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

Legal Notices

List of Chapters

Table of Contents

List of Figures

List of Tables

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 27: Machine Check Terms and Acronyms

Term	Description
MCA	Machine Check Architecture.
MCAX	Machine Check Architecture eXtensions.
WRIG	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 diagnosibility.
    - 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 28 [Legacy MCA MSR Layout].

Table 28: 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_DESTATUS
  - 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 29 [MCAX MSR Layout].

Access to unused MCAX MSRs is RAZ/WRIG. MCA Bank 4 is always Read-as-zero (RAZ/WRIG).

Table 29: 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 30 [MCAX Implementation-Specific Register Layout].

Table 30: 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 31 [Error Overwrite Priorities] indicates which errors are overwritten in the error status registers.

Table 31: Error Overwrite Priorities

		Older Error		
		Uncorrected	Deferred	Corrected
Newer Error	Uncorrected	-	Overwrite	Overwrite
	Deferred	-	-	Overwrite
	Corrected	-	-	-

Table 32 [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 32: 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 33: 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 34: Error code: transaction type (TT)

TT	Transaction Type
00	Instruction
01	Data
10	Generic
11	Reserved

Table 35: Error codes: cache level (LL)

LL	Cache Level
00	L0: Core
01	L1: Level 1

10	L2: Level 2
11	LG: Generic

Table 36: 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 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

Table 37: Blocks Capable of Supporting MCA Banks

Acronym	Block Function
LS	Load-Store Unit
IF	Instruction Fetch Unit
L2	L2 Cache Unit
DE	Decode Unit
EX	Execution Unit
FP	Floating Point Unit
L3	L3 Cache Unit
PIE	Power Management, Interrupts, Etc.
CS	Coherent Slave
UMC	Unified Memory Controller
NBIO	Northbridge IO Unit
PB	Parameter Block
PSP	Platform Security Processor
SMU	System Management Controller Unit
PCIE	PCIe® Root Port
MP5	Microprocessor5 Management Controller

Table 38: Mapping of Blocks to MCA_IPID[HwId] and MCA_IPID[McaType]

Block	**Hardware ID**	MCA **Type**
LS	0xb0	0x10
IF	0xb0	0x1
L2	0xb0	0x2
L3	0xb0	0x7
MP5	0x1	0x2
PB	0x5	0x0
UMC	0x96	0x0
NBIO	0x18	0x0
PCIE	0x46	0x0
SMU	0x1	0x1
PSP	0xff	0x1
PIE	0x2e	0x1
CS	0x2e	0x2
EX	0xb0	0x5
FP	0xb0	0x6
DE	0xb0	0x3

3.2.2 Implemented Machine Check Bank Registers

Table 39 [Legacy MCA Registers] provides links to the description of each block's Legacy MCA registers. Table 40 [MCAX Registers] provides links to the description of each block's MCA Extension Registers.

Table 39: 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
PB	MCA::PB::MCA_CTL_PB	MCA::PB::MCA_STATUS_PB	MCA::PB::MCA_ADDR_PB	MCA::PB::MCA_MISC0_PB	MCA::PB::MCA_CTL_MASK_PB
PSP	MCA::PSP::MCA_CTL_PSP	MCA::PSP::MCA_STATUS_PSP	MCA::PSP::MCA_ADDR_PSP	MCA::PSP::MCA_MISC0_PSP	MCA::PSP::MCA_CTL_MASK_PSP
SMU	MCA::SMU::MCA_CTL_SMU	MCA::SMU::MCA_STATUS_SMU	MCA::SMU::MCA_ADDR_SMU	MCA::SMU::MCA_MISC0_SMU	MCA::SMU::MCA_CTL_MASK_SMU
NBIO	MCA::NBIO::MCA_CTL_NBIO	MCA::NBIO::MCA_STATUS_NBIO	MCA::NBIO::MCA_ADDR_NBIO	MCA::NBIO::MCA_MISC0_NBIO	MCA::NBIO::MCA_CTL_MASK_NBIO
PCIE	MCA::PCIE::MCA_CTL_PCIE	MCA::PCIE::MCA_STATUS_PCIE	MCA::PCIE::MCA_ADDR_PCIE	MCA::PCIE::MCA_MISC0_PCIE	MCA::PCIE::MCA_CTL_MASK_PCIE

Table 40: 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
PB	MCA::PB::MCA_CONFIG_PB	MCA::PB::MCA_IPID_PB	MCA::PB::MCA_SYND_PB	--	--
PSP	MCA::PSP::MCA_CONFIG_PSP	MCA::PSP::MCA_IPID_PSP	MCA::PSP::MCA_SYND_PSP	--	--
SMU	MCA::SMU::MCA_CONFIG_SMU	MCA::SMU::MCA_IPID_SMU	MCA::SMU::MCA_SYND_SMU	--	--
NBIO	MCA::NBIO::MCA_CONFIG_NBIO	MCA::NBIO::MCA_IPID_NBIO	MCA::NBIO::MCA_SYND_NBIO	MCA::NBIO::MCA_DESTAT_NBIO	MCA::NBIO::MCA_DEADDR_NBIO
PCIE	MCA::PCIE::MCA_CONFIG_PCIE	MCA::PCIE::MCA_IPID_PCIE	MCA::PCIE::MCA_SYND_PCIE	MCA::PCIE::MCA_DESTAT_PCIE	MCA::PCIE::MCA_DEADDR_PCIE

3.2.3 Mapping of Banks to Blocks

Table 41 [Core MCA Bank to Block Mapping] shows MCA banks that are present in the address space of every logical core:

Table 41: Core MCA Bank to Block Mapping

Bank	Block
0	LS
1	IF
2	L2
3	DE
4	RAZ
5	EX
6	FP

Table 42 [Non-core MCA Bank to Block Mapping] shows MCA banks that are present in the address space of specific logical cores:

Table 42: Non-core MCA Bank to Block Mapping

Bank	Thread **0**	Thread **2**	Thread **4**	Thread **6**	Thread **8**	Thread **10**	Thread **12**	Thread **14**
7	L3	L3	L3	L3	L3	L3	L3	L3
8	L3	L3	L3	L3	L3	L3	L3	L3
9	L3	L3	L3	L3	L3	L3	L3	L3
10	L3	L3	L3	L3	L3	L3	L3	L3
11	L3	L3	L3	L3	L3	L3	L3	L3
12	L3	L3	L3	L3	L3	L3	L3	L3
13	L3	L3	L3	L3	L3	L3	L3	L3
14	L3	L3	L3	L3	L3	L3	L3	L3
15	MP5	MP5	MP5	MP5	MP5	MP5	MP5	MP5
16	PB	PB	PB	PB	PB	PB	PB	PB
17	UMC	UMC	UMC	UMC	RAZ	RAZ	RAZ	RAZ
18	UMC	UMC	UMC	UMC	RAZ	RAZ	RAZ	RAZ
19	CS	CS	CS	CS	RAZ	RAZ	RAZ	RAZ
20	CS	CS	CS	CS	RAZ	RAZ	RAZ	RAZ
21	CS	CS	CS	CS	RAZ	RAZ	RAZ	RAZ
22	NBIO	NBIO	NBIO	NBIO	RAZ	RAZ	RAZ	RAZ
23	PCIE	PCIE	PCIE	PCIE	RAZ	RAZ	RAZ	RAZ
24	SMU	RAZ	RAZ	RAZ	RAZ	RAZ	RAZ	RAZ
25	PSP	RAZ	RAZ	RAZ	RAZ	RAZ	RAZ	RAZ
26	PB	RAZ	RAZ	RAZ	RAZ	RAZ	RAZ	RAZ
27	PIE	RAZ	RAZ	RAZ	RAZ	RAZ	RAZ	RAZ

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 38 [Mapping of Blocks to MCA_IPID[HwId] and MCA_IPID[McaType]] 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 Thread 0] (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::Msr::MCG_CTL. Does not affect error detection, correction, or logging.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst0_aliasMSRLEGACY; MSR0000_0400

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst0_aliasMSR; MSRC000_2000

Bits	Description
63:24	Reserved.
23	**STORE_DATA_OTHER**. Read-write. Reset: 0. A parity error was detected in an STLF, SCB EMEM entry or SRB store data by any access. Subtract 33 from the error location to get the actual error index
22	**HWA**. Read-write. Reset: 0. A hardware assertion error was reported
21	**SystemReadDataErrorWcb**. Read-write. Reset: 0. A SystemReadDataError error was reported on read data returned from L2 for a WCB store
20	**SystemReadDataErrorScb**. Read-write. Reset: 0. A SystemReadDataError error was reported on read data returned from L2 for an SCB store
19	**SystemReadDataErrorLoad**. Read-write. Reset: 0. A SystemReadDataError error was reported on read data returned from L2 for a load
18	**SCB_POISON**. Read-write. Reset: 0. A poisoned line was detected in an SCB entry by any access
17	**WCB**. Read-write. Reset: 0. A parity error was detected in a WCB entry by any access
16	**SCB_DATA**. Read-write. Reset: 0. A parity error was detected in an SCB entry data field by any access
15	**SCB_ADDR**. Read-write. Reset: 0. A parity error was detected in an SCB entry address field by any access
14	**SCB_STATE**. Read-write. Reset: 0. A parity error was detected in an SCB entry state field by any access
13	**MAB**. Read-write. Reset: 0. A parity error was detected in a MAB entry by any access
12	**LDQ**. Read-write. Reset: 0. A parity error was detected in an LDQ entry by any access
11	**STQ**. Read-write. Reset: 0. A parity error was detected in an STQ entry by any access
10	**PWC**. Read-write. Reset: 0. A parity error was detected in a PWC entry by any access
9	**L2DTLB**. Read-write. Reset: 0. A parity error was detected in an L2 TLB entry by any access
8	**L1DTLB**. Read-write. Reset: 0. A parity error was detected in an L1 TLB entry by any access. This error only logs a valid address down through bit [12], in spite of the AddrLsbCnt value of 0
7	**EMEM_RMW**. Read-write. Reset: 0. An ECC error was detected on an EMEM read-modify-write by a store
6	**EMEM_LOAD**. Read-write. Reset: 0. An ECC error was detected on an EMEM read by a load.
5	**DC_TAG_STORE**. Read-write. Reset: 0. An ECC error or poison bit mismatch was detected on a tag read by a store (note overflow may get incorrectly set).

4	**DC_TAG_LOAD**. Read-write. Reset: 0. An ECC error or poison bit mismatch was detected on a tag read by a load (note overflow may get incorrectly set).
3	**DC_TAG_VICTIM**. Read-write. Reset: 0. An ECC error or poison bit mismatch was detected on a tag read by a probe or victimization.
2	**DC_DATA_RMW**. Read-write. Reset: 0. An ECC error was detected on a data cache read-modify-write by a store
1	**DC_DATA_LOAD**. Read-write. Reset: 0. An ECC error or L2 poison was detected on a data cache read by a load (note PhysLoc is set to zero and Poison is set to 1 for L2 poison).
0	**DC_DATA_VICTIM**. Read-write. Reset: 0. An ECC error was detected on a data cache read by a probe or victimization

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.	
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst0_aliasMSRLSLEGACY; MSR0000_0001	
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst0_aliasMSRLEGACY; MSR0000_0401	
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst0_aliasMSR; MSRC000_2001	
Bits	**Description**
63	**Val**. 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	**Overflow**. 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	**UC**. 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	**En**. 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	**MiscV**. Reset: Cold,0. 1=Valid thresholding in MCA::LS::MCA_MISC0_LS. In certain modes, MISC registers are owned by platform firmware and will RAZ when read by non-SMM code. Therefore, it is possible for MiscV=1 and the MISC register to read as all zeros.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
58	**AddrV**. 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	**PCC**. 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	**ErrCoreIdVal**. 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	**TCC**. Reset: Cold,0. 1=Hardware context of the process thread to which the error was reported may have been corrupted. Continued operation of the thread may have unpredictable results. The thread 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	**RESERV54**. Reset: Cold,0. MCA_STATUS Register Reserved bit.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
53	**SyndV**. 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	Reserved.
51:47	**RESERV47**. Reset: Cold,00h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
46	**CECC**. 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	**UECC**. 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	**Deferred**. 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	**Poison**. 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.
42:41	**RESERV41**. Reset: Cold,0h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
40	**Scrub**. Reset: Cold,0. 1=The error was the result of a scrub operation.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
39:38	**RESERV38**. Reset: Cold,0h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
37:32	**ErrCoreId**. 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.
31:30	**RESERV30**. Reset: Cold,0h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
29:24	**AddrLsb**. 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.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
23:22	**RESERV22**. Reset: Cold,0h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
21:16	**ErrorCodeExt**. 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::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
15:0	**ErrorCode**. 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 43: MCA_STATUS_LS

Error Type	ErrorCode Ext	UC	PCC	TCC	Deferred	Poison	AddrV
DC_DATA_	0x0	0	0	0	0/1	0	1

VICTIM							
DC_DATA_LOAD	0x1	0/1	0	0/1	0	0/1	1
DC_DATA_RMW	0x2	0	0	0	0/1	0	1
DC_TAG_VICTIM	0x3	0/1	0/1	0/1	0/1	0	0
DC_TAG_LOAD	0x4	0/1	0/1	0/1	0/1	0	0
DC_TAG_STORE	0x5	0/1	0/1	0/1	0/1	0	0
EMEM_LOAD	0x6	1	1	1	0	0	1
EMEM_RMW	0x7	0/1	0/1	0/1	0	0	0
L1DTLB	0x8	0	0	0	0	0	0
L2DTLB	0x9	0	0	0	0	0	1
PWC	0xa	0	0	0	0	0	1
STQ	0xb	1	1	1	0	0	0
LDQ	0xc	1	1	1	0	0	0
MAB	0xd	1	1	1	0	0	0
SCB_STATE	0xe	1	1	1	0	0	0
SCB_ADDR	0xf	1	1	1	0	0	0
SCB_DATA	0x10	1	1	1	0	0	0
WCB	0x11	1	1	1	0	0	0
SCB_POISON	0x12	0	0	0	1	0	0
SystemReadDataErrorLoad	0x13	1	0	1	0	0	1
SystemReadDataErrorScb	0x14	1	1	1	0	0	1
SystemReadDataErrorWcb	0x15	1	1	1	0	0	0
HWA	0x16	1	1	1	0	0	0
STORE_DATA_OTHER	0x17	1	1	1	0	0	0

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.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst0_aliasMSRLSLEGACY; MSR0000_0000

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst0_aliasMSRLEGACY; MSR0000_0402

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst0_aliasMSR; MSRC000_2002

Bits	Description
63:57	Reserved.
56:0	**ErrorAddr**. Read-write,Volatile. Reset: Cold,000_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 44: MCA_ADDR_LS

Error Type	Bits	Description
DC_DATA_VICTIM	[56:52]	Reserved
	[51:6]	Physical address
	[5]	Reserved
	[4:0]	Which data word had the error
DC_DATA_LOAD	[56:52]	Reserved
	[51:1]	Physical address
	[0]	Reserved
DC_DATA_RMW	[56:52]	Reserved
	[51:6]	Physical address
	[5:0]	Reserved
DC_TAG_VICTIM	[56:0]	Reserved
DC_TAG_LOAD	[56:0]	Reserved
DC_TAG_STORE	[56:0]	Reserved
EMEM_LOAD	[56:52]	Reserved
	[51:1]	Physical address
	[0]	Reserved
EMEM_RMW	[56:0]	Reserved
L1DTLB	[56:0]	Reserved
L2DTLB	[56:52]	Reserved
	[51:12]	Physical address
	[11:0]	Reserved
PWC	[56:52]	Reserved
	[51:12]	Physical address
	[11:0]	Reserved
STQ	[56:0]	Reserved
LDQ	[56:0]	Reserved
MAB	[56:0]	Reserved
SCB_STATE	[56:0]	Reserved
SCB_ADDR	[56:0]	Reserved
SCB_DATA	[56:0]	Reserved
WCB	[56:0]	Reserved
SCB_POISON	[56:0]	Reserved
SystemReadDataErrorLoad	[56:52]	Reserved
	[51:1]	Physical address
	[0]	Reserved
SystemReadDataErrorScb	[56:52]	Reserved
	[51:6]	Physical address
	[5:0]	Reserved
SystemReadDataErrorWcb	[56:0]	Reserved
HWA	[56:0]	Reserved
STORE_DATA_OTHER	[56:0]	Reserved

MSR0000_0403...MSRC000_2003 [LS Machine Check Miscellaneous 0 Thread 0] (MCA::LS::MCA_MISC0_LS)

Log miscellaneous information associated with errors.	
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst0_aliasMSRLEGACY; MSR0000_0403	
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst0_aliasMSR; MSRC000_2003	
Bits	**Description**
63	**Valid**. Reset: 1. 1=A valid CntP field is present in this register.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read-only.
62	**CntP**. Reset: 1. 1=A valid threshold counter is present.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read-only.
61	**Locked**. 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	**IntP**. 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::LS::MCA_MISC0_LS[Locked]) ? Read-write : Read-only.
59:56	Reserved.
55:52	**LvtOffset**. 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::LS::MCA_MISC0_LS[Locked]) ? Read-write : Read-only.
51	**CntEn**. Reset: 0. 1=Count thresholding errors.
	AccessType: (Core::X86::Msr::HWCR[McStatusWrEn] | !MCA::LS::MCA_MISC0_LS[Locked]) ? Read-write : Read-only.
50:49	**ThresholdIntType**. 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::LS::MCA_MISC0_LS[Locked]) ? Read-write : Read-only.
48	**Ovrflw**. 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::LS::MCA_MISC0_LS[Locked]) ? Read-write : Read-only.
47:44	Reserved.
43:32	**ErrCnt**. 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::LS::MCA_MISC0_LS[Locked]) ? Read-write : Read-only.
31:24	**BlkPtr**. Read-write. Reset: 00h. 00h=Extended MISC MSR block is not valid. 01h=Extended MSR block is valid.
23:0	Reserved.

MSRC000_2004 [LS Machine Check Configuration Thread 0] (MCA::LS::MCA_CONFIG_LS)

Reset: 0000_0002_0000_0125h.	
Controls configuration of the associated machine check bank.	
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst0_aliasMSR; MSRC000_2004	
Bits	**Description**
63:39	Reserved.
38:37	**DeferredIntType**. 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	**LogDeferredInMcaStat**. 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	**McaXEnable**. 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.
31:9	Reserved.
8	**McaLsbInStatusSupported**. Read-only. Reset: 1. 1=MCA::LS::MCA_CONFIG_LS[McaLsbInStatusSupported] indicates that AddrLbc is located in McaStatus registers.
7:6	Reserved.
5	**DeferredIntTypeSupported**. 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	**DeferredErrorLoggingSupported**. 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	**McaX**. 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 Thread 0] (MCA::LS::MCA_IPID_LS)

Reset: 0010_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.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst0_aliasMSR; MSRC000_2005

Bits	Description
63:48	**McaType**. Read-only. Reset: 0010h. The McaType of the MCA bank within this IP.
47:44	**InstanceIdHi**. Read-write. Reset: 0h. The HI value instance ID of this IP. This is initialized to a unique ID per instance of this register.
43:32	**HardwareID**. Read-only. Reset: 0B0h. The Hardware ID of the IP associated with this MCA bank.
31:0	**InstanceId**. 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

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst0_aliasMSR; MSRC000_2006

Bits	Description
63:39	Reserved.
38:32	**Syndrome**. 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	**ErrorPriority**. 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	**Length**. 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	**ErrorInformation**. Read-write,Volatile. Reset: Cold,0_0000h. Contains error-specific information about the location of the error. Decoding is available in Table 45 [MCA_SYND_LS].

Table 45: MCA_SYND_LS

Error Type	Bits	Description
DC_DATA_VICTIM	[17]	1
	[16]	1
	[15:14]	Reserved
	[13:8]	Cache line index
	[7:3]	Reserved
	[2:0]	Cache line way
DC_DATA_LOAD	[17]	1
	[16]	1
	[15:14]	Reserved
	[13:8]	Cache line index
	[7:3]	Reserved
	[2:0]	Cache line way
DC_DATA_RMW	[17]	0
	[16]	1
	[15:5]	Reserved
	[4:0]	Which data word had the error
DC_TAG_VICTIM	[17]	1
	[16]	1
	[15:14]	Reserved
	[13:8]	Cache line index
	[7:3]	Reserved
	[2:0]	Cache line way
DC_TAG_LOAD	[17]	1
	[16]	1
	[15:14]	Reserved
	[13:8]	Cache line index
	[7:3]	Reserved
	[2:0]	Cache line way
DC_TAG_STORE	[17]	1
	[16]	1
	[15:14]	Reserved

	[13:8]	Cache line index
	[7:3]	Reserved
	[2:0]	Cache line way
EMEM_LOAD	[17]	1
	[16]	0
	[15:14]	Reserved
	[13:8]	Cache line index
	[7:0]	Reserved
EMEM_RMW	[17]	0
	[16]	1
	[15:5]	Reserved
	[4:0]	Which data word had the error
L1DTLB	[17]	0
	[16]	1
	[15:7]	Reserved
	[6:0]	TLB entry index
L2DTLB	[17]	1
	[16]	1
	[15:11]	Reserved
	[10:7]	TLB way
	[6:0]	TLB entry index
PWC	[17]	0
	[16]	1
	[15:6]	Reserved
	[5:0]	PWC entry index
STQ	[17]	0
	[16]	1
	[15:6]	Reserved
	[5:0]	STQ entry index
LDQ	[17]	0
	[16]	1
	[15:6]	Reserved
	[5:0]	LDQ entry index
MAB	[17]	0
	[16]	1
	[15:5]	Reserved
	[4:0]	MAB entry index
SCB_STATE	[17]	0
	[16]	1
	[15:4]	Reserved
	[3:0]	SCB entry index
SCB_ADDR	[17]	0
	[16]	1
	[15:4]	Reserved
	[3:0]	SCB entry index
SCB_DATA	[17]	0
	[16]	1

	[15:4]	Reserved
	[3:0]	SCB entry index
WCB	[17]	0
	[16]	1
	[15:3]	Reserved
	[2:0]	WCB entry index
SCB_POISON	[17]	0
	[16]	1
	[15:4]	Reserved
	[3:0]	SCB entry index
SystemReadDataErrorLoad	[17]	0
	[16]	0
	[15:2]	Reserved
	[1:0]	SystemReadDataError response error type
SystemReadDataErrorScb	[17]	0
	[16]	0
	[15:2]	Reserved
	[1:0]	SystemReadDataError response error type
SystemReadDataErrorWcb	[17]	0
	[16]	0
	[15:2]	Reserved
	[1:0]	SystemReadDataError response error type
HWA	[17]	0
	[16]	0
	[15:8]	Reserved
	[7]	MCA was signaled
	[6]	Reserved
	[5:0]	assertion type
STORE_DATA_OTHER	[17]	0
	[16]	0
	[15:7]	Reserved
	[6]	if 1 then [5:0] are STQ index, if 0 then [3:0] are SCB index
	[5:0]	if bit 6 is 1 then STQ index if bit 6 is 0 then [5:4] are 2'b0 and [3:0] are STQ index

MSRC000_2008 [LS Machine Check Deferred Error Status Thread 0] (MCA::LS::MCA_DESTAT_LS)

Reset: Cold,0000_0000_0000_0000h.

Holds status information for the first deferred error seen in this bank.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst0_aliasMSR; MSRC000_2008

Bits	Description
63	**Val**. Read-write,Volatile. Reset: Cold,0. 1=A valid error has been detected (whether it is enabled or not).
62	**Overflow**. 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	**RESERV4**. Read-write. Reset: Cold,0h.
58	**AddrV**. Read-write,Volatile. Reset: Cold,0. 1=MCA::LS::MCA_DEADDR_LS contains address information associated with the error.
57:54	**RESERV3**. Read-write. Reset: Cold,0h.

53	**SyndV**. 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	**RESERV2**. Read-write. Reset: Cold,00h.
44	**Deferred**. 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.
43:30	**RESERV1**. Read-write. Reset: Cold,0000h.
29:24	**AddrLsb**. 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.
23:22	**RESERV0**. Read-write. Reset: Cold,0h.
21:16	**ErrorCodeExt**. Read-write,Volatile. Reset: Cold,00h. Logs an extended error code when an error is detected. This model-specific field is used in conjunction with ErrorCode? to identify the error sub-type for root cause analysis.
15:0	**ErrorCode**. Read-write,Volatile. Reset: Cold,0000h. Error code for this error.

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_DESTAT_LS[AddrLsb].

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst0_aliasMSR; MSRC000_2009

Bits	Description
63:57	Reserved.
56:0	**ErrorAddr**. Read-write,Volatile. Reset: Cold,000_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_DESTAT_LS[AddrLsb].

MSRC001_0400 [LS Machine Check Control Mask Thread 0] (MCA::LS::MCA_CTL_MASK_LS)

Read-write. Reset: 0000_0000_0000_0000h.

Inhibit detection of an error source.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst0_aliasMSR; MSRC001_0400

Bits	Description
63:24	Reserved.
23	**STORE_DATA_OTHER**. Read-write. Reset: 0. A parity error was detected in an STLF, SCB EMEM entry or SRB store data by any access. Subtract 33 from the error location to get the actual error index
22	**HWA**. Read-write. Reset: 0. A hardware assertion error was reported
21	**SystemReadDataErrorWcb**. Read-write. Reset: 0. Init: BIOS,1. A SystemReadDataError error was reported on read data returned from L2 for a WCB store
20	**SystemReadDataErrorScb**. Read-write. Reset: 0. Init: BIOS,1. A SystemReadDataError error was reported on read data returned from L2 for an SCB store
19	**SystemReadDataErrorLoad**. Read-write. Reset: 0. Init: BIOS,1. A SystemReadDataError error was reported on read data returned from L2 for a load
18	**SCB_POISON**. Read-write. Reset: 0. A poisoned line was detected in an SCB entry by any access
17	**WCB**. Read-write. Reset: 0. A parity error was detected in a WCB entry by any access

16	**SCB_DATA**. Read-write. Reset: 0. A parity error was detected in an SCB entry data field by any access
15	**SCB_ADDR**. Read-write. Reset: 0. A parity error was detected in an SCB entry address field by any access
14	**SCB_STATE**. Read-write. Reset: 0. A parity error was detected in an SCB entry state field by any access
13	**MAB**. Read-write. Reset: 0. A parity error was detected in a MAB entry by any access
12	**LDQ**. Read-write. Reset: 0. A parity error was detected in an LDQ entry by any access
11	**STQ**. Read-write. Reset: 0. A parity error was detected in an STQ entry by any access
10	**PWC**. Read-write. Reset: 0. A parity error was detected in a PWC entry by any access
9	**L2DTLB**. Read-write. Reset: 0. A parity error was detected in an L2 TLB entry by any access
8	**L1DTLB**. Read-write. Reset: 0. A parity error was detected in an L1 TLB entry by any access. This error only logs a valid address down through bit [12], in spite of the AddrLsbCnt value of 0
7	**EMEM_RMW**. Read-write. Reset: 0. An ECC error was detected on an EMEM read-modify-write by a store
6	**EMEM_LOAD**. Read-write. Reset: 0. An ECC error was detected on an EMEM read by a load.
5	**DC_TAG_STORE**. Read-write. Reset: 0. An ECC error or poison bit mismatch was detected on a tag read by a store (note overflow may get incorrectly set).
4	**DC_TAG_LOAD**. Read-write. Reset: 0. An ECC error or poison bit mismatch was detected on a tag read by a load (note overflow may get incorrectly set).
3	**DC_TAG_VICTIM**. Read-write. Reset: 0. An ECC error or poison bit mismatch was detected on a tag read by a probe or victimization.
2	**DC_DATA_RMW**. Read-write. Reset: 0. An ECC error was detected on a data cache read-modify-write by a store
1	**DC_DATA_LOAD**. Read-write. Reset: 0. An ECC error or L2 poison was detected on a data cache read by a load (note PhysLoc is set to zero and Poison is set to 1 for L2 poison).
0	**DC_DATA_VICTIM**. Read-write. Reset: 0. An ECC error was detected on a data cache read by a probe or victimization

3.2.5.2 IF

MSR0000_0404...MSRC000_2010 [IF Machine Check Control Thread 0] (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.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst1_aliasMSRLEGACY; MSR0000_0404

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst1_aliasMSR; MSRC000_2010

Bits	Description
63:19	Reserved.
18	**CtMceError**. Read-write. Reset: 0. CT MCE
17	**BsrParity**. Read-write. Reset: 0. BSR Parity Error.
16	**L2TlbMultiHit**. Read-write. Reset: 0. L2-TLB Multi-Hit
15	**L1TlbMultiHit**. Read-write. Reset: 0. L1-TLB Multi-Hit.
14	**HwAssert**. Read-write. Reset: 0. Hardware Assertion Error.
13	**SystemReadDataError**. Read-write. Reset: 0. L2 Cache Error Response.
12	**L2RespPoison**. Read-write. Reset: 0. L2 Cache Response Poison Error. Error is the result of consuming poison data.
11	**L2BtbMultiHit**. Read-write. Reset: 0. BP L2-BTB Multi-Hit Error.
10	**L1BtbMultiHit**. Read-write. Reset: 0. BP L1-BTB Multi-Hit Error.
9	**IcUtagParity**. Read-write. Reset: 0. Ic MicroTag Parity Error.
8	**RSVD_8**. Read-write. Reset: 0. Reserved. Will never trigger.
7	**L2ItlbParity**. Read-write. Reset: 0. L2-TLB Parity Error.

6	**L1ItlbParity**. Read-write. Reset: 0. L1-TLB Parity Error.
5	**RSVD_5**. Read-write. Reset: 0. Reserved. Will never trigger.
4	**DqParity**. Read-write. Reset: 0. PRQ Parity Error.
3	**DataParity**. Read-write. Reset: 0. IC Data Array Parity Error.
2	**TagParity**. Read-write. Reset: 0. IC Full Tag Parity Error.
1	**TagMultiHit**. Read-write. Reset: 0. IC Full Tag Multi-hit Error.
0	**OcUtagParity**. Read-write. Reset: 0. Op Cache Microtag Parity Error. Parity errors on PA and other relevant uTag fields are reported, independent of any utag probing. The parity error way and index are logged.

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.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst1_aliasMSRLEGACY; MSR0000_0405

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst1_aliasMSR; MSRC000_2011

Bits	Description
63	**Val**. 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	**Overflow**. 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	**UC**. 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	**En**. 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	**MiscV**. Reset: Cold,0. 1=Valid thresholding in MCA::IF::MCA_MISC0_IF. In certain modes, MISC registers are owned by platform firmware and will RAZ when read by non-SMM code. Therefore, it is possible for MiscV=1 and the MISC register to read as all zeros.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
58	**AddrV**. 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	**PCC**. 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	**ErrCoreIdVal**. 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	**TCC**. Reset: Cold,0. 1=Hardware context of the process thread to which the error was reported may have been corrupted. Continued operation of the thread may have unpredictable results. The thread 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	**RESERV54**. Reset: Cold,0. MCA_STATUS Register Reserved bit.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
53	**SyndV**. 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	Reserved.
51:47	**RESERV47**. Reset: Cold,00h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
46	**CECC**. 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	**UECC**. 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	**Deferred**. 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	**Poison**. 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.
42:41	**RESERV41**. Reset: Cold,0h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
40	**Scrub**. Reset: Cold,0. 1=The error was the result of a scrub operation.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
39:38	**RESERV38**. Reset: Cold,0h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
37:32	**ErrCoreId**. 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.
31:30	**RESERV30**. Reset: Cold,0h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
29:24	**AddrLsb**. 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.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
23:22	**RESERV22**. Reset: Cold,0h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
21:16	**ErrorCodeExt**. 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	**ErrorCode**. 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 46: MCA_STATUS_IF

Error Type	ErrorCode Ext	UC	PCC	TCC	Deferred	Poison	AddrV
OcUtagParity	0x0	1	1	1	0	0	0
TagMultiHit	0x1	0	0	0	0	0	1

TagParity	0x2	0	0	0	0	0	1
DataParity	0x3	0	0	0	0	0	1
DqParity	0x4	1	1	1	0	0	0
RSVD_5	0x5	1	1	1	0	0	0
L1ItlbParity	0x6	0	0	0	0	0	1
L2ItlbParity	0x7	0	0	0	0	0	1
RSVD_8	0x8	1	1	1	0	0	0
IcUtagParity	0x9	0	0	0	0	0	0
L1BtbMultiHit	0xa	0	0	0	0	0	0
L2BtbMultiHit	0xb	0	0	0	0	0	0
L2RespPoison	0xc	1	0	1	0	1	1
SystemReadDataError	0xd	1	0	1	0	0	1
HwAssert	0xe	1	1	1	0	0	0
L1TlbMultiHit	0xf	0	0	0	0	0	1
L2TlbMultiHit	0x10	0	0	0	0	0	1
BsrParity	0x11	1	1	1	0	0	0
CtMceError	0x12	1	1	1	0	0	0

MSR0000_0406...MSRC000_2012 [IF Machine Check Address Thread 0] (MCA::IF::MCA_ADDR_IF)

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.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst1_aliasMSRLEGACY; MSR0000_0406

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst1_aliasMSR; MSRC000_2012

Bits	Description
63:57	Reserved.
56:0	**ErrorAddr**. Read-write,Volatile. Reset: Cold,000_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 47: MCA_ADDR_IF

Error Type	Bits	Description
OcUtagParity	[56:0]	Reserved
TagMultiHit	[56:0]	VA
TagParity	[56:0]	VA
DataParity	[56:0]	VA
DqParity	[56:0]	Reserved
RSVD_5	[56:0]	Reserved
L1ItlbParity	[56:0]	VA
L2ItlbParity	[56:0]	VA
RSVD_8	[56:0]	Reserved
IcUtagParity	[56:0]	Reserved
L1BtbMultiHit	[56:0]	Reserved

L2BtbMultiHit	[56:0]	Reserved
L2RespPoison	[56:0]	VA
SystemReadDataError	[56:0]	VA
HwAssert	[56:0]	Reserved
L1TlbMultiHit	[56:0]	VA
L2TlbMultiHit	[56:0]	VA
BsrParity	[56:0]	Reserved
CtMceError	[56:0]	Reserved

MSR0000_0407...MSRC000_2013 [IF Machine Check Miscellaneous 0 Thread 0] (MCA::IF::MCA_MISC0_IF)

Log miscellaneous information associated with errors.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst1_aliasMSRLEGACY; MSR0000_0407

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst1_aliasMSR; MSRC000_2013

Bits	Description
63	**Valid**. Reset: 1. 1=A valid CntP field is present in this register.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read-only.
62	**CntP**. Reset: 1. 1=A valid threshold counter is present.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read-only.
61	**Locked**. 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	**IntP**. 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	**LvtOffset**. 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	**CntEn**. Reset: 0. 1=Count thresholding errors.
	AccessType: (Core::X86::Msr::HWCR[McStatusWrEn] | !MCA::IF::MCA_MISC0_IF[Locked]) ? Read-write : Read-only.
50:49	**ThresholdIntType**. 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::IF::MCA_MISC0_IF[Locked]) ? Read-write : Read-only.
48	**Ovrflw**. 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	**ErrCnt**. 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	**BlkPtr**. Read-write. Reset: 00h. 00h=Extended MISC MSR block is not valid. 01h=Extended MSR block is valid.
23:0	Reserved.

MSRC000_2014 [IF Machine Check Configuration Thread 0] (MCA::IF::MCA_CONFIG_IF)

Reset: 0000_0002_0000_0121h.

Controls configuration of the associated machine check bank.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst1_aliasMSR; MSRC000_2014

Bits	Description
63:39	Reserved.
38:37	**DeferredIntType**. 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	**McaXEnable**. 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.
31:9	Reserved.
8	**McaLsbInStatusSupported**. Read-only. Reset: 1. 1=MCA::IF::MCA_CONFIG_IF[McaLsbInStatusSupported] indicates that AddrLbc is located in McaStatus registers.
7:6	Reserved.
5	**DeferredIntTypeSupported**. 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	**DeferredErrorLoggingSupported**. 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	**McaX**. 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 Thread 0] (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.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst1_aliasMSR; MSRC000_2015

Bits	Description
63:48	**McaType**. Read-only. Reset: 0001h. The McaType of the MCA bank within this IP.
47:44	**InstanceIdHi**. Read-write. Reset: 0h. The HI value instance ID of this IP. This is initialized to a unique ID per instance of this register.
43:32	**HardwareID**. Read-only. Reset: 0B0h. The Hardware ID of the IP associated with this MCA bank.
31:0	**InstanceId**. 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

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst1_aliasMSR; MSRC000_2016

Bits	Description

63:33	Reserved.
32	**Syndrome**. 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	**ErrorPriority**. 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	**Length**. 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	**ErrorInformation**. Read-write,Volatile. Reset: Cold,0_0000h. Contains error-specific information about the location of the error. Decoding is available in Table 48 [MCA_SYND_IF].

