070 (IOMMUL2::IOMMU\_MSI\_DATA)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; D00F2x070

\_aliasSMN; IOMMUCFGx00000070; IOMMUCFG=13F0\_0000h

Bits	Description
31:16	Reserved.
15:0	<b>MSI_DATA</b> . Read-write. Reset: 0000h. Sets the data issued with MSI messages.

**D00F2x074 (IOMMUL2::IOMMU\_MSI\_MAPPING\_CAP)**

Read-only. Reset: Fixed,A803\_0008h.

\_aliasHOST; D00F2x074

\_aliasSMN; IOMMUCFGx00000074; IOMMUCFG=13F0\_0000h

Bits	Description
31:27	<b>MSI_MAP_CAP_TYPE</b> . Read-only. Reset: Fixed,15h. Indicates the MSI Mapping Capability.
26:18	Reserved.
17	<b>MSI_MAP_FIXD</b> . Read-only. Reset: Fixed,1. 0=MSI mapping range is programmable. 1=MSI mapping range is not programmable. Always set to 1 to indicate that this device only maps MSI interrupts with address 0xFEEX_XXXX onto Hypertransport interrupts and that the mapping range is not programmable.
16	<b>MSI_MAP_EN</b> . Read-only. Reset: Fixed,1. 0=MSI Mapping Capability disabled. 1=MSI Mapping Capability enabled. Always set to 1 to indicate that the MSI Mapping Capability is always enabled.
15:8	<b>MSI_MAP_CAP_PTR</b> . Read-only. Reset: Fixed,00h. Points to the next capability list item.
7:0	<b>MSI_MAP_CAP_ID</b> . Read-only. Reset: Fixed,08h. Indicates a Hypertransport capability list item.

**D00F2x084 (IOMMUL2::IOMMU\_MMIO\_CONTROL1\_W)**

Read-write. Reset: 0000\_0800h.

See also IOMMUMMIO::IOMMU\_MMIO\_EFR\_1.

\_aliasHOST; D00F2x084

\_aliasSMN; IOMMUCFGx00000084; IOMMUCFG=13F0\_0000h

Bits	Description										
31:21	Reserved.										
20	<b>HD_SUP_W</b> . Read-write. Reset: 0. 0=Hardware does not support Dirty bit updates for v1 Page tables. 1=Hardware supports Dirty bit update for v1 Page tables. Hardware Dirty bit update support.										
19:13	Reserved.										
12:11	<b>MARCnum_SUP_W</b> . Read-write. Reset: 1h. Writes number of MARC apoertures and thus the nubmer of MARC register 3-tuples supported.										
<b>ValidValues:</b>											
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>MARC feature not supported.</td></tr> <tr> <td>1h</td><td>4 MARC register 3-tuples supported.</td></tr> <tr> <td>2h</td><td>8 MARC register 4-tuples are supported.</td></tr> <tr> <td>3h</td><td>Reserved.</td></tr> </table>	Value	Description	0h	MARC feature not supported.	1h	4 MARC register 3-tuples supported.	2h	8 MARC register 4-tuples are supported.	3h	Reserved.
Value	Description										
0h	MARC feature not supported.										
1h	4 MARC register 3-tuples supported.										
2h	8 MARC register 4-tuples are supported.										
3h	Reserved.										
10:0	Reserved.										

**IOMMUL2Ax00000088 (IOMMUL2::L2A\_UPDATE\_FILTER\_CNTL)**

Read-write. Reset: 0000\_0001h.

\_aliasSMN; IOMMUL2Ax00000088; IOMMUL2A=1570\_0000h

Bits	Description
31:1	Reserved.
0	<b>L2a_Update_Filter_Bypass</b> . Read-write. Reset: 1. 0=Enable the dropping of updates that are already in the l2a_update_filter or in the destination L2a cache. 1=Disable duplicate update filtering. Dropping of updates that are already in the l2a_update_filter or in the destination L2a cache.

**IOMMUL2Ax00000094 (IOMMUL2::L2\_L2A\_DTRACE\_TRIGGER\_1)**

Read-write. Reset: 0000\_0000h.

\_aliasSMN; IOMMUL2Ax00000094; IOMMUL2A=1570\_0000h

Bits	Description
------	-------------

31:20	Reserved.
19:0	<b>DTraceVAPageUpper</b> . Read-write. Reset: 0_0000h. VA bits[63:44] for L2a debug bus transaction trigger.

**IOMMUL2Ax00000098 (IOMMUL2::L2\_L2A\_DTRACE\_TRIGGER\_2)**

Read-write. Reset: 0000_0000h.	
_aliasSMN; IOMMUL2Ax00000098; IOMMUL2A=1570_0000h	
Bits	Description
31:25	Reserved.
24:21	<b>DTraceClient</b> . Read-write. Reset: 0h. Client ID for L2a debug bus transaction trigger.
20	<b>DTraceZeroByteRead</b> . Read-write. Reset: 0. Zero-byte Read for L2a debug bus transaction trigger.
19:18	<b>DTraceAT</b> . Read-write. Reset: 0h. AT for L2a debug bus transaction trigger.
17	<b>DTraceInterrupt</b> . Read-write. Reset: 0. Interrupt for L2a debug bus transaction trigger.
16	<b>DTraceWr</b> . Read-write. Reset: 0. Write for L2a debug bus transaction trigger.
15:0	<b>DTraceReqID</b> . Read-write. Reset: 0000h. Request ID for L2a debug bus transaction trigger.

**IOMMUL2Ax0000009C (IOMMUL2::L2\_L2A\_DTRACE\_MASK\_0)**

Read-write. Reset: 0000_0000h.	
_aliasSMN; IOMMUL2Ax0000009C; IOMMUL2A=1570_0000h	
Bits	Description
31:0	<b>DTraceVAPageLowerMask</b> . Read-write. Reset: 0000_0000h. VA mask bits[43:12] for L2a debug bus transaction trigger.

**IOMMUL2Ax000000A0 (IOMMUL2::L2\_L2A\_DTRACE\_MASK\_1)**

Read-write. Reset: 0000_0000h.	
_aliasSMN; IOMMUL2Ax000000A0; IOMMUL2A=1570_0000h	
Bits	Description
31:20	Reserved.
19:0	<b>DTraceVAPageUpperMask</b> . Read-write. Reset: 0_0000h. VA mask bits[63:44] for L2a debug bus transaction trigger.

**IOMMUL2Ax000000A4 (IOMMUL2::L2\_L2A\_DTRACE\_MASK\_2)**

Read-write. Reset: 0000_0000h.	
_aliasSMN; IOMMUL2Ax000000A4; IOMMUL2A=1570_0000h	
Bits	Description
31:25	Reserved.
24:21	<b>DTraceClientMask</b> . Read-write. Reset: 0h. Client ID mask for L2a debug bus transaction trigger.
20	<b>DTraceZeroByteReadMask</b> . Read-write. Reset: 0. Zero-byte Read mask for L2a debug bus transaction trigger.
19:18	<b>DTraceATMask</b> . Read-write. Reset: 0h. AT mask for L2a debug bus transaction trigger.
17	<b>DTraceInterruptMask</b> . Read-write. Reset: 0. Interrupt mask for L2a debug bus transaction trigger.
16	<b>DTraceWrMask</b> . Read-write. Reset: 0. Write mask for L2a debug bus transaction trigger.
15:0	<b>DTraceReqIDMask</b> . Read-write. Reset: 0000h. Request ID mask for L2a debug bus transaction trigger.

**IOMMUL2Ax000000C0 (IOMMUL2::L2\_ERR\_RULE\_CONTROL\_3)**

Write-once. Reset: 0000_0000h.	
_aliasSMN; IOMMUL2Ax000000C0; IOMMUL2A=1570_0000h	
Bits	Description
31:1	Reserved.
0	<b>ERRRuleLock1</b> . Write-once. Reset: 0. 0=No effect. 1=Lock error detection rule. Setting this field bit locks the error detection rule.

**IOMMUL2Ax000000CC (IOMMUL2::L2\_L2A\_CK\_GATE\_CONTROL)**

Read-write. Reset: 0000_0057h.	
--------------------------------	--

_aliasSMN; IOMMUL2Ax000000CC; IOMMUL2A=1570_0000h	
Bits	Description
31:8	Reserved.
7:6	<b>CKGateL2AStop</b> . Read-write. Reset: 1h. Clock cycles to delay before stopping the clocks when clkready de-asserts.
<b>ValidValues:</b>	
<b>Value</b>	<b>Description</b>
0h	Allow 2 clock cycles of delay before stopping the clocks when clkready de-asserts.
1h	Allow 4 clock cycles of delay before stopping the clocks when clkready de-asserts.
2h	Allow 8 clock cycles of delay before stopping the clocks when clkready de-asserts.
3h	Allow 16 clock cycles of delay before stopping the clocks when clkready de-asserts.
5:4	<b>CKGateL2ALength</b> . Read-write. Reset: 1h. Clock cycles to delay before stopping the clocks when idle asserts.
<b>ValidValues:</b>	
<b>Value</b>	<b>Description</b>
0h	Allow 128 clock cycles of delay before stopping the clocks when idle asserts.
1h	Allow 256 clock cycles of delay before stopping the clocks when idle asserts.
2h	Allow 512 clock cycles of delay before stopping the clocks when idle asserts.
3h	Allow 1024 clock cycles of delay before stopping the clocks when idle asserts
3	Reserved.
2	<b>CKGateL2ACacheDisable</b> . Read-write. Reset: 1. Disable the gating of the L2b upper cache ways.
1	<b>CKGateL2ADynamicDisable</b> . Read-write. Reset: 1. Disable the gating of the L2b dynamic clock branch.
0	<b>CKGateL2ARegsDisable</b> . Read-write. Reset: 1. Disable the gating of the L2b register clock branch.

#### IOMMUL2Ax000000D0 (IOMMUL2::L2\_L2A\_PG\_SIZE\_CONTROL)

Read-write. Reset: 0000\_0101h.

\_aliasSMN; IOMMUL2Ax000000D0; IOMMUL2A=1570\_0000h

Bits	Description
31:15	Reserved.
14:8	<b>L2AREG_HOST_PG_SIZE</b> . Read-write. Reset: 01h. Controls the TLB search for different (final and partial) host page sizes.
<b>ValidValues:</b>	
<b>Bit</b>	<b>Description</b>
[0]	4k final TLB look up enabled (cannot be disabled).
[2:1]	Reserved.
[3]	2MB partial and final TLB look up enabled.
[5:4]	Reserved.
[6]	1GB partial and final TLB look up enabled.
7	Reserved.
6:0	<b>L2AREG_GST_PG_SIZE</b> . Read-write. Reset: 01h. Controls the TLB search for different (final and partial) guest page sizes.
<b>ValidValues:</b>	
<b>Bit</b>	<b>Description</b>
[0]	4k final TLB look up enabled (cannot be disabled).
[1]	Reserved.
[2]	Reserved.
[3]	2MB partial and final TLB look up enabled.
[4]	Reserved.
[5]	Reserved.
[6]	1GB partial and final TLB look up enabled.

**IOMMUL2Bx0000012C (IOMMUL2::L2\_SB\_LOCATION)**

Read-write. Reset: 0000\_0000h.

Specifies which PCIe® bridge is connected to the South Bridge (SB). Register set to 32'b0 indicates the GNB is secondary and has no SB connected.

\_aliasSMN; IOMMUL2Bx0000012C; IOMMUL2B=13F0\_1000h

Bits	Description												
31:16	<b>SBlocated_Core.</b> Read-write. Reset: 0000h.  <b>Description:</b> One hot encoding: 0x1 - SB is under GPP0 0x2 - SB is under GPP1 0x4 - SB is under GPP2 All other encodings are reserved.  <b>ValidValues:</b>												
	<table> <tr> <th>Bit</th><th>Description</th></tr> <tr> <td>[0]</td><td>SB is under GPP0.</td></tr> <tr> <td>[1]</td><td>SB is under GPP1.</td></tr> <tr> <td>[2]</td><td>SB is under GPP2.</td></tr> <tr> <td>[15:3]</td><td>Reserved.</td></tr> </table>	Bit	Description	[0]	SB is under GPP0.	[1]	SB is under GPP1.	[2]	SB is under GPP2.	[15:3]	Reserved.		
Bit	Description												
[0]	SB is under GPP0.												
[1]	SB is under GPP1.												
[2]	SB is under GPP2.												
[15:3]	Reserved.												
15:0	<b>SBlocated_Port.</b> Read-write. Reset: 0000h. One hot encoding.  <b>ValidValues:</b>												
	<table> <tr> <th>Bit</th><th>Description</th></tr> <tr> <td>[0]</td><td>SB is located on port A of SBlocated_Core.</td></tr> <tr> <td>[1]</td><td>SB is located on port B of SBlocated_Core.</td></tr> <tr> <td>[2]</td><td>SB is located on port C of SBlocated_Core.</td></tr> <tr> <td>[3]</td><td>SB is located on port D of SBlocated_Core.</td></tr> <tr> <td>[15:4]</td><td>Reserved.</td></tr> </table>	Bit	Description	[0]	SB is located on port A of SBlocated_Core.	[1]	SB is located on port B of SBlocated_Core.	[2]	SB is located on port C of SBlocated_Core.	[3]	SB is located on port D of SBlocated_Core.	[15:4]	Reserved.
Bit	Description												
[0]	SB is located on port A of SBlocated_Core.												
[1]	SB is located on port B of SBlocated_Core.												
[2]	SB is located on port C of SBlocated_Core.												
[3]	SB is located on port D of SBlocated_Core.												
[15:4]	Reserved.												

**IOMMUL2Bx00000130 (IOMMUL2::L2\_CONTROL\_5)**

Read-write. Reset: 0000\_1000h.

\_aliasSMN; IOMMUL2Bx00000130; IOMMUL2B=13F0\_1000h

Bits	Description
31:19	Reserved.
18:12	<b>GST_partial_ptc_cntrl.</b> Read-write. Reset: 01h. Prevent specified page sizes from stored in PTC cache.
11	<b>ForceTWonVC7.</b> Read-write. Reset: 0. 0=Don't force. 1=Force. Force table-walk fetches to use VC7 for all VC[1:0] and VC7 DMA requests.
10:0	Reserved.

**IOMMUL2Bx0000014C (IOMMUL2::L2B\_UPDATE\_FILTER\_CNTL)**

Read-write. Reset: 0000\_0001h.

\_aliasSMN; IOMMUL2Bx0000014C; IOMMUL2B=13F0\_1000h

Bits	Description
31:1	Reserved.
0	<b>L2b_Update_Filter_Bypass.</b> Read-write. Reset: 1. 1=Enable the dropping of updates that are already in the l2b_update_filter or in the PDC. L2 Update Filter bypass to enable the dropping of updates.

**IOMMUL2Bx00000150 (IOMMUL2::L2\_TW\_CONTROL)**

Read-write. Reset: 0040\_1000h.

\_aliasSMN; IOMMUL2Bx00000150; IOMMUL2B=13F0\_1000h

Bits	Description
31:25	Reserved.
24:22	<b>TWGuestPrefetchRange.</b> Read-write. Reset: 1h. Selects the number of pages to prefetch for Guest PTEs. A

	value of 0 is Reserved. Programming this register to 0 results in unpredictable hardware behavior.						
	<b>ValidValues:</b>						
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>Reserved.</td></tr> <tr> <td>7h-1h</td><td>Number of pages to prefetch.</td></tr> </table>	Value	Description	0h	Reserved.	7h-1h	Number of pages to prefetch.
Value	Description						
0h	Reserved.						
7h-1h	Number of pages to prefetch.						
21	<b>TWGuestPrefetchEn.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. Enable Guest prefetching in the table-walker.						
20:15	Reserved.						
14:12	<b>TWPrefetchRange.</b> Read-write. Reset: 1h. Selects the number of pages to prefetch.						
11:9	Reserved.						
8	<b>TWPrefetchEn.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. Enable prefetching in the table-walker.						
7:0	Reserved.						

**IOMMUL2Bx00000158 (IOMMUL2::L2\_CP\_CONTROL)**

Read-write. Reset: 0000\_0004h.

\_aliasSMN; IOMMUL2Bx00000158; IOMMUL2B=13F0\_1000h

Bits	Description
31:3	Reserved.
2	<b>CPFlushOnInv.</b> Read-write. Reset: 1. 0=No flush is performed during Invalidations. 1=Command Processor flushes out old requests on every Invalidation command. Command Processor flush on Invalidation command.
1	<b>CPFlushOnWait.</b> Read-write. Reset: 0. 0=No flush is performed on Completion Wait. 1=Command Processor flushes out old requests on Completion Wait. Command Processor flush on Completion Wait.
0	Reserved.

**IOMMUL2Bx0000015C (IOMMUL2::L2\_CP\_CONTROL\_1)**

Read-write. Reset: 0000\_0000h.

\_aliasSMN; IOMMUL2Bx0000015C; IOMMUL2B=13F0\_1000h

Bits	Description
31:16	Reserved.
15:0	<b>CPL1Off.</b> Read-write. Reset: 0000h. Each bit in this field indicates to the IOMMU Command Processor that a corresponding L1 TLB is inaccessible due to static clock or power gating. System software is responsible for programming this field.

**IOMMUL2Bx00000200 (IOMMUL2::L2\_ERR\_RULE\_CONTROL\_0)**

Read,Write-once. Reset: 0000\_0000h.

\_aliasSMN; IOMMUL2Bx00000200; IOMMUL2B=13F0\_1000h

Bits	Description
31:1	Reserved.
0	<b>ERRRuleLock0.</b> Read,Write-once. Reset: 0. This register is Write-once.

**IOMMUL2Bx00000240 (IOMMUL2::L2\_L2B\_CK\_GATE\_CONTROL)**

Read-write. Reset: 0000\_0057h.

\_aliasSMN; IOMMUL2Bx00000240; IOMMUL2B=13F0\_1000h

Bits	Description								
31:8	Reserved.								
7:6	<b>CKGateL2BStop.</b> Read-write. Reset: 1h. Clock cycles of delay before stopping the clocks when CLKREADY de-asserts.								
	<b>ValidValues:</b>								
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>Allow 2 clock cycles delay before stopping the clocks when CLKREADY de-asserts.</td></tr> <tr> <td>1h</td><td>Allow 4 clock cycles delay before stopping the clocks when CLKREADY de-asserts.</td></tr> <tr> <td>2h</td><td>Allow 8 clock cycles delay before stopping the clocks when CLKREADY de-asserts.</td></tr> </table>	Value	Description	0h	Allow 2 clock cycles delay before stopping the clocks when CLKREADY de-asserts.	1h	Allow 4 clock cycles delay before stopping the clocks when CLKREADY de-asserts.	2h	Allow 8 clock cycles delay before stopping the clocks when CLKREADY de-asserts.
Value	Description								
0h	Allow 2 clock cycles delay before stopping the clocks when CLKREADY de-asserts.								
1h	Allow 4 clock cycles delay before stopping the clocks when CLKREADY de-asserts.								
2h	Allow 8 clock cycles delay before stopping the clocks when CLKREADY de-asserts.								



	3h	Allow 16 clock cycles delay before stopping the clocks when CLKREADY de-asserts.
5:4	<b>CKGateL2BLength.</b> Read-write. Reset: 1h. Clock cycles of delay before stopping the clocks when IDLE asserts. <b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	Allow 128 clock cycles delay before stopping the clocks when IDLE asserts.
	1h	Allow 256 clock cycles delay before stopping the clocks when IDLE asserts.
	2h	Allow 512 clock cycles delay before stopping the clocks when IDLE asserts.
	3h	Allow 1024 clock cycles delay before stopping the clocks when IDLE asserts.
3	<b>CKGateL2BCacheDisable.</b> Read-write. Reset: 0. 0=Enabled. 1=Disabled. Disable the gating of the L2b upper cache ways.	
2	<b>CKGateL2BMiscDisable.</b> Read-write. Reset: 1. 0=Enabled. 1=Disabled. Disable the gating of the L2b miscellaneous clock branch.	
1	<b>CKGateL2BDynamicDisable.</b> Read-write. Reset: 1. 0=Enabled. 1=Disabled. Disable the gating of the L2b dynamic clock branch.	
0	<b>CKGateL2BRegsDisable.</b> Read-write. Reset: 1. 0=Enabled. 1=Disabled. Disable the gating of the L2b register clock branch.	

**IOMMUL2Bx00000250 (IOMMUL2::L2\_L2B\_PGSIZE\_CONTROL)**

Read-write. Reset: 0000\_0101h.

\_aliasSMN; IOMMUL2Bx00000250; IOMMUL2B=13F0\_1000h

Bits	Description
31:15	Reserved.
14:8	<b>L2BREG_HOST_PGSIZE.</b> Read-write. Reset: 01h. Controls the allowed final host page sizes stored in PTC cache. The table shows the valid programming bit values for this register. All other bits are Reserved. <b>ValidValues:</b>
	<b>Bit</b> <b>Description</b>
	[0]      Enable 4K (4K is always on, cannot be disabled).
	[2:1]      Reserved.
	[3]      Enable 2M final host page size caching.
	[5:4]      Reserved.
	[6]      Enabled 1G final host page size caching.
7	Reserved.
6:0	<b>L2BREG_GST_PGSIZE.</b> Read-write. Reset: 01h. Controls the allowed final guest page sizes stored in PTC cache. The table shows the valid programming bit values for this register. If a given final page size is disabled, the next enabled lowest page size will be cached. <b>ValidValues:</b>
	<b>Bit</b> <b>Description</b>
	[0]      Enable 4K (4K is always on, cannot be disabled).
	[1]      Reserved.
	[2]      Reserved.
	[3]      Enable 2M final guest page size caching.
	[4]      Reserved.
	[5]      Reserved.
	[6]      Enabled 1G final guest page size caching.

**8.1.2.3 IOMMUMMIO Registers****IOMMUBARx00000 (IOMMUMMIO::IOMMU\_MMIO\_DEVTBL\_BASE\_0)**

Read-write. Reset: 0000_0000h.	
_aliasHOST; IOMMUBARx00000; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI], IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO], 19'h0_0000}	
_aliasSMN; IOMMUMMIOx00000000; IOMMUMMIO=0240_0000h	
Bits	Description
31:12	<b>DEV_TBL_BASE_LO.</b> Read-write. Reset: 0_0000h. Specifies address bits[31:12] of the 4K byte-aligned base address of the first level device table.
11:9	Reserved.
8:0	<b>DEV_TBL_SIZE.</b> Read-write. Reset: 000h. This field contains 1 less than the length of the device table, in multiples of 4K bytes. A minimum size of 0 corresponds to a 4K-byte device table and a maximum size of 1FFh corresponds to a 2M-byte device table.

**IOMMUBARx00004 (IOMMUMMIO::IOMMU\_MMIO\_DEVTBL\_BASE\_1)**

Read-write. Reset: 0000_0000h.	
_aliasHOST; IOMMUBARx00004; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI], IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO], 19'h0_0000}	
_aliasSMN; IOMMUMMIOx00000004; IOMMUMMIO=0240_0000h	
Bits	Description
31:20	Reserved.
19:0	<b>DEV_TBL_BASE_HI.</b> Read-write. Reset: 0_0000h. Specifies address bits[51:32] of the 4K byte-aligned base address of the first level device table.

**IOMMUBARx00008 (IOMMUMMIO::IOMMU\_MMIO\_CMD\_BASE\_0)**

Read-write. Reset: 0000_0000h.	
_aliasHOST; IOMMUBARx00008; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI], IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO], 19'h0_0000}	
_aliasSMN; IOMMUMMIOx00000008; IOMMUMMIO=0240_0000h	
Bits	Description
31:12	<b>COM_BASE_LO.</b> Read-write. Reset: 0_0000h. Specifies address bits[31:12] of the base address of the command buffer. The base address programmed must be aligned to 4K bytes.
11:0	Reserved.

**IOMMUBARx0000C (IOMMUMMIO::IOMMU\_MMIO\_CMD\_BASE\_1)**

Read-write. Reset: 0800_0000h.													
_aliasHOST; IOMMUBARx0000C; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI], IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO], 19'h0_0000}													
_aliasSMN; IOMMUMMIOx0000000C; IOMMUMMIO=0240_0000h													
Bits	Description												
31:28	Reserved.												
27:24	<b>COM_LEN.</b> Read-write. Reset: 8h. Specifies the length of the command buffer in power of 2 increments. The minimum size is 256 entries (4K bytes); values less than 1000b are reserved.												
<b>Valid Values:</b>													
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>7h-0h</td><td>Reserved.</td></tr> <tr> <td>8h</td><td>256 entries (4K bytes).</td></tr> <tr> <td>9h</td><td>512 entries (8K bytes).</td></tr> <tr> <td>Eh-Ah</td><td>2<sup>^</sup>&lt;Value&gt; entries.</td></tr> <tr> <td>Fh</td><td>32768 entries (512K bytes).</td></tr> </table>	Value	Description	7h-0h	Reserved.	8h	256 entries (4K bytes).	9h	512 entries (8K bytes).	Eh-Ah	2 <sup>^</sup> <Value> entries.	Fh	32768 entries (512K bytes).
Value	Description												
7h-0h	Reserved.												
8h	256 entries (4K bytes).												
9h	512 entries (8K bytes).												
Eh-Ah	2 <sup>^</sup> <Value> entries.												
Fh	32768 entries (512K bytes).												
23:20	Reserved.												
19:0	<b>COM_BASE_HI.</b> Read-write. Reset: 0_0000h. Specifies address bits[51:32] of the base address of the command buffer. The base address programmed must be aligned to 4K bytes.												

**IOMMUBARx00010 (IOMMUMMIO::IOMMU\_MMIO\_EVENT\_BASE\_0)**

Read-write. Reset: 0000_0000h.	
--------------------------------	--

_aliasHOST; IOMMUBARx00010; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI] , IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO] , 19'h0_0000}	
_aliasSMN; IOMMUMMIOx00000010; IOMMUMMIO=0240_0000h	
Bits	Description
31:12	<b>EVENT_BASE_LO.</b> Read-write. Reset: 0_0000h. Specifies address bits[31:12] of the base address of the event log. The base address programmed must be aligned to 4K bytes.
11:0	Reserved.

#### IOMMUBARx00014 (IOMMUMMIO::IOMMU\_MMIO\_EVENT\_BASE\_1)

Read-write. Reset: 0800_0000h.													
_aliasHOST; IOMMUBARx00014; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI] , IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO] , 19'h0_0000}													
_aliasSMN; IOMMUMMIOx00000014; IOMMUMMIO=0240_0000h													
Bits	Description												
31:28	Reserved.												
27:24	<b>EVENT_LEN.</b> Read-write. Reset: 8h. Specifies the length of the event log in power of 2 increments. The minimum size is 256 entries (4K bytes); values less than 1000b are reserved. <b>ValidValues:</b>												
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>7h-0h</td><td>Reserved.</td></tr> <tr> <td>8h</td><td>256 entries (4K bytes).</td></tr> <tr> <td>9h</td><td>512 entries (8K bytes).</td></tr> <tr> <td>Eh-Ah</td><td>2^&lt;Value&gt; entries.</td></tr> <tr> <td>Fh</td><td>32768 entries (512K bytes).</td></tr> </table>	Value	Description	7h-0h	Reserved.	8h	256 entries (4K bytes).	9h	512 entries (8K bytes).	Eh-Ah	2^<Value> entries.	Fh	32768 entries (512K bytes).
Value	Description												
7h-0h	Reserved.												
8h	256 entries (4K bytes).												
9h	512 entries (8K bytes).												
Eh-Ah	2^<Value> entries.												
Fh	32768 entries (512K bytes).												
23:20	Reserved.												
19:0	<b>EVENT_BASE_HI.</b> Read-write. Reset: 0_0000h. Specifies address bits[51:32] of the base address of the event log. The base address programmed must be aligned to 4K bytes.												

#### IOMMUBARx00018 (IOMMUMMIO::IOMMU\_MMIO\_CNTRL\_0)

Reset: 0000_0400h.											
_aliasHOST; IOMMUBARx00018; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI] , IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO] , 19'h0_0000}											
_aliasSMN; IOMMUMMIOx00000018; IOMMUMMIO=0240_0000h											
Bits	Description										
31:30	<b>PPRQ.</b> Read-write. Reset: 0h. PPR (event) queue. <b>ValidValues:</b>										
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>Use default queue A for PPR logger with autoswap disabled.</td></tr> <tr> <td>1h</td><td>Use queue B for PPR logger with autoswap disabled.</td></tr> <tr> <td>2h</td><td>Autoswap is enabled, always start with queue A after reset NOTE: This register can be set on the fly.</td></tr> <tr> <td>3h</td><td>Reserved.</td></tr> </table>	Value	Description	0h	Use default queue A for PPR logger with autoswap disabled.	1h	Use queue B for PPR logger with autoswap disabled.	2h	Autoswap is enabled, always start with queue A after reset NOTE: This register can be set on the fly.	3h	Reserved.
Value	Description										
0h	Use default queue A for PPR logger with autoswap disabled.										
1h	Use queue B for PPR logger with autoswap disabled.										
2h	Autoswap is enabled, always start with queue A after reset NOTE: This register can be set on the fly.										
3h	Reserved.										
29	<b>GA_INT_EN.</b> Read,Write-1-to-clear. Reset: 0. 0=An interrupt is not signalled when IOMMUMMIO::IOMMU_MMIO_STATUS_0[GA_INT] changes from 0 to 1.. 1=An interrupt is signaled when IOMMUMMIO::IOMMU_MMIO_STATUS_0[GA_INT] changes from 0 to 1 using IOMMUL2::IOMMU_CAP_MISC_1[IOMMU_MSI_NUM_GA]. Guest virtual APIC interrupt enable. Writes to this bit are ignored when IOMMUMMIO::IOMMU_MMIO_EFR_0[GA_SUP] == 0.										
28	<b>GA_LOG_EN.</b> Read-write. Reset: 0. 0=Disable guest vAPIC virtual interrupt request logging. 1=Enable GA Logging. Guest virtual APIC GA Log Enabled. Writes to this but are ignored when IOMMUMMIO::IOMMU_MMIO_EFR_0{GASup} == 0.										
27:25	<b>GAM_EN.</b> Read-write. Reset: 0h. This field specifies advanced interrupt behaviors and the size of the IRTE. When set to 1, Virtual interrupt behavior as defined by the AVIC specification.										
24	<b>SMIF_LOG_EN.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. If SMI filtering is enabled, send EVENT_LOG when SMI is dropped.										

23	Reserved.
22	<b>SMIF_EN.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. Enable SMI filtering. Drop SMI from ReqID not found in the SMI filter registers.
21:18	<b>TLPT.</b> Read-write. Reset: 0h. TLPT[3:0] contains the 4-bit value matched to the PCIe® TLP Type field when the PCIe® TLP format value indicates the field carries a prefix.
17	<b>GA_EN.</b> Read-write. Reset: 0. 0=Prohibited. 1=Loose. Guest APIC enable.
16	<b>GT_EN.</b> Read-write. Reset: 0. 0=Guest translation disabled. 1=Guest translation may be enabled for a peripheral by programming DTE[GV]. NOTE: This bit must be programmed to zero when IOMMUMMIO::IOMMU_MMIO_EFR_0[GT_SUP] == 0.
15	<b>PPR_EN.</b> Read-write. Reset: 0. 0=Peripheral page service requests are treated as invalid device requests. 1=Peripheral page service requests are processed. NOTE: This bit must be programmed to zero when IOMMUMMIO::IOMMU_MMIO_EFR_0[PPR_SUP] == 0.
14	<b>PPR_INT_EN.</b> Read-write. Reset: 0. 0=An interrupt is not signaled when IOMMUMMIO::IOMMU_MMIO_STATUS_0[PPR_INT] == 1. 1=An interrupt is signaled when IOMMUMMIO::IOMMU_MMIO_STATUS_0[PPR_INT] == 1. NOTE: This bit must be programmed to zero when IOMMUMMIO::IOMMU_MMIO_EFR_0[PPR_SUP] == 0. The interrupt vector used is indicated in Capability Offset (See IOMMUMMIO::IOMMU_MMIO_EVENT_BASE_0), IOMMUL2::IOMMU_CAP_MISC[IOMMU_MSI_NUM_PPR].
13	<p><b>PPR_LOG_EN.</b> Read-write. Reset: 0. 0=Peripheral page service request logging is not enabled. Peripheral page service requests are discarded when the peripheral page service request log is not enabled or when IOMMUMMIO::IOMMU_MMIO_EFR_0[PPR_SUP] == 0. When IommuEn == 1 and software Writes PPR_LOG_EN with 1, the IOMMU clears the PPR_LOG_OVERFLOW bit and sets the PPR_RUN bit in the IOMMU Status register IOMMUMMIO::IOMMU_MMIO_STATUS_0. The IOMMU can now Write new entries to the event log if there are usable entries available. 1=The PPR Log Base Address Register IOMMUMMIO::IOMMU_MMIO_PPR_BASE_0 has been configured and peripheral page service request events are Written to the peripheral page service request log when IommuEn has also been set. Writing a 1 to this bit when PPR_LOG_EN == 1 has no effect.</p> <p><b>Description:</b> NOTE: Software can Read IOMMUMMIO::IOMMU_MMIO_STATUS_0[PPR_RUN] to determine the status of the peripheral page service request log Writing by the IOMMU. NOTE2: The peripheral page service request and event logs are independent. SOFTWARE NOTE: The PPR Log Base Address register IOMMUMMIO::IOMMU_MMIO_PPR_BASE_0, the IOMMU PPR Log Head Pointer Register IOMMUMMIO::IOMMU_MMIO_PPR_BUF_HDPTR_0, and the IOMMU PPR Log Tail Pointer Register IOMMUMMIO::IOMMU_MMIO_PPR_BUF_TAILPTR_0 must be set prior to enabling the event log.</p>
12	<p><b>CMD_BUF_EN.</b> Read-write. Reset: 0. 0=Halt command buffer processing. Writing a 0 to this bit causes the IOMMU to cease fetching new commands although commands previously fetched are completed. The IOMMU stops fetching commands upon reset and after errors as specified.</p> <p><b>Description:</b> See also IOMMUMMIO::IOMMU_MMIO_STATUS_0[CMD_BUFRUN]. NOTE: See IOMMUMMIO::IOMMU_MMIO_STATUS_0 to determine the status of command buffer processing. NOTE: Writing of event log entries is independently controlled by EventLogEn. SOFTWARE NOTE: The Command Buffer Base Address Register (See IOMMUMMIO::IOMMU_MMIO_CMD_BASE_0), the Command Buffer Head Pointer Register IOMMUMMIO::IOMMU_MMIO_CMD_BUF_HDPTR_0, and the Command Buffer Tail Pointer Register IOMMUMMIO::IOMMU_MMIO_CMD_BUF_TAILPTR_0 must be set prior to enabling the IOMMU command buffer processor.</p>
11	<b>ISOC.</b> Read-write. Reset: 0. 0=Request packet to use standard channel. 1=Request packet to use isochronous channel. This bit controls the state of the isochronous bit in the HyperTransport Read request packet when the IOMMU issues IO page table Reads and device table Reads on the HyperTransport link.
10	<b>COHERENT.</b> Read-write. Reset: 1. 0=Device table requests are not snooped by the processor. 1=Device table requests are snooped by the processor. This bit controls the state of the coherent bit in the HyperTransport Read request packet when the IOMMU issues device table Reads on the HyperTransport link.

9	<b>RES_PASS_PW</b> . Read-write. Reset: 0. 0=Response may not pass posted requests. 1=Response may pass posted requests. This bit controls the state of the ResPassPW bit in the HyperTransport Read request packet when the IOMMU issues IO page table Reads and device table Reads on the HyperTransport link.																
8	<b>PASS_PW</b> . Read-write. Reset: 0. 0=Request packet may not pass posted requests. 1=Request packet may pass posted requests. This bit controls the state of the PassPW bit in the HyperTransport Read request packet when the IOMMU issues IO page table Reads and device table Reads on the HyperTransport link.																
7:5	<b>INV_TIMEOUT</b> . Read-write. Reset: 0h. This field specifies the invalidation timeout for IOTLB invalidation requests. <b>ValidValues:</b>																
<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>No timeout.</td></tr> <tr> <td>1h</td><td>1 ms.</td></tr> <tr> <td>2h</td><td>10 ms.</td></tr> <tr> <td>3h</td><td>100 ms.</td></tr> <tr> <td>4h</td><td>1 sec.</td></tr> <tr> <td>5h</td><td>10 sec.</td></tr> <tr> <td>7h-6h</td><td>Reserved.</td></tr> </table>		Value	Description	0h	No timeout.	1h	1 ms.	2h	10 ms.	3h	100 ms.	4h	1 sec.	5h	10 sec.	7h-6h	Reserved.
Value	Description																
0h	No timeout.																
1h	1 ms.																
2h	10 ms.																
3h	100 ms.																
4h	1 sec.																
5h	10 sec.																
7h-6h	Reserved.																
4	<b>COM_WAIT_INTEN</b> . Read-write. Reset: 0. 0=No interrupt signaled. 1=An interrupt is signaled when IOMMUMMIO::IOMMU_MMIO_STATUS_0[COMWAIT_INT] == 1. Interrupt signaling.																
3	<b>EVENT_INT_EN</b> . Read-write. Reset: 0. 0=Disabled. 1=Enabled. An interrupt is signaled when the EventLogInt bit is set in the IOMMU Status Register IOMMUMMIO::IOMMU_MMIO_STATUS_0.																
2	<b>EVENT_LOG_EN</b> . Read-write. Reset: 0. 0=Disabled. 1=Enabled. <b>Description:</b> The Event Log Base Address Register IOMMUMMIO::IOMMU_MMIO_EVENT_BASE_0 has been configured and all events detected are Written to the event log when IommuEn has also been set. Writing a 1 to this bit when EventLogEn == 1 has no effect. When set to 0, event logging is not enabled. Events are discarded when the event log is not enabled. When IommuEn == 1 and software Writes EventLogEn with 1, the IOMMU clears the EventOverflow bit, and sets the EventLogRun bit in the IOMMU Status Register IOMMUMMIO::IOMMU_MMIO_STATUS_0. The IOMMU can now Write new entries to the event log if there are usable entries available. <b>NOTE:</b> Software can Read MMIO Offset IOMMUMMIO::IOMMU_MMIO_STATUS_0[EVENT_LOGRUN] to determine the status of event log Writing by the IOMMU. <b>NOTE:</b> The fetching of commands are independently controlled by CmdBufEn. <b>SOFTWARE NOTE:</b> The Event Log Base Address Register IOMMUMMIO::IOMMU_MMIO_EVENT_BASE_0, the Event Log Head Pointer Register IOMMUMMIO::IOMMU_MMIO_EVENT_BUF_HDPTR_0, and the Event Log Tail Pointer Register IOMMUMMIO::IOMMU_MMIO_EVENT_BUF_TAILPTR_0 must be set prior to enabling the event log.																
1	<b>HT_TUN_EN</b> . Read-write. Reset: 0. 0=Upstream traffic received by the HyperTransport tunnel is not translated by the IOMMU. 1=Upstream traffic received by the HyperTransport tunnel is translated by the IOMMU. The IOMMU ignores the state of this bit while IommuEn == 0. See also the HtTunnel bit in the IOMMU Capability Header IOMMUL2::IOMMU_CAP_HEADER.																
0	<b>IOMMU_EN</b> . Read-write. Reset: 0. 0=Disabled. 1=Enabled. <b>Description:</b> IOMMU enable. All upstream transactions are translated by the IOMMU. The Device Table Base Address Register IOMMUMMIO::IOMMU_MMIO_DEVTBL_BASE_0 must be configured by software before setting this bit. When set to 0, the IOMMU is disabled and no upstream transactions are translated or remapped by the IOMMU. When disabled, the IOMMU Reads no commands and creates no event log entries. <b>SOFTWARE NOTE:</b> Software must configure EventLogEn and CmdBufEn.																

**IOMMUBARx0001C (IOMMUMMIO::IOMMU\_MMIO\_CNTRL\_1)**

Read-write. Reset: 0000\_2200h.

\_aliasHOST; IOMMUBARx0001C; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI], IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx0000001C; IOMMUMMIO=0240\_0000h

Bits	Description										
31:17	Reserved.										
16	<b>V2_HD_Dis.</b> Read-write. Reset: 0. 0=Enable. 1=Disable. Disable v2 Hardware Dirty bit update.										
15:14	<b>HW_Prefetch_AD.</b> Read-write. Reset: 0h. Controls host A/D updates for HW prefetches.										
	<b>ValidValues:</b>										
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>Hardware prefetch is not allowed to update v1 Access and Dirty bits (Default).</td></tr> <tr> <td>1h</td><td>Hardware prefetch is allowed to update v1 Access bit, but not allowed to update v1 Dirty bit.</td></tr> <tr> <td>2h</td><td>Reserved. (Behaves the same as if 0).</td></tr> <tr> <td>3h</td><td>Hardware prefetch is allowed to update v1 Access and Dirty bits.</td></tr> </table>	Value	Description	0h	Hardware prefetch is not allowed to update v1 Access and Dirty bits (Default).	1h	Hardware prefetch is allowed to update v1 Access bit, but not allowed to update v1 Dirty bit.	2h	Reserved. (Behaves the same as if 0).	3h	Hardware prefetch is allowed to update v1 Access and Dirty bits.
Value	Description										
0h	Hardware prefetch is not allowed to update v1 Access and Dirty bits (Default).										
1h	Hardware prefetch is allowed to update v1 Access bit, but not allowed to update v1 Dirty bit.										
2h	Reserved. (Behaves the same as if 0).										
3h	Hardware prefetch is allowed to update v1 Access and Dirty bits.										
13	<b>EPH_EN.</b> Read-write. Reset: 1. 0=Disable. 1=Enable. Enables enhanced PRI handling support.										
12:11	Reserved.										
10	<b>PPR_Auto_resp_AON.</b> Read-write. Reset: 0. 0=Regular Auto Response behaviour, generate auto responses on buffer full or reaching the threshold based on the setting. 1=Always generate auto responses when PPR_Auto_resp_en == 1, otherwise no effect. Used to generate PPR auto responses all the time regardless of the threshold. PPR_Auto_resp_en has to be enabled in order for this to take effect.										
9	<b>Block_StopMark_En.</b> Read-write. Reset: 1. 0=Disable blocking of stop mark message. 1=Enable blocking of stop mark message. Blocking of stop mark message.										
8	<b>MARC_en.</b> Read-write. Reset: 0. 0=MARC disabled. 1=MARC enabled. Memory Address Routing and Control (MARC) feature Enable. When IOMMUMMIO::IOMMU_MMIO_EFR_1[MARCnum] == 0, this feild is reserved.										
7	<b>PPR_Auto_resp_en.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. Enable PPR auto response when PPR near overflow.										
6:5	<b>PRIV_ABORT_EN.</b> Read-write. Reset: 0h. Only effective when IOMMUMMIO::IOMMU_MMIO_EFR_1[US_SUP] == 1.										
	<b>ValidValues:</b>										
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>Blind abort is not performed, instead if US_SUP == 1, the PMR field from the request is used to check for required permission.</td></tr> <tr> <td>1h</td><td>Abort all guest accesses to pages with U/S == 0 (Supervisor level only).</td></tr> <tr> <td>2h</td><td>Future use if DTE enable bit is implemented.</td></tr> <tr> <td>3h</td><td>Reserved.</td></tr> </table>	Value	Description	0h	Blind abort is not performed, instead if US_SUP == 1, the PMR field from the request is used to check for required permission.	1h	Abort all guest accesses to pages with U/S == 0 (Supervisor level only).	2h	Future use if DTE enable bit is implemented.	3h	Reserved.
Value	Description										
0h	Blind abort is not performed, instead if US_SUP == 1, the PMR field from the request is used to check for required permission.										
1h	Abort all guest accesses to pages with U/S == 0 (Supervisor level only).										
2h	Future use if DTE enable bit is implemented.										
3h	Reserved.										
4	Reserved.										
3:2	<b>DTE_SEG_EN.</b> Read-write. Reset: 0h. DTE segment.										
	<b>ValidValues:</b>										
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>DTE is 1 segment. Only IOMMU_MMIO_DEVTBL_BASE is used.</td></tr> <tr> <td>1h</td><td>DTE is broken into 2 segments. IOMMU_MMIO_DEVTBL_BASE and IOMMU_MMIO_DEVTBL_1_BASE are used.</td></tr> <tr> <td>2h</td><td>DTE is broken into 4 segments. IOMMU_MMIO_DEVTBL_BASE and IOMMU_MMIO_DEVTBL_1/2/3_BASE are used.</td></tr> <tr> <td>3h</td><td>DTE is broken into 8 segments. All IOMMU_MMIO_DEVTBL&lt;n&gt;_BASE registers are used.</td></tr> </table>	Value	Description	0h	DTE is 1 segment. Only IOMMU_MMIO_DEVTBL_BASE is used.	1h	DTE is broken into 2 segments. IOMMU_MMIO_DEVTBL_BASE and IOMMU_MMIO_DEVTBL_1_BASE are used.	2h	DTE is broken into 4 segments. IOMMU_MMIO_DEVTBL_BASE and IOMMU_MMIO_DEVTBL_1/2/3_BASE are used.	3h	DTE is broken into 8 segments. All IOMMU_MMIO_DEVTBL<n>_BASE registers are used.
Value	Description										
0h	DTE is 1 segment. Only IOMMU_MMIO_DEVTBL_BASE is used.										
1h	DTE is broken into 2 segments. IOMMU_MMIO_DEVTBL_BASE and IOMMU_MMIO_DEVTBL_1_BASE are used.										
2h	DTE is broken into 4 segments. IOMMU_MMIO_DEVTBL_BASE and IOMMU_MMIO_DEVTBL_1/2/3_BASE are used.										
3h	DTE is broken into 8 segments. All IOMMU_MMIO_DEVTBL<n>_BASE registers are used.										
1:0	<b>EVENTQ.</b> Read-write. Reset: 0h. Event queue. NOTE: This register can be set on the fly.										
	<b>ValidValues:</b>										
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>Use default queue A for Event logger with autoswap disabled.</td></tr> <tr> <td>1h</td><td>Use queue B for Event logger with autoswap disabled.</td></tr> <tr> <td>2h</td><td>Autoswap is enabled, always start with queue A after reset.</td></tr> <tr> <td>3h</td><td>Reserved.</td></tr> </table>	Value	Description	0h	Use default queue A for Event logger with autoswap disabled.	1h	Use queue B for Event logger with autoswap disabled.	2h	Autoswap is enabled, always start with queue A after reset.	3h	Reserved.
Value	Description										
0h	Use default queue A for Event logger with autoswap disabled.										
1h	Use queue B for Event logger with autoswap disabled.										
2h	Autoswap is enabled, always start with queue A after reset.										
3h	Reserved.										



**IOMMUBARx00020 (IOMMUMMIO::IOMMU\_MMIO\_EXCL\_BASE\_0)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx00020; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI] ,  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO] , 19'h0\_0000}  
\_aliasSMN; IOMMUMMIOx00000020; IOMMUMMIO=0240\_0000h

Bits	Description
31:12	<b>EXCL_BASE_LO.</b> Read-write. Reset: 0_0000h. Specifies address bits[31:12] of the 4K byte-aligned base address of the exclusion range.
11:2	Reserved.
1	<b>EX_ALLOW.</b> Read-write. Reset: 0. 0=The EX bit in the device table entry specifies if accesses to the exclusion range are translated. 1=All accesses to the exclusion range are forwarded untranslated. Allow accesses to the exclusion range.
0	<b>EX_EN.</b> Read-write. Reset: 0. 0=The exclusion range is disabled. 1=The exclusion range is enabled. Exclusion range enable.

**IOMMUBARx00024 (IOMMUMMIO::IOMMU\_MMIO\_EXCL\_BASE\_1)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx00024; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI] ,  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO] , 19'h0\_0000}  
\_aliasSMN; IOMMUMMIOx00000024; IOMMUMMIO=0240\_0000h

Bits	Description
31:20	Reserved.
19:0	<b>EXCL_BASE_HI.</b> Read-write. Reset: 0_0000h. Specifies address bits[51:32] of the 4K byte-aligned base address of the exclusion range.

**IOMMUBARx00028 (IOMMUMMIO::IOMMU\_MMIO\_EXCL\_LIM\_0)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx00028; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI] ,  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO] , 19'h0\_0000}  
\_aliasSMN; IOMMUMMIOx00000028; IOMMUMMIO=0240\_0000h

Bits	Description
31:12	<b>EXCL_LIMIT_LO.</b> Read-write. Reset: 0_0000h. Specifies address bits[31:12] of the 4K-byte limit of the exclusion range.
11:0	Reserved.

**IOMMUBARx0002C (IOMMUMMIO::IOMMU\_MMIO\_EXCL\_LIM\_1)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx0002C; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI] ,  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO] , 19'h0\_0000}  
\_aliasSMN; IOMMUMMIOx0000002C; IOMMUMMIO=0240\_0000h

Bits	Description
31:20	Reserved.
19:0	<b>EXCL_LIMIT_HI.</b> Read-write. Reset: 0_0000h. Specifies address bits[63:32] of the 4K-byte limit of the exclusion range.

**IOMMUBARx00030 (IOMMUMMIO::IOMMU\_MMIO\_EFR\_0)**

Read-only.

\_aliasHOST; IOMMUBARx00030; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI] ,  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO] , 19'h0\_0000}  
\_aliasSMN; IOMMUMMIOx00000030; IOMMUMMIO=0240\_0000h

Bits	Description
31:30	Reserved.
29:28	<b>EVENTF.</b> Read-only. Reset: 2h. Event log dual buffer support.
	<b>ValidValues:</b>



	<b>Value</b>	<b>Description</b>
	0h	Event log dual buffer autoswap not supported.
	1h	Event log dual buffer supported without autoswap.
	2h	Event log dual buffer autoswap supported.
	3h	Reserved.
27:26	<b>GAF</b> . Read-only. Reset: Fixed,0h. GA log dual buffer support.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	GA log dual buffer autoswap not supported.
	1h	GA log dual buffer supported without autoswap.
	2h	GA log dual buffer autoswap supported.
	3h	Reserved.
25:24	<b>PPRF</b> . Read-only. Reset: 2h. PPR log dual buffer support.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	PPR log dual buffer not supported.
	1h	PPR log dual buffer supported without autoswap.
	2h	PPR log dual buffer autoswap supported.
	3h	Reserved.
23:21	<b>GAM_SUP</b> . Read-only. Reset: 1h. General AVIC modes supported.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	No advanced interrupt features supported (interrupt remapping only).
	1h	AVIC supported.
	7h-2h	Reserved.
20:18	<b>SMIF_RC</b> . Read-only. Reset: 2h. Number of SMI filter registers available.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	1 SMI filter register.
	1h	2 SMI filter registers.
	2h	4 SMI filter registers.
	3h	8 SMI filter registers.
	4h	16 SMI filter registers.
	7h-5h	Reserved.
17:16	<b>SMIF_SUP</b> . Read-only. Reset: 0h. Init: BIOS,1h. SMI filter register supported. Specifies that SMI interrupts may be filtered.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	SMI interrupts are always passed through.
	1h	SMI interrupts are filtered under the control of [SMIFEN] and the contents of the SMI filter registers.
	3h-2h	Reserved.
15:14	<b>GLX_SUP</b> . Read-only. Reset: 1h. Single-level GCR3 base table address translation is supported. When GLX_SUP == 0, GLX in the DTE is ignored. The value of GLXSup is not meaningful when GTSup == 0.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	Single-level guest CR3 base table address translation is support. When GLXSup == 0, the value of GLX in the DTE is ignored and the IOMMU performs only single-level Guest CR3 lookups.
	1h	Two-level GCR3 base address table is supported in hardware.

	2h	Three-level GCR3 base address table is supported in hardware for 20-bit PASID values in the future.
	3h	Reserved.
13:12	<b>GATS.</b> Read-only. Reset: Fixed,0h. The maximum number of translation levels supported for guest address translation GVA. This value is meaningful when IOMMUMMIO::IOMMU_MMIO_EFR_0[GT_SUP] == 0.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	4 levels (PML4E).
	1h	5 levels.
	2h	6 levels.
	3h	Reserved.
11:10	<b>HATS.</b> Read-only. Reset: 2h. The maximum number of host address translation levels supported. This value is not meaningful when IOMMUMMIO::IOMMU_MMIO_EFR_0[GT_SUP] == 0. IOMMU behavior is undefined if Next Level in a page directory entry exceeds the limit set by HATS.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	4 levels.
	1h	5 levels.
	2h	6 levels.
	3h	Reserved.
9	<b>PC_SUP.</b> Read-only. Reset: 1. 0=No performance counters are supported. 1=Performance counters are supported. Performance counter support.	
8	<b>HE_SUP.</b> Read-only. Reset: 0. 0=Hardware error registers do not report error information. 1=Error information is reported in hardware error registers. Hardware error register support.	
7	<b>GA_SUP.</b> Read-only. Reset: 1. 0=Guest APIC supported. 1=Guest APIC not supported. Guest APIC support.	
6	<b>IA_SUP.</b> Read-only. Reset: 1. 0=The INVALIDATE_IOMMU_ALL command is not supported and will generate an error when used. 1=The INVALIDATE_IOMMU_ALL command is supported. INVALIDATE_IOMMU_ALL command support.	
5	Reserved.	
4	<b>GT_SUP.</b> Read-only. Reset: 1. 0=Only nested address translation is supported. 1=Guest address translation is supported. Guest address translation support. When GT_SUP == 0, the following values in the DTE must be zero: GV, GLX and GCR3 Table Root Pointer. See also IOMMUMMIO::IOMMU_MMIO_CNTRL_0[GT_EN].	
3	<b>NX_SUP.</b> Read-only. Reset: 1. 0=No-execute protection is not supported. 1=No-execute protection is supported. 1=No-execute protection support.	
2	<b>XT_SUP.</b> Read-only. Reset: 0. 0=x2APIC support is disabled. 1=The interrupt remapping table is expanded to support x2APIC interrupt information. x2APIC interrupt support.	
1	<b>PPR_SUP.</b> Read-only. Reset: 1. 0=Peripheral page service requests are not supported, the paage service request queue is not supported and the PPR interrupt is not generated by the IOMMU. 1=Indicates that IOMMU handles page service request events from peripherals, the IOMMU supports the page service request queue, and that the second IOMMU interrupt can be used to signal peripheral page service request events. Peripheral page service request support.	
0	<b>PREF_SUP.</b> Read-only. Reset: 0. 0=IOMMU treats PREFETCH_IOMMU_PAGES commands as illegal commands. 1=Indicates that IOMMU accepts PREFETCH_IOMMU_PAGES commands. IOMMU PREFETCH_IOMMU_PAGES command acceptance.	

#### IOMMUBARx00034 (IOMMUMMIO::IOMMU\_MMIO\_EFR\_1)

Read-only. Reset: 000F\_77EFh.

\_aliasHOST; IOMMUBARx00034; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI], IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}  
\_aliasSMN; IOMMUMMIOx00000034; IOMMUMMIO=0240\_0000h

Bits	Description
31:23	Reserved.

22	<b>InvIotlbTypeSup.</b> Read-only. Reset: 0. 0=Hardware does not support invalidate IOTLB type. 1=Hardware supports invalidate IOTLB type. Invalidate IOTLB type support.										
21	<b>V2_HD_DIS_SUP.</b> Read-only. Reset: 0. 0=Hardware does not support disabling v2 Dirty bit updates. 1=Hardware supports disabling v2 Dirty bit updates. Support for disabling v2 Dirty bit updates by hardware.										
20	<b>HD_SUP.</b> Read-only. Reset: 0. 0= Hardware does not support Host Dirty bit update support. 1=Hardware supports Host Dirty bit update support for v1 Page tables. Support of Host Dirty bit update support for v1 Page tables.										
19	<b>ATTRFW_SUP.</b> Read-only. Reset: 1. 0=Hardware does not support forward page table memory. 1=Hardware supports forward page table memory attribute to ATC client. Support of forward page table memory attribute to ATC client.										
18	<b>EPH_SUP.</b> Read-only. Reset: 1. 0=Hardware does not support Enhanced PPR Handling. 1=Hardware supports Enhanced PPR Handling. Support of Enhanced PPR Handling.										
17	<b>HA_SUP.</b> Read-only. Reset: 1. 0=Hardware does not support Host Access bit update. 1=Hardware supports Host Access bit update support for v1 Page tables. Support of Host Access bit update support for v1 Page tables.										
16	<b>GIO_SUP.</b> Read-only. Reset: 1. 0=Hardware does not support nested IO protection. 1=Hardware supports nested IO protection. Support of nested IO protection.										
15	<b>SNOOP_ATTRS_SUP.</b> Read-only. Reset: 0. 0=Hardware does not support returning memory attributes of the guest page. 1=Hardware supports returning memory attributes of the guest page to ATS requester along with ATS completion. Support for returning memory attributes of the guest page to ATS requester along with ATS completion.										
14	<b>MMIO_MSI_CAP_SUP.</b> Read-only. Reset: 1. 0=IOMMU does not advertise MSI Capability registers. 1=IOMMU advertises that MSI Capability registers are accessible through IOMMU MMIO Space. IOMMU advertisement of MSI Capability registers accessible through IOMMU MMIO Space.										
13	<b>GMC_IOMMU_BYPASS_SUP.</b> Read-only. Reset: 1. 0=GMC IOMMU bypass not supported. 1=IOMMU GMC bypass supported. IOMMU Graphics Memory Controller (GMC) bypass support.										
12	<b>BLOCK_STOPMARK_SUP.</b> Read-only. Reset: 1. 0=No blocking Stop Mark Messages supported. 1=Block Stop Mark Messages supported. Blocking Stop Mark Messages support.										
11:10	<b>MARCnum.</b> Read-only. Reset: 1h. Specifies the number of MARC apertures and thus the number of MARC register 3-tuples supported. <b>ValidValues:</b>										
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>MARC feature not supported.</td></tr> <tr> <td>1h</td><td>4 MARC register 3-tuples supported.</td></tr> <tr> <td>2h</td><td>8 MARC register 4-tuples are supported.</td></tr> <tr> <td>3h</td><td>Reserved.</td></tr> </table>	Value	Description	0h	MARC feature not supported.	1h	4 MARC register 3-tuples supported.	2h	8 MARC register 4-tuples are supported.	3h	Reserved.
Value	Description										
0h	MARC feature not supported.										
1h	4 MARC register 3-tuples supported.										
2h	8 MARC register 4-tuples are supported.										
3h	Reserved.										
9	<b>PPR_AUTORESP_SUP.</b> Read-only. Reset: 1. 0=No hardware support for PPR auto response generation on log. 1=Hardware supports PPR auto response generation on log about to be full or generates auto response on reaching threshold if enabled with OVERFLOW_EARLY. Hardware support of PPR auto response generation.										
8	<b>PPR_OVERFLOW_EARLY_SUP.</b> Read-only. Reset: 1. 0=Hardware does not support PPR_OVERFLOW_EARLY notification on threshold. 1=Hardware supports PPR_OVERFLOW_EARLY notification on threshold. PPR_OVERFLOW_EARLY notification on threshold support.										
7:6	<b>DTE_seg.</b> Read-only. Reset: 3h. DTE segmentation support. <b>ValidValues:</b>										
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>Hardware supports no DTE segmentation.</td></tr> <tr> <td>1h</td><td>Hardware supports 2-way DTE segmentation.</td></tr> <tr> <td>2h</td><td>Hardware supports 4-way DTE segmentation.</td></tr> <tr> <td>3h</td><td>Hardware supports 8-way DTE segmentation.</td></tr> </table>	Value	Description	0h	Hardware supports no DTE segmentation.	1h	Hardware supports 2-way DTE segmentation.	2h	Hardware supports 4-way DTE segmentation.	3h	Hardware supports 8-way DTE segmentation.
Value	Description										
0h	Hardware supports no DTE segmentation.										
1h	Hardware supports 2-way DTE segmentation.										
2h	Hardware supports 4-way DTE segmentation.										
3h	Hardware supports 8-way DTE segmentation.										
5	<b>US_SUP.</b> Read-only. Reset: 1. 0=Privileged page protection is not supported. 1=Privileged page protection is supported. Privileged page protection support.										

4	Reserved.														
3:0	<b>PAS_MAX</b> . Read-only. Reset: Fh. The maximum PASID value supported is calculated as $2^{\text{PAS\_MAX}} - 1$ . This value is not meaningful when <code>IOMMUMMIO::IOMMU_MMIO_EFR_0[GT_SUP] == 0</code> . <b>ValidValues:</b>														
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>2h-0h</td><td>Reserved.</td></tr> <tr> <td>3h</td><td>4-bit PASIDs (0-15).</td></tr> <tr> <td>4h</td><td>5-bit PASIDs.</td></tr> <tr> <td>Dh-5h</td><td>(&lt;Value&gt; + 1)-bit PASIDs.</td></tr> <tr> <td>Eh</td><td>15-bit PASIDs.</td></tr> <tr> <td>Fh</td><td>16-bit PASIDs.</td></tr> </table>	Value	Description	2h-0h	Reserved.	3h	4-bit PASIDs (0-15).	4h	5-bit PASIDs.	Dh-5h	(<Value> + 1)-bit PASIDs.	Eh	15-bit PASIDs.	Fh	16-bit PASIDs.
Value	Description														
2h-0h	Reserved.														
3h	4-bit PASIDs (0-15).														
4h	5-bit PASIDs.														
Dh-5h	(<Value> + 1)-bit PASIDs.														
Eh	15-bit PASIDs.														
Fh	16-bit PASIDs.														

**IOMMUBARx00038 (IOMMUMMIO::IOMMU\_MMIO\_PPR\_BASE\_0)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx00038; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx00000038; IOMMUMMIO=0240\_0000h

Bits	Description
31:12	<b>PPR_BASE_LO</b> . Read-write. Reset: 0_0000h. Specifies bits[31:12] of the base address of the PPR log. The base address programmed must be aligned to 4K bytes.
11:0	Reserved.

**IOMMUBARx0003C (IOMMUMMIO::IOMMU\_MMIO\_PPR\_BASE\_1)**

Read-write. Reset: 0800\_0000h.

\_aliasHOST; IOMMUBARx0003C; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx0000003C; IOMMUMMIO=0240\_0000h

Bits	Description												
31:28	Reserved.												
27:24	<b>PPR_LEN</b> . Read-write. Reset: 8h. Specifies the length of the PPR log in power of two increments. The minimum size is 256 entries (4K bytes). Values less than 8 are reserved. <b>ValidValues:</b>												
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>7h-0h</td><td>Reserved.</td></tr> <tr> <td>8h</td><td>256 entries (4K bytes).</td></tr> <tr> <td>9h</td><td>512 entries (8K bytes).</td></tr> <tr> <td>Eh-Ah</td><td><math>2^{\text{&lt;Value&gt;}}</math> entries.</td></tr> <tr> <td>Fh</td><td>32768 entries (512K bytes).</td></tr> </table>	Value	Description	7h-0h	Reserved.	8h	256 entries (4K bytes).	9h	512 entries (8K bytes).	Eh-Ah	$2^{\text{<Value>}}$ entries.	Fh	32768 entries (512K bytes).
Value	Description												
7h-0h	Reserved.												
8h	256 entries (4K bytes).												
9h	512 entries (8K bytes).												
Eh-Ah	$2^{\text{<Value>}}$ entries.												
Fh	32768 entries (512K bytes).												
23:20	Reserved.												
19:0	<b>PPR_BASE_HI</b> . Read-write. Reset: 0_0000h. Specifies bits[51:32] of the base address of the PPR log. The base address programmed must be aligned to 4K bytes.												

**IOMMUBARx00040 (IOMMUMMIO::IOMMU\_MMIO\_HW\_ERR\_UPPER\_0)**

Read-only. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx00040; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx00000040; IOMMUMMIO=0240\_0000h

Bits	Description
31:0	<b>FIRST_EV_CODE_LO</b> . Read-only. Reset: 0000_0000h. Specifies bits[31:0] of the upper 64 bits of the most recent hardware error detected by the IOMMU.

**IOMMUBARx00044 (IOMMUMMIO::IOMMU\_MMIO\_HW\_ERR\_UPPER\_1)**

Read-only. Reset: 0000\_0000h.

_aliasHOST; IOMMUBARx00044; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI] , IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO] , 19'h0_0000}	
_aliasSMN; IOMMUMMIOx00000044; IOMMUMMIO=0240_0000h	
Bits	Description
31:28	<b>EV_CODE</b> . Read-only. Reset: 0h. Event code for the type of error logged.
27:0	<b>FIRST_EV_CODE_HI</b> . Read-only. Reset: 000_0000h. Specifies bits[59:32] of the upper 64 bits of the most recent hardware error detected by the IOMMU.

**IOMMUBARx00048 (IOMMUMMIO::IOMMU\_MMIO\_HW\_ERR\_LOWER\_0)**

Read-only. Reset: 0000_0000h.	
_aliasHOST; IOMMUBARx00048; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI] , IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO] , 19'h0_0000}	
_aliasSMN; IOMMUMMIOx00000048; IOMMUMMIO=0240_0000h	
Bits	Description
31:0	<b>SECOND_EV_CODE_LO</b> . Read-only. Reset: 0000_0000h. Specifies bits[31:0] of the lower 64 bits of the most recent hardware error detected by the IOMMU.

**IOMMUBARx0004C (IOMMUMMIO::IOMMU\_MMIO\_HW\_ERR\_LOWER\_1)**

Read-only. Reset: 0000_0000h.	
_aliasHOST; IOMMUBARx0004C; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI] , IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO] , 19'h0_0000}	
_aliasSMN; IOMMUMMIOx0000004C; IOMMUMMIO=0240_0000h	
Bits	Description
31:0	<b>SECOND_EV_CODE_HI</b> . Read-only. Reset: 0000_0000h. Specifies bits[63:32] of the lower 64 bits of the most recent hardware error detected by the IOMMU.

**IOMMUBARx00050 (IOMMUMMIO::IOMMU\_MMIO\_HW\_ERR\_STATUS\_0)**

Read,Write-1-to-clear. Reset: 0000_0000h.	
_aliasHOST; IOMMUBARx00050; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI] , IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO] , 19'h0_0000}	
_aliasSMN; IOMMUMMIOx00000050; IOMMUMMIO=0240_0000h	
Bits	Description
31:2	Reserved.
1	<b>HEO</b> . Read,Write-1-to-clear. Reset: 0. 0=Not overwritten. 1=Contents overwritten by new information. Defines the contents of the IOMMU hardware error registers as having been overwritten. HEO is not meaningful when HEV == 0.
0	<b>HEV</b> . Read,Write-1-to-clear. Reset: 0. 0=Not valid. 1=Contents are valid. Defines the contents of the IOMMU hardware error registers as valid.

**IOMMUBARx00054 (IOMMUMMIO::IOMMU\_MMIO\_HW\_ERR\_STATUS\_1)**

Reset: 0000_0000h.	
_aliasHOST; IOMMUBARx00054; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI] , IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO] , 19'h0_0000}	
_aliasSMN; IOMMUMMIOx00000054; IOMMUMMIO=0240_0000h	
Bits	Description
31:0	Reserved.

**IOMMUBARx00060 (IOMMUMMIO::SMI\_FILTER\_REGISTER\_0\_0)**

Reset: 0000_0000h.	
_aliasHOST; IOMMUBARx00060; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI] , IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO] , 19'h0_0000}	
_aliasSMN; IOMMUMMIOx00000060; IOMMUMMIO=0240_0000h	
Bits	Description
31:18	Reserved.
17	<b>SmiFLock_0</b> . Write-once. Reset: 0. 0=Unlocked. 1=Locked. Makes SmiDV and SmiDID Read-only. Can only be set to 0 on reset.

16	<b>SmiDV_0.</b> Read-write. Reset: 0. 0=Not Valid. 1=SMI filter is enabled. Valid. This SMI filter is enabled and the Device ID specified in SmiDID is valid for the SMI.
15:0	<b>SmiDID_0.</b> Read-write. Reset: 0000h. Specifies the Device ID for the SMIs which are forwarded upstream.

**IOMMUBARx00064 (IOMMUMMIO::SMI\_FILTER\_REGISTER\_0\_1)**

Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx00064; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}  
\_aliasSMN; IOMMUMMIOx00000064; IOMMUMMIO=0240\_0000h

Bits	Description
------	-------------

31:0	Reserved.
------	-----------

**IOMMUBARx00068 (IOMMUMMIO::SMI\_FILTER\_REGISTER\_1\_0)**

Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx00068; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}  
\_aliasSMN; IOMMUMMIOx00000068; IOMMUMMIO=0240\_0000h

Bits	Description
------	-------------

31:18	Reserved.
-------	-----------

17	<b>SmiFLock_1.</b> Write-once. Reset: 0. 0=Unlocked. 1=Locked. Makes SmiDV and SmiDID Read-only. Can only be set to 0 on reset.
----	---

16	<b>SmiDV_1.</b> Read-write. Reset: 0. 0=Not valid. 1=SMI filter is enabled. Valid. This SMI filter is enabled and the Device ID specified in SmiDID is valid for the SMI.
----	---

15:0	<b>SmiDID_1.</b> Read-write. Reset: 0000h. Specifies the Device ID for the SMIs which are forwarded upstream.
------	---

**IOMMUBARx0006C (IOMMUMMIO::SMI\_FILTER\_REGISTER\_1\_1)**

Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx0006C; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}  
\_aliasSMN; IOMMUMMIOx0000006C; IOMMUMMIO=0240\_0000h

Bits	Description
------	-------------

31:0	Reserved.
------	-----------

**IOMMUBARx00070 (IOMMUMMIO::SMI\_FILTER\_REGISTER\_2\_0)**

Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx00070; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}  
\_aliasSMN; IOMMUMMIOx00000070; IOMMUMMIO=0240\_0000h

Bits	Description
------	-------------

31:18	Reserved.
-------	-----------

17	<b>SmiFLock_2.</b> Write-once. Reset: 0. 0=Unlocked. 1=Locked. Makes SmiDV and SmiDID Read-only. Can only be set to 0 on reset.
----	---

16	<b>SmiDV_2.</b> Read-write. Reset: 0. 0=Not valid. 1=SMI filter is enabled. Valid. This SMI filter is enabled and the Device ID specified in SmiDID is valid for the SMI.
----	---

15:0	<b>SmiDID_2.</b> Read-write. Reset: 0000h. Specifies the Device ID for the SMIs which are forwarded upstream.
------	---

**IOMMUBARx00074 (IOMMUMMIO::SMI\_FILTER\_REGISTER\_2\_1)**

Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx00074; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}  
\_aliasSMN; IOMMUMMIOx00000074; IOMMUMMIO=0240\_0000h

Bits	Description
------	-------------

31:0	Reserved.
------	-----------

**IOMMUBARx00078 (IOMMUMMIO::SMI\_FILTER\_REGISTER\_3\_0)**



Reset: 0000_0000h.	
_aliasHOST; IOMMUBARx00078; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI], IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO], 19'h0_0000}	
_aliasSMN; IOMMUMMIOx00000078; IOMMUMMIO=0240_0000h	
Bits	Description
31:18	Reserved.
17	<b>SmiFLock_3.</b> Write-once. Reset: 0. 0=Unlocked. 1=Locked. Makes SmiDV and SmiDID Read-only. Can only be set to 0 on reset.
16	<b>SmiDV_3.</b> Read-write. Reset: 0. 0=Not valid. 1=SMI filter is enabled. Valid. This SMI filter is enabled and the Device ID specified in SmiDID is valid for SMI.
15:0	<b>SmiDID_3.</b> Read-write. Reset: 0000h. Specifies the Device ID for the SMIs which are forwarded upstream.

**IOMMUBARx0007C (IOMMUMMIO::SMI\_FILTER\_REGISTER\_3\_1)**

Reset: 0000_0000h.	
_aliasHOST; IOMMUBARx0007C; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI], IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO], 19'h0_0000}	
_aliasSMN; IOMMUMMIOx0000007C; IOMMUMMIO=0240_0000h	
Bits	Description
31:0	Reserved.

**IOMMUBARx000E0 (IOMMUMMIO::IOMMU\_MMIO\_GA\_LOG\_BASE\_0)**

Read-write. Reset: 0000_0000h.	
_aliasHOST; IOMMUBARx000E0; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI], IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO], 19'h0_0000}	
_aliasSMN; IOMMUMMIOx000000E0; IOMMUMMIO=0240_0000h	
Bits	Description
31:12	<b>GA_LOG_BASE_LO.</b> Read-write. Reset: 0_0000h. Guest virtual interrupt log base addresss. Specifies bits[31:12] of the base address of the guest virtual APIC log. The base address must be aligned to 4K bytes.
11:0	Reserved.

**IOMMUBARx000E4 (IOMMUMMIO::IOMMU\_MMIO\_GA\_LOG\_BASE\_1)**

Read-write. Reset: 0800_0000h.													
_aliasHOST; IOMMUBARx000E4; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI], IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO], 19'h0_0000}													
_aliasSMN; IOMMUMMIOx000000E4; IOMMUMMIO=0240_0000h													
Bits	Description												
31:28	Reserved.												
27:24	<b>GA_LOG_LEN.</b> Read-write. Reset: 8h. Guest virtual APIC log length. Specifies the length of the guest virtual APIC log in power of 2 increments. The minimum size is 512 entries (4K bytes). Values less than 8 are Reserved. <b>ValidValues:</b>												
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>7h-0h</td><td>Reserved.</td></tr> <tr> <td>8h</td><td>256 entries (4K bytes).</td></tr> <tr> <td>9h</td><td>512 entries (8K bytes).</td></tr> <tr> <td>Eh-Ah</td><td>2^&lt;Value&gt; entries.</td></tr> <tr> <td>Fh</td><td>32768 entries (512K bytes).</td></tr> </table>	Value	Description	7h-0h	Reserved.	8h	256 entries (4K bytes).	9h	512 entries (8K bytes).	Eh-Ah	2^<Value> entries.	Fh	32768 entries (512K bytes).
Value	Description												
7h-0h	Reserved.												
8h	256 entries (4K bytes).												
9h	512 entries (8K bytes).												
Eh-Ah	2^<Value> entries.												
Fh	32768 entries (512K bytes).												
23:20	Reserved.												
19:0	<b>GA_LOG_BASE_HI.</b> Read-write. Reset: 0_0000h. Guest virtual interrupt log base addresss. Specifies bits[51:32] of the base address of the guest virtual APIC log. The base address must be aligned to 4K bytes.												

**IOMMUBARx000E8 (IOMMUMMIO::IOMMU\_MMIO\_GA\_LOG\_TAILPTR\_ADDR\_0)**

Read-write. Reset: 0000_0000h.	
_aliasHOST; IOMMUBARx000E8; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI], IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO], 19'h0_0000}	



_aliasSMN; IOMMUMMIOx000000E8; IOMMUMMIO=0240_0000h	
Bits	Description
31:3	<b>GA_LOG_TAILPTR_ADDR_LO.</b> Read-write. Reset: 0000_0000h. Guest virtual APIC log tail address[31:3]. Specifies the SPA of the memory location containing the tail pointer of the guest virtual APIC log. The address must be aligned to an 8-byte boundary. When GATAddr is 0, the memory location is not updated and the software must Read the tail pointer from the Guest Virtual APIC Log Tail Pointer Register.
2:0	Reserved.

#### IOMMUBARx000EC (IOMMUMMIO::IOMMU\_MMIO\_GA\_LOG\_TAILPTR\_ADDR\_1)

Read-write. Reset: 0000_0000h.	
_aliasHOST; IOMMUBARx000EC; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI], IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO], 19'h0_0000}	
_aliasSMN; IOMMUMMIOx000000EC; IOMMUMMIO=0240_0000h	
Bits	Description
31:20	Reserved.
19:0	<b>GA_LOG_TAILPTR_ADDR_HI.</b> Read-write. Reset: 0_0000h. Guest virtual APIC log tail address[51:32]. Specifies the SPA of the memory location containing the tail pointer of the guest virtual APIC log. The address must be aligned to an 8-byte boundary. When GATAddr is 0, the memory location is not updated and the software must Read the tail pointer from the Guest Virtual APIC Log Tail Pointer Register.

#### IOMMUBARx000F0 (IOMMUMMIO::IOMMU\_MMIO\_PPR\_B\_BASE\_0)

Read-write. Reset: 0000_0000h.	
_aliasHOST; IOMMUBARx000F0; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI], IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO], 19'h0_0000}	
_aliasSMN; IOMMUMMIOx000000F0; IOMMUMMIO=0240_0000h	
Bits	Description
31:12	<b>PPR_B_BASE_LO.</b> Read-write. Reset: 0_0000h. Specifies bits[31:12] of the base address of the PPR log B. The base address programmed must be aligned to 4K bytes.
11:0	Reserved.

#### IOMMUBARx000F4 (IOMMUMMIO::IOMMU\_MMIO\_PPR\_B\_BASE\_1)

Read-write. Reset: 0800_0000h.													
_aliasHOST; IOMMUBARx000F4; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI], IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO], 19'h0_0000}													
_aliasSMN; IOMMUMMIOx000000F4; IOMMUMMIO=0240_0000h													
Bits	Description												
31:28	Reserved.												
27:24	<b>PPR_B_LEN.</b> Read-write. Reset: 8h. Specifies the length of the PPR log B in power of two increments. The minimum size is 256 entries (4K bytes). Values less than 8 are reserved.												
<b>ValidValues:</b>													
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>7h-0h</td><td>Reserved.</td></tr> <tr> <td>8h</td><td>256 entries (4K bytes).</td></tr> <tr> <td>9h</td><td>512 entries (8K bytes).</td></tr> <tr> <td>Eh-Ah</td><td>2^&lt;Value&gt; entries.</td></tr> <tr> <td>Fh</td><td>32768 entries (512K bytes).</td></tr> </table>	Value	Description	7h-0h	Reserved.	8h	256 entries (4K bytes).	9h	512 entries (8K bytes).	Eh-Ah	2^<Value> entries.	Fh	32768 entries (512K bytes).
Value	Description												
7h-0h	Reserved.												
8h	256 entries (4K bytes).												
9h	512 entries (8K bytes).												
Eh-Ah	2^<Value> entries.												
Fh	32768 entries (512K bytes).												
23:20	Reserved.												
19:0	<b>PPR_B_BASE_HI.</b> Read-write. Reset: 0_0000h. Specifies bits[51:32] of the base address of the PPR log B. The base address programmed must be aligned to 4K bytes.												

#### IOMMUBARx000F8 (IOMMUMMIO::IOMMU\_MMIO\_EVENT\_B\_BASE\_0)

Read-write. Reset: 0000_0000h.	
_aliasHOST; IOMMUBARx000F8; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI], IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO], 19'h0_0000}	

_aliasSMN; IOMMUMMIOx000000F8; IOMMUMMIO=0240_0000h	
Bits	Description
31:12	<b>EVENT_B_BASE_LO.</b> Read-write. Reset: 0_0000h. Specifies address bits[31:12] of the base address of the event log B. The base address programmed must be aligned to 4K bytes.
11:0	Reserved.

#### IOMMUBARx000FC (IOMMUMMIO::IOMMU\_MMIO\_EVENT\_B\_BASE\_1)

Read-write. Reset: 0800\_0000h.

\_aliasHOST; IOMMUBARx000FC; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI] , IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO] , 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx000000FC; IOMMUMMIO=0240\_0000h

Bits	Description												
31:28	Reserved.												
27:24	<b>EVENT_B_LEN.</b> Read-write. Reset: 8h. Specifies the length of the event log B in power of 2 increments. The minimum size is 256 entries (4K bytes). Values less than 8 are reserved. <b>ValidValues:</b>												
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>7h-0h</td><td>Reserved.</td></tr> <tr> <td>8h</td><td>256 entries (4K bytes).</td></tr> <tr> <td>9h</td><td>512 entries (8K bytes).</td></tr> <tr> <td>Eh-Ah</td><td>2^&lt;Value&gt; entries.</td></tr> <tr> <td>Fh</td><td>32768 entries (512K bytes).</td></tr> </table>	Value	Description	7h-0h	Reserved.	8h	256 entries (4K bytes).	9h	512 entries (8K bytes).	Eh-Ah	2^<Value> entries.	Fh	32768 entries (512K bytes).
Value	Description												
7h-0h	Reserved.												
8h	256 entries (4K bytes).												
9h	512 entries (8K bytes).												
Eh-Ah	2^<Value> entries.												
Fh	32768 entries (512K bytes).												
23:20	Reserved.												
19:0	<b>EVENT_B_BASE_HI.</b> Read-write. Reset: 0_0000h. Specifies address bits[51:32] of the base address of the event log B. The base address programmed must be aligned to 4K bytes.												

#### IOMMUBARx00100 (IOMMUMMIO::IOMMU\_MMIO\_DEVTBL\_1\_BASE\_0)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx00100; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI] , IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO] , 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx00000100; IOMMUMMIO=0240\_0000h

Bits	Description
31:12	<b>DEV_TBL_1_BASE_LO.</b> Read-write. Reset: 0_0000h. Specifies address bits[31:12] of the 4K byte-aligned base address of the specific segment of the first level device table.
11:9	Reserved.
8:0	<b>DEV_TBL_1_SIZE.</b> Read-write. Reset: 000h. This field contains 1 less than the length of the device table, in multiples of 4K bytes. A minimum size of 0 corresponds to a 4K-byte device table and a maximum size of 1FFh corresponds to a 2M-byte device table.

#### IOMMUBARx00104 (IOMMUMMIO::IOMMU\_MMIO\_DEVTBL\_1\_BASE\_1)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx00104; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI] , IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO] , 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx00000104; IOMMUMMIO=0240\_0000h

Bits	Description
31:20	Reserved.
19:0	<b>DEV_TBL_1_BASE_HI.</b> Read-write. Reset: 0_0000h. Specifies address bits[51:32] of the 4K byte-aligned base address of the first level device table.

#### IOMMUBARx00108 (IOMMUMMIO::IOMMU\_MMIO\_DEVTBL\_2\_BASE\_0)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx00108; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI] , IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO] , 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx00000108; IOMMUMMIO=0240\_0000h

Bits	Description
31:12	<b>DEV_TBL_2_BASE_LO.</b> Read-write. Reset: 0_0000h. Specifies address bits[31:12] of the 4K byte-aligned base address of the specific segment of the first level device table.
11:9	Reserved.
8:0	<b>DEV_TBL_2_SIZE.</b> Read-write. Reset: 000h. This field contains 1 less than the length of the device table, in multiples of 4K bytes. A minimum size of 0 corresponds to a 4K-byte device table and a maximum size of 1FFh corresponds to a 2M-byte device table.

#### **IOMMUBARx0010C (IOMMUMMIO::IOMMU\_MMIO\_DEVTBL\_2\_BASE\_1)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx0010C; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI], IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx0000010C; IOMMUMMIO=0240\_0000h

Bits	Description
31:20	Reserved.
19:0	<b>DEV_TBL_2_BASE_HI.</b> Read-write. Reset: 0_0000h. Specifies address bits[51:32] of the 4K byte-aligned base address of the first level device table.

#### **IOMMUBARx00110 (IOMMUMMIO::IOMMU\_MMIO\_DEVTBL\_3\_BASE\_0)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx00110; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI], IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx00000110; IOMMUMMIO=0240\_0000h

Bits	Description
31:12	<b>DEV_TBL_3_BASE_LO.</b> Read-write. Reset: 0_0000h. Specifies address bits[31:12] of the 4K byte-aligned base address of the specific segment of the first level device table.
11:9	Reserved.
8:0	<b>DEV_TBL_3_SIZE.</b> Read-write. Reset: 000h. This field contains 1 less than the length of the device table, in multiples of 4K bytes. A minimum size of 0 corresponds to a 4K-byte device table and a maximum size of 1FFh corresponds to a 2M-byte device table.

#### **IOMMUBARx00114 (IOMMUMMIO::IOMMU\_MMIO\_DEVTBL\_3\_BASE\_1)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx00114; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI], IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx00000114; IOMMUMMIO=0240\_0000h

Bits	Description
31:20	Reserved.
19:0	<b>DEV_TBL_3_BASE_HI.</b> Read-write. Reset: 0_0000h. Specifies address bits[51:32] of the 4K byte-aligned base address of the first level device table.

#### **IOMMUBARx00118 (IOMMUMMIO::IOMMU\_MMIO\_DEVTBL\_4\_BASE\_0)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx00118; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI], IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx00000118; IOMMUMMIO=0240\_0000h

Bits	Description
31:12	<b>DEV_TBL_4_BASE_LO.</b> Read-write. Reset: 0_0000h. Specifies address bits[31:12] of the 4K byte-aligned base address of the specific segment of the first level device table.
11:9	Reserved.
8:0	<b>DEV_TBL_4_SIZE.</b> Read-write. Reset: 000h. This field contains 1 less than the length of the device table, in multiples of 4K bytes. A minimum size of 0 corresponds to a 4K-byte device table and a maximum size of 1FFh corresponds to a 2M-byte device table.

**IOMMUBARx0011C (IOMMUMMIO::IOMMU\_MMIO\_DEVTBL\_4\_BASE\_1)**

Read-write. Reset: 0000_0000h.	
_aliasHOST; IOMMUBARx0011C; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI], IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO], 19'h0_0000}	
_aliasSMN; IOMMUMMIOx0000011C; IOMMUMMIO=0240_0000h	
Bits	Description
31:20	Reserved.
19:0	<b>DEV_TBL_4_BASE_HI.</b> Read-write. Reset: 0_0000h. Specifies address bits[51:32] of the 4K byte-aligned base address of the first level device table.

**IOMMUBARx00120 (IOMMUMMIO::IOMMU\_MMIO\_DEVTBL\_5\_BASE\_0)**

Read-write. Reset: 0000_0000h.	
_aliasHOST; IOMMUBARx00120; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI], IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO], 19'h0_0000}	
_aliasSMN; IOMMUMMIOx00000120; IOMMUMMIO=0240_0000h	
Bits	Description
31:12	<b>DEV_TBL_5_BASE_LO.</b> Read-write. Reset: 0_0000h. Specifies address bits[31:12] of the 4K byte-aligned base address of the specific segment of the first level device table.
11:9	Reserved.
8:0	<b>DEV_TBL_5_SIZE.</b> Read-write. Reset: 000h. This field contains 1 less than the length of the device table, in multiples of 4K bytes. A minimum size of 0 corresponds to a 4K-byte device table and a maximum size of 1FFh corresponds to a 2M-byte device table.

**IOMMUBARx00124 (IOMMUMMIO::IOMMU\_MMIO\_DEVTBL\_5\_BASE\_1)**

Read-write. Reset: 0000_0000h.	
_aliasHOST; IOMMUBARx00124; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI], IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO], 19'h0_0000}	
_aliasSMN; IOMMUMMIOx00000124; IOMMUMMIO=0240_0000h	
Bits	Description
31:20	Reserved.
19:0	<b>DEV_TBL_5_BASE_HI.</b> Read-write. Reset: 0_0000h. Specifies address bits[51:32] of the 4K byte-aligned base address of the first level device table.

**IOMMUBARx00128 (IOMMUMMIO::IOMMU\_MMIO\_DEVTBL\_6\_BASE\_0)**

Read-write. Reset: 0000_0000h.	
_aliasHOST; IOMMUBARx00128; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI], IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO], 19'h0_0000}	
_aliasSMN; IOMMUMMIOx00000128; IOMMUMMIO=0240_0000h	
Bits	Description
31:12	<b>DEV_TBL_6_BASE_LO.</b> Read-write. Reset: 0_0000h. Specifies address bits[31:12] of the 4K byte-aligned base address of the specific segment of the first level device table.
11:9	Reserved.
8:0	<b>DEV_TBL_6_SIZE.</b> Read-write. Reset: 000h. This field contains 1 less than the length of the device table, in multiples of 4K bytes. A minimum size of 0 corresponds to a 4K-byte device table and a maximum size of 1FFh corresponds to a 2M-byte device table.

**IOMMUBARx0012C (IOMMUMMIO::IOMMU\_MMIO\_DEVTBL\_6\_BASE\_1)**

Read-write. Reset: 0000_0000h.	
_aliasHOST; IOMMUBARx0012C; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI], IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO], 19'h0_0000}	
_aliasSMN; IOMMUMMIOx0000012C; IOMMUMMIO=0240_0000h	
Bits	Description
31:20	Reserved.
19:0	<b>DEV_TBL_6_BASE_HI.</b> Read-write. Reset: 0_0000h. Specifies address bits[51:32] of the 4K byte-aligned base address of the first level device table.

**IOMMUBARx00130 (IOMMUMMIO::IOMMU\_MMIO\_DEVTBL\_7\_BASE\_0)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOmmUBARx00130; IOmmUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI] ,  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO] , 19'h0\_0000}  
\_aliasSMN; IOMMUMMIOx00000130; IOMMUMMIO=0240\_0000h

Bits	Description
31:12	<b>DEV_TBL_7_BASE_LO.</b> Read-write. Reset: 0_0000h. Specifies address bits[31:12] of the 4K byte-aligned base address of the specific segment of the first level device table.
11:9	Reserved.
8:0	<b>DEV_TBL_7_SIZE.</b> Read-write. Reset: 000h. This field contains 1 less than the length of the device table, in multiples of 4K bytes. A minimum size of 0 corresponds to a 4K-byte device table and a maximum size of 1FFh corresponds to a 2M-byte device table.

**IOMMUBARx00134 (IOMMUMMIO::IOMMU\_MMIO\_DEVTBL\_7\_BASE\_1)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOmmUBARx00134; IOmmUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI] ,  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO] , 19'h0\_0000}  
\_aliasSMN; IOMMUMMIOx00000134; IOMMUMMIO=0240\_0000h

Bits	Description
31:20	Reserved.
19:0	<b>DEV_TBL_7_BASE_HI.</b> Read-write. Reset: 0_0000h. Specifies address bits[51:32] of the 4K byte-aligned base address of the first level device table.

**IOMMUBARx00138 (IOMMUMMIO::IOMMU\_MMIO\_DSFX)**

Read-only. Reset: 0000\_0000h.

\_aliasHOST; IOmmUBARx00138; IOmmUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI] ,  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO] , 19'h0\_0000}  
\_aliasSMN; IOMMUMMIOx00000138; IOMMUMMIO=0240\_0000h

Bits	Description
31:28	<b>REVISION_MAJOR.</b> Read-only. Reset: 0h. Specifies the major revision.
27:24	<b>REVISION_MINOR.</b> Read-only. Reset: 0h. Specifies the minor revision.
23:0	<b>DSFXSup.</b> Read-only. Reset: 00_0000h. Specifies to software which features (bit[n]) is supported.

**IOMMUBARx00140 (IOMMUMMIO::IOMMU\_MMIO\_DSCX)**

Reset: 0000\_0000h.

\_aliasHOST; IOmmUBARx00140; IOmmUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI] ,  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO] , 19'h0\_0000}  
\_aliasSMN; IOMMUMMIOx00000140; IOMMUMMIO=0240\_0000h

Bits	Description
31:28	<b>REVISION_MAJOR.</b> Read-only. Reset: 0h. Specifies the major revision.
27:24	<b>REVISION_MINOR.</b> Read-only. Reset: 0h. Specifies the minor revision.
23:0	<b>DSCX_CNTRL.</b> Read-write. Reset: 00_0000h. Register bits to be used in hardware as controls for features. If the corresponding IOMMUMMIO::IOMMU_MMIO_DSFX[DSFXSup] bit is low the register bit in DSCX_CNTRL should be Read-only.

**IOMMUBARx00148 (IOMMUMMIO::IOMMU\_MMIO\_DSSX)**

Reset: 0000\_0000h.

\_aliasHOST; IOmmUBARx00148; IOmmUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI] ,  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO] , 19'h0\_0000}  
\_aliasSMN; IOMMUMMIOx00000148; IOMMUMMIO=0240\_0000h

Bits	Description
31:28	<b>REVISION_MAJOR.</b> Read-only. Reset: 0h. Specifies the major revision.
27:24	<b>REVISION_MINOR.</b> Read-only. Reset: 0h. Specifies the minor revision.
23:0	<b>DSSX_status.</b> Read,Write-1-to-clear. Reset: 00_0000h. Status bits used to report hardware status to software.

**IOMMUBARx00150 (IOMMUMMIO::IOMMU\_MMIO\_CAP\_MISC)**

Read-only. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx00150; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}  
\_aliasSMN; IOMMUMMIOx00000150; IOMMUMMIO=0240\_0000h

Bits	Description
31:27	<b>IOMMU_MSI_NUM_PPR.</b> Read-only. Reset: 00h. This field must indicate which MSI vector is used for the interrupt message generated by the IOMMU for the peripheral page service request log when IOMMUMMIO::IOMMU_MMIO_EFR_0[PPR_SUP] == 1. IOMMU_MSI_NUM_PPR must be 0 when IOMMUMMIO::IOMMU_MMIO_EFR_0[PPR_SUP] == 0. For MSI there can be only one IOMMU so this field must be 0. This interrupt is not remapped by the IOMMU.
26:5	Reserved.
4:0	<b>IOMMU_MSI_NUM.</b> Read-only. Reset: 00h. Indicates the MSI vector used for interrupt messages generated by the IOMMU.

**IOMMUBARx00154 (IOMMUMMIO::IOMMU\_MMIO\_CAP\_MISC\_1)**

Read-only. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx00154; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}  
\_aliasSMN; IOMMUMMIOx00000154; IOMMUMMIO=0240\_0000h

Bits	Description
31:5	Reserved.
4:0	<b>IOMMU_MSI_NUM_GA.</b> Read-only. Reset: 00h. This field must indicate which MSI vector is used for the interrupt message generated by the IOMMU.

**IOMMUBARx00158 (IOMMUMMIO::IOMMU\_MMIO\_MSI\_CAP)**

\_aliasHOST; IOMMUBARx00158; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}  
\_aliasSMN; IOMMUMMIOx00000158; IOMMUMMIO=0240\_0000h

Bits	Description
31:24	Reserved.
23	<b>MSI_64_EN.</b> Read-only. Reset: Fixed,1. 0=32-bit MSI address supported. 1=64-bit MSI address supported. Set to indicate that a 64-bit MSI address is supported.
22:20	<b>MSI_MULT_MESS_EN.</b> Read-write. Reset: 0h. Sets the number of MSI messages assigned to this function.
19:17	<b>MSI_MULT_MESS_CAP.</b> Read-only. Reset: 2h. Indicates the number of MSI messages requested by this function.
16	<b>MSI_EN.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. Enables MSI for this function and causes legacy interrupts to be disabled.
15:8	<b>MSI_CAP_PTR.</b> Read-only. Reset: Fixed,74h. Pointer to the next configuration space capability.
7:0	<b>MSI_CAP_ID.</b> Read-only. Reset: Fixed,05h. Indicates that this is the MSI capability.

**IOMMUBARx0015C (IOMMUMMIO::IOMMU\_MMIO\_MSI\_ADDR\_LO)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx0015C; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}  
\_aliasSMN; IOMMUMMIOx0000015C; IOMMUMMIO=0240\_0000h

Bits	Description
31:2	<b>MSI_ADDR_LO.</b> Read-write. Reset: 0000_0000h. This register sets the address bits[31:2] that are used to issue MSI messages.
1:0	Reserved.

**IOMMUBARx00160 (IOMMUMMIO::IOMMU\_MMIO\_MSI\_ADDR\_HI)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx00160; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}



IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO] , 19'h0_0000} _aliasSMN; IOMMUMMIOx00000160; IOMMUMMIO=0240_0000h	
Bits	Description
31:0	<b>MSI_ADDR_HI.</b> Read-write. Reset: 0000_0000h. This register sets the address bits[63:32] that are used to issue MSI messages.

**IOMMUBARx00164 (IOMMUMMIO::IOMMU\_MMIO\_MSI\_DATA)**

Read-write. Reset: 0000_0000h. _aliasHOST; IOMMUBARx00164; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI] , IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO] , 19'h0_0000} _aliasSMN; IOMMUMMIOx00000164; IOMMUMMIO=0240_0000h	
Bits	Description
31:16	Reserved.
15:0	<b>MSI_DATA.</b> Read-write. Reset: 0000h. This register sets the data issued with MSI messages.

**IOMMUBARx00168 (IOMMUMMIO::IOMMU\_MMIO\_MSI\_MAPPING\_CAP)**

Read-only. Reset: Fixed,A803_0008h. _aliasHOST; IOMMUBARx00168; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI] , IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO] , 19'h0_0000} _aliasSMN; IOMMUMMIOx00000168; IOMMUMMIO=0240_0000h	
Bits	Description
31:27	<b>MSI_MAP_CAP_TYPE.</b> Read-only. Reset: Fixed,15h. Indicates the MSI mapping Capability.
26:18	Reserved.
17	<b>MSI_MAP_FIXD.</b> Read-only. Reset: Fixed,1. 0=No interrupt address limit. 1=Address 0xFEEX_XXXX mapped to HyperTransport interrupts. Always set to 1 to indicate that this device only maps MSI interrupts with address 0xFEEX_XXXX onto HyperTransport interrupts and that the mapping range is not programmable.
16	<b>MSI_MAP_EN.</b> Read-only. Reset: Fixed,1. 0=Disabled. 1=Enabled. Always set to 1 to indicate that the MSI Mapping Capability is always enabled.
15:8	<b>MSI_MAP_CAP_PTR.</b> Read-only. Reset: Fixed,00h. Points to the next capability list item.
7:0	<b>MSI_MAP_CAP_ID.</b> Read-only. Reset: Fixed,08h. Indicates a HyperTransport capability list item.

**IOMMUBARx0016C (IOMMUMMIO::IOMMU\_MMIO\_CONTROL\_W)**

Read-write. Reset: 0000_0000h. _aliasHOST; IOMMUBARx0016C; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI] , IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO] , 19'h0_0000} _aliasSMN; IOMMUMMIOx0000016C; IOMMUMMIO=0240_0000h	
Bits	Description
31:14	Reserved.
13	<b>GMC_IOMMU_BYPASS.</b> Read-write. Reset: 0. 0=Bypass disabled. 1=Bypass enabled. Set to 1 whenever IOMMU is not performing HOST translations or checks (INTGFX only). IOMMU indicates to Graphics Memory Controller (GMC) that the IOMMU is disabled, and GMC will bypass the IOMMU.
12:0	Reserved.

**IOMMUBARx00230 (IOMMUMMIO::IOMMU\_MARC\_BASE\_LO\_2)**

Read-write. Reset: 0000_0000h. _aliasHOST; IOMMUBARx00230; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI] , IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO] , 19'h0_0000} _aliasSMN; IOMMUMMIOx00000230; IOMMUMMIO=0240_0000h	
Bits	Description
31:12	<b>MARCBASEAddr_L_2.</b> Read-write. Reset: 0_0000h. MARC aperture base address. Specifies bits[31:12] of the 4K byte aligned base address of a MARC memory aperture in the device's address space.
11:0	Reserved.

**IOMMUBARx02000 (IOMMUMMIO::IOMMU\_MMIO\_CMD\_BUF\_HDPTR\_0)**

Read-write. Reset: 0000_0000h.	
--------------------------------	--



_aliasHOST; IOMMUBARx02000; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI], IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO], 19'h0_0000}	
_aliasSMN; IOMMUMMIOx00002000; IOMMUMMIO=0240_0000h	
Bits	Description
31:19	Reserved.
18:4	<b>CMD_HDPTR.</b> Read-write. Reset: 0000h. Specifies the 128-bit aligned offset from the command buffer base address register of the next command to be fetched by the IOMMU. The IOMMU increments this register, rolling over to zero at the end of the buffer and after fetching and validating the command in the command buffer. After incrementing this register, the IOMMU cannot re-fetch the command from the buffer. If this register is Written to by software while IOMMUMMIO::IOMMU_MMIO_STATUS_0[CMD_BUFRUN] == 1, the IOMMU behavior is undefined. If this register is set by software to a value outside the length specified by IOMMUMMIO::IOMMU_MMIO_CMD_BASE_1[COM_LEN], the IOMMU behavior is undefined.
3:0	Reserved.

**IOMMUBARx02004 (IOMMUMMIO::IOMMU\_MMIO\_CMD\_BUF\_HDPTR\_1)**

Reset: 0000_0000h.	
_aliasHOST; IOMMUBARx02004; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI], IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO], 19'h0_0000}	
_aliasSMN; IOMMUMMIOx00002004; IOMMUMMIO=0240_0000h	
Bits	Description
31:0	Reserved.

**IOMMUBARx02008 (IOMMUMMIO::IOMMU\_MMIO\_CMD\_BUF\_TAILPTR\_0)**

Read-write. Reset: 0000_0000h.	
_aliasHOST; IOMMUBARx02008; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI], IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO], 19'h0_0000}	
_aliasSMN; IOMMUMMIOx00002008; IOMMUMMIO=0240_0000h	
Bits	Description
31:19	Reserved.
18:4	<b>CMD_TAILPTR.</b> Read-write. Reset: 0000h. Specifies the 128-bit aligned offset from the command buffer base address register of the next command to be Written by the software. Software must increment this field, rolling over to zero at the end of the buffer and after Writing a command to the command buffer. If software advances the tail pointer equal to or beyond the head pointer after adding one or more commands to the buffer, the IOMMU behavior is undefined. If software sets the command buffer tail pointer to an offset beyond the length of the command buffer, the IOMMU behavior is undefined.
3:0	Reserved.

**IOMMUBARx0200C (IOMMUMMIO::IOMMU\_MMIO\_CMD\_BUF\_TAILPTR\_1)**

Reset: 0000_0000h.	
_aliasHOST; IOMMUBARx0200C; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI], IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO], 19'h0_0000}	
_aliasSMN; IOMMUMMIOx0000200C; IOMMUMMIO=0240_0000h	
Bits	Description
31:0	Reserved.

**IOMMUBARx02010 (IOMMUMMIO::IOMMU\_MMIO\_EVENT\_BUF\_HDPTR\_0)**

Read-write. Reset: 0000_0000h.	
_aliasHOST; IOMMUBARx02010; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI], IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO], 19'h0_0000}	
_aliasSMN; IOMMUMMIOx00002010; IOMMUMMIO=0240_0000h	
Bits	Description
31:19	Reserved.
18:4	<b>EVENT_HDPTR.</b> Read-write. Reset: 0000h. Specifies the 128-bit aligned offset from the event log base address register that is Read next by software. Software must increment this field, rolling over at the end of the buffer and after Reading an event from the event log. If software advances the head pointer beyond the tail pointer, the

	IOMMU behavior is undefined. If software sets the event log head pointer to an offset beyond the length of the event log, the IOMMU behavior is undefined.
3:0	Reserved.

**IOMMUBARx02014 (IOMMUMMIO::IOMMU\_MMIO\_EVENT\_BUF\_HDPTR\_1)**

Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx02014; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}  
\_aliasSMN; IOMMUMMIOx00002014; IOMMUMMIO=0240\_0000h

Bits	Description
31:0	Reserved.

**IOMMUBARx02018 (IOMMUMMIO::IOMMU\_MMIO\_EVENT\_BUF\_TAILPTR\_0)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx02018; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}  
\_aliasSMN; IOMMUMMIOx00002018; IOMMUMMIO=0240\_0000h

Bits	Description
31:19	Reserved.
18:4	<b>EVENT_TAILPTR.</b> Read-write. Reset: 0000h. Specifies the 128-bit aligned offset from the event log base address register that is Written next by the IOMMU when an event is detected. The IOMMU increments this register, rolling over at the end of the buffer and after Writing an event to the event log. If this register is Written while IOMMUMMIO::IOMMU_MMIO_STATUS_0[EVENT_LOGRUN] == 1, the IOMMU behavior is undefined. If this register is set by software to a value outside the length specified by IOMMUMMIO::IOMMU_MMIO_EVENT_BASE_1[EVENT_LEN], the IOMMU behavior is undefined.
3:0	Reserved.

**IOMMUBARx0201C (IOMMUMMIO::IOMMU\_MMIO\_EVENT\_BUF\_TAILPTR\_1)**

Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx0201C; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}  
\_aliasSMN; IOMMUMMIOx0000201C; IOMMUMMIO=0240\_0000h

Bits	Description
31:0	Reserved.

**IOMMUBARx02020 (IOMMUMMIO::IOMMU\_MMIO\_STATUS\_0)**

Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx02020; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}  
\_aliasSMN; IOMMUMMIOx00002020; IOMMUMMIO=0240\_0000h

Bits	Description
31:19	Reserved.
18	<b>PPR_OVERFLOW_EARLY.</b> Read,Write-1-to-clear. Reset: 0. 0=No early overflow reached. 1=IOMMU event log A has reached early overflow threshold. This bit is set when the respected PPR log has reached the programmable threshold and when IOMMUMMIO::IOMMU_MMIO_PPR_OVERFLOW_EARLY_0[PPR_Overflow_early_en] == 1. When IOMMUMMIO::IOMMU_MMIO_CNTRL_1[PPR_Auto_resp_en] == 1, hardware generates auto responses for the last PPR request and let Stop marker PPR msg through the log. Cleared on Writing to 1.
17	<b>PPR_B_OVERFLOW_EARLY.</b> Read,Write-1-to-clear. Reset: 0. 0=No early overflow reached. 1=IOMMU event log B has reached early overflow threshold. This bit is set when the respected PPR log has reached the programmable threshold and when IOMMUMMIO::IOMMU_MMIO_PPR_OVERFLOW_EARLY_0[PPR_Overflow_early_en] == 1. When IOMMUMMIO::IOMMU_MMIO_CNTRL_1[PPR_Auto_resp_en] == 1, hardware generates auto responses for the last PPR request and let Stop marker PPR msg through the log. Cleared on Writing to 1.

16	<b>EVENT_BUF_ACTIVE</b> . Read-only. Reset: 0. 0=EVENT default buffer being used by hardware. 1=EVENT buffer B being used by hardware. EVENT buffer active.
15	<b>EVENT_B_OVERFLOW</b> . Read,Write-1-to-clear. Reset: 0. 0=No event log B overflow. 1=IOMMU event log B overflow has occurred. This bit is set when a new event is to be Written to the event log and there is no usable entry in the event log, causing the new event information to be discarded. An interrupt is generated when <code>EVENT_B_OVERFLOW == 1</code> and <code>IOMMUMMIO::IOMMU_MMIO_CNTRL_0[EVENT_INT_EN] == 1</code> . No new event log entries are Written while this bit is set. Cleared on Writing to 1.
14:13	Reserved.
12	<b>PPR_BUF_ACTIVE</b> . Read-only. Reset: 0. 0=PPR default buffer being used by hardware. 1=PPR buffer B being used by hardware. PPR active buffer.
11	<b>PPR_B_OVERFLOW</b> . Read,Write-1-to-clear. Reset: 0. 0=No PPR log B overflow. 1=IOMMU PPR log B overflow has occurred. This bit is set when a new peripheral page service request is to be Written to the PPR log and there is no usable entry in the PPR log, causing the new information to be discarded. An interrupt is generated when <code>PPR_B_OVERFLOW == 1</code> and <code>IOMMUMMIO::IOMMU_MMIO_CNTRL_0[PPR_INT_EN] == 1</code> . No new PPR log entries are Written while this bit is set. Cleared on Writing to 1.
10	<b>GA_INT</b> . Read-write. Reset: 0. 0=No GA entry Written to the GA log by the IOMMU. 1=GA request entry Written to the GA log by the IOMMU. An interrupt is generated when <code>GA_INT == 1</code> and <code>IOMMUMMIO::IOMMU_MMIO_CNTRL_0[GA_INT_EN] == 1</code> . Cleared on Writing to 1.
9	<b>GA_OVERFLOW</b> . Read,Write-1-to-clear. Reset: 0. 0=No GA log overflow has occurred. IOMMU GA log overflow has occurred.
8	<b>GA_RUN</b> . Read-only. Reset: 0. 0=GA requests are discarded without logging. 1=GA requests are logged as they occur. When <code>GA_OVERFLOW == 1</code> , the IOMMU does not Write new GA log entries even when <code>GA_RUN == 1</code> . When halted, GA request logging is restarted by using <code>IOMMUMMIO::IOMMU_MMIO_CNTRL_0[GA_LOG_EN]</code> .
7	<b>PPR_RUN</b> . Read-only. Reset: 0. 0=PPR requests are discarded without logging. 1=PPR requests are logged as they occur. When <code>PPR_OVERFLOW == 1</code> , the IOMMU does not Write new PPR log entries even when <code>PPR_RUN == 1</code> . When halted, PPR request logging is restarted by using <code>IOMMUMMIO::IOMMU_MMIO_CNTRL_0[PPR_LOG_EN]</code> .
6	<b>PPR_INT</b> . Read,Write-1-to-clear. Reset: 0. 0=No PPR entry Written to the PPR log by the IOMMU. 1=PPR request entry Written to the PPR log by the IOMMU. An interrupt is generated when <code>PPR_INT == 1</code> and <code>IOMMUMMIO::IOMMU_MMIO_CNTRL_0[PPR_INT_EN] == 1</code> . Cleared on Writing to 1.
5	<b>PPR_OVERFLOW</b> . Read,Write-1-to-clear. Reset: 0. 0=No PPR log overflow has occurred. 1=IOMMU PPR log overflow has occurred. This bit is set when a new peripheral page service request is to be Written to the PPR log and there is no usable entry in the PPR log, causing the new information to be discarded. An interrupt is generated when <code>PPR_OVERFLOW == 1</code> and <code>IOMMUMMIO::IOMMU_MMIO_CNTRL_0[PPR_INT_EN] == 1</code> . No new PPR log entries are Written while this bit is set. Cleared on Writing to 1.
4	<b>CMD_BUFRUN</b> . Read-only. Reset: 0. 0=IOMMU has stopped fetching new commands. 1=Commands may be fetched from the command buffer. The IOMMU freezes command processing after <code>COMMAND_HARDWARE_ERROR</code> or <code>ILLEGAL_COMMAND_ERROR</code> errors. When frozen, command fetching is restarted by using <code>IOMMUMMIO::IOMMU_MMIO_CNTRL_0[CMD_BUF_EN]</code> .
3	<b>EVENT_LOGRUN</b> . Read-only. Reset: 0. 0=Event reports are discarded without logging. 1=Events are logged as they occur. When <code>EVENT_OVERFLOW == 1</code> , the IOMMU does not Write new event log entries even when <code>EVENT_LOGRUN == 1</code> . When halted, event logging is restarted by using <code>IOMMUMMIO::IOMMU_MMIO_CNTRL_0[EVENT_LOG_EN]</code> .
2	<b>COMWAIT_INT</b> . Read,Write-1-to-clear. Reset: 0. 0=COMPLETION_WAIT command not run or not completed. 1=COMPLETION_WAIT command completed. This bit is only set if the [i] bit is set in the COMPLETION_WAIT command. An interrupt is generated when <code>COMWAIT_INT == 1</code> and <code>IOMMUMMIO::IOMMU_MMIO_CNTRL_0[COM_WAIT_INTEN] == 1</code> . Cleared on Writing to 1.
1	<b>EVENT_LOGINT</b> . Read,Write-1-to-clear. Reset: 0. 0=No event entry Written to the event log by the IOMMU. 1=Event entry Written to the event log by the IOMMU. An interrupt is generated when <code>EVENT_LOGINT == 1</code> and <code>IOMMUMMIO::IOMMU_MMIO_CNTRL_0[EVENT_INT_EN] == 1</code> . Cleared on Writing to 1.
0	<b>EVENT_OVERFLOW</b> . Read,Write-1-to-clear. Reset: 0. 0=No IOMMU event log overflow has occurred.

	1=IOMMU event log overflow has occurred. This bit is set when a new event is to be Written to the event log and there is no usable entry in the event log, causing the new event information to be discarded. An interrupt is generated when <code>EVENT_OVERFLOW == 1</code> and <code>IOMMUMMIO::IOMMU_MMIO_CNTRL_0[EVENT_INT_EN] == 1</code> . No new event log entries are Written while this bit is set. Cleared on Writing to 1.
--	--

**IOMMUBARx02024 (IOMMUMMIO::IOMMU\_MMIO\_STATUS\_1)**

Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx02024; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}  
\_aliasSMN; IOMMUMMIOx00002024; IOMMUMMIO=0240\_0000h

**Bits Description**

31:0 Reserved.

**IOMMUBARx02030 (IOMMUMMIO::IOMMU\_MMIO\_PPR\_BUF\_HDPTR\_0)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx02030; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}  
\_aliasSMN; IOMMUMMIOx00002030; IOMMUMMIO=0240\_0000h

**Bits Description**

31:19 Reserved.

18:4 **PPR\_HDPTR**. Read-write. Reset: 0000h. Specifies the 128-bit aligned offset from the PPR log base address register that is Read next by software. Software must increment this field, rolling over at the end of the buffer and after Reading a PPR request entry from the PPR event log. If software advances the head pointer beyond the tail pointer, the IOMMU behavior is undefined. If software sets the PPR log head pointer to an offset beyond the length of the PPR log, the IOMMU behavior is undefined.

3:0 Reserved.

**IOMMUBARx02034 (IOMMUMMIO::IOMMU\_MMIO\_PPR\_BUF\_HDPTR\_1)**

Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx02034; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}  
\_aliasSMN; IOMMUMMIOx00002034; IOMMUMMIO=0240\_0000h

**Bits Description**

31:0 Reserved.

**IOMMUBARx02038 (IOMMUMMIO::IOMMU\_MMIO\_PPR\_BUF\_TAILPTR\_0)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx02038; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}  
\_aliasSMN; IOMMUMMIOx00002038; IOMMUMMIO=0240\_0000h

**Bits Description**

31:19 Reserved.

18:4 **PPR\_TAILPTR**. Read-write. Reset: 0000h. Specifies the 128-bit aligned offset from the PPR log base address register that is Written next by the IOMMU when a PPR request is detected. The IOMMU increments this register, rolling over at the end of the buffer and after Writing a PPR request to the PPR log. If this register is Written while `IOMMUMMIO::IOMMU_MMIO_STATUS_0[PPR_RUN] == 1`, the IOMMU behavior is undefined. If software sets the PPR log tail pointer to an offset beyond the length of the PPR log, defined by `iIOMMUMMIO::IOMMU_MMIO_PPR_BASE_1[PPR_LEN]`, the IOMMU behavior is undefined.

3:0 Reserved.

**IOMMUBARx0203C (IOMMUMMIO::IOMMU\_MMIO\_PPR\_BUF\_TAILPTR\_1)**

Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx0203C; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

_aliasSMN; IOMMUMMIOx0000203C; IOMMUMMIO=0240_0000h	
Bits	Description
31:0	Reserved.

**IOMMUBARx02040 (IOMMUMMIO::IOMMU\_MMIO\_GA\_BUF\_HDPTR\_0)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx02040; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI], IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx00002040; IOMMUMMIO=0240\_0000h

Bits	Description
31:16	Reserved.
15:3	<b>GA_HDPTR.</b> Read-write. Reset: 0000h. Specifies the 128-bit aligned offset from the GA log base address register that is Read next by software. Software must increment this field, rolling over at the end of the buffer, after Reading a GA request entry from the GA event log. If software advances the head pointer beyond the tail pointer, the IOMMU behavior is undefined. If software sets the GA log head pointer to an offset beyond the length of the GA log, the IOMMU behavior is undefined.
2:0	Reserved.

**IOMMUBARx02044 (IOMMUMMIO::IOMMU\_MMIO\_GA\_BUF\_HDPTR\_1)**

Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx02044; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI], IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx00002044; IOMMUMMIO=0240\_0000h

Bits	Description
31:0	Reserved.

**IOMMUBARx02048 (IOMMUMMIO::IOMMU\_MMIO\_GA\_BUF\_TAILPTR\_0)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx02048; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI], IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx00002048; IOMMUMMIO=0240\_0000h

Bits	Description
31:16	Reserved.
15:3	<b>GA_TAILPTR.</b> Read-write. Reset: 0000h. Guest virtual APIC log tail pointer. Specifies the 8-byte aligned offset from IOMMUMMIO::IOMMU_MMIO_GA_LOG_BASE_0 that is Written next by the IOMMU when an undelivered virtual interrupt request needs to be entered into the log. The IOMMU increments this register, rolling over at the end of the buffer and after Writing an entry into the log.
2:0	Reserved.

**IOMMUBARx0204C (IOMMUMMIO::IOMMU\_MMIO\_GA\_BUF\_TAILPTR\_1)**

Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx0204C; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI], IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx0000204C; IOMMUMMIO=0240\_0000h

Bits	Description
31:0	Reserved.

**IOMMUBARx02050 (IOMMUMMIO::IOMMU\_MMIO\_PPR\_B\_BUF\_HDPTR\_0)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx02050; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI], IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx00002050; IOMMUMMIO=0240\_0000h

Bits	Description
31:19	Reserved.
18:4	<b>PPR_B_HDPTR.</b> Read-write. Reset: 0000h. Specifies the 128-bit aligned offset from the PPR log base address

	register that is Read next by software. Software must increment this field, rolling over at the end of the buffer and after Reading a PPR request entry from the PPR event log. If software advances the head pointer beyond the tail pointer, the IOMMU behavior is undefined. If software sets the PPR log head pointer to an offset beyond the length of the PPR log, the IOMMU behavior is undefined.
3:0	Reserved.

**IOMMUBARx02054 (IOMMUMMIO::IOMMU\_MMIO\_PPR\_B\_BUF\_HDPTR\_1)**

Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx02054; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI], IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx00002054; IOMMUMMIO=0240\_0000h

Bits	Description
31:0	Reserved.

**IOMMUBARx02058 (IOMMUMMIO::IOMMU\_MMIO\_PPR\_B\_BUF\_TAILPTR\_0)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx02058; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI], IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx00002058; IOMMUMMIO=0240\_0000h

Bits	Description
31:19	Reserved.
18:4	<b>PPR_B_TAILPTR.</b> Read-write. Reset: 0000h. Specifies the 128-bit aligned offset from the PPR log base address register that is Written next by the IOMMU when a PPR request is detected. The IOMMU increments this register, rolling over at the end of the buffer and after Writing a PPR request to the PPR log. If this register is Written while IOMMUMMIO::IOMMU_MMIO_STATUS_0[PPR_RUN] == 1, the IOMMU behavior is undefined. If software sets the PPR log tail pointer to an offset beyond the length of the PPR log, defined by IOMMUMMIO::IOMMU_MMIO_PPR_BASE_1[PPR_LEN], the IOMMU behavior is undefined.
3:0	Reserved.

**IOMMUBARx0205C (IOMMUMMIO::IOMMU\_MMIO\_PPR\_B\_BUF\_TAILPTR\_1)**

Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx0205C; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI], IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx0000205C; IOMMUMMIO=0240\_0000h

Bits	Description
31:0	Reserved.

**IOMMUBARx02070 (IOMMUMMIO::IOMMU\_MMIO\_EVENT\_B\_BUF\_HDPTR\_0)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx02070; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI], IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx00002070; IOMMUMMIO=0240\_0000h

Bits	Description
31:19	Reserved.
18:4	<b>EVENT_B_HDPTR.</b> Read-write. Reset: 0000h. Specifies the 128-bit aligned offset from the event log B base address register that is Read next by software. Software must increment this field, rolling over at the end of the buffer and after Reading an event from the event log. If software advances the head pointer beyond the tail pointer, the IOMMU behavior is undefined. If software sets the event log head pointer to an offset beyond the length of the event log, the IOMMU behavior is undefined.
3:0	Reserved.

**IOMMUBARx02074 (IOMMUMMIO::IOMMU\_MMIO\_EVENT\_B\_BUF\_HDPTR\_1)**

Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx02074; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI], IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}



_aliasSMN; IOMMUMMIOx00002074; IOMMUMMIO=0240_0000h	
Bits	Description
31:0	Reserved.

**IOMMUBARx02078 (IOMMUMMIO::IOMMU\_MMIO\_EVENT\_B\_BUF\_TAILPTR\_0)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx02078; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI], IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx00002078; IOMMUMMIO=0240\_0000h

Bits	Description
31:19	Reserved.
18:4	<b>EVENT_B_TAILPTR.</b> Read-write. Reset: 0000h. Specifies the 128-bit aligned offset from the event log B base address register that is Written next by the IOMMU when an event is detected. The IOMMU increments this register, rolling over at the end of the buffer and after Writing an event to the event log. If this register is Written while IOMMUMMIO::IOMMU_MMIO_STATUS_0[EVENT_LOGRUN] == 1, the IOMMU behavior is undefined. If this register is set by software to a value outside the length specified by IOMMUMMIO::IOMMU_MMIO_EVENT_BASE_1[EVENT_LEN], the IOMMU behavior is undefined.
3:0	Reserved.

**IOMMUBARx0207C (IOMMUMMIO::IOMMU\_MMIO\_EVENT\_B\_BUF\_TAILPTR\_1)**

Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx0207C; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI], IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx0000207C; IOMMUMMIO=0240\_0000h

Bits	Description
31:0	Reserved.

**IOMMUBARx02080 (IOMMUMMIO::IOMMU\_MMIO\_PPR\_AUTORESP\_0)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx02080; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI], IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx00002080; IOMMUMMIO=0240\_0000h

Bits	Description
31:5	Reserved.
4	<b>PPR_Auto_resp_mask_gn.</b> Read-write. Reset: 0. 0=No Mask. 1=1=Mask GN bit and PASID to 0. Mask GN bit and PASID to 0 for Auto Response, or else return fields from PPR request.
3:0	<b>PPR_Auto_resp_code.</b> Read-write. Reset: 0h. Programable PPR code for Auto Response. 0=SUCCESSFUL on Default.

**IOMMUBARx02088 (IOMMUMMIO::IOMMU\_MMIO\_PPR\_OVERFLOW\_EARLY\_0)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx02088; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI], IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx00002088; IOMMUMMIO=0240\_0000h

Bits	Description
31	<b>PPR_Overflow_early_en.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. Enable PPR early overflow mechanism in hardware.
30	<b>PPR_Overflow_early_int_en.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. Enable PPR early overflow interrupt.
29:15	Reserved.
14:0	<b>PPR_Overflow_early_threshold.</b> Read-write. Reset: 0000h. Number of entries prior to overflow that hardware should signal early overflow.

**IOMMUBARx02090 (IOMMUMMIO::IOMMU\_MMIO\_PPR\_B\_OVERFLOW\_EARLY\_0)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx02090; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],



IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO], 19'h0_0000}	
_aliasSMN; IOMMUMMIOx00002090; IOMMUMMIO=0240_0000h	
Bits	Description
31	<b>PPR_B_Overflow_early_en.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. Enable PPR early overflow mechanism in hardware.
30	<b>PPR_B_Overflow_early_int_en.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. Enable PPR early overflow interrupt.
29:15	Reserved.
14:0	<b>PPR_B_Overflow_early_threshold.</b> Read-write. Reset: 0000h. Number of entries prior to overflow that hardware should signal early overflow.

**IOMMUBARx04000 (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_CONFIG\_0)**

Read-only. Reset: Fixed,0000\_2200h.

\_aliasHOST; IOMMUBARx04000; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}  
\_aliasSMN; IOMMUMMIOx00004000; IOMMUMMIO=0240\_0000h

Bits	Description								
31:18	Reserved.								
17:12	<b>N_COUNTER_BANKS.</b> Read-only. Reset: Fixed,02h.  <b>Description:</b> The number of counter banks supported by the IOMMU. Each bank contains two or more counter and control registers as specified by NCounter. For each counter bank, a corresponding control bit is in IOMMU Counter PASID Bank-Lock Register IOMMUMMIO::IOMMU_MMIO_COUNTER_PASID_BANK_LOCK_0, IOMMU Counter Domain Bank-Lock Register IOMMUMMIO::IOMMU_MMIO_COUNTER_DOMAIN_BANK_LOCK_0 and IOMMU Counter DeviceID Bank-Lock Register IOMMUMMIO::IOMMU_MMIO_COUNTER_DEVID_BANK_LOCK_0. Each supported event counter bank is in a distinct, consecutive 4K-byte page. NOTE: IOMMU event counter banks are numbered starting with 0.  <b>ValidValues:</b>								
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>00h</td><td>No event counter banks supported.</td></tr> <tr> <td>3Fh-01h</td><td>The number of event counter banks supported.</td></tr> </table>	Value	Description	00h	No event counter banks supported.	3Fh-01h	The number of event counter banks supported.		
Value	Description								
00h	No event counter banks supported.								
3Fh-01h	The number of event counter banks supported.								
11	Reserved.								
10:7	<b>N_COUNTER.</b> Read-only. Reset: Fixed,4h.  <b>Description:</b> Reports the number of individual counters in each IOMMU counter bank. Each counter bank contains the same number of counters. 0=No counters supported. 1=Reserved 2-15=number of counters in each counter bank.  <b>ValidValues:</b>								
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>No counters supported.</td></tr> <tr> <td>1h</td><td>Reserved.</td></tr> <tr> <td>Fh-2h</td><td>Number of counters in each counter bank.</td></tr> </table>	Value	Description	0h	No counters supported.	1h	Reserved.	Fh-2h	Number of counters in each counter bank.
Value	Description								
0h	No counters supported.								
1h	Reserved.								
Fh-2h	Number of counters in each counter bank.								
6:0	Reserved.								

**IOMMUBARx04004 (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_CONFIG\_1)**

Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx04004; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}  
\_aliasSMN; IOMMUMMIOx00004004; IOMMUMMIO=0240\_0000h

Bits	Description
31:0	Reserved.

**IOMMUBARx04008 (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_PASID\_BANK\_LOCK\_0)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx04008; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}  
\_aliasSMN; IOMMUMMIOx00004008; IOMMUMMIO=0240\_0000h

Bits	Description																						
31:0	<p><b>PASID_LOCK_LO.</b> Read-write. Reset: 0000_0000h.</p> <p><b>Description:</b> PASID lock enable. For each bit in PASID_LOCK_LO, the corresponding PASID-match registers in an IOMMU counter bank may be changed. See IOMMU PASID Match Registers</p> <ul style="list-style-type: none"> <li>- IOMMUMMIO::IOMMU_MMIO_PASID_MATCH_BANK_0_CNT_0_0 and IOMMUMMIO::IOMMU_MMIO_PASID_MATCH_BANK_0_CNT_0_1,</li> <li>- IOMMUMMIO::IOMMU_MMIO_PASID_MATCH_BANK_0_CNT_1_0 and IOMMUMMIO::IOMMU_MMIO_PASID_MATCH_BANK_0_CNT_1_1,</li> <li>- IOMMUMMIO::IOMMU_MMIO_PASID_MATCH_BANK_0_CNT_2_0 and IOMMUMMIO::IOMMU_MMIO_PASID_MATCH_BANK_0_CNT_2_1,</li> <li>- IOMMUMMIO::IOMMU_MMIO_PASID_MATCH_BANK_0_CNT_3_0 and IOMMUMMIO::IOMMU_MMIO_PASID_MATCH_BANK_0_CNT_3_1,</li> <li>- IOMMUMMIO::IOMMU_MMIO_PASID_MATCH_BANK_1_CNT_0_0 and IOMMUMMIO::IOMMU_MMIO_PASID_MATCH_BANK_1_CNT_0_1,</li> <li>- IOMMUMMIO::IOMMU_MMIO_PASID_MATCH_BANK_1_CNT_1_0 and IOMMUMMIO::IOMMU_MMIO_PASID_MATCH_BANK_1_CNT_1_1,</li> <li>- IOMMUMMIO::IOMMU_MMIO_PASID_MATCH_BANK_1_CNT_2_0 and IOMMUMMIO::IOMMU_MMIO_PASID_MATCH_BANK_1_CNT_2_1,</li> <li>- IOMMUMMIO::IOMMU_MMIO_PASID_MATCH_BANK_1_CNT_3_0 and IOMMUMMIO::IOMMU_MMIO_PASID_MATCH_BANK_1_CNT_3_1.</li> </ul> <p>Bit positions above the value reported in IOMMUMMIO::IOMMU_MMIO_COUNTER_CONFIG_0[N_COUNTER_BANKS] are ignored when Written and return zero when Read. The counter banks are numbered starting with zero (See IOMMUMMIO::IOMMU_MMIO_COUNTER_CONFIG_0[N_COUNTER_BANKS]), are ignored when Written and return zero when Read. The counter banks are numbered starting with zero; PASID_LOCK_LO[0] controls bank 0, etc.</p> <p><b>Valid Values:</b></p> <table> <tr> <th>Bit</th><th>Description</th></tr> <tr> <td>[0]</td><td>0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).</td></tr> <tr> <td>[1]</td><td>0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).</td></tr> <tr> <td>[2]</td><td>0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).</td></tr> <tr> <td>[3]</td><td>0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).</td></tr> <tr> <td>[4]</td><td>0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).</td></tr> <tr> <td>[5]</td><td>0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).</td></tr> <tr> <td>[6]</td><td>0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).</td></tr> <tr> <td>[7]</td><td>0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).</td></tr> <tr> <td>[8]</td><td>0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).</td></tr> <tr> <td>[9]</td><td>0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).</td></tr> </table>	Bit	Description	[0]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).	[1]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).	[2]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).	[3]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).	[4]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).	[5]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).	[6]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).	[7]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).	[8]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).	[9]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).
Bit	Description																						
[0]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).																						
[1]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).																						
[2]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).																						
[3]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).																						
[4]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).																						
[5]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).																						
[6]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).																						
[7]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).																						
[8]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).																						
[9]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).																						

[10]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).
[11]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).
[12]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).
[13]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).
[14]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).
[15]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).
[16]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).
[17]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).
[18]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).
[19]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).
[20]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).
[21]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).
[22]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).
[23]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).
[24]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).
[25]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).
[26]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).
[27]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).
[28]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).
[29]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).
[30]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).
[31]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).

