

# Intel<sup>®</sup> Core<sup>™</sup> i7 Processor Family for the LGA-2011 Socket

## Specification Update

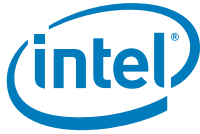
---

*Supporting Desktop Intel<sup>®</sup> Core<sup>™</sup> i7-3960X Extreme Edition Processor for the LGA-2011 Socket*

*Supporting Desktop Intel<sup>®</sup> Core<sup>™</sup> i7-39xxK and i7-38xx Processor Series for the LGA-2011 Socket*

*January 2012*

**Notice:** The products described in this document may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.



INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

A "Mission Critical Application" is any application in which failure of the Intel Product could result, directly or indirectly, in personal injury or death. SHOULD YOU PURCHASE OR USE INTEL'S PRODUCTS FOR ANY SUCH MISSION CRITICAL APPLICATION, YOU SHALL INDEMNIFY AND HOLD INTEL AND ITS SUBSIDIARIES, SUBCONTRACTORS AND AFFILIATES, AND THE DIRECTORS, OFFICERS, AND EMPLOYEES OF EACH, HARMLESS AGAINST ALL CLAIMS COSTS, DAMAGES, AND EXPENSES AND REASONABLE ATTORNEYS' FEES ARISING OUT OF, DIRECTLY OR INDIRECTLY, ANY CLAIM OF PRODUCT LIABILITY, PERSONAL INJURY, OR DEATH ARISING IN ANY WAY OUT OF SUCH MISSION CRITICAL APPLICATION, WHETHER OR NOT INTEL OR ITS SUBCONTRACTOR WAS NEGLIGENT IN THE DESIGN, MANUFACTURE, OR WARNING OF THE INTEL PRODUCT OR ANY OF ITS PARTS.

Intel may make changes to specifications and product descriptions at any time, without notice. Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined". Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. The information here is subject to change without notice. Do not finalize a design with this information.

The products described in this document may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Contact your local Intel sales office or your distributor to obtain the latest specifications and before placing your product order.

Copies of documents which have an order number and are referenced in this document, or other Intel literature, may be obtained by calling 1-800-548-4725, or go to: <http://www.intel.com/design/literature.htm>.

Code names featured are used internally within Intel to identify products that are in development and not yet publicly announced for release. Customers, licensees and other third parties are not authorized by Intel to use code names in advertising, promotion or marketing of any product or services and any such use of Intel's internal code names is at the sole risk of the user.

Intel processor numbers are not a measure of performance. Processor numbers differentiate features within each processor family, not across different processor families. Go to: [http://www.intel.com/products/processor\\_number](http://www.intel.com/products/processor_number).

Intel® Hyper-Threading Technology requires an Intel® HT Technology enabled system, check with your PC manufacturer. Performance will vary depending on the specific hardware and software used. Not available on Intel® Core™ i5-750. For more information including details on which processors support HT Technology, visit <http://www.intel.com/info/hyperthreading>.

Intel® Turbo Boost Technology requires a system with Intel® Turbo Boost Technology. Intel Turbo Boost Technology and Intel Turbo Boost Technology 2.0 are only available on select Intel® processors. Consult your PC manufacturer. Performance varies depending on hardware, software, and system configuration. For more information, visit: <http://www.intel.com/go/turbo>.

Intel® 64 architecture requires a system with a 64-bit enabled processor, chipset, BIOS and software. Performance will vary depending on the specific hardware and software you use. Consult your PC manufacturer for more information. For more information, visit: <http://www.intel.com/info/em64t>.

No computer system can provide absolute security under all conditions. Intel® Trusted Execution Technology (Intel® TXT) requires a computer system with Intel® Virtualization Technology, an Intel TXT-enabled processor, chipset, BIOS, Authenticated Code Modules and an Intel TXT-compatible measured launched environment (MLE). Intel TXT also requires the system to contain a TPM v1.s. For more information, visit <http://www.intel.com/technology/security>.

Intel® Virtualization Technology requires a computer system with an enabled Intel® processor, BIOS, virtual machine monitor (VMM). Functionality, performance or other benefits will vary depending on hardware and software configurations. Software applications may not be compatible with all operating systems. Consult your PC manufacturer. For more information, visit: <http://www.intel.com/go/virtualization>.

Intel and the Intel logo are trademarks of Intel Corporation in the U.S. and other countries.

\*Other names and brands may be claimed as the property of others.

Copyright © 2012, Intel Corporation. All rights reserved.



# Contents

---

<b>Revision History</b> .....	4
<b>Preface</b> .....	5
<b>Identification Information</b> .....	7
<b>Summary Table of Changes</b> .....	9
<b>Errata</b> .....	14
<b>Specification Changes</b> .....	41
<b>Specification Clarifications</b> .....	42
<b>Documentation Changes</b> .....	43

§ §



# Revision History

---

Revision	Description	Date
001	• Initial Release	November 2011
002	• Added Errata - <a href="#">BS92.</a> , <a href="#">BS93.</a> , <a href="#">BS94.</a>	January 2012



# Preface

This document is an update to the specifications contained in the [Affected Documents](#) table below. This document is a compilation of device and documentation sighting, specification clarifications and changes. It is intended for hardware system manufacturers and software developers of applications, operating systems, or tools.

Information types defined in [Nomenclature](#) are consolidated into the specification update and are no longer published in other documents.

This document may also contain information that was not previously published.

## Affected Documents

Document Title	Document Number/ Location
<i>Intel® Core™ i7 Processor Family for the LGA-2011 Socket Datasheet - Volume 1</i>	326196-001
<i>Intel® Core™ i7 Processor Family for the LGA-2011 Socket Datasheet - Volume 2</i>	326197-001

## Related Documents

Document Title	Document Number/Location
<i>AP-485, Intel® Processor Identification and the CPUID Instruction</i>	<a href="http://www.intel.com/Assets/en_US/PDF/appnote/241618.pdf">http://www.intel.com/Assets/en_US/PDF/appnote/241618.pdf</a>
<i>Intel® 64 and IA-32 Architecture Software Developer's Manual</i> <ul style="list-style-type: none"><li>• <i>Volume 1: Basic Architecture</i></li><li>• <i>Volume 2A: Instruction Set Reference Manual A-M</i></li><li>• <i>Volume 2B: Instruction Set Reference Manual N-Z</i></li><li>• <i>Volume 3A: System Programming Guide</i></li><li>• <i>Volume 3B: System Programming Guide</i></li><li>• <i>IA-32 Intel® Architecture Optimization Reference Manual</i></li></ul>	<a href="http://www.intel.com/products/processor/manuals/index.htm">http://www.intel.com/products/processor/manuals/index.htm</a>
<i>Intel® 64 and IA-32 Architectures Software Developer's Manual Documentation Changes</i>	<a href="http://www.intel.com/design/processor/specupdt/252046.htm">http://www.intel.com/design/processor/specupdt/252046.htm</a>
<i>Intel® 64 and IA-32 Architectures Optimization Reference Manual</i>	<a href="http://www.intel.com/Assets/PDF/manual/248966.htm">http://www.intel.com/Assets/PDF/manual/248966.htm</a>

**Notes:**

1. Documentation changes for the Intel® 64 and IA-32 Architecture Software Developer's Manual Volumes 1, 2A, 2B, 3A, and 3B and bug fixes are posted in the Intel® 64 and IA-32 Architecture Software Developer's Manual Documentation Changes. Follow the following link to become familiar with this file: <http://developer.intel.com/products/processor/manuals/index.htm>



## Nomenclature

**S-Spec Number** is a five-digit code used to identify products. Products are differentiated by their unique characteristics, such as, core speed, L2 cache size, package type, etc. as described in the processor identification information table. Read all notes associated with each S-Spec number.

**Sightings** are design defects or errors. These may cause the Product Name's behavior to deviate from published specifications. Hardware and software designed to be used with any given stepping must assume that all sightings documented for that stepping are present on all devices

**Specification Changes** are modifications to the current published specifications. These changes will be incorporated in any new release of the specification.

**Specification Clarifications** describe a specification in greater detail or further highlight a specification's impact to a complex design situation. These clarifications will be incorporated in any new release of the specification.

**Documentation Changes** include typos, errors, or omissions from the current published specifications. These will be incorporated in any new release of the specification.

*Note:* Specification changes, specification clarifications and documentation changes are removed from the specification update when the appropriate changes are made to the appropriate product specification or user documentation (datasheets, manuals, etc.).



# Identification Information

## Component Identification using Programming Interface

The Intel® Core™ i7 processor family for the LGA-2011 Socket stepping can be identified by the following register contents:

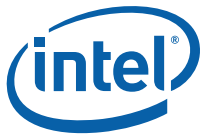
Reserved	Extended Family <sup>1</sup>	Extended Model <sup>2</sup>	Reserved	Processor Type <sup>3</sup>	Family Code <sup>4</sup>	Model Number <sup>5</sup>	Stepping ID
31:28	27:20	19:16	15:14	13:12	11:8	7:4	3:0
	00000000b	0010b		00b	0110b	1101b	xxxxb
				B0	36S	8086	3405h
				B2	36S	8086h	3405h

**Notes:**

1. The Extended Family, Bits [27:20] are used in conjunction with the Family Code, specified in Bits [11:8], to indicate whether the processor belongs to the Intel386™, Intel486™, Pentium®, Pentium 4, or Intel® Core™ processor family.
2. The Extended Model, Bits [19:16] in conjunction with the Model Number, specified in Bits [7:4], are used to identify the model of the processor within the processor's family.
3. The Family Code corresponds to Bits [11:8] of the EDX register after RESET, Bits [11:8] of the EAX register after the CPUID instruction is executed with a 1 in the EAX register, and the generation field of the Device ID register accessible through Boundary Scan.
4. The Model Number corresponds to Bits [7:4] of the EDX register after RESET, Bits [7:4] of the EAX register after the CPUID instruction is executed with a 1 in the EAX register, and the model field of the Device ID register accessible through Boundary Scan.
5. The Stepping ID in Bits [3:0] indicates the revision number of that model. See [Table 1](#) for the processor stepping ID number in the CPUID information.

When EAX is initialized to a value of '1', the CPUID instruction returns the *Extended Family, Extended Model, Processor Type, Family Code, Model Number and Stepping ID* value in the EAX register. The EDX processor signature value after reset is equivalent to the processor signature output value in the EAX register.

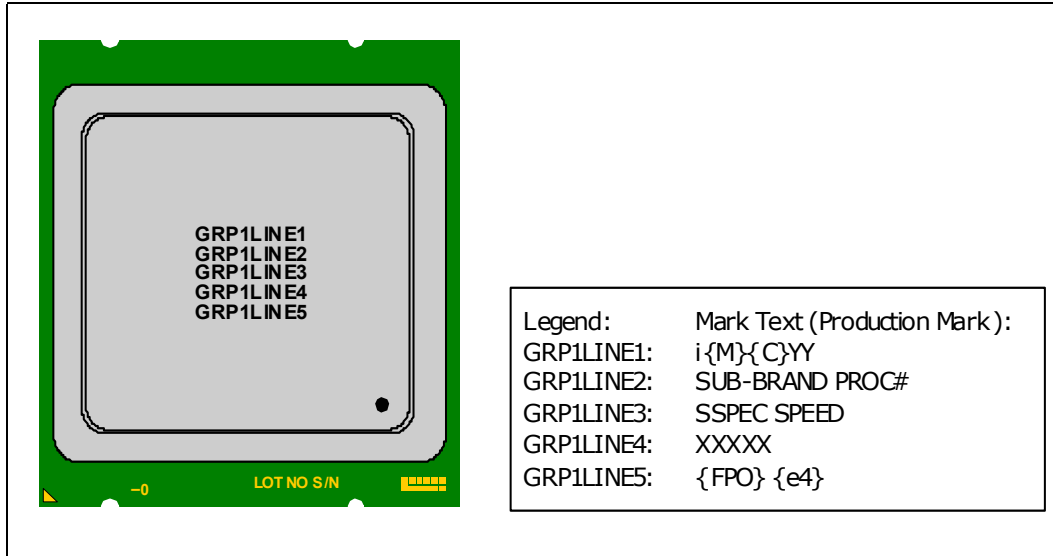
Cache and TLB descriptor parameters are provided in the EAX, EBX, ECX and EDX registers after the CPUID instruction is executed with a 2 in the EAX register.