Table 48: MCA_SYND_IF

Error Type	Bits	Description
OcUtagParity	[17:16]	Reserved
	[15:8]	Way bit vector
	[7:6]	Reserved
	[5:0]	Index
TagMultiHit	[17:16]	Reserved
	[15:8]	Way bit vector
	[7:6]	Reserved
	[5:0]	Index
TagParity	[17]	Reserved
	[16]	Way Valid
	[15:8]	Reserved
	[7:0]	Way bit vector
DataParity	[17:15]	Reserved
	[14:12]	Data Bank
	[11:8]	Sub-Bank
	[7:0]	Index
DqParity	[17]	Reserved
	[16]	PA Error Valid
	[15:0]	Way bit vector
RSVD_5	[17:0]	Reserved
L1ItlbParity	[17:6]	Reserved
	[5:0]	Index
L2ItlbParity	[17:16]	Reserved
	[15:8]	Way bit vector
	[7:6]	Reserved
	[5:0]	Index
RSVD_8	[17:0]	Reserved
IcUtagParity	[17:16]	Reserved
	[15:8]	Way bit vector
	[7:6]	Reserved

	[5:0]	Index
L1BtbMultiHit	[17:12] [11:8] [7:0]	Reserved Way bit vector Index
L2BtbMultiHit	[17:12] [11:9] [8:0]	Reserved Table Index
L2RespPoison	[17:0]	Reserved
SystemReadDataError	[17:4] [3] [2] [1] [0]	Reserved Protection Violation Transaction Error Target Abort Master Abort
HwAssert	[17:5] [4:0]	Assertion log Code
L1TlbMultiHit	[17:6] [5:0]	Reserved Index
L2TlbMultiHit	[17:16] [15:8] [7:6] [5:0]	Reserved Way bit vector Reserved Index
BsrParity	[17:8] [7:0]	Reserved Index
CtMceError	[17:2] [1:0]	Reserved Thread bit vector

MSRC001_0401 [IF Machine Check Control Mask Thread 0] (MCA::IF::MCA_CTL_MASK_IF)

Read-write. Reset: 0000_0000_0000_0000h.

Inhibit detection of an error source.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst1_aliasMSR; MSRC001_0401

Bits	Description
63:19	Reserved.
18	**CtMceError**. Read-write. Reset: 0. CT MCE
17	**BsrParity**. Read-write. Reset: 0. BSR Parity Error.
16	**L2TlbMultiHit**. Read-write. Reset: 0. Init: BIOS,1. L2-TLB Multi-Hit
15	**L1TlbMultiHit**. Read-write. Reset: 0. L1-TLB Multi-Hit.
14	**HwAssert**. Read-write. Reset: 0. Hardware Assertion Error.
13	**SystemReadDataError**. Read-write. Reset: 0. L2 Cache Error Response.
12	**L2RespPoison**. Read-write. Reset: 0. L2 Cache Response Poison Error. Error is the result of consuming poison data.
11	**L2BtbMultiHit**. Read-write. Reset: 0. Init: BIOS,1. BP L2-BTB Multi-Hit Error.
10	**L1BtbMultiHit**. Read-write. Reset: 0. BP L1-BTB Multi-Hit Error.
9	**IcUtagParity**. Read-write. Reset: 0. Ic MicroTag Parity Error.
8	**RSVD_8**. Read-write. Reset: 0. Reserved. Will never trigger.
7	**L2ItlbParity**. Read-write. Reset: 0. L2-TLB Parity Error.
6	**L1ItlbParity**. Read-write. Reset: 0. L1-TLB Parity Error.
5	**RSVD_5**. Read-write. Reset: 0. Reserved. Will never trigger.
4	**DqParity**. Read-write. Reset: 0. PRQ Parity Error.

3	**DataParity**. Read-write. Reset: 0. IC Data Array Parity Error.
2	**TagParity**. Read-write. Reset: 0. IC Full Tag Parity Error.
1	**TagMultiHit**. Read-write. Reset: 0. IC Full Tag Multi-hit Error.
0	**OcUtagParity**. Read-write. Reset: 0. Op Cache Microtag Parity Error. Parity errors on PA and other relevant uTag fields are reported, independent of any utag probing. The parity error way and index are logged.

3.2.5.3 L2

MSR0000_0408...MSRC000_2020 [L2 Machine Check Control Thread 0] (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.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst2_aliasMSRLEGACY; MSR0000_0408

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst2_aliasMSR; MSRC000_2020

Bits	Description
63:4	Reserved.
3	**Hwa**. Read-write. Reset: 0. Hardware Assert Error.
2	**Data**. Read-write. Reset: 0. L2M Data Array ECC Error.
1	**Tag**. Read-write. Reset: 0. L2M Tag or State Array ECC Error.
0	**MultiHit**. 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.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst2_aliasMSRLEGACY; MSR0000_0409

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst2_aliasMSR; MSRC000_2021

Bits	Description
63	**Val**. 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	**Overflow**. 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	**UC**. 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	**En**. 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	**MiscV**. Reset: Cold,0. 1=Valid thresholding in MCA::L2::MCA_MISC0_L2. In certain modes, MISC registers are owned by platform firmware and will RAZ when read by non-SMM code. Therefore, it is possible for MiscV=1 and the MISC register to read as all zeros.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
58	**AddrV**. 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	**PCC**. 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	**ErrCoreIdVal**. 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	**TCC**. Reset: Cold,0. 1=Hardware context of the process thread to which the error was reported may have been corrupted. Continued operation of the thread may have unpredictable results. The thread 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	**RESERV54**. Reset: Cold,0. MCA_STATUS Register Reserved bit.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
53	**SyndV**. 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	Reserved.
51:47	**RESERV47**. Reset: Cold,00h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
46	**CECC**. 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	**UECC**. 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	**Deferred**. 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	**Poison**. 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.
42:41	**RESERV41**. Reset: Cold,0h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
40	**Scrub**. Reset: Cold,0. 1=The error was the result of a scrub operation.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
39:38	**RESERV38**. Reset: Cold,0h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
37:32	**ErrCoreId**. 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.
31:30	**RESERV30**. Reset: Cold,0h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
29:24	**AddrLsb**. 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[54:0] contains a valid byte address. A value of 6 indicates that MCA::L2::MCA_ADDR_L2[54:6] 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[54:12] contain a valid 4KB memory page and that MCA::L2::MCA_ADDR_L2[11:0] should be ignored by error handling software.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
23:22	**RESERV22**. Reset: Cold,0h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
21:16	**ErrorCodeExt**. 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	**ErrorCode**. 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 49: MCA_STATUS_L2

Error Type	ErrorCode Ext	UC	PCC	TCC	Deferred	Poison	AddrV
MultiHit	0x0	1	1	1	0	0	1
Tag	0x1	0/1	0/1	0/1	0	0	1
Data	0x2	0/1	0/1	0/1	0/1	0	1
Hwa	0x3	1	1	1	0	0	1

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.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst2_aliasMSRLEGACY; MSR0000_040A

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst2_aliasMSR; MSRC000_2022

Bits	Description
63:56	Reserved.
55:0	**ErrorAddr**. 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 50: MCA_ADDR_L2

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.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst2_aliasMSRLEGACY; MSR0000_040B

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst2_aliasMSR; MSRC000_2023

Bits	Description
63	**Valid**. Reset: 1. 1=A valid CntP field is present in this register.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read-only.
62	**CntP**. Reset: 1. 1=A valid threshold counter is present.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read-only.

61	**Locked**. 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	**IntP**. 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	**LvtOffset**. 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	**CntEn**. Reset: 0. 1=Count thresholding errors.
	AccessType: (Core::X86::Msr::HWCR[McStatusWrEn] | !MCA::L2::MCA_MISC0_L2[Locked]) ? Read-write : Read-only.
50:49	**ThresholdIntType**. 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	**Ovrflw**. 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	**ErrCnt**. 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	**BlkPtr**. Read-write. Reset: 00h. 00h=Extended MISC MSR block is not valid. 01h=Extended MSR block is valid.
23:0	Reserved.

MSRC000_2024 [L2 Machine Check Configuration Thread 0] (MCA::L2::MCA_CONFIG_L2)

Reset: 0000_0000_0000_0125h.

Controls configuration of the associated machine check bank.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst2_aliasMSR; MSRC000_2024

Bits	Description
63:39	Reserved.
38:37	**DeferredIntType**. 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	**LogDeferredInMcaStat**. 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	**McaXEnable**. 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.
31:9	Reserved.
8	**McaLsbInStatusSupported**. Read-only. Reset: 1. 1=MCA::L2::MCA_CONFIG_L2[McaLsbInStatusSupported] indicates that AddrLbc is located in McaStatus registers.
7:6	Reserved.
5	**DeferredIntTypeSupported**. 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	**DeferredErrorLoggingSupported**. 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	**McaX**. 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 Thread 0] (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.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst2_aliasMSR; MSRC000_2025

Bits	Description
63:48	**McaType**. Read-only. Reset: 0002h. The McaType of the MCA bank within this IP.
47:44	**InstanceIdHi**. Read-write. Reset: 0h. The HI value instance ID of this IP. This is initialized to a unique ID per instance of this register.
43:32	**HardwareID**. Read-only. Reset: 0B0h. The Hardware ID of the IP associated with this MCA bank.
31:0	**InstanceId**. 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

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst2_aliasMSR; MSRC000_2026

Bits	Description
63:49	Reserved.
48:32	**Syndrome**. 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	**ErrorPriority**. 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	**Length**. 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	**ErrorInformation**. Read-write,Volatile. Reset: Cold,0_0000h. Contains error-specific information about the location of the error. Decoding is available in Table 51 [MCA_SYND_L2].

Table 51: MCA_SYND_L2

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] [4:3] [2:0]	Reserved Index Quarter-line Way
Hwa	[17:0]	Reserved

MSRC000_2028 [L2 Machine Check Deferred Error Status Thread 0] (MCA::L2::MCA_DESTAT_L2)

Reset: Cold,0000_0000_0000_0000h.

Holds status information for the first deferred error seen in this bank.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst2_aliasMSR; MSRC000_2028

Bits	Description
63	**Val**. Read-write,Volatile. Reset: Cold,0. 1=A valid error has been detected (whether it is enabled or not).
62	**Overflow**. 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	**RESERV4**. Read-write. Reset: Cold,0h.
58	**AddrV**. Read-write,Volatile. Reset: Cold,0. 1=MCA::L2::MCA_DEADDR_L2 contains address information associated with the error.
57:54	**RESERV3**. Read-write. Reset: Cold,0h.
53	**SyndV**. 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	**RESERV2**. Read-write. Reset: Cold,00h.
44	**Deferred**. 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.
43:30	**RESERV1**. Read-write. Reset: Cold,0000h.
29:24	**AddrLsb**. 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[54:0] contains a valid byte address. A value of 6 indicates that MCA::L2::MCA_ADDR_L2[54:6] 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[54:12] contain a valid 4KB memory page and that MCA::L2::MCA_ADDR_L2[11:0] should be ignored by error handling software.
23:22	**RESERV0**. Read-write. Reset: Cold,0h.
21:16	**ErrorCodeExt**. Read-write,Volatile. Reset: Cold,00h. Logs an extended error code when an error is detected. This model-specific field is used in conjunction with ErrorCode? to identify the error sub-type for root cause

	analysis.
15:0	**ErrorCode**. Read-write,Volatile. Reset: Cold,0000h. Error code for this error.

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_DESTAT_L2[AddrLsb].

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst2_aliasMSR; MSRC000_2029

Bits	Description
63:56	Reserved.
55:0	**ErrorAddr**. 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_DESTAT_L2[AddrLsb].

MSRC001_0402 [L2 Machine Check Control Mask Thread 0] (MCA::L2::MCA_CTL_MASK_L2)

Read-write. Reset: 0000_0000_0000_0000h.

Inhibit detection of an error source.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst2_aliasMSR; MSRC001_0402

Bits	Description
63:4	Reserved.
3	**Hwa**. Read-write. Reset: 0. Init: BIOS,1. Hardware Assert Error.
2	**Data**. Read-write. Reset: 0. L2M Data Array ECC Error.
1	**Tag**. Read-write. Reset: 0. L2M Tag or State Array ECC Error.
0	**MultiHit**. Read-write. Reset: 0. L2M Tag Multiple-Way-Hit error.

3.2.5.4 DE

MSR0000_040C...MSRC000_2030 [DE Machine Check Control Thread 0] (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.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst3_aliasMSRLEGACY; MSR0000_040C

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst3_aliasMSR; MSRC000_2030

Bits	Description
63:10	Reserved.
9	**HwAssertMca**. Read-write. Reset: 0. Hardware Assertion MCA Error
8	**OCBQ**. Read-write. Reset: 0. Micro-op buffer parity error.
7	**UcSeq**. Read-write. Reset: 0. Patch RAM sequencer parity error
6	**UcDat**. Read-write. Reset: 0. Patch RAM data parity error
5	**Faq**. Read-write. Reset: 0. Fetch address FIFO parity error
4	**Idq**. Read-write. Reset: 0. Instruction dispatch queue parity error
3	**UopQ**. Read-write. Reset: 0. Micro-op Queue parity error
2	**Ibq**. Read-write. Reset: 0. IBB Register File parity error
1	**OcDat**. Read-write. Reset: 0. Micro-op cache DATA Array parity error
0	**OcTag**. Read-write. Reset: 0. Micro-op cache TAG Array 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.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst3_aliasMSRLEGACY; MSR0000_040D

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst3_aliasMSR; MSRC000_2031

Bits	Description
63	**Val**. 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	**Overflow**. 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	**UC**. 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	**En**. 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	**MiscV**. Reset: Cold,0. 1=Valid thresholding in MCA::DE::MCA_MISC0_DE. In certain modes, MISC registers are owned by platform firmware and will RAZ when read by non-SMM code. Therefore, it is possible for MiscV=1 and the MISC register to read as all zeros.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
58	**AddrV**. 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	**PCC**. 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	**ErrCoreIdVal**. 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	**TCC**. Reset: Cold,0. 1=Hardware context of the process thread to which the error was reported may have been corrupted. Continued operation of the thread may have unpredictable results. The thread 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	**RESERV54**. Reset: Cold,0. MCA_STATUS Register Reserved bit.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
53	**SyndV**. 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	Reserved.
51:47	**RESERV47**. Reset: Cold,00h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
46	**CECC**. 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	**UECC**. 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	**Deferred**. 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	**Poison**. 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.
42:41	**RESERV41**. Reset: Cold,0h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
40	**Scrub**. Reset: Cold,0. 1=The error was the result of a scrub operation.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
39:38	**RESERV38**. Reset: Cold,0h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
37:32	**ErrCoreId**. 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.
31:30	**RESERV30**. Reset: Cold,0h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
29:24	**AddrLsb**. 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[54:0] contains a valid byte address. A value of 6 indicates that MCA::DE::MCA_ADDR_DE[54:6] 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[54:12] contain a valid 4KB memory page and that MCA::DE::MCA_ADDR_DE[11:0] should be ignored by error handling software.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
23:22	**RESERV22**. Reset: Cold,0h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
21:16	**ErrorCodeExt**. 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	**ErrorCode**. 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 52: MCA_STATUS_DE

Error Type	ErrorCode Ext	UC	PCC	TCC	Deferred	Poison	AddrV
OcTag	0x0					0	0
OcDat	0x1					0	0
Ibq	0x2					0	0
UopQ	0x3					0	0
Idq	0x4					0	0
Faq	0x5					0	0
UcDat	0x6					0	0
UcSeq	0x7					0	0
OCBQ	0x8					0	0
HwAssertMca	0x9					0	0

MSR0000_040E...MSRC000_2032 [DE Machine Check Address Thread 0] (MCA::DE::MCA_ADDR_DE)

Read-only. 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.	
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst3_aliasMSRLEGACY; MSR0000_040E	
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst3_aliasMSR; MSRC000_2032	
Bits	**Description**
63:0	**ErrorAddr**. Read-only. Reset: Cold,0000_0000_0000_0000h. Contains the address, if any, associated with the error logged in MCA::DE::MCA_STATUS_DE.

Table 53: MCA_ADDR_DE

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
HwAssertMca	[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.	
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst3_aliasMSRLEGACY; MSR0000_040F	
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst3_aliasMSR; MSRC000_2033	
Bits	**Description**
63	**Valid**. Reset: 1. 1=A valid CntP field is present in this register.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read-only.
62	**CntP**. Reset: 1. 1=A valid threshold counter is present.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read-only.
61	**Locked**. 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	**IntP**. 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::DE::MCA_MISC0_DE[Locked]) ? Read-write : Read-only.
59:56	Reserved.
55:52	**LvtOffset**. 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	**CntEn**. Reset: 0. 1=Count thresholding errors.
	AccessType: (Core::X86::Msr::HWCR[McStatusWrEn] | !MCA::DE::MCA_MISC0_DE[Locked]) ? Read-write : Read-only.
50:49	**ThresholdIntType**. 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	**Ovrflw**. 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	**ErrCnt**. 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	**BlkPtr**. Read-write. Reset: 00h. 00h=Extended MISC MSR block is not valid. 01h=Extended MSR block is valid.
23:0	Reserved.

MSRC000_2034 [DE Machine Check Configuration Thread 0] (MCA::DE::MCA_CONFIG_DE)

Reset: 0000_0002_0000_0121h.

Controls configuration of the associated machine check bank.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst3_aliasMSR; MSRC000_2034

Bits	Description
63:39	Reserved.
38:37	**DeferredIntType**. 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	**McaXEnable**. 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.
31:9	Reserved.
8	**McaLsbInStatusSupported**. Read-only. Reset: 1. 1=MCA::DE::MCA_CONFIG_DE[McaLsbInStatusSupported] indicates that AddrLbc is located in McaStatus registers.
7:6	Reserved.
5	**DeferredIntTypeSupported**. 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	**DeferredErrorLoggingSupported**. 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	**McaX**. 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 Thread 0] (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.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst3_aliasMSR; MSRC000_2035

Bits	Description
63:48	**McaType**. Read-only. Reset: 0003h. The McaType of the MCA bank within this IP.
47:44	**InstanceIdHi**. Read-write. Reset: 0h. The HI value instance ID of this IP. This is initialized to a unique ID per instance of this register.
43:32	**HardwareID**. Read-only. Reset: 0B0h. The Hardware ID of the IP associated with this MCA bank.
31:0	**InstanceId**. 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

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst3_aliasMSR; MSRC000_2036

Bits	Description
63:33	Reserved.
32	**Syndrome**. 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	**ErrorPriority**. 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	**Length**. 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	**ErrorInformation**. Read-write,Volatile. Reset: Cold,0_0000h. Contains error-specific information about the location of the error. Decoding is available in Table 54 [MCA_SYND_DE].

Table 54: MCA_SYND_DE

Error Type	Bits	Description
OcTag	[17:0]	Reserved
OcDat	[17:0]	Reserved
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
HwAssertMca	[17:0]	Reserved

MSRC001_0403 [DE Machine Check Control Mask Thread 0] (MCA::DE::MCA_CTL_MASK_DE)

Read-write. Reset: 0000_0000_0000_0000h.

Inhibit detection of an error source.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst3_aliasMSR; MSRC001_0403

Bits	Description
63:10	Reserved.
9	**HwAssertMca**. Read-write. Reset: 0. Hardware Assertion MCA Error
8	**OCBQ**. Read-write. Reset: 0. Micro-op buffer parity error.
7	**UcSeq**. Read-write. Reset: 0. Patch RAM sequencer parity error
6	**UcDat**. Read-write. Reset: 0. Patch RAM data parity error
5	**Faq**. Read-write. Reset: 0. Fetch address FIFO parity error
4	**Idq**. Read-write. Reset: 0. Instruction dispatch queue parity error
3	**UopQ**. Read-write. Reset: 0. Micro-op Queue parity error
2	**Ibq**. Read-write. Reset: 0. IBB Register File parity error
1	**OcDat**. Read-write. Reset: 0. Micro-op cache DATA Array parity error
0	**OcTag**. Read-write. Reset: 0. Micro-op cache TAG Array parity error

3.2.5.5 EX

MSR0000_0414...MSRC000_2050 [EX Machine Check Control Thread 0] (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.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst5_aliasMSRLEGACY; MSR0000_0414

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst5_aliasMSR; MSRC000_2050

Bits	Description
63:14	Reserved.
13	**RETMAP**. Read-write. Reset: 0. Retire Map parity error.
12	**SPECMAP**. Read-write. Reset: 0. Spec Map parity error.
11	**HWA**. Read-write. Reset: 0. Hardware Assertion Error.
10	**BBQ**. Read-write. Reset: 0. Branch Buffer Queue (BBQ) parity error.
9	**SQ**. Read-write. Reset: 0. Scheduler Queue parity error.
8	**STATQ**. Read-write. Reset: 0. Retire status queue parity error.
7	**RETDISP**. Read-write. Reset: 0. Retire Dispatch Queue parity error.
6	**CHKPTQ**. Read-write. Reset: 0. Checkpoint Queue (AKA Map_DispQ) parity error.
5	**PLDAL**. Read-write. Reset: 0. EX payload parity error.
4	**PLDAG**. Read-write. Reset: 0. Address generator payload parity error.
3	**IDRF**. Read-write. Reset: 0. Immediate displacement register file parity error.
2	**FRF**. Read-write. Reset: 0. Flag register file (FRF) parity error.
1	**PRF**. Read-write. Reset: 0. Physical register file (PRF) parity error.
0	**WDT**. Read-write. Reset: 0. Watchdog Timeout.

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.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst5_aliasMSRLEGACY; MSR0000_0415

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst5_aliasMSR; MSRC000_2051

Bits	Description
63	**Val**. 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	**Overflow**. 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	**UC**. 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	**En**. 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::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
59	**MiscV**. Reset: Cold,0. 1=Valid thresholding in MCA::EX::MCA_MISC0_EX. In certain modes, MISC registers are owned by platform firmware and will RAZ when read by non-SMM code. Therefore, it is possible for MiscV=1 and the MISC register to read as all zeros.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
58	**AddrV**. Reset: Cold,0. 1=MCA::EX::MCA_ADDR_EX contains address information associated with the error.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
57	**PCC**. 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	**ErrCoreIdVal**. 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	**TCC**. Reset: Cold,0. 1=Hardware context of the process thread to which the error was reported may have been corrupted. Continued operation of the thread may have unpredictable results. The thread must be terminated. Only meaningful when MCA::EX::MCA_STATUS_EX[PCC]=0.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
54	**RESERV54**. Reset: Cold,0. MCA_STATUS Register Reserved bit.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
53	**SyndV**. 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::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
52	Reserved.
51:47	**RESERV47**. Reset: Cold,00h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
46	**CECC**. 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	**UECC**. 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	**Deferred**. 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	**Poison**. 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.
42:41	**RESERV41**. Reset: Cold,0h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
40	**Scrub**. Reset: Cold,0. 1=The error was the result of a scrub operation.

	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
39:38	**RESERV38**. Reset: Cold,0h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
37:32	**ErrCoreId**. 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.
31:30	**RESERV30**. Reset: Cold,0h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
29:24	**AddrLsb**. 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.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
23:22	**RESERV22**. Reset: Cold,0h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
21:16	**ErrorCodeExt**. 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	**ErrorCode**. 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 55: MCA_STATUS_EX

Error Type	ErrorCode Ext	UC	PCC	TCC	Deferred	Poison	AddrV
WDT	0x0	1	1	1	0	0	1
PRF	0x1	1	1	1	0	0	0
FRF	0x2	1	1	1	0	0	0
IDRF	0x3	1	1	1	0	0	0
PLDAG	0x4	1	1	1	0	0	0
PLDAL	0x5	1	1	1	0	0	0
CHKPTQ	0x6	1	1	1	0	0	0
RETDISP	0x7	1	1	1	0	0	0
STATQ	0x8	1	1	1	0	0	0
SQ	0x9	1	1	1	0	0	0
BBQ	0xa	1	1	1	0	0	0
HWA	0xb	1	1	1	0	0	0
SPECMAP	0xc	1	1	1	0	0	0
RETMAP	0xd	1	1	1	0	0	0

MSR0000_0416...MSRC000_2052 [EX Machine Check Address Thread 0] (MCA::EX::MCA_ADDR_EX)

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.
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst5_aliasMSRLEGACY; MSR0000_0416
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst5_aliasMSR; MSRC000_2052

Bits	Description
63:57	Reserved.
56:0	**ErrorAddr**. Read-write,Volatile. Reset: Cold,000_0000_0000_0000h. Unless otherwise specified by an error, contains the address associated with the error logged in MCA::EX::MCA_STATUS_EX. For physical addresses, the most significant bit is given by Core::X86::Cpuid::LongModeInfo[PhysAddrSize].

Table 56: MCA_ADDR_EX

Error Type	Bits	Description
WDT	[56:0]	Instruction Pointer (RIP)
PRF	[56:0]	Reserved
FRF	[56:0]	Reserved
IDRF	[56:0]	Reserved
PLDAG	[56:0]	Reserved
PLDAL	[56:0]	Reserved
CHKPTQ	[56:0]	Reserved
RETDISP	[56:0]	Reserved
STATQ	[56:0]	Reserved
SQ	[56:0]	Reserved
BBQ	[56:0]	Reserved
HWA	[56:0]	Reserved
SPECMAP	[56:0]	Reserved
RETMAP	[56:0]	Reserved

MSR0000_0417...MSRC000_2053 [EX Machine Check Miscellaneous 0 Thread 0] (MCA::EX::MCA_MISC0_EX)

Log miscellaneous information associated with errors.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst5_aliasMSRLEGACY; MSR0000_0417

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst5_aliasMSR; MSRC000_2053

Bits	Description
63	**Valid**. Reset: 1. 1=A valid CntP field is present in this register.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read-only.
62	**CntP**. Reset: 1. 1=A valid threshold counter is present.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read-only.
61	**Locked**. 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	**IntP**. 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::EX::MCA_MISC0_EX[Locked]) ? Read-write : Read-only.
59:56	Reserved.
55:52	**LvtOffset**. 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::EX::MCA_MISC0_EX[Locked]) ? Read-write : Read-only.
51	**CntEn**. Reset: 0. 1=Count thresholding errors.
	AccessType: (Core::X86::Msr::HWCR[McStatusWrEn] | !MCA::EX::MCA_MISC0_EX[Locked]) ? Read-write : Read-only.
50:49	**ThresholdIntType**. 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::EX::MCA_MISC0_EX[Locked]) ? Read-write : Read-only.
48	**Ovrflw**. 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::EX::MCA_MISC0_EX[Locked]) ? Read-write : Read-only.
47:44	Reserved.
43:32	**ErrCnt**. 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::EX::MCA_MISC0_EX[Locked]) ? Read-write : Read-only.
31:24	**BlkPtr**. Read-write. Reset: 00h. 00h=Extended MISC MSR block is not valid. 01h=Extended MSR block is valid.
23:0	Reserved.

MSRC000_2054 [EX Machine Check Configuration Thread 0] (MCA::EX::MCA_CONFIG_EX)

Reset: 0000_0002_0000_0121h.

Controls configuration of the associated machine check bank.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst5_aliasMSR; MSRC000_2054

Bits	Description
63:39	Reserved.
38:37	**DeferredIntType**. 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	**McaXEnable**. 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.
31:9	Reserved.
8	**McaLsbInStatusSupported**. Read-only. Reset: 1. 1=MCA::EX::MCA_CONFIG_EX[McaLsbInStatusSupported] indicates that AddrLbc is located in McaStatus registers.
7:6	Reserved.
5	**DeferredIntTypeSupported**. 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	**DeferredErrorLoggingSupported**. 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	**McaX**. 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 Thread 0] (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.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst5_aliasMSR; MSRC000_2055

Bits	Description
63:48	**McaType**. Read-only. Reset: 0005h. The McaType of the MCA bank within this IP.
47:44	**InstanceIdHi**. Read-write. Reset: 0h. The HI value instance ID of this IP. This is initialized to a unique ID per instance of this register.
43:32	**HardwareID**. Read-only. Reset: 0B0h. The Hardware ID of the IP associated with this MCA bank.
31:0	**InstanceId**. 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

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst5_aliasMSR; MSRC000_2056

Bits	Description
63:33	Reserved.
32	**Syndrome**. 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	**ErrorPriority**. 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	**Length**. 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	**ErrorInformation**. Read-write,Volatile. Reset: Cold,0_0000h. Contains error-specific information about the location of the error. Decoding is available in Table 57 [MCA_SYND_EX].

Table 57: MCA_SYND_EX

Error Type	Bits	Description
WDT	[17:0]	Reserved
PRF	[17:0]	Reserved
FRF	[17:0]	Reserved
IDRF	[17:0]	Reserved
PLDAG	[17:0]	Reserved
PLDAL	[17:0]	Reserved
CHKPTQ	[17:0]	Reserved
RETDISP	[17:0]	Reserved
STATQ	[17:0]	Reserved
SQ	[17:0]	Reserved
BBQ	[17:0]	Reserved
HWA	[17:0]	Reserved
SPECMAP	[17:0]	Reserved
RETMAP	[17:0]	Reserved

MSRC001_0405 [EX Machine Check Control Mask Thread 0] (MCA::EX::MCA_CTL_MASK_EX)	
Read-write. Reset: 0000_0000_0000_0000h.	
Inhibit detection of an error source.	
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst5_aliasMSR; MSRC001_0405	
Bits	**Description**
63:14	Reserved.
13	**RETMAP**. Read-write. Reset: 0. Retire Map parity error.
12	**SPECMAP**. Read-write. Reset: 0. Spec Map parity error.
11	**HWA**. Read-write. Reset: 0. Hardware Assertion Error.
10	**BBQ**. Read-write. Reset: 0. Branch Buffer Queue (BBQ) parity error.
9	**SQ**. Read-write. Reset: 0. Scheduler Queue parity error.
8	**STATQ**. Read-write. Reset: 0. Retire status queue parity error.
7	**RETDISP**. Read-write. Reset: 0. Retire Dispatch Queue parity error.
6	**CHKPTQ**. Read-write. Reset: 0. Checkpoint Queue (AKA Map_DispQ) parity error.
5	**PLDAL**. Read-write. Reset: 0. EX payload parity error.
4	**PLDAG**. Read-write. Reset: 0. Address generator payload parity error.
3	**IDRF**. Read-write. Reset: 0. Immediate displacement register file parity error.
2	**FRF**. Read-write. Reset: 0. Flag register file (FRF) parity error.
1	**PRF**. Read-write. Reset: 0. Physical register file (PRF) parity error.
0	**WDT**. Read-write. Reset: 0. Watchdog Timeout.

3.2.5.6 FP

MSR0000_0418...MSRC000_2060 [FP Machine Check Control Thread 0] (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.	
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst6_aliasMSRLEGACY; MSR0000_0418	
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst6_aliasMSR; MSRC000_2060	
Bits	**Description**
63:7	Reserved.
6	**HWA**. Read-write. Reset: 0. Hardware assertion.
5	**SRF**. Read-write. Reset: 0. Status register file (SRF) parity error.
4	**RQ**. Read-write. Reset: 0. Retire queue (RQ) parity error.
3	**NSQ**. Read-write. Reset: 0. NSQ parity error.
2	**SCH**. Read-write. Reset: 0. Schedule queue parity error.
1	**FL**. Read-write. Reset: 0. Freelist (FL) parity error.
0	**PRF**. 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.	
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst6_aliasMSRLEGACY; MSR0000_0419	
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst6_aliasMSR; MSRC000_2061	
Bits	**Description**
63	**Val**. 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	**Overflow**. 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	**UC**. 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	**En**. 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	**MiscV**. Reset: Cold,0. 1=Valid thresholding in MCA::FP::MCA_MISC0_FP. In certain modes, MISC registers are owned by platform firmware and will RAZ when read by non-SMM code. Therefore, it is possible for MiscV=1 and the MISC register to read as all zeros.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
58	**AddrV**. 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	**PCC**. 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	**ErrCoreIdVal**. 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	**TCC**. Reset: Cold,0. 1=Hardware context of the process thread to which the error was reported may have been corrupted. Continued operation of the thread may have unpredictable results. The thread 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	**RESERV54**. Reset: Cold,0. MCA_STATUS Register Reserved bit.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
53	**SyndV**. 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	Reserved.
51:47	**RESERV47**. Reset: Cold,00h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
46	**CECC**. 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	**UECC**. 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	**Deferred**. 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	**Poison**. 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.
42:41	**RESERV41**. Reset: Cold,0h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
40	**Scrub**. Reset: Cold,0. 1=The error was the result of a scrub operation.

	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
39:38	**RESERV38**. Reset: Cold,0h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
37:32	**ErrCoreId**. 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.
31:30	**RESERV30**. Reset: Cold,0h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
29:24	**AddrLsb**. 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[54:0] contains a valid byte address. A value of 6 indicates that MCA::FP::MCA_ADDR_FP[54:6] 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[54:12] contain a valid 4KB memory page and that MCA::FP::MCA_ADDR_FP[11:0] should be ignored by error handling software.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
23:22	**RESERV22**. Reset: Cold,0h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
21:16	**ErrorCodeExt**. 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	**ErrorCode**. 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 58: MCA_STATUS_FP

Error Type	ErrorCode Ext	UC	PCC	TCC	Deferred	Poison	AddrV
PRF	0x0	1	1	1	0	0	0
FL	0x1	1	1	1	0	0	0
SCH	0x2	1	1	1	0	0	0
NSQ	0x3	1	1	1	0	0	0
RQ	0x4	1	1	1	0	0	0
SRF	0x5	1	1	1	0	0	0
HWA	0x6	1	1	1	0	0	0

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.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst6_aliasMSRLEGACY; MSR0000_041A

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst6_aliasMSR; MSRC000_2062

Bits	Description
63:0	**ErrorAddr**. Read-only. Reset: Cold,0000_0000_0000_0000h. Contains the address, if any, associated with the error logged in MCA::FP::MCA_STATUS_FP.

Table 59: MCA_ADDR_FP

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.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst6_aliasMSRLEGACY; MSR0000_041B

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst6_aliasMSR; MSRC000_2063

Bits	Description
63	**Valid**. Reset: 1. 1=A valid CntP field is present in this register.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read-only.
62	**CntP**. Reset: 1. 1=A valid threshold counter is present.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read-only.
61	**Locked**. 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	**IntP**. 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	**LvtOffset**. 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	**CntEn**. Reset: 0. 1=Count thresholding errors.
	AccessType: (Core::X86::Msr::HWCR[McStatusWrEn] | !MCA::FP::MCA_MISC0_FP[Locked]) ? Read-write : Read-only.
50:49	**ThresholdIntType**. 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	**Ovrflw**. 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	**ErrCnt**. 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	**BlkPtr**. Read-write. Reset: 00h. 00h=Extended MISC MSR block is not valid. 01h=Extended MSR block is valid.
23:0	Reserved.

MSRC000_2064 [FP Machine Check Configuration Thread 0] (MCA::FP::MCA_CONFIG_FP)

Reset: 0000_0002_0000_0121h.

Controls configuration of the associated machine check bank.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst6_aliasMSR; MSRC000_2064

Bits	Description
63:39	Reserved.
38:37	**DeferredIntType**. 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	**McaXEnable**. 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.
31:9	Reserved.
8	**McaLsbInStatusSupported**. Read-only. Reset: 1. 1=MCA::FP::MCA_CONFIG_FP[McaLsbInStatusSupported] indicates that AddrLbc is located in McaStatus registers.
7:6	Reserved.
5	**DeferredIntTypeSupported**. 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	**DeferredErrorLoggingSupported**. 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	**McaX**. 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 Thread 0] (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.

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst6_aliasMSR; MSRC000_2065

Bits	Description
63:48	**McaType**. Read-only. Reset: 0006h. The McaType of the MCA bank within this IP.
47:44	**InstanceIdHi**. Read-write. Reset: 0h. The HI value instance ID of this IP. This is initialized to a unique ID per instance of this register.
43:32	**HardwareID**. Read-only. Reset: 0B0h. The Hardware ID of the IP associated with this MCA bank.
31:0	**InstanceId**. 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

_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst6_aliasMSR; MSRC000_2066

Bits	Description
63:33	Reserved.
32	**Syndrome**. 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	**ErrorPriority**. 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	**Length**. 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	**ErrorInformation**. Read-write,Volatile. Reset: Cold,0_0000h. Contains error-specific information about the location of the error. Decoding is available in Table 60 [MCA_SYND_FP].