#### IOMMUBARx0400C (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_PASID\_BANK\_LOCK\_1)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx0400C; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx0000400C; IOMMUMMIO=0240\_0000h

Bits	Description
31:0	<p><b>PASID_LOCK_HI.</b> Read-write. Reset: 0000_0000h.</p> <p><b>Description:</b> PASID lock enable. For each bit in PASID_LOCK_HI, the corresponding PASID-match registers in an IOMMU counter bank may be changed. See IOMMU PASID Match Registers</p> <ul style="list-style-type: none"> <li>- IOMMUMMIO::IOMMU_MMIO_PASID_MATCH_BANK_0_CNT_0_0 and IOMMUMMIO::IOMMU_MMIO_PASID_MATCH_BANK_0_CNT_0_1,</li> <li>- IOMMUMMIO::IOMMU_MMIO_PASID_MATCH_BANK_0_CNT_1_0 and IOMMUMMIO::IOMMU_MMIO_PASID_MATCH_BANK_0_CNT_1_1,</li> <li>- IOMMUMMIO::IOMMU_MMIO_PASID_MATCH_BANK_0_CNT_2_0 and IOMMUMMIO::IOMMU_MMIO_PASID_MATCH_BANK_0_CNT_2_1,</li> <li>- IOMMUMMIO::IOMMU_MMIO_PASID_MATCH_BANK_0_CNT_3_0 and IOMMUMMIO::IOMMU_MMIO_PASID_MATCH_BANK_0_CNT_3_1,</li> <li>- IOMMUMMIO::IOMMU_MMIO_PASID_MATCH_BANK_1_CNT_0_0 and IOMMUMMIO::IOMMU_MMIO_PASID_MATCH_BANK_1_CNT_0_1,</li> <li>- IOMMUMMIO::IOMMU_MMIO_PASID_MATCH_BANK_1_CNT_1_0 and IOMMUMMIO::IOMMU_MMIO_PASID_MATCH_BANK_1_CNT_1_1,</li> </ul>

- IOMMUMMIO::IOMMU\_MMIO\_PASID\_MATCH\_BANK\_1\_CNT\_2\_0 and  
IOMMUMMIO::IOMMU\_MMIO\_PASID\_MATCH\_BANK\_1\_CNT\_2\_1,

- IOMMUMMIO::IOMMU\_MMIO\_PASID\_MATCH\_BANK\_1\_CNT\_3\_0 and  
IOMMUMMIO::IOMMU\_MMIO\_PASID\_MATCH\_BANK\_1\_CNT\_3\_1.

Bit positions above the value reported in

IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_CONFIG\_0[N\_COUNTER\_BANKS] are ignored when Written and return zero when Read. The counter banks are numbered starting with 32 (See IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_CONFIG\_0[N\_COUNTER\_BANKS]), are ignored when Written and return zero when Read. The counter banks are numbered starting with 32; PASID\_LOCK\_HI[0] controls bank 32, etc.

#### ValidValues:

Bit	Description
[0]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).
[1]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).
[2]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).
[3]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).
[4]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).
[5]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).
[6]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).
[7]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).
[8]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).
[9]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).
[10]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).
[11]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).
[12]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).
[13]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).
[14]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).
[15]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).
[16]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).
[17]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).
[18]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).
[19]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).
[20]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).
[21]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).
[22]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).
[23]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).
[24]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).
[25]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).
[26]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).
[27]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).
[28]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).
[29]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).
[30]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).
[31]	0=Corresponding counter bank of PASID-match registers is unlocked. 1=Locked (Writes are ignored).

#### IOMMUBARx04010 (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_DOMAIN\_BANK\_LOCK\_0)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx04010; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],

IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO] , 19'h0_0000}																	
_aliasSMN; IOMMUMMIOx00004010; IOMMUMMIO=0240_0000h																	
Bits	Description																
31:0	<p><b>DOMAIN_LOCK_LO.</b> Read-write. Reset: 0000_0000h.</p> <p><b>Description:</b> Domain lock enable. For each bit in DOMAIN_LOCK_LO, the corresponding Domain-match registers in an IOMMU counter bank may be changed. See IOMMU Domain Match Registers</p> <ul style="list-style-type: none"> <li>- IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_0_CNT_0_0 and IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_0_CNT_0_1,</li> <li>- IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_0_CNT_1_0 and IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_0_CNT_1_1,</li> <li>- IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_0_CNT_2_0 and IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_0_CNT_2_1,</li> <li>- IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_0_CNT_3_0 and IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_0_CNT_3_1,</li> <li>- IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_1_CNT_0_0 and IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_1_CNT_0_1,</li> <li>- IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_1_CNT_1_0 and IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_1_CNT_1_1,</li> <li>- IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_1_CNT_2_0 and IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_1_CNT_2_1,</li> <li>- IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_1_CNT_3_0 and IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_1_CNT_3_1.</li> </ul> <p>Bit positions above the value reported in IOMMUMMIO::IOMMU_MMIO_COUNTER_CONFIG_0[N_COUNTER_BANKS] are ignored when Written and return zero when Read. The counter banks are numbered starting with zero (See IOMMUMMIO::IOMMU_MMIO_COUNTER_CONFIG_0[N_COUNTER_BANKS]), are ignored when Written and return zero when Read. The counter banks are numbered starting with zero; DOMAIN_LOCK_LO[0] controls bank 0, etc.</p> <p><b>ValidValues:</b></p> <table> <tr> <th>Bit</th><th>Description</th></tr> <tr> <td>[0]</td><td>0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).</td></tr> <tr> <td>[1]</td><td>0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).</td></tr> <tr> <td>[2]</td><td>0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).</td></tr> <tr> <td>[3]</td><td>0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).</td></tr> <tr> <td>[4]</td><td>0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).</td></tr> <tr> <td>[5]</td><td>0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).</td></tr> <tr> <td>[6]</td><td>0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).</td></tr> </table>	Bit	Description	[0]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).	[1]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).	[2]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).	[3]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).	[4]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).	[5]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).	[6]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).
Bit	Description																
[0]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).																
[1]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).																
[2]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).																
[3]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).																
[4]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).																
[5]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).																
[6]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).																

	ignored).
[7]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).
[8]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).
[9]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).
[10]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).
[11]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).
[12]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).
[13]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).
[14]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).
[15]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).
[16]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).
[17]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).
[18]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).
[19]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).
[20]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).
[21]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).
[22]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).
[23]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).
[24]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).
[25]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).
[26]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).
[27]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).
[28]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).
[29]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).
[30]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).
[31]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are



ignored).

**IOMMUBARx04014 (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_DOMAIN\_BANK\_LOCK\_1)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx04014; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI] ,  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO] , 19'h0\_0000}  
\_aliasSMN; IOMMUMMIOx00004014; IOMMUMMIO=0240\_0000h

Bits	Description												
31:0	<p><b>DOMAIN_LOCK_HI.</b> Read-write. Reset: 0000_0000h.</p> <p><b>Description:</b> Domain lock enable. For each bit in DOMAIN_LOCK_HI, the corresponding Domain-match registers in an IOMMU counter bank may be changed. See IOMMU Domain Match Registers</p> <ul style="list-style-type: none"> <li>- IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_0_CNT_0_0 and IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_0_CNT_0_1,</li> <li>- IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_0_CNT_1_0 and IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_0_CNT_1_1,</li> <li>- IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_0_CNT_2_0 and IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_0_CNT_2_1,</li> <li>- IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_0_CNT_3_0 and IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_0_CNT_3_1,</li> <li>- IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_1_CNT_0_0 and IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_1_CNT_0_1,</li> <li>- IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_1_CNT_1_0 and IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_1_CNT_1_1,</li> <li>- IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_1_CNT_2_0 and IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_1_CNT_2_1,</li> <li>- IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_1_CNT_3_0 and IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_1_CNT_3_1.</li> </ul> <p>Bit positions above the value reported in IOMMUMMIO::IOMMU_MMIO_COUNTER_CONFIG_0[N_COUNTER_BANKS] are ignored when Written and return zero when Read. The counter banks are numbered starting with 32 (See IOMMUMMIO::IOMMU_MMIO_COUNTER_CONFIG_0[N_COUNTER_BANKS]), are ignored when Written and return zero when Read. The counter banks are numbered starting with 32; DOMAIN_LOCK_HI[0] controls bank 32, etc.</p> <p><b>Valid Values:</b></p> <table> <tr> <th>Bit</th><th>Description</th></tr> <tr> <td>[0]</td><td>0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).</td></tr> <tr> <td>[1]</td><td>0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).</td></tr> <tr> <td>[2]</td><td>0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).</td></tr> <tr> <td>[3]</td><td>0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).</td></tr> <tr> <td>[4]</td><td>0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).</td></tr> </table>	Bit	Description	[0]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).	[1]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).	[2]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).	[3]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).	[4]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).
Bit	Description												
[0]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).												
[1]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).												
[2]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).												
[3]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).												
[4]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).												



	ignored).
[5]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).
[6]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).
[7]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).
[8]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).
[9]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).
[10]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).
[11]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).
[12]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).
[13]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).
[14]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).
[15]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).
[16]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).
[17]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).
[18]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).
[19]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).
[20]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).
[21]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).
[22]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).
[23]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).
[24]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).
[25]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).
[26]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).
[27]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).
[28]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).
[29]	0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).

		ignored).
[30]		0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).
[31]		0=Corresponding counter bank of Domain-match registers is unlocked. 1=Locked (Writes are ignored).

### IOMMUBARx04018 (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_DEVID\_LOCK\_0)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx04018; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI], IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx00004018; IOMMUMMIO=0240\_0000h

Bits	Description								
31:0	<p><b>DEVID_LOCK_LO.</b> Read-write. Reset: 0000_0000h.</p> <p><b>Description:</b> DeviceID lock enable. For each bit in DEVID_LOCK_LO, the corresponding DeviceID-match registers in an IOMMU counter bank may be changed. See IOMMU DeviceID Match Registers</p> <ul style="list-style-type: none"> <li>- IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_0_CNT_0_0 and IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_0_CNT_0_1,</li> <li>- IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_0_CNT_1_0 and IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_0_CNT_1_1,</li> <li>- IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_0_CNT_2_0 and IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_0_CNT_2_1,</li> <li>- IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_0_CNT_3_0 and IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_0_CNT_3_1,</li> <li>- IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_1_CNT_0_0 and IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_1_CNT_0_1,</li> <li>- IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_1_CNT_1_0 and IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_1_CNT_1_1,</li> <li>- IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_1_CNT_2_0 and IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_1_CNT_2_1,</li> <li>- IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_1_CNT_3_0 and IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_1_CNT_3_1.</li> </ul> <p>Bit positions above the value reported in IOMMUMMIO::IOMMU_MMIO_COUNTER_CONFIG_0[N_COUNTER_BANKS] are ignored when Written and return zero when Read. The counter banks are numbered starting with zero (See IOMMUMMIO::IOMMU_MMIO_COUNTER_CONFIG_0[N_COUNTER_BANKS]), are ignored when Written and return zero when Read. The counter banks are numbered starting with zero; DEVID_LOCK_LO[0] controls bank 0, etc.</p> <p><b>Valid Values:</b></p> <table> <tr> <th>Bit</th><th>Description</th></tr> <tr> <td>[0]</td><td>0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).</td></tr> <tr> <td>[1]</td><td>0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).</td></tr> <tr> <td>[2]</td><td>0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).</td></tr> </table>	Bit	Description	[0]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).	[1]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).	[2]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).
Bit	Description								
[0]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).								
[1]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).								
[2]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).								

	ignored).
[3]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).
[4]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).
[5]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).
[6]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).
[7]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).
[8]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).
[9]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).
[10]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).
[11]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).
[12]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).
[13]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).
[14]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).
[15]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).
[16]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).
[17]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).
[18]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).
[19]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).
[20]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).
[21]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).
[22]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).
[23]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).
[24]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).
[25]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).
[26]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).
[27]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are

		ignored).
[28]		0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).
[29]		0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).
[30]		0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).
[31]		0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).

### IOMMUBARx0401C (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_DEVID\_LOCK\_1)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx0401C; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}  
\_aliasSMN; IOMMUMMIOx0000401C; IOMMUMMIO=0240\_0000h

Bits	Description				
31:0	<p><b>DEVID_LOCK_HI.</b> Read-write. Reset: 0000_0000h.</p> <p><b>Description:</b> DeviceID lock enable. For each bit in DEVID_LOCK_HI, the corresponding DeviceID-match registers in an IOMMU counter bank may be changed. See IOMMU DeviceID Match Registers</p> <ul style="list-style-type: none"> <li>- IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_0_CNT_0_0 and IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_0_CNT_0_1,</li> <li>- IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_0_CNT_1_0 and IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_0_CNT_1_1,</li> <li>- IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_0_CNT_2_0 and IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_0_CNT_2_1,</li> <li>- IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_0_CNT_3_0 and IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_0_CNT_3_1,</li> <li>- IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_1_CNT_0_0 and IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_1_CNT_0_1,</li> <li>- IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_1_CNT_1_0 and IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_1_CNT_1_1,</li> <li>- IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_1_CNT_2_0 and IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_1_CNT_2_1,</li> <li>- IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_1_CNT_3_0 and IOMMUMMIO::IOMMU_MMIO_DOMAIN_MATCH_BANK_1_CNT_3_1.</li> </ul> <p>Bit positions above the value reported in IOMMUMMIO::IOMMU_MMIO_COUNTER_CONFIG_0[N_COUNTER_BANKS] are ignored when Written and return zero when Read. The counter banks are numbered starting with 32 (See IOMMUMMIO::IOMMU_MMIO_COUNTER_CONFIG_0[N_COUNTER_BANKS]), are ignored when Written and return zero when Read. The counter banks are numbered starting with 32; DEVID_LOCK_HI[0] controls bank 32, etc.</p> <p><b>ValidValues:</b></p> <table> <tr> <th>Bit</th><th>Description</th></tr> <tr> <td>[0]</td><td>0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are</td></tr> </table>	Bit	Description	[0]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are
Bit	Description				
[0]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are				

	ignored).
[1]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).
[2]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).
[3]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).
[4]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).
[5]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).
[6]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).
[7]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).
[8]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).
[9]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).
[10]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).
[11]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).
[12]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).
[13]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).
[14]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).
[15]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).
[16]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).
[17]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).
[18]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).
[19]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).
[20]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).
[21]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).
[22]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).
[23]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).
[24]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).
[25]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are

		ignored).
[26]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).	
[27]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).	
[28]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).	
[29]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).	
[30]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).	
[31]	0=Corresponding counter bank of DeviceID-match registers is unlocked. 1=Locked (Writes are ignored).	

**IOMMUBARx40000 (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_BANK\_0\_CNT\_0\_0)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx40000; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI], IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx00040000; IOMMUMMIO=0240\_0000h

Bits	Description
31:0	<b>ICOUNTER_0_0_LO.</b> Read-write. Reset: 0000_0000h. Reports the counter value (bits[31:0]). The counter counts up continuously, wrapping at the maximum value. There is no overflow indicator.

**IOMMUBARx40004 (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_BANK\_0\_CNT\_0\_1)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx40004; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI], IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx00040004; IOMMUMMIO=0240\_0000h

Bits	Description
31:16	Reserved.
15:0	<b>ICOUNTER_0_0_HI.</b> Read-write. Reset: 0000h. Reports the counter value (bits[47:32]). The counter counts up continuously, wrapping at the maximum value. There is no overflow indicator. See also IOMMUMMIO::IOMMU_MMIO_COUNTER_BANK_0_CNT_0_0.

**IOMMUBARx40008 (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_SRC\_BANK\_0\_CNT\_0\_0)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx40008; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI], IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx00040008; IOMMUMMIO=0240\_0000h

Bits	Description
31	<b>CAC_0_0.</b> Read-write. Reset: 0. 0=Architectural counter input group. 1=Custom input group. Counter source architectural or custom.
30	<b>COUNT_UNITS_0_0.</b> Read-write. Reset: 0. 0=Counter counts events (level). 1=Counter counts clocks (edges). Count Units.
29:8	Reserved.
7:0	<b>CSOURCE_0_0.</b> Read-write. Reset: 00h. Counter source. Selects event counter input from the choices provided.

**IOMMUBARx4000C (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_SRC\_BANK\_0\_CNT\_0\_1)**

Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx4000C; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI], IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx0004000C; IOMMUMMIO=0240\_0000h

Bits	Description
31:0	Reserved.

**IOMMUBARx40010 (IOMMUMMIO::IOMMU\_MMIO\_PASID\_MATCH\_BANK\_0\_CNT\_0\_0)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx40010; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}  
\_aliasSMN; IOMMUMMIOx00040010; IOMMUMMIO=0240\_0000h

Bits	Description
31	<b>PASMEN_0_0.</b> Read-write. Reset: 0. 0=PASID is ignored. 1=Filtered PASID must match to count an event. PASID Match enable. An event with no PASID tag is only counted when PASMEN_0_0 == 0.
30:16	Reserved.
15:0	<b>PASID_MATCH_0_0.</b> Read-write. Reset: 0000h. PASID match. This value is compared with the masked (filtered) value of the incoming PASID of the transaction to decide to count the corresponding event. The event is counted if PASID_MATCH_0_0 is exactly equal to the masked incoming PASID. The event is not counted if they are not equal.

**IOMMUBARx40014 (IOMMUMMIO::IOMMU\_MMIO\_PASID\_MATCH\_BANK\_0\_CNT\_0\_1)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx40014; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}  
\_aliasSMN; IOMMUMMIOx00040014; IOMMUMMIO=0240\_0000h

Bits	Description
31:16	Reserved.
15:0	<b>PASID_MASK_0_0.</b> Read-write. Reset: 0000h. PASID Mask. This bit-mask is ANDed with the PASID of the transaction to decide to count the corresponding event.
<b>ValidValues:</b>	
<b>Value</b>	<b>Description</b>
0000h	Count events for all values of incoming PASID.
FFFFh-0001h	Bit-wise mask ANDed with incoming PASID.

**IOMMUBARx40018 (IOMMUMMIO::IOMMU\_MMIO\_DOMAIN\_MATCH\_BANK\_0\_CNT\_0\_0)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx40018; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}  
\_aliasSMN; IOMMUMMIOx00040018; IOMMUMMIO=0240\_0000h

Bits	Description
31	<b>DOMMEN_0_0.</b> Read-write. Reset: 0. 0=Domain is ignored. 1=Filtered Domain must match to count an event. Domain Match enable.
30:16	Reserved.
15:0	<b>DOMAIN_MATCH_0_0.</b> Read-write. Reset: 0000h. Domain Match. This value is compared with the masked (filtered) value of the incoming Domain of the transaction to decide to count the corresponding event. The event is counted if DOMAIN_MATCH_0_0 is exactly equal to the masked incoming Domain. The event is not counted if they are not equal.

**IOMMUBARx4001C (IOMMUMMIO::IOMMU\_MMIO\_DOMAIN\_MATCH\_BANK\_0\_CNT\_0\_1)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx4001C; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}  
\_aliasSMN; IOMMUMMIOx0004001C; IOMMUMMIO=0240\_0000h

Bits	Description
31:16	Reserved.
15:0	<b>DOMAIN_MASK_0_0.</b> Read-write. Reset: 0000h. Domain Mask. This bit-mask is ANDed with the Domain of the transaction to decide to count the corresponding event.
<b>ValidValues:</b>	
<b>Value</b>	<b>Description</b>



0000h	Count events for all values of incoming Domain.
FFFFh-0001h	Bit-wise mask ANDed with incoming Domain.

#### IOMMUBARx40020 (IOMMUMMIO::IOMMU\_MMIO\_DEVICEID\_MATCH\_BANK\_0\_CNT\_0\_0)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOmmUBARx40020; IOmmUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOmmUMMIOx00040020; IOmmUMMIO=0240\_0000h

Bits	Description
31	<b>DIDMEN_0_0.</b> Read-write. Reset: 0. 0=DeviceID is ignored. 1=Filtered DeviceID must match to count an event. DeviceID Match enable.
30:16	Reserved.
15:0	<b>DEVICEID_MATCH_0_0.</b> Read-write. Reset: 0000h. DeviceID Match. This value is compared with the masked (filtered) value of the incoming DeviceID of the transaction to decide to count the corresponding event. The event is counted if DEVICEID_MATCH_0_0 is exactly equal to the masked incoming DeviceID. The event is not counted if they are not equal.

#### IOmmUBARx40024 (IOMMUMMIO::IOMMU\_MMIO\_DEVICEID\_MATCH\_BANK\_0\_CNT\_0\_1)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOmmUBARx40024; IOmmUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOmmUMMIOx00040024; IOmmUMMIO=0240\_0000h

Bits	Description						
31:16	Reserved.						
15:0	<b>DEVICEID_MASK_0_0.</b> Read-write. Reset: 0000h. DeviceID Mask. This bit-mask is ANDed with the DeviceID of the transaction to decide to count the corresponding event.						
<b>Valid Values:</b>							
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0000h</td><td>Count events for all values of incoming DeviceID.</td></tr> <tr> <td>FFFFh-0001h</td><td>Bit-wise mask ANDed with incoming DeviceID.</td></tr> </table>	Value	Description	0000h	Count events for all values of incoming DeviceID.	FFFFh-0001h	Bit-wise mask ANDed with incoming DeviceID.
Value	Description						
0000h	Count events for all values of incoming DeviceID.						
FFFFh-0001h	Bit-wise mask ANDed with incoming DeviceID.						

#### IOmmUBARx40028 (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_RPT\_BANK\_0\_CNT\_0\_0)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOmmUBARx40028; IOmmUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOmmUMMIOx00040028; IOmmUMMIO=0240\_0000h

Bits	Description
31:0	<b>EVENT_NOTE_0_0_LO.</b> Read-write. Reset: 0000_0000h. Event Note (bits[31:0]). When IOmmUMMIO::IOMMU_MMIO_COUNTER_RPT_BANK_0_CNT_0_1[CERE_0_0] == 1 and the corresponding counter is incremented and wraps to zero, EVENT_NOTE_0_0[51:0] is reported in the EVENT_COUNTER_ZERO event log entry. EVENT_NOTE_0_0[51:0] = {EVENT_NOTE_0_0_LO, IOmmUMMIO::IOMMU_MMIO_COUNTER_RPT_BANK_0_CNT_0_1[EVENT_NOTE_0_0_HI]}

#### IOmmUBARx4002C (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_RPT\_BANK\_0\_CNT\_0\_1)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOmmUBARx4002C; IOmmUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOmmUMMIOx0004002C; IOmmUMMIO=0240\_0000h

Bits	Description
31	<b>CERE_0_0.</b> Read-write. Reset: 0. 0=No event report when counter wraps to zero. 1=IOMMU Writes an EVENT_COUNTER_ZERO event log entry when the counter wraps to zero.

	<b>Description:</b> Counter Event Report Enable. The counter-wrap event is treated like any other event. <b>SOFTWARE NOTE:</b> The counter-wrap event is delivered promptly but there is no defined latency expectation.
30:20	Reserved.
19:0	<b>EVENT_NOTE_0_0_HI.</b> Read-write. Reset: 0_0000h. Event Note (bits[51:32]). When CERE_0_0 == 1 and the corresponding counter is incremented and wraps to zero, EVENT_NOTE_0_0[51:0] is reported in the EVENT_COUNTER_ZERO event log entry. See also IOMMUMMIO::IOMMU_MMIO_COUNTER_RPT_BANK_0_CNT_0_0.

**IOMMUBARx40100 (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_BANK\_0\_CNT\_1\_0)**

Read-write. Reset: 0000_0000h.	
_aliasHOST; IOMMUBARx40100; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI], IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO], 19'h0_0000}	
_aliasSMN; IOMMUMMIOx00040100; IOMMUMMIO=0240_0000h	
Bits	Description
31:0	<b>ICOUNTER_0_1_LO.</b> Read-write. Reset: 0000_0000h. Reports the counter value (bits[31:0]). The counter counts up continuously, wrapping at the maximum value. There is no overflow indicator.

**IOMMUBARx40104 (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_BANK\_0\_CNT\_1\_1)**

Read-write. Reset: 0000_0000h.	
_aliasHOST; IOMMUBARx40104; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI], IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO], 19'h0_0000}	
_aliasSMN; IOMMUMMIOx00040104; IOMMUMMIO=0240_0000h	
Bits	Description
31:16	Reserved.
15:0	<b>ICOUNTER_0_1_HI.</b> Read-write. Reset: 0000h. Reports the counter value (bits[47:32]). The counter counts up continuously, wrapping at the maximum value. There is no overflow indicator. See also IOMMUMMIO::IOMMU_MMIO_COUNTER_BANK_0_CNT_1_0.

**IOMMUBARx40108 (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_SRC\_BANK\_0\_CNT\_1\_0)**

Read-write. Reset: 0000_0000h.	
_aliasHOST; IOMMUBARx40108; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI], IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO], 19'h0_0000}	
_aliasSMN; IOMMUMMIOx00040108; IOMMUMMIO=0240_0000h	
Bits	Description
31	<b>CAC_0_1.</b> Read-write. Reset: 0. 0=Architectural counter input group. 1=Custom input group. Counter source architectural or custom.
30	<b>COUNT_UNITS_0_1.</b> Read-write. Reset: 0. 0=Counter counts events (level). 1=Counter counts clocks (edges). Count Units.
29:8	Reserved.
7:0	<b>CSOURCE_0_1.</b> Read-write. Reset: 00h. Counter source. Selects event counter input from the choices provided.

**IOMMUBARx4010C (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_SRC\_BANK\_0\_CNT\_1\_1)**

Reset: 0000_0000h.	
_aliasHOST; IOMMUBARx4010C; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI], IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO], 19'h0_0000}	
_aliasSMN; IOMMUMMIOx0004010C; IOMMUMMIO=0240_0000h	
Bits	Description
31:0	Reserved.

**IOMMUBARx40110 (IOMMUMMIO::IOMMU\_MMIO\_PASID\_MATCH\_BANK\_0\_CNT\_1\_0)**

Read-write. Reset: 0000_0000h.	
_aliasHOST; IOMMUBARx40110; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI], IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO], 19'h0_0000}	
_aliasSMN; IOMMUMMIOx00040110; IOMMUMMIO=0240_0000h	
Bits	Description

31	<b>PASMEN_0_1.</b> Read-write. Reset: 0. 0=PASID is ignored. 1=Filtered PASID must match to count an event. PASID Match enable. An event with no PASID tag is only counted when PASMEN_0_1 == 0.
30:16	Reserved.
15:0	<b>PASID_MATCH_0_1.</b> Read-write. Reset: 0000h. PASID match. This value is compared with the masked (filtered) value of the incoming PASID of the transaction to decide to count the corresponding event. The event is counted if PASID_MATCH_0_1 is exactly equal to the masked incoming PASID. The event is not counted if they are not equal.

#### IO MMUBARx40114 (IO MMUMMIO::IO MMU\_MMIO\_PASID\_MATCH\_BANK\_0\_CNT\_1\_1)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IO MMUBARx40114; IO MMUBAR={IO MMUL2::IO MMU\_CAP\_BASE\_HI\_aliasHOST[IO MMU\_BASE\_ADDR\_HI] , IO MMUL2::IO MMU\_CAP\_BASE\_LO\_aliasHOST[IO MMU\_BASE\_ADDR\_LO] , 19'h0\_0000}

\_aliasSMN; IO MMUMMIOx00040114; IO MMUMMIO=0240\_0000h

Bits	Description						
31:16	Reserved.						
15:0	<b>PASID_MASK_0_1.</b> Read-write. Reset: 0000h. PASID Mask. This bit-mask is ANDed with the PASID of the transaction to decide to count the corresponding event. <b>ValidValues:</b>						
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0000h</td><td>Count events for all values of incoming PASID.</td></tr> <tr> <td>FFFFh-0001h</td><td>Bit-wise mask ANDed with incoming PASID.</td></tr> </table>	Value	Description	0000h	Count events for all values of incoming PASID.	FFFFh-0001h	Bit-wise mask ANDed with incoming PASID.
Value	Description						
0000h	Count events for all values of incoming PASID.						
FFFFh-0001h	Bit-wise mask ANDed with incoming PASID.						

#### IO MMUBARx40118 (IO MMUMMIO::IO MMU\_MMIO\_DOMAIN\_MATCH\_BANK\_0\_CNT\_1\_0)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IO MMUBARx40118; IO MMUBAR={IO MMUL2::IO MMU\_CAP\_BASE\_HI\_aliasHOST[IO MMU\_BASE\_ADDR\_HI] , IO MMUL2::IO MMU\_CAP\_BASE\_LO\_aliasHOST[IO MMU\_BASE\_ADDR\_LO] , 19'h0\_0000}

\_aliasSMN; IO MMUMMIOx00040118; IO MMUMMIO=0240\_0000h

Bits	Description
31	<b>DOMMEN_0_1.</b> Read-write. Reset: 0. 0=Domain is ignored. 1=Filtered Domain must match to count an event. Domain Match enable.
30:16	Reserved.
15:0	<b>DOMAIN_MATCH_0_1.</b> Read-write. Reset: 0000h. Domain Match. This value is compared with the masked (filtered) value of the incoming Domain of the transaction to decide to count the corresponding event. The event is counted if DOMAIN_MATCH_0_1 is exactly equal to the masked incoming Domain. The event is not counted if they are not equal.

#### IO MMUBARx4011C (IO MMUMMIO::IO MMU\_MMIO\_DOMAIN\_MATCH\_BANK\_0\_CNT\_1\_1)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IO MMUBARx4011C; IO MMUBAR={IO MMUL2::IO MMU\_CAP\_BASE\_HI\_aliasHOST[IO MMU\_BASE\_ADDR\_HI] , IO MMUL2::IO MMU\_CAP\_BASE\_LO\_aliasHOST[IO MMU\_BASE\_ADDR\_LO] , 19'h0\_0000}

\_aliasSMN; IO MMUMMIOx0004011C; IO MMUMMIO=0240\_0000h

Bits	Description						
31:16	Reserved.						
15:0	<b>DOMAIN_MASK_0_1.</b> Read-write. Reset: 0000h. Domain Mask. This bit-mask is ANDed with the Domain of the transaction to decide to count the corresponding event. <b>ValidValues:</b>						
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0000h</td><td>Count events for all values of incoming Domain.</td></tr> <tr> <td>FFFFh-0001h</td><td>Bit-wise mask ANDed with incoming Domain.</td></tr> </table>	Value	Description	0000h	Count events for all values of incoming Domain.	FFFFh-0001h	Bit-wise mask ANDed with incoming Domain.
Value	Description						
0000h	Count events for all values of incoming Domain.						
FFFFh-0001h	Bit-wise mask ANDed with incoming Domain.						

#### IO MMUBARx40120 (IO MMUMMIO::IO MMU\_MMIO\_DEVICEID\_MATCH\_BANK\_0\_CNT\_1\_0)

Read-write. Reset: 0000_0000h.	
_aliasHOST; IOMMUBARx40120; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI], IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO], 19'h0_0000}	
_aliasSMN; IOMMUMMIOx00040120; IOMMUMMIO=0240_0000h	
Bits	Description
31	<b>DIDMEN_0_1.</b> Read-write. Reset: 0. 0=DeviceID is ignored. 1=Filtered DeviceID must match to count an event. DeviceID Match enable.
30:16	Reserved.
15:0	<b>DEVICEID_MATCH_0_1.</b> Read-write. Reset: 0000h. DeviceID Match. This value is compared with the masked (filtered) value of the incoming DeviceID of the transaction to decide to count the corresponding event. The event is counted if DEVICEID_MATCH_0_1 is exactly equal to the masked incoming DeviceID. The event is not counted if they are not equal.

#### IOMMUBARx40124 (IOMMUMMIO::IOMMU\_MMIO\_DEVICEID\_MATCH\_BANK\_0\_CNT\_1\_1)

Read-write. Reset: 0000_0000h.	
_aliasHOST; IOMMUBARx40124; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI], IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO], 19'h0_0000}	
_aliasSMN; IOMMUMMIOx00040124; IOMMUMMIO=0240_0000h	
Bits	Description
31:16	Reserved.
15:0	<b>DEVICEID_MASK_0_1.</b> Read-write. Reset: 0000h. DeviceID Mask. This bit-mask is ANDed with the DeviceID of the transaction to decide to count the corresponding event.
<b>Valid Values:</b>	
Value	Description
0000h	Count events for all values of incoming DeviceID.
FFFFh-0001h	Bit-wise mask ANDed with incoming DeviceID.

#### IOMMUBARx40128 (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_RPT\_BANK\_0\_CNT\_1\_0)

Read-write. Reset: 0000_0000h.	
_aliasHOST; IOMMUBARx40128; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI], IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO], 19'h0_0000}	
_aliasSMN; IOMMUMMIOx00040128; IOMMUMMIO=0240_0000h	
Bits	Description
31:0	<b>EVENT_NOTE_0_1_LO.</b> Read-write. Reset: 0000_0000h. Event Note (bits[31:0]). When IOMMUMMIO::IOMMU_MMIO_COUNTER_RPT_BANK_0_CNT_1_1[CERE_0_1] == 1 and the corresponding counter is incremented and wraps to zero, EVENT_NOTE_0_1[51:0] is reported in the EVENT_COUNTER_ZERO event log entry. EVENT_NOTE_0_1[51:0] = {EVENT_NOTE_0_1_LO, IOMMUMMIO::IOMMU_MMIO_COUNTER_RPT_BANK_0_CNT_1_1[EVENT_NOTE_0_1_HI]}

#### IOMMUBARx4012C (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_RPT\_BANK\_0\_CNT\_1\_1)

Read-write. Reset: 0000_0000h.	
_aliasHOST; IOMMUBARx4012C; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI], IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO], 19'h0_0000}	
_aliasSMN; IOMMUMMIOx0004012C; IOMMUMMIO=0240_0000h	
Bits	Description
31	<b>CERE_0_1.</b> Read-write. Reset: 0. 0=No event report when counter wraps to zero. 1=IOMMU Writes an EVENT_COUNTER_ZERO event log entry when the counter wraps to zero. <b>Description:</b> Counter Event Report Enable. The counter-wrap event is treated like any other event. SOFTWARE NOTE: The counter-wrap event is delivered promptly but there is no defined latency expectation.
30:20	Reserved.
19:0	<b>EVENT_NOTE_0_1_HI.</b> Read-write. Reset: 0_0000h. Event Note (bits[51:32]). When CERE_0_1 == 1 and the corresponding counter is incremented and wraps to zero, EVENT_NOTE_0_1[51:0] is reported in the

	EVENT_COUNTER_ZERO event log entry. See also IOMMUMMIO::IOMMU_MMIO_COUNTER_RPT_BANK_0_CNT_1_0.
--	---

**IOMMUBARx40200 (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_BANK\_0\_CNT\_2\_0)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx40200; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx00040200; IOMMUMMIO=0240\_0000h

Bits	Description
31:0	<b>ICOUNTER_0_2_LO</b> . Read-write. Reset: 0000_0000h. Reports the counter value (bits[31:0]). The counter counts up continuously, wrapping at the maximum value. There is no overflow indicator.

**IOMMUBARx40204 (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_BANK\_0\_CNT\_2\_1)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx40204; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx00040204; IOMMUMMIO=0240\_0000h

Bits	Description
31:16	Reserved.
15:0	<b>ICOUNTER_0_2_HI</b> . Read-write. Reset: 0000h. Reports the counter value (bits[47:32]). The counter counts up continuously, wrapping at the maximum value. There is no overflow indicator. See also IOMMUMMIO::IOMMU_MMIO_COUNTER_BANK_0_CNT_2_0.

**IOMMUBARx40208 (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_SRC\_BANK\_0\_CNT\_2\_0)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx40208; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx00040208; IOMMUMMIO=0240\_0000h

Bits	Description
31	<b>CAC_0_2</b> . Read-write. Reset: 0. 0=Architectural counter input group. 1=Custom input group. Counter source architectural or custom.
30	<b>COUNT_UNITS_0_2</b> . Read-write. Reset: 0. 0=Counter counts events (level). 1=Counter counts clocks (edges). Count Units.
29:8	Reserved.
7:0	<b>CSOURCE_0_2</b> . Read-write. Reset: 00h. Counter source. Selects event counter input from the choices provided.

**IOMMUBARx4020C (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_SRC\_BANK\_0\_CNT\_2\_1)**

Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx4020C; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx0004020C; IOMMUMMIO=0240\_0000h

Bits	Description
31:0	Reserved.

**IOMMUBARx40210 (IOMMUMMIO::IOMMU\_MMIO\_PASID\_MATCH\_BANK\_0\_CNT\_2\_0)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx40210; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx00040210; IOMMUMMIO=0240\_0000h

Bits	Description
31	<b>PASMEN_0_2</b> . Read-write. Reset: 0. 0=PASID is ignored. 1=Filtered PASID must match to count an event. PASID Match enable. An event with no PASID tag is only counted when PASMEN_0_2 == 0.
30:16	Reserved.
15:0	<b>PASID_MATCH_0_2</b> . Read-write. Reset: 0000h. PASID match. This value is compared with the masked (filtered) value of the incoming PASID of the transaction to decide to count the corresponding event. The event is

	counted if PASID_MATCH_0_2 is exactly equal to the masked incoming PASID. The event is not counted if they are not equal.
--	---

#### IOMMUBARx40214 (IOMMUMMIO::IOMMU\_MMIO\_PASID\_MATCH\_BANK\_0\_CNT\_2\_1)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOmmUBARx40214; IOmmUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}  
\_aliasSMN; IOmmUMMIOx00040214; IOmmUMMIO=0240\_0000h

Bits	Description
31:16	Reserved.
15:0	<b>PASID_MASK_0_2.</b> Read-write. Reset: 0000h. PASID Mask. This bit-mask is ANDed with the PASID of the transaction to decide to count the corresponding event.
<b>Valid Values:</b>	
Value	Description
0000h	Count events for all values of incoming PASID.
FFFFh-0001h	Bit-wise mask ANDed with incoming PASID.

#### IOMMUBARx40218 (IOMMUMMIO::IOMMU\_MMIO\_DOMAIN\_MATCH\_BANK\_0\_CNT\_2\_0)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOmmUBARx40218; IOmmUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}  
\_aliasSMN; IOmmUMMIOx00040218; IOmmUMMIO=0240\_0000h

Bits	Description
31	<b>DOMMEN_0_2.</b> Read-write. Reset: 0. 0=Domain is ignored. 1=Filtered Domain must match to count an event. Domain Match enable.
30:16	Reserved.
15:0	<b>DOMAIN_MATCH_0_2.</b> Read-write. Reset: 0000h. Domain Match. This value is compared with the masked (filtered) value of the incoming Domain of the transaction to decide to count the corresponding event. The event is counted if DOMAIN_MATCH_0_2 is exactly equal to the masked incoming Domain. The event is not counted if they are not equal.

#### IOMMUBARx4021C (IOMMUMMIO::IOMMU\_MMIO\_DOMAIN\_MATCH\_BANK\_0\_CNT\_2\_1)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOmmUBARx4021C; IOmmUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}  
\_aliasSMN; IOmmUMMIOx0004021C; IOmmUMMIO=0240\_0000h

Bits	Description
31:16	Reserved.
15:0	<b>DOMAIN_MASK_0_2.</b> Read-write. Reset: 0000h. Domain Mask. This bit-mask is ANDed with the Domain of the transaction to decide to count the corresponding event.
<b>Valid Values:</b>	
Value	Description
0000h	Count events for all values of incoming Domain.
FFFFh-0001h	Bit-wise mask ANDed with incoming Domain.

#### IOMMUBARx40220 (IOMMUMMIO::IOMMU\_MMIO\_DEVICEID\_MATCH\_BANK\_0\_CNT\_2\_0)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOmmUBARx40220; IOmmUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}  
\_aliasSMN; IOmmUMMIOx00040220; IOmmUMMIO=0240\_0000h

Bits	Description
31	<b>DIDMEN_0_2.</b> Read-write. Reset: 0. 0=DeviceID is ignored. 1=Filtered DeviceID must match to count an event.



	DeviceID Match enable.
30:16	Reserved.
15:0	<b>DEVICEID_MATCH_0_2.</b> Read-write. Reset: 0000h. DeviceID Match. This value is compared with the masked (filtered) value of the incoming DeviceID of the transaction to decide to count the corresponding event. The event is counted if DEVICEID_MATCH_0_2 is exactly equal to the masked incoming DeviceID. The event is not counted if they are not equal.

#### IOMMUBARx40224 (IOMMUMMIO::IOMMU\_MMIO\_DEVICEID\_MATCH\_BANK\_0\_CNT\_2\_1)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOmmubarx40224; IOmmubar={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx00040224; IOMMUMMIO=0240\_0000h

Bits	Description						
31:16	Reserved.						
15:0	<b>DEVICEID_MASK_0_2.</b> Read-write. Reset: 0000h. DeviceID Mask. This bit-mask is ANDed with the DeviceID of the transaction to decide to count the corresponding event.						
	<b>ValidValues:</b>						
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0000h</td><td>Count events for all values of incoming DeviceID.</td></tr> <tr> <td>FFFFh-0001h</td><td>Bit-wise mask ANDed with incoming DeviceID.</td></tr> </table>	Value	Description	0000h	Count events for all values of incoming DeviceID.	FFFFh-0001h	Bit-wise mask ANDed with incoming DeviceID.
Value	Description						
0000h	Count events for all values of incoming DeviceID.						
FFFFh-0001h	Bit-wise mask ANDed with incoming DeviceID.						

#### IOMMUBARx40228 (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_RPT\_BANK\_0\_CNT\_2\_0)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOmmubarx40228; IOmmubar={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx00040228; IOMMUMMIO=0240\_0000h

Bits	Description
31:0	<b>EVENT_NOTE_0_2_LO.</b> Read-write. Reset: 0000_0000h. Event Note (bits[31:0]). When IOMMUMMIO::IOMMU_MMIO_COUNTER_RPT_BANK_0_CNT_2_1[CERE_0_2] == 1 and the corresponding counter is incremented and wraps to zero, EVENT_NOTE_0_2[51:0] is reported in the EVENT_COUNTER_ZERO event log entry. EVENT_NOTE_0_2[51:0] = {EVENT_NOTE_0_2_LO, IOMMUMMIO::IOMMU_MMIO_COUNTER_RPT_BANK_0_CNT_2_1[EVENT_NOTE_0_2_HI]}

#### IOMMUBARx4022C (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_RPT\_BANK\_0\_CNT\_2\_1)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOmmubarx4022C; IOmmubar={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx0004022C; IOMMUMMIO=0240\_0000h

Bits	Description
31	<b>CERE_0_2.</b> Read-write. Reset: 0. 0=No event report when counter wraps to zero. 1=IOMMU Writes an EVENT_COUNTER_ZERO event log entry when the counter wraps to zero. <b>Description:</b> Counter Event Report Enable. The counter-wrap event is treated like any other event. <b>SOFTWARE NOTE:</b> The counter-wrap event is delivered promptly but there is no defined latency expectation.
30:20	Reserved.
19:0	<b>EVENT_NOTE_0_2_HI.</b> Read-write. Reset: 0_0000h. Event Note (bits[51:32]). When CERE_0_2 == 1 and the corresponding counter is incremented and wraps to zero, EVENT_NOTE_0_2[51:0] is reported in the EVENT_COUNTER_ZERO event log entry. See also IOMMUMMIO::IOMMU_MMIO_COUNTER_RPT_BANK_0_CNT_2_0.

#### IOMMUBARx40300 (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_BANK\_0\_CNT\_3\_0)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOmmubarx40300; IOmmubar={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],



IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO] , 19'h0_0000} _aliasSMN; IOMMUMMIOx00040300; IOMMUMMIO=0240_0000h	
Bits	Description
31:0	<b>ICOUNTER_0_3_LO</b> . Read-write. Reset: 0000_0000h. Reports the counter value (bits[31:0]). The counter counts up continuously, wrapping at the maximum value. There is no overflow indicator.

**IOMMUBARx40304 (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_BANK\_0\_CNT\_3\_1)**

Read-write. Reset: 0000_0000h. _aliasHOST; IOMMUBARx40304; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI] , IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO] , 19'h0_0000} _aliasSMN; IOMMUMMIOx00040304; IOMMUMMIO=0240_0000h	
Bits	Description
31:16	Reserved.
15:0	<b>ICOUNTER_0_3_HI</b> . Read-write. Reset: 0000h. Reports the counter value (bits[47:32]). The counter counts up continuously, wrapping at the maximum value. There is no overflow indicator. See also IOMMUMMIO::IOMMU_MMIO_COUNTER_BANK_0_CNT_3_0.

**IOMMUBARx40308 (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_SRC\_BANK\_0\_CNT\_3\_0)**

Read-write. Reset: 0000_0000h. _aliasHOST; IOMMUBARx40308; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI] , IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO] , 19'h0_0000} _aliasSMN; IOMMUMMIOx00040308; IOMMUMMIO=0240_0000h	
Bits	Description
31	<b>CAC_0_3</b> . Read-write. Reset: 0. 0=Architectural counter input group. 1=Custom input group. Counter source architectural or custom.
30	<b>COUNT_UNITS_0_3</b> . Read-write. Reset: 0. 0=Counter counts events (level). 1=Counter counts clocks (edges). Count Units.
29:8	Reserved.
7:0	<b>CSOURCE_0_3</b> . Read-write. Reset: 00h. Counter source. Selects event counter input from the choices provided.

**IOMMUBARx4030C (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_SRC\_BANK\_0\_CNT\_3\_1)**

Reset: 0000_0000h. _aliasHOST; IOMMUBARx4030C; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI] , IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO] , 19'h0_0000} _aliasSMN; IOMMUMMIOx0004030C; IOMMUMMIO=0240_0000h	
Bits	Description
31:0	Reserved.

**IOMMUBARx40310 (IOMMUMMIO::IOMMU\_MMIO\_PASID\_MATCH\_BANK\_0\_CNT\_3\_0)**

Read-write. Reset: 0000_0000h. _aliasHOST; IOMMUBARx40310; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI] , IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO] , 19'h0_0000} _aliasSMN; IOMMUMMIOx00040310; IOMMUMMIO=0240_0000h	
Bits	Description
31	<b>PASMEN_0_3</b> . Read-write. Reset: 0. 0=PASID is ignored. 1=Filtered PASID must match to count an event. PASID Match enable. An event with no PASID tag is only counted when PASMEN_0_3 == 0.
30:16	Reserved.
15:0	<b>PASID_MATCH_0_3</b> . Read-write. Reset: 0000h. PASID match. This value is compared with the masked (filtered) value of the incoming PASID of the transaction to decide to count the corresponding event. The event is counted if PASID_MATCH_0_3 is exactly equal to the masked incoming PASID. The event is not counted if they are not equal.

**IOMMUBARx40314 (IOMMUMMIO::IOMMU\_MMIO\_PASID\_MATCH\_BANK\_0\_CNT\_3\_1)**

Read-write. Reset: 0000_0000h. _aliasHOST; IOMMUBARx40314; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI] ,	
--	--

IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO] , 19'h0_0000} _aliasSMN; IOMMUMMIOx00040314; IOMMUMMIO=0240_0000h	
Bits	Description
31:16	Reserved.
15:0	<b>PASID_MASK_0_3.</b> Read-write. Reset: 0000h. PASID Mask. This bit-mask is ANDed with the PASID of the transaction to decide to count the corresponding event.
<b>ValidValues:</b>	
<b>Value</b>	<b>Description</b>
0000h	Count events for all values of incoming PASID.
FFFFh-0001h	Bit-wise mask ANDed with incoming PASID.

#### IOMMUBARx40318 (IOMMUMMIO::IOMMU\_MMIO\_DOMAIN\_MATCH\_BANK\_0\_CNT\_3\_0)

Read-write. Reset: 0000_0000h. _aliasHOST; IOMMUBARx40318; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI] , IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO] , 19'h0_0000} _aliasSMN; IOMMUMMIOx00040318; IOMMUMMIO=0240_0000h	
Bits	Description
31	<b>DOMMEN_0_3.</b> Read-write. Reset: 0. 0=Domain is ignored. 1=Filtered Domain must match to count an event. Domain Match enable.
30:16	Reserved.
15:0	<b>DOMAIN_MATCH_0_3.</b> Read-write. Reset: 0000h. Domain Match. This value is compared with the masked (filtered) value of the incoming Domain of the transaction to decide to count the corresponding event. The event is counted if DOMAIN_MATCH_0_3 is exactly equal to the masked incoming Domain. The event is not counted if they are not equal.

#### IOMMUBARx4031C (IOMMUMMIO::IOMMU\_MMIO\_DOMAIN\_MATCH\_BANK\_0\_CNT\_3\_1)

Read-write. Reset: 0000_0000h. _aliasHOST; IOMMUBARx4031C; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI] , IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO] , 19'h0_0000} _aliasSMN; IOMMUMMIOx0004031C; IOMMUMMIO=0240_0000h	
Bits	Description
31:16	Reserved.
15:0	<b>DOMAIN_MASK_0_3.</b> Read-write. Reset: 0000h. Domain Mask. This bit-mask is ANDed with the Domain of the transaction to decide to count the corresponding event.
<b>ValidValues:</b>	
<b>Value</b>	<b>Description</b>
0000h	Count events for all values of incoming Domain.
FFFFh-0001h	Bit-wise mask ANDed with incoming Domain.

#### IOMMUBARx40320 (IOMMUMMIO::IOMMU\_MMIO\_DEVICEID\_MATCH\_BANK\_0\_CNT\_3\_0)

Read-write. Reset: 0000_0000h. _aliasHOST; IOMMUBARx40320; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI] , IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO] , 19'h0_0000} _aliasSMN; IOMMUMMIOx00040320; IOMMUMMIO=0240_0000h	
Bits	Description
31	<b>DIDMEN_0_3.</b> Read-write. Reset: 0. 0=DeviceID is ignored. 1=Filtered DeviceID must match to count an event. DeviceID Match enable.
30:16	Reserved.
15:0	<b>DEVICEID_MATCH_0_3.</b> Read-write. Reset: 0000h. DeviceID Match. This value is compared with the masked (filtered) value of the incoming DeviceID of the transaction to decide to count the corresponding event. The event is counted if DEVICEID_MATCH_0_3 is exactly equal to the masked incoming DeviceID. The event is not

	counted if they are not equal.
--	--------------------------------

### IOMMUBARx40324 (IOMMUMMIO::IOMMU\_MMIO\_DEVICEID\_MATCH\_BANK\_0\_CNT\_3\_1)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx40324; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx00040324; IOMMUMMIO=0240\_0000h

Bits	Description						
31:16	Reserved.						
15:0	<b>DEVICEID_MASK_0_3.</b> Read-write. Reset: 0000h. DeviceID Mask. This bit-mask is ANDed with the DeviceID of the transaction to decide to count the corresponding event. <b>Valid Values:</b>						
	<table border="1"> <thead> <tr> <th>Value</th><th>Description</th></tr> </thead> <tbody> <tr> <td>0000h</td><td>Count events for all values of incoming DeviceID.</td></tr> <tr> <td>FFFFh-0001h</td><td>Bit-wise mask ANDed with incoming DeviceID.</td></tr> </tbody> </table>	Value	Description	0000h	Count events for all values of incoming DeviceID.	FFFFh-0001h	Bit-wise mask ANDed with incoming DeviceID.
Value	Description						
0000h	Count events for all values of incoming DeviceID.						
FFFFh-0001h	Bit-wise mask ANDed with incoming DeviceID.						

### IOMMUBARx40328 (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_RPT\_BANK\_0\_CNT\_3\_0)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx40328; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx00040328; IOMMUMMIO=0240\_0000h

Bits	Description
31:0	<b>EVENT_NOTE_0_3_LO.</b> Read-write. Reset: 0000_0000h. Event Note (bits[31:0]). When IOMMUMMIO::IOMMU_MMIO_COUNTER_RPT_BANK_0_CNT_3_1[CERE_0_3] == 1 and the corresponding counter is incremented and wraps to zero, EVENT_NOTE_0_3[51:0] is reported in the EVENT_COUNTER_ZERO event log entry. EVENT_NOTE_0_3[51:0] = {EVENT_NOTE_0_3_LO, IOMMUMMIO::IOMMU_MMIO_COUNTER_RPT_BANK_0_CNT_3_1[EVENT_NOTE_0_3_HI]}

### IOMMUBARx4032C (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_RPT\_BANK\_0\_CNT\_3\_1)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx4032C; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx0004032C; IOMMUMMIO=0240\_0000h

Bits	Description
31	<b>CERE_0_3.</b> Read-write. Reset: 0. 0=No event report when counter wraps to zero. 1=IOMMU Writes an EVENT_COUNTER_ZERO event log entry when the counter wraps to zero. <b>Description:</b> Counter Event Report Enable. The counter-wrap event is treated like any other event. <b>SOFTWARE NOTE:</b> The counter-wrap event is delivered promptly but there is no defined latency expectation.
30:20	Reserved.
19:0	<b>EVENT_NOTE_0_3_HI.</b> Read-write. Reset: 0_0000h. Event Note (bits[51:32]). When CERE_0_3 == 1 and the corresponding counter is incremented and wraps to zero, EVENT_NOTE_0_3[51:0] is reported in the EVENT_COUNTER_ZERO event log entry. See also IOMMUMMIO::IOMMU_MMIO_COUNTER_RPT_BANK_0_CNT_3_0.

### IOMMUBARx41000 (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_BANK\_1\_CNT\_0\_0)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx41000; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx00041000; IOMMUMMIO=0240\_0000h

Bits	Description
31:0	<b>ICOUNTER_1_0_LO.</b> Read-write. Reset: 0000_0000h. Reports the counter value (bits[31:0]). The counter counts up continuously, wrapping at the maximum value. There is no overflow indicator.

**IOMMUBARx41004 (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_BANK\_1\_CNT\_0\_1)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx41004; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}  
\_aliasSMN; IOMMUMMIOx00041004; IOMMUMMIO=0240\_0000h

Bits	Description
31:16	Reserved.
15:0	<b>ICOUNTER_1_0_HI</b> . Read-write. Reset: 0000h. Reports the counter value (bits[47:32]). The counter counts up continuously, wrapping at the maximum value. There is no overflow indicator. See also IOMMUMMIO::IOMMU_MMIO_COUNTER_BANK_1_CNT_0_0.

**IOMMUBARx41008 (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_SRC\_BANK\_1\_CNT\_0\_0)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx41008; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}  
\_aliasSMN; IOMMUMMIOx00041008; IOMMUMMIO=0240\_0000h

Bits	Description
31	<b>CAC_1_0</b> . Read-write. Reset: 0. 0=Architectural counter input group. 1=Custom input group. Counter source architectural or custom.
30	<b>COUNT_UNITS_1_0</b> . Read-write. Reset: 0. 0=Counter counts events (level). 1=Counter counts clocks (edges). Count Units.
29:8	Reserved.
7:0	<b>CSOURCE_1_0</b> . Read-write. Reset: 00h. Counter source. Selects event counter input from the choices provided.

**IOMMUBARx4100C (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_SRC\_BANK\_1\_CNT\_0\_1)**

Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx4100C; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}  
\_aliasSMN; IOMMUMMIOx0004100C; IOMMUMMIO=0240\_0000h

Bits	Description
31:0	Reserved.

**IOMMUBARx41010 (IOMMUMMIO::IOMMU\_MMIO\_PASID\_MATCH\_BANK\_1\_CNT\_0\_0)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx41010; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}  
\_aliasSMN; IOMMUMMIOx00041010; IOMMUMMIO=0240\_0000h

Bits	Description
31	<b>PASMEN_1_0</b> . Read-write. Reset: 0. 0=PASID is ignored. 1=Filtered PASID must match to count an event. PASID Match enable. An event with no PASID tag is only counted when PASMEN_1_0 == 0.
30:16	Reserved.
15:0	<b>PASID_MATCH_1_0</b> . Read-write. Reset: 0000h. PASID match. This value is compared with the masked (filtered) value of the incoming PASID of the transaction to decide to count the corresponding event. The event is counted if PASID_MATCH_1_0 is exactly equal to the masked incoming PASID. The event is not counted if they are not equal.

**IOMMUBARx41014 (IOMMUMMIO::IOMMU\_MMIO\_PASID\_MATCH\_BANK\_1\_CNT\_0\_1)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx41014; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}  
\_aliasSMN; IOMMUMMIOx00041014; IOMMUMMIO=0240\_0000h

Bits	Description
31:16	Reserved.
15:0	<b>PASID_MASK_1_0</b> . Read-write. Reset: 0000h. PASID Mask. This bit-mask is ANDed with the PASID of the transaction to decide to count the corresponding event.

	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0000h	Count events for all values of incoming PASID.
	FFFFh-0001h	Bit-wise mask ANDed with incoming PASID.

#### IOMMUBARx41018 (IOMMUMMIO::IOMMU\_MMIO\_DOMAIN\_MATCH\_BANK\_1\_CNT\_0\_0)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOmmUBARx41018; IOmmUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOmmUMMIOx00041018; IOmmUMMIO=0240\_0000h

Bits	Description
31	<b>DOMMEN_1_0</b> . Read-write. Reset: 0. 0=Domain is ignored. 1=Filtered Domain must match to count an event. Domain Match enable.
30:16	Reserved.
15:0	<b>DOMAIN_MATCH_1_0</b> . Read-write. Reset: 0000h. Domain Match. This value is compared with the masked (filtered) value of the incoming Domain of the transaction to decide to count the corresponding event. The event is counted if DOMAIN_MATCH_1_0 is exactly equal to the masked incoming Domain. The event is not counted if they are not equal.

#### IOMMUBARx4101C (IOMMUMMIO::IOMMU\_MMIO\_DOMAIN\_MATCH\_BANK\_1\_CNT\_0\_1)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOmmUBARx4101C; IOmmUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOmmUMMIOx0004101C; IOmmUMMIO=0240\_0000h

Bits	Description						
31:16	Reserved.						
15:0	<b>DOMAIN_MASK_1_0</b> . Read-write. Reset: 0000h. Domain Mask. This bit-mask is ANDed with the Domain of the transaction to decide to count the corresponding event.						
	<b>ValidValues:</b>						
	<table><tr><th>Value</th><th>Description</th></tr><tr><td>0000h</td><td>Count events for all values of incoming Domain.</td></tr><tr><td>FFFFh-0001h</td><td>Bit-wise mask ANDed with incoming Domain.</td></tr></table>	Value	Description	0000h	Count events for all values of incoming Domain.	FFFFh-0001h	Bit-wise mask ANDed with incoming Domain.
	Value	Description					
0000h	Count events for all values of incoming Domain.						
FFFFh-0001h	Bit-wise mask ANDed with incoming Domain.						

#### IOMMUBARx41020 (IOMMUMMIO::IOMMU\_MMIO\_DEVICEID\_MATCH\_BANK\_1\_CNT\_0\_0)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOmmUBARx41020; IOmmUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOmmUMMIOx00041020; IOmmUMMIO=0240\_0000h

Bits	Description
31	<b>DIDMEN_1_0</b> . Read-write. Reset: 0. 0=DeviceID is ignored. 1=Filtered DeviceID must match to count an event. DeviceID Match enable.
30:16	Reserved.
15:0	<b>DEVICEID_MATCH_1_0</b> . Read-write. Reset: 0000h. DeviceID Match. This value is compared with the masked (filtered) value of the incoming DeviceID of the transaction to decide to count the corresponding event. The event is counted if DEVICEID_MATCH_1_0 is exactly equal to the masked incoming DeviceID. The event is not counted if they are not equal.

#### IOMMUBARx41024 (IOMMUMMIO::IOMMU\_MMIO\_DEVICEID\_MATCH\_BANK\_1\_CNT\_0\_1)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOmmUBARx41024; IOmmUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOmmUMMIOx00041024; IOmmUMMIO=0240\_0000h

Bits	Description						
31:16	Reserved.						
15:0	<b>DEVICEID_MASK_1_0.</b> Read-write. Reset: 0000h. DeviceID Mask. This bit-mask is ANDed with the DeviceID of the transaction to decide to count the corresponding event.						
	<b>Valid Values:</b>						
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0000h</td><td>Count events for all values of incoming DeviceID.</td></tr> <tr> <td>FFFFh-0001h</td><td>Bit-wise mask ANDed with incoming DeviceID.</td></tr> </table>	Value	Description	0000h	Count events for all values of incoming DeviceID.	FFFFh-0001h	Bit-wise mask ANDed with incoming DeviceID.
Value	Description						
0000h	Count events for all values of incoming DeviceID.						
FFFFh-0001h	Bit-wise mask ANDed with incoming DeviceID.						

**IOMMUBARx41028 (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_RPT\_BANK\_1\_CNT\_0\_0)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx41028; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}  
\_aliasSMN; IOMMUMMIOx00041028; IOMMUMMIO=0240\_0000h

Bits	Description
31:0	<b>EVENT_NOTE_1_0_LO.</b> Read-write. Reset: 0000_0000h. Event Note (bits[31:0]). When IOMMUMMIO::IOMMU_MMIO_COUNTER_RPT_BANK_1_CNT_0_1[CERE_1_0] == 1 and the corresponding counter is incremented and wraps to zero, EVENT_NOTE_1_0[51:0] is reported in the EVENT_COUNTER_ZERO event log entry. EVENT_NOTE_1_0[51:0] = {EVENT_NOTE_1_0_LO, IOMMUMMIO::IOMMU_MMIO_COUNTER_RPT_BANK_1_CNT_0_1[EVENT_NOTE_1_0_HI]}

**IOMMUBARx4102C (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_RPT\_BANK\_1\_CNT\_0\_1)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx4102C; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}  
\_aliasSMN; IOMMUMMIOx0004102C; IOMMUMMIO=0240\_0000h

Bits	Description
31	<b>CERE_1_0.</b> Read-write. Reset: 0. 0=No event report when counter wraps to zero. 1=IOMMU Writes an EVENT_COUNTER_ZERO event log entry when the counter wraps to zero. <b>Description:</b> Counter Event Report Enable. The counter-wrap event is treated like any other event. SOFTWARE NOTE: The counter-wrap event is delivered promptly but there is no defined latency expectation.
30:20	Reserved.
19:0	<b>EVENT_NOTE_1_0_HI.</b> Read-write. Reset: 0_0000h. Event Note (bits[51:32]). When CERE_1_0 == 1 and the corresponding counter is incremented and wraps to zero, EVENT_NOTE_1_0[51:0] is reported in the EVENT_COUNTER_ZERO event log entry. See also IOMMUMMIO::IOMMU_MMIO_COUNTER_RPT_BANK_1_CNT_0_0.

**IOMMUBARx41100 (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_BANK\_1\_CNT\_1\_0)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx41100; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}  
\_aliasSMN; IOMMUMMIOx00041100; IOMMUMMIO=0240\_0000h

Bits	Description
31:0	<b>ICOUNTER_1_1_LO.</b> Read-write. Reset: 0000_0000h. Reports the counter value (bits[31:0]). The counter counts up continuously, wrapping at the maximum value. There is no overflow indicator.

**IOMMUBARx41104 (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_BANK\_1\_CNT\_1\_1)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx41104; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}  
\_aliasSMN; IOMMUMMIOx00041104; IOMMUMMIO=0240\_0000h

Bits	Description
------	-------------



31:16	Reserved.
15:0	<b>ICOUNTER_1_1_HI</b> . Read-write. Reset: 0000h. Reports the counter value (bits[47:32]). The counter counts up continuously, wrapping at the maximum value. There is no overflow indicator. See also IOMMUMMIO::IOMMU_MMIO_COUNTER_BANK_1_CNT_1_0.

#### IOMMUBARx41108 (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_SRC\_BANK\_1\_CNT\_1\_0)

Read-write. Reset: 0000_0000h.	
_aliasHOST; IOMMUBARx41108; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI], IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO], 19'h0_0000}	
_aliasSMN; IOMMUMMIOx00041108; IOMMUMMIO=0240_0000h	
Bits	Description
31	<b>CAC_1_1</b> . Read-write. Reset: 0. 0=Architectural counter input group. 1=Custom input group. Counter source architectural or custom.
30	<b>COUNT_UNITS_1_1</b> . Read-write. Reset: 0. 0=Counter counts events (level). 1=Counter counts clocks (edges). Count Units.
29:8	Reserved.
7:0	<b>CSOURCE_1_1</b> . Read-write. Reset: 00h. Counter source. Selects event counter input from the choices provided.

#### IOMMUBARx4110C (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_SRC\_BANK\_1\_CNT\_1\_1)

Reset: 0000_0000h.	
_aliasHOST; IOMMUBARx4110C; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI], IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO], 19'h0_0000}	
_aliasSMN; IOMMUMMIOx0004110C; IOMMUMMIO=0240_0000h	
Bits	Description
31:0	Reserved.

#### IOMMUBARx41110 (IOMMUMMIO::IOMMU\_MMIO\_PASID\_MATCH\_BANK\_1\_CNT\_1\_0)

Read-write. Reset: 0000_0000h.	
_aliasHOST; IOMMUBARx41110; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI], IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO], 19'h0_0000}	
_aliasSMN; IOMMUMMIOx00041110; IOMMUMMIO=0240_0000h	
Bits	Description
31	<b>PASMEN_1_1</b> . Read-write. Reset: 0. 0=PASID is ignored. 1=Filtered PASID must match to count an event. PASID Match enable. An event with no PASID tag is only counted when PASMEN_1_1 == 0.
30:16	Reserved.
15:0	<b>PASID_MATCH_1_1</b> . Read-write. Reset: 0000h. PASID match. This value is compared with the masked (filtered) value of the incoming PASID of the transaction to decide to count the corresponding event. The event is counted if PASID_MATCH_1_1 is exactly equal to the masked incoming PASID. The event is not counted if they are not equal.

#### IOMMUBARx41114 (IOMMUMMIO::IOMMU\_MMIO\_PASID\_MATCH\_BANK\_1\_CNT\_1\_1)

Read-write. Reset: 0000_0000h.	
_aliasHOST; IOMMUBARx41114; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI], IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO], 19'h0_0000}	
_aliasSMN; IOMMUMMIOx00041114; IOMMUMMIO=0240_0000h	
Bits	Description
31:16	Reserved.
15:0	<b>PASID_MASK_1_1</b> . Read-write. Reset: 0000h. PASID Mask. This bit-mask is ANDed with the PASID of the transaction to decide to count the corresponding event.
<b>Valid Values:</b>	
Value	Description
0000h	Count events for all values of incoming PASID.
FFFFh-0001h	Bit-wise mask ANDed with incoming PASID.



**IOMMUBARx41118 (IOMMUMMIO::IOMMU\_MMIO\_DOMAIN\_MATCH\_BANK\_1\_CNT\_1\_0)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx41118; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}  
\_aliasSMN; IOMMUMMIOx00041118; IOMMUMMIO=0240\_0000h

Bits	Description
31	<b>DOMMEN_1_1.</b> Read-write. Reset: 0. 0=Domain is ignored. 1=Filtered Domain must match to count an event. Domain Match enable.
30:16	Reserved.
15:0	<b>DOMAIN_MATCH_1_1.</b> Read-write. Reset: 0000h. Domain Match. This value is compared with the masked (filtered) value of the incoming Domain of the transaction to decide to count the corresponding event. The event is counted if DOMAIN_MATCH_1_1 is exactly equal to the masked incoming Domain. The event is not counted if they are not equal.

**IOMMUBARx4111C (IOMMUMMIO::IOMMU\_MMIO\_DOMAIN\_MATCH\_BANK\_1\_CNT\_1\_1)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx4111C; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}  
\_aliasSMN; IOMMUMMIOx0004111C; IOMMUMMIO=0240\_0000h

Bits	Description						
31:16	Reserved.						
15:0	<b>DOMAIN_MASK_1_1.</b> Read-write. Reset: 0000h. Domain Mask. This bit-mask is ANDed with the Domain of the transaction to decide to count the corresponding event.						
	<b>ValidValues:</b>						
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0000h</td><td>Count events for all values of incoming Domain.</td></tr> <tr> <td>FFFFh-0001h</td><td>Bit-wise mask ANDed with incoming Domain.</td></tr> </table>	Value	Description	0000h	Count events for all values of incoming Domain.	FFFFh-0001h	Bit-wise mask ANDed with incoming Domain.
Value	Description						
0000h	Count events for all values of incoming Domain.						
FFFFh-0001h	Bit-wise mask ANDed with incoming Domain.						

**IOMMUBARx41120 (IOMMUMMIO::IOMMU\_MMIO\_DEVICEID\_MATCH\_BANK\_1\_CNT\_1\_0)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx41120; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}  
\_aliasSMN; IOMMUMMIOx00041120; IOMMUMMIO=0240\_0000h

Bits	Description
31	<b>DIDMEN_1_1.</b> Read-write. Reset: 0. 0=DeviceID is ignored. 1=Filtered DeviceID must match to count an event. DeviceID Match enable.
30:16	Reserved.
15:0	<b>DEVICEID_MATCH_1_1.</b> Read-write. Reset: 0000h. DeviceID Match. This value is compared with the masked (filtered) value of the incoming DeviceID of the transaction to decide to count the corresponding event. The event is counted if DEVICEID_MATCH_1_1 is exactly equal to the masked incoming DeviceID. The event is not counted if they are not equal.

**IOMMUBARx41124 (IOMMUMMIO::IOMMU\_MMIO\_DEVICEID\_MATCH\_BANK\_1\_CNT\_1\_1)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx41124; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}  
\_aliasSMN; IOMMUMMIOx00041124; IOMMUMMIO=0240\_0000h

Bits	Description		
31:16	Reserved.		
15:0	<b>DEVICEID_MASK_1_1.</b> Read-write. Reset: 0000h. DeviceID Mask. This bit-mask is ANDed with the DeviceID of the transaction to decide to count the corresponding event.		
	<b>ValidValues:</b>		
	<table> <tr> <th>Value</th><th>Description</th></tr> </table>	Value	Description
Value	Description		

0000h	Count events for all values of incoming DeviceID.
FFFFh-0001h	Bit-wise mask ANDed with incoming DeviceID.

**IOMMUBARx41128 (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_RPT\_BANK\_1\_CNT\_1\_0)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx41128; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI], IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx00041128; IOMMUMMIO=0240\_0000h

Bits	Description
31:0	<b>EVENT_NOTE_1_1_LO.</b> Read-write. Reset: 0000_0000h. Event Note (bits[31:0]). When IOMMUMMIO::IOMMU_MMIO_COUNTER_RPT_BANK_1_CNT_1_1[CERE_1_1] == 1 and the corresponding counter is incremented and wraps to zero, EVENT_NOTE_1_1[51:0] is reported in the EVENT_COUNTER_ZERO event log entry. EVENT_NOTE_1_1[51:0] = {EVENT_NOTE_1_1_LO, IOMMUMMIO::IOMMU_MMIO_COUNTER_RPT_BANK_1_CNT_1_1[EVENT_NOTE_1_1_HI]}

**IOMMUBARx4112C (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_RPT\_BANK\_1\_CNT\_1\_1)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx4112C; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI], IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx0004112C; IOMMUMMIO=0240\_0000h

Bits	Description
31	<b>CERE_1_1.</b> Read-write. Reset: 0. 0=No event report when counter wraps to zero. 1=IOMMU Writes an EVENT_COUNTER_ZERO event log entry when the counter wraps to zero. <b>Description:</b> Counter Event Report Enable. The counter-wrap event is treated like any other event. SOFTWARE NOTE: The counter-wrap event is delivered promptly but there is no defined latency expectation.
30:20	Reserved.
19:0	<b>EVENT_NOTE_1_1_HI.</b> Read-write. Reset: 0_0000h. Event Note (bits[51:32]). When CERE_1_1 == 1 and the corresponding counter is incremented and wraps to zero, EVENT_NOTE_1_1[51:0] is reported in the EVENT_COUNTER_ZERO event log entry. See also IOMMUMMIO::IOMMU_MMIO_COUNTER_RPT_BANK_1_CNT_1_0.

**IOMMUBARx41200 (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_BANK\_1\_CNT\_2\_0)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx41200; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI], IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx00041200; IOMMUMMIO=0240\_0000h

Bits	Description
31:0	<b>ICOUNTER_1_2_LO.</b> Read-write. Reset: 0000_0000h. Reports the counter value (bits[31:0]). The counter counts up continuously, wrapping at the maximum value. There is no overflow indicator.

**IOMMUBARx41204 (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_BANK\_1\_CNT\_2\_1)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx41204; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI], IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx00041204; IOMMUMMIO=0240\_0000h

Bits	Description
31:16	Reserved.
15:0	<b>ICOUNTER_1_2_HI.</b> Read-write. Reset: 0000h. Reports the counter value (bits[47:32]). The counter counts up continuously, wrapping at the maximum value. There is no overflow indicator. See also IOMMUMMIO::IOMMU_MMIO_COUNTER_BANK_1_CNT_2_0.

**IOMMUBARx41208 (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_SRC\_BANK\_1\_CNT\_2\_0)**

Read-write. Reset: 0000\_0000h.

_aliasHOST; IOMMUBARx41208; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI] , IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO] , 19'h0_0000}	
_aliasSMN; IOMMUMMIOx00041208; IOMMUMMIO=0240_0000h	
Bits	Description
31	<b>CAC_1_2.</b> Read-write. Reset: 0. 0=Architectural counter input group. 1=Custom input group. Counter source architectural or custom.
30	<b>COUNT_UNITS_1_2.</b> Read-write. Reset: 0. 0=Counter counts events (level). 1=Counter counts clocks (edges). Count Units.
29:8	Reserved.
7:0	<b>CSOURCE_1_2.</b> Read-write. Reset: 00h. Counter source. Selects event counter input from the choices provided.

#### IOMMUBARx4120C (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_SRC\_BANK\_1\_CNT\_2\_1)

Reset: 0000_0000h.	
_aliasHOST; IOMMUBARx4120C; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI] , IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO] , 19'h0_0000}	
_aliasSMN; IOMMUMMIOx0004120C; IOMMUMMIO=0240_0000h	
Bits	Description
31:0	Reserved.

#### IOMMUBARx41210 (IOMMUMMIO::IOMMU\_MMIO\_PASID\_MATCH\_BANK\_1\_CNT\_2\_0)

Read-write. Reset: 0000_0000h.	
_aliasHOST; IOMMUBARx41210; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI] , IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO] , 19'h0_0000}	
_aliasSMN; IOMMUMMIOx00041210; IOMMUMMIO=0240_0000h	
Bits	Description
31	<b>PASMEN_1_2.</b> Read-write. Reset: 0. 0=PASID is ignored. 1=Filtered PASID must match to count an event. PASID Match enable. An event with no PASID tag is only counted when PASMEN_1_2 == 0.
30:16	Reserved.
15:0	<b>PASID_MATCH_1_2.</b> Read-write. Reset: 0000h. PASID match. This value is compared with the masked (filtered) value of the incoming PASID of the transaction to decide to count the corresponding event. The event is counted if PASID_MATCH_1_2 is exactly equal to the masked incoming PASID. The event is not counted if they are not equal.

#### IOMMUBARx41214 (IOMMUMMIO::IOMMU\_MMIO\_PASID\_MATCH\_BANK\_1\_CNT\_2\_1)

Read-write. Reset: 0000_0000h.	
_aliasHOST; IOMMUBARx41214; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI] , IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO] , 19'h0_0000}	
_aliasSMN; IOMMUMMIOx00041214; IOMMUMMIO=0240_0000h	
Bits	Description
31:16	Reserved.
15:0	<b>PASID_MASK_1_2.</b> Read-write. Reset: 0000h. PASID Mask. This bit-mask is ANDed with the PASID of the transaction to decide to count the corresponding event.
<b>Valid Values:</b>	
Value	Description
0000h	Count events for all values of incoming PASID.
FFFFh-0001h	Bit-wise mask ANDed with incoming PASID.

#### IOMMUBARx41218 (IOMMUMMIO::IOMMU\_MMIO\_DOMAIN\_MATCH\_BANK\_1\_CNT\_2\_0)

Read-write. Reset: 0000_0000h.	
_aliasHOST; IOMMUBARx41218; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI] , IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO] , 19'h0_0000}	
_aliasSMN; IOMMUMMIOx00041218; IOMMUMMIO=0240_0000h	
Bits	Description
31	<b>DOMMEN_1_2.</b> Read-write. Reset: 0. 0=Domain is ignored. 1=Filtered Domain must match to count an event.

	Domain Match enable.
30:16	Reserved.
15:0	<b>DOMAIN_MATCH_1_2.</b> Read-write. Reset: 0000h. Domain Match. This value is compared with the masked (filtered) value of the incoming Domain of the transaction to decide to count the corresponding event. The event is counted if DOMAIN_MATCH_1_2 is exactly equal to the masked incoming Domain. The event is not counted if they are not equal.

#### IOMMUBARx4121C (IOMMUMMIO::IOMMU\_MMIO\_DOMAIN\_MATCH\_BANK\_1\_CNT\_2\_1)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOmmubarx4121C; IOmmubar={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx0004121C; IOMMUMMIO=0240\_0000h

Bits	Description
31:16	Reserved.
15:0	<b>DOMAIN_MASK_1_2.</b> Read-write. Reset: 0000h. Domain Mask. This bit-mask is ANDed with the Domain of the transaction to decide to count the corresponding event.
	<b>ValidValues:</b>
Value	Description
0000h	Count events for all values of incoming Domain.
FFFFh-0001h	Bit-wise mask ANDed with incoming Domain.

#### IOMMUBARx41220 (IOMMUMMIO::IOMMU\_MMIO\_DEVICEID\_MATCH\_BANK\_1\_CNT\_2\_0)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOmmubarx41220; IOmmubar={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx00041220; IOMMUMMIO=0240\_0000h

Bits	Description
31	<b>DIDMEN_1_2.</b> Read-write. Reset: 0. 0=DeviceID is ignored. 1=Filtered DeviceID must match to count an event. DeviceID Match enable.
30:16	Reserved.
15:0	<b>DEVICEID_MATCH_1_2.</b> Read-write. Reset: 0000h. DeviceID Match. This value is compared with the masked (filtered) value of the incoming DeviceID of the transaction to decide to count the corresponding event. The event is counted if DEVICEID_MATCH_1_2 is exactly equal to the masked incoming DeviceID. The event is not counted if they are not equal.

#### IOMMUBARx41224 (IOMMUMMIO::IOMMU\_MMIO\_DEVICEID\_MATCH\_BANK\_1\_CNT\_2\_1)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOmmubarx41224; IOmmubar={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx00041224; IOMMUMMIO=0240\_0000h

Bits	Description
31:16	Reserved.
15:0	<b>DEVICEID_MASK_1_2.</b> Read-write. Reset: 0000h. DeviceID Mask. This bit-mask is ANDed with the DeviceID of the transaction to decide to count the corresponding event.
	<b>ValidValues:</b>
Value	Description
0000h	Count events for all values of incoming DeviceID.
FFFFh-0001h	Bit-wise mask ANDed with incoming DeviceID.

#### IOMMUBARx41228 (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_RPT\_BANK\_1\_CNT\_2\_0)

Read-write. Reset: 0000\_0000h.

_aliasHOST; IOMMUBARx41228; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI], IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO], 19'h0_0000}	
_aliasSMN; IOMMUMMIOx00041228; IOMMUMMIO=0240_0000h	
Bits	Description
31:0	<b>EVENT_NOTE_1_2_LO.</b> Read-write. Reset: 0000_0000h. Event Note (bits[31:0]). When IOMMUMMIO::IOMMU_MMIO_COUNTER_RPT_BANK_1_CNT_2_1[CERE_1_2] == 1 and the corresponding counter is incremented and wraps to zero, EVENT_NOTE_1_2[51:0] is reported in the EVENT_COUNTER_ZERO event log entry. EVENT_NOTE_1_2[51:0] = {EVENT_NOTE_1_2_LO, IOMMUMMIO::IOMMU_MMIO_COUNTER_RPT_BANK_1_CNT_2_1[EVENT_NOTE_1_2_HI]}

#### IOMMUBARx4122C (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_RPT\_BANK\_1\_CNT\_2\_1)

Read-write. Reset: 0000_0000h.	
_aliasHOST; IOMMUBARx4122C; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI], IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO], 19'h0_0000}	
_aliasSMN; IOMMUMMIOx0004122C; IOMMUMMIO=0240_0000h	
Bits	Description
31	<b>CERE_1_2.</b> Read-write. Reset: 0. 0=No event report when counter wraps to zero. 1=IOMMU Writes an EVENT_COUNTER_ZERO event log entry when the counter wraps to zero. <b>Description:</b> Counter Event Report Enable. The counter-wrap event is treated like any other event. SOFTWARE NOTE: The counter-wrap event is delivered promptly but there is no defined latency expectation.
30:20	Reserved.
19:0	<b>EVENT_NOTE_1_2_HI.</b> Read-write. Reset: 0_0000h. Event Note (bits[51:32]). When CERE_1_2 == 1 and the corresponding counter is incremented and wraps to zero, EVENT_NOTE_1_2[51:0] is reported in the EVENT_COUNTER_ZERO event log entry. See also IOMMUMMIO::IOMMU_MMIO_COUNTER_RPT_BANK_1_CNT_2_0.

#### IOMMUBARx41300 (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_BANK\_1\_CNT\_3\_0)

Read-write. Reset: 0000_0000h.	
_aliasHOST; IOMMUBARx41300; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI], IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO], 19'h0_0000}	
_aliasSMN; IOMMUMMIOx00041300; IOMMUMMIO=0240_0000h	
Bits	Description
31:0	<b>ICOUNTER_1_3_LO.</b> Read-write. Reset: 0000_0000h. Reports the counter value (bits[31:0]). The counter counts up continuously, wrapping at the maximum value. There is no overflow indicator.

#### IOMMUBARx41304 (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_BANK\_1\_CNT\_3\_1)

Read-write. Reset: 0000_0000h.	
_aliasHOST; IOMMUBARx41304; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI], IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO], 19'h0_0000}	
_aliasSMN; IOMMUMMIOx00041304; IOMMUMMIO=0240_0000h	
Bits	Description
31:16	Reserved.
15:0	<b>ICOUNTER_1_3_HI.</b> Read-write. Reset: 0000h. Reports the counter value (bits[47:32]). The counter counts up continuously, wrapping at the maximum value. There is no overflow indicator. See also IOMMUMMIO::IOMMU_MMIO_COUNTER_BANK_1_CNT_3_0.

#### IOMMUBARx41308 (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_SRC\_BANK\_1\_CNT\_3\_0)

Read-write. Reset: 0000_0000h.	
_aliasHOST; IOMMUBARx41308; IOMMUBAR={IOMMUL2::IOMMU_CAP_BASE_HI_aliasHOST[IOMMU_BASE_ADDR_HI], IOMMUL2::IOMMU_CAP_BASE_LO_aliasHOST[IOMMU_BASE_ADDR_LO], 19'h0_0000}	
_aliasSMN; IOMMUMMIOx00041308; IOMMUMMIO=0240_0000h	
Bits	Description
31	<b>CAC_1_3.</b> Read-write. Reset: 0. 0=Architectural counter input group. 1=Custom input group. Counter source architectural or custom.

30	<b>COUNT_UNITS_1_3.</b> Read-write. Reset: 0. 0=Counter counts events (level). 1=Counter counts clocks (edges). Count Units.
29:8	Reserved.
7:0	<b>CSOURCE_1_3.</b> Read-write. Reset: 00h. Counter source. Selects event counter input from the choices provided.

#### IOMMUBARx4130C (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_SRC\_BANK\_1\_CNT\_3\_1)

Reset: 0000\_0000h.

\_aliasHOST; IOmmUBARx4130C; IOmmUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx0004130C; IOMMUMMIO=0240\_0000h

Bits	Description
31:0	Reserved.

#### IOMMUBARx41310 (IOMMUMMIO::IOMMU\_MMIO\_PASID\_MATCH\_BANK\_1\_CNT\_3\_0)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOmmUBARx41310; IOmmUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx00041310; IOMMUMMIO=0240\_0000h

Bits	Description
31	<b>PASMEN_1_3.</b> Read-write. Reset: 0. 0=PASID is ignored. 1=Filtered PASID must match to count an event. PASID Match enable. An event with no PASID tag is only counted when PASMEN_1_3 == 0.
30:16	Reserved.
15:0	<b>PASID_MATCH_1_3.</b> Read-write. Reset: 0000h. PASID match. This value is compared with the masked (filtered) value of the incoming PASID of the transaction to decide to count the corresponding event. The event is counted if PASID_MATCH_1_3 is exactly equal to the masked incoming PASID. The event is not counted if they are not equal.

#### IOMMUBARx41314 (IOMMUMMIO::IOMMU\_MMIO\_PASID\_MATCH\_BANK\_1\_CNT\_3\_1)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOmmUBARx41314; IOmmUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx00041314; IOMMUMMIO=0240\_0000h

Bits	Description
31:16	Reserved.
15:0	<b>PASID_MASK_1_3.</b> Read-write. Reset: 0000h. PASID Mask. This bit-mask is ANDed with the PASID of the transaction to decide to count the corresponding event.
<b>Valid Values:</b>	
Value	Description
0000h	Count events for all values of incoming PASID.
FFFFh-0001h	Bit-wise mask ANDed with incoming PASID.

#### IOMMUBARx41318 (IOMMUMMIO::IOMMU\_MMIO\_DOMAIN\_MATCH\_BANK\_1\_CNT\_3\_0)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOmmUBARx41318; IOmmUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx00041318; IOMMUMMIO=0240\_0000h

Bits	Description
31	<b>DOMMEN_1_3.</b> Read-write. Reset: 0. 0=Domain is ignored. 1=Filtered Domain must match to count an event. Domain Match enable.
30:16	Reserved.
15:0	<b>DOMAIN_MATCH_1_3.</b> Read-write. Reset: 0000h. Domain Match. This value is compared with the masked (filtered) value of the incoming Domain of the transaction to decide to count the corresponding event. The event is counted if DOMAIN_MATCH_1_3 is exactly equal to the masked incoming Domain. The event is not counted



	if they are not equal.
--	------------------------

**IOMMUBARx4131C (IOMMUMMIO::IOMMU\_MMIO\_DOMAIN\_MATCH\_BANK\_1\_CNT\_3\_1)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx4131C; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}  
\_aliasSMN; IOMMUMMIOx0004131C; IOMMUMMIO=0240\_0000h

Bits	Description						
31:16	Reserved.						
15:0	<b>DOMAIN_MASK_1_3.</b> Read-write. Reset: 0000h. Domain Mask. This bit-mask is ANDed with the Domain of the transaction to decide to count the corresponding event. <b>ValidValues:</b>						
	<table border="1"> <thead> <tr> <th>Value</th><th>Description</th></tr> </thead> <tbody> <tr> <td>0000h</td><td>Count events for all values of incoming Domain.</td></tr> <tr> <td>FFFFh-0001h</td><td>Bit-wise mask ANDed with incoming Domain.</td></tr> </tbody> </table>	Value	Description	0000h	Count events for all values of incoming Domain.	FFFFh-0001h	Bit-wise mask ANDed with incoming Domain.
Value	Description						
0000h	Count events for all values of incoming Domain.						
FFFFh-0001h	Bit-wise mask ANDed with incoming Domain.						

**IOMMUBARx41320 (IOMMUMMIO::IOMMU\_MMIO\_DEVICEID\_MATCH\_BANK\_1\_CNT\_3\_0)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx41320; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}  
\_aliasSMN; IOMMUMMIOx00041320; IOMMUMMIO=0240\_0000h

Bits	Description
31	<b>DIDMEN_1_3.</b> Read-write. Reset: 0. 0=DeviceID is ignored. 1=Filtered DeviceID must match to count an event. DeviceID Match enable.
30:16	Reserved.
15:0	<b>DEVICEID_MATCH_1_3.</b> Read-write. Reset: 0000h. DeviceID Match. This value is compared with the masked (filtered) value of the incoming DeviceID of the transaction to decide to count the corresponding event. The event is counted if DEVICEID_MATCH_1_3 is exactly equal to the masked incoming DeviceID. The event is not counted if they are not equal.

**IOMMUBARx41324 (IOMMUMMIO::IOMMU\_MMIO\_DEVICEID\_MATCH\_BANK\_1\_CNT\_3\_1)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx41324; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}  
\_aliasSMN; IOMMUMMIOx00041324; IOMMUMMIO=0240\_0000h

Bits	Description						
31:16	Reserved.						
15:0	<b>DEVICEID_MASK_1_3.</b> Read-write. Reset: 0000h. DeviceID Mask. This bit-mask is ANDed with the DeviceID of the transaction to decide to count the corresponding event. <b>ValidValues:</b>						
	<table border="1"> <thead> <tr> <th>Value</th><th>Description</th></tr> </thead> <tbody> <tr> <td>0000h</td><td>Count events for all values of incoming DeviceID.</td></tr> <tr> <td>FFFFh-0001h</td><td>Bit-wise mask ANDed with incoming DeviceID.</td></tr> </tbody> </table>	Value	Description	0000h	Count events for all values of incoming DeviceID.	FFFFh-0001h	Bit-wise mask ANDed with incoming DeviceID.
Value	Description						
0000h	Count events for all values of incoming DeviceID.						
FFFFh-0001h	Bit-wise mask ANDed with incoming DeviceID.						

**IOMMUBARx41328 (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_RPT\_BANK\_1\_CNT\_3\_0)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx41328; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI],  
IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}  
\_aliasSMN; IOMMUMMIOx00041328; IOMMUMMIO=0240\_0000h

Bits	Description
31:0	<b>EVENT_NOTE_1_3_LO.</b> Read-write. Reset: 0000_0000h. Event Note (bits[31:0]). When IOMMUMMIO::IOMMU_MMIO_COUNTER_RPT_BANK_1_CNT_3_1[CERE_1_3] == 1 and the



	corresponding counter is incremented and wraps to zero, EVENT_NOTE_1_3[51:0] is reported in the EVENT_COUNTER_ZERO event log entry. EVENT_NOTE_1_3[51:0] = {EVENT_NOTE_1_3_LO,IOMMUMMIO::IOMMU_MMIO_COUNTER_RPT_BANK_1_CNT_3_1[EVENT_NOTE_1_3_HI]}
--	--

#### IOMMUBARx4132C (IOMMUMMIO::IOMMU\_MMIO\_COUNTER\_RPT\_BANK\_1\_CNT\_3\_1)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOMMUBARx4132C; IOMMUBAR={IOMMUL2::IOMMU\_CAP\_BASE\_HI\_aliasHOST[IOMMU\_BASE\_ADDR\_HI], IOMMUL2::IOMMU\_CAP\_BASE\_LO\_aliasHOST[IOMMU\_BASE\_ADDR\_LO], 19'h0\_0000}

\_aliasSMN; IOMMUMMIOx0004132C; IOMMUMMIO=0240\_0000h

Bits	Description
31	<b>CERE_1_3.</b> Read-write. Reset: 0. 0=No event report when counter wraps to zero. 1=IOMMU Writes an EVENT_COUNTER_ZERO event log entry when the counter wraps to zero. <b>Description:</b> Counter Event Report Enable. The counter-wrap event is treated like any other event. <b>SOFTWARE NOTE:</b> The counter-wrap event is delivered promptly but there is no defined latency expectation.
30:20	Reserved.
19:0	<b>EVENT_NOTE_1_3_HI.</b> Read-write. Reset: 0_0000h. Event Note (bits[51:32]). When CERE_1_3 == 1 and the corresponding counter is incremented and wraps to zero, EVENT_NOTE_1_3[51:0] is reported in the EVENT_COUNTER_ZERO event log entry. See also IOMMUMMIO::IOMMU_MMIO_COUNTER_RPT_BANK_1_CNT_3_0.

## 9 FCH

### 9.1 FCH Overview

#### 9.1.1 Acronyms

*Table 70: List of Acronyms used in FCH*

Acronym	Definition
ASF	Alert Standard Format.
ASL	ACPI Source Language. See docACPI.
DASH	Desktop and mobile Architecture for System Hardware.
SCH	Server Controller Hub.
TPM	Trusted Platform Module.

#### 9.1.2 Functional

This section describes the integrated FCH. It utilizes a standard Scalable Data Fabric Port (SDP).

The following is the list of IP blocks and functions:

LPC/SPI/eSPI – Low Pin Count/Serial peripheral interface: this is the bridge logic to the BIOS/firmware flash and SPI TPM. eSPI is multiplexed on the SPI bus to support an eSPI device such as embedded controller (EC).

Family 17h processors have the following features:

1. LPC function
2. The LPC is multiplexed on SPI pins

eMMC – Embedded MMC: a solid state flash interface that is popular for low cost embedded applications. The latest revision v5.0 supports bandwidths up to 400MBs.

CLKGEN – Clock Generation: integrated clock generation function for the entire system. This block adds a non-spread display and more PCIe® GPP clock outputs. It also adds more internal reference clocks for various PHYs. See 9.2.10.1 [Miscellaneous (MISC) Registers] for register descriptions.

GPIO – General Purpose IO: defined by AOAC and used as GPIO or interrupt inputs. See 9.2.11 [GPIO Pin control registers] for register descriptions.

ACPI – Advanced Configuration and Power Interface: power management and reset functions.

Power Management (PM) Supervisory – custom logic to manage clock and power gating to support AOAC. See 9.2.10.2 [Power Management (PM) Registers] and 9.2.10.4 [Standard ACPI Registers] for register descriptions.

#### 9.1.3 MMIO Programming for Legacy Devices

The legacy devices, LPC, IOAPIC, ACPI, TPM, and Watchdog Timer, require the base address of the Memory Mapped IO registers to be assigned before these registers can be accessed. The Memory Mapped IO register base address and its entire range should be mapped to a non-posted memory region by programming the CPU register.

### 9.1.3.1 Description for FCH::IO::PCIInterrupt Map

FCH::IO::PciIntrIndex bit[7] means PciIntrApic, set as 0 means IRQ routing to PIC, set as 1 means IRQ routing to IOAPIC.

FCH::IO::PciIntrIndex bits[6:0] are PCI interrupt index. Select which PCI interrupt to map. Following are detail description.

Table 71: ValidValuesTable: PCI interrupt index list of PCI\_INTR\_INDEX bit[6:0]

Value	Description
04h-00h	INT[E:A]#.
05h	INTF#/GENINT2.
06h	INTG#.
07h	INTH#.
08h	Misc.
0Bh-09h	Misc[2:0].
0Fh-0Ch	INT[D:A] from serial IRQ.
10h	SCI.
11h	SMBUS0.
12h	ASF.
15h-13h	Reserved.
16h	PerMon.
17h	SD.
19h-18h	Reserved.
1Ah	SDIO.
1Fh-1Bh	Reserved.
20h	CIR, no IRQ connected.
21h	GPIOa, from PAD_FANIN0.
22h	GPIOb, from PAD_FANOUT0.
23h	GPIOc, no IRQ connected.
40h-24h	Reserved.
41h	SATA PCI interrupt.
42h	Reserved.
43h	eMMC.
4Fh-44h	Reserved.
53h-50h	GPPInt3/2/1/0.
61h-54h	Reserved.
62h	GPIO controller interrupt.
6Fh-63h	Reserved.
70h	I2C0.
71h	I2C1.
72h	I2C2.
73h	I2C3.
75h-74h	UART1/UART0.
77h-76h	I2C5/I2C4.
79h-78h	UART3/UART2.
FFh-7Ah	Reserved.

### 9.1.4 On-Chip Clock Generator

There are two clocking modes in which the processor can be brought up based on the boot strap pin SPI\_CLK. SBIOS should read FCH::MISC::StrapStat[ClkGenStrap] to determine the clock generator mode. SBIOS should program FCH::MISC::MiscClkCntl1[OscClkSwitchEn] = 1 to select an average 14 MHz OSC clock provided by internal PLL in both internal and external clocking modes.

### 9.1.5 EMMC

The eMMC (embedded MultiMedia Card controller) is controlled by driver software. There are no programming requirements for the BIOS. The eMMC interface pins can be either 1.8V or 3.3V.

### 9.1.6 eSPI

eSPI is configured to run at 16.67 MHz, 33 MHz, and 66 MHz. See 9.2.9.4 [eSPI Registers] about how to program the speed.

### 9.1.7 LPC Bus Interface

This section describes the LPC Bus related programming requirements. BIOS programs FCH::ITF::LPC::SPIBaseAddr[Spi\_eSpi\_BaseAddr] with a non-zero address to enable the MMIO access to the SPI ROM control registers.

RomProtect applies to ROM, regardless of ROM is located on SPI bus or LPC bus (Only one can exist, what means the ROM controls IP can only be applied to one ROM device which becomes the main BIOS ROM. That one device is selected by the hardware straps as to whether it is on the SPI or LPC bus.), thus RomProtect control is applied to both the LPC bus and SPI bus is correct.

Any control applies to ROM will apply to both SPI bus and LPC bus.

Those ROM controls does not apply to eSPI bus as eSPI controller is another independent IP.

#### 9.1.7.1 Read/Write SPI Flash through LPC

1. Configure register FCH::ITF::SPI::SPI100En[UseSpi100] = 1 to enable the support for SPI 100 MHz speed.
2. Configure register FCH::ITF::LPC::ROMAddrRng1 (D14F3x068), configuring ROM1 start address and end address.
3. Configure register FCH::ITF::LPC::ROMAddrRng2 (D14F3x06C), configuring ROM2 start address and end address.
4. Configure SPI chip select register FCH::ITF::SPI::AltSPICS[AltSpiCsEn] (SPIx001D[1:0]) to enable related SPI chip select.
5. Configure register FCH::ITF::LPC::RomProtect, keep [1:0] as 0 to ensure it is not in Read/Write protect status.
6. Access normal SPI flash space.

#### 9.1.7.2 Enabling SPI 100

To enable the support for SPI 100 MHz speed, software must program FCH::ITF::SPI::SPI100En[UseSpi100] = 1. SPI 100 should be enabled on after the auto ROM sizing has been completed (auto ROM sizing is done by the software only during the initial boot when the system power state transitions from G3->S5->S0). The actual Read speed also depends on the settings at FCH::ITF::SPI::SPI100En.

#### 9.1.7.3 Serial Peripheral Interface (SPI) ROM

SPI memory can be protected from being written to or read from by using the following sequence. This is optional and used as required.

### 9.1.7.3.1 Enable SPI ROM Protection

1. Program D14F3x050 FCH::ITF::LPC::RomProtect to enable protection for Read or Write memory accesses to SPI flash memory space. Up to four memory ranges specified by Rom Protect registers can be protected.
2. Program SPIx04 FCH::ITF::SPI::SPIRestrictedCmd or SPIx08 FCH::ITF::SPI::SPIRestrictedCmd2 with SPI commands that are required before doing write accesses to the SPI ROM. Use commands such as WR\_EN, WR\_Status and Erase when programming these restricted command registers.
3. Program SPIx00 FCH::ITF::SPI::SPICntrl0[SpiHostAccessRomEn] = 0 to protect the registers above from being reprogrammed.
4. Program SPIx1D FCH::ITF::SPI::AltSPICS[SpiProtectEn0] = 1 to apply Read/Write protection on ranges defined by Rom Protect registers (D14F3x50, D14F3x54, D14F3x58, D14F3x5C).
5. Program SPIx1D FCH::ITF::SPI::AltSPICS[SpiProtectLock] = 1 to make bits 3, 4, and 5 non-writable.

Accesses are now blocked by the SPI host preventing write accesses to the SPI ROM.

## 9.1.8 GPIO

There are three groups of GPIO, GPIOBank0 (FCH::GPIO::GPIOBank0Ctl), GPIOBank1 (FCH::GPIO::GPIOBank1Ctl) and GPIOBank2 (FCH::GPIO::GPIOBank2Ctl). See 9.2.11.2 [GPIO Registers].

To enable the GPIO controller, follow the programming steps listed below:

### 9.1.8.1 Interrupt GPIO

All the configuration bits, (Bit[x]), used in the following steps come from GPIOBank0 (FCH::GPIO::GPIOBank0Ctl), GPIOBank1 (FCH::GPIO::GPIOBank1Ctl) and GPIOBank2 (FCH::GPIO::GPIOBank2Ctl). If the pin is used as an interrupt, see 9.2.11.2 [GPIO Registers]:

1. The driver programs the DebounceTmrOut/DebounceTmrOutUnit/DebounceCntrl bits according to the data passed by BIOS through the ASL code.
2. Configure FCH::GPIO::GPIOBank0Ctl[10:8] according to the data passed by BIOS through ASL code.
3. Set FCH::GPIO::GPIOBank0Ctl[11] = 1 and FCH::GPIO::GPIOBank0Ctl[10] = 1 to enable interrupt delivery and interrupt status. Interrupt status is at FCH::GPIO::GPIOBank0Ctl[28] when the GPIO input is asserted.
4. Configure the GPIO controller interrupt. The GPIO controller interrupt is an active low level interrupt. The driver clears the interrupt status, then sets FCH::GPIO::GPIOWakeIntMasSwitch[EOI] = 1 to de-assert the interrupt request after acknowledging the interrupt.

### 9.1.8.2 Wakeup GPIO

All the configuration bits used in the following steps come from GPIOBank0 (FCH::GPIO::GPIOBank0Ctl), GPIOBank1 (FCH::GPIO::GPIOBank1Ctl) and GPIOBank2 (FCH::GPIO::GPIOBank2Ctl). If the pin is used as wake, see 9.2.11.2 [GPIO Registers]:

1. BIOS programs FCH::GPIO::GPIOBank0Ctl[15:13] to enable wake in S0i3/S3/S5 by BIOS. The wake status is FCH::GPIO::GPIOBank0Ctl[29]. BIOS/SMU reads the wake status to determine the wake source, then clears them after wake.
2. The driver programs DebounceTmrOut/DebounceTmrOutUnit/DebounceCntrl bits according the data from BIOS through the ASL code.
3. The driver sets FCH::GPIO::GPIOBank0Ctl[10:8] to configure assertion condition according to the data from BIOS through the ASL code.
4. Set FCH::GPIO::GPIOBank0Ctl[11] = 1 and FCH::GPIO::GPIOBank0Ctl[10] = 1 to enable interrupt delivery and

interrupt status then its interrupt status is found at FCH::GPIO::GPIOBank0Ctl[28] when the GPIO input is asserted.

- Configure the GPIO controller interrupt. The GPIO controller interrupt is an active low level interrupt. BIOS configures it and sets it to 0s. The driver clears the interrupt status then sets FCH::GPIO::GPIOWakeIntMasSwitch[EOI] = 1 after the interrupt is acknowledged.

### 9.1.8.3 Pure GPIO

All the configuration bits (Bit[x]) used in the following steps come from GPIOBank0 (FCH::GPIO::GPIOBank0Ctl), GPIOBank1 (FCH::GPIO::GPIOBank1Ctl) and GPIOBank2 (FCH::GPIO::GPIOBank2Ctl). If the pin is used as a GPIO, see 9.2.11.2 [GPIO Registers]:

FCH::GPIO::GPIOBank0Ctl[23:17] provides the state of the control on pull-up, pull-down, drive strength, output enable, output value, and pin value through FCH::GPIO::GPIOBank0Ctl[16].

### 9.1.8.4 GPIO Driving Strength

Following Table is for Driving Strength of 1.8V PAD, the connection of DrvStrengthSel[1:0] to {iS1, iS0} is reverted, DrvStrengthSel[1]=iS0, DrvStrengthSel[0]=iS1

Table 72: Driving Strength Configuration

iS1 DrvStrengthSel[0]	iS1 DrvStrengthSel[1]	Zout(ohms)
L	L	Do not use
L	H	40
H	L	60
H	H	80

### 9.1.8.5 PCIE\_RST1 Enhancement

PCIE\_RESET1\_L can be enhanced because some PCIe lane may require extra time to bring up, and BIOS can program FCH::GPIO::GPIOBank0Ctl[OutputValue] = 1 to take them out of reset.

- When set FCH::IOMUX::Gpio27[X48M0\_EGPIO27] == 1, it behaves as PCIE\_RST1\_L.
- Configure FCH::GPIO::GPIOBank0Ctl[OutputValue] = 0/1 toggles PCIE\_RST1\_L and create a reset signal.
- When set FCH::IOMUX::Gpio27[X48M0\_EGPIO27] == 0, it works as normal EGPIO27.

### 9.1.9 I2C Slave

The I2C Slave supports only one function: To be written by an external USB Power Delivery Engine chip; its address location is 0xFEDC\_6XXX.

### 9.1.10 FCH Register Access Information Guide

Following registers can only be accessed in 8-bit (byte access), using 16-bit (word access) or 32-bit (double word access) will have quirky behavior. For Read, byte0 data will be returned on all 4 bytes, for Write, only byte0 data will be written.

- IOMUX: Memory mapped address 0xFED8\_0D00 – 0xFED8\_0DFF.
- BIOS\_RAM: Access using IO (0xCD4: Index, 0xCD5: Data) or Memory mapped address 0xFED8\_0500 – 0xFED8\_05FF.
- CMOS\_RAM: Access using IO (0x72: Index, 0x73: Data) or Memory mapped address 0xFED8\_0600 –

0xFED8\_06FF.

- CMOS: Access using IO (0x70: Index, 0x71: Data) or Memory mapped address 0xFED8\_0700 – 0xFED8\_07FF.
- PMIO2: Memory mapped address 0xFED8\_0400 – 0xFED8\_04FF.
- ACPI: registers can be 8/16/32-bit, please refer to following table. AcpiMMioAddr=0xFED8\_0000.

**Table 73: Register Access Information**

Register Name	IO Base Address definition register	IO Offset Address*	MMIO Access
Pm1Status	PM_60: AcpiPm1EvtBlk	00h, 16-bit	AcpiMMioAddr + 800
Pm1Enable		02h, 16-bit	AcpiMMioAddr + 802
PmControl	PM_62: AcpiPm1CntBlk	00h, 16-bit	AcpiMMioAddr + 804
TmrValue/ETmrValue	PM_64: AcpiPmTmrBlk	00h, 32-bit, Read Only	AcpiMMioAddr + 808
CLKVALUE	PM_66: CpuControl	00h, 32-bit	AcpiMMioAddr + 80C
PLvl2		04h, 8-bit, Read Only	AcpiMMioAddr + 810
PLvl3		05h, 8-bit, Read Only	AcpiMMioAddr + 811
EVENT_STATUS	PM_68: AcpiGpe0Blk	00h, 32-bit	AcpiMMioAddr + 814
EVENT_ENABLE		04h, 32-bit	AcpiMMioAddr + 818
SmiCmdPort	PM_6A: AcpiSmiCmd	00h, 8-bit	AcpiMMioAddr + 81C
SmiCmdStatus		01h, 8-bit	AcpiMMioAddr + 81D
PmaControl	PM_6E: AcpiPmaCntBlk	00h, 8-bit	AcpiMMioAddr + 824

### 9.1.11 Reset Overview

Below are definitions of the various reset types:

- Type 0 reset (S5 Reset): RsmRst and UserRst.
- Type 1 reset (reset initiated by software or system): CF9, KBRst, Sync\_flood, ASF\_remote\_reset, Fail\_boot, Watchdog Timer reset, toggling of PwrGood (SLP\_S3#/SLP\_S5# remain de-asserted at high), SHUTDOWN command, INIT/PORT92.
- Type 2 reset (Sleep Reset): S3/S4/S5 reset.
- Type 3 reset (Fatal\_error\_reset or reset caused by hardware exception): 4s-shutdown, thermal trip, ASF\_remotePowerDown.
- Type 4 reset (any reset from above): Type 0 or Type 1 or Type 2 or Type 3.

**Table 74: Reset Type**

Register block	Power domain	Reset source
PCI Configure	S0	Type 4
SMI (9.2.4 [SMI Registers])	S5	Some register: Type 0 or Type 3 Some register: Type 4
PM (9.2.10.2 [Power Management (PM) Registers])	S5	Some register: Type 0 Some register: Type 4
PMIO2 (9.2.10.3 [Power Management (PM2) Registers])	S5	(FCH::PM::ResetCtl 1[RstToCpuPwrGdEn] == 1) ? Type 1 : Type 0
ACPI (9.2.10.2 [Power Management (PM) Registers])	S5	Type 0 or Type 3
ASF	S5	Type 0
SMBUS	S5	Type 0 Type 4
WatchDog (9.2.6 [Watchdog Timer (WDT) Registers])	S5	Type 0



HPET (9.2.5 [High Precision Event Timer (HPET) Registers])	S0	Type 0
IOMUX (9.2.11.1 [IOMUX Registers])	S5 (0-42) S0 (67-148)	S5-IoMux: Type 0 S0-IoMux: Type 4
MISC (9.2.10.1 [Miscellaneous (MISC) Registers])	S5	Most registers are Type 0. Some CLK register are Type 4 (i.e., FCH::MISC::MiscClkCntl4, FCH::MISC::AutoAddrLo)
Serial Debug	S5	Type 0
Shadow System Counter	S5	Type 0 or Type 1
GPIO-0 (9.2.11.2 [GPIO Registers])	S5	Type 0 or Type 1 or Type3
GPIO-1 (9.2.11.2 [GPIO Registers])	S0	Type 4
GPIO-2 (9.2.11.2 [GPIO Registers])	S0	Type 4
GPIO-3 (9.2.11.2 [GPIO Registers])	S0	Type 4
Wake Alarm (ACDC timer) (9.2.7 [Wake Alarm Device (AcDcTimer) Registers])	S5	Type 0
AOAC (9.2.8 [Always On Always Connected (AOAC) Registers])	S5	Type 0 or Type 1

Below is a set of simplified top level reset paths.

When a reset is generated from acpi\_s5, all S0 logic within the FCH are reset by it. It also resets devices on the motherboard that use PCIE\_RST\_L or LPC\_RST\_L. Other IPs within the SOC are not reset by it directly. Instead, they are being reset by SMU. when CpuPwrGood is connected to the SMU and the SMU treats it as an interrupt. Upon assertion of this signal, the SMU proceeds to do its "house cleaning" activities first; then it will issue resets to all other IPs that are not inside of the FCH. CpuPwrGood is the actual reset to the SMU. When this signal is de-asserted (to low), the SMU will propagate its reset to all IPs within the SOC. Warm reset is defined as an assertion of CpuRstB and cold reset the toggling of both CpuRstB (low) and CpuPwrGood (low).

Table 75: Reset Overview

S5 reset	Behavior
RSMRST_L	Reset all logic (S0 and S5). In addition, SOC will go back to default power state (always-off, always-on, previous state).
SYS_RST_L	Reset all logic (S0 and S5).
Sleep induced reset	Behavior
SLEEP3/4/5	Sleep entry to S3/4/5 will assert cold reset to all S0 logic prior to assertion of SLP_S3_L/SLP_S5_L.
Pin included reset	Behavior
PWRBUTTON_L	4 second override – unconditional shutdown (Sleep S5) → generates cold reset and will reset all S0 logic.
THERMTRIP_L	Over temperature indicator from CPU – unconditional shutdown (Sleep S5) → generates cold reset and will reset all S0 logic.
KB_RST_L	Keyboard reset → reset to S0 logic; configurable to warm or cold reset.
PWRGOOD	De-assertion of PWRGOOD will always generate cold reset to S0 logic.
Software induced reset	Behavior
CF9	Write to CF9 → reset S0 logic; configurable to warm or cold reset; or momentary S0 → S5 → S0 transition.
Port92	Write Port 92 (FAST_INIT) → reset S0 logic; configurable to warm or cold reset.
Software writes to PM_Reg	Generates reset to S0 and S5 logic.

0xC4[6], resetallacpi	
Hardware based reset	Behavior
Watchdog timer	Watchdog timer → reset S0 logic; configurable to warm or cold reset.
AMD (boot_timer) Watchdog timer	AMD defined Watchdog timer → reset S0 logic; configurable to warm or cold reset.
Remote Command reset	Behavior
ASF – remote reset	Reset S0 logic; configurable to warm or cold reset.
ASF – remote sleep	Remote sleep S5 command – logic will sequence to S5 and assert cold reset to all S0 logic.
Internal events	Behavior
Shutdown (message from CPU)	Triple faults in CPU will cause an internal SHUTDOWN message to be broadcasted. FCH will generate a reset to S0 logic; configurable to warm or cold reset.
SYNC_FLOOD (message)	Internal data fabric logic detects an error (e.g., parity error) and broadcasts an internal SYNC_FLOOD message. FCH will generate a reset to S0 logic; configurable to warm or cold reset.

## 9.1.12 RTC

### 9.1.12.1 RTC Register

Please note that, the value in Time/Alarm or CMOS registers or CMOS RAM is undefined/indeterministic when power up first time. It is highly recommended to add a checksum or CRC over CMOS RAM so BIOS can detect whether CMOS has been corrupted or erased.

You can see RTC registers at 9.2.1 [Legacy Block Configuration Registers (IO)].

### 9.1.13 Address Mapping Table

Table 76: Address Space Mapping

Function name	Address Mapping in HOST Space	NOTES
LPC/SPI ROM_1	00_000F_FFFFh - 00_0000_0000h	
ACPI	00_FED8_[1:0]XXXh	
AL2AHB	00_FEDC_0XXXh	
Reserved	00_FEDC_3FFFh - 00_FEDC_1000h	
I2C_2 Master	00_FEDC_4XXXh	
I2C_3 Master	00_FEDC_5XXXh	
I2C_4 Slave for USB Power Delivery	00_FEDC_6XXXh	
DMAC0	00_FEDC_7XXXh	
DMAC1	00_FEDC_8XXXh	
UART0	00_FEDC_9XXXh	
UART1	00_FEDC_AXXXh	
Reserved	00_FEDC_BFFFh -	

	00_FEDC_B000h	
DMA_2	00_FEDC_CXXXh	
DMA_3	00_FEDC_DXXXh	
UART_2	00_FEDC_EXXXh	
UART_3	00_FEDC_FXXXh	
Reserved	00_FEDD_4FFFh - 00_FEDD_0000h	
EMMC Control	00_FEDD_5XXXh	
LPC/SPI ROM_2	00_FFFF_FFFFh - 00_FF00_0000h	
LPC/SPI ROM_3	FD_03FF_FFFFh - FD_0000_0000h	

## 9.2 Registers

### 9.2.1 Legacy Block Configuration Registers (IO)

#### 9.2.1.1 Registers

See 1.4.3 [Register Mnemonics] for a description of the register naming convention.

IOx0000 [Dma_Ch 0] (FCH::IO::DmaCh0)	
Read-write. Reset: 0000h.	
_aliasIO; IOx0000; IO=0000_0000h	
Bits	Description
15:0	<b>DmaCh0</b> . Read-write. Reset: 0000h. DMA1 channel 0 base and current address.
IOx0002 [Dma_Ch 1] (FCH::IO::DmaCh1)	
Read-write. Reset: 0000h.	
_aliasIO; IOx0002; IO=0000_0000h	
Bits	Description
15:0	<b>DmaCh1</b> . Read-write. Reset: 0000h. DMA1 channel 1 base and current address.
IOx0004 [Dma_Ch 2] (FCH::IO::DmaCh2)	
Read-write. Reset: 0000h.	
_aliasIO; IOx0004; IO=0000_0000h	
Bits	Description
15:0	<b>DmaCh2</b> . Read-write. Reset: 0000h. DMA1 channel 2 base and current address.
IOx0006 [Dma_Ch 3] (FCH::IO::DmaCh3)	
Read-write. Reset: 0000h.	
_aliasIO; IOx0006; IO=0000_0000h	
Bits	Description
15:0	<b>DmaCh3</b> . Read-write. Reset: 0000h. DMA1 channel 3 base and current address.
IOx0008 [Dma_Status] (FCH::IO::DmaStat)	
Read-write. Reset: 00h.	

_aliasIO; IOx0008; IO=0000_0000h	
Bits	Description
7:0	<b>DmaStatus</b> . Read-write. Reset: 00h. Specifies the DMA status register for channels[3:0] for reads and specifies the DMA control register for channels[3:0] for writes.

**IOx0009 [Dma\_WriteRequest] (FCH::IO::DmaWriteRequest)**

Read-write. Reset: 00h.

\_aliasIO; IOx0009; IO=0000\_0000h

Bits	Description
7:0	<b>DmaWriteRequest</b> . Read-write. Reset: 00h. Request register.

**IOx000A [Dma\_WriteMask] (FCH::IO::DmaWriteMask)**

Read-write. Reset: 00h.

\_aliasIO; IOx000A; IO=0000\_0000h

Bits	Description
7:0	<b>DmaWriteMask</b> . Read-write. Reset: 00h. DMA channel mask register.

**IOx000B [Dma\_WriteMode] (FCH::IO::DmaWriteMode)**

Read-write. Reset: 00h.

\_aliasIO; IOx000B; IO=0000\_0000h

Bits	Description
7:0	<b>DmaWriteMode</b> . Read-write. Reset: 00h. DMA mode register.

**IOx000C [Dma\_Clear] (FCH::IO::DmaClear)**

Read-write. Reset: 00h.

\_aliasIO; IOx000C; IO=0000\_0000h

Bits	Description
7:0	<b>DmaClear</b> . Read-write. Reset: 00h. Channel[3:0] DMA clear byte pointer.

**IOx000D [Dma\_MasterClr] (FCH::IO::DmaMasClr)**

Read-write. Reset: 00h.

Dma\_MasterClr register

\_aliasIO; IOx000D; IO=0000\_0000h

Bits	Description
7:0	<b>DmaMasterClr</b> . Read-write. Reset: 00h. Write: Channel[3:0] master clear register. Read: Intermediate register.

**IOx000E [Dma\_ClrMask] (FCH::IO::DmaClrMask)**

Read-write. Reset: 00h.

\_aliasIO; IOx000E; IO=0000\_0000h

Bits	Description
7:0	<b>DmaClrmask</b> . Read-write. Reset: 00h. Channel[3:0] DMA clear mask.

**IOx000F [Dma\_AllMask] (FCH::IO::DmaAllMask)**

Read-write. Reset: 00h.

\_aliasIO; IOx000F; IO=0000\_0000h

Bits	Description
7:0	<b>DmaAllMask</b> . Read-write. Reset: 00h. General mask register.

**IOx0020 [IntrCntl1Reg1] (FCH::IO::IntrCntl1Reg1)**

Read-write. Reset: 00h.

\_aliasIO; IOx0020; IO=0000\_0000h

Bits	Description
7:0	<b>IntrCntl1Reg1</b> . Read-write. Reset: 00h. IRQ[7:0] status and control. Read: IRR, ISR. Write: ICW1, OCW2, OCW3.

**IOx0021 [IntrCntrl1Reg2] (FCH::IO::IntrCntrl1Reg2)**

Read-write. Reset: 00h.

\_aliasIO; IOx0021; IO=0000\_0000h

Bits	Description
7:0	<b>IntrCntrl1Reg2</b> . Read-write. Reset: 00h. IRQ[7:0] status and control. Read: IMR. Write: ICW2, ICW3, ICW4, OCW1.

**IOx0022 [IMCR\_Index] (FCH::IO::IMCRIndex)**

Read-write. Reset: 00h.

\_aliasIO; IOx0022; IO=0000\_0000h

Bits	Description
7:0	<b>ImcrIndex</b> . Read-write. Reset: 00h. Index port for IMCR register. See FCH::IO::IMCRData.

**IOx0023 [IMCR\_Data] (FCH::IO::IMCRData)**

Read-write. Reset: 00h.

\_aliasIO; IOx0023; IO=0000\_0000h

Bits	Description
7:0	<b>ImcrData</b> . Read-write. Reset: 00h. Data port for IMCR register. The actual IMCR register is located at index 70h and it is at bit[0].

**IOx0040 [TimerCh0] (FCH::IO::TimerCh0)**

Read-write. Reset: 00h.

\_aliasIO; IOx0040; IO=0000\_0000h

Bits	Description
7:0	<b>TimerCh0</b> . Read-write. Reset: 00h. 8254 Timer 1 Counter 0 Data Port.

**IOx0041 [TimerCh1] (FCH::IO::TimerCh1)**

Read-write. Reset: 00h.

\_aliasIO; IOx0041; IO=0000\_0000h

Bits	Description
7:0	<b>TimerCh1</b> . Read-write. Reset: 00h. 8254 Timer 1 Counter 1 Data Port.

**IOx0042 [TimerCh2] (FCH::IO::TimerCh2)**

Read-write. Reset: 00h.

\_aliasIO; IOx0042; IO=0000\_0000h

Bits	Description
7:0	<b>TimerCh2</b> . Read-write. Reset: 00h. 8254 Timer 1 Counter 2 Data Port.

**IOx0043 [Tmr1CntrlWord] (FCH::IO::Tmr1CntrlWord)**

Write-only. Reset: 00h.

This is the control register for 8254 timer 1. If a counter is programmed to read or write two-byte counts, the following precaution applies: Software must not transfer control between writing the first and second byte to another routine which also writes into that same counter; otherwise, the counter is be loaded with an incorrect value. The count must always be completely loaded with both bytes. Reads of this register always returns 0xFF.

\_aliasIO; IOx0043; IO=0000\_0000h

Bits	Description								
7:6	<b>CounterSelect</b> . Write-only. Reset: 0h.								
<b>ValidValues:</b>									
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>Select counter 0.</td></tr> <tr> <td>1h</td><td>Select counter 1.</td></tr> <tr> <td>2h</td><td>Select counter 2.</td></tr> </table>	Value	Description	0h	Select counter 0.	1h	Select counter 1.	2h	Select counter 2.
Value	Description								
0h	Select counter 0.								
1h	Select counter 1.								
2h	Select counter 2.								

	3h	Read back command.
5:4	<b>CommandSelect.</b> Write-only. Reset: 0h.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	Counter latch command.
	1h	Read/Write least significant byte.
	2h	Read/Write most significant byte.
	3h	Read/Write least, and then most significant byte.
3:1	<b>ModeSelect.</b> Write-only. Reset: 0h.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	Asserts OUT signal at end of count.
	1h	Hardware re-triggerable one-shot.
	2h	Rate generator.
	3h	Square wave output.
	4h	Software triggered strobe.
	5h	Hardware triggered strobe.
	7h-6h	Reserved.
0	<b>CntDownSelect.</b> Write-only. Reset: 0. 0=Binary countdown. 1=BCD countdown.	

**IOx0061 [Nmi Status] (FCH::IO::NmiStat)**

\_aliasIO; IOx0061; IO=0000\_0000h

Bits	Description
7	<b>ParityErrNmi.</b> Read-only. Reset: X. NMI is caused by parity error (either PERR# or SERR#).
6	<b>IoChkNmi.</b> Read-only. Reset: X. NMI is triggered by serial IOCHK.
5	<b>SpkrClk.</b> Read-only. Reset: X. The output of 8254 timer counter 2.
4	<b>RefClk.</b> Read-only. Reset: X. The output of 8254 timer counter 1.
3	<b>IoChkNmiEn.</b> Read-write. Reset: 1. 0=Enable IoChk to NMI generation. 1=Disable IoChk to NMI generation.
2	<b>ParityErrNmiEn.</b> Read-write. Reset: 1. 0=Enable Parity Error to NMI generation (from SERR# or PERR#). 1=Disable Parity Error to NMI generation and clear bit[7].
1	<b>SpkrTmrEnable.</b> Read-write. Reset: 0. 0=Speaker timer off. 1=Speaker timer on.
0	<b>SpkrEnable.</b> Read-write. Reset: 0. 0=Disable counter 2. 1=Enable counter 2.

**IOx0070 [RtcAddrPort and NmiMask] (FCH::IO::RtcAddrPortNmiMask)**

Reset: 00h.

\_aliasIO; IOx0070; IO=0000\_0000h

Bits	Description
7	<b>NmiMask.</b> Write-only. Reset: 0. 0=NMI enabled. 1=NMI masked.
6:0	<b>RtcAddrPort: RTC Address Port.</b> Read-write. Reset: 00h. This is used with either the internal RTC or external RTC. This port specifies the index to access RTC time registers and the CMOS RAM space. FCH::IO::RTCA[DV0] should be programmed first to select bank when accessing the CMOS RAM space. See FCH::IO::RTCSeconds.

**IOx0071 [RtcDataPort] (FCH::IO::RtcDataPort)**

Read-write. Reset: 00h.

\_aliasIO; IOx0071; IO=0000\_0000h

Bits	Description
7:0	<b>RtcDataPort: RTC Data Port.</b> Read-write. Reset: 00h. This is used with either the internal RTC or external RTC in conjunction with FCH::IO::RtcAddrPortNmiMask.

**IOx0072 [Alternate RTC AddrPort] (FCH::IO::AltRTCAddrPort)**

Read-write. Reset: 00h.	
_aliasIO; IOx0072; IO=0000_0000h	
Bits	Description
7:0	<b>AlternatRTCAddrPort.</b> Read-write. Reset: 00h. This is used with the internal RTC. This port allows the user to specify the full 8-bit address (instead of bank0/bank1 indexing) to access the 256 bytes of RTC RAM.

**IOx0073 [Alternate RTC DataPort] (FCH::IO::AltRTCDataPort)**

Read-write. Reset: 00h.	
_aliasIO; IOx0073; IO=0000_0000h	
Bits	Description
7:0	<b>AlternatRTCDataPort.</b> Read-write. Reset: 00h. This is used with the internal RTC in conjunction with FCH::IO::AltRTCAddrPort.

**IOx00073\_x00 [RTC Seconds] (FCH::IO::RTCSeconds)**

Read-write.	
_aliasIO; IOx00073_x00; IO=0000_0000h; DataPortWrite=FCH::IO::AltRTCAddrPort	
_aliasHOST; RTCHOSTx00000; RTCHOST=FED8_0700h	
Bits	Description
7:0	<b>Seconds.</b> Read-write. Reset: XXh. BCD format. The value range for this register is 00 through 59. If (FCH::IO::RTCB[SET] == 1), this register is programmed by software and hardware updating is disabled. If (FCH::IO::RTCB[SET] == 0), this register is updated by the RTC logic once per second. The value of this register is undefined/non-deterministic when powered up for the first time.

**IOx00073\_x01 [RTC Seconds Alarm] (FCH::IO::RtcSecAlarm)**

Read-write.	
_aliasIO; IOx00073_x01; IO=0000_0000h; DataPortWrite=FCH::IO::AltRTCAddrPort	
_aliasHOST; RTCHOSTx00001; RTCHOST=FED8_0700h	
Bits	Description
7:0	<b>SecondsAlarm: Seconds Alarm.</b> Read-write. Reset: XXh. BCD format. If (FCH::IO::RTCB[SET] == 1), the Seconds Alarm register never matches FCH::IO::RTCSeconds. Else, if (bits[7:6] == 11b) the RTC Seconds Alarm register always matches RTC Seconds register and causes an RTC alarm event to be generated once per second. See FCH::IO::RTCC[AF]. The value of this register is undefined/non-deterministic when power up first time.

**IOx00073\_x02 [RTC Minutes] (FCH::IO::RTCMinutes)**

Read-write.	
_aliasIO; IOx00073_x02; IO=0000_0000h; DataPortWrite=FCH::IO::AltRTCAddrPort	
_aliasHOST; RTCHOSTx00002; RTCHOST=FED8_0700h	
Bits	Description
7:0	<b>Minutes.</b> Read-write. Reset: XXh. BCD format. The value range for this register is 00 through 59. If (FCH::IO::RTCB[SET] == 1), this register is programmed by software and hardware updating is disabled. If (FCH::IO::RTCB[SET] == 0), this register is updated by the RTC logic, once per second. The value of this register is undefined/non-deterministic when powered up for the first time.

**IOx00073\_x03 [RTC Minutes Alarm] (FCH::IO::RTCMinutesAlarm)**

Read-write.	
_aliasIO; IOx00073_x03; IO=0000_0000h; DataPortWrite=FCH::IO::AltRTCAddrPort	
_aliasHOST; RTCHOSTx00003; RTCHOST=FED8_0700h	
Bits	Description
7:0	<b>MinutesAlarm: Minutes Alarm.</b> Read-write. Reset: XXh. BCD format. If (FCH::IO::RTCB[SET] == 1), the RTC Minutes Alarm register never matches FCH::IO::RTCMinutes. Else, if (bits[7:6] == 11b), the RTC Minutes Alarm register always matches RTC Minutes register and causes an RTC alarm event to be generated once per minute. See FCH::IO::RTCC[AF]. The value of this register is undefined/non-deterministic when powered up for the first time.



**IOx00073\_x04 [RTC Hours] (FCH::IO::RTCHours)**

Read-write.

\_aliasIO; IOx00073\_x04; IO=0000\_0000h; DataPortWrite=FCH::IO::AltRTCAddrPort\_aliasHOST; RTCHOSTx00004; RTCHOST=FED8\_0700h

Bits	Description
7:0	<b>Hours.</b> Read-write. Reset: XXh. BCD format. The value range for this register is 00 through 23. If (FCH::IO::RTCB[SET] == 1), this register can be set by software and hardware updating is disabled. If (FCH::IO::RTCB[SET] == 0), this register is updated by the RTC logic once per second. The value of this register is undefined/non-deterministic when powered up for the first time.

**IOx00073\_x05 [RTC Hours Alarm] (FCH::IO::RTCHoursAlarm)**

Read-write.

\_aliasIO; IOx00073\_x05; IO=0000\_0000h; DataPortWrite=FCH::IO::AltRTCAddrPort\_aliasHOST; RTCHOSTx00005; RTCHOST=FED8\_0700h

Bits	Description
7:0	<b>HoursAlarm: Hours Alarm.</b> Read-write. Reset: XXh. BCD format. If (FCH::IO::RTCB[SET] == 1), the RTC Hours Alarm register never matches FCH::IO::RTCHours. Else, if (bits[7:6] == 11b) the RTC Hours Alarm register always matches with the RTC Hours register and causes an RTC alarm event to be generated once per hour. See FCH::IO::RTCC[AF]. The value of this register is undefined/non-deterministic when powered up for the first time.

**IOx00073\_x06 [RTC Day of Week] (FCH::IO::RTCDayOfWeek)**

Read-write.

\_aliasIO; IOx00073\_x06; IO=0000\_0000h; DataPortWrite=FCH::IO::AltRTCAddrPort\_aliasHOST; RTCHOSTx00006; RTCHOST=FED8\_0700h

Bits	Description
7:0	<b>DayOfWeek: Day of Week.</b> Read-write. Reset: XXh. BCD format. The value range for this register is 01 through 07 (Sunday = 1). Leap year correction is performed by software. If (FCH::IO::RTCB[SET] == 1), this register is programmed by software and hardware updating is disabled. If (FCH::IO::RTCB[SET] == 0), this register is updated by hardware. The value of this register is undefined/non-deterministic when powered up for the first time.