## Component Marking Information

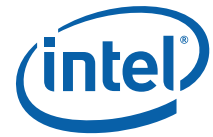
The Intel® Core™ i7 processor family for the LGA-2011 Socket can be identified by the following component markings.

**Figure 1. Intel® Core™ i7 Processor Family for the LGA-2011 Socket Top-side Markings (Example)**



**Table 1. Intel® Core™ i7 Processor Family for the LGA-2011 Socket Identification**

S-Spec Number	Stepping	CPUID	Core Frequency (GHz)/ DDR3(MHz)	TDP (W)	# Cores	Cache Size (MB)	Notes
SR0GW	C-1	0X206D6	3.3/1600	130	6	15	
SR0H9	C-1	0X206D6	3.2/1600	130	6	12	
SR0KF	C-2	0X206D7	3.3/1600	130	6	15	
SR0KY	C-2	0X206D7	3.2/1600	130	6	12	



# Summary Table of Changes

---

The table included in this section indicate the errata, Specification Changes, Specification Clarifications, or Document Changes which apply to the processor. Intel may fix some of the errata in a future stepping of the component, and account for the other outstanding issues through documentation or specification changes as noted.

Definitions are listed below for terminology used in the following Summary Tables.

## Codes Used in Summary Tables

### Stepping

X:	Errata exists in the stepping indicated. Specification Change or Clarification that applies to this stepping.
(No mark)	
or (Blank box):	This erratum is fixed in listed stepping or specification change does not apply to listed stepping.

### Page

(Page):	Page location of item in this document.
---------	---

### Status

Doc:	Document change or update will be implemented.
Plan Fix:	This erratum may be fixed in a future stepping of the product.
Fixed:	This erratum has been previously fixed.
No Fix:	There are no plans to fix this erratum.

### Row

Change bar to left of a table row indicates this erratum is either new or modified from the previous version of the document.



## Errata Summary Table (Sheet 1 of 4)

Errata Number	Steppings		Status	ERRATA
	C-1	C-2		
BS1	X	X	No Fix	An Enabled Debug Breakpoint or Single Step Trap May Be Taken after MOV SS/ POP SS Instruction if it is Followed by an Instruction That Signals a Floating Point Exception
BS2	X	X	No Fix	APIC Error "Received Illegal Vector" May be Lost
BS3	X	X	No Fix	An Uncorrectable Error Logged in IA32_CR_MC2_STATUS May also Result in a System Hang
BS4	X	X	No Fix	B0-B3 Bits in DR6 For Non-Enabled Breakpoints May be Incorrectly Set
BS5	X	X	No Fix	Changing the Memory Type for an In-Use Page Translation May Lead to Memory-Ordering Violations
BS6	X	X	No Fix	Code Segment Limit/Canonical Faults on RSM May be Serviced before Higher Priority Interrupts/Exceptions and May Push the Wrong Address Onto the Stack
BS7	X	X	No Fix	Corruption of CS Segment Register During RSM While Transitioning From Real Mode to Protected Mode
BS8	X	X	No Fix	Debug Exception Flags DR6.B0-B3 Flags May be Incorrect for Disabled Breakpoints
BS9	X	X	No Fix	DR6 May Contain Incorrect Information When the First Instruction After a MOV SS,r/m or POP SS is a Store
BS10	X	X	No Fix	EFLAGS Discrepancy on Page Faults and on EPT-Induced VM Exits after a Translation Change
BS11	X	X	No Fix	Fault on ENTER Instruction May Result in Unexpected Values on Stack Frame
BS12	X	X	No Fix	Faulting MMX Instruction May Incorrectly Update x87 FPU Tag Word
BS13	X	X	No Fix	FREEZE_WHILE_SMM Does Not Prevent Event From Pending PEBS During SMM
BS14	X	X	No Fix	General Protection Fault (#GP) for Instructions Greater than 15 Bytes May be Preempted
BS15	X	X	No Fix	#GP on Segment Selector Descriptor that Straddles Canonical Boundary May Not Provide Correct Exception Error Code
BS16	X	X	No Fix	IO_SMI Indication in SMRAM State Save Area May be Set Incorrectly
BS17	X	X	No Fix	IRET under Certain Conditions May Cause an Unexpected Alignment Check Exception
BS18	X	X	No Fix	LER MSRs May Be Unreliable
BS19	X	X	No Fix	LBR, BTS, BTM May Report a Wrong Address when an Exception/Interrupt Occurs in 64-bit Mode
BS20	X	X	No Fix	MCI_Status Overflow Bit May Be Incorrectly Set on a Single Instance of a DTLB Error
BS21	X	X	No Fix	MONITOR or CLFLUSH on the Local XAPIC's Address Space Results in Hang
BS22	X	X	No Fix	MOV To/From Debug Registers Causes Debug Exception
BS23	X	X	No Fix	PEBS Record not Updated when in Probe Mode
BS24	X	X	No Fix	Performance Monitor Counter INST_RETIRED.STORES May Count Higher than Expected
BS25	X	X	No Fix	Performance Monitoring Event FP_MMX_TRANS_TO_MMX May Not Count Some Transitions



## Errata Summary Table (Sheet 2 of 4)

Errata Number	Steppings		Status	ERRATA
	C-1	C-2		
BS26	X	X	No Fix	REP MOVSt/STOS Executing with Fast Strings Enabled and Crossing Page Boundaries with Inconsistent Memory Types may use an Incorrect Data Size or Lead to Memory-Ordering Violations.
BS27	X	X	No Fix	Reported Memory Type May Not Be Used to Access the VMCS and Referenced Data Structures
BS28	X	X	No Fix	Single Step Interrupts with Floating Point Exception Pending May Be Mishandled
BS29	X	X	No Fix	The Processor May Report a #TS Instead of a #GP Fault
BS30	X	X	No Fix	VM Exits Due to "NMI-Window Exiting" May Be Delayed by One Instruction
BS31	X	X	No Fix	Values for LBR/BTS/BTM Will be Incorrect after an Exit from SMM
BS32	X	X	No Fix	VPHMINPOSUW Instruction in VEX Format Does Not Signal #UD (Invalid Opcode Exception) When vex.vvvv !=1111
BS33	X	X	No Fix	Pending x87 FPU Exceptions (#MF) May be Signaled Earlier Than Expected
BS34	X	X	No Fix	VMREAD/VMWRITE Instruction May Not Fail When Accessing an Unsupported Field in VMCS
BS35	X	X	No Fix	Unexpected #UD on VZEROALL/VZEROUPPER
BS36	X	X	No Fix	Execution of Opcode 9BH with the VEX Opcode Extension May Produce a #NM Exception
BS37				Erratum Removed
BS38	X	X	No Fix	An Unexpected Page Fault or EPT Violation May Occur After Another Logical Processor Creates a Valid Translation for a Page
BS39	X	X	No Fix	Faulting Executions of XRSTOR May Update State Inconsistently
BS40	X	X	No Fix	Execution of FXSAVE or FXRSTOR With the VEX Prefix May Produce a #NM Exception
BS41	X	X	No Fix	Unexpected #UD on VPXSTRD/VPINSRD
BS42	X	X	No Fix	#GP May be Signaled When Invalid VEX Prefix Precedes Conditional Branch Instructions
BS43	X	X	No Fix	LBR, BTM or BTS Records May have Incorrect Branch From Information After an EIST/T-state/S-state/C1E Transition or Adaptive Thermal Throttling
BS44	X	X	No Fix	A Write to the IA32_FIXED_CTR1 MSR May Result in Incorrect Value in Certain Conditions
BS45	X	X	No Fix	L1 Data Cache Errors May be Logged With Level Set to 1 Instead of 0
BS46	X	X	No Fix	Warm Reset May Leave the System in an Invalid Poisoning State and Could Cause The Feature to be Disabled
BS47	X	X	No Fix	VM Entries That Return From SMM Using VMLAUNCH May Not Update The Launch State of the VMCS
BS48	X	X	No Fix	Interrupt From Local APIC Timer May Not Be Detectable While Being Delivered
BS49				Erratum Removed
BS50	X	X	No Fix	Poison Packets Will be Reported to PCIe* Port 1a When Forwarded to Port 1b
BS51	X	X	No Fix	IA32_MCi_ADDR Overwritten in The Case of Multiple Recoverable Instruction Fetch Errors
BS52	X	X	No Fix	The QPI Link Status Register LinkInitStatus Field Incorrectly Reports "Internal Stall Link Initialization" For Certain Stall Conditions



## Errata Summary Table (Sheet 3 of 4)

Errata Number	Steppings		Status	ERRATA
	C-1	C-2		
BS53	X	X	No Fix	PROCHOT_N Assertion During Warm Reset May Disable a Processor Via The FRB Mechanism
BS54	X	X	No Fix	The PCIe* Current Compensation Value Default is Incorrect
BS55	X	X	No Fix	The PCIe* Link at 8.0 GT/s is Transitioning Too Soon to Normal Operation While Training
BS56	X	X	No Fix	A First Level Data Cache Parity Error May Result in Unexpected Behavior
BS57	X	X	No Fix	PECI Write Requests That Require a Retry Will Always Time Out
BS58	X	X	No Fix	The Vswing of the PCIe* Transmitter Exceeds The Specification
BS59	X	X	No Fix	When a Link is Degraded on a Port due to PCIe* Signaling Issues Correctable Receiver Errors May be Reported on The Neighboring Port
BS60	X	X	No Fix	A CMCI is Only Generated When the Memory Controller's Correctable Error Count Threshold is Exceeded
BS61	X	X	No Fix	PCIe* Rx DC Common Mode Impedance is Not Meeting the Specification
BS62	X	X	No Fix	A Modification to the Multiple Message Enable Field Does Not Affect The AER Interrupt Message Number field
BS63	X	X	No Fix	Unexpected PCIe* Set_Slot_Power_Limit Message on Writes to LNKCON
BS64				Erratum Removed
BS65	X	X	No Fix	Locked Accesses Spanning Cachelines That Include PCI Space May Lead to a System Hang
BS66	X	X	No Fix	Cold Boot May Fail Due to Internal Timer Error
BS67	X	X	No Fix	PCIe* Rx Common Mode Return Loss is Not Meeting The Specification
BS68	X	X	No Fix	The Most Significant Bit of the CEC Cannot be Cleared Once Set
BS69	X	X	No Fix	PCIe* Adaptive Equalization May Not Train to the Optimal Settings
BS70	X	X	No Fix	A Core May Not Complete Transactions to The Caching Agent When C-States Are Enabled Leading to an Internal Timer Error
BS71	X	X	No Fix	TSC is Not Affected by Warm Reset
BS72	X	X	No Fix	Warm Resets May be Converted to Power-on Resets When Recovering From an IERR
BS73				Erratum Removed
BS74	X	X	No Fix	Processor May not Restore the VR12 DDR3 Voltage Regulator Phases upon Pkg C3 State Exit
BS75	X	X	No Fix	The Equalization Phase Successful Bits Are Not Compliant to The PCIe* Specification
BS76	X	X	No Fix	Using DMA XOR With DCA May Cause a Machine Check
BS77	X	X	No Fix	Mixed DMA XOR and Legacy Operations in The Same Channel May Cause Data to be Observed Out of Order
BS78	X	X	No Fix	Unexpected DMA XOR Halt and Errors When Using Descriptors With P or Q Operations Disabled
BS79	X	X	No Fix	DMA XOR Channel May Hang on Source Read Completion Data Parity Error For >8K Descriptors
BS80	X	X	No Fix	DMA CB_BAR Decode May be Incorrect After DMA FLR
BS81	X	X	No Fix	XOR DMA Restricted to <=8KB Transfers When Multiple Channels Are in use