Table 60: MCA_SYND_FP

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 Thread 0] (MCA::FP::MCA_CTL_MASK_FP)

Read-write. Reset: 0000_0000_0000_0000h.
Inhibit detection of an error source.
_ccd[7:0]_lthree0_core[7:0]_thread[1:0]_inst6_aliasMSR; MSRC001_0406

Bits	Description
63:7	Reserved.
6	**HWA**. Read-write. Reset: 0. Init: BIOS,1. Hardware assertion.
5	**SRF**. Read-write. Reset: 0. Status register file (SRF) parity error.
4	**RQ**. Read-write. Reset: 0. Retire queue (RQ) parity error.
3	**NSQ**. Read-write. Reset: 0. NSQ parity error.
2	**SCH**. Read-write. Reset: 0. Schedule queue parity error.
1	**FL**. Read-write. Reset: 0. Freelist (FL) parity error.
0	**PRF**. Read-write. Reset: 0. Physical register file (PRF) parity error.

3.2.5.7 L3

MSR0000_041C...MSRC000_20E0 [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.
_ccd[7:0]_lthree0_inst7_n[56,48,40,32,24,16,8,0]_aliasMSRLEGACY; MSR0000_041C
_ccd[7:0]_lthree0_inst8_n[57,49,41,33,25,17,9,1]_aliasMSRLEGACY; MSR0000_0420
_ccd[7:0]_lthree0_inst9_n[5[8,0],[42,34,26,18,10,2]]_aliasMSRLEGACY; MSR0000_0424
_ccd[7:0]_lthree0_inst10_n[5[9,1],[43,35,27,19,11,3]]_aliasMSRLEGACY; MSR0000_0428
_ccd[7:0]_lthree0_inst11_n[60,52,44,36,28,20,12,4]_aliasMSRLEGACY; MSR0000_042C

_ccd[7:0]_lthree0_inst12_n[61,53,45,37,29,21,13,5]_aliasMSRLEGACY; MSR0000_0430
_ccd[7:0]_lthree0_inst13_n[62,54,46,38,30,22,14,6]_aliasMSRLEGACY; MSR0000_0434
_ccd[7:0]_lthree0_inst14_n[63,55,47,39,31,23,15,7]_aliasMSRLEGACY; MSR0000_0438
_ccd[7:0]_lthree0_inst7_n[56,48,40,32,24,16,8,0]_aliasMSR; MSRC000_2070
_ccd[7:0]_lthree0_inst8_n[57,49,41,33,25,17,9,1]_aliasMSR; MSRC000_2080
_ccd[7:0]_lthree0_inst9_n[5[8,0],[42,34,26,18,10,2]]_aliasMSR; MSRC000_2090
_ccd[7:0]_lthree0_inst10_n[5[9,1],[43,35,27,19,11,3]]_aliasMSR; MSRC000_20A0
_ccd[7:0]_lthree0_inst11_n[60,52,44,36,28,20,12,4]_aliasMSR; MSRC000_20B0
_ccd[7:0]_lthree0_inst12_n[61,53,45,37,29,21,13,5]_aliasMSR; MSRC000_20C0
_ccd[7:0]_lthree0_inst13_n[62,54,46,38,30,22,14,6]_aliasMSR; MSRC000_20D0
_ccd[7:0]_lthree0_inst14_n[63,55,47,39,31,23,15,7]_aliasMSR; MSRC000_20E0

Bits	Description
63:8	Reserved.
7	**Hwa**. Read-write. Reset: 0. L3 Hardware Assertion.
6	**XiVictimQueue**. Read-write. Reset: 0. L3 Victim Queue Parity Error.
5	**SdpParity**. Read-write. Reset: 0. SDP Parity Error from XI.
4	**DataArray**. Read-write. Reset: 0. L3M Data ECC Error.
3	**MultiHitTag**. Read-write. Reset: 0. L3M Tag Multi-way-hit Error.
2	**Tag**. Read-write. Reset: 0. L3M Tag ECC Error.
1	**MultiHitShadowTag**. Read-write. Reset: 0. Shadow Tag Macro Multi-way-hit Error.
0	**ShadowTag**. Read-write. Reset: 0. Shadow Tag Macro ECC Error.

MSR0000_041D...MSRC000_20E1 [L3 Machine Check Status] (MCA::L3::MCA_STATUS_L3)

Reset: Cold,0000_0000_0000_0000h.

Logs information associated with errors.

_ccd[7:0]_lthree0_inst7_n[56,48,40,32,24,16,8,0]_aliasMSRLEGACY; MSR0000_041D
_ccd[7:0]_lthree0_inst8_n[57,49,41,33,25,17,9,1]_aliasMSRLEGACY; MSR0000_0421
_ccd[7:0]_lthree0_inst9_n[5[8,0],[42,34,26,18,10,2]]_aliasMSRLEGACY; MSR0000_0425
_ccd[7:0]_lthree0_inst10_n[5[9,1],[43,35,27,19,11,3]]_aliasMSRLEGACY; MSR0000_0429
_ccd[7:0]_lthree0_inst11_n[60,52,44,36,28,20,12,4]_aliasMSRLEGACY; MSR0000_042D
_ccd[7:0]_lthree0_inst12_n[61,53,45,37,29,21,13,5]_aliasMSRLEGACY; MSR0000_0431
_ccd[7:0]_lthree0_inst13_n[62,54,46,38,30,22,14,6]_aliasMSRLEGACY; MSR0000_0435
_ccd[7:0]_lthree0_inst14_n[63,55,47,39,31,23,15,7]_aliasMSRLEGACY; MSR0000_0439
_ccd[7:0]_lthree0_inst7_n[56,48,40,32,24,16,8,0]_aliasMSR; MSRC000_2071
_ccd[7:0]_lthree0_inst8_n[57,49,41,33,25,17,9,1]_aliasMSR; MSRC000_2081
_ccd[7:0]_lthree0_inst9_n[5[8,0],[42,34,26,18,10,2]]_aliasMSR; MSRC000_2091
_ccd[7:0]_lthree0_inst10_n[5[9,1],[43,35,27,19,11,3]]_aliasMSR; MSRC000_20A1
_ccd[7:0]_lthree0_inst11_n[60,52,44,36,28,20,12,4]_aliasMSR; MSRC000_20B1
_ccd[7:0]_lthree0_inst12_n[61,53,45,37,29,21,13,5]_aliasMSR; MSRC000_20C1
_ccd[7:0]_lthree0_inst13_n[62,54,46,38,30,22,14,6]_aliasMSR; MSRC000_20D1
_ccd[7:0]_lthree0_inst14_n[63,55,47,39,31,23,15,7]_aliasMSR; MSRC000_20E1

Bits	Description
63	**Val**. 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	**Overflow**. 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	**UC**. 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	**En**. 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::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
59	**MiscV**. Reset: Cold,0. 1=Valid thresholding in MCA::L3::MCA_MISC0_L3. In certain modes, MISC registers are owned by platform firmware and will RAZ when read by non-SMM code. Therefore, it is possible for

	MiscV=1 and the MISC register to read as all zeros.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
58	**AddrV**. Reset: Cold,0. 1=MCA::L3::MCA_ADDR_L3 contains address information associated with the error.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
57	**PCC**. 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	**ErrCoreIdVal**. 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	**TCC**. Reset: Cold,0. 1=Hardware context of the process thread to which the error was reported may have been corrupted. Continued operation of the thread may have unpredictable results. The thread must be terminated. Only meaningful when MCA::L3::MCA_STATUS_L3[PCC]=0.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
54	**RESERV54**. Reset: Cold,0. MCA_STATUS Register Reserved bit.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
53	**SyndV**. 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::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
52	Reserved.
51:47	**RESERV47**. Reset: Cold,00h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
46	**CECC**. 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	**UECC**. 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	**Deferred**. 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	**Poison**. 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.
42:41	**RESERV41**. Reset: Cold,0h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
40	**Scrub**. Reset: Cold,0. 1=The error was the result of a scrub operation.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
39:38	**RESERV38**. Reset: Cold,0h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
37:32	**ErrCoreId**. 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.
31:30	**RESERV30**. Reset: Cold,0h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
29:24	**AddrLsb**. 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[54:0] contains a valid byte address. A value of 6 indicates that MCA::L3::MCA_ADDR_L3[54:6] 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[54:12] contain a valid 4KB memory page and that MCA::L3::MCA_ADDR_L3[11:0] should be ignored by error handling software.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
23:22	**RESERV22**. Reset: Cold,0h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
21:16	**ErrorCodeExt**. 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::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
15:0	**ErrorCode**. 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 61: MCA_STATUS_L3

Error Type	ErrorCode Ext	UC	PCC	TCC	Deferred	Poison	AddrV
ShadowTag	0x0					-	-
MultiHitShadowTag	0x1					-	-
Tag	0x2					-	-
MultiHitTag	0x3					-	-
DataArray	0x4					-	-
SdpParity	0x5					-	-
XiVictimQueue	0x6					-	-
Hwa	0x7					-	-

MSR0000_041E...MSRC000_20E2 (MCA::L3::MCA_ADDR_L3)

Reset: Cold,0000_0000_0000_0000h.

_ccd[7:0]_lthree0_inst7_n[56,48,40,32,24,16,8,0]_aliasMSRLEGACY; MSR0000_041E
_ccd[7:0]_lthree0_inst8_n[57,49,41,33,25,17,9,1]_aliasMSRLEGACY; MSR0000_0422
_ccd[7:0]_lthree0_inst9_n[5[8,0],[42,34,26,18,10,2]]_aliasMSRLEGACY; MSR0000_0426
_ccd[7:0]_lthree0_inst10_n[5[9,1],[43,35,27,19,11,3]]_aliasMSRLEGACY; MSR0000_042A
_ccd[7:0]_lthree0_inst11_n[60,52,44,36,28,20,12,4]_aliasMSRLEGACY; MSR0000_042E
_ccd[7:0]_lthree0_inst12_n[61,53,45,37,29,21,13,5]_aliasMSRLEGACY; MSR0000_0432
_ccd[7:0]_lthree0_inst13_n[62,54,46,38,30,22,14,6]_aliasMSRLEGACY; MSR0000_0436
_ccd[7:0]_lthree0_inst14_n[63,55,47,39,31,23,15,7]_aliasMSRLEGACY; MSR0000_043A
_ccd[7:0]_lthree0_inst7_n[56,48,40,32,24,16,8,0]_aliasMSR; MSRC000_2072
_ccd[7:0]_lthree0_inst8_n[57,49,41,33,25,17,9,1]_aliasMSR; MSRC000_2082
_ccd[7:0]_lthree0_inst9_n[5[8,0],[42,34,26,18,10,2]]_aliasMSR; MSRC000_2092
_ccd[7:0]_lthree0_inst10_n[5[9,1],[43,35,27,19,11,3]]_aliasMSR; MSRC000_20A2
_ccd[7:0]_lthree0_inst11_n[60,52,44,36,28,20,12,4]_aliasMSR; MSRC000_20B2
_ccd[7:0]_lthree0_inst12_n[61,53,45,37,29,21,13,5]_aliasMSR; MSRC000_20C2
_ccd[7:0]_lthree0_inst13_n[62,54,46,38,30,22,14,6]_aliasMSR; MSRC000_20D2
_ccd[7:0]_lthree0_inst14_n[63,55,47,39,31,23,15,7]_aliasMSR; MSRC000_20E2

Bits	Description
63:56	Reserved.
55:0	**ErrorAddr**. 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 62: MCA_ADDR_L3

Error Type	Bits	Description
ShadowTag	[55:16] [15:0]	Reserved 16'b{7'b{Index}, 3'b{Slice}, 6'b{0}}
MultiHitShadowTag	[55:16] [15:0]	Reserved 16'b{7'b{Index}, 3'b{Slice}, 6'b{0}}
Tag	[55:21] [20:0]	Reserved 21'b{1'b{Odd/Even}, 7'b{Index}, 4'b{Bank[3:0]}, 3'b{Slice}, 6'b{0}}
MultiHitTag	[55:21] [20:0]	Reserved 21'b{1'b{Odd/Even}, 7'b{Index}, 4'b{Bank[3:0]}, 3'b{Slice}, 6'b{0}}
DataArray	[55:52] [51:0]	Reserved Physical Address
SdpParity	[55:52] [51:0]	Reserved Physical Address
XiVictimQueue	[55:52] [51:0]	Reserved Physical Address
Hwa	[55:45] [44:0]	Reserved Reserved

MSR0000_041F...MSRC000_20E3 [L3 Machine Check Miscellaneous 0] (MCA::L3::MCA_MISC0_L3)

Log miscellaneous information associated with errors.

_ccd[7:0]_lthree0_inst7_n[56,48,40,32,24,16,8,0]_aliasMSRLEGACY; MSR0000_041F
_ccd[7:0]_lthree0_inst8_n[57,49,41,33,25,17,9,1]_aliasMSRLEGACY; MSR0000_0423
_ccd[7:0]_lthree0_inst9_n[5[8,0],[42,34,26,18,10,2]]_aliasMSRLEGACY; MSR0000_0427
_ccd[7:0]_lthree0_inst10_n[5[9,1],[43,35,27,19,11,3]]_aliasMSRLEGACY; MSR0000_042B
_ccd[7:0]_lthree0_inst11_n[60,52,44,36,28,20,12,4]_aliasMSRLEGACY; MSR0000_042F
_ccd[7:0]_lthree0_inst12_n[61,53,45,37,29,21,13,5]_aliasMSRLEGACY; MSR0000_0433
_ccd[7:0]_lthree0_inst13_n[62,54,46,38,30,22,14,6]_aliasMSRLEGACY; MSR0000_0437
_ccd[7:0]_lthree0_inst14_n[63,55,47,39,31,23,15,7]_aliasMSRLEGACY; MSR0000_043B
_ccd[7:0]_lthree0_inst7_n[56,48,40,32,24,16,8,0]_aliasMSR; MSRC000_2073
_ccd[7:0]_lthree0_inst8_n[57,49,41,33,25,17,9,1]_aliasMSR; MSRC000_2083
_ccd[7:0]_lthree0_inst9_n[5[8,0],[42,34,26,18,10,2]]_aliasMSR; MSRC000_2093
_ccd[7:0]_lthree0_inst10_n[5[9,1],[43,35,27,19,11,3]]_aliasMSR; MSRC000_20A3
_ccd[7:0]_lthree0_inst11_n[60,52,44,36,28,20,12,4]_aliasMSR; MSRC000_20B3
_ccd[7:0]_lthree0_inst12_n[61,53,45,37,29,21,13,5]_aliasMSR; MSRC000_20C3
_ccd[7:0]_lthree0_inst13_n[62,54,46,38,30,22,14,6]_aliasMSR; MSRC000_20D3
_ccd[7:0]_lthree0_inst14_n[63,55,47,39,31,23,15,7]_aliasMSR; MSRC000_20E3

Bits	Description
63	**Valid**. Reset: 1. 1=A valid CntP field is present in this register.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read-only.
62	**CntP**. Reset: 1. 1=A valid threshold counter is present.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read-only.
61	**Locked**. 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	**IntP**. 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::L3::MCA_MISC0_L3[Locked]) ? Read-write : Read-only.
59:56	Reserved.
55:52	**LvtOffset**. 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::L3::MCA_MISC0_L3[Locked]) ? Read-write : Read-only.
51	**CntEn**. Reset: 0. 1=Count thresholding errors.
	AccessType: (Core::X86::Msr::HWCR[McStatusWrEn] | !MCA::L3::MCA_MISC0_L3[Locked]) ? Read-write : Read-only.
50:49	**ThresholdIntType**. 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::L3::MCA_MISC0_L3[Locked]) ? Read-write : Read-only.
48	**Ovrflw**. 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::L3::MCA_MISC0_L3[Locked]) ? Read-write : Read-only.
47:44	Reserved.
43:32	**ErrCnt**. 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	**BlkPtr**. Read-write. Reset: 00h. 00h=Extended MISC MSR block is not valid. 01h=Extended MSR block is valid.
23:0	Reserved.

MSRC000_20[7...E]4 [L3 Machine Check Configuration] (MCA::L3::MCA_CONFIG_L3)

Reset: 0000_0000_0000_0125h.

Controls configuration of the associated machine check bank.

_ccd[7:0]_lthree0_inst7_n[56,48,40,32,24,16,8,0]_aliasMSR; MSRC000_2074
_ccd[7:0]_lthree0_inst8_n[57,49,41,33,25,17,9,1]_aliasMSR; MSRC000_2084
_ccd[7:0]_lthree0_inst9_n[5[8,0],[42,34,26,18,10,2]]_aliasMSR; MSRC000_2094
_ccd[7:0]_lthree0_inst10_n[5[9,1],[43,35,27,19,11,3]]_aliasMSR; MSRC000_20A4
_ccd[7:0]_lthree0_inst11_n[60,52,44,36,28,20,12,4]_aliasMSR; MSRC000_20B4
_ccd[7:0]_lthree0_inst12_n[61,53,45,37,29,21,13,5]_aliasMSR; MSRC000_20C4
_ccd[7:0]_lthree0_inst13_n[62,54,46,38,30,22,14,6]_aliasMSR; MSRC000_20D4
_ccd[7:0]_lthree0_inst14_n[63,55,47,39,31,23,15,7]_aliasMSR; MSRC000_20E4

Bits	Description
63:39	Reserved.
38:37	**DeferredIntType**. 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	**LogDeferredInMcaStat**. 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	**McaXEnable**. 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.
31:9	Reserved.
8	**McaLsbInStatusSupported**. Read-only. Reset: 1. 1=MCA::L3::MCA_CONFIG_L3[McaLsbInStatusSupported] indicates that AddrLbc is located in McaStatus registers.
7:6	Reserved.
5	**DeferredIntTypeSupported**. 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	**DeferredErrorLoggingSupported**. 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	**McaX**. 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...E]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.

_ccd[7:0]_lthree0_inst7_n[56,48,40,32,24,16,8,0]_aliasMSR; MSRC000_2075
_ccd[7:0]_lthree0_inst8_n[57,49,41,33,25,17,9,1]_aliasMSR; MSRC000_2085
_ccd[7:0]_lthree0_inst9_n[5[8,0],[42,34,26,18,10,2]]_aliasMSR; MSRC000_2095
_ccd[7:0]_lthree0_inst10_n[5[9,1],[43,35,27,19,11,3]]_aliasMSR; MSRC000_20A5
_ccd[7:0]_lthree0_inst11_n[60,52,44,36,28,20,12,4]_aliasMSR; MSRC000_20B5
_ccd[7:0]_lthree0_inst12_n[61,53,45,37,29,21,13,5]_aliasMSR; MSRC000_20C5
_ccd[7:0]_lthree0_inst13_n[62,54,46,38,30,22,14,6]_aliasMSR; MSRC000_20D5
_ccd[7:0]_lthree0_inst14_n[63,55,47,39,31,23,15,7]_aliasMSR; MSRC000_20E5

Bits	Description
63:48	**McaType**. Read-only. Reset: 0007h. The McaType of the MCA bank within this IP.
47:44	**InstanceIdHi**. Read-write. Reset: 0h. The HI value instance ID of this IP. This is initialized to a unique ID per instance of this register.
43:32	**HardwareID**. Read-only. Reset: 0B0h. The Hardware ID of the IP associated with this MCA bank.
31:0	**InstanceId**. Read-write. Reset: 0000_0000h. The instance ID of this IP. This is initialized to a unique ID per instance of this register.
	Init: _ccd0_lthree0_inst7_n0_aliasMSR: 2035_0000h Init: _ccd0_lthree0_inst8_n1_aliasMSR: 2035_0100h Init: _ccd0_lthree0_inst9_n2_aliasMSR: 2035_0200h Init: _ccd0_lthree0_inst10_n3_aliasMSR: 2035_0300h Init: _ccd0_lthree0_inst11_n4_aliasMSR: 2035_0400h Init: _ccd0_lthree0_inst12_n5_aliasMSR: 2035_0500h Init: _ccd0_lthree0_inst13_n6_aliasMSR: 2035_0600h Init: _ccd0_lthree0_inst14_n7_aliasMSR: 2035_0700h Init: _ccd1_lthree0_inst7_n8_aliasMSR: 20B5_0000h Init: _ccd1_lthree0_inst8_n9_aliasMSR: 20B5_0100h Init: _ccd1_lthree0_inst9_n10_aliasMSR: 20B5_0200h Init: _ccd1_lthree0_inst10_n11_aliasMSR: 20B5_0300h Init: _ccd1_lthree0_inst11_n12_aliasMSR: 20B5_0400h Init: _ccd1_lthree0_inst12_n13_aliasMSR: 20B5_0500h Init: _ccd1_lthree0_inst13_n14_aliasMSR: 20B5_0600h Init: _ccd1_lthree0_inst14_n15_aliasMSR: 20B5_0700h Init: _ccd2_lthree0_inst7_n16_aliasMSR: 2135_0000h Init: _ccd2_lthree0_inst8_n17_aliasMSR: 2135_0100h Init: _ccd2_lthree0_inst9_n18_aliasMSR: 2135_0200h Init: _ccd2_lthree0_inst10_n19_aliasMSR: 2135_0300h Init: _ccd2_lthree0_inst11_n20_aliasMSR: 2135_0400h

	Init: _ccd2_lthree0_inst12_n21_aliasMSR: 2135_0500h Init: _ccd2_lthree0_inst13_n22_aliasMSR: 2135_0600h Init: _ccd2_lthree0_inst14_n23_aliasMSR: 2135_0700h Init: _ccd3_lthree0_inst7_n24_aliasMSR: 21B5_0000h Init: _ccd3_lthree0_inst8_n25_aliasMSR: 21B5_0100h Init: _ccd3_lthree0_inst9_n26_aliasMSR: 21B5_0200h Init: _ccd3_lthree0_inst10_n27_aliasMSR: 21B5_0300h Init: _ccd3_lthree0_inst11_n28_aliasMSR: 21B5_0400h Init: _ccd3_lthree0_inst12_n29_aliasMSR: 21B5_0500h Init: _ccd3_lthree0_inst13_n30_aliasMSR: 21B5_0600h Init: _ccd3_lthree0_inst14_n31_aliasMSR: 21B5_0700h Init: _ccd4_lthree0_inst7_n32_aliasMSR: 2235_0000h Init: _ccd4_lthree0_inst8_n33_aliasMSR: 2235_0100h Init: _ccd4_lthree0_inst9_n34_aliasMSR: 2235_0200h Init: _ccd4_lthree0_inst10_n35_aliasMSR: 2235_0300h Init: _ccd4_lthree0_inst11_n36_aliasMSR: 2235_0400h Init: _ccd4_lthree0_inst12_n37_aliasMSR: 2235_0500h Init: _ccd4_lthree0_inst13_n38_aliasMSR: 2235_0600h Init: _ccd4_lthree0_inst14_n39_aliasMSR: 2235_0700h Init: _ccd5_lthree0_inst7_n40_aliasMSR: 22B5_0000h Init: _ccd5_lthree0_inst8_n41_aliasMSR: 22B5_0100h Init: _ccd5_lthree0_inst9_n42_aliasMSR: 22B5_0200h Init: _ccd5_lthree0_inst10_n43_aliasMSR: 22B5_0300h Init: _ccd5_lthree0_inst11_n44_aliasMSR: 22B5_0400h Init: _ccd5_lthree0_inst12_n45_aliasMSR: 22B5_0500h Init: _ccd5_lthree0_inst13_n46_aliasMSR: 22B5_0600h Init: _ccd5_lthree0_inst14_n47_aliasMSR: 22B5_0700h Init: _ccd6_lthree0_inst7_n48_aliasMSR: 2335_0000h Init: _ccd6_lthree0_inst8_n49_aliasMSR: 2335_0100h Init: _ccd6_lthree0_inst9_n50_aliasMSR: 2335_0200h Init: _ccd6_lthree0_inst10_n51_aliasMSR: 2335_0300h Init: _ccd6_lthree0_inst11_n52_aliasMSR: 2335_0400h Init: _ccd6_lthree0_inst12_n53_aliasMSR: 2335_0500h Init: _ccd6_lthree0_inst13_n54_aliasMSR: 2335_0600h Init: _ccd6_lthree0_inst14_n55_aliasMSR: 2335_0700h Init: _ccd7_lthree0_inst7_n56_aliasMSR: 23B5_0000h Init: _ccd7_lthree0_inst8_n57_aliasMSR: 23B5_0100h Init: _ccd7_lthree0_inst9_n58_aliasMSR: 23B5_0200h Init: _ccd7_lthree0_inst10_n59_aliasMSR: 23B5_0300h Init: _ccd7_lthree0_inst11_n60_aliasMSR: 23B5_0400h Init: _ccd7_lthree0_inst12_n61_aliasMSR: 23B5_0500h Init: _ccd7_lthree0_inst13_n62_aliasMSR: 23B5_0600h Init: _ccd7_lthree0_inst14_n63_aliasMSR: 23B5_0700h

MSRC000_20[7...E]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

_ccd[7:0]_lthree0_inst7_n[56,48,40,32,24,16,8,0]_aliasMSR; MSRC000_2076
_ccd[7:0]_lthree0_inst8_n[57,49,41,33,25,17,9,1]_aliasMSR; MSRC000_2086
_ccd[7:0]_lthree0_inst9_n[5[8,0],[42,34,26,18,10,2]]_aliasMSR; MSRC000_2096
_ccd[7:0]_lthree0_inst10_n[5[9,1],[43,35,27,19,11,3]]_aliasMSR; MSRC000_20A6
_ccd[7:0]_lthree0_inst11_n[60,52,44,36,28,20,12,4]_aliasMSR; MSRC000_20B6
_ccd[7:0]_lthree0_inst12_n[61,53,45,37,29,21,13,5]_aliasMSR; MSRC000_20C6
_ccd[7:0]_lthree0_inst13_n[62,54,46,38,30,22,14,6]_aliasMSR; MSRC000_20D6
_ccd[7:0]_lthree0_inst14_n[63,55,47,39,31,23,15,7]_aliasMSR; MSRC000_20E6

Bits	Description
63:49	Reserved.
48:32	**Syndrome**. 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	**ErrorPriority**. 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	**Length**. 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	**ErrorInformation**. Read-write,Volatile. Reset: Cold,0_0000h. Contains error-specific information about the location of the error. Decoding is available in Table 63 [MCA_SYND_L3].

Table 63: MCA_SYND_L3

Error Type	Bits	Description
ShadowTag	[17:16]	Reserved
	[15:8]	Pack
	[7:3]	Reserved
	[2:0]	Way
MultiHitShadowTag	[17:12]	Reserved
	[11:8]	Pack
	[7:0]	Reserved
Tag	[17:14]	4'b1100
	[13]	1'b0
	[12]	PA[20]
	[11]	Bank[3]
	[10:8]	Bank[2:0]
	[7:4]	Reserved
	[3:0]	Way
MultiHitTag	[17:0]	Reserved
DataArray	[17:14]	4'b1100
	[13]	1'b0
	[12]	1'b0
	[11]	1'b0
	[10:8]	Bank[2:0]
	[7:4]	Reserved
	[3:0]	Way
SdpParity	[17:0]	Reserved
XiVictimQueue	[17:0]	Reserved
Hwa	[17:0]	Reserved

MSRC000_20[7...E]8 [L3 Machine Check Deferred Error Status] (MCA::L3::MCA_DESTAT_L3)

Reset: Cold,0000_0000_0000_0000h.

Holds status information for the first deferred error seen in this bank.

_ccd[7:0]_lthree0_inst7_n[56,48,40,32,24,16,8,0]_aliasMSR; MSRC000_2078
_ccd[7:0]_lthree0_inst8_n[57,49,41,33,25,17,9,1]_aliasMSR; MSRC000_2088
_ccd[7:0]_lthree0_inst9_n[5[8,0],[42,34,26,18,10,2]]_aliasMSR; MSRC000_2098
_ccd[7:0]_lthree0_inst10_n[5[9,1],[43,35,27,19,11,3]]_aliasMSR; MSRC000_20A8
_ccd[7:0]_lthree0_inst11_n[60,52,44,36,28,20,12,4]_aliasMSR; MSRC000_20B8
_ccd[7:0]_lthree0_inst12_n[61,53,45,37,29,21,13,5]_aliasMSR; MSRC000_20C8
_ccd[7:0]_lthree0_inst13_n[62,54,46,38,30,22,14,6]_aliasMSR; MSRC000_20D8
_ccd[7:0]_lthree0_inst14_n[63,55,47,39,31,23,15,7]_aliasMSR; MSRC000_20E8

Bits	Description
63	**Val**. Read-write,Volatile. Reset: Cold,0. 1=A valid error has been detected (whether it is enabled or not).
62	**Overflow**. 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	**RESERV4**. Read-write. Reset: Cold,0h.
58	**AddrV**. Read-write,Volatile. Reset: Cold,0. 1=MCA::L3::MCA_DEADDR_L3 contains address information associated with the error.
57:54	**RESERV3**. Read-write. Reset: Cold,0h.
53	**SyndV**. 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	**RESERV2**. Read-write. Reset: Cold,00h.
44	**Deferred**. 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.
43:30	**RESERV1**. Read-write. Reset: Cold,0000h.
29:24	**AddrLsb**. 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[54:0] contains a valid byte address. A value of 6 indicates that MCA::L3::MCA_ADDR_L3[54:6] 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[54:12] contain a valid 4KB memory page and that MCA::L3::MCA_ADDR_L3[11:0] should be ignored by error handling software.
23:22	**RESERV0**. Read-write. Reset: Cold,0h.
21:16	**ErrorCodeExt**. Read-write,Volatile. Reset: Cold,00h. Logs an extended error code when an error is detected. This model-specific field is used in conjunction with ErrorCode? to identify the error sub-type for root cause analysis.
15:0	**ErrorCode**. Read-write,Volatile. Reset: Cold,0000h. Error code for this error.

MSRC000_20[7...E]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_DESTAT_L3[AddrLsb].

_ccd[7:0]_lthree0_inst7_n[56,48,40,32,24,16,8,0]_aliasMSR; MSRC000_2079
_ccd[7:0]_lthree0_inst8_n[57,49,41,33,25,17,9,1]_aliasMSR; MSRC000_2089
_ccd[7:0]_lthree0_inst9_n[5[8,0],[42,34,26,18,10,2]]_aliasMSR; MSRC000_2099
_ccd[7:0]_lthree0_inst10_n[5[9,1],[43,35,27,19,11,3]]_aliasMSR; MSRC000_20A9
_ccd[7:0]_lthree0_inst11_n[60,52,44,36,28,20,12,4]_aliasMSR; MSRC000_20B9
_ccd[7:0]_lthree0_inst12_n[61,53,45,37,29,21,13,5]_aliasMSR; MSRC000_20C9
_ccd[7:0]_lthree0_inst13_n[62,54,46,38,30,22,14,6]_aliasMSR; MSRC000_20D9
_ccd[7:0]_lthree0_inst14_n[63,55,47,39,31,23,15,7]_aliasMSR; MSRC000_20E9

Bits	Description
63:56	Reserved.
55:0	**ErrorAddr**. 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_DESTAT_L3[AddrLsb].

MSRC001_040[7...E] [L3 Machine Check Control Mask] (MCA::L3::MCA_CTL_MASK_L3)

Read-write. Reset: 0000_0000_0000_0000h.

Inhibit detection of an error source.

_ccd[7:0]_lthree0_inst7_n[56,48,40,32,24,16,8,0]_aliasMSR; MSRC001_0407
_ccd[7:0]_lthree0_inst8_n[57,49,41,33,25,17,9,1]_aliasMSR; MSRC001_0408
_ccd[7:0]_lthree0_inst9_n[5[8,0],[42,34,26,18,10,2]]_aliasMSR; MSRC001_0409
_ccd[7:0]_lthree0_inst10_n[5[9,1],[43,35,27,19,11,3]]_aliasMSR; MSRC001_040A
_ccd[7:0]_lthree0_inst11_n[60,52,44,36,28,20,12,4]_aliasMSR; MSRC001_040B
_ccd[7:0]_lthree0_inst12_n[61,53,45,37,29,21,13,5]_aliasMSR; MSRC001_040C

_ccd[7:0]_lthree0_inst13_n[62,54,46,38,30,22,14,6]_aliasMSR; MSRC001_040D
_ccd[7:0]_lthree0_inst14_n[63,55,47,39,31,23,15,7]_aliasMSR; MSRC001_040E

Bits	Description
63:8	Reserved.
7	**Hwa**. Read-write. Reset: 0. Init: BIOS,1. L3 Hardware Assertion.
6	**XiVictimQueue**. Read-write. Reset: 0. L3 Victim Queue Parity Error.
5	**SdpParity**. Read-write. Reset: 0. SDP Parity Error from XI.
4	**DataArray**. Read-write. Reset: 0. L3M Data ECC Error.
3	**MultiHitTag**. Read-write. Reset: 0. L3M Tag Multi-way-hit Error.
2	**Tag**. Read-write. Reset: 0. L3M Tag ECC Error.
1	**MultiHitShadowTag**. Read-write. Reset: 0. Shadow Tag Macro Multi-way-hit Error.
0	**ShadowTag**. Read-write. Reset: 0. Shadow Tag Macro ECC Error.

3.2.5.8 CS

MSR0000_044C...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.

_inst[CS[6,4,2,0]]_n[9,6,3,0]_aliasMSRLEGACY; MSR0000_044C
_inst[CS[7,5,3,1]]_n[10,7,4,1]_aliasMSRLEGACY; MSR0000_0450
_inst[CCIX[3:0]]_n[11,8,5,2]_aliasMSRLEGACY; MSR0000_0454
_inst[CS[6,4,2,0]]_n[9,6,3,0]_aliasMSR; MSRC000_2130
_inst[CS[7,5,3,1]]_n[10,7,4,1]_aliasMSR; MSRC000_2140
_inst[CCIX[3:0]]_n[11,8,5,2]_aliasMSR; MSRC000_2150

Bits	Description
63:14	Reserved.
13	**CNTR_UNFL**. Read-write. Reset: 0. Counter underflow error.
12	**CNTR_OVFL**. Read-write. Reset: 0. Counter overflow error.
11	**SDP_UNEXP_RETRY**. Read-write. Reset: 0. SDP read response had an unexpected RETRY error.
10	**SPF_ECC_ERR**. Read-write. Reset: 0. Probe Filter ECC Error: An ECC error occurred on a probe filter access.
9	**SPF_PRT_ERR**. Read-write. Reset: 0. Probe Filter Protocol Error: Indicates a Cache Coherence Issue.
8	**SDP_RSP_NO_MTCH**. Read-write. Reset: 0. SDP read response had no match in the CS queue.
7	**ATM_PAR_ERR**. Read-write. Reset: 0. Atomic Request Parity Error: Parity error on read of an atomic transaction.
6	**SDP_PAR_ERR**. Read-write. Reset: 0. Read Response Parity Error: Parity error on incoming read response data.
5	**FTI_PAR_ERR**. Read-write. Reset: 0. Request or Probe Parity Error: Parity error on incoming request or probe response data.
4	**FTI_RSP_NO_MTCH**. Read-write. Reset: 0. Unexpected Response: A response was received from the transport layer which does not match any request.
3	**FTI_ILL_RSP**. Read-write. Reset: 0. Illegal Response: An illegal response was received from the transport layer.
2	**FTI_SEC_VIOL**. Read-write. Reset: 0. Security Violation: A security violation was received from the transport layer.
1	**FTI_ADDR_VIOL**. Read-write. Reset: 0. Address Violation: An address violation was received from the transport layer.
0	**FTI_ILL_REQ**. Read-write. Reset: 0. Illegal Request: An illegal request was received from the transport layer.

MSR0000_044D...MSRC000_2151 [CS Machine Check Status] (MCA::CS::MCA_STATUS_CS)

Reset: Cold,0000_0000_0000_0000h.

Logs information associated with errors.