**IOx00073\_x07 [RTC Date of Month] (FCH::IO::RTCDateofMonth)**

Read-write.

\_aliasIO; IOx00073\_x07; IO=0000\_0000h; DataPortWrite=FCH::IO::AltRTCAddrPort\_aliasHOST; RTCHOSTx00007; RTCHOST=FED8\_0700h

Bits	Description
7:0	<b>DateOfMonth: Date of Month.</b> Read-write. Reset: XXh. BCD format. The range is 01 through 31. Leap year correction is performed by software. If (FCH::IO::RTCB[SET] == 1), this register is programmed by software and hardware updating is disabled. If (FCH::IO::RTCB[SET] == 0), this register is updated by hardware. The value of this register is undefined/non-deterministic when powered up for the first time.

**IOx00073\_x08 [RTC Month] (FCH::IO::RTCMonth)**

Read-write.

\_aliasIO; IOx00073\_x08; IO=0000\_0000h; DataPortWrite=FCH::IO::AltRTCAddrPort\_aliasHOST; RTCHOSTx00008; RTCHOST=FED8\_0700h

Bits	Description
7:0	<b>Month.</b> Read-write. Reset: XXh. BCD format. The range is 01 through 12. Leap year correction is performed by software. If (FCH::IO::RTCB[SET] == 1), this register is programmed by software and hardware updating is disabled. If (FCH::IO::RTCB[SET] == 0), this register is updated by hardware. The value of this register is undefined/non-deterministic when powered up for the first time.

**IOx00073\_x09 [RTC Year] (FCH::IO::RTCYear)**

Read-write.

_aliasIO; IOx00073_x09; IO=0000_0000h; DataPortWrite=FCH::IO::AltRTCAddrPort	
_aliasHOST; RTCHOSTx00009; RTCHOST=FED8_0700h	
Bits	Description
7:0	<b>Year.</b> Read-write. Reset: XXh. BCD format. Range is 00 through 99. No leap year correction capability. Leap year correction must be done by software. If (FCH::IO::RTCB[SET] == 1), this register is programmed by software and hardware updating is disabled. If (FCH::IO::RTCB[SET] == 0), this register is updated by hardware. The value of this register is undefined/non-deterministic when powered up for the first time.

**IOx00073\_x0A [RTC Register A] (FCH::IO::RTCA)**

_aliasIO; IOx00073_x0A; IO=0000_0000h; DataPortWrite=FCH::IO::AltRTCAddrPort	
_aliasHOST; RTCHOSTx0000A; RTCHOST=FED8_0700h	
Bits	Description
7	<b>UIP: Update In Progress.</b> Read-only. Reset: X. 0=The update transfer does not occur for at least 244 us. 1=The update transfer occurs soon. If (FCH::IO::RTCB[SET] == 1), UIP is cleared. The value of this register is undefined/non-deterministic when powered up for the first time.
6:5	Reserved.
4	<b>DV0: Bank Selection.</b> Read-write. Reset: X. 0=Select bank 0 when accessing the RTC CMOS RAM space through FCH::IO::RtcAddrPortNmiMask and FCH::IO::RtcDataPort. 1=Select bank 1 when accessing the RTC CMOS RAM space through FCH::IO::RtcAddrPortNmiMask and FCH::IO::RtcDataPort. The FCH has an alternate way to access the RAM without the use of bank select bit. FCH::IO::AltRTCAddrPort and FCH::IO::AltRTCDataPort provides indexed access to the full 256 bytes of RAM. The value of this register is undefined/non-deterministic when powered up for the first time.
3:0	<b>RS[3:0]: Rate Selection.</b> Read-write. Reset: Xh. These four rate selection bits select one of the 13 taps on the 15-stage frequency divider or disable the divider output (flat output signal). The tap selected can be used to generate a periodic interrupt. See the following table for the frequency selection. The value of this register is undefined/non-deterministic when powered up for the first time.
<b>Valid Values:</b>	
Value	Description
0h	Flat Signal (None).
1h	256 Hz (3.90625 ms).
2h	128 Hz (7.8125 ms).
3h	8.192 kHz (122.070 us).
4h	4.096 kHz (244.141 us).
5h	2.048 kHz (488.281 us).
6h	1.024 kHz (976.5625 us).
7h	512 Hz (1.953125 ms).
8h	256 Hz (3.90625 ms).
9h	128 Hz (7.8125 ms).
Ah	64 Hz (15.625 ms).
Bh	32 Hz (31.25 ms).
Ch	16 Hz (62.5 ms).
Dh	8 Hz (125 ms).
Eh	4 Hz (250 ms).
Fh	2 Hz (500 ms).

**IOx00073\_x0B [RTC Register B] (FCH::IO::RTCB)**

Read-write.	
_aliasIO; IOx00073_x0B; IO=0000_0000h; DataPortWrite=FCH::IO::AltRTCAddrPort	
_aliasHOST; RTCHOSTx0000B; RTCHOST=FED8_0700h	
Bits	Description
7	<b>SET: Set new time.</b> Read-write. Reset: X. 0=The RTC time registers are updated every second. 1=No internal updating for RTC time registers occurs. The value of this register is undefined/non-deterministic when powered

	up for the first time.
6	<b>PIE: Periodic Interrupt Enable.</b> Read-write. Reset: X. 1=Enable the FCH::IO::RTCC[PF] bit to assert the IRQ. The value of this register is undefined/non-deterministic when powered up for the first time.
5	<b>AIE: Alarm Interrupt Enable.</b> Read-write. Reset: X. 1=Enable the FCH::IO::RTCC[AF] bit to assert the IRQ. When the alarm time is written in the appropriate hours, minutes, and seconds alarm registers, the alarm interrupt is initiated at the specified time each day if AIE == 1. The value of this register is undefined/non-deterministic when powered up for the first time.
4	<b>UIE: Update Ended Interrupt Enable.</b> Read-write. Reset: X. 1=Enable the FCH::IO::RTCC[UF] bit to assert the IRQ. If [SET] == 1, UIE is cleared. The value of this register is undefined/non-deterministic when powered up for the first time.
3:2	Reserved.
1	<b>HourMode.</b> Read-write. Reset: X. 0=12 hour mode. 1=24 hour mode. The value of this register is undefined/non-deterministic when powered up for the first time.
0	<b>DaylightSavingEnable.</b> Read-write. Reset: X. Init: BIOS,0. 1=RTC daylight saving is enabled. The value of this register is undefined/non-deterministic when powered up for the first time.

**IOx00073\_x0C [RTC Register C] (FCH::IO::RTCC)**

Read-only.

\_aliasIO; IOx00073\_x0C; IO=0000\_0000h; DataPortWrite=FCH::IO::AltRTCAddrPort

\_aliasHOST; RTCHOSTx0000C; RTCHOST=FED8\_0700h

Bits	Description
7	<b>IRQF: Interrupt Request Flag.</b> Read-only. Reset: X. 1=The IRQ# pin is driven low. $IRQF = (PF * PIE) + (AF * AIE) + (UF * UIE)$ . Reading RTC Register C clears the IRQF bit. The value of this register is undefined/non-deterministic when powered up for the first time.
6	<b>PF: Periodic Interrupt Flag.</b> Read-only. Reset: X. 1=An edge is detected on the selected tap (through RS3 to RS0) of the frequency divider. Reading RTC Register C clears PF bit. The value of this register is undefined/non-deterministic when powered up for the first time.
5	<b>AF: Alarm Interrupt Flag.</b> Read-only. Reset: X. 1=FCH::IO::RTCSeconds, FCH::IO::RTCMinutes, FCH::IO::RTCHours and FCH::IO::RTCDateofMonth match FCH::IO::RtcSecAlarm, FCH::IO::RTCMinutesAlarm, FCH::IO::RTCHoursAlarm, and FCH::IO::RTCDateAlarm respectively. Reading RTC Register C clears AF bit. The value of this register is undefined/non-deterministic when powered up for the first time.
4	<b>UF: Update Ended Interrupt Flag.</b> Read-only. Reset: X. 1=Update cycle complete. Reading RTC Register C clears UF. The value of this register is undefined/non-deterministic when powered up for the first time.
3:0	Reserved.

**IOx00073\_x0D [RTC Date Alarm] (FCH::IO::RTCDateAlarm)**

\_aliasIO; IOx00073\_x0D; IO=0000\_0000h; DataPortWrite=FCH::IO::AltRTCAddrPort

\_aliasHOST; RTCHOSTx0000D; RTCHOST=FED8\_0700h

Bits	Description
7	<b>VRT: Valid RAM and Time.</b> Read-only. Reset: X. 0=RTC date, time, and CMOS RAM are invalid due to low RTC battery being monitored. 1=RTC date, time, and CMOS RAM are valid. See FCH::PM::VRT_T1 and FCH::PM::VRT_T2. The value of this register is undefined/non-deterministic when powered up for the first time.
6	<b>ScratchBit.</b> Read-write. Reset: X. Scratch bit. The value of this register is undefined/non-deterministic when powered up for the first time.
5:0	<b>DateAlarm.</b> Read-write. Reset: XXXXXXb. If (DateAlarm != 0), then DateAlarm is considered for alarm generation. DateAlarm is in BCD format. 0=DateAlarm is not compared for alarm generation. 1=DateAlarm is compared for alarm generation. The value of this register is undefined/non-deterministic when powered up for the first time.
<b>ValidValues:</b>	
<b>Bit</b>	<b>Description</b>
[0]	DateAlarm 0

[1]	DateAlarm 1
[2]	DateAlarm 2
[3]	DateAlarm 3
[4]	DateAlarm 4
[5]	DateAlarm 5

**IOx00073\_x32 [RTC AltCentury] (FCH::IO::RTCAltCentury)**

Read-write.

\_aliasIO; IOx00073\_x32; IO=0000\_0000h; DataPortWrite=FCH::IO::AltRTCAAddrPort\_aliasHOST; RTCHOSTx00032; RTCHOST=FED8\_0700h

Bits	Description
7:0	<b>AltCentury</b> . Read-write. Reset: XXh. BCD format. Leap year correction is performed by hardware. This register is accessed only when (FCH::IO::RTCA[DV0] == 0) and (FCH::PM::RTCCtl[CenturyEn] == 1). If (FCH::IO::RTCB[SET] == 1), this register is programmed by software and hardware updating is disabled. If (FCH::IO::RTCB[SET] == 0), this register is automatically updated by hardware every century. The value of this register is undefined/non-deterministic when powered up for the first time.

**IOx00073\_x48 [RTC Century] (FCH::IO::RTCCentury)**

Read-write.

\_aliasIO; IOx00073\_x48; IO=0000\_0000h; DataPortWrite=FCH::IO::AltRTCAAddrPort\_aliasHOST; RTCHOSTx00048; RTCHOST=FED8\_0700h

Bits	Description
7:0	<b>Century</b> . Read-write. Reset: XXh. BCD format. Leap year correction is done through hardware. This register is accessed only when (FCH::IO::RTCA[DV0] == 1). If (FCH::IO::RTCB[SET] == 1), this register is programmed by software and hardware updating is disabled. If (FCH::IO::RTCB[SET] == 0), this register is automatically updated by hardware every century. The value of this register is undefined/non-deterministic when powered up for the first time.

**IOx00073\_x50 [RTC Extended RAM Address Port] (FCH::IO::RTCExtRAMAddrPort)**

Read-write.

\_aliasIO; IOx00073\_x50; IO=0000\_0000h; DataPortWrite=FCH::IO::AltRTCAAddrPort\_aliasHOST; RTCHOSTx00050; RTCHOST=FED8\_0700h

Bits	Description
7	Reserved.
6:0	<b>ExtendedRamAddrPort</b> . Read-write. Reset: XXXXXXXXb. Because only 7 address bits are used in FCH::IO::RtcAddrPortNmiMask[RtcAddrPort], only the lower 128 bytes at offset 7Fh:00h are accessible through FCH::IO::RtcDataPort. Extended RAMs are physically located at addresses 80h to FFh. In order to access these addresses, an address offset should be programmed into this register and access to them is done through FCH::IO::RTCExtRAMDataPort. An offset of 80h is automatically added to this 7-bit address. The value of this register is undefined/non-deterministic when powered up for the first time.

**IOx00073\_x53 [RTC Extended RAM Data Port] (FCH::IO::RTCExtRAMDataPort)**

Read-write.

\_aliasIO; IOx00073\_x53; IO=0000\_0000h; DataPortWrite=FCH::IO::AltRTCAAddrPort\_aliasHOST; RTCHOSTx00053; RTCHOST=FED8\_0700h

Bits	Description
7:0	<b>ExtendedRamDataPort</b> . Read-write. Reset: XXh. Extended RAM data port. The value of this register is undefined/non-deterministic when powered up for the first time.

**IOx00073\_x7E [RTC Time Clear] (FCH::IO::RTCTimeClear)**

Read-write.

\_aliasIO; IOx00073\_x7E; IO=0000\_0000h; DataPortWrite=FCH::IO::AltRTCAAddrPort\_aliasHOST; RTCHOSTx0007E; RTCHOST=FED8\_0700h

Bits	Description
------	-------------

7:1	Reserved.
0	<b>RtcTimeClear</b> . Read-write. Reset: X. 1=Clear the RTC second and stop RTC time. The value of this register is undefined/non-deterministic when powered up for the first time.

**IOx00073\_x7F [RTC RAM Enable] (FCH::IO::RTCRAMEn)**

Read-write.

\_aliasIO; IOx00073\_x7F; IO=0000\_0000h; DataPortWrite=FCH::IO::AltRTCAddrPort\_aliasHOST; RTCHOSTx0007F; RTCHOST=FED8\_0700h

Bits	Description
7:1	Reserved.
0	<b>RtcRamEnable</b> . Read-write. Reset: X. 1=Enable access to the RTC RAM. The value of this register is undefined/non-deterministic when powered up for the first time.

**IOx00080 [PostCode 0] (FCH::IO::PostCode0)**

Read-write. Reset: 00h.

\_aliasIO; IOx00080; IO=0000\_0000h

Bits	Description
7:0	<b>PostCode[7:0]: BIOS post code</b> . Read-write. Reset: 00h. BIOS post code register. This can be 32 bits for Writes (using IOx008[3:1]) and 8 bits for Reads.

**IOx00092 [FastInit] (FCH::IO::FastInit)**

Read-write. Reset: 00h.

\_aliasIO; IOx00092; IO=0000\_0000h

Bits	Description
7:2	Reserved.
1	<b>A20EnB: A20 Enable Bar bit</b> . Read-write. Reset: 0. 1=A20M# function is disabled.
0	<b>FastInit</b> . Read-write. Reset: 0. 0=Before another INIT pulse can be generated via this register, this bit must be written back to a 0. 1=Generate INIT assertion for approximately 4 ms. This bit provides a fast software executed processor reset function.

**IOx00A0 [IntrCntrl2Reg1] (FCH::IO::IntrCntrl2Reg1)**

Read-write. Reset: 00h.

\_aliasIO; IOx00A0; IO=0000\_0000h

Bits	Description
7:0	<b>IntrCntrl2Reg1</b> . Read-write. Reset: 00h. IRQ[15:8] status and control. Read: IRR, ISR. Write: ICW1, OCW2, OCW3.

**IOx00A1 [IntrCntrl2Reg2] (FCH::IO::IntrCntrl2Reg2)**

Read-write. Reset: 00h.

\_aliasIO; IOx00A1; IO=0000\_0000h

Bits	Description
7:0	<b>IntrCntrl2Reg2</b> . Read-write. Reset: 00h. IRQ[15:8] status and control. Read: IMR. Write: ICW2, ICW3, ICW4, OCW1.

**IOx00C0 [Dma2\_Ch4Addr] (FCH::IO::Dma2Ch4Addr)**

Read-write. Reset: 00h.

\_aliasIO; IOx00C0; IO=0000\_0000h

Bits	Description
7:0	<b>Dma2Ch4Addr</b> . Read-write. Reset: 00h. DMA2 channel[4] base and current address.

**IOx00C2 [Dma2\_Ch4Cnt] (FCH::IO::Dma2Ch4Cnt)**

Read-write. Reset: 00h.

\_aliasIO; IOx00C2; IO=0000\_0000h

Bits	Description
7:0	<b>Dma2Ch4Cnt.</b> Read-write. Reset: 00h. DMA2 channel[4] base and current count.

**IOx00C4 [Dma2\_Ch5Addr] (FCH::IO::Dma2Ch5Addr)**

Read-write. Reset: 00h.

\_aliasIO; IOx00C4; IO=0000\_0000h

Bits	Description
7:0	<b>Dma2Ch5Addr.</b> Read-write. Reset: 00h. DMA2 channel[5] base and current address.

**IOx00C6 [Dma2\_Ch5Cnt] (FCH::IO::Dma2Ch5Cnt)**

Read-write. Reset: 00h.

\_aliasIO; IOx00C6; IO=0000\_0000h

Bits	Description
7:0	<b>Dma2Ch5Cnt.</b> Read-write. Reset: 00h. DMA2 channel[5] base and current count.

**IOx00C8 [Dma2\_Ch6Addr] (FCH::IO::Dma2Ch6Addr)**

Read-write. Reset: 00h.

\_aliasIO; IOx00C8; IO=0000\_0000h

Bits	Description
7:0	<b>Dma2Ch6Addr.</b> Read-write. Reset: 00h. DMA2 channel[6] base and current address.

**IOx00CA [Dma2\_Ch6Cnt] (FCH::IO::Dma2Ch6Cnt)**

Read-write. Reset: 00h.

\_aliasIO; IOx00CA; IO=0000\_0000h

Bits	Description
7:0	<b>Dma2Ch6Cnt.</b> Read-write. Reset: 00h. DMA2 channel[6] base and current count.

**IOx00CC [Dma2\_Ch7Addr] (FCH::IO::Dma2Ch7Addr)**

Read-write. Reset: 00h.

\_aliasIO; IOx00CC; IO=0000\_0000h

Bits	Description
7:0	<b>Dma2Ch7Addr.</b> Read-write. Reset: 00h. DMA2 channel[7] base and current address.

**IOx00CE [Dma2\_Ch7Cnt] (FCH::IO::Dma2Ch7Cnt)**

Read-write. Reset: 00h.

\_aliasIO; IOx00CE; IO=0000\_0000h

Bits	Description
7:0	<b>Dma2Ch7Cnt.</b> Read-write. Reset: 00h. DMA2 channel[7] base and current count.

**IOx00D0 [Dma2\_Status] (FCH::IO::Dma2Stat)**

Read-write. Reset: 00h.

\_aliasIO; IOx00D0; IO=0000\_0000h

Bits	Description
7:0	<b>Dma2Status.</b> Read-write. Reset: 00h. DMA2 status register.

**IOx00D2 [Dma2\_WriteRequest] (FCH::IO::Dma2WriteReq)**

Read-write. Reset: 00h.

\_aliasIO; IOx00D2; IO=0000\_0000h

Bits	Description
7:0	<b>Dma2WriteRequest.</b> Read-write. Reset: 00h. DMA2 request register.

**IOx00D4 [Dma2\_WriteMask] (FCH::IO::Dma2WriteMask)**

Read-write. Reset: 00h.

\_aliasIO; IOx00D4; IO=0000\_0000h

Bits	Description
7:0	<b>Dma2WriteMask.</b> Read-write. Reset: 00h. DMA2 channel mask register.

**IOx00D6 [Dma2\_WriteMode] (FCH::IO::Dma2WriteMode)**

Read-write. Reset: 00h.

\_aliasIO; IOx00D6; IO=0000\_0000h

Bits	Description
7:0	<b>Dma2WriteMode.</b> Read-write. Reset: 00h. DMA2 mode register.

**IOx00D8 [Dma2\_Clear] (FCH::IO::Dma2Clear)**

Read-write. Reset: 00h.

\_aliasIO; IOx00D8; IO=0000\_0000h

Bits	Description
7:0	<b>Dma2Clear.</b> Read-write. Reset: 00h. Channel[7:4] clear byte pointer.

**IOx00DA [Dma2\_MasterClr] (FCH::IO::Dma2MasClr)**

Read-write. Reset: 00h.

\_aliasIO; IOx00DA; IO=0000\_0000h

Bits	Description
7:0	<b>Dma2MasterClear.</b> Read-write. Reset: 00h. Write: Channel[7:4] DMA master clear. Read: Intermediate register.

**IOx00DC [Dma2\_ClrMask] (FCH::IO::Dma2ClrMask)**

Read-write. Reset: 00h.

\_aliasIO; IOx00DC; IO=0000\_0000h

Bits	Description
7:0	<b>Dma2ClrMask.</b> Read-write. Reset: 00h. Channel[7:4] DMA clear mask.

**IOx00DE [Dma2\_AllMask] (FCH::IO::Dma2AllMask)**

Read-write. Reset: 00h.

\_aliasIO; IOx00DE; IO=0000\_0000h

Bits	Description
7:0	<b>Dma2AllMask.</b> Read-write. Reset: 00h. DMA2 mask register.

**IOx00F0 [NCP Error] (FCH::IO::NCPerr)**

Read,Write-once. Reset: Cold,00h.

Writes to this port assert IGNNE# if FERR# is TRUE. If FERR# is FALSE, writes to this port do not assert IGNNE#.

The first write sets WarmBoot.

\_aliasIO; IOx00F0; IO=0000\_0000h

Bits	Description
7	<b>WarmBoot.</b> Read,Write-once. Reset: Cold,0. 0=Cold. 1=Warm. Warm or cold boot indicator.
6:0	Reserved.

**IOx04D0 [IntrEdgeControl] (FCH::IO::IntrEdgeCtl)**

Reset: 0000h.

\_aliasIO; IOx04D0; IO=0000\_0000h

Bits	Description
15	<b>IRQ15Control.</b> Read-write. Reset: 0. 0=Edge. 1=Level.
14	<b>IRQ14Control.</b> Read-write. Reset: 0. 0=Edge. 1=Level.
13	Reserved.
12	<b>IRQ12Control.</b> Read-write. Reset: 0. 0=Edge. 1=Level.
11	<b>IRQ11Control.</b> Read-write. Reset: 0. 0=Edge. 1=Level.
10	<b>IRQ10Control.</b> Read-write. Reset: 0. 0=Edge. 1=Level.
9	<b>IRQ9Control.</b> Read-write. Reset: 0. 0=Edge. 1=Level.



8	<b>IRQ8Control.</b> Read-only. Reset: 0. Always edge.
7	<b>IRQ7Control.</b> Read-write. Reset: 0. 0=Edge. 1=Level.
6	<b>IRQ6Control.</b> Read-write. Reset: 0. 0=Edge. 1=Level.
5	<b>IRQ5Control.</b> Read-write. Reset: 0. 0=Edge. 1=Level.
4	<b>IRQ4Control.</b> Read-write. Reset: 0. 0=Edge. 1=Level.
3	<b>IRQ3Control.</b> Read-write. Reset: 0. 0=Edge. 1=Level.
2	Reserved.
1	<b>IRQ1Control.</b> Read-write. Reset: 0. 0=Edge. 1=Level.
0	<b>IRQ0Control.</b> Read-write. Reset: 0. 0=Edge. 1=Level.

**IOx0C00 [Pci\_Intr\_Index] (FCH::IO::PciIntrIndex)**

Read-write. Reset: 00h.

\_aliasIO; IOx0C00; IO=0000\_0000h

Bits	Description
7	<b>PciIntrApic.</b> Read-write. Reset: 0. 0=IRQ routing to PIC. 1=IRQ routing to IOAPIC.
6:0	<b>PciIntrIndex.</b> Read-write. Reset: 00h. PCI interrupt index. Selects which PCI interrupt to map.
<b>ValidValues:</b>	
Value	Description
04h-00h	INT[E:A]#.
05h	INTF#/GENINT2.
06h	INTG#.
07h	INTH#.
08h	Misc.
0Bh-09h	Misc[2:0].
0Fh-0Ch	INT[D:A] from serial IRQ.
10h	SCI.
11h	SMBUS0.
12h	ASF.
15h-13h	Reserved.
16h	PerMon.
17h	SD.
19h-18h	Reserved.
1Ah	SDIO.
1Fh-1Bh	Reserved.
20h	CIR, no IRQ connected.
21h	GPIOa, from PAD_FANIN0.
22h	GPIOb, from PAD_FANOUT0.
23h	GPIOc, no IRQ connected.
40h-24h	Reserved.
41h	SATA PCI interrupt.
42h	Reserved.
43h	eMMC.
4Fh-44h	Reserved.
53h-50h	GPPInt3/2/1/0.
61h-54h	Reserved.
62h	GPIO controller interrupt.

6Fh-63h	Reserved.
70h	I2C0.
71h	I2C1.
72h	I2C2.
73h	I2C3.
75h-74h	UART1/UART0.
77h-76h	I2C5/I2C4.
79h-78h	UART3/UART2.
7Fh-7Ah	Reserved.

**IOx0C01 [Pci\_Intr\_Data] (FCH::IO::PciIntrData)**

Read-write. Reset: 00h.

\_aliasIO; IOx0C01; IO=0000\_0000h

Bits	Description
7:0	<b>PciIntrData.</b> Read-write. Reset: 00h. PCI interrupt data.

**IOx00C01\_x00 [PCI INT[H#,G#,F#,E#,D#,C#,B#,A#] Map] (FCH::IO::PciIntMap)**

Read-write. Reset: 1Fh.

\_aliasIO; IOx00C01\_x00; IO=0000\_0000h; DataPortWrite=FCH::IO::PciIntrIndex

Bits	Description
7:5	Reserved.
4:0	<b>Pci2IntrMap.</b> Read-write. Reset: 1Fh. If (FCH::IO::PciIntrIndex[PciIntrApic] == 1), then Pci2IntrMap specifies mapping of INT[H#:A#] to APIC interrupt number. If (FCH::IO::PciIntrIndex[PciIntrApic] == 0), then Pci2IntrMap specifies mapping of INT[H#:A#] to PIC interrupt number.

**IOx00C01\_x08 [Intr\_Misc\_Map] (FCH::IO::IntrMiscMap)**

Read-write. Reset: 00h.

\_aliasIO; IOx00C01\_x08; IO=0000\_0000h; DataPortWrite=FCH::IO::PciIntrIndex

Bits	Description										
7:6	<b>Pci2Intr15Map.</b> Read-write. Reset: 0h. <b>ValidValues:</b> <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>IRQ15 mapped to legacy IDE.</td></tr> <tr> <td>1h</td><td>IRQ15 mapped to SATA IDE.</td></tr> <tr> <td>2h</td><td>IRQ15 mapped to SATA2.</td></tr> <tr> <td>3h</td><td>IRQ15 come from Serial IRQ or PCI interrupt.</td></tr> </table>	Value	Description	0h	IRQ15 mapped to legacy IDE.	1h	IRQ15 mapped to SATA IDE.	2h	IRQ15 mapped to SATA2.	3h	IRQ15 come from Serial IRQ or PCI interrupt.
Value	Description										
0h	IRQ15 mapped to legacy IDE.										
1h	IRQ15 mapped to SATA IDE.										
2h	IRQ15 mapped to SATA2.										
3h	IRQ15 come from Serial IRQ or PCI interrupt.										
5:4	<b>Pci2Intr14Map.</b> Read-write. Reset: 0h. <b>ValidValues:</b> <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>IRQ14 mapped to legacy IDE.</td></tr> <tr> <td>1h</td><td>IRQ14 mapped to SATA IDE.</td></tr> <tr> <td>2h</td><td>IRQ14 mapped to SATA2.</td></tr> <tr> <td>3h</td><td>IRQ14 mapped to Serial IRQ or PCI interrupt.</td></tr> </table>	Value	Description	0h	IRQ14 mapped to legacy IDE.	1h	IRQ14 mapped to SATA IDE.	2h	IRQ14 mapped to SATA2.	3h	IRQ14 mapped to Serial IRQ or PCI interrupt.
Value	Description										
0h	IRQ14 mapped to legacy IDE.										
1h	IRQ14 mapped to SATA IDE.										
2h	IRQ14 mapped to SATA2.										
3h	IRQ14 mapped to Serial IRQ or PCI interrupt.										
3	<b>PciIntrIrq12.</b> Read-write. Reset: 0. 0=IMC as IRQ12 input source. 1=Serial IRQ or PCI devices as IRQ12 input source.										
2	<b>PciIntrIrq8.</b> Read-write. Reset: 0. 0=RTC is IRQ8 input source. 1=Serial IRQ or PCI devices as IRQ8 input source.										
1	<b>PciIntrIrq1.</b> Read-write. Reset: 0. 0=IMC as IRQ1 input source. 1=Serial IRQ or PCI devices as IRQ1 input source.										

0	<b>PciIntrIrq0</b> . Read-write. Reset: 0. 0=8254 timer as IRQ0 input source. 1=Serial IRQ or PCI devices as IRQ0 input source.
---	---

**IOx00C01\_x09 [Intr\_Misc0Map] (FCH::IO::IntrMisc0Map)**

Read-write. Reset: E7h.

\_aliasIO; IOx00C01\_x09; IO=0000\_0000h; DataPortWrite=FCH::IO::PciIntrIndex

Bits	Description
7	<b>IntrDelay</b> . Read-write. Reset: 1. INTR 600 ns delay.
6	<b>IRQ12FilterEnable</b> . Read-write. Reset: 1. IRQ12 filter enable.
5	<b>IRQ1FilterEnable</b> . Read-write. Reset: 1. IRQ1 filter enable.
4	<b>IrqInputEn</b> . Read-write. Reset: 0. 0=Mask off IRQ input. 1=Enable IRQ input.
3	<b>MaskIrq1Irq12</b> . Read-write. Reset: 0. 0=Enable IRQ1 and IRQ12. 1=Mask off IRQ1 and IRQ12.
2	<b>Merge_Ec_irq12</b> . Read-write. Reset: 1. 0=Route serial IRQ12 to USB IRQ12 input. 1=Route IMC IRQ12 to USB IRQ12 input.
1	<b>Merge_Ec_irq1</b> . Read-write. Reset: 1. 0=Route serial IRQ1 to USB IRQ1 input. 1=Route IMC IRQ1 to USB IRQ1 input.
0	<b>IntMap</b> . Read-write. Reset: 1. 0=INT0 in IOAPIC comes from IRQ0 in PIC, INT2 in IOAPIC comes from INTR in PIC. 1=INT2 in IOAPIC comes from IRQ0 in PIC, INT0 in IOAPIC comes from INTR in PIC.

**IOx00C01\_x0A [Intr\_Misc1Map] (FCH::IO::IntrMisc1Map)**

Read-write. Reset: 00h.

\_aliasIO; IOx00C01\_x0A; IO=0000\_0000h; DataPortWrite=FCH::IO::PciIntrIndex

Bits	Description
7:0	<b>HPET</b> . Read-write. Reset: 00h. FCH::IO::IntrMisc1Map updates the lower 8 bits. FCH::IO::IntrMisc2Map updates the upper 8 bits.

**IOx00C01\_x0B [Intr\_Misc2Map] (FCH::IO::IntrMisc2Map)**

Read-write. Reset: 00h.

\_aliasIO; IOx00C01\_x0B; IO=0000\_0000h; DataPortWrite=FCH::IO::PciIntrIndex

Bits	Description
7:0	<b>HPET</b> . Read-write. Reset: 00h. FCH::IO::IntrMisc1Map updates the lower 8 bits, FCH::IO::IntrMisc2Map updates the upper 8 bits.

**IOx00C01\_x0C [PciInterruptMap] (FCH::IO::PciInterruptMap)**

Read-write. Reset: 1Fh.

\_aliasIO; IOx00C01\_x0C; IO=0000\_0000h; DataPortWrite=FCH::IO::PciIntrIndex

Bits	Description
7:5	Reserved.
4:0	<b>Pci2IntrMap</b> . Read-write. Reset: 1Fh. If (FCH::IO::PciIntrIndex[PciIntrApic] == 1), then Pci2IntrMap specifies the APIC interrupt number that the corresponding PCI interrupt maps to. If (FCH::IO::PciIntrIndex[PciIntrApic] == 0), then Pci2IntrMap specifies the PIC interrupt number that the corresponding PCI interrupt maps to. See FCH::IO::PciIntrIndex for the PCI interrupt list.

**IOx0C14 [Pci\_Error] (FCH::IO::PciErr)**

\_aliasIO; IOx0C14; IO=0000\_0000h

Bits	Description
7:4	Reserved.
3	<b>PerrNmi</b> . Read-write. Reset: 1. 0=Enable. 1=Disable. Enable NMI generation from PERR#.
2	<b>SerrNmi</b> . Read-write. Reset: 1. 0=Enable. 1=Disable. Enable NMI generation from SERR#.
1	<b>PerrNmiStatus</b> . Read-only. Reset: X. 1=NMI generation is enabled and PERR# is asserted due to a PCI data parity error. This bit is cleared by writing FCH::IO::NmiStat[2] = 1.
0	<b>SerrNmiStatus</b> . Read-only. Reset: X. 1=NMI generation is enabled and SERR# is asserted due to a PCI error.

	This bit is cleared by writing FCH::IO::NmiStat[2] = 1.
--	---

**IOx0CD0 [PM2\_Index] (FCH::IO::PM2Index)**

Read-write. Reset: 00h.

\_aliasIO; IOx0CD0; IO=0000\_0000h

Bits	Description
7:0	<b>Pm2Index: Power management 2 index register.</b> Read-write. Reset: 00h. This register specifies the index of the power management 2 register.

**IOx0CD1 [PM2\_Data] (FCH::IO::PM2Data)**

Read-write. Reset: 00h.

\_aliasIO; IOx0CD1; IO=0000\_0000h

Bits	Description
7:0	<b>Pm2Data: Power management 2 data register.</b> Read-write. Reset: 00h. This register specifies the data Read from/Written to the power management 2 register pointed by FCH::IO::PM2Index.

**IOx0CD4 [BIOSRAM Index] (FCH::IO::BiosRAMIndex)**

Read-write. Reset: 00h.

\_aliasIO; IOx0CD4; IO=0000\_0000h

Bits	Description
7:0	<b>BiosRamIndex: BIOS RAM index register.</b> Read-write. Reset: 00h. This register specifies the index in the 256-byte BIOS RAM. Data in this RAM is preserved until RSMRST# is asserted or S5 power is lost.

**IOx0CD5 [BIOSRAM Data] (FCH::IO::BiosRAMData)**

Read-write. Reset: 00h.

\_aliasIO; IOx0CD5; IO=0000\_0000h

Bits	Description
7:0	<b>BiosRamData: BIOS RAM data register.</b> Read-write. Reset: 00h. This register specifies the data Read from/Written to the BIOS RAM pointed by FCH::IO::BiosRAMIndex.

**IOx0CD6 [PM\_Index] (FCH::IO::PmIndex)**

Read-write. Reset: 00h.

\_aliasIO; IOx0CD6; IO=0000\_0000h

Bits	Description
7:0	<b>PmIndex: Power management index register.</b> Read-write. Reset: 00h. This register specifies the index of the power management register. See 9.2.10.2 [Power Management (PM) Registers].

**IOx0CD7 [PM\_Data] (FCH::IO::PmData)**

Read-write. Reset: 00h.

\_aliasIO; IOx0CD7; IO=0000\_0000h

Bits	Description
7:0	<b>PmData: Power management data register.</b> Read-write. Reset: 00h. This register specifies the data Read from or Written to the power management register pointed by FCH::IO::PmIndex. See 9.2.1 [Legacy Block Configuration Registers (IO)].

**IOx0CF9 [System Reset Register] (FCH::IO::SysReset)**

Reset: 00h.

This register can be accessed through FCH::PM::CF9Shadow.

\_aliasIO; IOx0CF9; IO=0000\_0000h

Bits	Description
7:4	Reserved.
3	<b>FullRst.</b> Read-write. Reset: 0. 0=Assert reset signals only. 1=Place system in S5 state for 3 to 5 seconds.
2	<b>RstCmd.</b> Read-write, Volatile. Reset: 0. 1=Generate reset as specified by FullRst and SysRst.

1	<b>SysRst.</b> Read-write. Reset: 0. 0=Send an INIT HT message. 1=Reset as specified by FullRst.
0	Reserved.

## 9.2.2 eMMC

This embedded MultiMedia Card (eMMC) controller is compliant to the eMMC v5.0 specification and is capable of speed up to 400MB/s.

### 9.2.2.1 eMMC Configuration Registers

This chapter describes eMMC configuration registers.

<b>EMMCCFGx00000000 (FCH::EMMC::EMMC_DEV_VEN_ID)</b>	
Read-only. Reset: 7813_1022h.	
_aliasHOST; EMMCCFGx00000000; EMMCCFG=FEDD_5800h	
Bits	Description
31:16	<b>DEV_ID.</b> Read-only. Reset: 7813h. Device ID.
15:0	<b>VEND_ID.</b> Read-only. Reset: 1022h. Vendor ID.
<b>EMMCCFGx00000004 (FCH::EMMC::EMMC_CMD_STS)</b>	
_aliasHOST; EMMCCFGx00000004; EMMCCFG=FEDD_5800h	
Bits	Description
31	<b>Det_Perr.</b> Read-only. Reset: 0. This bit is set by the device whenever it detects a parity error, even if parity error handling is disabled (as controlled by bit[6] in the Command register).
30	<b>Sig_SysErr.</b> Read-only. Reset: 0. Set whenever the device asserts SERR#.
29	<b>Rec_Mabort.</b> Read-only. Reset: 0. This bit is set by a master device whenever its transaction (except for Special Cycle) is terminated with Master-Abort.
28	<b>Rec_Tabort.</b> Read-only. Reset: 0. This bit is set by a master device whenever its transaction is terminated with Target-Abort.
27	<b>Sig_Tabort.</b> Read-only. Reset: 0. This bit is set by a target device whenever it terminates a transaction with Target-Abort.
26:25	<b>DEVSEL_Timing.</b> Read-only. Reset: Fixed,1h. Medium timing.
24	<b>Master_DPerr.</b> Read-only. Reset: 0. Master Data Parity Error.
23	<b>Fast_B2B_Cap.</b> Read-only. Reset: Fixed,0. 0=Disallow. 1=Allow. Allow Fast Back-to-Back capability.
22	<b>PCI_ST_Reserved2.</b> Read-only. Reset: Fixed,0. Reserved.
21	<b>En_66MHz.</b> Read-only. Reset: Fixed,1. Indicating 66MHz capable.
20	<b>Cap_List.</b> Read-only. Reset: 1. Indicating no Capabilities Linked List available.
19	<b>Int_status.</b> Read-only. Reset: 0. Reflects the state of the interrupt in the device/function.
18:16	<b>PCI_ST_Reserved1.</b> Read-only. Reset: Fixed,0h. Reserved.
15:11	<b>PCI_CMD_Reserved2.</b> Read-write. Reset: Fixed,00h. Reserved.
10	<b>Int_Dis.</b> Read-write. Reset: 0. 0=Enables the assertion of the device/function's INTx# signal. 1=Disable. INTx# signal enable.
9	<b>Fast_B2B_En.</b> Read-only. Reset: Fixed,0. Hard-wired to 0, indicating fast back-to-back transactions only to the same agent are allowed.
8	<b>SERRB_En.</b> Read-write. Reset: 0. 0=Disable the SERR# driver. 1=Enable the SERR# driver. SERR# driver enable.
7	<b>PCI_CMD_Reserved1.</b> Read-only. Reset: Fixed,0. Reserved.
6	<b>Parity_En.</b> Read-write. Reset: 0. 0=Parity check disabled. 1=Device must take action when a parity error is detected. Parity check enabled.
5	<b>VGA_Pal_Access.</b> Read-only. Reset: Fixed,0. 0=Disallow. 1=Allow. Allow snoop VGA palette cycles.

4	<b>MemWr_Inv_Cmd.</b> Read-only. Reset: Fixed,0. 0=Disallow. 1=Allow. Allow Memory Write and Invalidate command.
3	<b>Special_Cycle.</b> Read-only. Reset: Fixed,0. 0=Disable. 1=Enable. Special cycle recognition enable.
2	<b>Bus_Master.</b> Read-write. Reset: 0. 0=Disallow. 1=Allow. Allow the device to behave as a bus master.
1	<b>Mem_Space_Access.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. Memory Space access enable.
0	<b>IO_Space_Access.</b> Read-only. Reset: Fixed,0. 0=Disable. 1=Enable. IO Space access enable.

**EMMCCFGx00000008 (FCH::EMMC::EMMC\_REV\_ID)**

\_aliasHOST; EMMCCFGx00000008; EMMCCFG=FEDD\_5800h

Bits	Description
31:24	<b>BC.</b> Read-write. Reset: 08h. Base Class Code. Hard-wired to 08h, indicating general peripheral.
23:16	<b>SC.</b> Read-write. Reset: 05h. Sub Class Code. Hard-wired to 05h, indicating SD Host controller.
15:8	<b>PI.</b> Read-only. Reset: 01h. Programming Interface Code. Hard-wired to 01h for Standard host supporting DMA.
7:0	<b>Revision_ID.</b> Read-only. Reset: Fixed,01h. Revision ID.

**EMMCCFGx0000000C (FCH::EMMC::EMMC\_MISC)**

\_aliasHOST; EMMCCFGx0000000C; EMMCCFG=FEDD\_5800h

Bits	Description
31:24	<b>BIST.</b> Read-only. Reset: Fixed,00h. Hard-wired to 00h, indicating no built-in BIST support.
23:16	<b>Header_Type.</b> Read-only. Reset: Fixed,80h. Bit[23] hard-wired to 1, indicating a single-function device. Bits[22:16] are hard-wired to 00h.
15:11	<b>Latency_Timer.</b> Read-write. Reset: 00h. Latency_Timer_HW (bits[10:8]) is hard-wired to 000b, resulting in a timer granularity of at least eight clocks. This field specifies, in units of PCI bus clocks, the value of the Latency Timer for this PCI bus master.
10:8	<b>Latency_Timer_HW.</b> Read-only. Reset: Fixed,0h. Specifies the value of the Latency Timer in units of PCICLKs.
7:0	<b>Cache_Size.</b> Read-write. Reset: 00h. This Read-write field specifies the system cacheline size in units of DWORDs and must be initialized to 00h.

**EMMCCFGx00000010 (FCH::EMMC::EMMC\_BAR)**

\_aliasHOST; EMMCCFGx00000010; EMMCCFG=FEDD\_5800h

Bits	Description
31:8	<b>BAR.</b> Read-write. Reset: 00_0000h. Base Address. Specifies the upper 15 bits of the 32-bit starting base address.
7:4	<b>BAR_Reserved.</b> Read-only. Reset: Fixed,0h. Reserved.
3	<b>PM.</b> Read-only. Reset: Fixed,0. Prefetch memory. A constant value of 0 indicates that there is no support for prefetchable memory.
2:1	<b>TP.</b> Read-only. Reset: 2h. 10b indicates that the base register has a 64-bit width.
0	<b>IND.</b> Read-only. Reset: Fixed,0. Resource Type Indicator. A constant value of 0 indicates that the operational registers of the device are mapped into memory space of the main memory of the PC host system.

**EMMCCFGx00000014 (FCH::EMMC::EMMC\_UPPER\_BAR)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; EMMCCFGx00000014; EMMCCFG=FEDD\_5800h

Bits	Description
31:0	<b>UBAR.</b> Read-write. Reset: 0000_0000h. Upper 32 bits of the Base Address.

**EMMCCFGx0000002C (FCH::EMMC::EMMC\_SUB\_VEN\_SYS\_ID)**

Read-write. Reset: 7806\_1022h.

\_aliasHOST; EMMCCFGx0000002C; EMMCCFG=FEDD\_5800h

Bits	Description
31:16	<b>Sub_System_ID.</b> Read-write. Reset: 7806h. Can only be written once by software.
15:0	<b>Sub_Vendor_ID.</b> Read-write. Reset: 1022h. Can only be written once by software.

**EMMCCFGx00000034 (FCH::EMMC::EMMC\_CAP\_PTR)**

Read-only. Reset: 80h.

\_aliasHOST; EMMCCFGx00000034; EMMCCFG=FEDD\_5800h

Bits	Description
7:0	<b>CAP_PTR.</b> Read-only. Reset: 80h. The first pointer of Capability block. Points to the FLR register.

**EMMCCFGx0000003C (FCH::EMMC::EMMC\_INT\_LINE)**

Reset: 0000\_0100h.

\_aliasHOST; EMMCCFGx0000003C; EMMCCFG=FEDD\_5800h

Bits	Description
31:24	<b>MAX_LAT.</b> Read-only. Reset: 00h. Hardwired to 00h to indicate no major requirements for the settings of Latency Timers.
23:16	<b>MIN_GNT.</b> Read-only. Reset: 00h. Hardwired to 00h to indicate no major requirements for the settings of Latency Timers.
15:8	<b>Int_Pin.</b> Read-write. Reset: 01h. Hard-wired to 01h, corresponding to using INTA#.
7:0	<b>Int_Line.</b> Read-write. Reset: 00h. The Interrupt Line register used to communicate interrupt line routing information.

**EMMCCFGx00000040 (FCH::EMMC::EMMC\_SLOT\_INFORMATION)**

Read-only.

\_aliasHOST; EMMCCFGx00000040; EMMCCFG=FEDD\_5800h

Bits	Description
31:8	Reserved.
7	<b>Slot_Reserved2.</b> Read-only. Reset: Fixed,0. Reserved.
6:4	<b>Num_Of_Slot.</b> Read-only. Reset: Fixed,0h. Hardwired to 000h to indicate only 1 slot.
3	<b>Slot_Reserved1.</b> Read-only. Reset: Fixed,0. Reserved.
2:0	<b>First_BAR.</b> Read-only. Reset: 0h. Hardwired to 000b to indicate BAR0 (Base address 10h).

**EMMCCFGx00000080 (FCH::EMMC::EMMC\_MSI\_CAP\_HEADER)**

Read-only. Reset: 9005h.

\_aliasHOST; EMMCCFGx00000080; EMMCCFG=FEDD\_5800h

Bits	Description
15:8	<b>CAP_NXT_PTR.</b> Read-only. Reset: 90h. Pointer to the next item in the capabilities list.
7:0	<b>CAP_ID.</b> Read-only. Reset: 05h. 05h indicates MSI.

**EMMCCFGx00000082 (FCH::EMMC::EMMC\_MSI\_CTRL)**\_aliasHOST; EMMCCFGx00000082; EMMCCFG=FEDD\_5800h

Bits	Description
15:8	<b>MSI_CTRL_Reserved.</b> Read-only. Reset: Fixed,00h. Reserved.
7	<b>Extend_Addr_En.</b> Read-only. Reset: 1. 64-bit address capable.
6:4	<b>Mul_Msg_En.</b> Read-write. Reset: 0h. Multiple message enable.
3:1	<b>Mul_Msg_Cap.</b> Read-only. Reset: Fixed,0h. Multiple message capable.
0	<b>MSI_Enable.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. MSI enable.

**EMMCCFGx00000084 (FCH::EMMC::EMMC\_MSI\_ADDR)**\_aliasHOST; EMMCCFGx00000084; EMMCCFG=FEDD\_5800h

Bits	Description
31:2	<b>Msg_Addr.</b> Read-write. Reset: 0000_0000h. Message address.
1:0	<b>MSI_ADDR_Reserved.</b> Read-only. Reset: Fixed,0h. Reserved.

**EMMCCFGx00000088 (FCH::EMMC::EMMC\_MSI\_U\_ADDR)**

Read-write. Reset: 0000\_0000h.



_aliasHOST; EMMCCFGx00000088; EMMCCFG=FEDD_5800h	
Bits	Description
31:0	<b>Msg_Upper_Addr.</b> Read-write. Reset: 0000_0000h. Message upper address. Msg_Addr={Msg_Upper_Addr,FCH::EMMC::EMMC_MSI_ADDR[Msg_Addr]}

**EMMCCFGx0000008C (FCH::EMMC::EMMC\_MSI\_DATA)**

Read-write. Reset: 0000h.

\_aliasHOST; EMMCCFGx0000008C; EMMCCFG=FEDD\_5800h

Bits	Description
15:0	<b>Msg_Data.</b> Read-write. Reset: 0000h. Message data.

**EMMCCFGx00000090 (FCH::EMMC::EMMC\_PMC\_CAP\_HEADER)**

Read-only. Reset: 0001h.

\_aliasHOST; EMMCCFGx00000090; EMMCCFG=FEDD\_5800h

Bits	Description
15:8	<b>CAP_NXT_PTR.</b> Read-only. Reset: 00h. Pointer to the next item in the capabilities list.
7:0	<b>CAP_ID.</b> Read-only. Reset: 01h. Power management.

**EMMCCFGx00000092 (FCH::EMMC::EMMC\_PMC\_CAP)**

Read-only. Reset: Fixed,8003h.

\_aliasHOST; EMMCCFGx00000092; EMMCCFG=FEDD\_5800h

Bits	Description
15:11	<b>PME_Support.</b> Read-only. Reset: Fixed,10h. This 5-bit field indicates the power states in which the function may assert PME#. A value of 0 for any bits indicates that the function is not capable of asserting the PME# signal while in that power state.
10	<b>D2_Support.</b> Read-only. Reset: Fixed,0. D2 Power Management state.
9	<b>D1_Support.</b> Read-only. Reset: Fixed,0. D1 Power Management state.
8:6	<b>Aux_Current.</b> Read-only. Reset: Fixed,0h. 3.3 Vaux auxiliary current requirements.
5	<b>DSI.</b> Read-only. Reset: Fixed,0. Device specific initialization.
4	<b>PMC_Reserved.</b> Read-only. Reset: Fixed,0. Reserved.
3	<b>PME_Clock.</b> Read-only. Reset: Fixed,0. If PME depends on PCI clock.
2:0	<b>PMC_Version.</b> Read-only. Reset: Fixed,3h. 011b indicates compliance with revision 1.2 of the PCI PM Interface specification.

**EMMCCFGx00000094 (FCH::EMMC::EMMC\_PMCSR)**

\_aliasHOST; EMMCCFGx00000094; EMMCCFG=FEDD\_5800h

Bits	Description
15	<b>PME_Status.</b> Read-write. Reset: 0. Set when the function normally asserts the PME# signal independent of the state of PME# Enable.
14:13	<b>Data_Scale.</b> Read-only. Reset: Fixed,0h. Indicates the scaling factor to be used when interpreting the value of the Data register. The value and meaning of this field varies depending on which data value has been selected by the DataSelect field.
12:9	<b>Data_Select.</b> Read-write. Reset: 0h. Selects which data is to be reported through the Data register and DataScale field.
8	<b>PME_EN.</b> Read-write. Reset: 0. 0=PME# assertion is disabled. 1=Enable the function to assert PME#. PME# Enable.
7:4	<b>PMCSR_Reserved2.</b> Read-only. Reset: Fixed,0h. Reserved.
3	<b>No_Soft_Reset.</b> Read-only. Reset: Fixed,1. 1=Indicates that devices transitioning from D3hot to D0 because of power state commands do not perform an internal reset.
2	<b>PMCSR_Reserved1.</b> Read-only. Reset: Fixed,0. Reserved.
1:0	<b>PowerState.</b> Read-write. Reset: 0h. Initiates power state.

**EMMCCFGx00000096 (FCH::EMMC::EMMC\_PMC\_BSE)**

Read-only. Reset: Fixed,00h.

\_aliasHOST; EMMCCFGx00000096; EMMCCFG=FEDD\_5800h

Bits	Description
7	<b>BPCC_En.</b> Read-only. Reset: Fixed,0. Bus Power/Clock Control enable.
6	<b>B2_B3.</b> Read-only. Reset: Fixed,0. B2/B3 support for D3hot.
5:0	<b>PMC_BSE_Reserved.</b> Read-only. Reset: Fixed,00h. Reserved.

**EMMCCFGx00000097 (FCH::EMMC::EMMC\_PMC\_DATA)**

Read-only. Reset: 00h.

\_aliasHOST; EMMCCFGx00000097; EMMCCFG=FEDD\_5800h

Bits	Description
7:0	<b>PMC_Data.</b> Read-only. Reset: 00h. Reports the state dependent data requested by the DataSelect field. The value of this register is scaled by the value reported by the DataScale field.

**EMMCCFGx00000108 (FCH::EMMC::EMMC\_DLL\_CFG)**

Read-write. Reset: 3232h.

\_aliasHOST; EMMCCFGx00000108; EMMCCFG=FEDD\_5800h

Bits	Description
15:8	<b>I_PARAM_INCREMENT.</b> Read-write. Reset: 32h. Next step search value, when first search fails. Next jump step search value. Good value setting will reduce initial time when first lock searching. Typical value = start_point.
7:0	<b>I_PARAM_START_POINT.</b> Read-write. Reset: 32h. Initial DLL search value. Good value setting decreases initial time when first lock searching. Assumes every delay cell == 0.1 ns. For 200MHz, set <VALUE> = 50 (32h). For 100MHz, set <VALUE> = 100 (64h).

**EMMCCFGx0000010A (FCH::EMMC::EMMC\_DLL\_CFG1)**

Read-write. Reset: 3F00h.

\_aliasHOST; EMMCCFGx0000010A; EMMCCFG=FEDD\_5800h

Bits	Description
15:8	<b>I_DELAY.</b> Read-write. Reset: 3Fh. DLL delay select. (I_BYPASS == 0) ? (This register is used to phase shift setting. 7Fh=Full cycle delay; 2Fh=1/4 cycle delay; 80h=Before one cycle; ...) : (This register is used to statically delay the cell select.)
7:5	Reserved.
4	<b>I_BYPASS.</b> Read-write. Reset: 0. 0=Disabled. 1=Enabled. DLL bypass. When enabled, the delay cell value is I_DELAY (bits[15:8]).
3	Reserved.
2	<b>I_RESYNC.</b> Read-write. Reset: 0. 0=No effect. 1=Re-sync. DLL re-sync dll slave line.
1	<b>SOFT_RESET.</b> Read-write. Reset: 0. 0=No effect. 1=Reset. DLL reset. Active is low.
0	<b>DLL_ENABLE.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. DLL enable. Default is off for saving power.

**EMMCCFGx0000010C (FCH::EMMC::EMMC\_DLL\_CFG2)**

Reset: 2F00h.

\_aliasHOST; EMMCCFGx0000010C; EMMCCFG=FEDD\_5800h

Bits	Description
15:8	<b>O_LOCK_VALUE.</b> Read-only. Reset: 2Fh. DLL lock value, which can be obtained by software to view the DLL state.
7:1	Reserved.
0	<b>O_LOCK.</b> Read-only. Reset: 0. 0=DLL not locked. 1=DLL locked. DLL lock flag. When asserted, the DLL is locked, then software can do next things.

### 9.2.2.2 eMMC Host Controller Registers

This chapter describes eMMC Host Controller registers.

EMMCHC <sub>x00000000</sub> (FCH::EMMC::MMIO::EMMCHC_SYS_ADDR)	
Read-write. Reset: 0000_0000h.	
_aliasHOST; EMMCHC <sub>x00000000</sub> ; EMMCHC=FEDD_5000h	
Bits	Description
31:0	<b>SYS_ADDR.</b> Read-write. Reset: 0000_0000h. By updating this register, DMA_WAIT will be cleared. System Address. It indicates system memory address for DMA. When DMA transfer detects the DMA Buffer Boundary specified by the Host DMA Buffer Boundary in the Block Size register, SD controller asserts DMA_WAIT. Also SD controller generates DMA interrupt at this time in the case that corresponding bits in the Normal Interrupt Status Enable register and Normal Interrupt Signal Enable register are set. While ADMA is enabled, this register will not be used.
EMMCHC <sub>x00000004</sub> (FCH::EMMC::MMIO::EMMCHC_BLK_CS)	
_aliasHOST; EMMCHC <sub>x00000004</sub> ; EMMCHC=FEDD_5000h	
Bits	Description
31:16	<b>BLK_CNT.</b> Read-write. Reset: 0000h. Block Count. It indicates the block count of multiple data transfers. It is enabled when the Block Count Enable bit (D01) in the Transfer Mode Register is set to 1. It is decremented after each block data transmission. During infinite data transmission, setting of this bit is meaningless. Block Size and Block Count.
15	Reserved.
14:12	<b>SDMA_BUF_BNDRY.</b> Read-write. Reset: 0h. Host DMA Buffer Boundary. Indicates the contiguous buffer size in the system memory. Internally, when this boundary is reached, DMA interrupt will be generated.
<b>ValidValues:</b>	
Value	Description
0h	4K bytes.
1h	8K bytes.
2h	16K bytes.
3h	32K bytes.
4h	64K bytes.
5h	128K bytes.
6h	256K bytes.
7h	512K bytes.
11:0	<b>BLK_SIZE.</b> Read-write. Reset: 000h. Transfer Data Length (Maximum block size is 2K bytes). When the CE-ATA enable bit of the CE-ATA control register is set, a value of 0x000 indicates block size of 4K bytes.
EMMCHC <sub>x00000008</sub> (FCH::EMMC::MMIO::EMMCHC_CMD_ARG)	
Read-write. Reset: 0000_0000h.	
_aliasHOST; EMMCHC <sub>x00000008</sub> ; EMMCHC=FEDD_5000h	
Bits	Description
31:0	<b>ARGUMENT1.</b> Read-write. Reset: 0000_0000h. Command Argument. Command arguments specified as bits[39-8] of the command format.
EMMCHC <sub>x0000000C</sub> (FCH::EMMC::MMIO::EMMCHC_CMD_TRN)	
_aliasHOST; EMMCHC <sub>x0000000C</sub> ; EMMCHC=FEDD_5000h	
Bits	Description
31:30	Reserved.
29:24	<b>CMD_IDX.</b> Read-write. Reset: 00h. Command index.
23:22	<b>CMD_TYPE.</b> Read-write. Reset: 0h. Command type.
<b>ValidValues:</b>	

	Value	Description
	0h	Normal.
	1h	Suspend CMD52 for writing BR in CCCR.
	2h	Resume CMD52 for writing Function Select in CCCR.
	3h	Abort CMD12 (SD Memory) or Abort CMD52 (SDIO).
21	<b>DATA_PRSENT</b> . Read-write. Reset: 0. 0=No data. 1=Data. Data Present Select. Indicates that data is present and will be transferred on the DAT line. When a command is issued with this bit enabled, the internal buffer will be cleared.	
20	<b>CMD_IDX_CHK_EN</b> . Read-write. Reset: 0. 0=Disable. 1=Enable. Command Index Check enable.	
19	<b>CRC_CHK_EN</b> . Read-write. Reset: 0. 0=Disable. 1=Enable. Command CRC Check enable.	
18	Reserved.	
17:16	<b>RESP_TYPE</b> . Read-write. Reset: 0h. Response type select.	
	<b>ValidValues:</b>	
	Value	Description
	0h	No Response.
	1h	Response length of 136 bits.
	2h	Response length of 48 bits with no busy.
	3h	Response length of 48 bits with busy.
15:6	Reserved.	
5	<b>MULTI_BLK</b> . Read-write. Reset: 0. 0=Single block. 1=Multiple blocks. Multiple/Single block select.	
4	<b>DATA_DIR</b> . Read-write. Reset: 0. 0=Write. 1=Read. Data Transfer Direction.	
3:2	<b>ACMD_EN</b> . Read-write. Reset: 0h. Auto CMD Enable.	
	<b>ValidValues:</b>	
	Value	Description
	0h	Disable.
	1h	Enable.
	3h-2h	Reserved.
1	<b>BLK_CNT_EN</b> . Read-write. Reset: 0. 0=Disable. 1=Enable. Block Count Enable.	
0	<b>DMA_EN</b> . Read-write. Reset: 0. 0=Disable. 1=Enable. DMA Enable.	

**EMMCHCx00000010 (FCH::EMMC::MMIO::EMMCHC\_RESP1\_0)**

Read-only. Reset: 0000\_0000h.

\_aliasHOST; EMMCHCx00000010; EMMCHC=FEDD\_5000h

Bits	Description
31:16	<b>RESPONSE1</b> . Read-only. Reset: 0000h. R[39:24] of the response is saved in this field. The response value is preseved until the next response.
15:0	<b>RESPONSE0</b> . Read-only. Reset: 0000h. R[23:8] of the response is saved in this field.

**EMMCHCx00000014 (FCH::EMMC::MMIO::EMMCHC\_RESP3\_2)**

Read-only. Reset: 0000\_0000h.

\_aliasHOST; EMMCHCx00000014; EMMCHC=FEDD\_5000h

Bits	Description
31:16	<b>RESPONSE3</b> . Read-only. Reset: 0000h. R[71:56] of the response is saved in this field. The response value is preseved until the next response.
15:0	<b>RESPONSE2</b> . Read-only. Reset: 0000h. R[55:40] of the response is saved in this field.

**EMMCHCx00000018 (FCH::EMMC::MMIO::EMMCHC\_RESP5\_4)**

Read-only. Reset: 0000\_0000h.

\_aliasHOST; EMMCHCx00000018; EMMCHC=FEDD\_5000h

Bits	Description
------	-------------

31:16	<b>RESPONSE5.</b> Read-only. Reset: 0000h. R[103:88] of the response is saved in this field. The response value is preseeded until the next response.
15:0	<b>RESPONSE4.</b> Read-only. Reset: 0000h. R[87:72] of the response is saved in this field.

**EMMCHCx0000001C (FCH::EMMC::MMIO::EMMCHC\_RESP7\_6)**

Read-only. Reset: 0000\_0000h.

\_aliasHOST; EMMCHCx0000001C; EMMCHC=FEDD\_5000h

Bits	Description
31:16	<b>RESPONSE7.</b> Read-only. Reset: 0000h. R[127:120] of the response or R[39:24] of the Auto CMD12 response is saved in this field. The response value is preseeded until the next response.
15:0	<b>RESPONSE6.</b> Read-only. Reset: 0000h. R[119:104] of the response or R[23:8] of the Auto CMD12 response is saved in this field.

**EMMCHCx00000020 (FCH::EMMC::MMIO::EMMCHC\_BUF\_PORT)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; EMMCHCx00000020; EMMCHC=FEDD\_5000h

Bits	Description
31:0	<b>DATA_BUF_PORT.</b> Read-write. Reset: 0000_0000h. Data Buffer. Data will be accessed through this register. Data which exceeds the size designated by the Block Size Register will not be written in the data buffer.

**EMMCHCx00000024 (FCH::EMMC::MMIO::EMMCHC\_PRNT\_STATE)**

Read-only.

\_aliasHOST; EMMCHCx00000024; EMMCHC=FEDD\_5000h

Bits	Description
31:29	Reserved.
28:25	<b>DAT_LEVEL2.</b> Read-only. Reset: 0h. DAT[7:4].
24	<b>CMD_LEVEL.</b> Read-only. Reset: 0. CMD Line Signal Level. Reflects signal level of CMD line. This is used for testing.
23:20	<b>DAT_LEVEL1.</b> Read-only. Reset: 0h. DAT Line Signal level. Reflects signal level of DAT line. This is used for testing.
19	<b>WP_LEVEL.</b> Read-only. Reset: 0. 0=Write protected. 1=Write enable. Write Protect Switch level.
18	<b>CD_LEVEL.</b> Read-only. Reset: 0. 0=No card present. 1=Card present. Card Detect Pin level. This bit is used for testing.
17	<b>CARD_STABLE.</b> Read-only. Reset: 0. 0=Not stable (debouncing or resetting). 1=Card stable. Card state stable. Indicates Card Detect signal level is stable. This bit is used for testing.
16	<b>CARD_INSERT.</b> Read-only. Reset: 0. 0=No card inserted or debouncing state or resetting. 1=Card inserted. Card inserted.
15:12	Reserved.
11	<b>BUF_RD_EN.</b> Read-only. Reset: 0. 0=Read disable. 1=Read enable. Buffer Read Enable. Indicates buffer is ready for reading.
10	<b>BUF_WR_EN.</b> Read-only. Reset: 0. 0=Write disable. 1=Write enable. Buffer Write Enable. Indicates buffer is ready for Writing.
9	<b>RD_TX_ACT.</b> Read-only. Reset: 0. 0=No data transferring. 1=Read data transferring. Read Transfer Active. Indicates duration of Read data transfer.
8	<b>WR_TX_ACT.</b> Read-only. Reset: 0. 0=No data transferring. 1=Write data transferring. Write Transfer Active. Indicates duration of Write data transfer.
7:3	Reserved.
2	<b>DAT_LINE_ACT.</b> Read-only. Reset: 0. 0=DAT line inactive. 1=DAT line active. DAT Line Active. Indicates the DAT line on SD Bus is active.
1	<b>CMD_INHIBIT_DAT.</b> Read-only. Reset: 0. 0=Can issue commands which use the DAT line. 1=Cannot issue any commands which use the DAT line. Command Inhibit (DAT). Indicates that commands which also use the DAT line can be issued.

0	<b>CMD_INHIBIT_CMD.</b> Read-only. Reset: 0. 0=Can issue commands which use the CMD line. 1=Cannot issue any commands. Command Inhibit (CMD). Indicates that commands which use only the CMD line can be issued.
---	--

**EMMCHC<sub>x00000028</sub> (FCH::EMMC::MMIO::EMMCHC\_CTRL1)**\_aliasHOST; EMMCHC<sub>x00000028</sub>; EMMCHC=FEDD\_5000h

Bits	Description
31:27	Reserved.
26	<b>WAKEUP_RM_EN.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. SD Card Removal Wakeup.
25	<b>WAKEUP_INS_EN.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. SD Card Insertion Wakeup.
24	<b>WAKEUP_INT_EN.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. SD Card Interrupt Wakeup.
23:20	Reserved.
19	<b>BG_INT_EN.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. Interrupt at Block Gap. Enable interrupt detection during 4-bit block transmission.
18	<b>RD_WAIT_EN.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. Read Wait Control. Indicates Read Wait will be inserted when needed.
17	<b>CONT_REQ.</b> Read-write. Reset: 0. 0=No effect. 1=Restart. Continue Request. Writing 1 to this bit triggers restart of halted data transaction with current register setting. Once this bit is 1, the internal buffer will be cleared and data transfer sequence will be restarted.
16	<b>BG_STOP_REQ.</b> Read-write. Reset: 0. 0=Transfer. 1=Stop. Stop at Block Gap Request. Writing 1 to this bit triggers halting of the current data transfer after the next block gap. For using this request, the Read Wait function is necessary in a Read transaction. Even if the Auto CMD12 enable bit is set to 1, Auto CMD12 is not issued in case this bit is set to 1. This bit is cleared by not only writing 0 to this bit, but also issuing abort commands.
15:12	Reserved.
11:9	<b>BUS_VOLT_SEL.</b> Read-write. Reset: 0h. Reserved. Not used in EMMC. Use FCH::EMMC::MMIO::EMMCHC_ACMD_ERR_CTRL2[VOL18_EN] instead.
8	<b>BUS_PWR.</b> Read-write. Reset: 0. 0=Off. 1=On. SD Bus Power. When card is removed, this bit is cleared automatically.
7	<b>CD_SIG_SEL.</b> Read-write. Reset: 0. 0=IO pin. 1=SD_TEST_LEVEL. Card Detect Signal selection. This is for testing.
6	<b>CD_TEST_LEVEL.</b> Read-write. Reset: 0. 0=Card removed. 1=Card inserted. Card Detect Test Level. This is for testing.
5	<b>EXT_WIDTH.</b> Read-write. Reset: 0. 0=Use width set by DAT_XFER_WIDTH. 1=8-bit width. Extended Data Transfer Width (MMC).
4:3	<b>DMA_SEL.</b> Read-write. Reset: 0h. DMA Select. Valid only when DMA is enabled.
<b>Valid Values:</b>	
Value	Description
0h	No DMA or SDMA selected.
1h	Reserved.
2h	32-bit ADMA.
3h	64-bit ADMA
2	<b>HS_EN.</b> Read-write. Reset: 0. 0=Normal speed. 1=High-speed. High Speed Enable. When disabled, SD controller outputs commands and data on the falling edge of the SD clock (Up to 25 MHz SD clock can be supported). When enabled, SD controller outputs commands and data on the rising edge of the SD clock (Up to 50 MHz SD clock can be supported).
1	<b>DAT_XFER_WIDTH.</b> Read-write. Reset: 0. 0=1-bit. 1=4-bit. Data Transfer Width.
0	<b>LED_CTRL.</b> Read-write. Reset: 0. 0=Off. 1=On. LED control. Drives the LED_ON output.

**EMMCHC<sub>x0000002C</sub> (FCH::EMMC::MMIO::EMMCHC\_CTRL2)**\_aliasHOST; EMMCHC<sub>x0000002C</sub>; EMMCHC=FEDD\_5000h

Bits	Description
31:27	Reserved.

26	<b>SRST_DAT.</b> Read-write. Reset: 0. 0=No effect. 1=Reset DAT Line. <b>Description:</b> Software Reset for DAT Line. Following registers will be cleared: <ul style="list-style-type: none"> <li>- (Buffer Data Port Register) (Buffer is cleared).</li> <li>- (Present State Register).</li> <li>- Buffer Read Enable.</li> <li>- Buffer Write Enable.</li> <li>- Read Transfer Active.</li> <li>- Write Transfer Active.</li> <li>- DAT Line Active.</li> <li>- Command Inhibit (DAT).</li> <li>- (Block Gap Control Register).</li> <li>- Continue Request.</li> <li>- Stop At Block Gap Request.</li> <li>- (Normal Interrupt Status Register).</li> <li>- Buffer Read Ready.</li> <li>- Buffer Write Ready.</li> <li>- Block Gap Event.</li> <li>- Transfer Complete.</li> </ul>												
25	<b>SRST_CMD.</b> Read-write. Reset: 0. 0=No effect. 1=Reset CMD Line. <b>Description:</b> Software Reset for CMD Line. Following registers will be cleared: <ul style="list-style-type: none"> <li>- (Present State Register).</li> <li>- Command Inhibit (CMD).</li> <li>- (Normal Interrupt Status Register).</li> <li>- Command Complete.</li> </ul>												
24	<b>SRST_ALL.</b> Read-write. Reset: 0. 0=No effect. 1=Reset all. <b>Description:</b> Software Reset for all. Following register will not be cleared: <ul style="list-style-type: none"> <li>- CMD Line Signal Level.</li> <li>- DAT[3:0] Line Signal Level.</li> <li>- Write Protect Switch Pin Level.</li> <li>- Card Detect Pin Level.</li> <li>- Card State Stable.</li> <li>- Card Inserted.</li> <li>- All bits in the Capabilities Register.</li> <li>- All bits in the Maximum Current Capabilities register.</li> </ul>												
23:20	Reserved.												
19:16	<b>DAT_TO_CNT.</b> Read-write. Reset: 0h. Data Timeout Counter Value. By using this counter value, DAT line timeouts are detected. <b>ValidValues:</b> <table border="1"> <thead> <tr> <th>Value</th><th>Description</th></tr> </thead> <tbody> <tr> <td>0h</td><td>2<sup>13</sup></td></tr> <tr> <td>1h</td><td>2<sup>14</sup></td></tr> <tr> <td>Dh-2h</td><td>Omitted.</td></tr> <tr> <td>Eh</td><td>2<sup>27</sup></td></tr> <tr> <td>Fh</td><td>Reserved.</td></tr> </tbody> </table>	Value	Description	0h	2 <sup>13</sup>	1h	2 <sup>14</sup>	Dh-2h	Omitted.	Eh	2 <sup>27</sup>	Fh	Reserved.
Value	Description												
0h	2 <sup>13</sup>												
1h	2 <sup>14</sup>												
Dh-2h	Omitted.												
Eh	2 <sup>27</sup>												
Fh	Reserved.												
15:8	<b>CLK_FREQ_SEL.</b> Read-write. Reset: 00h. SDCLK Frequency Select. If multiple bits are set, most significant bit will be selected. <b>ValidValues:</b> <table border="1"> <thead> <tr> <th>Bit</th><th>Description</th></tr> </thead> <tbody> <tr> <td>[0]</td><td>0=Divide by 1. 1=Divide by 2.</td></tr> <tr> <td>[1]</td><td>1=Divide by 4.</td></tr> <tr> <td>[2]</td><td>1=Divide by 8.</td></tr> </tbody> </table>	Bit	Description	[0]	0=Divide by 1. 1=Divide by 2.	[1]	1=Divide by 4.	[2]	1=Divide by 8.				
Bit	Description												
[0]	0=Divide by 1. 1=Divide by 2.												
[1]	1=Divide by 4.												
[2]	1=Divide by 8.												



	[3]	1=Divide by 16.
	[4]	1=Divide by 32.
	[5]	1=Divide by 64.
	[6]	1=Divide by 128.
	[7]	1=Divide by 256.
7:6	<b>CLK_FREQ_SEL_UP</b> . Read-write. Reset: 0h. Upper Bits of SDCLK Frequency Select.	
5	<b>CLK_GEN_SEL</b> . Read-write. Reset: 0. 0=Divided Clock mode. 1=Programmable Clock mode. Clock Generator select.	
4:3	Reserved.	
2	<b>CLK_EN</b> . Read-write. Reset: 0. 0=Disable. 1=Enable. SD Clock Enable. SDCLK Frequency Select can be changed when this bit is 0. When card is removed, this bit is cleared to 0 automatically.	
1	<b>INTNCLK_STABLE</b> . Read-only. Reset: 0. 0=Unstable. 1=Stable. Internal clock stable.	
0	<b>INTNCLK_EN</b> . Read-write. Reset: 0. 0=Disable. 1=Enable. Internal clock enable.	

**EMMCHCx00000030 (FCH::EMMC::MMIO::EMMCHC\_INT\_STS)**

\_aliasHOST; EMMCHCx00000030; EMMCHC=FEDD\_5000h

Bits	Description
31:30	Reserved.
29	<b>CE_ATA_ERR</b> . Read-only. Reset: 0. CE-ATA Error.
28	<b>SDMA_ERR</b> . Read,Write-1-to-clear. Reset: 0. SDMA Error.
27	Reserved.
26	<b>TUNING_ERR</b> . Read,Write-1-to-clear. Reset: 0. Tuning Error.
25	<b>ADMA_ERR</b> . Read,Write-1-to-clear. Reset: 0. ADMA Error.
24	<b>ACMD_ERR</b> . Read,Write-1-to-clear. Reset: 0. Auto CMD12 Error. Logical OR of Auto CMD12 Error Status register.
23	<b>CUR_LIM_ERR</b> . Read,Write-1-to-clear. Reset: 0. Reserved.
22	<b>DAT_END_ERR</b> . Read,Write-1-to-clear. Reset: 0. Data CRC Error.
21	<b>DAT_CRC_ERR</b> . Read,Write-1-to-clear. Reset: 0. Data CRC Error.
20	<b>DAT_TO_ERR</b> . Read,Write-1-to-clear. Reset: 0. Data Timeout Error.
19	<b>CMD_IDX_ERR</b> . Read,Write-1-to-clear. Reset: 0. Command Index Error. Mismatch of Command Index and index of response.
18	<b>CMD_END_ERR</b> . Read,Write-1-to-clear. Reset: 0. Command End Bit Error.
17	<b>CMD_CRC_ERR</b> . Read,Write-1-to-clear. Reset: 0. Command CRC Error. If both CMD_TO_ERR and CMD_CRC_ERR set, this indicates Command Conflict Error.
16	<b>CMD_TO_ERR</b> . Read,Write-1-to-clear. Reset: 0. Command Timeout Error. Response not returned within 128 SDCLK cycles.
15	<b>ERR_INT</b> . Read-only. Reset: 0. Error Interrupt.
14:13	Reserved.
12	<b>RE_RUNNING</b> . Read-only. Reset: 0. Re-Tuning Event. This status is set if Re-Tuning Request in the Present State register changes from 0 to 1.
11	<b>INT_C</b> . Read-only. Reset: 0. INT_C. This Status is set if INT_C is enabled and INT_C# pin is in low level.
10	<b>INT_B</b> . Read-only. Reset: 0. INT_B. This Status is set if INT_B is enabled and INT_B# pin is in low level.
9	<b>INT_A</b> . Read-only. Reset: 0. INT_A. This Status is set if INT_A is enabled and INT_A# pin is in low level.
8	<b>CARD_INT</b> . Read-only. Reset: 0. SDIO Card Interrupt. Writing 1 to this register does not clear this bit. For clearing this bit, interrupt factor of SDIO cards should be cleared. The value of this bit is latched internally as long as the Card Interrupt bit (D08) in the Normal Interrupt Status Enable Register is 1.
7	<b>CARD_RM</b> . Read,Write-1-to-clear. Reset: 0. Card Removal.
6	<b>CARD_INS</b> . Read,Write-1-to-clear. Reset: 0. Card Insertion.
5	<b>BUF_RD_RDY</b> . Read,Write-1-to-clear. Reset: 0. Buffer Read Ready. In case that Auto CMD12 is enabled and last block has been transferred, Auto CMD12 will be issued prior to this bit being set to 1. Clearing this bit should

	be done before buffer reading, because SD controller has dual buffer and the next Buffer Read Ready interrupt may occur immediately.
4	<b>BUF_WR_RDY</b> . Read,Write-1-to-clear. Reset: 0. Buffer Write Ready. Clearing this bit should be done before buffer writing, because SD controller has dual buffer and the next Buffer Write Ready interrupt may occur immediately.
3	<b>DMA_INT</b> . Read,Write-1-to-clear. Reset: 0. DMA Interrupt. It is set when the internal counter reaches the value designated by Host DMA Buffer Boundary. It should be cleared by Host Driver after System Address Register is updated.
2	<b>BLOCK_GAP_EVT</b> . Read,Write-1-to-clear. Reset: 0. Block Gap Event. It indicates the timing of next block gap, which is requested by the Stop At Block Gap Request. In case of a write transaction, this interrupt will be generated before busy completion.
1	<b>XFER_DONE</b> . Read,Write-1-to-clear. Reset: 0. Data Transfer Complete. Indicates the timing for completion of data transaction which includes the completion at the block gap by the Stop At Block Gap Request. When some errors are detected during data transaction, this bit will not be set. In case that Auto CMD12 is enabled, Auto CMD12 will be issued prior to this bit being set to 1.
0	<b>CMD_DONE</b> . Read,Write-1-to-clear. Reset: 0. Command Complete. The end bit of the command response is received. In the case of commands with no response, the end of the command.

**EMMCHCx00000034 (FCH::EMMC::MMIO::EMMCHC\_INT\_STS\_EN)**

\_aliasHOST; EMMCHCx00000034; EMMCHC=FEDD\_5000h

Bits	Description
31:30	Reserved.
29	<b>CE_ATA_ERR_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enable. CE-ATA Error Status enable.
28	<b>SDMA_ERR_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enable. SDMA Error Status enable.
27:26	Reserved.
25	<b>ADMA_ERR_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enable. ADMA Error Status enable.
24	<b>ACMD_ERR_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enable. Auto CMD12 Error Status enable.
23	<b>CUR_LIM_ERR_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enable. Current Limit Error Status enable.
22	<b>DAT_END_ERR_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enable. Data End Bit Error Status enable.
21	<b>DAT_CRC_ERR_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enable. Data CRC Error Status enable.
20	<b>DAT_TO_ERR_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enable. Data Timeout Error Status enable.
19	<b>CMD_IDX_ERR_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enable. Command Index Error Status enable.
18	<b>CMD_END_ERR_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enable. Command End Bit Error Status enable.
17	<b>CMD_CRC_ERR_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enable. Command CRC Error Status enable.
16	<b>CMD_TO_ERR_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enable. Command Timeout Error Status enable.
15:13	Reserved.
12	<b>RE_RUNNING_EN</b> . Read-only. Reset: 0. 0=Masked. 1=Enable. Re-Tuning Event Status enable.
11	<b>INT_C_EN</b> . Read-only. Reset: 0. 0=Masked. 1=Enable. INT_C Status enable.
10	<b>INT_B_EN</b> . Read-only. Reset: 0. 0=Masked. 1=Enable. INT_B Status enable.
9	<b>INT_A_EN</b> . Read-only. Reset: 0. 0=Masked. 1=Enable. INT_A Status enable.
8	<b>CARD_INT_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enable. Card Interrupt Status enable.
7	<b>CARD_RM_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enable. Card Removal Status enable.
6	<b>CARD_INS_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enable. Card Insertion Status enable.
5	<b>BUF_RD_RDY_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enable. Buffer Read Ready Status enable.
4	<b>BUF_WR_RDY_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enable. Buffer Write Ready Status enable.
3	<b>DMA_INT_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enable. DMA Interrupt Status enable.
2	<b>BLOCK_GAP_EVT_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enable. Block Gap Event Status enable.
1	<b>XFER_DONE_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enable. Transfer Complete Status enable.
0	<b>CMD_DONE_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enable. Command Complete Status enable.

**EMMCHC<sub>x00000038</sub> (FCH::EMMC::MMIO::EMMCHC\_INT\_EN)**

_aliasHOST; EMMCHC <sub>x00000038</sub> ; EMMCHC=FEDD_5000h	
Bits	Description
31:30	Reserved.
29	<b>CE_ATA_ERR_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enable. CE-ATA Error Signal enable.
28	<b>SDMA_ERR_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enable. SDMA Error Signal enable.
27:26	Reserved.
25	<b>ADMA_ERR_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enable. ADMA Error Signal enable.
24	<b>ACMD_ERR_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enable. Auto CMD12 Error Signal enable.
23	<b>CUR_LIM_ERR_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enable. Current Limit Error Signal enable.
22	<b>DAT_END_ERR_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enable. Data End Bit Error Signal enable.
21	<b>DAT_CRC_ERR_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enable. Data CRC Error Signal enable.
20	<b>DAT_TO_ERR_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enable. Data Timeout Error Signal enable.
19	<b>CMD_IDX_ERR_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enable. Command Index Error Signal enable.
18	<b>CMD_END_ERR_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enable. Command End Bit Error Signal enable.
17	<b>CMD_CRC_ERR_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enable. Command CRC Error Signal enable.
16	<b>CMD_TO_ERR_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enable. Command Timeout Error Signal enable.
15:13	Reserved.
12	<b>RE_RUNING_EN</b> . Read-only. Reset: 0. 0=Masked. 1=Enable. Re-Tuning Event Signal enable.
11	<b>INT_C_EN</b> . Read-only. Reset: 0. 0=Masked. 1=Enable. INT_C Signal enable.
10	<b>INT_B_EN</b> . Read-only. Reset: 0. 0=Masked. 1=Enable. INT_B Signal enable.
9	<b>INT_A_EN</b> . Read-only. Reset: 0. 0=Masked. 1=Enable. INT_A Signal enable.
8	<b>CARD_INT_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enable. Card Interrupt Signal enable.
7	<b>CARD_RM_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enable. Card Removal Signal enable.
6	<b>CARD_INS_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enable. Card Insertion Signal enable.
5	<b>BUF_RD_RDY_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enable. Buffer Read Ready Signal enable.
4	<b>BUF_WR_RDY_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enable. Buffer Write Ready Signal enable.
3	<b>DMA_INT_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enable. DMA Interrupt Signal enable.
2	<b>BLOCK_GAP_EVT_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enable. Block Gap Event Signal enable.
1	<b>XFER_DONE_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enable. Transfer Complete Signal enable.
0	<b>CMD_DONE_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enable. Command Complete Signal enable.

**EMMCHC<sub>x0000003C</sub> (FCH::EMMC::MMIO::EMMCHC\_ACMD\_ERR\_CTRL2)**

_aliasHOST; EMMCHC <sub>x0000003C</sub> ; EMMCHC=FEDD_5000h	
Bits	Description
31	<b>RESET_VAL_EN</b> . Read-write. Reset: 0. 0=Disable. 1=Enable. Preset value enable.
30	<b>ASYN_INT_EN</b> . Read-write. Reset: 0. 0=Disable. 1=Enable. Asynchronous Interrupt enable.
29:24	Reserved.
23	<b>SAMPLING_CLK</b> . Read-write. Reset: 0. Sampling Clock select.
22	<b>EXE_TUNING</b> . Read-write. Reset: 0. 0=No effect. 1=Execute tuning. Execute tuning.
21:20	<b>DRV_STR</b> . Read-write. Reset: 0h. Driver strength select.
<b>ValidValues:</b>	
Value	Description
0h	Driver Type B is selected (default).
1h	Driver Type A is selected.
2h	Driver Type C is selected.
3h	Driver Type D is selected.
19	<b>VOL18_EN</b> . Read-write. Reset: 0. 1.8V Signaling Enable; this bit controls voltage 3.3/1.8v select. we should setting this bit first when use emmc

18:16	<b>UHS_MODE.</b> Read-write. Reset: 0h. UHS mode select.																
<b>ValidValues:</b>																	
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>SDR12</td></tr> <tr> <td>1h</td><td>SDR25</td></tr> <tr> <td>2h</td><td>SDR50</td></tr> <tr> <td>3h</td><td>SDR104/HS200</td></tr> <tr> <td>4h</td><td>DDR50</td></tr> <tr> <td>5h</td><td>HS400</td></tr> <tr> <td>7h-6h</td><td>Reserved.</td></tr> </table>	Value	Description	0h	SDR12	1h	SDR25	2h	SDR50	3h	SDR104/HS200	4h	DDR50	5h	HS400	7h-6h	Reserved.
Value	Description																
0h	SDR12																
1h	SDR25																
2h	SDR50																
3h	SDR104/HS200																
4h	DDR50																
5h	HS400																
7h-6h	Reserved.																
15:8	Reserved.																
7	<b>NOT_ISSUE_ERR.</b> Read-only. Reset: 0. Command not issued by Auto CMD12 error.																
6:5	Reserved.																
4	<b>IND_ERR.</b> Read-only. Reset: 0. Auto CMD12 Index error.																
3	<b>END_ERR.</b> Read-only. Reset: 0. Auto CMD12 End Bit error.																
2	<b>CRC_ERR.</b> Read-only. Reset: 0. Auto CMD12 CRC error.																
1	<b>TO_ERR.</b> Read-only. Reset: 0. Auto CMD12 Timeout error.																
0	<b>NOT_EXE_ERR.</b> Read-only. Reset: 0. Auto CMD12 not executed error.																

**EMMCHCx00000040 (FCH::EMMC::MMIO::EMMCHC\_CAP1)**

Read-only.

\_aliasHOST; EMMCHCx00000040; EMMCHC=FEDD\_5000h

Bits	Description										
31:30	<b>SLOT_TYPE.</b> Read-only. Reset: 0h. Slot type.										
29	<b>ASYN_INT_SUP.</b> Read-only. Reset: 0. Asynchronous Interrupt support.										
28	<b>SYS_BUS64_SUP.</b> Read-only. Reset: 0. 64-bit System Bus support.										
27	Reserved.										
26	<b>V18_SUP.</b> Read-only. Reset: 0. Voltage support 1.8V.										
25	<b>V30_SUP.</b> Read-only. Reset: 0. Voltage support 3.0V.										
24	<b>V33_SUP.</b> Read-only. Reset: 1. Voltage support 3.3V.										
23	<b>SUS_RES_SUP.</b> Read-only. Reset: 1. Suspend/Resume support.										
22	<b>SDMA_SUP.</b> Read-only. Reset: 0. SDMA support.										
21	<b>HS_SUP.</b> Read-only. Reset: 1. High-speed support.										
20	Reserved.										
19	<b>ADMA2_SUP.</b> Read-only. Reset: 0. ADMA2 support.										
18	<b>WIDTH_8_SUP.</b> Read-only. Reset: 1. 8-bit support for Embedded Device.										
17:16	<b>MAX_BLK_LEN.</b> Read-only. Reset: 2h. Max block length.										
<b>ValidValues:</b>											
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>512 bytes</td></tr> <tr> <td>1h</td><td>1K bytes</td></tr> <tr> <td>2h</td><td>2K bytes</td></tr> <tr> <td>3h</td><td>Reserved.</td></tr> </table>	Value	Description	0h	512 bytes	1h	1K bytes	2h	2K bytes	3h	Reserved.
Value	Description										
0h	512 bytes										
1h	1K bytes										
2h	2K bytes										
3h	Reserved.										
15:8	<b>BASE_CLK_FREQ.</b> Read-only. Reset: 18h. Base clock frequency for SD clock.										
7	<b>TO_CLK_UNIT.</b> Read-only. Reset: 1. Timeout clock unit.										
6	Reserved.										
5:0	<b>TO_CLK_FREQ.</b> Read-only. Reset: 18h. Timeout clock frequency.										

**EMMCHCx00000044 (FCH::EMMC::MMIO::EMMCHC\_CAP2)**

Read-only.	
_aliasHOST; EMMCHCx00000044; EMMCHC=FEDD_5000h	
Bits	Description
31:24	Reserved.
23:16	<b>CLK_MULTI</b> . Read-only. Reset: 00h. Clock multiplier.
15:14	<b>RETUNING_MODE</b> . Read-only. Reset: 0h. Re-Tuning modes.
13	<b>SDR50_TUNING</b> . Read-only. Reset: 0. Use Tuning for SDR50.
12	Reserved.
11:8	<b>TO_RETUNING</b> . Read-only. Reset: 0h. Timer Count for Re-Tuning.
7	Reserved.
6	<b>DRV_D_SUP</b> . Read-only. Reset: 0. Driver Type D support.
5	<b>DRV_C_SUP</b> . Read-only. Reset: 0. Driver Type C support.
4	<b>DRV_A_SUP</b> . Read-only. Reset: 0. Driver Type A support.
3	Reserved.
2	<b>DDR50_SUP</b> . Read-only. Reset: 1. DDR50 support.
1	<b>SDR104_SUP</b> . Read-only. Reset: 1. SDR104 support.
0	<b>SDR50_SUP</b> . Read-only. Reset: 1. SDR50 support.

**EMMCHCx00000048 (FCH::EMMC::MMIO::EMMCHC\_CURR\_CAP)**

Read-only.	
_aliasHOST; EMMCHCx00000048; EMMCHC=FEDD_5000h	
Bits	Description
31:24	Reserved.
23:16	<b>MAX_CURR_V18</b> . Read-only. Reset: 00h. Maximum current for 1.8V.
15:8	<b>MAX_CURR_V30</b> . Read-only. Reset: 00h. Maximum current for 3.0V.
7:0	<b>MAX_CURR_V33</b> . Read-only. Reset: 64h. Maximum current for 3.3V.

**EMMCHCx0000004C (FCH::EMMC::MMIO::EMMCHC\_CURR\_CAP2)**

Read-only. Reset: Fixed,0000_0000h.	
_aliasHOST; EMMCHCx0000004C; EMMCHC=FEDD_5000h	
Bits	Description
31:0	Reserved.

**EMMCHCx00000050 (FCH::EMMC::MMIO::EMMCHC\_FORCE\_EVT)**

_aliasHOST; EMMCHCx00000050; EMMCHC=FEDD_5000h	
Bits	Description
31:30	Reserved.
29	<b>CE_ATA_ERR</b> . Write-only. Reset: 0. 0=No effect. 1=Force event. Force CE-ATA Error.
28	<b>SDMA_ERR</b> . Write-only. Reset: 0. 0=No effect. 1=Force event. Force SDMA error.
27:26	Reserved.
25	<b>ADMA_ERR</b> . Write-only. Reset: 0. 0=No effect. 1=Force event. Force ADMA error.
24	<b>ACMD_ERR</b> . Write-only. Reset: 0. 0=No effect. 1=Force event. Force Auto CMD12 error.
23	<b>CUR_LIM_ERR</b> . Write-only. Reset: 0. 0=No effect. 1=Force event. Force Current Limit error.
22	<b>DAT_END_ERR</b> . Write-only. Reset: 0. 0=No effect. 1=Force event. Force Data End Bit error.
21	<b>DAT_CRC_ERR</b> . Write-only. Reset: 0. 0=No effect. 1=Force event. Force Data CRC error.
20	<b>DAT_TO_ERR</b> . Write-only. Reset: 0. 0=No effect. 1=Force event. Force Data Timeout error.
19	<b>CMD_IDX_ERR</b> . Write-only. Reset: 0. 0=No effect. 1=Force event. Force Command Index error.
18	<b>CMD_END_ERR</b> . Write-only. Reset: 0. 0=No effect. 1=Force event. Force Command End Bit error.
17	<b>CMD_CRC_ERR</b> . Write-only. Reset: 0. 0=No effect. 1=Force event. Force Command CRC error.
16	<b>CMD_TO_ERR</b> . Write-only. Reset: 0. 0=No effect. 1=Force event. Force Command Timeout error.

15:8	Reserved.
7	<b>ACMD12_ISS_ERR.</b> Write-only. Reset: 0. 0=No effect. 1=Force event. Force Command Not Issued By Auto CMD12 error.
6:5	Reserved.
4	<b>ACMD_IDX_ERR.</b> Write-only. Reset: 0. 0=No effect. 1=Force event. Force Auto CMD12 Index error.
3	<b>ACMD_END_ERR.</b> Write-only. Reset: 0. 0=No effect. 1=Force event. Force Auto CMD12 End Bit error.
2	<b>ACMD_CRC_ERR.</b> Write-only. Reset: 0. 0=No effect. 1=Force event. Force Auto CMD12 CRC error.
1	<b>ACMD_TO_ERR.</b> Write-only. Reset: 0. 0=No effect. 1=Force event. Force Auto CMD12 Timeout error.
0	<b>ACMD12_EXE_ERR.</b> Write-only. Reset: 0. 0=No effect. 1=Force event. Force Auto CMD12 Not Executed error.

#### EMMCHCx00000054 (FCH::EMMC::MMIO::EMMCHC\_ADMA\_ERR\_STS)

Read-only.

\_aliasHOST; EMMCHCx00000054; EMMCHC=FEDD\_5000h

Bits	Description										
31:3	Reserved.										
2	<b>ADDR_LEN_MISMATCH.</b> Read-only. Reset: 0. <b>Description:</b> ADMA Address Length Mismatch error. This error occurs in the following 2 cases: 1. While Block Count Enable being set, the total data length specified by the descriptor table is different from that specified by the Block Count and Block Length. 2. Total data length can not be divided by the block length.										
1:0	<b>ERR_STATE.</b> Read-only. Reset: 0h. ADMA state when error occurred. <b>ValidValues:</b> <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>Stop DMA.</td></tr> <tr> <td>1h</td><td>Fetch descriptor.</td></tr> <tr> <td>2h</td><td>Change address.</td></tr> <tr> <td>3h</td><td>Transfer data.</td></tr> </table>	Value	Description	0h	Stop DMA.	1h	Fetch descriptor.	2h	Change address.	3h	Transfer data.
Value	Description										
0h	Stop DMA.										
1h	Fetch descriptor.										
2h	Change address.										
3h	Transfer data.										

#### EMMCHCx00000058 (FCH::EMMC::MMIO::EMMCHC\_ADMA\_SYSADDR)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; EMMCHCx00000058; EMMCHC=FEDD\_5000h

Bits	Description
31:0	<b>ADMA_SYS_ADDR0.</b> Read-write. Reset: 0000_0000h. Lower bits of ADMA system address. See also FCH::EMMC::MMIO::EMMCHC_ADMA_SYSADDR_UP.

#### EMMCHCx0000005C (FCH::EMMC::MMIO::EMMCHC\_ADMA\_SYSADDR\_UP)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; EMMCHCx0000005C; EMMCHC=FEDD\_5000h

Bits	Description
31:0	<b>ADMA_SYS_ADDR1.</b> Read-write. Reset: 0000_0000h. Upper bits of ADMA System Address. Before ADMA data transfer, the descriptor address should be set by the Host driver. This address needs to be set with 4-byte alignment, since the descriptor table is 32-bit (4 byte) information formatted.

#### EMMCHCx00000080 (FCH::EMMC::MMIO::EMMCHC\_CE\_ATA\_CTRL)

\_aliasHOST; EMMCHCx00000080; EMMCHC=FEDD\_5000h

Bits	Description
31:5	Reserved.
4	<b>CE_ATA_SUPPORT.</b> Read-only. Reset: 1. 0=Unsupported. 1=Supported. CE-ATA support capability. When supported, bits[3:0] can be written.
3	<b>CE_ATA_EN.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. CE-ATA enable. When enabled bits[2:0] can be written.



2	<b>P_DRIVE_DIS.</b> Read-write. Reset: 0. 0=Enable. 1=Disable. P-Drive disable.
1	<b>CMD_COMP_DIS_EN.</b> Read-write. Reset: 0. 0=Wait after command. 1=Issued after command. Command Completion Signal Disable enable.
0	<b>CMD_COMP_SIG_EN.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. Command Completion Signal enable. When the Command Completion Signal is enabled, the Auto CMD12 Enable bit in the Transfer Mode register becomes of no use.

#### EMMCHCx00000090 (FCH::EMMC::MMIO::EMMCHC\_DAT3\_CARD\_DET)

\_aliasHOST; EMMCHCx00000090; EMMCHC=FEDD\_5000h

Bits	Description
31:5	Reserved.
4	<b>DEBOUNCING_EN.</b> Read-write. Reset: 1. 0=Disable. 1=Enable. Debouncing enable. This bit should be cleared before starting command or data transfer if debouncing state is not needed in the error cases.
3	<b>DEBOUNCING_STATUS.</b> Read-only. Reset: 0. 0=Not debouncing. 1=Debouncing. Debouncing State status. Indicates the card is debouncing. Should not issue any command during debouncing.
2	<b>CARD_DETECT_SEL.</b> Read-only. Reset: 0. 0=Standard Card Detect mode. 1=DAT3 Card Detect mode. Card Detect Select status.
1	<b>DAT3_DETECT_EN.</b> Read-write. Reset: 1. 0=Disable. 1=Enable. DAT3 Card Detect control. Disable this bit before writing the Command Register. After Command or Transfer complete, this bit can be re-enabled. During command and data transfer, enabling DAT3 Card Detection is prohibited. This bit is fixed to 1 in Standard Card Detect mode.
0	<b>DAT3_CARD_INS.</b> Read-only. Reset: 0. 0=No card inserted. 1=Card inserted. DAT3 Card inserted.

#### EMMCHCx000000FC (FCH::EMMC::MMIO::EMMCHC\_SLOT\_VER)

Read-only.

\_aliasHOST; EMMCHCx000000FC; EMMCHC=FEDD\_5000h

Bits	Description
31:24	<b>VENDOR_VERSION.</b> Read-only. Reset: C3h. Vendor version.
23:16	<b>SPEC_VERSION.</b> Read-only. Reset: 02h. Specification version.
15:8	Reserved.
7:0	<b>SLOT_INTRPT.</b> Read-only. Reset: 00h. Interrupt Signal for each slot. The value of XSLT_INT[7:0] inputs, which indicates the logical OR of the Interrupt signal and Wakeup signal, are inverted and referred by this register. In case of multiple slots, Interrupt signal and Wakeup signal should be logically ORed externally and should be inputted to each of XSLT_INT[7:0].

## 9.2.3 I/O Advanced Programmable Interrupt Control

### 9.2.3.1 IOAPIC Registers

IOAPIC configuration registers can be accessed at memory region from FEC0\_0000h to FEC0\_007Fh. Program FCH::PM::PmDecodeEn[IoApicEn]=1 to enable IOAPIC decoding.

#### IOAPICx0000 [IO Register Select Register] (FCH::IOAPIC::IOSel)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOAPICx0000; IOAPIC=FEC0\_0000h

Bits	Description
31:8	Reserved.
7:0	<b>IndirectAddressOffset.</b> Read-write. Reset: 00h. Indirect Address Offset to IO Window Register. Determines which register is manipulated during an IO window register Read/Write operation.

#### IOAPICx0010 [IO Window Register] (FCH::IOAPIC::IOWindow)



Read-write. Reset: 0000_0000h.	
_aliasHOST; IOAPICx0010; IOAPIC=FEC0_0000h	
Bits	Description
31:0	<b>IOWindow.</b> Read-write. Reset: 0000_0000h. Mapped by FCH::IOAPIC::IOSEL to the designated indirect access register.

#### IOAPICx0010\_indirectaddressoffset00 [IOAPICID] (FCH::IOAPIC::IOAPICID)

Read-write. Reset: 0000_0000h.	
This register is not used in IOxAPIC PCI bus delivery mode.	
_n0_aliasHOST; IOAPICx0010_indirectaddressoffset00; IOAPIC=FEC0_0000h; DataPortWrite=FCH::IOAPIC::IOSEL[IndirectAddressOffset]	
Bits	Description
31:28	Reserved.
27:24	<b>ID.</b> Read-write. Reset: 0h. IOAPIC device ID.
23:0	Reserved.

#### IOAPICx0010\_indirectaddressoffset01 [IOAPICVersion] (FCH::IOAPIC::IOAPICVersion)

Read-only. Reset: 0017_8021h.	
_n1_aliasHOST; IOAPICx0010_indirectaddressoffset01; IOAPIC=FEC0_0000h; DataPortWrite=FCH::IOAPIC::IOSEL[IndirectAddressOffset]	
Bits	Description
31:24	Reserved.
23:16	<b>MaxRedirectionEntries.</b> Read-only. Reset: 17h. Indicates 24 entries [23:0].
15	<b>PRQ.</b> Read-only. Reset: 1. IRQ pin assertion supported.
14:8	Reserved.
7:0	<b>Version.</b> Read-only. Reset: 21h. PCI 2.2 compliant.

#### IOAPICx0010\_indirectaddressoffset02 [IOAPICArb] (FCH::IOAPIC::IOAPICArb)

Read-only, Volatile. Reset: 0000_0000h.	
This register is not used in IOxAPIC PCI bus delivery mode.	
_n2_aliasHOST; IOAPICx0010_indirectaddressoffset02; IOAPIC=FEC0_0000h; DataPortWrite=FCH::IOAPIC::IOSEL[IndirectAddressOffset]	
Bits	Description
31:28	Reserved.
27:24	<b>ArbitrationID: Arbitration ID.</b> Read-only, Volatile. Reset: 0h. Arbitration ID.
23:0	Reserved.

#### IOAPICx0010\_indirectaddressoffset[10...3E] [Redirection Table Entry] (FCH::IOAPIC::RedirectionTabEntryLow32)

Reset: 0001_0000h.	
_n[23:0]_aliasHOST; IOAPICx0010_indirectaddressoffset[3E,3C,3A,38,36,34,32,30,2E,2C,2A,28,26,24,22,20,1E,1C,1A,18,16,14,12,10]; IOAPIC=FEC0_0000h; DataPortWrite=FCH::IOAPIC::IOSEL[IndirectAddressOffset]	
Bits	Description
31:17	Reserved.
16	<b>Mask.</b> Read-write. Reset: 1. 0=Unmask. 1=Mask the interrupt injection at the input of this device.
15	<b>TriggerMode.</b> Read-write. Reset: 0. 0=Edge. 1=Level.
14	<b>RemoteIRR.</b> Read-only. Reset: 0. 1=Interrupt message is delivered. Used for level triggered interrupts only. It is cleared by EOI special cycle transaction or a Write to FCH::IOAPIC::EOI.
13	<b>InterruptPinPolarity.</b> Read-write. Reset: 0. 0=High. 1=Low.
12	<b>DeliveryStatus.</b> Read-only. Reset: 0. 0=Idle. 1=Send Pending.
11	<b>DestinationMode.</b> Read-write. Reset: 0. 0=Physical. 1=Logical.
10:8	<b>DeliveryMode.</b> Read-write. Reset: 0h.
<b>ValidValues:</b>	
Value	Description

	0h	Fixed.
	1h	Lowest Priority.
	2h	SMI/PMI.
	3h	Reserved.
	4h	NMI.
	5h	INIT.
	6h	Reserved.
	7h	ExtINT.
7:0	<b>Vector.</b> Read-write. Reset: 00h. Interrupt vector associated with this interrupt input.	

#### IOAPICx0010\_indirectaddressoffset[11...3F] [Redirection Table Entry] (FCH::IOAPIC::RedirectionTabEntryHigh32)

Read-write. Reset: 0000\_0000h.

\_n[23:0]\_aliasHOST; IOAPICx0010\_indirectaddressoffset[3F,3D,3B,39,37,35,33,31,2F,2D,2B,29,27,25,23,21,1F,1D,1B,19,17,15,13,11]; IOAPIC=FEC0\_0000h;  
DataPortWrite=FCH::IOAPIC::IOSEL[IndirectAddressOffset]

Bits	Description
31:24	<b>DestinationID.</b> Read-write. Reset: 00h. Bits[19:12] of the address field of the interrupt message.
23:0	Reserved.

#### IOAPICx0020 [IRQ Pin Assertion Register] (FCH::IOAPIC::IRQPinAssertion)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; IOAPICx0020; IOAPIC=FEC0\_0000h

Bits	Description
31:8	Reserved.
7:0	<b>InputIrq.</b> Read-write. Reset: 00h. IRQ number for the requested interrupt. A write to this register triggers an interrupt associated with the redirection table entry referenced by the IRQ number. Currently the redirection table has 24 entries. Writes with IRQ number greater than 17h have no effect.

#### IOAPICx0040 [EOI Register] (FCH::IOAPIC::EOI)

Write-only. Reset: 0000\_0000h.

\_aliasHOST; IOAPICx0040; IOAPIC=FEC0\_0000h

Bits	Description
31:8	Reserved.
7:0	<b>Vector.</b> Write-only. Reset: 00h. Interrupt vector. A write to this register clears the remote IRR bit in the redirection table entry found matching the interrupt vector (see FCH::IOAPIC::RedirectionTabEntryLow32[RemoteIRR]). This provides an alternate mechanism other than the PCI special cycle for EOI to reach IOxAPIC.

## 9.2.4 SMI Registers

SMI register space is accessed through the AcpiMmio region. The SMI registers range from FED8\_0000h+200h to FED8\_0000h+2FFh. See FCH::PM::IsaControl[MmioEn].

#### SMIx000 [Event\_Status] (FCH::SMI::EventStat)

Read, Write-1-to-clear.

\_aliasHOST; SMIx000; SMI=FED8\_0200h

Bits	Description
31:0	<b>EventStatus.</b> Read, Write-1-to-clear. Reset: XXXX_XXXXh. This is a mirror register of the standard ACPI register FCH::PM::EventStat. Each event status bit is set when the selected event input equals to the corresponding value in FCH::SMI::SciTrig.

**SMIx004 [Event\_Enable] (FCH::SMI::EventEn)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; SMIx004; SMI=FED8\_0200h

Bits	Description
31:0	<b>EventEnable.</b> Read-write. Reset: 0000_0000h. This is the mirror register of the standard ACPI register FCH::PM::EventEnable. Each bit controls whether ACPI should generate wake up and SCI interrupt.

**SMIx008 [SciTrig] (FCH::SMI::SciTrig)**

Read-write. Reset: FFFF\_FFFFh.

Each bit in this register controls the way to set each corresponding bit in FCH::SMI::EventStat.

\_aliasHOST; SMIx008; SMI=FED8\_0200h

Bits	Description
31	<b>SciTrig31.</b> Read-write. Reset: 1. 0=Active low. 1=Active high. The bit controls the way to set FCH::SMI::EventStat bit[31].
30	<b>SciTrig30.</b> Read-write. Reset: 1. 0=Active low. 1=Active high. The bit controls the way to set FCH::SMI::EventStat bit[30].
29	<b>SciTrig29.</b> Read-write. Reset: 1. 0=Active low. 1=Active high. The bit controls the way to set FCH::SMI::EventStat bit[29].
28	<b>SciTrig28.</b> Read-write. Reset: 1. 0=Active low. 1=Active high. The bit controls the way to set FCH::SMI::EventStat bit[28].
27	<b>SciTrig27.</b> Read-write. Reset: 1. 0=Active low. 1=Active high. The bit controls the way to set FCH::SMI::EventStat bit[27].
26	<b>SciTrig26.</b> Read-write. Reset: 1. 0=Active low. 1=Active high. The bit controls the way to set FCH::SMI::EventStat bit[26].
25	<b>SciTrig25.</b> Read-write. Reset: 1. 0=Active low. 1=Active high. The bit controls the way to set FCH::SMI::EventStat bit[25].
24	<b>SciTrig24.</b> Read-write. Reset: 1. 0=Active low. 1=Active high. The bit controls the way to set FCH::SMI::EventStat bit[24].
23	<b>SciTrig23.</b> Read-write. Reset: 1. 0=Active low. 1=Active high. The bit controls the way to set FCH::SMI::EventStat bit[23].
22	<b>SciTrig22.</b> Read-write. Reset: 1. 0=Active low. 1=Active high. The bit controls the way to set FCH::SMI::EventStat bit[22].
21	<b>SciTrig21.</b> Read-write. Reset: 1. 0=Active low. 1=Active high. The bit controls the way to set FCH::SMI::EventStat bit[21].
20	<b>SciTrig20.</b> Read-write. Reset: 1. 0=Active low. 1=Active high. The bit controls the way to set FCH::SMI::EventStat bit[20].
19	<b>SciTrig19.</b> Read-write. Reset: 1. 0=Active low. 1=Active high. The bit controls the way to set FCH::SMI::EventStat bit[19].
18	<b>SciTrig18.</b> Read-write. Reset: 1. 0=Active low. 1=Active high. The bit controls the way to set FCH::SMI::EventStat bit[18].
17	<b>SciTrig17.</b> Read-write. Reset: 1. 0=Active low. 1=Active high. The bit controls the way to set FCH::SMI::EventStat bit[17].
16	<b>SciTrig16.</b> Read-write. Reset: 1. 0=Active low. 1=Active high. The bit controls the way to set FCH::SMI::EventStat bit[16].
15	<b>SciTrig15.</b> Read-write. Reset: 1. 0=Active low. 1=Active high. The bit controls the way to set FCH::SMI::EventStat bit[15].
14	<b>SciTrig14.</b> Read-write. Reset: 1. 0=Active low. 1=Active high. The bit controls the way to set FCH::SMI::EventStat bit[14].
13	<b>SciTrig13.</b> Read-write. Reset: 1. 0=Active low. 1=Active high. The bit controls the way to set FCH::SMI::EventStat bit[13].
12	<b>SciTrig12.</b> Read-write. Reset: 1. 0=Active low. 1=Active high. The bit controls the way to set

	FCH::SMI::EventStat bit[12].
11	<b>SciTrig11.</b> Read-write. Reset: 1. 0=Falling edge. 1=Active high. The bit controls the way to set FCH::SMI::EventStat bit[11].
10	<b>SciTrig10.</b> Read-write. Reset: 1. 0=Active low. 1=Active high. The bit controls the way to set FCH::SMI::EventStat bit[10].
9	<b>SciTrig9.</b> Read-write. Reset: 1. 0=Active low. 1=Active high. The bit controls the way to set FCH::SMI::EventStat bit[9].
8	<b>SciTrig8.</b> Read-write. Reset: 1. 0=Active low. 1=Active high. The bit controls the way to set FCH::SMI::EventStat bit[8].
7	<b>SciTrig7.</b> Read-write. Reset: 1. 0=Active low. 1=Active high. The bit controls the way to set FCH::SMI::EventStat bit[7].
6	<b>SciTrig6.</b> Read-write. Reset: 1. 0=Active low. 1=Active high. The bit controls the way to set FCH::SMI::EventStat bit[6].
5	<b>SciTrig5.</b> Read-write. Reset: 1. 0=Active low. 1=Active high. The bit controls the way to set FCH::SMI::EventStat bit[5].
4	<b>SciTrig4.</b> Read-write. Reset: 1. 0=Active low. 1=Active high. The bit controls the way to set FCH::SMI::EventStat bit[4].
3	<b>SciTrig3.</b> Read-write. Reset: 1. 0=Active low. 1=Active high. The bit controls the way to set FCH::SMI::EventStat bit[3].
2	<b>SciTrig2.</b> Read-write. Reset: 1. 0=Active low. 1=Active high. The bit controls the way to set FCH::SMI::EventStat bit[2].
1	<b>SciTrig1.</b> Read-write. Reset: 1. 0=Active low. 1=Active high. The bit controls the way to set FCH::SMI::EventStat bit[1].
0	<b>SciTrig0.</b> Read-write. Reset: 1. 0=Active low. 1=Active high. The bit controls the way to set FCH::SMI::EventStat bit[0].

**SMIx00C [SciLevl] (FCH::SMI::SciLevl)**

Read-write. Reset: 0000\_0000h.

Reset: 0000\_0000h. This register specifies the trigger mode for each of the corresponding bit in FCH::SMI::EventStat.

\_aliasHOST; SMIx00C; SMI=FED8\_0200h

Bits	Description
31	<b>SciLevl31.</b> Read-write. Reset: 0. 0=Edge trigger. 1=Level trigger.
30	<b>SciLevl30.</b> Read-write. Reset: 0. 0=Edge trigger. 1=Level trigger.
29	<b>SciLevl29.</b> Read-write. Reset: 0. 0=Edge trigger. 1=Level trigger.
28	<b>SciLevl28.</b> Read-write. Reset: 0. 0=Edge trigger. 1=Level trigger.
27	<b>SciLevl27.</b> Read-write. Reset: 0. 0=Edge trigger. 1=Level trigger.
26	<b>SciLevl26.</b> Read-write. Reset: 0. 0=Edge trigger. 1=Level trigger.
25	<b>SciLevl25.</b> Read-write. Reset: 0. 0=Edge trigger. 1=Level trigger.
24	<b>SciLevl24.</b> Read-write. Reset: 0. 0=Edge trigger. 1=Level trigger.
23	<b>SciLevl23.</b> Read-write. Reset: 0. 0=Edge trigger. 1=Level trigger.
22	<b>SciLevl22.</b> Read-write. Reset: 0. 0=Edge trigger. 1=Level trigger.
21	<b>SciLevl21.</b> Read-write. Reset: 0. 0=Edge trigger. 1=Level trigger.
20	<b>SciLevl20.</b> Read-write. Reset: 0. 0=Edge trigger. 1=Level trigger.
19	<b>SciLevl19.</b> Read-write. Reset: 0. 0=Edge trigger. 1=Level trigger.
18	<b>SciLevl18.</b> Read-write. Reset: 0. 0=Edge trigger. 1=Level trigger.
17	<b>SciLevl17.</b> Read-write. Reset: 0. 0=Edge trigger. 1=Level trigger.
16	<b>SciLevl16.</b> Read-write. Reset: 0. 0=Edge trigger. 1=Level trigger.
15	<b>SciLevl15.</b> Read-write. Reset: 0. 0=Edge trigger. 1=Level trigger.
14	<b>SciLevl14.</b> Read-write. Reset: 0. 0=Edge trigger. 1=Level trigger.
13	<b>SciLevl13.</b> Read-write. Reset: 0. 0=Edge trigger. 1=Level trigger.

12	<b>SciLevl12.</b> Read-write. Reset: 0. 0=Edge trigger. 1=Level trigger.
11	<b>SciLevl11.</b> Read-write. Reset: 0. 0=Edge trigger. 1=Level trigger.
10	<b>SciLevl10.</b> Read-write. Reset: 0. 0=Edge trigger. 1=Level trigger.
9	<b>SciLevl9.</b> Read-write. Reset: 0. 0=Edge trigger. 1=Level trigger.
8	<b>SciLevl8.</b> Read-write. Reset: 0. 0=Edge trigger. 1=Level trigger.
7	<b>SciLevl7.</b> Read-write. Reset: 0. 0=Edge trigger. 1=Level trigger.
6	<b>SciLevl6.</b> Read-write. Reset: 0. 0=Edge trigger. 1=Level trigger.
5	<b>SciLevl5.</b> Read-write. Reset: 0. 0=Edge trigger. 1=Level trigger.
4	<b>SciLevl4.</b> Read-write. Reset: 0. 0=Edge trigger. 1=Level trigger.
3	<b>SciLevl3.</b> Read-write. Reset: 0. 0=Edge trigger. 1=Level trigger.
2	<b>SciLevl2.</b> Read-write. Reset: 0. 0=Edge trigger. 1=Level trigger.
1	<b>SciLevl1.</b> Read-write. Reset: 0. 0=Edge trigger. 1=Level trigger.
0	<b>SciLevl0.</b> Read-write. Reset: 0. 0=Edge trigger. 1=Level trigger.

**SMIx010 [SmiSciStatus] (FCH::SMI::SmiSciStat)**

Read,Write-1-to-clear. Reset: 0000\_0000h.

\_aliasHOST; SMIx010; SMI=FED8\_0200h

Bits	Description
31:0	<b>SmiSciStatus.</b> Read,Write-1-to-clear. Reset: 0000_0000h. Each bit indicates the corresponding SmiSci status. The input of each bit is controlled by the corresponding FCH::SMI::SciTrig bit. NOTE: This function can be considered as a superset of FCH::SMI::EventStat. When one of the bits is set and its corresponding FCH::SMI::SmiSciEn is also set, it triggers an SMI to call the BIOS. After the BIOS has serviced the SMM and cleared its status, the internal logic automatically sets the corresponding FCH::SMI::EventStat bit and thereby triggering an SCI.

**SMIx014 [SmiSciEn] (FCH::SMI::SmiSciEn)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; SMIx014; SMI=FED8\_0200h

Bits	Description																				
31:0	<p><b>SmiSciEn.</b> Read-write. Reset: 0000_0000h. Each bit controls if an SMI message is generated when the corresponding FCH::SMI::SmiSciStat bit is set to 1.</p> <p><b>ValidValues:</b></p> <table> <tr> <th>Bit</th><th>Description</th></tr> <tr> <td>[0]</td><td>0=Not to send SMI message when the corresponding FCH::SMI::SmiSciStat bit is set. 1=Send SMI message when the corresponding FCH::SMI::SmiSciStat bit is set.</td></tr> <tr> <td>[1]</td><td>0=Not to send SMI message when the corresponding FCH::SMI::SmiSciStat bit is set. 1=Send SMI message when the corresponding FCH::SMI::SmiSciStat bit is set.</td></tr> <tr> <td>[2]</td><td>0=Not to send SMI message when the corresponding FCH::SMI::SmiSciStat bit is set. 1=Send SMI message when the corresponding FCH::SMI::SmiSciStat bit is set.</td></tr> <tr> <td>[3]</td><td>0=Not to send SMI message when the corresponding FCH::SMI::SmiSciStat bit is set. 1=Send SMI message when the corresponding FCH::SMI::SmiSciStat bit is set.</td></tr> <tr> <td>[4]</td><td>0=Not to send SMI message when the corresponding FCH::SMI::SmiSciStat bit is set. 1=Send SMI message when the corresponding FCH::SMI::SmiSciStat bit is set.</td></tr> <tr> <td>[5]</td><td>0=Not to send SMI message when the corresponding FCH::SMI::SmiSciStat bit is set. 1=Send SMI message when the corresponding FCH::SMI::SmiSciStat bit is set.</td></tr> <tr> <td>[6]</td><td>0=Not to send SMI message when the corresponding FCH::SMI::SmiSciStat bit is set. 1=Send SMI message when the corresponding FCH::SMI::SmiSciStat bit is set.</td></tr> <tr> <td>[7]</td><td>0=Not to send SMI message when the corresponding FCH::SMI::SmiSciStat bit is set. 1=Send SMI message when the corresponding FCH::SMI::SmiSciStat bit is set.</td></tr> <tr> <td>[8]</td><td>0=Not to send SMI message when the corresponding FCH::SMI::SmiSciStat bit is set. 1=Send SMI message when the corresponding FCH::SMI::SmiSciStat bit is set.</td></tr> </table>	Bit	Description	[0]	0=Not to send SMI message when the corresponding FCH::SMI::SmiSciStat bit is set. 1=Send SMI message when the corresponding FCH::SMI::SmiSciStat bit is set.	[1]	0=Not to send SMI message when the corresponding FCH::SMI::SmiSciStat bit is set. 1=Send SMI message when the corresponding FCH::SMI::SmiSciStat bit is set.	[2]	0=Not to send SMI message when the corresponding FCH::SMI::SmiSciStat bit is set. 1=Send SMI message when the corresponding FCH::SMI::SmiSciStat bit is set.	[3]	0=Not to send SMI message when the corresponding FCH::SMI::SmiSciStat bit is set. 1=Send SMI message when the corresponding FCH::SMI::SmiSciStat bit is set.	[4]	0=Not to send SMI message when the corresponding FCH::SMI::SmiSciStat bit is set. 1=Send SMI message when the corresponding FCH::SMI::SmiSciStat bit is set.	[5]	0=Not to send SMI message when the corresponding FCH::SMI::SmiSciStat bit is set. 1=Send SMI message when the corresponding FCH::SMI::SmiSciStat bit is set.	[6]	0=Not to send SMI message when the corresponding FCH::SMI::SmiSciStat bit is set. 1=Send SMI message when the corresponding FCH::SMI::SmiSciStat bit is set.	[7]	0=Not to send SMI message when the corresponding FCH::SMI::SmiSciStat bit is set. 1=Send SMI message when the corresponding FCH::SMI::SmiSciStat bit is set.	[8]	0=Not to send SMI message when the corresponding FCH::SMI::SmiSciStat bit is set. 1=Send SMI message when the corresponding FCH::SMI::SmiSciStat bit is set.
Bit	Description																				
[0]	0=Not to send SMI message when the corresponding FCH::SMI::SmiSciStat bit is set. 1=Send SMI message when the corresponding FCH::SMI::SmiSciStat bit is set.																				
[1]	0=Not to send SMI message when the corresponding FCH::SMI::SmiSciStat bit is set. 1=Send SMI message when the corresponding FCH::SMI::SmiSciStat bit is set.																				
[2]	0=Not to send SMI message when the corresponding FCH::SMI::SmiSciStat bit is set. 1=Send SMI message when the corresponding FCH::SMI::SmiSciStat bit is set.																				
[3]	0=Not to send SMI message when the corresponding FCH::SMI::SmiSciStat bit is set. 1=Send SMI message when the corresponding FCH::SMI::SmiSciStat bit is set.																				
[4]	0=Not to send SMI message when the corresponding FCH::SMI::SmiSciStat bit is set. 1=Send SMI message when the corresponding FCH::SMI::SmiSciStat bit is set.																				
[5]	0=Not to send SMI message when the corresponding FCH::SMI::SmiSciStat bit is set. 1=Send SMI message when the corresponding FCH::SMI::SmiSciStat bit is set.																				
[6]	0=Not to send SMI message when the corresponding FCH::SMI::SmiSciStat bit is set. 1=Send SMI message when the corresponding FCH::SMI::SmiSciStat bit is set.																				
[7]	0=Not to send SMI message when the corresponding FCH::SMI::SmiSciStat bit is set. 1=Send SMI message when the corresponding FCH::SMI::SmiSciStat bit is set.																				
[8]	0=Not to send SMI message when the corresponding FCH::SMI::SmiSciStat bit is set. 1=Send SMI message when the corresponding FCH::SMI::SmiSciStat bit is set.																				

[illegible]

## SMIx018 [SwSciEn] (FCH::SMI::SwSciEn)

Read-write. Reset: 0000 0000h.

aliasHOST; SMIx018; SMI=FED8\_0200h

Bits	Description
------	-------------



31:0 **SwSciEn.** Read-write. Reset: 0000\_0000h. This register is used as a software mechanism to trigger the SCI.  
NOTE: The setting of this bit needs to match with FCH::SMI::SciTrig and FCH::SMI::SciLevl in order to set the status bit.

**Valid Values:**

Bit	Description
[0]	0=No effect. 1=Software can write to FCH::SMI::SwSciData and set the corresponding FCH::SMI::EventStat bit.
[1]	0=No effect. 1=Software can write to FCH::SMI::SwSciData and set the corresponding FCH::SMI::EventStat bit.
[2]	0=No effect. 1=Software can write to FCH::SMI::SwSciData and set the corresponding FCH::SMI::EventStat bit.
[3]	0=No effect. 1=Software can write to FCH::SMI::SwSciData and set the corresponding FCH::SMI::EventStat bit.
[4]	0=No effect. 1=Software can write to FCH::SMI::SwSciData and set the corresponding FCH::SMI::EventStat bit.
[5]	0=No effect. 1=Software can write to FCH::SMI::SwSciData and set the corresponding FCH::SMI::EventStat bit.
[6]	0=No effect. 1=Software can write to FCH::SMI::SwSciData and set the corresponding FCH::SMI::EventStat bit.
[7]	0=No effect. 1=Software can write to FCH::SMI::SwSciData and set the corresponding FCH::SMI::EventStat bit.
[8]	0=No effect. 1=Software can write to FCH::SMI::SwSciData and set the corresponding FCH::SMI::EventStat bit.
[9]	0=No effect. 1=Software can write to FCH::SMI::SwSciData and set the corresponding FCH::SMI::EventStat bit.
[10]	0=No effect. 1=Software can write to FCH::SMI::SwSciData and set the corresponding FCH::SMI::EventStat bit.
[11]	0=No effect. 1=Software can write to FCH::SMI::SwSciData and set the corresponding FCH::SMI::EventStat bit.
[12]	0=No effect. 1=Software can write to FCH::SMI::SwSciData and set the corresponding FCH::SMI::EventStat bit.
[13]	0=No effect. 1=Software can write to FCH::SMI::SwSciData and set the corresponding FCH::SMI::EventStat bit.
[14]	0=No effect. 1=Software can write to FCH::SMI::SwSciData and set the corresponding FCH::SMI::EventStat bit.
[15]	0=No effect. 1=Software can write to FCH::SMI::SwSciData and set the corresponding FCH::SMI::EventStat bit.
[16]	0=No effect. 1=Software can write to FCH::SMI::SwSciData and set the corresponding FCH::SMI::EventStat bit.
[17]	0=No effect. 1=Software can write to FCH::SMI::SwSciData and set the corresponding FCH::SMI::EventStat bit.
[18]	0=No effect. 1=Software can write to FCH::SMI::SwSciData and set the corresponding FCH::SMI::EventStat bit.
[19]	0=No effect. 1=Software can write to FCH::SMI::SwSciData and set the corresponding FCH::SMI::EventStat bit.
[20]	0=No effect. 1=Software can write to FCH::SMI::SwSciData and set the corresponding FCH::SMI::EventStat bit.
[21]	0=No effect. 1=Software can write to FCH::SMI::SwSciData and set the corresponding FCH::SMI::EventStat bit.
[22]	0=No effect. 1=Software can write to FCH::SMI::SwSciData and set the corresponding FCH::SMI::EventStat bit.



[23]	0=No effect. 1=Software can write to FCH::SMI::SwSciData and set the corresponding FCH::SMI::EventStat bit.
[24]	0=No effect. 1=Software can write to FCH::SMI::SwSciData and set the corresponding FCH::SMI::EventStat bit.
[25]	0=No effect. 1=Software can write to FCH::SMI::SwSciData and set the corresponding FCH::SMI::EventStat bit.
[26]	0=No effect. 1=Software can write to FCH::SMI::SwSciData and set the corresponding FCH::SMI::EventStat bit.
[27]	0=No effect. 1=Software can write to FCH::SMI::SwSciData and set the corresponding FCH::SMI::EventStat bit.
[28]	0=No effect. 1=Software can write to FCH::SMI::SwSciData and set the corresponding FCH::SMI::EventStat bit.
[29]	0=No effect. 1=Software can write to FCH::SMI::SwSciData and set the corresponding FCH::SMI::EventStat bit.
[30]	0=No effect. 1=Software can write to FCH::SMI::SwSciData and set the corresponding FCH::SMI::EventStat bit.
[31]	0=No effect. 1=Software can write to FCH::SMI::SwSciData and set the corresponding FCH::SMI::EventStat bit.

**SMIx01C [SwSciData] (FCH::SMI::SwSciData)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; SMIx01C; SMI=FED8\_0200h

Bits	Description
31:0	<b>SwSciData.</b> Read-write. Reset: 0000_0000h. This is the software data path to set the corresponding FCH::SMI::EventStat bit when FCH::SMI::SwSciEn is set.