## Errata Summary Table (Sheet 4 of 4)

Errata Number	Steppings		Status	ERRATA
	C-1	C-2		
BS82	X	X	No Fix	Unable to Restart DMA After Poisoned Error During an XOR Operation
BS83	X	X	No Fix	DMA Restart Hang When First Descriptor is a Legacy Type Following Channel HALT Due to an Extended Descriptor Error
BS84	X	X	No Fix	JSP CBDMA errata BF508S: Operation With DMA XOR Interrupts/Completions Enabled Restricted to Channel 0 and 1
BS85	X	X	No Fix	Suspending/Resetting an Active DMA XOR Channel May Cause an Incorrect Data Transfer on Other Active Channels
BS86	X	X	No Fix	DWORD-Aligned DMA XOR Descriptors With Fencing And Multi-Channel Operation May Cause a Channel Hang
BS87	X	X	No Fix	Storage of PEBS Record Delayed Following Execution of MOV SS or STI
BS88	X	X	No Fix	Instruction Fetch May Cause Machine Check if Page Size and Memory Type Was Changed Without Invalidation
BS89	X	X	No Fix	PCIe* Adaptive Equalization May Not Train to the Optimal Settings
BS90	X		Fixed	The VT-d Queued Invalidation Status Write May Fail
BS91	X	X	No Fix	Executing The GETSEC Instruction While Throttling May Result in a Processor Hang
BS92	X	X	No Fix	Platform Idle Power Higher May be Higher Than Expected
BS93	X	X	No Fix	PECI Transactions during an S-State Transition May Result in a Platform Cold Reset
BS94	X	X	No Fix	Complex Platform Conditions during a Transition to S4 or S5 State May Result in an Internal Timeout Error

## Specification Changes

Number	SPECIFICATION CHANGES
	There are no Specification Changes at this time

## Specification Clarifications

Number	SPECIFICATION CLARIFICATIONS
	There are no Specification Clarifications at this time.

## Documentation Changes

Number	DOCUMENTATION CHANGES
	There are no Documentation Changes at this time



# Errata

---

## **BS1. An Enabled Debug Breakpoint or Single Step Trap May Be Taken after MOV SS/POP SS Instruction if it is Followed by an Instruction That Signals a Floating Point Exception**

**Problem:** A MOV SS/POP SS instruction should inhibit all interrupts including debug breakpoints until after execution of the following instruction. This is intended to allow the sequential execution of MOV SS/POP SS and MOV [r/e]SP, [r/e]BP instructions without having an invalid stack during interrupt handling. However, an enabled debug breakpoint or single step trap may be taken after MOV SS/POP SS if this instruction is followed by an instruction that signals a floating point exception rather than a MOV [r/e]SP, [r/e]BP instruction. This results in a debug exception being signaled on an unexpected instruction boundary since the MOV SS/POP SS and the following instruction should be executed atomically.

**Implication:** This can result in incorrect signaling of a debug exception and possibly a mismatched Stack Segment and Stack Pointer. If MOV SS/POP SS is not followed by a MOV [r/e]SP, [r/e]BP, there may be a mismatched Stack Segment and Stack Pointer on any exception. Intel has not observed this erratum with any commercially available software or system.

**Workaround:** As recommended in the *IA32 Intel® Architecture Software Developer's Manual*, the use of MOV SS/POP SS in conjunction with MOV [r/e]SP, [r/e]BP will avoid the failure since the MOV [r/e]SP, [r/e]BP will not generate a floating point exception. Developers of debug tools should be aware of the potential incorrect debug event signaling created by this erratum.

**Status:** For the steppings affected, see the Summary Tables of Changes.

## **BS2. APIC Error "Received Illegal Vector" May be Lost**

**Problem:** APIC (Advanced Programmable Interrupt Controller) may not update the ESR (Error Status Register) flag Received Illegal Vector bit [6] properly when an illegal vector error is received on the same internal clock that the ESR is being written (as part of the write-read ESR access flow). The corresponding error interrupt will also not be generated for this case.

**Implication:** Due to this erratum, an incoming illegal vector error may not be logged into ESR properly and may not generate an error interrupt.

**Workaround:** None identified.

**Status:** For the steppings affected, see the Summary Tables of Changes.

## **BS3. An Uncorrectable Error Logged in IA32\_CR\_MC2\_STATUS May also Result in a System Hang**

**Problem:** Uncorrectable errors logged in IA32\_CR\_MC2\_STATUS MSR (409H) may also result in a system hang causing an Internal Timer Error (MCACOD = 0x0400h) to be logged in another machine check bank (IA32\_MCi\_STATUS).

**Implication:** Uncorrectable errors logged in IA32\_CR\_MC2\_STATUS can further cause a system hang and an Internal Timer Error to be logged.

**Workaround:** None identified.

**Status:** For the steppings affected, see the Summary Tables of Changes.



#### **BS4. B0-B3 Bits in DR6 For Non-Enabled Breakpoints May be Incorrectly Set**

**Problem:** Some of the B0-B3 bits (breakpoint conditions detect flags, bits [3:0]) in DR6 may be incorrectly set for non-enabled breakpoints when the following sequence happens:

1. MOV or POP instruction to SS (Stack Segment) selector;
2. Next instruction is FP (Floating Point) that gets FP assist
3. Another instruction after the FP instruction completes successfully
4. A breakpoint occurs due to either a data breakpoint on the preceding instruction or a code breakpoint on the next instruction.

Due to this erratum a non-enabled breakpoint triggered on step 1 or step 2 may be reported in B0-B3 after the breakpoint occurs in step 4.

**Implication:** Due to this erratum, B0-B3 bits in DR6 may be incorrectly set for non-enabled breakpoints.

**Workaround:** Software should not execute a floating point instruction directly after a MOV SS or POP SS instruction.

**Status:** For the steppings affected, see the Summary Tables of Changes.

#### **BS5. Changing the Memory Type for an In-Use Page Translation May Lead to Memory-Ordering Violations**

**Problem:** Under complex microarchitectural conditions, if software changes the memory type for data being actively used and shared by multiple threads without the use of semaphores or barriers, software may see load operations execute out of order.

**Implication:** Memory ordering may be violated. Intel has not observed this erratum with any commercially available software.

**Workaround:** Software should ensure pages are not being actively used before requesting their memory type be changed.

**Status:** For the steppings affected, see the Summary Tables of Changes.

#### **BS6. Code Segment Limit/Canonical Faults on RSM May be Serviced before Higher Priority Interrupts/Exceptions and May Push the Wrong Address Onto the Stack**

**Problem:** Normally, when the processor encounters a Segment Limit or Canonical Fault due to code execution, a #GP (General Protection Exception) fault is generated after all higher priority Interrupts and exceptions are serviced. Due to this erratum, if RSM (Resume from System Management Mode) returns to execution flow that results in a Code Segment Limit or Canonical Fault, the #GP fault may be serviced before a higher priority Interrupt or Exception (e.g. NMI (Non-Maskable Interrupt), Debug break(#DB), Machine Check (#MC), etc.). If the RSM attempts to return to a non-canonical address, the address pushed onto the stack for this #GP fault may not match the non-canonical address that caused the fault.

**Implication:** Operating systems may observe a #GP fault being serviced before higher priority Interrupts and Exceptions. Intel has not observed this erratum on any commercially available software.

**Workaround:** None identified.

**Status:** For the steppings affected, see the Summary Tables of Changes.



### **BS7. Corruption of CS Segment Register During RSM While Transitioning From Real Mode to Protected Mode**

**Problem:** During the transition from real mode to protected mode, if an SMI (System Management Interrupt) occurs between the MOV to CR0 that sets PE (Protection Enable, bit 0) and the first far JMP, the subsequent RSM (Resume from System Management Mode) may cause the lower two bits of CS segment register to be corrupted.

**Implication:** The corruption of the bottom two bits of the CS segment register will have no impact unless software explicitly examines the CS segment register between enabling protected mode and the first far JMP. Intel® 64 and IA-32 Architectures Software Developer's Manual Volume 3A: System Programming Guide, Part 1, in the section titled "Switching to Protected Mode" recommends the far JMP immediately follows the write to CR0 to enable protected mode. Intel has not observed this erratum with any commercially available software.

**Workaround:** None identified.

**Status:** For the steppings affected, see the Summary Tables of Changes.

### **BS8. Debug Exception Flags DR6.B0-B3 Flags May be Incorrect for Disabled Breakpoints**

**Problem:** When a debug exception is signaled on a load that crosses cache lines with data forwarded from a store and whose corresponding breakpoint enable flags are disabled (DR7.G0-G3 and DR7.L0-L3), the DR6.B0-B3 flags may be incorrect.

**Implication:** The debug exception DR6.B0-B3 flags may be incorrect for the load if the corresponding breakpoint enable flag in DR7 is disabled.

**Workaround:** None identified.

**Status:** For the steppings affected, see the Summary Tables of Changes.

### **BS9. DR6 May Contain Incorrect Information When the First Instruction After a MOV SS,r/m or POP SS is a Store**

**Problem:** Normally, each instruction clears the changes in DR6 (Debug Status Register) caused by the previous instruction. However, the instruction following a MOV SS,r/m (MOV to the stack segment selector) or POP SS (POP stack segment selector) instruction will not clear the changes in DR6 because data breakpoints are not taken immediately after a MOV SS,r/m or POP SS instruction. Due to this erratum, any DR6 changes caused by a MOV SS,r/m or POP SS instruction may be cleared if the following instruction is a store.

**Implication:** When this erratum occurs, incorrect information may exist in DR6. This erratum will not be observed under normal usage of the MOV SS,r/m or POP SS instructions (i.e., following them with an instruction that writes [e/r]SP). When debugging or when developing debuggers, this behavior should be noted.

**Workaround:** None identified.

**Status:** For the steppings affected, see the Summary Tables of Changes.



### **BS10. EFLAGS Discrepancy on Page Faults and on EPT-Induced VM Exits after a Translation Change**

**Problem:** This erratum is regarding the case where paging structures are modified to change a linear address from writable to non-writable without software performing an appropriate TLB invalidation. When a subsequent access to that address by a specific instruction (ADD, AND, BTC, BTR, BTS, CMPXCHG, DEC, INC, NEG, NOT, OR, ROL/ROR, SAL/SAR/SHL/SHR, SHLD, SHRD, SUB, XOR, and XADD) causes a page fault or an EPT-induced VM exit, the value saved for EFLAGS may incorrectly contain the arithmetic flag values that the EFLAGS register would have held had the instruction completed without fault or VM exit. For page faults, this can occur even if the fault causes a VM exit or if its delivery causes a nested fault.