_inst[CS[6,4,2,0]]_n[9,6,3,0]_aliasMSRLEGACY; MSR0000_044D
_inst[CS[7,5,3,1]]_n[10,7,4,1]_aliasMSRLEGACY; MSR0000_0451
_inst[CCIX[3:0]]_n[11,8,5,2]_aliasMSRLEGACY; MSR0000_0455
_inst[CS[6,4,2,0]]_n[9,6,3,0]_aliasMSR; MSRC000_2131
_inst[CS[7,5,3,1]]_n[10,7,4,1]_aliasMSR; MSRC000_2141
_inst[CCIX[3:0]]_n[11,8,5,2]_aliasMSR; MSRC000_2151

Bits	Description
63	**Val**. 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	**Overflow**. 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	**UC**. 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	**En**. 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	**MiscV**. Reset: Cold,0. 1=Valid thresholding in MCA::CS::MCA_MISC0_CS. In certain modes, MISC registers are owned by platform firmware and will RAZ when read by non-SMM code. Therefore, it is possible for MiscV=1 and the MISC register to read as all zeros.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
58	**AddrV**. 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	**PCC**. 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	**ErrCoreIdVal**. 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	**TCC**. Reset: Cold,0. 1=Hardware context of the process thread to which the error was reported may have been corrupted. Continued operation of the thread may have unpredictable results. The thread 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	**RESERV4**. Reset: Cold,0. MCA_STATUS Register Reserved bit.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
53	**SyndV**. 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	Reserved.
51:47	**RESERV3**. Reset: Cold,00h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
46	**CECC**. 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	**UECC**. 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	**Deferred**. 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	**Poison**. 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.
42:41	**RESERV2**. Reset: Cold,0h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
40	**Scrub**. Reset: Cold,0. 1=The error was the result of a scrub operation.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
39:38	**RESERV1**. Reset: Cold,0h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
37:32	**ErrCoreId**. 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	**RESERV0**. Reset: Cold,000h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
21:16	**ErrorCodeExt**. 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	**ErrorCode**. 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 64: MCA_STATUS_CS

Error Type	ErrorCode Ext	UC	PCC	TCC	Deferred	Poison	AddrV
FTI_ILL_REQ	0x0	0	0	0	1	0	1
FTI_ADDR_VIOL	0x1	0	0	0	1	0	1
FTI_SEC_VIOL	0x2	0	0	0	1	0	1
FTI_ILL_RSP	0x3	1	1	1	0	0	0
FTI_RSP_NO_MTCH	0x4	1	1	1	0	0	0
FTI_PAR_ERR	0x5	0	0	0	1	0	1
SDP_PAR_ERR	0x6	0	0	0	1	0	1
ATM_PAR_ERR	0x7	0	0	0	1	0	1
SDP_RSP_NO_MTCH	0x8	1	1	1	0	0	0
SPF_PRT_ERR	0x9	1	1	1	0	0	0
SPF_ECC_ERR	0xa	0	0	0	0	0	1

SDP_UNEXP_RETRY	0xb	1	1	1	0	0	1
CNTR_OVFL	0xc	1	1	1	0	0	0
CNTR_UNFL	0xd	1	1	1	0	0	0

MSR0000_044E...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.

_inst[CS[6,4,2,0]]_n[9,6,3,0]_aliasMSRLEGACY; MSR0000_044E
_inst[CS[7,5,3,1]]_n[10,7,4,1]_aliasMSRLEGACY; MSR0000_0452
_inst[CCIX[3:0]]_n[11,8,5,2]_aliasMSRLEGACY; MSR0000_0456
_inst[CS[6,4,2,0]]_n[9,6,3,0]_aliasMSR; MSRC000_2132
_inst[CS[7,5,3,1]]_n[10,7,4,1]_aliasMSR; MSRC000_2142
_inst[CCIX[3:0]]_n[11,8,5,2]_aliasMSR; MSRC000_2152

Bits	Description
63:62	Reserved.
61:56	**LSB**. Read-write,Volatile. Reset: Cold,00h. Specifies the least significant valid bit of the address contained in MCA::CS::MCA_ADDR_CS[ErrorAddr]. For example, 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	**ErrorAddr**. 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 65: MCA_ADDR_CS

Error Type	Bits	Description
FTI_ILL_REQ	[51:2]	Address
FTI_ADDR_VIOL	[51:2]	Address
FTI_SEC_VIOL	[51:2]	Address
FTI_ILL_RSP	[55:0]	Reserved
FTI_RSP_NO_MTCH	[55:0]	Reserved
FTI_PAR_ERR	[51:2]	Address
SDP_PAR_ERR	[51:2]	Address
ATM_PAR_ERR	[51:2]	Address
SDP_RSP_NO_MTCH	[55:0]	Reserved
SPF_PRT_ERR	[55:0]	Reserved
SPF_ECC_ERR	[51:2]	Address
SDP_UNEXP_RETRY	[51:2]	Address
CNTR_OVFL	[55:0]	Reserved
CNTR_UNFL	[55:0]	Reserved

MSR0000_044F...MSRC000_2153 [CS Machine Check Miscellaneous 0] (MCA::CS::MCA_MISC0_CS)

Log miscellaneous information associated with errors.

_inst[CS[6,4,2,0]]_n[9,6,3,0]_aliasMSRLEGACY; MSR0000_044F
_inst[CS[7,5,3,1]]_n[10,7,4,1]_aliasMSRLEGACY; MSR0000_0453

_inst[CCIX[3:0]]_n[11,8,5,2]_aliasMSRLEGACY; MSR0000_0457	
_inst[CS[6,4,2,0]]_n[9,6,3,0]_aliasMSR; MSRC000_2133	
_inst[CS[7,5,3,1]]_n[10,7,4,1]_aliasMSR; MSRC000_2143	
_inst[CCIX[3:0]]_n[11,8,5,2]_aliasMSR; MSRC000_2153	
Bits	**Description**
63	**Valid**. Reset: 1. 1=A valid CntP field is present in this register.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read-only.
62	**CntP**. Reset: 1. 1=A valid threshold counter is present.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read-only.
61	**Locked**. 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	**IntP**. 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	**LvtOffset**. 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	**CntEn**. Reset: 0. 1=Count thresholding errors.
	AccessType: (Core::X86::Msr::HWCR[McStatusWrEn] | !MCA::CS::MCA_MISC0_CS[Locked]) ? Read-write : Read-only.
50:49	**ThresholdIntType**. 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::CS::MCA_MISC0_CS[Locked]) ? Read-write : Read-only.
48	**Ovrflw**. 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	**ErrCnt**. 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	**BlkPtr**. Read-write. Reset: 00h. 00h=Extended MISC MSR block is not valid. 01h=Extended MSR block is valid.
23:0	Reserved.

MSRC000_21[3...5]4 [CS Machine Check Configuration] (MCA::CS::MCA_CONFIG_CS)

Reset: 0000_0000_0000_0025h.	
Controls configuration of the associated machine check bank.	
_inst[CS[6,4,2,0]]_n[9,6,3,0]_aliasMSR; MSRC000_2134	
_inst[CS[7,5,3,1]]_n[10,7,4,1]_aliasMSR; MSRC000_2144	
_inst[CCIX[3:0]]_n[11,8,5,2]_aliasMSR; MSRC000_2154	
Bits	**Description**

63:39	Reserved.
38:37	**DeferredIntType**. 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	**LogDeferredInMcaStat**. 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	**McaXEnable**. 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	**DeferredIntTypeSupported**. 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	**DeferredErrorLoggingSupported**. 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	**McaX**. 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[3...5]5 [CS IP Identification] (MCA::CS::MCA_IPID_CS)

Reset: 0002_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.

_inst[CS[6,4,2,0]]_n[9,6,3,0]_aliasMSR; MSRC000_2135
_inst[CS[7,5,3,1]]_n[10,7,4,1]_aliasMSR; MSRC000_2145
_inst[CCIX[3:0]]_n[11,8,5,2]_aliasMSR; MSRC000_2155

Bits	Description
63:48	**McaType**. Read-only. Reset: 0002h. The McaType of the MCA bank within this IP.
47:44	Reserved.
43:32	**HardwareID**. Read-only. Reset: 02Eh. The Hardware ID of the IP associated with this MCA bank.
31:0	**InstanceId**. Read-write. Reset: 0000_0000h. The instance ID of this IP. This is initialized to a unique ID per instance of this register. Init: _instCCIX0_n2_aliasMSR: 0000_0800h Init: _instCCIX1_n5_aliasMSR: 0000_0900h Init: _instCCIX2_n8_aliasMSR: 0000_0A00h Init: _instCCIX3_n11_aliasMSR: 0000_0B00h Init: _instCS0_n0_aliasMSR: 0000_0000h Init: _instCS1_n1_aliasMSR: 0000_0100h Init: _instCS2_n3_aliasMSR: 0000_0200h Init: _instCS3_n4_aliasMSR: 0000_0300h Init: _instCS4_n6_aliasMSR: 0000_0400h Init: _instCS5_n7_aliasMSR: 0000_0500h Init: _instCS6_n9_aliasMSR: 0000_0600h Init: _instCS7_n10_aliasMSR: 0000_0700h

MSRC000_21[3...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

_inst[CS[6,4,2,0]]_n[9,6,3,0]_aliasMSR; MSRC000_2136

_inst[CS[7,5,3,1]]_n[10,7,4,1]_aliasMSR; MSRC000_2146

_inst[CCIX[3:0]]_n[11,8,5,2]_aliasMSR; MSRC000_2156

Bits	Description
63:48	Reserved.
47:32	**Syndrome**. 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	**ErrorPriority**. 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	**Length**. 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	**ErrorInformation**. Read-write,Volatile. Reset: Cold,0_0000h. Contains error-specific information about the location of the error. Decoding is available in Table 66 [MCA_SYND_CS].

Table 66: MCA_SYND_CS

Error Type	Bits	Description
FTI_ILL_REQ	[17:0]	
FTI_ADDR_VIOL	[17:0]	
FTI_SEC_VIOL	[17:0]	
FTI_ILL_RSP	[17:0]	
FTI_RSP_NO_MTCH	[17:0]	
FTI_PAR_ERR	[5:0]	
SDP_PAR_ERR	[5:0]	
ATM_PAR_ERR	[5:0]	
SDP_RSP_NO_MTCH	[6:0]	
SPF_PRT_ERR	[17:0]	
SPF_ECC_ERR	[17:0]	
SDP_UNEXP_RETRY	[5:0]	
CNTR_OVFL	[17:0]	
CNTR_UNFL	[17:0]	

MSRC000_21[3...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.

_inst[CS[6,4,2,0]]_n[9,6,3,0]_aliasMSR; MSRC000_2138

_inst[CS[7,5,3,1]]_n[10,7,4,1]_aliasMSR; MSRC000_2148

_inst[CCIX[3:0]]_n[11,8,5,2]_aliasMSR; MSRC000_2158

Bits	Description
63	**Val**. Read-write,Volatile. Reset: Cold,0. 1=A valid error has been detected (whether it is enabled or not).
62	**Overflow**. 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	**AddrV**. Read-write,Volatile. Reset: Cold,0. 1=MCA::CS::MCA_DEADDR_CS contains address information associated with the error.
57:54	Reserved.
53	**SyndV**. 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	**Deferred**. 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[3...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].

_inst[CS[6,4,2,0]]_n[9,6,3,0]_aliasMSR; MSRC000_2139
_inst[CS[7,5,3,1]]_n[10,7,4,1]_aliasMSR; MSRC000_2149
_inst[CCIX[3:0]]_n[11,8,5,2]_aliasMSR; MSRC000_2159

Bits	Description
63:62	Reserved.
61:56	**LSB**. Read-write,Volatile. Reset: Cold,00h. Specifies the least significant valid bit of the address contained in MCA::CS::MCA_DEADDR_CS[ErrorAddr]. For example, 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	**ErrorAddr**. 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[3...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.

_inst[CS[6,4,2,0]]_n[9,6,3,0]_aliasMSR; MSRC001_0413
_inst[CS[7,5,3,1]]_n[10,7,4,1]_aliasMSR; MSRC001_0414
_inst[CCIX[3:0]]_n[11,8,5,2]_aliasMSR; MSRC001_0415

Bits	Description
63:14	Reserved.
13	**CNTR_UNFL**. Read-write. Reset: 0. Counter underflow error.
12	**CNTR_OVFL**. Read-write. Reset: 0. Counter overflow error.
11	**SDP_UNEXP_RETRY**. Read-write. Reset: 0. SDP read response had an unexpected RETRY error.
10	**SPF_ECC_ERR**. Read-write. Reset: 0. Probe Filter ECC Error: An ECC error occurred on a probe filter access.
9	**SPF_PRT_ERR**. Read-write. Reset: 0. Probe Filter Protocol Error: Indicates a Cache Coherence Issue.
8	**SDP_RSP_NO_MTCH**. Read-write. Reset: 0. SDP read response had no match in the CS queue.
7	**ATM_PAR_ERR**. Read-write. Reset: 0. Atomic Request Parity Error: Parity error on read of an atomic transaction.

6	**SDP_PAR_ERR**. Read-write. Reset: 0. Read Response Parity Error: Parity error on incoming read response data.
5	**FTI_PAR_ERR**. Read-write. Reset: 0. Request or Probe Parity Error: Parity error on incoming request or probe response data.
4	**FTI_RSP_NO_MTCH**. Read-write. Reset: 0. Unexpected Response: A response was received from the transport layer which does not match any request.
3	**FTI_ILL_RSP**. Read-write. Reset: 0. Illegal Response: An illegal response was received from the transport layer.
2	**FTI_SEC_VIOL**. Read-write. Reset: 0. Security Violation: A security violation was received from the transport layer.
1	**FTI_ADDR_VIOL**. Read-write. Reset: 0. Address Violation: An address violation was received from the transport layer.
0	**FTI_ILL_REQ**. Read-write. Reset: 0. Illegal Request: An illegal request was received from the transport layer.

3.2.5.9 PIE

MSR0000_046C...MSRC000_21B0 [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.

_instPIE0_n0_aliasMSRLEGACY; MSR0000_046C

_instPIE0_n0_aliasMSR; MSRC000_21B0

Bits	Description
63:5	Reserved.
4	**DEF**. Read-write. Reset: 0. A deferred error was detected in the DF.
3	**FTI_DAT_STAT**. Read-write. Reset: 0. Poison data consumption: Poison data was written to an internal PIE register.
2	**GMI**. Read-write. Reset: 0. Link Error: An error occurred on a GMI or xGMI link.
1	**CSW**. Read-write. Reset: 0. Register security violation: A security violation was detected on an access to an internal PIE register.
0	**HW_ASSERT**. Read-write. Reset: 0. Hardware Assert: A hardware assert was detected.

MSR0000_046D...MSRC000_21B1 [PIE Machine Check Status] (MCA::PIE::MCA_STATUS_PIE)

Reset: Cold,0000_0000_0000_0000h.

Logs information associated with errors.

_instPIE0_n0_aliasMSRLEGACY; MSR0000_046D

_instPIE0_n0_aliasMSR; MSRC000_21B1

Bits	Description
63	**Val**. 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	**Overflow**. 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	**UC**. 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	**En**. 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	**MiscV**. Reset: Cold,0. 1=Valid thresholding in MCA::PIE::MCA_MISC0_PIE. In certain modes, MISC registers

	are owned by platform firmware and will RAZ when read by non-SMM code. Therefore, it is possible for MiscV=1 and the MISC register to read as all zeros.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
58	**AddrV**. 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	**PCC**. 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	**ErrCoreIdVal**. 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	**TCC**. Reset: Cold,0. 1=Hardware context of the process thread to which the error was reported may have been corrupted. Continued operation of the thread may have unpredictable results. The thread 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	**RESERV4**. Reset: Cold,0. MCA_STATUS Register Reserved bit.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
53	**SyndV**. 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	Reserved.
51:47	**RESERV3**. Reset: Cold,00h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
46	**CECC**. 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	**UECC**. 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	**Deferred**. 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	**Poison**. 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.
42:41	**RESERV2**. Reset: Cold,0h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
40	**Scrub**. Reset: Cold,0. 1=The error was the result of a scrub operation.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
39:38	**RESERV1**. Reset: Cold,0h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
37:32	**ErrCoreId**. 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	**RESERV0**. Reset: Cold,000h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
21:16	**ErrorCodeExt**. 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	**ErrorCode**. 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 67: MCA_STATUS_PIE

Error Type	ErrorCode Ext	UC	PCC	TCC	Deferred	Poison	AddrV
HW_ASSERT	0x0	1	1	1	0	0	0
CSW	0x1	0	0	0	1	0	0
GMI	0x2	0/1	0/1	0/1	0	0	0
FTI_DAT_STAT	0x3	1	1	1	0	0	0
DEF	0x4	0	0	0	1	0	0

MSR0000_046E...MSRC000_21B2 [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.

_instPIE0_n0_aliasMSRLEGACY; MSR0000_046E

_instPIE0_n0_aliasMSR; MSRC000_21B2

Bits	Description
63:62	Reserved.
61:56	**LSB**. Read-only. Reset: Cold,00h. Specifies the least significant valid bit of the address contained in MCA::PIE::MCA_ADDR_PIE[ErrorAddr]. For example, 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	**ErrorAddr**. 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 68: MCA_ADDR_PIE

Error Type	Bits	Description
HW_ASSERT	[55:0]	Reserved
CSW	[55:0]	Reserved
GMI	[55:0]	Reserved
FTI_DAT_STAT	[55:0]	Reserved
DEF	[55:0]	Reserved

MSR0000_046F...MSRC000_21B3 [PIE Machine Check Miscellaneous 0] (MCA::PIE::MCA_MISC0_PIE)

Log miscellaneous information associated with errors.

_instPIE0_n0_aliasMSRLEGACY; MSR0000_046F

_instPIE0_n0_aliasMSR; MSRC000_21B3

Bits	Description
63	**Valid**. Reset: 1. 1=A valid CntP field is present in this register.

	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read-only.
62	**CntP**. Reset: 1. 1=A valid threshold counter is present.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read-only.
61	**Locked**. 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	**IntP**. 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	**LvtOffset**. 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	**CntEn**. Reset: 0. 1=Count thresholding errors.
	AccessType: (Core::X86::Msr::HWCR[McStatusWrEn] | !MCA::PIE::MCA_MISC0_PIE[Locked]) ? Read-write : Read-only.
50:49	**ThresholdIntType**. 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	**Ovrflw**. 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	**ErrCnt**. 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	**BlkPtr**. Read-write. Reset: 00h. 00h=Extended MISC MSR block is not valid. 01h=Extended MSR block is valid.
23:0	Reserved.

MSRC000_21B4 [PIE Machine Check Configuration] (MCA::PIE::MCA_CONFIG_PIE)

Reset: 0000_0002_0000_0025h.

Controls configuration of the associated machine check bank.

_instPIE0_n0_aliasMSR; MSRC000_21B4

Bits	Description
63:39	Reserved.
38:37	**DeferredIntType**. 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	**LogDeferredInMcaStat**. 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	**McaXEnable**. 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	**DeferredIntTypeSupported**. 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	**DeferredErrorLoggingSupported**. 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	**McaX**. 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_21B5 [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.

_instPIE0_n0_aliasMSR; MSRC000_21B5

Bits	Description
63:48	**McaType**. Read-only. Reset: 0001h. The McaType of the MCA bank within this IP.
47:44	Reserved.
43:32	**HardwareID**. Read-only. Reset: 02Eh. The Hardware ID of the IP associated with this MCA bank.
31:0	**InstanceId**. Read-write. Reset: 0000_0000h. Init: 0000_0000h. The instance ID of this IP. This is initialized to a unique ID per instance of this register.

MSRC000_21B6 [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

_instPIE0_n0_aliasMSR; MSRC000_21B6

Bits	Description
63:33	Reserved.
32	**Syndrome**. 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	**ErrorPriority**. 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	**Length**. 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	**ErrorInformation**. Read-write,Volatile. Reset: Cold,0_0000h. Contains error-specific information about the location of the error. Decoding is available in Table 69 [MCA_SYND_PIE].

Table 69: MCA_SYND_PIE

Error Type	Bits	Description
HW_ASSERT	[17:0]	Reserved
CSW	[17:0]	
GMI	[17:0]	
FTI_DAT_STAT	[3:0]	
DEF	[17:0]	Reserved

MSRC000_21B8 [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.

_instPIE0_n0_aliasMSR; MSRC000_21B8

Bits	Description
63	**Val**. Read-write,Volatile. Reset: Cold,0. 1=A valid error has been detected (whether it is enabled or not).
62	**Overflow**. 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	**AddrV**. Read-write,Volatile. Reset: Cold,0. 1=MCA::PIE::MCA_DEADDR_PIE contains address information associated with the error.
57:54	Reserved.
53	**SyndV**. 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	**Deferred**. 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_21B9 [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].

_instPIE0_n0_aliasMSR; MSRC000_21B9

Bits	Description
63:62	Reserved.
61:56	**LSB**. Read-write,Volatile. Reset: Cold,00h. Specifies the least significant valid bit of the address contained in MCA::PIE::MCA_DEADDR_PIE[ErrorAddr]. For example, 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	**ErrorAddr**. 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_041B [PIE Machine Check Control Mask] (MCA::PIE::MCA_CTL_MASK_PIE)

Read-write. Reset: 0000_0000_0000_0000h.	
Inhibit detection of an error source.	
_instPIE0_n0_aliasMSR; MSRC001_041B	
Bits	**Description**
63:5	Reserved.
4	**DEF**. Read-write. Reset: 0. A deferred error was detected in the DF.
3	**FTI_DAT_STAT**. Read-write. Reset: 0. Poison data consumption: Poison data was written to an internal PIE register.
2	**GMI**. Read-write. Reset: 0. Link Error: An error occurred on a GMI or xGMI link.
1	**CSW**. Read-write. Reset: 0. Register security violation: A security violation was detected on an access to an internal PIE register.
0	**HW_ASSERT**. Read-write. Reset: 0. Hardware Assert: A hardware assert was detected.

3.2.5.10 UMC

MSR0000_0444...MSRC000_2120 [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.	
_ch0_inst[UMCWPHY[3:0]UMC]_n[6,4,2,0]_umc0_aliasMSRLEGACY; MSR0000_0444	
_ch0_inst[UMCWPHY[3:0]UMC]_n[7,5,3,1]_umc1_aliasMSRLEGACY; MSR0000_0448	
_ch0_inst[UMCWPHY[3:0]UMC]_n[6,4,2,0]_umc0_aliasMSR; MSRC000_2110	
_ch0_inst[UMCWPHY[3:0]UMC]_n[7,5,3,1]_umc1_aliasMSR; MSRC000_2120	
Bits	**Description**
63:8	Reserved.
7	**AesSramEccErr**. Read-write. Reset: 0. AES SRAM ECC error. An ECC error occured on a AES SRAM in the processor.
6	**DcqSramEccErr**. Read-write. Reset: 0. DCQ SRAM ECC error. An ECC error occured on a DCQ SRAM in the processor.
5	**WriteDataCrcErr**. Read-write. Reset: 0. Write data CRC error. A write data CRC error occurred on the DRAM data bus.
4	**AddressCommandParityErr**. Read-write. Reset: 0. Address/Command parity error. A parity error occurred on the DRAM address/command bus.
3	**ApbErr**. Read-write. Reset: 0. Advanced peripheral bus error. An error occurred on the advanced peripheral bus.
2	**SdpParityErr**. Read-write. Reset: 0. SDP parity error. A parity error was detected on write data from the data fabric in the processor.
1	**WriteDataPoisonErr**. Read-write. Reset: 0. Data poison error. The system tried to write poison data to DRAM and either DRAM does not support ECC or UMC_CH.EccCtrl.WrEccEn is cleared.
0	**DramEccErr**. Read-write. Reset: 0. DRAM ECC error. An ECC error occurred on a DRAM read.

MSR0000_0445...MSRC000_2121 [UMC Machine Check Status] (MCA::UMC::MCA_STATUS_UMC)

Reset: Cold,0000_0000_0000_0000h.
Logs information associated with errors.
_ch0_inst[UMCWPHY[3:0]UMC]_n[6,4,2,0]_umc0_aliasMSRLEGACY; MSR0000_0445
_ch0_inst[UMCWPHY[3:0]UMC]_n[7,5,3,1]_umc1_aliasMSRLEGACY; MSR0000_0449

_ch0_inst[UMCWPHY[3:0]UMC]_n[6,4,2,0]_umc0_aliasMSR; MSRC000_2111	
_ch0_inst[UMCWPHY[3:0]UMC]_n[7,5,3,1]_umc1_aliasMSR; MSRC000_2121	
Bits	**Description**
63	**Val**. 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	**Overflow**. 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	**UC**. 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	**En**. 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	**MiscV**. Reset: Cold,0. 1=Valid thresholding in MCA::UMC::MCA_MISC0_UMC. In certain modes, MISC registers are owned by platform firmware and will RAZ when read by non-SMM code. Therefore, it is possible for MiscV=1 and the MISC register to read as all zeros.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
58	**AddrV**. 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	**PCC**. 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	**ErrCoreIdVal**. 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	**TCC**. Reset: Cold,0. 1=Hardware context of the process thread to which the error was reported may have been corrupted. Continued operation of the thread may have unpredictable results. The thread 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	**RESERV4**. Reset: Cold,0. MCA_STATUS Register Reserved bit.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
53	**SyndV**. 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	Reserved.
51:47	**RESERV3**. Reset: Cold,00h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
46	**CECC**. 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	**UECC**. 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	**Deferred**. 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	**Poison**. 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.
42:41	**RESERV2**. Reset: Cold,0h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
40	**Scrub**. Reset: Cold,0. 1=The error was the result of a scrub operation.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
39:38	**RESERV1**. Reset: Cold,0h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
37:32	**ErrCoreId**. 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	**RESERV0**. Reset: Cold,000h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
21:16	**ErrorCodeExt**. 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	**ErrorCode**. 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 70: MCA_STATUS_UMC

Error Type	ErrorCode Ext	UC	PCC	TCC	Deferred	Poison	AddrV
DramEccErr	0x0	0/1	0/1	0/1	0/1	0	1
WriteDataPoisonErr	0x1	1	1	1	0	0	0
SdpParityErr	0x2	1	1	1	0	0	0
ApbErr	0x3	1	1	1	0	0	1
AddressCommandParityErr	0x4	0/1	0/1	0/1	0	0	0/1
WriteDataCrcErr	0x5	0/1	0/1	0/1	0	0	0
DcqSramEccErr	0x6	0/1	0/1	0/1	0	0	0
AesSramEccErr	0x7	0/1	0/1	0/1	0	0	0

MSR0000_0446...MSRC000_2122 [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.

_ch0_inst[UMCWPHY[3:0]UMC]_n[6,4,2,0]_umc0_aliasMSRLEGACY; MSR0000_0446

_ch0_inst[UMCWPHY[3:0]UMC]_n[7,5,3,1]_umc1_aliasMSRLEGACY; MSR0000_044A

_ch0_inst[UMCWPHY[3:0]UMC]_n[6,4,2,0]_umc0_aliasMSR; MSRC000_2112

_ch0_inst[UMCWPHY[3:0]UMC]_n[7,5,3,1]_umc1_aliasMSR; MSRC000_2122

Bits	Description

63:62	Reserved.
61:56	**LSB**. Read-write,Volatile. Reset: Cold,00h. Specifies the least significant valid bit of the address contained in MCA::UMC::MCA_ADDR_UMC[ErrorAddr]. For example, 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	**ErrorAddr**. 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 71: MCA_ADDR_UMC

Error Type	Bits	Description
DramEccErr	[55:39]	Reserved
	[39:4]	Reserved
WriteDataPoisonErr	[55:0]	Reserved
SdpParityErr	[55:0]	Reserved
ApbErr	[55:30]	Reserved
	[29:0]	Reserved
AddressCommandParityErr	[55:38]	Reserved
	[37:36]	Reserved
	[35:32]	Chip Select.
	[31:0]	Reserved
WriteDataCrcErr	[55:0]	Reserved
DcqSramEccErr	[55:0]	Reserved
AesSramEccErr	[55:0]	Reserved

MSR0000_0447...MSRC000_2123 [UMC Machine Check Miscellaneous 0] (MCA::UMC::MCA_MISC0_UMC)

Log miscellaneous information associated with errors.

_ch0_inst[UMCWPHY[3:0]UMC]_n[6,4,2,0]_umc0_aliasMSRLEGACY; MSR0000_0447
_ch0_inst[UMCWPHY[3:0]UMC]_n[7,5,3,1]_umc1_aliasMSRLEGACY; MSR0000_044B
_ch0_inst[UMCWPHY[3:0]UMC]_n[6,4,2,0]_umc0_aliasMSR; MSRC000_2113
_ch0_inst[UMCWPHY[3:0]UMC]_n[7,5,3,1]_umc1_aliasMSR; MSRC000_2123

Bits	Description
63	**Valid**. Reset: 1. 1=A valid CntP field is present in this register.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read-only.
62	**CntP**. Reset: 1. 1=A valid threshold counter is present.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read-only.
61	**Locked**. 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	**IntP**. 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	**LvtOffset**. 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	**CntEn**. Reset: 0. 1=Count thresholding errors.
	AccessType: (Core::X86::Msr::HWCR[McStatusWrEn] | !MCA::UMC::MCA_MISC0_UMC[Locked]) ? Read-write : Read-only.
50:49	**ThresholdIntType**. 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	**Ovrflw**. 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::UMC::MCA_MISC0_UMC[Locked]) ? Read-write : Read-only.
47:44	Reserved.
43:32	**ErrCnt**. 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::UMC::MCA_MISC0_UMC[Locked]) ? Read-write : Read-only.
31:24	**BlkPtr**. Read-write. Reset: 01h. 00h=Extended MISC MSR block is not valid. 01h=Extended MSR block is valid.
23:0	Reserved.

MSRC000_21[1...2]4 [UMC Machine Check Configuration] (MCA::UMC::MCA_CONFIG_UMC)

Reset: 0000_0002_0000_0025h.

Controls configuration of the associated machine check bank.

_ch0_inst[UMCWPHY[3:0]UMC]_n[6,4,2,0]_umc0_aliasMSR; MSRC000_2114

_ch0_inst[UMCWPHY[3:0]UMC]_n[7,5,3,1]_umc1_aliasMSR; MSRC000_2124

Bits	Description
63:39	Reserved.
38:37	**DeferredIntType**. 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	**LogDeferredInMcaStat**. 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	**McaXEnable**. 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	**DeferredIntTypeSupported**. 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	**DeferredErrorLoggingSupported**. 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	**McaX**. 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_21[1...2]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.

_ch0_inst[UMCWPHY[3:0]UMC]_n[6,4,2,0]_umc0_aliasMSR; MSRC000_2115

_ch0_inst[UMCWPHY[3:0]UMC]_n[7,5,3,1]_umc1_aliasMSR; MSRC000_2125

Bits	Description
63:48	**McaType**. Read-only. Reset: 0000h. The McaType of the MCA bank within this IP.
47:44	Reserved.
43:32	**HardwareID**. Read-only. Reset: 096h. The Hardware ID of the IP associated with this MCA bank.
31:0	**InstanceId**. Read-write. Reset: 0000_0000h. The instance ID of this IP. This is initialized to a unique ID per instance of this register. Init: _ch0_instUMCWPHY0UMC_n0_umc0_aliasMSR: 0005_0F00h Init: _ch0_instUMCWPHY0UMC_n1_umc1_aliasMSR: 0015_0F00h Init: _ch0_instUMCWPHY1UMC_n2_umc0_aliasMSR: 0025_0F00h Init: _ch0_instUMCWPHY1UMC_n3_umc1_aliasMSR: 0035_0F00h Init: _ch0_instUMCWPHY2UMC_n4_umc0_aliasMSR: 0045_0F00h Init: _ch0_instUMCWPHY2UMC_n5_umc1_aliasMSR: 0055_0F00h Init: _ch0_instUMCWPHY3UMC_n6_umc0_aliasMSR: 0065_0F00h Init: _ch0_instUMCWPHY3UMC_n7_umc1_aliasMSR: 0075_0F00h

MSRC000_21[1...2]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

_ch0_inst[UMCWPHY[3:0]UMC]_n[6,4,2,0]_umc0_aliasMSR; MSRC000_2116

_ch0_inst[UMCWPHY[3:0]UMC]_n[7,5,3,1]_umc1_aliasMSR; MSRC000_2126

Bits	Description
63:32	**Syndrome**. Read-write,Volatile. Reset: Cold,0000_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	**ErrorPriority**. 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	**Length**. 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	**ErrorInformation**. Read-write,Volatile. Reset: Cold,0_0000h. Contains error-specific information about the location of the error. Decoding is available in Table 72 [MCA_SYND_UMC].

Table 72: MCA_SYND_UMC

Error Type	Bits	Description

DramEccErr	[17:16]	Reserved
	[15]	Software-Managed Bad Symbol ID Error
	[14]	Reserved
	[13:8]	Symbol. Only contains valid information on 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
DcqSramEccErr	[17:14]	Reserved
	[13:0]	Reserved
AesSramEccErr	[17]	Reserved
	[16:8]	Reserved
	[7:4]	Reserved
	[3:2]	Reserved
	[1:0]	Reserved

MSRC000_21[1...2]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.

_ch0_inst[UMCWPHY[3:0]UMC]_n[6,4,2,0]_umc0_aliasMSR; MSRC000_2118

_ch0_inst[UMCWPHY[3:0]UMC]_n[7,5,3,1]_umc1_aliasMSR; MSRC000_2128

Bits	Description
63	**Val**. Read-write,Volatile. Reset: Cold,0. 1=A valid error has been detected (whether it is enabled or not).
62	**Overflow**. 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	**AddrV**. Read-write,Volatile. Reset: Cold,0. 1=MCA::UMC::MCA_DEADDR_UMC contains address information associated with the error.
57:54	Reserved.
53	**SyndV**. 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	**Deferred**. 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[1...2]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].	
_ch0_inst[UMCWPHY[3:0]UMC]_n[6,4,2,0]_umc0_aliasMSR; MSRC000_2119	
_ch0_inst[UMCWPHY[3:0]UMC]_n[7,5,3,1]_umc1_aliasMSR; MSRC000_2129	
Bits	**Description**
63:62	Reserved.
61:56	**LSB**. Read-write,Volatile. Reset: Cold,00h. Specifies the least significant valid bit of the address contained in MCA::UMC::MCA_DEADDR_UMC[ErrorAddr]. For example, 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	**ErrorAddr**. 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_21[1...2]A [UMC Machine Check Miscellaneous 1] (MCA::UMC::MCA_MISC1_UMC)

Log miscellaneous information associated with errors, as defined by each error type.	
_ch0_inst[UMCWPHY[3:0]UMC]_n[6,4,2,0]_umc0_aliasMSR; MSRC000_211A	
_ch0_inst[UMCWPHY[3:0]UMC]_n[7,5,3,1]_umc1_aliasMSR; MSRC000_212A	
Bits	**Description**
63	**Valid**. Reset: 1. 1=A valid CntP field is present in this register.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read-only.
62	**CntP**. Reset: 1. 1=A valid threshold counter is present.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read-only.
61	**Locked**. 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	**IntP**. 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_MISC1_UMC[Locked]) ? Read-write : Read-only.
59:52	Reserved.
51	**CntEn**. Reset: 0. 1=Count thresholding errors.
	AccessType: (Core::X86::Msr::HWCR[McStatusWrEn] | !MCA::UMC::MCA_MISC1_UMC[Locked]) ? Read-write : Read-only.
50:49	**ThresholdIntType**. 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.
	AccessType: (Core::X86::Msr::HWCR[McStatusWrEn] | !MCA::UMC::MCA_MISC1_UMC[Locked]) ? Read-write : Read-only.
48	**Ovrflw**. 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.
	AccessType: (Core::X86::Msr::HWCR[McStatusWrEn] | !MCA::UMC::MCA_MISC1_UMC[Locked]) ? Read-write : Read-only.
47:44	Reserved.
43:32	**ErrCnt**. 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::UMC::MCA_MISC1_UMC[Locked]) ? Read-

	write : Read-only.
31:24	**BlkPtr**. Read-write. Reset: 01h. 00h=Extended MISC MSR block is not valid. 01h=Extended MSR block is valid.
23:0	Reserved.

MSRC001_041[1...2] [UMC Machine Check Control Mask] (MCA::UMC::MCA_CTL_MASK_UMC)

Read-write. Reset: 0000_0000_0000_0000h.

Inhibit detection of an error source.

_ch0_inst[UMCWPHY[3:0]UMC]_n[6,4,2,0]_umc0_aliasMSR; MSRC001_0411

_ch0_inst[UMCWPHY[3:0]UMC]_n[7,5,3,1]_umc1_aliasMSR; MSRC001_0412

Bits	Description
63:8	Reserved.
7	**AesSramEccErr**. Read-write. Reset: 0. AES SRAM ECC error. An ECC error occured on a AES SRAM in the processor.
6	**DcqSramEccErr**. Read-write. Reset: 0. DCQ SRAM ECC error. An ECC error occured on a DCQ SRAM in the processor.
5	**WriteDataCrcErr**. Read-write. Reset: 0. Write data CRC error. A write data CRC error occurred on the DRAM data bus.
4	**AddressCommandParityErr**. Read-write. Reset: 0. Address/Command parity error. A parity error occurred on the DRAM address/command bus.
3	**ApbErr**. Read-write. Reset: 0. Advanced peripheral bus error. An error occurred on the advanced peripheral bus.
2	**SdpParityErr**. Read-write. Reset: 0. SDP parity error. A parity error was detected on write data from the data fabric in the processor.
1	**WriteDataPoisonErr**. Read-write. Reset: 0. Data poison error. The system tried to write poison data to DRAM and either DRAM does not support ECC or UMC_CH.EccCtrl.WrEccEn is cleared.
0	**DramEccErr**. Read-write. Reset: 0. DRAM ECC error. An ECC error occurred on a DRAM read.