**SMIx020 [SciSleepDisable] (FCH::SMI::SciSleepDisable)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; SMIx020; SMI=FED8\_0200h

Bits	Description																				
31:0	<p><b>SciSleepDisable.</b> Read-write. Reset: 0000_0000h. This register is used to ignore EVENT pins that are powered in the main power domain instead of auxiliary power domain. For each bit:</p> <p><b>ValidValues:</b></p> <table> <tr> <th>Bit</th><th>Description</th></tr> <tr> <td>[0]</td><td>0=No effect. 1=The corresponding FCH::SMI::EventStat bit is masked off whenever the system goes to S3 or higher sleep state.</td></tr> <tr> <td>[1]</td><td>0=No effect. 1=The corresponding FCH::SMI::EventStat bit is masked off whenever the system goes to S3 or higher sleep state.</td></tr> <tr> <td>[2]</td><td>0=No effect. 1=The corresponding FCH::SMI::EventStat bit is masked off whenever the system goes to S3 or higher sleep state.</td></tr> <tr> <td>[3]</td><td>0=No effect. 1=The corresponding FCH::SMI::EventStat bit is masked off whenever the system goes to S3 or higher sleep state.</td></tr> <tr> <td>[4]</td><td>0=No effect. 1=The corresponding FCH::SMI::EventStat bit is masked off whenever the system goes to S3 or higher sleep state.</td></tr> <tr> <td>[5]</td><td>0=No effect. 1=The corresponding FCH::SMI::EventStat bit is masked off whenever the system goes to S3 or higher sleep state.</td></tr> <tr> <td>[6]</td><td>0=No effect. 1=The corresponding FCH::SMI::EventStat bit is masked off whenever the system goes to S3 or higher sleep state.</td></tr> <tr> <td>[7]</td><td>0=No effect. 1=The corresponding FCH::SMI::EventStat bit is masked off whenever the system goes to S3 or higher sleep state.</td></tr> <tr> <td>[8]</td><td>0=No effect. 1=The corresponding FCH::SMI::EventStat bit is masked off whenever the system goes</td></tr> </table>	Bit	Description	[0]	0=No effect. 1=The corresponding FCH::SMI::EventStat bit is masked off whenever the system goes to S3 or higher sleep state.	[1]	0=No effect. 1=The corresponding FCH::SMI::EventStat bit is masked off whenever the system goes to S3 or higher sleep state.	[2]	0=No effect. 1=The corresponding FCH::SMI::EventStat bit is masked off whenever the system goes to S3 or higher sleep state.	[3]	0=No effect. 1=The corresponding FCH::SMI::EventStat bit is masked off whenever the system goes to S3 or higher sleep state.	[4]	0=No effect. 1=The corresponding FCH::SMI::EventStat bit is masked off whenever the system goes to S3 or higher sleep state.	[5]	0=No effect. 1=The corresponding FCH::SMI::EventStat bit is masked off whenever the system goes to S3 or higher sleep state.	[6]	0=No effect. 1=The corresponding FCH::SMI::EventStat bit is masked off whenever the system goes to S3 or higher sleep state.	[7]	0=No effect. 1=The corresponding FCH::SMI::EventStat bit is masked off whenever the system goes to S3 or higher sleep state.	[8]	0=No effect. 1=The corresponding FCH::SMI::EventStat bit is masked off whenever the system goes
Bit	Description																				
[0]	0=No effect. 1=The corresponding FCH::SMI::EventStat bit is masked off whenever the system goes to S3 or higher sleep state.																				
[1]	0=No effect. 1=The corresponding FCH::SMI::EventStat bit is masked off whenever the system goes to S3 or higher sleep state.																				
[2]	0=No effect. 1=The corresponding FCH::SMI::EventStat bit is masked off whenever the system goes to S3 or higher sleep state.																				
[3]	0=No effect. 1=The corresponding FCH::SMI::EventStat bit is masked off whenever the system goes to S3 or higher sleep state.																				
[4]	0=No effect. 1=The corresponding FCH::SMI::EventStat bit is masked off whenever the system goes to S3 or higher sleep state.																				
[5]	0=No effect. 1=The corresponding FCH::SMI::EventStat bit is masked off whenever the system goes to S3 or higher sleep state.																				
[6]	0=No effect. 1=The corresponding FCH::SMI::EventStat bit is masked off whenever the system goes to S3 or higher sleep state.																				
[7]	0=No effect. 1=The corresponding FCH::SMI::EventStat bit is masked off whenever the system goes to S3 or higher sleep state.																				
[8]	0=No effect. 1=The corresponding FCH::SMI::EventStat bit is masked off whenever the system goes																				

		to S3 or higher sleep state.
[9]		0=No effect. 1=The corresponding FCH::SMI::EventStat bit is masked off whenever the system goes to S3 or higher sleep state.
[10]		0=No effect. 1=The corresponding FCH::SMI::EventStat bit is masked off whenever the system goes to S3 or higher sleep state.
[11]		0=No effect. 1=The corresponding FCH::SMI::EventStat bit is masked off whenever the system goes to S3 or higher sleep state.
[12]		0=No effect. 1=The corresponding FCH::SMI::EventStat bit is masked off whenever the system goes to S3 or higher sleep state.
[13]		0=No effect. 1=The corresponding FCH::SMI::EventStat bit is masked off whenever the system goes to S3 or higher sleep state.
[14]		0=No effect. 1=The corresponding FCH::SMI::EventStat bit is masked off whenever the system goes to S3 or higher sleep state.
[15]		0=No effect. 1=The corresponding FCH::SMI::EventStat bit is masked off whenever the system goes to S3 or higher sleep state.
[16]		0=No effect. 1=The corresponding FCH::SMI::EventStat bit is masked off whenever the system goes to S3 or higher sleep state.
[17]		0=No effect. 1=The corresponding FCH::SMI::EventStat bit is masked off whenever the system goes to S3 or higher sleep state.
[18]		0=No effect. 1=The corresponding FCH::SMI::EventStat bit is masked off whenever the system goes to S3 or higher sleep state.
[19]		0=No effect. 1=The corresponding FCH::SMI::EventStat bit is masked off whenever the system goes to S3 or higher sleep state.
[20]		0=No effect. 1=The corresponding FCH::SMI::EventStat bit is masked off whenever the system goes to S3 or higher sleep state.
[21]		0=No effect. 1=The corresponding FCH::SMI::EventStat bit is masked off whenever the system goes to S3 or higher sleep state.
[22]		0=No effect. 1=The corresponding FCH::SMI::EventStat bit is masked off whenever the system goes to S3 or higher sleep state.
[23]		0=No effect. 1=The corresponding FCH::SMI::EventStat bit is masked off whenever the system goes to S3 or higher sleep state.
[24]		0=No effect. 1=The corresponding FCH::SMI::EventStat bit is masked off whenever the system goes to S3 or higher sleep state.
[25]		0=No effect. 1=The corresponding FCH::SMI::EventStat bit is masked off whenever the system goes to S3 or higher sleep state.
[26]		0=No effect. 1=The corresponding FCH::SMI::EventStat bit is masked off whenever the system goes to S3 or higher sleep state.
[27]		0=No effect. 1=The corresponding FCH::SMI::EventStat bit is masked off whenever the system goes to S3 or higher sleep state.
[28]		0=No effect. 1=The corresponding FCH::SMI::EventStat bit is masked off whenever the system goes to S3 or higher sleep state.
[29]		0=No effect. 1=The corresponding FCH::SMI::EventStat bit is masked off whenever the system goes to S3 or higher sleep state.
[30]		0=No effect. 1=The corresponding FCH::SMI::EventStat bit is masked off whenever the system goes to S3 or higher sleep state.
[31]		0=No effect. 1=The corresponding FCH::SMI::EventStat bit is masked off whenever the system goes to S3 or higher sleep state.

**SMI<sub>x</sub>030 [CapturedData] (FCH::SMI::CapturedData)**

Read-only.

\_aliasHOST; SMI<sub>x</sub>030; SMI=FED8\_0200h

Bits	Description
31:0	<b>CapturedData.</b> Read-only. Reset: XXXX_XXXXh. This is the buffer to capture Write data for the last transaction that caused an SMI. NOTE: This buffer has no meaning for a Read trap.

**SMIx034 [CapturedValid] (FCH::SMI::CapturedVal)**

Read-only.

\_aliasHOST; SMIx034; SMI=FED8\_0200h

Bits	Description										
31:4	Reserved.										
3:0	<b>CapturedValid.</b> Read-only. Reset: Xh. This is the byte valid buffer to signal which byte is captured for the last transaction that caused the SMI. <b>ValidValues:</b>										
	<table> <tr> <th>Bit</th><th>Description</th></tr> <tr> <td>[0]</td><td>Byte[0]: 0=Not valid. 1=Valid.</td></tr> <tr> <td>[1]</td><td>Byte[1]: 0=Not valid. 1=Valid.</td></tr> <tr> <td>[2]</td><td>Byte[2]: 0=Not valid. 1=Valid.</td></tr> <tr> <td>[3]</td><td>Byte[3]: 0=Not valid. 1=Valid.</td></tr> </table>	Bit	Description	[0]	Byte[0]: 0=Not valid. 1=Valid.	[1]	Byte[1]: 0=Not valid. 1=Valid.	[2]	Byte[2]: 0=Not valid. 1=Valid.	[3]	Byte[3]: 0=Not valid. 1=Valid.
Bit	Description										
[0]	Byte[0]: 0=Not valid. 1=Valid.										
[1]	Byte[1]: 0=Not valid. 1=Valid.										
[2]	Byte[2]: 0=Not valid. 1=Valid.										
[3]	Byte[3]: 0=Not valid. 1=Valid.										

**SMIx038 [EPBIF\_AER\_Straps] (FCH::SMI::EPBIF\_AER\_Straps)**

Read-write. Reset: 07FE\_FFEh.

\_aliasHOST; SMIx038; SMI=FED8\_0200h

Bits	Description
31:28	Reserved.
27	<b>StrapBifInternalErrEnFch.</b> Read-write. Reset: 0. Internal error enable.
26	<b>StrapBifPoisonedAdvisoryNonfatalAFch.</b> Read-write. Reset: 1. Poisoned TLP as advisory non-fatal.
25	<b>StrapBifAcsDirectTranslatedP2pFch.</b> Read-write. Reset: 1. ACS direct translated P2P enable.
24	<b>StrapBifAcsUpstreamForwardingFch.</b> Read-write. Reset: 1. ACS upstream forwarding enable.
23	<b>StrapBifAcsP2pCompletionRedirectFch.</b> Read-write. Reset: 1. ACS P2P completion redirect enable.
22	<b>StrapBifAcsP2pRequestRedirectFch.</b> Read-write. Reset: 1. ACS P2P request redirect enable.
21	<b>StrapBifAcsTranslationBlockingFch.</b> Read-write. Reset: 1. ACS translation blocking enable.
20	<b>StrapBifAcsSourceValidationFch.</b> Read-write. Reset: 1. ACS source validation enable.
19	<b>StrapBifAcsEnFch.</b> Read-write. Reset: 1. ACS enable.
18	<b>StrapBifFirstRcvdErrLogFch.</b> Read-write. Reset: 1. First received error log.
17	<b>StrapBifEcrcCheckEnFch.</b> Read-write. Reset: 1. ECRC check enable.
16	<b>StrapBifEcrcGenEnFch.</b> Read-write. Reset: 0. ECRC generate enable.
15	<b>StrapBifCplAbortErrEnFch.</b> Read-write. Reset: 1. Completer abort error enable.
14	<b>StrapBifRxIgnoreVend0UrFch.</b> Read-write. Reset: 1. Ignore Vendor 0 error.
13	<b>StrapBifRxIgnoreTcErrFch.</b> Read-write. Reset: 1. Ignore traffic class error.
12	<b>StrapBifRxIgnoreMsgErrFch.</b> Read-write. Reset: 1. Ignore message error.
11	<b>StrapBifRxIgnoreMaxPayloadErrFch.</b> Read-write. Reset: 1. Ignore maximum payload error.
10	<b>StrapBifRxIgnoreLenMismatchErrFch.</b> Read-write. Reset: 1. Ignore length mismatch error.
9	<b>StrapBifRxIgnoreIoUrErrFch.</b> Read-write. Reset: 1. Ignore IO UR error.
8	<b>StrapBifRxIgnoreIoErrFch.</b> Read-write. Reset: 1. Ignore IO error.
7	<b>StrapBifRxIgnoreEpErrFch.</b> Read-write. Reset: 1. Ignore poisoned TLP error.
6	<b>StrapBifRxIgnoreCplErrFch.</b> Read-write. Reset: 1. Ignore completion error.
5	<b>StrapBifRxIgnoreCfgUrFch.</b> Read-write. Reset: 1. Ignore configuration UR error.
4	<b>StrapBifRxIgnoreCfgErrFch.</b> Read-write. Reset: 1. Ignore configuration error.
3	<b>StrapBifRxIgnoreBeErrFch.</b> Read-write. Reset: 1. Ignore byte enable error.
2	<b>StrapBifErrReportingDisFch.</b> Read-write. Reset: 1. Error reporting disable.
1	<b>StrapBifAerEnFch.</b> Read-write. Reset: 1. AER enable.

0	<b>StrapBifStickyOverrideS5</b> . Read-write. Reset: 0. 1=Values in this register would override straps loaded from EEPROM.
---	---

**SMIx03C [DataErrorStatus] (FCH::SMI::DataErrStat)**

Read,Write-1-to-clear. Reset: 0000\_0000h.

\_aliasHOST; SMIx03C; SMI=FED8\_0200h

Bits	Description
31:8	Reserved.
7	<b>SlrqIochk</b> . Read,Write-1-to-clear. Reset: 0. Serial Iochk error.
6	<b>SataPerr</b> . Read,Write-1-to-clear. Reset: 0. SATA controller internal parity error status.
5	<b>UmiUncorrectableErr</b> . Read,Write-1-to-clear. Reset: 0. UMI uncorrectable error status.
4	<b>UmiCorrectableErr</b> . Read,Write-1-to-clear. Reset: 0. UMIcorrectable error status.
3	<b>AbUmiGppPerr</b> . Read,Write-1-to-clear. Reset: 0. AB/UMI/GPP parity error status.
2	<b>InternalGPPSerr</b> . Read,Write-1-to-clear. Reset: 0. Internal error status: FCH has detected an internal error from upstream bridge.
1	<b>InternalPerr</b> . Read,Write-1-to-clear. Reset: 0. Internal devices PERR error status.
0	<b>InternalSerr</b> . Read,Write-1-to-clear. Reset: 0. Internal devices SERR error status.

**SMIx040 [SciMap0] (FCH::SMI::SciMap0)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; SMIx040; SMI=FED8\_0200h

Bits	Description
31:29	Reserved.
28:24	<b>SciMap3</b> . Read-write. Reset: 00h. Specifies the mapping of LPC_PME_L to one of the 32 bits in FCH::SMI::EventStat.
23:21	Reserved.
20:16	<b>SciMap2</b> . Read-write. Reset: 00h. Specifies the mapping of AGPIO3 to one of the 32 bits in FCH::SMI::EventStat.
15:13	Reserved.
12:8	<b>SciMap1</b> . Read-write. Reset: 00h. Specifies the mapping of GENINT2_L to one of the 32 bits in FCH::SMI::EventStat.
7:5	Reserved.
4:0	<b>SciMap0</b> . Read-write. Reset: 00h. Specifies the mapping of GENINT1_L to one of the 32 bits in FCH::SMI::EventStat.

**SMIx044 [SciMap1] (FCH::SMI::SciMap1)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; SMIx044; SMI=FED8\_0200h

Bits	Description
31:29	Reserved.
28:24	<b>SciMap7</b> . Read-write. Reset: 00h. Specifies the mapping of AGPIO5 to one of the 32 bits in FCH::SMI::EventStat.
23:21	Reserved.
20:16	<b>SciMap6</b> . Read-write. Reset: 00h. Specifies the mapping of SPKR to one of the 32 bits in FCH::SMI::EventStat.
15:13	Reserved.
12:8	<b>SciMap5</b> . Read-write. Reset: 00h. Specifies the mapping of LPC_PD_L to one of the 32 bits in FCH::SMI::EventStat.
7:5	Reserved.
4:0	<b>SciMap4</b> . Read-write. Reset: 00h. Specifies the mapping of AGPIO4 to one of the 32 bits in FCH::SMI::EventStat.

**SMIx048 [SciMap2] (FCH::SMI::SciMap2)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; SMIx048; SMI=FED8\_0200h

Bits	Description
31:29	Reserved.
28:24	<b>SciMap11.</b> Read-write. Reset: 00h. Specifies the mapping of AGPIO7 to one of the 32 bits in FCH::SMI::EventStat. CopyFrom: SciMap8.
23:21	Reserved.
20:16	<b>SciMap10.</b> Read-write. Reset: 00h. Specifies the mapping of AGPIO6 to one of the 32 bits in FCH::SMI::EventStat. CopyFrom: SciMap8.
15:13	Reserved.
12:8	<b>SciMap9.</b> Read-write. Reset: 00h. Specifies the mapping of LPC_SMI_L to one of the 32 bits in FCH::SMI::EventStat. CopyFrom: SciMap8.
7:5	Reserved.
4:0	<b>SciMap8.</b> Read-write. Reset: 00h. Specifies the mapping of WAKE_L to one of the 32 bits in FCH::SMI::EventStat.

**SMIx04C [SciMap3] (FCH::SMI::SciMap3)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; SMIx04C; SMI=FED8\_0200h

Bits	Description
31:29	Reserved.
28:24	<b>SciMap15.</b> Read-write. Reset: 00h. Specifies the mapping of USB_OC3_L to one of the 32 bits in FCH::SMI::EventStat. CopyFrom: SciMap12.
23:21	Reserved.
20:16	<b>SciMap14.</b> Read-write. Reset: 00h. Specifies the mapping of USB_OC2_L to one of the 32 bits in FCH::SMI::EventStat. CopyFrom: SciMap12.
15:13	Reserved.
12:8	<b>SciMap13.</b> Read-write. Reset: 00h. Specifies the mapping of USB_OC1_L to one of the 32 bits in FCH::SMI::EventStat. CopyFrom: SciMap12.
7:5	Reserved.
4:0	<b>SciMap12.</b> Read-write. Reset: 00h. Specifies the mapping of USB_OC0_L to one of the 32 bits in FCH::SMI::EventStat.

**SMIx050 [SciMap4] (FCH::SMI::SciMap4)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; SMIx050; SMI=FED8\_0200h

Bits	Description
31:29	Reserved.
28:24	<b>SciMap19.</b> Read-write. Reset: 00h. Specifies the mapping of SYS_RESET_L to one of the 32 bits in FCH::SMI::EventStat. CopyFrom: SciMap16.
23:21	Reserved.
20:16	<b>SciMap18.</b> Read-write. Reset: 00h. Specifies the mapping of FANIN0 to one of the 32 bits in FCH::SMI::EventStat. CopyFrom: SciMap16.
15:13	Reserved.
12:8	<b>SciMap17.</b> Read-write. Reset: 00h. Specifies the mapping of ESPI_RESET_L to one of the 32 bits in FCH::SMI::EventStat. CopyFrom: SciMap16.
7:5	Reserved.
4:0	<b>SciMap16.</b> Read-write. Reset: 00h. Specifies the mapping of AGPIO23 to one of the 32 bits in FCH::SMI::EventStat.

**SMIx054 [SciMap5] (FCH::SMI::SciMap5)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; SMIx054; SMI=FED8\_0200h

Bits	Description
31:29	Reserved.
28:24	<b>SciMap23.</b> Read-write. Reset: 00h. Specifies the mapping of AGPIO8 to one of the 32 bits in FCH::SMI::EventStat. CopyFrom: SciMap20.
23:21	Reserved.
20:16	<b>SciMap22.</b> Read-write. Reset: 00h. Specifies the mapping of AGPIO9 to one of the 32 bits in FCH::SMI::EventStat. CopyFrom: SciMap20.
15:13	Reserved.
12:8	<b>SciMap21.</b> Read-write. Reset: 00h. Specifies the mapping of PWR_BTN_L to one of the 32 bits in FCH::SMI::EventStat. CopyFrom: SciMap20.
7:5	Reserved.
4:0	<b>SciMap20.</b> Read-write. Reset: 00h. Specifies the mapping of AGPIO40 to one of the 32 bits in FCH::SMI::EventStat.

**SMIx058 [SciMap6] (FCH::SMI::SciMap6)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; SMIx058; SMI=FED8\_0200h

Bits	Description
31:29	Reserved.
28:24	<b>SciMap27.</b> Read-write. Reset: 00h. Specifies the mapping of eSPI_WAKE_PME (active high) to one of the 32 bits in FCH::SMI::EventStat.
23:21	Reserved.
20:16	<b>SciMap26.</b> Read-write. Reset: 00h. Specifies the mapping of eSPI system event (active high) to one of the 32 bits in FCH::SMI::EventStat.
15:0	Reserved.

**SMIx05C [SciMap7] (FCH::SMI::SciMap7)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; SMIx05C; SMI=FED8\_0200h

Bits	Description
31:21	Reserved.
20:16	<b>SciMap30.</b> Read-write. Reset: 00h. Specifies the mapping of NB GPP Hot Plug to one of the 32 bits in FCH::SMI::EventStat.
15:13	Reserved.
12:8	<b>SciMap29.</b> Read-write. Reset: 00h. Specifies the mapping of NB GPP_PME to one of the 32 bits in FCH::SMI::EventStat.
7:0	Reserved.

**SMIx060 [SciMap8] (FCH::SMI::SciMap8)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; SMIx060; SMI=FED8\_0200h

Bits	Description
31:5	Reserved.
4:0	<b>SciMap32.</b> Read-write. Reset: 00h. Specifies the mapping of WAKE_L to one of the 32 bits in FCH::SMI::EventStat.

**SMIx064 [SciMap9] (FCH::SMI::SciMap9)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; SMIx064; SMI=FED8\_0200h



Bits	Description
31:29	Reserved.
28:24	<b>SciMap39</b> . Read-write. Reset: 00h. Specifies the mapping of Azalia PME to one of the 32 bits in FCH::SMI::EventStat.
23:0	Reserved.

**SMIx068 [SciMap10] (FCH::SMI::SciMap10)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; SMIx068; SMI=FED8\_0200h

Bits	Description
31:29	Reserved.
28:24	<b>SciMap43</b> . Read-write. Reset: 00h. Specifies the mapping of AltHPET timer event to one of the 32 bits in FCH::SMI::EventStat.
23:13	Reserved.
12:8	<b>SciMap41</b> . Read-write. Reset: 00h. Specifies the mapping of GPIO Interrupt to one of the 32 bits in FCH::SMI::EventStat.
7:5	Reserved.
4:0	<b>SciMap40</b> . Read-write. Reset: 00h. Specifies the mapping of USB_PD_I2C4 (active high) interrupt to one of the 32 bits in FCH::SMI::EventStat.

**SMIx06C [SciMap11] (FCH::SMI::SciMap11)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; SMIx06C; SMI=FED8\_0200h

Bits	Description
31:29	Reserved.
28:24	<b>SciMap47</b> . Read-write. Reset: 00h. Specifies the mapping of SMBUS0 Interrupt event to one of the 32 bits in FCH::SMI::EventStat. CopyFrom: SciMap44.
23:21	Reserved.
20:16	<b>SciMap46</b> . Read-write. Reset: 00h. Specifies the mapping of I2S wake event to one of the 32 bits in FCH::SMI::EventStat. CopyFrom: SciMap44.
15:13	Reserved.
12:8	<b>SciMap45</b> . Read-write. Reset: 00h. Specifies the mapping of ASF Master and Slave interrupt event to one of the 32 bits in FCH::SMI::EventStat. CopyFrom: SciMap44.
7:5	Reserved.
4:0	<b>SciMap44</b> . Read-write. Reset: 00h. Specifies the mapping of internal FAN/THERMAL event to one of the 32 bits in FCH::SMI::EventStat.

**SMIx070 [SciMap12] (FCH::SMI::SciMap12)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; SMIx070; SMI=FED8\_0200h

Bits	Description
31:29	Reserved.
28:24	<b>SciMap51</b> . Read-write. Reset: 00h. Specifies the mapping of Power Button event to one of the 32 bits in FCH::SMI::EventStat.
23:21	Reserved.
20:16	<b>SciMap50</b> . Read-write. Reset: 00h. Specifies the mapping of LLB# to one of the 32 bits in FCH::SMI::EventStat.
15:13	Reserved.
12:8	<b>SciMap49</b> . Read-write. Reset: 00h. Specifies the mapping of internal traffic monitor to one of the 32 bits in FCH::SMI::EventStat.
7:5	Reserved.
4:0	<b>SciMap48</b> . Read-write. Reset: 00h. Specifies the mapping of TWARN pin to one of the 32 bits in



	FCH::SMI::EventStat.
--	----------------------

**SMIx074 [SciMap13] (FCH::SMI::SciMap13)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; SMIx074; SMI=FED8\_0200h

Bits	Description
31:29	Reserved.
28:24	<b>SciMap55.</b> Read-write. Reset: 00h. Specifies the mapping of RAS_event status to one of the 32 bits in FCH::SMI::EventStat. CopyFrom: SciMap52.
23:21	Reserved.
20:16	<b>SciMap54.</b> Read-write. Reset: 00h. Specifies the mapping of SCI assertion message from APU to one of the 32 bits in FCH::SMI::EventStat. CopyFrom: SciMap52.
15:13	Reserved.
12:8	<b>SciMap53.</b> Read-write. Reset: 00h. Specifies the mapping of hardware assertion message from APU to one of the 32 bits in FCH::SMI::EventStat. CopyFrom: SciMap52.
7:5	Reserved.
4:0	<b>SciMap52.</b> Read-write. Reset: 00h. Specifies the mapping of PROHOT# pin to one of the 32 bits in FCH::SMI::EventStat.

**SMIx078 [SciMap14] (FCH::SMI::SciMap14)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; SMIx078; SMI=FED8\_0200h

Bits	Description
31:21	Reserved.
20:16	<b>SciMap58.</b> Read-write. Reset: 00h. Specifies the mapping of AcDcTimerEvent to one of the 32 bits in FCH::SMI::EventStat.
15:13	Reserved.
12:8	<b>SciMap57.</b> Read-write. Reset: 00h. Specifies the mapping of XHC1 PME event to one of the 32 bits in FCH::SMI::EventStat.
7:5	Reserved.
4:0	<b>SciMap56.</b> Read-write. Reset: 00h. Specifies the mapping of XHC0 PME event to one of the 32 bits in FCH::SMI::EventStat.

**SMIx080 [SmiStatus0] (FCH::SMI::SmiStat0)**

Read,Write-1-to-clear. Reset: 0000\_0000h.

\_aliasHOST; SMIx080; SMI=FED8\_0200h

Bits	Description
31	Reserved.
30	<b>NBGPPHP_event30.</b> Read,Write-1-to-clear. Reset: 0. Status of NB Hot Plug event.
29	<b>NBGppPme_event29.</b> Read,Write-1-to-clear. Reset: 0. Status of NB GPP PME.
28	Reserved.
27	<b>eSPI_WAKE_PME_event27.</b> Read,Write-1-to-clear. Reset: 0. Status of eSPI_WAKE_PME_event27.
26	<b>eSPI_system_event26.</b> Read,Write-1-to-clear. Reset: 0. Status of eSPI system event.
25:24	Reserved.
23	<b>Gevent23StatusEvent23.</b> Read,Write-1-to-clear. Reset: 0. Status of AGPIO8.
22	<b>Gevent22StatusEvent22.</b> Read,Write-1-to-clear. Reset: 0. Status of AGPIO9.
21	<b>Gevent21StatusEvent21.</b> Read,Write-1-to-clear. Reset: 0. Status of PWR_BTN_L.
20	<b>Gevent20StatusEvent20.</b> Read,Write-1-to-clear. Reset: 0. Status of AGPIO40.
19	<b>Gevent19StatusEvent19.</b> Read,Write-1-to-clear. Reset: 0. Status of SYS_RESET_L.
18	<b>Gevent18StatusEvent18.</b> Read,Write-1-to-clear. Reset: 0. Status of FANIN0.
17	<b>Gevent17StatusEvent17.</b> Read,Write-1-to-clear. Reset: 0. Status of ESPI_RESET_L.

16	<b>Gevent16StatusEvent16.</b> Read,Write-1-to-clear. Reset: 0. Status of AGPIO23.
15	<b>Gevent15StatusEvent15.</b> Read,Write-1-to-clear. Reset: 0. Status of USB_OC3_L.
14	<b>Gevent14StatusEvent14.</b> Read,Write-1-to-clear. Reset: 0. Status of USB_OC2_L.
13	<b>Gevent13StatusEvent13.</b> Read,Write-1-to-clear. Reset: 0. Status of USB_OC1_L.
12	<b>Gevent12StatusEvent12.</b> Read,Write-1-to-clear. Reset: 0. Status of USB_OC0_L.
11	<b>Gevent11StatusEvent11.</b> Read,Write-1-to-clear. Reset: 0. Status of AGPIO7.
10	<b>Gevent10StatusEvent10.</b> Read,Write-1-to-clear. Reset: 0. Status of AGPIO6.
9	<b>Gevent9StatusEvent9.</b> Read,Write-1-to-clear. Reset: 0. Status of LPC_SMI_L.
8	<b>Gevent8StatusEvent8.</b> Read,Write-1-to-clear. Reset: 0. Status of WAKE_L.
7	<b>Gevent7StatusEvent7.</b> Read,Write-1-to-clear. Reset: 0. Status of AGPIO5.
6	<b>Gevent6StatusEvent6.</b> Read,Write-1-to-clear. Reset: 0. Status of SPKR.
5	<b>Gevent5StatusEvent5.</b> Read,Write-1-to-clear. Reset: 0. Status of LPC_PD_L.
4	<b>Gevent4StatusEvent4.</b> Read,Write-1-to-clear. Reset: 0. Status of AGPIO4.
3	<b>Gevent3StatusEvent3.</b> Read,Write-1-to-clear. Reset: 0. Status of LPC_PME_L.
2	<b>Gevent2StatusEvent2.</b> Read,Write-1-to-clear. Reset: 0. Status of AGPIO3.
1	<b>Gevent1StatusEvent1.</b> Read,Write-1-to-clear. Reset: 0. Status of GENINT2_L.
0	<b>Gevent0StatusEvent0.</b> Read,Write-1-to-clear. Reset: 0. Status of GENINT1_L.

**SMIx084 [SmiStatus1] (FCH::SMI::SmiStat1)**

Read,Write-1-to-clear. Reset: 0000\_0000h.

\_aliasHOST; SMIx084; SMI=FED8\_0200h

Bits	Description
31:27	Reserved.
26	<b>AcDcTimerEventEvent58.</b> Read,Write-1-to-clear. Reset: 0. Status of AcDcTimer wake up event.
25	<b>Xhc1PmeEvent57.</b> Read,Write-1-to-clear. Reset: 0. Status of XHC1 PME.
24	<b>Xhc0PmeEvent56.</b> Read,Write-1-to-clear. Reset: 0. Status of XHC0 PME.
23	<b>RasEvent55.</b> Read,Write-1-to-clear. Reset: 0. Internal devices SERR error status.
22	<b>ApuSciAssertionEvent54.</b> Read,Write-1-to-clear. Reset: 0. Status of APU SCI request.
21	<b>ApuHwAssertionEvent53.</b> Read,Write-1-to-clear. Reset: 0. Status of APU hardware assertion.
20	<b>ProcHotEvent52.</b> Read,Write-1-to-clear. Reset: 0. Status of ProcHot event.
19	<b>PwrButtonEvent51.</b> Read,Write-1-to-clear. Reset: 0. Status of PwrButton (rising edge).
18	<b>PwrButtonEvent50.</b> Read,Write-1-to-clear. Reset: 0. Status of iLLB# assertion.
17	<b>TrafficMonitorIntrEvent49.</b> Read,Write-1-to-clear. Reset: 0. Status of FCH Traffic Monitor Interrupt.
16	<b>TwarnEvent48.</b> Read,Write-1-to-clear. Reset: 0. Status of FCH TWARN.
15	<b>Smbus0Event47.</b> Read,Write-1-to-clear. Reset: 0. Status of FCH SMBUS0 master interrupt.
14	<b>Smbus0Event46.</b> Read,Write-1-to-clear. Reset: 0. Status of I2S wake event.
13	<b>AsfrIntrEvent45.</b> Read,Write-1-to-clear. Reset: 0. Status of FCH ASF master and slave interrupt.
12	<b>FanThermalGeventEvent44.</b> Read,Write-1-to-clear. Reset: 0. Status of FCH fan thermal.
11	<b>AltMmTimerStsEvent43.</b> Read,Write-1-to-clear. Reset: 0. Status of AltMmTimer Alarm.
10	Reserved.
9	<b>GpioEvent41.</b> Read,Write-1-to-clear. Reset: 0. Status of FCH GPIO Control interrupt.
8	<b>GpioEvent40.</b> Read,Write-1-to-clear. Reset: 0. Status of USB_PD_I2C4.
7	<b>AZPME_event39.</b> Read,Write-1-to-clear. Reset: 0. Status of Azalia wake event.
6:1	Reserved.
0	<b>WakeLEvent32.</b> Read,Write-1-to-clear. Reset: 0. Status of WAKE_L.

**SMIx088 [SmiStatus2] (FCH::SMI::SmiStat2)**

Read,Write-1-to-clear. Reset: 0000\_0000h.

\_aliasHOST; SMIx088; SMI=FED8\_0200h

Bits	Description
------	-------------

31:27	Reserved.
26	<b>Emulate64Event90.</b> Read,Write-1-to-clear. Reset: 0. Status of Emulation IO Port 60h/64h.
25:21	Reserved.
20	<b>PciSerrEvent84.</b> Read,Write-1-to-clear. Reset: 0. Status of SERR assertion on PCI bus.
19	<b>ProtHotEvent83.</b> Read,Write-1-to-clear. Reset: 0. Status of ProtHot event.
18	<b>VbatLowEvent82.</b> Read,Write-1-to-clear. Reset: 0. Status of VBAT low.
17	<b>IntruderAlertStsEvent81.</b> Read,Write-1-to-clear. Reset: 0. Status of IntruderAlertSts.
16:15	Reserved.
14	<b>Smbus0IntrEvent78.</b> Read,Write-1-to-clear. Reset: 0. Status of SMBUS0 interrupt request.
13	<b>SerialIrqSmiEvent77.</b> Read,Write-1-to-clear. Reset: 0. Status of SMI request from Serial IRQ.
12	<b>UsbSmiEvent76.</b> Read,Write-1-to-clear. Reset: 0. Status of USB SMI request.
11	<b>SmiCmdportEvent75.</b> Read,Write-1-to-clear. Reset: 0. Status of Writing SMI Command Port.
10	<b>PwrbtnEvent74.</b> Read,Write-1-to-clear. Reset: 0. Status of Power Button being pressed.
9	<b>BiosRlsEvent73.</b> Read,Write-1-to-clear. Reset: 0. Status of BIOS_RLS. See FCH::PM::AcpiCfg[BiosRls].
8	<b>GblRlsEvent72.</b> Read,Write-1-to-clear. Reset: 0. Status of GBL event. See FCH::PM::PmCtl[GblRls].
7	Reserved.
6	<b>UsbPdiI2C4IntrBSmiEvent70.</b> Read,Write-1-to-clear. Reset: 0. Status of USB_PD_I2C4_intrB SMI request.
5	<b>NBGppHpPulseSmiEvent69.</b> Read,Write-1-to-clear. Reset: 0. Status of NBGppHpPulse SMI request.
4	<b>NBGppPmePulseSmiEvent68.</b> Read,Write-1-to-clear. Reset: 0. Status of NBGppPmePulse SMI request.
3	<b>SataAhciSmiEvent67.</b> Read,Write-1-to-clear. Reset: 0. Status of SATA AHCI SMI request.
2	<b>AL2H_ACPI_Assertion_Event67.</b> Read,Write-1-to-clear. Reset: 0. Status of AL2H ACPI Assertion.
1	<b>SlpTypeEvent65.</b> Read,Write-1-to-clear. Reset: 0. Status of writing FCH::PM::PmCtl[SlpTyp].
0	<b>KBRstEvent64.</b> Read,Write-1-to-clear. Reset: 0. Status is set when KeyBoard Reset.

**SMIx090 [SmiStatus4] (FCH::SMI::SmiStat4)**

Read,Write-1-to-clear. Reset: 0000\_0000h.

\_aliasHOST; SMIx090; SMI=FED8\_0200h

Bits	Description
31:29	Reserved.
28	<b>CfgTrapping0Event156.</b> Read,Write-1-to-clear. Reset: 0. Status of PCI configuration cycle Trapping0 SMI request.
27:25	Reserved.
24	<b>MemTrapping0Event152.</b> Read,Write-1-to-clear. Reset: 0. Status of memory Trapping0 SMI request.
23	<b>IoTrapping3Event151.</b> Read,Write-1-to-clear. Reset: 0. Status of IO Trapping3 SMI request.
22	<b>IoTrapping2Event150.</b> Read,Write-1-to-clear. Reset: 0. Status of IO Trapping2 SMI request.
21	<b>IoTrapping1Event149.</b> Read,Write-1-to-clear. Reset: 0. Status of IO Trapping1 SMI request.
20	<b>IoTrapping0Event148.</b> Read,Write-1-to-clear. Reset: 0. Status of IO Trapping0 SMI request.
19	Reserved.
18	<b>eSPISMIEvent146.</b> Read,Write-1-to-clear. Reset: 0. Status of eSPI SMI event.
17	Reserved.
16	<b>AbSmiTrapEvent144.</b> Read,Write-1-to-clear. Reset: 0. Status of AB SMI trapping request.
15	<b>LongTimerEvent143.</b> Read,Write-1-to-clear. Reset: 0. Status of Long Timer SMI request.
14	<b>ShortTimerEvent142.</b> Read,Write-1-to-clear. Reset: 0. Status of Short Timer SMI request.
13	<b>Cf9WriteEvent141.</b> Read,Write-1-to-clear. Reset: 0. 0=CF9 bit[2] is not written to 1. 1=CF9 bit[2] has been written to 1.
12:6	Reserved.
5	<b>Fanin0StsEvent133.</b> Read,Write-1-to-clear. Reset: 0. Status of FanIn0 event.
4:0	Reserved.

**SMIx094 [SmiPointer] (FCH::SMI::SmiPtr)**

Read-only. Reset: 0000h.	
This register is meant as a faster mechanism to locate the SMI source. BIOS can examine this register to find out the SMI source instead of reading FCH::SMI::SmiStat0 through FCH::SMI::SmiStat4 individually.	
_aliasHOST; SMIx094; SMI=FED8_0200h	
Bits	Description
15:6	Reserved.
5	<b>SmiStatusSource4</b> . Read-only. Reset: 0. Indicates whether the SMI source is from FCH::SMI::SmiStat4 if the corresponding SMI enable is selected.
4	Reserved.
3	<b>SmiStatusSource2</b> . Read-only. Reset: 0. Indicates whether the SMI source is from FCH::SMI::SmiStat2 if the corresponding SMI enable is selected.
2	<b>SmiStatusSource1</b> . Read-only. Reset: 0. Indicates whether the SMI source is from FCH::SMI::SmiStat1 if the corresponding SMI enable is selected.
1	<b>SmiStatusSource0</b> . Read-only. Reset: 0. Indicates whether the SMI source is from FCH::SMI::SmiStat0 if the corresponding SMI enable is selected.
0	<b>SmiSciSource</b> . Read-only. Reset: 0. Indicates whether the SMI source is from FCH::SMI::SmiSciStat.

#### SMIx096 [SmiTimer] (FCH::SMI::SmiTimer)

Read-write. Reset: 0000h.	
There are two SMI timers: the short timer runs at 1 us unit time and the long timer runs at 1 ms unit time. This register is actually made up of two sets of registers depending on the setting of FCH::SMI::SmiTrig0[SmiTimerSel]. If FCH::SMI::SmiTrig0[SmiTimerSel] == 0, then FCH::SMI::SmiTimer is for the short timer. If FCH::SMI::SmiTrig0[SmiTimerSel] == 1, then FCH::SMI::SmiTimer is for the long timer. The default setting selects this register as "SmiShortTimer"; software needs to set the "SmiTimerSel = 1" to select this register as "SmiLongTimer".	
_aliasHOST; SMIx096; SMI=FED8_0200h	
Bits	Description
15	<b>TimerEn</b> . Read-write. Reset: 0. 0=Disable. 1=Enable the SMI short Timer or long timer, which is selected by FCH::SMI::SmiTrig0[SmiTimerSel].
14:0	<b>SmiTimerCount</b> . Read-write. Reset: 0000h. Actual timer duration = TimerTime + 1 unit.

#### SMIx098 [SmiTrig0] (FCH::SMI::SmiTrig0)

Read-write. Reset: 81FF_FFFFh.	
Bits[23:0] defines the trigger mode for 24 GEVENTS in FCH::SMI::SmiStat0[23:0]. Note these are different from FCH::SMI::SciTrig.	
_aliasHOST; SMIx098; SMI=FED8_0200h	
Bits	Description
31	<b>SmiEnB</b> . Read-write. Reset: 1. 0=Enable SMI function. 1=Disable.
30	Reserved.
29	<b>SmiTimerSel</b> . Read-write. Reset: 0. 0=Select FCH::SMI::SmiTimer to be SmiShortTimer register. 1=Select FCH::SMI::SmiTimer to be SmiLongTimer register.
28	<b>Eos</b> . Read-write. Reset: 0. This bit is set to 1 by software to enable SMI generation. It is cleared by hardware after an SMI event has occurred. When Eos is clear, subsequent pending SMI event will be blocked.
27:25	Reserved.
24	<b>TrappingIrqOnPic</b> . Read-write. Reset: 1. 0=SMI is generated when trapping IRQ[15:0] of IOAPIC. 1=SMI is generated when trapping IRQ[15:0] of PIC.
23	<b>SmiTrig23</b> . Read-write. Reset: 1. 0=Active low. 1=Active high.
22	<b>SmiTrig22</b> . Read-write. Reset: 1. 0=Active low. 1=Active high.
21	<b>SmiTrig21</b> . Read-write. Reset: 1. 0=Active low. 1=Active high.
20	<b>SmiTrig20</b> . Read-write. Reset: 1. 0=Active low. 1=Active high.
19	<b>SmiTrig19</b> . Read-write. Reset: 1. 0=Active low. 1=Active high.
18	<b>SmiTrig18</b> . Read-write. Reset: 1. 0=Active low. 1=Active high.

17	<b>SmiTrig17.</b> Read-write. Reset: 1. 0=Active low. 1=Active high.
16	<b>SmiTrig16.</b> Read-write. Reset: 1. 0=Active low. 1=Active high.
15	<b>SmiTrig15.</b> Read-write. Reset: 1. 0=Active low. 1=Active high.
14	<b>SmiTrig14.</b> Read-write. Reset: 1. 0=Active low. 1=Active high.
13	<b>SmiTrig13.</b> Read-write. Reset: 1. 0=Active low. 1=Active high.
12	<b>SmiTrig12.</b> Read-write. Reset: 1. 0=Active low. 1=Active high.
11	<b>SmiTrig11.</b> Read-write. Reset: 1. 0=Active low. 1=Active high.
10	<b>SmiTrig10.</b> Read-write. Reset: 1. 0=Active low. 1=Active high.
9	<b>SmiTrig9.</b> Read-write. Reset: 1. 0=Active low. 1=Active high.
8	<b>SmiTrig8.</b> Read-write. Reset: 1. 0=Active low. 1=Active high.
7	<b>SmiTrig7.</b> Read-write. Reset: 1. 0=Active low. 1=Active high.
6	<b>SmiTrig6.</b> Read-write. Reset: 1. 0=Active low. 1=Active high.
5	<b>SmiTrig5.</b> Read-write. Reset: 1. 0=Active low. 1=Active high.
4	<b>SmiTrig4.</b> Read-write. Reset: 1. 0=Active low. 1=Active high.
3	<b>SmiTrig3.</b> Read-write. Reset: 1. 0=Active low. 1=Active high.
2	<b>SmiTrig2.</b> Read-write. Reset: 1. 0=Active low. 1=Active high.
1	<b>SmiTrig1.</b> Read-write. Reset: 1. 0=Active low. 1=Active high.
0	<b>SmiTrig0.</b> Read-write. Reset: 1. 0=Active low. 1=Active high.

**SMIx09C [SmiTrig1] (FCH::SMI::SmiTrig1)**

Read-write.

Bits[23:0] defines the trigger mode for 24 IRQs.

\_aliasHOST; SMIx09C; SMI=FED8\_0200h

Bits	Description
31:24	Reserved.
23	<b>SmiIrq23Trig.</b> Read-write. Reset: 0. 0=Active low. 1=Active high.
22	<b>SmiIrq22Trig.</b> Read-write. Reset: 0. 0=Active low. 1=Active high.
21	<b>SmiIrq21Trig.</b> Read-write. Reset: 0. 0=Active low. 1=Active high.
20	<b>SmiIrq20Trig.</b> Read-write. Reset: 0. 0=Active low. 1=Active high.
19	<b>SmiIrq19Trig.</b> Read-write. Reset: 0. 0=Active low. 1=Active high.
18	<b>SmiIrq18Trig.</b> Read-write. Reset: 0. 0=Active low. 1=Active high.
17	<b>SmiIrq17Trig.</b> Read-write. Reset: 0. 0=Active low. 1=Active high.
16	<b>SmiIrq16Trig.</b> Read-write. Reset: 0. 0=Active low. 1=Active high.
15	<b>SmiIrq15Trig.</b> Read-write. Reset: 0. 0=Active low. 1=Active high.
14	<b>SmiIrq14Trig.</b> Read-write. Reset: 0. 0=Active low. 1=Active high.
13	<b>SmiIrq13Trig.</b> Read-write. Reset: 0. 0=Active low. 1=Active high.
12	<b>SmiIrq12Trig.</b> Read-write. Reset: 0. 0=Active low. 1=Active high.
11	<b>SmiIrq11Trig.</b> Read-write. Reset: 0. 0=Active low. 1=Active high.
10	<b>SmiIrq10Trig.</b> Read-write. Reset: 0. 0=Active low. 1=Active high.
9	<b>SmiIrq9Trig.</b> Read-write. Reset: 0. 0=Active low. 1=Active high.
8	<b>SmiIrq8Trig.</b> Read-write. Reset: 0. 0=Active low. 1=Active high.
7	<b>SmiIrq7Trig.</b> Read-write. Reset: 0. 0=Active low. 1=Active high.
6	<b>SmiIrq6Trig.</b> Read-write. Reset: 0. 0=Active low. 1=Active high.
5	<b>SmiIrq5Trig.</b> Read-write. Reset: 0. 0=Active low. 1=Active high.
4	<b>SmiIrq4Trig.</b> Read-write. Reset: 0. 0=Active low. 1=Active high.
3	<b>SmiIrq3Trig.</b> Read-write. Reset: 0. 0=Active low. 1=Active high.
2	<b>SmiIrq2Trig.</b> Read-write. Reset: 0. 0=Active low. 1=Active high.
1	<b>SmiIrq1Trig.</b> Read-write. Reset: 0. 0=Active low. 1=Active high.
0	<b>SmiIrq0Trig.</b> Read-write. Reset: 0. 0=Active low. 1=Active high.

**SMIx0A0 [SmiControl0] (FCH::SMI::SmiCtl0)**

Read-write. Reset: 0000\_0000h.

This register specifies the control mechanism for SMI sources in FCH::SMI::SmiStat0[15:0].

\_aliasHOST; SMIx0A0; SMI=FED8\_0200h

Bits	Description										
31:30	<b>Smicontrol15.</b> Read-write. Reset: 0h. Control for GEVENT15. <b>ValidValues:</b> <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>Disable.</td></tr> <tr> <td>1h</td><td>SMI.</td></tr> <tr> <td>2h</td><td>NMI.</td></tr> <tr> <td>3h</td><td>IRQ13.</td></tr> </table>	Value	Description	0h	Disable.	1h	SMI.	2h	NMI.	3h	IRQ13.
Value	Description										
0h	Disable.										
1h	SMI.										
2h	NMI.										
3h	IRQ13.										
29:28	<b>Smicontrol14.</b> Read-write. Reset: 0h. Control for GEVENT14. <b>ValidValues:</b> <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>Disable.</td></tr> <tr> <td>1h</td><td>SMI.</td></tr> <tr> <td>2h</td><td>NMI.</td></tr> <tr> <td>3h</td><td>IRQ13.</td></tr> </table>	Value	Description	0h	Disable.	1h	SMI.	2h	NMI.	3h	IRQ13.
Value	Description										
0h	Disable.										
1h	SMI.										
2h	NMI.										
3h	IRQ13.										
27:26	<b>Smicontrol13.</b> Read-write. Reset: 0h. Control for GEVENT13. <b>ValidValues:</b> <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>Disable.</td></tr> <tr> <td>1h</td><td>SMI.</td></tr> <tr> <td>2h</td><td>NMI.</td></tr> <tr> <td>3h</td><td>IRQ13.</td></tr> </table>	Value	Description	0h	Disable.	1h	SMI.	2h	NMI.	3h	IRQ13.
Value	Description										
0h	Disable.										
1h	SMI.										
2h	NMI.										
3h	IRQ13.										
25:24	<b>Smicontrol12.</b> Read-write. Reset: 0h. Control for GEVENT12. <b>ValidValues:</b> <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>Disable.</td></tr> <tr> <td>1h</td><td>SMI.</td></tr> <tr> <td>2h</td><td>NMI.</td></tr> <tr> <td>3h</td><td>IRQ13.</td></tr> </table>	Value	Description	0h	Disable.	1h	SMI.	2h	NMI.	3h	IRQ13.
Value	Description										
0h	Disable.										
1h	SMI.										
2h	NMI.										
3h	IRQ13.										
23:22	<b>Smicontrol11.</b> Read-write. Reset: 0h. Control for GEVENT11. <b>ValidValues:</b> <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>Disable.</td></tr> <tr> <td>1h</td><td>SMI.</td></tr> <tr> <td>2h</td><td>NMI.</td></tr> <tr> <td>3h</td><td>IRQ13.</td></tr> </table>	Value	Description	0h	Disable.	1h	SMI.	2h	NMI.	3h	IRQ13.
Value	Description										
0h	Disable.										
1h	SMI.										
2h	NMI.										
3h	IRQ13.										
21:20	<b>Smicontrol10.</b> Read-write. Reset: 0h. Control for GEVENT10. <b>ValidValues:</b> <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>Disable.</td></tr> <tr> <td>1h</td><td>SMI.</td></tr> <tr> <td>2h</td><td>NMI.</td></tr> <tr> <td>3h</td><td>IRQ13.</td></tr> </table>	Value	Description	0h	Disable.	1h	SMI.	2h	NMI.	3h	IRQ13.
Value	Description										
0h	Disable.										
1h	SMI.										
2h	NMI.										
3h	IRQ13.										
19:18	<b>Smicontrol9.</b> Read-write. Reset: 0h. Control for GEVENT9.										



	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	Disable.
	1h	SMI.
	2h	NMI.
17:16	3h	IRQ13.
	<b>Smicontrol8.</b> Read-write. Reset: 0h. Control for GEVENT8.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	Disable.
15:14	1h	SMI.
	2h	NMI.
	3h	IRQ13.
	<b>Smicontrol7.</b> Read-write. Reset: 0h. Control for GEVENT7.	
	<b>ValidValues:</b>	
13:12	<b>Value</b>	<b>Description</b>
	0h	Disable.
	1h	SMI.
	2h	NMI.
	3h	IRQ13.
11:10	<b>Smicontrol6.</b> Read-write. Reset: 0h. Control for GEVENT6.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	Disable.
	1h	SMI.
9:8	2h	NMI.
	3h	IRQ13.
	<b>Smicontrol5.</b> Read-write. Reset: 0h. Control for GEVENT5.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
7:6	0h	Disable.
	1h	SMI.
	2h	NMI.
	3h	IRQ13.
	<b>Smicontrol4.</b> Read-write. Reset: 0h. Control for GEVENT4.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	Disable.
	1h	SMI.
	2h	NMI.
	3h	IRQ13.
	<b>Smicontrol3.</b> Read-write. Reset: 0h. Control for GEVENT3.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	Disable.
	1h	SMI.
	2h	NMI.
	3h	IRQ13.



5:4	<b>Smicontrol2.</b> Read-write. Reset: 0h. Control for GEVENT2.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	Disable.
	1h	SMI.
3:2	2h	NMI.
	3h	IRQ13.
	<b>Smicontrol1.</b> Read-write. Reset: 0h. Control for GEVENT1.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
1:0	0h	Disable.
	1h	SMI.
	2h	NMI.
	3h	IRQ13.
	<b>Smicontrol0.</b> Read-write. Reset: 0h. Control for GEVENT0.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	Disable.
	1h	SMI.
	2h	NMI.
	3h	IRQ13.

**SMIx0A4 [SmiControl1] (FCH::SMI::SmiCtl1)**

Read-write. Reset: 0000\_0000h.

This register specifies the control mechanism for SMI sources in FCH::SMI::SmiStat0[31:16].

\_aliasHOST; SMIx0A4; SMI=FED8\_0200h

Bits	Description
31:30	Reserved.
29:28	<b>Smicontrol30.</b> Read-write. Reset: 0h. Control for NB GPP Hot Plug.
	<b>ValidValues:</b>
	<b>Value</b>
	0h
	1h
27:26	2h
	3h
	<b>Smicontrol29.</b> Read-write. Reset: 0h. Control for GPP PME.
	<b>ValidValues:</b>
	<b>Value</b>
25:24	0h
	1h
	2h
	3h
	<b>Smicontrol27.</b> Read-write. Reset: 0h. Control for eSPI_WAKE_PME.
23:22	<b>ValidValues:</b>
	<b>Value</b>
	0h
	1h
	2h

	3h	IRQ13.
21:20	<b>Smicontrol26.</b> Read-write. Reset: 0h. Control for eSPI system event.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	Disable.
	1h	SMI.
	2h	NMI.
	3h	IRQ13.
19:16	Reserved.	
15:14	<b>Smicontrol23.</b> Read-write. Reset: 0h. Control for GEVENT23.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	Disable.
	1h	SMI.
	2h	NMI.
	3h	IRQ13.
13:12	<b>Smicontrol22.</b> Read-write. Reset: 0h. Control for GEVENT22.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	Disable.
	1h	SMI.
	2h	NMI.
	3h	IRQ13.
11:10	<b>Smicontrol21.</b> Read-write. Reset: 0h. Control for GEVENT21.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	Disable.
	1h	SMI.
	2h	NMI.
	3h	IRQ13.
9:8	<b>Smicontrol20.</b> Read-write. Reset: 0h. Control for GEVENT20.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	Disable.
	1h	SMI.
	2h	NMI.
	3h	IRQ13.
7:6	<b>Smicontrol19.</b> Read-write. Reset: 0h. Control for GEVENT19.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	Disable.
	1h	SMI.
	2h	NMI.
	3h	IRQ13.
5:4	<b>Smicontrol18.</b> Read-write. Reset: 0h. Control for GEVENT18.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	Disable.

	1h	SMI.
	2h	NMI.
	3h	IRQ13.
3:2	<b>Smicontrol17.</b> Read-write. Reset: 0h. Control for GEVENT17.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	Disable.
	1h	SMI.
	2h	NMI.
	3h	IRQ13.
1:0	<b>Smicontrol16.</b> Read-write. Reset: 0h. Control for GEVENT16.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	Disable.
	1h	SMI.
	2h	NMI.
	3h	IRQ13.

**SMIx0A8 [SmiControl2] (FCH::SMI::SmiCtl2)**

Read-write. Reset: 0000\_0000h.

This register specifies the control mechanism for SMI sources in FCH::SMI::SmiStat1[15:0].

\_aliasHOST; SMIx0A8; SMI=FED8\_0200h

Bits	Description
31:30	<b>Smicontrol47.</b> Read-write. Reset: 0h. Control for SMBUS0 interrupt.
	<b>ValidValues:</b>
	<b>Value</b> <b>Description</b>
	0h Disable.
	1h SMI.
	2h NMI.
	3h IRQ13.
29:28	<b>Smicontrol46.</b> Read-write. Reset: 0h. Control for ACP_FCH_I2S_Wake.
	<b>ValidValues:</b>
	<b>Value</b> <b>Description</b>
	0h Disable.
	1h SMI.
	2h NMI.
	3h IRQ13.
27:26	<b>Smicontrol45.</b> Read-write. Reset: 0h. Control for ASF Master and Slave interrupt.
	<b>ValidValues:</b>
	<b>Value</b> <b>Description</b>
	0h Disable.
	1h SMI.
	2h NMI.
	3h IRQ13.
25:24	<b>Smicontrol44.</b> Read-write. Reset: 0h. Control for fan thermal GEvent.
	<b>ValidValues:</b>
	<b>Value</b> <b>Description</b>
	0h Disable.
	1h SMI.

	2h	NMI.
	3h	IRQ13.
23:22	<b>SimControl43.</b> Read-write. Reset: 0h. Control for ALTHPET_TimerSts.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	Disable.
	1h	SMI.
	2h	NMI.
	3h	IRQ13.
21:20	Reserved.	
19:18	<b>SimControl41.</b> Read-write. Reset: 0h. Control for GPIO interrupt.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	Disable.
	1h	SMI.
	2h	NMI.
	3h	IRQ13.
17:16	<b>Smicontrol40.</b> Read-write. Reset: 0h. Control for USB_PD_I2C4_intrB.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	Disable.
	1h	SMI.
	2h	NMI.
	3h	IRQ13.
15:14	<b>Smicontrol39.</b> Read-write. Reset: 0h. Control for Azalia PME.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	Disable.
	1h	SMI.
	2h	NMI.
	3h	IRQ13.
13:2	Reserved.	
1:0	<b>Smicontrol32.</b> Read-write. Reset: 0h. Control for WAKE_L.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	Disable.
	1h	SMI.
	2h	NMI.
	3h	IRQ13.

**SMIx0AC [SmiControl3] (FCH::SMI::SmiCtl3)**

Read-write. Reset: 0000\_0000h.

This register specifies the control mechanism for SMI sources in FCH::SMI::SmiStat1[31:16].

\_aliasHOST; SMIx0AC; SMI=FED8\_0200h

Bits	Description
31:30	<b>Smicontrol63.</b> Read-write. Reset: 0h. Control for TempTsi event.
	<b>ValidValues:</b>
	<b>Value</b>
	0h

	1h	SMI.
	2h	NMI.
	3h	IRQ13.
29:22	Reserved.	
21:20	<b>Smicontrol58.</b> Read-write. Reset: 0h. Control for AcDcTimer wake up event (Wake Device in ACPI 4.0).	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	Disable.
	1h	SMI.
	2h	NMI.
	3h	IRQ13.
19:18	<b>Smicontrol57.</b> Read-write. Reset: 0h. Control for XHC1 PME.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	Disable.
	1h	SMI.
	2h	NMI.
	3h	IRQ13.
17:16	<b>Smicontrol56.</b> Read-write. Reset: 0h. Control for XHC0 PME.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	Disable.
	1h	SMI.
	2h	NMI.
	3h	IRQ13.
15:14	<b>Smicontrol55.</b> Read-write. Reset: 0h. Control for internal devices SERR error status.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	Disable.
	1h	SMI.
	2h	NMI.
	3h	IRQ13.
13:12	<b>Smicontrol54.</b> Read-write. Reset: 0h. Control for APU SCI request.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	Disable.
	1h	SMI.
	2h	NMI.
	3h	IRQ13.
11:10	<b>Smicontrol53.</b> Read-write. Reset: 0h. Control for APU hardware assertion.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	Disable.
	1h	SMI.
	2h	NMI.
	3h	IRQ13.
9:8	<b>Smicontrol52.</b> Read-write. Reset: 0h. Control for ProcHot event.	
	<b>ValidValues:</b>	

	<b>Value</b>	<b>Description</b>
	0h	Disable.
	1h	SMI.
	2h	NMI.
	3h	IRQ13.
7:6	<b>Smicontrol51.</b> Read-write. Reset: 0h. Control for iLLB#.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	Disable.
	1h	SMI.
	2h	NMI.
	3h	IRQ13.
5:4	<b>Smicontrol50.</b> Read-write. Reset: 0h. Control for Power Button event.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	Disable.
	1h	SMI.
	2h	NMI.
	3h	IRQ13.
3:2	<b>Smicontrol49.</b> Read-write. Reset: 0h. Control for internal Traffic monitor interrupt.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	Disable.
	1h	SMI.
	2h	NMI.
	3h	IRQ13.
1:0	<b>Smicontrol48.</b> Read-write. Reset: 0h. Control for TWARN#.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	Disable.
	1h	SMI.
	2h	NMI.
	3h	IRQ13.

**SMIx0B0 [SmiControl4] (FCH::SMI::SmiCtl4)**

Read-write. Reset: 0000\_0000h.

This register specifies the control mechanism for SMI sources in FCH::SMI::SmiStat2[15:0].

\_aliasHOST; SMIx0B0; SMI=FED8\_0200h

Bits	Description	
31:30	Smicontrol79. Read-write. Reset: 0h. Control for EC SMI request0.	
	ValidValues:	
	Value	Description
	0h	Disable.
	1h	SMI.
	2h	NMI.
	3h	IRQ13.
29:28	Smicontrol78. Read-write. Reset: 0h. Control for SMBUS0 interrupt.	
	ValidValues:	
	Value	Description

	0h	Disable.
	1h	SMI.
	2h	NMI.
	3h	IRQ13.
27:26	<b>Smicontrol77.</b> Read-write. Reset: 0h. Control for SMI request form serial IRQ.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	Disable.
	1h	SMI.
	2h	NMI.
	3h	IRQ13.
25:24	<b>Smicontrol76.</b> Read-write. Reset: 0h. Control for USB SMI request.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	Disable.
	1h	SMI.
	2h	NMI.
	3h	IRQ13.
23:22	<b>Smicontrol75.</b> Read-write. Reset: 0h. Control for Writing SMI command port at FCH::PM::SmiCmdPort.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	Disable.
	1h	SMI.
	2h	NMI.
	3h	IRQ13.
21:20	<b>Smicontrol74.</b> Read-write. Reset: 0h. Control for Bower Button being pressed.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	Disable.
	1h	SMI.
	2h	NMI.
	3h	IRQ13.
19:18	<b>Smicontrol73.</b> Read-write. Reset: 0h. Control for Writing FCH::PM::AcpiCfg[BiosRls].	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	Disable.
	1h	SMI.
	2h	NMI.
	3h	IRQ13.
17:16	<b>Smicontrol72.</b> Read-write. Reset: 0h. Control for Writing FCH::PM::PmCtl[GblRls].	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	Disable.
	1h	SMI.
	2h	NMI.
	3h	IRQ13.
15:14	<b>Smicontrol71.</b> Read-write. Reset: 0h. Control for USB_PD_I2C4_intrB event.	
	<b>ValidValues:</b>	



	<b>Value</b>	<b>Description</b>
	0h	Disable.
	1h	SMI.
	2h	NMI.
	3h	IRQ13.
13:12	<b>Smicontrol70.</b> Read-write. Reset: 0h. Control for USB_PD_I2C4_intrB event.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	Disable.
	1h	SMI.
	2h	NMI.
	3h	IRQ13.
11:10	<b>Smicontrol69.</b> Read-write. Reset: 0h. Control for NBGppHpPulse event.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	Disable.
	1h	SMI.
	2h	NMI.
	3h	IRQ13.
9:8	<b>Smicontrol68.</b> Read-write. Reset: 0h. Control for NBGppPmePulse event.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	Disable.
	1h	SMI.
	2h	NMI.
	3h	IRQ13.
7:6	<b>Smicontrol67.</b> Read-write. Reset: 0h. Control for SATA AHCI event.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	Disable.
	1h	SMI.
	2h	NMI.
	3h	IRQ13.
5:4	<b>Smicontrol66.</b> Read-write. Reset: 0h. Control for iAL2H ACPI Assertion.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	Disable.
	1h	SMI.
	2h	NMI.
	3h	IRQ13.
3:2	<b>Smicontrol65.</b> Read-write. Reset: 0h. Control for Writing FCH::PM::PmCtl[SlpTyp] to put the system in S-state.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	Disable.
	1h	SMI.
	2h	NMI.
	3h	IRQ13.
1:0	<b>Smicontrol64.</b> Read-write. Reset: 0h. Control for Keyboard Reset.	

	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	Disable.
	1h	SMI.
	2h	NMI.
	3h	IRQ13.

**SMIx0B4 [SmiControl5] (FCH::SMI::SmiCtl5)**

Read-write. Reset: 0000\_0000h.

This register specifies the control mechanism for SMI sources in FCH::SMI::SmiStat2[31:16].

\_aliasHOST; SMIx0B4; SMI=FED8\_0200h

Bits	Description
31:22	Reserved.
21:20	<b>Smicontrol90.</b> Read-write. Reset: 0h. Control for Emulation64.
	<b>ValidValues:</b>
	<b>Value</b>
	<b>Description</b>
	0h
	Disable.
	1h
	SMI.
	2h
	NMI.
	3h
	IRQ13.
19:18	<b>Smicontrol89.</b> Read-write. Reset: 0h. Control for ThermalTrip# assertion.
	<b>ValidValues:</b>
	<b>Value</b>
	<b>Description</b>
	0h
	Disable.
	1h
	SMI.
	2h
	NMI.
	3h
	IRQ13.
17:10	Reserved.
9:8	<b>Smicontrol84.</b> Read-write. Reset: 0h. Control for SERR#.
	<b>ValidValues:</b>
	<b>Value</b>
	<b>Description</b>
	0h
	Disable.
	1h
	SMI.
	2h
	NMI.
	3h
	IRQ13.
7:6	<b>Smicontrol83.</b> Read-write. Reset: 0h. Control for ProcHot.
	<b>ValidValues:</b>
	<b>Value</b>
	<b>Description</b>
	0h
	Disable.
	1h
	SMI.
	2h
	NMI.
	3h
	IRQ13.
5:4	<b>Smicontrol82.</b> Read-write. Reset: 0h. Control for VBAT low.
	<b>ValidValues:</b>
	<b>Value</b>
	<b>Description</b>
	0h
	Disable.
	1h
	SMI.
	2h
	NMI.
	3h
	IRQ13.

3:2	<b>Smicontrol81.</b> Read-write. Reset: 0h. Control for Intruder event.
<b>ValidValues:</b>	
<b>Value</b>	<b>Description</b>
0h	Disable.
1h	SMI.
2h	NMI.
3h	IRQ13.
1:0	Reserved.

**SMIx0C0 [SmiControl8] (FCH::SMI::SmiCtl8)**

Read-write. Reset: 0000\_0000h.

This register specifies the control mechanism for SMI sources in FCH::SMI::SmiStat4[15:0].

\_aliasHOST; SMIx0C0; SMI=FED8\_0200h

<b>Bits</b>	<b>Description</b>
31:30	<b>Smicontrol143.</b> Read-write. Reset: 0h. Control for Long Timer.
<b>ValidValues:</b>	
<b>Value</b>	<b>Description</b>
0h	Disable.
1h	SMI.
2h	NMI.
3h	IRQ 13.
29:28	<b>Smicontrol142.</b> Read-write. Reset: 0h. Control for Short Timer.
<b>ValidValues:</b>	
<b>Value</b>	<b>Description</b>
0h	Disable.
1h	SMI.
2h	NMI.
3h	IRQ 13.
27:26	<b>Smicontrol141.</b> Read-write. Reset: 0h. Control for CF9 IO Write.
<b>ValidValues:</b>	
<b>Value</b>	<b>Description</b>
0h	Disable.
1h	SMI.
2h	NMI.
3h	IRQ 13.
25:12	Reserved.
11:10	<b>Smicontrol133.</b> Read-write. Reset: 0h. Control for Fan0 Tach 0 too slow event.
<b>ValidValues:</b>	
<b>Value</b>	<b>Description</b>
0h	Disable.
1h	SMI.
2h	NMI.
3h	IRQ 13.
9:0	Reserved.

**SMIx0C4 [SmiControl9] (FCH::SMI::SmiCtl9)**

Read-write. Reset: 0000\_0000h.

This register specifies the control mechanism for SMI sources in FCH::SMI::SmiStat4[31:16].

\_aliasHOST; SMIx0C4; SMI=FED8\_0200h

Bits	Description										
31:26	Reserved.										
25:24	<b>Smicontrol156.</b> Read-write. Reset: 0h. Control for configuration cycle trapping 0.										
	<b>ValidValues:</b>										
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>Disable.</td></tr> <tr> <td>1h</td><td>SMI.</td></tr> <tr> <td>2h</td><td>NMI.</td></tr> <tr> <td>3h</td><td>IRQ13.</td></tr> </table>	Value	Description	0h	Disable.	1h	SMI.	2h	NMI.	3h	IRQ13.
Value	Description										
0h	Disable.										
1h	SMI.										
2h	NMI.										
3h	IRQ13.										
23:18	Reserved.										
17:16	<b>Smicontrol152.</b> Read-write. Reset: 0h. Control for memory trapping 0.										
	<b>ValidValues:</b>										
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>Disable.</td></tr> <tr> <td>1h</td><td>SMI.</td></tr> <tr> <td>2h</td><td>NMI.</td></tr> <tr> <td>3h</td><td>IRQ13.</td></tr> </table>	Value	Description	0h	Disable.	1h	SMI.	2h	NMI.	3h	IRQ13.
Value	Description										
0h	Disable.										
1h	SMI.										
2h	NMI.										
3h	IRQ13.										
15:14	<b>Smicontrol151.</b> Read-write. Reset: 0h. Control for IO trapping 3.										
	<b>ValidValues:</b>										
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>Disable.</td></tr> <tr> <td>1h</td><td>SMI.</td></tr> <tr> <td>2h</td><td>NMI.</td></tr> <tr> <td>3h</td><td>IRQ13.</td></tr> </table>	Value	Description	0h	Disable.	1h	SMI.	2h	NMI.	3h	IRQ13.
Value	Description										
0h	Disable.										
1h	SMI.										
2h	NMI.										
3h	IRQ13.										
13:12	<b>Smicontrol150.</b> Read-write. Reset: 0h. Control for IO trapping 2.										
	<b>ValidValues:</b>										
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>Disable.</td></tr> <tr> <td>1h</td><td>SMI.</td></tr> <tr> <td>2h</td><td>NMI.</td></tr> <tr> <td>3h</td><td>IRQ13.</td></tr> </table>	Value	Description	0h	Disable.	1h	SMI.	2h	NMI.	3h	IRQ13.
Value	Description										
0h	Disable.										
1h	SMI.										
2h	NMI.										
3h	IRQ13.										
11:10	<b>Smicontrol149.</b> Read-write. Reset: 0h. Control for IO trapping 1.										
	<b>ValidValues:</b>										
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>Disable.</td></tr> <tr> <td>1h</td><td>SMI.</td></tr> <tr> <td>2h</td><td>NMI.</td></tr> <tr> <td>3h</td><td>IRQ13.</td></tr> </table>	Value	Description	0h	Disable.	1h	SMI.	2h	NMI.	3h	IRQ13.
Value	Description										
0h	Disable.										
1h	SMI.										
2h	NMI.										
3h	IRQ13.										
9:8	<b>Smicontrol148.</b> Read-write. Reset: 0h. Control for IO trapping 0.										
	<b>ValidValues:</b>										
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>Disable.</td></tr> <tr> <td>1h</td><td>SMI.</td></tr> <tr> <td>2h</td><td>NMI.</td></tr> <tr> <td>3h</td><td>IRQ13.</td></tr> </table>	Value	Description	0h	Disable.	1h	SMI.	2h	NMI.	3h	IRQ13.
Value	Description										
0h	Disable.										
1h	SMI.										
2h	NMI.										
3h	IRQ13.										
7:6	Reserved.										
5:4	<b>Smicontrol146.</b> Read-write. Reset: 0h. Control for ESPI_SMI_B.										
	<b>ValidValues:</b>										

	Value	Description
	0h	Disable.
	1h	SMI.
	2h	NMI.
	3h	IRQ13.
3:2	Reserved.	
1:0	<b>Smicontrol144.</b> Read-write. Reset: 0h. Control for AB SMI trapping request.	
	<b>Valid Values:</b>	
	Value	Description
	0h	Disable.
	1h	SMI.
	2h	NMI.
	3h	IRQ13.

### 9.2.5 High Precision Event Timer (HPET) Registers

HPET registers are accessed through two methods:

- Memory access to HPET memory address range from FED0\_0000h to FED0\_01FFh. Program PMx00[HpetEn] = 1 to enable HPET decoding.
- Memory mapped access through the AcpiMmio region. The HPET registers range from FED8\_0000h+C00h to FED8\_0000h+CFFh. See PMx04[MmioEn].

#### HPETx000 [ID] (FCH::TMR::HPET::HpetId)

Read-only. Reset: 1022\_8201h.

\_aliasHOST; HPETx000; HPET=FED0\_0000h

Bits	Description
31:16	<b>VendorID.</b> Read-only. Reset: 1022h. Vendor ID.
15	<b>LegacyCap.</b> Read-only. Reset: 1. Legacy replacement interrupt is supported.
14	Reserved.
13	<b>CounterSizeCap.</b> Read-only. Reset: 0. 0=32-bits wide. 1=64-bits wide.
12:8	<b>NumTmrCap.</b> Read-only. Reset: 02h. Three timers are supported.
7:0	<b>RevID.</b> Read-only. Reset: 01h. Revision ID.

#### HPETx004 [ClkPeriod] (FCH::TMR::HPET::ClkPeriod)

Read-only. Reset: 0429\_B17Eh.

\_aliasHOST; HPETx004; HPET=FED0\_0000h

Bits	Description
31:0	<b>CounterClkPeriod.</b> Read-only. Reset: 0429_B17Eh. Specifies the clock period of each HPET timer tick. HPET main counter runs at 14.31818 MHz. The unit is femtosecond ( $10^{-15}$ seconds). The value of this register can be modified through FCH::TMR::HPET::ClkPeriod.

#### HPETx010 [Config] (FCH::TMR::HPET::Cfg)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; HPETx010; HPET=FED0\_0000h

Bits	Description
31:2	Reserved.
1	<b>LegacyEn.</b> Read-write. Reset: 0. 1=Timer0 interrupt goes to IRQ0 of PIC controller, INT2 of IOAPIC; Timer1 interrupt goes to IRQ8 of PIC controller, INT8 of IOAPIC.
0	<b>TmrEn.</b> Read-write. Reset: 0. 0=Pause main counter and disable all timer interrupts. 1=Allow main counter to run and allow timer interrupts if enabled.

**HPETx020 [Interrupt Status] (FCH::TMR::HPET::IntStat)**

Reset: 0000\_0000h.

\_aliasHOST; HPETx020; HPET=FED0\_0000h

**Bits Description**

31:0 Reserved.

**HPETx0F0 [Main Counter] (FCH::TMR::HPET::MainCtr)**

Read-write. Reset: 0000\_0000\_0000\_0000h.

\_aliasHOST; HPETx0F0; HPET=FED0\_0000h

**Bits Description**

63:32 **MainCounterHi.** Read-write. Reset: 0000\_0000h. IF (FCH::PM::PmDecodeEn[HpetWidthSel] == 1) THEN Specifies the upper 32 bits of the 64-bit HPET main counter. Bits [31:0] contain the lower 32 bits of the 64-bit HPET main counter. Should be written to only when it is halted. ELSE Reserved. ENDIF.

31:0 **MainCounterLo.** Read-write. Reset: 0000\_0000h. Specified the HPET main counter, incremented by 1 on every clock. Should be written to only when it is halted.

**HPETx1[0...4]0 [Timer[2:0] Config Capability] (FCH::TMR::HPET::TimerCfgCap)**

Reset: 00C0\_0000\_0000\_8010h.

\_n[2:0]\_aliasHOST; HPETx1[4,2,0]0; HPET=FED0\_0000h

**Bits Description**

63:32 **TmrIntRouteCap.** Read-only. Reset: 00C0\_0000h. Indicates which INT entry of IOAPIC can be assigned to the timer interrupt.

31:16 Reserved.

15 **TmrFsbCap.** Read-only. Reset: 1. 1=Front side bus delivery is supported.

14 **TmrFsbEn.** Read-write. Reset: 0. 1=Enable front side bus delivery of interrupt.

13:9 **TmrIntRoute.** Read-write. Reset: 00h. This specifies which INT entry of IOAPIC the timer is routed to if FCH::TMR::HPET::Cfg[LegacyEn] == 0.

8 **Tmr32ModeEn.** Read-only. Reset: 0. 0=64-bit timer is not supported. 1=64-bit timer is supported.

7 Reserved.

6 **TmrSetPer.** Read-write. Reset: 0. 1=Allow software to set the timer's accumulator if the timer is set to periodic mode. The bit is automatically cleared when FCH::TMR::HPET::TimerComparator is written by software.

5 **TmrSizeCap.** Read-only. Reset: 0. The timer is 32-bits wide.

4 **TmrTypCap.** Read-only. Reset: 1. The timer supports periodic interrupt delivery mode.

3 **TmrTyp.** Read-write. Reset: 0. 0=Non-periodic. 1=Periodic. Selects the timer interrupt type.

2 **TmrIntEn.** Read-write. Reset: 0. 1=Enable the timer interrupt.

1 **TmrIntTyp.** Read-write. Reset: 0. 0=Edge triggered. 1=Level triggered. Specifies the timer interrupt polarity.

0 Reserved.

**HPETx1[0...4]8 [Timer[2:0] Comparator] (FCH::TMR::HPET::TimerComparator)**

Read-write. Reset: 0000\_0000\_FFFF\_FFFFh.

\_n[2:0]\_aliasHOST; HPETx1[4,2,0]8; HPET=FED0\_0000h

**Bits Description**

63:32 **ComparatorHi.** Read-write. Reset: 0000\_0000h. IF (FCH::PM::PmDecodeEn[HpetWidthSel] == 1) THEN ELSE ENDIF. Specifies the upper 32 bits of the 64-bit timer comparator. Bits [31:0] contain the lower 32 bits of the 64-bit timer comparator. ELSE Reserved. ENDIF.

31:0 **ComparatorLo.** Read-write. Reset: FFFF\_FFFFh. IF (FCH::TMR::HPET::TimerCfgCap[TmrTyp] == 0) THEN Read-write. ELSEIF ((FCH::TMR::HPET::TimerCfgCap[TmrTyp] == 1) && (FCH::TMR::HPET::TimerCfgCap[TmrSetPer] == 1)) THEN Read-write. ELSE Read-only. ENDIF. The comparator is periodically incremented by the value last written to this register. This register is updated by hardware when (FCH::TMR::HPET::TimerCfgCap[TmrTyp] == 1).

**HPETx1[1...5]0 [Timer[2:0] FSB Interrupt Data] (FCH::TMR::HPET::TimerFSBIntData)**

Read-write. Reset: 0000_0000h.	
_n[2:0]_aliasHOST; HPETx1[5,3,1]0; HPET=FED0_0000h	
Bits	Description
31:0	<b>TnFsbIntVal.</b> Read-write. Reset: 0000_0000h. Software sets this 32-bit field to specify the write data of Front Side Bus (FSB) Interrupt Message.

HPETx1[1...5]4 [Timer[2:0] FSB Interrupt Address] (FCH::TMR::HPET::TimerFSBIntAddr)	
Read-write. Reset: 0000_0000h.	
_n[2:0]_aliasHOST; HPETx1[5,3,1]4; HPET=FED0_0000h	
Bits	Description
31:0	<b>TnFsbIntAddr.</b> Read-write. Reset: 0000_0000h. Software sets this 32-bit field to specify the address of Front Side Bus (FSB) Interrupt Message.

HPETx1[B...D]0 [Timer[2:0] Comparator Base Shadow] (FCH::TMR::HPET::TimerComparatorBaseShadow)	
Read-write. Reset: 0000_0000_FFFF_FFFFh.	
_n[2:0]_aliasHOST; HPETx1[D:B]0; HPET=FED0_0000h	
Bits	Description
63:32	<b>TmrCompBaseShadowHi.</b> Read-write. Reset: 0000_0000h. IF (FCH::PM::PmDecodeEn[HpetWidthSel] == 1) THEN ELSE ENDIF. This is a shadow of the base value of FCH::TMR::HPET::TimerComparator_n[2:0][ComparatorHi]. Reading this register returns the base value of FCH::TMR::HPET::TimerComparator_n[2:0][ComparatorHi]. Writing the register changes the base value of FCH::TMR::HPET::TimerComparator_n[2:0][ComparatorHi].
31:0	<b>TmrCompBaseShadowLo.</b> Read-write. Reset: FFFF_FFFFh. This is shadow of the base value of FCH::TMR::HPET::TimerComparator_n[2:0][ComparatorLo]. Reading this register returns the base value of FCH::TMR::HPET::TimerComparator_n[2:0][ComparatorLo]. Writing the register changes the base value of FCH::TMR::HPET::TimerComparator_n[2:0][ComparatorLo].

HPETx1[B...D]8 [Timer[2:0] Comparator Shadow] (FCH::TMR::HPET::TimerComparatorShadow)	
Read-write. Reset: 0000_0000_FFFF_FFFFh.	
_n[2:0]_aliasHOST; HPETx1[D:B]8; HPET=FED0_0000h	
Bits	Description
63:32	<b>TmrCompShadowHi.</b> Read-write. Reset: 0000_0000h. IF (FCH::PM::PmDecodeEn[HpetWidthSel] == 1) THEN ELSE ENDIF. This is a shadow of the current value of FCH::TMR::HPET::TimerComparator_n[2:0][ComparatorHi]. Reading this register returns the current value of FCH::TMR::HPET::TimerComparator_n[2:0][ComparatorHi]. Writing the register changes the current value of FCH::TMR::HPET::TimerComparator_n[2:0][ComparatorHi].
31:0	<b>TmrCompShadowLo.</b> Read-write. Reset: FFFF_FFFFh. This is shadow of the current value of FCH::TMR::HPET::TimerComparator_n[2:0][ComparatorLo]. Reading this register returns the current value of FCH::TMR::HPET::TimerComparator_n[2:0][ComparatorLo]. Writing the register changes the current value of FCH::TMR::HPET::TimerComparator_n[2:0][ComparatorLo].

HPETx1E0 [Main Counter RTC] (FCH::TMR::HPET::MainCtrRTC)	
Read-write. Reset: 0000_0000_0000_0000h.	
_aliasHOST; HPETx1E0; HPET=FED0_0000h	
Bits	Description
63:32	<b>MainCounterRtcHi.</b> Read-write. Reset: 0000_0000h. IF (FCH::PM::PmDecodeEn[HpetWidthSel] == 1) THEN Specifies the shadow of FCH::TMR::HPET::MainCtr[MainCounterHi]. See MainCounterRtcLo. ELSE Reserved. ENDIF.
31:0	<b>MainCounterRtcLo.</b> Read-write. Reset: 0000_0000h. This is a shadow of FCH::TMR::HPET::MainCtr[MainCounterLo]. It samples the value of Main Counter at every falling edge of RTC 32 KHz clock for software to Read. When software Writes this register, FCH::TMR::HPET::MainCtr[MainCounterLo] is updated with the same value written to this register, and then enabled counting at the next RTC clock falling edge. The purpose of this register is for the convenience of



	software save-restore HPET.
<b>HPETx1E8 [Next Timer Remain] (FCH::TMR::HPET::NextTimerRemain)</b>	
Reset: 0000_0000_FFFF_FFFFh.	
_aliasHOST; HPETx1E8; HPET=FED0_0000h	
Bits	Description
63:32	<b>NxtTmrRemainHi.</b> Read-write. Reset: 0000_0000h. IF (FCH::PM::PmDecodeEn[HpetWidthSel] == 1) THEN Specifies the upper 32 bits of the timer ticks remaining before the next enabled comparator interrupt. ELSE Reserved. ENDIF.
31:0	<b>NxtTmrRemainLo.</b> Read-only. Reset: FFFF_FFFFh. Specifies how many timer ticks remaining before the next enabled comparator interrupt.

## 9.2.6 Watchdog Timer (WDT) Registers

Watchdog timer registers are accessed through two methods:

- Memory access to Watchdog Timer memory address range from FEB0\_0000h to FEB0\_000Fh. Program FCH::PM::PmDecodeEn[WatchdogTmrEn] = 1 to enable Watchdog Timer decoding.
- Memory mapped access through the AcpiMmio region. The Watchdog Timer registers start from FED8\_0000h+B00h. See FCH::PM::IsaControl[MmioEn].

<b>WDTx000 [WatchdogControl] (FCH::TMR::WDT::WatchdogCtl)</b>	
_aliasHOST; WDTx000; WDT=FED8_0B00h	
Bits	Description
31:8	Reserved.
7	<b>WatchdogTrigger.</b> Write-1-only. Reset: 0. Writing 1 to this bit triggers the watchdog to start a new count interval, counting down from the value that was last written to FCH::TMR::WDT::WatchdogCnt.
6:4	Reserved.
3	<b>WatchdogDisable.</b> Read-only. Reset: 1. 0=Enable. 1=Disable. This bit reflects the state of the watchdog timer hardware. This bit reflects the state of FCH::PM::PmDecodeEn[WatchdogTmrEn].
2	<b>WatchdogAction.</b> Read-write. Reset: 0. 0=System reset. 1=System power off. This bit determines the action to be taken when the watchdog timer expires. The bit is only valid when the watchdog is enabled.
1	<b>WatchdogFired.</b> Read,Write-1-to-clear. Reset: Cold,0. 1=The watchdog timer has expired and caused the current restart. The bit is cleared by a power cycle or by the operating system and it must remain cleared for any restart that is not caused by the watchdog timer firing. The bit is only valid when the watchdog is enabled.
0	<b>WatchdogRunStopB.</b> Read-write. Reset: 0. 0=Watchdog is in the stopped state. 1=Watchdog is in the running state. This bit is used to control or indicate whether the watchdog is in the running or stopped states. If the watchdog is in the stopped state and a 1 is written to this bit, the watchdog moves to the running state, but a count interval is not started until a 1 is written to WatchdogTrigger. If the watchdog is in the running state, writing 1 to this bit has no effect. The bit is only valid when the watchdog is enabled.

<b>WDTx004 [WatchdogCount] (FCH::TMR::WDT::WatchdogCnt)</b>	
Read-write.	
_aliasHOST; WDTx004; WDT=FED8_0B00h	
Bits	Description
31:16	Reserved.
15:0	<b>WatchdogCount.</b> Read-write. Reset: XXXXh. Writing this field specifies the countdown time for the counter.

## 9.2.7 Wake Alarm Device (AcDcTimer) Registers

In Family 17h Models 10-1Fh, signal iBatteryModeB is connected to pad AGPIO23 that can be toggled to use ac\_timer or dc\_timer.

The AC/DC timer registers are used to control the wake alarm device. They are accessed through the AcpiMmio region. The AC DC timer registers range from FED8\_0000h+1D00h to FED8\_0000h+1DFFh.

<b>ACDCx000 [AcTimerValue] (FCH::TMR::ACDC::AcTimerVal)</b>	
Read-write. Reset: FFFF_FFFFh.	
_aliasHOST; ACDCx000; ACDC=FED8_1D00h	
Bits	Description
31:0	<b>AcTimerValue.</b> Read-write. Reset: FFFF_FFFFh. <b>Description:</b> Writing the register with a value other than FFFF_FFFFh starts the AC timer. Reading this register returns the current value of the AC timer. When the AC or DC timer generates a wake up event, this register is reset to FFFF_FFFFh by hardware.
<b>ACDCx004 [AcExpiredTimerPolicy] (FCH::TMR::ACDC::AcExpiredTimerPolicy)</b>	
Read-write. Reset: FFFF_FFFFh.	
_aliasHOST; ACDCx004; ACDC=FED8_1D00h	
Bits	Description
31:0	<b>AcExpiredTimerPolicy.</b> Read-write. Reset: FFFF_FFFFh. When the AC timer expires, the wake signal is asserted if the current power source is AC; the wake signal is not asserted if the current power source is DC. If the power source is switched from DC to AC after the AC timer expired, we wait for the number of seconds defined in this register and then wake up the system. When the AC or DC timer generates a wake up event, this register is reset to FFFF_FFFFh by hardware.
<b>ACDCx008 [AcTimerStatus] (FCH::TMR::ACDC::AcTimerStatus)</b>	
Read,Write-1-to-clear. Reset: 0000_0000h.	
_aliasHOST; ACDCx008; ACDC=FED8_1D00h	
Bits	Description
31:2	Reserved.
1	<b>AcTimerWakeup.</b> Read,Write-1-to-clear. Reset: 0. 0=Wake-up was not caused by AC timer expiration. 1=Wake up was caused by AC timer expiration.
0	<b>AcTimerExpired.</b> Read,Write-1-to-clear. Reset: 0. 0=AC timer has not expired. 1=AC timer expired.
<b>ACDCx010 [DcTimerValue] (FCH::TMR::ACDC::DcTimerVal)</b>	
Read-write. Reset: FFFF_FFFFh.	
_aliasHOST; ACDCx010; ACDC=FED8_1D00h	
Bits	Description
31:0	<b>DcTimerValue.</b> Read-write. Reset: FFFF_FFFFh. <b>Description:</b> Writing the register with a value other than FFFF_FFFFh starts the DC timer. Reading this register returns the current value of the DC timer. When the AC or DC timer generates a wake up event, this register is reset to FFFF_FFFFh by hardware.
<b>ACDCx014 [DcExpiredTimerPolicy] (FCH::TMR::ACDC::DcExpiredTimerPolicy)</b>	
Read-write. Reset: FFFF_FFFFh.	
_aliasHOST; ACDCx014; ACDC=FED8_1D00h	
Bits	Description
31:0	<b>DcExpiredTimerPolicy.</b> Read-write. Reset: FFFF_FFFFh. When the DC timer expires, the wake signal is asserted if the current power source is DC; the wake signal is not asserted if the current power source is AC. If the power source is switched from AC to DC after the DC timer expired, the device waits for the number of seconds defined in this register and then wakes up the system. When the AC or DC timer generates a wake up event, this register is reset to FFFF_FFFFh by hardware.
<b>ACDCx018 [DcTimerStatus] (FCH::TMR::ACDC::DcTimerStatus)</b>	

Read,Write-1-to-clear. Reset: 0000_0000h.	
_aliasHOST; ACDCx018; ACDC=FED8_1D00h	
Bits	Description
31:2	Reserved.
1	<b>DcTimerWakeup.</b> Read,Write-1-to-clear. Reset: 0. 0=Wake-up was not caused by DC timer expiration. 1=Wake-up was caused by DC timer expiration.
0	<b>DcTimerExpired.</b> Read,Write-1-to-clear. Reset: 0. 0=DC timer has not expired. 1=DC timer expired.

ACDCx020 [AcDcTimerCtrl] (FCH::TMR::ACDC::AcDcTimerCtrl)	
Reset: 0000_0000h.	
_aliasHOST; ACDCx020; ACDC=FED8_1D00h	
Bits	Description
31:10	Reserved.
9	<b>DcTimerEventEn.</b> Read-write. Reset: 0. 0=Disable. 1=Enable DC Timer to wake up system.
8	<b>AcTimerEventEn.</b> Read-write. Reset: 0. 0=Disable. 1=Enable AC Timer to wake up system.
7:1	Reserved.
0	<b>Busy.</b> Read-only. Reset: 0. Right after FCH::TMR::ACDC::AcTimerVal or FCH::TMR::ACDC::DcTimerVal is programmed, the hardware sets this bit to 1. The hardware clears this bit once the programming is done and the corresponding timer is started properly. Before the software writes to FCH::TMR::ACDC::AcTimerVal or FCH::TMR::ACDC::DcTimerVal, it has to read this bit and make sure it is 0. Otherwise, the hardware just ignores the programming action from software. For the registers other than FCH::TMR::ACDC::AcTimerVal or FCH::TMR::ACDC::DcTimerVal, there is no such limitation.

## 9.2.8 Always On Always Connected (AOAC) Registers

Table 77: Device D3 Control

Controller Block	FCH Power Group	D3 Control Register	D3 Status Register	ClkOk (From block to ACPI)	AOACCtrl (From ACPI to block)
Clk Gen	PG1	DevCtrl 0	DevSts 0	Tied high	Connected PllRstB and PllLock
AB	PG1	DevCtrl 1	DevSts 1	Tied high	Connected
ACPI S0 (fch_acpismbus)	PG1	DevCtrl 2	DevSts 2	Tied high	Connected
ACPI S5 (fch_acpi_s5)	S5	DevCtrl 3	DevSts 3	Tied high	Not used
LPC	PG1	DevCtrl 4	DevSts 4	Tied high	Connected
I2C2	PG2	DevCtrl 7	DevSts 7	Tied high	Connected
I2C3	PG2	DevCtrl 8	DevSts 8	Tied high	Connected
I2C4	S5	DevCtrl 9	DevSts 9	Tied high	Connected
UART0	PG2	DevCtrl 11	DevSts 11	Tied high	Connected
UART1	PG2	DevCtrl 12	DevSts 12	Tied high	Connected
UART2	PG2	DevCtrl 16	DevSts 16	Tied high	Connected
AMBA	PG1	DevCtrl 17	DevSts 17	Tied high	Connected
SD1	PG2	DevCtrl 25	DevSts 25	Tied high	Connected
UART3	PG2	DevCtrl 26	DevSts 26	Tied high	Connected
eSPI	PG1	DevCtrl 27	DevSts 27	Tied high	Connected
EMMC	PG1a	DevCtrl 28	DevSts 28	Tied high	Connected

NB	PG1	DevCtrl 30	DevSts 30	Tied high	Not used
----	-----	------------	-----------	-----------	----------

**AOACx0000[40...7E] [Device D3 Control] (FCH::AOAC::DevD3Ctl)**

Read-write.

\_link[63:32]\_aliasHOST; AOACx0000[7E,7C,7A,78,76,74,72,70,6E,6C,6A,68,66,64,62,60,5E,5C,5A,58,56,54,52,50,4E,4C,4A,48,46,44,42,40];  
AOAC=FED8\_1E00h

Bits	Description										
7	<b>IsSwControl.</b> Read-write. Reset: 0. 0=Hardware controls control signals (PwrRstB, RefClkOk, RstB) to the device. 1=Software controls control signals (PwrRstB, RefClkOk, RstB) to the device. Software must set bit IsSwControl first before programming bits SwPwrOnRstB, SwRefClkOk and SwRstB, otherwise, unpredictable behavior may result.										
6	<b>SwRstB.</b> Read-write. Reset: 1. 0=RstB is asserted to the device if IsSwControl == 1. 1=RstB is de-asserted to the device if IsSwControl == 1.										
5	<b>SwRefClkOk.</b> Read-write. Reset: 1. 0=RefClkOk is de-asserted to the device if IsSwControl == 1. 1=RefClkOk is asserted to the device if IsSwControl == 1.										
4	<b>SwPwrOnRstB.</b> Read-write. Reset: 1. 0=PwrRstB is asserted to the device if IsSwControl == 1. 1=PwrRstB is de-asserted to the device if IsSwControl == 1.										
3	<b>PwrOnDev.</b> Read-write. Reset: X. 0=Put device in reset state. 1=Put device out of reset state. If SwControl == 0, software can write this bit to trigger a hardware controlled reset sequence to the device. Default value is 0 when N=13-15, 18-24, 29. Default value is 1 when N!=13-15, 18-24, 29.										
2	<b>DeviceState.</b> Read-write. Reset: 1. 0=Device power is removed. 1=Device power is applied. Software records the device power state in this field.										
1:0	<b>TargetedDeviceState.</b> Read-write. Reset: 1h. Software records D-State of the device in this field. Default value is 01b when N=0-12, 16-17, 25-28, 30-31. Default value is 00b when N=13-15, 18-24, 29. <b>ValidValues:</b>										
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>D0 un-initialized</td></tr> <tr> <td>1h</td><td>D0 initialized</td></tr> <tr> <td>2h</td><td>D1/D2/D3hot</td></tr> <tr> <td>3h</td><td>D3cold</td></tr> </table>	Value	Description	0h	D0 un-initialized	1h	D0 initialized	2h	D1/D2/D3hot	3h	D3cold
Value	Description										
0h	D0 un-initialized										
1h	D0 initialized										
2h	D1/D2/D3hot										
3h	D3cold										

**AOACx0000[41...7F] [Device D3 State] (FCH::AOAC::DevD3State)**

Read-only. Reset: 27h.

\_link[63:32]\_aliasHOST; AOACx0000[7F,7D,7B,79,77,75,73,71,6F,6D,6B,69,67,65,63,61,5F,5D,5B,59,57,55,53,51,4F,4D,4B,49,47,45,43,41];  
AOAC=FED8\_1E00h

Bits	Description
7	<b>Stat1.</b> Read-only. Reset: 0. State of device.
6	<b>Stat0.</b> Read-only. Reset: 0. State of device.
5	<b>ClkOkState.</b> Read-only. Reset: 1. Specifies the state of device's ClkOk signal.
4	<b>D3Cold.</b> Read-only. Reset: 0. Specifies the state of device's D3 Cold signal.
3	<b>DevOffGatingState.</b> Read-only. Reset: 0. Specifies the state of device's DevOffGating signal.
2	<b>RstBState.</b> Read-only. Reset: 1. Specifies the state of device's RstB signal.
1	<b>RefClkOkState.</b> Read-only. Reset: 1. Specifies the state of device's RefClkOk signal.
0	<b>PwrRstBState.</b> Read-only. Reset: 1. Specifies the state of device's PwrRstB signal.

**AOACx000084 [Shadow Register Status] (FCH::AOAC::ShadowRegStat)**

Read-write. Reset: 3FFF\_FFFFh.

\_aliasHOST; AOACx000084; AOAC=FED8\_1E00h

Bits	Description
31:30	Reserved.
29:0	<b>Status.</b> Read-write. Reset: 3FFF_FFFFh. Reading these bits returns the status of the store or restore action. Each

bit represents the status of a device. For each bit, Software can also write this register to 1 or 0. This provides software a way to override the status.

**Valid Values:**

Bit	Description
[0]	0=Shadow register logic has finished store action for a device. 1=Shadow register has finished restore action for a device.
[1]	0=Shadow register logic has finished store action for a device. 1=Shadow register has finished restore action for a device.
[2]	0=Shadow register logic has finished store action for a device. 1=Shadow register has finished restore action for a device.
[3]	0=Shadow register logic has finished store action for a device. 1=Shadow register has finished restore action for a device.
[4]	0=Shadow register logic has finished store action for a device. 1=Shadow register has finished restore action for a device.
[5]	0=Shadow register logic has finished store action for a device. 1=Shadow register has finished restore action for a device.
[6]	0=Shadow register logic has finished store action for a device. 1=Shadow register has finished restore action for a device.
[7]	0=Shadow register logic has finished store action for a device. 1=Shadow register has finished restore action for a device.
[8]	0=Shadow register logic has finished store action for a device. 1=Shadow register has finished restore action for a device.
[9]	0=Shadow register logic has finished store action for a device. 1=Shadow register has finished restore action for a device.
[10]	0=Shadow register logic has finished store action for a device. 1=Shadow register has finished restore action for a device.
[11]	0=Shadow register logic has finished store action for a device. 1=Shadow register has finished restore action for a device.
[12]	0=Shadow register logic has finished store action for a device. 1=Shadow register has finished restore action for a device.
[13]	0=Shadow register logic has finished store action for a device. 1=Shadow register has finished restore action for a device.
[14]	0=Shadow register logic has finished store action for a device. 1=Shadow register has finished restore action for a device.
[15]	0=Shadow register logic has finished store action for a device. 1=Shadow register has finished restore action for a device.
[16]	0=Shadow register logic has finished store action for a device. 1=Shadow register has finished restore action for a device.
[17]	0=Shadow register logic has finished store action for a device. 1=Shadow register has finished restore action for a device.
[18]	0=Shadow register logic has finished store action for a device. 1=Shadow register has finished restore action for a device.
[19]	0=Shadow register logic has finished store action for a device. 1=Shadow register has finished restore action for a device.
[20]	0=Shadow register logic has finished store action for a device. 1=Shadow register has finished restore action for a device.
[21]	0=Shadow register logic has finished store action for a device. 1=Shadow register has finished restore action for a device.
[22]	0=Shadow register logic has finished store action for a device. 1=Shadow register has finished restore action for a device.

[23]	0=Shadow register logic has finished store action for a device. 1=Shadow register has finished restore action for a device.
[24]	0=Shadow register logic has finished store action for a device. 1=Shadow register has finished restore action for a device.
[25]	0=Shadow register logic has finished store action for a device. 1=Shadow register has finished restore action for a device.
[26]	0=Shadow register logic has finished store action for a device. 1=Shadow register has finished restore action for a device.
[27]	0=Shadow register logic has finished store action for a device. 1=Shadow register has finished restore action for a device.
[28]	0=Shadow register logic has finished store action for a device. 1=Shadow register has finished restore action for a device.
[29]	0=Shadow register logic has finished store action for a device. 1=Shadow register has finished restore action for a device.

#### AOACx000088 [Shadow Register SRAM Addr] (FCH::AOAC::ShadowRegSRAMAddr)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; AOACx000088; AOAC=FED8\_1E00h

Bits	Description
31:16	Reserved.
15:0	<b>ShadowSRamAddr.</b> Read-write. Reset: 0000h. Software has indirect access to the SRAM in shadow register block via FCH::AOAC::ShadowRegSRAMAddr and FCH::AOAC::ShadowRegSRAMData. The field specifies the shadow SRAM address to be accessed.

#### AOACx00008C [Shadow Register SRAM Data] (FCH::AOAC::ShadowRegSRAMData)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; AOACx00008C; AOAC=FED8\_1E00h

Bits	Description
31:0	<b>ShadowSRamData.</b> Read-write. Reset: 0000_0000h. Specifies the data to be written or read for the shadow SRAM address specified by FCH::AOAC::ShadowRegSRAMAddr.

#### AOACx000094 [S0I3 Control] (FCH::AOAC::S0I3Ctl)

Reset: 0000\_0000h.

\_aliasHOST; AOACx000094; AOAC=FED8\_1E00h

Bits	Description
31:18	Reserved.
17	<b>PllOffReq.</b> Write-only. Reset: 0. Read back always 0. SMU can write 1 to this bit to request FCH to shutdown CGPLL.
16	<b>IntrBlocked.</b> Read-only. Reset: 0. 0=FCH interrupts are not blocked. 1=FCH interrupts are blocked. If the FCH is sending an interrupt message when software or SMU write 1 to InterruptDis, the interrupt is blocked "gracefully". Wait until the interrupt is sent and then block the interrupt. This bit indicates the interrupt blocking status.
15	<b>InterruptDis.</b> Read-write. Reset: 0. 0=Interrupt from FCH is enabled. 1=Interrupt from FCH is disabled. Software is able to set and clear this bit. Hardware can only set this bit.
14	<b>ArbiterDis.</b> Read-write. Reset: 0. 0=FCH upstream arbiter is enabled. 1=FCH upstream arbiter is disabled. Software is able to set and clear this bit. Hardware can only set this bit.
13:11	Reserved.
10	<b>ShadowTimerWakeSts.</b> Read-only. Reset: 0. 0=Shadow Timer did not fire. 1=Shadow Timer fired and caused a wake-up from the S0I3 state. Hardware clears this bit during S0I3 entry.
9	<b>S0I3Resume.</b> Read,Write-1-to-clear. Reset: 0. Set to 1 by S0I3 exit to indicate the system has resumed from the S0I3 state. The bit is cleared by S0I3 entry or write-1-to-cleared by software.
8	<b>S0I3Enter.</b> Read-only. Reset: 0. 0=Indicate the system is in S0 state. 1=Indicate the system is in S0I3 state.



7:5	Reserved.
4	<b>S0I3Trigger</b> . Write-only. Reset: 0. Setting the bit to 1 triggers S0I3 power down sequence.
3	Reserved.
2	<b>S0I3Ready2</b> . Read-write. Reset: 0. 0=S0I3 can't be entered. 1=S0I3 can be entered. Programmed by Driver.
1	<b>S0I3OnSlpS3B</b> . Read-write. Reset: 0. 0=SLP_S3# doesn't assert in S0I3 state. 1=SLP_S3# asserts in S0I3 state.
0	<b>S0I3Ready</b> . Read-write. Reset: 0. 0=S0I3 cannot be entered. 1=S0I3 can be entered. Programmed by BIOS.

#### AOACx00009C [Shadow Timer Control] (FCH::AOAC::ShadowTimerCtl)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; AOACx00009C; AOAC=FED8\_1E00h

Bits	Description
31:24	<b>RestoredOffset64</b> . Read-write. Reset: 00h. Specifies a programmable offset to be added to restored HPET 64-bit timer; this is to account for restore time uncertainty.
23:16	<b>RestoredOffset</b> . Read-write. Reset: 00h. Specifies a programmable offset to be added to restored HPET 32-bit timer and ACPI timer; this is to account for restore time uncertainty.
15:8	<b>EarlyCount</b> . Read-write. Reset: 00h. Specifies a programmable offset to be subtracted from the alarm value.
7:3	Reserved.
2	<b>EarlyCountUnit</b> . Read-write. Reset: 0. 0=The unit of EarlyCount is 1 RTC clock period. 1=The unit of EarlyCount is 16 RTC clock period.
1	<b>ShadowAcpiTimerEn</b> . Read-write. Reset: 0. 1=Enable control to perform store/restore operation for ACPI timer.
0	<b>ShadowHpetEn</b> . Read-write. Reset: 0. 1=Enable control to perform store/restore operation from HPET timer.

#### AOACx0000A0 [PwrGood Control] (FCH::AOAC::PwrGoodCtl)

Read-write. Reset: 0000\_8003h.

\_aliasHOST; AOACx0000A0; AOAC=FED8\_1E00h

Bits	Description
31:25	Reserved.
24	<b>SpiPadDisable</b> . Read-write. Reset: 0. 1=Disable the OE and PU of the SPI pads.
23	<b>LpcPadDisable</b> . Read-write. Reset: 0. 1=Disable the OE and PU of the LPC pads.
22:16	Reserved.
15	<b>PowerAllPwrIsland</b> . Read-write. Reset: 1. 0=When software programs [PG1a/PG2/XHC], ACPI powers up all groups immediately. 1=Indicates that when software programs [PG1a/PG2/XHC], ACPI powers up all power groups sequentially to avoid current surge. If the platform supports AOAC, this bit must be 0 during boot, otherwise, this bit remains set.
14:2	Reserved.
1	<b>PG2PwrGood</b> . Read-write. Reset: 1. 0=PG2 is powered down. 1=PG2 is powered up. Controls the power of Power Group 2. Waking up from S3 or S5 sets this bit to default value.
0	<b>PG1aPwrGood</b> . Read-write. Reset: 1. 0=PG1a is powered down. 1=PG1a is powered up. Controls the power of Power Group 1a. Waking up from S3 or S5 sets this bit to default value.

## 9.2.9 ISA Bridge

### 9.2.9.1 Device 14h Function 3 (LPC Bridge) Configuration Registers

#### D14F3x000 [Device/Vendor ID] (FCH::ITF::LPC::PciDevVendID)

Read-only.

\_aliasHOST; D14F3x000

Bits	Description
31:16	<b>DeviceID: Device Identifier</b> . Read-only. Reset: Fixed,790Eh. This 16-bit field is assigned by the device



	manufacturer and identifies the type of device.
15:0	<b>VendorID: Vendor Identifier.</b> Read-only. Reset: 1022h. PCI vendor identifier.

**D14F3x004 [Status/Command] (FCH::ITF::LPC::PciStatCmd)**

_aliasHOST; D14F3x004	
Bits	Description
31	<b>DetectedParityError: Detected Parity Error.</b> Read,Write-1-to-clear. Reset: 0. 1=The FCH detects a parity error. Parity error detection.
30	<b>SignaledSystemError: Signaled System Error.</b> Read,Write-1-to-clear. Reset: 0. 1=The FCH detects a PCI address parity error. System error detection.
29	<b>ReceivedMasterAbort: Received Master Abort.</b> Read,Write-1-to-clear. Reset: 0. 1=The FCH acts as a PCI master and aborts a PCI bus memory cycle. Master abort received.
28	<b>ReceivedTargetAbort: Received Target Abort.</b> Read,Write-1-to-clear. Reset: 0. 1=An FCH generated PCI cycle (the FCH is the PCI master) is aborted by a PCI target. Target abort received.
27	<b>SignaledTargetAbort: Signaled Target Abort.</b> Read,Write-1-to-clear. Reset: 0. 1=The FCH signals target abort. Target abort signal.
26:25	<b>DeviceSelectTiming: Device Select Timing.</b> Read-only. Reset: Fixed,1h. Indicates DEVSEL# timing when performing a positive decode. DEVSEL# is asserted to meet the medium timing.
24	<b>MasterDataParityError: Master Data Parity Error.</b> Read,Write-1-to-clear. Reset: 0. 1=The FCH detects PERR# asserted while acting as PCI master regardless whether PERR# was driven by the FCH or not. Master data parity error.
23:21	Reserved.
20	<b>CapabilitiesList: Capabilities List.</b> Read-only. Reset: 0. Always reads 0.
19:10	Reserved.
9	<b>FastBack2BackEnable: Fast Back-to-Back Enable.</b> Read-only. Reset: Fixed,0. Enable fast back-to-back transactions.
8	<b>SERREnable: SERR# Enable.</b> Read-only. Reset: 0. 0=SERR# is not asserted. 1=The FCH asserts SERR# when it detects an address parity error. SERR# assertion on address parity error.
7	<b>SteppingControl: Stepping Control.</b> Read-only. Reset: Fixed,0. Stepping control.
6	<b>ParityErrorResponse: Parity Error Response.</b> Read-write. Reset: 0. 0=PERR# is not asserted. 1=The FCH asserts PERR# when it is the agent receiving data and it detects a parity error. PERR# (Response) Detection enable bit.
5	<b>VGAPaletteSnoop: VGA Palette Snoop.</b> Read-only. Reset: Fixed,0. Snoop VGA color palette.
4	<b>MemoryWriteAndInvalidateEnable: Memory Write and Invalidate Enable.</b> Read-only. Reset: Fixed,0. Enables Memory Write and Invalidate command.
3	<b>SpecialCycles: Special Cycles.</b> Read-only. Reset: Fixed,1. Controls Special Cycle operations.
2	<b>BusMaster: Bus Master.</b> Read-only. Reset: 1. 1=Bus master enabled.
1	<b>MemorySpace: Memory Space.</b> Read-only. Reset: Fixed,1. Controls a device's response to Memory Space accesses.
0	<b>IOSpace: IO Space.</b> Read-only. Reset: 1. 1=Enable access to the legacy IDE ports and PCI bus master IDE IO registers are enabled. This bit controls access to the IO space registers.

**D14F3x008 [Revision ID/Class Code] (FCH::ITF::LPC::PciClassCodeRevId)**

Read-only.	
_aliasHOST; D14F3x008	
Bits	Description
31:8	<b>ClassCode: Class Code.</b> Read-only. Reset: 06_0100h. Indicates an ISA bridge.
7:0	<b>RevisionID: Revision ID.</b> Read-only. Reset: Fixed,51h. Indicates the revision level of the design.

**D14F3x00C [Cache Line Size] (FCH::ITF::LPC::PciHeadType)**

Read-only.	
------------	--

_aliasHOST; D14F3x00C	
Bits	Description
31:24	<b>Bist.</b> Read-only. Reset: 00h. No BIST modes.
23:16	<b>HeaderType: Header Type.</b> Read-only. Reset: 80h. Identifies the type of the predefined header in the configuration space. The most significant bit is set to 1 to indicate a multi-function device.
15:8	<b>LatencyTimer: Latency Timer.</b> Read-only. Reset: 00h. Specifies the value of the latency timer in units of PCICLKs.
7:0	<b>CacheLineSize: Cache Line Size.</b> Read-only. Reset: Fixed,00h. Specifies the size of the cache line.

**D14F3x010 [Base Address Reg 0] (FCH::ITF::LPC::BaseAddrReg0)**

_aliasHOST; D14F3x010	
Bits	Description
31:5	<b>BaseAddress0[31:5]: Base Address 0.</b> Write-only. Reset: Fixed,7F6_0000h. Read as zero. This register has an internal value used as base address BaseAddress0[31:5] for APIC memory space. Writing to the register changes its internal value. The default internal base address is FEC0_0000h.
4:0	<b>BaseAddress0[4:0].</b> Read-only. Reset: 00h. This register has an internal value used as base address BaseAddress0[4:0] for APIC memory space.

**D14F3x02C [Subsystem ID and Subsystem Vendor ID] (FCH::ITF::LPC::SubsystemIDSubsystemVendorID)**

Read,Write-once. Reset: 790E_1022h.	
_aliasHOST; D14F3x02C	
Bits	Description
31:16	<b>SubsystemID: Subsystem ID.</b> Read,Write-once. Reset: 790Eh. Subsystem identifier.
15:0	<b>SubsystemVendorID: Subsystem Vendor ID.</b> Read,Write-once. Reset: 1022h. PCI vendor identifier.

**D14F3x034 [Capabilities Pointer] (FCH::ITF::LPC::PciCapPtr)**

Read-only. Reset: 0000_0000h.	
_aliasHOST; D14F3x034	
Bits	Description
31:8	Reserved.
7:0	<b>CapabilitiesPointer: Capabilities Pointer.</b> Read-only. Reset: 00h. No capability, always Read 00h.

**D14F3x040 [PCI Control] (FCH::ITF::LPC::PCICtl)**

Reset: 0000_001Ch.	
_aliasHOST; D14F3x040	
Bits	Description
31:7	Reserved.
6	<b>EcSemaphore.</b> Read-only. Reset: 0. 0=SPI ROM is not locked by External EC. 1=SPI ROM is locked by External EC. SPI ROM locking.
5	<b>BiosSemaphore.</b> Read-write. Reset: 0. 0=SPI ROM is not locked by Host. 1=SPI ROM is locked by Host. This bit is writable by host to block EC ROM access. Whenever the host needs to lock down ROM access then the host should continue writing the bit to 1 until it reads back 1.
4	<b>ExtRomSharingEn.</b> Read-write. Reset: 1. 0=Disable ROM sharing in External mode. 1=Support ROM sharing in External mode.
3	<b>VWRomSharingEn.</b> Read-write. Reset: 1. 0=Disable ROM sharing in Virtual write mode. 1=Support ROM sharing in Virtual write mode.
2	<b>LegacyDmaEnable: Legacy DMA Enable.</b> Read-write. Reset: 1. 1=Enable LPC DMA cycle. Transfer size for channel[3:0] is 8 bits; Transfer size for channel[7:5] is 16 bits. 32-bit DMA is not supported.
1:0	Reserved.

**D14F3x044 [IO Port Decode Enable] (FCH::ITF::LPC::IOPortDecodeEn)**

Read-write. Reset: 0000_0000h.	
_aliasHOST; D14F3x044	

Bits	Description
31	<b>AdLibPortEnable: Ad-Lib Port Enable.</b> Read-write. Reset: 0. 1=Enable the IO range. Port enable for Ad-Lib port, 388h-389h.
30	<b>AcpiMicroControllerPortEnable: ACPI Micro-Controller Port Enable.</b> Read-write. Reset: 0. 1=Enable the IO range. Port enable for ACPI micro-controller port, 62h & 66h.
29	<b>KBCPortEnable: KBC Port Enable.</b> Read-write. Reset: 0. 1=Enable the IO range. Port enable for KBC port, 60h & 64h.
28	<b>GamePortEnable: Game Port Enable.</b> Read-write. Reset: 0. 1=Enable the IO range. Port enable for game port, 200h-20Fh.
27	<b>FDCPortEnable1: FDC Port Enable 1.</b> Read-write. Reset: 0. 1=Enable the IO range. Port enable for FDC port, 370h-377h.
26	<b>FDCPortEnable0: FDC Port Enable 0.</b> Read-write. Reset: 0. 1=Enable the IO range. Port enable for FDC port, 3F0h-3F7h.
25	<b>MSSPortEnable3: MSS Port Enable 3.</b> Read-write. Reset: 0. 1=Enable the IO range. Port enable for MSS port, F40h-F47h.
24	<b>MSSPortEnable2: MSS Port Enable 2.</b> Read-write. Reset: 0. 1=Enable the IO range. Port enable for MSS port, E80h-E87h.
23	<b>MSSPortEnable1: MSS Port Enable 1.</b> Read-write. Reset: 0. 1=Enable the IO range. Port enable for MSS port, 604h-60bh.
22	<b>MSSPortEnable0: MSS Port Enable 0.</b> Read-write. Reset: 0. 1=Enable the IO range. Port enable for MSS port, 530h-537h.
21	<b>MIDIPortEnable3: MIDI Port Enable 3.</b> Read-write. Reset: 0. 1=Enable the IO range. Port enable for MIDI port, 330h-331h.
20	<b>MIDIPortEnable2: MIDI Port Enable 2.</b> Read-write. Reset: 0. 1=Enable the IO range. Port enable for MIDI port, 320h-321h.
19	<b>MIDIPortEnable1: MIDI Port Enable 1.</b> Read-write. Reset: 0. 1=Enable the IO range. Port enable for MIDI port, 310h-311h.
18	<b>MIDIPortEnable0: MIDI Port Enable 0.</b> Read-write. Reset: 0. 1=Enable the IO range. Port enable for MIDI port, 300h-301h.
17	<b>AudioPortEnable3: Audio Port Enable 3.</b> Read-write. Reset: 0. 1=Enable the IO range. Port enable for audio port, 280h-293h.
16	<b>AudioPortEnable2: Audio Port Enable 2.</b> Read-write. Reset: 0. 1=Enable the IO range. Port enable for audio port, 260h-273h.
15	<b>AudioPortEnable1: Audio Port Enable 1.</b> Read-write. Reset: 0. 1=Enable the IO range. Port enable for audio port, 240h-253h.
14	<b>AudioPortEnable0: Audio Port Enable 0.</b> Read-write. Reset: 0. 1=Enable the IO range. Port enable for audio port, 230h-233h.
13	<b>SerialPortEnable7: Serial Port Enable 7.</b> Read-write. Reset: 0. 1=Enable the IO range. Port enable for serial port, 3E8h-3EFh.
12	<b>SerialPortEnable6: Serial Port Enable 6.</b> Read-write. Reset: 0. 1=Enable the IO range. Port enable for serial port, 338h-33Fh.
11	<b>SerialPortEnable5: Serial Port Enable 5.</b> Read-write. Reset: 0. 1=Enable the IO range. Port enable for serial port, 2E8h-2EFh.
10	<b>SerialPortEnable4: Serial Port Enable 4.</b> Read-write. Reset: 0. 1=Enable the IO range. Port enable for serial port, 238h-23Fh.
9	<b>SerialPortEnable3: Serial Port Enable 3.</b> Read-write. Reset: 0. 1=Enable the IO range. Port enable for serial port, 228h-22Fh.
8	<b>SerialPortEnable2: Serial Port Enable 2.</b> Read-write. Reset: 0. 1=Enable the IO range. Port enable for serial port, 220h-227h.
7	<b>SerialPortEnable1: Serial Port Enable 1.</b> Read-write. Reset: 0. 1=Enable the IO range. Port enable for serial port, 2F8h-2FFh.

6	<b>SerialPortEnable0: Serial Port Enable 0.</b> Read-write. Reset: 0. 1=Enable the IO range. Port enable for serial port, 3F8h-3FFh.
5	<b>ParallelPortEnable5: Parallel Port Enable 5.</b> Read-write. Reset: 0. 1=Enable the IO range. Port enable for parallel port, 7BCh-7BFh.
4	<b>ParallelPortEnable4: Parallel Port Enable 4.</b> Read-write. Reset: 0. 1=Enable the IO range. Port enable for parallel port, 3BCh-3BFh.
3	<b>ParallelPortEnable3: Parallel Port Enable 3.</b> Read-write. Reset: 0. 1=Enable the IO range. Port enable for parallel port, 678h-67Fh.
2	<b>ParallelPortEnable2: Parallel Port Enable 2.</b> Read-write. Reset: 0. 1=Enable the IO range. Port enable for parallel port, 278h-27Fh.
1	<b>ParallelPortEnable1: Parallel Port Enable 1.</b> Read-write. Reset: 0. 1=Enable the IO range. Port enable for parallel port, 778h-77Fh.
0	<b>ParallelPortEnable0: Parallel Port Enable 0.</b> Read-write. Reset: 0. 1=Enable the IO range. Port enable for parallel port, 378h-37Fh.

#### D14F3x048 [IO/Mem Port Decode Enable] (FCH::ITF::LPC::IOMemPortDecodeEn)

Read-write. Reset: 0000\_FF00h.

\_aliasHOST; D14F3x048

Bits	Description
31:26	Reserved.
25	<b>WideIO2Enable.</b> Read-write. Reset: 0. 1=Enable the IO range. Port enable for wide generic IO port 2 defined by FCH::ITF::LPC::WideIO2[IOBaseAddress2].
24	<b>WideIO1Enable.</b> Read-write. Reset: 0. 1=Enable the IO range. Port enable for wide generic IO port 1 defined by FCH::ITF::LPC::PciIoBaseAddrWideGenPort[IOBaseAddress1].
23	<b>IOPortEnable6: IO port enable 6.</b> Read-write. Reset: 0. 1=Enable the IO range. Port enable for IO port FD60h-FD6Fh.
22	<b>IOPortEnable5: IO port enable 5.</b> Read-write. Reset: 0. 1=Enable the IO range. Port enable for IO port 4700h-470Bh.
21	<b>IOPortEnable4: IO port enable 4.</b> Read-write. Reset: 0. 1=Enable the IO range. Port enable for IO port 80h.
20	<b>MemPortEnable: Mem port enable.</b> Read-write. Reset: 0. 1=Enable the memory range. Port enable for 4K byte memory range defined in FCH::ITF::LPC::MemRng.
19	<b>IOPortEnable3: IO port enable 3.</b> Read-write. Reset: 0. 1=Enable the IO range. Port enable for IO port 580h-5BFh.
18	<b>IOPortEnable2: IO port enable 2.</b> Read-write. Reset: 0. 1=Enable the IO range. Port enable for IO port 500h-53Fh.
17	<b>IOPortEnable1: IO port enable 1.</b> Read-write. Reset: 0. 1=Enable the IO range. Port enable for IO port 480h-4BFh.
16	<b>IOPortEnable0: IO port enable 0.</b> Read-write. Reset: 0. 1=Enable the IO range. Port enable for IO port 400h-43Fh.
15:8	<b>SyncTimeoutCount: Synchronization Timeout Count.</b> Read-write. Reset: FFh. When [SyncTimeoutCounterEnable] == 1, this field specifies the number of LPC clocks that the state machine waits during LPC data synchronization before aborting the cycle.
7	<b>SyncTimeoutCounterEnable: Synchronization Timeout Counter Enable.</b> Read-write. Reset: 0. 0=The counter is disabled. 1=LPC sync timeout counter is enabled. This counter is used to avoid a deadlock condition if an LPC device drives sync forever. Timeout count is programmed in [SyncTimeoutCount]. Write 0 to this bit if an LPC device is extremely slow and takes more than 255 LPC clocks to complete a cycle.
6	<b>RtcIORangePortEnable: RTC IO Range Port Enable.</b> Read-write. Reset: 0. 1=Enable the IO range. Port enable for RTC IO range 70h-73h.
5	<b>MemoryRangePortEnable: Memory Range Port Enable.</b> Read-write. Reset: 0. 1=Enable the memory range. Port enable for LPC memory target range defined by FCH::ITF::LPC::PCIMemAddrforLPCTargCyc.
4:3	Reserved.

2	<b>WideIO0Enable.</b> Read-write. Reset: 0. 1=Enable the IO range. Port enable for wide generic IO port defined by FCH::ITF::LPC::PciIoBaseAddrWideGenPort[IOBaseAddress0].
1	<b>AlternateSuperIOConfigurationPortEnable: Alternate Super IO Configuration Port Enable.</b> Read-write. Reset: 0. 1=Enable the IO range. Port enable for alternate Super IO configuration port, 4Eh-4Fh.
0	<b>SuperIOConfigurationPortEnable: Super IO Configuration Port Enable.</b> Read-write. Reset: 0. 1=Enable the IO range. Port enable for Super IO configuration port, 2Eh-2Fh.

**D14F3x04C [Memory Range] (FCH::ITF::LPC::MemRng)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; D14F3x04C

Bits	Description
31:12	<b>BaseAddress: Base Address.</b> Read-write. Reset: 0_0000h. Specifies a 4K byte memory range from {Base Address, 000h} to {Base Address, FFFh}. The range is enabled by FCH::ITF::LPC::IOMemPortDecodeEn[MemPortEnable].
11:0	Reserved.

**D14F3x05[0...C] [ROM Protect 3, 2, 1, 0] (FCH::ITF::LPC::RomProtect)**

Read,Write-once. Reset: 0000\_0000h.

This register specifies different ROM ranges to be protected. FCH::ITF::SPI::AltSPICS[SpiProtectEn0] enables the protection ranges. The addresses are within the defined ROM range if:

{RomBase, 000\_0000\_0000b} <= address[31:0] <= ({RomBase, 000\_0000\_0000b} + (Range << (RangeUnit ? 16 : 12)))

For the host, this register can only be written once after hardware reset; subsequent writes have no effect.

To enable writing to this register again, one can generate an SMI through FCH or apply hardware reset.

\_n[3:0]\_aliasHOST; D14F3x05[C,8,4,0]

Bits	Description
31:12	<b>RomBase: ROM Base.</b> Read,Write-once. Reset: 0_0000h. <b>Description:</b> Protected range is defined below: {RomBase, 12'h0} ~ {RomBase, 12'h0} + (Range+1) *4K (or 64K if bit 2 is 1)
11	Reserved.
10	<b>WriteProtect: Write Protect.</b> Read,Write-once. Reset: 0. 1=The memory range defined by this register is write-protected and writing to the range has no effect.
9	<b>ReadProtect: Read Protect.</b> Read,Write-once. Reset: 0. 1=The memory range defined by this register is read-protected and reading any location in the range returns FFFF_FFFFh.
8	<b>RangeUnit.</b> Read,Write-once. Reset: 0. 0=4 KB. 1=64 KB.
7:0	<b>Range.</b> Read,Write-once. Reset: 00h. <b>Description:</b> Specifies the protected range. The unit is defined at bit[8] in the same register. NOTE: The protection is limited to 4GB boundary. Base + Range cannot cross 4GB boundary; otherwise, hardware will not behave correctly. BIOS should make sure the values are within a valid range.

**D14F3x060 [PCI Memory Address for LPC Target Cycles] (FCH::ITF::LPC::PCIMemAddrforLPC TargCyc)**

Read-write. Reset: 0000\_0000h.

This register contains the upper 16 bits of the start and end address of the LPC memory target range. The lower 16 bits of MemoryStartAddress are 0000h. The lower 16 bits of MemoryEndAddress are FFFFh. This range can be enabled or disabled using FCH::ITF::LPC::IOMemPortDecodeEn[MemoryRangePortEnable].

\_aliasHOST; D14F3x060

Bits	Description
31:16	<b>MemoryEndAddress: Memory End Address.</b> Read-write. Reset: 0000h. Specifies the upper 16 bits of the end address of the LPC target memory range.
15:0	<b>MemoryStartAddress: Memory Start Address.</b> Read-write. Reset: 0000h. Specifies the upper 16 bits of the start address of the LPC target memory range.



**D14F3x064 [PCI IO base Address for Wide Generic Port] (FCH::ITF::LPC::PciIoBaseAddrWideGenPort)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; D14F3x064

Bits	Description
31:16	<b>IOBaseAddress1: IO Base Address 1.</b> Read-write. Reset: 0000h. 16-bit PCI IO base address for wide generic IO port range. This function is enabled by FCH::ITF::LPC::IOMemPortDecodeEn[WideIO1Enable]. If FCH::ITF::LPC::AlternativeWideIORngEn[AlternativeWideIO1RangeEnable] == 1, the range is 16 bytes; else the range is 512 bytes.
15:0	<b>IOBaseAddress0: IO Base Address 0.</b> Read-write. Reset: 0000h. 16-bit PCI IO base address for wide generic IO port range. This function is enabled by FCH::ITF::LPC::IOMemPortDecodeEn[WideIO0Enable]. If FCH::ITF::LPC::AlternativeWideIORngEn[AlternativeWideIO0RangeEnable] == 1, the range is 16 bytes; else the range is 512 bytes.

**D14F3x068 [ROM Address Range 1] (FCH::ITF::LPC::ROMAddrRng1)**

Read-write. Reset: 000F\_0008h.

This register contains the upper 16 bits of the start and end address of the ROM address range 1. The lower 16 bits of RomStartAddress1 are 0000h. The lower 16 bits of the RomEndAddress1 are FFFFh. This register is used for both LPC and SPI flash.

\_aliasHOST; D14F3x068

Bits	Description
31:16	<b>RomEndAddress1: ROM End Address 1.</b> Read-write. Reset: 000Fh. Specifies the upper 16 bits of the end address of the ROM memory address range 1. If the strap is disabled, the reset value is 0h.
15:0	<b>RomStartAddress1: ROM Start Address 1.</b> Read-write. Reset: 0008h. Specifies the upper 16 bits of the start address of the ROM memory address range 1. If the strap is disabled, the reset value is 0h. Default is set to 512K below 1M.

**D14F3x06C [ROM Address Range 2] (FCH::ITF::LPC::ROMAddrRng2)**

Read-write. Reset: FFFF\_FF00h.

This register contains the upper 16 bits of the start and end address of the ROM address range 2. The lower 16 bits of RomStartAddress2 are 0000h. The lower 16 bits of RomEndAddress2 are FFFFh. This register is used for both LPC and SPI flash.

\_aliasHOST; D14F3x06C

Bits	Description
31:16	<b>RomEndAddress2: ROM End Address 2.</b> Read-write. Reset: FFFFh. Specifies the upper 16 bits of the end address of the ROM memory address range 2. If the strap is disabled, the reset value is 0h.
15:0	<b>RomStartAddress2: ROM Start Address 2.</b> Read-write. Reset: FF00h. Specifies the upper 16 bits of the start address of the ROM memory address range 2. If the strap is disabled, the reset value is 0h.

**D14F3x074 [Alternative Wide IO Range Enable] (FCH::ITF::LPC::AlternativeWideIORngEn)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; D14F3x074

Bits	Description
31:4	Reserved.
3	<b>AlternativeWideIO2RangeEnable: Alternative Wide IO 2 Range Enable.</b> Read-write. Reset: 0. This bit is similar to bit[AlternativeWideIO0RangeEnable], but it applies to the IO range defined by FCH::ITF::LPC::WideIO2[IOBaseAddress2]. See bit[AlternativeWideIO0RangeEnable] for a detailed description.
2	<b>AlternativeWideIO1RangeEnable: Alternative Wide IO 1 Range Enable.</b> Read-write. Reset: 0. This bit is similar to bit[AlternativeWideIO0RangeEnable], but it applies to the IO range defined by FCH::ITF::LPC::PciIoBaseAddrWideGenPort[IOBaseAddress1]. See bit[AlternativeWideIO0RangeEnable] for a detailed description.
1	Reserved.

0	<b>AlternativeWideIO0RangeEnable: Alternative Wide IO 0 Range Enable.</b> Read-write. Reset: 0. 0=Wide IO range defined by FCH::ITF::LPC::PciIoBaseAddrWideGenPort[IOBaseAddress0] is 512 bytes. 1=The range is 16 bytes. To use this feature, address in FCH::ITF::LPC::PciIoBaseAddrWideGenPort[IOBaseAddress0] must be aligned to 16 bytes, i.e., bits[3:0] must be 0. If the address is not aligned to 16 bytes, the IO range is from address[15:0] to {address[15:4], 0xF}.
---	--

**D14F3x078 [Miscellaneous Control Bits] (FCH::ITF::LPC::MiscCtlBits)**

Read-write. Reset: 0000\_0091h.