**Implication:** None identified. Although the EFLAGS value saved by an affected event (a page fault or an EPT-induced VM exit) may contain incorrect arithmetic flag values, Intel has not identified software that is affected by this erratum. This erratum will have no further effects once the original instruction is restarted because the instruction will produce the same results as if it had initially completed without fault or VM exit.

**Workaround:** If the handler of the affected events inspects the arithmetic portion of the saved EFLAGS value, then system software should perform a synchronized paging structure modification and TLB invalidation.

**Status:** For the steppings affected, see the Summary Tables of Changes.

### **BS11. Fault on ENTER Instruction May Result in Unexpected Values on Stack Frame**

**Problem:** The ENTER instruction is used to create a procedure stack frame. Due to this erratum, if execution of the ENTER instruction results in a fault, the dynamic storage area of the resultant stack frame may contain unexpected values (i.e. residual stack data as a result of processing the fault).

**Implication:** Data in the created stack frame may be altered following a fault on the ENTER instruction. Please refer to "Procedure Calls For Block-Structured Languages" in IA-32 Intel® Architecture Software Developer's Manual, Vol. 1, Basic Architecture, for information on the usage of the ENTER instructions. This erratum is not expected to occur in ring 3. Faults are usually processed in ring 0 and stack switch occurs when transferring to ring 0. Intel has not observed this erratum on any commercially available software.

**Workaround:** None identified.

**Status:** For the steppings affected, see the Summary Tables of Changes.

### **BS12. Faulting MMX Instruction May Incorrectly Update x87 FPU Tag Word**

**Problem:** Under a specific set of conditions, MMX stores (MOVD, MOVQ, MOVNTQ, MASKMOVQ) which cause memory access faults (#GP, #SS, #PF, or #AC), may incorrectly update the x87 FPU tag word register.

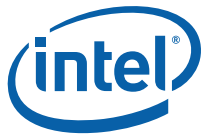
This erratum will occur when the following additional conditions are also met.

- The MMX store instruction must be the first MMX instruction to operate on x87 FPU state (i.e. the x87 FP tag word is not already set to 0x0000).
- For MOVD, MOVQ, MOVNTQ stores, the instruction must use an addressing mode that uses an index register (this condition does not apply to MASKMOVQ).

**Implication:** If the erratum conditions are met, the x87 FPU tag word register may be incorrectly set to a 0x0000 value when it should not have been modified.

**Workaround:** None identified

**Status:** For the steppings affected, see the Summary Tables of Changes.



### **BS13. FREEZE\_WHILE\_SMM Does Not Prevent Event From Pending PEBS During SMM**

**Problem:** In general, a PEBS record should be generated on the first count of the event after the counter has overflowed. However, IA32\_DEBUGCTL\_MSR.FREEZE\_WHILE\_SMM (MSR 1D9H, bit [14]) prevents performance counters from counting during SMM (System Management Mode). Due to this erratum, if

1. A performance counter overflowed before an SMI
2. A PEBS record has not yet been generated because another count of the event has not occurred
3. The monitored event occurs during SMM

then a PEBS record will be saved after the next RSM instruction.

When FREEZE\_WHILE\_SMM is set, a PEBS should not be generated until the event occurs outside of SMM.

**Implication:** A PEBS record may be saved after an RSM instruction due to the associated performance counter detecting the monitored event during SMM; even when FREEZE\_WHILE\_SMM is set.

**Workaround:** None identified.

**Status:** For the steppings affected, see the Summary Tables of Changes.

### **BS14. General Protection Fault (#GP) for Instructions Greater than 15 Bytes May be Preempted**

**Problem:** When the processor encounters an instruction that is greater than 15 bytes in length, a #GP is signaled when the instruction is decoded. Under some circumstances, the #GP fault may be preempted by another lower priority fault (e.g. Page Fault (#PF)). However, if the preempting lower priority faults are resolved by the operating system and the instruction retried, a #GP fault will occur.

**Implication:** Software may observe a lower-priority fault occurring before or in lieu of a #GP fault. Instructions of greater than 15 bytes in length can only occur if redundant prefixes are placed before the instruction.

**Workaround:** None identified.

**Status:** For the steppings affected, see the Summary Tables of Changes.

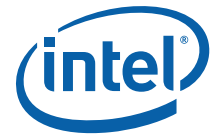
### **BS15. #GP on Segment Selector Descriptor that Straddles Canonical Boundary May Not Provide Correct Exception Error Code**

**Problem:** During a #GP (General Protection Exception), the processor pushes an error code on to the exception handler's stack. If the segment selector descriptor straddles the canonical boundary, the error code pushed onto the stack may be incorrect.

**Implication:** An incorrect error code may be pushed onto the stack. Intel has not observed this erratum with any commercially available software.

**Workaround:** None identified.

**Status:** For the steppings affected, see the Summary Tables of Changes.



### **BS16. IO\_SMI Indication in SMRAM State Save Area May be Set Incorrectly**

**Problem:** The IO\_SMI bit in SMRAM's location 7FA4H is set to "1" by the CPU to indicate a System Management Interrupt (SMI) occurred as the result of executing an instruction that reads from an I/O port. Due to this erratum, the IO\_SMI bit may be incorrectly set by:

- A non-I/O instruction
- SMI is pending while a lower priority event interrupts
- A REP I/O read
- A I/O read that redirects to MWAIT

**Implication:** SMM handlers may get false IO\_SMI indication.

**Workaround:** The SMM handler has to evaluate the saved context to determine if the SMI was triggered by an instruction that read from an I/O port. The SMM handler must not restart an I/O instruction if the platform has not been configured to generate a synchronous SMI for the recorded I/O port address.

**Status:** For the steppings affected, see the Summary Tables of Changes.

### **BS17. IRET under Certain Conditions May Cause an Unexpected Alignment Check Exception**

**Problem:** In IA-32e mode, it is possible to get an Alignment Check Exception (#AC) on the IRET instruction even though alignment checks were disabled at the start of the IRET. This can only occur if the IRET instruction is returning from CPL3 code to CPL3 code. IRETs from CPL0/1/2 are not affected. This erratum can occur if the EFLAGS value on the stack has the AC flag set, and the interrupt handler's stack is misaligned. In IA-32e mode, RSP is aligned to a 16-byte boundary before pushing the stack frame.

**Implication:** In IA-32e mode, under the conditions given above, an IRET can get a #AC even if alignment checks are disabled at the start of the IRET. This erratum can only be observed with a software generated stack frame.

**Workaround:** Software should not generate misaligned stack frames for use with IRET.

**Status:** For the steppings affected, see the Summary Tables of Changes.

### **BS18. LER MSRs May Be Unreliable**

**Problem:** Due to certain internal processor events, updates to the LER (Last Exception Record) MSRs, MSR\_LER\_FROM\_LIP (1DDH) and MSR\_LER\_TO\_LIP (1DEH), may happen when no update was expected.

**Implication:** The values of the LER MSRs may be unreliable.

**Workaround:** None Identified.

**Status:** For the steppings affected, see the Summary Tables of Changes.

### **BS19. LBR, BTS, BTM May Report a Wrong Address when an Exception/Interrupt Occurs in 64-bit Mode**

**Problem:** An exception/interrupt event should be transparent to the LBR (Last Branch Record), BTS (Branch Trace Store) and BTM (Branch Trace Message) mechanisms. However, during a specific boundary condition where the exception/interrupt occurs right after the execution of an instruction at the lower canonical boundary (0x00007FFFFFFFFF) in 64-bit mode, the LBR return registers will save a wrong return address with bits 63 to 48 incorrectly sign extended to all 1's. Subsequent BTS and BTM operations which report the LBR will also be incorrect.

**Implication:** LBR, BTS and BTM may report incorrect information in the event of an exception/interrupt.

**Workaround:** None identified.

**Status:** For the steppings affected, see the Summary Tables of Changes.



### **BS20. MCI\_Status Overflow Bit May Be Incorrectly Set on a Single Instance of a DTLB Error**

**Problem:** A single Data Translation Look Aside Buffer (DTLB) error can incorrectly set the Overflow (bit [62]) in the MCI\_Status register. A DTLB error is indicated by MCA error code (bits [15:0]) appearing as binary value, 000x 0000 0001 0100, in the MCI\_Status register.

**Implication:** Due to this erratum, the Overflow bit in the MCI\_Status register may not be an accurate indication of multiple occurrences of DTLB errors. There is no other impact to normal processor functionality.

**Workaround:** None identified.

**Status:** For the steppings affected, see the Summary Tables of Changes.

### **BS21. MONITOR or CLFLUSH on the Local xAPIC's Address Space Results in Hang**

**Problem:** If the target linear address range for a MONITOR or CLFLUSH is mapped to the local xAPIC's address space, the processor will hang.

**Implication:** When this erratum occurs, the processor will hang. The local xAPIC's address space must be uncached. The MONITOR instruction only functions correctly if the specified linear address range is of the type write-back. CLFLUSH flushes data from the cache. Intel has not observed this erratum with any commercially available software.

**Workaround:** Do not execute MONITOR or CLFLUSH instructions on the local xAPIC address space.

**Status:** For the steppings affected, see the Summary Tables of Changes.

### **BS22. MOV To/From Debug Registers Causes Debug Exception**

**Problem:** When in V86 mode, if a MOV instruction is executed to/from a debug registers, a general-protection exception (#GP) should be generated. However, in the case when the general detect enable flag (GD) bit is set, the observed behavior is that a debug exception (#DB) is generated instead.

**Implication:** With debug-register protection enabled (i.e., the GD bit set), when attempting to execute a MOV on debug registers in V86 mode, a debug exception will be generated instead of the expected general-protection fault.

**Workaround:** In general, operating systems do not set the GD bit when they are in V86 mode. The GD bit is generally set and used by debuggers. The debug exception handler should check that the exception did not occur in V86 mode before continuing. If the exception did occur in V86 mode, the exception may be directed to the general-protection exception handler.

**Status:** For the steppings affected, see the Summary Tables of Changes.

### **BS23. PEBS Record not Updated when in Probe Mode**

**Problem:** When a performance monitoring counter is configured for PEBS (Precise Event Based Sampling), overflows of the counter can result in storage of a PEBS record in the PEBS buffer. Due to this erratum, if the overflow occurs during probe mode, it may be ignored and a new PEBS record may not be added to the PEBS buffer.

**Implication:** Due to this erratum, the PEBS buffer may not be updated by overflows that occur during probe mode.

**Workaround:** None identified.

**Status:** For the steppings affected, see the Summary Tables of Changes.



#### **BS24. Performance Monitor Counter INST\_RETIREDD.STORES May Count Higher than Expected**

**Problem:** Performance Monitoring counter INST\_RETIREDD.STORES (Event: C0H) is used to track retired instructions which contain a store operation. Due to this erratum, the processor may also count other types of instructions including WRMSR and MFENCE.