3.2.5.11 PB

MSR0000_0440...MSRC000_21A0 [PB Machine Check Control] (MCA::PB::MCA_CTL_PB)

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::PB::MCA_CTL_PB register must be enabled by the corresponding enable bit in Core::X86::Msr::MCG_CTL. Does not affect error detection, correction, or logging.

_ccd[7:0]_instPBCCD_n[8:2,0]_aliasMSRLEGACY; MSR0000_0440

_instPB_n1_aliasMSRLEGACY; MSR0000_0468

_ccd[7:0]_instPBCCD_n[8:2,0]_aliasMSR; MSRC000_2100

_instPB_n1_aliasMSR; MSRC000_21A0

Bits	Description
63:1	Reserved.
0	**EccError**. Read-write. Reset: 0. An ECC error in the Parameter Block RAM array.

MSR0000_0441...MSRC000_21A1 [PB Machine Check Status] (MCA::PB::MCA_STATUS_PB)

Reset: Cold,0000_0000_0000_0000h.

Logs information associated with errors.

_ccd[7:0]_instPBCCD_n[8:2,0]_aliasMSRLEGACY; MSR0000_0441

_instPB_n1_aliasMSRLEGACY; MSR0000_0469

_ccd[7:0]_instPBCCD_n[8:2,0]_aliasMSR; MSRC000_2101

_instPB_n1_aliasMSR; MSRC000_21A1

Bits	Description
63	**Val**. 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	**Overflow**. 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	**UC**. 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	**En**. Reset: Cold,0. 1=MCA error reporting is enabled for this error, as indicated by the corresponding bit in MCA::PB::MCA_CTL_PB. This bit is a copy of bit in MCA::PB::MCA_CTL_PB for this error.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
59	**MiscV**. Reset: Cold,0. 1=Valid thresholding in MCA::PB::MCA_MISC0_PB. In certain modes, MISC registers are owned by platform firmware and will RAZ when read by non-SMM code. Therefore, it is possible for MiscV=1 and the MISC register to read as all zeros.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
58	**AddrV**. Reset: Cold,0. 1=MCA::PB::MCA_ADDR_PB contains address information associated with the error.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
57	**PCC**. 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	**ErrCoreIdVal**. 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	**TCC**. Reset: Cold,0. 1=Hardware context of the process thread to which the error was reported may have been corrupted. Continued operation of the thread may have unpredictable results. The thread must be terminated. Only meaningful when MCA::PB::MCA_STATUS_PB[PCC]=0.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
54	**RESERV4**. Reset: Cold,0. MCA_STATUS Register Reserved bit.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
53	**SyndV**. Reset: Cold,0. 1=This error logged information in MCA::PB::MCA_SYND_PB. If MCA::PB::MCA_SYND_PB[ErrorPriority] is the same as the priority of the error in MCA::PB::MCA_STATUS_PB, then the information in MCA::PB::MCA_SYND_PB is associated with the error in MCA::PB::MCA_STATUS_PB.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
52	Reserved.
51:47	**RESERV3**. Reset: Cold,00h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
46	**CECC**. 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	**UECC**. 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	**Deferred**. 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	**Poison**. 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.
42:41	**RESERV2**. Reset: Cold,0h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
40	**Scrub**. Reset: Cold,0. 1=The error was the result of a scrub operation.

	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
39:38	**RESERV1**. Reset: Cold,0h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
37:32	**ErrCoreId**. 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	**RESERV0**. Reset: Cold,000h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
21:16	**ErrorCodeExt**. 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::PB::MCA_CTL_PB 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	**ErrorCode**. 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 73: MCA_STATUS_PB

Error Type	ErrorCode Ext	UC	PCC	TCC	Deferred	Poison	AddrV
EccError	0x0	0/1	0/1	0/1	0	0	0

MSR0000_0442...MSRC000_21A2 (MCA::PB::MCA_ADDR_PB)

Read-only. Reset: Cold,0000_0000_0000_0000h.

_ccd[7:0]_instPBCCD_n[8:2,0]_aliasMSRLEGACY; MSR0000_0442

_instPB_n1_aliasMSRLEGACY; MSR0000_046A

_ccd[7:0]_instPBCCD_n[8:2,0]_aliasMSR; MSRC000_2102

_instPB_n1_aliasMSR; MSRC000_21A2

Bits	Description
63:62	Reserved.
61:56	**LSB**. Read-only. Reset: Cold,00h. Specifies the least significant valid bit of the address contained in MCA::PB::MCA_ADDR_PB[ErrorAddr]. For example, a value of 0 indicates that MCA::PB::MCA_ADDR_PB[55:0] contains a valid byte address. A value of 6 indicates that MCA::PB::MCA_ADDR_PB[55:6] contains a valid cache line address and that MCA::PB::MCA_ADDR_PB[5:0] are not part of the address and should be ignored by error handling software. A value of 12 indicates that MCA::PB::MCA_ADDR_PB[55:12] contain a valid 4KB memory page and that MCA::PB::MCA_ADDR_PB[11:0] should be ignored by error handling software.
55:0	**ErrorAddr**. Read-only. Reset: Cold,00_0000_0000_0000h. Contains the address, if any, associated with the error logged in MCA::PB::MCA_STATUS_PB.

Table 74: MCA_ADDR_PB

Error Type	Bits	Description
EccError	[55:0]	Reserved

MSR0000_0443...MSRC000_21A3 [PB Machine Check Miscellaneous 0] (MCA::PB::MCA_MISC0_PB)

Log miscellaneous information associated with errors.

_ccd[7:0]_instPBCCD_n[8:2,0]_aliasMSRLEGACY; MSR0000_0443

_instPB_n1_aliasMSRLEGACY; MSR0000_046B

_ccd[7:0]_instPBCCD_n[8:2,0]_aliasMSR; MSRC000_2103

_instPB_n1_aliasMSR; MSRC000_21A3

Bits	Description
63	**Valid**. Reset: 1. 1=A valid CntP field is present in this register.

	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read-only.
62	**CntP**. Reset: 1. 1=A valid threshold counter is present.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read-only.
61	**Locked**. 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	**IntP**. 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::PB::MCA_MISC0_PB[Locked]) ? Read-write : Read-only.
59:56	Reserved.
55:52	**LvtOffset**. 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::PB::MCA_MISC0_PB[Locked]) ? Read-write : Read-only.
51	**CntEn**. Reset: 0. 1=Count thresholding errors.
	AccessType: (Core::X86::Msr::HWCR[McStatusWrEn] | !MCA::PB::MCA_MISC0_PB[Locked]) ? Read-write : Read-only.
50:49	**ThresholdIntType**. 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::PB::MCA_MISC0_PB[Locked]) ? Read-write : Read-only.
48	**Ovrflw**. 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::PB::MCA_MISC0_PB[Locked]) ? Read-write : Read-only.
47:44	Reserved.
43:32	**ErrCnt**. 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::PB::MCA_MISC0_PB[Locked]) ? Read-write : Read-only.
31:24	**BlkPtr**. Read-write. Reset: 00h. 00h=Extended MISC MSR block is not valid. 01h=Extended MSR block is valid.
23:0	Reserved.

MSRC000_21[0...A]4 [PB Machine Check Configuration] (MCA::PB::MCA_CONFIG_PB)

Reset: 0000_0000_0000_0021h.	
Controls configuration of the associated machine check bank.	
_ccd[7:0]_instPBCCD_n[8:2,0]_aliasMSR; MSRC000_2104	
_instPB_n1_aliasMSR; MSRC000_21A4	
Bits	**Description**
63:39	Reserved.
38:37	**DeferredIntType**. 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	**McaXEnable**. 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	**DeferredIntTypeSupported**. Read-only. Reset: 1. 1=MCA::PB::MCA_CONFIG_PB[DeferredIntType] controls the type of interrupt generated on a deferred error. Deferred errors are supported in this bank only if MCA::PB::MCA_CONFIG_PB[DeferredErrorLoggingSupported]=1.
4:3	Reserved.
2	**DeferredErrorLoggingSupported**. 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	**McaX**. Read-only. Reset: 1. 1=This bank provides Machine Check Architecture Extensions. Up to 4 additional MISC registers (MISC1-MISC4) are supported. MCA::PB::MCA_MISC0_PB[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::PB::MCA_STATUS_PB[TCC] is present.

MSRC000_21[0...A]5 [PB IP Identification] (MCA::PB::MCA_IPID_PB)

Reset: 0000_0005_0000_0000h.

The MCA::PB::MCA_IPID_PB register is used by software to determine what IP type and revision is associated with the MCA bank.

_ccd[7:0]_instPBCCD_n[8:2,0]_aliasMSR; MSRC000_2105

_instPB_n1_aliasMSR; MSRC000_21A5

Bits	Description
63:48	**McaType**. Read-only. Reset: 0000h. The McaType of the MCA bank within this IP.
47:44	Reserved.
43:32	**HardwareID**. Read-only. Reset: 005h. The Hardware ID of the IP associated with this MCA bank.
31:0	**InstanceId**. Read-write. Reset: 0000_0000h. The instance ID of this IP. This is initialized to a unique ID per instance of this register. Init: _ccd0_instPBCCD_n0_aliasMSR: 3008_2900h Init: _ccd1_instPBCCD_n2_aliasMSR: 3208_2900h Init: _ccd2_instPBCCD_n3_aliasMSR: 3408_2900h Init: _ccd3_instPBCCD_n4_aliasMSR: 3608_2900h Init: _ccd4_instPBCCD_n5_aliasMSR: 3808_2900h Init: _ccd5_instPBCCD_n6_aliasMSR: 3A08_2900h Init: _ccd6_instPBCCD_n7_aliasMSR: 3C08_2900h Init: _ccd7_instPBCCD_n8_aliasMSR: 3E08_2900h Init: _instPB_n1_aliasMSR: 0005_E100h

MSRC000_21[0...A]6 [PB Machine Check Syndrome] (MCA::PB::MCA_SYND_PB)

Read-write,Volatile. Reset: Cold,0000_0000_0000_0000h.

Logs physical location information associated with error in MCA::PB::MCA_STATUS_PB Thread 0

_ccd[7:0]_instPBCCD_n[8:2,0]_aliasMSR; MSRC000_2106

_instPB_n1_aliasMSR; MSRC000_21A6

Bits	Description
63:33	Reserved.
32	**Syndrome**. Read-write,Volatile. Reset: Cold,0. Contains the syndrome, if any, associated with the error logged in MCA::PB::MCA_STATUS_PB. The low-order bit of the syndrome is stored in bit 0, and the syndrome has a length specified by MCA::PB::MCA_SYND_PB[Length]. The Syndrome field is only valid when MCA::PB::MCA_SYND_PB[Length] is not 0.
31:27	Reserved.
26:24	**ErrorPriority**. Read-write,Volatile. Reset: Cold,0h. Encodes the priority of the error logged in MCA::PB::MCA_SYND_PB. 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	**Length**. Read-write,Volatile. Reset: Cold,00h. Specifies the length in bits of the syndrome contained in MCA::PB::MCA_SYND_PB[Syndrome]. A value of 0 indicates that there is no valid syndrome in MCA::PB::MCA_SYND_PB. For example, a syndrome length of 9 means that MCA::PB::MCA_SYND_PB[Syndrome] bits [8:0] contains a valid syndrome.
17:0	**ErrorInformation**. Read-write,Volatile. Reset: Cold,0_0000h. Contains error-specific information about the location of the error. Decoding is available in Table 75 [MCA_SYND_PB].

Table 75: MCA_SYND_PB

Error Type	Bits	Description
EccError	[17:0]	Reserved

MSRC001_041[0...A] [PB Machine Check Control Mask] (MCA::PB::MCA_CTL_MASK_PB)

Read-write. Reset: 0000_0000_0000_0000h.

Inhibit detection of an error source.

_ccd[7:0]_instPBCCD_n[8:2,0]_aliasMSR; MSRC001_0410

_instPB_n1_aliasMSR; MSRC001_041A

Bits	Description
63:1	Reserved.
0	**EccError**. Read-write. Reset: 0. An ECC error in the Parameter Block RAM array.

3.2.5.12 PSP

MSR0000_0464...MSRC000_2190 [PSP Machine Check Control] (MCA::PSP::MCA_CTL_PSP)

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::PSP::MCA_CTL_PSP register must be enabled by the corresponding enable bit in Core::X86::Msr::MCG_CTL. Does not affect error detection, correction, or logging.

_instMP0MP0_n0_aliasMSRLEGACY; MSR0000_0464

_instMP0MP0_n0_aliasMSR; MSRC000_2190

Bits	Description
63:18	Reserved.
17	**Mp0SHubIfRdBufError**. Read-write. Reset: 0. System Hub Read Buffer ECC or parity error.
16	**Mp0TlbBank1Error**. Read-write. Reset: 0. TLB Bank 1 parity error.
15	**Mp0TlbBank0Error**. Read-write. Reset: 0. TLB Bank 0 parity error.
14	**Mp0DDirtyRamError**. Read-write. Reset: 0. Dirty Data Ram parity error.
13	**Mp0DTagBank3Error**. Read-write. Reset: 0. Data Tag Bank 3 parity error.
12	**Mp0DTagBank2Error**. Read-write. Reset: 0. Data Tag Bank 2 parity error.
11	**Mp0DTagBank1Error**. Read-write. Reset: 0. Data Tag Bank 1 parity error.
10	**Mp0DTagBank0Error**. Read-write. Reset: 0. Data Tag Bank 0 parity error.
9	**Mp0DDataBank3Error**. Read-write. Reset: 0. Data Cache Bank 3 ECC or parity error.
8	**Mp0DDataBank2Error**. Read-write. Reset: 0. Data Cache Bank 2 ECC or parity error.
7	**Mp0DDataBank1Error**. Read-write. Reset: 0. Data Cache Bank 1 ECC or parity error.
6	**Mp0DDataBank0Error**. Read-write. Reset: 0. Data Cache Bank 0 ECC or parity error.
5	**Mp0ITagRam1Error**. Read-write. Reset: 0. Instruction Tag Ram 1 parity error.
4	**Mp0ITagRam0Error**. Read-write. Reset: 0. Instruction Tag Ram 0 parity error.
3	**Mp0IDataBank1Error**. Read-write. Reset: 0. Instruction Cache Bank 1 ECC or parity error.
2	**Mp0IDataBank0Error**. Read-write. Reset: 0. Instruction Cache Bank 0 ECC or parity error.
1	**Mp0LowSramError**. Read-write. Reset: 0. Low SRAM ECC or parity error.
0	**Mp0HighSramError**. Read-write. Reset: 0. High SRAM ECC or parity error.

MSR0000_0465...MSRC000_2191 [PSP Machine Check Status] (MCA::PSP::MCA_STATUS_PSP)

Reset: Cold,0000_0000_0000_0000h.

Logs information associated with errors.

_instMP0MP0_n0_aliasMSRLEGACY; MSR0000_0465

_instMP0MP0_n0_aliasMSR; MSRC000_2191

Bits	Description
63	**Val**. 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	**Overflow**. 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	**UC**. 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	**En**. Reset: Cold,0. 1=MCA error reporting is enabled for this error, as indicated by the corresponding bit in MCA::PSP::MCA_CTL_PSP. This bit is a copy of bit in MCA::PSP::MCA_CTL_PSP for this error.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
59	**MiscV**. Reset: Cold,0. 1=Valid thresholding in MCA::PSP::MCA_MISC0_PSP. In certain modes, MISC registers are owned by platform firmware and will RAZ when read by non-SMM code. Therefore, it is possible for MiscV=1 and the MISC register to read as all zeros.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
58	**AddrV**. Reset: Cold,0. 1=MCA::PSP::MCA_ADDR_PSP contains address information associated with the error.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
57	**PCC**. 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	**ErrCoreIdVal**. 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	**TCC**. Reset: Cold,0. 1=Hardware context of the process thread to which the error was reported may have been corrupted. Continued operation of the thread may have unpredictable results. The thread must be terminated. Only meaningful when MCA::PSP::MCA_STATUS_PSP[PCC]=0.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
54	**RESERV4**. Reset: Cold,0. MCA_STATUS Register Reserved bit.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
53	**SyndV**. Reset: Cold,0. 1=This error logged information in MCA::PSP::MCA_SYND_PSP. If MCA::PSP::MCA_SYND_PSP[ErrorPriority] is the same as the priority of the error in MCA::PSP::MCA_STATUS_PSP, then the information in MCA::PSP::MCA_SYND_PSP is associated with the error in MCA::PSP::MCA_STATUS_PSP.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
52	Reserved.
51:47	**RESERV3**. Reset: Cold,00h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
46	**CECC**. 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	**UECC**. 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	**Deferred**. 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	**Poison**. 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.
42:41	**RESERV2**. Reset: Cold,0h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
40	**Scrub**. Reset: Cold,0. 1=The error was the result of a scrub operation.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
39:38	**RESERV1**. Reset: Cold,0h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
37:32	**ErrCoreId**. 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	**RESERV0**. Reset: Cold,000h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
21:16	**ErrorCodeExt**. 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::PSP::MCA_CTL_PSP 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	**ErrorCode**. 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 76: MCA_STATUS_PSP

Error Type	ErrorCode Ext	UC	PCC	TCC	Deferred	Poison	AddrV
Mp0HighSramError	0x0	0/1	0/1	0/1	0	0	1
Mp0LowSramError	0x1	0/1	0/1	0/1	0	0	1
Mp0IDataBank0Error	0x2	0/1	0/1	0/1	0	0	1
Mp0IDataBank1Error	0x3	0/1	0/1	0/1	0	0	1
Mp0ITagRam0Error	0x4	1	1	1	0	0	1
Mp0ITagRam1Error	0x5	1	1	1	0	0	1
Mp0DDataBank0Error	0x6	0/1	0/1	0/1	0	0	1
Mp0DDataBank1Error	0x7	0/1	0/1	0/1	0	0	1
Mp0DDataBank2Error	0x8	0/1	0/1	0/1	0	0	1
Mp0DDataBank3Error	0x9	0/1	0/1	0/1	0	0	1
Mp0DTagBank0Error	0xa	1	1	1	0	0	1

Mp0DTagBank1Error	0xb	1	1	1	0	0	1
Mp0DTagBank2Error	0xc	1	1	1	0	0	1
Mp0DTagBank3Error	0xd	1	1	1	0	0	1
Mp0DDirtyRamError	0xe	1	1	1	0	0	1
Mp0TlbBank0Error	0xf	1	1	1	0	0	1
Mp0TlbBank1Error	0x10	1	1	1	0	0	1
Mp0SHubIfRdBufError	0x11	1	1	1	0	0	1
TwixError	0x3E	0	0	0	0	0	0
WaflError	0x3F	0	0	0	0	0	0

MSR0000_0466...MSRC000_2192 (MCA::PSP::MCA_ADDR_PSP)

Read-only. Reset: Cold,0000_0000_0000_0000h.

_instMP0MP0_n0_aliasMSRLEGACY; MSR0000_0466

_instMP0MP0_n0_aliasMSR; MSRC000_2192

Bits	Description
63:62	Reserved.
61:56	**LSB**. Read-only. Reset: Cold,00h. Specifies the least significant valid bit of the address contained in MCA::PSP::MCA_ADDR_PSP[ErrorAddr]. For example, a value of 0 indicates that MCA::PSP::MCA_ADDR_PSP[55:0] contains a valid byte address. A value of 6 indicates that MCA::PSP::MCA_ADDR_PSP[55:6] contains a valid cache line address and that MCA::PSP::MCA_ADDR_PSP[5:0] are not part of the address and should be ignored by error handling software. A value of 12 indicates that MCA::PSP::MCA_ADDR_PSP[55:12] contain a valid 4KB memory page and that MCA::PSP::MCA_ADDR_PSP[11:0] should be ignored by error handling software.
55:0	**ErrorAddr**. Read-only. Reset: Cold,00_0000_0000_0000h. Contains the address, if any, associated with the error logged in MCA::PSP::MCA_STATUS_PSP.

Table 77: MCA_ADDR_PSP

Error Type	Bits	Description
Mp0HighSramError	[55:0]	Reserved
Mp0LowSramError	[55:0]	Reserved
Mp0IDataBank0Error	[55:0]	Reserved
Mp0IDataBank1Error	[55:0]	Reserved
Mp0ITagRam0Error	[55:0]	Reserved
Mp0ITagRam1Error	[55:0]	Reserved
Mp0DDataBank0Error	[55:0]	Reserved
Mp0DDataBank1Error	[55:0]	Reserved
Mp0DDataBank2Error	[55:0]	Reserved
Mp0DDataBank3Error	[55:0]	Reserved
Mp0DTagBank0Error	[55:0]	Reserved
Mp0DTagBank1Error	[55:0]	Reserved
Mp0DTagBank2Error	[55:0]	Reserved
Mp0DTagBank3Error	[55:0]	Reserved
Mp0DDirtyRamError	[55:0]	Reserved

Mp0TlbBank0Error	[55:0]	Reserved
Mp0TlbBank1Error	[55:0]	Reserved
Mp0SHubIfRdBufError	[55:0]	Reserved
TwixError	[55:0]	Reserved
WaflError	[55:0]	Reserved

MSR0000_0467...MSRC000_2193 [PSP Machine Check Miscellaneous 0] (MCA::PSP::MCA_MISC0_PSP)

Log miscellaneous information associated with errors.

_instMP0MP0_n0_aliasMSRLEGACY; MSR0000_0467

_instMP0MP0_n0_aliasMSR; MSRC000_2193

Bits	Description
63	**Valid**. Reset: 1. 1=A valid CntP field is present in this register.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read-only.
62	**CntP**. Reset: 1. 1=A valid threshold counter is present.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read-only.
61	**Locked**. 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	**IntP**. 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::PSP::MCA_MISC0_PSP[Locked]) ? Read-write : Read-only.
59:56	Reserved.
55:52	**LvtOffset**. 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::PSP::MCA_MISC0_PSP[Locked]) ? Read-write : Read-only.
51	**CntEn**. Reset: 0. 1=Count thresholding errors.
	AccessType: (Core::X86::Msr::HWCR[McStatusWrEn] | !MCA::PSP::MCA_MISC0_PSP[Locked]) ? Read-write : Read-only.
50:49	**ThresholdIntType**. 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::PSP::MCA_MISC0_PSP[Locked]) ? Read-write : Read-only.
48	**Ovrflw**. 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::PSP::MCA_MISC0_PSP[Locked]) ? Read-write : Read-only.
47:44	Reserved.
43:32	**ErrCnt**. 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::PSP::MCA_MISC0_PSP[Locked]) ? Read-write : Read-only.
31:24	**BlkPtr**. Read-write. Reset: 00h. 00h=Extended MISC MSR block is not valid. 01h=Extended MSR block is valid.
23:0	Reserved.

MSRC000_2194 [PSP Machine Check Configuration] (MCA::PSP::MCA_CONFIG_PSP)

Reset: 0000_0002_0000_0021h.

Controls configuration of the associated machine check bank.

_instMP0MP0_n0_aliasMSR; MSRC000_2194

Bits	Description
63:39	Reserved.
38:37	**DeferredIntType**. 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	**McaXEnable**. 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	**DeferredIntTypeSupported**. Read-only. Reset: 1. 1=MCA::PSP::MCA_CONFIG_PSP[DeferredIntType] controls the type of interrupt generated on a deferred error. Deferred errors are supported in this bank only if MCA::PSP::MCA_CONFIG_PSP[DeferredErrorLoggingSupported]=1.
4:3	Reserved.
2	**DeferredErrorLoggingSupported**. 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	**McaX**. Read-only. Reset: 1. 1=This bank provides Machine Check Architecture Extensions. Up to 4 additional MISC registers (MISC1-MISC4) are supported. MCA::PSP::MCA_MISC0_PSP[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::PSP::MCA_STATUS_PSP[TCC] is present.

MSRC000_2195 [PSP IP Identification] (MCA::PSP::MCA_IPID_PSP)

Reset: 0001_00FF_0000_0000h.

The MCA::PSP::MCA_IPID_PSP register is used by software to determine what IP type and revision is associated with the MCA bank.

_instMP0MP0_n0_aliasMSR; MSRC000_2195

Bits	Description
63:48	**McaType**. Read-only. Reset: 0001h. The McaType of the MCA bank within this IP.
47:44	Reserved.
43:32	**HardwareID**. Read-only. Reset: 0FFh. The Hardware ID of the IP associated with this MCA bank.
31:0	**InstanceId**. Read-write. Reset: 0000_0000h. Init: 0383_0400h. The instance ID of this IP. This is initialized to a unique ID per instance of this register.

MSRC000_2196 [PSP Machine Check Syndrome] (MCA::PSP::MCA_SYND_PSP)

Read-write,Volatile. Reset: Cold,0000_0000_0000_0000h.

Logs physical location information associated with error in MCA::PSP::MCA_STATUS_PSP Thread 0

_instMP0MP0_n0_aliasMSR; MSRC000_2196

Bits	Description
63:27	Reserved.
26:24	**ErrorPriority**. Read-write,Volatile. Reset: Cold,0h. Encodes the priority of the error logged in MCA::PSP::MCA_SYND_PSP. 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	**Length**. Read-write,Volatile. Reset: Cold,00h. Specifies the length in bits of any syndromes logged. Only meaningful if the Syndrome field exists in this register.

17:0	**ErrorInformation**. Read-write,Volatile. Reset: Cold,0_0000h. Contains error-specific information about the location of the error. Decoding is available in Table 78 [MCA_SYND_PSP].

Table 78: MCA_SYND_PSP

Error Type	Bits	Description
Mp0HighSramError	[17:0]	Reserved
Mp0LowSramError	[17:0]	Reserved
Mp0IDataBank0Error	[17:9] [8:0]	Reserved Reserved
Mp0IDataBank1Error	[17:9] [8:0]	Reserved Reserved
Mp0ITagRam0Error	[17:7] [6:0]	Reserved Reserved
Mp0ITagRam1Error	[17:7] [6:0]	Reserved Reserved
Mp0DDataBank0Error	[17:9] [8:0]	Reserved Reserved
Mp0DDataBank1Error	[17:9] [8:0]	Reserved Reserved
Mp0DDataBank2Error	[17:9] [8:0]	Reserved Reserved
Mp0DDataBank3Error	[17:9] [8:0]	Reserved Reserved
Mp0DTagBank0Error	[17:6] [5:0]	Reserved Reserved
Mp0DTagBank1Error	[17:6] [5:0]	Reserved Reserved
Mp0DTagBank2Error	[17:6] [5:0]	Reserved Reserved
Mp0DTagBank3Error	[17:6] [5:0]	Reserved Reserved
Mp0DDirtyRamError	[17:6] [5:0]	Reserved Reserved
Mp0TlbBank0Error	[17:6] [5:0]	Reserved Reserved
Mp0TlbBank1Error	[17:6] [5:0]	Reserved Reserved
Mp0SHubIfRdBufError	[17:6] [5:0]	Reserved Reserved
TwixError	[17:0]	Reserved
WaflError	[17:0]	Reserved

MSRC001_0419 [PSP Machine Check Control Mask] (MCA::PSP::MCA_CTL_MASK_PSP)

Read-write. Reset: 0000_0000_0000_0000h.

Inhibit detection of an error source.

_instMP0MP0_n0_aliasMSR; MSRC001_0419

Bits	Description
63:18	Reserved.

17	**Mp0SHubIfRdBufError**. Read-write. Reset: 0. System Hub Read Buffer ECC or parity error.
16	**Mp0TlbBank1Error**. Read-write. Reset: 0. TLB Bank 1 parity error.
15	**Mp0TlbBank0Error**. Read-write. Reset: 0. TLB Bank 0 parity error.
14	**Mp0DDirtyRamError**. Read-write. Reset: 0. Dirty Data Ram parity error.
13	**Mp0DTagBank3Error**. Read-write. Reset: 0. Data Tag Bank 3 parity error.
12	**Mp0DTagBank2Error**. Read-write. Reset: 0. Data Tag Bank 2 parity error.
11	**Mp0DTagBank1Error**. Read-write. Reset: 0. Data Tag Bank 1 parity error.
10	**Mp0DTagBank0Error**. Read-write. Reset: 0. Data Tag Bank 0 parity error.
9	**Mp0DDataBank3Error**. Read-write. Reset: 0. Data Cache Bank 3 ECC or parity error.
8	**Mp0DDataBank2Error**. Read-write. Reset: 0. Data Cache Bank 2 ECC or parity error.
7	**Mp0DDataBank1Error**. Read-write. Reset: 0. Data Cache Bank 1 ECC or parity error.
6	**Mp0DDataBank0Error**. Read-write. Reset: 0. Data Cache Bank 0 ECC or parity error.
5	**Mp0ITagRam1Error**. Read-write. Reset: 0. Instruction Tag Ram 1 parity error.
4	**Mp0ITagRam0Error**. Read-write. Reset: 0. Instruction Tag Ram 0 parity error.
3	**Mp0IDataBank1Error**. Read-write. Reset: 0. Instruction Cache Bank 1 ECC or parity error.
2	**Mp0IDataBank0Error**. Read-write. Reset: 0. Instruction Cache Bank 0 ECC or parity error.
1	**Mp0LowSramError**. Read-write. Reset: 0. Low SRAM ECC or parity error.
0	**Mp0HighSramError**. Read-write. Reset: 0. High SRAM ECC or parity error.

3.2.5.13 SMU

MSR0000_0460...MSRC000_2180 [SMU Machine Check Control] (MCA::SMU::MCA_CTL_SMU)

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::SMU::MCA_CTL_SMU register must be enabled by the corresponding enable bit in Core::X86::Msr::MCG_CTL. Does not affect error detection, correction, or logging.

_instMP1MP1_n0_aliasMSRLEGACY; MSR0000_0460

_instMP1MP1_n0_aliasMSR; MSRC000_2180

Bits	Description
63:11	Reserved.
10	**Mp1SHubIfRdBufError**. Read-write. Reset: 0. System Hub Read Buffer ECC or parity error.
9	**Mp1ITagBError**. Read-write. Reset: 0. Instruction Tag Cache Bank B ECC or parity error.
8	**Mp1ITagAError**. Read-write. Reset: 0. Instruction Tag Cache Bank A ECC or parity error.
7	**Mp1ICacheBError**. Read-write. Reset: 0. Instruction Cache Bank B ECC or parity error.
6	**Mp1ICacheAError**. Read-write. Reset: 0. Instruction Cache Bank A ECC or parity error.
5	**Mp1DTagBError**. Read-write. Reset: 0. Data Tag Cache Bank B ECC or parity error.
4	**Mp1DTagAError**. Read-write. Reset: 0. Data Tag Cache Bank A ECC or parity error.
3	**Mp1DCacheBError**. Read-write. Reset: 0. Data Cache Bank B ECC or parity error.
2	**Mp1DCacheAError**. Read-write. Reset: 0. Data Cache Bank A ECC or parity error.
1	**Mp1LowSramError**. Read-write. Reset: 0. Low SRAM ECC or parity error.
0	**Mp1HighSramError**. Read-write. Reset: 0. High SRAM ECC or parity error.

MSR0000_0461...MSRC000_2181 [SMU Machine Check Status] (MCA::SMU::MCA_STATUS_SMU)

Reset: Cold,0000_0000_0000_0000h.

Logs information associated with errors.

_instMP1MP1_n0_aliasMSRLEGACY; MSR0000_0461

_instMP1MP1_n0_aliasMSR; MSRC000_2181

Bits	Description
63	**Val**. 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	**Overflow**. 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	**UC**. 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	**En**. Reset: Cold,0. 1=MCA error reporting is enabled for this error, as indicated by the corresponding bit in MCA::SMU::MCA_CTL_SMU. This bit is a copy of bit in MCA::SMU::MCA_CTL_SMU for this error.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
59	**MiscV**. Reset: Cold,0. 1=Valid thresholding in MCA::SMU::MCA_MISC0_SMU. In certain modes, MISC registers are owned by platform firmware and will RAZ when read by non-SMM code. Therefore, it is possible for MiscV=1 and the MISC register to read as all zeros.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
58	**AddrV**. Reset: Cold,0. 1=MCA::SMU::MCA_ADDR_SMU contains address information associated with the error.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
57	**PCC**. 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	**ErrCoreIdVal**. 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	**TCC**. Reset: Cold,0. 1=Hardware context of the process thread to which the error was reported may have been corrupted. Continued operation of the thread may have unpredictable results. The thread must be terminated. Only meaningful when MCA::SMU::MCA_STATUS_SMU[PCC]=0.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
54	**RESERV4**. Reset: Cold,0. MCA_STATUS Register Reserved bit.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
53	**SyndV**. Reset: Cold,0. 1=This error logged information in MCA::SMU::MCA_SYND_SMU. If MCA::SMU::MCA_SYND_SMU[ErrorPriority] is the same as the priority of the error in MCA::SMU::MCA_STATUS_SMU, then the information in MCA::SMU::MCA_SYND_SMU is associated with the error in MCA::SMU::MCA_STATUS_SMU.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
52	Reserved.
51:47	**RESERV3**. Reset: Cold,00h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
46	**CECC**. 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	**UECC**. 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	**Deferred**. 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	**Poison**. 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.

42:41	**RESERV2**. Reset: Cold,0h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
40	**Scrub**. Reset: Cold,0. 1=The error was the result of a scrub operation.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
39:38	**RESERV1**. Reset: Cold,0h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
37:32	**ErrCoreId**. 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	**RESERV0**. Reset: Cold,000h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
21:16	**ErrorCodeExt**. 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::SMU::MCA_CTL_SMU 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	**ErrorCode**. 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 79: MCA_STATUS_SMU

Error Type	ErrorCode Ext	UC	PCC	TCC	Deferred	Poison	AddrV
Mp1HighSramError	0x0	0/1	0/1	0/1	0	0	1
Mp1LowSramError	0x1	0/1	0/1	0/1	0	0	1
Mp1DCacheAError	0x2	0/1	0/1	0/1	0	0	1
Mp1DCacheBError	0x3	0/1	0/1	0/1	0	0	1
Mp1DTagAError	0x4	0/1	0/1	0/1	0	0	1
Mp1DTagBError	0x5	0/1	0/1	0/1	0	0	1
Mp1ICacheAError	0x6	0/1	0/1	0/1	0	0	1
Mp1ICacheBError	0x7	0/1	0/1	0/1	0	0	1
Mp1ITagAError	0x8	0/1	0/1	0/1	0	0	1
Mp1ITagBError	0x9	0/1	0/1	0/1	0	0	1
Mp1SHubIfRdBufError	0xa	0/1	0/1	0/1	0	0	1
PhyRamEccError	0xb	0	0	0	0	0	0
EdcIndicator	0x39	0	0	0	0	0	0

MSR0000_0462...MSRC000_2182 (MCA::SMU::MCA_ADDR_SMU)

Read-only. Reset: Cold,0000_0000_0000_0000h.

_instMP1MP1_n0_aliasMSRLEGACY; MSR0000_0462	
_instMP1MP1_n0_aliasMSR; MSRC000_2182	
Bits	**Description**
63:62	Reserved.
61:56	**LSB**. Read-only. Reset: Cold,00h. Specifies the least significant valid bit of the address contained in MCA::SMU::MCA_ADDR_SMU[ErrorAddr]. For example, a value of 0 indicates that MCA::SMU::MCA_ADDR_SMU[55:0] contains a valid byte address. A value of 6 indicates that MCA::SMU::MCA_ADDR_SMU[55:6] contains a valid cache line address and that MCA::SMU::MCA_ADDR_SMU[5:0] are not part of the address and should be ignored by error handling software. A value of 12 indicates that MCA::SMU::MCA_ADDR_SMU[55:12] contain a valid 4KB memory page and that MCA::SMU::MCA_ADDR_SMU[11:0] should be ignored by error handling software.
55:0	**ErrorAddr**. Read-only. Reset: Cold,00_0000_0000_0000h. Contains the address, if any, associated with the error logged in MCA::SMU::MCA_STATUS_SMU.