\_aliasHOST; D14F3x078

Bits	Description
31:11	Reserved.
10	<b>LDRQ0_PD_EN.</b> Read-write. Reset: 0. 0=Disable the pull-down of LDRQ0 pad. 1=Enable the pull-down of LDRQ0 pad. LDRQ0_PD_EN.
9	<b>LDRQ0_PU_EN.</b> Read-write. Reset: 0. 0=Disable the pull-up of LDRQ0 pad. 1=Enable the pull-up of LDRQ0 pad. LDRQ0_PU_EN.
8	Reserved.
7	<b>AllowHostInDma.</b> Read-write. Reset: 1. 0=DMA hold LPC even ACPI has not given GNT to LPC during DMA transfer. 1=Allow Host to access LPC if ACPI has not given GNT to LPC during DMA transfer.
6	<b>GateWrongRx.</b> Read-write. Reset: 0. 1=Allow AltRxByteCount to be 0.
5	<b>GateSpiAccessDis.</b> Read-write. Reset: 0. 1=Pass ROM access to SPI even if it is strapped as LPC.
4	<b>SMMWriteRomEn.</b> Read-write. Reset: 1. 1=Enable ROM access in SMM mode.
3	Reserved.
2	<b>LDRQ0.</b> Read-write. Reset: 0. 1=Enable LDRQ0# on LPC bus. Enable LDRQ0.
1	Reserved.
0	<b>NoHog: No Hog.</b> Read-write. Reset: 1. 0=LPC may hold the internal bus during a DMA transfer. 1=The internal bus is not locked by LPC bridge during a slave access. LPC DMA fetch.

**D14F3x07C [TPM] (FCH::ITF::LPC::TPM)**

\_aliasHOST; D14F3x07C

Bits	Description
31:14	Reserved.
13	<b>TpmBufferEn.</b> Read-write. Reset: 0. 0=Disable TPM buffer. 1=Enable TPM buffer.
12	<b>TpmPfetchEn.</b> Read-write. Reset: 0. 0=Disable TPM burst Read. 1=Enable TPM burst Read.
11	<b>LpcClk1IsGpio.</b> Read-write. Reset: 1. 0=Treat LpcClk1 as LpcClk1. 1=Treat LpcClk1 as GPIO.
10	<b>GpioLpcClk1Out.</b> Read-write. Reset: 0. Control GpioLpcClk1 output value.
9	<b>GpioLpcClk1OeB.</b> Read-write. Reset: 1. 0=Enable GpioLpcClk1 output. 1=Disable GpioLpcClk1 output.
8	<b>GpioLpcClk1.</b> Read-only. Reset: X. Status of LpcClk1 port.
7	<b>WiderTpmEn.</b> Read-write. Reset: 0. 1=Force logic to decode FED4_XXXXh as TPM cycles instead of FED4_0XXXh, FED4_1XXXh, FED4_2XXXh, FED4_3XXXh, and FED4_4XXXh.
6:3	Reserved.
2	<b>TpmLegacy.</b> Read-write. Reset: 0. 1=Enable decoding of legacy TPM addresses: IO addresses 7Eh/7Fh and EEh/EFh.
1	Reserved.
0	<b>Tpm12En.</b> Read-write. Reset: 1. 1=Enable decoding of TPM cycles defined in TPM1. Enables decoding of TPM cycles defined in TPM 1.2 spec. Note that this bit and [TpmLegacy] are independent bits; they respectively turn on decoding of different TPM addresses.

**D14F3x090 [Wide IO 2] (FCH::ITF::LPC::WideIO2)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; D14F3x090

Bits	Description
------	-------------



31:16	Reserved.
15:0	<b>IOBaseAddress2: IO Base Address 2.</b> Read-write. Reset: 0000h. 16-bit PCI IO base address for wide generic IO port range. This function is enabled by FCH::ITF::LPC::IOMemPortDecodeEn[WideIO2Enable]. If FCH::ITF::LPC::AlternativeWideIORngEn[AlternativeWideIO2RangeEnable] == 1, the range is 16 bytes; else, the range is 512 bytes.

**D14F3x0A0 [SPI Base\_Addr] (FCH::ITF::LPC::SPIBaseAddr)**

Read-write. Reset: FEC1\_0002h.

\_aliasHOST; D14F3x0A0

Bits	Description
31:8	<b>Spi_eSpi_BaseAddr.</b> Read-write. Reset: FE_C100h. <b>Description:</b> This register defines the MMIO base address for the SPI ROM controller and eSPI host controller: SPI BAR = {SPI_BaseAddr[31:8],8'b0} eSPI BAR = {SPI_BaseAddr[31:8],8'b0} + 0x00010000h
7:5	Reserved.
4	<b>PspSpiMmioSel.</b> Read-write. Reset: 0. 0=SPI MMIO register for Host. 1=SPI MMIO register for PSP. PSP SPI MMIO select.
3	<b>RouteTpm2Spi.</b> Read-write. Reset: 0. 1=TPM cycles are routed to SPI bus with TPM_SPI_CS# asserted. Route TPM to SPI.
2	<b>AbortEnable.</b> Read-write. Reset: 0. LPC Abort enable.
1	<b>SpiRomEnable.</b> Read-write. Reset: 1. 0=SPI ROM is disabled. 1=SPI ROM is enabled if chip is strapped to SPI ROM.
0	<b>AltSpiCSEnable.</b> Read-write. Reset: 0. Alternative SPI CS enable.

**D14F3x0B0 [RomDmaSrcAddr] (FCH::ITF::LPC::RomDmaSrcAddr)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; D14F3x0B0

Bits	Description
31:6	<b>DmaStartAddr.</b> Read-write. Reset: 000_0000h. Specifies the starting DMA address to read from the ROM. NOTE: This is not the same as the legacy DMA function. This is meant to be used by BIOS to fetch the BOOT code quicker.
5:0	Reserved.

**D14F3x0B4 [RomDmaDstAddr] (FCH::ITF::LPC::RomDmaDstAddr)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; D14F3x0B4

Bits	Description
31:6	<b>DmaDstAddr.</b> Read-write. Reset: 000_0000h. Specifies the target DMA address to be written in the system memory. NOTE: This is not the same as the legacy DMA function. This is meant to be used by BIOS to fetch the BOOT code quicker.
5:0	Reserved.

**D14F3x0B8 [RomDmaControl/HostControl] (FCH::ITF::LPC::RomDmaCtlHostCtl)**

Reset: 0300\_0004h.

\_aliasHOST; D14F3x0B8

Bits	Description
31:28	Reserved.
27	<b>AutoSizeDone_pmio.</b> Read-write. Reset: 0. Init: BIOS,1. 1=Enable hand-instance of the pulse generator.
26	Reserved.
25	<b>LpcBusPullUpEn.</b> Read-write. Reset: 1. Init: BIOS,1. 0=Disable LAD[3:0] internal pull-up. 1=Enable LAD internal Pull-up. LPC Bus Pull-up enable.
24	<b>PrefetchEnSpiFromHost.</b> Read-write. Reset: 1. Init: BIOS,1. 1=SPI controller prefetches from the flash on

	behalf of the host. Host enabled SPI prefetch.
23:16	Reserved.
15:6	<b>DWCount.</b> Read-write. Reset: 000h. This register specifies the number (DWCount + 1) of cache lines (64 bytes) to be fetched from the ROM when the DMA is used.
5:3	Reserved.
2	<b>RomcpSupportAbRetry.</b> Read-write. Reset: 1. 0=ROM copy does not support AB retry. 1=ROM copy supports AB retry.
1	<b>DmaErrorStatus.</b> Read-only. Reset: 0. 0=Previous transfer has completed successfully. 1=Previous transfer has error. DmaErrorStatus.
0	<b>DmaStart.</b> Read, Write-1-only. Reset: 0. 1=LPC bridge starts the DMA function, with starting addresses defined by FCH::ITF::LPC::RomDmaSrcAddr and FCH::ITF::LPC::RomDmaDstAddr. This bit returns the status of the DMA transfer. 0=DMA transfer is complete. A return value of 1 means the DMA transfer is running.

**D14F3x0D0 [ClkCntrl] (FCH::ITF::LPC::ClkCntrl)**

Reset: 08FF\_E086h.

\_aliasHOST; D14F3x0D0

Bits	Description
31	<b>ClkRunEn.</b> Read-write. Reset: 0. 0=ClkRun function is disabled and LPCCLK0/LPCCLK1 can be running all the time. 1=ClkRun function is enabled and LPCCLK0/LPCCLK1 can be stopped. Should be set to 1 for mobile platforms to enhance energy savings.
30:24	<b>ClkRunDlyCounter.</b> Read-write. Reset: 08h. Specifies the amount of clocks to be extended before stopping the LPCCLK0/LPCCLK1.
23	Reserved.
22	<b>Lclk1ClkrunOvrid.</b> Read-write. Reset: 1. 0=LPCCLK1 is forced to run. 1=LPCCLK1 functions with CLKRUN protocol.
21	<b>Lclk0ClkrunOvrid.</b> Read-write. Reset: 1. 0=LPCCLK0 is forced to run. 1=LPCCLK0 is functioning with CLKRUN protocol.
20:15	Reserved.
14	<b>Lclk1En.</b> Read-write. Reset: 1. 0=LPCCLK1 is forced to stop. 1=LPCCLK1 is functioning with CLKRUN protocol.
13	<b>Lclk0En.</b> Read-write. Reset: 1. 0=LPCCLK0 is forced to stop. 1=LPCCLK0 is functioning with CLKRUN protocol.
12:8	Reserved.
7	<b>LpcClkRunEn.</b> Read-write. Reset: 1. 0=Disable. 1=Enable. LPC Clock run enable.
6:3	Reserved.
2	<b>SpiOnClkRun.</b> Read-write. Reset: 1. 0=SPI request can assert ClkRun#. 1=SPI request doesn't assert ClkRun#.
1:0	<b>ClkGateCntrl.</b> Read-write. Reset: 2h. These two bits control whether the LPC module allows clock gating to the internal core clock.
<b>Valid Values:</b>	
Value	Description
0h	Disable the clock gating function.
1h	Wait 16 clocks before allowing clock gating to the LPC module.
2h	Wait 64 clocks before allowing clock gating to the LPC module.
3h	Wait 256 clocks before allowing clock gating to the LPC module.

**D14F3x0D4 [ClkRunOption] (FCH::ITF::LPC::ClkRunOption)**

Read-write. Reset: 0000\_0040h.

\_aliasHOST; D14F3x0D4

Bits	Description
31:8	Reserved.
7:4	<b>MinAssertion.</b> Read-write. Reset: 4h. Specifies the minimum time of ClkRun# assertion. In units of 30 ns.

3:1	Reserved.
0	<b>ExtendClkRunB.</b> Read-write. Reset: 0. 0=Delay 30 ns before capturing ClkRun# input. 1=Delay 60 ns before capturing ClkRun# input.

### 9.2.9.2 SPI Registers

FCH SPI design is mode 0 only.

This field SpiReadMode[2:0] specifies the SPI read mode:

*Table 78: SpiReadMode[2:0]*

Bits	Definition
000b	Normal read (up to 33M)
001b	Reserved
010b	Dual IO (1-1-2)
011b	Quad IO (1-1-4)
100b	Dual IO (1-2-2)
101b	Quad IO (1-4-4)
110b	Normal read (up to 66M)
111b	Fast Read

The SPI configuration registers are accessed through SPI base address specified by FCH::ITF::LPC::SPIBaseAddr. Software can communicate with the SPI ROM through the default memory or an alternate programming (a.k.a "Indexed mode") method:

- Memory access to the BIOS ROM address space is automatically handled by the hardware. The SPI ROM controller translates the memory address onto the SPI bus and accesses the SPI ROM data. Any other commands besides memory read or memory write to the SPI ROM need to go through the alternate programming (a.k.a "Indexed mode") method. SPI ROM access through the memory address is limited to a 24-bit address (16MB addressable space).
- With the alternate programming (a.k.a "Indexed mode") method, software needs to program the SpiOpCode, SpiAddress, TxByteCount, RxByteCount, put the data into the transmit FIFO, and then execute the command. The hardware communicates with the SPI ROM using these parameters. This alternate programming (a.k.a "Indexed mode") method basically allows software to issue any flash vendor specific commands such as ERASE and STATUS. The alternate programming method can generate up to 32-bit addresses, but transfers are limited to "normal" (1-bit) mode.

### 9.2.9.3 Programming for ROM Protection register

#### 9.2.9.3.1 Index Mode (Indirect) Access

Opcode used in Index Mode determines if it is using 24-bit address mode or 32-bit address mode. For example:

OpCode = 0x03, 24-bit address Read command

= 0x13, 32-bit address Read command

= 0x02, 24-bit address byte program command

= 0x12, 32-bit address byte program command

##### 9.2.9.3.1.1 24-bit Address Index Mode

For Index mode, only address[23:0] are used for ROM protection address comparison. The address mapping of 24-bit

address Index mode is:

Address[23:16]: SPI\_regx80

Address[15:8]: SPI\_regx81

Address[7:0]: SPI\_regx82

### 9.2.9.3.2 FCH::ITF::SPI Registers

SPIx000 [SPI_Cntrl0] (FCH::ITF::SPI::SPICntrl0)	
Reset: 0FC0_0000h.	
_aliasHOST; SPIx000; SPI=FEC1_0000h	
Bits	Description
31	<b>SpiBusy.</b> Read-only. Reset: 0. 0=SPI bus is idle. 1=SPI bus is busy.
30:29	<b>SpiReadMode[2:1].</b> Read-write. Reset: 0h. <b>Description:</b> See Table 78 [SpiReadMode[2:0]]. NOTE: SPI modes supported are listed below, Old engine (spi_sie_org + spi_sie) support: <ol style="list-style-type: none"> <li>1-1-1 AltOpCode (Index Mode) Read</li> <li>1-1-1 AltOpCode (Index Mode) Write</li> <li>1-1-1 Host Read to ROM (command value is 03)</li> <li>1-1-1 Host fast Read to ROM (command value is 0B)</li> <li>1-1-2 Host Read to ROM (command value is programmable)</li> <li>1-1-4 Host Read to ROM (command value is programmable)</li> <li>1-2-2 Host Read to ROM (command value is programmable)</li> <li>1-4-4 Host Read to ROM (command value is programmable)</li> <li>1-1-1 TPM Read</li> <li>1-1-1 TPM Write</li> <li>All Writes except TPM-write should use AltOpCode (Index Mode).</li> <li>Host read prefetch</li> <li>TPM read prefetch</li> <li>ROM copy (DMA)</li> <li>SPI Clock speed 66/33/22/16 MHz</li> </ol> New engine (spi_sie_100) support: All the features listed above, plus 100 MHz clock speed.  PS: "1-1-1" means command, address and data are transmitted through 1 wire, 1 wire and 1 wire, respectively. "1-1-2" means command, address and data are transmitted through 1 wire, 1 wire and 2 wires, respectively. "1-1-4" means command, address and data are transmitted through 1 wire, 1 wire and 4 wires, respectively. "1-2-2" means command, address and data are transmitted through 1 wire, 2 wires and 2 wires, respectively. "1-4-4" means command, address and data are transmitted through 1 wire, 4 wires and 4 wires, respectively.
28	<b>SpiClkGate.</b> Read-write. Reset: 0. 1=Skip the 8th SPI clock at the end data when doing read.
27	<b>SpiBridgeDisable.</b> Read-write. Reset: 1. Setting this bit disables the SPI bridge mode (SB acts as a SPI-LPC bridge to the MAC).
26:24	<b>ArbWaitCount.</b> Read-write. Reset: 7h. Specifies the amount of wait time the SPI controller asserts HOLD# before it should access the SPI ROM, under ROM sharing mode with the MAC. This time is to allow the MAC to sample HOLD#.
23	<b>SpiHostAccessRomEn.</b> Read,Write-0-only. Reset: 1. 0=MAC cannot access BIOS ROM space (upper 512 KB). 1=MAC can access BIOS ROM space. This is a clear-once protection bit. Once set, some SPI registers can't be written and discards a SPI request if it is an illegal request.
22	<b>SpiAccessRomEn.</b> Read,Write-0-only. Reset: 1. 0=Software cannot access MAC's portion of the ROM space

	(lower 512 KB). 1=Software can access MAC's portion of the ROM space. This is a clear-once protection bit. Once set, some SPI registers can't be written and discards a SPI request if it is an illegal request.
21	<b>IllegalAccess</b> . Read-only. Reset: 0. 0=Legal index mode access. 1=Illegal index mode access.
20:19	Reserved.
18	<b>SpiReadMode[0]</b> . Read-write. Reset: 0. Bit[0] of SpiReadMode. See the definition of SpiReadMode[2:1] in this register. SpiReadMode = {SpiReadMode[2:1],SpiReadMode[0]}.
17:0	Reserved.

**SPIx004 [SPI\_RestrictedCmd] (FCH::ITF::SPI::SPIRestrictedCmd)**

Reset: 0000\_0000h.

\_aliasHOST; SPIx004; SPI=FEC1\_0000h

Bits	Description
31:24	<b>RestrictedCmd3</b> . Reset: 00h. Same as RestrictedCmd0. AccessType: (FCH::ITF::SPI::SPICntrl0[SpiAccessRomEn] && FCH::ITF::SPI::SPICntrl0[SpiHostAccessRomEn]) ? Read-write : Read-only.
23:16	<b>RestrictedCmd2</b> . Reset: 00h. Same as RestrictedCmd0. AccessType: (FCH::ITF::SPI::SPICntrl0[SpiAccessRomEn] && FCH::ITF::SPI::SPICntrl0[SpiHostAccessRomEn]) ? Read-write : Read-only.
15:8	<b>RestrictedCmd1</b> . Reset: 00h. Same as RestrictedCmd0. AccessType: (FCH::ITF::SPI::SPICntrl0[SpiAccessRomEn] && FCH::ITF::SPI::SPICntrl0[SpiHostAccessRomEn]) ? Read-write : Read-only.
7:0	<b>RestrictedCmd0</b> . Reset: 00h. This defines a restricted command issued by the MAC which is checked by the SB. If the opcode issued by the MAC matches with this register and the address space is in the BIOS space, this controller simply ignores the command for the case of bridge mode. For peer mode, the SPI controller stalls the entire interface as an attempt to stop that transaction. Note when either SpiAccessRomEn and/or SpiHostAccessRomEn bit are cleared, these registers become read-only and cannot be changed any more. AccessType: (FCH::ITF::SPI::SPICntrl0[SpiAccessRomEn] && FCH::ITF::SPI::SPICntrl0[SpiHostAccessRomEn]) ? Read-write : Read-only.

**SPIx008 [SPI\_RestrictedCmd2] (FCH::ITF::SPI::SPIRestrictedCmd2)**

Reset: 0000\_0000h.

\_aliasHOST; SPIx008; SPI=FEC1\_0000h

Bits	Description
31:24	<b>RestrictedCmdWoAddr2</b> . Reset: 00h. Same as [RestrictedCmdWoAddr0]. AccessType: (FCH::ITF::SPI::SPICntrl0[SpiAccessRomEn] && FCH::ITF::SPI::SPICntrl0[SpiHostAccessRomEn]) ? Read-write : Read-only.
23:16	<b>RestrictedCmdWoAddr1</b> . Reset: 00h. Same as [RestrictedCmdWoAddr0]. AccessType: (FCH::ITF::SPI::SPICntrl0[SpiAccessRomEn] && FCH::ITF::SPI::SPICntrl0[SpiHostAccessRomEn]) ? Read-write : Read-only.
15:8	<b>RestrictedCmdWoAddr0</b> . Reset: 00h. Same as FCH::ITF::SPI::SPIRestrictedCmd[RestrictedCmd0] except that this field defines a restricted command that does not have an address. AccessType: (FCH::ITF::SPI::SPICntrl0[SpiAccessRomEn] && FCH::ITF::SPI::SPICntrl0[SpiHostAccessRomEn]) ? Read-write : Read-only.
7:0	<b>RestrictedCmd4</b> . Reset: 00h. Same as FCH::ITF::SPI::SPIRestrictedCmd[RestrictedCmd0]. AccessType: (FCH::ITF::SPI::SPICntrl0[SpiAccessRomEn] && FCH::ITF::SPI::SPICntrl0[SpiHostAccessRomEn]) ? Read-write : Read-only.

**SPIx00C [SPI\_Cntrl1] (FCH::ITF::SPI::SPICntrl1)**

Reset: 0222\_0000h.

\_aliasHOST; SPIx00C; SPI=FEC1\_0000h

Bits	Description
31:24	<b>ByteCommand</b> . Reset: 02h. Specifies the command byte for the opcode transaction.

	AccessType: (FCH::ITF::SPI::SPICntrl0[SpiAccessRomEn] && FCH::ITF::SPI::SPICntrl0[SpiHostAccessRomEn]) ? Read-write : Read-only.								
23	<b>SpiDoLockOnNxtCmd.</b> Read-write. Reset: 0. SPI does a lock on the next command.								
22	<b>SpiDoUnLockOnNxtCmd.</b> Read-write. Reset: 0. SPI does an unlock on the next command.								
21:16	<b>WaitCount.</b> Read-write. Reset: 22h. Specifies the time, where units = 15 ns * (WaitCount + 1).								
	<b>ValidValues:</b>								
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>00h</td><td>15 ns.</td></tr> <tr> <td>01h</td><td>2 *15 ns (30 ns).</td></tr> <tr> <td>3Fh-02h</td><td>&lt;VALUE + 1&gt; * 15 ns.</td></tr> </table>	Value	Description	00h	15 ns.	01h	2 *15 ns (30 ns).	3Fh-02h	<VALUE + 1> * 15 ns.
Value	Description								
00h	15 ns.								
01h	2 *15 ns (30 ns).								
3Fh-02h	<VALUE + 1> * 15 ns.								
15:12	Reserved.								
11	<b>TrackMacLockEn.</b> Read-write. Reset: 0. When set, the controller will lock the SPI for the MAC when it has detected a command (from the MAC) matching the value defined in offset 10h or 11h. Conversely, it will unlock the bus when it has detected a command (from the MAC) matching the value defined in offset 12h or 13h.								
10:8	Reserved.								
7:0	<b>SpiParameters.</b> Read-write. Reset: 00h. This is the TX/RX FIFO port which can take up to 8 bytes. To send data to SPI ROM, software Writes data into this port. To retrieve data that are received from the SPI ROM, software Reads from this port.								

**SPIx010 [SPI\_CmdValue0] (FCH::ITF::SPI::SPICmdVal0)**

Reset: 0404\_2006h.

\_aliasHOST; SPIx010; SPI=FEC1\_0000h

Bits	Description
31:24	<b>MacUnlockCmd1.</b> Reset: 04h. Same as MacUnlockCmd0. AccessType: (FCH::ITF::SPI::SPICntrl0[SpiAccessRomEn] && FCH::ITF::SPI::SPICntrl0[SpiHostAccessRomEn]) ? Read-write : Read-only.
23:16	<b>MacUnlockCmd0.</b> Reset: 04h. This field is used to compare against the opcode sent out by the MAC. If FCH::ITF::SPI::SPICntrl1[TrackMacLockEn] == 1, the controller unlocks the SPI bus for the MAC. In other words, access by the CPU is allowed again. AccessType: (FCH::ITF::SPI::SPICntrl0[SpiAccessRomEn] && FCH::ITF::SPI::SPICntrl0[SpiHostAccessRomEn]) ? Read-write : Read-only.
15:8	<b>MacLockCmd1.</b> Reset: 20h. Same as MacLockCmd0. AccessType: (FCH::ITF::SPI::SPICntrl0[SpiAccessRomEn] && FCH::ITF::SPI::SPICntrl0[SpiHostAccessRomEn]) ? Read-write : Read-only.
7:0	<b>MacLockCmd0.</b> Reset: 06h. This is used to compare against the opcode sent out by the MAC. If FCH::ITF::SPI::SPICntrl1[TrackMacLockEn] == 1, the controller locks the SPI bus for the MAC. In other words, the MAC has the exclusive access to the ROM and access by the CPU is delayed until this is unlocked. This allows the MAC to do a certain sequence of operations without interruption. AccessType: (FCH::ITF::SPI::SPICntrl0[SpiAccessRomEn] && FCH::ITF::SPI::SPICntrl0[SpiHostAccessRomEn]) ? Read-write : Read-only.

**SPIx014 [SPI\_CmdValue1] (FCH::ITF::SPI::SPICmdVal1)**

Reset: 059F\_0406h.

\_aliasHOST; SPIx014; SPI=FEC1\_0000h

Bits	Description
31:24	<b>RDSR.</b> Reset: 05h. This is used to compare against the opcode sent out by the MAC. This is a predefined value to decode for the RDSR (Read Status Register) command from the MAC. AccessType: (FCH::ITF::SPI::SPICntrl0[SpiAccessRomEn] && FCH::ITF::SPI::SPICntrl0[SpiHostAccessRomEn]) ? Read-write : Read-only.
23:16	<b>RDID.</b> Reset: 9Fh. This is used to compare against the opcode sent out by the MAC. This is a predefined value to decode for the RDID (Read ID) command from the MAC.



	AccessType: (FCH::ITF::SPI::SPICntrl0[SpiAccessRomEn] && FCH::ITF::SPI::SPICntrl0[SpiHostAccessRomEn]) ? Read-write : Read-only.
15:8	<b>WRDI</b> . Reset: 04h. This is used to compare against the opcode sent out by the MAC. This is a predefined value to decode for the WRDI (Write Disable) command from the MAC.
	AccessType: (FCH::ITF::SPI::SPICntrl0[SpiAccessRomEn] && FCH::ITF::SPI::SPICntrl0[SpiHostAccessRomEn]) ? Read-write : Read-only.
7:0	<b>WREN</b> . Reset: 06h. This is used to compare against the opcode sent out by the MAC. This is a predefined value to decode for the WREN (Write Enable) command from the MAC.
	AccessType: (FCH::ITF::SPI::SPICntrl0[SpiAccessRomEn] && FCH::ITF::SPI::SPICntrl0[SpiHostAccessRomEn]) ? Read-write : Read-only.

**SPIx018 [SPI\_CmdValue2] (FCH::ITF::SPI::SPICmdVal2)**

Reset: 020A\_0B03h.

\_aliasHOST; SPIx018; SPI=FEC1\_0000h

Bits	Description
31:24	<b>BYTEWR</b> . Reset: 02h. This is used to compare against the opcode sent out by the MAC. This is a predefined value to decode for the BYTEWR (Byte Write) command from the MAC.
	AccessType: (FCH::ITF::SPI::SPICntrl0[SpiAccessRomEn] && FCH::ITF::SPI::SPICntrl0[SpiHostAccessRomEn]) ? Read-write : Read-only.
23:16	<b>PAGWR</b> . Reset: 0Ah. Page Write command.
	AccessType: (FCH::ITF::SPI::SPICntrl0[SpiAccessRomEn] && FCH::ITF::SPI::SPICntrl0[SpiHostAccessRomEn]) ? Read-write : Read-only.
15:8	<b>FRead</b> . Reset: 0Bh. This is used to compare against the opcode sent out by the MAC. This is a predefined value to decode for the FRead (Fast Read) command from the MAC.
	AccessType: (FCH::ITF::SPI::SPICntrl0[SpiAccessRomEn] && FCH::ITF::SPI::SPICntrl0[SpiHostAccessRomEn]) ? Read-write : Read-only.
7:0	<b>Read</b> . Reset: 03h. This is used to compare against the opcode sent out by the MAC. This is a predefined value to decode for the Read (Read Byte) command from the MAC.
	AccessType: (FCH::ITF::SPI::SPICntrl0[SpiAccessRomEn] && FCH::ITF::SPI::SPICntrl0[SpiHostAccessRomEn]) ? Read-write : Read-only.

**SPIx01C [Reserved] (FCH::ITF::SPI::Reserved)**

Read-write. Reset: FFh.

\_aliasHOST; SPIx01C; SPI=FEC1\_0000h

Bits	Description
7:0	Reserved.

**SPIx01D [Alt\_SPI\_CS] (FCH::ITF::SPI::AltSPICS)**

Reset: 00h.

\_aliasHOST; SPIx01D; SPI=FEC1\_0000h

Bits	Description
7	<b>SpiCsDlySel</b> . Read-write. Reset: 0. 0=75 ns minimum SPI_CS# de-assertion time. 1=125 ns minimum SPI_CS# de-assertion time.
6	Reserved.
5	<b>SpiProtectLock</b> . Read-write. Reset: 0. 1=Bits[3,4,5] are no longer writable.
4	<b>SpiProtectEn1</b> . Reset: 0. 1=Enable SPI protection to prevent host from accessing IMC and USB3 space.
	AccessType: FCH::ITF::SPI::AltSPICS[SpiProtectLock] ? Read-only : Read-write.
3	<b>SpiProtectEn0</b> . Reset: 0. 1=Enable SPI Read/Write protection ranges specified by FCH::ITF::LPC::RomProtect.
	AccessType: FCH::ITF::SPI::AltSPICS[SpiProtectLock] ? Read-only : Read-write.
2	<b>WriteBufferEn</b> . Read-write. Reset: 0. 1=SPI bridge can take burst write from the host and transfer it to the SPI flash. SPI write performance enhancement.



1:0	<b>AltSpiCsEn.</b> Reset: 0h. These two bits select the alternate SPI_CS# for BIOS_ROM.	
	AccessType: (FCH::ITF::SPI::SPICntrl0[SpiAccessRomEn] && FCH::ITF::SPI::SPICntrl0[SpiHostAccessRomEn]) ? Read-write : Read-only.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	BIOS ROM select SPI_CS1_L.

1h	BIOS ROM select SPI_CS2_L.
3h-2h	Reserved.

**SPIx020 [SPI100 Enable] (FCH::ITF::SPI::SPI100En)**

Read-write. Reset: 00h.

\_aliasHOST; SPIx020; SPI=FEC1\_0000h

Bits	Description
7:1	Reserved.
0	<b>UseSpi100.</b> Read-write. Reset: 0. 0=Does not support 100 MHz speed. 1=Supports 100 MHz speed. The actual Read speed also depends on FCH::ITF::SPI::SPI100SpeedCfg.

**SPIx022 [SPI100 Speed Config] (FCH::ITF::SPI::SPI100SpeedCfg)**

Read-write. Reset: 3133h.

\_aliasHOST; SPIx022; SPI=FEC1\_0000h

Bits	Description																
15:12	<b>NormSpeedNew[3:0].</b> Read-write. Reset: 3h. Configures the SPI bus normal speed in SPI100 engine. If the command is not using TpmSpeed and FastSpeed, it uses NormSpeed. <b>ValidValues:</b> <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>66.66 MHz.</td></tr> <tr> <td>1h</td><td>33.33 MHz.</td></tr> <tr> <td>2h</td><td>22.22 MHz.</td></tr> <tr> <td>3h</td><td>16.66 MHz.</td></tr> <tr> <td>4h</td><td>100 MHz.</td></tr> <tr> <td>5h</td><td>800 KHz.</td></tr> <tr> <td>Fh-6h</td><td>Reserved.</td></tr> </table>	Value	Description	0h	66.66 MHz.	1h	33.33 MHz.	2h	22.22 MHz.	3h	16.66 MHz.	4h	100 MHz.	5h	800 KHz.	Fh-6h	Reserved.
Value	Description																
0h	66.66 MHz.																
1h	33.33 MHz.																
2h	22.22 MHz.																
3h	16.66 MHz.																
4h	100 MHz.																
5h	800 KHz.																
Fh-6h	Reserved.																
11:8	<b>FastSpeedNew[3:0].</b> Read-write. Reset: 1h. <b>Description:</b> Configures the SPI bus speed for the following command in SPI100 engine: <ul style="list-style-type: none"> <li>FAST READ</li> <li>DDR READ (1-1-2)</li> <li>QDR READ (1-1-4)</li> <li>DPR READ (1-2-2)</li> <li>QPR READ (1-4-4)</li> </ul> <b>ValidValues:</b> <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>66.66 MHz.</td></tr> <tr> <td>1h</td><td>33.33 MHz.</td></tr> <tr> <td>2h</td><td>22.22 MHz.</td></tr> <tr> <td>3h</td><td>16.66 MHz.</td></tr> <tr> <td>4h</td><td>100 MHz.</td></tr> <tr> <td>5h</td><td>800 KHz.</td></tr> <tr> <td>Fh-6h</td><td>Reserved.</td></tr> </table>	Value	Description	0h	66.66 MHz.	1h	33.33 MHz.	2h	22.22 MHz.	3h	16.66 MHz.	4h	100 MHz.	5h	800 KHz.	Fh-6h	Reserved.
Value	Description																
0h	66.66 MHz.																
1h	33.33 MHz.																
2h	22.22 MHz.																
3h	16.66 MHz.																
4h	100 MHz.																
5h	800 KHz.																
Fh-6h	Reserved.																
7:4	<b>AltSpeedNew[3:0].</b> Read-write. Reset: 3h. Configures the SPI bus speed for the AltOpCode mode in SPI100 engine.																

	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	66.66 MHz.
	1h	33.33 MHz.
	2h	22.22 MHz.
	3h	16.66 MHz.
	4h	100 MHz.
	5h	800 KHz.
	Fh-6h	Reserved.
3:0	<b>TpmSpeedNew[3:0]</b> . Read-write. Reset: 3h. Configures the SPI bus speed for TPM Read and Write to the SPI100 engine.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	66.66 MHz.
	1h	33.33 MHz.
	2h	22.22 MHz.
	3h	16.66 MHz.
	4h	100 MHz.
	5h	800 KHz.
	Fh-6h	Reserved.

#### SPIx02C [SPI100 Host Prefetch Config] (FCH::ITF::SPI::SPI100HostPrefetchCfg)

Read-write. Reset: D4C0h.

- We have reached the maximum prefetch size defined in HostPrefetchSize and HostPrefOn64ByteBoundary register.
- When the host requests an address that is not already prefetched and not going to be prefetched shortly, we stop current prefetch action and re-start a new prefetch with the first address being the current address requested by the Host.
- When there is a ROM-Write or AltOpCode request from host, the on-going prefetch is terminated, and the prefetch buffer is flushed.
- When there is a TPM-Write, TPM-Read, USB-Read or EC-Read request, the on-going prefetch is halted. The contents of prefetch buffer is preserved so that the host can access them later.
- "Will Hit" algorithm. The current requested address is fetched "shortly" if this equation is true: Current Requested Address <= (First Prefetched Address + HostPrefetchSize).
- "Hit Soon" algorithm. The current requested address is fetched "shortly" if this equation is true: Current Requested Address <= (Last Prefetched Address + HostHitRange).

\_aliasHOST; SPIx02C; SPI=FEC1\_0000h

Bits	Description
15	<b>Rd4dw_en_host</b> . Read-write. Reset: 1. 1=Enable Host burst to 4 DWORD. Rd4dw_en_host.
14	<b>HostBurstEn</b> . Read-write. Reset: 1. 1=Enable Host buffer burst data out. Host burst enable.
13	<b>HostHitSoonEn</b> . Read-write. Reset: 0. 1=Enable the "Hit Soon" algorithm.
12	<b>HostWillHitEn</b> . Read-write. Reset: 1. 1=Enable the "Will Hit" algorithm. When "Will Hit" algorithm is enabled, "Hit Soon" algorithm is automatically disabled regardless of the setting in HostHitSoonEn.
11:8	<b>HostHitRange[3:0]</b> . Read-write. Reset: 4h. Configures the "Hit Range" in the "Hit Soon" algorithm.
7	<b>HostPrefOn64ByteBoundary</b> . Read-write. Reset: 1. 0=Always fetch 64 bytes no matter whether the first Host requested address lies on the 64-byte boundary or not. 1=Fetch 64 bytes if the first Host requested address lies on the 64-byte boundary, otherwise, get the 4 bytes the Host is currently requesting.
6:0	<b>HostPrefetchSize[6:0]</b> . Read-write. Reset: 40h. Configures the maximum prefetch byte count for the host prefetch buffer. The value has to be less than or equal to 64.

**SPIx040 [DDRCmdCode] (FCH::ITF::SPI::DDRCmdCode)**

Read-write. Reset: 3Bh.

\_aliasHOST; SPIx040; SPI=FEC1\_0000h

Bits	Description
7:0	<b>DDR_CMD</b> . Read-write. Reset: 3Bh. Double Data Rate command.

**SPIx041 [QDRCmdCode] (FCH::ITF::SPI::QDRCmdCode)**

Read-write. Reset: 6Bh.

\_aliasHOST; SPIx041; SPI=FEC1\_0000h

Bits	Description
7:0	<b>QDR_CMD</b> . Read-write. Reset: 6Bh. Quad Data Rate command.

**SPIx042 [DPRCmdCode] (FCH::ITF::SPI::DPRCmdCode)**

Read-write. Reset: BBh.

\_aliasHOST; SPIx042; SPI=FEC1\_0000h

Bits	Description
7:0	<b>DPR_CMD</b> . Read-write. Reset: BBh. DPR command.

**SPIx043 [QPRCmdCode] (FCH::ITF::SPI::QPRCmdCode)**

Read-write. Reset: EBh.

\_aliasHOST; SPIx043; SPI=FEC1\_0000h

Bits	Description
7:0	<b>QPR_CMD</b> . Read-write. Reset: EBh. QPR command.

**SPIx044 [ModeByte] (FCH::ITF::SPI::ModeByte)**

Read-write. Reset: 00h.

\_aliasHOST; SPIx044; SPI=FEC1\_0000h

Bits	Description
7:0	<b>ModeByte</b> . Read-write. Reset: 00h. SPI Read Mode.

**SPIx045 [CmdCode] (FCH::ITF::SPI::CmdCode)**

Read-write. Reset: 00h.

\_aliasHOST; SPIx045; SPI=FEC1\_0000h

Bits	Description
7:0	<b>SpiOpCode</b> . Read-write. Reset: 00h. Specifies the SPI opcode in alternate program method.

**SPIx047 [CmdTrigger] (FCH::ITF::SPI::CmdTrig)**

Write-1-only. Reset: 00h.

\_aliasHOST; SPIx047; SPI=FEC1\_0000h

Bits	Description
7	<b>Execute</b> . Write-1-only. Reset: 0. Trigger to execute command.
6:0	Reserved.

**SPIx048 [TxByteCount] (FCH::ITF::SPI::TxByteCnt)**

Read-write. Reset: 00h.

\_aliasHOST; SPIx048; SPI=FEC1\_0000h

Bits	Description
7:0	<b>TxByteCount</b> . Read-write. Reset: 00h. Number of bytes to be sent to SPI ROM.

**SPIx04B [RxByteCount] (FCH::ITF::SPI::RxByteCnt)**

Read-write. Reset: 00h.

\_aliasHOST; SPIx04B; SPI=FEC1\_0000h

Bits	Description
------	-------------

7:0	<b>RxByteCount.</b> Read-write. Reset: 00h. Number of bytes to be received from the SPI ROM.
-----	--

**SPIx04C [SpiStatus] (FCH::ITF::SPI::SpiStat)**

Read-only. Reset: 0000\_0000h.

\_aliasHOST; SPIx04C; SPI=FEC1\_0000h

Bits	Description
31	<b>SpiBusy.</b> Read-only. Reset: 0. 0=SPI bus is idle. 1=SPI bus is busy.
30:23	Reserved.
22:16	<b>FiFoRdPtr.</b> Read-only. Reset: 00h. The current Data FIFO read pointer.
15	Reserved.
14:8	<b>FiFoWrPtr.</b> Read-only. Reset: 00h. The current Data FIFO write pointer.
7:0	<b>DoneByteCount.</b> Read-only. Reset: 00h. Indicates how many bytes has been received or sent in the previous SPI transaction.

**SPIx080 [FIFO[70:0]] (FCH::ITF::SPI::FIFO)**

Read-write. Reset: 00h.

\_aliasHOST; SPIx080; SPI=FEC1\_0000h

Bits	Description
7:0	<b>FiFo.</b> Read-write. Reset: 00h. Contains the Data FIFO byte which is used in command mode to send or receive data.

**9.2.9.4 eSPI Registers**

The MMIO base address for accessing eSPI registers is defined in FCH::ITF::LPC::SPIBaseAddr.

**ESPIx00000000 (FCH::ITF::ESPI::DN\_TXHDR\_0th)**

Reset: 0000\_0000h.

\_aliasHOST; ESPIx00000000; ESPI=FEC2\_0000h

Bits	Description
31:24	<b>DNCMD_HDATA2.</b> Read-write. Reset: 00h. <b>Description:</b> The definition for this field is dependent upon SW_CMD_TYPE.  Independent channel command selected: - Reserved, should always be 00h.  Peripheral selected : Length[7:0].  VW selected: Reserved, should always be 00h.  OOB selected: Length[7:0].  FLASH selected: Length[7:0].
23:16	<b>DNCMD_HDATA1.</b> Read-write. Reset: 00h. <b>Description:</b> The definition for this field is dependent upon SW_CMD_TYPE.  Independent channel command selected: - Addres [7:0] of SET_CONFIGURATION/GET_CONFIGURATION. - Bit[1:0] needs to be 00b. - Note: In-Band command. These bits are ignored.  Peripheral selected:

	<ul style="list-style-type: none"> <li>- [23:20]: Tag</li> <li>- [19:16]: Length[11:8]</li> </ul> <p>VW selected: Reserved, should always be 00h.</p> <p>OOB selected:</p> <ul style="list-style-type: none"> <li>- [23:20]: Tag</li> <li>- [19:16]: Length[11:8]</li> </ul> <p>FLASH selected:</p> <ul style="list-style-type: none"> <li>- [23:20]: Tag</li> <li>- [19:16]: Length[11:8]</li> </ul>										
15:8	<p><b>DNCMD_HDATA0.</b> Read-write. Reset: 00h.</p> <p><b>Description:</b> The definition for this field is dependent upon SW_CMD_TYPE.</p> <p>Independent command selected:</p> <ul style="list-style-type: none"> <li>- Address [15:8] of SET_CONFIGURATION/GET_CONFIGURATION.</li> <li>- [15:12]: 0h</li> <li>- [11:8]: address[11:8]</li> <li>- Note: In-Band command. These bits are ignored.</li> </ul> <p>Peripheral selected:</p> <ul style="list-style-type: none"> <li>- a) Software programs this byte to be Message cycle type(0001xxxy) to instruct the eSPI controller to send down peripheral message with data (8 bytes + data byte N) or without data (total 8 bytes).</li> <li>- b) Software programs this byte to be unsuccessful completion to instruct the eSPI controller to send down unsuccessful completion.</li> </ul> <p>VW selected: It indicates the Virtual Wire Count is sent down. Bit[5:0] represents how many Virtual Wire groups to be communicated in the same packets.</p> <ul style="list-style-type: none"> <li>- NOTE: In the current design, it is limited to 16 groups (bit[5:4] = 00b), to save the registers needed.</li> </ul> <p>OOB selected: Software programs this byte to be 0x21 to instruct the eSPI controller to send down Tunneled SMBUS message to the slave.</p> <ul style="list-style-type: none"> <li>- Software prgrames this byte to be CycleType for Flash Completion, including Cpl/Unsuccessful Cpl/CplID.</li> </ul>										
7:6	Reserved.										
5:4	<p><b>SLAVE_SEL.</b> Read-write. Reset: 0h. Slave N selected.</p> <p><b>ValidValues:</b></p> <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>Slave0.</td></tr> <tr> <td>3h-1h</td><td>Reserved.</td></tr> </table>	Value	Description	0h	Slave0.	3h-1h	Reserved.				
Value	Description										
0h	Slave0.										
3h-1h	Reserved.										
3	<p><b>DNCMD_STATUS.</b> Read,Write-0-only. Reset: 0. 0=Clear: Hardware will automatically clear this bit after the packet is sent down. 1=Set: The bit needs to be set last by software after all eSPI specific registers are all programmed to inform the protocoal layer to send down command or packet. Downstream command status. Cleared when Writting to 0.</p>										
2:0	<p><b>DNCMD_TYPE.</b> Read-write. Reset: 0h. TX Command Type.</p> <p><b>ValidValues:</b></p> <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>Set Configuration (Independent command).</td></tr> <tr> <td>1h</td><td>Get Configuration (Independent command).</td></tr> <tr> <td>2h</td><td>In-band RESET command (Independent command).</td></tr> <tr> <td>3h</td><td>Peripheral Unsuccesful Cpl down stream.</td></tr> </table>	Value	Description	0h	Set Configuration (Independent command).	1h	Get Configuration (Independent command).	2h	In-band RESET command (Independent command).	3h	Peripheral Unsuccesful Cpl down stream.
Value	Description										
0h	Set Configuration (Independent command).										
1h	Get Configuration (Independent command).										
2h	In-band RESET command (Independent command).										
3h	Peripheral Unsuccesful Cpl down stream.										

4h	Peripheral Channel message down stream.
5h	VW Channel down stream.
6h	OOB Channel down stream.
7h	Flash Channel Cpl/CplD/Unsuccessful Cpl down stream.

**ESPIx00000004 (FCH::ITF::ESPI::DN\_TXHDR\_1)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; ESPIx00000004; ESPI=FEC2\_0000h

Bits	Description
31:24	<b>DNCMD_HDATA6.</b> Read-write. Reset: 00h. <b>Description:</b> The definition for this field is dependent upon SW_CMD_TYPE.  Indipendent channel command selected: Data[31:24].  Peripheral selected: Message specific byte[2].  VW selected: Reserved, should always be 00h.  OOB selected: Reserved, should always be 00h.  FLASH selected: Reserved, should always be 00h.
23:16	<b>DNCMD_HDATA5.</b> Read-write. Reset: 00h. <b>Description:</b> The definition for this field is dependent upon SW_CMD_TYPE.  Indipendent channel command selected: Data[23:16].  Peripheral selected: Message specific byte[1].  VW selected: Reserved, should always be 00h.  OOB selected: - SMBus Byte Count. Need to program to be not greater than 128 bytes.  FLASH selected: Reserved, shouls always be 00h.
15:8	<b>DNCMD_HDATA4.</b> Read-write. Reset: 00h. <b>Description:</b> The definition for this field is dependent upon SW_CMD_TYPE.  Indipendent channel command selected: - Data[15:8].  Peripheral selected: Message specific byte[0].  VW selected: Reserved, should always be 00h.  OOB selected: SMBus Command Op Code.  FLASH selected: Reserved, should always be 00h.
7:0	<b>DNCMD_HDATA3.</b> Read-write. Reset: 00h. <b>Description:</b> The definition for this field is dependent upon SW_CMD_TYPE.  Indipendent channel command selected: Data[7:0].

	Peripheral selected: Message code [7:0].
	VW selected: Reserved, should always be 00h.
	OOB selected: - SMBus Slave Address. Bit[0] needs to be programmed to 1.
	FLASH selected: Reserved, should always be 00h.

**ESPIx00000008 (FCH::ITF::ESPI::DN\_TXHDR\_2)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; ESPIx00000008; ESPI=FEC2\_0000h

Bits	Description
31:8	Reserved.
7:0	<b>DNCMD_HDATA7.</b> Read-write. Reset: 00h. <b>Description:</b> The definition for this field is dependent upon SW_CMD_TYPE.  Independent channel command selected: Reserved, should always be 00h.  Peripheral selected: Message specific byte[3].  VW selected: Reserved, should always be 00h.  OOB selected: Reserved, should always be 00h.  FLASH selected: Reserved, should always be 00h.

**ESPIx0000000C (FCH::ITF::ESPI::DN\_TXDATA\_PORT)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; ESPIx0000000C; ESPI=FEC2\_0000h

Bits	Description
31:24	<b>DN_TXDATA_B3.</b> Read-write. Reset: 00h. <b>Description:</b> The definition for this field is dependent upon SW_CMD_TYPE.  Independent channel command selected: Reserved, should always be 00h.  Peripheral selected: Message Data DWn[31:24].  VW selected: VW Index Group (2n + 1) data.  OOB selected: OOB Message DWn[31:24].  FLASH selected: Flash Cpl Data DWn[31:24].
23:16	<b>DN_TXDATA_B2.</b> Read-write. Reset: 00h. <b>Description:</b> The definition for this field is dependent upon SW_CMD_TYPE.  Independent channel command selected: Reserved, should always be 00h.  Peripheral selected: Message Data DWn[23:16].  VW selected: VW Index Group (2n + 1).  OOB selected: OOB Message DWn[23:16].



	FLASH selected: Flash Cpl Data DWn[23:16].
15:8	<b>DN_TXDATA_B1.</b> Read-write. Reset: 00h. <b>Description:</b> The definition for this field is dependent upon SW_CMD_TYPE.  Indipendent channel command selected: Reserved, should always be 00h.  Peripheral selected: Message Data DWn[15:8].  VW selected: VW Index Group 2n Data.  OOB selected: OOB Message DWn[15:8].  FLASH selected: Flash Cpl Data DWn[15:8].
7:0	<b>DN_TXDATA_B0.</b> Read-write. Reset: 00h. <b>Description:</b> The definition for this field is dependent upon SW_CMD_TYPE.  Indipendent channel command selected: Reserved, should always be 00h.  Peripheral selected: Message Data DWn[7:0].  VW selected: VW Index Group 2n.  OOB selected: OOB Message DWn[7:0].  FLASH selected: Flash Cpl Data DWn[7:0].

**ESPIx00000010 (FCH::ITF::ESPI::UP\_RXHDR\_0)**

Reset: 0000\_0000h.

\_aliasHOST; ESPIx00000010; ESPI=FEC2\_0000h

Bits	Description						
31:24	<b>UPCMD_HDATA2.</b> Read-only. Reset: 00h. <b>Description:</b> RX_LOW_LEN: This field stores the Length[7:0] from GET_FLASH_NP/GET_OOB. - Bits[31:24]: Length[7:0].						
23:16	<b>UPCMD_HDATA1.</b> Read-only. Reset: 00h. <b>Description:</b> RX_TAG_LEN: This field stores the Tag and Length[11:8] from which the eSPI packet recieved by GET_FLASH_NP/GET_OOB. - Bits[23:20]: Tag. - Bits[19:16]: Length[11:8].						
15:8	<b>UPCMD_HDATA0.</b> Read-only. Reset: 00h. Cycle Type: This field stores the cycle type from GET_FLASH_NP and GET_OOB.						
7:6	Reserved.						
5:4	<b>SLAVE_SEL.</b> Read-only. Reset: 0h. Slave N receive select. <b>ValidValues:</b>						
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>The upstream packet is from Slave0.</td></tr> <tr> <td>3h-1h</td><td>Reserved.</td></tr> </table>	Value	Description	0h	The upstream packet is from Slave0.	3h-1h	Reserved.
Value	Description						
0h	The upstream packet is from Slave0.						
3h-1h	Reserved.						
3	<b>UPCMD_STATUS.</b> Read,Write-1-to-clear. Reset: 0. 0=OOB message packet or Flash request packet not received. 1=This bit will be set after OOB message packet or Flash request packet is recieved, and eSPI will not send down another GET_OOB or GET_FLASH_NP before the Valid bit. Cleared by software. Upstream						

	command status. Valid bit Status. This bit can be cleared by software writing 1 to this field.								
2:0	<b>UPCMD_TYPE</b> . Read-only. Reset: 0h. Upstream command type.								
	<b>ValidValues:</b>								
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>Flash Channel Request (GET_FLASH_NP).</td></tr> <tr> <td>1h</td><td>Upstream OOB message (GET_OOB).</td></tr> <tr> <td>7h-2h</td><td>Reserved.</td></tr> </table>	Value	Description	0h	Flash Channel Request (GET_FLASH_NP).	1h	Upstream OOB message (GET_OOB).	7h-2h	Reserved.
Value	Description								
0h	Flash Channel Request (GET_FLASH_NP).								
1h	Upstream OOB message (GET_OOB).								
7h-2h	Reserved.								

**ESPIx00000014 (FCH::ITF::ESPI::UP\_RXHDR\_1)**

Read-only. Reset: 0000\_0000h.

\_aliasHOST; ESPIx00000014; ESPI=FEC2\_0000h

Bits	Description
31:24	<b>UPCMD_HDATA6</b> . Read-only. Reset: 00h. <b>Description:</b> The definition for this field is dependent upon RX_REQ_TYPE. - OOB selected: Reserved. - FLASH selected: Address[7:0].
23:16	<b>UPCMD_HDATA5</b> . Read-only. Reset: 00h. <b>Description:</b> The definition for this field is dependent upon RX_REQ_TYPE. - OOB selected: SMBus Byte Count. - FLASH selected: Address[15:8].
15:8	<b>UPCMD_HDATA4</b> . Read-only. Reset: 00h. <b>Description:</b> The definition for this field is dependent upon RX_REQ_TYPE. - OOB selected: SMBus Command Opcode. - FLASH selected: Address[23:16].
7:0	<b>UPCMD_HDATA3</b> . Read-only. Reset: 00h. <b>Description:</b> The definition for this field is dependent upon RX_REQ_TYPE. - OOB selected: SMBus Slave Address. - FLASH selected: Address[31:24].

**ESPIx00000018 (FCH::ITF::ESPI::UP\_RXDATA\_PORT)**

Read-only. Reset: 0000\_0000h.

\_aliasHOST; ESPIx00000018; ESPI=FEC2\_0000h

Bits	Description
31:0	<b>UP_RXDATA</b> . Read-only. Reset: 0000_0000h. Receive data.

**ESPIx0000001C (FCH::ITF::ESPI::RESERVED\_REG0)**

Reset: 0000\_0000h.

\_aliasHOST; ESPIx0000001C; ESPI=FEC2\_0000h

Bits	Description
31:0	Reserved.

**ESPIx00000020 (FCH::ITF::ESPI::RESERVED\_REG1)**

Reset: 0000\_0000h.

\_aliasHOST; ESPIx00000020; ESPI=FEC2\_0000h

Bits	Description
31:0	Reserved.

**ESPIx00000024 (FCH::ITF::ESPI::RESERVED\_REG2)**

Reset: 0000\_0000h.

\_aliasHOST; ESPIx00000024; ESPI=FEC2\_0000h

Bits	Description
31:0	Reserved.

**ESPIx0000002C (FCH::ITF::ESPI::MASTER\_CAP)**

Read-only. Reset: Fixed,E649\_E91Fh.

\_aliasHOST; ESPIx0000002C; ESPI=FEC2\_0000h

Bits	Description												
31	<b>CRC_CHECK_SUPPORT.</b> Read-only. Reset: Fixed,1. 0=Master supports CRC checking. 1=Master doesn't support CRC checking. CRC checking supported by Master.												
30	<b>ALERT_MODE_SUPPORT.</b> Read-only. Reset: Fixed,1. 0=IO[1] pin is used to signal the Alert event. 1=A dedicated Alert# pin is used to signal the Alert event, or IO[1] pin used for Alert. Alert mode supported by the Master.												
29:28	<b>IO_MODE_SUPPORT.</b> Read-only. Reset: Fixed,2h. IO Mode support by Controller, Quad mode, Dual mode, single mode <b>ValidValues:</b> <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>Single mode.</td></tr> <tr> <td>1h</td><td>Dual mode and Single mode.</td></tr> <tr> <td>2h</td><td>Quad mode, Dual mode and Single mode.</td></tr> <tr> <td>3h</td><td>Reserved.</td></tr> </table>	Value	Description	0h	Single mode.	1h	Dual mode and Single mode.	2h	Quad mode, Dual mode and Single mode.	3h	Reserved.		
Value	Description												
0h	Single mode.												
1h	Dual mode and Single mode.												
2h	Quad mode, Dual mode and Single mode.												
3h	Reserved.												
27:25	<b>CLK_FREQ_SUPPORT.</b> Read-only. Reset: Fixed,3h. Operating frequency supported. <b>ValidValues:</b> <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>16.7 MHz.</td></tr> <tr> <td>1h</td><td>16.7 MHz, 33 MHz.</td></tr> <tr> <td>2h</td><td>Reserved.</td></tr> <tr> <td>3h</td><td>16.7 MHz, 33 MHz and 66 MHz.</td></tr> <tr> <td>7h-4h</td><td>Reserved.</td></tr> </table>	Value	Description	0h	16.7 MHz.	1h	16.7 MHz, 33 MHz.	2h	Reserved.	3h	16.7 MHz, 33 MHz and 66 MHz.	7h-4h	Reserved.
Value	Description												
0h	16.7 MHz.												
1h	16.7 MHz, 33 MHz.												
2h	Reserved.												
3h	16.7 MHz, 33 MHz and 66 MHz.												
7h-4h	Reserved.												
24:22	<b>SLAVE_NUM.</b> Read-only. Reset: Fixed,1h. Indicates the number of slaves. A value of 1 indicates one slave supported, and is the minimum requirement. A value of 0 indicates 8 slaves, and is the masimum supported number. <b>ValidValues:</b> <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>8 slaves (maximum supported).</td></tr> <tr> <td>1h</td><td>1 slave (minimum supported).</td></tr> <tr> <td>7h-2h</td><td>&lt;VALUE&gt; slaves supported.</td></tr> </table>	Value	Description	0h	8 slaves (maximum supported).	1h	1 slave (minimum supported).	7h-2h	<VALUE> slaves supported.				
Value	Description												
0h	8 slaves (maximum supported).												
1h	1 slave (minimum supported).												
7h-2h	<VALUE> slaves supported.												
21:19	<b>PR_MAX_SIZE.</b> Read-only. Reset: Fixed,1h. Peripheral Channel maximum payload size supported. The payload of the transaction must not cross the naturally aligned address boundary of the corresponding maximum payload size. <b>ValidValues:</b> <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>Reserved.</td></tr> <tr> <td>1h</td><td>64-byte address aligned maximum payload size.</td></tr> <tr> <td>7h-2h</td><td>Reserved.</td></tr> </table>	Value	Description	0h	Reserved.	1h	64-byte address aligned maximum payload size.	7h-2h	Reserved.				
Value	Description												
0h	Reserved.												
1h	64-byte address aligned maximum payload size.												
7h-2h	Reserved.												
18:13	<b>VW_MAX_SIZE.</b> Read-only. Reset: Fixed,0Fh. Operating maximum Virtual Wire Count supported. The maximum number of Virtual Wire groups that can be sent in a single Virtual Wire packet. This is a 0-based count. The default value of 0 indicates a count of 1.												
12:10	<b>OOB_MAX_SIZE.</b> Read-only. Reset: Fixed,2h. OOB Message Channel maximum payload size supported. <b>ValidValues:</b> <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>Reserved.</td></tr> <tr> <td>1h</td><td>64-byte maximum payload size.</td></tr> </table>	Value	Description	0h	Reserved.	1h	64-byte maximum payload size.						
Value	Description												
0h	Reserved.												
1h	64-byte maximum payload size.												

	2h	128-byte maximum payload size.
	7h-3h	Reserved.
9:7	<b>FLASH_MAX_SIZE.</b> Read-only. Reset: Fixed,2h. Flash Access Channel maximum payload size supported.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	Reserved.
	1h	64-byte maximum payload size.
	2h	128-byte maximum payload size.
	7h-3h	Reserved.
6:4	<b>ESPI_VERSION.</b> Read-only. Reset: Fixed,1h. eSPI version.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	Master supports eSPI 0.7 version.
	1h	Master supports eSPI 0.75 version.
	7h-2h	Reserved.
3	<b>PR_SUPPORT.</b> Read-only. Reset: Fixed,1. 0=Not supported. 1=Supported. Peripheral Channel support by Master.	
2	<b>VW_SUPPORT.</b> Read-only. Reset: Fixed,1. 0=Not supported. 1=Supported. Virtual Wire Channel support by Master.	
1	<b>OOB_SUPPORT.</b> Read-only. Reset: Fixed,1. 0=Not supported. 1=Supported. OOB Message Channel support by Master.	
0	<b>FLASH_SUPPORT.</b> Read-only. Reset: Fixed,1. 0=Not supported. 1=Supported. Flash Access Channel support by Master.	

**ESPIx00000030 (FCH::ITF::ESPI::GLOBAL\_CONTROL\_0)**

Read-write. Reset: 0000_0008h.		
_aliasHOST; ESPIx00000030; ESPI=FEC2_0000h		
Bits	Description	
31:30	Reserved.	
29:24	WAIT_CNT. Read-write. Reset: 00h. Specifies the timeout count for the wait state.	
23:8	WDG_CNT. Read-write. Reset: 0000h. Specifies the timeout retry count for PCI downstream retries.	
7	Reserved.	
6:4	AL_IDLE_TIMER. Read-write. Reset: 0h. Selects the idle timer timeout value. Once the idle timer reaches the timeout value and AL_STOP_EN == 1, eSPI will output ESPI_STOP_AIClk to do global Alink clock gating.	
	ValidValues:	
	Value	Description
	0h	16 clocks.
	1h	32 clocks.
	2h	64 clocks.
	3h	128 clocks.
	4h	256 clocks.
	5h	512 clocks.
	6h	1024 clocks.
7h	2048 clocks.	
3	AL_STOP_EN. Read-write. Reset: 1. 0=Disable. 1=Enable. Global Alink clock gating enable. set this bit to enable the eSPI to generate ESPI_STOP_AIClk to do global clock gating once the global Alink Idle Timer reaches the timeout value.	
2	PR_CLKGAT_EN. Read-write. Reset: 0. 0=Disable. 1=Enable. Peripheral clock gating enable. Set this bit to enable peripheral block to dynamically clock gate once the Slave peripheral channel is disabled.	

1	<b>WAIT_CHKEN.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. Wait State Control enable. Set this bit to enable the Wait State counter during eSPI bus turn around.
0	<b>WDG_EN.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. Watchdog enable. Set this bit to enable the watchdog counter for all the PCI downstream transactions for eSPI.

**ESPIx00000034 (FCH::ITF::ESPI::GLOBAL\_CONTROL\_1)**

Read-write. Reset: 0002\_FF00h.

\_aliasHOST; ESPIx00000034; ESPI=FEC2\_0000h

Bits	Description																
31:18	Reserved.																
17:13	<b>RGCMD_INT_MAP.</b> Read-write. Reset: 17h. Register Command interrupt mapping, When Register command (Downstream/Upstream peripheral message, Downstream/Upstream OOB, Downstream VW, Channel Independent command) has finished, eSPI controller generated interrupt will map to the interrupt pin according to the following register setting. <b>ValidValues:</b> <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>00h</td><td>IRQ0</td></tr> <tr> <td>01h</td><td>IRQ1</td></tr> <tr> <td>02h</td><td>IRQ2</td></tr> <tr> <td>16h-03h</td><td>IRQ&lt;VALUE&gt;</td></tr> <tr> <td>17h</td><td>IRQ23</td></tr> <tr> <td>1Eh-18h</td><td>Reserved.</td></tr> <tr> <td>1Fh</td><td>SMI#</td></tr> </table>	Value	Description	00h	IRQ0	01h	IRQ1	02h	IRQ2	16h-03h	IRQ<VALUE>	17h	IRQ23	1Eh-18h	Reserved.	1Fh	SMI#
Value	Description																
00h	IRQ0																
01h	IRQ1																
02h	IRQ2																
16h-03h	IRQ<VALUE>																
17h	IRQ23																
1Eh-18h	Reserved.																
1Fh	SMI#																
12:8	<b>ERR_INT_MAP.</b> Read-write. Reset: 1Fh. Error interrupt mapping. When a Slave transaction error has happened, and the error interrupt enable has been set, the error interrupt will map to the interrupt pin according to the following register setting. <b>ValidValues:</b> <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>00h</td><td>IRQ0</td></tr> <tr> <td>01h</td><td>IRQ1</td></tr> <tr> <td>02h</td><td>IRQ2</td></tr> <tr> <td>16h-03h</td><td>IRQ&lt;VALUE&gt;</td></tr> <tr> <td>17h</td><td>IRQ23</td></tr> <tr> <td>1Eh-18h</td><td>Reserved.</td></tr> <tr> <td>1Fh</td><td>SMI#</td></tr> </table>	Value	Description	00h	IRQ0	01h	IRQ1	02h	IRQ2	16h-03h	IRQ<VALUE>	17h	IRQ23	1Eh-18h	Reserved.	1Fh	SMI#
Value	Description																
00h	IRQ0																
01h	IRQ1																
02h	IRQ2																
16h-03h	IRQ<VALUE>																
17h	IRQ23																
1Eh-18h	Reserved.																
1Fh	SMI#																
7:5	Reserved.																
4:3	<b>SUB_DECODE_SLV.</b> Read-write. Reset: 0h. Selects which slave to do subtractive decode. <b>ValidValues:</b> <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>Slave0.</td></tr> <tr> <td>1h</td><td>Slave1.</td></tr> <tr> <td>2h</td><td>Slave2.</td></tr> <tr> <td>3h</td><td>Slave3.</td></tr> </table>	Value	Description	0h	Slave0.	1h	Slave1.	2h	Slave2.	3h	Slave3.						
Value	Description																
0h	Slave0.																
1h	Slave1.																
2h	Slave2.																
3h	Slave3.																
2	<b>SUB_DECODE_EN.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. Enable eSPI to do subtractive decode.																
1	<b>BUS_MASTER_EN.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. Bus Master enable. Enables eSPI Upstream Memory cycle posting.																
0	<b>SW_RST.</b> Read-write. Reset: 0. 0=No effect. 1=Controller reset. Set the bit to perform global controller resets for the eSPI controller. All the state machines will return to idle and all the requests will be flushed. All the																

configuration registers will reset to default values and software needs to send In-Band Resets to each Slave device after the controller reset so that both Master and Slave run in the same configuration mode.

#### ESPIx00000044 (FCH::ITF::ESPI::SLAVE0\_IO\_BASE\_REG0)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; ESPIx00000044; ESPI=FEC2\_0000h

Bits	Description
31:16	<b>RANGE1.</b> Read-write. Reset: 0000h. IO decode base address for Range 1.
15:0	<b>RANGE0.</b> Read-write. Reset: 0000h. IO decode base address for Range 0.

#### ESPIx00000048 (FCH::ITF::ESPI::SLAVE0\_IO\_BASE\_REG1)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; ESPIx00000048; ESPI=FEC2\_0000h

Bits	Description
31:16	<b>RANGE3.</b> Read-write. Reset: 0000h. IO decode base address for Range 3.
15:0	<b>RANGE2.</b> Read-write. Reset: 0000h. IO decode base address for Range 2.

#### ESPIx0000004C (FCH::ITF::ESPI::SLAVE0\_IO\_SIZE)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; ESPIx0000004C; ESPI=FEC2\_0000h

Bits	Description
31:24	<b>RANGE3.</b> Read-write. Reset: 00h. Programmable IO Range3 size.
23:16	<b>RANGE2.</b> Read-write. Reset: 00h. Programmable IO Range2 size.
15:8	<b>RANGE1.</b> Read-write. Reset: 00h. Programmable IO Range1 size.
7:0	<b>RANGE0.</b> Read-write. Reset: 00h. Programmable IO Range0 size.

#### ESPIx00000050 (FCH::ITF::ESPI::SLAVE0\_MMIO\_BASE\_REG0)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; ESPIx00000050; ESPI=FEC2\_0000h

Bits	Description
31:0	<b>RANGE0.</b> Read-write. Reset: 0000_0000h. MMIO decode base address for Range 0.

#### ESPIx00000054 (FCH::ITF::ESPI::SLAVE0\_MMIO\_BASE\_REG1)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; ESPIx00000054; ESPI=FEC2\_0000h

Bits	Description
31:0	<b>RANGE1.</b> Read-write. Reset: 0000_0000h. MMIO decode base address for Range 1.

#### ESPIx00000058 (FCH::ITF::ESPI::SLAVE0\_MMIO\_BASE\_REG2)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; ESPIx00000058; ESPI=FEC2\_0000h

Bits	Description
31:0	<b>RANGE2.</b> Read-write. Reset: 0000_0000h. MMIO decode base address for Range 2.

#### ESPIx0000005C (FCH::ITF::ESPI::SLAVE0\_MMIO\_BASE\_REG3)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; ESPIx0000005C; ESPI=FEC2\_0000h

Bits	Description
31:0	<b>RANGE3.</b> Read-write. Reset: 0000_0000h. MMIO decode base address for Range 3.

#### ESPIx00000060 (FCH::ITF::ESPI::SLAVE0\_MMIO\_SIZE\_REG0)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; ESPIx00000060; ESPI=FEC2\_0000h

Bits	Description
31:16	<b>RANGE1.</b> Read-write. Reset: 0000h. Programmable MMIO Range1 size.
15:0	<b>RANGE0.</b> Read-write. Reset: 0000h. Programmable MMIO Range0 size.

**ESPIx00000064 (FCH::ITF::ESPI::SLAVE0\_MMIO\_SIZE\_REG1)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; ESPIx00000064; ESPI=FEC2\_0000h

Bits	Description
31:16	<b>RANGE3.</b> Read-write. Reset: 0000h. Programmable MMIO Range3 size.
15:0	<b>RANGE2.</b> Read-write. Reset: 0000h. Programmable MMIO Range2 size.

**ESPIx00000068 (FCH::ITF::ESPI::SLAVE0\_CONFIG)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; ESPIx00000068; ESPI=FEC2\_0000h

Bits	Description										
31	<b>CRC_CHECK_EN.</b> Read-write. Reset: 0. 0=CRC checking is disabled. 1=CRC checking is enabled. This bit is set to 1 by the eSPI Master to enable CRC checking on the eSPI bus. By default, CRC checking is disabled.										
30	<b>ALERT_MODE_SEL.</b> Read-write. Reset: 0. 0=IO bit[1] pin is used to signal the Alert event. 1=A dedicated Alert# pin is used to signal the Alert event. This bit serves to configure the Alert mechanism used by the slave to initiate a transaction on the eSPI interface. Note: This bit can only be 0 in a single Master-single Slave topology. For single Master-multiple Slave topology, this bit must be programmed to 1.										
29:28	<b>IO_MODE_SEL.</b> Read-write. Reset: 0h. IO Mode Select. eSPI Master programs this field to enable the appropriate mode of operation, which takes effect at the de-assertion edge of the Chip Select#. The IO Mode configured in this field must be supported by both the Master and Slave. Single IO mode is supported by default. <b>ValidValues:</b>										
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>Single IO.</td></tr> <tr> <td>1h</td><td>Dual IO.</td></tr> <tr> <td>2h</td><td>Quad IO.</td></tr> <tr> <td>3h</td><td>Reserved.</td></tr> </table>	Value	Description	0h	Single IO.	1h	Dual IO.	2h	Quad IO.	3h	Reserved.
Value	Description										
0h	Single IO.										
1h	Dual IO.										
2h	Quad IO.										
3h	Reserved.										
27:25	<b>CLK_FREQ_SEL.</b> Read-write. Reset: 0h. Identifies the operating frequency. <b>ValidValues:</b>										
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>16.6 MHz.</td></tr> <tr> <td>1h</td><td>33 MHz.</td></tr> <tr> <td>2h</td><td>66 MHz.</td></tr> <tr> <td>7h-3h</td><td>Reserved.</td></tr> </table>	Value	Description	0h	16.6 MHz.	1h	33 MHz.	2h	66 MHz.	7h-3h	Reserved.
Value	Description										
0h	16.6 MHz.										
1h	33 MHz.										
2h	66 MHz.										
7h-3h	Reserved.										
24:4	Reserved.										
3	<b>PR_EN.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. Peripheral Channel enable. This bit is set to 1 by the eSPI Master to enable the Peripheral Channel.										
2	<b>VW_EN.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. Virtual Wire Channel enable. This bit is set to 1 by the eSPI Master to enable the Virtual Wire Channel.										
1	<b>OOB_EN.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. OOB Message Channel enable. This bit is set to 1 by the eSPI Master to enable the OOB Message Channel.										
0	<b>FLASH_EN.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. Flash Access Channel enable. This bit is set to 1 by the eSPI Master to enable the Flash Access Channel.										

**ESPIx0000006C (FCH::ITF::ESPI::SLAVE0\_INT\_EN)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; ESPIx0000006C; ESPI=FEC2\_0000h

Bits	Description
------	-------------



31	<b>FLASH_REQ_INT_EN</b> . Read-write. Reset: 0. 0=Disable. 1=Enable. Flash Request Received enable. Enables the generation of a command interrupt when an Upstream Flash Request is recieved and valid to Read.
30	<b>RXOOB_INT_EN</b> . Read-write. Reset: 0. 0=Disable. 1=Enable. OOB Message Received enable. Enables the generation of a command interrupt when an Upstream OOB Message is recieved and valid to Read.
29	<b>RXMSG_INT_EN</b> . Read-write. Reset: 0. 0=Disable. 1=Enable. Peripheral Message Received enable. Enables the generation of a command interrupt when an Upstream Peripheral Message is recieved and valid to Read.
28	<b>DNCMD_INT_EN</b> . Read-write. Reset: 0. 0=Disable. 1=Enable. Downstream Register Command Complete enable. Enables the generation of a command interrupt when a Downstream eSPI register's programming command has completed.
27	<b>RXVW_GRP3_INT_EN</b> . Read-write. Reset: 0. 0=Disable. 1=Enable. Virtual Wire Index Group3 Received enable. Enables the generation of a command interrupt when a Virtual Wire Index Group3 register specified Virtual Wire Packet is received.
26	<b>RXVW_GRP2_INT_EN</b> . Read-write. Reset: 0. 0=Disable. 1=Enable. Virtual Wire Index Group2 Received enable. Enables the generation of a command interrupt when a Virtual Wire Index Group2 register specified Virtual Wire Packet is received.
25	<b>RXVW_GRP1_INT_EN</b> . Read-write. Reset: 0. 0=Disable. 1=Enable. Virtual Wire Index Group1 Received enable. Enables the generation of a command interrupt when a Virtual Wire Index Group1 register specified Virtual Wire Packet is received.
24	<b>RXVW_GRP0_INT_EN</b> . Read-write. Reset: 0. 0=Disable. 1=Enable. Virtual Wire Index Group0 Received enable. Enables the generation of a command interrupt when a Virtual Wire Index Group0 register specified Virtual Wire Packet is received.
23:20	Reserved.
19	<b>WDG_TIMEOUT_INT_EN</b> . Read-write. Reset: 0. 0=Disable. 1=Enable. Alink Bus Watchdog Timer timeout enable. Enables the generation of an error interrupt when an Alink Bus Watchdog Timer timeout occurs.
18	<b>MST_ABORT_INT_EN</b> . Read-write. Reset: 0. 0=Disable. 1=Enable. Alink Bus Master Abort enable. Enables the generation of an error interrupt when an eSPI is doing a Master Abort.
17:16	Reserved.
15	<b>PROTOCOL_ERR_INT_EN</b> . Read-write. Reset: 0. 0=Disable. 1=Enable. Protocol Error detected enable. Enables the generation of an error interrupt when a Protocol Error is detected.
14	<b>RXFLASH_OVERFLOW_INT_EN</b> . Read-write. Reset: 0. 0=Disable. 1=Enable. Flash Packet data length over 128 bytes enable. Enables the generation of an error interrupt when a Flash Packet data is over 128 bytes.
13	<b>RXMSG_OVERFLOW_INT_EN</b> . Read-write. Reset: 0. 0=Disable. 1=Enable. Peripheral Message data length over 32 bytes enable. Enables the generation of an error interrupt when a Peripheral Packet Message data is over 32 bytes.
12	<b>RXOOB_OVERFLOW_INT_EN</b> . Read-write. Reset: 0. 0=Disable. 1=Enable. OOB Packet data length over 128 bytes enable. Enables the generation of an error interrupt when an OOB Packet data is over 128 bytes.
11	<b>ILLEGAL_LEN_INT_EN</b> . Read-write. Reset: 0. 0=Disable. 1=Enable. Illegal Response Length Received enable. Enables the generation of an error interrupt when an Illegal length is received.
10	<b>ILLEGAL_TAG_INT_EN</b> . Read-write. Reset: 0. 0=Disable. 1=Enable. Illegal Response Tag Received enable. Enables the generation of an error interrupt when an Illegal tag is received.
9	<b>UNSUCSS_CPL_INT_EN</b> . Read-write. Reset: 0. 0=Disable. 1=Enable. Unsuccessful CPL Received enable. Enables the generation of an error interrupt when an Unsuccessful Completion without Data is received.
8	<b>INVALID_CT_INT_EN</b> . Read-write. Reset: 0. 0=Disable. 1=Enable. Unrecognized/Invalid Cycle Type Received enable. Enables the generation of an error interrupt when an Unrecognized Cycle Type is received.
7	<b>INVALID_RSP_INT_EN</b> . Read-write. Reset: 0. 0=Disable. 1=Enable. Unrecognized/Invalid Response Code Received enable. Enables the generation of an error interrupt when an Unrecognized Response Code is received.
6	<b>NON_FATAL_ERR_INT_EN</b> . Read-write. Reset: 0. 0=Disable. 1=Enable. NON_FATAL_ERROR Response Code Received enable. Enables the generation of an error interrupt when a NON_FATAL_ERROR Response Code is received.
5	<b>FATAL_ERR_INT_EN</b> . Read-write. Reset: 0. 0=Disable. 1=Enable. FATAL_ERROR Response Code Received enable. Enables the generation of an error interrupt when a FATAL_ERROR Response Code is received.

4	<b>NO_RSP_INT_EN.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. NO_RESPONSE Response Code Received enable. Enables the generation of an error interrupt when a NO_RESPONSE Response Code is received.
3	Reserved.
2	<b>CRC_ERR_INT_EN.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. CRC Error detected enable. Enables the generation of an error interrupt when CRC Error is detected on the response phase.
1	<b>WAIT_TIMEOUT_INT_EN.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. eSPI Bus Wait State Insertion Maximum Out enable. Enables the generation of an error interrupt when eSPI Wait State timer timeout occurs.
0	<b>BUS_ERR_INT_EN.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. eSPI Bus Timing Error enable. Enables generation of an error interrupt when eSPI Bus Timing Error occurs.

**ESPIx00000070 (FCH::ITF::ESPI::SLAVE0\_INT\_STS)**

Read, Write-1-to-clear. Reset: 0000\_0000h.

\_aliasHOST; ESPIx00000070; ESPI=FEC2\_0000h

Bits	Description
31	<b>FLASH_REQ_INT.</b> Read, Write-1-to-clear. Reset: 0. 0=Upstream Flash Request not received. 1=Upstream Flash Request has been received. Flash Request Received status. Upstream Flash Request has been received and is valid to Read.
30	<b>RXOOB_INT.</b> Read, Write-1-to-clear. Reset: 0. 0=Upstream OOB Message not received. 1=Upstream OOB Message has been received. OOB Message Received status. Upstream OOB Message has been received and is valid to Read.
29	<b>RXMSG_INT.</b> Read, Write-1-to-clear. Reset: 0. 0=Upstream Peripheral Message not received. 1=Upstream Peripheral Message has been received. Peripheral Message Received status. Upstream Peripheral Message has been received and is valid to Read.
28	<b>DNCMD_INT.</b> Read, Write-1-to-clear. Reset: 0. 0=Downstream Register Command not complete. 1=Downstream Register Command complete. Downstream Register Command Complete status. When once set to 1, software can program the next command or get the data.
27	<b>RXVW_GRP3_INT.</b> Read, Write-1-to-clear. Reset: 0. 0=Specified VW packet has not been received. 1=Virtual Wire Index Group3 register specified VW packet has been received. Virtual Wire Index Group3 Received status.
26	<b>RXVW_GRP2_INT.</b> Read, Write-1-to-clear. Reset: 0. 0=Specified VW packet has not been received. 1=Virtual Wire Index Group2 register specified VW packet has been received. Virtual Wire Index Group2 Received status.
25	<b>RXVW_GRP1_INT.</b> Read, Write-1-to-clear. Reset: 0. 0=Specified VW packet has not been received. 1=Virtual Wire Index Group1 register specified VW packet has been received. Virtual Wire Index Group1 Received status.
24	<b>RXVW_GRP0_INT.</b> Read, Write-1-to-clear. Reset: 0. 0=Specified VW packet has not been received. 1=Virtual Wire Index Group0 register specified VW packet has been received. Virtual Wire Index Group0 Received status.
23:20	Reserved.
19	<b>WDG_TIMEOUT_INT.</b> Read, Write-1-to-clear. Reset: 0. 0=eSPI Watchdog Timer Timeout has not occurred. 1=eSPI Watchdog Timer Timeout has occurred. Alink Bus Watchdog Timer Timeout status.
18	<b>MST_ABORT_INT.</b> Read, Write-1-to-clear. Reset: 0. 0=eSPI not doing a Master Abort. 1=eSPI doing a Master Abort. Alink Bus Master Abort status.
17:16	Reserved.
15	<b>PROTOCOL_ERR_INT.</b> Read, Write-1-to-clear. Reset: 0. 0=Protocol Error has not occurred. 1=Protocol Error has occurred. Protocol Error status.
14	<b>RXFLASH_OVERFLOW_INT.</b> Read, Write-1-to-clear. Reset: 0. 0=Flash Packet Data Length not over 128 bytes. 1=Flash Packet Data Length over 128 bytes. Flash Packet Data Length over 128 bytes status.
13	<b>RXMSG_OVERFLOW_INT.</b> Read, Write-1-to-clear. Reset: 0. 0=Peripheral Message Data Length not over 32 bytes. 1=Peripheral Message Data Length over 32 bytes. Peripheral Message Data Length over 32 bytes status.
12	<b>RXOOB_OVERFLOW_INT.</b> Read, Write-1-to-clear. Reset: 0. 0=OOB Packet Length not over 128 bytes. 1=OOB Packet Length over 128 bytes. OOB Packet Data Length over 128 bytes status.
11	<b>ILLEGAL_LEN_INT.</b> Read, Write-1-to-clear. Reset: 0. 0=Illegal Response Length not received. 1=Illegal Response Length received. Illegal Response Length Received status. Set when an Illegal Response Length is received from a Slave.

10	<b>ILLEGAL_TAG_INT.</b> Read,Write-1-to-clear. Reset: 0. 0=Illegal Response Tag not received. 1=Illegal Response Tag received. Illegal Response Tag Received status. Set when an Illegal Response Tag is received.
9	<b>UNSUCSS_CPL_INT.</b> Read,Write-1-to-clear. Reset: 0. 0=Unsuccessful Completion Packet not received. 1=Unsuccessful Completion Packet received. Unsuccessful CPL Received status. Set when an Unsuccessful Completion without data is received.
8	<b>UNKNOWN_CT_INT.</b> Read,Write-1-to-clear. Reset: 0. 0=Unrecognized Cycle Type not received. 1=Unrecognized Cycle Type received. Invalid Cycle Type Received status. Set when an Unrecognized Cycle Type is received.
7	<b>UNKNOWN_RSP_INT.</b> Read,Write-1-to-clear. Reset: 0. 0=Unrecognized Response not received. 1=Unrecognized Response received. Invalid Response Code Received status. Set when an Unrecognized Response code is received.
6	<b>NON_FATAL_ERR_INT.</b> Read,Write-1-to-clear. Reset: 0. 0=NON_FATAL_ERROR not received. 1=NON_FATAL_ERROR received from the Slave. NON_FATAL_ERROR Response Code Received status. Set when a NON_FATAL_ERROR Response code is received from the Slave.
5	<b>FATAL_ERR_INT.</b> Read,Write-1-to-clear. Reset: 0. 0=FATAL_ERROR not received. 1=FATAL_ERROR received from the Slave. FATAL_ERROR Response Code Received status. Set when a FATAL_ERROR Response code is received from the Slave.
4	<b>NO_RSP_INT.</b> Read,Write-1-to-clear. Reset: 0. 0=NO_RESPONSE not received. 1=NO_RESPONSE received from the Slave. NO_RESPONSE Code Received status. Set when a NO_RESPONSE code is received from the Slave.
3	Reserved.
2	<b>CRC_ERR_INT.</b> Read,Write-1-to-clear. Reset: 0. 0=CRC Error not detected. 1=CRC Error detected. CRC Error detected status. CRC error detected on response phase. Set when a CRC Error is detected on the response phase.
1	<b>WAIT_TIMEOUT_INT.</b> Read,Write-1-to-clear. Reset: 0. 0=Timer timeout has not occurred. 1=Timer timeout has occurred. eSPI Bus Wait State Intersection maximum out status. eSPI Wait State timer timeout. Set when the Slave inserts more Wait States than the counter specified in eSPI Global Control and Status Register0.
0	<b>BUS_ERR_INT.</b> Read,Write-1-to-clear. Reset: 0. 0=Timing error not detected. 1=eSPI Bus timing error detected. eSPI Bus Timing error status. Set when eSPI link block detect Slave doesn't drive 1 after response CRC and before CS# is de-asserted.

**ESPIx00000074 (FCH::ITF::ESPI::SLAVE0\_RXMSG\_HDR0)**

Read-only. Reset: 0000\_0000h.

\_aliasHOST; ESPIx00000074; ESPI=FEC2\_0000h

Bits	Description
31:24	<b>BYTE3.</b> Read-only. Reset: 00h. Received Periperal Message code.
23:16	<b>BYTE2.</b> Read-only. Reset: 00h. Received Periperal Message Length[7:0].
15:8	<b>BYTE1.</b> Read-only. Reset: 00h. Bits[15:12]: Tag, Bits[11:8]: Length[11:8].
7:0	<b>CYCLETYPE.</b> Read-only. Reset: 00h. CycleType[7:0] for Periperal Msg/MsgD.

**ESPIx00000078 (FCH::ITF::ESPI::SLAVE0\_RXMSG\_HDR1)**

Read-only. Reset: 0000\_0000h.

\_aliasHOST; ESPIx00000078; ESPI=FEC2\_0000h

Bits	Description
31:24	<b>SPECIFIC_BYTE3.</b> Read-only. Reset: 00h. Periperal Message Specific Byte3.
23:16	<b>SPECIFIC_BYTE2.</b> Read-only. Reset: 00h. Periperal Message Specific Byte2.
15:8	<b>SPECIFIC_BYTE1.</b> Read-only. Reset: 00h. Periperal Message Specific Byte1.
7:0	<b>SPECIFIC_BYTE0.</b> Read-only. Reset: 00h. Periperal Message Specific Byte0.

**ESPIx0000007C (FCH::ITF::ESPI::SLAVE0\_RXMSG\_DATA\_PORT)**

Read-only. Reset: 0000\_0000h.

\_aliasHOST; ESPIx0000007C; ESPI=FEC2\_0000h

Bits	Description
------	-------------

31:0	<b>RXMSG_DATA</b> . Read-only. Reset: 0000_0000h. Receive message data.
------	---

**ESPIx00000080 (FCH::ITF::ESPI::RESERVED\_RXMSG\_REG0)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; ESPIx00000080; ESPI=FEC2\_0000h

Bits	Description
31:0	Reserved.

**ESPIx00000084 (FCH::ITF::ESPI::RESERVED\_RXMSG\_REG1)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; ESPIx00000084; ESPI=FEC2\_0000h

Bits	Description
31:0	Reserved.

**ESPIx00000088 (FCH::ITF::ESPI::RESERVED\_RXMSG\_REG2)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; ESPIx00000088; ESPI=FEC2\_0000h

Bits	Description
31:0	Reserved.

**ESPIx0000008C (FCH::ITF::ESPI::RESERVED\_RXMSG\_REG3)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; ESPIx0000008C; ESPI=FEC2\_0000h

Bits	Description
31:0	Reserved.

**ESPIx00000090 (FCH::ITF::ESPI::RESERVED\_RXMSG\_REG4)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; ESPIx00000090; ESPI=FEC2\_0000h

Bits	Description
31:0	Reserved.

**ESPIx00000094 (FCH::ITF::ESPI::RESERVED\_RXMSG\_REG5)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; ESPIx00000094; ESPI=FEC2\_0000h

Bits	Description
31:0	Reserved.

**ESPIx00000098 (FCH::ITF::ESPI::RESERVED\_RXMSG\_REG6)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; ESPIx00000098; ESPI=FEC2\_0000h

Bits	Description
31:0	Reserved.

**ESPIx0000009C (FCH::ITF::ESPI::SLAVE0\_RXVW)**

Reset: 0007\_0C00h.

\_aliasHOST; ESPIx0000009C; ESPI=FEC2\_0000h

Bits	Description
31:20	Reserved.
19	<b>HOST_RST_ACK</b> . Read-only. Reset: 0. 0=No acknowledge sent. 1=Acknowledge sent. Host Reset Acknowledge (HOST_RST_ACK). Sent by the Slave to acknowledge it received the HOST_RST_WARN virtual wire.
18	<b>RCIN_B</b> . Read-only. Reset: 1. 0=No INIT. 1=INIT sent. Reset CPU INIT (RCIN#). Sent to request a CPU reset

	on behalf of the Keyboard controller.																		
17	<b>SMI_B</b> . Read-only. Reset: 1. 0=No interrupt. 1=System Management Interrupt sent. System Management Interrupt (SMI#). Sent as a general purpose alert resulting in an SMI code being invoked by the BIOS.																		
16	<b>SCI_B</b> . Read-only. Reset: 1. 0=No interrupt. 1=System Controller Interrupt sent. System Controller Interrupt (SCI#). Sent as a general purpose alert resulting in an ACPI method being invoked by the OS.																		
15	<b>SLAVE_BOOT_LOAD_STS</b> . Read-only. Reset: 0. 0=The boot image is corrupted, incomplete or otherwise unusable. 1=The boot code load was successful and that the integrity of the image is intact, or the boot code load from the Master attached flash is not required. Slave boot load status. Sent upon completion of the Slave boot load from the Master attached flash. NOTE: The Slave_Boot_Load_Status must be sent in either the same or a previous virtual wire message as the Slave_Boot_Load_Done.																		
14	<b>ERROR_NONFATAL</b> . Read-only. Reset: 0. 0=No non-fatal error. 1=Non-Fatal error detected. Non-Fatal error is detected, not due to eSPI transaction on the bus. NOTE: Non-Fatal error due to transaction on the eSPI bus will be signaled through the Response (RSP) phase.																		
13	<b>ERROR_FATAL</b> . Read-only. Reset: 0. 0=No fatal error. 1=Fatal error detected. Fatal error is detected, not due to eSPI transaction on the bus. NOTE: Fatal error due to transaction on the eSPI bus will be signaled through the Response (RSP) phase.																		
12	<b>SLAVE_BOOT_LOAD_DONE</b> . Read-only. Reset: 0. 0=No event. 1=Slave Boot Load done. Slave Boot Load done. Sent upon completed boot process, as indication to eSPI Master to continue with the G3 to S0 exit.																		
11	<b>PME_B</b> . Read-only. Reset: 1. 0=No event. 1=Power Management Event occurred. PCI Power Management Event (PME#). Shared by multiple eSPI.																		
10	<b>WAKE_B</b> . Read-only. Reset: 1. 0=No effect. 1=Wake the Host. To wake the Host from Slave[x] on any event (WAKE#).																		
9	Reserved.																		
8	<b>OOB_RST_ACK</b> . Read-only. Reset: 0. 0=OOB_RST_ACK not acknowledged. 1=OOB_RST_ACK acknowledged. OOB Reset acknowledge. Sent by the Slave to acknowledge the received OOB_RST_ACK virtual wire from the Host.																		
7:5	<b>IRQ_STS</b> . Read-only. Reset: 0h. IRQ status specified by IRQ selection. <b>ValidValues:</b> <table border="1"> <thead> <tr> <th>Value</th><th>Description</th></tr> </thead> <tbody> <tr> <td>0h</td><td>IRQ keep 0 unchanged.</td></tr> <tr> <td>1h</td><td>IRQ keep 1 unchanged.</td></tr> <tr> <td>2h</td><td>IRQ changed from 1 to 0 (Clear).</td></tr> <tr> <td>3h</td><td>IRQ changed from 0 to 1 (Set).</td></tr> <tr> <td>4h</td><td>IRQ changed from 0-&gt;1-&gt;0 (High pulse).</td></tr> <tr> <td>5h</td><td>IRQ changed from 1-&gt;0-&gt;1 (Low pulse).</td></tr> <tr> <td>6h</td><td>IRQ changed from 1-&gt;1-&gt;0 or 1-&gt;0-&gt;0 or 0-&gt;0-&gt;0.</td></tr> <tr> <td>7h</td><td>IRQ changed from 0-&gt;0-&gt;1 or 0-&gt;1-&gt;1 or 1-&gt;1-&gt;1.</td></tr> </tbody> </table>	Value	Description	0h	IRQ keep 0 unchanged.	1h	IRQ keep 1 unchanged.	2h	IRQ changed from 1 to 0 (Clear).	3h	IRQ changed from 0 to 1 (Set).	4h	IRQ changed from 0->1->0 (High pulse).	5h	IRQ changed from 1->0->1 (Low pulse).	6h	IRQ changed from 1->1->0 or 1->0->0 or 0->0->0.	7h	IRQ changed from 0->0->1 or 0->1->1 or 1->1->1.
Value	Description																		
0h	IRQ keep 0 unchanged.																		
1h	IRQ keep 1 unchanged.																		
2h	IRQ changed from 1 to 0 (Clear).																		
3h	IRQ changed from 0 to 1 (Set).																		
4h	IRQ changed from 0->1->0 (High pulse).																		
5h	IRQ changed from 1->0->1 (Low pulse).																		
6h	IRQ changed from 1->1->0 or 1->0->0 or 0->0->0.																		
7h	IRQ changed from 0->0->1 or 0->1->1 or 1->1->1.																		
4:0	<b>IRQ_SEL</b> . Read-write. Reset: 00h. This field determines the Slave[N] Received Virtual Wires Register bits[7:5] output's IRQ status. <b>ValidValues:</b> <table border="1"> <thead> <tr> <th>Value</th><th>Description</th></tr> </thead> <tbody> <tr> <td>00h</td><td>IRQ0</td></tr> <tr> <td>01h</td><td>IRQ1</td></tr> <tr> <td>02h</td><td>IRQ2</td></tr> <tr> <td>16h-03h</td><td>IRQ&lt;VALUE&gt;</td></tr> <tr> <td>17h</td><td>IRQ23</td></tr> <tr> <td>1Fh-18h</td><td>Reserved.</td></tr> </tbody> </table>	Value	Description	00h	IRQ0	01h	IRQ1	02h	IRQ2	16h-03h	IRQ<VALUE>	17h	IRQ23	1Fh-18h	Reserved.				
Value	Description																		
00h	IRQ0																		
01h	IRQ1																		
02h	IRQ2																		
16h-03h	IRQ<VALUE>																		
17h	IRQ23																		
1Fh-18h	Reserved.																		

**ESPIx000000A0 (FCH::ITF::ESPI::SLAVE0\_RXVW\_DATA)**

Read-only. Reset: 0000\_0000h.



_aliasHOST; ESPIx000000A0; ESPI=FEC2_0000h	
Bits	Description
31:24	<b>GRP3.</b> Read-only. Reset: 00h. Group3 Virtual Wire Data Register. When FCH::ITF::ESPI::SLAVE0_RXVW_MISC_CNTL[GRP3_EN] == 1, eSPI Master will check each received VW Index. If the received Index matches with FCH::ITF::ESPI::SLAVE0_RXVW_INDEX[GRP3], eSPI Master will update this field with the new received value.
23:16	<b>GRP2.</b> Read-only. Reset: 00h. Group2 Virtual Wire Data Register. When FCH::ITF::ESPI::SLAVE0_RXVW_MISC_CNTL[GRP2_EN] == 1, eSPI Master will check each received VW Index. If the received Index matches with FCH::ITF::ESPI::SLAVE0_RXVW_INDEX[GRP2], eSPI Master will update this field with the new received value.
15:8	<b>GRP1.</b> Read-only. Reset: 00h. Group1 Virtual Wire Data Register. When FCH::ITF::ESPI::SLAVE0_RXVW_MISC_CNTL[GRP1_EN] == 1, eSPI Master will check each received VW Index. If the received Index matches with FCH::ITF::ESPI::SLAVE0_RXVW_INDEX[GRP1], eSPI Master will update this field with the new received value.
7:0	<b>GRP0.</b> Read-only. Reset: 00h. Group0 Virtual Wire Data Register. When FCH::ITF::ESPI::SLAVE0_RXVW_MISC_CNTL[GRP0_EN] == 1, eSPI Master will check each received VW Index. If the received Index matches with FCH::ITF::ESPI::SLAVE0_RXVW_INDEX[GRP0], eSPI Master will update this field with the new received value.

#### ESPIx000000A4 (FCH::ITF::ESPI::SLAVE0\_RXVW\_INDEX)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; ESPIx000000A4; ESPI=FEC2\_0000h

Bits	Description
31:24	<b>GRP3.</b> Read-write. Reset: 00h. Group3 Virtual Wire Index selection.
23:16	<b>GRP2.</b> Read-write. Reset: 00h. Group2 Virtual Wire Index selection.
15:8	<b>GRP1.</b> Read-write. Reset: 00h. Group1 Virtual Wire Index selection.
7:0	<b>GRP0.</b> Read-write. Reset: 00h. Group0 Virtual Wire Index selection.

#### ESPIx000000A8 (FCH::ITF::ESPI::SLAVE0\_RXVW\_MISC\_CNTL)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; ESPIx000000A8; ESPI=FEC2\_0000h

Bits	Description
31	<b>IRQ23_MASK.</b> Read-write. Reset: 0. 0=Unmasked: IRQ23 will be sent to the ACPI. 1=Masked: IRQ23 will not be sent to the ACPI. When set, IRQ23 received from a Virtual Wire packet will be masked, and eSPI_IRQ23 will not be sent to the ACPI.
30	<b>IRQ22_MASK.</b> Read-write. Reset: 0. 0=Unmasked: IRQ22 will be sent to the ACPI. 1=Masked: IRQ22 will not be sent to the ACPI. When set, IRQ22 received from a Virtual Wire packet will be masked, and eSPI_IRQ22 will not be sent to the ACPI.
29	<b>IRQ21_MASK.</b> Read-write. Reset: 0. 0=Unmasked: IRQ21 will be sent to the ACPI. 1=Masked: IRQ21 will not be sent to the ACPI. When set, IRQ21 received from a Virtual Wire packet will be masked, and eSPI_IRQ21 will not be sent to the ACPI.
28	<b>IRQ20_MASK.</b> Read-write. Reset: 0. 0=Unmasked: IRQ20 will be sent to the ACPI. 1=Masked: IRQ20 will not be sent to the ACPI. When set, IRQ20 received from a Virtual Wire packet will be masked, and eSPI_IRQ20 will not be sent to the ACPI.
27	<b>IRQ19_MASK.</b> Read-write. Reset: 0. 0=Unmasked: IRQ19 will be sent to the ACPI. 1=Masked: IRQ19 will not be sent to the ACPI. When set, IRQ19 received from a Virtual Wire packet will be masked, and eSPI_IRQ19 will not be sent to the ACPI.
26	<b>IRQ18_MASK.</b> Read-write. Reset: 0. 0=Unmasked: IRQ18 will be sent to the ACPI. 1=Masked: IRQ18 will not be sent to the ACPI. When set, IRQ18 received from a Virtual Wire packet will be masked, and eSPI_IRQ18 will not be sent to the ACPI.
25	<b>IRQ17_MASK.</b> Read-write. Reset: 0. 0=Unmasked: IRQ17 will be sent to the ACPI. 1=Masked: IRQ17 will not be sent to the ACPI. When set, IRQ17 received from a Virtual Wire packet will be masked, and eSPI_IRQ17 will not be sent to the ACPI.

	not be sent to the ACPI.
24	<b>IRQ16_MASK.</b> Read-write. Reset: 0. 0=Unmasked: IRQ16 will be sent to the ACPI. 1=Masked: IRQ16 will not be sent to the ACPI. When set, IRQ16 received from a Virtual Wire packet will be masked, and eSPI_IRQ16 will not be sent to the ACPI.
23	<b>IRQ15_MASK.</b> Read-write. Reset: 0. 0=Unmasked: IRQ15 will be sent to the ACPI. 1=Masked: IRQ15 will not be sent to the ACPI. When set, IRQ15 received from a Virtual Wire packet will be masked, and eSPI_IRQ15 will not be sent to the ACPI.
22	<b>IRQ14_MASK.</b> Read-write. Reset: 0. 0=Unmasked: IRQ14 will be sent to the ACPI. 1=Masked: IRQ14 will not be sent to the ACPI. When set, IRQ14 received from a Virtual Wire packet will be masked, and eSPI_IRQ14 will not be sent to the ACPI.
21	<b>IRQ13_MASK.</b> Read-write. Reset: 0. 0=Unmasked: IRQ13 will be sent to the ACPI. 1=Masked: IRQ13 will not be sent to the ACPI. When set, IRQ13 received from a Virtual Wire packet will be masked, and eSPI_IRQ13 will not be sent to the ACPI.
20	<b>IRQ12_MASK.</b> Read-write. Reset: 0. 0=Unmasked: IRQ12 will be sent to the ACPI. 1=Masked: IRQ12 will not be sent to the ACPI. When set, IRQ12 received from a Virtual Wire packet will be masked, and eSPI_IRQ12 will not be sent to the ACPI.
19	<b>IRQ11_MASK.</b> Read-write. Reset: 0. 0=Unmasked: IRQ11 will be sent to the ACPI. 1=Masked: IRQ11 will not be sent to the ACPI. When set, IRQ11 received from a Virtual Wire packet will be masked, and eSPI_IRQ11 will not be sent to the ACPI.
18	<b>IRQ10_MASK.</b> Read-write. Reset: 0. 0=Unmasked: IRQ10 will be sent to the ACPI. 1=Masked: IRQ10 will not be sent to the ACPI. When set, IRQ10 received from a Virtual Wire packet will be masked, and eSPI_IRQ10 will not be sent to the ACPI.
17	<b>IRQ9_MASK.</b> Read-write. Reset: 0. 0=Unmasked: IRQ9 will be sent to the ACPI. 1=Masked: IRQ9 will not be sent to the ACPI. When set, IRQ9 received from a Virtual Wire packet will be masked, and eSPI_IRQ9 will not be sent to the ACPI.
16	<b>IRQ8_MASK.</b> Read-write. Reset: 0. 0=Unmasked: IRQ8 will be sent to the ACPI. 1=Masked: IRQ8 will not be sent to the ACPI. When set, IRQ8 received from a Virtual Wire packet will be masked, and eSPI_IRQ8 will not be sent to the ACPI.
15	<b>IRQ7_MASK.</b> Read-write. Reset: 0. 0=Unmasked: IRQ7 will be sent to the ACPI. 1=Masked: IRQ7 will not be sent to the ACPI. When set, IRQ7 received from a Virtual Wire packet will be masked, and eSPI_IRQ7 will not be sent to the ACPI.
14	<b>IRQ6_MASK.</b> Read-write. Reset: 0. 0=Unmasked: IRQ6 will be sent to the ACPI. 1=Masked: IRQ6 will not be sent to the ACPI. When set, IRQ6 received from a Virtual Wire packet will be masked, and eSPI_IRQ6 will not be sent to the ACPI.
13	<b>IRQ5_MASK.</b> Read-write. Reset: 0. 0=Unmasked: IRQ5 will be sent to the ACPI. 1=Masked: IRQ5 will not be sent to the ACPI. When set, IRQ5 received from a Virtual Wire packet will be masked, and eSPI_IRQ5 will not be sent to the ACPI.
12	<b>IRQ4_MASK.</b> Read-write. Reset: 0. 0=Unmasked: IRQ4 will be sent to the ACPI. 1=Masked: IRQ4 will not be sent to the ACPI. When set, IRQ4 received from a Virtual Wire packet will be masked, and eSPI_IRQ4 will not be sent to the ACPI.
11	<b>IRQ3_MASK.</b> Read-write. Reset: 0. 0=Unmasked: IRQ3 will be sent to the ACPI. 1=Masked: IRQ3 will not be sent to the ACPI. When set, IRQ3 received from a Virtual Wire packet will be masked, and eSPI_IRQ3 will not be sent to the ACPI.
10	<b>IRQ2_MASK.</b> Read-write. Reset: 0. 0=Unmasked: IRQ2 will be sent to the ACPI. 1=Masked: IRQ2 will not be sent to the ACPI. When set, IRQ2 received from a Virtual Wire packet will be masked, and eSPI_IRQ2 will not be sent to the ACPI.
9	<b>IRQ1_MASK.</b> Read-write. Reset: 0. 0=Unmasked: IRQ1 will be sent to the ACPI. 1=Masked: IRQ1 will not be sent to the ACPI. When set, IRQ1 received from a Virtual Wire packet will be masked, and eSPI_IRQ1 will not be sent to the ACPI.
8	<b>IRQ0_MASK.</b> Read-write. Reset: 0. 0=Unmasked: IRQ0 will be sent to the ACPI. 1=Masked: IRQ0 will not be sent to the ACPI. When set, IRQ0 received from a Virtual Wire packet will be masked, and eSPI_IRQ0 will not



	be sent to the ACPI.
7:5	Reserved.
4	<b>SUS_STAT_VWEN</b> . Read-write. Reset: 0. 0=Disable. 1=Enable. SUS_STAT# enable. Hardware sends Virtual Wire packet when SUS_STAT# changes.
3	<b>GRP3_EN</b> . Read-write. Reset: 0. 0=Don't check the received Index. 1=Check the received Index. Group3 enable. When set, VW channel will check the received Index. If Index is same as FCH::ITF::ESPI::SLAVE0_RXVW_INDEX[GRP3]; VW will store the Data into FCH::ITF::ESPI::SLAVE0_RXVW_DATA[GRP3].
2	<b>GRP2_EN</b> . Read-write. Reset: 0. 0=Don't check the received Index. 1=Check the received Index. Group2 enable. When set, VW channel will check the received Index. If Index is same as FCH::ITF::ESPI::SLAVE0_RXVW_INDEX[GRP2]; VW will store the Data into FCH::ITF::ESPI::SLAVE0_RXVW_DATA[GRP2].
1	<b>GRP1_EN</b> . Read-write. Reset: 0. 0=Don't check the received Index. 1=Check the received Index. Group1 enable. When set, VW channel will check the received Index. If Index is same as FCH::ITF::ESPI::SLAVE0_RXVW_INDEX[GRP1]; VW will store the Data into FCH::ITF::ESPI::SLAVE0_RXVW_DATA[GRP1].
0	<b>GRP0_EN</b> . Read-write. Reset: 0. 0=Don't check the received Index. 1=Check the received Index. Group0 enable. When set, VW channel will check the received Index. If Index is same as FCH::ITF::ESPI::SLAVE0_RXVW_INDEX[GRP0]; VW will store the Data into FCH::ITF::ESPI::SLAVE0_RXVW_DATA[GRP0].

#### ESPIx000000AC (FCH::ITF::ESPI::SLAVE0\_RXVW\_POLARITY)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; ESPIx000000AC; ESPI=FEC2\_0000h

Bits	Description
31:24	Reserved.
23	<b>IRQ23_POLARITY</b> . Read-write. Reset: 0. 0=Invert VW IRQ data. 1=Don't invert VW IRQ packet data. If the Slave IRQ23 is low active level interrupt or high active edge interrupt, no need to invert VW IRQ data, this bit must be set. If the Slave IRQ is high active level interrupt or low active edge interrupt, it needs to invert VW IRQ data and this bit must be cleared.
22	<b>IRQ22_POLARITY</b> . Read-write. Reset: 0. 0=Invert VW IRQ data. 1=Don't invert VW IRQ packet data. If the Slave IRQ22 is low active level interrupt or high active edge interrupt, no need to invert VW IRQ data, this bit must be set. If the Slave IRQ is high active level interrupt or low active edge interrupt, it needs to invert VW IRQ data and this bit must be cleared.
21	<b>IRQ21_POLARITY</b> . Read-write. Reset: 0. 0=Invert VW IRQ data. 1=Don't invert VW IRQ packet data. If the Slave IRQ21 is low active level interrupt or high active edge interrupt, no need to invert VW IRQ data, this bit must be set. If the Slave IRQ is high active level interrupt or low active edge interrupt, it needs to invert VW IRQ data and this bit must be cleared.
20	<b>IRQ20_POLARITY</b> . Read-write. Reset: 0. 0=Invert VW IRQ data. 1=Don't invert VW IRQ packet data. If the Slave IRQ20 is low active level interrupt or high active edge interrupt, no need to invert VW IRQ data, this bit must be set. If the Slave IRQ is high active level interrupt or low active edge interrupt, it needs to invert VW IRQ data and this bit must be cleared.
19	<b>IRQ19_POLARITY</b> . Read-write. Reset: 0. 0=Invert VW IRQ data. 1=Don't invert VW IRQ packet data. If the Slave IRQ19 is low active level interrupt or high active edge interrupt, no need to invert VW IRQ data, this bit must be set. If the Slave IRQ is high active level interrupt or low active edge interrupt, it needs to invert VW IRQ data and this bit must be cleared.
18	<b>IRQ18_POLARITY</b> . Read-write. Reset: 0. 0=Invert VW IRQ data. 1=Don't invert VW IRQ packet data. If the Slave IRQ18 is low active level interrupt or high active edge interrupt, no need to invert VW IRQ data, this bit must be set. If the Slave IRQ is high active level interrupt or low active edge interrupt, it needs to invert VW IRQ data and this bit must be cleared.
17	<b>IRQ17_POLARITY</b> . Read-write. Reset: 0. 0=Invert VW IRQ data. 1=Don't invert VW IRQ packet data. If the Slave IRQ17 is low active level interrupt or high active edge interrupt, no need to invert VW IRQ data, this bit

472

	must be set. If the Slave IRQ is high active level interrupt or low active edge interrupt, it needs to invert VW IRQ data and this bit must be cleared.
3	<b>IRQ3_POLARITY</b> . Read-write. Reset: 0. 0=Invert VW IRQ data. 1=Don't invert VW IRQ packet data. If the Slave IRQ3 is low active level interrupt or high active edge interrupt, no need to invert VW IRQ data, this bit must be set. If the Slave IRQ is high active level interrupt or low active edge interrupt, it needs to invert VW IRQ data and this bit must be cleared.
2	<b>IRQ2_POLARITY</b> . Read-write. Reset: 0. 0=Invert VW IRQ data. 1=Don't invert VW IRQ packet data. If the Slave IRQ2 is low active level interrupt or high active edge interrupt, no need to invert VW IRQ data, this bit must be set. If the Slave IRQ is high active level interrupt or low active edge interrupt, it needs to invert VW IRQ data and this bit must be cleared.
1	<b>IRQ1_POLARITY</b> . Read-write. Reset: 0. 0=Invert VW IRQ data. 1=Don't invert VW IRQ packet data. If the Slave IRQ1 is low active level interrupt or high active edge interrupt, no need to invert VW IRQ data, this bit must be set. If the Slave IRQ is high active level interrupt or low active edge interrupt, it needs to invert VW IRQ data and this bit must be cleared.
0	<b>IRQ0_POLARITY</b> . Read-write. Reset: 0. 0=Invert VW IRQ data. 1=Don't invert VW IRQ packet data. If the Slave IRQ0 is low active level interrupt or high active edge interrupt, no need to invert VW IRQ data, this bit must be set. If the Slave IRQ is high active level interrupt or low active edge interrupt, it needs to invert VW IRQ data and this bit must be cleared.

## 9.2.10 MISC Registers

### 9.2.10.1 Miscellaneous (MISC) Registers

The MISC register space is accessed through the AcpiMmio region. The MISC registers range from FED8\_0000h+E00h to FED8\_0000h+EFFh. See FCH::PM::IsaControl[MmioEn].

<b>MISCx000 [GPPClkControl] (FCH::MISC::GPPClkCntrl)</b>											
Read-write. Reset: 0000_3FFFh.											
_aliasHOST; MISCx000; MISC=FED8_0E00h											
Bits	Description										
31:14	Reserved.										
13:12	<b>GPP_CLK6_Clock_Request_mapping</b> . Read-write. Reset: 3h.  <b>Description:</b> GPP6 PCIE clock pins (GPP_CLK6P/GPP_CLK6N) output control by CLKREQ6# pin. GPP_CLK6P/GPP_CLK6N pins are powered off when FCH is strapped to use an external clock, and powered on when FCH is strapped to operate in integrated clock mode. When FCH is in integrated clock mode, GPP6 PCIE clock can be powered off according to the CLK_REQ mapping table below, and the selected CLK_REQ6# input can power off the GPP6 PCIE clock output pins if it is asserted.  <b>ValidValues:</b>										
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>Off.</td></tr> <tr> <td>1h</td><td>CLK_REQ6#.</td></tr> <tr> <td>2h</td><td>Off, Reserved.</td></tr> <tr> <td>3h</td><td>On (default).</td></tr> </table>	Value	Description	0h	Off.	1h	CLK_REQ6#.	2h	Off, Reserved.	3h	On (default).
Value	Description										
0h	Off.										
1h	CLK_REQ6#.										
2h	Off, Reserved.										
3h	On (default).										
11:10	<b>GPP_CLK5_Clock_Request_mapping</b> . Read-write. Reset: 3h.  <b>Description:</b> GPP5 PCIE clock pins (GPP_CLK5P/GPP_CLK5N) output control by CLKREQ5# pin. GPP_CLK5P/GPP_CLK5N pins are powered off when FCH is strapped to use an external clock, and powered on when FCH is strapped to operate in integrated clock mode. When FCH is in integrated clock mode, GPP5 PCIE										

	<p>clock can be powered off according to the CLK_REQ mapping table below, and the selected CLK_REQ5# input can power off the GPP5 PCIE clock output pins if it is asserted.</p> <p><b>ValidValues:</b></p> <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>Off.</td></tr> <tr> <td>1h</td><td>CLK_REQ5#.</td></tr> <tr> <td>2h</td><td>Off, Reserved.</td></tr> <tr> <td>3h</td><td>On (default).</td></tr> </table>	Value	Description	0h	Off.	1h	CLK_REQ5#.	2h	Off, Reserved.	3h	On (default).
Value	Description										
0h	Off.										
1h	CLK_REQ5#.										
2h	Off, Reserved.										
3h	On (default).										
9:8	<p><b>GPP_CLK3_Clock_Request_mapping.</b> Read-write. Reset: 3h. GPP3 PCIE clock pins (GPP_CLK3P/GPP_CLK3N) output control by CLKREQ3# pin. GPP_CLK3P/GPP_CLK3N pins are powered off when FCH is strapped to use an external clock, and powered on when FCH is strapped to operate in integrated clock mode. When FCH is in integrated clock mode, GPP3 PCIE clock can be powered off according to the CLK_REQ mapping table below, and the selected CLK_REQ3# input can power off the GPP3 PCIE clock output pins if it is asserted.</p> <p><b>ValidValues:</b></p> <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>Off.</td></tr> <tr> <td>1h</td><td>CLK_REQ3#.</td></tr> <tr> <td>2h</td><td>Off, Reserved.</td></tr> <tr> <td>3h</td><td>On (default).</td></tr> </table>	Value	Description	0h	Off.	1h	CLK_REQ3#.	2h	Off, Reserved.	3h	On (default).
Value	Description										
0h	Off.										
1h	CLK_REQ3#.										
2h	Off, Reserved.										
3h	On (default).										
7:6	<p><b>GPP_CLK2_Clock_Request_mapping.</b> Read-write. Reset: 3h. GPP2 PCIE clock pins (GPP_CLK2P/GPP_CLK2N) output control by CLKREQ2# pin. GPP_CLK2P/GPP_CLK2N pins are powered off when FCH is strapped to use an external clock, and powered on when FCH is strapped to operate in integrated clock mode. When FCH is in integrated clock mode, GPP2 PCIE clock can be powered off according to the CLK_REQ mapping table below, and the selected CLK_REQ2# input can power off the GPP2 PCIE clock output pins if it is asserted.</p> <p><b>ValidValues:</b></p> <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>Off.</td></tr> <tr> <td>1h</td><td>CLK_REQ2#.</td></tr> <tr> <td>2h</td><td>Off, Reserved.</td></tr> <tr> <td>3h</td><td>On (default).</td></tr> </table>	Value	Description	0h	Off.	1h	CLK_REQ2#.	2h	Off, Reserved.	3h	On (default).
Value	Description										
0h	Off.										
1h	CLK_REQ2#.										
2h	Off, Reserved.										
3h	On (default).										
5:4	<p><b>GPP_CLK4_Clock_Request_mapping.</b> Read-write. Reset: 3h.</p> <p><b>Description:</b> GPP4 PCIE clock pins (GPP_CLK4P/GPP_CLK4N) output control by CLKREQ4# pin. GPP_CLK4P/GPP_CLK4N pins are powered off when FCH is strapped to use an external clock, and powered on when FCH is strapped to operate in integrated clock mode. When FCH is in the integrated clock mode, GPP4 PCIE clock can be powered off according to the CLK_REQ mapping table below and the selected CLK_REQ4# input can power off the GPP4 PCIE clock output pins if it is asserted.</p> <p><b>ValidValues:</b></p> <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>Off.</td></tr> <tr> <td>1h</td><td>CLK_REQ4#.</td></tr> <tr> <td>2h</td><td>Off, Reserved.</td></tr> <tr> <td>3h</td><td>On (default).</td></tr> </table>	Value	Description	0h	Off.	1h	CLK_REQ4#.	2h	Off, Reserved.	3h	On (default).
Value	Description										
0h	Off.										
1h	CLK_REQ4#.										
2h	Off, Reserved.										
3h	On (default).										
3:2	<p><b>GPP_CLK1_Clock_Request_mapping.</b> Read-write. Reset: 3h. GPP1 PCIE clock pins (GPP_CLK1P/GPP_CLK1N) output control by CLKREQ0# pin. GPP_CLK1P/GPP_CLK1N pins are powered off when FCH is strapped to use an external clock, and powered on when FCH is strapped to operate in integrated clock mode. When FCH is in integrated clock mode, GPP1 PCIE clock can be powered off according to the CLK_REQ mapping table below, and the selected CLK_REQ1# input can power off the GPP0 PCIE clock output pins if it is asserted.</p>										

	<b>ValidValues:</b>										
	<table><tr><th>Value</th><th>Description</th></tr><tr><td>0h</td><td>Off.</td></tr><tr><td>1h</td><td>CLK_REQ1#.</td></tr><tr><td>2h</td><td>Off, Reserved.</td></tr><tr><td>3h</td><td>On (default).</td></tr></table>	Value	Description	0h	Off.	1h	CLK_REQ1#.	2h	Off, Reserved.	3h	On (default).
Value	Description										
0h	Off.										
1h	CLK_REQ1#.										
2h	Off, Reserved.										
3h	On (default).										
1:0	<p><b>GPP_CLK0_Clock_Request_mapping.</b> Read-write. Reset: 3h. GPP0 PCIE clock pins (GPP_CLK0P/GPP_CLK0N) output control by CLKREQ0# pin. GPP_CLK0P/GPP_CLK0N pins are powered off when FCH is strapped to use an external clock, and powered on when FCH is strapped to operate in integrated clock mode. When FCH is in integrated clock mode, GPP0 PCIE clock can be powered off according to the CLK_REQ mapping table below, and the selected CLK_REQ0# input can power off the GPP0 PCIE clock output pins if it is asserted.</p> <p><b>ValidValues:</b></p> <table><tr><th>Value</th><th>Description</th></tr><tr><td>0h</td><td>Off.</td></tr><tr><td>1h</td><td>CLK_REQ0#.</td></tr><tr><td>2h</td><td>Off, Reserved.</td></tr><tr><td>3h</td><td>On (default).</td></tr></table>	Value	Description	0h	Off.	1h	CLK_REQ0#.	2h	Off, Reserved.	3h	On (default).
Value	Description										
0h	Off.										
1h	CLK_REQ0#.										
2h	Off, Reserved.										
3h	On (default).										

#### MISCx004 [ClkOutputCntl] (FCH::MISC::ClkOutCntl)

Reset: 0000\_0000h.

\_aliasHOST; MISCx004; MISC=FED8\_0E00h

Bits	Description
31:0	Reserved.

#### MISCx008 [CGPLLConfig1] (FCH::MISC::CGPlLCfg1)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; MISCx008; MISC=FED8\_0E00h

Bits	Description
31	<b>XTAL_REFCLK2X_CLKEN.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. REFCLK2X clock enable/disable.
30	<b>XTAL_CLKGEN_S5_CLKEN.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. CLKGEN_S5 RefClk enable/disable.
29	<b>XTAL_CLKGEN_S0_CLKEN.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. CLKGEN_S0 RefClk enable/disable.
28:9	Reserved.
8	<p><b>CG1_REFDIVSRC_Override.</b> Read-write. Reset: 0.</p> <p><b>Description:</b> CG1PLL Refclk Source Select Override. This bit is used to override CG1PLL REFCLK source select.</p> <p>By default, Master_die CG1PLL REFCLK source is from 48 MHz CG_XTAL and Slave_die is 100 MHz from external clock chip.</p> <p>For Master_die:            0=CG1PLL REFCLK source is from 48 MHz CG_XTAL.            1=CG1PLL REFCLK source is 100 MHz from external clock chip.</p> <p>For Slave_die:            0=CG1PLL REFCLK source is 100MHz from external clock chip.            1=CG1PLL REFCLK source is from 48 MHz CG_XTAL.</p> <p>NOTE: Need to apply a reset after setting this bit.</p>
7:5	Reserved.

4	<b>RefClk_Source_Switch_Mode.</b> Read-write. Reset: 0. 0=Need to apply a PllRstB for switching different REFCLK source. 1=Switch REFCLK source on-the-fly. For PCIe®/USB3/SATA/UFS/SSIC/GBE/DPLL REFCLK source. It can be from either CG_XTAL/CG_PLL generated or external reference source.
3:1	Reserved.
0	<b>CG1PLL_SpreadSpectrumEnable.</b> Read-write. Reset: 0. 0=Disable Spread Spectrum. 1=Enable CG1_PLL Spread Spectrum.

**MISCx00C [CGPLLConfig2] (FCH::MISC::CGPllCfg2)**

Reset: 0000\_0000h.

\_aliasHOST; MISCx00C; MISC=FED8\_0E00h

Bits	Description
31:0	Reserved.

**MISCx010 [CGPLLConfig3] (FCH::MISC::CGPllCfg3)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; MISCx010; MISC=FED8\_0E00h

Bits	Description
31:30	Reserved.
29	<b>CG1PLL_fracn_en_Override.</b> Read-write. Reset: 0. Used with CG1PLL frac-N and SSC clocking only.
28:13	Reserved.
12:4	<b>CG1PLL_fcw0_int_Override.</b> Read-write. Reset: 000h. CG1PLL Override: Integer portion of Frequency Control Word0 (feedback divisor0).
3:2	Reserved.
1:0	<b>CG1PLL_refclk_div_Override.</b> Read-write. Reset: 0h. CG1PLL Override: Reference clock divisor.
<b>ValidValues:</b>	
Value	Description
0h	1x.
1h	2x.
3h-2h	4x.

**MISCx014 [CGPLLConfig4] (FCH::MISC::CGPllCfg4)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; MISCx014; MISC=FED8\_0E00h

Bits	Description
31:24	Reserved.
23:8	<b>CG1PLL_fcw1_frac_Override.</b> Read-write. Reset: 0000h. CG1PLL Override: Fractional portion of Frequency Control Word1 (feedback divisor1). Intended to be used with frequency ramping. Also used to step PLL frequency and phase for DFT.
7:0	Reserved.

**MISCx018 [CGPLLConfig5] (FCH::MISC::CGPllCfg5)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; MISCx018; MISC=FED8\_0E00h

Bits	Description
31:16	<b>CG1PLL_fcw_slew_frac_Override.</b> Read-write. Reset: 0000h. <b>Description:</b> CG1PLL Override: Sets SSC freq ramp rate. Set fractional change in programmed frequency per REFCLK cycle. e.g., 0.5% downspread SSC at 33.3 KHz and FB DIV=80. FCW_slewrates_frac = $(2^{16}) * 0.00485 * 80 / (15\mu s / 10ns) = 17$ Need 31.5kHz and -0.375%.
15:0	Reserved.



**MISCx01C [CGPLLConfig6] (FCH::MISC::CGPllCfg6)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; MISCx01C; MISC=FED8\_0E00h

Bits	Description
31:17	Reserved.
16:13	<b>CG1PLL_gp_coarse_mant_Override</b> . Read-write. Reset: 0h. CG1PLL Override: Coarse proportional path FP multiply mantissa.
12:0	Reserved.

**MISCx024 [ClkDrvStr1] (FCH::MISC::ClkDrvStr1)**

Read-write. Reset: 0000\_0249h.

\_aliasHOST; MISCx024; MISC=FED8\_0E00h

Bits	Description												
31:12	Reserved.												
11:9	<b>GppClk3_ClockBufferDrivingStrengthControl</b> . Read-write. Reset: 1h. Drive Strength control for GPP_CLK_3 differential Clock Buffers. Drive strength addition/subtraction relative to IMP_CTRL[4:0]. Default: 3'b001. <b>ValidValues:</b> <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>~ -10%.</td></tr> <tr> <td>1h</td><td>No change from IMP_CTRL.</td></tr> <tr> <td>2h</td><td>~ +10%.</td></tr> <tr> <td>3h</td><td>~ + 20%.</td></tr> <tr> <td>7h-4h</td><td>Reserved.</td></tr> </table>	Value	Description	0h	~ -10%.	1h	No change from IMP_CTRL.	2h	~ +10%.	3h	~ + 20%.	7h-4h	Reserved.
Value	Description												
0h	~ -10%.												
1h	No change from IMP_CTRL.												
2h	~ +10%.												
3h	~ + 20%.												
7h-4h	Reserved.												
8:6	<b>GppClk2_ClockBufferDrivingStrengthControl</b> . Read-write. Reset: 1h. Drive Strength control for GPP_CLK_2 differential Clock Buffers. Drive strength addition/subtraction relative to IMP_CTRL[4:0]. Default: 3'b001. <b>ValidValues:</b> <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>~ -10%.</td></tr> <tr> <td>1h</td><td>No change from IMP_CTRL.</td></tr> <tr> <td>2h</td><td>~ +10%.</td></tr> <tr> <td>3h</td><td>~ + 20%.</td></tr> <tr> <td>7h-4h</td><td>Reserved.</td></tr> </table>	Value	Description	0h	~ -10%.	1h	No change from IMP_CTRL.	2h	~ +10%.	3h	~ + 20%.	7h-4h	Reserved.
Value	Description												
0h	~ -10%.												
1h	No change from IMP_CTRL.												
2h	~ +10%.												
3h	~ + 20%.												
7h-4h	Reserved.												
5:3	<b>GppClk1_ClockBufferDrivingStrengthControl</b> . Read-write. Reset: 1h. Drive Strength control for GPP_CLK_1 differential Clock Buffers. Drive strength addition/subtraction relative to IMP_CTRL[4:0]. <b>ValidValues:</b> <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>~ -10%.</td></tr> <tr> <td>1h</td><td>No change from IMP_CTRL.</td></tr> <tr> <td>2h</td><td>~ +10%.</td></tr> <tr> <td>3h</td><td>~ + 20%.</td></tr> <tr> <td>7h-4h</td><td>Reserved.</td></tr> </table>	Value	Description	0h	~ -10%.	1h	No change from IMP_CTRL.	2h	~ +10%.	3h	~ + 20%.	7h-4h	Reserved.
Value	Description												
0h	~ -10%.												
1h	No change from IMP_CTRL.												
2h	~ +10%.												
3h	~ + 20%.												
7h-4h	Reserved.												
2:0	<b>GppClk0_ClockBufferDrivingStrengthControl</b> . Read-write. Reset: 1h. Drive Strength control for GPP_CLK_0 differential Clock Buffers. Drive strength addition/subtraction relative to IMP_CTRL[4:0]. <b>ValidValues:</b> <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>~ -10%.</td></tr> <tr> <td>1h</td><td>No change from IMP_CTRL.</td></tr> <tr> <td>2h</td><td>~ +10%.</td></tr> <tr> <td>3h</td><td>~ + 20%.</td></tr> </table>	Value	Description	0h	~ -10%.	1h	No change from IMP_CTRL.	2h	~ +10%.	3h	~ + 20%.		
Value	Description												
0h	~ -10%.												
1h	No change from IMP_CTRL.												
2h	~ +10%.												
3h	~ + 20%.												



	7h-4h	Reserved.
--	-------	-----------

**MISCx02C [ClkGatedCntl] (FCH::MISC::ClkGatedCntl)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; MISCx02C; MISC=FED8\_0E00h

Bits	Description
31:18	Reserved.
17	<b>BlinkClkGateOffEn.</b> Read-write. Reset: 0. 0=Disable B-Link Clock Gate Off function. 1=Enable B-Link Clock Gate Off function. B-Link Clock Gate-Off Enable. Internal B-Link clock has two clock trees: one is a free running clock and the other is a gated clock. When all controllers agree to stop the gated B-Link clock and this bit is set, clock gating logic gates off the clock tree from clock root.
16	<b>AlinkClkGateOffEn.</b> Read-write. Reset: 0. 0=Disable A-Link Clock Gate-Off function. 1=Enable A-Link Clock Gate-Off function. A-Link Clock Gate-Off Enable. Internal A-Link clock has two clock trees: one is a free-running clock and the other is a gated clock. When all controllers agree to stop the gated A-Link clock and this bit is set, clock gating logic gates off the clock tree from clock root.
15:0	Reserved.

**MISCx034 [CGPLLConfig8] (FCH::MISC::CGPLLcfg8)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; MISCx034; MISC=FED8\_0E00h

Bits	Description
31:23	<b>CG1PLL_fcw1_int_Override.</b> Read-write. Reset: 000h. CG1PLL Override: Integer portion of Frequency Control Word0. Intended to be used with frequency ramping.
22:0	Reserved.

**MISCx03C [CGPLLConfig10] (FCH::MISC::CGPLLcfg10)**

Reset: 0000\_0000h.

\_aliasHOST; MISCx03C; MISC=FED8\_0E00h

Bits	Description
31:0	Reserved.

**MISCx040 [MiscClkCntl1] (FCH::MISC::MiscClkCntl1)**

Read-write. Reset: 0000\_4004h.