**Implication:** Performance Monitoring counter INST\_RETIREDD.STORES may report counts higher than expected.

**Workaround:** None identified.

**Status:** For the steppings affected, see the Summary Tables of Changes.

#### **BS25. Performance Monitoring Event FP\_MMX\_TRANS\_TO\_MMX May Not Count Some Transitions**

**Problem:** Performance Monitor Event FP\_MMX\_TRANS\_TO\_MMX (Event CCH, Umask 01H) counts transitions from x87 Floating Point (FP) to MMX™ instructions. Due to this erratum, if only a small number of MMX instructions (including EMMS) are executed immediately after the last FP instruction, a FP to MMX transition may not be counted.

**Implication:** The count value for Performance Monitoring Event FP\_MMX\_TRANS\_TO\_MMX may be lower than expected. The degree of undercounting is dependent on the occurrences of the erratum condition while the counter is active. Intel has not observed this erratum with any commercially available software.

**Workaround:** None Identified.

**Status:** For the steppings affected, see the Summary Tables of Changes.

#### **BS26. REP MOVSS/STOSS Executing with Fast Strings Enabled and Crossing Page Boundaries with Inconsistent Memory Types may use an Incorrect Data Size or Lead to Memory-Ordering Violations.**

**Problem:** Under certain conditions as described in the Software Developers Manual section “Out-of-Order Stores For String Operations in Pentium 4, Intel Xeon, and P6 Family Processors” the processor performs REP MOVSS or REP STOSS as fast strings. Due to this erratum fast string REP MOVSS/REP STOSS instructions that cross page boundaries from WB/WC memory types to UC/WP/WT memory types, may start using an incorrect data size or may observe memory ordering violations.

**Implication:** Upon crossing the page boundary the following may occur, dependent on the new page memory type:

- UC the data size of each write will now always be 8 bytes, as opposed to the original data size.
- WP the data size of each write will now always be 8 bytes, as opposed to the original data size and there may be a memory ordering violation.
- WT there may be a memory ordering violation.

**Workaround:** Software should avoid crossing page boundaries from WB or WC memory type to UC, WP or WT memory type within a single REP MOVSS or REP STOSS instruction that will execute with fast strings enabled.

**Status:** For the steppings affected, see the Summary Tables of Changes.



### **BS27. Reported Memory Type May Not Be Used to Access the VMCS and Referenced Data Structures**

**Problem:** Bits 53:50 of the IA32\_VMX\_BASIC MSR report the memory type that the processor uses to access the VMCS and data structures referenced by pointers in the VMCS. Due to this erratum, a VMX access to the VMCS or referenced data structures will instead use the memory type that the MTRRs (memory-type range registers) specify for the physical address of the access.

**Implication:** Bits 53:50 of the IA32\_VMX\_BASIC MSR report that the WB (write-back) memory type will be used but the processor may use a different memory type.

**Workaround:** Software should ensure that the VMCS and referenced data structures are located at physical addresses that are mapped to WB memory type by the MTRRs.

**Status:** For the steppings affected, see the Summary Tables of Changes.

### **BS28. Single Step Interrupts with Floating Point Exception Pending May Be Mishandled**

**Problem:** In certain circumstances, when a floating point exception (#MF) is pending during single-step execution, processing of the single-step debug exception (#DB) may be mishandled.

**Implication:** When this erratum occurs, #DB will be incorrectly handled as follows:

- #DB is signaled before the pending higher priority #MF (Interrupt 16)
- #DB is generated twice on the same instruction

**Workaround:** None identified.

**Status:** For the steppings affected, see the Summary Tables of Changes.

### **BS29. The Processor May Report a #TS Instead of a #GP Fault**

**Problem:** A jump to a busy TSS (Task-State Segment) may cause a #TS (invalid TSS exception) instead of a #GP fault (general protection exception).

**Implication:** Operation systems that access a busy TSS may get invalid TSS fault instead of a #GP fault. Intel has not observed this erratum with any commercially available software.

**Workaround:** None identified.

**Status:** For the steppings affected, see the Summary Tables of Changes.

### **BS30. VM Exits Due to "NMI-Window Exiting" May Be Delayed by One Instruction**

**Problem:** If VM entry is executed with the "NMI-window exiting" VM-execution control set to 1, a VM exit with exit reason "NMI window" should occur before execution of any instruction if there is no virtual-NMI blocking, no blocking of events by MOV SS, and no blocking of events by STI. If VM entry is made with no virtual-NMI blocking but with blocking of events by either MOV SS or STI, such a VM exit should occur after execution of one instruction in VMX non-root operation. Due to this erratum, the VM exit may be delayed by one additional instruction.

**Implication:** VMM software using "NMI-window exiting" for NMI virtualization should generally be unaffected, as the erratum causes at most a one-instruction delay in the injection of a virtual NMI, which is virtually asynchronous. The erratum may affect VMMs relying on deterministic delivery of the affected VM exits.

**Workaround:** None identified.

**Status:** For the steppings affected, see the Summary Tables of Changes.



### **BS31. Values for LBR/BTS/BTM Will be Incorrect after an Exit from SMM**

**Problem:** After a return from SMM (System Management Mode), the CPU will incorrectly update the LBR (Last Branch Record) and the BTS (Branch Trace Store), hence rendering their data invalid. The corresponding data if sent out as a BTM on the system bus will also be incorrect.

Note: This issue would only occur when one of the 3 above mentioned debug support facilities are used.

**Implication:** The value of the LBR, BTS, and BTM immediately after an RSM operation should not be used.

**Workaround:** None identified.

**Status:** For the steppings affected, see the Summary Tables of Changes.

### **BS32. VPHMINPOSUW Instruction in VEX Format Does Not Signal #UD (Invalid Opcode Exception) When vex.vvvv != 1111**

**Problem:** Processor does not signal #UD fault when executing the reserved instruction VPHMINPOSUW with vex.vvvv!=1111. The VPHMINPOSUW instruction is described in greater detail in the Intel® Advanced Vector Extensions Programming Reference.

**Implication:** Executing VPHMINPOSUW with vex.vvvv != 1111 results in same behavior as vex.vvvv= 1111.

**Workaround:** SW should not use VPHMINPOSUW with vex.vvvv != 1111 in order to ensure future compatibility.

**Status:** For the steppings affected, see the Summary Tables of Changes.

### **BS33. Pending x87 FPU Exceptions (#MF) May be Signaled Earlier Than Expected**

**Problem:** x87 instructions that trigger #MF normally service interrupts before the #MF. Due to this erratum, if an instruction that triggers #MF is executed while Enhanced Intel SpeedStep® Technology transitions, Intel® Turbo Boost Technology transitions, or Thermal Monitor events occur, the pending #MF may be signaled before pending interrupts are serviced.

**Implication:** Software may observe #MF being signaled before pending interrupts are serviced.

**Workaround:** None identified.

**Status:** For the steppings affected, see the Summary Tables of Changes.

### **BS34. VMREAD/VMWRITE Instruction May Not Fail When Accessing an Unsupported Field in VMCS**

**Problem:** The Intel® 64 and IA-32 Architectures Software Developer's Manual, Volume 2B states that execution of VMREAD or VMWRITE should fail if the value of the instruction's register source operand corresponds to an unsupported field in the VMCS (Virtual Machine Control Structure). The correct operation is that the logical processor will set the ZF (Zero Flag), write 0CH into the VM-instruction error field and for VMREAD leave the instruction's destination operand unmodified. Due to this erratum, the instruction may instead clear the ZF, leave the VM-instruction error field unmodified and for VMREAD modify the contents of its destination operand.

**Implication:** Accessing an unsupported field in VMCS will fail to properly report an error. In addition, VMREAD from an unsupported VMCS field may unexpectedly change its destination operand. Intel has not observed this erratum with any commercially available software.

**Workaround:** Software should avoid accessing unsupported fields in a VMCS.

**Status:** For the steppings affected, see the Summary Tables of Changes.



### **BS35. Unexpected #UD on VZEROALL/VZERoupper**

**Problem:** Execution of the VZEROALL or VZERoupper instructions in 64-bit mode with VEX.W set to 1 may erroneously cause a #UD (invalid-opcode exception).

**Implication:** The affected instructions may produce unexpected invalid-opcode exceptions in 64-bit mode.

**Workaround:** Compilers should encode VEX.W = 0 for the VZEROALL and VZERoupper instructions.

**Status:** For the steppings affected, see the Summary Tables of Changes.

### **BS36. Execution of Opcode 9BH with the VEX Opcode Extension May Produce a #NM Exception**

**Problem:** Attempt to use opcode 9BH with a VEX opcode extension should produce a #UD (Invalid-Opcode) exception. Due to this erratum, if CR0.MP and CR0.TS are both 1, the processor may produce a #NM (Device-Not-Available) exception if one of the following conditions exists:

- 66H, F2H, F3H or REX as a preceding prefix;
- An illegal map specified in the VEX.mmmmm field;

**Implication:** Due to this erratum, some undefined instruction encodings may produce a #NM instead of a #UD exception.

**Workaround:** Software should not use opcode 9BH with the VEX opcode extension.

**Status:** For the steppings affected, see the Summary Tables of Changes.

### **BS37. Erratum Removed**

### **BS38. An Unexpected Page Fault or EPT Violation May Occur After Another Logical Processor Creates a Valid Translation for a Page**

**Problem:** An unexpected page fault (#PF) or EPT violation may occur for a page under the following conditions:

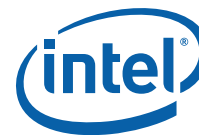
- The paging structures initially specify no valid translation for the page.
- Software on one logical processor modifies the paging structures so that there is a valid translation for the page (e.g., by setting to 1 the present bit in one of the paging-structure entries used to translate the page).
- Software on another logical processor observes this modification (e.g., by accessing a linear address on the page or by reading the modified paging-structure entry and seeing value 1 for the present bit).
- Shortly thereafter, software on that other logical processor performs a store to a linear address on the page.

In this case, the store may cause a page fault or EPT violation that indicates that there is no translation for the page (e.g., with bit 0 clear in the page-fault error code, indicating that the fault was caused by a not-present page). Intel has not observed this erratum with any commercially available software.

**Implication:** An unexpected page fault may be reported. There are no other side effects due to this erratum.

**Workaround:** System software can be constructed to tolerate these unexpected page faults. See Section "Propagation of Paging-Structure Changes to Multiple Processors" of Volume 3B of IA-32 Intel® Architecture Software Developer's Manual, for recommendations for software treatment of asynchronous paging-structure updates.

**Status:** For the steppings affected, see the Summary Tables of Changes.



### **BS39. Faulting Executions of XRSTOR May Update State Inconsistently**

**Problem:** The state updated by a faulting XRSTOR instruction may vary from one execution to another.

**Implication:** Software that relies on x87/SSE/AVX state following a faulting execution of XRSTOR may behave inconsistently.

**Workaround:** Software handling a fault on an execution of XRSTOR can compensate for execution variability by correcting the cause of the fault and executing XRSTOR again.

**Status:** For the steppings affected, see the Summary Tables of Changes.

### **BS40. Execution of FXSAVE or FXRSTOR With the VEX Prefix May Produce a #NM Exception**

**Problem:** Attempt to use FXSAVE or FXRSTOR with a VEX prefix should produce a #UD (Invalid-Opcode) exception. If either the TS or EM flag bits in CR0 are set, a #NM (device-not-available) exception will be raised instead of #UD exception.

**Implication:** Due to this erratum a #NM exception may be signaled instead of a #UD exception on an FXSAVE or an FXRSTOR with a VEX prefix.