Table 80: MCA_ADDR_SMU

Error Type	Bits	Description
Mp1HighSramError	[55:0]	Reserved
Mp1LowSramError	[55:0]	Reserved
Mp1DCacheAError	[55:0]	Reserved
Mp1DCacheBError	[55:0]	Reserved
Mp1DTagAError	[55:0]	Reserved
Mp1DTagBError	[55:0]	Reserved
Mp1ICacheAError	[55:0]	Reserved
Mp1ICacheBError	[55:0]	Reserved
Mp1ITagAError	[55:0]	Reserved
Mp1ITagBError	[55:0]	Reserved
Mp1SHubIfRdBufError	[55:0]	Reserved
PhyRamEccError	[55:0]	Reserved
EdcIndicator	[55:0]	Reserved

MSR0000_0463...MSRC000_2183 [SMU Machine Check Miscellaneous 0] (MCA::SMU::MCA_MISC0_SMU)

Log miscellaneous information associated with errors.	
_instMP1MP1_n0_aliasMSRLEGACY; MSR0000_0463	
_instMP1MP1_n0_aliasMSR; MSRC000_2183	
Bits	**Description**
63	**Valid**. Reset: 1. 1=A valid CntP field is present in this register.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read-only.
62	**CntP**. Reset: 1. 1=A valid threshold counter is present.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read-only.
61	**Locked**. 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	**IntP**. 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::SMU::MCA_MISC0_SMU[Locked]) ? Read-write : Read-only.
59:56	Reserved.
55:52	**LvtOffset**. 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::SMU::MCA_MISC0_SMU[Locked]) ? Read-write : Read-only.
51	**CntEn**. Reset: 0. 1=Count thresholding errors.
	AccessType: (Core::X86::Msr::HWCR[McStatusWrEn] | !MCA::SMU::MCA_MISC0_SMU[Locked]) ? Read-write : Read-only.
50:49	**ThresholdIntType**. 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::SMU::MCA_MISC0_SMU[Locked]) ? Read-write : Read-only.
48	**Ovrflw**. 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::SMU::MCA_MISC0_SMU[Locked]) ? Read-write : Read-only.
47:44	Reserved.
43:32	**ErrCnt**. 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::SMU::MCA_MISC0_SMU[Locked]) ? Read-write : Read-only.
31:24	**BlkPtr**. Read-write. Reset: 00h. 00h=Extended MISC MSR block is not valid. 01h=Extended MSR block is valid.
23:0	Reserved.

MSRC000_2184 [SMU Machine Check Configuration] (MCA::SMU::MCA_CONFIG_SMU)

Reset: 0000_0002_0000_0021h.

Controls configuration of the associated machine check bank.

_instMP1MP1_n0_aliasMSR; MSRC000_2184

Bits	Description
63:39	Reserved.
38:37	**DeferredIntType**. 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	**McaXEnable**. 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	**DeferredIntTypeSupported**. Read-only. Reset: 1. 1=MCA::SMU::MCA_CONFIG_SMU[DeferredIntType] controls the type of interrupt generated on a deferred error. Deferred errors are supported in this bank only if MCA::SMU::MCA_CONFIG_SMU[DeferredErrorLoggingSupported]=1.
4:3	Reserved.
2	**DeferredErrorLoggingSupported**. 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	**McaX**. Read-only. Reset: 1. 1=This bank provides Machine Check Architecture Extensions. Up to 4 additional MISC registers (MISC1-MISC4) are supported. MCA::SMU::MCA_MISC0_SMU[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::SMU::MCA_STATUS_SMU[TCC] is present.

MSRC000_2185 [SMU IP Identification] (MCA::SMU::MCA_IPID_SMU)

Reset: 0001_0001_0000_0000h.
The MCA::SMU::MCA_IPID_SMU register is used by software to determine what IP type and revision is associated with the MCA bank.
_instMP1MP1_n0_aliasMSR; MSRC000_2185

Bits	Description
63:48	**McaType**. Read-only. Reset: 0001h. The McaType of the MCA bank within this IP.
47:44	Reserved.
43:32	**HardwareID**. Read-only. Reset: 001h. The Hardware ID of the IP associated with this MCA bank.
31:0	**InstanceId**. Read-write. Reset: 0000_0000h. Init: 03B3_0400h. The instance ID of this IP. This is initialized to a unique ID per instance of this register.

MSRC000_2186 [SMU Machine Check Syndrome] (MCA::SMU::MCA_SYND_SMU)

Read-write,Volatile. Reset: Cold,0000_0000_0000_0000h.
Logs physical location information associated with error in MCA::SMU::MCA_STATUS_SMU Thread 0
_instMP1MP1_n0_aliasMSR; MSRC000_2186

Bits	Description
63:27	Reserved.
26:24	**ErrorPriority**. Read-write,Volatile. Reset: Cold,0h. Encodes the priority of the error logged in MCA::SMU::MCA_SYND_SMU. 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	**Length**. Read-write,Volatile. Reset: Cold,00h. Specifies the length in bits of any syndromes logged. Only meaningful if the Syndrome field exists in this register.
17:0	**ErrorInformation**. Read-write,Volatile. Reset: Cold,0_0000h. Contains error-specific information about the location of the error. Decoding is available in Table 81 [MCA_SYND_SMU].

Table 81: MCA_SYND_SMU

Error Type	Bits	Description
Mp1HighSramError	[17:15] [14:0]	Reserved Reserved
Mp1LowSramError	[17:15] [14:0]	Reserved Reserved
Mp1DCacheAError	[17:8] [7:0]	Reserved Reserved
Mp1DCacheBError	[17:8] [7:0]	Reserved Reserved
Mp1DTagAError	[17:7] [6:0]	Reserved Reserved
Mp1DTagBError	[17:7] [6:0]	Reserved Reserved
Mp1ICacheAError	[17:8] [7:0]	Reserved Reserved
Mp1ICacheBError	[17:8] [7:0]	Reserved Reserved
Mp1ITagAError	[17:6] [5:0]	Reserved Reserved
Mp1ITagBError	[17:6]	Reserved

	[5:0]	Reserved
Mp1SHubIfRdBufError	[17:6] [5:0]	Reserved Reserved
PhyRamEccError	[17:0]	Reserved
EdcIndicator	[17:0]	Reserved

MSRC001_0418 [SMU Machine Check Control Mask] (MCA::SMU::MCA_CTL_MASK_SMU)

Read-write. Reset: 0000_0000_0000_0000h.

Inhibit detection of an error source.

_instMP1MP1_n0_aliasMSR; MSRC001_0418

Bits	Description
63:11	Reserved.
10	**Mp1SHubIfRdBufError**. Read-write. Reset: 0. System Hub Read Buffer ECC or parity error.
9	**Mp1ITagBError**. Read-write. Reset: 0. Instruction Tag Cache Bank B ECC or parity error.
8	**Mp1ITagAError**. Read-write. Reset: 0. Instruction Tag Cache Bank A ECC or parity error.
7	**Mp1ICacheBError**. Read-write. Reset: 0. Instruction Cache Bank B ECC or parity error.
6	**Mp1ICacheAError**. Read-write. Reset: 0. Instruction Cache Bank A ECC or parity error.
5	**Mp1DTagBError**. Read-write. Reset: 0. Data Tag Cache Bank B ECC or parity error.
4	**Mp1DTagAError**. Read-write. Reset: 0. Data Tag Cache Bank A ECC or parity error.
3	**Mp1DCacheBError**. Read-write. Reset: 0. Data Cache Bank B ECC or parity error.
2	**Mp1DCacheAError**. Read-write. Reset: 0. Data Cache Bank A ECC or parity error.
1	**Mp1LowSramError**. Read-write. Reset: 0. Low SRAM ECC or parity error.
0	**Mp1HighSramError**. Read-write. Reset: 0. High SRAM ECC or parity error.

3.2.5.14 MP5

MSR0000_043C...MSRC000_20F0 [MP5 Machine Check Control] (MCA::MP5::MCA_CTL_MP5)

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::MP5::MCA_CTL_MP5 register must be enabled by the corresponding enable bit in Core::X86::Msr::MCG_CTL. Does not affect error detection, correction, or logging.

_ccd[7:0]_instMP5MP5_n[7:0]_aliasMSRLEGACY; MSR0000_043C

_ccd[7:0]_instMP5MP5_n[7:0]_aliasMSR; MSRC000_20F0

Bits	Description
63:10	Reserved.
9	**Mp5ITagBError**. Read-write. Reset: 0. Instruction Tag Cache Bank B ECC or parity error.
8	**Mp5ITagAError**. Read-write. Reset: 0. Instruction Tag Cache Bank A ECC or parity error.
7	**Mp5ICacheBError**. Read-write. Reset: 0. Instruction Cache Bank B ECC or parity error.
6	**Mp5ICacheAError**. Read-write. Reset: 0. Instruction Cache Bank A ECC or parity error.
5	**Mp5DTagBError**. Read-write. Reset: 0. Data Tag Cache Bank B ECC or parity error.
4	**Mp5DTagAError**. Read-write. Reset: 0. Data Tag Cache Bank A ECC or parity error.
3	**Mp5DCacheBError**. Read-write. Reset: 0. Data Cache Bank B ECC or parity error.
2	**Mp5DCacheAError**. Read-write. Reset: 0. Data Cache Bank A ECC or parity error.
1	**Mp5LowSramError**. Read-write. Reset: 0. Low SRAM ECC or parity error.
0	**Mp5HighSramError**. Read-write. Reset: 0. High SRAM ECC or parity error.

MSR0000_043D...MSRC000_20F1 [MP5 Machine Check Status] (MCA::MP5::MCA_STATUS_MP5)

Reset: Cold,0000_0000_0000_0000h.

Logs information associated with errors.

_ccd[7:0]_instMP5MP5_n[7:0]_aliasMSRLEGACY; MSR0000_043D
_ccd[7:0]_instMP5MP5_n[7:0]_aliasMSR; MSRC000_20F1

Bits	Description
63	**Val**. 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	**Overflow**. 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	**UC**. 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	**En**. Reset: Cold,0. 1=MCA error reporting is enabled for this error, as indicated by the corresponding bit in MCA::MP5::MCA_CTL_MP5. This bit is a copy of bit in MCA::MP5::MCA_CTL_MP5 for this error.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
59	**MiscV**. Reset: Cold,0. 1=Valid thresholding in MCA::MP5::MCA_MISC0_MP5. In certain modes, MISC registers are owned by platform firmware and will RAZ when read by non-SMM code. Therefore, it is possible for MiscV=1 and the MISC register to read as all zeros.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
58	**AddrV**. Reset: Cold,0. 1=MCA::MP5::MCA_ADDR_MP5 contains address information associated with the error.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
57	**PCC**. 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	**ErrCoreIdVal**. 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	**TCC**. Reset: Cold,0. 1=Hardware context of the process thread to which the error was reported may have been corrupted. Continued operation of the thread may have unpredictable results. The thread must be terminated. Only meaningful when MCA::MP5::MCA_STATUS_MP5[PCC]=0.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
54	**RESERV4**. Reset: Cold,0. MCA_STATUS Register Reserved bit.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
53	**SyndV**. Reset: Cold,0. 1=This error logged information in MCA::MP5::MCA_SYND_MP5. If MCA::MP5::MCA_SYND_MP5[ErrorPriority] is the same as the priority of the error in MCA::MP5::MCA_STATUS_MP5, then the information in MCA::MP5::MCA_SYND_MP5 is associated with the error in MCA::MP5::MCA_STATUS_MP5.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
52	Reserved.
51:47	**RESERV3**. Reset: Cold,00h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
46	**CECC**. 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	**UECC**. 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	**Deferred**. 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	**Poison**. 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.
42:41	**RESERV2**. Reset: Cold,0h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
40	**Scrub**. Reset: Cold,0. 1=The error was the result of a scrub operation.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
39:38	**RESERV1**. Reset: Cold,0h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
37:32	**ErrCoreId**. 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	**RESERV0**. Reset: Cold,000h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
21:16	**ErrorCodeExt**. 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::MP5::MCA_CTL_MP5 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	**ErrorCode**. 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 82: MCA_STATUS_MP5

Error Type	ErrorCode Ext	UC	PCC	TCC	Deferred	Poison	AddrV
Mp5HighSramError	0x0	0/1	0/1	0/1	0	0	1
Mp5LowSramError	0x1	0/1	0/1	0/1	0	0	1
Mp5DCacheAError	0x2	0/1	0/1	0/1	0	0	1
Mp5DCacheBError	0x3	0/1	0/1	0/1	0	0	1
Mp5DTagAError	0x4	0/1	0/1	0/1	0	0	1
Mp5DTagBError	0x5	0/1	0/1	0/1	0	0	1
Mp5ICacheAError	0x6	0/1	0/1	0/1	0	0	1
Mp5ICacheBError	0x7	0/1	0/1	0/1	0	0	1
Mp5ITagAError	0x8	0/1	0/1	0/1	0	0	1
Mp5ITagBError	0x9	0/1	0/1	0/1	0	0	1

MSR0000_043E...MSRC000_20F2 (MCA::MP5::MCA_ADDR_MP5)

Read-only. Reset: Cold,0000_0000_0000_0000h.

_ccd[7:0]_instMP5MP5_n[7:0]_aliasMSRLEGACY; MSR0000_043E

_ccd[7:0]_instMP5MP5_n[7:0]_aliasMSR; MSRC000_20F2

Bits	Description
63:62	Reserved.
61:56	**LSB**. Read-only. Reset: Cold,00h. Specifies the least significant valid bit of the address contained in MCA::MP5::MCA_ADDR_MP5[ErrorAddr]. For example, a value of 0 indicates that MCA::MP5::MCA_ADDR_MP5[55:0] contains a valid byte address. A value of 6 indicates that MCA::MP5::MCA_ADDR_MP5[55:6] contains a valid cache line address and that MCA::MP5::MCA_ADDR_MP5[5:0] are not part of the address and should be ignored by error handling software. A value of 12 indicates that MCA::MP5::MCA_ADDR_MP5[55:12] contain a valid 4KB memory page and that MCA::MP5::MCA_ADDR_MP5[11:0] should be ignored by error handling software.
55:0	**ErrorAddr**. Read-only. Reset: Cold,00_0000_0000_0000h. Contains the address, if any, associated with the error logged in MCA::MP5::MCA_STATUS_MP5.

Table 83: MCA_ADDR_MP5

Error Type	Bits	Description
Mp5HighSramError	[55:0]	Reserved
Mp5LowSramError	[55:0]	Reserved
Mp5DCacheAError	[55:0]	Reserved
Mp5DCacheBError	[55:0]	Reserved
Mp5DTagAError	[55:0]	Reserved
Mp5DTagBError	[55:0]	Reserved
Mp5ICacheAError	[55:0]	Reserved
Mp5ICacheBError	[55:0]	Reserved
Mp5ITagAError	[55:0]	Reserved
Mp5ITagBError	[55:0]	Reserved

MSR0000_043F...MSRC000_20F3 [MP5 Machine Check Miscellaneous 0] (MCA::MP5::MCA_MISC0_MP5)

Log miscellaneous information associated with errors.

_ccd[7:0]_instMP5MP5_n[7:0]_aliasMSRLEGACY; MSR0000_043F

_ccd[7:0]_instMP5MP5_n[7:0]_aliasMSR; MSRC000_20F3

Bits	Description
63	**Valid**. Reset: 1. 1=A valid CntP field is present in this register.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read-only.
62	**CntP**. Reset: 1. 1=A valid threshold counter is present.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read-only.
61	**Locked**. 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	**IntP**. 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::MP5::MCA_MISC0_MP5[Locked]) ? Read-write : Read-only.
59:56	Reserved.
55:52	**LvtOffset**. 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::MP5::MCA_MISC0_MP5[Locked]) ? Read-write : Read-only.
51	**CntEn**. Reset: 0. 1=Count thresholding errors.
	AccessType: (Core::X86::Msr::HWCR[McStatusWrEn] | !MCA::MP5::MCA_MISC0_MP5[Locked]) ? Read-write : Read-only.

50:49	**ThresholdIntType**. 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::MP5::MCA_MISC0_MP5[Locked]) ? Read-write : Read-only.
48	**Ovrflw**. 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::MP5::MCA_MISC0_MP5[Locked]) ? Read-write : Read-only.
47:44	Reserved.
43:32	**ErrCnt**. 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::MP5::MCA_MISC0_MP5[Locked]) ? Read-write : Read-only.
31:24	**BlkPtr**. Read-write. Reset: 00h. 00h=Extended MISC MSR block is not valid. 01h=Extended MSR block is valid.
23:0	Reserved.

MSRC000_20F4 [MP5 Machine Check Configuration] (MCA::MP5::MCA_CONFIG_MP5)

Reset: 0000_0002_0000_0021h.

Controls configuration of the associated machine check bank.

_ccd[7:0]_instMP5MP5_n[7:0]_aliasMSR; MSRC000_20F4

Bits	Description
63:39	Reserved.
38:37	**DeferredIntType**. 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	**McaXEnable**. 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	**DeferredIntTypeSupported**. Read-only. Reset: 1. 1=MCA::MP5::MCA_CONFIG_MP5[DeferredIntType] controls the type of interrupt generated on a deferred error. Deferred errors are supported in this bank only if MCA::MP5::MCA_CONFIG_MP5[DeferredErrorLoggingSupported]=1.
4:3	Reserved.
2	**DeferredErrorLoggingSupported**. 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	**McaX**. Read-only. Reset: 1. 1=This bank provides Machine Check Architecture Extensions. Up to 4 additional MISC registers (MISC1-MISC4) are supported. MCA::MP5::MCA_MISC0_MP5[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::MP5::MCA_STATUS_MP5[TCC] is present.

MSRC000_20F5 [MP5 IP Identification] (MCA::MP5::MCA_IPID_MP5)

Reset: 0002_0001_0000_0000h.

The MCA::MP5::MCA_IPID_MP5 register is used by software to determine what IP type and revision is associated with

the MCA bank.	
_ccd[7:0]_instMP5MP5_n[7:0]_aliasMSR; MSRC000_20F5	
Bits	**Description**
63:48	**McaType**. Read-only. Reset: 0002h. The McaType of the MCA bank within this IP.
47:44	Reserved.
43:32	**HardwareID**. Read-only. Reset: 001h. The Hardware ID of the IP associated with this MCA bank.
31:0	**InstanceId**. Read-write. Reset: 0000_0000h. The instance ID of this IP. This is initialized to a unique ID per instance of this register.
	Init: _ccd0_instMP5MP5_n0_aliasMSR: 3043_0400h Init: _ccd1_instMP5MP5_n1_aliasMSR: 3243_0400h Init: _ccd2_instMP5MP5_n2_aliasMSR: 3443_0400h Init: _ccd3_instMP5MP5_n3_aliasMSR: 3643_0400h Init: _ccd4_instMP5MP5_n4_aliasMSR: 3843_0400h Init: _ccd5_instMP5MP5_n5_aliasMSR: 3A43_0400h Init: _ccd6_instMP5MP5_n6_aliasMSR: 3C43_0400h Init: _ccd7_instMP5MP5_n7_aliasMSR: 3E43_0400h

MSRC000_20F6 [MP5 Machine Check Syndrome] (MCA::MP5::MCA_SYND_MP5)

Read-write,Volatile. Reset: Cold,0000_0000_0000_0000h.	
Logs physical location information associated with error in MCA::MP5::MCA_STATUS_MP5 Thread 0	
_ccd[7:0]_instMP5MP5_n[7:0]_aliasMSR; MSRC000_20F6	
Bits	**Description**
63:27	Reserved.
26:24	**ErrorPriority**. Read-write,Volatile. Reset: Cold,0h. Encodes the priority of the error logged in MCA::MP5::MCA_SYND_MP5. 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	**Length**. Read-write,Volatile. Reset: Cold,00h. Specifies the length in bits of any syndromes logged. Only meaningful if the Syndrome field exists in this register.
17:0	**ErrorInformation**. Read-write,Volatile. Reset: Cold,0_0000h. Contains error-specific information about the location of the error. Decoding is available in Table 84 [MCA_SYND_MP5].

Table 84: MCA_SYND_MP5

Error Type	Bits	Description
Mp5HighSramError	[17:15] [14:0]	Reserved Reserved
Mp5LowSramError	[17:15] [14:0]	Reserved Reserved
Mp5DCacheAError	[17:8] [7:0]	Reserved Reserved
Mp5DCacheBError	[17:8] [7:0]	Reserved Reserved
Mp5DTagAError	[17:7] [6:0]	Reserved Reserved
Mp5DTagBError	[17:7] [6:0]	Reserved Reserved
Mp5ICacheAError	[17:8] [7:0]	Reserved Reserved
Mp5ICacheBError	[17:8] [7:0]	Reserved Reserved
Mp5ITagAError	[17:6] [5:0]	Reserved Reserved

Mp5ITagBError	[17:6] [5:0]	Reserved Reserved

MSRC001_040F [MP5 Machine Check Control Mask] (MCA::MP5::MCA_CTL_MASK_MP5)

Read-write. Reset: 0000_0000_0000_0000h.

Inhibit detection of an error source.

_ccd[7:0]_instMP5MP5_n[7:0]_aliasMSR; MSRC001_040F

Bits	Description
63:10	Reserved.
9	**Mp5ITagBError**. Read-write. Reset: 0. Instruction Tag Cache Bank B ECC or parity error.
8	**Mp5ITagAError**. Read-write. Reset: 0. Instruction Tag Cache Bank A ECC or parity error.
7	**Mp5ICacheBError**. Read-write. Reset: 0. Instruction Cache Bank B ECC or parity error.
6	**Mp5ICacheAError**. Read-write. Reset: 0. Instruction Cache Bank A ECC or parity error.
5	**Mp5DTagBError**. Read-write. Reset: 0. Data Tag Cache Bank B ECC or parity error.
4	**Mp5DTagAError**. Read-write. Reset: 0. Data Tag Cache Bank A ECC or parity error.
3	**Mp5DCacheBError**. Read-write. Reset: 0. Data Cache Bank B ECC or parity error.
2	**Mp5DCacheAError**. Read-write. Reset: 0. Data Cache Bank A ECC or parity error.
1	**Mp5LowSramError**. Read-write. Reset: 0. Low SRAM ECC or parity error.
0	**Mp5HighSramError**. Read-write. Reset: 0. High SRAM ECC or parity error.

3.2.5.15 NBIO

MSR0000_0458...MSRC000_2160 [NBIO Machine Check Control] (MCA::NBIO::MCA_CTL_NBIO)

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::NBIO::MCA_CTL_NBIO register must be enabled by the corresponding enable bit in Core::X86::Msr::MCG_CTL. Does not affect error detection, correction, or logging.

_instIOHC_n[3:0]_nbio[3:0]_aliasMSRLEGACY; MSR0000_0458

_instIOHC_n[3:0]_nbio[3:0]_aliasMSR; MSRC000_2160

Bits	Description
63:5	Reserved.
4	**IOHC_Internal_Poison**. Read-write. Reset: 0. Internal Poison Error. Poison data was sent to an internal client.
3	**Egress_Poison**. Read-write. Reset: 0. SDP Egress Poison Error. Poison was propagated to an egress port.
2	**ErrEvent**. Read-write. Reset: 0. SDP ErrEvent error. A system fatal error event from data fabric was detected.
1	**PCIE_Sideband**. Read-write. Reset: 0. PCIE error. A PCIe error was logged in a PCIe root port.
0	**EccParityError**. Read-write. Reset: 0. ECC or Parity error. An SRAM ECC or parity error was detected.

MSR0000_0459...MSRC000_2161 [NBIO Machine Check Status] (MCA::NBIO::MCA_STATUS_NBIO)

Reset: Cold,0000_0000_0000_0000h.

Logs information associated with errors.

_instIOHC_n[3:0]_nbio[3:0]_aliasMSRLEGACY; MSR0000_0459

_instIOHC_n[3:0]_nbio[3:0]_aliasMSR; MSRC000_2161

Bits	Description
63	**Val**. 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	**Overflow**. 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	**UC**. 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	**En**. Reset: Cold,0. 1=MCA error reporting is enabled for this error, as indicated by the corresponding bit in MCA::NBIO::MCA_CTL_NBIO. This bit is a copy of bit in MCA::NBIO::MCA_CTL_NBIO for this error.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
59	**MiscV**. Reset: Cold,0. 1=Valid thresholding in MCA::NBIO::MCA_MISC0_NBIO. In certain modes, MISC registers are owned by platform firmware and will RAZ when read by non-SMM code. Therefore, it is possible for MiscV=1 and the MISC register to read as all zeros.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
58	**AddrV**. Reset: Cold,0. 1=MCA::NBIO::MCA_ADDR_NBIO contains address information associated with the error.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
57	**PCC**. 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	**ErrCoreIdVal**. 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	**TCC**. Reset: Cold,0. 1=Hardware context of the process thread to which the error was reported may have been corrupted. Continued operation of the thread may have unpredictable results. The thread must be terminated. Only meaningful when MCA::NBIO::MCA_STATUS_NBIO[PCC]=0.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
54	**RESERV4**. Reset: Cold,0. MCA_STATUS Register Reserved bit.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
53	**SyndV**. Reset: Cold,0. 1=This error logged information in MCA::NBIO::MCA_SYND_NBIO. If MCA::NBIO::MCA_SYND_NBIO[ErrorPriority] is the same as the priority of the error in MCA::NBIO::MCA_STATUS_NBIO, then the information in MCA::NBIO::MCA_SYND_NBIO is associated with the error in MCA::NBIO::MCA_STATUS_NBIO.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
52	Reserved.
51:47	**RESERV3**. Reset: Cold,00h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
46	**CECC**. 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	**UECC**. 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	**Deferred**. 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	**Poison**. 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.
42:41	**RESERV2**. Reset: Cold,0h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
40	**Scrub**. Reset: Cold,0. 1=The error was the result of a scrub operation.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
39:38	**RESERV1**. Reset: Cold,0h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.

37:32	**ErrCoreId**. 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	**RESERV0**. Reset: Cold,000h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
21:16	**ErrorCodeExt**. 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::NBIO::MCA_CTL_NBIO 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	**ErrorCode**. 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 85: MCA_STATUS_NBIO

Error Type	ErrorCode Ext	UC	PCC	TCC	Deferred	Poison	AddrV
EccParityError	0x0	0/1	0/1	0/1	0/1	0	0
PCIE_Sideband	0x1	0/1	0/1	0/1	0/1	0	0
ErrEvent	0x2	1	1	1	0	0	0
Egress_Poison	0x3	0/1	0/1	0/1	0/1	0	0
IOHC_Internal_Poison	0x4	1	1	1	0	0	0

MSR0000_045A...MSRC000_2162 [NBIO Machine Check Address] (MCA::NBIO::MCA_ADDR_NBIO)

Read-only. Reset: Cold,0000_0000_0000_0000h.	
MCA::NBIO::MCA_ADDR_NBIO stores an address and other information associated with the error in MCA::NBIO::MCA_STATUS_NBIO. The register is only meaningful if MCA::NBIO::MCA_STATUS_NBIO[Val]=1 and MCA::NBIO::MCA_STATUS_NBIO[AddrV]=1.	
_instIOHC_n[3:0]_nbio[3:0]_aliasMSRLEGACY; MSR0000_045A	
_instIOHC_n[3:0]_nbio[3:0]_aliasMSR; MSRC000_2162	
Bits	**Description**
63:62	Reserved.
61:56	**LSB**. Read-only. Reset: Cold,00h. Specifies the least significant valid bit of the address contained in MCA::NBIO::MCA_ADDR_NBIO[ErrorAddr]. For example, a value of 0 indicates that MCA::NBIO::MCA_ADDR_NBIO[55:0] contains a valid byte address. A value of 6 indicates that MCA::NBIO::MCA_ADDR_NBIO[55:6] contains a valid cache line address and that MCA::NBIO::MCA_ADDR_NBIO[5:0] are not part of the address and should be ignored by error handling software. A value of 12 indicates that MCA::NBIO::MCA_ADDR_NBIO[55:12] contain a valid 4KB memory page and that MCA::NBIO::MCA_ADDR_NBIO[11:0] should be ignored by error handling software.
55:0	**ErrorAddr**. Read-only. Reset: Cold,00_0000_0000_0000h. Contains the address, if any, associated with the error logged in MCA::NBIO::MCA_STATUS_NBIO.

Table 86: MCA_ADDR_NBIO

Error Type	Bits	Description
EccParityError	[55:0]	Reserved
PCIE_Sideband	[55:0]	Reserved
ErrEvent	[55:0]	Reserved

Egress_Poison	[55:0]	Reserved
IOHC_Internal_Poison	[55:0]	Reserved

MSR0000_045B...MSRC000_2163 [NBIO Machine Check Miscellaneous 0] (MCA::NBIO::MCA_MISC0_NBIO)

Log miscellaneous information associated with errors.

_instIOHC_n[3:0]_nbio[3:0]_aliasMSRLEGACY; MSR0000_045B

_instIOHC_n[3:0]_nbio[3:0]_aliasMSR; MSRC000_2163

Bits	Description
63	**Valid**. Reset: 1. 1=A valid CntP field is present in this register.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read-only.
62	**CntP**. Reset: 1. 1=A valid threshold counter is present.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read-only.
61	**Locked**. 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	**IntP**. 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::NBIO::MCA_MISC0_NBIO[Locked]) ? Read-write : Read-only.
59:56	Reserved.
55:52	**LvtOffset**. 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::NBIO::MCA_MISC0_NBIO[Locked]) ? Read-write : Read-only.
51	**CntEn**. Reset: 0. 1=Count thresholding errors.
	AccessType: (Core::X86::Msr::HWCR[McStatusWrEn] | !MCA::NBIO::MCA_MISC0_NBIO[Locked]) ? Read-write : Read-only.
50:49	**ThresholdIntType**. 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::NBIO::MCA_MISC0_NBIO[Locked]) ? Read-write : Read-only.
48	**Ovrflw**. 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::NBIO::MCA_MISC0_NBIO[Locked]) ? Read-write : Read-only.
47:44	Reserved.
43:32	**ErrCnt**. 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::NBIO::MCA_MISC0_NBIO[Locked]) ? Read-write : Read-only.
31:24	**BlkPtr**. Read-write. Reset: 00h. 00h=Extended MISC MSR block is not valid. 01h=Extended MSR block is valid.
23:0	Reserved.

MSRC000_2164 [NBIO Machine Check Configuration] (MCA::NBIO::MCA_CONFIG_NBIO)

Reset: 0000_0002_0000_0025h.

Controls configuration of the associated machine check bank.

_instIOHC_n[3:0]_nbio[3:0]_aliasMSR; MSRC000_2164

Bits	Description
63:39	Reserved.
38:37	**DeferredIntType**. 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	**LogDeferredInMcaStat**. Read-write. Reset: 0. Init: BIOS,1. 1=Log deferred errors in MCA::NBIO::MCA_STATUS_NBIO and MCA::NBIO::MCA_ADDR_NBIO in addition to MCA::NBIO::MCA_DESTAT_NBIO and MCA::NBIO::MCA_DEADDR_NBIO. 0=Only log deferred errors in MCA::NBIO::MCA_DESTAT_NBIO and MCA::NBIO::MCA_DEADDR_NBIO. This bit does not affect logging of deferred errors in MCA::NBIO::MCA_SYND_NBIO, MCA::NBIO::MCA_MISC0_NBIO.
33	Reserved.
32	**McaXEnable**. 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	**DeferredIntTypeSupported**. Read-only. Reset: 1. 1=MCA::NBIO::MCA_CONFIG_NBIO[DeferredIntType] controls the type of interrupt generated on a deferred error. Deferred errors are supported in this bank only if MCA::NBIO::MCA_CONFIG_NBIO[DeferredErrorLoggingSupported]=1.
4:3	Reserved.
2	**DeferredErrorLoggingSupported**. Read-only. Reset: 1. 1=Deferred errors are supported in this MCA bank, and MCA::NBIO::MCA_CONFIG_NBIO[LogDeferredInMcaStat] controls the logging behavior of these errors. MCA::NBIO::MCA_DESTAT_NBIO and MCA::NBIO::MCA_DEADDR_NBIO are supported in this MCA bank. 0=Deferred errors are not supported in this bank.
1	Reserved.
0	**McaX**. Read-only. Reset: 1. 1=This bank provides Machine Check Architecture Extensions. Up to 4 additional MISC registers (MISC1-MISC4) are supported. MCA::NBIO::MCA_MISC0_NBIO[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::NBIO::MCA_STATUS_NBIO[TCC] is present.

MSRC000_2165 [NBIO IP Identification] (MCA::NBIO::MCA_IPID_NBIO)

Reset: 0000_0018_0000_0000h.

The MCA::NBIO::MCA_IPID_NBIO register is used by software to determine what IP type and revision is associated with the MCA bank.

_instIOHC_n[3:0]_nbio[3:0]_aliasMSR; MSRC000_2165

Bits	Description
63:48	**McaType**. Read-only. Reset: 0000h. The McaType of the MCA bank within this IP.
47:44	Reserved.
43:32	**HardwareID**. Read-only. Reset: 018h. The Hardware ID of the IP associated with this MCA bank.
31:0	**InstanceId**. Read-write. Reset: 0000_0000h. The instance ID of this IP. This is initialized to a unique ID per instance of this register. Init: _instIOHC_n0_nbio0_aliasMSR: 13B1_7000h Init: _instIOHC_n1_nbio1_aliasMSR: 13C1_7000h Init: _instIOHC_n2_nbio2_aliasMSR: 13D1_7000h Init: _instIOHC_n3_nbio3_aliasMSR: 13E1_7000h

MSRC000_2166 [NBIO Machine Check Syndrome] (MCA::NBIO::MCA_SYND_NBIO)

Read-write,Volatile. Reset: Cold,0000_0000_0000_0000h.

Logs physical location information associated with error in MCA::NBIO::MCA_STATUS_NBIO Thread 0

_instIOHC_n[3:0]_nbio[3:0]_aliasMSR; MSRC000_2166

Bits	Description

63:33	Reserved.
32	**Syndrome**. Read-write,Volatile. Reset: Cold,0. Contains the syndrome, if any, associated with the error logged in MCA::NBIO::MCA_STATUS_NBIO. The low-order bit of the syndrome is stored in bit 0, and the syndrome has a length specified by MCA::NBIO::MCA_SYND_NBIO[Length]. The Syndrome field is only valid when MCA::NBIO::MCA_SYND_NBIO[Length] is not 0.
31:27	Reserved.
26:24	**ErrorPriority**. Read-write,Volatile. Reset: Cold,0h. Encodes the priority of the error logged in MCA::NBIO::MCA_SYND_NBIO. 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	**Length**. Read-write,Volatile. Reset: Cold,00h. Specifies the length in bits of the syndrome contained in MCA::NBIO::MCA_SYND_NBIO[Syndrome]. A value of 0 indicates that there is no valid syndrome in MCA::NBIO::MCA_SYND_NBIO. For example, a syndrome length of 9 means that MCA::NBIO::MCA_SYND_NBIO[Syndrome] bits [8:0] contains a valid syndrome.
17:0	**ErrorInformation**. Read-write,Volatile. Reset: Cold,0_0000h. Contains error-specific information about the location of the error. Decoding is available in Table 87 [MCA_SYND_NBIO].

Table 87: MCA_SYND_NBIO

Error Type	Bits	Description
EccParityError	[17:5] [4:0]	Group ID Structure ID
PCIE_Sideband	[5:0]	EgressPortNum
ErrEvent	[3:0]	Reserved
Egress_Poison	[5:0]	Egress Port Number
IOHC_Internal_Poison	[0]	0:CfgMaster 1:TrapClient

MSRC000_2168 [NBIO Machine Check Deferred Error Status] (MCA::NBIO::MCA_DESTAT_NBIO)

Read-write,Volatile. Reset: Cold,0000_0000_0000_0000h.

Holds status information for the first deferred error seen in this bank.

_instIOHC_n[3:0]_nbio[3:0]_aliasMSR; MSRC000_2168

Bits	Description
63	**Val**. Read-write,Volatile. Reset: Cold,0. 1=A valid error has been detected (whether it is enabled or not).
62	**Overflow**. 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	**AddrV**. Read-write,Volatile. Reset: Cold,0. 1=MCA::NBIO::MCA_DEADDR_NBIO contains address information associated with the error.
57:54	Reserved.
53	**SyndV**. Read-write,Volatile. Reset: Cold,0. 1=This error logged information in MCA::NBIO::MCA_SYND_NBIO. If MCA::NBIO::MCA_SYND_NBIO[ErrorPriority] is the same as the priority of the error in MCA::NBIO::MCA_STATUS_NBIO, then the information in MCA::NBIO::MCA_SYND_NBIO is associated with the error in MCA::NBIO::MCA_DESTAT_NBIO.
52:45	Reserved.
44	**Deferred**. 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 [NBIO Deferred Error Address] (MCA::NBIO::MCA_DEADDR_NBIO)

Reset: Cold,0000_0000_0000_0000h.