\_aliasHOST; MISCx040; MISC=FED8\_0E00h

Bits	Description
31	Reserved.
30	<b>CG1_cfg_update_req.</b> Read-write. Reset: 0. Setting this bit requests CG1_PLL to load spread related value into CG1_PLL. The bit is cleared by hardware after the request is sent to CG1_PLL.
29:26	Reserved.
25	<b>CG1_FBDIV_LoadEn.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. Enable loading CG1_PLL FBDIV value form register.
24:20	Reserved.
19	<b>USB31_CLK_Source.</b> Read-write. Reset: 0. 0=343 MHz. 1=400 MHz. <b>Description:</b> USB31 Host Clock Source (in mission mode). USB31 Host Clock can be 400 MHz or 343 MHz. NOTE: In scan mode, there is a TDR bit that can make the selection. TDR default is 0 to select 400 MHz output.
18	Reserved.
17	<b>USB3_Refclk_Drvier_PWDN.</b> Read-write. Reset: 0. 1=USB3 REFCLK Driver is turned off in S5/S3 state. USB3 REFCLK Driver (differential) Power-Down in S3/S5.
16:15	Reserved.
14	<b>OscClkSwitchEn.</b> Read-write. Reset: 1. 0=OSC is 12 MHz (48 MHz CG_XTAL divided by 4). 1=OSC is average 14.318 MHz. When this bit is set, the FCH uses the 48 MHz CG_XTAL to generate the average 14 MHz

	clock as OSC.
13:12	Reserved.
11	<b>AZ_48MCLK_PWDN.</b> Read-write. Reset: 0. 0=AZ 48 MHz Clock output driver is enabled. 1= AZ 48 MHz Clock output driver is disabled. AZ 48 MHz Clock Output Driver PWDN. FCH provides 48 MHz clock for Azalia (AZ) ACP.
10	<b>The48MXtal_S0i3_PWDN_En.</b> Read-write. Reset: 0. 0=Don't enable CG XTAL pad power down for S0i3. 1=Enable powering down CG XTAL pad in S0i3 state. CG 48 MHz XTAL Pad power down enable when in S0i3 state.
9	<b>The48MXtal_S5_PWDN_En.</b> Read-write. Reset: 0. 0=Don't enable CG XTAL pad power down for S5. 1=Enable to power down CG XTAL pad in S5 state. CG 48 MHz XTAL Pad power down enable when in S5 state.
8	<b>The48MXtal_S3_PWDN_En.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. <b>Description:</b> When XTAL_PAD gets powered down, 48 MHz clock is off. (USB3 wake-up clock, ACP 48 MHz clock, and CGPLL REFCLK are from CG_XTAL directly). CG 48 MHz XTAL Pad power down is enabled when in S3 state, and CG XTAL pad can be powered off in following conditions. <ol style="list-style-type: none"> <li>1. S3 State with setting of FCH::MISC::MiscClkCntl1[8] == 1.</li> <li>2. S5 State with setting of FCH::MISC::MiscClkCntl1[9] == 1.</li> <li>3. S0i3 State (connected standby state) with setting of FCH::MISC::MiscClkCntl1[10] == 1.</li> </ol>
7:3	Reserved.
2	<b>BP_X48M0_Output_Enable.</b> Read-write. Reset: 1. BP_X48M0 Clock Output Enable. This is S0 type clock and is OFF in sleep state. When this bit is set, 48 MHz clock is On regardless of S0A3. Programming this bit to 0 turns off the clock when in S0 state.
1	<b>CoreSpeedMode.</b> Read-write. Reset: 0. 0=Full speed B-Link clock. 1=Slow speed B-Link clock. Slow down Internal Core Clock (B-Link clock) for power saving.
0	<b>BP_X48M0_OutputEn_S0i3.</b> Read-write. Reset: 0. <b>Description:</b> BP_X48M0 Clock Output Enable at S0i3 state. This is S0 type clock and is Off in sleep state. If a platform does not use this clock, it can be disabled by programming FCH::MISC::MiscClkCntl1[BP_X48M0_Output_Enable] = 0. If a platform use this clock, but would like this clock always running at S0 regardless S0i3, then program BP_X48M0_Output_En_S0i3 = 1. If a platform uses this clock and would like to disable this clock when at S0i3 state, then program BP_X48M0_Output_En_S0i3 = 0 (default).

**MISCx044 [MiscClkCntl2] (FCH::MISC::MiscClkCntl2)**

Reset: 0000\_0000h.

\_aliasHOST; MISCx044; MISC=FED8\_0E00h

Bits	Description
31:0	Reserved.

**MISCx048 [MiscClkCntl3] (FCH::MISC::MiscClkCntl3)**

Read-write. Reset: 0C60\_0000h.

\_aliasHOST; MISCx048; MISC=FED8\_0E00h

Bits	Description
31	Reserved.
30:26	<b>GPP_CLK5_Driver_Impedance_Control.</b> Read-write. Reset: 03h. GPP_CLK5 Driver Impedance Control. Driver impedance control. Controls the number of slices to be turned on.
25:21	<b>GPP_CLK4_Driver_Impedance_Control.</b> Read-write. Reset: 03h. GPP_CLK4 Driver Impedance Control. Driver impedance control. Controls the number of slices to be turned on.
20:0	Reserved.

**MISCx04C [MiscClkCntl4] (FCH::MISC::MiscClkCntl4)**

Read-write. Reset: 0000_0000h.	
_aliasHOST; MISCx04C; MISC=FED8_0E00h	
Bits	Description
31	<b>LowPowerDisplay400MClkEnB.</b> Read-write. Reset: 0. 0=Enable low power display 400 MHz clock output. 1=Disable. Non-sticky bit.
30	<b>LowPowerDisplay300MClkEnB.</b> Read-write. Reset: 0. 0=Enable low power display 300 MHz clock output. 1=Disable. Non-sticky bit.
29:0	Reserved.

**MISCx060 [IdleCntl] (FCH::MISC::IdleCntl)**

Read-write.	
_aliasHOST; MISCx060; MISC=FED8_0E00h	
Bits	Description
31:24	<b>IdleCount.</b> Read-write. Reset: XXh. This returns the idle count from the latest monitored period.
23:0	Reserved.

**MISCx068 [Memory Power Saving Control] (FCH::MISC::MemPwrSavCntl)**

Read-write. Reset: 0000_0006h.	
_aliasHOST; MISCx068; MISC=FED8_0E00h	
Bits	Description
31:3	Reserved.
2	<b>ABBypassMemDsd.</b> Read-write. Reset: 1. 0=Enable memory deep sleep and shutdown features. 1=Disable memory deep sleep and shutdown. AB memory BypassMemdsd control.
1	<b>SBGBypassMemDsd.</b> Read-write. Reset: 1. 0=Enable memory deep sleep and shutdown features. 1=Disable memory deep sleep and shutdown. SBG memory BypassMemdsd control.
0	Reserved.

**MISCx070 [OscFreqCounter] (FCH::MISC::OscFreqCtr)**

Reset: 0000_0000h.	
_aliasHOST; MISCx070; MISC=FED8_0E00h	
Bits	Description
31	<b>CountEnable.</b> Read-write. Reset: 0. 1=Enable the internal counter to count the number of ocellator (OSC) clocks. When software is not using this function, it should always set it to 0 to conserve power.
30	<b>CountIsValid.</b> Read-only. Reset: 0. 1=OscCountPerSec is valid. Software should always wait for this bit to be set before reading OscCountPerSec.
29:28	Reserved.
27:0	<b>OscCountPerSec.</b> Read-write. Reset: 000_0000h. Number of OSC clocks per 1 second. Whenever [CountEnable] is set, an internal counter starts counting the number of OSC clocks per second and records the count value here.

**MISCx078 [PostCode] (FCH::MISC::PostCode)**

Read-only. Reset: 0000_0000h.	
_aliasHOST; MISCx078; MISC=FED8_0E00h	
Bits	Description
31:0	<b>PostCode[31:0].</b> Read-only. Reset: 0000_0000h. BIOS' 32-bit writes to FCH::MISC::PostCode go to this internal 32-bit PostCode Register. Reads from FCH::MISC::PostCode return PostCode[7:0].

**MISCx07C [PostCodeStack] (FCH::MISC::PostCodeStack)**

Read-only. Reset: FFFF_FFFFh.	
_aliasHOST; MISCx07C; MISC=FED8_0E00h	
Bits	Description
31:0	<b>PostCodeStack[31:0].</b> Read-only. Reset: FFFF_FFFFh. 32 deep post code STACK Read out window. Reads get

	32 bits of PostCode. Unused bytes return 0s. If Write is full, the oldest data is lost and the new data fills in. When software Reads out of the stack, it Reads from new data to old data and doesn't get flushed. Extra Reads produce duplicate data.
--	---

**MISCx080 [StrapStatus] (FCH::MISC::StrapStat)**

Read-only.

\_aliasHOST; MISCx080; MISC=FED8\_0E00h

Bits	Description
31:18	Reserved.
17	<b>ClkGenStrap</b> . Read-only. Reset: X. 0=External clocking mode; Uses 100 MHz differential spread clock as the reference clock. 1=Internal clocking mode; Uses 48 MHz crystal clock as the reference clock.
16:12	Reserved.
11	<b>ShortResetStrap</b> . Read-only. Reset: 0. Generate short reset.
10:2	Reserved.
1	<b>UseLpcRomStrap</b> . Read-only. Reset: X. 0=SPI ROM strap. 1=LPC ROM strap. Use LPC ROM.
0	Reserved.

**MISCx088 [PostCode Control] (FCH::MISC::PostCodeControl)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; MISCx088; MISC=FED8\_0E00h

Bits	Description
31:1	Reserved.
0	<b>Post_rd_ptr_rst</b> . Read-write. Reset: 0. Software reset bit for Read pointer. When the new turn of Read is needed, set this bit and the Read pointer goes back for the next cycle's Read.

**MISCx090 [AutoTransaction/Allow EC] (FCH::MISC::AutoTransactionAllowEC)**

Reset: 0000\_0000h.

\_aliasHOST; MISCx090; MISC=FED8\_0E00h

Bits	Description										
31:10	Reserved.										
9	<b>DisableAuto</b> . Read,Write-1-only. Reset: 0. 1=The entire Auto Transaction logic is disabled. Once this bit is set, it cannot be cleared except by system reset.										
8	<b>AllowECToAutoTransactEn</b> . Read-write. Reset: 0. 0=IMC cannot write to any of the registers (FCH::MISC::AutoTransactionAllowEC, FCH::MISC::AutoAddrLo, FCH::MISC::AutoAddrHi, FCH::MISC::AutoData) in the Auto Transaction Generation logic. 1=IMC can change any of these registers. Only BIOS can change this bit.										
7:4	<b>TransactionType</b> . Read-write. Reset: 0h. PCI Command type used for this auto transaction. See the PCI specification for PCI command types.										
3:2	<b>ByteCount</b> . Read-write. Reset: 0h. <b>ValidValues:</b>										
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>1 byte</td></tr> <tr> <td>1h</td><td>2 bytes</td></tr> <tr> <td>2h</td><td>4 bytes</td></tr> <tr> <td>3h</td><td>4 bytes</td></tr> </table>	Value	Description	0h	1 byte	1h	2 bytes	2h	4 bytes	3h	4 bytes
Value	Description										
0h	1 byte										
1h	2 bytes										
2h	4 bytes										
3h	4 bytes										
1	<b>DualAddr</b> . Read-write. Reset: 0. 0=Use single address cycle. 1=Use dual address cycle.										
0	<b>AutoExecute</b> . Read-write. Reset: 0. 1=Causes the hardware to execute the transaction defined by bits[7:1]. Writing 1 to this bit causes the hardware to execute the transaction defined by bits[7:1]. Once it is written, this bit stays set until the transaction is completed, in which case it returns to 0.										

**MISCx094 [AutoAddrLow] (FCH::MISC::AutoAddrLo)**

Read-write. Reset: 0000_0000h.	
_aliasHOST; MISCx094; MISC=FED8_0E00h	
Bits	Description
31:0	<b>AutoAddrLow.</b> Read-write. Reset: 0000_0000h. Low address to be used by the FCH::MISC::AutoTransactionAllowEC[AutoExecute] operation. See also FCH::MISC::AutoAddrHi[AutoAddrHigh].

**MISCx098 [AutoAddrHigh] (FCH::MISC::AutoAddrHi)**

Read-write. Reset: 0000_0000h.	
_aliasHOST; MISCx098; MISC=FED8_0E00h	
Bits	Description
31:0	<b>AutoAddrHigh.</b> Read-write. Reset: 0000_0000h. High address to be used by the FCH::MISC::AutoTransactionAllowEC[AutoExecute] operation. This register is only applicable when FCH::MISC::AutoTransactionAllowEC[DualAddr] == 1. AutoAddr = {AutoAddrHigh, FCH::MISC::AutoAddrLo[AutoAddrLow]}.

**MISCx09C [AutoData] (FCH::MISC::AutoData)**

Read-write. Reset: 0000_0000h.	
_aliasHOST; MISCx09C; MISC=FED8_0E00h	
Bits	Description
31:0	<b>AutoData.</b> Read-write. Reset: 0000_0000h. If FCH::MISC::AutoTransactionAllowEC[TransactionType] is Read, this register returns the Read data. If the FCH::MISC::AutoTransactionAllowEC[TransactionType] is a Write command, this register contains the Write data. NOTE: The byte is aligned accordingly.

**MISCx0A0 [CGPLLCntrl Reg1] (FCH::MISC::CGPLLCntrlreg1)**

Read-write. Reset: 0000_0000h.	
_aliasHOST; MISCx0A0; MISC=FED8_0E00h	
Bits	Description
31	Reserved.
30:23	<b>CG1PLL_kdco_ratio[7:0].</b> Read-write. Reset: 00h. <b>Description:</b> CG1PLL Override: - 5b mantissa, 3b exponent (signed exponent, negative exponent means fraction < 1.0).
22:0	Reserved.

**MISCx0A4 [CGPLLCntrl Reg2] (FCH::MISC::CGPLLCntrlReg2)**

Read-write. Reset: 0000_0000h.	
_aliasHOST; MISCx0A4; MISC=FED8_0E00h	
Bits	Description
31:22	Reserved.
21:8	<b>CG1PLL_lock_timer_13_0_override.</b> Read-write. Reset: 0000h. CG1PLL Override: Lock timer control.
7:0	Reserved.

**MISCx0A8 [CGPLLCntrl Reg3] (FCH::MISC::CGPLLCntrlReg3)**

Reset: 0000_0000h.	
_aliasHOST; MISCx0A8; MISC=FED8_0E00h	
Bits	Description
31:0	Reserved.

**MISCx0AC [CGPLLCntrl Reg4] (FCH::MISC::CGPLLCntrlReg4)**

Read-write. Reset: 0000_0000h.	
_aliasHOST; MISCx0AC; MISC=FED8_0E00h	
Bits	Description
31:24	Reserved.

23:22	<b>CPU_Refclk_Selection_Override.</b> Read-write. Reset: 0h. <b>Description:</b> CPU REFCLK Selection Override (XOR). <b>NOTE:</b> 1. Misc_Reg xBC[13] to select whether EXT_GPP0_SRC is from GPP0 external input or GPP0 external input with divide-by-2. 2. Should apply a cold reset with CPU REFCLK source change. <b>ValidValues:</b> <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>100MHz CG1_PLL generated (default single die package with internal clock mode).</td></tr> <tr> <td>1h</td><td>EXT_GPP0_SRC.</td></tr> <tr> <td>2h</td><td>External input thru GPP1 (default in single die package with external clock mode).</td></tr> <tr> <td>3h</td><td>Reserved.</td></tr> </table>	Value	Description	0h	100MHz CG1_PLL generated (default single die package with internal clock mode).	1h	EXT_GPP0_SRC.	2h	External input thru GPP1 (default in single die package with external clock mode).	3h	Reserved.
Value	Description										
0h	100MHz CG1_PLL generated (default single die package with internal clock mode).										
1h	EXT_GPP0_SRC.										
2h	External input thru GPP1 (default in single die package with external clock mode).										
3h	Reserved.										
21:20	<b>SMU_100M_Refclk_Selection.</b> Read-write. Reset: 0h. SMU_100M REFCLK Selection. <b>ValidValues:</b> <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>100MHz CG1_PLL generated (default).</td></tr> <tr> <td>1h</td><td>100MHz GPP0 input from external.</td></tr> <tr> <td>2h</td><td>GPP0 input with divide-by-2.</td></tr> <tr> <td>3h</td><td>Reserved.</td></tr> </table>	Value	Description	0h	100MHz CG1_PLL generated (default).	1h	100MHz GPP0 input from external.	2h	GPP0 input with divide-by-2.	3h	Reserved.
Value	Description										
0h	100MHz CG1_PLL generated (default).										
1h	100MHz GPP0 input from external.										
2h	GPP0 input with divide-by-2.										
3h	Reserved.										
19:8	Reserved.										
7:6	<b>PHY_A3_4_Refclk_Selection.</b> Read-write. Reset: 0h. <b>Description:</b> PHY_A3_4 REFCLK Selection. <b>NOTE:</b> FCH::MISC::CGPLLCtrlReg8[EXT_GPP0_REFCLK_SEL] to select whether EXT_GPP0_SRC is from GPP0 external input or GPP0 external input with divide-by-2. <b>ValidValues:</b> <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>100 MHz CG1_PLL generated (default).</td></tr> <tr> <td>1h</td><td>133 MHz CG1_PLL generated.</td></tr> <tr> <td>2h</td><td>EXT_GPP0_SRC.</td></tr> <tr> <td>3h</td><td>48 MHz, buffered version from CG_XTAL input.</td></tr> </table>	Value	Description	0h	100 MHz CG1_PLL generated (default).	1h	133 MHz CG1_PLL generated.	2h	EXT_GPP0_SRC.	3h	48 MHz, buffered version from CG_XTAL input.
Value	Description										
0h	100 MHz CG1_PLL generated (default).										
1h	133 MHz CG1_PLL generated.										
2h	EXT_GPP0_SRC.										
3h	48 MHz, buffered version from CG_XTAL input.										
5:4	<b>PHY_A2_Refclk_Selection.</b> Read-write. Reset: 0h. <b>Description:</b> PHY_A2 REFCLK Selection. <b>NOTE:</b> FCH::MISC::CGPLLCtrlReg8[EXT_GPP0_REFCLK_SEL] to select whether EXT_GPP0_SRC is from GPP0 external input or GPP0 external input with divide-by-2. <b>ValidValues:</b> <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>100 MHz CG1_PLL generated (default).</td></tr> <tr> <td>1h</td><td>133 MHz CG1_PLL generated.</td></tr> <tr> <td>2h</td><td>EXT_GPP0_SRC.</td></tr> <tr> <td>3h</td><td>48 MHz, buffered version from CG_XTAL input.</td></tr> </table>	Value	Description	0h	100 MHz CG1_PLL generated (default).	1h	133 MHz CG1_PLL generated.	2h	EXT_GPP0_SRC.	3h	48 MHz, buffered version from CG_XTAL input.
Value	Description										
0h	100 MHz CG1_PLL generated (default).										
1h	133 MHz CG1_PLL generated.										
2h	EXT_GPP0_SRC.										
3h	48 MHz, buffered version from CG_XTAL input.										
3:2	<b>PHY_A1_Refclk_Selection.</b> Read-write. Reset: 0h. <b>Description:</b> PHY_A1 REFCLK Selection. <b>NOTE:</b> FCH::MISC::CGPLLCtrlReg8[EXT_GPP0_REFCLK_SEL] to select whether EXT_GPP0_SRC is from GPP0 external input or GPP0 external input with divide-by-2. <b>ValidValues:</b> <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>100 MHz CG1_PLL generated (default).</td></tr> <tr> <td>1h</td><td>133 MHz CG1_PLL generated.</td></tr> <tr> <td>2h</td><td>EXT_GPP0_SRC.</td></tr> </table>	Value	Description	0h	100 MHz CG1_PLL generated (default).	1h	133 MHz CG1_PLL generated.	2h	EXT_GPP0_SRC.		
Value	Description										
0h	100 MHz CG1_PLL generated (default).										
1h	133 MHz CG1_PLL generated.										
2h	EXT_GPP0_SRC.										

	3h	48 MHz, buffered version from CG_XTAL input.
1:0	<b>PHY_A0_Refclk_Selection.</b> Read-write. Reset: 0h. <b>Description:</b> PHY_A0 REFCLK Selection. NOTE: FCH::MISC::CGPLLCntlReg8[EXT_GPP0_REFCLK_SEL] to select whether EXT_GPP0_SRC is from GPP0 external input or GPP0 external input with divide-by-2. <b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	100 MHz CG1_PLL generated (default).
	1h	133 MHz CG1_PLL generated.
	2h	EXT_GPP0_SRC.
	3h	48 MHz, buffered version from CG_XTAL input.

#### MISCx0BC [CGPLLCntl Reg8] (FCH::MISC::CGPLLCntlReg8)

Read-write. Reset: 0000\_0000h.

\_aliasHOST; MISCx0BC; MISC=FED8\_0E00h

Bits	Description								
31:28	Reserved.								
27:26	<b>GPP5_Refclk_Selection.</b> Read-write. Reset: 0h. GPP5 REFCLK selection. <b>ValidValues:</b> <table><tr><th>Value</th><th>Description</th></tr><tr><td>0h</td><td>CG_PLL generated 100 MHz.</td></tr><tr><td>3h-1h</td><td>Reserved.</td></tr></table>	Value	Description	0h	CG_PLL generated 100 MHz.	3h-1h	Reserved.		
Value	Description								
0h	CG_PLL generated 100 MHz.								
3h-1h	Reserved.								
25:24	<b>GPP4_Refclk_Selection.</b> Read-write. Reset: 0h. GPP4 REFCLK selection. <b>ValidValues:</b> <table><tr><th>Value</th><th>Description</th></tr><tr><td>0h</td><td>CG_PLL generated 100 MHz.</td></tr><tr><td>3h-1h</td><td>Reserved.</td></tr></table>	Value	Description	0h	CG_PLL generated 100 MHz.	3h-1h	Reserved.		
Value	Description								
0h	CG_PLL generated 100 MHz.								
3h-1h	Reserved.								
23:22	<b>GPP3_Refclk_Selection.</b> Read-write. Reset: 0h. GPP3 REFCLK selection. NOTE: FCH::MISC::CGPLLCntrlReg8[EXT_GPP0_REFCLK_SEL] to select EXT_GPP0_SRC either is from GPP0 external input or GPP0 external input with divide-by-2. <b>ValidValues:</b> <table><tr><th>Value</th><th>Description</th></tr><tr><td>0h</td><td>CG_PLL generated 100 MHz.</td></tr><tr><td>1h</td><td>EXT_GPP0_SRC.</td></tr><tr><td>3h-2h</td><td>Reserved.</td></tr></table>	Value	Description	0h	CG_PLL generated 100 MHz.	1h	EXT_GPP0_SRC.	3h-2h	Reserved.
Value	Description								
0h	CG_PLL generated 100 MHz.								
1h	EXT_GPP0_SRC.								
3h-2h	Reserved.								
21:20	<b>GPP2_Refclk_Selection.</b> Read-write. Reset: 0h. GPP2 REFCLK selection. NOTE: FCH::MISC::CGPLLCntrlReg8[EXT_GPP0_REFCLK_SEL] to select EXT_GPP0_SRC either is from GPP0 external input or GPP0 external input with divide-by-2. <b>ValidValues:</b> <table><tr><th>Value</th><th>Description</th></tr><tr><td>0h</td><td>CG_PLL generated 100 MHz.</td></tr><tr><td>1h</td><td>EXT_GPP0_SRC.</td></tr><tr><td>3h-2h</td><td>Reserved.</td></tr></table>	Value	Description	0h	CG_PLL generated 100 MHz.	1h	EXT_GPP0_SRC.	3h-2h	Reserved.
Value	Description								
0h	CG_PLL generated 100 MHz.								
1h	EXT_GPP0_SRC.								
3h-2h	Reserved.								
19:18	<b>GPP1_Refclk_Selection.</b> Read-write. Reset: 0h. GPP1 REFCLK selection. NOTE: FCH::MISC::CGPLLCntrlReg8[EXT_GPP0_REFCLK_SEL] to select EXT_GPP0_SRC either is from GPP0 external input or GPP0 external input with divide-by-2. <b>ValidValues:</b> <table><tr><th>Value</th><th>Description</th></tr><tr><td>0h</td><td>CG_PLL generated 100 MHz.</td></tr></table>	Value	Description	0h	CG_PLL generated 100 MHz.				
Value	Description								
0h	CG_PLL generated 100 MHz.								



	1h	EXT_GPP0_SRC.
	3h-2h	Reserved.
17:16	<b>GPP0_Refclk_Selection.</b> Read-write. Reset: 0h. GPP0 REFCLK selection. <b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	CG_PLL generated 100 MHz.
	3h-1h	Reserved.
15	<b>EXT_BYPASSCLK_EN.</b> Read-write. Reset: 0. 0=Enable. 1=Disable. EXT_BYPASSCLK_EN. Enable/Disable REF_BYPASSCLK external input thru GFX0.	
14	<b>CLK_CGPLL_EXT_PWDN.</b> Read-write. Reset: 0. 0=Enable. 1=Disable. CLK_CGPLL_EXT_PWDN. Enable/Disable CGPLL external REFCLK source thru GPP0 input.	
13	<b>EXT_GPP0_REFCLK_SEL.</b> Read-write. Reset: 0. 0=GPP0 external input. 1=GPP0 external input divided-by-2. EXT_GPP0_REFCLK_SEL.	
12	Reserved.	
11	<b>CPU_Refclk_Driver_PWDN.</b> Read-write. Reset: 0. 0=Enable. 1=Disable. CPU REFCLK driver power down.	
10	<b>SMU_100M_Refclk_Driver_PWDN.</b> Read-write. Reset: 0. 0=Enable. 1=Disable. SMU_100M REFCLK driver power down.	
9	Reserved.	
8	<b>PHY_USB3_Refclk_Driver_PWDN.</b> Read-write. Reset: 0. 0=Enable. 1=Disable. PHY_USB3 REFCLK driver power down in S0i3.	
7:0	Reserved.	

**MISCx0C0 [IoTrapping0 IoTrapping1] (FCH::MISC::IoTrapping0IoTrapping1)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; MISCx0C0; MISC=FED8\_0E00h

Bits	Description
31:16	<b>IoTrappingAdr1.</b> Read-write. Reset: 0000h. Specify the IO address 1, which causes an SMI event.
15:0	<b>IoTrappingAdr0.</b> Read-write. Reset: 0000h. Specify the IO address 0, which causes an SMI event.

**MISCx0C4 [IoTrapping2 IoTrapping3] (FCH::MISC::IoTrapping2IoTrapping3)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; MISCx0C4; MISC=FED8\_0E00h

Bits	Description
31:16	<b>IoTrappingAdr3.</b> Read-write. Reset: 0000h. Specify the IO address 3, which causes an SMI event.
15:0	<b>IoTrappingAdr2.</b> Read-write. Reset: 0000h. Specify the IO address 2, which causes an SMI event.

**MISCx0C8 [CfgTrapping0] (FCH::MISC::CfgTrapping0)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; MISCx0C8; MISC=FED8\_0E00h

Bits	Description
31:2	<b>CfgTrappingAdr0.</b> Read-write. Reset: 0000_0000h. Specify the configuration address 0, which causes an SMI event.
1:0	Reserved.

**MISCx0CC [SMITrappingRwRdOvr] (FCH::MISC::SMITrappingRwRdOvr)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; MISCx0CC; MISC=FED8\_0E00h

Bits	Description
31:17	Reserved.
16	<b>CfgTrappingRw0.</b> Read-write. Reset: 0. 0=Trap on configuration Read access on the address specified in FCH::MISC::CfgTrapping0[CfgTrappingAdr0]. 1=Trap on configuration Write access on the address specified in

	FCH::MISC::CfgTrapping0[CfgTrappingAdr0].
15:13	Reserved.
12	<b>MemTrappingRdOvr0</b> . Read-write. Reset: 0. Set to 1 to force Read data to be replaced by FCH::MISC::MemRdOvrData0.
11:9	Reserved.
8	<b>MemTrappingRw0</b> . Read-write. Reset: 0. 0=Trap on memory Read access on the address specified in FCH::MISC::MemTrapping0[MemTrappingAdr0]. 1=Trap on memory Write access on the address specified in FCH::MISC::MemTrapping0[MemTrappingAdr0].
7:4	Reserved.
3	<b>IoTrappingRw3</b> . Read-write. Reset: 0. 0=Trap on IO Read access on the address specified in FCH::MISC::IoTrapping2IoTrapping3[IoTrappingAdr3]. 1=Trap on IO Write access on the address specified in FCH::MISC::IoTrapping2IoTrapping3[IoTrappingAdr3].
2	<b>IoTrappingRw2</b> . Read-write. Reset: 0. 0=Trap on IO Read access on the address specified in FCH::MISC::IoTrapping2IoTrapping3[IoTrappingAdr2]. 1=Trap on IO Write access on the address specified in FCH::MISC::IoTrapping2IoTrapping3[IoTrappingAdr2].
1	<b>IoTrappingRw1</b> . Read-write. Reset: 0. 0=Trap on IO Read access on the address specified in FCH::MISC::IoTrapping0IoTrapping1[IoTrappingAdr1]. 1=Trap on IO Write access on the address specified in FCH::MISC::IoTrapping0IoTrapping1[IoTrappingAdr1].
0	<b>IoTrappingRw0</b> . Read-write. Reset: 0. 0=Trap on IO Read access on the address specified in FCH::MISC::IoTrapping0IoTrapping1[IoTrappingAdr0]. 1=Trap on IO Write access on the address specified in FCH::MISC::IoTrapping0IoTrapping1[IoTrappingAdr0].

**MISCx0D0 [MemTrapping0] (FCH::MISC::MemTrapping0)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; MISCx0D0; MISC=FED8\_0E00h

Bits	Description
31:2	<b>MemTrappingAdr0</b> . Read-write. Reset: 0000_0000h. Specify the 30-bit memory address 0 which causes an SMI even. Lowest 2 bits are ignored.
1:0	Reserved.

**MISCx0D4 [MemRdOvrData0] (FCH::MISC::MemRdOvrData0)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; MISCx0D4; MISC=FED8\_0E00h

Bits	Description
31:0	<b>MemTrappingRData0</b> . Read-write. Reset: 0000_0000h. The 32-bit data is used as the return data when the memory Read trapping is enabled in FCH::MISC::MemTrapping0, with FCH::MISC::SMITrappingRwRdOvr[MemTrappingRdOvr0] == 1.

**MISCx0D8 [I2C0\_PadCtrl] (FCH::MISC::I2C0\_PadCtrl)**

Read-write. Reset: 0000\_0010h.

\_aliasHOST; MISCx0D8; MISC=FED8\_0E00h

Bits	Description
31:19	Reserved.
18	<b>Spare1</b> . Read-write. Reset: 0. Spare pins.
17	<b>Spare0</b> . Read-write. Reset: 0. Spare pins.
16	<b>BiasCrtEn</b> . Read-write. Reset: 0. 0=Disable. 1=Enable. Pbias should be on in Fast/Fast+. To save power, it can be turned off in I2C Standard Mode.
15	<b>RSel1p1</b> . Read-write. Reset: 0. 1=Increases resistance by 10% for all RC timers. When asserted, increases resistance by 10% for all RC timers.
14	<b>RSel0p9</b> . Read-write. Reset: 0. 1=Decreases resistance by 10% for all RC timers. When asserted, decreases resistance by 10% for all RC timers.
13	<b>CSel1p1</b> . Read-write. Reset: 0. 1=Increases capacitance by 10% for all RC timers. When asserted, increases

	capacitance by 10% for all RC timers.										
12	<b>CSel0p9</b> . Read-write. Reset: 0. 1=Decreases capacitance by 10% for all RC timers. When asserted, decreases capacitance by 10% for all RC timers.										
11	<b>SpikeRcSel</b> . Read-write. Reset: 0. 0=50ns. 1=20ns. Select RC constant for I2C spike suppression.										
10	<b>SpikeRcEn</b> . Read-write. Reset: 0. 0=Disable. 1=Enable RX spike suppression. Enable spike suppression.										
9	<b>Slewn</b> . Read-write. Reset: 0. 0=Disabled. 1=Enabled. Enable changing strength of pre-driver for fall time compensation.										
8:7	<b>FallSlewSel</b> . Read-write. Reset: 0h. Slew rate. <b>ValidValues:</b>										
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>Standard Mode TX frequency is 100 KHz.</td></tr> <tr> <td>1h</td><td>Low speed from 12ns to 120ns; TX frequency &lt; 1MHz.</td></tr> <tr> <td>3h-2h</td><td>Reserved.</td></tr> </table>	Value	Description	0h	Standard Mode TX frequency is 100 KHz.	1h	Low speed from 12ns to 120ns; TX frequency < 1MHz.	3h-2h	Reserved.		
Value	Description										
0h	Standard Mode TX frequency is 100 KHz.										
1h	Low speed from 12ns to 120ns; TX frequency < 1MHz.										
3h-2h	Reserved.										
6	<b>PdEn</b> . Read-write. Reset: 0. 0=Disable. 1=Enable. Pull-down enable for DDCCLK (PADP).										
5:4	<b>I2cRxSel</b> . Read-write. Reset: 1h. I2C Receiver select. <b>ValidValues:</b>										
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>All receivers Off.</td></tr> <tr> <td>1h</td><td>Schmitt trigger for 3.3v input.</td></tr> <tr> <td>2h</td><td>Schmitt trigger for 3.3v input.</td></tr> <tr> <td>3h</td><td>Schmitt trigger for 1.8v input.</td></tr> </table>	Value	Description	0h	All receivers Off.	1h	Schmitt trigger for 3.3v input.	2h	Schmitt trigger for 3.3v input.	3h	Schmitt trigger for 1.8v input.
Value	Description										
0h	All receivers Off.										
1h	Schmitt trigger for 3.3v input.										
2h	Schmitt trigger for 3.3v input.										
3h	Schmitt trigger for 1.8v input.										
3:0	<b>NG: N Strength Control</b> . Read-write. Reset: 0h. Init: BIOS,Ch. N Strength control. Software should program to 0Ch for normal drive strength.										

**MISCx0DC [I2C1\_PadCtrl] (FCH::MISC::I2C1\_PadCtrl)**

Read-write. Reset: 0000_0010h.							
_aliasHOST; MISCx0DC; MISC=FED8_0E00h							
Bits	Description						
31:19	Reserved.						
18	<b>Spare1</b> . Read-write. Reset: 0. Spare pins.						
17	<b>Spare0</b> . Read-write. Reset: 0. Spare pins.						
16	<b>BiasCrtEn</b> . Read-write. Reset: 0. 0=Disable. 1=Enable. Pbias should be on in Fast/Fast+. To save power, it can be turned off in I2C Standard Mode.						
15	<b>RSel1p1</b> . Read-write. Reset: 0. 1=Increases resistance by 10% for all RC timers. When asserted, increases resistance by 10% for all RC timers.						
14	<b>RSel0p9</b> . Read-write. Reset: 0. 1=Decreases resistance by 10% for all RC timers. When asserted, decreases resistance by 10% for all RC timers.						
13	<b>CSel1p1</b> . Read-write. Reset: 0. 1=Increases capacitance by 10% for all RC timers. When asserted, increases capacitance by 10% for all RC timers.						
12	<b>CSel0p9</b> . Read-write. Reset: 0. 1=Decreases capacitance by 10% for all RC timers. When asserted, decreases capacitance by 10% for all RC timers.						
11	<b>SpikeRcSel</b> . Read-write. Reset: 0. 0=50ns. 1=20ns. Select RC constant for I2C spike suppression.						
10	<b>SpikeRcEn</b> . Read-write. Reset: 0. 0=Disable. 1=Enable RX spike suppression. Enable spike suppression.						
9	<b>Slewn</b> . Read-write. Reset: 0. 0=Disabled. 1=Enabled. Enable changing strength of pre-driver for fall time compensation.						
8:7	<b>FallSlewSel</b> . Read-write. Reset: 0h. Slew rate. <b>ValidValues:</b>						
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>Standard Mode TX frequency is 100 KHz.</td></tr> <tr> <td>1h</td><td>Low speed from 12ns to 120ns; TX frequency &lt; 1 MHz.</td></tr> </table>	Value	Description	0h	Standard Mode TX frequency is 100 KHz.	1h	Low speed from 12ns to 120ns; TX frequency < 1 MHz.
Value	Description						
0h	Standard Mode TX frequency is 100 KHz.						
1h	Low speed from 12ns to 120ns; TX frequency < 1 MHz.						

	3h-2h	Reserved.
6	<b>PdEn.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. Pull-down enable for DDCCLK (PADP).	
5:4	<b>I2cRxSel.</b> Read-write. Reset: 1h. I2C receiver select.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	All receivers Off.
	1h	Schmitt trigger for 3.3v input.
	2h	Schmitt trigger for 3.3v input.
	3h	Schmitt trigger for 1.8v input.
3:0	<b>NG: N Strength Control.</b> Read-write. Reset: 0h. Init: BIOS,Ch. N Strength Control. Software should program to 0Ch for normal drive strength.	

**MISCx0E0 [I2C2\_PadCtrl] (FCH::MISC::I2C2\_PadCtrl)**

Read-write. Reset: 0000\_0010h.

\_aliasHOST; MISCx0E0; MISC=FED8\_0E00h

Bits	Description
31:19	Reserved.
18	<b>Spare1.</b> Read-write. Reset: 0. Spare pins.
17	<b>Spare0.</b> Read-write. Reset: 0. Spare pins.
16	<b>BiasCrtEn.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. Pbias should be on in Fast/Fast+. To save power, it can be turned off in I2C Standard Mode.
15	<b>RSel1p1.</b> Read-write. Reset: 0. 1=Increases resistance by 10% for all RC timers. When asserted, increases resistance by 10% for all RC timers.
14	<b>RSel0p9.</b> Read-write. Reset: 0. 1=Decreases resistance by 10% for all RC timers. When asserted, decreases resistance by 10% for all RC timers.
13	<b>CSel1p1.</b> Read-write. Reset: 0. 1=Increases capacitance by 10% for all RC timers. When asserted, increases capacitance by 10% for all RC timers.
12	<b>CSel0p9.</b> Read-write. Reset: 0. 1=Decreases capacitance by 10% for all RC timers. When asserted, decreases capacitance by 10% for all RC timers.
11	<b>SpikeRcSel.</b> Read-write. Reset: 0. 0=50ns. 1=20ns. Select RC constant for I2C spike suppression.
10	<b>SpikeRcEn.</b> Read-write. Reset: 0. 0=Disable. 1=Enable RX spike suppression. Enable spike suppression.
9	<b>Slewn.</b> Read-write. Reset: 0. 0=Disabled. 1=Enabled. Enable changing strength of pre-driver for fall time compensation.
8:7	<b>FallSlewSel.</b> Read-write. Reset: 0h. Slew rate.
	<b>ValidValues:</b>
	<b>Value</b> <b>Description</b>
	0h Standard Mode TX frequency is 100 KHz.
	1h Low speed from 12ns to 120ns; TX frequency < 1 MHz.
	3h-2h Reserved.
6	<b>PdEn.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. Pull-down enable for DDCCLK (PADP).
5:4	<b>I2cRxSel.</b> Read-write. Reset: 1h. I2C receiver select.
	<b>ValidValues:</b>
	<b>Value</b> <b>Description</b>
	0h All receivers Off.
	1h Schmitt trigger for 3.3v input.
	2h Schmitt trigger for 3.3v input.
	3h Schmitt trigger for 1.8v input.
3:0	<b>NG: N Strength Control.</b> Read-write. Reset: 0h. Init: BIOS,Ch. N Strength Control. Software should program to 0Ch for normal drive strength.

**MISCx0E4 [I2C3\_PadCtrl] (FCH::MISC::I2C3\_PadCtrl)**

Read-write. Reset: 0000\_0010h.

\_aliasHOST; MISCx0E4; MISC=FED8\_0E00h

Bits	Description										
31:19	Reserved.										
18	<b>Spare1</b> . Read-write. Reset: 0. Spare pin.										
17	<b>Spare0</b> . Read-write. Reset: 0. Spare pin.										
16	<b>BiasCrtEn</b> . Read-write. Reset: 0. 0=Disable. 1=Enable. Pbias should be on in Fast/Fast+. To save power, it can be turned off in I2C Standard Mode.										
15	<b>RSel1p1</b> . Read-write. Reset: 0. 1=Increases resistance by 10% for all RC timers. When asserted, increases resistance by 10% for all RC timers.										
14	<b>RSel0p9</b> . Read-write. Reset: 0. 1=Decreases resistance by 10% for all RC timers. When asserted, decreases resistance by 10% for all RC timers.										
13	<b>CSel1p1</b> . Read-write. Reset: 0. 1=Increases capacitance by 10% for all RC timers. When asserted, increases capacitance by 10% for all RC timers.										
12	<b>CSel0p9</b> . Read-write. Reset: 0. 1=Decreases capacitance by 10% for all RC timers. When asserted, decreases capacitance by 10% for all RC timers.										
11	<b>SpikeRcSel</b> . Read-write. Reset: 0. 0=50ns. 1=20ns. Select RC constant for I2C spike suppression.										
10	<b>SpikeRcEn</b> . Read-write. Reset: 0. 0=Disable. 1=Enable RX spike suppression. Enable spike suppression.										
9	<b>Slewn</b> . Read-write. Reset: 0. 0=Disabled. 1=Enabled. Enable changing strength of pre-driver for fall time compensation.										
8:7	<b>FallSlewSel</b> . Read-write. Reset: 0h. Slew rate. <b>ValidValues:</b>										
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>Standard Mode TX frequency is 100 KHz.</td></tr> <tr> <td>1h</td><td>Low speed from 12ns to 120ns; TX frequency &lt; 1 MHz.</td></tr> <tr> <td>3h-2h</td><td>Reserved.</td></tr> </table>	Value	Description	0h	Standard Mode TX frequency is 100 KHz.	1h	Low speed from 12ns to 120ns; TX frequency < 1 MHz.	3h-2h	Reserved.		
Value	Description										
0h	Standard Mode TX frequency is 100 KHz.										
1h	Low speed from 12ns to 120ns; TX frequency < 1 MHz.										
3h-2h	Reserved.										
6	<b>PdEn</b> . Read-write. Reset: 0. 0=Disable. 1=Enable. Pull-down enable for DDCCLK (PADP).										
5:4	<b>I2cRxSel</b> . Read-write. Reset: 1h. I2C receiver select. <b>ValidValues:</b>										
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>All receivers Off.</td></tr> <tr> <td>1h</td><td>Schmitt trigger for 3.3v input.</td></tr> <tr> <td>2h</td><td>Schmitt trigger for 3.3v input.</td></tr> <tr> <td>3h</td><td>Schmitt trigger for 1.8v input.</td></tr> </table>	Value	Description	0h	All receivers Off.	1h	Schmitt trigger for 3.3v input.	2h	Schmitt trigger for 3.3v input.	3h	Schmitt trigger for 1.8v input.
Value	Description										
0h	All receivers Off.										
1h	Schmitt trigger for 3.3v input.										
2h	Schmitt trigger for 3.3v input.										
3h	Schmitt trigger for 1.8v input.										
3:0	<b>NG: N Strength Control</b> . Read-write. Reset: 0h. Init: BIOS,Ch. N Strength Control. Software should program to 0Ch for normal drive strength.										

**MISCx0F0 [SataPortSts] (FCH::MISC::SataPortSts)**

Read-write.

\_aliasHOST; MISCx0F0; MISC=FED8\_0E00h

Bits	Description										
31:26	Reserved.										
25:24	<b>SataPortSel</b> . Read-write. Reset: 0h. <b>ValidValues:</b>										
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>Select "led" for Port 0 to 1.</td></tr> <tr> <td>1h</td><td>Select "det" for Port 0 to 1.</td></tr> <tr> <td>2h</td><td>Select "err" for Port 0 to 1.</td></tr> <tr> <td>3h</td><td>Select "led" for Port 0 to 1.</td></tr> </table>	Value	Description	0h	Select "led" for Port 0 to 1.	1h	Select "det" for Port 0 to 1.	2h	Select "err" for Port 0 to 1.	3h	Select "led" for Port 0 to 1.
Value	Description										
0h	Select "led" for Port 0 to 1.										
1h	Select "det" for Port 0 to 1.										
2h	Select "err" for Port 0 to 1.										
3h	Select "led" for Port 0 to 1.										

23:2	Reserved.
1	<b>Port1Sts.</b> Read-write. Reset: X. The selected status of Port 1. This status bit indicates the internal status of SATA port 1.
0	<b>Port0Sts.</b> Read-write. Reset: X. The selected status of Port 0. This status bit indicates the internal status of SATA port 0.

**MISCx0F4 [ClkCntrlSts] (FCH::MISC::ClkCntrlSts)**

Reset: 0000\_0000h.

\_aliasHOST; MISCx0F4; MISC=FED8\_0E00h

Bits	Description
31:0	Reserved.

**MISCx0FC [ClkGatingCntrl] (FCH::MISC::ClkGatingCntrl)**

Read-write. Reset: 0000\_1FFFh.

\_aliasHOST; MISCx0FC; MISC=FED8\_0E00h

Bits	Description
31:13	Reserved.
12	<b>WatchDogTimerBlkClkGatingDis.</b> Read-write. Reset: 1. 0=The clock of the WatchDog Timer block can be stopped if not being accessed. 1=The clock of the WatchDog Timer block can't be stopped if not being accessed. Watchdog Timer block clock gating.
11	<b>AcDcTmrBlkClkGatingDis.</b> Read-write. Reset: 1. 0=The clock of the AcDc Timer block can be stopped if not being accessed. 1=The clock of the AcDc Timer block can't be stopped if not being accessed. AcDc Timer block clock gating.
10	<b>BiosRamBlkClkGatingDis.</b> Read-write. Reset: 1. 0=The clock of the BIOS RAM block can be stopped if not being accessed. 1=The clock of the BIOS RAM block can't be stopped if not being accessed. BIOS RAM block clock gating.
9	<b>HwThrotBlkClkGatingDis.</b> Read-write. Reset: 1. 0=The clock of the hardware throttling block can be stopped if not being accessed. 1=The clock of the hardware throttling block can't be stopped if not being accessed. Hardware throttling block clock gating.
8	<b>CStateBlkClkGatingDis.</b> Read-write. Reset: 1. 0=The clock of the C-State block can be stopped if not being accessed. 1=The clock of the C-State block can't be stopped if not being accessed. C-State block clock gating.
7	<b>FusionCBlkClkGatingDis.</b> Read-write. Reset: 1. 0=The clock of the FusionC block can be stopped if not being accessed. 1=The clock of the FusionC block can't be stopped if not being accessed. FusionC block clock gating.
6	<b>OBFFBlkClkGatingDis.</b> Read-write. Reset: 1. 0=The clock of the OBFF block can be stopped if not being accessed. 1=The clock of the OBFF block can't be stopped if not being accessed. OBFF block clock gating.
5	<b>Smbus0ClkGatingDis.</b> Read-write. Reset: 1. 0=The clock of the SMBUS0 block can be stopped if not being accessed. 1=The clock of the SMBUS0 block can't be stopped if not being accessed. SMBUS0 block clock gating.
4	<b>VWClkGatingDis.</b> Read-write. Reset: 1. 0=The clock of the Virtual Wire block can be stopped if not being accessed. 1=The clock of the Virtual Wire block can't be stopped if not being accessed. Virtual Wire block clock gating.
3	<b>AoacRegBlkClkGatingDis.</b> Read-write. Reset: 1. 0=The clock of the AOAC register block can be stopped if not being accessed. 1=The clock of the AOAC register block can't be stopped if not being accessed. Always On Always Connected register block clock gating.
2	<b>Pmio2RegBlkClkGatingDis.</b> Read-write. Reset: 1. 0=The clock of PMIO2 register block can be stopped if not being accessed. 1=The clock of the PMIO2 register block can't be stopped if not being accessed. Power Management IO 2 register block clock gating.
1	<b>PmioRegBlkClkGatingDis.</b> Read-write. Reset: 1. 0=The clock of the PMIO register block can be stopped if not being accessed. 1=The clock of the PMIO register block can't be stopped if not being accessed. Power Management IO register block clock gating.
0	<b>MIscRegBlkClkGatingDis.</b> Read-write. Reset: 1. 0=The clock of the MISC register block can be stopped if not being accessed. 1=The clock of the MISC register block can't be stopped if not being accessed. Miscellaneous register block clock gating.



**9.2.10.2 Power Management (PM) Registers***Table 79: ACPI MMIO Space Allocation*

00FFh-0000h	SMBus PCI configuration registers
01FFh-0100h	Reserved.
02FFh-0200h	SMI
03FFh-0300h	PMIO
04FFh-0400h	PMIO2
05FFh-0500h	BIOS RAM
06FFh-0600h	CMOS RAM
07FFh-0700h	CMOS
08FFh-0800h	ACPI
09FFh-0900h	ASF registers
0AFFh-0A00h	SMBus registers
0BFFh-0B00h	Watchdog registers
0CFFh-0C00h	HPET
0DFFh-0D00h	IOMux
0EFFh-0E00h	Miscellaneous registers
10FFh-1000h	Reserved. Serial Debug bus
11FFh-1100h	Shadow System Counter.
13FFh-1200h	Reserved
14FFh-1400h	DP-VGA
15FFh-1500h	GPIO Bank 0
16FFh-1600h	GPIO Bank 1
17FFh-1700h	GPIO Bank 2
1BFFh-1800h	Reserved.
1CFFh-1C00h	Reserved.
1DFFh-1D00h	Wake Device (AC DC timer)
1EFFh-1E00h	AOAC Registers
1FFFh-1F00h	Reserved.

The PM register space is accessed through two methods:

- Indirect IO access through IOCD6 [PM\_Index] and IOCD7 [PM\_Data]. Software first programs the offset into the index register IOCD6 and then reads from or writes to the data register IOCD7.
- Direct memory mapped access through the AcpiMmio region. The PM registers range from FED8\_0000h+300h to FED8\_0000h+3FFh. See PMx04[MmioEn] for details on the AcpiMmio region.

**PMx000 [DecodeEn] (FCH::PM::PmDecodeEn)**

Read-write. Reset: 2302\_0B10h.

\_aliasHOST; PMx000; PM=FED8\_0300h

Bits	Description
31:30	Reserved.
29	<b>HpetMsiEn.</b> Read-write. Reset: 1. 1=Expose MSI capability in HPET Capability register.
28	<b>HpetWidthSel.</b> Read-write. Reset: 0. 0=HPET is 32 bits. 1=HPET is 64 bits.
27:26	<b>WatchDogOptions.</b> Read-write. Reset: 0h. Enable for normal WatchDog Timer operation.
<b>Valid Values:</b>	
Value	Description
0h	Enable WatchDog Timer.
3h-1h	Reserved.



25:24	<b>WatchDogFreq.</b> Read-write. Reset: 3h. Defines the clock frequency used by the WatchDog Timer. <b>ValidValues:</b>																		
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>32us</td></tr> <tr> <td>1h</td><td>10ms</td></tr> <tr> <td>2h</td><td>100ms</td></tr> <tr> <td>3h</td><td>1s</td></tr> </table>	Value	Description	0h	32us	1h	10ms	2h	100ms	3h	1s								
Value	Description																		
0h	32us																		
1h	10ms																		
2h	100ms																		
3h	1s																		
23:21	<b>AsfClkSel.</b> Read-write. Reset: 0h. Specifies the frequency of ASF master clock. <b>ValidValues:</b>																		
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>~100kHz</td></tr> <tr> <td>1h</td><td>~200kHz</td></tr> <tr> <td>2h</td><td>~300kHz</td></tr> <tr> <td>3h</td><td>~400kHz</td></tr> <tr> <td>4h</td><td>~600kHz</td></tr> <tr> <td>5h</td><td>~800kHz</td></tr> <tr> <td>6h</td><td>~900kHz</td></tr> <tr> <td>7h</td><td>~1MHz</td></tr> </table>	Value	Description	0h	~100kHz	1h	~200kHz	2h	~300kHz	3h	~400kHz	4h	~600kHz	5h	~800kHz	6h	~900kHz	7h	~1MHz
Value	Description																		
0h	~100kHz																		
1h	~200kHz																		
2h	~300kHz																		
3h	~400kHz																		
4h	~600kHz																		
5h	~800kHz																		
6h	~900kHz																		
7h	~1MHz																		
20:19	<b>Smbus0Sel.</b> Read-write. Reset: 0h. Specifies the SMBus port selection. There is only one SMBus engine controlling four SMBus ports. This register routes the SMBus engine to the desired port. <b>ValidValues:</b>																		
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>Port 0, for SMBus on the board.</td></tr> <tr> <td>1h</td><td>Port 2, dedicated for TSI polling.</td></tr> <tr> <td>2h</td><td>Port 3, Reserved.</td></tr> <tr> <td>3h</td><td>Port 4, Reserved.</td></tr> </table>	Value	Description	0h	Port 0, for SMBus on the board.	1h	Port 2, dedicated for TSI polling.	2h	Port 3, Reserved.	3h	Port 4, Reserved.								
Value	Description																		
0h	Port 0, for SMBus on the board.																		
1h	Port 2, dedicated for TSI polling.																		
2h	Port 3, Reserved.																		
3h	Port 4, Reserved.																		
18	<b>AsfClkSwitch.</b> Read-write. Reset: 0. 1=Change ASF master clock from RTC (32 kHz) to the clock defined in AsfClkSel.																		
17	<b>AsfClkStretchEn.</b> Read-write. Reset: 1. 1=Enable clock stretching support.																		
16	<b>AsfSmMasterEn.</b> Read-write. Reset: 0. 1=Enable ASF SMBus master function.																		
15:8	<b>SmbusAsfIoBase.</b> Read-write. Reset: 0Bh. <b>Description:</b> Specifies SMBus and ASF IO base address. <ul style="list-style-type: none"> <li>SMBus IO base={Smbus0AsfIoBase[7:0],0x00}.</li> <li>ASF IO base={Smbus0AsfIoBase[7:0],0x20}.</li> </ul> By default SMBus IO base is 0xB00 and ASF IO base is 0xB20.																		
7	<b>WatchdogTmrEn.</b> Read-write. Reset: 0. 1=Enable IOAPIC memory (FEB0_0000h ~ FEB0_0003h) decoding, and enable WatchDog Timer operation.																		
6	<b>HpetEn.</b> Read-write. Reset: 0. 1=Enable HPET memory (FED0_0000h ~ FED0_01FFh) decoding.																		
5	<b>IoApicEn.</b> Read-write. Reset: 0. 1=Enable IOAPIC memory (FEC0_0000h ~ FEC0_007Fh) decoding.																		
4	<b>SmbusAsfIoEn.</b> Read-write. Reset: 1. 1=Enable SMBus and ASF IO decoding. SMBUS and ASF IO ranges are defined in Smbus0AsfIoBase.																		
3	<b>DmaPort80.</b> Read-write. Reset: 0. 1=Pass IO ports 0x80, 0x81, 0x82, 0x83 to legacy DMA IO range.																		
2	<b>LegacyDmaIoEn.</b> Read-write. Reset: 0. 1=Enable legacy DMA IO range.																		
1	<b>Cf9IoEn.</b> Read-write. Reset: 0. 1=Enable CF9h IO port decoding.																		
0	<b>LegacyIoEn.</b> Read-write. Reset: 0. 1=Enable the following IO decoding as described. <b>Description:</b> 0x20, 0x21, 0xA0, 0xA1 (PIC); 0x40, 0x41, 0x42, 0x43, 0x61 (8254 timer); 0x70, 0x71, 0x72, 0x73 (RTC); 0x92.																		

**PMx004 [IsaControl] (FCH::PM::IsaControl)**

Read-write. Reset: 0000\_0002h.

\_aliasHOST; PMx004; PM=FED8\_0300h

Bits	Description
31:17	Reserved.
16	<b>ABClkGateEn.</b> Read-write. Reset: 0. 0=Disabled. 1=Enabled. Master switch for A-Link and B-Link clock gating.
15:9	Reserved.
8	<b>DmaEnhanceEn.</b> Read-write. Reset: 0. 1=Enable enhancement of legacy DMA.
7:6	Reserved.
5	<b>ReadShadow.</b> Read-write. Reset: 0. 1=Allow to Read PIC ICWX, OCWX registers through IO port 21h and A1h.
4:2	Reserved.
1	<b>MmioEn.</b> Read-write. Reset: 1. Init: BIOS, 1. 1=Enable ACPI MMIO range (FED8_0000h-FED8_1FFFh). MmioEn.
0	<b>BiosRamEn.</b> Read-write. Reset: 0. 1=Enable BIOS RAM whose base is FED1_0000h (256 bytes).

**PMx008 [PciControl] (FCH::PM::PciCtl)**

Read-write. Reset: 0000\_0100h.

\_aliasHOST; PMx008; PM=FED8\_0300h

Bits	Description
31:26	Reserved.
25	<b>ForceSlpStateRetry.</b> Read-write. Reset: 0. 1=Send out an SMI message before the completion response of IO writes to FCH::PM::PmCtl[SlpTyp]. This is to be used in conjunction with SMI trapping on writes to FCH::PM::PmCtl[SlpTyp].
24	Reserved.
23	<b>AbStallEn.</b> Read-write. Reset: 0. 1=Allows the legacy DMA engine to hold the internal bus before completing legacy DMA transaction on the LPC bus. This is only needed for certain old LPC devices.
22:21	Reserved.
20	<b>ShutDownOption.</b> Read-write. Reset: 0. 0=Issue Init message upstream when receiving shutdown message. 1=Generate PCI reset when receiving shutdown message.
19	<b>MasterNoWait.</b> Read-write. Reset: 0. 0=PCI Master waits for Slave idle. 1=ACPI PCI Master doesn't wait for Slave idle when it wants to request bus.
18:15	Reserved.
14:12	<b>ExtIntrTime.</b> Read-write. Reset: 0h. Specifies the extended interrupt time in 2 microsecond intervals. This is used for preventing APU from re-entering C-state right away when it just breaks out from a C-state.
11:10	Reserved.
9	<b>BlockAcpiS5IntrSt.</b> Read-write. Reset: 0. Block ACPI S5 interrupt.
8	<b>PicApicArbiter.</b> Read-write. Reset: 1. 1=Enable arbitration between PIC request and IOAPIC request.
7	<b>ForceSmafMatch.</b> Read-write. Reset: 0. 1=Enable STPGNT message matching to the expected SMAF.
6	<b>NmiMsgSel.</b> Read-write. Reset: 0. 0=Encode NMI request as legacy PIC NMI message type. 1=NMI request as NMI message type.
5	<b>PicMsgSel.</b> Read-write. Reset: 0. 0=Encode PIC interrupt request as Legacy PIC external interrupt (ExtInt) message type. 1=Encode PIC interrupt request as external interrupt (ExtInt) message type.
4	<b>MsgIntrEnable.</b> Read-write. Reset: 0. 1=Deliver legacy PIC interrupt as message type.
3	<b>MaskMsgBmStsEn.</b> Read-write. Reset: 0. Set to 1 to enable A20#, IGNNE#, INIT#, NMI, SMI# message delivery.
2:0	Reserved.

**PMx010 [Power Reset Config] (FCH::PM::PwrResetCfg)**

Read-write. Reset: 0000h.

_aliasHOST; PMx010; PM=FED8_0300h	
Bits	Description
15:2	Reserved.
1	<b>ToggleAllPwrGoodOnCf9</b> . Read-write. Reset: 0. Init: BIOS,1. 0=During CF9 reset, PG1_PwrGood stays high; PG1a_PwrGood, PG2_PwrGood and Xhc_PwrGood behavior depend on TogglePG1aPG2PGXhcOnCf9 bit during CF9 reset. 1=De-assert and then assert all PwrGood signals (for PG1, PG1a, PG2 and XHC) during CF9 reset.
0	Reserved.

**PMx014 [PGPwrGoodTmr] (FCH::PM::PGPwrGoodTmr)**

Reset: 0000\_0000h.

\_aliasHOST; PMx014; PM=FED8\_0300h

Bits	Description
31:0	Reserved.

**PMx01E [PLLLockTmr] (FCH::PM::PllLockTmr)**

Reset: 0000h.

\_aliasHOST; PMx01E; PM=FED8\_0300h

Bits	Description
15:0	Reserved.

**PMx02A [S5ToS0EnTmr] (FCH::PM::S5ToS0EnTmr)**

Reset: 0000h.

\_aliasHOST; PMx02A; PM=FED8\_0300h

Bits	Description
15:0	Reserved.

**PMx02C [SlpRstTmr] (FCH::PM::SlpRstTmr)**

Reset: 0000h.

\_aliasHOST; PMx02C; PM=FED8\_0300h

Bits	Description
15:0	Reserved.

**PMx02E [OverrideDevRstTmr] (FCH::PM::OverrideDevRstTmr)**

Reset: 0000h.

\_aliasHOST; PMx02E; PM=FED8\_0300h

Bits	Description
15:0	Reserved.

**PMx030 [OverridePwrRstBASrtTmr] (FCH::PM::OverridePwrRstBASrtTmr)**

Reset: 0000h.

\_aliasHOST; PMx030; PM=FED8\_0300h

Bits	Description
15:0	Reserved.

**PMx032 [OverrideRefClkOkDsrtTmr] (FCH::PM::OverrideRefClkOkDsrtTmr)**

Reset: 0000h.

\_aliasHOST; PMx032; PM=FED8\_0300h

Bits	Description
15:0	Reserved.

**PMx034 [OverrideEnable] (FCH::PM::OverrideEnable)**

Reset: 0000\_0000h.

\_aliasHOST; PMx034; PM=FED8\_0300h

Bits	Description
31:0	Reserved.

**PMx040 [eSPIIntrCtrl] (FCH::PM::ESPIIntrCtrl)**

Read-write. Reset: 00FF\_FFFFh.

\_aliasHOST; PMx040; PM=FED8\_0300h

Bits	Description
31:24	Reserved.
23:0	<b>eSPIDevIntrMask.</b> Read-write. Reset: FF_FFFFh. Set these bits to mask of eSPI Device IRQ[23:0].
<b>Valid Values:</b>	
Value	Description
0	Enable eSPI IRQ[23:0]
1	Mask off the interrupt

**PMx044 [BootTimerEn] (FCH::PM::BootTimerEn)**

Reset: Cold,1800\_0000h.

\_aliasHOST; PMx044; PM=FED8\_0300h

Bits	Description
31	<b>BootTmrDisable.</b> Read-write. Reset: Cold,0. Init: BIOS,1. 1=Boot timer is stopped. Once set it to 1, timer will reset back to 0. When this bit is set back to 0, timer starts counting from 0. This bit will clear itself on any reset (this bit is non-sticky).
30	<b>FailBootRstSts.</b> Read,Write-1-to-clear. Reset: Cold,0. 0=Boot timer has not been fired. 1=Boot timer has been fired. Write-1-to-clear it to 0.
29	<b>ExpireBootTmr.</b> Read-write. Reset: Cold,0. 1=Force boot timer to expire; then, NB PwrGood can be asserted.
28	<b>BootTmrStopOnGAlink.</b> Read-write. Reset: Cold,1. Init: BIOS,0. 0=Enable the boot timer to ensure a good boot. 1=Boot timer stops when FCH observes a good boot after PCI reset or S3/S4/S5 resume. BootTmrStopOnGAlink.
27	<b>BootTmrFuncEn.</b> Read-write. Reset: Cold,1. Init: BIOS,1. 0=Disable boot timer function. 1=Enable boot timer function. The boot timer will start to count down and toggle NBPwrGood after 1.26s ( $2^{24}/(66.6M/5)$ ) if software has not set ExpireBootTmr = 1 after PCI reset or resuming from S3/S4/S5, This bit is persistent through any reset or sleep (this bit is sticky).
26:25	Reserved.
24:0	<b>FailBootTimer.</b> Read-only. Reset: Cold,000_0000h. <b>Description:</b> Specifies the counter of APU boot timer (66.6 MHz/5), which starts counting when the following conditions are met: <ul style="list-style-type: none"> <li>• BootTmrDisable == 0.</li> <li>• BootTmrFuncEn == 1.</li> <li>• PCI reset is not asserted.</li> </ul> Setting BootTmrDisable = 1 will stop the timer and reset timer back to 0. It will count from 0 again if BootTmrDisable is set back to 0, BootTmrDisable is a non-sticky bit and will get cleared to 0 on any reset.  Setting BootTmrFuncEn = 0 will disable the timer permanently because BootTmrFuncEn is a sticky bit. BootTmrFuncEn will reset back to 1 if there is an S5 power loss; or it can be written back to 1 by software.  The timer itself cannot be written directly. It will reset back to 0 whenever BootTmrDisable = 1; or BootTmrFuncEn = 0; or a reset has occurred.

**PMx048 [PGPwrEnDly] (FCH::PM::WatchdogTimerEn)**

Read-write. Reset: 5935\_2B21h.

\_aliasHOST; PMx048; PM=FED8\_0300h

Bits	Description
31:24	<b>PG1PwrDownDlyTmr.</b> Read-write. Reset: 59h. PG1 Power Down delay timer.

23:16	<b>XhcPwrEnDlyTmr.</b> Read-write. Reset: 35h. XHC Power Enable delay timer.
15:8	<b>PG2PwrEnDlyTmr.</b> Read-write. Reset: 2Bh. PG2 Power Enable delay timer.
7:0	<b>PG1aPwrEnDlyTmr.</b> Read-write. Reset: 21h. PG1a Power En delay timer.

**PMx04C [I2CInputThreshold] (FCH::PM::I2CInputThreshold)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; PMx04C; PM=FED8\_0300h

Bits	Description
31:6	Reserved.
5	<b>I2C5InputThresholdHi.</b> Read-write. Reset: 0. 0=Low threshold. 1=High threshold. Configure I2C5 input threshold.
4	<b>I2C4InputThresholdHi.</b> Read-write. Reset: 0. 0=Low threshold. 1=High threshold. Configure I2C4 input threshold.
3	<b>I2C3InputThresholdHi.</b> Read-write. Reset: 0. 0=Low threshold. 1=High threshold. Configure I2C3 input threshold.
2	<b>I2C2InputThresholdHi.</b> Read-write. Reset: 0. 0=Low threshold. 1=High threshold. Configure I2C2 input threshold.
1	<b>I2C1InputThresholdHi.</b> Read-write. Reset: 0. 0=Low threshold. 1=High threshold. Configure I2C1 input threshold.
0	<b>I2C0InputThresholdHi.</b> Read-write. Reset: 0. 0=Low threshold. 1=High threshold. Configure I2C0 input threshold.

**PMx050 [ApuPllCtrl] (FCH::PM::ApuPllCtrl)**

Read-write. Reset: 3F3E\_0000h.

\_aliasHOST; PMx050; PM=FED8\_0300h

Bits	Description
31:24	<b>ApuPllRstBTmr.</b> Read-write. Reset: 3Fh. Configure the PllLock assertion time. Unit is 16 usec.
23:16	<b>ApuPllPwrRstBTmr.</b> Read-write. Reset: 3Eh. Configure the PllRstB de-assertion time. Unit is 16 usec.
15:0	Reserved.

**PMx054 [SerialIrqConfig] (FCH::PM::SerialIrqCfg)**

Read-write.

\_aliasHOST; PMx054; PM=FED8\_0300h

Bits	Description								
15:8	Reserved.								
7	<b>SerialIrqEnable.</b> Read-write. Reset: 0. 1=Enable the serial IRQ function.								
6	<b>SerIrqMode.</b> Read-write. Reset: 0. 0=Continuous mode. 1=Active (quiet) mode.								
5:2	<b>NumSerIrqBits.</b> Read-write. Reset: Cold,0h. <b>Description:</b> Specifies the total number of serial IRQs. The serial IRQ can support 15 IRQ#, SMI#, IOCHK#, INTA#, INTB#, INTC#, and INTD#. <b>ValidValues:</b>								
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>17 serial IRQs (15 IRQ#, SMI#, IOCHK#)</td></tr> <tr> <td>1h</td><td>18 serial IRQs (15 IRQ#, SMI#, IOCHK#, INTA#)</td></tr> <tr> <td>Fh-2h</td><td>&lt;NumSerIrqBits+17&gt;</td></tr> </table>	Value	Description	0h	17 serial IRQs (15 IRQ#, SMI#, IOCHK#)	1h	18 serial IRQs (15 IRQ#, SMI#, IOCHK#, INTA#)	Fh-2h	<NumSerIrqBits+17>
Value	Description								
0h	17 serial IRQs (15 IRQ#, SMI#, IOCHK#)								
1h	18 serial IRQs (15 IRQ#, SMI#, IOCHK#, INTA#)								
Fh-2h	<NumSerIrqBits+17>								
1:0	<b>NumStartBits.</b> Read-write. Reset: Cold,0h. This field specifies the number of clocks in the start frame. Start Frame Width = 4 + 2 * NumStartBits.								

**PMx056 [RTC Control] (FCH::PM::RTCctl)**

Reset: Cold,1400h.

\_aliasHOST; PMx056; PM=FED8\_0300h

Bits	Description
15	Reserved.
14	<b>ExtraRtcCmosEn.</b> Read-write. Reset: Cold,0. 0=Access to the extra RTC CMOS RAM space is disabled. 1=Software can access the extra 16 bytes of RTC CMOS RAM through FCH::IO::AltRTCAddrPort and FCH::IO::AltRTCDDataPort; The extra RAM spaces are located at index 0x00-0x03, 0x40-0x43, 0x80-0x83, 0xC0-0xC3.
13	<b>AltCmosMapEn.</b> Read-write. Reset: Cold,0. 1=When accessing the RTC CMOS RAM through FCH::IO::RtcAddrPortNmiMask and FCH::IO::RtcDataPort, Bank 1 of CMOS RAM is changed: Index[0Dh:00h] still returns the time and alarm settings; Index[7Fh:0Eh] returns the absolute offset[FFh:8Eh] which maps to the extended RAM space.
12	<b>CenturyEn.</b> Read-write. Reset: Cold,1. 1=Enable RTC Century support.
11	<b>MaskRtcClkOut.</b> Read-write. Reset: Cold,0. 1=Disable RtcClk output.
10	<b>RtcClkDrive.</b> Read-write. Reset: Cold,1. 0=HIGHDRIVE is tied low for RtcClkOut pad. 1=HIGHDRIVE is tied high for RtcClkOut pad.
9:5	Reserved.
4	<b>RtcProtectC0_CF.</b> Write-once. Reset: Cold,0. 1=RTC RAM index[CFh:C0h] are locked from Read/Write.
3	<b>RtcProtectD0_DF.</b> Write-once. Reset: Cold,0. 1=RTC RAM index[DFh:D0h] are locked from Read/Write.
2	<b>RtcProtectE0_EF.</b> Write-once. Reset: Cold,0. 1=RTC RAM index[EFh:E0h] are locked from Read/Write.
1	<b>RtcProtectF0_FF.</b> Write-once. Reset: Cold,0. 1=RTC RAM index[FFh:F0h] are locked from Read/Write.
0	<b>RtcProtect38_3F.</b> Write-once. Reset: Cold,0. 1=RTC RAM index[3Fh:38h] are locked from Read/Write.

**PMx058 [VRT\_T1] (FCH::PM::VRT\_T1)**

Read-write. Reset: Cold,01h.

\_aliasHOST; PMx058; PM=FED8\_0300h

Bits	Description
7:0	<b>VRT_T1.</b> Read-write. Reset: Cold,01h. This field specifies the time of VRT_Enable being high for RTC battery monitor circuit in milliseconds. To conserve power, the RTC battery is sampled periodically for checking its state of health. VRT_T1 and FCH::PM::VRT_T2[VRT_T2] make up the interval of the checking. When VRT_Enable is high, the battery is being sampled. When VRT_Enable is low, the battery is not being sampled.

**PMx059 [VRT\_T2] (FCH::PM::VRT\_T2)**

Read-write. Reset: Cold,FFh.

\_aliasHOST; PMx059; PM=FED8\_0300h

Bits	Description
7:0	<b>VRT_T2.</b> Read-write. Reset: Cold,FFh. This field specifies the time of VRT_Enable being low for the RTC battery monitor circuit in 4 ms increments. See FCH::PM::VRT_T1 for detailed description.

**PMx05B [RTC Shadow] (FCH::PM::RTCShadow)**

Read-write. Reset: Cold,00h.

NOTE: These four bits don't have any default value. After power on, their values are undetermined. Software has to program PwrFailShadow to give them values.

\_aliasHOST; PMx05B; PM=FED8\_0300h

Bits	Description
7:4	Reserved.
3:0	<b>PwrFailShadow.</b> Read-write. Reset: Cold,0h. Init: BIOS,4h. Power Fail Shadow.

**PMx05C [LLBCntrl] (FCH::PM::LLBCntrl)**

Read-write. Reset: 00h.

\_aliasHOST; PMx05C; PM=FED8\_0300h

Bits	Description
7:3	Reserved.
2	<b>AllowWakeS3En.</b> Read-write. Reset: 0. 1=Allow LLB# as wake event in S3.

	<b>Description:</b> NOTE: LLB (Low low battery) event should trigger an SCI if system is in S0. It should block wake-up if the system is in S-States. To meet that required behavior, software should use the following settings: <ul style="list-style-type: none"> <li>• LLB_En (FCH::PM::Misc[LibEn]) = 1.</li> <li>• BlockWakeEn = 0.</li> <li>• UseAsWakeEn = 1.</li> <li>• AllowWakeS3En = 0 or 1 (don't care).</li> </ul>
1	<b>UseAsWakeEn.</b> Read-write. Reset: 0. 1=Treat LLB# as wake event.
0	<b>BlockWakeEn.</b> Read-write. Reset: 0. 1=Block wake event if LLB# is asserted; if (UseAsWakeEn == 1) && (AllowWakeS3En == 1), LLB# and other wake events can wake the system up from S3.

**PMx05E [RTC ExtIndex] (FCH::PM::RTCExtIndex)**

Read-write.

\_aliasHOST; PMx05E; PM=FED8\_0300h

Bits	Description
7:0	<b>Index.</b> Read-write. Reset: Cold,XXh. Specifies the offset of the RTC Extended Register to be Read or Written from FCH::PM::RTCExtData.

**PMx05F [RTC ExtData] (FCH::PM::RTCExtData)**

Read-write.

\_aliasHOST; PMx05F; PM=FED8\_0300h

Bits	Description
7:0	<b>Data.</b> Read-write. Reset: Cold,XXh. Specifies the Read data or Write data of the RTC Extended Register.

**PMx060 [AcpiPm1EvtBlk] (FCH::PM::AcpiPm1EvtBlk)**

Read-write. Reset: Cold,0000h.

\_aliasHOST; PMx060; PM=FED8\_0300h

Bits	Description
15:2	<b>AcpiPm1EvtBlk.</b> Read-write. Reset: Cold,0000h. Specifies the 16-bit IO range base address[15:2] of the ACPI power management event block.
1:0	Reserved.

**PMx062 [AcpiPm1CntBlk] (FCH::PM::AcpiPm1CntBlk)**

Read-write. Reset: Cold,0400h.

\_aliasHOST; PMx062; PM=FED8\_0300h

Bits	Description
15:1	<b>AcpiPm1CntBlk.</b> Read-write. Reset: Cold,0200h. Specifies the 16-bit IO base address[15:1] of the ACPI power management control block.
0	Reserved.

**PMx064 [AcpiPmTmrBlk] (FCH::PM::AcpiPmTmrBlk)**

Read-write. Reset: Cold,0000h.

\_aliasHOST; PMx064; PM=FED8\_0300h

Bits	Description
15:1	<b>AcpiPmTmrBlk.</b> Read-write. Reset: Cold,0000h. Specifies the 16-bit IO base address[15:1] of the ACPI power management timer block.
0	Reserved.

**PMx066 [PCntBlk] (FCH::PM::PCntBlk)**

Read-write. Reset: Cold,0000h.

\_aliasHOST; PMx066; PM=FED8\_0300h

Bits	Description
15:3	<b>CpuControl.</b> Read-write. Reset: Cold,0000h. These bits define the least significant byte of the 16 bit I/O base address of the ACPI power management APU Control block. Bit 3 corresponds to Addr[3] and bit 7 corresponds



	to Addr[7]. Addr[2:0] are ignored because this register is 6 byte long.
2:0	Reserved.

**PMx068 [AcpiGpe0Blk] (FCH::PM::AcpiGpe0Blk)**

Read-write. Reset: Cold,0000h.

\_aliasHOST; PMx068; PM=FED8\_0300h

Bits	Description
15:2	<b>AcpiGpe0Blk.</b> Read-write. Reset: Cold,0000h. Specifies the 16-bit IO base address[15:2] of the ACPI power management general purpose event block.
1:0	Reserved.

**PMx06A [AcpiSmiCmd] (FCH::PM::AcpiSmiCmd)**

Read-write. Reset: Cold,00B0h.

\_aliasHOST; PMx06A; PM=FED8\_0300h

Bits	Description
15:0	<b>AcpiSmiCmd.</b> Read-write. Reset: Cold,00B0h. Defines the 16-bit IO base address[15:0] of the ACPI SMI command block.

**PMx06E [AcpiPm2CntBlk] (FCH::PM::AcpiPm2CntBlk)**

Read-write. Reset: Cold,0000h.

\_aliasHOST; PMx06E; PM=FED8\_0300h

Bits	Description
15:0	<b>AcpiPm2CntBlk.</b> Read-write. Reset: Cold,0000h. Defines the 16-bit IO base address[15:0] of the ACPI power management additional control block.

**PMx074 [AcpiConfig] (FCH::PM::AcpiCfg)**

\_aliasHOST; PMx074; PM=FED8\_0300h

Bits	Description
31:30	Reserved.
29	<b>RtcWakeAlarm.</b> Read-write. Reset: Cold,0. 0=Use FCH::IO::RTCC[IRQF] as one of the system wake-up events if IRQF is enabled. 1=Use FCH::IO::RTCC[AF] as one of the system wake-up events if AF is enabled. RTC wake alarm.
28	<b>PcieGeventMap.</b> Read-write. Reset: Cold,1. 1=Route PME message from NB to GEVENT 24, hot plug message from APU to GEVENT 7.
27	<b>WakePinAsGevent.</b> Read-write. Reset: Cold,0. 1=Treat WAKE# pin as GEVENT input.
26	Reserved.
25	<b>PcieWakMask.</b> Read-write. Reset: Cold,0. 1=Disable the PCIE_WAK_STS and PCIE_WAK_DIS function defined in FCH::PM::Pm1Stat[PciExpWakeStatus] and FCH::PM::Pm1En[PciExpWakeDis].
24	<b>PcieNative.</b> Read-write. Reset: Cold,0. 1=Block PCIe® GPP PME message and hot plug message from generating SCI.
23	<b>RstUSBS5.</b> Read-write. Reset: Cold,0. 0=De-assert Cpl_VDDCR_S5_RESEtN = 1. 1=Make Cpl_VDDCR_S5_RESEtN = 0 to reset USB, MP2, etc.
22:10	Reserved.
9	<b>AcpiReducedHwEn.</b> Read-write. Reset: 0. 0=ACPI fixed register interface is enabled. 1=The decoding of ACPI fixed registers and SCI are disabled; In addition, wake function from AcpiPmEvtBlk is disabled as well.
8	<b>PwnBtnEn.</b> Read-write. Reset: 1. 1=Enable power button support in AcpiPmEvtBlk block.
7	<b>BiosRIs.</b> Read,Write-1-only. Reset: Fixed,0. Writing 1 to this bit generates SMI, NMI or IRQ13 depending on FCH::SMI::SmiCtl4[Smicontrol73].
6:5	Reserved.
4	<b>TmrEnEn.</b> Read-write. Reset: Cold,0. 1=Enable the TMR_EN function defined in FCH::PM::Pm1En[TmrEn].
3	Reserved.
2	<b>RtcEnEn.</b> Read-write. Reset: Cold,0. 1=Enable the RTC_EN function defined in FCH::PM::Pm1En[RtcEn].

1	<b>GblEnEn.</b> Read-write. Reset: Cold,0. 1=Enable the GBL_EN function defined in FCH::PM::Pm1En[GblEn].
0	<b>DecEnAcpi.</b> Read-write. Reset: Cold,0. 1=Enable decoding of the standard ACPI registers.

**PMx07E [CStateEn] (FCH::PM::CStateEn)**

Read-write. Reset: 0020h.

\_aliasHOST; PMx07E; PM=FED8\_0300h

Bits	Description
15:6	Reserved.
5	<b>C1eToC3En.</b> Read-write. Reset: 1. 1=Put APU into C3 state in C1e state. Software needs to program C1eToC3En = 1.
4	<b>C1eToC2En.</b> Read-write. Reset: 0. 1=Put APU into C2 state in C1e state.
3:0	Reserved.

**PMx088 [CStateControl] (FCH::PM::CStateCtl)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; PMx088; PM=FED8\_0300h

Bits	Description
31:6	Reserved.
5	<b>SlpEn.</b> Read-write. Reset: 0. 1=Enable LDTSTOP# as an output.
4:3	Reserved.
2	<b>DlySlpEn.</b> Read-write. Reset: 0. 1=Delay recognition of STPGNT# until there is no pending Read in AB.
1:0	Reserved.

**PMx0A0 [MessageCState] (FCH::PM::MsgCState)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; PMx0A0; PM=FED8\_0300h

Bits	Description
31:5	Reserved.
4	<b>FusionSerrEn.</b> Read-write. Reset: 0. 1=FCH C-state coordination logic causes CPU to exit from C-state when there is a system error within the FCH.
3	<b>FusionPerrEn.</b> Read-write. Reset: 0. 1=FCH C-state coordination logic causes CPU to exit from C-state when there is a parity error within the FCH.
2	Reserved.
1	<b>TimerTickChgMsgEn.</b> Read-write. Reset: 0. 1=FCH sends a message to APU indicating the latest periodic timer interval. FCH automatically determines which timer (PIT, RTC, or HPET) is being used.
0	<b>BattModeChgMsgEn.</b> Read-write. Reset: 0. 1=FCH automatically sends a message to CPU indicating the power mode (AC vs battery); In addition, every time it is changed, FCH generates a message to indicate the update.

**PMx0A4 [MiscDebug] (FCH::PM::TrafStat)**

Reset: 0000\_0000h.

\_aliasHOST; PMx0A4; PM=FED8\_0300h

Bits	Description
31:0	Reserved.

**PMx0A8 [VirtualWire] (FCH::PM::TrafMonIdleTime)**

Read-write. Reset: 00FF\_CEF8h.

\_aliasHOST; PMx0A8; PM=FED8\_0300h

Bits	Description
31	<b>VW_Enable.</b> Read-write. Reset: 0. 0=The state machine is disabled. 1=The state machine is enabled. <b>Description:</b> This bit enables the VirtualWire function. It is possible that Virtual Wire state machine is sending Virtual Wire messages when software clears the VW_Enable bit. In that case, finish Virtual Wire message delivery first and then disable Virtual Wire state

	machine. Reading VW_Enable bit returns the current state of Virtual Wire state machine.
30	<b>VW_InvValue.</b> Read-write. Reset: 0. 0=Normal polarity. 1=Inverted polarity. This is to change the polarity of the virtual wire value bit in the message.
29:24	Reserved.
23:0	<b>VW_SrcActiveLow.</b> Read-write. Reset: FF_CEF8h. Each bit configures polarity of the corresponding interrupt wires. These 24 interrupt wires are connected to IOAPIC input and virtual wire input.
<b>ValidValues:</b>	
Bit	Description
[0]	0=Interrupt Wire is active high. 1=Interrupt wire is active low.
[1]	0=Interrupt Wire is active high. 1=Interrupt wire is active low.
[2]	0=Interrupt Wire is active high. 1=Interrupt wire is active low.
[3]	0=Interrupt Wire is active high. 1=Interrupt wire is active low.
[4]	0=Interrupt Wire is active high. 1=Interrupt wire is active low.
[5]	0=Interrupt Wire is active high. 1=Interrupt wire is active low.
[6]	0=Interrupt Wire is active high. 1=Interrupt wire is active low.
[7]	0=Interrupt Wire is active high. 1=Interrupt wire is active low.
[8]	0=Interrupt Wire is active high. 1=Interrupt wire is active low.
[9]	0=Interrupt Wire is active high. 1=Interrupt wire is active low.
[10]	0=Interrupt Wire is active high. 1=Interrupt wire is active low.
[11]	0=Interrupt Wire is active high. 1=Interrupt wire is active low.
[12]	0=Interrupt Wire is active high. 1=Interrupt wire is active low.
[13]	0=Interrupt Wire is active high. 1=Interrupt wire is active low.
[14]	0=Interrupt Wire is active high. 1=Interrupt wire is active low.
[15]	0=Interrupt Wire is active high. 1=Interrupt wire is active low.
[16]	0=Interrupt Wire is active high. 1=Interrupt wire is active low.
[17]	0=Interrupt Wire is active high. 1=Interrupt wire is active low.
[18]	0=Interrupt Wire is active high. 1=Interrupt wire is active low.
[19]	0=Interrupt Wire is active high. 1=Interrupt wire is active low.
[20]	0=Interrupt Wire is active high. 1=Interrupt wire is active low.
[21]	0=Interrupt Wire is active high. 1=Interrupt wire is active low.
[22]	0=Interrupt Wire is active high. 1=Interrupt wire is active low.
[23]	0=Interrupt Wire is active high. 1=Interrupt wire is active low.

**PMx0B0 [DeferTimeTick] (FCH::PM::DeferTimeTickOBFFctl)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; PMx0B0; PM=FED8\_0300h

Bits	Description
31:11	Reserved.
10:8	<b>DeferTimerTickValue.</b> Read-write. Reset: 0h.
<b>ValidValues:</b>	
Value	Description
0h	No skipping.
1h	Skip 1 timer tick.
2h	Skip 2 timer ticks.
3h	Skip 3 timer ticks.
4h	Skip 4 timer ticks.
5h	Skip 5 timer ticks.
6h	Skip 6 timer ticks.
7h	Skip 7 timer ticks.

7:2	Reserved.
1	<b>ForceTmrTickEn.</b> Read-write. Reset: 0. 1=If (DeferTimerTickEn == 1) && FCH has skipped a timer tick interrupt, FCH immediately generates the timer tick interrupt upon C-state exit.
0	<b>DeferTimerTickEn.</b> Read-write. Reset: 0. 1=FCH skips a number of timer tick interrupts based on the value defined in DeterTimeTickValue when CPU is in C-state. When CPU is not in C-state, the FCH does not skip any timer tick interrupt.

**PMx0B8 [Tpreset2] (FCH::PM::Tpreset2)**

Read-write. Reset: 80h.

\_aliasHOST; PMx0B8; PM=FED8\_0300h

Bits	Description										
7:6	<b>ClkGateCntrl.</b> Read-write. Reset: 2h. These two bits control whether SMBUS module allows clock gating to the internal 66 MHz core clock. <b>ValidValues:</b> <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>Disable the clock gating function.</td></tr> <tr> <td>1h</td><td>Wait 16 clocks before allowing clock gating to the SMBUS module.</td></tr> <tr> <td>2h</td><td>Wait 64 clocks before allowing clock gating to the SMBUS module.</td></tr> <tr> <td>3h</td><td>Wait 256 clocks before allowing clock gating to the SMBUS module.</td></tr> </table>	Value	Description	0h	Disable the clock gating function.	1h	Wait 16 clocks before allowing clock gating to the SMBUS module.	2h	Wait 64 clocks before allowing clock gating to the SMBUS module.	3h	Wait 256 clocks before allowing clock gating to the SMBUS module.
Value	Description										
0h	Disable the clock gating function.										
1h	Wait 16 clocks before allowing clock gating to the SMBUS module.										
2h	Wait 64 clocks before allowing clock gating to the SMBUS module.										
3h	Wait 256 clocks before allowing clock gating to the SMBUS module.										
5:0	Reserved.										

**PMx0B9 [LpcMisc] (FCH::PM::LpcMisc)**

Read-write. Reset: 00h.

\_aliasHOST; PMx0B9; PM=FED8\_0300h

Bits	Description
7:4	Reserved.
3	<b>ClkRunDisable.</b> Read-write. Reset: 0. 1=Disable this module's ability to support CLKRUN# function from PCIBridge; In other words, this module prevents PCIBridge from stopping the 33 MHz clock. Legacy DMA and serial IRQ logic reside in this module and they are running on the 33 MHz LPCCLK.
2:0	Reserved.

**PMx0BA [S\_StateControl] (FCH::PM::SStateCtl)**

Read-write. Reset: 0000h.

\_aliasHOST; PMx0BA; PM=FED8\_0300h

Bits	Description
15	<b>MaskPmeMsgEn.</b> Read-write. Reset: 0. 1=If (PmeMsgEn == 1), PmeAck messages are ignored and ACPI S-state logic solely uses the timeout mechanism to sequence through the S3 state.
14	<b>WakePinEnable.</b> Read-write. Reset: 0. 1=Enable wake up from WAKE# pin. WAKE# pin enable.
13:1	Reserved.
0	<b>LongSLPS3.</b> Read-write. Reset: 0. 1=Extend SLP_S3# assertion to 1 s minimum.

**PMx0BC [ThrottlingControl] (FCH::PM::ThrottleCtl)**

Read-write. Reset: 0000h.

\_aliasHOST; PMx0BC; PM=FED8\_0300h

Bits	Description
15	<b>Therm2SecDelay.</b> Read-write. Reset: 0. 1=Enable 2 second delay for thermal clock throttle. This bit affects both hardware and software throttle.
14	<b>NoWaitStpGntEn.</b> Read-write. Reset: 0. 0=Wait for STPGNT after asserting STPCLK. 1=Do not wait for STPGNT after asserting STPCLK. This bit affects both hardware and software throttle.
13	<b>ThermThrotPeriod.</b> Read-write. Reset: 0. 0=30 us. 1=244 us. Specifies the clock throttle period for hardware thermal throttle.
12:8	Reserved.

7	<b>ThrottleControl[3]</b> . Read-write. Reset: 0. 1=Enable hardware thermal clock throttle. See ThrottleControl[2:0].																		
6:4	<b>ThrottleControl[2:0]</b> . Read-write. Reset: 0h. Specifies the throttle interval for STPCLK de-assertion in hardware thermal clock throttle.																		
<b>ValidValues:</b>																			
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>50%</td></tr> <tr> <td>1h</td><td>12.5%</td></tr> <tr> <td>2h</td><td>25%</td></tr> <tr> <td>3h</td><td>37.5%</td></tr> <tr> <td>4h</td><td>50%</td></tr> <tr> <td>5h</td><td>62.5%</td></tr> <tr> <td>6h</td><td>75%</td></tr> <tr> <td>7h</td><td>87.5%</td></tr> </table>	Value	Description	0h	50%	1h	12.5%	2h	25%	3h	37.5%	4h	50%	5h	62.5%	6h	75%	7h	87.5%
Value	Description																		
0h	50%																		
1h	12.5%																		
2h	25%																		
3h	37.5%																		
4h	50%																		
5h	62.5%																		
6h	75%																		
7h	87.5%																		
3:0	Reserved.																		

**PMx0BE [ResetControl1] (FCH::PM::ResetCtl1)**

Read-write. Reset: 72h.

\_aliasHOST; PMx0BE; PM=FED8\_0300h

Bits	Description										
7	<b>RstToCpuPwrGdEn</b> . Read-write. Reset: 0. 1=FCH toggles CpuPwrGd on every reset.										
6	<b>HwmResetOption</b> . Read-write. Reset: 1. 0=Hwm function is reset by RsmRst. 1=Hwm function is reset by PciRst.										
5	<b>SlpTypeControl</b> . Read-write. Reset: 1. 0=FCH::PM::PmCtl[SlpTypeEn] bit has no effect. 1=Enable the function of FCH::PM::PmCtl[SlpTypeEn].										
4	<b>KbRstEn</b> . Read-write. Reset: 1. 1=Enable KBRST# pin to trigger keyboard reset.										
3:2	<b>CpuRstControl</b> . Read-write. Reset: 0h.										
<b>ValidValues:</b>											
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>CpuReset is de-asserted after PciReset.</td></tr> <tr> <td>1h</td><td>CpuReset is de-asserted as PciReset.</td></tr> <tr> <td>2h</td><td>CpuReset is de-asserted before PciReset.</td></tr> <tr> <td>3h</td><td>CpuReset is de-asserted after PciReset.</td></tr> </table>	Value	Description	0h	CpuReset is de-asserted after PciReset.	1h	CpuReset is de-asserted as PciReset.	2h	CpuReset is de-asserted before PciReset.	3h	CpuReset is de-asserted after PciReset.
Value	Description										
0h	CpuReset is de-asserted after PciReset.										
1h	CpuReset is de-asserted as PciReset.										
2h	CpuReset is de-asserted before PciReset.										
3h	CpuReset is de-asserted after PciReset.										
1	<b>KbPciRstEn</b> . Read-write. Reset: 1. 1=Make PCI reset during keyboard reset, which can be triggered by KBRST# pin or IMC. This bit must not be programmed by the software. It should be left with the power up default value of 1.										
0	<b>SoftResetEn</b> . Read-write. Reset: 0. 1=Block any reset request until the system is not in C state.										

**PMx0C0 [S5/Reset Status] (FCH::PM::S5ResetStat)**

Reset: 0000\_0800h.

This register shows the source of previous reset.

\_aliasHOST; PMx0C0; PM=FED8\_0300h

Bits	Description
31:28	Reserved.
27	<b>SyncFlood</b> . Read,Write-1-to-clear. Reset: 0. System reset was caused by a SYNC_FLOOD event which was due to an UE error or caused by a SHUTDOWN command from CPU if FCH::PM::PciCtl[ShutDownOption]. Bits[27:16] will be cleared by the last reset event, except the associated bit will be set.
26	<b>RemoteResetFromASF</b> . Read,Write-1-to-clear. Reset: 0. System reset was caused by a remote RESET command from ASF. Bits[27:16] will be cleared by the last reset event, except the associated bit will be set.
25	<b>WatchdogIssueReset</b> . Read,Write-1-to-clear. Reset: 0. Bits[27:16] will be cleared by the last reset event except the associated bit will be set.

24	<b>FailBootRst.</b> Read,Write-1-to-clear. Reset: 0. Bits[27:16] will be cleared by the last reset event, except the associated bit will be set.										
23	Reserved.										
22	<b>KbReset.</b> Read,Write-1-to-clear. Reset: 0. System reset was caused by assertion of KB_RST_L. Bits[27:16] will be cleared by the last reset event, except the associated bit will be set.										
21	<b>SleepReset.</b> Read,Write-1-to-clear. Reset: 0. Reset status from Sleep state (s0i3, S3, S4 or S5) transition. Bits[27:16] will be cleared by the last reset event, except the associated bit will be set.										
20	Reserved.										
19	<b>Do_k8_Reset.</b> Read,Write-1-to-clear. Reset: 0. System reset was caused by CF9 = 0x06. Bits[27:16] will be cleared by the last reset event, except the associated bit will be set.										
18	<b>Do_k8_Init.</b> Read,Write-1-to-clear. Reset: 0. System reset was caused by CF9 = 0x04. Bits[27:16] will be cleared by the last reset event, except the associated bit will be set.										
17	<b>SoftPciRst.</b> Read,Write-1-to-clear. Reset: 0. System reset was caused by writing to FCH::PM::ResetCmd[Reset]. Bits[27:16] will be cleared by the last reset event, except the associated bit will be set.										
16	<b>UserRst.</b> Read,Write-1-to-clear. Reset: 0. Last reset was caused by BP_SYS_RST_L assertion. Bits[27:16] will be cleared by the last reset event, except the associated bit will be set.										
15:14	<b>PmeTurnOffTime.</b> Read-write. Reset: 0h.										
<b>ValidValues:</b>											
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>1ms</td></tr> <tr> <td>1h</td><td>2ms</td></tr> <tr> <td>2h</td><td>4ms</td></tr> <tr> <td>3h</td><td>8ms</td></tr> </table>	Value	Description	0h	1ms	1h	2ms	2h	4ms	3h	8ms
Value	Description										
0h	1ms										
1h	2ms										
2h	4ms										
3h	8ms										
13:10	Reserved.										
9	<b>IntThermalTrip.</b> Read,Write-1-to-clear. Reset: 0. System was shutdown due to an internal ThermalTrip event.										
8:5	Reserved.										
4	<b>RemotePowerDownFromASF.</b> Read,Write-1-to-clear. Reset: 0. SOC has received a remote Power Off command from ASF.										
3	Reserved.										
2	<b>Shutdown.</b> Read,Write-1-to-clear. Reset: 0. System was shutdown due to ShutDown event (SHUTDOWN# pin).										
1	<b>FourSecondPwrBtn.</b> Read,Write-1-to-clear. Reset: 0. System was shutdown due to 4s PwrButton event.										
0	<b>ThermalTrip.</b> Read,Write-1-to-clear. Reset: 0. System was shutdown due to BP_THERMTRIP_L assertion.										

**PMx0C4 [ResetCommand] (FCH::PM::ResetCmd)**

Reset: 28h.

\_aliasHOST; PMx0C4; PM=FED8\_0300h

Bits	Description
7	<b>ResetEn.</b> Read-write. Reset: 0. 0=Writing to bit [Reset] is not allowed. 1=Writing to bit [Reset] is allowed.
6	<b>ResetAllAcpi.</b> Write-1-only. Reset: 0. Writing 1 to this bit emulates a reset button event.
5	<b>ResetButtonEn.</b> Read-write. Reset: 1. 0=Disable Reset button. 1=Enable Reset button. 1=Reset from reset button (SYS_RESET_L) will generate S5Reset. 0=Disable SYS_RESET_L to generate S5Reset, which can be used as GEVENT19/GPIO1.
4	<b>ResetPcie.</b> Read-write. Reset: 0. 0=Release the PCIe® reset. 1=Reset the GPP ports. Set to 1 to reset GPP ports.
3	<b>UsrRst2Pll.</b> Read-write. Reset: 1. 1=Stop the PLL when the reset button is pressed.
2	<b>SelectDebug.</b> Read-write. Reset: 0. 0=Select FCH::PM::S5ResetStat to be S5/Reset Status register. 1=Select FCH::PM::S5ResetStat to be a debug status register.
1	<b>MemRstDisable.</b> Read-write. Reset: 0. 1=The memory reset function at DDR_RST# pin is disabled.
0	<b>Reset.</b> Write-1-only. Reset: 0. Writing 1 to this bit causes a PCI reset. This bit is enabled by the [ResetEn] bit.

**PMx0C5 [CF9 Shadow] (FCH::PM::CF9Shadow)**

Reset: 00h.	
_aliasHOST; PMx0C5; PM=FED8_0300h	
Bits	Description
7:4	Reserved.
3	<b>FullRst.</b> Read-write. Reset: 0. 0=Assert reset signals only. 1=Place system in S5 state for 3 to 5 seconds.
2	<b>RstCmd.</b> Write-only. Reset: 0. Write to 1 to generate reset as specified by bits [FullRst] and [SysRst]. Always Read as 0.
1	<b>SysRst.</b> Read-write. Reset: 0. 0=Send INIT HT message. 1=Reset as specified by [FullRst].
0	Reserved.

**PMx0C6 [HTControl] (FCH::PM::HTCtl)**

Reset: 0000h.	
_aliasHOST; PMx0C6; PM=FED8_0300h	
Bits	Description
15:0	Reserved.

**PMx0C8 [Misc] (FCH::PM::Misc)**

Read-write. Reset: 0008_000Ch.											
_aliasHOST; PMx0C8; PM=FED8_0300h											
Bits	Description										
31:24	<b>ClkIntrVectorOrd.</b> Read-write. Reset: 00h. Specifies the value used to identify the clock interrupt.										
23	Reserved.										
22	<b>ClkIntrVectorOrdEn.</b> Read-write. Reset: 0. 1=The system timer interrupt in the IOAPIC is tagged with a value defined by [ClkIntrVectorOrd].										
21:20	Reserved.										
19	<b>UseCpuRst.</b> Read-write. Reset: 1. 0=System reset causes INIT# instead of CPURST#.										
18	<b>UseBypassRom.</b> Read-write. Reset: 0. 1=Override the ROM straps and use BypassRomSel to determine which type of ROM to use. This is for BIOS debugging purposes or for systems having multiple BIOSes on board.										
17:16	<b>BypassRomSel.</b> Read-write. Reset: 0h. These two bits override the two ROM strap pins if [UseBypassRom] == 1.										
<b>ValidValues:</b>											
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>LPC ROM</td></tr> <tr> <td>1h</td><td>Reserved</td></tr> <tr> <td>2h</td><td>Reserved</td></tr> <tr> <td>3h</td><td>SPI ROM</td></tr> </table>	Value	Description	0h	LPC ROM	1h	Reserved	2h	Reserved	3h	SPI ROM
Value	Description										
0h	LPC ROM										
1h	Reserved										
2h	Reserved										
3h	SPI ROM										
15	<b>HideSmbus.</b> Read-write. Reset: 0. 1=Hide the SMBUS PCI configuration space and promote LPC bridge PCI configuration space to function 0.										
14	Reserved.										
13	<b>IdChangeEn.</b> Read-write. Reset: 0. 1=Allow the software to change.										
12	<b>S5ResetOverride.</b> Read-write. Reset: 0. 1=Mask off internet PCI reset used in ACPI.										
11	<b>WriteBackEnable.</b> Read-write. Reset: 0. 1=The WakeOnRing status bit is written back to HD audio controller upon system power up.										
10	<b>LlbEn.</b> Read-write. Reset: 0. 1=LLB function is enabled and system won't wake up from ACPI S-State until LLB# is de-asserted.										
9:8	<b>TempPolarity.</b> Read-write. Reset: 0h. Temperature polarity control for THRMTRIP and TALERT respectively.										
<b>ValidValues:</b>											
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0</td><td>Active low</td></tr> <tr> <td>1</td><td>Active high</td></tr> </table>	Value	Description	0	Active low	1	Active high				
Value	Description										
0	Active low										
1	Active high										



7	<b>DisablePciRom.</b> Read-write. Reset: 0. 1=Disable PCI from strap.
6	<b>TwarnEn.</b> Read-write. Reset: 0. 1=Enable TALERT# pin.
5:4	Reserved.
3	<b>TDeadEn.</b> Read-write. Reset: 1. 1=GEVENT2 takes up the THRMTRIP function; When THRMTRIP pin is low and [TFatalEn] is set, hardware switches the system to S5 automatically.
2	<b>TFatalEn.</b> Read-write. Reset: 1. 1=Enable both the soft PciRst and the THRMTRIP function.
1	Reserved.
0	<b>CpuIoPullDownDrvStrength.</b> Read-write. Reset: 0. 1=The integrated pull-down drive strength of all CPU IOs are increased by 50%.

**PMx0D3 [ManualReset] (FCH::PM::ManualReset)**

Read-write. Reset: 01h.

\_aliasHOST; PMx0D3; PM=FED8\_0300h

Bits	Description
7:1	Reserved.
0	<b>AssertLdtRstB.</b> Read-write. Reset: 1. 0=Assert LDT_RST# low.

**PMx0D5 [AltAcpiMmioEn] (FCH::PM::AltAcpiMmioEn)**

Read-write. Reset: 00h.

\_aliasHOST; PMx0D5; PM=FED8\_0300h

Bits	Description
7:2	Reserved.
1	<b>Alt_addr_width_sel.</b> Read-write. Reset: 0. 0=Alternate address is 32-bit width. 1=Alternate address is 64-bit width.
0	<b>Alt_addr_en.</b> Read-write. Reset: 0. 0=Alternate address disable. 1=Alternate address enable.

**PMx0D6 [AltAcpiMmioBase] (FCH::PM::AltAcpiMmioBase)**

Read-write. Reset: 0000h.

\_aliasHOST; PMx0D6; PM=FED8\_0300h

Bits	Description
15:0	<b>Lower_addr_alt.</b> Read-write. Reset: 0000h. Lower bits of base address. Its value only takes effect when alternate address is enabled (FCH::PM::AltAcpiMmioEn[Alt_addr_en] == 1). For example, when it is set as 0xABCD and alternate address width is 32-bit (FCH::PM::AltAcpiMmioEn[Alt_addr_width_sel] == 0), base address is 0xABCD_0000. If it is 64-bit then the base address is 0xFFFF_FFFF_ABCD_0000.

**PMx0D8 [Eprom Index] (FCH::PM::EpromEfuseIndex)**

Reset: 00h.

\_aliasHOST; PMx0D8; PM=FED8\_0300h

Bits	Description
7:0	Reserved.

**PMx0D9 [Eprom Data] (FCH::PM::EpromEfuseData)**

Reset: 00h.

\_aliasHOST; PMx0D9; PM=FED8\_0300h

Bits	Description
7:0	Reserved.

**PMx0DC [SataConfig2] (FCH::PM::SataCfg2)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; PMx0DC; PM=FED8\_0300h

Bits	Description
31:0	<b>Pmio_xDC_SataConfig.</b> Read-write. Reset: 0000_0000h. Used by Sata DEVSLP. This register only reset by RsmRstB or UserRstB, BIOS need to clear this register to 0x0 after Warm Reset so that device can response with

	SATA controller's OOB.
--	------------------------

**PMx0E0 [ABRegBar] (FCH::PM::ABRegBar)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; PMx0E0; PM=FED8\_0300h

Bits	Description
31:16	Reserved.
15:3	<b>ABRegBar</b> . Read-write. Reset: 0000h. IO Base address of AB Configuration Registers.
2:0	Reserved.

**PMx0E4 [AB Misc Control] (FCH::PM::ABDbg)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; PMx0E4; PM=FED8\_0300h

Bits	Description										
31:2	Reserved.										
1:0	<b>BlinkControl</b> . Read-write. Reset: 0h. This field controls the BLINK pin behavior. BLINK pin can be used to control the on-off state of an LED on the board. <b>ValidValues:</b>										
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>Disabled. BLINK pin can be controlled by GPIO logic if so configured.</td></tr> <tr> <td>1h</td><td>BLINK pin is repeatedly driven to low for 1 sec and then released for 3 sec.</td></tr> <tr> <td>2h</td><td>BLINK pin is repeatedly driven to low for 2 sec and then released for 2 sec.</td></tr> <tr> <td>3h</td><td>BLINK pin is always driven to low.</td></tr> </table>	Value	Description	0h	Disabled. BLINK pin can be controlled by GPIO logic if so configured.	1h	BLINK pin is repeatedly driven to low for 1 sec and then released for 3 sec.	2h	BLINK pin is repeatedly driven to low for 2 sec and then released for 2 sec.	3h	BLINK pin is always driven to low.
Value	Description										
0h	Disabled. BLINK pin can be controlled by GPIO logic if so configured.										
1h	BLINK pin is repeatedly driven to low for 1 sec and then released for 3 sec.										
2h	BLINK pin is repeatedly driven to low for 2 sec and then released for 2 sec.										
3h	BLINK pin is always driven to low.										

**PMx0EC [LpcGating] (FCH::PM::LpcGating)**

Read-write. Reset: 01h.

\_aliasHOST; PMx0EC; PM=FED8\_0300h

Bits	Description
7:3	Reserved.
2	<b>AbNoBypassEn</b> . Read-write. Reset: 0. Init: BIOS,1. 0=Disable. 1=Enable. Signals the A-Link that the LPC cycle should not be bypassed when a retry has timed out.
1	Reserved.
0	<b>LpcEnable</b> . Read-write. Reset: 1. 0=Disable. 1=Enable LPC bridge.

**PMx0FC [TraceMemoryEn] (FCH::PM::TraceMemoryEn)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; PMx0FC; PM=FED8\_0300h

Bits	Description
31:20	<b>TraceMemoryBaseAddr</b> . Read-write. Reset: 000h. The base address of trace memory. It is 1M memory space.
19:1	Reserved.
0	<b>TraceMemoryEn</b> . Read-write. Reset: 0. 0=Disable trace memory decoding. 1=Enable trace memory decoding. Set to 1 to enable trace memory decoding.

**9.2.10.3 Power Management (PM2) Registers**

The PM2 register space is accessed through two methods:

- Indirect IO access through index/data address pair at IOCD0 [PM2\_Index] and IOCD1 [PM2\_Data]. Software first programs the offset into the index register IOCD0 [PM2\_Index] and then Reads/Writes to/from the data register IOCD1 [PM2\_Data].
- Memory mapped access through the AcpiMmio region. The ACPI registers range from FED8\_0000h+400h to

FED8\_0000h+4FFh. See FCH::PM::IsaControl[MmioEn] for details on the AcpiMmio region.

#### PM2x000 [Fan0InputControl] (FCH::PM2::Fan0InCtl)

Read-write. Reset: 05h.

\_aliasHOST; PM2x000; PM2=FED8\_0400h

Bits	Description
7:4	Reserved.
3	<b>TwoRampAlgorithmEn.</b> Read-write. Reset: 0. 0=Disabled. 1=The two ramp fan control algorithm is enabled.
2:0	<b>FanInputControl.</b> Read-write. Reset: 5h.
<b>ValidValues:</b>	
Value	Description
4h-0h	Reserved.
5h	FanOut0 is enabled and temperature input is from TempTsi.
6h	FanOut0 is enabled and temperature input is 0.
7h	FanOut0 is disabled.

#### PM2x001 [Fan0Control] (FCH::PM2::Fan0Ctl)

Read-write. Reset: 00h.

When [AutoMode] == 1, the active duty cycle is controlled by the hardware automatically.

- If ActualTemperature < LowTemp{FCH::PM2::LoTemp0Hi,FCH::PM2::LoTemp0Lo}, DutyCycle = 0;
  - If LowTemp{FCH::PM2::LoTemp0Hi,FCH::PM2::LoTemp0Lo} <= ActualTemperature < MedTemp{FCH::PM2::MedTemp0Hi,FCH::PM2::MedTemp0Lo}, DutyCycle = FCH::PM2::LoDuty0[LowDuty0].
- If MedTemp{FCH::PM2::MedTemp0Hi,FCH::PM2::MedTemp0Lo} <= ActualTemperature < HighTemp{FCH::PM2::HiTemp0Hi,FCH::PM2::HiTemp0Lo},
  - If [LinearMode] == 0, DutyCycle = FCH::PM2::MedDuty0[MedDuty].
  - If [LinearMode] == 1, DutyCycle = ((ActualTemperature - LowTemp{FCH::PM2::LoTemp0Hi,FCH::PM2::LoTemp0Lo}) \* (FCH::PM2::Multiplier0[Multiplier] + 1) >> FCH::PM2::Multiplier0[DutySel]) + FCH::PM2::LoDuty0[LowDuty0].
- If ActualTemperature >= HighTemp{FCH::PM2::HiTemp0Hi,FCH::PM2::HiTemp0Lo}, DutyCycle = 100%.

In Automode, the hysteresis limit defined by FCH::PM2::LinearRng0 is applied to keep the fan from oscillating erratically.

\_aliasHOST; PM2x001; PM2=FED8\_0400h

Bits	Description
7:3	<b>LinearAdjust.</b> Read-write. Reset: 00h. Specifies the additional offset to effective duty cycle under linear mode.
2	<b>FanPolarity.</b> Read-write. Reset: 0. 0=FanOut0 drives low. 1=FanOut0 drives high.
1	<b>LinearMode.</b> Read-write. Reset: 0. 0=Use step function. 1=Use Linear function. See above description for details.
0	<b>AutoMode.</b> Read-write. Reset: 0. 0=FanOut0 is controlled by FCH::PM2::LoDuty0. 1=FanOut0 is controlled by the temperature input. See the above description for details on AutoMode.

#### PM2x002 [Fan0Freq] (FCH::PM2::Fan0Freq)

Read-write. Reset: 00h.

\_aliasHOST; PM2x002; PM2=FED8\_0400h

Bits	Description
7:0	<b>FanFreq.</b> Read-write. Reset: 00h. Normally, 4-wire fan runs at 25 KHz and 3-wire fan runs at 100 Hz. FanOut0 frequency is programmed as follows.
<b>ValidValues:</b>	
Value	Description
00h	28.64KHz
01h	25.78KHz

02h	23.44KHz
03h	21.48KHz
04h	19.83KHz
05h	18.41KHz
F6h-06h	1/(FanFreq*2048*15ns)
F7h	100Hz
F8h	87Hz
F9h	58Hz
FAh	44Hz
FBh	35Hz
FCh	29Hz
FDh	22Hz
FEh	14Hz
FFh	11Hz

**PM2x003 [LowDuty0] (FCH::PM2::LoDuty0)**

Read-write. Reset: 00h.

\_aliasHOST; PM2x003; PM2=FED8\_0400h

Bits	Description								
7:0	<p><b>LowDuty0.</b> Read-write. Reset: 00h. Specifies Fan0 duty number if (LowTemp{FCH::PM2::LoTemp0Hi, FCH::PM2::LoTemp0Lo} &lt;= temperature &lt; MedTemp{FCH::PM2::MedTemp0Hi, FCH::PM2::MedTemp0Lo}). There are 256 time slots in one fan cycle. Duty number N represents the (N+1)th time slot. Fan actively spins in time slot 0 ~ slot N, and stops from slot (N+1) ~ slot 255.</p> <p><b>ValidValues:</b></p> <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>00h</td><td>Always stop.</td></tr> <tr> <td>FEh-01h</td><td>Fan spins for time slots &lt;= LowDuty, stops for time slots &gt; LowDuty.</td></tr> <tr> <td>FFh</td><td>Run full speed.</td></tr> </table>	Value	Description	00h	Always stop.	FEh-01h	Fan spins for time slots <= LowDuty, stops for time slots > LowDuty.	FFh	Run full speed.
Value	Description								
00h	Always stop.								
FEh-01h	Fan spins for time slots <= LowDuty, stops for time slots > LowDuty.								
FFh	Run full speed.								

**PM2x004 [MedDuty0] (FCH::PM2::MedDuty0)**

Read-write. Reset: 00h.

\_aliasHOST; PM2x004; PM2=FED8\_0400h

Bits	Description								
7:0	<p><b>MedDuty.</b> Read-write. Reset: 00h. Specifies Fan0 duty number using step function if (MedTemp{FCH::PM2::MedTemp0Hi, FCH::PM2::MedTemp0Lo} &lt;= temperature &lt;= HighTemp{FCH::PM2::HiTemp0Hi, FCH::PM2::HiTemp0Lo}). There are 256 time slots in one fan cycle. Duty number N represents the (N+1)th time slot. Fan actively spins in time slot 0 ~ slot N, and stops from slot (N+1) ~ slot 255.</p> <p><b>ValidValues:</b></p> <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>00h</td><td>Always stop.</td></tr> <tr> <td>FEh-01h</td><td>Fan spins for time slots &lt;= MedDuty, stops for time slots &gt; MedDuty.</td></tr> <tr> <td>FFh</td><td>Run full speed.</td></tr> </table>	Value	Description	00h	Always stop.	FEh-01h	Fan spins for time slots <= MedDuty, stops for time slots > MedDuty.	FFh	Run full speed.
Value	Description								
00h	Always stop.								
FEh-01h	Fan spins for time slots <= MedDuty, stops for time slots > MedDuty.								
FFh	Run full speed.								

**PM2x005 [Multiplier0] (FCH::PM2::Multiplier0)**

Read-write. Reset: 00h.

- HighTemp0[15:0] = {FCH::PM2::HiTemp0Hi[HighTemp0[15:8]], FCH::PM2::HiTemp0Lo[HighTemp0[7:0]]}.
- Hysteresis0[15:0] = FCH::PM2::Fan0Hysteresis[HysteresisHi, HysteresisLo].
- Med2Temp0[15:0] =

- {FCH::PM2::Med2Temp0Hi[Med2Temp0[15:8]],FCH::PM2::Med2Temp0Lo[Med2Temp0[7:0]]}.
- SlopeHi[7:2] are the integer bits and SlopeHi[1:0] are the fractional bits of the ramp slope.
  - BIOS has to calculate SlopeHi[7:0] using the following equation:
    - $$\text{SlopeHi} = (16'hFF00 - \{FCH::PM2::Med2Duty0[Med2Duty0[7:0]], 8'b0\}) / (\text{HighTemp0}[15:0] - \text{Hysteresis0}[15:0] - \text{Med2Temp0}[15:0])$$

For example, if our setting is:

- Med2Duty0[7:0] = 25% = 40h
- HighTemp0[15:0] = 90°C = 5A00h
- Med2Temp0[15:0] = 40°C = 2800h
- Hysteresis0[15:0] = 10°C = 0A00h

Then:

- $$\text{SlopeHi} = (FF00h - 4000h) / (5A00h - 0A00h - 2800h) = 48896 / 10240 = 4.775$$

Converting the number 4.775 into the 8-bit format: 00010011 (13h), BIOS should program the Multiplier0 register as 13h in this example.

\_aliasHOST; PM2x005; PM2=FED8\_0400h

Bits	Description
7:6	<b>DutySel.</b> Read-write. Reset: 0h. When Fan0 is programmed in AutoMode with linear function being selected, this field selects part of duty to be fed into fan as described in FCH::PM2::Fan0Ctl register description. When Fan0 is programmed in TwoRamp mode, this field specifies the upper 2 bits in Slope[7:0]. See the above register description for detail.
5:0	<b>Multiplier.</b> Read-write. Reset: 00h. When Fan0 is programmed in AutoMode with linear function being selected, this field specifies the factor to calculate duty number as described in FCH::PM2::Fan0Ctl register description. When Fan0 is programmed in TwoRamp mode, this field specifies the lower 6 bits in Slope[7:0]. See the above register description for detail.

#### PM2x006 [LowTemp0Lo] (FCH::PM2::LoTemp0Lo)

Read-write. Reset: 00h.

\_aliasHOST; PM2x006; PM2=FED8\_0400h

Bits	Description
7:0	<b>LowTemp0[7:0].</b> Read-write. Reset: 00h. Specifies the lower 8 bits of the low temperature threshold. See also FCH::PM2::LoTemp0Hi[LowTemp0[15:8]].

#### PM2x007 [LowTemp0Hi] (FCH::PM2::LoTemp0Hi)

Read-write. Reset: 00h.

\_aliasHOST; PM2x007; PM2=FED8\_0400h

Bits	Description
7:0	<b>LowTemp0[15:8].</b> Read-write. Reset: 00h. Specifies the higher 8 bits of the low temperature threshold. LowTemp0 = {LowTemp0[15:8],FCH::PM2::LoTemp0Lo[LowTemp0[7:0]]}.

#### PM2x008 [MedTemp0Lo] (FCH::PM2::MedTemp0Lo)

Read-write. Reset: 00h.

\_aliasHOST; PM2x008; PM2=FED8\_0400h

Bits	Description
7:0	<b>MedTemp0[7:0].</b> Read-write. Reset: 00h. Specifies the lower 8 bits of the medium temperature threshold. See also FCH::PM2::MedTemp0Hi[MedTemp0[15:8]].

#### PM2x009 [MedTemp0Hi] (FCH::PM2::MedTemp0Hi)

Read-write. Reset: 00h.

\_aliasHOST; PM2x009; PM2=FED8\_0400h

Bits	Description
------	-------------

7:0	<b>MedTemp0[15:8]</b> . Read-write. Reset: 00h. Specifies the higher 8 bits of the medium temperature threshold. MedTemp0 = {MedTemp0[15:8], FCH::PM2::MedTemp0Lo[MedTemp0[7:0]]}.
-----	--

**PM2x00A [HighTemp0Lo] (FCH::PM2::HiTemp0Lo)**

Read-write. Reset: 00h.

\_aliasHOST; PM2x00A; PM2=FED8\_0400h

Bits	Description
7:0	<b>HighTemp0[7:0]</b> . Read-write. Reset: 00h. Specifies the lower 8 bits of the high temperature threshold. See also FCH::PM2::HiTemp0Hi[HighTemp0[15:8]].

**PM2x00B [HighTemp0Hi] (FCH::PM2::HiTemp0Hi)**

Read-write. Reset: 00h.

\_aliasHOST; PM2x00B; PM2=FED8\_0400h

Bits	Description
7:0	<b>HighTemp0[15:8]</b> . Read-write. Reset: 00h. Specifies the higher 8 bits of the high temperature threshold. HighTemp0 = {HighTemp0[15:8], FCH::PM2::HiTemp0Lo[HighTemp0[7:0]]}.

**PM2x00C [LinearRange0] (FCH::PM2::LinearRng0)**

Read-write. Reset: 00h.

\_aliasHOST; PM2x00C; PM2=FED8\_0400h

Bits	Description
7:0	<b>LinearRange</b> . Read-write. Reset: 00h. Specifies a variable range that Fan0 can tolerate. Fan0 is not affected if temperature varies within this range.

**PM2x00D [LinearHoldCount0] (FCH::PM2::LinearHoldCnt0)**

Read-write. Reset: 00h.

\_aliasHOST; PM2x00D; PM2=FED8\_0400h

Bits	Description
7:0	<b>LinearHoldCount</b> . Read-write. Reset: 00h. Specifies the fan cycles to wait before the duty cycle can be changed.

**PM2x00E [Fan0Hysteresis] (FCH::PM2::Fan0Hysteresis)**

Read-write. Reset: 0000h.

\_aliasHOST; PM2x00E; PM2=FED8\_0400h

Bits	Description
15:8	<b>HysteresisHi</b> . Read-write. Reset: 00h. Specifies the hysteresis value (in temperature) of the Two Ramp Fan Control Algorithm. The unit is °C. See FCH::PM2::Multiplier0 on how to program this register.
7:0	<b>HysteresisLo</b> . Read-write. Reset: 00h. This byte should always be programmed as 0.

**PM2x050 [Med2Temp0Lo] (FCH::PM2::Med2Temp0Lo)**

Read-write. Reset: 00h.

\_aliasHOST; PM2x050; PM2=FED8\_0400h

Bits	Description
7:0	<b>Med2Temp0[7:0]</b> . Read-write. Reset: 00h. IF (FCH::PM2::Fan0InCtl[TwoRampAlgorithmEn] == 1) THEN It specifies the lower byte of the temperature value of the turning point on the ramp. See also FCH::PM2::Med2Temp0Hi[Med2Temp0[15:8]]. ELSE Unused. ENDIF.

**PM2x051 [Med2Temp0Hi] (FCH::PM2::Med2Temp0Hi)**

Read-write. Reset: 00h.

\_aliasHOST; PM2x051; PM2=FED8\_0400h

Bits	Description
7:0	<b>Med2Temp0[15:8]</b> . Read-write. Reset: 00h. IF (FCH::PM2::Fan0InCtl[TwoRampAlgorithmEn] == 1) THEN It specifies the higher byte of the temperature value of the turning point on the ramp. Med2Temp0 = {Med2Temp0[15:8], FCH::PM2::Med2Temp0Lo[Med2Temp0[7:0]]}. ELSE Unused. ENDIF.

**PM2x052 [Med2Duty0] (FCH::PM2::Med2Duty0)**

Read-write. Reset: 00h.

\_aliasHOST; PM2x052; PM2=FED8\_0400h

Bits	Description
7:0	<b>Med2Duty0[7:0]</b> . Read-write. Reset: 00h. IF (FCH::PM2::Fan0InCtl[TwoRampAlgorithmEn] == 1) THEN It specifies the fan duty value of the turning point on the ramp. ELSE Unused. ENDIF.

**PM2x053 [Multiplier2\_0] (FCH::PM2::Multiplier20)**

Read-write. Reset: 00h.

- MedTemp0[15:0] = {FCH::PM2::MedTemp0Hi[MedTemp0[15:8]], FCH::PM2::MedTemp0Lo[MedTemp0[7:0]]}.
- Med2Temp0[15:0] = {FCH::PM2::Med2Temp0Hi[Med2Temp0[15:8]], FCH::PM2::Med2Temp0Lo[Med2Temp0[7:0]]}.
- Multiplier2\_0[7:2] are the integer bits and Multiplier2\_0[1:0] are the fractional bits of the ramp slope.
- BIOS has to calculate Multiplier2\_0[7:0] using the following equation:
  - $\text{Multiplier2\_0} = (\{ \text{FCH::PM2::Med2Duty0}[\text{Med2Duty0}[7:0]], 8'b0 \} - \{ \text{FCH::PM2::LoDuty0}[\text{LowDuty0}[7:0]], 8'b0 \}) / (\text{Med2Temp0}[15:0] - \text{MedTemp0}[15:0])$

For example, if our setting is:

- Med2Duty0[7:0] = 50% = 80h
- LowDuty0[7:0] = 25% = 40h
- Med2Temp0[15:0] = 70°C = 4600h
- MedTemp0[15:0] = 40°C = 2800h

Then:

- $\text{Multiplier2\_0} = (8000h - 4000h) / (4600h - 2800h) = 16384 / 7680 = 2.133$

Converting the number 2.133 into the 8-bit format: 00001000b (08h), BIOS should program Multiplier20 register to 08h in this example.

\_aliasHOST; PM2x053; PM2=FED8\_0400h

Bits	Description
7:0	<b>Multiplier2_0[7:0]</b> . Read-write. Reset: 00h. IF (FCH::PM2::Fan0InCtl[TwoRampAlgorithmEn] == 1) THEN Specifies the slope value of ramp1lo and ramp0lo in Fan Control Algorithm. Bits[7:2] are integer bits. Bits[1:0] are fractional bits. ELSE Unused. ENDIF.

**PM2x060 [FanStatus] (FCH::PM2::FanStat)**

Read,Write-1-to-clear. Reset: 00h.

\_aliasHOST; PM2x060; PM2=FED8\_0400h

Bits	Description
7:1	Reserved.
0	<b>Fan0SpeedTooSlow</b> . Read,Write-1-to-clear. Reset: 0. 1=Fan0 runs slower than the value in the Fan0SpeedLimit{FCH::PM2::Fan0SpeedLimitHi, FCH::PM2::Fan0SpeedLimitLo}. Fan0 speed too slow.

**PM2x061 [FanINTRouteLo] (FCH::PM2::FanINTRouteLo)**

Read-write. Reset: 00h.

\_aliasHOST; PM2x061; PM2=FED8\_0400h

Bits	Description				
7:2	Reserved.				
1:0	<b>Fan0INTRoute</b> . Read-write. Reset: 0h. <b>ValidValues:</b>				
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>No SCI/SMI generated.</td></tr> </table>	Value	Description	0h	No SCI/SMI generated.
Value	Description				
0h	No SCI/SMI generated.				



1h	SMI.
2h	SMI or SCI depending on GEVENT13 routing.
3h	No SCI/SMI generated.

**PM2x063 [SampleFreqDiv] (FCH::PM2::SampleFreqDiv)**

Read-write. Reset: 00h.

\_aliasHOST; PM2x063; PM2=FED8\_0400h

Bits	Description										
7:4	<b>LinearRangeOutLimit[4:1]</b> . Read-write. Reset: 0h. LinearRangeOutLimit[7:0] = {000b, LinearRangeOutLimit[4:1], 1b}. LinearRangeOutLimit specifies how close the fan duty follows the target duty cycle and is only used when the fan duty is changing. It is different from the FCH::PM2::LinearRng0, which works like a hysteresis and used when fan duty is not changing.										
3	Reserved.										
2	<b>FanLinearEnhanceEn2</b> . Read-write. Reset: 0. 1=The positive hysteresis of fan duty is removed; FCH::PM2::LinearRng0 only applies to the negative direction; as a result, the fan duty increases once the temperature is increased instead of waiting for a hysteresis.										
1:0	<b>SampleFreqDiv</b> . Read-write. Reset: 0h. This field specifies the sampling rate of Fan Speed. <b>ValidValues:</b> <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>Base (22.5KHz)</td></tr> <tr> <td>1h</td><td>Base (22.5KHz) / 2</td></tr> <tr> <td>2h</td><td>Base (22.5KHz) / 4</td></tr> <tr> <td>3h</td><td>Base (22.5KHz) / 8</td></tr> </table>	Value	Description	0h	Base (22.5KHz)	1h	Base (22.5KHz) / 2	2h	Base (22.5KHz) / 4	3h	Base (22.5KHz) / 8
Value	Description										
0h	Base (22.5KHz)										
1h	Base (22.5KHz) / 2										
2h	Base (22.5KHz) / 4										
3h	Base (22.5KHz) / 8										

**PM2x064 [FanDebounceCounterLo] (FCH::PM2::FanDebounceCtrLo)**

Read-write. Reset: 00h.

\_aliasHOST; PM2x064; PM2=FED8\_0400h

Bits	Description
7:0	<b>FanDebounceCounter[7:0]</b> . Read-write. Reset: 00h. Specifies the lower 8 bits of the debounce counter when measuring fan speed. See also FCH::PM2::FanDebounceCtrHi[FanDebounceCounter[15:8]].

**PM2x065 [FanDebounceCounterHi] (FCH::PM2::FanDebounceCtrHi)**

Read-write. Reset: 00h.

\_aliasHOST; PM2x065; PM2=FED8\_0400h

Bits	Description
7:0	<b>FanDebounceCounter[15:8]</b> . Read-write. Reset: 00h. Specifies the high 8 bits of the debounce counter when measuring fan speed. FanDebounceCounter = {FanDebounceCounter[15:8], FCH::PM2::FanDebounceCtrLo[FanDebounceCounter[7:0]]}.

**PM2x066 [Fan0DetectorControl] (FCH::PM2::Fan0DetectorCtl)**

Read-write. Reset: 00h.

\_aliasHOST; PM2x066; PM2=FED8\_0400h

Bits	Description
7:5	Reserved.
4	<b>ShutDownEnable</b> . Read-write. Reset: 0. 1=The system is shutdown if FCH::PM2::FanStat[Fan0SpeedTooSlow] remains 1 for more than 4 seconds.
3:2	Reserved.
1	<b>UseAverage</b> . Read-write. Reset: 0. 0=Do not use average fan0 speed. 1=Use average fan0 speed.
0	<b>FanDetectorEnable</b> . Read-write. Reset: 0. 0=Disable fan0 speed measurement. 1=Enable fan0 speed measurement.

**PM2x067 [Fan0SpeedLimitLo] (FCH::PM2::Fan0SpeedLimitLo)**

Read-write. Reset: 00h.	
_aliasHOST; PM2x067; PM2=FED8_0400h	
Bits	Description
7:0	<b>FanSpeedLimit[7:0]</b> . Read-write. Reset: 00h. Specifies the lower 8 bits of Fan0SpeedLimit. See also FCH::PM2::Fan0SpeedLimitHi[FanSpeedLimit[15:8]].