**Workaround:** Software should not use FXSAVE or FXRSTOR with the VEX prefix.

**Status:** For the steppings affected, see the Summary Tables of Changes.

### **BS41. Unexpected #UD on VPEXTRD/VPINSRD**

**Problem:** Execution of the VPEXTRD or VPINSRD instructions outside of 64-bit mode with VEX.W set to 1 may erroneously cause a #UD (invalid-opcode exception).

**Implication:** The affected instructions may produce unexpected invalid-opcode exceptions outside 64-bit mode.

**Workaround:** Software should encode VEX.W = 0 for executions of the VPEXTRD and VPINSRD instructions outside 64-bit mode.

**Status:** For the steppings affected, see the Summary Tables of Changes.

### **BS42. #GP May be Signaled When Invalid VEX Prefix Precedes Conditional Branch Instructions**

**Problem:** When a 2-byte opcode of a conditional branch (opcodes 0F8xH, for any value of x) instruction resides in 16-bit code-segment and is associated with invalid VEX prefix, it may sometimes signal a #GP fault (illegal instruction length > 15-bytes) instead of a #UD (illegal opcode) fault.

**Implication:** Due to this erratum, #GP fault instead of a #UD may be signaled on an illegal instruction.

**Workaround:** None identified.

**Status:** For the steppings affected, see the Summary Tables of Changes.



**BS43. LBR, BTM or BTS Records May have Incorrect Branch From Information After an EIST/T-state/S-state/C1E Transition or Adaptive Thermal Throttling**

**Problem:** The “From” address associated with the LBR (Last Branch Record), BTM (Branch Trace Message) or BTS (Branch Trace Store) may be incorrect for the first branch after a transition of:

- EIST (Enhanced Intel® SpeedStep Technology)
- T-state (Thermal Monitor states)
- S1-state (ACPI package sleep state)
- C1E (Enhanced C1 Low Power state)
- Adaptive Thermal Throttling

**Implication:** When the LBRs, BTM or BTS are enabled, some records may have incorrect branch “From” addresses for the first branch after a transition of EIST, T-states, S-states, C1E, or Adaptive Thermal Throttling.

**Workaround:** None identified.

**Status:** For the steppings affected, see the Summary Tables of Changes.

**BS44. A Write to the IA32\_FIXED\_CTR1 MSR May Result in Incorrect Value in Certain Conditions**

**Problem:** Under specific internal conditions, if software tries to write the IA32\_FIXED\_CTR1 MSR (30AH) a value that has all bits [31:1] set while the counter was just about to overflow when the write is attempted (i.e. its value was 0xFFFF FFFF FFFF), then due to this erratum the new value in the MSR may be corrupted.

**Implication:** Due to this erratum, IA32\_FIXED\_CTR1 MSR may be written with a corrupted value.

**Workaround:** Software may avoid this erratum by writing zeros to the IA32\_FIXED\_CTR1 MSR, before the desired write operation.

**Status:** For the steppings affected, see the Summary Tables of Changes.

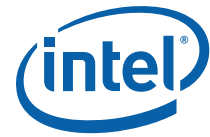
**BS45. L1 Data Cache Errors May be Logged With Level Set to 1 Instead of 0**

**Problem:** When an L1 Data Cache error is logged in IA32\_MCi\_STATUS[15:0], which is the MCA Error Code Field, with a cache error type of the format 0000 0001 RRRR TTLL, the LL field may be incorrectly encoded as 01b instead of 00b.

**Implication:** An error in the L1 Data Cache may report the same LL value as the L2 Cache. Software should not assume that an LL value of 01b is the L2 Cache.

**Workaround:** None identified.

**Status:** For the steppings affected, see the Summary Tables of Changes.



#### **BS46. Warm Reset May Leave the System in an Invalid Poisoning State and Could Cause The Feature to be Disabled**

**Problem:** Due to this erratum, the PCIe Poison forwarding enable and Intel® QPI Poison Enable bits are cleared by warm reset, but other bits related to the Poisoning feature remain set. After the warm reset the system may be in an invalid state in regards to the Poisoning bits. This invalid state may cause the feature to be disabled.

**Implication:** This invalid state may prevent the propagation of the poisoning indication, effectively disabling the feature.

**Workaround:** If poisoning is disabled, program the following bits to 0 after reset. If poisoning is enabled, program the following bits to 1 after reset:

The IA32\_MCG\_CONTAIN.POISON\_ENABLE bit (MSR 178H, bit 0). It should be noted that each thread must perform this action.

The IA32\_MCG\_CONTAIN.POISON\_ENABLE (MSR 178H, bit 0).

The POISFEN bit (IIOMISCCTRL; CPUBUS(0); Device 5; Function 0; Offset 1C0H; Bit 37).

The DMASK bit (UNCEDMASK; CPUBUS(0); Device 0, 1, 2, 3; Functions 0, 1, 2, 3; Offset 218H; bit 12).

**Status:** For the steppings affected, see the Summary Tables of Changes.

#### **BS47. VM Entries That Return From SMM Using VMLAUNCH May Not Update The Launch State of the VMCS**

**Problem:** Successful VM entries using the VMLAUNCH instruction should set the launch state of the VMCS to "launched". Due to this erratum, such a VM entry may not update the launch state of the current VMCS if the VM entry is returning from SMM.

**Implication:** Subsequent VM entries using the VMRESUME instruction with this VMCS will fail. RFLAGS.ZF is set to 1 and the value 5 (indicating VMRESUME with non-launched VMCS) is stored in the VM-instruction error field. This erratum applies only if dual monitor treatment of SMI and SMM is active.

**Workaround:** It is possible for the BIOS to contain a workaround for this erratum.

**Status:** For the steppings affected, see the Summary Tables of Changes.

#### **BS48. Interrupt From Local APIC Timer May Not Be Detectable While Being Delivered**

**Problem:** If the local-APIC timer's CCR (current-count register) is 0, software should be able to determine whether a previously generated timer interrupt is being delivered by first reading the delivery-status bit in the LVT timer register and then reading the bit in the IRR (interrupt-request register) corresponding to the vector in the LVT timer register. If both values are read as 0, no timer interrupt should be in the process of being delivered. Due to this erratum, a timer interrupt may be delivered even if the CCR is 0 and the LVT and IRR bits are read as 0. This can occur only if the DCR (Divide Configuration Register) is greater than or equal to 4. The erratum does not occur if software writes zero to the Initial Count Register before reading the LVT and IRR bits.

**Implication:** Software that relies on reads of the LVT and IRR bits to determine whether a timer interrupt is being delivered may not operate properly.

**Workaround:** Software that uses the local-APIC timer must be prepared to handle the timer interrupts, even those that would not be expected based on reading CCR and the LVT and IRR bits; alternatively, software can avoid the problem by writing zero to the Initial Count Register before reading the LVT and IRR bits.

**Status:** For the steppings affected, see the Summary Tables of Changes.



**BS49. Erratum Removed**

**BS50. Poison Packets Will be Reported to PCIe\* Port 1a When Forwarded to Port 1b**

**Problem:** With respect to data poisoning, the processor IIO module supports forwarding poisoned information between the coherent interface and PCIe and vice-versa. Also the processor IIO module supports forwarding poisoned data between peer PCIe ports. When the PCIe Ports 1a and 1b are configured as x4, the outbound Poison Error is reported on Port 1a when a poison packet is forwarded to Port 1b.

**Implication:** When Ports 1a and 1b are configured as x4 ports, Poison Errors reported on the root port are unreliable.

**Workaround:** None identified.

**Status:** For the steppings affected, see the Summary Tables of Changes.

**BS51. IA32\_MCi\_ADDR Overwritten in The Case of Multiple Recoverable Instruction Fetch Errors**

**Problem:** The instruction fetch machine check error (MCACOD 0x150) is a SRAR (Software Recoverable Action Required) error. The address of the location with the error is provided in the corresponding IA32\_MCi\_ADDR MSR. When multiple instruction fetch errors are logged as part of a single machine check event, as indicated by setting of the Overflow (bit 62) in the IA32\_MCi\_STATUS MSR, then recovery is not possible. Due to this erratum, when multiple instruction fetch errors are logged in the same bank, the IA32\_MCi\_MISC MSR contains all of the correct information including the proper setting for Overflow (bit 62); however, the IA32\_MCi\_ADDR MSR is overwritten with a value that corresponds to neither the first or second error.

**Implication:** When debugging failures associated with the instruction fetch machine check error and the Overflow bit is set, the value in IA32\_MCi\_ADDR will not be valid.

**Workaround:** It is possible for the BIOS to contain a workaround for this erratum.

**Status:** For the steppings affected, see the Summary Tables of Changes.

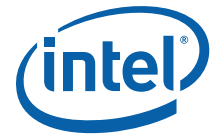
**BS52. The QPI Link Status Register LinkInitStatus Field Incorrectly Reports "Internal Stall Link Initialization" For Certain Stall Conditions**

**Problem:** The Intel® QPI Link Control register (CPUBUS(1), Devices 8, 9; Function 0; Offset 0x44) bits 17 and 16 allow for the control of the Link Layer Initialization by forcing the link to stall the initialization process until cleared. The Intel® QPI Link Status register (CPUBUS(1), Device 8, 9; Function 0; Offset 0x48) bits 27:24 report the Link Initialization Status (LinkInitStatus). The LinkInitStatus incorrectly reports "Internal Stall Link Initialization" (0001b) for non- QPI Link Control register, bit[17,16] stall conditions. The Intel® QPI Specification does not intend for internal stall conditions to report that status, but rather report the normal "Waiting for Physical Layer Ready" (0000b).

**Implication:** There is no known problem with this behavior since there is no usage model that relies on polling of the LinkInitStatus state in the "Waiting for Physical Layer Ready" versus "Internal Stall Link Initialization" state, and it only advertises the "Internal Stall Link Initialization" state for a brief period of time during Link Layer Initialization.

**Workaround:** None identified.

**Status:** For the steppings affected, see the Summary Tables of Changes.



### **BS53. PROCHOT\_N Assertion During Warm Reset May Disable a Processor Via The FRB Mechanism**

**Problem:** FRB (Fault Resilient Booting) is defined as the ability to boot even when one or more processors in the system fail, as long as there is one processor functional. If a warm reset is asserted during the boot flow before the Intel® QPI Interface is enumerated and while a processor is hot and drives PROCHOT\_N, the processor that is driving PROCHOT\_N will mistakenly observe PROCHOT\_N as a signal to transition itself into FRB mode.

**Implication:** It is possible that a processor may be incorrectly isolated via the FRB mechanism if the same processor asserts PROCHOT\_N during a warm reset.

**Workaround:** Case 1: Systems with a BMC

Case 1.1: Legacy Processor gets disabled: The system will not boot. The BMC can detect this case by observing that legacy socket is occupied, but the processor times out on PECI Ping() command. Since BMC knows it did not disable legacy socket, it can assume this is an error case.

Case 1.2: Non-Legacy Processor gets disabled: If system boots with one or more fewer sockets, BMC will observe a discrepancy between socket occupied pins and response to PECI Ping() command. If BMC did not disable the affected socket, it can conclude they were accidentally disabled due to this issue. The BMC can respond to either case by issuing a cold reset to the platform.

Case 2: Systems without a BMC

Case 2.1: Legacy Socket gets disabled: This will prevent booting. The PCH (Platform Control Hub) TCO logic can be strapped to reset the platform if the CPU does not fetch code after reset.