The MCA::NBIO::MCA_DEADDR_NBIO register stores the address associated with the error in MCA::NBIO::MCA_DESTAT_NBIO. The register is only meaningful if MCA::NBIO::MCA_DESTAT_NBIO[Val]=1 and MCA::NBIO::MCA_DESTAT_NBIO[AddrV]=1. The lowest valid bit of the address is defined by MCA::NBIO::MCA_DEADDR_NBIO[LSB].

_instIOHC_n[3:0]_nbio[3:0]_aliasMSR; MSRC000_2169

Bits	Description
63:62	Reserved.
61:56	**LSB**. Read-write,Volatile. Reset: Cold,00h. Specifies the least significant valid bit of the address contained in MCA::NBIO::MCA_DEADDR_NBIO[ErrorAddr]. For example, a value of 0 indicates that MCA::NBIO::MCA_DEADDR_NBIO[55:0] contains a valid byte address. A value of 6 indicates that MCA::NBIO::MCA_DEADDR_NBIO[55:6] contains a valid cache line address and that MCA::NBIO::MCA_DEADDR_NBIO[5:0] are not part of the address and should be ignored by error handling software. A value of 12 indicates that MCA::NBIO::MCA_DEADDR_NBIO[55:12] contain a valid 4KB memory page and that MCA::NBIO::MCA_DEADDR_NBIO[11:0] should be ignored by error handling software.
55:0	**ErrorAddr**. Read-write,Volatile. Reset: Cold,00_0000_0000_0000h. Contains the address, if any, associated with the error logged in MCA::NBIO::MCA_DESTAT_NBIO. The lowest-order valid bit of the address is specified in MCA::NBIO::MCA_DEADDR_NBIO[LSB].

MSRC001_0416 [NBIO Machine Check Control Mask] (MCA::NBIO::MCA_CTL_MASK_NBIO)

Read-write. Reset: 0000_0000_0000_0000h.

Inhibit detection of an error source.

_instIOHC_n[3:0]_nbio[3:0]_aliasMSR; MSRC001_0416

Bits	Description
63:5	Reserved.
4	**IOHC_Internal_Poison**. Read-write. Reset: 0. Internal Poison Error. Poison data was sent to an internal client.
3	**Egress_Poison**. Read-write. Reset: 0. SDP Egress Poison Error. Poison was propagated to an egress port.
2	**ErrEvent**. Read-write. Reset: 0. SDP ErrEvent error. A system fatal error event from data fabric was detected.
1	**PCIE_Sideband**. Read-write. Reset: 0. PCIE error. A PCIe error was logged in a PCIe root port.
0	**EccParityError**. Read-write. Reset: 0. ECC or Parity error. An SRAM ECC or parity error was detected.

3.2.5.16 PCIE

MSR0000_045C...MSRC000_2170 [PCIE Machine Check Control] (MCA::PCIE::MCA_CTL_PCIE)

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::PCIE::MCA_CTL_PCIE register must be enabled by the corresponding enable bit in Core::X86::Msr::MCG_CTL. Does not affect error detection, correction, or logging.

_instPCIE1PCIE_n[3:0]_nbio[3:0]_aliasMSRLEGACY; MSR0000_045C

_instPCIE1PCIE_n[3:0]_nbio[3:0]_aliasMSR; MSRC000_2170

Bits	Description
63:5	Reserved.
4	**CCIX_WRRSP_DATA_ERR**. Read-write. Reset: 0. CCIX Non-okay write response with data error. Tied off. These should never happen.
3	**CCIX_RDRSP_DATA_ERR**. Read-write. Reset: 0. CCIX Read Response with Status: Data Error.
2	**CCIX_WRRSP_NONDATA_ERR**. Read-write. Reset: 0. CCIX Write Response with Status: Non-Data Error.
1	**CCIX_RDRSP_NONDATA_ERR**. Read-write. Reset: 0. CCIX Read Response with Status: Non-Data Error.
0	**CCIX_PER_MSG_LOG**. Read-write. Reset: 0. CCIX PER Message logging.

MSR0000_045D...MSRC000_2171 [PCIE Machine Check Status] (MCA::PCIE::MCA_STATUS_PCIE)

Reset: Cold,0000_0000_0000_0000h.

Logs information associated with errors.

_instPCIE1PCIE_n[3:0]_nbio[3:0]_aliasMSRLEGACY; MSR0000_045D	
_instPCIE1PCIE_n[3:0]_nbio[3:0]_aliasMSR; MSRC000_2171	
Bits	**Description**
63	**Val**. 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	**Overflow**. 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	**UC**. 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	**En**. Reset: Cold,0. 1=MCA error reporting is enabled for this error, as indicated by the corresponding bit in MCA::PCIE::MCA_CTL_PCIE. This bit is a copy of bit in MCA::PCIE::MCA_CTL_PCIE for this error.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
59	**MiscV**. Reset: Cold,0. 1=Valid thresholding in MCA::PCIE::MCA_MISC0_PCIE. In certain modes, MISC registers are owned by platform firmware and will RAZ when read by non-SMM code. Therefore, it is possible for MiscV=1 and the MISC register to read as all zeros.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
58	**AddrV**. Reset: Cold,0. 1=MCA::PCIE::MCA_ADDR_PCIE contains address information associated with the error.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
57	**PCC**. 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	**ErrCoreIdVal**. 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	**TCC**. Reset: Cold,0. 1=Hardware context of the process thread to which the error was reported may have been corrupted. Continued operation of the thread may have unpredictable results. The thread must be terminated. Only meaningful when MCA::PCIE::MCA_STATUS_PCIE[PCC]=0.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
54	**RESERV4**. Reset: Cold,0. MCA_STATUS Register Reserved bit.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
53	**SyndV**. Reset: Cold,0. 1=This error logged information in MCA::PCIE::MCA_SYND_PCIE. If MCA::PCIE::MCA_SYND_PCIE[ErrorPriority] is the same as the priority of the error in MCA::PCIE::MCA_STATUS_PCIE, then the information in MCA::PCIE::MCA_SYND_PCIE is associated with the error in MCA::PCIE::MCA_STATUS_PCIE.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
52	Reserved.
51:47	**RESERV3**. Reset: Cold,00h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
46	**CECC**. 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	**UECC**. 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	**Deferred**. 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	**Poison**. 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.
42:41	**RESERV2**. Reset: Cold,0h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
40	**Scrub**. Reset: Cold,0. 1=The error was the result of a scrub operation.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
39:38	**RESERV1**. Reset: Cold,0h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
37:32	**ErrCoreId**. 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	**RESERV0**. Reset: Cold,000h. MCA_STATUS Register Reserved bits.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read,Write-0-only,Error-on-write-1.
21:16	**ErrorCodeExt**. 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::PCIE::MCA_CTL_PCIE 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	**ErrorCode**. 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 88: MCA_STATUS_PCIE

Error Type	ErrorCode Ext	UC	PCC	TCC	Deferred	Poison	AddrV
CCIX_PER_MSG_LOG	0x0	0/1	0/1	0/1	0/1	0	PER Message ADRV field
CCIX_RDRSP_NONDATA_ERR	0x1	0/1	0/1	0/1	0/1	0	1
CCIX_WRRSP_NONDATA_ERR	0x2	1	1	1	0	0	1
CCIX_RDRSP_DATA_ERR	0x3	0	0	0	1	0	1
CCIX_WRRSP_DATA_ERR	0x4	1	1	1	0	0	0

MSR0000_045E...MSRC000_2172 [PCIE Machine Check Address] (MCA::PCIE::MCA_ADDR_PCIE)

Read-only. Reset: Cold,0000_0000_0000_0000h.

MCA::PCIE::MCA_ADDR_PCIE stores an address and other information associated with the error in MCA::PCIE::MCA_STATUS_PCIE. The register is only meaningful if MCA::PCIE::MCA_STATUS_PCIE[Val]=1 and MCA::PCIE::MCA_STATUS_PCIE[AddrV]=1.

_instPCIE1PCIE_n[3:0]_nbio[3:0]_aliasMSRLEGACY; MSR0000_045E

_instPCIE1PCIE_n[3:0]_nbio[3:0]_aliasMSR; MSRC000_2172

Bits	Description
63:62	Reserved.

61:56	**LSB**. Read-only. Reset: Cold,00h. Specifies the least significant valid bit of the address contained in MCA::PCIE::MCA_ADDR_PCIE[ErrorAddr]. For example, a value of 0 indicates that MCA::PCIE::MCA_ADDR_PCIE[55:0] contains a valid byte address. A value of 6 indicates that MCA::PCIE::MCA_ADDR_PCIE[55:6] contains a valid cache line address and that MCA::PCIE::MCA_ADDR_PCIE[5:0] are not part of the address and should be ignored by error handling software. A value of 12 indicates that MCA::PCIE::MCA_ADDR_PCIE[55:12] contain a valid 4KB memory page and that MCA::PCIE::MCA_ADDR_PCIE[11:0] should be ignored by error handling software.
55:0	**ErrorAddr**. Read-only. Reset: Cold,00_0000_0000_0000h. Contains the address, if any, associated with the error logged in MCA::PCIE::MCA_STATUS_PCIE.

Table 89: MCA_ADDR_PCIE

Error Type	Bits	Description
CCIX_PER_MSG_LOG	[55:0]	Reserved
CCIX_RDRSP_NONDATA_ERR	[55:0]	Reserved
CCIX_WRRSP_NONDATA_ERR	[55:0]	Reserved
CCIX_RDRSP_DATA_ERR	[55:0]	Reserved
CCIX_WRRSP_DATA_ERR	[55:0]	Reserved

MSR0000_045F...MSRC000_2173 [PCIE Machine Check Miscellaneous 0] (MCA::PCIE::MCA_MISC0_PCIE)

Log miscellaneous information associated with errors.

_instPCIE1PCIE_n[3:0]_nbio[3:0]_aliasMSRLEGACY; MSR0000_045F

_instPCIE1PCIE_n[3:0]_nbio[3:0]_aliasMSR; MSRC000_2173

Bits	Description
63	**Valid**. Reset: 1. 1=A valid CntP field is present in this register.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read-only.
62	**CntP**. Reset: 1. 1=A valid threshold counter is present.
	AccessType: Core::X86::Msr::HWCR[McStatusWrEn] ? Read-write : Read-only.
61	**Locked**. 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	**IntP**. 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::PCIE::MCA_MISC0_PCIE[Locked]) ? Read-write : Read-only.
59:56	Reserved.
55:52	**LvtOffset**. 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::PCIE::MCA_MISC0_PCIE[Locked]) ? Read-write : Read-only.
51	**CntEn**. Reset: 0. 1=Count thresholding errors.
	AccessType: (Core::X86::Msr::HWCR[McStatusWrEn] | !MCA::PCIE::MCA_MISC0_PCIE[Locked]) ? Read-write : Read-only.
50:49	**ThresholdIntType**. 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::PCIE::MCA_MISC0_PCIE[Locked]) ? Read-write : Read-only.
48	**Ovrflw**. 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::PCIE::MCA_MISC0_PCIE[Locked]) ? Read-write : Read-only.
47:44	Reserved.
43:32	**ErrCnt**. 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::PCIE::MCA_MISC0_PCIE[Locked]) ? Read-write : Read-only.
31:24	**BlkPtr**. Read-write. Reset: 00h. 00h=Extended MISC MSR block is not valid. 01h=Extended MSR block is valid.
23:0	Reserved.

MSRC000_2174 [PCIE Machine Check Configuration] (MCA::PCIE::MCA_CONFIG_PCIE)

Reset: 0000_0002_0000_0025h.

Controls configuration of the associated machine check bank.

_instPCIE1PCIE_n[3:0]_nbio[3:0]_aliasMSR; MSRC000_2174

Bits	Description
63:39	Reserved.
38:37	**DeferredIntType**. 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	**LogDeferredInMcaStat**. Read-write. Reset: 0. Init: BIOS,1. 1=Log deferred errors in MCA::PCIE::MCA_STATUS_PCIE and MCA::PCIE::MCA_ADDR_PCIE in addition to MCA::PCIE::MCA_DESTAT_PCIE and MCA::PCIE::MCA_DEADDR_PCIE. 0=Only log deferred errors in MCA::PCIE::MCA_DESTAT_PCIE and MCA::PCIE::MCA_DEADDR_PCIE. This bit does not affect logging of deferred errors in MCA::PCIE::MCA_SYND_PCIE, MCA::PCIE::MCA_MISC0_PCIE.
33	Reserved.
32	**McaXEnable**. 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	**DeferredIntTypeSupported**. Read-only. Reset: 1. 1=MCA::PCIE::MCA_CONFIG_PCIE[DeferredIntType] controls the type of interrupt generated on a deferred error. Deferred errors are supported in this bank only if MCA::PCIE::MCA_CONFIG_PCIE[DeferredErrorLoggingSupported]=1.
4:3	Reserved.
2	**DeferredErrorLoggingSupported**. Read-only. Reset: 1. 1=Deferred errors are supported in this MCA bank, and MCA::PCIE::MCA_CONFIG_PCIE[LogDeferredInMcaStat] controls the logging behavior of these errors. MCA::PCIE::MCA_DESTAT_PCIE and MCA::PCIE::MCA_DEADDR_PCIE are supported in this MCA bank. 0=Deferred errors are not supported in this bank.
1	Reserved.
0	**McaX**. Read-only. Reset: 1. 1=This bank provides Machine Check Architecture Extensions. Up to 4 additional MISC registers (MISC1-MISC4) are supported. MCA::PCIE::MCA_MISC0_PCIE[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::PCIE::MCA_STATUS_PCIE[TCC] is present.

MSRC000_2175 [PCIE IP Identification] (MCA::PCIE::MCA_IPID_PCIE)

Reset: 0000_0046_0000_0000h.

The MCA::PCIE::MCA_IPID_PCIE register is used by software to determine what IP type and revision is associated

with the MCA bank.	
_instPCIE1PCIE_n[3:0]_nbio[3:0]_aliasMSR; MSRC000_2175	
Bits	**Description**
63:48	**McaType**. Read-only. Reset: 0000h. The McaType of the MCA bank within this IP.
47:44	Reserved.
43:32	**HardwareID**. Read-only. Reset: 046h. The Hardware ID of the IP associated with this MCA bank.
31:0	**InstanceId**. Read-write. Reset: 0000_0000h. The instance ID of this IP. This is initialized to a unique ID per instance of this register.
	Init: _instPCIE1PCIE_n0_nbio0_aliasMSR: 115C_0000h Init: _instPCIE1PCIE_n1_nbio1_aliasMSR: 116C_0000h Init: _instPCIE1PCIE_n2_nbio2_aliasMSR: 117C_0000h Init: _instPCIE1PCIE_n3_nbio3_aliasMSR: 118C_0000h

MSRC000_2176 [PCIE Machine Check Syndrome] (MCA::PCIE::MCA_SYND_PCIE)

Read-write,Volatile. Reset: Cold,0000_0000_0000_0000h.	
Logs physical location information associated with error in MCA::PCIE::MCA_STATUS_PCIE Thread 0	
_instPCIE1PCIE_n[3:0]_nbio[3:0]_aliasMSR; MSRC000_2176	
Bits	**Description**
63:32	**Syndrome**. Read-write,Volatile. Reset: Cold,0000_0000h. Contains the syndrome, if any, associated with the error logged in MCA::PCIE::MCA_STATUS_PCIE. The low-order bit of the syndrome is stored in bit 0, and the syndrome has a length specified by MCA::PCIE::MCA_SYND_PCIE[Length]. The Syndrome field is only valid when MCA::PCIE::MCA_SYND_PCIE[Length] is not 0.
31:27	Reserved.
26:24	**ErrorPriority**. Read-write,Volatile. Reset: Cold,0h. Encodes the priority of the error logged in MCA::PCIE::MCA_SYND_PCIE. 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	**Length**. Read-write,Volatile. Reset: Cold,00h. Specifies the length in bits of the syndrome contained in MCA::PCIE::MCA_SYND_PCIE[Syndrome]. A value of 0 indicates that there is no valid syndrome in MCA::PCIE::MCA_SYND_PCIE. For example, a syndrome length of 9 means that MCA::PCIE::MCA_SYND_PCIE[Syndrome] bits [8:0] contains a valid syndrome.
17:0	**ErrorInformation**. Read-write,Volatile. Reset: Cold,0_0000h. Contains error-specific information about the location of the error. Decoding is available in Table 90 [MCA_SYND_PCIE].

Table 90: MCA_SYND_PCIE

Error Type	Bits	Description
CCIX_PER_MSG_LOG	[17:16] [15:8] [7:4] [3:0]	Reserved Source ID Component Type Source Port ID
CCIX_RDRSP_NONDATA_ERR	[17:3] [2:0]	Reserved Port Number
CCIX_WRRSP_NONDATA_ERR	[17:3] [2:0]	Reserved Port Number
CCIX_RDRSP_DATA_ERR	[17:3] [2:0]	Reserved Port Number
CCIX_WRRSP_DATA_ERR	[17:0]	Reserved

MSRC000_2178 [PCIE Machine Check Deferred Error Status] (MCA::PCIE::MCA_DESTAT_PCIE)

Read-write,Volatile. Reset: Cold,0000_0000_0000_0000h.
Holds status information for the first deferred error seen in this bank.
_instPCIE1PCIE_n[3:0]_nbio[3:0]_aliasMSR; MSRC000_2178

Bits	Description
63	**Val**. Read-write,Volatile. Reset: Cold,0. 1=A valid error has been detected (whether it is enabled or not).
62	**Overflow**. 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	**AddrV**. Read-write,Volatile. Reset: Cold,0. 1=MCA::PCIE::MCA_DEADDR_PCIE contains address information associated with the error.
57:54	Reserved.
53	**SyndV**. Read-write,Volatile. Reset: Cold,0. 1=This error logged information in MCA::PCIE::MCA_SYND_PCIE. If MCA::PCIE::MCA_SYND_PCIE[ErrorPriority] is the same as the priority of the error in MCA::PCIE::MCA_STATUS_PCIE, then the information in MCA::PCIE::MCA_SYND_PCIE is associated with the error in MCA::PCIE::MCA_DESTAT_PCIE.
52:45	Reserved.
44	**Deferred**. 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_2179 [PCIE Deferred Error Address] (MCA::PCIE::MCA_DEADDR_PCIE)

Reset: Cold,0000_0000_0000_0000h.

The MCA::PCIE::MCA_DEADDR_PCIE register stores the address associated with the error in MCA::PCIE::MCA_DESTAT_PCIE. The register is only meaningful if MCA::PCIE::MCA_DESTAT_PCIE[Val]=1 and MCA::PCIE::MCA_DESTAT_PCIE[AddrV]=1. The lowest valid bit of the address is defined by MCA::PCIE::MCA_DEADDR_PCIE[LSB].

_instPCIE1PCIE_n[3:0]_nbio[3:0]_aliasMSR; MSRC000_2179

Bits	Description
63:62	Reserved.
61:56	**LSB**. Read-write,Volatile. Reset: Cold,00h. Specifies the least significant valid bit of the address contained in MCA::PCIE::MCA_DEADDR_PCIE[ErrorAddr]. For example, a value of 0 indicates that MCA::PCIE::MCA_DEADDR_PCIE[55:0] contains a valid byte address. A value of 6 indicates that MCA::PCIE::MCA_DEADDR_PCIE[55:6] contains a valid cache line address and that MCA::PCIE::MCA_DEADDR_PCIE[5:0] are not part of the address and should be ignored by error handling software. A value of 12 indicates that MCA::PCIE::MCA_DEADDR_PCIE[55:12] contain a valid 4KB memory page and that MCA::PCIE::MCA_DEADDR_PCIE[11:0] should be ignored by error handling software.
55:0	**ErrorAddr**. Read-write,Volatile. Reset: Cold,00_0000_0000_0000h. Contains the address, if any, associated with the error logged in MCA::PCIE::MCA_DESTAT_PCIE. The lowest-order valid bit of the address is specified in MCA::PCIE::MCA_DEADDR_PCIE[LSB].

MSRC001_0417 [PCIE Machine Check Control Mask] (MCA::PCIE::MCA_CTL_MASK_PCIE)

Read-write. Reset: 0000_0000_0000_0000h.

Inhibit detection of an error source.

_instPCIE1PCIE_n[3:0]_nbio[3:0]_aliasMSR; MSRC001_0417

Bits	Description
63:5	Reserved.
4	**CCIX_WRRSP_DATA_ERR**. Read-write. Reset: 0. CCIX Non-okay write response with data error. Tied off. These should never happen.
3	**CCIX_RDRSP_DATA_ERR**. Read-write. Reset: 0. CCIX Read Response with Status: Data Error.
2	**CCIX_WRRSP_NONDATA_ERR**. Read-write. Reset: 0. CCIX Write Response with Status: Non-Data Error.
1	**CCIX_RDRSP_NONDATA_ERR**. Read-write. Reset: 0. CCIX Read Response with Status: Non-Data Error.
0	**CCIX_PER_MSG_LOG**. Read-write. Reset: 0. CCIX PER Message logging.

4 System Management Network (SMN)

4.1 Definitions

Table 91: Definitions

Term	Description
SMN	System Management Network.

4.2 SMN Network Overview

System Management Network (SMN) is a packetized SOC network with a minimum 8-bit packet width in each direction with either synchronous or source synchronous clocking. It is expected that the packetized width of SMN may be adjusted to provide increased bandwidth over a SMN segment as appropriate to meet SOC and IP requirements. Network and IP interface partitioning is such that the IP interface is immune to changes in packet widths or other characteristics of SMN.

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 SBI Temperature Sensor Interface (SB-TSI). For related specifications, see 1.2 [Reference Documents].

5.1.1 Definitions

Table 92: APML Definitions

Term	Description
ARA	Alert response address.
ARP	Address Resolution Protocol
EC	Embedded Controller.
KBC	Keyboard Controller.
Master or SMBus Master	The device that initiates and terminates all communication and drives the clock, SCL.
PEC	Packet error code.
POR	Power on reset.
RTS	Remote temperature sensor, typical examples are ADM1032, LM99, MAX6657, EMC1002.
SBI	Sideband interface.
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.
TSI	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).

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.

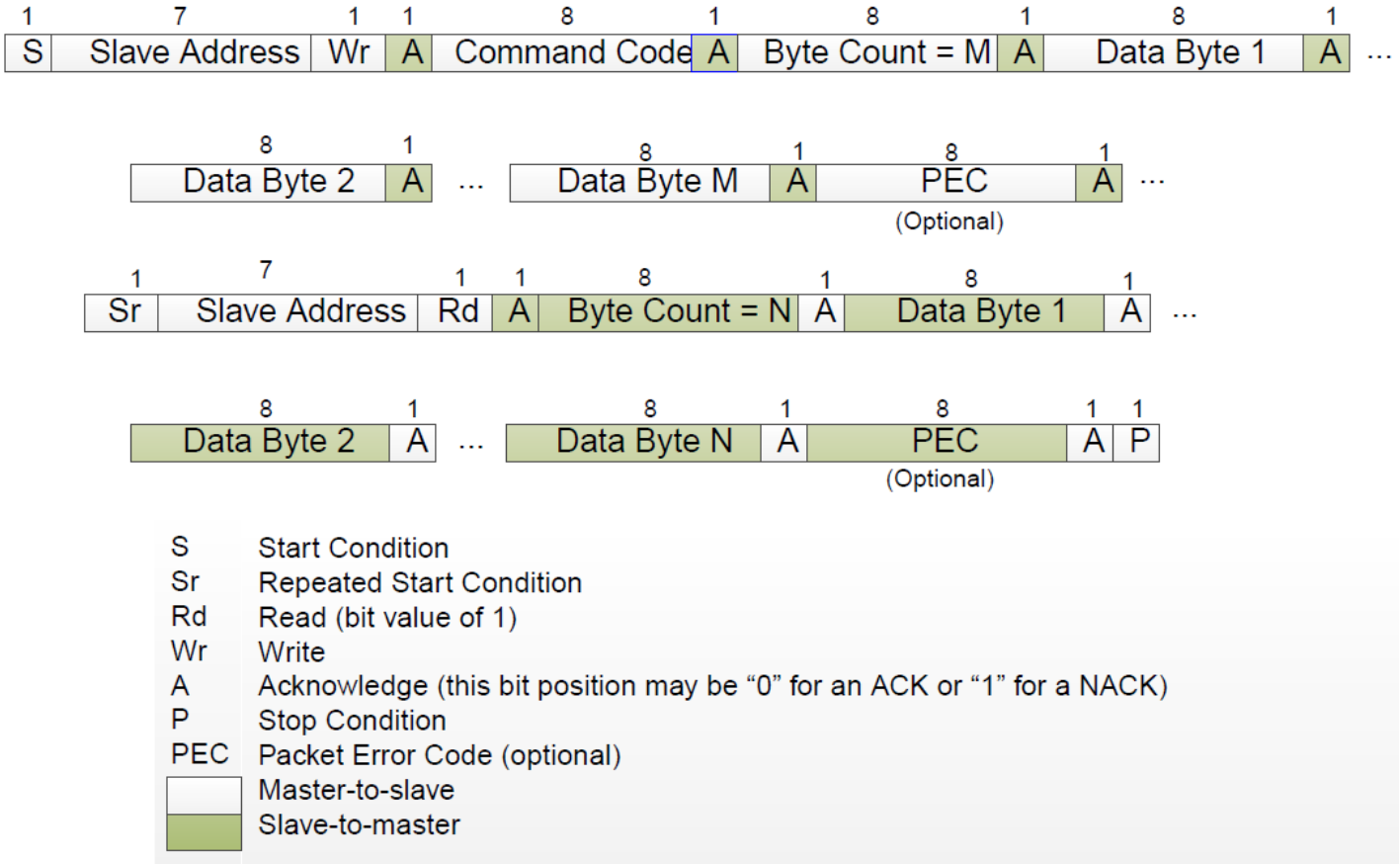

Figure 28: SBI Transmission Protocol

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 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.

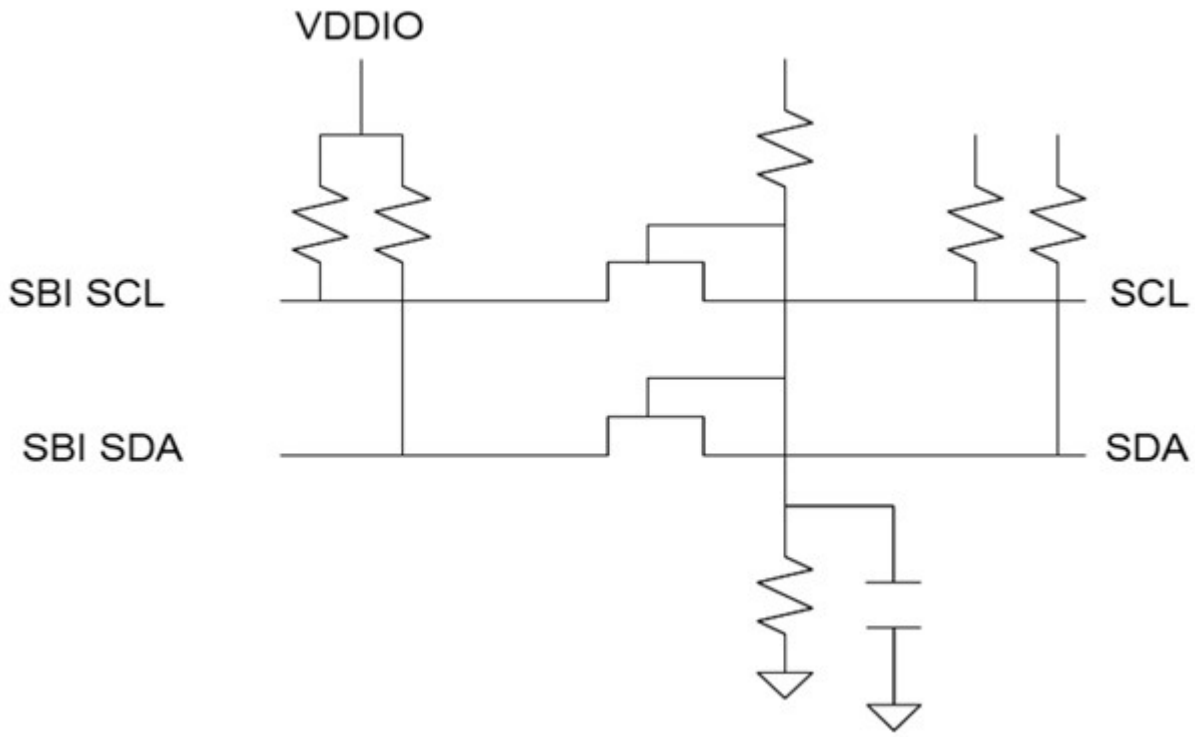

Figure 29: Pass FET Implementation

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 Vgs 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 Vol < Vil 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 30 [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 31 [SB-TSI Thermal Management Example]. SB-TSI supports the SMBus protocols that typical RTS supports.

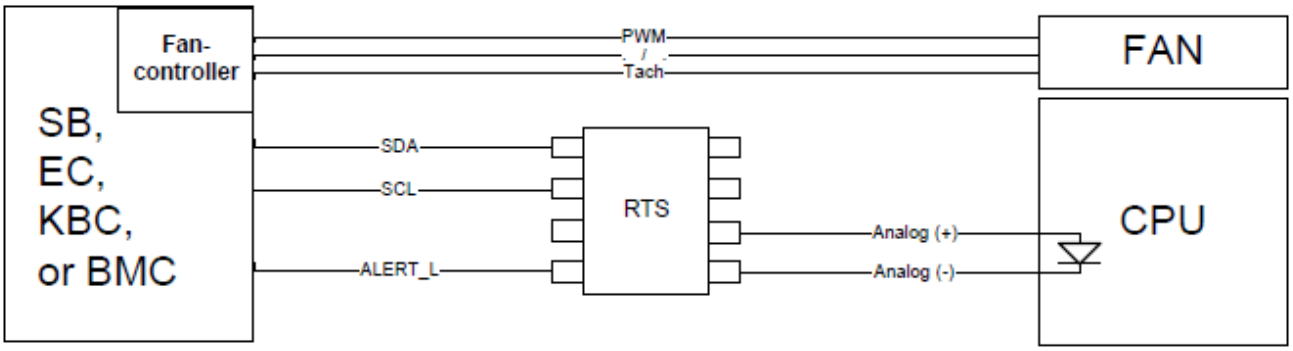

Figure 30: RTS Thermal Management Example

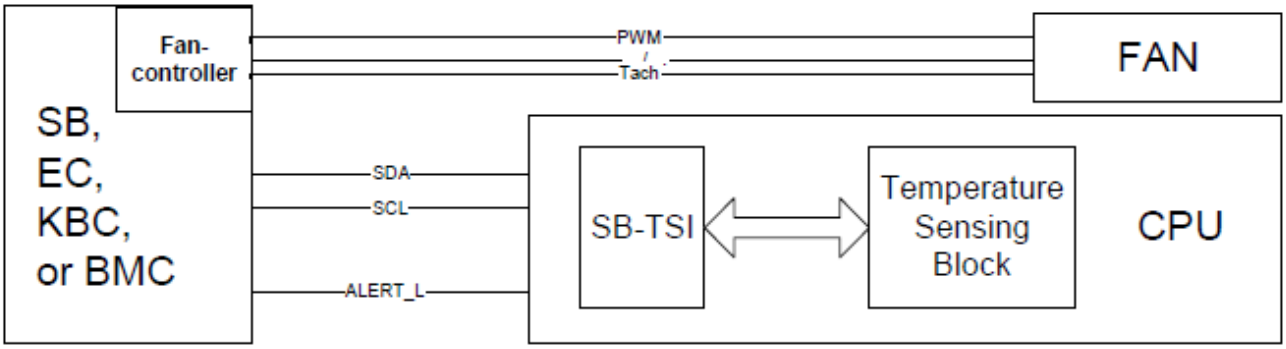

Figure 31: SB-TSI Thermal Management Example

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 93: SB-TSI Definitions

Term	Description
BMC	Base management controller.
TCC	Temperature calculation circuit.
Tctl	Processor temperature control value.

TSM	Temperature sensor macro.
SB-TSI	Sideband Internal Temperature Sensor Interface. See APML.

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 in 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 Ttimeout.max, as specified for SMBus. The time-out needs to be enabled by SBTSI::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 (SBTSI::Status[TempHighAlert] || SBTSI::Status[TempLowAlert]) && ~SBTSI::Config[AlertMask] as shown in Figure 3. The following registers also affect temperature alert behavior.

- SBTSI::Config[AraDis]: Disables ARA response.
- SBTSI::UpdateRate[UpRate]: Specifies rate at which temperature thresholds are checked.
- {SBTSI::HiTempInt[HiTempInt], SBTSI::HiTempDec[HiTempDec]}: Sets high temperature threshold.
- {SBTSI::LoTempInt[LoTempInt], SBTSI::LoTempDec[LoTempDec]}: Sets low temperature threshold.
- SBTSI::AlertThreshold[AlertThr]: Specifies number of consecutive temperature samples to assert an alert.
- SBTSI::AlertConfig[AlertCompEn]: Specifies ALERT_L pin to be in latched or comparator mode. Affects ARA.

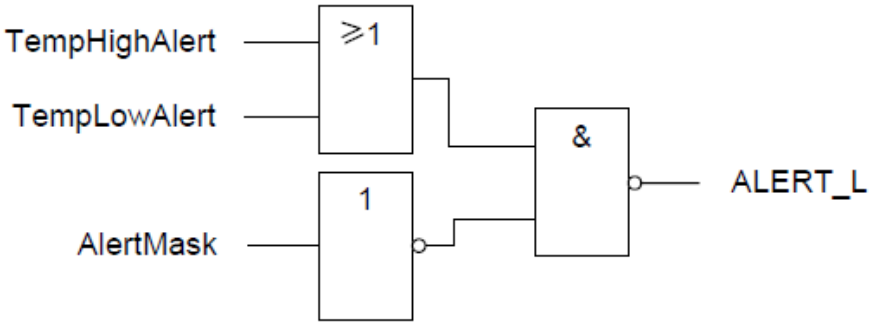

Figure 32: Alert Assertion Diagram

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 94: 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, SBTSI::CpuTempInt and SBTSI::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. SBTSI::CpuTempOffInt[CpuTempOffInt] and SBTSI::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 95: 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 96: 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.

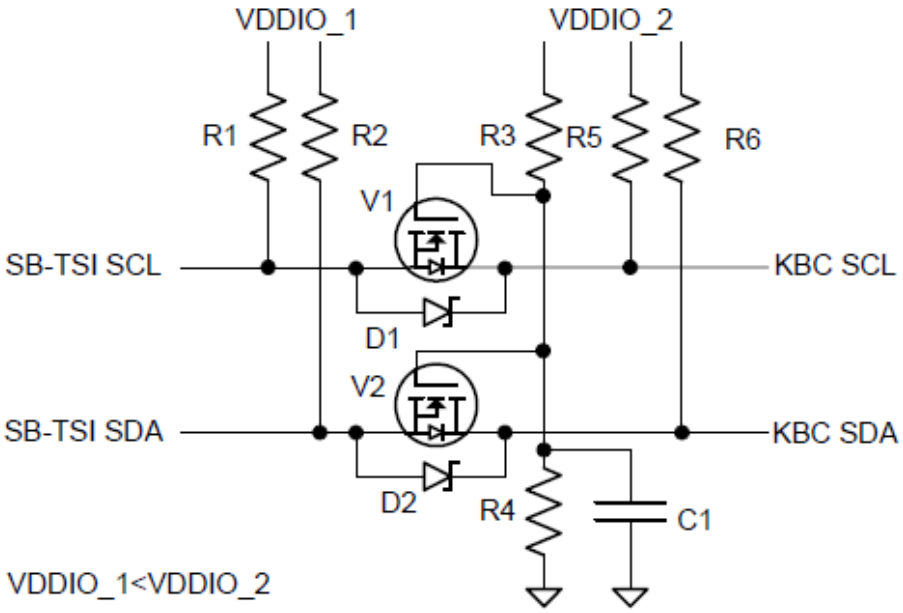

Figure 33: Pass FET Implementation

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 Vgs 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 Vol < Vil 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	**CpuTempInt**: **integer CPU temperature value**. 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	**TempHighAlert**: **temperature high alert**. 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	**TempLowAlert**: **temperature low alert**. 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	**AlertMask**: **alert mask**. 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	**RunStop**: **run stop**. 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	**ReadOrder**: **atomic read order**. 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	**AraDis**: **ARA disable**. 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	**UpRate**: **update rate**. 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]).