PM2x068 [Fan0SpeedLimitHi] (FCH::PM2::Fan0SpeedLimitHi)	
Read-write. Reset: 00h.	
_aliasHOST; PM2x068; PM2=FED8_0400h	
Bits	Description
7:0	<b>FanSpeedLimit[15:8]</b> . Read-write. Reset: 00h. Specifies the higher 8 bits of Fan0SpeedLimit. FanSpeedLimit = {FanSpeedLimit[15:8],FCH::PM2::Fan0SpeedLimitLo[FanSpeedLimit[7:0]]}.

PM2x069 [Fan0SpeedLo] (FCH::PM2::Fan0SpeedLo)	
Read-only. Reset: 00h.	
_aliasHOST; PM2x069; PM2=FED8_0400h	
Bits	Description
7:0	<b>FanSpeed[7:0]</b> . Read-only. Reset: 00h. Specifies the lower 8 bits of fan0 speed. See also FCH::PM2::Fan0SpeedHi[FanSpeed[15:8]].

PM2x06A [Fan0SpeedHi] (FCH::PM2::Fan0SpeedHi)	
Read-only. Reset: 00h.	
_aliasHOST; PM2x06A; PM2=FED8_0400h	
Bits	Description
7:0	<b>FanSpeed[15:8]</b> . Read-only. Reset: 00h. Specifies the higher 8 bits of fan0 speed. FanSpeed = {FanSpeed[15:8],FCH::PM2::Fan0SpeedLo[FanSpeed[7:0]]}.

PM2x08A [TempTsiLo] (FCH::PM2::TempTsiLo)	
Reset: 00h.	
_aliasHOST; PM2x08A; PM2=FED8_0400h	
Bits	Description
7:0	<b>TempTsi[7:0]</b> . Reset: 00h. Specifies the lower 8 bits of TempTsi. See also FCH::PM2::TempTsiHi[TempTsi[15:8]].
AccessType: FCH::PM2::TempTsiWe[TempTsiWe] ? Read-write, Volatile : Read-only, Volatile.	

PM2x08B [TempTsiHi] (FCH::PM2::TempTsiHi)	
Reset: 00h.	
_aliasHOST; PM2x08B; PM2=FED8_0400h	
Bits	Description
7:0	<b>TempTsi[15:8]</b> . Reset: 00h. Specifies the higher 8 bits of TempTsi. TempTsi = {TempTsi[15:8],FCH::PM2::TempTsiLo[TempTsi[7:0]]}.
AccessType: FCH::PM2::TempTsiWe[TempTsiWe] ? Read-write, Volatile : Read-only, Volatile.	

PM2x08C [TempTsiLimitLo] (FCH::PM2::TempTsiLimitLo)	
Read-write. Reset: 00h.	
_aliasHOST; PM2x08C; PM2=FED8_0400h	
Bits	Description
7:0	<b>TempTsiLimit[7:0]</b> . Read-write. Reset: 00h. Specifies the lower 8 bits of TempTsiLimit. See FCH::PM2::TempTsiLimitHi[TempTsiLimit[15:8]].

PM2x08D [TempTsiLimitHi] (FCH::PM2::TempTsiLimitHi)	
Read-write. Reset: 00h.	
_aliasHOST; PM2x08D; PM2=FED8_0400h	

Bits	Description
7:0	<b>TempTsiLimit[15:8]</b> . Read-write. Reset: 00h. Specifies the higher 8 bits of TempTsiLimit. TempTsiLimit[15:0] = {TempTsiLimit[15:8], FCH::PM2::TempTsiLimitLo[TempTsiLimit[7:0]]}.

#### PM2x08E [TempTsiChangeLimit] (FCH::PM2::TempTsiChangeLimit)

Read-write. Reset: 00h.

If (TempTsiChangeLimit != 0), filtering is applied to TempTsi{FCH::PM2::TempTsiHi, FCH::PM2::TempTsiLo} as below:

- If TempTsi\_New > (TempTsi\_Old + (TempTsiChangeLimit<<6)), then ... (Action to take.)
- If TempTsi\_New < (TempTsi\_Old - (TempTsiChangeLimit<<6)), then ... (Action to take.)
- If (TempTsi\_Old - (TempTsiChangeLimit<<6)) <= TempTsi\_New <= (TempTsi\_Old + (TempTsiChangeLimit<<6)), then ... (Action to take.)

\_aliasHOST; PM2x08E; PM2=FED8\_0400h

Bits	Description
7:0	<b>TempTsiChangeLimit</b> . Read-write. Reset: 00h. Limit of the change seen by the temperature sensor to be used when defining actions to be taken on temperature changes.

#### PM2x08F [TempTsiWe] (FCH::PM2::TempTsiWe)

Read-write. Reset: 00h.

\_aliasHOST; PM2x08F; PM2=FED8\_0400h

Bits	Description
7:6	Reserved.
5	<b>TempTsiWe: TempTsi Write Enable</b> . Read-write. Reset: 0. 0=TempTsi registers, FCH::PM2::TempTsiHi and FCH::PM2::TempTsiLo, are Read-only and updated by hardware with the result from TempTsi sensor. 1=TempTsi registers are Writable only by host and IMC; they are not updated by hardware with the result from the TempTsi sensor.
4:0	Reserved.

#### PM2x090 [TempTsiStatus] (FCH::PM2::TempTsiStat)

Read, Write-1-to-clear. Reset: 00h.

\_aliasHOST; PM2x090; PM2=FED8\_0400h

Bits	Description
7:6	Reserved.
5	<b>TempTsiStatus</b> . Read, Write-1-to-clear. Reset: 0. 1=TempTsi{FCH::PM2::TempTsiHi, FCH::PM2::TempTsiLo} is out of the limit.
4:0	Reserved.

#### PM2x092 [TempTsiControl] (FCH::PM2::TempTsiCtl)

Read-write. Reset: 00h.

\_aliasHOST; PM2x092; PM2=FED8\_0400h

Bits	Description										
7:4	Reserved.										
3:2	<b>TempTsiControl</b> . Read-write. Reset: 0h. TempTsi sensor is enabled if (TempTsiControl != 0). <b>ValidValues:</b>										
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>Disable.</td></tr> <tr> <td>1h</td><td>If (TempTsi &gt; TempTsiLimit), then [TempTsiStatus] = 1.</td></tr> <tr> <td>2h</td><td>If (TempTsi &lt; TempTsiLimit), then [TempTsiStatus] = 1.</td></tr> <tr> <td>3h</td><td>If ((TempTsiHi &gt; TempTsiLimitLo)    (TempTsiHi &lt; TempTsiLimitHi)), then [TempTsiStatus] = 1.</td></tr> </table>	Value	Description	0h	Disable.	1h	If (TempTsi > TempTsiLimit), then [TempTsiStatus] = 1.	2h	If (TempTsi < TempTsiLimit), then [TempTsiStatus] = 1.	3h	If ((TempTsiHi > TempTsiLimitLo)    (TempTsiHi < TempTsiLimitHi)), then [TempTsiStatus] = 1.
Value	Description										
0h	Disable.										
1h	If (TempTsi > TempTsiLimit), then [TempTsiStatus] = 1.										
2h	If (TempTsi < TempTsiLimit), then [TempTsiStatus] = 1.										
3h	If ((TempTsiHi > TempTsiLimitLo)    (TempTsiHi < TempTsiLimitHi)), then [TempTsiStatus] = 1.										
1:0	Reserved.										

#### PM2x094 [TempTsiINTRoute] (FCH::PM2::TempTsiINTRoute)

Read-write. Reset: 00h.											
_aliasHOST; PM2x094; PM2=FED8_0400h											
Bits	Description										
7:4	Reserved.										
3:2	<b>TempTsiINTRoute</b> . Read-write. Reset: 0h.										
	<b>ValidValues:</b>										
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>No SCI/SMI generated.</td></tr> <tr> <td>1h</td><td>SMI.</td></tr> <tr> <td>2h</td><td>SMI or SCI according to GEVENT 13 INT routing.</td></tr> <tr> <td>3h</td><td>No SCI/SMI generated.</td></tr> </table>	Value	Description	0h	No SCI/SMI generated.	1h	SMI.	2h	SMI or SCI according to GEVENT 13 INT routing.	3h	No SCI/SMI generated.
Value	Description										
0h	No SCI/SMI generated.										
1h	SMI.										
2h	SMI or SCI according to GEVENT 13 INT routing.										
3h	No SCI/SMI generated.										
1:0	Reserved.										

**PM2x0DF [TempTsiRstSel] (FCH::PM2::TempTsiRstSel)**

Read-write. Reset: 00h.	
_aliasHOST; PM2x0DF; PM2=FED8_0400h	
Bits	Description
7:6	Reserved.
5	<b>TempTsiRstSel</b> . Read-write. Reset: 0. 1=Thermal diode monitoring function is not stopped by reset.
4:0	Reserved.

**PM2x0E0 [AlertThermaltripStatus] (FCH::PM2::AlertThermtripStat)**

Read-only. Reset: 00h.	
_aliasHOST; PM2x0E0; PM2=FED8_0400h	
Bits	Description
7:2	Reserved.
1	<b>ThermalTripStatus</b> . Read-only. Reset: 0. 0=Current temperature is not above ThermalTripLimit. 1=Current temperature is above ThermalTripLimit.
0	<b>AlertStatus</b> . Read-only. Reset: 0. 0=Current temperature is not above AlertLimit. 1=Current temperature is above AlertLimit.

**PM2x0E1 [AlertLimitLo] (FCH::PM2::AlertLimitLo)**

Read-write. Reset: 00h.	
_aliasHOST; PM2x0E1; PM2=FED8_0400h	
Bits	Description
7:0	<b>AlertLimit[7:0]</b> . Read-write. Reset: 00h. Thermal alert limit lower 8 bits. See also FCH::PM2::AlertLimitHi[AlertLimit[15:8]].

**PM2x0E2 [AlertLimitHi] (FCH::PM2::AlertLimitHi)**

Read-write. Reset: 00h.	
_aliasHOST; PM2x0E2; PM2=FED8_0400h	
Bits	Description
7:0	<b>AlertLimit[15:8]</b> . Read-write. Reset: 00h. Thermal alert limit upper 8 bits. AlertLimit[15:0] = {AlertLimit[15:8], FCH::PM2::AlertLimitLo[AlertLimit[7:0]]}.

**PM2x0E3 [ThermalTripLimitLo] (FCH::PM2::ThermTripLimitLo)**

Read-write. Reset: 00h.	
_aliasHOST; PM2x0E3; PM2=FED8_0400h	
Bits	Description
7:0	<b>ThermalTripLimit[7:0]</b> . Read-write. Reset: 00h. Thermal trip limit to enable system shutdown. Lower 8 bit of the trip limit. See also FCH::PM2::ThermTripLimitHi[ThermalTripLimit[15:8]].

**PM2x0E4 [ThermalTripLimitHi] (FCH::PM2::ThermTripLimitHi)**

Read-write. Reset: 00h.

\_aliasHOST; PM2x0E4; PM2=FED8\_0400h

Bits	Description
7:0	<b>ThermalTripLimit[15:8]</b> . Read-write. Reset: 00h. Thermal trip limit upper 8 bits. ThermalTripLimit[15:0] = {ThermalTripLimit[15:8], FCH::PM2::ThermTripLimitLo[ThermalTripLimit[7:0]]}.

**PM2x0E5 [AlertThermaltripControl] (FCH::PM2::AlertThermtripCtl)**

Read-write. Reset: 00h.

\_aliasHOST; PM2x0E5; PM2=FED8\_0400h

Bits	Description								
7:5	<b>TempSelAlert</b> . Read-write. Reset: 0h. This field selects which temperature sensor is the event source. It converts the selected Temp input pin into either TAlert or ThermalTrip function. <b>ValidValues:</b> <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>4h-0h</td><td>Reserved.</td></tr> <tr> <td>5h</td><td>TempTsi.</td></tr> <tr> <td>7h-6h</td><td>Reserved.</td></tr> </table>	Value	Description	4h-0h	Reserved.	5h	TempTsi.	7h-6h	Reserved.
Value	Description								
4h-0h	Reserved.								
5h	TempTsi.								
7h-6h	Reserved.								
4:2	Reserved.								
1:0	<b>AlertControl</b> . Read-write. Reset: 0h. Enable ThermalTrip or Talert on the selected Temp input. <b>ValidValues:</b> <table> <tr> <th>Bit</th><th>Description</th></tr> <tr> <td>[0]</td><td>1=Enable Talert on the selected temperature input.</td></tr> <tr> <td>[1]</td><td>1=Enable ThermalTrip on the selected temperature input.</td></tr> </table>	Bit	Description	[0]	1=Enable Talert on the selected temperature input.	[1]	1=Enable ThermalTrip on the selected temperature input.		
Bit	Description								
[0]	1=Enable Talert on the selected temperature input.								
[1]	1=Enable ThermalTrip on the selected temperature input.								

**9.2.10.4 Standard ACPI Registers****9.2.10.4.1 AcpiPmEvtBlk**

The IO mapped base address of this register block is defined by FCH::PM::AcpiPm1EvtBlk.

**PMx0800 [Pm1Status] (FCH::PM::Pm1Stat)**

Read,Write-1-to-clear. Reset: 0000h.

\_aliasPM; PMx0800; PM=FED8\_0000h

Bits	Description
15	<b>WakeStatus</b> . Read,Write-1-to-clear. Reset: 0. 1=The system is in the sleep state and a wake-up event occurs. Wake status.
14	<b>PciExpWakeStatus</b> . Read,Write-1-to-clear. Reset: 0. 1=The system wake is due to a PCI Express wakeup event. PCI Express® wake status.
13:11	Reserved.
10	<b>RtcStatus</b> . Read,Write-1-to-clear. Reset: 0. 1=RTC generates an alarm. RTC status, set-by-hardware. If both FCH::PM::Pm1En[RtcEn] and this bit are set to 1, a power management event: SCI, SMI or resume event, is generated.
9	Reserved.
8	<b>PwrBtnStatus</b> . Read,Write-1-to-clear. Reset: 0. 1=The Power Button is pressed. Power Button status bit. In the system working state, if FCH::PM::Pm1En[PwrBtnEn] and this bit are both set to 1, an interrupt event is raised. In the sleeping or soft-off state, a wake event is generated when the Power Button is pressed regardless of the setting of FCH::PM::Pm1En[PwrBtnEn].
7:6	Reserved.

5	<b>GblStatus.</b> Read,Write-1-to-clear. Reset: 0. 1=An SCI is generated due to the BIOS wanting the attention of the SCI handler; Writing 1 to FCH::PM::AcpiCfg[BiosRls] sets this bit. Global status.
4:1	Reserved.
0	<b>TmrStatus.</b> Read,Write-1-to-clear. Reset: 0. 1=The 31st bit of the 32-bit counter changes from low to high or high to low. Timer status, set-by-hardware. Timer carry status bit. If both FCH::PM::Pm1En[TmrEn] and this bit are set to 1, an interrupt event is raised.

**PMx0802 [Pm1Enable] (FCH::PM::Pm1En)**

Read-write. Reset: 4000h.

\_aliasPM; PMx0802; PM=FED8\_0000h

Bits	Description
15	Reserved.
14	<b>PciExpWakeDis.</b> Read-write. Reset: 1. 1=Disable the inputs to FCH::PM::Pm1Stat[PciExpWakeStatus] from waking the system.
13:11	Reserved.
10	<b>RtcEn.</b> Read-write. Reset: 0. 1=A wake event is generated whenever FCH::PM::Pm1Stat[RtcStatus] is also set to 1.
9	Reserved.
8	<b>PwrBtnEn.</b> Read-write. Reset: 0. 1=A power management event (SCI or wake) is generated whenever FCH::PM::Pm1Stat[PwrBtnStatus] is also set to 1.
7:6	Reserved.
5	<b>GblEn.</b> Read-write. Reset: 0. 1=An SCI is raised whenever FCH::PM::Pm1Stat[GblStatus] is also set to 1.
4:1	Reserved.
0	<b>TmrEn.</b> Read-write. Reset: 0. 0=No interrupt is generated when the FCH::PM::Pm1Stat[TmrStatus] bit is set to 1. 1=An SCI event is generated whenever the FCH::PM::Pm1Stat[TmrStatus] is set to 1. This is the timer carry interrupt enable bit.

**9.2.10.4.2 AcpiPm1CntBlk**

The IO mapped base address of this register block is defined by FCH::PM::AcpiPm1CntBlk.

**PMx0804 [PmControl] (FCH::PM::PmCtl)**

Reset: 0000\_0000h.

\_aliasPM; PMx0804; PM=FED8\_0000h

Bits	Description
31:14	Reserved.
13	<b>SlpTypeEn.</b> Write-1-only. Reset: 0. 1=The system sequences into the sleeping state defined by [SlpTyp] when FCH::PM::ResetCtl1[SlpTypeControl] is set to 1.
12:10	<b>SlpTyp.</b> Read-write. Reset: 0h. Specifies the sleep state the system enters when SlpTypeEn == 1. This design currently implements 5 states: S0, S1, S3, S4, and S5.
9:3	Reserved.
2	<b>GblIRIs.</b> Write-only. Reset: 0. If FCH::SMI::SmiCtl4[Smicontrol72] is set to 01b, writing 1 to this bit generates SMI and sets FCH::SMI::SmiStat2[GblIRIsEvent72].
1	Reserved.
0	<b>SciEn.</b> Read-write. Reset: 0. 0=Power management events generate SMI interrupts. 1=Power management events generate SCI interrupts. Selects the power management event to be either an SCI or SMI interrupt for the power management events.

**9.2.10.4.3 AcpiPmTmrBlk**

The IO mapped base address of this register block is defined by FCH::PM::AcpiPmTmrBlk.

#### PMx0808 [TmrValue/ETmrValue] (FCH::PM::TmrValue\_ETmrValu)

Read-only.	
_aliasPM; PMx0808; PM=FED8_0000h	
Bits	Description
31:0	<b>TmrValue.</b> Read-only. Reset: XXXX_XXXXh. It returns the running count of the power management timer (ACPI timer).

#### 9.2.10.4.4 AcpiGpe0Blk

The IO mapped base address of this register block is defined by FCH::PM::AcpiGpe0Blk.

#### PMx0814 [EventStatus] (FCH::PM::EventStat)

Read,Write-1-to-clear.	
_aliasPM; PMx0814; PM=FED8_0000h	
Bits	Description
31:0	<b>EventStatus.</b> Read,Write-1-to-clear. Reset: XXXX_XXXXh. Each bit represents an ACPI event status. For each bit, configuration for the events are located at FCH::SMI::SciTrig through FCH::SMI::SciMap14. The status bits are also mirrored in FCH::PM::EventStat.
<b>ValidValues:</b>	
Bit	Description
[0]	1=The selected event input equals to the corresponding value in FCH::SMI::SciTrig[SciTrig0].
[1]	1=The selected event input equals to the corresponding value in FCH::SMI::SciTrig[SciTrig1].
[2]	1=The selected event input equals to the corresponding value in FCH::SMI::SciTrig[SciTrig2].
[3]	1=The selected event input equals to the corresponding value in FCH::SMI::SciTrig[SciTrig3].
[4]	1=The selected event input equals to the corresponding value in FCH::SMI::SciTrig[SciTrig4].
[5]	1=The selected event input equals to the corresponding value in FCH::SMI::SciTrig[SciTrig5].
[6]	1=The selected event input equals to the corresponding value in FCH::SMI::SciTrig[SciTrig6].
[7]	1=The selected event input equals to the corresponding value in FCH::SMI::SciTrig[SciTrig7].
[8]	1=The selected event input equals to the corresponding value in FCH::SMI::SciTrig[SciTrig8].
[9]	1=The selected event input equals to the corresponding value in FCH::SMI::SciTrig[SciTrig9].
[10]	1=The selected event input equals to the corresponding value in FCH::SMI::SciTrig[SciTrig10].
[11]	1=The selected event input equals to the corresponding value in FCH::SMI::SciTrig[SciTrig11].
[12]	1=The selected event input equals to the corresponding value in FCH::SMI::SciTrig[SciTrig12].
[13]	1=The selected event input equals to the corresponding value in FCH::SMI::SciTrig[SciTrig13].
[14]	1=The selected event input equals to the corresponding value in FCH::SMI::SciTrig[SciTrig14].
[15]	1=The selected event input equals to the corresponding value in FCH::SMI::SciTrig[SciTrig15].
[16]	1=The selected event input equals to the corresponding value in FCH::SMI::SciTrig[SciTrig16].
[17]	1=The selected event input equals to the corresponding value in FCH::SMI::SciTrig[SciTrig17].
[18]	1=The selected event input equals to the corresponding value in FCH::SMI::SciTrig[SciTrig18].
[19]	1=The selected event input equals to the corresponding value in FCH::SMI::SciTrig[SciTrig19].
[20]	1=The selected event input equals to the corresponding value in FCH::SMI::SciTrig[SciTrig20].
[21]	1=The selected event input equals to the corresponding value in FCH::SMI::SciTrig[SciTrig21].
[22]	1=The selected event input equals to the corresponding value in FCH::SMI::SciTrig[SciTrig22].
[23]	1=The selected event input equals to the corresponding value in FCH::SMI::SciTrig[SciTrig23].
[24]	1=The selected event input equals to the corresponding value in FCH::SMI::SciTrig[SciTrig24].
[25]	1=The selected event input equals to the corresponding value in FCH::SMI::SciTrig[SciTrig25].
[26]	1=The selected event input equals to the corresponding value in FCH::SMI::SciTrig[SciTrig26].
[27]	1=The selected event input equals to the corresponding value in FCH::SMI::SciTrig[SciTrig27].



[28]	1=The selected event input equals to the corresponding value in FCH::SMI::SciTrig[SciTrig28].
[29]	1=The selected event input equals to the corresponding value in FCH::SMI::SciTrig[SciTrig29].
[30]	1=The selected event input equals to the corresponding value in FCH::SMI::SciTrig[SciTrig30].
[31]	1=The selected event input equals to the corresponding value in FCH::SMI::SciTrig[SciTrig31].

**PMx0818 [EventEnable] (FCH::PM::EventEnable)**

Read-write. Reset: 0000\_0000h.

\_aliasPM; PMx0818; PM=FED8\_0000h

Bits	Description																																																																		
31:0	<b>EventEnable.</b> Read-write. Reset: 0000_0000h. Each bit controls whether ACPI should generate wakeup and SCI interrupt. The enable bits are also mirrored in FCH::SMI::EventEn. <b>ValidValues:</b> <table> <tr> <th>Bit</th><th>Description</th></tr> <tr><td>[0]</td><td>1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig0].</td></tr> <tr><td>[1]</td><td>1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig1].</td></tr> <tr><td>[2]</td><td>1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig2].</td></tr> <tr><td>[3]</td><td>1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig3].</td></tr> <tr><td>[4]</td><td>1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig4].</td></tr> <tr><td>[5]</td><td>1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig5].</td></tr> <tr><td>[6]</td><td>1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig6].</td></tr> <tr><td>[7]</td><td>1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig7].</td></tr> <tr><td>[8]</td><td>1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig8].</td></tr> <tr><td>[9]</td><td>1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig9].</td></tr> <tr><td>[10]</td><td>1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig10].</td></tr> <tr><td>[11]</td><td>1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig11].</td></tr> <tr><td>[12]</td><td>1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig12].</td></tr> <tr><td>[13]</td><td>1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig13].</td></tr> <tr><td>[14]</td><td>1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig14].</td></tr> <tr><td>[15]</td><td>1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig15].</td></tr> <tr><td>[16]</td><td>1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig16].</td></tr> <tr><td>[17]</td><td>1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig17].</td></tr> <tr><td>[18]</td><td>1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig18].</td></tr> <tr><td>[19]</td><td>1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig19].</td></tr> <tr><td>[20]</td><td>1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig20].</td></tr> <tr><td>[21]</td><td>1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig21].</td></tr> <tr><td>[22]</td><td>1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig22].</td></tr> <tr><td>[23]</td><td>1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig23].</td></tr> <tr><td>[24]</td><td>1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig24].</td></tr> <tr><td>[25]</td><td>1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig25].</td></tr> <tr><td>[26]</td><td>1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig26].</td></tr> <tr><td>[27]</td><td>1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig27].</td></tr> <tr><td>[28]</td><td>1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig28].</td></tr> <tr><td>[29]</td><td>1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig29].</td></tr> <tr><td>[30]</td><td>1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig30].</td></tr> <tr><td>[31]</td><td>1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig31].</td></tr> </table>	Bit	Description	[0]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig0].	[1]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig1].	[2]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig2].	[3]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig3].	[4]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig4].	[5]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig5].	[6]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig6].	[7]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig7].	[8]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig8].	[9]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig9].	[10]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig10].	[11]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig11].	[12]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig12].	[13]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig13].	[14]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig14].	[15]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig15].	[16]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig16].	[17]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig17].	[18]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig18].	[19]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig19].	[20]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig20].	[21]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig21].	[22]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig22].	[23]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig23].	[24]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig24].	[25]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig25].	[26]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig26].	[27]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig27].	[28]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig28].	[29]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig29].	[30]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig30].	[31]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig31].
Bit	Description																																																																		
[0]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig0].																																																																		
[1]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig1].																																																																		
[2]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig2].																																																																		
[3]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig3].																																																																		
[4]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig4].																																																																		
[5]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig5].																																																																		
[6]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig6].																																																																		
[7]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig7].																																																																		
[8]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig8].																																																																		
[9]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig9].																																																																		
[10]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig10].																																																																		
[11]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig11].																																																																		
[12]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig12].																																																																		
[13]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig13].																																																																		
[14]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig14].																																																																		
[15]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig15].																																																																		
[16]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig16].																																																																		
[17]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig17].																																																																		
[18]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig18].																																																																		
[19]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig19].																																																																		
[20]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig20].																																																																		
[21]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig21].																																																																		
[22]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig22].																																																																		
[23]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig23].																																																																		
[24]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig24].																																																																		
[25]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig25].																																																																		
[26]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig26].																																																																		
[27]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig27].																																																																		
[28]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig28].																																																																		
[29]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig29].																																																																		
[30]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig30].																																																																		
[31]	1=Generate wakeup and SCI interrupt for the event defined in FCH::SMI::SciTrig[SciTrig31].																																																																		

**9.2.10.4.5 SmiCmdBlk**

The IO mapped base address of this register block is defined by FCH::PM::AcpiSmiCmd.

**PMx081C [SmiCmdPort] (FCH::PM::SmiCmdPort)**

Read-write. Reset: 00h.

\_aliasPM; PMx081C; PM=FED8\_0000h

Bits	Description
7:0	<b>SmiCmdPort.</b> Read-write. Reset: 00h. When SMI command port is enabled, a Write to this port generates SMI# (only IOW can generate SMI#, MEMW does not generate SMI#). A Read of this address will return the previously Written value but does not generate an SMI.

**PMx081D [SmiCmdStatus] (FCH::PM::SmiCmdStat)**

Read-write. Reset: 00h.

\_aliasPM; PMx081D; PM=FED8\_0000h

Bits	Description
7:0	<b>SmiCmdStatus.</b> Read-write. Reset: 00h. Used by BIOS and OS.

**9.2.11 GPIO Pin control registers****9.2.11.1 IOMUX Registers**

Note: When IOMUXx43, EGPI067\_SPI\_ROM\_REQ is selecting EGPI067, FCH::ITF::LPC::PCICtl[ExtRomSharingEn] = 0 must be set to disable External ROM sharing.

Table 80: IOMUX Function Table

IOMUX	Pin Name	GPI O#	GEVEN T#	reset value	Ove rrid e_0	Ove rrid e_1	IOM UX= =0	IOM UX= =1	IOM UX= =2	IOM UX= =3	Doma in	Type	Sci	Def ault
IOMUXx0	PWR_BTN_L_AGPIO0	0	21	0			PWR_BTN_L	GPIOxx			S5	AGPIO	YES	PU
IOMUXx1	RST_strapSYS_RESET_L_AGPIO1	1	19	0		RST_strap	SYS_RESE T_L	GPIOxx			S5	AGPIO	YES	PU
IOMUXx2	WAKE_L_AGPIO2	2	8	0			WAKE_L	GPIOxx			S5	AGPIO	YES	PU
IOMUXx3	AGPIO3	3	2	0			GPIOxx				S5	AGPIO	YES	PU
IOMUXx4	AGPIO4	4	4	0			GPIOxx				S5	AGPIO	YES	PD
IOMUXx5	AGPIO5_DEVSLP0	5	7	0			GPIOxx	DEVSLP0			S5	AGPIO	YES	PD
IOMUXx6	AGPIO6_DEVSLP1	6	10	0			GPIOxx	DEVSLP1			S5	AGPIO	YES	PD
IOMUXx7	AGPIO7_FCH_ACP_I2S_SDIN	7	11	0			GPIOxx	FCH_ACP_I2S_S DIN			S5	AGPIO	YES	PD
IOMUXx8	AGPIO8_FC	8	23	0			GPIO	FCH_			S5	AGPIO	YES	PD

	H_ACP_I2S_LRCLK						xx	ACP_I2S_LRCLK						
IOMUXx9	AGPIO9_SGPIOTAO MDIO1_SCL	9	22	0			GPIOxx	SGPIOTAO MDIO1_SCL			S5	AGPIO	YES	PD
IOMUXxA	S0A3_GPIO_AGPIOT0_S GPIO_CLK_MDIO0_SCL	10		0			GPIOxx	S0A3_GPIO SGPIOT0_SCL			S5	AGPIO	YES	PU
IOMUXxB	BLINK_AGPIO11	11		0			GPIOxx	BLINK			S5	AGPIO	YES	PU
IOMUXxC	AGPIO12_LB_L	12		0			LLB_L	GPIOxx			S5	AGPIO	YES	n/a
IOMUXxD	AGPIO13_USB_OC5_L	13		0			USB_OC5_L	GPIOxx			S5	AGPIO	YES	PU
IOMUXxE	AGPIO14_USB_OC4_L	14		0			USB_OC4_L	GPIOxx			S5	AGPIO	YES	PU
IOMUXx10	USB_OC0_L_AGPIO16	16	12	0			USB_OC0_L	GPIOxx			S5	AGPIO	YES	PU
IOMUXx11	USB_OC1_L_AGPIO17	17	13	0			USB_OC1_L	GPIOxx			S5	AGPIO	YES	PU
IOMUXx12	USB_OC2_L_AGPIO18	18	14	0			USB_OC2_L	GPIOxx			S5	AGPIO	YES	PU
IOMUXx13	SCL1_I2C3_SCL_AGPIO19	19		0			SCL1	I2C3_SCL	GPIOxx		S5	AGPIO	YES	n/a
IOMUXx14	SDA1_I2C3_SDA_AGPIO20	20		0			SDA1	I2C3_SDA	GPIOxx		S5	AGPIO	YES	n/a
IOMUXx15	LPC_PD_L_EMMC_CMD_AGPIO21	21	5	0			LPC_PD_L	EMMC_CMD	GPIOxx		S5	AGPIO	YES	n/a
IOMUXx16	LPC_PME_L_EMMC_PWR_CTRL_AGPIO22	22	3	0			LPC_PME_L	EMMC_PWR_CTRL	GPIOxx		S5	AGPIO	YES	PU
IOMUXx17	AGPIO23_ACPRES_SGPIOT_LOAD MDIO1_SDA	23	16	0			ACPRES	SGPIOT_LOAD MDIO1_SDA	GPIOxx		S5	AGPIO	YES	PU
IOMUXx18	USB_OC3_L_AGPIO24	24	15	0			USB_OC3_L	GPIOxx			S5	AGPIO	YES	PU

IOMUXx1 A	PCIE_RST_L _EGPIO26	26		0			PCIE _RST _L	GPIO xx			S5	EGPIO	NO	n/a
IOMUXx1 B	EGPIO27_PC IE_RST1_L	27		0			GPIO xx	PCIE _RST 1_L			S5	EGPIO	NO	PD
IOMUXx1 D	SPI_TPM_CS _L_AGPIO29 _USB_OC6_ L	29		0			SPI_T PM_ CS_L	GPIO xx	USB_ OC6_ L		S5	AGPIO	YES	PU
IOMUXx1 E	SPI_CS2_L_ ESPI_CS_L_ AGPIO30	30		0			SPI_ CS2_ L	ESPI_ CS_L	GPIO xx		S5	AGPIO	YES	PU
IOMUXx1 F	SPI_CS3_L_ AGPIO31	31		0			SPI_ CS3_ L	ESPI_ CS_L	GPIO xx		S5	AGPIO	YES	PU
IOMUXx2 0	LPC_RST_L _SD_WP_L_ AGPIO32	32		0			LPC_ RST_ L	EMM C_W P	GPIO xx		S5	AGPIO	YES	PD
IOMUXx2 8	AGPIO40_S GPIO_DATA IN_MDIO0_ SDA	40	20	0			GPIO xx	SGPI O_DA TAIN	MDI O0_S DA		S5	AGPIO	YES	n/a
IOMUXx2 A	EGPIO42	42		0			GPIO xx				S5	EGPIO	NO	PU
IOMUXx4 3	EGPIO67_SP I_ROM_REQ	67		0			SPI_ ROM _REQ	GPIO xx			S5	EGPIO	NO	PD
IOMUXx4 4	AGPIO68_E MMC_CD	68		0			GPIO xx	EMM C_CD			S0	AGPIO	YES	PD
IOMUXx4 5	AGPIO69	69		0			GPIO xx				S0	AGPIO	YES	PD
IOMUXx4 6	EGPIO70_E MMC_CLK	70		0			GPIO xx	EMM C_CL K	SD_C LK		S0	EGPIO	NO	PD
IOMUXx4 A	LPCCCLK0_E MMC_DATA 4_EGPIO74	74		0			LPCC LK0	EMM C_DA TA4	GPIO xx		S0	EGPIO	NO	n/a
IOMUXx4 B	LPCCCLK1_E MMC_DATA 6_EGPIO75	75		0			LPCC LK1	EMM C_DA TA6	GPIO xx		S0	EGPIO	NO	n/a
IOMUXx4 C	EGPIO76/SPI _ROM_GNT	76		0			SPI_ ROM _GNT	GPIO xx			S5	EGPIO	NO	PD
IOMUXx5 4	FANIN0_AG PIO84	84	18	0			FANI N0	GPIO xx			S0	AGPIO	YES	PU
IOMUXx5 5	FanOut_intB_ FANOUT0_A GPIO85	85		1		Fan Out _int B	FAN OUT0	GPIO xx			S0	AGPIO	YES	PU
IOMUXx5	LPC_SMI_L_	86	9	0			LPC_	GPIO			S0	AGPIO	YES	PU

6	AGPIO86						SMI_L	xx						
IOMUXx57	SERIRQ_EM MC_DATA7_ AGPIO87	87		0			SERIRQ	EMMC_DATA7	GPIOxx		S0	AGPIO	YES	PU
IOMUXx58	LPC_CLKRUN_L_EMMC_DATA5_AG PIO88	88		0			LPC_CLKRUN_L	EMMC_DATA5	GPIOxx		S0	AGPIO	YES	PU
IOMUXx59	GENINT1_L_PSP_INTR0_AG PIO89	89	0	0			GENINT1_L	PSP_INTR0	GPIOxx		S0	AGPIO	YES	PU
IOMUXx5A	GENINT2_L_PSP_INTR1_AG PIO90	90	1	0			GENINT2_L	PSP_INTR1	GPIOxx		S0	AGPIO	YES	PU
IOMUXx5B	SPKR_AGPI O91	91	6	1			SPKR	GPIOxx			S0	AGPIO	YES	PD
IOMUXx5C	CLK_REQ0_L_SATA_IS0_L_SATA_ZP 0_L_AGPI092	92		0			CLK_REQ0_L	SATA_IS0_L	SATA_ZP0_L	GPIOxx	S0	AGPIO	YES	PU
IOMUXx5F	SD0_CLK_E GPIO95	95		0			SD0_CLK	GPIOxx			S0	EGPIO	NO	PD
IOMUXx60	SD0_CMD_E GPIO96	96		0			SD0_CMD	GPIOxx			S0	EGPIO	NO	PU
IOMUXx61	SD0_DATA0_E GPIO97	97		0			SD0_DATA0	GPIOxx			S0	EGPIO	NO	PU
IOMUXx62	SD0_DATA1_E GPIO98	98		0			SD0_DATA1	GPIOxx			S0	EGPIO	NO	PU
IOMUXx63	SD0_DATA2_E GPIO99	99		0			SD0_DATA2	GPIOxx			S0	EGPIO	NO	PU
IOMUXx64	SD0_DATA3_E GPIO100	100		0			SD0_DATA3	GPIOxx			S0	EGPIO	NO	PU
IOMUXx68	LAD0_EMMC_DATA0_E GPIO104	104		0			LAD0	EMMC_DATA0	reserved	GPIOxx	S0	EGPIO	NO	PU
IOMUXx69	LAD1_EMMC_DATA1_E GPIO105	105		0			LAD1	EMMC_DATA1	reserved	GPIOxx	S0	EGPIO	NO	PU
IOMUXx6A	LAD2_EMMC_DATA2_E GPIO106	106		0			LAD2	EMMC_DATA2	reserved	GPIOxx	S0	EGPIO	NO	PU
IOMUXx6B	LAD3_EMMC_DATA3_E GPIO107	107		0			LAD3	EMMC_DATA3	reserved	GPIOxx	S0	EGPIO	NO	PU
IOMUXx6	ESPI_ALERT	10		0		ESP	LDR	ESPI_	GPIO		S0	EGPIO	NO	PD

C	_L_LDRQ0_ L_ESPI_ALE RT_D1_EGPI O108	8				I_A LE RT_ L	Q0_L	ALER T_D1	xx					
IOMUXx6 D	ROMTYPE_s trap_LFRAM E_L_EMMC _DS_EGPI01 09	10 9		0		RO MT YP E_s trap	LFRA ME_ L	EMM C_DS	GPIO xx		S0	EGPIO	NO	n/a
IOMUXx7 1	SCL0_I2C2_ SCL_EGPI0 113	11 3		0			SCL0	I2C2_ SCL	GPIO xx		S0	EGPIO	NO	n/a
IOMUXx7 2	SDA0_I2C2_ SDA_EGPI0 114	11 4		0			SDA0	I2C2_ SDA	GPIO xx		S0	EGPIO	NO	n/a
IOMUXx7 3	CLK_REQ1_ L_AGPIO115	11 5		0			CLK_ REQ1_ _L	GPIO xx			S0	AGPIO	YES	PU
IOMUXx7 4	CLK_REQ2_ L_AGPIO116	11 6		0			CLK_ REQ2_ _L	GPIO xx			S0	AGPIO	YES	PU
IOMUXx7 8	CLK_REQ5_ L_EGPI0120	12 0		0			CLK_ REQ5_ _L	GPIO xx			S0	EGPIO	NO	n/a
IOMUXx7 9	CLK_REQ6_ L_EGPI0121	12 1		0			CLK_ REQ6_ _L	GPIO xx			S0	EGPIO	NO	PD
IOMUXx8 1	ESPI_RESET_ L_KBRST_ L_AGPIO129	12 9	17	0		ESP I_R ES ET_ L	KBR ST_L	GPIO xx			S0	AGPIO	YES	PU
IOMUXx8 2	SATA_ACT_ L_AGPIO130	13 0		0			SATA_ ACT_ _L	GPIO xx			S0	AGPIO	YES	PU
IOMUXx8 3	CLK_REQ3_ L_SATA_IS1_ _L_SATA_ZP 1_L_EGPI01 31	13 1		0			CLK_ REQ3_ _L	SATA_ IS1_ L	SATA_ ZP1_ _L	GPIO xx	S0	EGPIO	NO	PU
IOMUXx8 4	CLK_REQ4_ L_OSCIN_E GPIO132	13 2		0			CLK_ REQ4_ _L	OSCI N	GPIO xx		S0	EGPIO	NO	PU
IOMUXx8 7	UART0_CTS_ L_UART2_ TXD_EGPI0 135	13 5		10			UAR T0_C TS_L	UAR T2_R XD	GPIO xx		S0	EGPIO	NO	PD
IOMUXx8 8	UART0_RX_ D_EGPI0136	13 6		0			UAR T0_R XD	GPIO xx			S0	EGPIO	NO	PD
IOMUXx8 9	UART0_RTS_ L_UART2_ _L	13 7		10			UAR T0_R	UAR T2_T	GPIO xx		S0	EGPIO	NO	PU

	RXD_EGPIO 137					TS_L	XD						
IOMUXx8 A	UART0_TXD _EGPIO138	13 8		1		UAR T0_T XD	GPIO xx			S0	EGPIO	NO	PU
IOMUXx8 B	UART0_INT R_AGPIO139	13 9		0		UAR T0_I NTR	GPIO xx			S0	AGPIO	YES	PD
IOMUXx8 C	UART1_CTS _L_UART3_ TXD_EGPIO 140	14 0		10		UAR T1_C TS_L	UAR T3_T XD	GPIO xx		S0	EGPIO	NO	PD
IOMUXx8 D	UART1_RX D_EGPIO141	14 1		0		UAR T1_R XD	GPIO xx			S0	EGPIO	NO	PD
IOMUXx8 E	UART1_RTS _L_UART3_ RXD_EGPIO 142	14 2		10		UAR T1_R TS_L	UAR T3_R XD	GPIO xx		S0	EGPIO	NO	PU
IOMUXx8 F	UART1_TXD _EGPIO143	14 3		1		UAR T1_T XD	GPIO xx			S0	EGPIO	NO	PU
IOMUXx9 0	UART1_INT R_AGPIO144	14 4		0		UAR T1_I NTR	GPIO xx			S0	AGPIO	YES	PD

The IOMUX register space is accessed through the AcpiMmio region. The registers range from FED8\_0000h+D00h to FED8\_0000h+DFFh. See FCH::PM::IsaControl[MmioEn].

- AGPIO: Advanced GPIO - this pin can be used for interrupt, wake, or GPIO.
- EGPIO: Enhanced GPIO - this pin can be used for GPIO.

NOTE: Pay close attention to following conditions:

- FCH::IOMUX::Gpio0[PWR\_BTN\_L\_AGPIO0] can only be used as PWR\_BTN\_L. Since ACPI 5.0 requires Power Button be claimed as GPIO, and BIOS need the GPIO number to program its debouncing time, thus AGPIO0 is assigned.
- FCH::IOMUX::Gpio74[LPCCLK0\_EMMC\_DATA4\_EGPIO74] is used to supply clock to internal LPC logic. If LPCCLK0 is disabled, any transaction to LPC bus will cause system to lock up. Sending transaction to SPI/eSPI is OK.

#### IOMUXx000 [PWR\_BTN\_L\_AGPIO0] (FCH::IOMUX::Gpio0)

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx000; IOMUX=FED8\_0D00h

Bits	Description
7:2	Reserved.
1:0	<b>PWR_BTN_L_AGPIO0.</b> Read-write. Reset: 0h. Multi-function IO pin function select for pin PWR_BTN_L. The default IO state is pull-up.
<b>ValidValues:</b>	
<b>Value</b>	<b>Description</b>
0h	PWR_BTN_L
1h	AGPIO0
3h-2h	Reserved.

#### IOMUXx001 [RST\_strap\_SYS\_RESET\_L\_AGPIO1] (FCH::IOMUX::Gpio1)



Read-write. Reset: 00h.	
_aliasHOST; IOMUXx001; IOMUX=FED8_0D00h	
Bits	Description
7:2	Reserved.
1:0	<b>RST_strap_SYS_RESET_L_AGPI01.</b> Read-write. Reset: 0h. Multi-function IO pin function select for pin SYS_RESET_L. The default IO state is pull-up.
<b>ValidValues:</b>	
Value	Description
0h	SYS_RESET_L
1h	AGPI01
3h-2h	Reserved.

#### IOMUXx002 [WAKE\_L\_AGPI02] (FCH::IOMUX::Gpio2)

Read-write. Reset: 00h.	
_aliasHOST; IOMUXx002; IOMUX=FED8_0D00h	
Bits	Description
7:2	Reserved.
1:0	<b>WAKE_L_AGPI02.</b> Read-write. Reset: 0h. Multi-function IO pin function select for pin WAKE_L. The default IO state is pull-up.
<b>ValidValues:</b>	
Value	Description
0h	WAKE_L
1h	AGPI02
3h-2h	Reserved.

#### IOMUXx003 [AGPI03] (FCH::IOMUX::Gpio3)

Read-write. Reset: 00h.	
_aliasHOST; IOMUXx003; IOMUX=FED8_0D00h	
Bits	Description
7:2	Reserved.
1:0	<b>AGPI03.</b> Read-write. Reset: 0h. Multi-function IO pin function select for pin AGPI03. The default IO state is pull-up.
<b>ValidValues:</b>	
Value	Description
0h	AGPI03
3h-1h	Reserved.

#### IOMUXx004 [AGPI04] (FCH::IOMUX::Gpio4)

Read-write. Reset: 00h.	
_aliasHOST; IOMUXx004; IOMUX=FED8_0D00h	
Bits	Description
7:2	Reserved.
1:0	<b>AGPI04.</b> Read-write. Reset: 0h. Multi-function IO pin function select for pin AGPI04. The default IO state is pull-down.
<b>ValidValues:</b>	
Value	Description
0h	AGPI04
3h-1h	Reserved.

#### IOMUXx005 [AGPI05\_DEVSLP0] (FCH::IOMUX::Gpio5)

Read-write. Reset: 00h.	
-------------------------	--

_aliasHOST; IOMUXx005; IOMUX=FED8_0D00h	
Bits	Description
7:2	Reserved.
1:0	<b>AGPIO5_DEVSLP0</b> . Read-write. Reset: 0h. Multi-function IO pin function select for pin AGPIO5. The default IO state is pull-down.
<b>ValidValues:</b>	
Value	Description
0h	AGPIO5
1h	DEVSLP0
3h-2h	Reserved.

#### IOMUXx006 [AGPIO6\_DEVSLP1] (FCH::IOMUX::Gpio6)

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx006; IOMUX=FED8\_0D00h

Bits	Description
7:2	Reserved.
1:0	<b>AGPIO6_DEVSLP1</b> . Read-write. Reset: 0h. Multi-function IO pin function select for pin AGPIO6. The default IO state is pull-down.
<b>ValidValues:</b>	
Value	Description
0h	AGPIO6
1h	DEVSLP1
3h-2h	Reserved.

#### IOMUXx007 [AGPIO7\_FCH\_ACP\_I2S\_SDIN] (FCH::IOMUX::Gpio7)

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx007; IOMUX=FED8\_0D00h

Bits	Description
7:2	Reserved.
1:0	<b>AGPIO7_FCH_ACP_I2S_SDIN</b> . Read-write. Reset: 0h. Multi-function IO pin function select for pin AGPIO7. The default IO state is pull-down.
<b>ValidValues:</b>	
Value	Description
0h	AGPIO7
1h	FCH_ACP_I2S_SDIN
3h-2h	Reserved.

#### IOMUXx008 [AGPIO8\_FCH\_ACP\_I2S\_LRCLK] (FCH::IOMUX::Gpio8)

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx008; IOMUX=FED8\_0D00h

Bits	Description
7:2	Reserved.
1:0	<b>AGPIO8_FCH_ACP_I2S_LRCLK</b> . Read-write. Reset: 0h. Multi-function IO pin function select for pin AGPIO8. The default IO state is pull-down.
<b>ValidValues:</b>	
Value	Description
0h	AGPIO8
1h	FCH_ACP_I2S_LRCLK
3h-2h	Reserved.

#### IOMUXx009 [AGPIO9\_SGPIO\_DATAOUT\_MDIO1\_SCL] (FCH::IOMUX::Gpio9)

Read-write. Reset: 00h.	
_aliasHOST; IOMUXx009; IOMUX=FED8_0D00h	
Bits	Description
7:2	Reserved.
1:0	<b>AGPIO9_SGPIO_DATAOUT_MDIO1_SCL</b> . Read-write. Reset: 0h. Multi-function IO pin function select for pin AGPIO9. The default IO state is pull-down.
<b>ValidValues:</b>	
Value	Description
0h	AGPIO9
1h	SGPIO_DATAOUT
2h	MDIO1_SCL
3h	Reserved.

**IOMUXx00A [S0A3\_GPIO\_AGPIO10\_SGPIO\_CLK\_MDIO0\_SCL] (FCH::IOMUX::Gpio10)**

Read-write. Reset: 00h.	
_aliasHOST; IOMUXx00A; IOMUX=FED8_0D00h	
Bits	Description
7:2	Reserved.
1:0	<b>S0A3_GPIO_AGPIO10_SGPIO_CLK_MDIO0_SCL</b> . Read-write. Reset: 0h. Multi-function IO pin function select for pin S0A3_GPIO.
<b>ValidValues:</b>	
Value	Description
0h	AGPIO10
1h	S0A3_GPIO
2h	SGPIO_CLK
3h	MDIO0_SCL

**IOMUXx00B [BLINK\_AGPIO11] (FCH::IOMUX::Gpio11)**

Read-write. Reset: 00h.	
_aliasHOST; IOMUXx00B; IOMUX=FED8_0D00h	
Bits	Description
7:2	Reserved.
1:0	<b>BLINK_AGPIO11</b> . Read-write. Reset: 0h. Multi-function IO pin function select for pin BLINK. The default IO state is pull-up.
<b>ValidValues:</b>	
Value	Description
0h	AGPIO11
1h	BLINK
3h-2h	Reserved.

**IOMUXx00C [AGPIO12\_LLB\_L] (FCH::IOMUX::Gpio12)**

Read-write. Reset: 00h.	
_aliasHOST; IOMUXx00C; IOMUX=FED8_0D00h	
Bits	Description
7:2	Reserved.
1:0	<b>PWRGD_OUT_AGPIO12</b> . Read-write. Reset: 0h. Multi-function IO pin function select for pin LLB_L.
<b>ValidValues:</b>	
Value	Description
0h	LLB_L
1h	AGPIO12

	3h-2h	Reserved.
--	-------	-----------

**IOMUXx00D [AGPIO13\_USB\_OC5\_L] (FCH::IOMUX::Gpio13)**

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx00D; IOMUX=FED8\_0D00h

Bits	Description
7:2	Reserved.
1:0	<b>AGPIO13_USB_OC5_L.</b> Read-write. Reset: 0h. Multi-function IO pin function select for pin USB_OC5_L.
<b>ValidValues:</b>	
Value	Description
0h	USB_OC5_L
1h	AGPIO13
3h-2h	Reserved.

**IOMUXx00E [AGPIO14\_USB\_OC4\_L] (FCH::IOMUX::Gpio14)**

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx00E; IOMUX=FED8\_0D00h

Bits	Description
7:2	Reserved.
1:0	<b>AGPIO14_USB_OC4_L.</b> Read-write. Reset: 0h. Multi-function IO pin function select for pin USB_OC4_L.
<b>ValidValues:</b>	
Value	Description
0h	USB_OC4_L
1h	AGPIO14
3h-2h	Reserved.

**IOMUXx010 [USB\_OC0\_L\_AGPIO16] (FCH::IOMUX::Gpio16)**

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx010; IOMUX=FED8\_0D00h

Bits	Description
7:2	Reserved.
1:0	<b>USB_OC0_L_AGPIO16.</b> Read-write. Reset: 0h. Multi-function IO pin function select for pin USB_OC0_L. The default IO state is pull-up.
<b>ValidValues:</b>	
Value	Description
0h	USB_OC0_L
1h	AGPIO16
3h-2h	Reserved.

**IOMUXx011 [USB\_OC1\_L\_AGPIO17] (FCH::IOMUX::Gpio17)**

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx011; IOMUX=FED8\_0D00h

Bits	Description
7:2	Reserved.
1:0	<b>USB_OC1_L_AGPIO17.</b> Read-write. Reset: 0h. Multi-function IO pin function select for pin USB_OC1_L. The default IO state is pull-up.
<b>ValidValues:</b>	
Value	Description
0h	USB_OC1_L
1h	AGPIO17
3h-2h	Reserved.

**IOMUXx012 [USB\_OC2\_L\_AGPIO18] (FCH::IOMUX::Gpio18)**

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx012; IOMUX=FED8\_0D00h

Bits	Description
7:2	Reserved.
1:0	<b>USB_OC2_L_AGPIO18.</b> Read-write. Reset: 0h. Multi-function IO pin function select for pin USB_OC2_L. The default IO state is pull-up.
<b>ValidValues:</b>	
Value	Description
0h	USB_OC2_L
1h	AGPIO18
3h-2h	Reserved.

**IOMUXx013 [SCL1\_I2C3\_SCL\_AGPIO19] (FCH::IOMUX::Gpio19)**

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx013; IOMUX=FED8\_0D00h

Bits	Description
7:2	Reserved.
1:0	<b>SCL1_I2C3_SCL_AGPIO19.</b> Read-write. Reset: 0h. Multi-function IO pin function select for pin SCL1.
<b>ValidValues:</b>	
Value	Description
0h	SCL1
1h	I2C3_SCL
2h	AGPIO19
3h	Reserved.

**IOMUXx014 [SDA1\_I2C3\_SDA\_AGPIO20] (FCH::IOMUX::Gpio20)**

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx014; IOMUX=FED8\_0D00h

Bits	Description
7:2	Reserved.
1:0	<b>SDA1_I2C3_SDA_AGPIO20.</b> Read-write. Reset: 0h. Multi-function IO pin function select for pin SDA1.
<b>ValidValues:</b>	
Value	Description
0h	SDA1
1h	I2C3_SDA
2h	AGPIO20
3h	Reserved.

**IOMUXx015 [LPC\_PD\_L\_EMMC\_CMD\_AGPIO21] (FCH::IOMUX::Gpio21)**

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx015; IOMUX=FED8\_0D00h

Bits	Description
7:2	Reserved.
1:0	<b>LPC_PD_L_EMMC_CMD_AGPIO21.</b> Read-write. Reset: 0h. Multi-function IO pin function select for pin LPC_PD_L.
<b>ValidValues:</b>	
Value	Description
0h	LPC_PD_L
1h	EMMC_CMD

2h	AGPIO21
3h	Reserved.

**IOMUXx016 [LPC\_PME\_L\_EMMC\_PWR\_CTRL\_AGPIO22] (FCH::IOMUX::Gpio22)**

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx016; IOMUX=FED8\_0D00h

Bits	Description										
7:2	Reserved.										
1:0	<b>LPC_PME_L_EMMC_PWR_CTRL_AGPIO22.</b> Read-write. Reset: 0h. Multi-function IO pin function select for pin LPC_PME_L. The default IO state is pull-up. <b>ValidValues:</b>										
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>LPC_PME_L</td></tr> <tr> <td>1h</td><td>EMMC_PWR_CTRL</td></tr> <tr> <td>2h</td><td>AGPIO22</td></tr> <tr> <td>3h</td><td>Reserved.</td></tr> </table>	Value	Description	0h	LPC_PME_L	1h	EMMC_PWR_CTRL	2h	AGPIO22	3h	Reserved.
Value	Description										
0h	LPC_PME_L										
1h	EMMC_PWR_CTRL										
2h	AGPIO22										
3h	Reserved.										

**IOMUXx017 [AGPIO23\_AC\_PRES\_SGPIO\_LOAD\_MDIO1\_SDA] (FCH::IOMUX::Gpio23)**

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx017; IOMUX=FED8\_0D00h

Bits	Description										
7:2	Reserved.										
1:0	<b>AGPIO23_AC_PRES_SGPIO_LOAD_MDIO1_SDA.</b> Read-write. Reset: 0h. Multi-function IO pin function select for pin AGPIO23. The default IO state is pull-up. <b>ValidValues:</b>										
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>AC_PRES</td></tr> <tr> <td>1h</td><td>SGPIO_LOAD</td></tr> <tr> <td>2h</td><td>MDIO1_SDA</td></tr> <tr> <td>3h</td><td>AGPIO23</td></tr> </table>	Value	Description	0h	AC_PRES	1h	SGPIO_LOAD	2h	MDIO1_SDA	3h	AGPIO23
Value	Description										
0h	AC_PRES										
1h	SGPIO_LOAD										
2h	MDIO1_SDA										
3h	AGPIO23										

**IOMUXx018 [USB\_OC3\_L\_AGPIO24] (FCH::IOMUX::Gpio24)**

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx018; IOMUX=FED8\_0D00h

Bits	Description								
7:2	Reserved.								
1:0	<b>USB_OC3_L_AGPIO24.</b> Read-write. Reset: 0h. Multi-function IO pin function select for pin USB_OC3_L. The default IO state is pull-up. <b>ValidValues:</b>								
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>USB_OC3_L</td></tr> <tr> <td>1h</td><td>AGPIO24</td></tr> <tr> <td>3h-2h</td><td>Reserved.</td></tr> </table>	Value	Description	0h	USB_OC3_L	1h	AGPIO24	3h-2h	Reserved.
Value	Description								
0h	USB_OC3_L								
1h	AGPIO24								
3h-2h	Reserved.								

**IOMUXx01A [PCIE\_RST\_L\_EGPIO26] (FCH::IOMUX::Gpio26)**

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx01A; IOMUX=FED8\_0D00h

Bits	Description
7:2	Reserved.
1:0	<b>PCIE_RST_L_EGPIO26.</b> Read-write. Reset: 0h. Multi-function IO pin function select for pin PCIE_RST_L. <b>ValidValues:</b>

Value	Description
0h	PCIE_RST_L
1h	EGPIO26
3h-2h	Reserved.

**IOMUXx01B [EGPIO27\_PCIE\_RST1\_L] (FCH::IOMUX::Gpio27)**

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx01B; IOMUX=FED8\_0D00h

Bits	Description								
7:2	Reserved.								
1:0	<b>X48M0_EGPIO27.</b> Read-write. Reset: 0h. Multi-function IO pin function select for pin PCIE_RST1_L. The default IO state is pull-down. <b>ValidValues:</b>								
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>EGPIO27</td></tr> <tr> <td>1h</td><td>PCIE_RST1_L</td></tr> <tr> <td>3h-2h</td><td>Reserved.</td></tr> </table>	Value	Description	0h	EGPIO27	1h	PCIE_RST1_L	3h-2h	Reserved.
Value	Description								
0h	EGPIO27								
1h	PCIE_RST1_L								
3h-2h	Reserved.								

**IOMUXx01D [SPI\_TPM\_CS\_L\_AGPIO29\_USB\_OC6\_L] (FCH::IOMUX::Gpio29)**

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx01D; IOMUX=FED8\_0D00h

Bits	Description										
7:2	Reserved.										
1:0	<b>SPI_TPM_CS_L_AGPIO29_USB_OC6_L.</b> Read-write. Reset: 0h. Multi-function IO pin function select for pin AGPIO29. The default IO state is pull-down. <b>ValidValues:</b>										
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>SPI_TPM_CS_L</td></tr> <tr> <td>1h</td><td>AGPIO29</td></tr> <tr> <td>2h</td><td>USB_OC6_L</td></tr> <tr> <td>3h</td><td>Reserved.</td></tr> </table>	Value	Description	0h	SPI_TPM_CS_L	1h	AGPIO29	2h	USB_OC6_L	3h	Reserved.
Value	Description										
0h	SPI_TPM_CS_L										
1h	AGPIO29										
2h	USB_OC6_L										
3h	Reserved.										

**IOMUXx01E [SPI\_CS2\_L\_ESPI\_CS\_L\_AGPIO30] (FCH::IOMUX::Gpio30)**

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx01E; IOMUX=FED8\_0D00h

Bits	Description										
7:2	Reserved.										
1:0	<b>SPI_CS2_L_ESPI_CS_L_AGPIO30.</b> Read-write. Reset: 0h. Multi-function IO pin function select for pin AGPIO30. The default IO state is pull-down. <b>ValidValues:</b>										
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>SPI_CS2_L</td></tr> <tr> <td>1h</td><td>ESPI_CS_L</td></tr> <tr> <td>2h</td><td>AGPIO30</td></tr> <tr> <td>3h</td><td>Reserved.</td></tr> </table>	Value	Description	0h	SPI_CS2_L	1h	ESPI_CS_L	2h	AGPIO30	3h	Reserved.
Value	Description										
0h	SPI_CS2_L										
1h	ESPI_CS_L										
2h	AGPIO30										
3h	Reserved.										

**IOMUXx01F [SPI\_CS3\_L\_AGPIO31] (FCH::IOMUX::Gpio31)**

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx01F; IOMUX=FED8\_0D00h

Bits	Description
7:2	Reserved.



1:0	<b>SPI_CS3_L_AGPIO31.</b> Read-write. Reset: 0h. Multi-function IO pin function select for pin AGPIO31. The default IO state is pull-down.
<b>ValidValues:</b>	
Value	Description
0h	SPI_CS3_L
1h	ESPI_CS_L
2h	AGPIO31
3h	Reserved.

**IOMUXx020 [LPC\_RST\_L\_SD\_WP\_L\_AGPIO32] (FCH::IOMUX::Gpio32)**

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx020; IOMUX=FED8\_0D00h

Bits	Description
7:2	Reserved.
1:0	<b>LPC_RST_L_SD_WP_L_AGPIO32.</b> Read-write. Reset: 0h. Multi-function IO pin function select for pin AGPIO32. The default IO state is pull-down.
<b>ValidValues:</b>	
Value	Description
0h	LPC_RST_L
1h	SD_WP_L
2h	AGPIO32
3h	Reserved.

**IOMUXx028 [AGPIO40\_SGPIO\_DATAIN\_MDIO0\_SDA] (FCH::IOMUX::Gpio40)**

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx028; IOMUX=FED8\_0D00h

Bits	Description
7:2	Reserved.
1:0	<b>AGPIO40_SGPIO_DATAIN_MDIO0_SDA.</b> Read-write. Reset: 0h. Multi-function IO pin function select for pin AGPIO40.
<b>ValidValues:</b>	
Value	Description
0h	AGPIO40
1h	SGPIO_DATAIN
2h	MDIO0_SDA
3h	Reserved.

**IOMUXx02A [EGPIO42] (FCH::IOMUX::Gpio42)**

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx02A; IOMUX=FED8\_0D00h

Bits	Description
7:2	Reserved.
1:0	<b>EGPIO42.</b> Read-write. Reset: 0h. Multi-function IO pin function select for pin EGPIO42.
<b>ValidValues:</b>	
Value	Description
0h	EGPIO42
3h-1h	Reserved.

**IOMUXx043 [EGPIO67\_SPI\_ROM\_REQ] (FCH::IOMUX::Gpio67)**

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx043; IOMUX=FED8\_0D00h

Bits	Description
7:2	Reserved.
1:0	<b>EGPIO67_SPI_ROM_REQ.</b> Read-write. Reset: 0h. Multi-function IO pin function select for pin EGPIO67. The default IO state is pull-down.
<b>ValidValues:</b>	
<b>Value</b>	<b>Description</b>
0h	SPI_ROM_REQ
1h	EGPIO67
3h-2h	Reserved.

**IOMUXx044 [AGPIO68\_EMMC\_CD] (FCH::IOMUX::Gpio68)**

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx044; IOMUX=FED8\_0D00h

Bits	Description
7:2	Reserved.
1:0	<b>AGPIO68_EMMC_CD.</b> Read-write. Reset: 0h. Multi-function IO pin function select for pin AGPIO68. The default IO state is pull-down.
<b>ValidValues:</b>	
<b>Value</b>	<b>Description</b>
0h	AGPIO68
1h	EMMC_CD
3h-2h	Reserved.

**IOMUXx045 [AGPIO69] (FCH::IOMUX::Gpio69)**

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx045; IOMUX=FED8\_0D00h

Bits	Description
7:2	Reserved.
1:0	<b>AGPIO69.</b> Read-write. Reset: 0h. Multi-function IO pin function select for pin AGPIO69. The default IO state is pull-down.
<b>ValidValues:</b>	
<b>Value</b>	<b>Description</b>
0h	AGPIO69
3h-1h	Reserved.

**IOMUXx046 [EGPIO70\_EMMC\_CLK] (FCH::IOMUX::Gpio70)**

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx046; IOMUX=FED8\_0D00h

Bits	Description
7:2	Reserved.
1:0	<b>EGPIO70_EMMC_CLK.</b> Read-write. Reset: 0h. Multi-function IO pin function select for pin EGPIO70. The default IO state is pull-down.
<b>ValidValues:</b>	
<b>Value</b>	<b>Description</b>
0h	EGPIO70
1h	EMMC_CLK
2h	SD_CLK
3h	Reserved.

**IOMUXx04A [LPCCLK0\_EMMC\_DATA4\_EGPIO74] (FCH::IOMUX::Gpio74)**

Read-write. Reset: 00h.

_aliasHOST; IOMUXx04A; IOMUX=FED8_0D00h	
Bits	Description
7:2	Reserved.
1:0	<b>LPCCLK0_EMMC_DATA4_EGPIO74.</b> Read-write. Reset: 0h. Multi-function IO pin function select for pin LPCCLK0.
<b>ValidValues:</b>	
Value	Description
0h	LPCCLK0
1h	EMMC_DATA4
2h	EGPIO74
3h	Reserved.

#### IOMUXx04B [LPCCLK1\_EMMC\_DATA6\_EGPIO75] (FCH::IOMUX::Gpio75)

Read-write. Reset: 00h.	
_aliasHOST; IOMUXx04B; IOMUX=FED8_0D00h	
Bits	Description
7:2	Reserved.
1:0	<b>LPCCLK1_EMMC_DATA6_EGPIO75.</b> Read-write. Reset: 0h. Multi-function IO pin function select for pin LPCCLK1.
<b>ValidValues:</b>	
Value	Description
0h	LPCCLK1
1h	EMMC_DATA6
2h	EGPIO75
3h	Reserved.

#### IOMUXx04C [EGPIO76\_SPI\_ROM\_GNT] (FCH::IOMUX::Gpio76)

Read-write. Reset: 01h.	
_aliasHOST; IOMUXx04C; IOMUX=FED8_0D00h	
Bits	Description
7:2	Reserved.
1:0	<b>EGPIO76_SPI_ROM_GNT.</b> Read-write. Reset: 1h. Multi-function IO pin function select for pin EGPIO76. The default IO state is pull-up.
<b>ValidValues:</b>	
Value	Description
0h	SPI_ROM_GNT
1h	EGPIO76
3h-2h	Reserved.

#### IOMUXx054 [FANIN0\_AGPIO84] (FCH::IOMUX::Gpio84)

Read-write. Reset: 00h.	
_aliasHOST; IOMUXx054; IOMUX=FED8_0D00h	
Bits	Description
7:2	Reserved.
1:0	<b>FANIN0_AGPIO84.</b> Read-write. Reset: 0h. Multi-function IO pin function select for pin FANIN0. The default IO state is pull-up.
<b>ValidValues:</b>	
Value	Description
0h	FANIN0
1h	AGPIO84
3h-2h	Reserved.

**IOMUXx055 [FanOut\_intB\_FANOUT0\_AGPIO85] (FCH::IOMUX::Gpio85)**

Read-write. Reset: 01h.

\_aliasHOST; IOMUXx055; IOMUX=FED8\_0D00h

Bits	Description
7:2	Reserved.
1:0	<b>FanOut_intB_FANOUT0_AGPIO85.</b> Read-write. Reset: 1h. Multi-function IO pin function select for pin FANOUT0. The default IO state is pull-up.
<b>ValidValues:</b>	
Value	Description
0h	FANOUT0
1h	AGPIO85
3h-2h	Reserved.

**IOMUXx056 [LPC\_SMI\_L\_AGPIO86] (FCH::IOMUX::Gpio86)**

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx056; IOMUX=FED8\_0D00h

Bits	Description
7:2	Reserved.
1:0	<b>LPC_SMI_L_AGPIO86.</b> Read-write. Reset: 0h. Multi-function IO pin function select for pin LPC_SMI_L. The default IO state is pull-up.
<b>ValidValues:</b>	
Value	Description
0h	LPC_SMI_L
1h	AGPIO86
3h-2h	Reserved.

**IOMUXx057 [SERIRQ\_EMMC\_DATA7\_AGPIO87] (FCH::IOMUX::Gpio87)**

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx057; IOMUX=FED8\_0D00h

Bits	Description
7:2	Reserved.
1:0	<b>SERIRQ_EMMC_DATA7_AGPIO87.</b> Read-write. Reset: 0h. Multi-function IO pin function select for pin SERIRQ. The default IO state is pull-up.
<b>ValidValues:</b>	
Value	Description
0h	SERIRQ
1h	EMMC_DATA7
2h	AGPIO87
3h	Reserved.

**IOMUXx058 [LPC\_CLKRUN\_L\_EMMC\_DATA5\_AGPIO88] (FCH::IOMUX::Gpio88)**

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx058; IOMUX=FED8\_0D00h

Bits	Description
7:2	Reserved.
1:0	<b>LPC_CLKRUN_L_EMMC_DATA5_AGPIO88.</b> Read-write. Reset: 0h. Multi-function IO pin function select for pin LPC_CLKRUN_L. The default IO state is pull-up.
<b>ValidValues:</b>	
Value	Description
0h	LPC_CLKRUN_L
1h	EMMC_DATA5

2h	AGPIO88
3h	Reserved.

**IOMUXx059 [GENINT1\_L\_PSP\_INTR0\_AGPIO89] (FCH::IOMUX::Gpio89)**

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx059; IOMUX=FED8\_0D00h

Bits	Description
7:2	Reserved.
1:0	<b>GENINT1_L_PSP_INTR0_AGPIO89.</b> Read-write. Reset: 0h. Multi-function IO pin function select for pin GENINT1_L. The default IO state is pull-up.
<b>ValidValues:</b>	
Value	Description
0h	GENINT1_L
1h	PSP_INTR0
2h	AGPIO89
3h	Reserved.

**IOMUXx05A [GENINT2\_L\_PSP\_INTR1\_AGPIO90] (FCH::IOMUX::Gpio90)**

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx05A; IOMUX=FED8\_0D00h

Bits	Description
7:2	Reserved.
1:0	<b>GENINT2_L_PSP_INTR1_AGPIO90.</b> Read-write. Reset: 0h. Multi-function IO pin function select for GENINT2_L. The default IO state is pull-up.
<b>ValidValues:</b>	
Value	Description
0h	GENINT2_L
1h	PSP_INTR1
2h	AGPIO90
3h	Reserved.

**IOMUXx05B [SPKR\_AGPIO91] (FCH::IOMUX::Gpio91)**

Read-write. Reset: 01h.

\_aliasHOST; IOMUXx05B; IOMUX=FED8\_0D00h

Bits	Description
7:2	Reserved.
1:0	<b>SPKR_AGPIO91.</b> Read-write. Reset: 1h. Multi-function IO pin function select for SPKR. The default IO state is pull-down.
<b>ValidValues:</b>	
Value	Description
0h	SPKR
1h	AGPIO91
3h-2h	Reserved.

**IOMUXx05C [CLK\_REQ0\_L\_SATA\_IS0\_L\_SATA\_ZP0\_L\_AGPIO92] (FCH::IOMUX::Gpio92)**

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx05C; IOMUX=FED8\_0D00h

Bits	Description
7:2	Reserved.
1:0	<b>CLK_REQ0_L_SATA_IS0_L_SATA_ZP0_L_AGPIO92.</b> Read-write. Reset: 0h. Multi-function IO pin function select for pin CLK_REQ0_L. The default IO state is pull-up.

	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	CLK_REQ0_L
	1h	SATA_IS0_L
	2h	SATA_ZP0_L
	3h	AGPIO92

#### IOMUXx05F [SD0\_CLK\_EGPIO95] (FCH::IOMUX::Gpio95)

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx05F; IOMUX=FED8\_0D00h

Bits	Description
7:2	Reserved.
1:0	<b>SD0_CLK_EGPIO95.</b> Read-write. Reset: 0h. Multi-function IO pin function select for pin SD0_CLK. The default IO state is pull-down.
<b>ValidValues:</b>	
<b>Value</b>	<b>Description</b>
0h	EGPIO95
1h	SD0_CLK
3h-2h	Reserved.

#### IOMUXx060 [SD0\_CMD\_EGPIO96] (FCH::IOMUX::Gpio96)

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx060; IOMUX=FED8\_0D00h

Bits	Description
7:2	Reserved.
1:0	<b>SD0_CMD_EGPIO96.</b> Read-write. Reset: 0h. Multi-function IO pin function select for pin SD0_CMD. The default IO state is pull-up.
<b>ValidValues:</b>	
<b>Value</b>	<b>Description</b>
0h	EGPIO96
1h	SD0_CMD
3h-2h	Reserved.

#### IOMUXx061 [SD0\_DATA0\_EGPIO97] (FCH::IOMUX::Gpio97)

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx061; IOMUX=FED8\_0D00h

Bits	Description
7:2	Reserved.
1:0	<b>SD0_DATA0_EGPIO97.</b> Read-write. Reset: 0h. Multi-function IO pin function select for pin SD0_DATA0. The default IO state is pull-up.
<b>ValidValues:</b>	
<b>Value</b>	<b>Description</b>
0h	EGPIO97
1h	SD0_DATA0
3h-2h	Reserved.

#### IOMUXx062 [SD0\_DATA1\_EGPIO98] (FCH::IOMUX::Gpio98)

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx062; IOMUX=FED8\_0D00h

Bits	Description
7:2	Reserved.

1:0	<b>SD0_DATA1_EGPIO98.</b> Read-write. Reset: 0h. Multi-function IO pin function select for pin SD0_DATA1. The default IO state is pull-up.
<b>ValidValues:</b>	
<b>Value</b>	<b>Description</b>
0h	EGPIO98
1h	SD0_DATA1
3h-2h	Reserved.

#### IOMUXx063 [SD0\_DATA2\_EGPIO99] (FCH::IOMUX::Gpio99)

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx063; IOMUX=FED8\_0D00h

<b>Bits</b>	<b>Description</b>
7:2	Reserved.
1:0	<b>SD0_DATA2_EGPIO99.</b> Read-write. Reset: 0h. Multi-function IO pin function select for pin SD0_DATA2. The default IO state is pull-up.
<b>ValidValues:</b>	
<b>Value</b>	<b>Description</b>
0h	EGPIO99
1h	SD0_DATA2
3h-2h	Reserved.

#### IOMUXx064 [SD0\_DATA3\_EGPIO100] (FCH::IOMUX::Gpio100)

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx064; IOMUX=FED8\_0D00h

<b>Bits</b>	<b>Description</b>
7:2	Reserved.
1:0	<b>SD0_DATA3_EGPIO100.</b> Read-write. Reset: 0h. Multi-function IO pin function select for pin SD0_DATA3. The default IO state is pull-up.
<b>ValidValues:</b>	
<b>Value</b>	<b>Description</b>
0h	EGPIO100
1h	SD0_DATA3
3h-2h	Reserved.

#### IOMUXx068 [LAD0\_EMMC\_DATA0\_EGPIO104] (FCH::IOMUX::Gpio104)

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx068; IOMUX=FED8\_0D00h

<b>Bits</b>	<b>Description</b>
7:2	Reserved.
1:0	<b>LAD0_EMMC_DATA0_EGPIO104.</b> Read-write. Reset: 0h. Multi-function IO pin function select for LAD0. The default IO state is pull-up.
<b>ValidValues:</b>	
<b>Value</b>	<b>Description</b>
0h	LAD0
1h	EMMC_DATA0
2h	Reserved.
3h	EGPIO104

#### IOMUXx069 [LAD1\_EMMC\_DATA1\_EGPIO105] (FCH::IOMUX::Gpio105)

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx069; IOMUX=FED8\_0D00h



Bits	Description										
7:2	Reserved.										
1:0	<b>LAD1_EMMC_DATA1_EGPIO105.</b> Read-write. Reset: 0h. Multi-function IO pin function select for LAD1. The default IO state is pull-up.										
	<b>ValidValues:</b>										
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>LAD1</td></tr> <tr> <td>1h</td><td>EMMC_DATA1</td></tr> <tr> <td>2h</td><td>Reserved.</td></tr> <tr> <td>3h</td><td>EGPIO105</td></tr> </table>	Value	Description	0h	LAD1	1h	EMMC_DATA1	2h	Reserved.	3h	EGPIO105
Value	Description										
0h	LAD1										
1h	EMMC_DATA1										
2h	Reserved.										
3h	EGPIO105										

**IOMUXx06A [LAD2\_EMMC\_DATA2\_EGPIO106] (FCH::IOMUX::Gpio106)**

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx06A; IOMUX=FED8\_0D00h

Bits	Description										
7:2	Reserved.										
1:0	<b>LAD2_EMMC_DATA2_EGPIO106.</b> Read-write. Reset: 0h. Multi-function IO pin function select for LAD2. The default IO state is pull-up.										
	<b>ValidValues:</b>										
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>LAD2</td></tr> <tr> <td>1h</td><td>EMMC_DATA2</td></tr> <tr> <td>2h</td><td>Reserved.</td></tr> <tr> <td>3h</td><td>EGPIO106</td></tr> </table>	Value	Description	0h	LAD2	1h	EMMC_DATA2	2h	Reserved.	3h	EGPIO106
Value	Description										
0h	LAD2										
1h	EMMC_DATA2										
2h	Reserved.										
3h	EGPIO106										

**IOMUXx06B [LAD3\_EMMC\_DATA3\_EGPIO107] (FCH::IOMUX::Gpio107)**

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx06B; IOMUX=FED8\_0D00h

Bits	Description										
7:2	Reserved.										
1:0	<b>LAD3_EMMC_DATA3_EGPIO107.</b> Read-write. Reset: 0h. Multi-function IO pin function select for LAD3. The default IO state is pull-up.										
	<b>ValidValues:</b>										
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>LAD3</td></tr> <tr> <td>1h</td><td>EMMC_DATA3</td></tr> <tr> <td>2h</td><td>Reserved.</td></tr> <tr> <td>3h</td><td>EGPIO107</td></tr> </table>	Value	Description	0h	LAD3	1h	EMMC_DATA3	2h	Reserved.	3h	EGPIO107
Value	Description										
0h	LAD3										
1h	EMMC_DATA3										
2h	Reserved.										
3h	EGPIO107										

**IOMUXx06C [ESPI\_ALERT\_L\_LDRQ0\_L\_ESPI\_ALERT\_D1\_EGPIO108] (FCH::IOMUX::Gpio108)**

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx06C; IOMUX=FED8\_0D00h

Bits	Description						
7:2	Reserved.						
1:0	<b>ESPI_ALERT_L_LDRQ0_L_ESPI_ALERT_D1_EGPIO108.</b> Read-write. Reset: 0h. Multi-function IO pin function select for LDRQ0_L. The default IO state is pull-down.						
	<b>ValidValues:</b>						
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>LDRQ0_L</td></tr> <tr> <td>1h</td><td>ESPI_ALERT_D1</td></tr> </table>	Value	Description	0h	LDRQ0_L	1h	ESPI_ALERT_D1
Value	Description						
0h	LDRQ0_L						
1h	ESPI_ALERT_D1						

2h	EGPIO108
3h	Reserved.

**IOMUXx06D [ROMTYPE\_strap\_LFRAME\_L\_EMMC\_DS\_EGPIO109] (FCH::IOMUX::Gpio109)**

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx06D; IOMUX=FED8\_0D00h

Bits	Description
7:2	Reserved.
1:0	<b>ROMTYPE_strap_LFRAME_L_EMMC_DS_EGPIO109.</b> Read-write. Reset: 0h. Multi-function IO pin function select for LFRAME_L.
<b>ValidValues:</b>	
Value	Description
0h	LFRAME_L
1h	EMMC_DS
2h	EGPIO109
3h	Reserved.

**IOMUXx071 [SCL0\_I2C2\_SCL\_EGPIO113] (FCH::IOMUX::Gpio113)**

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx071; IOMUX=FED8\_0D00h

Bits	Description
7:2	Reserved.
1:0	<b>SCL0_I2C2_SCL_EGPIO113.</b> Read-write. Reset: 0h. Multi-function IO pin function select for SCL0.
<b>ValidValues:</b>	
Value	Description
0h	SCL0
1h	I2C2_SCL
2h	EGPIO113
3h	Reserved.

**IOMUXx072 [SDA0\_I2C2\_SDA\_EGPIO114] (FCH::IOMUX::Gpio114)**

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx072; IOMUX=FED8\_0D00h

Bits	Description
7:2	Reserved.
1:0	<b>SDA0_I2C2_SDA_EGPIO114.</b> Read-write. Reset: 0h. Multi-function IO pin function select for SDA0.
<b>ValidValues:</b>	
Value	Description
0h	SDA0
1h	I2C2_SDA
2h	EGPIO114
3h	Reserved.

**IOMUXx073 [CLK\_REQ1\_L\_AGPIO115] (FCH::IOMUX::Gpio115)**

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx073; IOMUX=FED8\_0D00h

Bits	Description
7:2	Reserved.
1:0	<b>CLK_REQ1_L_AGPIO115.</b> Read-write. Reset: 0h. Multi-function IO pin function select for CLK_REQ1_L. The default IO state is pull-up.
<b>ValidValues:</b>	

Value	Description
0h	CLK_REQ1_L
1h	AGPIO115
3h-2h	Reserved.

**IOMUXx074 [CLK\_REQ2\_L\_AGPIO116] (FCH::IOMUX::Gpio116)**

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx074; IOMUX=FED8\_0D00h

Bits	Description								
7:2	Reserved.								
1:0	<b>CLK_REQ2_L_AGPIO116.</b> Read-write. Reset: 0h. Multi-function IO pin function select for CLK_REQ2_L. The default IO state is pull-up. <b>ValidValues:</b>								
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>CLK_REQ2_L</td></tr> <tr> <td>1h</td><td>AGPIO116</td></tr> <tr> <td>3h-2h</td><td>Reserved.</td></tr> </table>	Value	Description	0h	CLK_REQ2_L	1h	AGPIO116	3h-2h	Reserved.
Value	Description								
0h	CLK_REQ2_L								
1h	AGPIO116								
3h-2h	Reserved.								

**IOMUXx078 [CLK\_REQ5\_L\_EGPIO120] (FCH::IOMUX::Gpio120)**

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx078; IOMUX=FED8\_0D00h

Bits	Description								
7:2	Reserved.								
1:0	<b>CLK_REQ5_L_EGPIO120.</b> Read-write. Reset: 0h. Multi-function IO pin function select for pin CLK_REQ5_L. The default IO state is pull-down. <b>ValidValues:</b>								
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>CLK_REQ5_L</td></tr> <tr> <td>1h</td><td>EGPIO120</td></tr> <tr> <td>3h-2h</td><td>Reserved.</td></tr> </table>	Value	Description	0h	CLK_REQ5_L	1h	EGPIO120	3h-2h	Reserved.
Value	Description								
0h	CLK_REQ5_L								
1h	EGPIO120								
3h-2h	Reserved.								

**IOMUXx079 [CLK\_REQ6\_L\_EGPIO121] (FCH::IOMUX::Gpio121)**

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx079; IOMUX=FED8\_0D00h

Bits	Description								
7:2	Reserved.								
1:0	<b>CLK_REQ6_L_EGPIO121.</b> Read-write. Reset: 0h. Multi-function IO pin function select for pin EGPIO121. The default IO state is pull-down. <b>ValidValues:</b>								
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>CLK_REQ6_L</td></tr> <tr> <td>1h</td><td>EGPIO121</td></tr> <tr> <td>3h-2h</td><td>Reserved.</td></tr> </table>	Value	Description	0h	CLK_REQ6_L	1h	EGPIO121	3h-2h	Reserved.
Value	Description								
0h	CLK_REQ6_L								
1h	EGPIO121								
3h-2h	Reserved.								

**IOMUXx081 [ESPI\_RESET\_L\_KBRST\_L\_AGPIO129] (FCH::IOMUX::Gpio129)**

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx081; IOMUX=FED8\_0D00h

Bits	Description
7:2	Reserved.
1:0	<b>ESPI_RESET_L_KBRST_L_AGPIO129.</b> Read-write. Reset: 0h. Multi-function IO pin function select for pin KBRST_L. The default IO state is pull-up.

	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	KBRST_L
	1h	Reserved.
	2h	AGPIO129
	3h	Reserved.

**IOMUXx082 [SATA\_ACT\_L\_AGPIO130] (FCH::IOMUX::Gpio130)**

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx082; IOMUX=FED8\_0D00h

Bits	Description
7:2	Reserved.
1:0	<b>SATA_ACT_L_AGPIO130.</b> Read-write. Reset: 0h. Multi-function IO pin function select for pin SATA_ACT_L. The default IO state is pull-up.
<b>ValidValues:</b>	
	<b>Value</b> <b>Description</b>
	0h   SATA_ACT_L
	1h   AGPIO130
	3h-2h   Reserved.

**IOMUXx083 [CLK\_REQ3\_L\_SATA\_IS1\_L\_SATA\_ZP1\_L\_EGPIO131] (FCH::IOMUX::Gpio131)**

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx083; IOMUX=FED8\_0D00h

Bits	Description
7:2	Reserved.
1:0	<b>CLK_REQ3_L_SATA_IS1_L_SATA_ZP1_L_EGPIO131.</b> Read-write. Reset: 0h. Multi-function IO pin function select for pin CLK_REQ3_L. The default IO state is pull-up.
<b>ValidValues:</b>	
	<b>Value</b> <b>Description</b>
	0h   CLK_REQ3_L
	1h   SATA_IS1_L
	2h   SATA_ZP1_L
	3h   EGPIO131

**IOMUXx084 [CLK\_REQG\_L\_OSCIN\_EGPIO132] (FCH::IOMUX::Gpio132)**

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx084; IOMUX=FED8\_0D00h

Bits	Description
7:2	Reserved.
1:0	<b>CLK_REQG_L_OSCIN_EGPIO132.</b> Read-write. Reset: 0h. Multi-function IO pin function select for pin CLK_REQG_L. The default IO state is pull-up.
<b>ValidValues:</b>	
	<b>Value</b> <b>Description</b>
	0h   CLK_REQG_L
	1h   OSCIN
	2h   EGPIO132
	3h   Reserved.

**IOMUXx087 [UART0\_CTS\_L\_UART2\_TXD\_EGPIO135] (FCH::IOMUX::Gpio135)**

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx087; IOMUX=FED8\_0D00h

--	--

Bits	Description										
7:2	Reserved.										
1:0	<b>UART0_CTS_L_UART2_TXD_EGPIO135.</b> Read-write. Reset: 0h. Multi-function IO pin function select for pin UART0_CTS_L. The default IO state is pull-down. <b>ValidValues:</b>										
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>UART0_CTS_L</td></tr> <tr> <td>1h</td><td>UART2_TXD</td></tr> <tr> <td>2h</td><td>EGPIO135</td></tr> <tr> <td>3h</td><td>Reserved.</td></tr> </table>	Value	Description	0h	UART0_CTS_L	1h	UART2_TXD	2h	EGPIO135	3h	Reserved.
Value	Description										
0h	UART0_CTS_L										
1h	UART2_TXD										
2h	EGPIO135										
3h	Reserved.										

**IOMUXx088 [UART0\_RXD\_EGPIO136] (FCH::IOMUX::Gpio136)**

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx088; IOMUX=FED8\_0D00h

Bits	Description								
7:2	Reserved.								
1:0	<b>UART0_RXD_EGPIO136.</b> Read-write. Reset: 0h. Multi-function IO pin function select for pin UART0_RXD. The default IO state is pull-down. <b>ValidValues:</b>								
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>UART0_RXD</td></tr> <tr> <td>1h</td><td>EGPIO136</td></tr> <tr> <td>3h-2h</td><td>Reserved.</td></tr> </table>	Value	Description	0h	UART0_RXD	1h	EGPIO136	3h-2h	Reserved.
Value	Description								
0h	UART0_RXD								
1h	EGPIO136								
3h-2h	Reserved.								

**IOMUXx089 [UART0\_RTS\_L\_UART2\_RXD\_EGPIO137] (FCH::IOMUX::Gpio137)**

Read-write. Reset: 01h.

\_aliasHOST; IOMUXx089; IOMUX=FED8\_0D00h

Bits	Description										
7:2	Reserved.										
1:0	<b>UART0_RTS_L_UART2_RXD_EGPIO137.</b> Read-write. Reset: 1h. Multi-function IO pin function select for pin UART0_RTS_L. The default IO state is pull-up. <b>ValidValues:</b>										
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>UART0_RTS_L</td></tr> <tr> <td>1h</td><td>UART2_RXD</td></tr> <tr> <td>2h</td><td>EGPIO137</td></tr> <tr> <td>3h</td><td>Reserved.</td></tr> </table>	Value	Description	0h	UART0_RTS_L	1h	UART2_RXD	2h	EGPIO137	3h	Reserved.
Value	Description										
0h	UART0_RTS_L										
1h	UART2_RXD										
2h	EGPIO137										
3h	Reserved.										

**IOMUXx08A [UART0\_TXD\_EGPIO138] (FCH::IOMUX::Gpio138)**

Read-write. Reset: 01h.

\_aliasHOST; IOMUXx08A; IOMUX=FED8\_0D00h

Bits	Description								
7:2	Reserved.								
1:0	<b>UART0_TXD_EGPIO138.</b> Read-write. Reset: 1h. Multi-function IO pin function select for pin UART0_TXD. The default IO state is pull-up. <b>ValidValues:</b>								
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>UART0_TXD</td></tr> <tr> <td>1h</td><td>EGPIO138</td></tr> <tr> <td>3h-2h</td><td>Reserved.</td></tr> </table>	Value	Description	0h	UART0_TXD	1h	EGPIO138	3h-2h	Reserved.
Value	Description								
0h	UART0_TXD								
1h	EGPIO138								
3h-2h	Reserved.								

**IOMUXx08B [UART0\_INTR\_AGPIO139] (FCH::IOMUX::Gpio139)**

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx08B; IOMUX=FED8\_0D00h

Bits	Description
7:2	Reserved.
1:0	<b>UART0_INTR_AGPIO139.</b> Read-write. Reset: 0h. Multi-function IO pin function select for pin UART0_INTR. The default IO state is pull-down.
<b>ValidValues:</b>	
Value	Description
0h	UART0_INTR
1h	AGPIO139
3h-2h	Reserved.

**IOMUXx08C [UART1\_CTS\_L\_UART3\_TXD\_EGPIO140] (FCH::IOMUX::Gpio140)**

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx08C; IOMUX=FED8\_0D00h

Bits	Description
7:2	Reserved.
1:0	<b>UART1_CTS_L_UART3_TXD_EGPIO140.</b> Read-write. Reset: 0h. Multi-function IO pin function select for pin UART1_CTS_L. The default IO state is pull-down.
<b>ValidValues:</b>	
Value	Description
0h	UART1_CTS_L
1h	UART3_TXD
2h	EGPIO140
3h	Reserved.

**IOMUXx08D [UART1\_RXD\_EGPIO141] (FCH::IOMUX::Gpio141)**

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx08D; IOMUX=FED8\_0D00h

Bits	Description
7:2	Reserved.
1:0	<b>UART1_RXD_EGPIO141.</b> Read-write. Reset: 0h. Multi-function IO pin function select for pin UART1_RXD. The default IO state is pull-down.
<b>ValidValues:</b>	
Value	Description
0h	UART1_RXD
1h	EGPIO141
3h-2h	Reserved.

**IOMUXx08E [UART1\_RTS\_L\_UART3\_RXD\_EGPIO142] (FCH::IOMUX::Gpio142)**

Read-write. Reset: 01h.

\_aliasHOST; IOMUXx08E; IOMUX=FED8\_0D00h

Bits	Description
7:2	Reserved.
1:0	<b>UART1_RTS_L_UART3_RXD_EGPIO142.</b> Read-write. Reset: 1h. Multi-function IO pin function select for pin UART1_RTS_L. The default IO state is pull-up.
<b>ValidValues:</b>	
Value	Description
0h	UART1_RTS_L
1h	UART3_RXD

2h	EGPIO142
3h	Reserved.

**IOMUXx08F [UART1\_TXD\_EGPIO143] (FCH::IOMUX::Gpio143)**

Read-write. Reset: 01h.

\_aliasHOST; IOMUXx08F; IOMUX=FED8\_0D00h

Bits	Description
7:2	Reserved.
1:0	<b>UART1_TXD_EGPIO143.</b> Read-write. Reset: 1h. Multi-function IO pin function select for pin UART1_TXD. The default IO state is pull-up.
<b>ValidValues:</b>	
Value	Description
0h	UART1_TXD
1h	EGPIO143
3h-2h	Reserved.

**IOMUXx090 [UART1\_INTR\_AGPIO144] (FCH::IOMUX::Gpio144)**

Read-write. Reset: 00h.

\_aliasHOST; IOMUXx090; IOMUX=FED8\_0D00h

Bits	Description
7:2	Reserved.
1:0	<b>UART1_INTR_AGPIO144.</b> Read-write. Reset: 0h. Multi-function IO pin function select for pin UART1_INTR. The default IO state is pull-down.
<b>ValidValues:</b>	
Value	Description
0h	UART1_INTR
1h	AGPIO144
3h-2h	Reserved.

**9.2.11.2 GPIO Registers**

The GPIO pins are controlled by a combination of device enables and by their specific GPIO and IOMUX register pair.

GPIO registers are accessed through memory-mapped ACPIMMIO region. The offset is relative to FED8\_0000h+1500h. GPIO bank 0 registers range from FED8\_0000h+1500h to FED8\_0000h+15FFh. GPIO Bank 1 registers range from FED8\_0000h+1600h to FED8\_0000h+16FFh. GPIO Bank 2 registers range from FED8\_0000h+1700h to FED8\_0000h+17FFh. GPIO Bank 3 registers range from FED8\_0000h+1800h to FED8\_0000h+18FFh.

*Table 81: I2C Pad Configuration Method*

	PAD name	GPIO register(bank, offset) (Not used)	Misc_Reg(Control I2C PAD)
I2C2	SCL0_I2C2_SCL_EGPIO113	0x1C4	0xE0
	SDA0_I2C2_SDA_EGPIO114	0x1C8	
I2C3	SCL1_I2C3_SCL_AGPIO19	0x04C	0xE4
	SDA1_I2C3_SDA_AGPIO20	0x050	

*Table 82: Reset Value for GPIO BANK0*



Register	Reset Value	Index	Pin Name
GPIOx00000	0014_0000h	0	PWR_BTN_L_AGPI00
GPIOx00004	0014_0000h	1	RST_strap_SYS_RESET_L_AGPI01
GPIOx00008	0014_0000h	2	WAKE_L_AGPI02
GPIOx0000C	0014_0000h	3	AGPI03
GPIOx00010	0024_0000h	4	AGPI04
GPIOx00014	0024_0000h	5	AGPI05_DEVSLP0
GPIOx00018	0024_0000h	6	AGPI06_DEVSLP1
GPIOx0001C	0024_0000h	7	AGPI07_FCH_ACP_I2S_SDIN
GPIOx00020	0024_0000h	8	AGPI08_FCH_ACP_I2S_LRCLK
GPIOx00024	0024_0000h	9	AGPI09_SGPIO_DATAOUT_MDIO1_SCL
GPIOx00028	0004_0000h	10	S0A3_GPIO_AGPI010_SGPIO_CLK_MDIO0_SCL
GPIOx0002C	0014_0000h	11	BLINK_AGPI011
GPIOx00030	0004_0000h	12	AGPI012_LLB_L
GPIOx00034	0014_0000h	13	AGPI013_USB_OC5_L
GPIOx00038	0014_0000h	14	AGPI014_USB_OC4_L
GPIOx0003C	0000_0000h	15	
GPIOx00040	0014_0000h	16	USB_OC0_L_AGPI016
GPIOx00044	0014_0000h	17	USB_OC1_L_AGPI017
GPIOx00048	0014_0000h	18	USB_OC2_L_AGPI018
GPIOx0004C	0004_0000h	19	SCL1_I2C3_SCL_AGPI019
GPIOx00050	0004_0000h	20	SDA1_I2C3_SDA_AGPI020
GPIOx00054	0004_0000h	21	LPC_PD_L_EMMC_CMD_AGPI021
GPIOx00058	0014_0000h	22	LPC_PME_L_EMMC_PWR_CTRL_AGPI022
GPIOx0005C	0014_0000h	23	AGPI023_SGPIO_LOAD_MDIO1_SDA
GPIOx00060	0014_0000h	24	USB_OC3_L_AGPI024
GPIOx00064	0000_0000h	25	
GPIOx00068	0004_0000h	26	PCIE_RST_L_EGPI026
GPIOx0006C	0024_0000h	27	EGPI027_PCIE_RST1_L
GPIOx00070	0000_0000h	28	
GPIOx00074	0014_0000h	29	X48M2_EGPI029
GPIOx00078	0014_0000h	30	SPI_CS2_L_ESPI_CS_L_AGPI030
GPIOx0007C	0014_0000h	31	SPI_CS3_L_AGPI031
GPIOx00080	0024_0000h	32	LPC_RST_L_SD_WP_L_AGPI032
GPIOx00084	0000_0000h	33	
GPIOx00088	0000_0000h	34	
GPIOx0008C	0000_0000h	35	
GPIOx00090	0000_0000h	36	
GPIOx00094	0000_0000h	37	
GPIOx00098	0000_0000h	38	
GPIOx0009C	0000_0000h	39	
GPIOx000A0	0004_0000h	40	AGPI040_SGPIO_DATAIN_MDIO0_SDA
GPIOx000A4	0000_0000h	41	
GPIOx000A8	0014_0000h	42	EGPI042
GPIOx000AC	0000_0000h	43	
GPIOx000B0	0000_0000h	44	ShdwSysAlarmFire
GPIOx000B4	0000_0000h	45	Pwr Button
GPIOx000B8	0000_0000h	46	

GPIOx000BC	0000_0000h	47	Int_iEcSci
GPIOx000C0	0000_0000h	48	Int_iCIR_Wake
GPIOx000C4	0000_0000h	49	Int_ASFSlaveIntr
GPIOx000C8	0000_0000h	50	Int_~ec_sm_irq_
GPIOx000CC	0000_0000h	51	Int_WakeFromLLB
GPIOx000D0	0000_0000h	52	Int_AcDcTimerEvent
GPIOx000D4	0000_0000h	53	Int_ALTHPET_TimerSts
GPIOx000D8	0000_0000h	54	
GPIOx000DC	0000_0000h	55	
GPIOx000E0	0000_0000h	56	
GPIOx000E4	0000_0000h	57	
GPIOx000E8	0000_0000h	58	Int_usb_xhc_0_acpi_pme
GPIOx000EC	0000_0000h	59	Int_usb_xhc_1_acpi_pme
GPIOx000F0	0000_0000h	60	
GPIOx000F4	0000_0000h	61	Int_ACP_FCH_AZ_Wake
GPIOx000F8	0000_0000h	62	Int_ACP_FCH_I2S_Wake
GPIOx000FC	0014_0000h	63	PWR_BTN_L_AGPI00

Table 83: Reset Value for GPIO BANK1

Register	Reset Value	Index	Pin Name
GPIOx00100	0000_0000h	64	
GPIOx00104	0000_0000h	65	
GPIOx00108	0000_0000h	66	
GPIOx0010C	0024_0000h	67	EGPIO67_SPI_ROM_REQ
GPIOx00110	0024_0000h	68	AGPIO68_EMMC_CD
GPIOx00114	0024_0000h	69	AGPIO69
GPIOx00118	0024_0000h	70	EGPIO70_EMMC_CLK
GPIOx0011C	0000_0000h	71	
GPIOx00120	0000_0000h	72	
GPIOx00124	0000_0000h	73	
GPIOx00128	0004_0000h	74	LPCCLK0_EMMC_DATA4_EGPIO74
GPIOx0012C	0004_0000h	75	LPCCLK1_EMMC_DATA6_EGPIO75
GPIOx00130	0024_0000h	76	EGPIO76_SPI_ROM_GNT
GPIOx00134	0000_0000h	77	
GPIOx00138	0000_0000h	78	
GPIOx0013C	0000_0000h	79	
GPIOx00140	0000_0000h	80	
GPIOx00144	0000_0000h	81	
GPIOx00148	0000_0000h	82	
GPIOx0014C	0000_0000h	83	
GPIOx00150	0014_0000h	84	FANIN0_AGPI084
GPIOx00154	0014_0000h	85	FANOUT0_AGPI085
GPIOx00158	0014_0000h	86	LPC_SMI_L_AGPI086
GPIOx0015C	0014_0000h	87	SERIRQ_EMMC_DATA7_AGPI087
GPIOx00160	0014_0000h	88	LPC_CLKRUN_L_EMMC_DATA5_AGPI088
GPIOx00164	0014_0000h	89	GENINT1_L_PSP_INTR0_AGPI089
GPIOx00168	0014_0000h	90	GENINT2_L_PSP_INTR1_AGPI090

GPIOx0016C	0024_0000h	91	SPKR_AGPI091
GPIOx00170	0014_0000h	92	CLK_REQ0_L_SATA_IS0_L_SATA_ZP0_L_AGPI092
GPIOx00174	0000_0000h	93	
GPIOx00178	0000_0000h	94	
GPIOx0017C	0024_0000h	95	SD0_CLK_EGPI095
GPIOx00180	0014_0000h	96	SD0_CMD_EGPI096
GPIOx00184	0014_0000h	97	SD0_DATA0_EGPI097
GPIOx00188	0014_0000h	98	SD0_DATA1_EGPI098
GPIOx0018C	0014_0000h	99	SD0_DATA2_EGPI099
GPIOx00190	0014_0000h	100	SD0_DATA3_EGPI100
GPIOx00194	0000_0000h	101	
GPIOx00198	0000_0000h	102	
GPIOx0019C	0000_0000h	103	
GPIOx001A0	0014_0000h	104	LAD0_EMMC_DATA0_EGPI104
GPIOx001A4	0014_0000h	105	LAD1_EMMC_DATA1_EGPI105
GPIOx001A8	0014_0000h	106	LAD2_EMMC_DATA2_EGPI106
GPIOx001AC	0014_0000h	107	LAD3_EMMC_DATA3_EGPI107
GPIOx001B0	0024_0000h	108	ESPI_ALERT_L_LDRQ0_L_ESPI_ALERT_D1_EGPI108
GPIOx001B4	0004_0000h	109	ROMTYPE_strap_LFRAME_L_EMMC_DS_EGPI109
GPIOx001B8	0000_0000h	110	
GPIOx001BC	0000_0000h	111	
GPIOx001C0	0000_0000h	112	
GPIOx001C4	0004_0000h	113	SCL0_I2C2_SCL_EGPI113
GPIOx001C8	0004_0000h	114	SDA0_I2C2_SDA_EGPI114
GPIOx001CC	0014_0000h	115	CLK_REQ1_L_AGPI115
GPIOx001D0	0014_0000h	116	CLK_REQ2_L_AGPI11
GPIOx001D4	0000_0000h	117	
GPIOx001D8	0000_0000h	118	
GPIOx001DC	0000_0000h	119	
GPIOx001E0	0004_0000h	120	CLK_REQ5_L_EGPI120
GPIOx001E4	0024_0000h	121	CLK_REQ6_L_EGPI121
GPIOx001E8	0000_0000h	122	
GPIOx001EC	0000_0000h	123	
GPIOx001F0	0000_0000h	124	
GPIOx001F4	0000_0000h	125	
GPIOx001F8	0000_0000h	126	
GPIOx001FC	0000_0000h	127	

Table 84: Reset Value for GPIO BANK2

Register	Reset Value	Index	Function
GPIOx00200	0000_0000h	128	
GPIOx00204	0014_0000h	129	ESPI_RESET_L_KBRST_L_AGPI129
GPIOx00208	0014_0000h	130	SATA_ACT_L_AGPI130
GPIOx0020C	0014_0000h	131	CLK_REQ3_L_SATA_IS1_L_SATA_ZP1_L_EGPI131
GPIOx00210	0014_0000h	132	CLK_REQ4_L_OSCIN_EGPI132
GPIOx00214	0000_0000h	133	
GPIOx00218	0000_0000h	134	

GPIOx0021C	0024_0000h	135	UART0_CTS_L_UART2_RXD_EGPIO135
GPIOx00220	0024_0000h	136	UART0_RXD_EGPIO136
GPIOx00224	0014_0000h	137	UART0_RTS_L_UART2_TXD_EGPIO137
GPIOx00228	0014_0000h	138	UART0_TXD_EGPIO138
GPIOx0022C	0024_0000h	139	UART0_INTR_AGPIO139
GPIOx00230	0024_0000h	140	UART1_CTS_L_UART3_TXD_EGPIO140
GPIOx00234	0024_0000h	141	UART1_RXD_EGPIO141
GPIOx00238	0014_0000h	142	UART1_RTS_L_UART3_RXD_EGPIO142
GPIOx0023C	0014_0000h	143	UART1_TXD_EGPIO143
GPIOx00240	0024_0000h	144	UART1_INTR_AGPIO144
GPIOx00244	0000_0000h	145	
GPIOx00248	0000_0000h	146	
GPIOx0024C	0000_0000h	147	
GPIOx00250	0000_0000h	148	
GPIOx00254	0000_0000h	149	
GPIOx00258	0000_0000h	150	
GPIOx0025C	0000_0000h	151	
GPIOx00260	0000_0000h	152	
GPIOx00264	0000_0000h	153	
GPIOx00268	0000_0000h	154	
GPIOx0026C	0000_0000h	155	
GPIOx00270	0000_0000h	156	
GPIOx00274	0000_0000h	157	
GPIOx00278	0000_0000h	158	
GPIOx0027C	0000_0000h	159	
GPIOx00280	0000_0000h	160	
GPIOx00284	0000_0000h	161	
GPIOx00288	0000_0000h	162	
GPIOx0028C	0000_0000h	163	
GPIOx00290	0000_0000h	164	
GPIOx00294	0000_0000h	165	
GPIOx00298	0000_0000h	166	
GPIOx0029C	0000_0000h	167	
GPIOx002A0	0000_0000h	168	
GPIOx002A4	0000_0000h	169	
GPIOx002A8	0000_0000h	170	
GPIOx002AC	0000_0000h	171	
GPIOx002B0	0000_0000h	172	NBGppPmePulse
GPIOx002B4	0000_0000h	173	NBGppHpPulse
GPIOx002B8	0000_0000h	174	AcpiPerfIntr
GPIOx002BC	0000_0000h	175	sata_sci_ & sata_sci2_
GPIOx002C0	0000_0000h	176	FanThermal_SCIOut
GPIOx002C4	0000_0000h	177	ASFMasterIntr
GPIOx002C8	0000_0000h	178	Ras_event
GPIOx002CC	0000_0000h	179	GBL_RLS
GPIOx002D0	0000_0000h	180	ShortTimerEvent   LongTimerEvent
GPIOx002D4	0000_0000h	181	NBHwAssertion_r[3]

			NBSciAssertion_r[3]
GPIOx002D8	0000_0000h	182	eSPI_WAKE_PME_B
GPIOx002DC	0000_0000h	183	eSPI_SYS_EVT_B
GPIOx002E0	0000_0000h	184	
GPIOx002E4	0000_0000h	185	
GPIOx002E8	0000_0000h	186	
GPIOx002EC	0000_0000h	187	
GPIOx002F0	0000_0000h	188	
GPIOx002F4	0000_0000h	189	
GPIOx002F8	0000_0000h	190	
GPIOx002FC	0000_0000h	191	

Table 85: Reset Value for GPIO BANK3

Register	Reset Value	Index	Function
GPIOx00300	0000_0000h	192	Int_NBGppPortDev0Pme
GPIOx00304	0000_0000h	193	Int_NBGppPortDev1Pme
GPIOx00308	0000_0000h	194	Int_NBGppPortDev2Pme
GPIOx0030C	0000_0000h	195	Int_NBGppPortDev3Pme
GPIOx00310	0000_0000h	196	Int_NBGppPortDev4Pme
GPIOx00314	0000_0000h	197	Int_NBGppPortDev5Pme
GPIOx00318	0000_0000h	198	Int_NBGppPortDev6Pme
GPIOx0031C	0000_0000h	199	Int_NBGppPortDev7Pme
GPIOx00320	0000_0000h	200	Int_NBGppPortDev8Pme
GPIOx00324	0000_0000h	201	Int_NBGppPortDev9Pme
GPIOx00328	0000_0000h	202	Int_NBGppPortDev10Pme
GPIOx0032C	0000_0000h	203	Int_NBGppPortDev11Pme
GPIOx00330	0000_0000h	204	Int_NBGppPortDev12Pme
GPIOx00334	0000_0000h	205	Int_NBGppPortDev13Pme
GPIOx00338	0000_0000h	206	Int_NBGppPortDev14Pme
GPIOx0033C	0000_0000h	207	Int_NBGppPortDev15Pme
GPIOx00340	0000_0000h	208	Int_NBGppPortDev16Pme
GPIOx00344	0000_0000h	209	Int_NBGppPortDev17Pme
GPIOx00348	0000_0000h	210	Int_NBGppPortDev18Pme
GPIOx0034C	0000_0000h	211	Int_NBGppPortDev19Pme
GPIOx00350	0000_0000h	212	Int_NBGppPortDev20Pme
GPIOx00354	0000_0000h	213	Int_NBGppPortDev21Pme
GPIOx00358	0000_0000h	214	Int_NBGppPortDev22Pme
GPIOx0035C	0000_0000h	215	Int_NBGppPortDev23Pme
GPIOx00360	0000_0000h	216	Int_NBGppPortDev24Pme
GPIOx00364	0000_0000h	217	Int_NBGppPortDev25Pme
GPIOx00368	0000_0000h	218	Int_NBGppPortDev26Pme
GPIOx0036C	0000_0000h	219	Int_NBGppPortDev27Pme
GPIOx00370	0000_0000h	220	Int_NBGppPortDev28Pme
GPIOx00374	0000_0000h	221	Int_NBGppPortDev29Pme
GPIOx00378	0000_0000h	222	Int_NBGppPortDev30Pme
GPIOx0037C	0000_0000h	223	Int_NBGppPortDev31Pme
GPIOx00380	0000_0000h	224	

GPIOx00384	0000_0000h	225	
GPIOx00388	0000_0000h	226	
GPIOx0038C	0000_0000h	227	
GPIOx00390	0000_0000h	228	
GPIOx00394	0000_0000h	229	
GPIOx00398	0000_0000h	230	
GPIOx0039C	0000_0000h	231	
GPIOx003A0	0000_0000h	232	
GPIOx003A4	0000_0000h	233	
GPIOx003A8	0000_0000h	234	
GPIOx003AC	0000_0000h	235	
GPIOx003B0	0000_0000h	236	
GPIOx003B4	0000_0000h	237	
GPIOx003B8	0000_0000h	238	
GPIOx003BC	0000_0000h	239	
GPIOx003C0	0000_0000h	240	
GPIOx003C4	0000_0000h	241	
GPIOx003C8	0000_0000h	242	
GPIOx003CC	0000_0000h	243	
GPIOx003D0	0000_0000h	244	
GPIOx003D4	0000_0000h	245	
GPIOx003D8	0000_0000h	246	
GPIOx003DC	0000_0000h	247	
GPIOx003E0	0000_0000h	248	
GPIOx003E4	0000_0000h	249	
GPIOx003E8	0000_0000h	250	
GPIOx003EC	0000_0000h	251	
GPIOx003F0	0000_0000h	252	
GPIOx003F4	0000_0000h	253	
GPIOx003F8	0000_0000h	254	
GPIOx003FC	0000_0000h	255	

Table 86: Debounce Timer Definition

DebounceTmrLarge	DebounceTmrOutUnit	Timer Unit	Max Debounce Time
0	0	61 usec (2 RtcClk)	915 usec
0	1	183 usec (6 RtcClk)	2.75 msec
1	0	15.56 msec (510 RtcClk)	233 msec
1	1	62.44 msec (2046 RtcClk)	936 msec

All registers in GpioBank0 except FCH::GPIO::GPIOWakeIntMasSwitch[GpioInterruptEn] are reset by the following conditions:

- Resume Reset: This reset is asserted in G3 state, and de asserted during G3 to S5 transition.
- System Reset: From system reset button.
- S0 Reset events: Some events that happen in S0 state, such as CF9 and Key-board Reset.

FCH::GPIO::GPIOWakeIntMasSwitch[GpioInterruptEn] are set to 1 by PciRstB, which will be asserted in the following conditions:

- Resume Reset: This reset is asserted in G3 state, and de asserted during G3 to S5 transition.
- System Reset: From system reset button.
- S0 Reset events: Some events that happen in S0 state, such as CF9 and Key-board Reset.
- Sleep states: S3,S5 and S0i3 states.

**GPIOx000[00...F8] [GPIO Bank 0 Control Register] (FCH::GPIO::GPIOBank0Ctl)**

Each GPIO pin is controlled by 4 bytes. These registers control GPIO bank 0 pins: GPIO[62:00].

_link[62:0]_aliasHOST; GPIOx000[F8,F4,F0,EC,E8,E4,E0,DC,D8,D4,D0,CC,C8,C4,C0,BC,B8,B4,B0,AC,A8,A4,A0,9C,98,94,90,8C,88,84,80,7C,78,74,70,6C,68,64,60,5C,58,54,50,4C,48,44,40,3C,38,34,30,2C,28,24,20,1C,18,14,10,0C,08,04,00]; GPIO=FED8_1500h											
Bits	Description										
31	<b>Less10secSts.</b> Read-only. Reset: X. 1=The power button is pressed for less than 10 second in S0 state. This bit is only valid for PWR_BTN_L_AGPIO0. For other GPIOs, this bit is Reserved. This bit can be cleared by writing 1 to InterruptSts bit.										
30	<b>Less2secSts.</b> Read-only. Reset: X. 1=The power button is pressed for less than 2 second in S0 state. This bit is only valid for PWR_BTN_L_AGPIO0. For other GPIOs, this bit is Reserved. When Less2secSts becomes 1, Less10secSts also becomes 1. This bit can be cleared by writing 1 to InterruptSts bit.										
29	<b>WakeSts.</b> Read,Write-1-to-clear. Reset: X. 0=The pin didn't generate a wake event. 1=The pin is a wake source. Wake Status.										
28	<b>InterruptSts.</b> Read,Write-1-to-clear. Reset: X. 0=The pin did not generate an interrupt. 1=The pin is an interrupt source. Interrupt status.										
27:24	Reserved.										
23	<b>OutputEnable.</b> Read-write. Reset: X. 0=Output is disabled on the pin. 1=Output is enabled on the pin. NOTE: I2C buffers are OD only and cannot support push-pull regardless if they are set in GPIO mode or I2C mode.										
22	<b>OutputValue.</b> Read-write. Reset: X. 0=Low. 1=High. When selecting I2C pad, OutputValue == don't care. NOTE: I2C buffers are OD only and cannot support push-pull regardless if they are set in GPIO mode or I2C mode.										
21	<b>PullDownEnable.</b> Read-write. Reset: X. 0=Pull-down is disabled on the pin. 1=Pull-down is enabled on the pin. Pull-down enable is not applicable for an I2C pad.										
20	<b>PullUpEnable.</b> Read-write. Reset: X. 0=Pull-up is disabled on the pin. 1=Pull-up is enabled on the pin. Pull-up enable is not applicable for an I2C pad.										
19	Reserved.										
18:17	<b>DrvStrengthSel.</b> Read-write. Reset: XXb. 3.3V PAD: x0: Z=40ohms x1: Z=80ohms; 1.8VPAD. NOTE: Drive strengths of 40/60/80 ohms using DrvStrengthSel bit of this GPIO Bank Control Register are not applicable for an I2C pad. I2C pad parameters are controlled by FCH::MISC::I2C3_PadCtrl and FCH::MISC::I2C2_PadCtrl. For example, Drive strength is defined by bits[3:0].										
<b>ValidValues:</b>											
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>Not supported</td></tr> <tr> <td>1h</td><td>60 ohms</td></tr> <tr> <td>2h</td><td>40 ohms</td></tr> <tr> <td>3h</td><td>80 ohms</td></tr> </table>	Value	Description	0h	Not supported	1h	60 ohms	2h	40 ohms	3h	80 ohms
Value	Description										
0h	Not supported										
1h	60 ohms										
2h	40 ohms										
3h	80 ohms										
16	<b>PinSts.</b> Read-only. Reset: X. 0=The pin is low. 1=The pin is high. This bit is not affected by the debounce logic.										
15:13	<b>WakeCntrl[2:0].</b> Read-write. Reset: XXXb. Wake control.										
<b>ValidValues:</b>											
	<table> <tr> <th>Bit</th><th>Description</th></tr> <tr> <td>[0]</td><td>1=Enables wake in S0i3 state.</td></tr> <tr> <td>[1]</td><td>1=Enables wake in S3 state (SLP_TYP ==3 and !G0_State).</td></tr> <tr> <td>[2]</td><td>1=Enables wake in S4/S5 state ((SLP_TYP == (4  5)) &amp;&amp; !G0_State).</td></tr> </table>	Bit	Description	[0]	1=Enables wake in S0i3 state.	[1]	1=Enables wake in S3 state (SLP_TYP ==3 and !G0_State).	[2]	1=Enables wake in S4/S5 state ((SLP_TYP == (4  5)) && !G0_State).		
Bit	Description										
[0]	1=Enables wake in S0i3 state.										
[1]	1=Enables wake in S3 state (SLP_TYP ==3 and !G0_State).										
[2]	1=Enables wake in S4/S5 state ((SLP_TYP == (4  5)) && !G0_State).										
12	<b>InterruptEnable[1].</b> Read-write. Reset: X. 0=Disable. 1=Enable interrupt delivery.										
11	<b>InterruptEnable[0].</b> Read-write. Reset: X. 0=Disable. 1=Enable interrupt status.										
10:9	<b>ActiveLevel.</b> Read-write. Reset: XXb.										
<b>ValidValues:</b>											
	<table> <tr> <th>Value</th><th>Description</th></tr> </table>	Value	Description								
Value	Description										



	0h	Active high.
	1h	Active low.
	2h	Active on both of edges if LevelTrig == 0.
	3h	Reserved.
8	<b>LevelTrig.</b> Read-write. Reset: X. 0=Edge trigger. 1=Level trigger.	
7	<b>DebounceTmrLarge.</b> Read-write. Reset: X. Combined with DebounceTmrOutUnit, this bit changes the unit and max debounce time for the debounce timer. See Table 86 [Debounce Timer Definition].	
6:5	<b>DebounceCntrl.</b> Read-write. Reset: XXb.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	No debounce.
	1h	Preserve low glitch.
	2h	Preserve high glitch.
	3h	Remove glitch.
4	<b>DebounceTmrOutUnit.</b> Read-write. Reset: X. DebounceTmrLarge and DebounceTmrOutUnit defines the unit and max debounce time for the debounce timer. See Table 86 [Debounce Timer Definition].	
3:0	<b>DebounceTmrOut.</b> Read-write. Reset: XXXXb. Specifies the debounce timer out number.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	Debounceing logic is disabled.
	Fh-1h	Debounce timer out number.

#### GPIOx00FC [GPIO\_Wake\_Inter\_Master Switch] (FCH::GPIO::GPIOWakeIntMasSwitch)

Read-write. Reset: FF00\_0000h.

\_aliasHOST; GPIOx00FC; GPIO=FED8\_1500h

Bits	Description
31	<b>GpioWakeEn.</b> Read-write. Reset: 1. 0=All GPIO wakes are blocked.
30	<b>GpioInterruptEn.</b> Read-write. Reset: 1. 0=All GPIO interrupts are blocked.
29	<b>EOI.</b> Read-write. Reset: 1. 0=The GPIO interrupt is blocked. 1=Allow GPIO interrupt, hardware clears this bit when interrupt occurs. This software/hardware handshake mechanism is the same as EOS of SMI.
28	<b>MaskStsEn.</b> Read-write. Reset: 1. 1=Enable hardware to block all wake/interrupt status generation when software Writes to any debounce registers. The length of blocking depends on mask_sts_length[3:0].
27:24	<b>MaskStsLength[3:0].</b> Read-write. Reset: Fh. See MaskStsEn and MaskStsLength[11:4]. The length of blocking = {MaskStsLength[11:0], 14'h3FFF}.
23:16	<b>MaskStsLength[11:4].</b> Read-write. Reset: 00h. See MaskStsEn and MaskStsLength[3:0].
15	<b>EnWinBlueBtn.</b> Read-write. Reset: 0. 0=GPIO0 detects debounced power button; Power button override is 4 sec. 1=GPIO0 detects debounced power button in S3/S5/S0I3, and detects "pressed less than 2 sec" and "pressed 2~10 sec" in S0; Power button override is 10 sec.
14	<b>IntrOutActiveHi.</b> Read-write. Reset: 0. 0=GPIO controller interrupt output is active low. 1=GPIO controller interrupt output is active high.
13	<b>SelGpio0Src: Select the source for GPIO0 detection.</b> Read-write. Reset: 0. 0=Power button goes to a processing logic first and then goes to GPIO0 detection logic. 1=Power button goes to GPIO0 debounce and then goes to GPIO0 detection logic. The "processing logic" includes 16 ms debounce counter and a logic to detect how long the button has been pressed to generate press_less2s_sts and press_less4s_sts. Only "GPIO0 debounce block" has debounce function.
12	<b>IntrOutPulse.</b> Read-write. Reset: 0. 0=GPIO controller interrupt output is a level signal. 1=GPIO controller interrupt output is pulse signal. The polarity is defined by IntrOutActiveHi register bit.
11:0	Reserved.

#### GPIOx001[00...FC] [GPIO Bank 1 Control Register] (FCH::GPIO::GPIOBank1Ctl)

Each GPIO pin is controlled by 4 bytes. These registers control GPIO bank 1 pins: GPIO[127:64].											
_link[63:0]_aliasHOST; GPIOx001[FC,F8,F4,F0,EC,E8,E4,E0,DC,D8,D4,D0,CC,C8,C4,C0,BC,B8,B4,B0,AC,A8,A4,A0,9C,98,94,90,8C,88,84,80,7C,78,74,70,6C,68,64,60,5C,58,54,50,4C,48,44,40,3C,38,34,30,2C,28,24,20,1C,18,14,10,0C,08,04,00]; GPIO=FED8_1500h											
Bits	Description										
31:30	Reserved.										
29	<b>WakeSts.</b> Read,Write-1-to-clear. Reset: X. 0=The pin didn't generate a wake event. 1=The pin is a wake source. Wake status.										
28	<b>InterruptSts.</b> Read,Write-1-to-clear. Reset: X. 0=The pin didn't generate an interrupt. 1=The pin is an interrupt source. Interrupt status.										
27:24	Reserved.										
23	<b>OutputEnable.</b> Read-write. Reset: X. 0=Output is disabled on the pin. 1=Output is enabled on the pin. NOTE: I2C buffers are OD only and cannot support push-pull regardless if they are set in GPIO mode or I2C mode.										
22	<b>OutputValue.</b> Read-write. Reset: X. 0=Low. 1=High. When selecting I2C pad, OutputValue == don't care. NOTE: I2C buffers are OD only and cannot support push-pull regardless if they are set in GPIO mode or I2C mode.										
21	<b>PullDownEnable.</b> Read-write. Reset: X. 0=Pull-down is disabled on the pin. 1=Pull-down is enabled on the pin. Pull-down enable is not applicable for an I2C pad.										
20	<b>PullUpEnable.</b> Read-write. Reset: X. 0=Pull-up is disabled on the pin. 1=Pull-up is enabled on the pin. Pull-up enable is not applicable for an I2C pad.										
19	Reserved.										
18:17	<b>DrvStrengthSel.</b> Read-write. Reset: XXb. 3.3V PAD: x0: Z=40ohms x1: Z=80ohms; 1.8VPAD. NOTE: Drive strengths of 40/60/80 ohms using DrvStrengthSel bit of this GPIO Bank Control Register are not applicable for an I2C pad. I2C pad parameters are controlled by FCH::MISC::I2C3_PadCtrl and FCH::MISC::I2C2_PadCtrl. For example, Drive strength is defined by bits[3:0].										
<b>ValidValues:</b>											
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>Not supported</td></tr> <tr> <td>1h</td><td>60 ohms</td></tr> <tr> <td>2h</td><td>40 ohms</td></tr> <tr> <td>3h</td><td>80 ohms</td></tr> </table>	Value	Description	0h	Not supported	1h	60 ohms	2h	40 ohms	3h	80 ohms
Value	Description										
0h	Not supported										
1h	60 ohms										
2h	40 ohms										
3h	80 ohms										
16	<b>PinSts.</b> Read-only. Reset: X. 0=The pin is low. 1=The pin is high. This bit is not affected by the debounce logic.										
15:13	<b>WakeCntrl[2:0].</b> Read-write. Reset: XXXb. Wake control.										
<b>ValidValues:</b>											
	<table> <tr> <th>Bit</th><th>Description</th></tr> <tr> <td>[0]</td><td>1=Enables wake in S0i3 state.</td></tr> <tr> <td>[1]</td><td>1=Enables wake in S3 state (SLP_TYP == 3 and !G0_State).</td></tr> <tr> <td>[2]</td><td>1=Enables wake in S4/S5 state ((SLP_TYP == (4  5)) &amp;&amp; !G0_State).</td></tr> </table>	Bit	Description	[0]	1=Enables wake in S0i3 state.	[1]	1=Enables wake in S3 state (SLP_TYP == 3 and !G0_State).	[2]	1=Enables wake in S4/S5 state ((SLP_TYP == (4  5)) && !G0_State).		
Bit	Description										
[0]	1=Enables wake in S0i3 state.										
[1]	1=Enables wake in S3 state (SLP_TYP == 3 and !G0_State).										
[2]	1=Enables wake in S4/S5 state ((SLP_TYP == (4  5)) && !G0_State).										
12:11	<b>InterruptEnable[1:0].</b> Read-write. Reset: XXb. Enable interrupt status and delivery.										
<b>ValidValues:</b>											
	<table> <tr> <th>Bit</th><th>Description</th></tr> <tr> <td>[0]</td><td>1=Enable interrupt status.</td></tr> <tr> <td>[1]</td><td>1=Enable interrupt delivery.</td></tr> </table>	Bit	Description	[0]	1=Enable interrupt status.	[1]	1=Enable interrupt delivery.				
Bit	Description										
[0]	1=Enable interrupt status.										
[1]	1=Enable interrupt delivery.										
10:9	<b>ActiveLevel.</b> Read-write. Reset: XXb.										
<b>ValidValues:</b>											
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>Active high.</td></tr> <tr> <td>1h</td><td>Active low.</td></tr> <tr> <td>2h</td><td>Active on both of edges if LevelTrig == 0.</td></tr> <tr> <td>3h</td><td>Reserved.</td></tr> </table>	Value	Description	0h	Active high.	1h	Active low.	2h	Active on both of edges if LevelTrig == 0.	3h	Reserved.
Value	Description										
0h	Active high.										
1h	Active low.										
2h	Active on both of edges if LevelTrig == 0.										
3h	Reserved.										

8	<b>LevelTrig.</b> Read-write. Reset: X. 0=Edge trigger. 1=Level trigger.										
7	<b>DebounceTmrLarge.</b> Read-write. Reset: X. Combined with DebounceTmrOutUnit, this bit changes the unit and max debounce time for the debounce timer. See Table 86 [Debounce Timer Definition].										
6:5	<b>DebounceCntl.</b> Read-write. Reset: XXb. <b>ValidValues:</b>										
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>No debounce.</td></tr> <tr> <td>1h</td><td>Preserve low glitch.</td></tr> <tr> <td>2h</td><td>Preserve high glitch.</td></tr> <tr> <td>3h</td><td>Remove glitch.</td></tr> </table>	Value	Description	0h	No debounce.	1h	Preserve low glitch.	2h	Preserve high glitch.	3h	Remove glitch.
Value	Description										
0h	No debounce.										
1h	Preserve low glitch.										
2h	Preserve high glitch.										
3h	Remove glitch.										
4	<b>DebounceTmrOutUnit.</b> Read-write. Reset: X. DebounceTmrLarge and DebounceTmrOutUnit defines the unit and max debounce time for the debounce timer. See Table 86 [Debounce Timer Definition].										
3:0	<b>DebounceTmrOut.</b> Read-write. Reset: XXXXb. Specifies the debounce timer out number. <b>ValidValues:</b>										
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>Debouncing logic is disabled.</td></tr> <tr> <td>Fh-1h</td><td>Debounce timer out number.</td></tr> </table>	Value	Description	0h	Debouncing logic is disabled.	Fh-1h	Debounce timer out number.				
Value	Description										
0h	Debouncing logic is disabled.										
Fh-1h	Debounce timer out number.										

#### GPIOx002[00...EC] [GPIO Bank 2 Control Register] (FCH::GPIO::GPIOBank2Ctl)

Each GPIO pin is controlled by 4 bytes. These registers control GPIO Bank 2 pins: GPIO[187:128].