Case 2.2: Non-Legacy Socket gets disabled: BIOS cannot read socket occupied pin from other socket. Therefore, BIOS cannot tell the difference between a tri-stated

socket and unpopulated socket. Enable Autoack in the Intel® QPI Interface Enumeration which will ensure that a warm reset asserted before the Intel® Quick Path Interface Enumeration will be converted into a power-cycle reset.

**Status:** For the steppings affected, see the Summary Tables of Changes.

### **BS54. The PCIe\* Current Compensation Value Default is Incorrect**

**Problem:** The default current compensation values for PCIe buffers may result in non-optimal performance.

**Implication:** The PCIe buffers will not perform as well as possible and performance could be compromised.

**Workaround:** A BIOS code change has been identified and may be implemented as a workaround for this erratum.

**Status:** For the steppings affected, see the Summary Tables of Changes.



### **BS55. The PCIe\* Link at 8.0 GT/s is Transitioning Too Soon to Normal Operation While Training**

**Problem:** The PCIe bus uses high speed serial links that must go through a training process to allow both transmitter and receiver to make adjustments in behavior to optimize the signaling between the transmitter and receiver. When a PCIe compliant device must train or retrain the link, training sequences are used. The device must allow enough time for the training to complete before transitioning to normal operation. In the case of PCIe equalization at 8.0 GT/s the processor is not allowing enough time to optimize signaling before attempting normal operation.

**Implication:** Due to this erratum, unexpected system behavior may be observed.

**Workaround:** A BIOS code change has been identified and may be implemented as a workaround for this erratum.

**Status:** For the steppings affected, see the Summary Tables of Changes.

### **BS56. A First Level Data Cache Parity Error May Result in Unexpected Behavior**

**Problem:** When a load occurs to a first level data cache line resulting in a parity error in close proximity to other software accesses to the same cache line and other locked accesses the processor may exhibit unexpected behavior.

**Implication:** Due to this erratum unpredictable system behavior may occur. Intel has not observed this erratum with any commercially available system.

**Workaround:** None identified.

**Status:** For the steppings affected, see the Summary Tables of Changes.

### **BS57. PECI Write Requests That Require a Retry Will Always Time Out**

**Problem:** Peci 3.0 introduces a 'Host Identification' field as a way for the Peci host device to identify itself to the Peci client. This is intended for use in future Peci systems that may support more than one Peci originator. Since Peci 3.0 systems do not support the use of multiple originators, Peci 3.0 host devices should zero out the unused Host ID field. Peci 3.0 also introduces a 'retry' bit as a way for the Peci host to indicate to the client that the current request is a 'retry' of a previous read or write operation. Unless the Peci 3.0 host device zeroes out the byte containing the 'Host ID & Retry bit' information, Peci write requests that require a retry will never complete successfully.

**Implication:** Peci write requests that require a retry may never complete successfully. Instead, they will return a timeout completion code of 81H for a period ranging from 1ms to 30ms if the 'RETRY' bit is asserted.

**Workaround:** Peci 3.0 host devices should zero out the byte that contains the Host ID and Retry bit information for all Peci requests at all times including retries.

**Status:** For the steppings affected, see the Summary Tables of Changes.

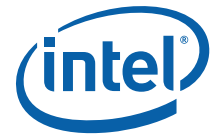
### **BS58. The Vswing of the PCIe\* Transmitter Exceeds The Specification**

**Problem:** The PCIe Specification defines a limit for the Vswing (Voltage Swing) of the differential lines that make up a lane to be 1200 mV peak-to-peak when operating at 2.5 GT/s and 5 GT/s. Intel has found that the processor's PCIe transmitter may exceed this specification. Peak-to-peak swings on a limited number of samples have been observed up to 1450 mV.

**Implication:** For those taking direct measurements of the PCIe transmit traffic coming from the processor may detect that the Vswing exceeds the PCIe Specification. Intel has not observed any functional failures due to this erratum.

**Workaround:** None identified.

**Status:** For the steppings affected, see the Summary Tables of Changes.



**BS59. When a Link is Degraded on a Port due to PCIe\* Signaling Issues Correctable Receiver Errors May be Reported on The Neighboring Port**

**Problem:** PCI Express interface incorporates a recovery mechanism when certain link degradation occurs by retraining the link without impacting the pending transactions. When a link is degraded on a specific port due to PCIe signaling issues, it is possible that correctable receiver errors are reported on the neighboring (logically adjacent) port. The correctable receiver errors are indicated by the PCIe AER Correctable error bit (XPGLBERRSTS CPUBUS(0); Device 0-3; Function 0-3; Offset 230H; Bit 2).

**Implication:** Software that logs errors on the PCIe interface must be aware that errors detected on a specific port could be due to either an error on that specific port or on a neighboring port.

**Workaround:** None identified.

**Status:** For the steppings affected, see the Summary Tables of Changes.

**BS60. A CMCI is Only Generated When the Memory Controller's Correctable Error Count Threshold is Exceeded**

**Problem:** A CMCI (corrected machine-check error interrupt) should be generated when the number of corrected errors for a bank reaches the corrected error threshold programmed into the IA32\_MCi\_CTL2 bits [14:0]. For memory scrubbing errors, IA32\_MCi\_STATUS.MCACOD (bits [15:0]) with value of 000x\_0000\_1100\_xxxx (where x stands for zero or one), a CMCI will not be generated until the number of errors has exceeded the threshold in IA32\_MCi\_CTL2 by 1.

**Implication:** The CMCI will not be generated when expected but rather will be generated on the next corrected error for the bank.

**Workaround:** It is possible for BIOS to contain a workaround for this issue. It should be noted that with this workaround if the threshold is programmed to a value of 0, a read of the value will return 1 and the threshold will be 1. All other valid threshold values for the bank will be read back correctly and function as expected.

**Status:** For the steppings affected, see the Summary Tables of Changes.

**BS61. PCIe\* Rx DC Common Mode Impedance is Not Meeting the Specification**

**Problem:** When the PCIe Rx termination is not powered, the DC Common Mode impedance has the following requirement:  $\geq 10 \text{ k}\Omega$  over 0-200 mV range with respect to ground and  $\geq 20 \text{ k}\Omega$  for voltages  $\geq 200 \text{ mV}$  with respect to ground. The processor's PCIe Rx do not meet this requirement at 85 degrees C or greater. In a limited number of samples Intel has measured an impedance as low as 9.85 k $\Omega$  at 50mV.

**Implication:** Intel has not observed any functional impact due to this violation with any commercially available system.

**Workaround:** None identified.

**Status:** For the steppings affected, see the Summary Tables of Changes.



### **BS62. A Modification to the Multiple Message Enable Field Does Not Affect The AER Interrupt Message Number field**

**Problem:** The (Advanced Error Interrupt) Message Number field (RPERRSTS Devices 0-3; Functions 0-3; Offset 178H; bits[31:27]) should be updated when the number of messages allocated to the root port is changed by writing the Multiple Message Enable field (MSIMSGCTL Device 3; Function 0; Offset 62H; bits[6:4]). However, writing the Multiple Message Enable in the root port does not update the Advanced Error Interrupt Message Number field.

**Implication:** Due to this erratum, software can allocate only one MSI (Message Signaled Interrupt) to the root port.

**Workaround:** None identified.

**Status:** For the steppings affected, see the Summary Tables of Changes.

### **BS63. Unexpected PCIe\* Set\_Slot\_Power\_Limit Message on Writes to LNKCON**

**Problem:** The processor sends the PCIe Set\_Slot\_Power\_Limit message on writes to the Slot Capabilities (SLTCAP Devices 0-3; Functions 0-3; Offset A4H) register. Due to this erratum, the processor also sends PCIe the Set\_Slot\_Power\_Limit message on writes to the LNKCON (CPUBUS(0); Devices 0-3; Functions 0-3; Offset A0H) register.

**Implication:** For those monitoring the PCIe\* traffic going across the link, the unexpected PCIe Set\_Slot\_Power\_Limit Message will be detected whenever a write to the LNKCON register occurs. Intel has not observed any functional failures due to this erratum on any commercially available system.

**Workaround:** None identified.

**Status:** For the steppings affected, see the Summary Tables of Changes.

### **BS64. Erratum Removed**

### **BS65. Locked Accesses Spanning Cachelines That Include PCI Space May Lead to a System Hang**

**Problem:** A locked memory access which splits across a cacheline boundary that suffers a master abort on a PCI bus may lead to a system hang.

**Implication:** Aborted split lock accesses may cause PCI devices to become inoperable until a platform reset. Intel has not observed this erratum with commercially available software.

**Workaround:** None identified.

**Status:** For the steppings affected, see the Summary Tables of Changes.

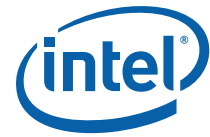
### **BS66. Cold Boot May Fail Due to Internal Timer Error**

**Problem:** The processors may not complete a cold boot (i.e. a boot from a power-off state) due to an internal timer error machine check, IA32\_MCi\_STATUS.MCACOD of 0000\_0100\_0000\_0000. This will result in the processor asserting IERR (Internal Error).

**Implication:** The processor may signal IERR during a cold boot when the system is initializing.

**Workaround:** It is possible for the BIOS to contain a workaround for this erratum.

**Status:** For the steppings affected, see the Summary Tables of Changes.



### **BS67. PCIe\* Rx Common Mode Return Loss is Not Meeting The Specification**

**Problem:** The PCIe Specification requires that the Rx Common Mode Return Loss in the range of 0.05 to 2.5 GHz must be limited to -6 dB. The processor's PCIe Rx do not meet this requirement. The PCIe Rx Common Mode Return at 500MHz has been found to be between -3.5 and -4 dB on a limited number of samples.

**Implication:** Intel has not observed any functional failures due to this erratum with any commercially available PCIe devices.

**Workaround:** None identified.

**Status:** For the steppings affected, see the Summary Tables of Changes.

### **BS68. The Most Significant Bit of the CEC Cannot be Cleared Once Set**

**Problem:** The most significant bit of the CEC (Corrected Error Count IA32\_MCi\_STATUS (i=12-19), bit 52) cannot be cleared once it has been set.

**Implication:** In the case that software attempts to clear the CEC and the count exceeds 3FFFH, software will read incorrect CEC values on subsequent accesses and additional CMCIs (Corrected Machine Check Error Interrupts) will not be generated.

**Workaround:** None identified. Software can avoid this erratum by setting corrected error threshold to a value less than 3FFFH, enable CMCI and clearing the error count before it exceeds 3FFFH.

**Status:** For the steppings affected, see the Summary Tables of Changes.

### **BS69. PCIe\* Adaptive Equalization May Not Train to the Optimal Settings**

**Problem:** In the case of the PCIe equalization procedure for 8 GT/s, the Downstream Port's (e.g. the processor's) TXEQ (transmitter equalization settings) can be fine tuned for each Lane during a process called Adaptive Equalization Phase 3. Due to this erratum, the processor may not direct the end-agent to the optimal TXEQ settings.

**Implication:** The PCIe link may not be as robust as possible potentially leading to a higher bit error rate than expected.

**Workaround:** A BIOS code change has been identified and may be implemented as a workaround for this erratum.

**Status:** For the steppings affected, see the Summary Tables of Changes.

### **BS70. A Core May Not Complete Transactions to The Caching Agent When C-States Are Enabled Leading to an Internal Timer Error**

**Problem:** When multiple cores have outstanding transactions targeted to a single caching agent and one of the cores enters a Core C-state before completing the transaction with the targeted caching agent an internal timer machine check error may occur (IA32\_MCi\_STATUS.MCACOD of 0000\_0100\_0000\_0000).