ValidValues:

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	**HiTempInt**: **high temperature integer threshold**. 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	**LoTempInt**: **low temperature integer threshold**. 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	**AlertMask**: **alert mask**. Read-write. Reset: Cold,0. See SBTSI::Config[AlertMask].
6	**RunStop**: **run stop**. Read-write. Reset: Cold,0. See SBTSI::Config[RunStop].
5	**ReadOrder**: **atomic read order**. Read-write. Reset: Cold,0. See SBTSI::Config[ReadOrder].
4:2	Reserved.
1	**AraDis**: **ARA disable**. 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	**CpuTempDec**: **decimal CPU temperature value**. 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	**CpuTempOffInt**: **CPU temperature integer offset**. 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	**CpuTempOffDec**: **CPU temperature decimal offset**. 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	**HiTempDec**: **high temperature decimal threshold**. 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	**LoTempDec**: **low temperature decimal threshold**. 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	**TimeoutEn**: **SMBus timeout enable**. 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	**AlertThr**: **alert threshold**. 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.

ValidValues:

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	**AlertCompEn**: **alert comparator mode enable**. 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	**ManId**: **Manufacture ID**. Read-only. Reset: Cold,0. Returns the AMD manufacture ID.

SBTSIxFF [**Revision**] **(SBTSI::Revision)**

Read-only. Reset: Cold,04h.

Bits	Description
7:0	**Revision**: **SB-TSI revision**. Read-only. Reset: Cold,04h. Specifies the SBI temperature sensor interface revision.

7 Host System Management Port (HSMP)

Table 97: Definitions

Term	Description
HSMP	Host System Management Port
SMN	System Management Network

7.1 Overview

The Host System Management Port (HSMP) is an interface to provide OS-level software with access to system management functions via a set of mailbox registers.

7.2 SMN Mailbox Registers

The set of HSMP mailbox registers can be accessed in SMN space via a pair of SMN_INDEX and SMN_DATA registers:

- SMN_INDEX = Table 103 [BXXD00F0x0C4 (IOHC::NB_SMN_INDEX_3)].
- SMN_DATA = Table 104 [BXXD00F0x0C8 (IOHC::NB_SMN_DATA_3)].

To access a specific HSMP mailbox register, software writes the SMN Address of the HSMP mailbox register into the SMN_INDEX register, and then reads from or writes to the SMN_DATA register.

Table 98: SMN Address for HSMP Mailbox Registers

HSMP Mailbox Register	SMN Address (SMN_INDEX)
Message ID	3B10534h
Message Response	3B10980h
Message Argument_0	3B109E0h
Message Argument_1	3B109E4h
Message Argument_2	3B109E8h
Message Argument_3	3B109ECh
Message Argument_4	3B109F0h
Message Argument_5	3B109F4h
Message Argument_6	3B109F8h
Message Argument_7	3B109FCh

7.2.1 Message ID

A 32-bit read-write register to specify the value for the requested system management function. Writes to this register initiate the command message sequence.

7.2.2 Message Arguments

A set of eight 32-bit read-write registers to specify the input data and to capture the output data for each command message. Software writes input values to these register prior to writing to the Message ID register.

7.2.3 Message Response

An 8-bit read-write register that provides the response when the command message has completed. Software writes zero to this register to clear the Response value prior to writing to the Message ID register.

7.3 Mailbox Protocol

To request a given HSMP function, software executes the following sequence:

- Write zero (0) to the Message Response register.
- Write function arguments into the Message Argument registers.
- Write the function ID into the Message ID register.
- Wait (poll) until the Message Response register returns a non-zero value.

7.3.1 Response Codes

Upon completion of the command sequence, if the Message Response contains a RESULT_OK (01h) value, the command completed successfully and results may be read from the Message Argument registers. Otherwise an error occurred and the Message Argument registers are invalid.

Table 99: HSMP Response Codes

Response Value	Description
01h	RESULT_OK. Command completed successfully.
FFh	Command failed – Invalid input Arguments.
FEh	Command failed – Invalid message ID
02h..FDh	Command failed – Other reasons

7.4 HSMP Functions

Table 100 [HSMP Functions] below describes the supported HSMP functions. The APML column denotes if a given HSMP function is also supported by APML. Unless otherwise noted, Input and Output Arguments are specified via the Message Argument_0 (Arg0) register.

Table 100: HSMP Functions

Function ID	Function Name	Description	Input Arguments	Output Arguments
01h	TestMessage	Increments the input argument value by 1. This is used to check if the HSMP interface is functioning correctly.	[31:0] Any value	[ARG0] + 1
02h	GetSmuVersion	Provides the SMU FW version.	None	SMU FW Version
03h	GetInterfaceVersion	Provides the HSMP interface version. Each version supports varying Function IDs as specified in the Table 101 [HSMP Supported Functions Per Interface Version].	None	Interface Version
04h	ReadSocketPower	Provides the average package power consumption. This data may be useful to assess relative variations in power under	None	Socket power (mWatts)

		different operating conditions and workloads. This data should not be used for system power budgeting.		
05h	WriteSocketPowerLimit	Sets the socket power limit. The value written is clipped to the maximum cTDP range for the processor. Note: there is a limit on the minimum power that the processor can operate at; no further socket power reduction occurs if the socket power limit is set below that minimum. NOTE: There there are independent Power Limit registers through HSMP and APML; whichever is the most constraining between the two at any given time is enforced.	[31:0] Socket Power Limit (mWatts)	None
06h	ReadSocketPowerLimit	Provides the current socket power limit.	None	Socket power limit (mWatts)
07h	ReadMaxSocketPowerLimit	Provides the maximum socket power limit. This specifies the clip value for the WriteSocketPowerLimit function.	None	Max Socket power limit (mWatts)
08h	WriteBoostLimit	Sets a Maximum Frequency Limit on a CPU core defined by the specified ApicId. 1. For processors with SMT enabled, writes to different ApicIds that map to the same physical core overwrite the previous write to that core. 2. Values written are constrained to the supported frequency range of the processor. 3. Writes that contain an invalid ApicId result in a Failed response. NOTE: There are independent Boost Limit registers through HSMP and APML; whichever is the most constraining between the two at any given time is enforced.	[31:16] ApicId, 15:0] Max Frequency (MHz)	None
09h	WriteBoostLimitAllCores	Similar to WriteBoostLimit, however the Frequency Limit applies to all cores in the socket.	[15:0] Max Frequency (MHz)	None

		NOTE: There are independent Boost Limit registers through HSMP and APML; whichever is the most constraining between the two at any given time is enforced.		
0Ah	ReadBoostLimit	Provides the frequency limit currently enforced through WriteBoostLimit and WriteBoostLimitAllCores. If no boost limits have been specified, Fmax is returned. Writes that contain an invalid ApicId result in a Failed response. NOTE: There are independent Boost Limit registers through HSMP and APML; this message provides only the boost limit associated with HSMP.	[15:0] ApicId	[15:0] Frequency (MHz)
0Bh	ReadProchotStatus	Provides the current PROCHOT status: 0 = PROCHOT not asserted. 1 = PROCHOT asserted.	None	PROCHOT Status
0Ch	SetXgmiLinkWidthRange	Sets Max and Min width of xGMI Link as follows: 0 = x2 1 = x8 2 = x16	[15:8] MinLinkWidth [7:0] MaxLinkWidth NOTE: Max value must be >= Min value. Invalid configurations result in a Failed response.	None
0Dh	APBDisable	Messages APBEnable and APBDisable specify DF (Data Fabric) P-state behavior. DF P-states specify the frequency of clock domains from the CPU core boundary through to and including system memory, where 0 is the highest DF P-state and 3 is the lowest. By default, an algorithm adjusts DF P-states automatically in order to optimize performance. However, this default may be changed to a fixed DF P-state through a CBS option at boot-time. APBDisable may also be used to disable this algorithm and force a fixed DF P-state.	[7:0] DfPstate (0 to 3)	None

		NOTE: While the socket is in PC6 or if PROCHOT_L is asserted, the lowest DF P-state (highest value) is enforced regardless of the APBEnable/APBDisable state.		
0Eh	APBEnable	This function enables the DF P-state Performance Boost algorithm. See the APBDisable function for more information.	None	None
0Fh	ReadCurrentFclkMemclk	Provides the Data Fabric clock (FCLK) and DRAM memory clock (MEMCLK) frequencies for the current socket DF Pstate.	None	Arg0 = FCLK (MHz), Arg1 = MEMCLK (MHz)
10h	ReadCclkFrequencyLimit	Provides the CPU core clock (CCLK) frequency limit for the socket, from the most restrictive infrastructure limit at the time of the message.	None	Frequency (MHz)
11h	ReadSocketC0Residency	Provides the average C0 residency across all cores in the socket. 100% specifies that all enabled cores in the socket are running in C0.	None	Socket C0 Residency (%)
12h	SetLclkDpmLevelRange	Sets the Max & Min LCLK DPM Level on a given NBIO per socket. The DPM Level is an encoding to represent the PCIe® Link Frequency (LCLK) under a root complex (NBIO), as shown in Table 102 [HSMP LCLK Frequency Per DPM Level].	[24:16] NBIO ID (0 to 3), [15:8] Max DPM Level (0 to 3), [7:0] Min DPM Level (0 to 3). NOTE: Max value must be >= Min value.	None
13h	Reserved	N/A	N/A	N/A
14h	GetMaxDDRBandwidthAndUtilization	Provides: 1. Theoretical maximum DDR Bandwidth of the system in GB/s. 2. Current utilized DDR Bandwidth (Read+Write) in GB/s. 3. Current utilized DDR Bandwidth as a percentage of theoretical maximum.	None	[31:20] Max BW in Gbps [19:8] Utilized (R+W) BW in Gbps [7:0] Utilized BW in %

Table 101: HSMP Supported Functions Per Interface Version

Interface Version	Supported Function IDs
0001h	01h through 11h
0002h	01h through 12h
0003h	01h through 14h

Table 102: HSMP LCLK Frequency Per DPM Level

DPM Level	LCLK Frequency
0	300 MHz
1	400 MHz
2	593 MHz
3	770 MHz

7.4.1 Boost Limit

The WriteBoostLimit function (ID 08h), provides the ability to limit frequency on a per core basis. However, processors are constrained to frequency resolution per the following requirements:

- Cores are grouped into Core Complexes (CCXes), each of which contains 1 or more cores, depending on the OPN.
- Each CCX includes a clock generator that supports a resolution of 25 MHz.
- If all cores of a CCX are not programmed for the same boost limit frequency, then the lower-frequency cores are limited to a frequency resolution that can be as low as 20% of the requested frequency.
- If the specified boost limit frequency of a core is not supported, then the processor selects the next lower supported frequency.

Example 1: If all cores in a CCX are programmed for a boost limit of 3.024 GHz, then all cores of the CCX are limited to 3.000 GHz (next lower supported frequency, with a 25MHz resolution).

Example 2: If three out of four cores of a CCX are programmed for a boost limit of 3.000 GHz, and the last core for 2.350 GHz, then the three high-frequency cores remain set to 3.000 GHz, while the low-frequency core is limited to 2.000 GHz! (~15% lower than requested).

7.4.2 ApicId Mapping

The WriteBoostLimit (ID 08h) and ReadBoostLimit (ID 0Ah) functions are directed at logical cores, as specified by their ApicId. Hence, this section describes how to determine which cores are grouped into the same Core Complex (CCX), and therefore abide by the boost limit frequency resolution rules in section 7.4.1 [Boost Limit] above.

All cores grouped into a common CCX share a Last-Level (L3) cache, hence Core::X86::Cpuid::CachePropEax3[NumSharingCache] can be used to first determine all logical processors (threads) in the cores that share an L3 cache, as follows:

Calculate L3ShareId from the ApicId of each thread as follows:

L3SharedId = ApicId >> log2 (NumSharingCache + 1)

If (NumSharingCache + 1) is not a power of two, round it up to the next power of two. Then, threads with the same L3ShareId share the L3 cache.

Once all threads grouped into a given CCX have been determined, a representative ApicId per core can be derived by filtering out redundant ApicIds when there are multiple threads enabled per core (SMT). The number of threads per core can be determined from Core::X86::Cpuid::CoreId[ThreadsPerCore], as follows:

Calculate CoreSharedId from the ApicId of each thread as follows:

CoreSharedId = ApicId >> log2 (ThreadsPerCore + 1)

If (ThreadsPerCore + 1) is not a power of two, round it up to the next power of two. Then, threads with the same

CoreSharedId belong to the same core.

7.5 Registers

7.5.1 IOHC Registers

Table 103: BXXD00F0x0C4 (IOHC::NB_SMN_INDEX_3)

Read-write. Reset: 0000_0000h.	
_nbio0_aliasHOST; BXXD00F0x0C4; BXX = _nbio0 instance of NB_BUS_NUM from Table 105 [IOHCMISC[0...3]x00000044 (IOHC::NB_BUS_NUM_CNTL)]	
_nbio1_aliasHOST; BXXD00F0x0C4; BXX = _nbio1 instance of NB_BUS_NUM from Table 105 [IOHCMISC[0...3]x00000044 (IOHC::NB_BUS_NUM_CNTL)]	
_nbio2_aliasHOST; BXXD00F0x0C4; BXX = _nbio2 instance of NB_BUS_NUM from Table 105 [IOHCMISC[0...3]x00000044 (IOHC::NB_BUS_NUM_CNTL)]	
_nbio3_aliasHOST; BXXD00F0x0C4; BXX = _nbio3 instance of NB_BUS_NUM from Table 105 [IOHCMISC[0...3]x00000044 (IOHC::NB_BUS_NUM_CNTL)]	
_nbio[3:0]_aliasSMN; NBCFG[3:0]x000000C4; NBCFG[3:0]=13[E:B]0_0000h	
Bits	**Description**
31:0	NB_SMN_INDEX_3. Read-write. Reset: 0000_0000h. Index value SMN Index/Data pair access.

Table 104: BXXD00F0x0C8 (IOHC::NB_SMN_DATA_3)

Read-write. Reset: 0000_0000h.	
_nbio0_aliasHOST; BXXD00F0x0C8; BXX = _nbio0 instance of NB_BUS_NUM from Table 105 [IOHCMISC[0...3]x00000044 (IOHC::NB_BUS_NUM_CNTL)]	
_nbio1_aliasHOST; BXXD00F0x0C8; BXX = _nbio1 instance of NB_BUS_NUM from Table 105 [IOHCMISC[0...3]x00000044 (IOHC::NB_BUS_NUM_CNTL)]	
_nbio2_aliasHOST; BXXD00F0x0C8; BXX = _nbio2 instance of NB_BUS_NUM from Table 105 [IOHCMISC[0...3]x00000044 (IOHC::NB_BUS_NUM_CNTL)]	
_nbio3_aliasHOST; BXXD00F0x0C8; BXX = _nbio3 instance of NB_BUS_NUM from Table 105 [IOHCMISC[0...3]x00000044 (IOHC::NB_BUS_NUM_CNTL)]	
_nbio[3:0]_aliasSMN; NBCFG[3:0]x000000C9; NBCFG[3:0]=13[E:B]0_0000h	
Bits	**Description**
31:0	NB_SMN_DATA_3. Read-write. Reset: 0000_0000h. Index value SMN Index/Data pair access.

Table 105: IOHCMISC[0...3]x00000044 (IOHC::NB_BUS_NUM_CNTL)

Read-write. Reset: 0000_0000h.	
GNB Bus Number Control.	
_nbio[3:0]_aliasSMN; IOHCMISC[3:0]x00000044; IOHCMISC[3:0]=13[E:B]1_0000h	
Bits	**Description**
31:9	NB_SMN_DATA_3. Read-write. Reset: 0000_0000h. Index value SMN Index/Data pair access.
8	NB_BUS_LAT_Mode. Read-write. Reset: 0.
	Description: NBIO bus number is specified by NB_BUS_NUM. 0 = Local bus number of NBIO is capture from any type 0 configuration request. 1= Use the NB_BUS_NUM to decode for configuration cycles targeting the NBIO bus.

7:0	NB_BUS_NUM. Read-write. Reset: 00h. Specifies the number of the NBIO local bus when NB_BUS_LAT_mode is set.

8 Northbridge IO (NBIO)

8.1 IOHC

8.1.1 Definitions

Table 106: Link Definitions

Term	Description
IOHC	IOHUB Core. The I/O crossbar.
MCTP	Management Component Transport Protocol. A manageability protocol that supports communication over a variety of interfaces including PCI Express® and SMBus.
VDM	Vendor Defined Message. A type of PCI Express message that is vendor defined.
P2P	Operations sent directly from one I/O device to another I/O device

8.1.2 Peer-to-Peer Support

The processor supports passing high bandwidth peer-to-peer memory read and write operations between I/O devices. Any PCIe® device may be a source or target of a P2P operation. Additionally, internal peripherals such as the CCP or NTB may also be the source or target of a P2P operation.

To generate a peer-to-peer request, an I/O device generates a DMA read or write operation containing an MMIO address rather than a DRAM address.

8.1.2.1 Peer-to-Peer Synchronization

In order to optimize DMA performance to DRAM, the processor limits the methods peer I/O devices may use to synchronize between each other in order to achieve producer/consumer ordering. These methods are a subset of what is allowed for in the PCIe specification.

8.1.2.1.1 Peer-to-Peer Status Polling

After issuing a set of P2P write operations, a P2P source may ensure that they are flushed to the target by performing a subsequent P2P read operation to the same target. The P2P source may then signal that the peer writes have been completed by setting an internal status register that is polled by the P2P target.

If the P2P source does not have read permissions to access the target, as enforced by the IOMMU, a zero-byte read operation may still be used.

8.1.2.1.2 Writing Data and Flag to a Peer

After issuing a set of P2P write operations, a P2P source may signal the target that the writes have completed by writing a status flag to the target also using a P2P write with RO=0. The target may either locally poll the status flag or the P2P write may trigger hardware that then operates on the received P2P write data.

8.1.2.1.3 Signaling via the CPU

After issuing a set of P2P write operations, a P2P source may signal the target indirectly via the CPU. It may generate an interrupt or write a flag into DRAM that is polled by the CPU. The CPU would then do an MMIO write to the P2P target to indicate that the P2P writes have been delivered from the P2P source.

8.1.2.1.4 Unsupported Peer-to-Peer Synchronization Methods

The processor does not support the below peer-to-peer synchronization methods.
Case 1:
Any synchronization scheme that relies on completions pushing posted writes in the downstream direction away from the processor is not supported.
In the following example, the read response of the status register in step 3 may pass the P2P writes in step 1.

1. Device A does P2P writes to device B.
2. Device A sets an internal status register to indicate the P2P writes are complete.
3. In parallel, device B polls the status register in device A.

In order to make the case pass, device A may use a flushing read to device B before setting the internal status register in step 2.

Case 2:
Any synchronization scheme that relies on ordering between the writes of one device and the reads and writes of another device is not supported, regardless of whether or not the operations target DRAM or a 3rd I/O device.
In the following example, it is possible that the DMA operations from device B in step 3 may pass the DMA writes in step 1.

1. Device A does DMA writes to DRAM.
2. Device A does a P2P write to device B to indicate that data has been written to DRAM.
3. Once device B observes the write in step 2, device B does DMA reads or writes to the same DRAM locations accessed by device A and requires that these operations will be ordered after the writes in step 1.

In order to make this case pass, device A should perform a flushing read to DRAM before issuing the P2P write to device B in step 2.

Similarly, for the following example it is possible that the P2P operations in step 3 may reach device C before the P2P writes in step 1.

1. Device A does P2P writes to device C.
2. Device A does a P2P write to device B to indicate that data has been written to device C.
3. Once device B observes the write in step 2, device B does P2P reads or writes to device C to operate on the data sent by device A or otherwise assumes that these operations will be ordered after the writes in step 1.

In order to make this case pass, device A should perform a flushing read to device C before issuing the P2P write to device B in step 2.

8.1.2.2 Peer-to-Peer Interaction with PCIe® ACS

The processor PCIe root ports treat a received completion that does not match a corresponding non-posted request previously issued by the same root port as an unexpected completion. Configurations that generate non-matching completions to the root ports are not supported. Non-matching completions may be sent to a root port during specific combinations of peer-to-peer traffic in conjunction with an external PCIe switch and specific settings of the PCIe ACS

features in the switch. In these cases, completions to peer-to-peer requests do not follow the reverse of the request path back to the requester through the PCIe topology.

In general, if IOMMU is enabled, software should enable ACS Request Redirection, ACS Upstream Forwarding, and ACS Completion Redirection in root ports, external switches and multi-function endpoints to ensure that DMA requests are forced up towards the IOMMU for translation. ACS Direct Translated P2P should be disabled in switches to avoid triggering unexpected completions at the processor root ports unless one of the enabling conditions described below is met.

At least one of the following sets of requirements must be met before ACS Direct Translated P2P may be enabled in an external PCIe switch downstream port or in a multi-function endpoint:

- All of the following are true:
    1. At least one of the functions located behind the switch downstream port or in the multi-function endpoint generates translated peer-to-peer requests.
    2. None of the endpoints located behind the switch downstream port or in the multi-function endpoint generate translated peer-to-peer reads or atomics.
- Or all of the following are true:
    1. At least one of the functions below the switch downstream port or in the multi-function endpoint is capable of generating translated peer-to-peer requests.
    2. The peer-to-peer traffic in (1) is capable of targeting one or more functions behind one of the downstream ports of the same switch or in the same multi-function endpoint.
    3. All of the targets in (2) are not also targets of untranslated peer-to-peer traffic whose source is located behind one of downstream ports of the same switch or in the same multi-function endpoint. PCIe completion redirection must be disabled in the switch downstream ports or the multi-function endpoint in (3).

Similarly, the processor root ports will trigger a timeout event if the root port issues downstream non-posted request without the corresponding completion returning to the same root port. This can occur whenever PCIe ACS features are enabled such that a function located behind a switch generates a P2P read or atomic targeting an endpoint located behind the same switch, where the request is routed through the PCIe root port, but the completion is directly routed back to the originator without going back through the PCIe root port.

8.2 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.

8.2.1 Functional Description

8.2.1.1 Definitions

Table 107: Link Definitions

Term	Description
IOMMU	IO Memory Management Unit

9 DXIO

9.1 DXIO Registers

9.1.1 PCS_DXIO Registers

XGMIPCS[0...5]x00010050 (PCS::DXIO::PCS_GOPX16_PCS_STATUS1)

Read-only.

Status registers R

_instSERDESAG0_pcs20_aliasSMN; XGMIPCS0x00010050; XGMIPCS0=12EE0000h
_instSERDESAG1_pcs21_aliasSMN; XGMIPCS1x00010050; XGMIPCS1=12FE0000h
_instSERDESAG2_pcs22_aliasSMN; XGMIPCS2x00010050; XGMIPCS2=130E0000h
_instSERDESAG3_pcs23_aliasSMN; XGMIPCS3x00010050; XGMIPCS3=131E0000h
_instSERDESAP3_pcs24_aliasSMN; XGMIPCS4x00010050; XGMIPCS4=132E0000h
_instSERDESAP2_pcs25_aliasSMN; XGMIPCS5x00010050; XGMIPCS5=133E0000h

Bits	Description
31:22	Reserved.
21	**LinkWidth_x16Status**. Read-only. Link Width x16 Status
20:19	Reserved.
18	**LinkWidth_x8Status**. Read-only. Link Width x8 Status
17:0	Reserved.

XGMIPCS[0...5]x00010114 (PCS::DXIO::PCS_GOPX16_PCS_PSTATE_CONTEXT5)

Read-only. Reset: 0000_0000h.

_instSERDESAG0_pcs20_aliasSMN; XGMIPCS0x00010114; XGMIPCS0=12EE0000h
_instSERDESAG1_pcs21_aliasSMN; XGMIPCS1x00010114; XGMIPCS1=12FE0000h
_instSERDESAG2_pcs22_aliasSMN; XGMIPCS2x00010114; XGMIPCS2=130E0000h
_instSERDESAG3_pcs23_aliasSMN; XGMIPCS3x00010114; XGMIPCS3=131E0000h
_instSERDESAP3_pcs24_aliasSMN; XGMIPCS4x00010114; XGMIPCS4=132E0000h
_instSERDESAP2_pcs25_aliasSMN; XGMIPCS5x00010114; XGMIPCS5=133E0000h

Bits	Description
31:11	Reserved.
10:4	**FrequencyCount**. Read-only. Reset: 00h.
3:0	Reserved.

10 Miscellaneous Information

10.1 RMTPLLCNTL0 Register

MISC2x30[7:0] is useful to determine the current SOC reference clock, so adding it here for public reference.

Table 108: RMT PLLCNTL0 REG

RMT_PLLCNTL_0_REG - RW - 32 bits			
Field Name	Bits	Default	Description
reg_clkb_phy_4_refclksel_override	7:6	00b	PHY_4 Refclk Selection Override (CLKGEN_TOP, G-PHY) CLKB_PCIE_PHY_G_3_P/N (fch_tile/CLKGEN) 00 = 100MHz CG1_PLL generated 01 = 133MHz CG2_PLL generated 10 = EXT_GPP0_SRC 11 = 100MHz CG2_PLL generated (default)
reg_clkb_phy_3_refclksel_override	5:4	00b	PHY_3 Refclk Selection Override (CLKGEN_TOP, G-PHY) CLKB_PCIE_PHY_G_2_P/N (fch_tile/CLKGEN) 00 = 100MHz CG1_PLL generated 01 = 133MHz CG2_PLL generated 10 = EXT_GPP0_SRC 11 = 100MHz CG2_PLL generated (default)
reg_clkb_phy_2_refclksel_override	3:2	00b	PHY_2 Refclk Selection Override (CLKGEN_TOP, G-PHY) CLKB_PCIE_PHY_1_P/N (fch_tile/CLKGEN) 00 = 100MHz CG1_PLL generated 01 = 133MHz CG2_PLL generated 10 = EXT_GPP0_SRC 11 = 100MHz CG2_PLL generated (default)
reg_clkb_phy_1_refclksel_override	1:0	00b	PHY_1_Refclk Selection Override (CLKGEN_TOP, G-PHY) CLKB_PCIE_PHY_G0_P/N (fch_tile/CLKGEN) 00 = 100MHz CG1_PLL generated 01 = 133MHz CG2_PLL generated 10 = EXT_GPP0_SRC from CLKGEN_BOT 11 = 100MHz CG2_PLL generated (default)

10.2 SMU::SMUIO::SMUSVI0_TEL_PLANE0 and SMU::SMUIO::SMUSVI1_TEL_PLANE0 Registers

Table 109: SMU::SMUIO::SMUSVI0_TEL_PLANE0 REG

SMUIOx00000010 (SMU::SMUIO::SMUSVI0_TEL_PLANE0)	
Read-only. Reset: 0000_0000h.	
_aliasSMN; SMUIOx00000010; SMUIO=0005_A000h	
31:25	Reserved.
24:16	SVI0_PLANE0_VDDCOR. Read-only. Reset: 000h. Read only. VDD by Telemetry for PLANE0 in VR0

15:8	Reserved.
7:0	SVI0_PLANE0_IDDCOR. Read-only. Reset: 000h. Read only. IDD by Telemetry for PLANE0 in VR0

Table 110: SMU::SMUIO::SMUSVI1_TEL_PLANE0 REG

SMUIOx00000010 (SMU::SMUIO::SMUSVI1_TEL_PLANE0)	
Read-only. Reset: 0000_0000h.	
_aliasSMN; SMUIOx00000014; SMUIO=0005_A000h	
31:25	Reserved.
24:16	SVI1_PLANE0_VDDCOR. Read-only. Reset: 000h. Read only. VDD by Telemetry for PLANE0 in VR1
15:8	Reserved.
7:0	SVI1_PLANE0_IDDCOR. Read-only. Reset: 000h. Read only. IDD by Telemetry for PLANE0 in VR1

10.3 SMU::THM::THM_TCON_CUR_TMP Register

Table 111: SMU::THM::THM_TCON_CUR_TMP REG

SMUTHMx00000000 (SMU::THM::THM_TCON_CUR_TMP)	
Reset: 0000_0000h.	
Provides the current control temperature (T ctl) after the slew-rate controls have been applied.	
_aliasSMN; SMUTHMx00000000; SMUTHM=0005_9800h	
Bits	Description
31:21	CUR_TEMP. Reset: 000h. Provides the current control temperature
	AccessType: Read-only.

List of Namespaces

Namespace	Heading(s)
Core::X86::Apic	2.1.14.2.2 [Local APIC Registers]
Core::X86::Cpuid	2.1.15.1 [CPUID Instruction Functions]
Core::X86::Msr	2.1.16.1 [MSRs - MSR0000_xxxx] 2.1.16.2 [MSRs - MSRC000_xxxx] 2.1.16.3 [MSRs - MSRC001_0xxx] 2.1.16.4 [MSRs - MSRC001_1xxx]
Core::X86::Pmc::Core	2.1.17.3 [Large Increment per Cycle Events] 2.1.17.4.1 [Floating Point (FP) Events] 2.1.17.4.2 [LS Events] 2.1.17.4.3 [IC and BP Events] 2.1.17.4.4 [DE Events] 2.1.17.4.5 [EX (SC) Events] 2.1.17.4.6 [L2 Cache Events]
Core::X86::Pmc::L3	2.1.17.5.1 [L3 Cache PMC Events]
Core::X86::Smm	2.1.14.1.6 [System Management State]
IO	2.1.9 [PCI Configuration Legacy Access]
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::MP5	3.2.5.14 [MP5]
MCA::NBIO	3.2.5.15 [NBIO]
MCA::PB	3.2.5.11 [PB]
MCA::PCIE	3.2.5.16 [PCIE]
MCA::PIE	3.2.5.9 [PIE]
MCA::PSP	3.2.5.12 [PSP]
MCA::SMU	3.2.5.13 [SMU]
MCA::UMC	3.2.5.10 [UMC]
PCS::DXIO	9.1.1 [PCS_DXIO Registers]
SBTSI	6.4 [SB-TSI Registers]

List of Definitions

ABS: ABS(integer expression): Remove sign from signed value.
AGESA: AMD Generic Encapsulated Software Architecture.
APML: Advanced Platform Management Link.
ARA: Alert response address.
ARP: Address Resolution Protocol
BAR: The BAR, or base address register, physical register mnemonic format is of the form PREFIXxZZZ.
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].
CCD: Core-Complex Die.
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::Msr::PStateDef[CpuFid[7:0]]/Core::X86::Msr::PStateDef[CpuDfsId])*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.
docAPM1: AMD64 Architecture Programmer's Manual Volume 1: Application Programming, Publication No. 24592.
docAPM2: AMD64 Architecture Programmer's Manual Volume 2: System Programming, Publication No. 24593.
docAPM3: AMD64 Architecture Programmer's Manual Volume 3: Instruction-Set Reference, Publication No. 24594.
docAPM4: AMD64 Architecture Programmer's Manual Volume 4: 128-Bit and 256-Bit Media Instructions, Publication No. 26568.
docAPM5: AMD64 Architecture Programmer's Manual Volume 5: 64-Bit Media and x87 Floating-Point Instructions, Publication No. 26569.
docJEDEC: JEDEC Standards. http://www.jedec.org.
docMBDG: Socket SP3 Processor Motherboard Design Guide (MBDG), Publication No. 55414.
docPCIe: PCI Express® Specification. http://www.pcisig.com.
docPCIlb: PCI Local Bus Specification. http://www.pcisig.com.
docRevG: Revision Guide for AMD Family 19h Models 00h-0Fh Processors, Publication No. 56683.
docSMB: System Management Bus (SMBus) Specification. http://www.smbus.org.
docSSP3: Socket SP3 Functional Data Sheet (FDS), Publication No. 55426.
Doubleword: A 32-bit value.
DW: Doubleword.
EC: Embedded Controller.
ECS: Extended Configuration Space.
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.
FID: Frequency Identifier. Specifies the PLL frequency multiplier for a given clock domain.
FLOOR: FLOOR(integer expression): Rounds real number down to nearest integer.
GB: Gbyte or Gigabyte; 1,073,741,824 bytes.
GT/s: Giga-Transfers per second.
HSMP: Host System Management Port
IBS: Instruction based sampling.
IFCM: Isochronous flow-control mode, as defined in the link specification.
Inaccessible: Not readable or writable (e.g., Hide ? Inaccessible : Read-Write).
IO configuration: Access to configuration space though IO ports CF8h and CFCh.
IOD: IO die.
IOHC: IOHUB Core. The I/O crossbar.
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].
LRU: Least recently used.
LVT: Local vector table. A collection of APIC registers that define interrupts for local events (e.g., APIC[530:500] [Extended Interrupt [3:0] Local Vector Table]).
Macro-op: The front-end of the pipeline breaks instructions into macro-ops and transfers (dispatches) them to the back-end of the pipeline for scheduling and execution. See Software Optimization Guide.
Master abort: This is a PCI-defined term that is applied to transactions on other than PCI buses. It indicates that the transaction is terminated without affecting the intended target; reads return all 1s; write are discarded; the master abort error code is returned in the response, if applicable; master abort error bits are set if applicable.
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.
MCTP: Management Component Transport Protocol. A manageability protocol that supports communication over a variety of interfaces including PCI Express and SMBus.
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 = (CPUID Fn00000001_EBX[LocalApicId[3:0]]==0). Node Base Core. The lowest numbered core in the node.
NTA: Non-Temporal Access.
OW: Octword. An 128-bit value.
P2P: Operations sent directly from one I/O device to another I/O device
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: Die of the System on Chip (SoC) covered by this PPR. See XX [Processor Overview].
PTE: Page table entry.
QW: Quadword. A 64-bit value.
RAS: Reliability, availability and serviceability (industry term). See XX [Machine Check Architecture].
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 cannotes 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-TSI: Sideband Internal Temperature Sensor Interface. See APML.
SBI: Sideband 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.
SMAF: 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.
SMN: System Management Network
SMT: Simultaneous multithreading. See Core::X86::Cpuid::CoreId[ThreadsPerCore].
Speculative event: A performance monitor event counter that counts all occurrences of the event even if the event occurs during speculative code execution.
SVM: Secure virtual machine.
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 Northbridge 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.
VDM: Vendor Defined Message. A type of PCI Express message that is vendor defined.
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.
X2APICEN: x2 APIC is enabled. X2APICEN = (Core::X86::Msr::APIC_BAR[ApicEn] && Core::X86::Msr::APIC_BAR[x2ApicEn]).
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_043Bh	MCA::L3
0000_043Ch...0000_043Fh	MCA::MP5
0000_0440h...0000_0443h	MCA::PB
0000_0444h...0000_044Bh	MCA::UMC
0000_044Ch...0000_0457h	MCA::CS
0000_0458h...0000_045Bh	MCA::NBIO
0000_045Ch...0000_045Fh	MCA::PCIE
0000_0460h...0000_0463h	MCA::SMU
0000_0464h...0000_0467h	MCA::PSP
0000_0468h...0000_046Bh	MCA::PB
0000_046Ch...0000_046Fh	MCA::PIE
0000_06A0h...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_20E9h	MCA::L3
C000_20F0h...C000_20F6h	MCA::MP5
C000_2100h...C000_2106h	MCA::PB
C000_2110h...C000_212Ah	MCA::UMC
C000_2130h...C000_2159h	MCA::CS
C000_2160h...C000_2169h	MCA::NBIO
C000_2170h...C000_2179h	MCA::PCIE
C000_2180h...C000_2186h	MCA::SMU
C000_2190h...C000_2196h	MCA::PSP
C000_21A0h...C000_21A6h	MCA::PB
C000_21B0h...C000_21B9h	MCA::PIE
C001_0000h...C001_031Fh	Core::X86::Msr
C0010400	MCA::LS
C0010401	MCA::IF
C0010402	MCA::L2
C0010403	MCA::DE
C0010405	MCA::EX
C0010406	MCA::FP
C001_0407h...C001_040Eh	MCA::L3

C001040F	MCA::MP5
C0010410	MCA::PB
C001_0411h...C001_0412h	MCA::UMC
C001_0413h...C001_0415h	MCA::CS
C0010416	MCA::NBIO
C0010417	MCA::PCIE
C0010418	MCA::SMU
C0010419	MCA::PSP
C001041A	MCA::PB
C001041B	MCA::PIE
C001_1002h...C001_103Ch	Core::X86::Msr

Memory Map - SMN

Physical Mnemonic	Namespace
12EE0000: XGMIPCS0x00010050...x00010114	PCS::DXIO
12FE0000: XGMIPCS1x00010050...x00010114	PCS::DXIO
130E0000: XGMIPCS2x00010050...x00010114	PCS::DXIO
131E0000: XGMIPCS3x00010050...x00010114	PCS::DXIO
132E0000: XGMIPCS4x00010050...x00010114	PCS::DXIO
133E0000: XGMIPCS5x00010050...x00010114	PCS::DXIO