\_link[59:0]\_aliasHOST;  
GPIOx002[EC,E8,E4,E0,DC,D8,D4,D0,CC,C8,C4,C0,BC,B8,B4,B0,AC,A8,A4,A0,9C,98,94,90,8C,88,84,80,7C,78,74,70,6C,68,64,60,5C,58,54,50,4C,48,44,40,3C,38,34,30,2C,28,24,20,1C,18,14,10,0C,08,04,00]; GPIO=FED8\_1500h

Bits	Description										
31:30	Reserved.										
29	<b>WakeSts.</b> Read,Write-1-to-clear. Reset: X. 0=The pin did not generate a wake event. 1=The pin is a wake source. Wake status.										
28	<b>InterruptSts.</b> Read,Write-1-to-clear. Reset: X. 0=The pin did not generate an interrupt. 1=The pin is an interrupt source. Interrupt status.										
27:24	Reserved.										
23	<b>OutputEnable.</b> Read-write. Reset: X. 0=Output is disabled on the pin. 1=Output is enabled on the pin. NOTE: I2C buffers are OD only and cannot support push-pull regardless if they are set in GPIO mode or I2C mode.										
22	<b>OutputValue.</b> Read-write. Reset: X. 0=Low. 1=High. When selecting I2C pad, Outputvalue == don't care. NOTE: I2C buffers are OD only and cannot support push-pull regardless if they are set in GPIO mode or I2C mode.										
21	<b>PullDownEnable.</b> Read-write. Reset: X. 0=Pull-down is disabled on the pin. 1=Pull-down is enabled on the pin. Pull-down enable is not applicable to I2C pad.										
20	<b>PullUpEnable.</b> Read-write. Reset: X. 0=Pull-up is disabled on the pin. 1=Pull-up is enabled on the pin. Pull-up enable is not applicable to I2C pad.										
19	Reserved.										
18:17	<b>DrvStrengthSel.</b> Read-write. Reset: XXb. 3.3V PAD: x0: Z=40ohms x1: Z=80ohms; 1.8VPAD. NOTE: Drive strengths of 40/60/80 ohms using DrvStrengthSel bit of this GPIO Bank Control Register are not applicable to I2C pad. I2C pad parameters are controlled by FCH::MISC::I2C3_PadCtrl and FCH::MISC::I2C2_PadCtrl. For example, Drive strength is defined by bits[3:0]. <b>ValidValues:</b>										
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>Not supported</td></tr> <tr> <td>1h</td><td>60 ohms</td></tr> <tr> <td>2h</td><td>40 ohms</td></tr> <tr> <td>3h</td><td>80 ohms</td></tr> </table>	Value	Description	0h	Not supported	1h	60 ohms	2h	40 ohms	3h	80 ohms
Value	Description										
0h	Not supported										
1h	60 ohms										
2h	40 ohms										
3h	80 ohms										
16	<b>PinSts.</b> Read-only. Reset: X. 0=The pin is low. 1=The pin is high. This bit is not affected by the debounce logic.										

15:13	<b>WakeCntrl[2:0]</b> . Read-write. Reset: XXXb. Wake control.
<b>ValidValues:</b>	
<b>Bit</b>	<b>Description</b>
[0]	1=Enables wake in S0i3 state.
[1]	1=Enables wake in S3 state (SLP_TYP == 3 and !G0_State).
[2]	1=Enables wake in S4/S5 state ((SLP_TYP == (4  5)) && !G0_State).
12:11	<b>InterruptEnable[1:0]</b> . Read-write. Reset: XXb. Enable interrupt status and delivery.
<b>ValidValues:</b>	
<b>Bit</b>	<b>Description</b>
[0]	1=Enable interrupt status.
[1]	1=Enable interrupt delivery.
10:9	<b>ActiveLevel</b> . Read-write. Reset: XXb.
<b>ValidValues:</b>	
<b>Value</b>	<b>Description</b>
0h	Active high.
1h	Active low.
2h	Active on both of edges if LevelTrig == 0.
3h	Reserved.
8	<b>LevelTrig</b> . Read-write. Reset: X. 0=Edge trigger. 1=Level trigger.
7	<b>DebounceTmrLarge</b> . Read-write. Reset: X. Combined with DebounceTmrOutUnit, this bit changes the unit and max debounce time for the debounce timer. See Table 86 [Debounce Timer Definition].
6:5	<b>DebounceCntrl</b> . Read-write. Reset: XXb.
<b>ValidValues:</b>	
<b>Value</b>	<b>Description</b>
0h	No debounce.
1h	Preserve low glitch.
2h	Preserve high glitch.
3h	Remove glitch.
4	<b>DebounceTmrOutUnit</b> . Read-write. Reset: X. DebounceTmrLarge and DebounceTmrOutUnit defines the unit and max debounce time for the debounce timer. See Table 86 [Debounce Timer Definition].
3:0	<b>DebounceTmrOut</b> . Read-write. Reset: XXXXb. Specifies the debounce timer out number.
<b>ValidValues:</b>	
<b>Value</b>	<b>Description</b>
0h	Debounceing logic is disabled.
Fh-1h	Debounce timer out number.

#### GPIOx02F0 [GPIO\_Wake\_Status\_Index\_0] (FCH::GPIO::GPIOWakeStatIndex0)

Read-only. Reset: 0000\_0000h.

\_aliasHOST; GPIOx02F0; GPIO=FED8\_1500h

Bits	Description
31:16	<b>WakeStatusIndex[31:16]</b> . Read-only. Reset: 0000h. When bit[N] is 1, at least one wake status of GPIO N*4 ~ N*4+3 is 1.
<b>ValidValues:</b>	
<b>Bit</b>	<b>Description</b>
[16]	1=At least one wake status of GPIO[67:64] is 1. 0=None of the GPIO[67:64] wake status are 1.
[17]	1=At least one wake status of GPIO[71:68] is 1. 0=None of the GPIO[71:68] wake status are 1.
[18]	1=At least one wake status of GPIO[75:72] is 1. 0=None of the GPIO[75:72] wake status are 1.
[19]	1=At least one wake status of GPIO[79:76] is 1. 0=None of the GPIO[79:76] wake status are 1.
[20]	1=At least one wake status of GPIO[83:80] is 1. 0=None of the GPIO[83:80] wake status are 1.

	[21]	1=At least one wake status of GPIO[87:84] is 1. 0=None of the GPIO[87:84] wake status are 1.
	[22]	1=At least one wake status of GPIO[91:88] is 1. 0=None of the GPIO[91:88] wake status are 1.
	[23]	1=At least one wake status of GPIO[95:92] is 1. 0=None of the GPIO[95:92] wake status are 1.
	[24]	1=At least one wake status of GPIO[99:96] is 1. 0=None of the GPIO[99:96] wake status are 1.
	[25]	1=At least one wake status of GPIO[103:100] is 1. 0=None of the GPIO[103:100] wake status are 1.
	[26]	1=At least one wake status of GPIO[107:104] is 1. 0=None of the GPIO[107:104] wake status are 1.
	[27]	1=At least one wake status of GPIO[111:108] is 1. 0=None of the GPIO[111:108] wake status are 1.
	[28]	1=At least one wake status of GPIO[115:112] is 1. 0=None of the GPIO[115:112] wake status are 1.
	[29]	1=At least one wake status of GPIO[119:116] is 1. 0=None of the GPIO[119:116] wake status are 1.
	[30]	1=At least one wake status of GPIO[123:120] is 1. 0=None of the GPIO[123:120] wake status are 1.
	[31]	1=At least one wake status of GPIO[127:124] is 1. 0=None of the GPIO[127:124] wake status are 1.
15	<b>WakeStatusIndex[15]</b> . Read-only. Reset: 0. 0=None of the GPIO[62:60] wake status are 1. 1=At least one wake status of GPIO[62:60] is 1. When this bit is 1, at least one wake status of GPIO[62:60] is 1.	
14:0	<b>WakeStatusIndex[14:0]</b> . Read-only. Reset: 0000h. When bit[N] is 1, at least one wake status of GPIO N*4 ~ N*4+3 is 1.	
	<b>ValidValues:</b>	
	<b>Bit</b>	<b>Description</b>
	[0]	1=At least one wake status of GPIO[3:0] is 1. 0=None of the GPIO[3:0] wake status are 1.
	[1]	1=At least one wake status of GPIO[7:4] is 1. 0=None of the GPIO[7:4] wake status are 1.
	[2]	1=At least one wake status of GPIO[11:8] is 1. 0=None of the GPIO[11:8] wake status are 1.
	[3]	1=At least one wake status of GPIO[15:12] is 1. 0=None of the GPIO[15:12] wake status are 1.
	[4]	1=At least one wake status of GPIO[19:16] is 1. 0=None of the GPIO[19:16] wake status are 1.
	[5]	1=At least one wake status of GPIO[23:20] is 1. 0=None of the GPIO[23:20] wake status are 1.
	[6]	1=At least one wake status of GPIO[27:24] is 1. 0=None of the GPIO[27:24] wake status are 1.
	[7]	1=At least one wake status of GPIO[31:28] is 1. 0=None of the GPIO[31:28] wake status are 1.
	[8]	1=At least one wake status of GPIO[35:32] is 1. 0=None of the GPIO[35:32] wake status are 1.
	[9]	1=At least one wake status of GPIO[39:36] is 1. 0=None of the GPIO[39:36] wake status are 1.
	[10]	1=At least one wake status of GPIO[43:40] is 1. 0=None of the GPIO[43:40] wake status are 1.
	[11]	1=At least one wake status of GPIO[47:44] is 1. 0=None of the GPIO[47:44] wake status are 1.
	[12]	1=At least one wake status of GPIO[51:48] is 1. 0=None of the GPIO[51:48] wake status are 1.
	[13]	1=At least one wake status of GPIO[55:52] is 1. 0=None of the GPIO[55:52] wake status are 1.
	[14]	1=At least one wake status of GPIO[59:56] is 1. 0=None of the GPIO[59:56] wake status are 1.

**GPIOx02F4 [GPIO\_Wake\_Status\_Index\_1] (FCH::GPIO::GPIOWakeStatIndex1)**

Read-only. Reset: 0000\_0000h.

\_aliasHOST; GPIOx02F4; GPIO=FED8\_1500h

Bits	Description										
31:16	Reserved.										
15	<b>NBGppPmeWake</b> . Read-only. Reset: 0. 0=No NBGPP PME wake event. 1=NBGPP has sent PME to wake the system. One wake status is set in GPIO Bank3 register.										
14	Reserved.										
13:0	<b>WakeStatusIndex[13:0]</b> . Read-only. Reset: 0000h. For each bit, 1=At least one of the wake status of GPIO N*4 ~ N*4+3 is 1.										
	<b>ValidValues:</b>										
	<table><tr><th>Bit</th><th>Description</th></tr><tr><td>[0]</td><td>1=At least one wake status of GPIO[131:128] is 1.</td></tr><tr><td>[1]</td><td>1=At least one wake status of GPIO[135:132] is 1.</td></tr><tr><td>[2]</td><td>1=At least one wake status of GPIO[139:136] is 1.</td></tr><tr><td>[3]</td><td>1=At least one wake status of GPIO[143:140] is 1.</td></tr></table>	Bit	Description	[0]	1=At least one wake status of GPIO[131:128] is 1.	[1]	1=At least one wake status of GPIO[135:132] is 1.	[2]	1=At least one wake status of GPIO[139:136] is 1.	[3]	1=At least one wake status of GPIO[143:140] is 1.
Bit	Description										
[0]	1=At least one wake status of GPIO[131:128] is 1.										
[1]	1=At least one wake status of GPIO[135:132] is 1.										
[2]	1=At least one wake status of GPIO[139:136] is 1.										
[3]	1=At least one wake status of GPIO[143:140] is 1.										

[4]	1=At least one wake status of GPIO[147:144] is 1.
[5]	1=At least one wake status of GPIO[151:148] is 1.
[6]	1=At least one wake status of GPIO[155:152] is 1.
[7]	1=At least one wake status of GPIO[159:156] is 1.
[8]	1=At least one wake status of GPIO[163:160] is 1.
[9]	1=At least one wake status of GPIO[167:164] is 1.
[10]	1=At least one wake status of GPIO[171:168] is 1.
[11]	1=At least one wake status of GPIO[175:172] is 1.
[12]	1=At least one wake status of GPIO[179:176] is 1.
[13]	1=At least one wake status of GPIO[183:180] is 1.

**GPIOx02F8 [GPIO\_Interrupt\_Status\_Index\_0] (FCH::GPIO::GPIOIntStatIndex0)**

Read-only. Reset: 0000\_0000h.

\_aliasHOST; GPIOx02F8; GPIO=FED8\_1500h

Bits	Description																																		
31:16	<b>InterruptStatusIndex[31:16].</b> Read-only. Reset: 0000h. When bit[N] is 1, at least one interrupt status of GPIO N*4 ~ N*4+3 is 1. <b>ValidValues:</b> <table> <tr> <th>Bit</th><th>Description</th></tr> <tr><td>[16]</td><td>1=At least one interrupt status of GPIO[67:64] is 1.</td></tr> <tr><td>[17]</td><td>1=At least one interrupt status of GPIO[71:68] is 1.</td></tr> <tr><td>[18]</td><td>1=At least one interrupt status of GPIO[75:72] is 1.</td></tr> <tr><td>[19]</td><td>1=At least one interrupt status of GPIO[79:76] is 1.</td></tr> <tr><td>[20]</td><td>1=At least one interrupt status of GPIO[83:80] is 1.</td></tr> <tr><td>[21]</td><td>1=At least one interrupt status of GPIO[87:84] is 1.</td></tr> <tr><td>[22]</td><td>1=At least one interrupt status of GPIO[91:88] is 1.</td></tr> <tr><td>[23]</td><td>1=At least one interrupt status of GPIO[95:92] is 1.</td></tr> <tr><td>[24]</td><td>1=At least one interrupt status of GPIO[99:96] is 1.</td></tr> <tr><td>[25]</td><td>1=At least one interrupt status of GPIO[103:100] is 1.</td></tr> <tr><td>[26]</td><td>1=At least one interrupt status of GPIO[107:104] is 1.</td></tr> <tr><td>[27]</td><td>1=At least one interrupt status of GPIO[111:108] is 1.</td></tr> <tr><td>[28]</td><td>1=At least one interrupt status of GPIO[115:112] is 1.</td></tr> <tr><td>[29]</td><td>1=At least one interrupt status of GPIO[119:116] is 1.</td></tr> <tr><td>[30]</td><td>1=At least one interrupt status of GPIO[123:120] is 1.</td></tr> <tr><td>[31]</td><td>1=At least one interrupt status of GPIO[127:124] is 1.</td></tr> </table>	Bit	Description	[16]	1=At least one interrupt status of GPIO[67:64] is 1.	[17]	1=At least one interrupt status of GPIO[71:68] is 1.	[18]	1=At least one interrupt status of GPIO[75:72] is 1.	[19]	1=At least one interrupt status of GPIO[79:76] is 1.	[20]	1=At least one interrupt status of GPIO[83:80] is 1.	[21]	1=At least one interrupt status of GPIO[87:84] is 1.	[22]	1=At least one interrupt status of GPIO[91:88] is 1.	[23]	1=At least one interrupt status of GPIO[95:92] is 1.	[24]	1=At least one interrupt status of GPIO[99:96] is 1.	[25]	1=At least one interrupt status of GPIO[103:100] is 1.	[26]	1=At least one interrupt status of GPIO[107:104] is 1.	[27]	1=At least one interrupt status of GPIO[111:108] is 1.	[28]	1=At least one interrupt status of GPIO[115:112] is 1.	[29]	1=At least one interrupt status of GPIO[119:116] is 1.	[30]	1=At least one interrupt status of GPIO[123:120] is 1.	[31]	1=At least one interrupt status of GPIO[127:124] is 1.
Bit	Description																																		
[16]	1=At least one interrupt status of GPIO[67:64] is 1.																																		
[17]	1=At least one interrupt status of GPIO[71:68] is 1.																																		
[18]	1=At least one interrupt status of GPIO[75:72] is 1.																																		
[19]	1=At least one interrupt status of GPIO[79:76] is 1.																																		
[20]	1=At least one interrupt status of GPIO[83:80] is 1.																																		
[21]	1=At least one interrupt status of GPIO[87:84] is 1.																																		
[22]	1=At least one interrupt status of GPIO[91:88] is 1.																																		
[23]	1=At least one interrupt status of GPIO[95:92] is 1.																																		
[24]	1=At least one interrupt status of GPIO[99:96] is 1.																																		
[25]	1=At least one interrupt status of GPIO[103:100] is 1.																																		
[26]	1=At least one interrupt status of GPIO[107:104] is 1.																																		
[27]	1=At least one interrupt status of GPIO[111:108] is 1.																																		
[28]	1=At least one interrupt status of GPIO[115:112] is 1.																																		
[29]	1=At least one interrupt status of GPIO[119:116] is 1.																																		
[30]	1=At least one interrupt status of GPIO[123:120] is 1.																																		
[31]	1=At least one interrupt status of GPIO[127:124] is 1.																																		
15	<b>InterruptStatusIndex[15].</b> Read-only. Reset: 0. 0=None of the GPIO[62:60] interrupt status are 1. 1=At least one interrupt status of GPIO[62:60] is 1. When this bit is 1, at least one of the interrupt status of GPIO[62:60] is 1.																																		
14:0	<b>InterruptStatusIndex[14:0].</b> Read-only. Reset: 0000h. When bit[N] is 1, at least one interrupt status of GPIO N*4 ~ N*4+3 is 1. <b>ValidValues:</b> <table> <tr> <th>Bit</th><th>Description</th></tr> <tr><td>[0]</td><td>1=At least one interrupt status of GPIO[3:0] is 1.</td></tr> <tr><td>[1]</td><td>1=At least one interrupt status of GPIO[7:4] is 1.</td></tr> <tr><td>[2]</td><td>1=At least one interrupt status of GPIO[11:8] is 1.</td></tr> <tr><td>[3]</td><td>1=At least one interrupt status of GPIO[15:12] is 1.</td></tr> <tr><td>[4]</td><td>1=At least one interrupt status of GPIO[19:16] is 1.</td></tr> <tr><td>[5]</td><td>1=At least one interrupt status of GPIO[23:20] is 1.</td></tr> <tr><td>[6]</td><td>1=At least one interrupt status of GPIO[27:24] is 1.</td></tr> </table>	Bit	Description	[0]	1=At least one interrupt status of GPIO[3:0] is 1.	[1]	1=At least one interrupt status of GPIO[7:4] is 1.	[2]	1=At least one interrupt status of GPIO[11:8] is 1.	[3]	1=At least one interrupt status of GPIO[15:12] is 1.	[4]	1=At least one interrupt status of GPIO[19:16] is 1.	[5]	1=At least one interrupt status of GPIO[23:20] is 1.	[6]	1=At least one interrupt status of GPIO[27:24] is 1.																		
Bit	Description																																		
[0]	1=At least one interrupt status of GPIO[3:0] is 1.																																		
[1]	1=At least one interrupt status of GPIO[7:4] is 1.																																		
[2]	1=At least one interrupt status of GPIO[11:8] is 1.																																		
[3]	1=At least one interrupt status of GPIO[15:12] is 1.																																		
[4]	1=At least one interrupt status of GPIO[19:16] is 1.																																		
[5]	1=At least one interrupt status of GPIO[23:20] is 1.																																		
[6]	1=At least one interrupt status of GPIO[27:24] is 1.																																		



[7]	1=At least one interrupt status of GPIO[31:28] is 1.
[8]	1=At least one interrupt status of GPIO[35:32] is 1.
[9]	1=At least one interrupt status of GPIO[39:36] is 1.
[10]	1=At least one interrupt status of GPIO[43:40] is 1.
[11]	1=At least one interrupt status of GPIO[47:44] is 1.
[12]	1=At least one interrupt status of GPIO[51:48] is 1.
[13]	1=At least one interrupt status of GPIO[55:52] is 1.
[14]	1=At least one interrupt status of GPIO[59:56] is 1.

#### GPIOx02FC [GPIO\_Interrupt\_Status\_Index\_1] (FCH::GPIO::GPIOIntStatIndex1)

Reset: 1F00\_0000h.

\_aliasHOST; GPIOx02FC; GPIO=FED8\_1500h

Bits	Description																														
31:29	Reserved.																														
28	<b>MaskStsEn.</b> Read-write. Reset: 1. 0=Disable. 1=Enable hardware to block all wake/interrupt status generation when software writes to any debounce registers. The length of blocking depends on MaskStsLength[3:0].																														
27:24	<b>MaskStsLength[3:0].</b> Read-write. Reset: Fh. See also MaskStsEn and MaskStsLength[11:4]. The length of blocking = {MaskStsLength[11:0], 14'h3FFF}.																														
23:16	<b>MaskStsLength[11:4].</b> Read-write. Reset: 00h. See MaskStsEn and MaskStsLength[3:0]. MaskStsLength[11:0] = {MaskStsLength[11:4],MaskStsLength[3:0]}.																														
15	<b>NBGppPmeIntr.</b> Read-only. Reset: 0. 0=No NB GPP PME event. 1=NB GPP has sent PME (one of the interrupt status bits are set in the GPIO Bank3 register).																														
14	Reserved.																														
13:0	<b>InterruptStatusIndex[13:0].</b> Read-only. Reset: 0000h. For each bit, at least one interrupt status of GPIO N*4 ~ N*4+3 is 1. <b>ValidValues:</b>																														
	<table> <tr> <th>Bit</th><th>Description</th></tr> <tr><td>[0]</td><td>1=At least one interrupt status of GPIO[131:128] is 1.</td></tr> <tr><td>[1]</td><td>1=At least one interrupt status of GPIO[135:132] is 1.</td></tr> <tr><td>[2]</td><td>1=At least one interrupt status of GPIO[139:136] is 1.</td></tr> <tr><td>[3]</td><td>1=At least one interrupt status of GPIO[143:140] is 1.</td></tr> <tr><td>[4]</td><td>1=At least one interrupt status of GPIO[147:144] is 1.</td></tr> <tr><td>[5]</td><td>1=At least one interrupt status of GPIO[151:148] is 1.</td></tr> <tr><td>[6]</td><td>1=At least one interrupt status of GPIO[155:152] is 1.</td></tr> <tr><td>[7]</td><td>1=At least one interrupt status of GPIO[159:156] is 1.</td></tr> <tr><td>[8]</td><td>1=At least one interrupt status of GPIO[163:160] is 1.</td></tr> <tr><td>[9]</td><td>1=At least one interrupt status of GPIO[167:164] is 1.</td></tr> <tr><td>[10]</td><td>1=At least one interrupt status of GPIO[171:168] is 1.</td></tr> <tr><td>[11]</td><td>1=At least one interrupt status of GPIO[175:172] is 1.</td></tr> <tr><td>[12]</td><td>1=At least one interrupt status of GPIO[179:176] is 1.</td></tr> <tr><td>[13]</td><td>1=At least one interrupt status of GPIO[183:180] is 1.</td></tr> </table>	Bit	Description	[0]	1=At least one interrupt status of GPIO[131:128] is 1.	[1]	1=At least one interrupt status of GPIO[135:132] is 1.	[2]	1=At least one interrupt status of GPIO[139:136] is 1.	[3]	1=At least one interrupt status of GPIO[143:140] is 1.	[4]	1=At least one interrupt status of GPIO[147:144] is 1.	[5]	1=At least one interrupt status of GPIO[151:148] is 1.	[6]	1=At least one interrupt status of GPIO[155:152] is 1.	[7]	1=At least one interrupt status of GPIO[159:156] is 1.	[8]	1=At least one interrupt status of GPIO[163:160] is 1.	[9]	1=At least one interrupt status of GPIO[167:164] is 1.	[10]	1=At least one interrupt status of GPIO[171:168] is 1.	[11]	1=At least one interrupt status of GPIO[175:172] is 1.	[12]	1=At least one interrupt status of GPIO[179:176] is 1.	[13]	1=At least one interrupt status of GPIO[183:180] is 1.
Bit	Description																														
[0]	1=At least one interrupt status of GPIO[131:128] is 1.																														
[1]	1=At least one interrupt status of GPIO[135:132] is 1.																														
[2]	1=At least one interrupt status of GPIO[139:136] is 1.																														
[3]	1=At least one interrupt status of GPIO[143:140] is 1.																														
[4]	1=At least one interrupt status of GPIO[147:144] is 1.																														
[5]	1=At least one interrupt status of GPIO[151:148] is 1.																														
[6]	1=At least one interrupt status of GPIO[155:152] is 1.																														
[7]	1=At least one interrupt status of GPIO[159:156] is 1.																														
[8]	1=At least one interrupt status of GPIO[163:160] is 1.																														
[9]	1=At least one interrupt status of GPIO[167:164] is 1.																														
[10]	1=At least one interrupt status of GPIO[171:168] is 1.																														
[11]	1=At least one interrupt status of GPIO[175:172] is 1.																														
[12]	1=At least one interrupt status of GPIO[179:176] is 1.																														
[13]	1=At least one interrupt status of GPIO[183:180] is 1.																														

#### GPIOx003[00...FC] [GPIO Bank 3 Control Register] (FCH::GPIO::GPIOBank3Ctl)

Each GPIO pin is controlled by 4 bytes. These registers control GPIO bank 3 pins: GPIO[224:193].

\_link[63:0]\_aliasHOST;

GPIOx003[FC,F8,F4,F0,EC,E8,E4,E0,DC,D8,D4,D0,CC,C8,C4,C0,BC,B8,B4,B0,AC,A8,A4,A0,9C,98,94,90,8C,88,84,80,7C,78,74,70,6C,68,64,60,5C,58,54,50,4C,48,44,40,3C,38,34,30,2C,28,24,20,1C,18,14,10,0C,08,04,00]; GPIO=FED8\_1500h

Bits	Description
31:30	Reserved.
29	<b>WakeSts.</b> Read,Write-1-to-clear. Reset: X. 0=The pin did not generate a wake event. 1=The pin is a wake source. Wake status.



28	<b>InterruptSts.</b> Read, Write-1-to-clear. Reset: X. 0=The pin did not generate an interrupt. 1=The pin is an interrupt source. Interrupt status.										
27:17	Reserved.										
16	<b>PinSts.</b> Read-only. Reset: X. 0=The pin is low. 1=The pin is high. This bit is not affected by the debounce logic.										
15:13	<b>WakeCntrl[2:0].</b> Read-write. Reset: XXXb. Wake control.										
	<b>ValidValues:</b>										
	<table> <tr> <th>Bit</th><th>Description</th></tr> <tr> <td>[0]</td><td>1=Enables wake in S0i3 state.</td></tr> <tr> <td>[1]</td><td>1=Enables wake in S3 state (SLP_TYP == 3 and !G0_State).</td></tr> <tr> <td>[2]</td><td>1=Enables wake in S4/S5 state ((SLP_TYP == (4  5)) &amp;&amp; !G0_State).</td></tr> </table>	Bit	Description	[0]	1=Enables wake in S0i3 state.	[1]	1=Enables wake in S3 state (SLP_TYP == 3 and !G0_State).	[2]	1=Enables wake in S4/S5 state ((SLP_TYP == (4  5)) && !G0_State).		
Bit	Description										
[0]	1=Enables wake in S0i3 state.										
[1]	1=Enables wake in S3 state (SLP_TYP == 3 and !G0_State).										
[2]	1=Enables wake in S4/S5 state ((SLP_TYP == (4  5)) && !G0_State).										
12:11	<b>InterruptEnable[1:0].</b> Read-write. Reset: XXb. Enable interrupt status and delivery.										
	<b>ValidValues:</b>										
	<table> <tr> <th>Bit</th><th>Description</th></tr> <tr> <td>[0]</td><td>1=Enable interrupt status.</td></tr> <tr> <td>[1]</td><td>1=Enable interrupt delivery.</td></tr> </table>	Bit	Description	[0]	1=Enable interrupt status.	[1]	1=Enable interrupt delivery.				
Bit	Description										
[0]	1=Enable interrupt status.										
[1]	1=Enable interrupt delivery.										
10:9	<b>ActiveLevel.</b> Read-write. Reset: XXb.										
	<b>ValidValues:</b>										
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>Active high.</td></tr> <tr> <td>1h</td><td>Active low.</td></tr> <tr> <td>2h</td><td>Active on both of edges if LevelTrig == 0.</td></tr> <tr> <td>3h</td><td>Reserved.</td></tr> </table>	Value	Description	0h	Active high.	1h	Active low.	2h	Active on both of edges if LevelTrig == 0.	3h	Reserved.
Value	Description										
0h	Active high.										
1h	Active low.										
2h	Active on both of edges if LevelTrig == 0.										
3h	Reserved.										
8	<b>LevelTrig.</b> Read-write. Reset: X. 0=Edge trigger. 1=Level trigger.										
7:0	Reserved.										

### 9.2.12 Secure Digital (SD) Controller

SD does not have a dedicated interface, but is multiplexed with GPIO pins. Because of this, it is speed limited and not expected to run faster than 50 MHz.

#### 9.2.12.1 SD Configuration Registers (SD)

D14F6x000 (FCH::SD::SD_PCI_DEV_VEN_ID)	
Read-only. Reset: 7906_1022h.	
_aliasHOST; D14F6x000	
Bits	Description
31:16	<b>DEV_ID.</b> Read-only. Reset: 7906h. Device ID.
15:0	<b>VEND_ID.</b> Read-only. Reset: 1022h. Vendor ID.
D14F6x004 (FCH::SD::SD_PCI_CMD_STS)	
_aliasHOST; D14F6x004	
Bits	Description
31	<b>Det_Perr.</b> Read-only. Reset: 0. This bit is set by the device whenever it detects a parity error, even if parity error handling is disabled (as controlled by Parity_EN bit).
30	<b>Sig_SysErr.</b> Read-only. Reset: 0. 0=SERR# not asserted. 1=SERR# asserted. Set whenever the device asserts SERR#.
29	<b>Rec_Mabort.</b> Read-only. Reset: 0. 0=Cycle not terminated with Master Abort. 1=Cycle terminated with Master Abort. This bit is set by a master device whenever its transaction (except for Special Cycle) is terminated with

	Master Abort.
28	<b>Rec_Tabort.</b> Read-only. Reset: 0. 0=PCI cycle not aborted. 1=PCI cycle is aborted by a PCI target. This bit is set by a master device whenever its transaction is terminated with Target-Abort.
27	<b>Sig_Tabort.</b> Read-only. Reset: 0. 0=Target Abort not signaled. 1=Target Abort signaled. This bit is set by a target device whenever it terminates a transaction with Target Abort.
26:25	<b>DEVSEL_Timing.</b> Read-only. Reset: Fixed,1h. Medium timing selected.
24	<b>Master_DPerr.</b> Read-only. Reset: 0. 0=No data parity error. 1=Detects PERR# asserted while acting as PCI master. Master Data Parity error.
23	<b>Fast_B2B_Cap.</b> Read-only. Reset: Fixed,0. 0=Disallow. 1=Allow. Allows Fast Back-to-Back capability.
22	<b>PCI_ST_Reserved2.</b> Read-only. Reset: Fixed,0. Reserved.
21	<b>En_66MHz.</b> Read-only. Reset: Fixed,1. indicating 66MHz capable.
20	<b>Cap_List.</b> Read-only. Reset: 1. indicating no Capabilities linked list available.
19	<b>Int_status.</b> Read-only. Reset: 0. Reflects the state of the interrupt in the device/function.
18:16	<b>PCI_ST_Reserved1.</b> Read-only. Reset: Fixed,0h. Reserved.
15:11	<b>PCI_CMD_Reserved2.</b> Read-write. Reset: Fixed,00h. Reserved.
10	<b>Int_Dis.</b> Read-write. Reset: 0. 0=Enables the assertion of the device/function's INTx# signal. 1=Disable. Interrupt disable.
9	<b>Fast_B2B_En.</b> Read-only. Reset: Fixed,0. 0=Fast back-to-back transactions allowed only to the same agent. 1=Fast back-to-back transactions not allowed. Hard-wired to 0, indicating fast back-to-back transactions to the same agent only are allowed.
8	<b>SERRB_En.</b> Read-write. Reset: 0. 0=Disable the SERR# driver. 1=Enable the SERR# driver. SERR# driver enable.
7	<b>PCI_CMD_Reserved1.</b> Read-only. Reset: Fixed,0. Reserved.
6	<b>Parity_En.</b> Read-write. Reset: 0. 0=Parity check disabled. 1=Device must take action when a parity error is detected. Parity error detect.
5	<b>VGA_Pal_Access.</b> Read-only. Reset: Fixed,0. 0=Disallow. 1=Allow. Allow snoop VGA palette cycles.
4	<b>MemWr_Inv_Cmd.</b> Read-only. Reset: Fixed,0. 0=Disallow. 1=Allow. Allow Memory Write and Invalidate command.
3	<b>Special_Cycle.</b> Read-only. Reset: Fixed,0. 0=Special cycle recognition disable. 1=Special cycle recognition enable. Special cycle recognition.
2	<b>Bus_Master.</b> Read-write. Reset: 0. 0=Disallow. 1=Allow. Allow the device to behave as a bus master.
1	<b>Mem_Space_Access.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. Memory Space access enable.
0	<b>IO_Space_Access.</b> Read-only. Reset: Fixed,0. 0=Disable. 1=Enable. IO Space access enable.

**D14F6x008 (FCH::SD::SD\_PCI\_REV\_ID)**

_aliasHOST; D14F6x008	
Bits	Description
31:24	<b>BC.</b> Read-write. Reset: 08h. Base Class Code. Hard-wired to 08h, indicating general peripheral.
23:16	<b>SC.</b> Read-write. Reset: 05h. Sub Class Code. Hard-wired to 05h, indicating SD host controller.
15:8	<b>PI.</b> Read-only. Reset: 01h. Programming Interface Code. Hard-wired to 01h for Standard host supporting DMA.
7:0	<b>Revision_ID.</b> Read-only. Reset: Fixed,01h. Revision ID.

**D14F6x00C (FCH::SD::SD\_PCI\_MISC)**

_aliasHOST; D14F6x00C	
Bits	Description
31:24	<b>BIST.</b> Read-only. Reset: Fixed,00h. Hard-wired to 00h, indicating no build-in BIST support.
23:16	<b>Header_Type.</b> Read-only. Reset: Fixed,80h. Bit[23] hard-wired to 1, indicating a single-function device. Bit[22:16] hard-wired to 00h.
15:11	<b>Latency_Timer.</b> Read-write. Reset: 00h. Latency_timer_HW bits[10:8] hard-wired to 000b, resulting in a timer granularity of at least eight clocks. This field specifies, in units of PCI bus clocks, the value of the Latency Timer for this PCI bus master.

10:8	<b>Latency_Timer_HW.</b> Read-only. Reset: Fixed,0h. Specifies the value of the Latency Timer in units of PCICLKs.
7:0	<b>Cache_Size.</b> Read-write. Reset: 00h. This Read/Write field specifies the system cacheline size in units of DWORDs and must be initialized to 00h.

**D14F6x010 (FCH::SD::SD\_PCI\_BAR)**

_aliasHOST; D14F6x010	
Bits	Description
31:8	<b>BAR.</b> Read-write. Reset: 00_0000h. Base Address. Specifies the upper 15 bits of the 32-bit starting base address.
7:4	<b>BAR_Reserved.</b> Read-only. Reset: Fixed,0h. Reserved.
3	<b>PM.</b> Read-only. Reset: Fixed,0. Prefetch memory. A constant value of 0 indicates that there is no support for memory prefetch.
2:1	<b>TP.</b> Read-only. Reset: 2h. 2'b10 indicates that the base register is 64-bit width
<b>ValidValues:</b>	
Value	Description
0h	32-bit width.
1h	Reserved.
2h	64-bit width.
3h	Reserved.
0	<b>IND.</b> Read-only. Reset: Fixed,0. Resource Type Indicator. A constant value of 0 indicates that the operational registers of the device are mapped into memory space of the main memory of the PC host system.

**D14F6x014 (FCH::SD::SD\_PCI\_UPPER\_BAR)**

Read-write. Reset: 0000_0000h.	
_aliasHOST; D14F6x014	
Bits	Description
31:0	<b>UBAR.</b> Read-write. Reset: 0000_0000h. Upper 32 bits of Base Address.

**D14F6x02C (FCH::SD::SD\_PCI\_SUB\_VEN\_SYS\_ID)**

Read,Write-once. Reset: 7806_1022h.	
_aliasHOST; D14F6x02C	
Bits	Description
31:16	<b>Sub_System_ID.</b> Read,Write-once. Reset: 7806h. Can only be written once by software.
15:0	<b>Sub_Vendor_ID.</b> Read,Write-once. Reset: 1022h. Can only be written once by software.

**D14F6x034 (FCH::SD::SD\_PCI\_CAP\_PTR)**

Read-only. Reset: 80h.	
_aliasHOST; D14F6x034	
Bits	Description
7:0	<b>CAP_PTR.</b> Read-only. Reset: 80h. The first pointer of the Capability block. To the Function Level Reset (FLR) register.

**D14F6x03C (FCH::SD::SD\_PCI\_INT\_LINE)**

Reset: 0000_0100h.	
_aliasHOST; D14F6x03C	
Bits	Description
31:24	<b>MAX_LAT.</b> Read-only. Reset: 00h. Hardwired to 00h to indicate no major requirements for the settings of the Latency Timers.
23:16	<b>MIN_GNT.</b> Read-only. Reset: 00h. Hardwired to 00h to indicate no major requirements for the settings of the Latency Timers.
15:8	<b>Int_Pin.</b> Read-write. Reset: 01h. Hard-wired to 01h, corresponding to using INTA#
7:0	<b>Int_Line.</b> Read-write. Reset: 00h. The Interrupt Line register used to communicate interrupt line routing information.

**D14F6x040 (FCH::SD::SLOT\_INFORMATION)**

Read-only.

\_aliasHOST; D14F6x040

Bits	Description
31:8	Reserved.
7	<b>Slot_Reserved2.</b> Read-only. Reset: Fixed,0. Reserved.
6:4	<b>Num_Of_Slot.</b> Read-only. Reset: Fixed,0h. Hardwired to 000h to indicate only 1 slot.
3	<b>Slot_Reserved1.</b> Read-only. Reset: Fixed,0. Reserved.
2:0	<b>First_BAR.</b> Read-only. Reset: 0h. Hardwired to 000b to indicate BAR0 (Base address 10h).

**D14F6x080 (FCH::SD::SD\_PCI\_MSI\_CAP\_HEADER)**

Read-only. Reset: 9005h.

\_aliasHOST; D14F6x080

Bits	Description
15:8	<b>CAP_NXT_PTR.</b> Read-only. Reset: 90h. Pointer to the next item in the capabilities list.
7:0	<b>CAP_ID.</b> Read-only. Reset: 05h. 05h indicates MSI.

**D14F6x082 (FCH::SD::SD\_PCI\_MSI\_CTRL)**

\_aliasHOST; D14F6x082

Bits	Description
15:8	<b>MSI_CTRL_Reserved.</b> Read-only. Reset: Fixed,00h. Reserved.
7	<b>Extend_Addr_En.</b> Read-only. Reset: 1. 0=Not capable. 1=Capable. 64-bit Address Capable.
6:4	<b>Mul_Msg_En.</b> Read-write. Reset: 0h. Multiple Message Enable.
3:1	<b>Mul_Msg_Cap.</b> Read-only. Reset: Fixed,0h. Multiple Message Capable.
0	<b>MSI_Enable.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. MSI Enable.

**D14F6x084 (FCH::SD::SD\_PCI\_MSI\_ADDR)**

\_aliasHOST; D14F6x084

Bits	Description
31:2	<b>Msg_Addr.</b> Read-write. Reset: 0000_0000h. Message Address.
1:0	<b>MSI_ADDR_Reserved.</b> Read-only. Reset: Fixed,0h. Reserved.

**D14F6x088 (FCH::SD::SD\_PCI\_MSI\_U\_ADDR)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; D14F6x088

Bits	Description
31:0	<b>Msg_Upper_Addr.</b> Read-write. Reset: 0000_0000h. Message Upper Address.

**D14F6x08C (FCH::SD::SD\_PCI\_MSI\_DATA)**

Read-write. Reset: 0000h.

\_aliasHOST; D14F6x08C

Bits	Description
15:0	<b>Msg_Data.</b> Read-write. Reset: 0000h. Message Data.

**D14F6x090 (FCH::SD::SD\_PCI\_PMC\_CAP\_HEADER)**

Read-only. Reset: 0001h.

\_aliasHOST; D14F6x090

Bits	Description
15:8	<b>CAP_NXT_PTR.</b> Read-only. Reset: 00h. Pointer to the next item in the capabilities list.
7:0	<b>CAP_ID.</b> Read-only. Reset: 01h. Power Management.

**D14F6x092 (FCH::SD::SD\_PCI\_PMC\_CAP)**

Read-only. Reset: Fixed,8003h.	
_aliasHOST; D14F6x092	
Bits	Description
15:11	<b>PME_Support.</b> Read-only. Reset: Fixed,10h. This 5-bit field indicates the power states in which the function may assert PME#. A value of 0 for any bits indicates that the function is not capable of asserting the PME# signal while in that power state.
10	<b>D2_Support.</b> Read-only. Reset: Fixed,0. D2 Power Management State.
9	<b>D1_Support.</b> Read-only. Reset: Fixed,0. D1 Power Management State.
8:6	<b>Aux_Current.</b> Read-only. Reset: Fixed,0h. 3.3V auxillary current requirements.
5	<b>DSI.</b> Read-only. Reset: Fixed,0. Device Specific Initialization.
4	<b>PMC_Reserved.</b> Read-only. Reset: Fixed,0. Reserved.
3	<b>PME_Clock.</b> Read-only. Reset: Fixed,0. If PME depends on PCI clock.
2:0	<b>PMC_Version.</b> Read-only. Reset: Fixed,3h. 011b indicates compliance to revision 1.2 of the PCI Power Management Interface Spec.

**D14F6x094 (FCH::SD::SD\_PCI\_PMCSR)**

_aliasHOST; D14F6x094	
Bits	Description
15	<b>PME_Status.</b> Read-write. Reset: 0. Set when the function normally asserts the PME# signal independent of the state of PME# Enable
14:13	<b>Data_Scale.</b> Read-only. Reset: Fixed,0h. Indicates the scaling factor to be used when interpreting the value of the Data register. The value and meaning of this field varies depending on which data value has been selected by the Data_Select field.
12:9	<b>Data_Select.</b> Read-write. Reset: 0h. Selects which data is to be reported through the Data register and Data_Scale field.
8	<b>PME_EN.</b> Read-write. Reset: 0. 0=PME# assertion is disabled. 1=Enable the function to assert PME#. PME# Enable.
7:4	<b>PMCSR_Reserved2.</b> Read-only. Reset: Fixed,0h. Reserved.
3	<b>No_Soft_Reset.</b> Read-only. Reset: Fixed,1. 1=Indicates that devices transitioning from D3hot to D0 because of PowerState commands do not perform an internal reset.
2	<b>PMCSR_Reserved1.</b> Read-only. Reset: Fixed,0. Reserved.
1:0	<b>PowerState.</b> Read-write. Reset: 0h. Initiates Power State.

**D14F6x096 (FCH::SD::SD\_PCI\_PMC\_BSE)**

Read-only. Reset: Fixed,00h.	
_aliasHOST; D14F6x096	
Bits	Description
7	<b>BPCC_En.</b> Read-only. Reset: Fixed,0. Bus Power/Clock Control enable.
6	<b>B2_B3.</b> Read-only. Reset: Fixed,0. B2/B3 support for D3hot
5:0	<b>PMC_BSE_Reserved.</b> Read-only. Reset: Fixed,00h. Reserved.

**D14F6x097 (FCH::SD::SD\_PCI\_PMC\_DATA)**

Read-only. Reset: 00h.	
_aliasHOST; D14F6x097	
Bits	Description
7:0	<b>PMC_Data.</b> Read-only. Reset: 00h. Reports the state dependent data requested by the Data_Select field. The value of this register is scaled by the value reported by the Data_Scale field.

**9.2.12.2 SD Host Controller Memory Mapped Registers (SDHC)**

**SDHCx000 (FCH::SD::MMIO::SDHC\_SYS\_ADDR)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; SDHCx000; SDHC={FCH::SD::SD\_PCI\_BAR\_aliasHOST[BAR], 00h}

Bits	Description
31:16	<b>SYS_ADDR1</b> . Read-write. Reset: 0000h. Upper bits. By updating this register, DMA_WAIT will be cleared. System Address. It indicates system memory address for the DMA. When DMA transfer detects the DMA Buffer Boundary specified by FCH::SD::MMIO::SDHC_BLK_CS[DMA_BUF_BNDRY], the SD controller asserts DMA_WAIT. Also, the SD controller generates a DMA interrupt at this time in the case that corresponding bits in the Normal Interrupt Status Enable register and Normal Interrupt Signal Enable register are set. While ADMA is enabled, this register will not be used.
15:0	<b>SYS_ADDR0</b> . Read-write. Reset: 0000h. Lower bits. By updating this register, DMA_WAIT will be cleared.

**SDHCx004 (FCH::SD::MMIO::SDHC\_BLK\_CS)**

\_aliasHOST; SDHCx004; SDHC={FCH::SD::SD\_PCI\_BAR\_aliasHOST[BAR], 00h}

Bits	Description																		
31:16	<b>BLK_CNT</b> . Read-write. Reset: 0000h. Block count. It indicates block count of multiple data transfer. It is enabled when FCH::SD::MMIO::SDHC_CMD_TRN[BLK_CNT_EN] == 1. It is decremented after each block data transmission. During infinite data transmission, the setting of this bit is meaningless.																		
15	Reserved.																		
14:12	<b>DMA_BUF_BNDRY</b> . Read-write. Reset: 0h. Host DMA buffer boundary. Indicates the contiguous buffer size in the system memory. When this boundary is reached internally, a DMA interrupt will be generated. <b>ValidValues:</b>																		
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>4K bytes</td></tr> <tr> <td>1h</td><td>8K bytes</td></tr> <tr> <td>2h</td><td>16K bytes</td></tr> <tr> <td>3h</td><td>32K bytes</td></tr> <tr> <td>4h</td><td>64K bytes</td></tr> <tr> <td>5h</td><td>128K bytes</td></tr> <tr> <td>6h</td><td>256K bytes</td></tr> <tr> <td>7h</td><td>512K bytes</td></tr> </table>	Value	Description	0h	4K bytes	1h	8K bytes	2h	16K bytes	3h	32K bytes	4h	64K bytes	5h	128K bytes	6h	256K bytes	7h	512K bytes
Value	Description																		
0h	4K bytes																		
1h	8K bytes																		
2h	16K bytes																		
3h	32K bytes																		
4h	64K bytes																		
5h	128K bytes																		
6h	256K bytes																		
7h	512K bytes																		
11:0	<b>BLK_SIZE</b> . Read-write. Reset: 000h. Transfer data length (Max. block size is 2K bytes). When FCH::SD::MMIO::SDHC_CE_ATA_CTRL[CE_ATA_EN] == 1, a value of 0x000 indicates a block size of 4K bytes.																		

**SDHCx008 (FCH::SD::MMIO::SDHC\_CMD\_ARG)**

Read-write. Reset: 0000\_0000h.

\_aliasHOST; SDHCx008; SDHC={FCH::SD::SD\_PCI\_BAR\_aliasHOST[BAR], 00h}

Bits	Description
31:16	<b>ARGUMENT1</b> . Read-write. Reset: 0000h. Upper bits. Command Argument. Command arguments specified as bits[39:8] of the command format.
15:0	<b>ARGUMENT0</b> . Read-write. Reset: 0000h. Lower bits.

**SDHCx00C (FCH::SD::MMIO::SDHC\_CMD\_TRN)**

\_aliasHOST; SDHCx00C; SDHC={FCH::SD::SD\_PCI\_BAR\_aliasHOST[BAR], 00h}

Bits	Description		
31:30	Reserved.		
29:24	<b>CMD_IDX</b> . Read-write. Reset: 00h. Command index.		
23:22	<b>CMD_TYPE</b> . Read-write. Reset: 0h. Command type. <b>ValidValues:</b>		
	<table> <tr> <th>Value</th><th>Description</th></tr> </table>	Value	Description
Value	Description		

	0h	Normal.
	1h	Suspend CMD52 for writing BR in CCCR.
	2h	Resume CMD52 for writing Func Sel in CCCR.
	3h	Abort CMD12 (SD Memory) or Abort CMD52 (SDIO).
21	<b>DATA_PRSENT</b> . Read-write. Reset: 0. 0=No data. 1=Data. Data present select. Indicates that data is present and will be transferred on the DAT line. When a command is issued with this bit enabled, the internal buffer will be cleared.	
20	<b>CMD_IDX_CHK_EN</b> . Read-write. Reset: 0. 0=Disable. 1=Enable. Command index check enable.	
19	<b>CRC_CHK_EN</b> . Read-write. Reset: 0. 0=Disable. 1=Enable. Command CRC check enable.	
18	Reserved.	
17:16	<b>RESP_TYPE</b> . Read-write. Reset: 0h. Response type select.	
	<b>Valid Values:</b>	
	<b>Value</b>	<b>Description</b>
	0h	No response.
	1h	Response length of 136 bits.
	2h	Response length of 48 bits with no busy.
	3h	Response length of 48 bits with busy.
15:6	Reserved.	
5	<b>MULTI_BLK</b> . Read-write. Reset: 0. 0=Single block. 1=Multiple blocks. Multiple/Single block select.	
4	<b>DATA_DIR</b> . Read-write. Reset: 0. 0=Write. 1=Read. Data transfer direction.	
3	Reserved.	
2	<b>ACMD12_EN</b> . Read-write. Reset: 0. 0=Disable. 1=Enable. Auto CMD12 enable. It is not valid when FCH::SD::MMIO::SDHC_CMD_TRN[BLK_CNT_EN] == 0.	
1	<b>BLK_CNT_EN</b> . Read-write. Reset: 0. 0=Disable. 1=Enable. Block count enable.	
0	<b>DMA_EN</b> . Read-write. Reset: 0. 0=Disable. 1=Enable. DMA enable.	

**SDHCx010 (FCH::SD::MMIO::SDHC\_RESP1\_0)**

Read-only. Reset: 0000\_0000h.

\_aliasHOST; SDHCx010; SDHC={FCH::SD::SD\_PCI\_BAR\_aliasHOST[BAR], 00h}

Bits	Description
31:16	<b>RESPONSE1</b> . Read-only. Reset: 0000h. Response. R[39:24] of the response is saved in this field. The value is preseeded until the next response.
15:0	<b>RESPONSE0</b> . Read-only. Reset: 0000h. R[23:8] of the response is saved in this field.

**SDHCx014 (FCH::SD::MMIO::SDHC\_RESP3\_2)**

Read-only. Reset: 0000\_0000h.

\_aliasHOST; SDHCx014; SDHC={FCH::SD::SD\_PCI\_BAR\_aliasHOST[BAR], 00h}

Bits	Description
31:16	<b>RESPONSE3</b> . Read-only. Reset: 0000h. Response. R[71:56] of response is saved in this field. The value is preseeded until the next response.
15:0	<b>RESPONSE2</b> . Read-only. Reset: 0000h. R[55:40] of response is saved in this field.

**SDHCx018 (FCH::SD::MMIO::SDHC\_RESP5\_4)**

Read-only. Reset: 0000\_0000h.

\_aliasHOST; SDHCx018; SDHC={FCH::SD::SD\_PCI\_BAR\_aliasHOST[BAR], 00h}

Bits	Description
31:16	<b>RESPONSE5</b> . Read-only. Reset: 0000h. Response. R[103:88] of response is saved in this field. The value is preseeded until the next response.
15:0	<b>RESPONSE4</b> . Read-only. Reset: 0000h. R[87:72] of response is saved in this field.

**SDHCx01C (FCH::SD::MMIO::SDHC\_RESP7\_6)**



Read-only. Reset: 0000_0000h.	
_aliasHOST; SDHCx01C; SDHC={FCH::SD::SD_PCI_BAR_aliasHOST[BAR] , 00h}	
Bits	Description
31:16	<b>RESPONSE7.</b> Read-only. Reset: 0000h. Response. R[127:120] of response or R[39:24] of Auto CMD12 response is saved in this field. The value is preseved until the next response.
15:0	<b>RESPONSE6.</b> Read-only. Reset: 0000h. R[119:104] of response or R[23:8] of Auto CMD12 response is saved in this field.

**SDHCx020 (FCH::SD::MMIO::SDHC\_BUFFER)**

Read-write. Reset: 0000_0000h.	
_aliasHOST; SDHCx020; SDHC={FCH::SD::SD_PCI_BAR_aliasHOST[BAR] , 00h}	
Bits	Description
31:16	<b>BUFF_DATA1.</b> Read-write. Reset: 0000h. Data buffer. Upper bits. Data will be accessed through this register. Data which exceeds the size designated by FCH::SD::MMIO::SDHC_BLK_CS[BLK_SIZE] will not be written in the data buffer.
15:0	<b>BUFF_DATA0.</b> Read-write. Reset: 0000h. Lower bits.

**SDHCx024 (FCH::SD::MMIO::SDHC\_PRSENT\_STATE)**

Read-only.	
_aliasHOST; SDHCx024; SDHC={FCH::SD::SD_PCI_BAR_aliasHOST[BAR] , 00h}	
Bits	Description
31:20	Reserved.
19	<b>WP_LEVEL.</b> Read-only. Reset: 0. 0=Write protected. 1=Write enable. Write Protect switch Level.
18	<b>CD_LEVEL.</b> Read-only. Reset: 0. 0=No card present. 1=Card present. Card Detect pin level. This bit is used for testing.
17	Reserved.
16	<b>CARD_INS.</b> Read-only. Reset: 0. 0=No card inserted or debouncing state or resetting. 1=Card inserted. Card inserted.
15:12	Reserved.
11	<b>BUF_RD_EN.</b> Read-only. Reset: 0. 0=Read disable. 1=Read enable. Buffer Read enable. Indicates buffer is ready for reading.
10	<b>BUF_WR_EN.</b> Read-only. Reset: 0. 0=Write disable. 1=Write enable. Buffer Write enable. Indicates buffer is ready for writing.
9	<b>RD_TX_ACTIVE.</b> Read-only. Reset: 0. 0=No data transferring. 1=Read data transferring. Read Transfer active. Indicates duration of read data transfer.
8	<b>WR_TX_ACTIVE.</b> Read-only. Reset: 0. 0=No data transferring. 1=Write data transferring. Write Transfer active. Indicates duration of write data transfer.
7:3	Reserved.
2	<b>DAT_LINE_ACTIVE.</b> Read-only. Reset: 0. 0=DAT line inactive. 1=DAT line active. DAT Line Active. Indicates DAT line on SD Bus is active.
1	<b>CMD_INHIB_DAT.</b> Read-only. Reset: 0. 0=Can issue commands which uses the DAT line. 1=Cannot issue any commands which uses the DAT line. Command inhibit (DAT). Indicates that commands which use also use the DAT line can be issued.
0	<b>CMD_INHIB_CMD.</b> Read-only. Reset: 0. 0=Can issue commands which use CMD line. 1=Cannot issue any commands. Command Inhibit (CMD). Indicates that commands which use only the CMD line can be issued.

**SDHCx028 (FCH::SD::MMIO::SDHC\_CTRL1)**

_aliasHOST; SDHCx028; SDHC={FCH::SD::SD_PCI_BAR_aliasHOST[BAR] , 00h}	
Bits	Description
31:27	Reserved.
26	<b>SD_REM_WAKE_EN.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. SD Card removal wakeup.
25	<b>SD_INS_WAKE_EN.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. SD Card insertion wakeup.

24	<b>SD_INT_WAKE_EN.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. SD Card interrupt wakeup.										
23:20	Reserved.										
19	<b>BG_INT_EN.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. Interrupt at Block Gap. Enable interrupt detection during 4-bit block transmission.										
18	<b>READ_WAIT_EN.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. Read Wait Control. Indicates Read Wait will be inserted when needed.										
17	<b>CONT_REQ.</b> Read-write. Reset: 0. 0=No effect. 1=Restart. Continue request. Writing 1 to this bit triggers a restart of halted data transaction with the current register setting. Once this bit is 1, the internal buffer is cleared and the data transfer sequence will be restarted.										
16	<b>BG_STOP_REQ.</b> Read-write. Reset: 0. 0=Transfer. 1=Stop. Stop at Block Gap request. Writing 1 to this bit triggers the halting of the current data transfer after the next block gap. For using this request, the Read Wait function is necessary in the Read transaction. Even if FCH::SD::MMIO::SDHC_CMD_TRN[ACMD12_EN] == 1, Auto CMD12 is not issued in the case of this bit being set to 1. This bit is cleared by not only Writing 0 to this bit, but also issuing abort commands.										
15:12	Reserved.										
11:9	<b>SD_BUS_VOLTAGE.</b> Read-write. Reset: 0h. SD bus voltage.										
	<b>ValidValues:</b>										
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>4h-0h</td><td>Reserved.</td></tr> <tr> <td>5h</td><td>1.8V</td></tr> <tr> <td>6h</td><td>3.0V</td></tr> <tr> <td>7h</td><td>3.3V (Not supported.)</td></tr> </table>	Value	Description	4h-0h	Reserved.	5h	1.8V	6h	3.0V	7h	3.3V (Not supported.)
Value	Description										
4h-0h	Reserved.										
5h	1.8V										
6h	3.0V										
7h	3.3V (Not supported.)										
8	<b>SD_BUS_EN.</b> Read-write. Reset: 0. 0=Off. 1=On. SD bus power. When card is removed, this bit is cleared automatically.										
7:6	Reserved.										
5	<b>MMC_WIDTH.</b> Read-write. Reset: 0. 0=Use width set by FCH::SD::MMIO::SDHC_CTRL1[DAT_TX_WIDTH]. 1=8-bit. Extended data transfer width (MMC).										
4:3	<b>DMA_SELECT.</b> Read-write. Reset: 0h. DMA select. Valid only when DMA is enabled.										
	<b>ValidValues:</b>										
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>No DMA or SDMA selected.</td></tr> <tr> <td>1h</td><td>Reserved.</td></tr> <tr> <td>2h</td><td>32-bit ADMA.</td></tr> <tr> <td>3h</td><td>64-bit ADMA.</td></tr> </table>	Value	Description	0h	No DMA or SDMA selected.	1h	Reserved.	2h	32-bit ADMA.	3h	64-bit ADMA.
Value	Description										
0h	No DMA or SDMA selected.										
1h	Reserved.										
2h	32-bit ADMA.										
3h	64-bit ADMA.										
2	<b>HIGH_SPEED_EN.</b> Read-write. Reset: 0. 0=Normal speed. 1=High speed. High Speed enable. When disabled, SD controller outputs commands and data on the falling edge of the SD clock (Up to 25MHz SD clock can be supported). When enabled, SD controller outputs commands and data on the rising edge of the SD clock (Up to 50MHz SD clock can be supported).										
1	<b>DAT_TX_WIDTH.</b> Read-write. Reset: 0. 0=1-bit. 1=4-bit. Data transfer width.										
0	<b>LED_CTRL.</b> Read-write. Reset: 0. 0=Off. 1=On. LED control. Drives the LED_ON output.										

**SDHCx02C (FCH::SD::MMIO::SDHC\_CTRL2)**

\_aliasHOST; SDHCx02C; SDHC={FCH::SD::SD\_PCI\_BAR\_aliasHOST[BAR] , 00h}

Bits	Description
31:27	Reserved.
26	<b>SOFT_RST_DAT.</b> Read-write. Reset: 0.
	<b>Description:</b> Software reset for the DAT Line. The following registers will be cleared: <ul style="list-style-type: none"> <li>- Buffer Data Port Register (Buffer is cleared)</li> <li>- (Present State Register)</li> <li>- Buffer Read Enable</li> </ul>

	<ul style="list-style-type: none"> <li>- Buffer Write Enable</li> <li>- Read Transfer Active</li> <li>- Write Transfer Active</li> <li>- DAT Line Active</li> <li>- Command Inhibit (DAT)</li> <li>- (Block Gap Control Register)</li> <li>- Continue Request</li> <li>- Stop At Block Gap Request</li> <li>- (Normal Interrupt Status Register)</li> <li>- Buffer Read Ready</li> <li>- Buffer Write Ready</li> <li>- Block Gap Event</li> <li>- Transfer Complete</li> </ul>																		
25	<b>SOFT_RST_CMD.</b> Read-write. Reset: 0. <b>Description:</b> Software reset for the CMD Line. The following registers will be cleared: <ul style="list-style-type: none"> <li>- (Present State Register)</li> <li>- Command Inhibit (CMD)</li> <li>- (Normal Interrupt Status Register)</li> <li>- Command Complete</li> </ul>																		
24	<b>SOFT_RST_ALL.</b> Read-write. Reset: 0. <b>Description:</b> Software reset for all. The following registers will not be cleared: <ul style="list-style-type: none"> <li>- CMD Line Signal Level</li> <li>- DAT[3:0] Line Signal Level</li> <li>- Write Protect Switch Pin Level</li> <li>- Card Detect Pin Level</li> <li>- Card State Stable</li> <li>- Card Inserted</li> <li>- All bits in the Capabilities Register</li> <li>- All bits in the Maximum Current Capabilities Register</li> </ul>																		
23:20	Reserved.																		
19:16	<b>DATA_TO_CNT.</b> Read-write. Reset: 0h. Data Timeout Counter Value. By using this counter value, DAT line timeouts are detected. <b>ValidValues:</b> <table border="1"> <thead> <tr> <th>Value</th><th>Description</th></tr> </thead> <tbody> <tr> <td>0h</td><td>2<sup>13</sup></td></tr> <tr> <td>1h</td><td>2<sup>14</sup></td></tr> <tr> <td>Dh-2h</td><td>Omitted.</td></tr> <tr> <td>Eh</td><td>2<sup>27</sup></td></tr> <tr> <td>Fh</td><td>Reserved.</td></tr> </tbody> </table>	Value	Description	0h	2 <sup>13</sup>	1h	2 <sup>14</sup>	Dh-2h	Omitted.	Eh	2 <sup>27</sup>	Fh	Reserved.						
Value	Description																		
0h	2 <sup>13</sup>																		
1h	2 <sup>14</sup>																		
Dh-2h	Omitted.																		
Eh	2 <sup>27</sup>																		
Fh	Reserved.																		
15:8	<b>SDCLK_DIV.</b> Read-write. Reset: 00h. SDCLK frequency select. If multiple bits are set, the most significant bit will be selected. <b>ValidValues:</b> <table border="1"> <thead> <tr> <th>Bit</th><th>Description</th></tr> </thead> <tbody> <tr> <td>[0]</td><td>0=Divide by 1. 1=Divide by 2.</td></tr> <tr> <td>[1]</td><td>1=Divide by 4.</td></tr> <tr> <td>[2]</td><td>1=Divide by 8.</td></tr> <tr> <td>[3]</td><td>1=Divide by 16.</td></tr> <tr> <td>[4]</td><td>1=Divide by 32.</td></tr> <tr> <td>[5]</td><td>1=Divide by 64.</td></tr> <tr> <td>[6]</td><td>1=Divide by 128.</td></tr> <tr> <td>[7]</td><td>1=Divide by 256.</td></tr> </tbody> </table>	Bit	Description	[0]	0=Divide by 1. 1=Divide by 2.	[1]	1=Divide by 4.	[2]	1=Divide by 8.	[3]	1=Divide by 16.	[4]	1=Divide by 32.	[5]	1=Divide by 64.	[6]	1=Divide by 128.	[7]	1=Divide by 256.
Bit	Description																		
[0]	0=Divide by 1. 1=Divide by 2.																		
[1]	1=Divide by 4.																		
[2]	1=Divide by 8.																		
[3]	1=Divide by 16.																		
[4]	1=Divide by 32.																		
[5]	1=Divide by 64.																		
[6]	1=Divide by 128.																		
[7]	1=Divide by 256.																		

7:3	Reserved.
2	<b>SDCLK_EN</b> . Read-write. Reset: 0. 0=Disable. 1=Enable. SD clock enable. SDCLK frequency select can be changed when this bit is 0. When card is removed, this bit is cleared to 0 automatically.
1	<b>SYSCLK_STABLE</b> . Read-only. Reset: 0. 0=Unstable. 1=Stable. Internal clock stable.
0	<b>SYSCLK_EN</b> . Read-write. Reset: 0. 0=Disable. 1=Enable. Internal clock enable.

**SDHCx030 (FCH::SD::MMIO::SDHC\_INT\_STATUS)**

Read-only.

\_aliasHOST; SDHCx030; SDHC={FCH::SD::SD\_PCI\_BAR\_aliasHOST[BAR], 00h}

Bits	Description
31:30	Reserved.
29	<b>CE_ATA_ERR</b> . Read-only. Reset: 0. CE-ATA error.
28	<b>SDMA_ERR</b> . Read-only. Reset: 0. SDMA error.
27:26	Reserved.
25	<b>ADMA_ERR</b> . Read-only. Reset: 0. ADMA error.
24	<b>ACMD12_ERR</b> . Read-only. Reset: 0. Auto CMD12 error. Logical OR of Auto CMD12 Error Status Register.
23	Reserved.
22	<b>DAT_END_ERR</b> . Read-only. Reset: 0. Data End error.
21	<b>DAT_CRC_ERR</b> . Read-only. Reset: 0. Data CRC error.
20	<b>DAT_TO_ERR</b> . Read-only. Reset: 0. Data Timeout error.
19	<b>CMD_IDX_ERR</b> . Read-only. Reset: 0. Command Index error. Mismatch of Command Index and index of response.
18	<b>CMD_END_ERR</b> . Read-only. Reset: 0. Command End Bit error.
17	<b>CMD_CRC_ERR</b> . Read-only. Reset: 0. Command CRC Error. If (CMD_TO_ERR && CMD_CRC_ERR) == 1, this indicates a Command Conflict Error.
16	<b>CMD_TO_ERR</b> . Read-only. Reset: 0. Command Timeout error. Response not returned within 128 SDCLK cycles.
15	<b>ERROR</b> . Read-only. Reset: 0. Error interrupt.
14:9	Reserved.
8	<b>SDIO</b> . Read-only. Reset: 0. SDIO Card interrupt. Writing 1 to this register does not clear this bit. For clearing this bit, the interrupt factor of SDIO cards should be cleared. The value of this bit is latched internally as long as the Card Interrupt bit (D08) in the Normal Interrupt Status Enable Register is 1.
7	<b>CARD_REM</b> . Read-only. Reset: 0. Card removal.
6	<b>CARD_INS</b> . Read-only. Reset: 0. Card insertion.
5	<b>BUF_RD_RDY</b> . Read-only. Reset: 0. Buffer Read Ready. In case FCH::SD::MMIO::SDHC_CMD_TRN[ACMD12_EN] == 1 and last block has been transferred, Auto CMD12 will be issued prior to this bit being set to 1. Clearing this bit should be done before Reading the buffer because the SD controller has a dual buffer and the next Buffer Read Ready interrupt may occur immediately.
4	<b>BUF_WR_RDY</b> . Read-only. Reset: 0. Buffer Write Ready. Clearing this bit should be done before writing to the buffer because the SD controller has a dual buffer and the next Buffer Write Ready interrupt may occur immediately.
3	<b>DMA_EVT</b> . Read-only. Reset: 0. DMA interrupt. It is set when the internal counter reaches the value designated by the Host DMA Buffer Boundary. It should be cleared by the Host Driver after the System Address Register is updated.
2	<b>BLOCK_GAP_EVT</b> . Read-only. Reset: 0. Block gap event. It indicates the timing of next block gap, which was requested by the Stop At Block Gap Request. In the case of a Write transaction, this interrupt will be generated before busy completion.
1	<b>DAT_DONE</b> . Read-only. Reset: 0. 0=Data not done. 1=Data done. Data transfer complete. Indicates the timing for completion of data transaction, which includes the completion at the block gap by the Stop At Block Gap Request. When some errors are detected during data transaction, this bit will not be set. In case that FCH::SD::MMIO::SDHC_CMD_TRN[ACMD12_EN] == 1, Auto CMD12 will be issued prior to this bit being

	set to 1.
0	<b>CMD_DONE</b> . Read-only. Reset: 0. 0=Command not done. 1=Command done. Command complete. The end bit of the command response is received. In the case of commands with no response, the end of the command.

**SDHCx034 (FCH::SD::MMIO::SDHC\_INT\_MASK)**

_aliasHOST; SDHCx034; SDHC={FCH::SD::SD_PCI_BAR_aliasHOST[BAR], 00h}	
Bits	Description
31:30	Reserved.
29	<b>CE_ATA_ERR_MASK</b> . Read-write. Reset: 0. 0=Masked. 1=Enabled. CE-ATA Error interrupt mask.
28	<b>SDMA_ERR_MASK</b> . Read-write. Reset: 0. 0=Masked. 1=Enabled. SDMA Error interrupt mask.
27:26	Reserved.
25	<b>ADMA_ERR_MASK</b> . Read-write. Reset: 0. 0=Masked. 1=Enabled. ADMA Error interrupt mask.
24	<b>ACMD12_ERR_MASK</b> . Read-write. Reset: 0. 0=Masked. 1=Enabled. Auto CMD12 Error interrupt mask.
23	<b>CUR_LIM_ERR_MASK</b> . Read-write. Reset: 0. 0=Masked. 1=Enabled. Current Limit Error interrupt mask.
22	<b>DAT_END_ERR_MASK</b> . Read-write. Reset: 0. 0=Masked. 1=Enabled. Data End Bit Error interrupt mask.
21	<b>DAT_CRC_ERR_MASK</b> . Read-write. Reset: 0. 0=Masked. 1=Enabled. Data CRC Error interrupt mask.
20	<b>DAT_TO_ERR_MASK</b> . Read-write. Reset: 0. 0=Masked. 1=Enabled. Data Timeout Error interrupt mask.
19	<b>CMD_IDX_ERR_MASK</b> . Read-write. Reset: 0. 0=Masked. 1=Enabled. Command Index Error interrupt mask.
18	<b>CMD_END_ERR_MASK</b> . Read-write. Reset: 0. 0=Masked. 1=Enabled. Command End Bit Error interrupt mask.
17	<b>CMD_CRC_ERR_MASK</b> . Read-write. Reset: 0. 0=Masked. 1=Enabled. Command CRC Error interrupt mask.
16	<b>CMD_TO_ERR_MASK</b> . Read-write. Reset: 0. 0=Masked. 1=Enabled. Command Timeout Error interrupt mask.
15:9	Reserved.
8	<b>SDIO_MASK</b> . Read-write. Reset: 0. 0=Masked. 1=Enabled. Card interrupt mask.
7	<b>CARD_REM_MASK</b> . Read-write. Reset: 0. 0=Masked. 1=Enabled. Card Removal interrupt mask.
6	<b>CARD_INS_MASK</b> . Read-write. Reset: 0. 0=Masked. 1=Enabled. Card Insertion interrupt mask.
5	<b>BUF_RD_RDY_MASK</b> . Read-write. Reset: 0. 0=Masked. 1=Enabled. Buffer Read Ready interrupt mask.
4	<b>BUF_WR_RDY_MASK</b> . Read-write. Reset: 0. 0=Masked. 1=Enabled. Buffer Write Ready interrupt mask.
3	<b>DMA_EVT_MASK</b> . Read-write. Reset: 0. 0=Masked. 1=Enabled. DMA Interrupt interrupt mask.
2	<b>BLOCK_GAP_EVT_MASK</b> . Read-write. Reset: 0. 0=Masked. 1=Enabled. Block Gap Event interrupt mask.
1	<b>DAT_DONE_MASK</b> . Read-write. Reset: 0. 0=Masked. 1=Enabled. Transfer Complete interrupt mask.
0	<b>CMD_DONE_MASK</b> . Read-write. Reset: 0. 0=Masked. 1=Enabled. Command Complete interrupt mask.

**SDHCx038 (FCH::SD::MMIO::SDHC\_SIG\_MASK)**

_aliasHOST; SDHCx038; SDHC={FCH::SD::SD_PCI_BAR_aliasHOST[BAR], 00h}	
Bits	Description
31:30	Reserved.
29	<b>CE_ATA_ERR_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enabled. CE-ATA error.
28	<b>SDMA_ERR_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enabled. SDMA error.
27:26	Reserved.
25	<b>ADMA_ERR_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enabled. ADMA error.
24	<b>ACMD12_ERR_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enabled. Auto CMD12 error.
23	<b>CUR_LIM_ERR_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enabled. Current Limit error.
22	<b>DAT_END_ERR_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enabled. Data End Bit error.
21	<b>DAT_CRC_ERR_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enabled. Data CRC error.
20	<b>DAT_TO_ERR_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enabled. Data Timeout error.
19	<b>CMD_IDX_ERR_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enabled. Command Index error.
18	<b>CMD_END_ERR_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enabled. Command End Bit error.
17	<b>CMD_CRC_ERR_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enabled. Command CRC error.
16	<b>CMD_TO_ERR_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enabled. Command Timeout error.

15:9	Reserved.
8	<b>SDIO_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enabled. Card Interrupt.
7	<b>CARD_REM_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enabled. Card Removal.
6	<b>CARD_INS_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enabled. Card Insertion.
5	<b>BUF_RD_RDY_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enabled. Buffer Read Ready.
4	<b>BUF_WR_RDY_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enabled. Buffer Write Ready.
3	<b>DMA_EVT_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enabled. DMA Interrupt.
2	<b>BLOCK_GAP_EVT_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enabled. Block Gap Event.
1	<b>DAT_DONE_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enabled. Transfer Complete.
0	<b>CMD_DONE_EN</b> . Read-write. Reset: 0. 0=Masked. 1=Enabled. Command Complete.

**SDHCx03C (FCH::SD::MMIO::SDHC\_ACMD12\_ERR)**

Read-only.

\_aliasHOST; SDHCx03C; SDHC={FCH::SD::SD\_PCI\_BAR\_aliasHOST[BAR], 00h}

Bits	Description
31:8	Reserved.
7	<b>CMD_ERR</b> . Read-only. Reset: 0. Command not issued by Auto CMD12 error.
6:5	Reserved.
4	<b>INDEX_ERR</b> . Read-only. Reset: 0. Auto CMD12 Index error.
3	<b>END_ERR</b> . Read-only. Reset: 0. Auto CMD12 End Bit error.
2	<b>CRC_ERR</b> . Read-only. Reset: 0. Auto CMD12 CRC error.
1	<b>TO_ERR</b> . Read-only. Reset: 0. Auto CMD12 Timeout error.
0	<b>EXE_ERR</b> . Read-only. Reset: 0. Auto CMD12 Not Executed error.

**SDHCx040 (FCH::SD::MMIO::SDHC\_CAPABILITY)**

Read-only.

\_aliasHOST; SDHCx040; SDHC={FCH::SD::SD\_PCI\_BAR\_aliasHOST[BAR], 00h}

Bits	Description										
31:27	Reserved.										
26	<b>SUPPORT_1_8V</b> . Read-only. Reset: 0. 0=Not supported. 1=Supported. Voltage support for 1.8V.										
25	<b>SUPPORT_3_0V</b> . Read-only. Reset: 0. 0=Not supported. 1=Supported. Voltage support for 3.0V.										
24	<b>SUPPORT_3_3V</b> . Read-only. Reset: 1. 0=Not supported. 1=Supported. Voltage support for 3.3V.										
23	<b>SUS_RES_SUPPORT</b> . Read-only. Reset: 1. 0=Not supported. 1=Supported. Suspend and Resume support.										
22	<b>DMA_SUPPORT</b> . Read-only. Reset: 0. 0=Not supported. 1=Supported. DMA support.										
21	<b>HISPEED_SUPPORT</b> . Read-only. Reset: 1. 0=Not supported. 1=Supported. High Speed support.										
20	<b>ADMA_SUPPORT</b> . Read-only. Reset: 1. 0=Not supported. 1=Supported. ADMA support.										
19	<b>SYS_SUPPORT</b> . Read-only. Reset: 0. Fixed to 0 in this release.										
18	<b>MMC8_SUPPORT</b> . Read-only. Reset: 1. 0=Not supported. 1=Supported. Extended Media Bus Support (MMC).										
17:16	<b>MAX_BLK_LEN</b> . Read-only. Reset: 2h. Max block length.										
<b>ValidValues:</b>											
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>512 bytes.</td></tr> <tr> <td>1h</td><td>1K bytes.</td></tr> <tr> <td>2h</td><td>2K bytes.</td></tr> <tr> <td>3h</td><td>Reserved.</td></tr> </table>	Value	Description	0h	512 bytes.	1h	1K bytes.	2h	2K bytes.	3h	Reserved.
Value	Description										
0h	512 bytes.										
1h	1K bytes.										
2h	2K bytes.										
3h	Reserved.										
15:14	Reserved.										
13:8	<b>BASE_CLK_FREQ</b> . Read-only. Reset: 18h. Base clock frequency for SD clock.										
<b>ValidValues:</b>											
	<table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>00h</td><td>Get Information via another method.</td></tr> </table>	Value	Description	00h	Get Information via another method.						
Value	Description										
00h	Get Information via another method.										



	01h	1 MHz.
	3Eh-02h	<Value> MHz.
	3Fh	63 MHz.
7	<b>TO_CLK_UNIT</b> . Read-only. Reset: 1. 0=Clock units are in KHz. 1=Clock units are in MHz. Timeout clock Unit.	
6	Reserved.	
5:0	<b>TO_CLK_FREQ</b> . Read-only. Reset: 18h. Timeout clock frequency. See also TO_CLK_UNIT to get the units.	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	00h	Get Information via another method.
	01h	1 clock unit.
	3Eh-02h	<Value> clock units.
	3Fh	63 clock units.

**SDHCx048 (FCH::SD::MMIO::SDHC\_CURR\_CAPABILITY)**

Read-only.

\_aliasHOST; SDHCx048; SDHC={FCH::SD::SD\_PCI\_BAR\_aliasHOST[BAR] , 00h}

Bits	Description
31:24	Reserved.
23:16	<b>MAX_CURR_1_8V</b> . Read-only. Reset: 00h. Max current for 1.8V.
	<b>ValidValues:</b>
	<b>Value</b> <b>Description</b>
	00h Get Information via another method.
	01h 4 mA.
	FEh-02h (<Value> * 4) mA.
	FFh 1020 mA.
15:8	<b>MAX_CURR_3_0V</b> . Read-only. Reset: 00h. Max current for 3.0V.
	<b>ValidValues:</b>
	<b>Value</b> <b>Description</b>
	00h Get Information via another method.
	01h 4 mA.
	FEh-02h (<Value> * 4) mA.
	FFh 1020 mA.
7:0	<b>MAX_CURR_3_3V</b> . Read-only. Reset: 64h. Max current for 3.3V.
	<b>ValidValues:</b>
	<b>Value</b> <b>Description</b>
	00h Get Information via another method.
	01h 4 mA.
	FEh-02h (<Value> * 4) mA.
	FFh 1020 mA.

**SDHCx050 (FCH::SD::MMIO::SDHC\_FORCE\_EVT)**

\_aliasHOST; SDHCx050; SDHC={FCH::SD::SD\_PCI\_BAR\_aliasHOST[BAR] , 00h}

Bits	Description
31:30	Reserved.
29	<b>CE_ATA_ERR_FRC</b> . Write-only. Reset: 0. 0=No Interrupt. 1=Interrupt is generated. Force CE-ATA error.



28	<b>SDMA_ERR_FRC.</b> Write-only. Reset: 0. 0=No Interrupt. 1=Interrupt is generated. Force SDMA error.
27:26	Reserved.
25	<b>ADMA_ERR_FRC.</b> Write-only. Reset: 0. 0=No Interrupt. 1=Interrupt is generated. Force ADMA error.
24	<b>ACMD12_ERR_FRC.</b> Write-only. Reset: 0. 0=No Interrupt. 1=Interrupt is generated. Force Auto CMD12 error.
23	<b>CUR_LIM_ERR_FRC.</b> Write-only. Reset: 0. 0=No Interrupt. 1=Interrupt is generated. Force current limit error.
22	<b>DAT_END_ERR_FRC.</b> Write-only. Reset: 0. 0=No Interrupt. 1=Interrupt is generated. Force data end bit error.
21	<b>DAT_CRC_ERR_FRC.</b> Write-only. Reset: 0. 0=No Interrupt. 1=Interrupt is generated. Force data CRC error.
20	<b>DAT_TO_ERR_FRC.</b> Write-only. Reset: 0. 0=No Interrupt. 1=Interrupt is generated. Force data timeout error.
19	<b>CMD_IDX_ERR_FRC.</b> Write-only. Reset: 0. 0=No Interrupt. 1=Interrupt is generated. Force command index error.
18	<b>CMD_END_ERR_FRC.</b> Write-only. Reset: 0. 0=No Interrupt. 1=Interrupt is generated. Force command end bit error.
17	<b>CMD_CRC_ERR_FRC.</b> Write-only. Reset: 0. 0=No Interrupt. 1=Interrupt is generated. Force command CRC error.
16	<b>CMD_TO_ERR_FRC.</b> Write-only. Reset: 0. 0=No Interrupt. 1=Interrupt is generated. Force command timeout error.
15:8	Reserved.
7	<b>ACMD12_CMD_ERR_FRC.</b> Write-only. Reset: 0. 0=No Interrupt. 1=Interrupt is generated. Force command not issued by Auto CMD12 error.
6:5	Reserved.
4	<b>ACMD12_IDX_ERR_FRC.</b> Write-only. Reset: 0. 0=No Interrupt. 1=Interrupt is generated. Force Auto CMD12 index error.
3	<b>ACMD12_END_ERR_FRC.</b> Write-only. Reset: 0. 0=No Interrupt. 1=Interrupt is generated. Force Auto CMD12 end bit error.
2	<b>ACMD12_CRC_ERR_FRC.</b> Write-only. Reset: 0. 0=No Interrupt. 1=Interrupt is generated. Force Auto CMD12 CRC error.
1	<b>ACMD12_TO_ERR_FRC.</b> Write-only. Reset: 0. 0=No Interrupt. 1=Interrupt is generated. Force Auto CMD12 timeout error.
0	<b>ACMD12_EXE_ERR_FRC.</b> Write-only. Reset: 0. 0=No Interrupt. 1=Interrupt is generated. Force Auto CMD12 not executed error.

**SDHCx054 (FCH::SD::MMIO::SDHC\_ADMA\_ERR)**

Read-only.

\_aliasHOST; SDHCx054; SDHC={FCH::SD::SD\_PCI\_BAR\_aliasHOST[BAR] , 00h}

Bits	Description										
31:3	Reserved.										
2	<b>ADDR_LEN_MISMATCH.</b> Read-only. Reset: 0. 0=No error. 1=Error. <b>Description:</b> ADMA address length mismatch error. This error occurs in the following 2 cases: 1. While Block Count Enable is being set, the total data length specified by the descriptor table is different from that specified by the Block Count and Block Length. 2. The total data length can not be divided by the block length.										
1:0	<b>ADMA_STATE.</b> Read-only. Reset: 0h. ADMA states when an error occurred. <b>ValidValues:</b> <table> <tr> <th>Value</th><th>Description</th></tr> <tr> <td>0h</td><td>Stop DMA.</td></tr> <tr> <td>1h</td><td>Fetch descriptor.</td></tr> <tr> <td>2h</td><td>Change address.</td></tr> <tr> <td>3h</td><td>Transfer data.</td></tr> </table>	Value	Description	0h	Stop DMA.	1h	Fetch descriptor.	2h	Change address.	3h	Transfer data.
Value	Description										
0h	Stop DMA.										
1h	Fetch descriptor.										
2h	Change address.										
3h	Transfer data.										

**SDHCx058 (FCH::SD::MMIO::SDHC\_ADMA\_SAD)**

Read-write. Reset: 0000_0000h.	
_aliasHOST; SDHCx058; SDHC={FCH::SD::SD_PCI_BAR_aliasHOST[BAR] , 00h}	
Bits	Description
31:16	<b>ADMA_SYS_ADDR1</b> . Read-write. Reset: 0000h. Upper bits of ADMA system address. Before the ADMA data transfer, the descriptor address should be set by the Host driver. This address needs to be set with 4-byte alignment, since the descriptor table is formatted for 32-bit (4 bytes) information.
15:0	<b>ADMA_SYS_ADDR0</b> . Read-write. Reset: 0000h. Lower bits of ADMA system address.

**SDHCx080 (FCH::SD::MMIO::SDHC\_CE\_ATA\_CTRL)**

_aliasHOST; SDHCx080; SDHC={FCH::SD::SD_PCI_BAR_aliasHOST[BAR] , 00h}	
Bits	Description
31:5	Reserved.
4	<b>CE_ATA_SUPPORT</b> . Read-only. Reset: 1. 0=Unsupported. 1=Supported. CE-ATA support capability. When supported, bits[3:0] can be written.
3	<b>CE_ATA_EN</b> . Read-write. Reset: 0. 0=Disable. 1=Enable. CE-ATA enable. When enabled bits[2:0] can be written.
2	<b>P_DRIVE_DIS</b> . Read-write. Reset: 0. 0=Enable. 1=Disable. P-Drive disable.
1	<b>CMD_COMP_DIS_EN</b> . Read-write. Reset: 0. 0=Wait after command. 1=Issued after command. Command Completion Signal Disable enable.
0	<b>CMD_COMP_SIG_EN</b> . Read-write. Reset: 0. 0=Disable. 1=Enable. Command Completion Signal enable. When the Command Completion Signal is enabled, the FCH::SD::MMIO::SDHC_CMD_TRN[ACMD12_EN] bit in the Transfer Mode register becomes of no use.

**SDHCx090 (FCH::SD::MMIO::SDHC\_DAT3\_CARD\_DET)**

_aliasHOST; SDHCx090; SDHC={FCH::SD::SD_PCI_BAR_aliasHOST[BAR] , 00h}	
Bits	Description
31:5	Reserved.
4	<b>DEBOUNCING_EN</b> . Read-write. Reset: 1. 0=Disable. 1=Enable. Debouncing enable. This bit should be cleared before starting command or data transfer if debouncing state is not needed in the error cases.
3	<b>DEBOUNCING_STATUS</b> . Read-only. Reset: 0. 0=Not debouncing. 1=Debouncing. Debouncing State status. Indicates the card is debouncing. Should not issue any commands during debouncing.
2	<b>CARD_DETECT_SEL</b> . Read-only. Reset: 0. 0=Standard Card Detect mode. 1=DAT3 Card Detect mode. Card Detect Select status.
1	<b>DAT3_DETECT_EN</b> . Read-write. Reset: 1. 0=Disable. 1=Enable. DAT3 Card Detect control. Disable this bit before writing to the Command Register. After Command or Transfer complete, this bit can be re-enabled. During command and data transfer, enabling DAT3 Card Detection is prohibited. This bit is fixed to 1 in Standard Card Detect mode.
0	<b>DAT3_CARD_INS</b> . Read-only. Reset: 0. 0=No card inserted. 1=Card inserted. DAT3 Card inserted.

**SDHCx0FC (FCH::SD::MMIO::SDHC\_VER\_SLOT)**

Read-only.	
_aliasHOST; SDHCx0FC; SDHC={FCH::SD::SD_PCI_BAR_aliasHOST[BAR] , 00h}	
Bits	Description
31:24	<b>VENDOR_VERSION</b> . Read-only. Reset: C3h. Vendor Version.
23:16	<b>SPEC_VERSION</b> . Read-only. Reset: 02h. Specification Version.
15:8	Reserved.
7:0	<b>SLOT_INTRPT</b> . Read-only. Reset: 00h. Interrupt signal for each slot. The value of XSLT_INT[7:0] inputs, which indicates the logical OR of Interrupt signal and Wakeup signal, are inverted and referred by this register. In case of multiple slots, Interrupt signal and Wakeup signal should be logical ORed externally and should be inputted to each of XSLT_INT[7:0].
<b>ValidValues:</b>	
Bit	Description

[0]	Slot 1.
[1]	Slot 2.
[2]	Slot 3.
[3]	Slot 4.
[4]	Slot 5.
[5]	Slot 6.
[6]	Slot 7.
[7]	Slot 8.

### 9.2.13 USB Legacy Registers

This register space is used to provide USB legacy support. Each of the registers are located on a 32-bit boundary. The offsets of the registers are relative to the ACPIMMIO base (FED8\_0000h).

#### HCEx040 [HCE Control] (FCH::USBLEGACY::HceControl)

Used to enable and control the emulation hardware and report various status information.

\_aliasHOST; HCEx040; HCE=FED8\_0000h

Bits	Description
31:9	Reserved.
8	<b>A20State</b> . Read-write. Reset: 0. Indicates current state of Gate A20 on keyboard controller.
7	<b>IRQ12Active</b> . Read,Write-1-to-clear. Reset: 0. Indicates that a positive transition on IRQ12 from the keyboard controller has occurred. Software may write a 1 to this bit to clear it (set it to 0). Software write of a 0 to this bit has no effect.
6	<b>IRQ1Active</b> . Read,Write-1-to-clear. Reset: 0. Indicates that a positive transition on IRQ1 from the keyboard controller has occurred. Software may write a 1 to this bit to clear it (set it to 0). Software write of a 0 to this bit has no effect.
5	<b>GateA20Sequence</b> . Read-write. Reset: 0. Set by the Host Controller (HC).
4	<b>ExternalIRQEn</b> . Read-write. Reset: 0. 1=IRQ1 and IRQ12 from the keyboard controller causes an emulation interrupt. The function controlled by this bit is independent of the setting of the EmulationEnable bit in this register.
3	<b>IRQEn</b> . Read-write. Reset: 0. 1=The HC generates IRQ1 or IRQ12 as long as the FCH::USBLEGACY::HceStatus[OutputFull] == 1. If FCH::USBLEGACY::HceStatus[AuxOutputFull] == 0, then IRQ1 is generated, else if it is 1, then an IRQ12 is generated.
2	<b>CharacterPending</b> . Read-write. Reset: 0. 1=An emulation interrupt is generated when FCH::USBLEGACY::HceStatus[OutputFull] = 0.
1	<b>EmulationInterrupt</b> . Read-only. This bit is a static decode of the emulation interrupt condition.
0	<b>EmulationEnable</b> . Read-write. Reset: 0. 0=Disabled. 1=The HC is enabled for legacy emulation. The HC decodes accesses and generates IRQ1 and/or IRQ12 when appropriate. Additionally, the HC generates an emulation interrupt at appropriate times to invoke the emulation software.

#### HCEx044 [HCE Input] (FCH::USBLEGACY::HceInput)

Read-write. Reset: 0000\_0000h.

This register may be Read or Written directly by accessing it with its memory address in the Host Controller's operational register space. When accessed directly with a memory cycle, Reads and Writes of this register have no side effects.

\_aliasHOST; HCEx044; HCE=FED8\_0000h

Bits	Description
31:8	Reserved.
7:0	<b>InputData</b> . Read-write. Reset: 00h. This register holds data that is written.

**HCEx048 [HCE Output] (FCH::USBLEGACY::HceOutput)**

Read-write. Reset: 0000\_0000h.

The data placed in this register by the emulation software is returned when I/O port 60h is Read and emulation is enabled (FCH::USBLEGACY::HceControl[EmulationEnable] == 1). On a Read of this location, FCH::USBLEGACY::HceStatus[OutputFull] = 0 is set.

\_aliasHOST; HCEx048; HCE=FED8\_0000h

Bits	Description
31:8	Reserved.
7:0	<b>OutputData.</b> Read-write. Reset: 00h. This register hosts data that is returned when an IO Read of port 60h is performed by application software.

**HCEx04C [HCE Status] (FCH::USBLEGACY::HceStatus)**

Read-write. Reset: 0000\_0000h.

The contents of the HceStatus Register are returned on an IO Read of port 64h when emulation is enabled. Reads and Writes of port 60h and Writes to port 64h can cause changes in this register. Emulation software can directly access this register through its memory address in the Host Controller's operational register space. Accessing this register through its memory address produces no side effects.

\_aliasHOST; HCEx04C; HCE=FED8\_0000h

Bits	Description
31:8	Reserved.
7	<b>Parity.</b> Read-write. Reset: 0. Indicates parity error on keyboard/mouse data.
6	<b>TimeOut.</b> Read-write. Reset: 0. Used to indicate a time-out.
5	<b>AuxOutputFull.</b> Read-write. Reset: 0. IRQ12 is asserted whenever this bit is set to 1 and (OutputFull == 1 && FCH::USBLEGACY::HceControl[IRQEn] == 1).
4	<b>InhibitSwitch.</b> Read-write. Reset: 0. This bit reflects the state of the keyboard inhibit switch and is set if the keyboard is NOT inhibited.
3	<b>CmdData.</b> Read-write. Reset: 0. The HC sets this bit to 0 on an IO Write to port 60h and to 1 on an IO Write to port 64h.
2	<b>Flag.</b> Read-write. Reset: 0. Nominally used as a system flag by software to indicate a warm or cold boot.
1	<b>InputFull.</b> Read-write. Reset: 0. Except for the case of a Gate A20 sequence, this bit is set to 1 on an IO Write to address 60h or 64h. While this bit is set to 1 and FCH::USBLEGACY::HceControl[EmulationEnable] == 1, an emulation interrupt condition exists.
0	<b>OutputFull.</b> Read-write. Reset: 0. The HC sets this bit to 0 on a Read of IO port 60h. If (FCH::USBLEGACY::HceControl[IRQEn] == 1 && AuxOutputFull == 0), then an IRQ1 is generated as long as this bit is set to 1. If (FCH::USBLEGACY::HceControl[IRQEn] == 1 && AuxOutputFull == 1), then an IRQ12 is generated as long as this bit is set to 1. While this bit is 0 and FCH::USBLEGACY::HceControl[CharacterPending] == 1, an emulation interrupt condition exists.

**HCEx050 [HCE Interrupts Enable] (FCH::USBLEGACY::HceIntrEn)**

Read-write. Reset: 0000\_00A3h.

\_aliasHOST; HCEx050; HCE=FED8\_0000h

Bits	Description
31:11	Reserved.
10	<b>A20State_inv.</b> Read-write. Reset: 0. 1=Use inverted version of A20State.
9:8	<b>SOFSrc.</b> Read-write. Reset: 0h. Indicates which HCI_MSofN source is used in USB Legacy.
<b>ValidValues:</b>	
Value	Description
0h	HCI_MSofN[0]
1h	HCI_MSofN[1]
2h	HCI_MSofN[2]
3h	HCI_MSofN[3]

7:5	<b>SOFCounter</b> . Read-write. Reset: 5h. The amount of SOF number that emulation interrupt has to wait before SMI is sent.
4	<b>EmulationSmiEn</b> . Read-write. Reset: 0. 1=Generate SMI if emulation interrupt happens.
3	<b>BlockIRQ12</b> . Read-write. Reset: 0. 1=Block IRQ12.
2	<b>BlockIRQ1</b> . Read-write. Reset: 0. 1=Block IRQ1.
1	<b>EnIRQ12</b> . Read-write. Reset: 1. 1=Enable IRQ12Active.
0	<b>EnIRQ1</b> . Read-write. Reset: 1. 1=Enable IRQ1Active.

## 10 MMIP – Audio

This section describes the integrated Multimedia blocks for efficient audio processing within the APU.

### 10.1 Audio Coprocessor (ACP)

#### 10.1.1 Function and Registers

The Audio Coprocessor (ACP) component offloads specific real-time audio processing tasks from the main processor that are important for low power use cases, and provides optional support for device control to offload the main processor.

ACPx01400 (ACP::ACPPGFSM::ACP_I2S_PIN_CONFIG)		
Read-write. Reset: 0000_0007h.		
I2S PIN CONFIGURATION		
_aliasHOST; ACPx01400; ACP=NBIFEPFNCFG::BASE_ADDR_1_instACP_aliasHOST[BASE_ADDR]		
_aliasSMN; ACPMMIOx00041400; ACPMMIO=0120_0000h		
Bits	Description	
31:3	Reserved.	
2:0	<b>ACP_I2S_PIN_CONFIG</b> . Read-write. Reset: 7h. I2S pin configuration.	
	<b>ValidValues:</b>	
	Value	Description
	0h	MAX HDA; MIN SoundWire® [Azalia with Reset and 3 SDI lines; Soundwire with only Data[0] line].
	1h	MAX m-HDA; MIN SoundWire [Azalia without Reset and 3 SDI lines; Soundwire with Data[1:0] line].
	2h	MIN HDA; MAX SoundWire [Azalia with Reset and 1 SDI lines; Soundwire with Data[2:0] line].
	3h	MIN m-HDA; MAX SoundWire [Azalia without Reset and 1 SDI lines; Soundwire with Data[3:0] line].
	4h	I2S/TDM.
	6h-5h	Reserved.
	7h	Pins not Configured, all pads will be in input mode.

ACPx01404 (ACP::ACPPGFSM::ACP_PAD_PULLUP_PULLDOWN_CTRL)		
Read-write. Reset: 01FF_0000h.		
ACP Audio Pads PullUp and PullDown control.		
_aliasHOST; ACPx01404; ACP=NBIFEPFNCFG::BASE_ADDR_1_instACP_aliasHOST[BASE_ADDR]		
Bits	Description	
31:25	Reserved.	
24:16	<b>ACP_PAD_PULLDOWN_CTRL</b> . Read-write. Reset: 1FFh. Each bit corresponds to PullDown control of respective Audio Pad. Setting this bit will put the pad under PullDown control.	
	<b>ValidValues:</b>	
	Bit	Description
	[0]	BP_AZ_BITCLK_TDM_BCLK_MIC
	[1]	BP_AZ_RST_L_SW_DATA1_SW_DATA3_TDM_DATA_MIC
	[2]	BP_AZ_SYNC_TDM_FRM_MIC
	[3]	BP_AZ_SDIN0_CODEC_GPI
	[4]	BP_AZ_SDIN1_SW_DATA1_TDM_BCLK_PLAYBACK

	[5]	BP_AZ_SDIN2_SW_DATA2_TDM_DATA_PLAYBACK
	[6]	BP_AZ_SDOUT_TDM_FRM_PLAYBACK
	[7]	BP_SW_MCLK_TDM_BCLK_BT
	[8]	BP_SW_DATA0_TDM_DOUT_BT
15:9	Reserved.	
8:0	<b>ACP_PAD_PULLUP_CTRL</b> . Read-write. Reset: 000h. Each bit corresponds to PullUp control of the respective Audio Pad. Setting this bit will put the pad under PullUp control.	
	<b>ValidValues:</b>	
	<b>Bit</b>	<b>Description</b>
	[0]	BP_AZ_BITCLK_TDM_BCLK_MIC
	[1]	BP_AZ_RST_L_SW_DATA1_SW_DATA3_TDM_DATA_MIC
	[2]	BP_AZ_SYNC_TDM_FRM_MIC
	[3]	BP_AZ_SDIN0_CODEC_GPI
	[4]	BP_AZ_SDIN1_SW_DATA1_TDM_BCLK_PLAYBACK
	[5]	BP_AZ_SDIN2_SW_DATA2_TDM_DATA_PLAYBACK
	[6]	BP_AZ_SDOUT_TDM_FRM_PLAYBACK
	[7]	BP_SW_MCLK_TDM_BCLK_BT
	[8]	BP_SW_DATA0_TDM_DOUT_BT

#### ACPx01408 (ACP::ACPPGFSM::ACP\_PAD\_DRIVE\_STRENGTH\_CTRL)

Read-write. Reset: 0001\_5556h.

ACP Audio Pads Drive Strength control.

\_aliasHOST; ACPx01408; ACP=NBIFEPFNCFG::BASE\_ADDR\_1\_instACP\_aliasHOST[BASE\_ADDR]

Bits	Description										
31:18	Reserved.										
17:16	<b>ACP_PAD_PULLUP_CTRL_17_16</b> . Read-write. Reset: 1h. Audio pad BP_SW_DATA0_TDM_DOUT_BT <b>ValidValues:</b> <table border="1"> <thead> <tr> <th>Value</th><th>Description</th></tr> </thead> <tbody> <tr><td>0h</td><td>GPIO operation 1.2V only. All driver transistors turned on 100% drive. Z mode is Z40.</td></tr> <tr><td>1h</td><td>GPIO operation 1.5V &amp; 1.8V. 83.4% drive. Z mode is Z40.</td></tr> <tr><td>2h</td><td>GPIO operation 1.2V / 1.5V / 1.8V. 66.7% drive. Z mode is Z60.</td></tr> <tr><td>3h</td><td>GPIO operation 1.2V / 1.5V / 1.8V. Only 80 Ohms driver is on. 50% drive. Z mode is Z80.</td></tr> </tbody> </table>	Value	Description	0h	GPIO operation 1.2V only. All driver transistors turned on 100% drive. Z mode is Z40.	1h	GPIO operation 1.5V & 1.8V. 83.4% drive. Z mode is Z40.	2h	GPIO operation 1.2V / 1.5V / 1.8V. 66.7% drive. Z mode is Z60.	3h	GPIO operation 1.2V / 1.5V / 1.8V. Only 80 Ohms driver is on. 50% drive. Z mode is Z80.
Value	Description										
0h	GPIO operation 1.2V only. All driver transistors turned on 100% drive. Z mode is Z40.										
1h	GPIO operation 1.5V & 1.8V. 83.4% drive. Z mode is Z40.										
2h	GPIO operation 1.2V / 1.5V / 1.8V. 66.7% drive. Z mode is Z60.										
3h	GPIO operation 1.2V / 1.5V / 1.8V. Only 80 Ohms driver is on. 50% drive. Z mode is Z80.										
15:14	<b>ACP_PAD_PULLUP_CTRL_15_14</b> . Read-write. Reset: 1h. Audio pad BP_SW_MCLK_TDM_BCLK_BT <b>ValidValues:</b> <table border="1"> <thead> <tr> <th>Value</th><th>Description</th></tr> </thead> <tbody> <tr><td>0h</td><td>GPIO operation 1.2V only. All driver transistors turned on 100% drive. Z mode is Z40.</td></tr> <tr><td>1h</td><td>GPIO operation 1.5V &amp; 1.8V. 83.4% drive. Z mode is Z40.</td></tr> <tr><td>2h</td><td>GPIO operation 1.2V / 1.5V / 1.8V. 66.7% drive. Z mode is Z60.</td></tr> <tr><td>3h</td><td>GPIO operation 1.2V / 1.5V / 1.8V. Only 80 Ohms driver is on. 50% drive. Z mode is Z80.</td></tr> </tbody> </table>	Value	Description	0h	GPIO operation 1.2V only. All driver transistors turned on 100% drive. Z mode is Z40.	1h	GPIO operation 1.5V & 1.8V. 83.4% drive. Z mode is Z40.	2h	GPIO operation 1.2V / 1.5V / 1.8V. 66.7% drive. Z mode is Z60.	3h	GPIO operation 1.2V / 1.5V / 1.8V. Only 80 Ohms driver is on. 50% drive. Z mode is Z80.
Value	Description										
0h	GPIO operation 1.2V only. All driver transistors turned on 100% drive. Z mode is Z40.										
1h	GPIO operation 1.5V & 1.8V. 83.4% drive. Z mode is Z40.										
2h	GPIO operation 1.2V / 1.5V / 1.8V. 66.7% drive. Z mode is Z60.										
3h	GPIO operation 1.2V / 1.5V / 1.8V. Only 80 Ohms driver is on. 50% drive. Z mode is Z80.										
13:12	<b>ACP_PAD_PULLUP_CTRL_13_12</b> . Read-write. Reset: 1h. Audio pad BP_AZ_SDOUT_TDM_FRM_PLAYBACK <b>ValidValues:</b> <table border="1"> <thead> <tr> <th>Value</th><th>Description</th></tr> </thead> <tbody> <tr><td>0h</td><td>GPIO operation 1.2V only. All driver transistors turned on 100% drive. Z mode is Z40.</td></tr> <tr><td>1h</td><td>GPIO operation 1.5V &amp; 1.8V. 83.4% drive. Z mode is Z40.</td></tr> <tr><td>2h</td><td>GPIO operation 1.2V / 1.5V / 1.8V. 66.7% drive. Z mode is Z60.</td></tr> <tr><td>3h</td><td>GPIO operation 1.2V / 1.5V / 1.8V. Only 80 Ohms driver is on. 50% drive. Z mode is Z80.</td></tr> </tbody> </table>	Value	Description	0h	GPIO operation 1.2V only. All driver transistors turned on 100% drive. Z mode is Z40.	1h	GPIO operation 1.5V & 1.8V. 83.4% drive. Z mode is Z40.	2h	GPIO operation 1.2V / 1.5V / 1.8V. 66.7% drive. Z mode is Z60.	3h	GPIO operation 1.2V / 1.5V / 1.8V. Only 80 Ohms driver is on. 50% drive. Z mode is Z80.
Value	Description										
0h	GPIO operation 1.2V only. All driver transistors turned on 100% drive. Z mode is Z40.										
1h	GPIO operation 1.5V & 1.8V. 83.4% drive. Z mode is Z40.										
2h	GPIO operation 1.2V / 1.5V / 1.8V. 66.7% drive. Z mode is Z60.										
3h	GPIO operation 1.2V / 1.5V / 1.8V. Only 80 Ohms driver is on. 50% drive. Z mode is Z80.										
11:10	<b>ACP_PAD_PULLUP_CTRL_11_10</b> . Read-write. Reset: 1h. Audio pad BP_AZ_SDIN2_SW_DATA2_TDM_DATA_PLAYBACK										



	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	GPIO operation 1.2V only. All driver transistors turned on 100% drive. Z mode is Z40.
	1h	GPIO operation 1.5V & 1.8V. 83.4% drive. Z mode is Z40.
	2h	GPIO operation 1.2V / 1.5V / 1.8V. 66.7% drive. Z mode is Z60.
	3h	GPIO operation 1.2V / 1.5V / 1.8V. Only 80 Ohms driver is on. 50% drive. Z mode is Z80.
9:8	<b>ACP_PAD_PULLUP_CTRL_9_8.</b> Read-write. Reset: 1h. Audio pad BP_AZ_SDIN1_SW_DATA1_TDM_BCLK_PLAYBACK	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	GPIO operation 1.2V only. All driver transistors turned on 100% drive. Z mode is Z40.
	1h	GPIO operation 1.5V & 1.8V. 83.4% drive. Z mode is Z40.
7:6	<b>ACP_PAD_PULLUP_CTRL_7_6.</b> Read-write. Reset: 1h. Audio pad BP_AZ_SDIN0_CODECS_GPI	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	GPIO operation 1.2V only. All driver transistors turned on 100% drive. Z mode is Z40.
	1h	GPIO operation 1.5V & 1.8V. 83.4% drive. Z mode is Z40.
5:4	<b>ACP_PAD_PULLUP_CTRL_5_4.</b> Read-write. Reset: 1h. Audio pad BP_AZ_SYNC_TDM_FRM_MIC	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	GPIO operation 1.2V only. All driver transistors turned on 100% drive. Z mode is Z40.
	1h	GPIO operation 1.5V & 1.8V. 83.4% drive. Z mode is Z40.
3:2	<b>ACP_PAD_PULLUP_CTRL_3_2.</b> Read-write. Reset: 1h. Audio pad BP_AZ_RST_L_SW_DATA1_SW_DATA3_TDM_DATA_MIC	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	GPIO operation 1.2V only. All driver transistors turned on 100% drive. Z mode is Z40.
	1h	GPIO operation 1.5V & 1.8V. 83.4% drive. Z mode is Z40.
1:0	<b>ACP_PAD_PULLUP_CTRL_1_0.</b> Read-write. Reset: 2h. Audio pad BP_AZ_BITCLK_TDM_BCLK_MIC	
	<b>ValidValues:</b>	
	<b>Value</b>	<b>Description</b>
	0h	GPIO operation 1.2V only. All driver transistors turned on 100% drive. Z mode is Z40.
	1h	GPIO operation 1.5V & 1.8V. 83.4% drive. Z mode is Z40.

**ACPx0140C (ACP::ACPPGFSM::ACP\_SW\_PAD\_KEEPER\_EN)**

Read-write. Reset: 0000_0000h.	
ACP SoundWire Audio Pads Bus Keeper enable.	
_aliasHOST; ACPx0140C; ACP=NBIFEPFNCFG::BASE_ADDR_1_instACP_aliasHOST[BASE_ADDR]	
<b>Bits</b>	<b>Description</b>

31:4	Reserved.										
3:0	<b>ACP_SW_PAD_KEEPER_EN.</b> Read-write. Reset: 0h. Each bit corresponds to the Keeper Enable of respective the Audio Pad. Setting each bit enables the Bus Keeper feature of respective SoundWire pad. <b>ValidValues:</b>										
	<table> <tr> <th>Bit</th><th>Description</th></tr> <tr> <td>[0]</td><td>BP_SW_DATA0_TDM_DOUT_BT</td></tr> <tr> <td>[1]</td><td>BP_AZ_SDIN1_SW_DATA1_TDM_BCLK_PLAYBACK</td></tr> <tr> <td>[2]</td><td>BP_AZ_RST_L_SW_DATA1_SW_DATA3_TDM_DATA_MIC</td></tr> <tr> <td>[3]</td><td>BP_AZ_SDIN2_SW_DATA2_TDM_DATA_PLAYBACK</td></tr> </table>	Bit	Description	[0]	BP_SW_DATA0_TDM_DOUT_BT	[1]	BP_AZ_SDIN1_SW_DATA1_TDM_BCLK_PLAYBACK	[2]	BP_AZ_RST_L_SW_DATA1_SW_DATA3_TDM_DATA_MIC	[3]	BP_AZ_SDIN2_SW_DATA2_TDM_DATA_PLAYBACK
Bit	Description										
[0]	BP_SW_DATA0_TDM_DOUT_BT										
[1]	BP_AZ_SDIN1_SW_DATA1_TDM_BCLK_PLAYBACK										
[2]	BP_AZ_RST_L_SW_DATA1_SW_DATA3_TDM_DATA_MIC										
[3]	BP_AZ_SDIN2_SW_DATA2_TDM_DATA_PLAYBACK										

**ACPx01410 (ACP::ACPPGFSM::ACP\_SW\_WAKE\_EN)**

Read-write. Reset: 0000\_0000h.

ACP SoundWire Wake enable.

\_aliasHOST; ACPx01410; ACP=NBIFEPFNCFG::BASE\_ADDR\_1\_instACP\_aliasHOST[BASE\_ADDR]

Bits	Description
31:1	Reserved.
0	<b>ACP_SW_WAKE_EN.</b> Read-write. Reset: 0. 0=Disable software wake. 1=Enable software wake. ACP SoundWire Wake feature enable. If (ACP_SW_WAKE_EN && ACP::ACPPGFSM::ACP_PME_EN[ACP_PME_EN]) == 1, receiving a high signal on SoundWire Data[0] pin triggers the ACP_Wake interrupt to the FCH. The interrupt is lowered when any of ACP::ACPPGFSM::ACP_PME_EN[ACP_PME_EN], ACP_SW_WAKE_EN or SoundWire Data[0] pin is low.

**ACPx01414 (ACP::ACPPGFSM::ACP\_I2S\_WAKE\_EN)**

Read-write. Reset: 0000\_0000h.

ACP SoundWire Wake enable.

\_aliasHOST; ACPx01414; ACP=NBIFEPFNCFG::BASE\_ADDR\_1\_instACP\_aliasHOST[BASE\_ADDR]

Bits	Description
31:1	Reserved.
0	<b>ACP_I2S_WAKE_EN.</b> Read-write. Reset: 0. 0=Disable I2S wake. 1=Enable I2S wake. ACP I2S Wake feature enable. If (ACP_I2S_WAKE_EN && ACP::ACPPGFSM::ACP_PME_EN[ACP_PME_EN]) == 1, receiving a high signal on CODEC_GPI pad triggers the ACP_Wake Interrupt to the FCH. The interrupt is lowered when any of ACP::ACPPGFSM::ACP_PME_EN[ACP_PME_EN], ACP_I2S_WAKE_EN or CODEC_GPI is low.

**ACPx01418 (ACP::ACPPGFSM::ACP\_PME\_EN)**

Read-write. Reset: 0000\_0000h.

ACP SoundWire Wake enable.

\_aliasHOST; ACPx01418; ACP=NBIFEPFNCFG::BASE\_ADDR\_1\_instACP\_aliasHOST[BASE\_ADDR]

Bits	Description
31:1	Reserved.
0	<b>ACP_PME_EN.</b> Read-write. Reset: 0. 0=Disable. 1=Enable. ACP PME Enable. Setting this bit raises a wake interrupt to the FCH if (ACP::ACPPGFSM::ACP_SW_WAKE_EN[ACP_SW_WAKE_EN]    ACP::ACPPGFSM::ACP_I2S_WAKE_EN[ACP_I2S_WAKE_EN]) == 1.

**ACPx0141C (ACP::ACPPGFSM::ACP\_PGFSM\_CONTROL)**

Read-write. Reset: 0000\_0000h.

ACP PGFSM Control and Status register.

\_aliasHOST; ACPx0141C; ACP=NBIFEPFNCFG::BASE\_ADDR\_1\_instACP\_aliasHOST[BASE\_ADDR]

Bits	Description
31:1	Reserved.
0	<b>ACP_PGFSM_CTRL.</b> Read-write. Reset: 0. 0=Power OFF ACP. 1=Power ON ACP. ACP ON/OFF power domain control.

**ACPx01420 (ACP::ACPPGFSM::ACP\_PGFSM\_STATUS)**

Read-only. Reset: 0000\_0002h.

ACP PGFSM Control and Status register.

\_aliasHOST; ACPx01420; ACP=NBIFEPFNCFG::BASE\_ADDR\_1\_instACP\_aliasHOST[BASE\_ADDR]

Bits	Description
31:2	Reserved.
1:0	<b>ACP_PGFSM_STATUS.</b> Read-only. Reset: 2h. ACP ON/OFF power domain status.
<b>ValidValues:</b>	
Value	Description
0h	Power is ON.
1h	In power ON process.
2h	Power is OFF.
3h	In power OFF process.

## List of Namespaces

Namespace	Heading(s)
ACP::ACPPGFSM	10.1.1 [Function and Registers]
Core::X86::Apic	2.1.11.2.2 [Local APIC Registers]
Core::X86::Cpuid	2.1.12.1 [CPUID Instruction Functions]
Core::X86::MsR	2.1.13.1 [MSRs - MSR0000_0xxx] 2.1.13.2 [MSRs - MSRC000_0xxx] 2.1.13.3 [MSRs - MSRC001_0xxx] 2.1.13.4 [MSRs - MSRC001_1xxx]
Core::X86::Pmc::Core	2.1.14.2 [Large Increment per Cycle Events] 2.1.14.3.1 [Floating Point (FP) Events] 2.1.14.3.2 [LS Events] 2.1.14.3.3 [IC and BP Events] 2.1.14.3.4 [DE Events] 2.1.14.3.5 [EX (SC) Events] 2.1.14.3.6 [L2 Cache Events]
Core::X86::Pmc::L3	2.1.14.4.1 [L3 Cache PMC Events]
Core::X86::Smm	2.1.11.1.6 [System Management State]
FCH::AOAC	9.2.8 [Always On Always Connected (AOAC) Registers]
FCH::EMMC	9.2.2.1 [eMMC Configuration Registers]
FCH::EMMC::MMIO	9.2.2.2 [eMMC Host Controller Registers]
FCH::GPIO	9.2.11.2 [GPIO Registers]
FCH::IO	9.2.1.1 [Registers]
FCH::IOAPIC	9.2.3.1 [IOAPIC Registers]
FCH::IOMUX	9.2.11.1 [IOMUX Registers]
FCH::ITF::ESPI	9.2.9.4 [eSPI Registers]
FCH::ITF::LPC	9.2.9.1 [Device 14h Function 3 (LPC Bridge) Configuration Registers]
FCH::ITF::SPI	9.2.9.3.2 [FCH::ITF::SPI Registers]
FCH::MISC	9.2.10.1 [Miscellaneous (MISC) Registers]
FCH::PM	9.2.10.2 [Power Management (PM) Registers]

	9.2.10.4.1 [AcpiPmEvtBlk] 9.2.10.4.2 [AcpiPm1CntBlk] 9.2.10.4.3 [AcpiPmTmrBlk] 9.2.10.4.4 [AcpiGpe0Blk] 9.2.10.4.5 [SmiCmdBlk]
FCH::PM2	9.2.10.3 [Power Management (PM2) Registers]
FCH::SD	9.2.12.1 [SD Configuration Registers (SD)]
FCH::SD::MMIO	9.2.12.2 [SD Host Controller Memory Mapped Registers (SDHC)]
FCH::SMI	9.2.4 [SMI Registers]
FCH::TMR::ACDC	9.2.7 [Wake Alarm Device (AcDcTimer) Registers]
FCH::TMR::HPET	9.2.5 [High Precision Event Timer (HPET) Registers]
FCH::TMR::WDT	9.2.6 [Watchdog Timer (WDT) Registers]
FCH::USBLEGACY	9.2.13 [USB Legacy Registers]
IO	2.1.7 [PCI Configuration Legacy Access]
IOMMUL1	8.1.2.1 [IOMMUL1 Registers]
IOMMUL2	8.1.2.2 [IOMMUL2 Registers]
IOMMUMMIO	8.1.2.3 [IOMMUMMIO Registers]
MCA::CS	3.2.5.8 [CS]
MCA::DE	3.2.5.4 [DE]
MCA::EX	3.2.5.5 [EX]
MCA::FP	3.2.5.6 [FP]
MCA::IF	3.2.5.2 [IF]
MCA::L2	3.2.5.3 [L2]
MCA::L3	3.2.5.7 [L3]
MCA::LS	3.2.5.1 [LS]
MCA::PIE	3.2.5.9 [PIE]
MCA::UMC	3.2.5.10 [UMC]
SBTSI	6.4 [SB-TSI Registers]
SMU::THM	4.2.1 [Registers]

## List of Definitions

**ABS:** ABS(integer expression): Remove sign from signed value.

**AGESA™:** AMD Generic Encapsulated Software Architecture.

**AM4:** Desktop, single die, single socket. For client desktop platform (uPGA) DDR4. AM4 = (Core::X86::CpuId::BrandId[PkgType] == 02h).

**AP:** Applications Processor.

**APML:** Advanced Platform Management Link.

**APU:** Accelerated Processing Unit.

**ARA:** Alert response address.

**ARP:** Address Resolution Protocol

**BAR:** The BAR, or base address register, physical register mnemonic format is of the form PREFIXxZZZ.

**BatteryPower:** The system is running from a limited energy or battery power source or otherwise undocked from a continuous power supply. Setting using this definition may be required to change during run time.

**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).

**BMC:** Base management controller.

**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.

**BSC:** Boot strap core. Core 0 of the BSP.

**BSP:** Boot strap processor.

**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.

**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.

**CEIL:** CEIL(real expression): Rounds real number up to nearest integer.

**CMP:** Specifies the core number.

**COF:** Current operating frequency of a given clock domain.

**Cold reset:** PWROK is de-asserted and RESET\_L is asserted.

**Configurable:** Indicates that the access type is configurable as described by the documentation.

**CoreCOF:** Core current operating frequency in MHz. CoreCOF = (Core::X86::Msrr::PStateDef[CpuFid[7:0]]/Core::X86::Msrr::PStateDef[CpuFid[fsId]])\*200. A nominal frequency reduction can occur if spread spectrum clocking is enabled.

**COUNT:** COUNT(integer expression): Returns the number of binary 1's in the integer.

**CpuCoreNum:** Specifies the core number.

**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.

**DID:** Divisor Identifier. Specifies the post-PLL divisor used to reduce the COF.

**docACPI:** Advanced Configuration and Power Interface (ACPI) Specification. <http://www.acpi.info>.

**docAM4:** Socket AM4 Processor Functional Data Sheet, order# 55509.

**docAPM1:** AMD64 Architecture Programmer's Manual Volume 1: Application Programming, order# 24592.

**docAPM2:** AMD64 Architecture Programmer's Manual Volume 2: System Programming, order# 24593.

**docAPM3:** AMD64 Architecture Programmer's Manual Volume 3: Instruction-Set Reference, order# 24594.

**docAPM4:** AMD64 Architecture Programmer's Manual Volume 4: 128-Bit and 256-Bit Media Instructions, order# 26568.

**docAPM5:** AMD64 Architecture Programmer's Manual Volume 5: 64-Bit Media and x87 Floating-Point Instructions, order# 26569.

**docASF:** Alert Standard Format Specification. <http://dmtf.org/standards/asf>.

**docATA:** AT Attachment with Packet Interface. <http://www.t13.org>.

**docDP:** VESA DisplayPort Standard. <http://www.vesa.org/vesa-standards>.

**docI2C:** I2C Bus Specification. [http://www.nxp.com/documents/user\\_manual/UM10204.pdf](http://www.nxp.com/documents/user_manual/UM10204.pdf)

**docIOMMU:** AMD I/O Virtualization Technology Specification, order#

48882.

**docJEDEC:** JEDEC Standards. <http://www.jedec.org>.

**docPCIe:** PCI Express® Specification. <http://www.pcisig.org>.

**docPCIlb:** PCI Local Bus Specification. <http://www.pcisig.org>.

**docRevG:** Revision Guide for AMD Family 17h Models 10h-1Fh Processors, order# 55606.

**docSATA:** Serial ATA Specification. <http://www.sata-io.org>.

**docSDHC:** Secure Digital Host Controller Standard Specification. <https://www.sdcard.org>.

**docSFP5:** AMD Socket FP5 Processor Functional Data Sheet, order# 55594.

**docSMB:** System Management Bus (SMBus) Specification. <http://www.smbus.org>.

**docUSB:** Universal Serial Bus Specification. <http://www.usb.org>.

**Doubleword:** A 32-bit value.

**DW:** Doubleword.

**EC:** Embedded Controller.

**ECS:** Extended Configuration Space.

**EDC:** Electrical design current. Indicates the maximum current the voltage rail can demand for a short, thermally insignificant time.

**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.

**FCH:** The integrated platform subsystem that contains the IO interfaces and bridges them to the system BIOS. Previously included in the Southbridge.

**FDS:** Functional Data Sheet. There is one FDS for each package type. See docAM4 or docSFP5.

**FID:** Frequency Identifier. Specifies the PLL frequency multiplier for a given clock domain.

**FLOOR:** FLOOR(integer expression): Rounds real number down to nearest integer.

**FP5:** Notebook package for direct solder boards (BGA). FP5 = (Core::X86::CpuId::BrandId[PkgType] == 00h).

**FreeRunSampleTimer:** An internal free running timer used by many power management features.

**GB:** Gbyte or Gigabyte; 1,073,741,824 bytes.

**GT/s:** Giga-Transfers per second.

**HTC:** Hardware Thermal Control.

**HTC-active state:** Hardware-controlled lower-power, lower performance state used to reduce temperature.

**IBS:** Instruction based sampling.

**Inaccessible:** Not readable or writable (e.g., Hide ? Inaccessible : Read-Write).

**IO configuration:** Access to configuration space though IO ports CF8h and CFCh.

**IOMMU:** IO Memory Management Unit

**IORR:** IO range register.

**KB:** Kbyte or Kilobyte; 1024 bytes.

**KBC:** Keyboard Controller.

**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.

**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 XX [Logical Mnemonic].

**LRL:** 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.

**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.

**Master or SMBus Master:** The device that initiates and terminates all communication and drives the clock, SCL.

**MAX:** MAX(integer expression list): Picks maximum integer or real value of comma separated list.

**MB:** Megabyte; 1024 KB.

**MCA:** Machine Check Architecture.

**MCAX:** Machine Check Architecture eXtensions.

**MergeEvent:** A PMC event that is capable of counter increments greater than 15, thus requiring merging a pair of even/odd performance monitors.

**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.

**MIN:** MIN(integer expression list): Picks minimum integer or real value of comma separated list.

**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.

**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.

**MTRR:** Memory-type range register. The MTRRs specify the type of memory associated with various memory ranges.

**NBC:** NBC=(CUID Fn00000001\_EBX[LocalApicId[3:0]] == 0). Node Base Core. The lowest numbered core in the node.

**Node:** A node, is an integrated circuit device that includes one to 4 cores (one Core Complex).

**NTA:** Non-Temporal Access.

**OW:** Octword. An 128-bit value.

**PCICFG:** The PCICFG, or PCI defined configuration space, physical register mnemonic format is of the form DXFYxZZZ.

**PCIe@:** PCI Express.

**PCS:** Physical Coding Sublayer.

**PEC:** Packet error code.

**physical mnemonic:** The register mnemonic that is formed based on the physical address used to access the register (e.g., D18F3x00). See XX [Physical Mnemonic].

**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.

**POR:** Power on reset.

**POW:** POW(base, exponent): POW(x,y) returns the value x to the power of y.

**Processor:** A package containing one or more Nodes. See Node.

**PTE:** Page table entry.

**QW:** Quadword. A 64-bit value.

**REFCLK:** Reference clock. Refers to the clock frequency (100 MHz) or the clock period (10 ns) depending on the context used.

**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 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 name:** A name that annotates the function of the register.

**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 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.

**Reserved-write-as-0:** Reads are undefined. Must always write 0.

**Reserved-write-as-1:** Reads are undefined. Must always write 1.

**ROUND:** ROUND(real expression): Rounds to the nearest integer; halfway rounds away from zero.

**RTS:** Remote temperature sensor, typical examples are ADM1032, LM99, MAX6657, EMC1002.

**SB-RMI:** Remote Management interface.

**SB-TSI:** Sideband Internal Temperature Sensor Interface. See APML.

**SDXI:** Smart Data Accelerator Interface

**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.

**Slave or SMBus slave:** The slave cannot initiate SMBus communication and cannot drive the clock but can drive the data signal SDA and the alert signal ALERT\_L.

**SMAC:** System Management Action Field. This is the code passed from the SMC to the processors in STPCLK assertion messages.

**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.

**SSC:** Spread Spectrum Clocking.

**SVM:** Secure virtual machine.

**TCC:** Temperature calculation circuit.

**Tctl:** Processor temperature control value.

**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.

**Thread:** One architectural context for instruction execution.

**Token:** A scheduler entry used in various DF queues to track outstanding requests.

**TOM2:** Top of extended Memory.

**TSI:** Temperature sensor interface.

**TSM:** Temperature sensor macro.

**UMI:** Unified Media Interface. The link between the processor and the FCH.

**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.

**Unpredictable:** The behavior of both reads and writes is unpredictable.

**VID:** Voltage level identifier.

**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.

**Warm reset:** RESET\_L is asserted only (while PWROK stays high).

**WDT:** Watchdog timer. A timer that detects activity and triggers an error if a specified period of time expires without the activity.

**WRIG:** Writes Ignored.

**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.

**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-once:** Capable of being written once; all subsequent writes have no effect. If not associated with Read, then reads are undefined.

**XBAR:** Cross bar; command packet switch.

## Memory Map - MSR

Physical Mnemonic	Namespace
0000_0000h...0000_0001h	MCA::LS
0000_0010h...0000_02FFh	Core::X86::Msr
0000_0400h...0000_0403h	MCA::LS
0000_0404h...0000_0407h	MCA::IF
0000_0408h...0000_040Bh	MCA::L2
0000_040Ch...0000_040Fh	MCA::DE
0000_0414h...0000_0417h	MCA::EX
0000_0418h...0000_041Bh	MCA::FP
0000_041Ch...0000_042Bh	MCA::L3
0000_043Ch...0000_0443h	MCA::UMC
0000_0450h...0000_0457h	MCA::CS
0000_0458h...0000_045Bh	MCA::PIE
C000_0080h...C000_0410h	Core::X86::Msr
C000_2000h...C000_2009h	MCA::LS
C000_2010h...C000_2016h	MCA::IF
C000_2020h...C000_2029h	MCA::L2
C000_2030h...C000_2036h	MCA::DE
C000_2050h...C000_2056h	MCA::EX
C000_2060h...C000_2066h	MCA::FP
C000_2070h...C000_20A9h	MCA::L3
C000_20F0h...C000_210Ah	MCA::UMC
C000_2140h...C000_2159h	MCA::CS
C000_2160h...C000_2169h	MCA::PIE
C001_0000h...C001_029Bh	Core::X86::Msr
C0010400	MCA::LS
C0010401	MCA::IF
C0010402	MCA::L2
C0010403	MCA::DE
C0010405	MCA::EX
C0010406	MCA::FP
C001_0407h...C001_040Ah	MCA::L3
C001_040Fh...C001_0410h	MCA::UMC
C001_0414h...C001_0415h	MCA::CS
C0010416	MCA::PIE
C001_1002h...C001_103Ch	Core::X86::Msr



## Memory Map - Main Memory

Physical Mnemonic	Namespace
0000_0000h: GPUF0REGx59800	SMU::THM
FEC00000: IOAPICx0000...x0010	FCH::IOAPIC
FEC00000: IOAPICx0010_indirectaddressoffset00...x0010_indirectaddressoffset3F	FCH::IOAPIC
FEC00000: IOAPICx0020...x0040	FCH::IOAPIC
FEC10000: SPIx000...x080	FCH::ITF::SPI
FEC20000: ESPIx00000000...x000000AC	FCH::ITF::ESPI
FED00000: HPETx000...x1E8	FCH::TMR::HPET
FED80000: HCEx040...x050	FCH::USBLEGACY
FED80200: SMIx000...x0C4	FCH::SMI
FED80300: PMx000...x0FC	FCH::PM
FED80400: PM2x000...x0E5	FCH::PM2
FED80700: RTCHOSTx00000...x0007F	FCH::IO
FED80000: PMx0800...x081D	FCH::PM
FED80B00: WDTx000...x004	FCH::TMR::WDT
FED80D00: IOMUXx000...x090	FCH::IOMUX
FED80E00: MISCx000...x0FC	FCH::MISC
FED81500: GPIOx00000...x000F8	FCH::GPIO
FED8_1500h: GPIOx00FC	FCH::GPIO
FED81500: GPIOx00100...x002EC	FCH::GPIO
FED81500: GPIOx02F0...x02FC	FCH::GPIO
FED81500: GPIOx00300...x003FC	FCH::GPIO
FED81D00: ACDCx000...x020	FCH::TMR::ACDC
FED81E00: AOACx000040...x0000A0	FCH::AOAC
FEDD5000: EMMCHCx00000000...x000000FC	FCH::EMMC::MMIO
FEDD5800: EMMCCFGx00000000...x0000010C	FCH::EMMC

# Memory Map - PCICFG

Physical Mnemonic	Namespace
D00F2x000...x084	IOMMUL2
D14F3x000...x0D4	FCH::ITF::LPC
D14F6x000...x097	FCH::SD

## Memory Map - SMN

Physical Mnemonic	Namespace
0005_9800h: SMUTHMx00000000	SMU::THM
0120_0000h: ACPMMIOx00041400	ACP::ACPPGFSM
02400000: IOMMUMMIOx00000000...x0004132C	IOMMUMMIO
13F00000: IOMMUCFGx00000000...x00000084	IOMMUL2
13F01000: IOMMUL2Bx0000012C...x00000250	IOMMUL2
14700000: IOMMUL1INT0x0000001C...x000000F4	IOMMUL1
14800000: IOMMUL1INT1x0000001C...x000000F4	IOMMUL1
15700000: IOMMUL2Ax00000088...x000000D0	IOMMUL2