**Implication:** Due to this erratum, the processor may experience an internal timer error.

**Workaround:** None identified.

**Status:** For the steppings affected, see the Summary Tables of Changes.

### **BS71. TSC is Not Affected by Warm Reset**

**Problem:** The TSC (Time Stamp Counter MSR 10H) should be cleared on reset. Due to this erratum the TSC is not affected by warm reset.

**Implication:** The TSC is not cleared by a warm reset. The TSC is cleared by power-on reset as expected. Intel has not observed any functional failures due to this erratum.

**Workaround:** None identified.

**Status:** For the steppings affected, see the Summary Tables of Changes.



## **BS72. Warm Resets May be Converted to Power-on Resets When Recovering From an IERR**

**Problem:** When a warm reset is attempted and an IERR (Internal Error) happens as indicated by the IA32\_MCi\_STATUS.MCACOD of 0000\_0100\_0000\_0000, a power-on reset occurs instead.

**Implication:** The values in the machine check bank will be lost as a result of the power-on reset. This prevents a OS, BIOS or the BMC (Baseboard Management Controller) from logging the content of the error registers or taking any post-reset actions that are dependent on the machine check information.

**Workaround:** It is possible for the BIOS to contain a workaround for this erratum.

**Status:** For the steppings affected, see the Summary Tables of Changes.

## **BS73. Erratum Removed**

## **BS74. Processor May not Restore the VR12 DDR3 Voltage Regulator Phases upon Pkg C3 State Exit**

**Problem:** During the Pkg (Package) C3 state entry, the processor directs the VR12 DDR3 voltage regulators to shed phases to reduce power consumption. Due to this erratum, the processor may not restore all VR12 DDR3 voltage regulator phases upon Pkg C3 state exit. The VR12 DDR3 voltage regulators require all phases to keep the DDR3 voltage plane in tolerance for proper memory subsystem functioning during normal system operation.

**Implication:** Due to this erratum, unpredictable system behavior may occur.

**Workaround:** It is possible for the BIOS to contain a workaround for this erratum.

**Status:** For the steppings affected, see the Summary Tables of Changes.

## **BS75. The Equalization Phase Successful Bits Are Not Compliant to The PCIe\* Specification**

**Problem:** PCIe Specification states that if the Phase 1 of Transmitter Equalization completes successfully as indicated by the LNKSTS2.Equalization Phase 1 Successful (Devices 0-3; Functions 0-3; bit[2]) bit being set to one and if the Phase 2 and 3 link training phases are bypassed, the LNKSTS2.Equalization Phase 3 Successful (Devices 0-3; Functions 0-3; bit[4]) and LNKSTS2.Equalization Phase 2 Successful (bit[3]) bits should be set to one. Due to this erratum, the processor will only set the Equalization Phase 2 or 3 Successful bits if the phases are completed successfully.

**Implication:** Due to this erratum, Equalization Phase 2 and 3 Successful bits may not be set. Intel has not observed any functional failure with commercially available PCIe devices.

**Workaround:** None identified.

**Status:** For the steppings affected, see the Summary Tables of Changes.

## **BS76. Using DMA XOR With DCA May Cause a Machine Check**

**Problem:** If both DCA (direct cache access) and DMA XOR operations are active at the same time, then invalid prefetch hints may be generated. These prefetch transactions may not complete and could result in a timeout machine check, which will cause CATERR# to become asserted.

**Implication:** Invalid prefetch hints may not complete resulting in a machine check.

**Workaround:** If using DMA XOR operations, disable DCA by clearing CHANCTRL.Completion\_Write\_DCA\_Enable (Offset 80H; Bit 9) in the region described by CB\_BAR (Device: 10; Function 0-7; Offset 80H).

**Status:** For the steppings affected, see the Summary Tables of Changes.



**BS77. Mixed DMA XOR and Legacy Operations in The Same Channel May Cause Data to be Observed Out of Order**

**Problem:** For mixed channel DMA (XOR and legacy operations active on the same channel) completion writes from legacy operations may pass completion writes from XOR operations resulting in out of order descriptor updates/completions.

**Implication:** DMA descriptor progress may appear out of order with incorrect data.

**Workaround:** In the DMA driver each DMA XOR descriptor must be followed by an additional legacy descriptor. The legacy descriptor must have a non-zero transfer length and the "NULL Transfer" bit and "Completion Interrupt" in the Descriptor Control field set to '1'. The transfer will not actually occur, but a completion interrupt will be generated that indicates that the XOR operation has completed. This causes all completion interrupts to be of the legacy type.

**Status:** For the steppings affected, see the Summary Tables of Changes.

**BS78. Unexpected DMA XOR Halt and Errors When Using Descriptors With P or Q Operations Disabled**

**Problem:** If a Galois Field Generate/Validate base descriptor has either the P Operations Disable or Q Operation Disable bit set and the corresponding disabled P Parity Address or Q Parity Address field of the descriptor does not contain a valid/aligned address, the DMA channel may halt unexpectedly with destination address errors. The destination address errors will be logged in CHANERR\_INT. DMA Transfer Destination Address Error (Device 4; Function 0-7; Offset 180H; Bit 1).

**Implication:** The DMA may only partially process a DMA XOR descriptor when a disabled P or Q Parity Address field of the descriptor does not contain a valid/aligned address, resulting in incomplete data, an unexpected DMA channel halt and destination address errors.

**Workaround:** At all times, software must place a valid/aligned address in both the P Parity Address field and the Q Parity Address field of a DMA XOR with Galois Field Generate/Validate base descriptor even if the P Operations Disable or Q Operations Disable descriptor fields are set to disable either P or Q operations for the descriptor.

**Status:** For the steppings affected, see the Summary Tables of Changes.

**BS79. DMA XOR Channel May Hang on Source Read Completion Data Parity Error For >8K Descriptors**

**Problem:** If a parity error occurs of source read completion data while inside the DMA for >8K descriptor transfer lengths, the DMA channel will hang until the next platform reset. This behavior only applies if the data arrived at the DMA unit error free (from DRAM and QPI) but then had a parity error in the completion data FIFO inside the DMA.

**Implication:** The effected DMA channel will hang until the next platform reset.

**Workaround:** None.

**Status:** For the steppings affected, see the Summary Tables of Changes.



## **BS80. DMA CB\_BAR Decode May be Incorrect After DMA FLR**

**Problem:** PCIe FLR (function level reset) of the DMA function, may result in an incorrect CB\_BAR (Device 4; Function 0-7; Offset 10h) decode when a memory read of the CB\_BAR occurs around the same time as the FLR.

**Implication:** A FLR may cause a PCIe memory read to decode to channel 0 instead of the intended channel resulting in incorrect read data returned.

**Workaround:** Software must quiesce the DMA function before issuing FLR including:

- Ensure clients are no longer referencing the driver.
- Ensure all outstanding descriptors have completed via the normal completion writeback notifications by reading CHANCMP, CHANSTS, and DMACOUNT.
- Issue FLR and ensure no new DMA transactions are started until FLR has completed.

CHANCMP (Offset 98H) and CHANSTS (Offset 88H), and DMACOUNT (Offset 86H) are offsets relative to CB\_BAR on the processor's internal IO bus (as defined in the IIOBUSNO register).

**Status:** For the steppings affected, see the Summary Tables of Changes.

## **BS81. XOR DMA Restricted to <=8KB Transfers When Multiple Channels Are in use**

**Problem:** Incorrect data transfers can occur if more than one DMA channel is in operation and >8KB XOR DMA transfer sizes are being used. XOR DMA transfer size is set by software in the Block Size field of the XOR with Galios Field Generate/Validate base descriptor.

**Implication:** XOR DMA operation is restricted to <=8KB transfer sizes when multiple DMA channels are in use. Legacy DMA operations may still use up to the maximum 1MB transfer length.

**Workaround:** Software may either:

- Use a single DMA channel for both legacy and XOR operation types both up to the maximum 1MB transfer size.
- Use multiple DMA channels where XOR operation types are <=8KB transfer size and legacy operation types are up to 1MB transfer size.

**Status:** For the steppings affected, see the Summary Tables of Changes.

## **BS82. Unable to Restart DMA After Poisoned Error During an XOR Operation**

**Problem:** If the CHANERR field Read Data Error (Offset A8H; Bit 8) is set due to a poisoned completion error during a DMA XOR operation, the DMA stays in the halted state and the Read Data Error bit does not clear

**Implication:** The XOR operations on the DMA can not be restarted after a read data error due to a poisoned XOR operation.

**Workaround:** At least one XOR descriptor with no read data errors has to be processed for a new chain of XOR descriptors to work correctly with the corresponding CHANERRMSK (Offset ACH; Bit 8) bit set. Upon detection of a Read Data Error, software must clear the CHANERR and CHANERR\_INT (Device 4; Function 0-7; Offset 180H) registers and disable the corresponding error mask bit by setting CHANERRMSK. Then new descriptors can be added to the chain and the DMA started by writing the DMACOUNT (Offset 86H). Once the DMA channel is in the running state, software can clear the CHANERRMSK. CHANERR, CHANERRMSK, and DMACOUNT are offsets relative to CB\_BAR (Device 4; Function 0-7; Offset 10H) on the processors internal IO bus (as defined in the IIOBUSNO register).

**Status:** For the steppings affected, see the Summary Tables of Changes.



**BS83. DMA Restart Hang When First Descriptor is a Legacy Type Following Channel HALT Due to an Extended Descriptor Error**

**Problem:** When using multiple DMA channels, all DMA channels may hang if a DMA channel restart is attempted with a Legacy descriptor as the first descriptor following an error/HALT on an Extended descriptor on Channel 0 or 1.

**Implication:** Following an extended descriptor error on Channel 0 or 1, the channel must be not be restarted with a first descriptor of legacy type including NULL. Does not apply for single channel operation.

**Workaround:** Software must guarantee that the first descriptor processed on restart is an XOR GF Multiply Generation (base type) before using legacy descriptors with interrupts and completions.

**Status:** For the steppings affected, see the Summary Tables of Changes.

**BS84. JSP CBDMA errata BF508S: Operation With DMA XOR Interrupts/ Completions Enabled Restricted to Channel 0 and 1**

**Problem:** If DMA XOR interrupts and completions are enabled on channel 0 or 1 concurrent with operation on channels 2-7, incorrect data transfers can occur on DMA channels 2-7. DMA XOR interrupts and completions are enabled by setting bits 0 and 3 of descriptor control field of a DMA XOR with Galios Field Generate/Validate base descriptor.

**Implication:** If DMA XOR interrupts and completions are enabled, only one interrupt/completion type may be used on any single channel and only channels 0 and 1 may be used.

**Workaround:** Software must either:

- Only use only legacy interrupts and completions on all channels.
- Use only DMA channels 0 and 1 where:
  - Only DMA XOR interrupts/completions are enabled on channel 0 and is only used for DMA XOR operations.
  - Only legacy interrupts/completions are enabled on channel 1 and is only used for DMA legacy operations.

**Status:** For the steppings affected, see the Summary Tables of Changes.

**BS85. Suspending/Resetting an Active DMA XOR Channel May Cause an Incorrect Data Transfer on Other Active Channels**

**Problem:** Suspending an active DMA XOR channel by setting CHANCMD.Suspend DMA bit (Offset 84; Bit 2) while XOR type DMA channels are active may cause incorrect data transfer on the other active legacy channels. This erratum may also occur while resetting an active DMA XOR channel CHANCMD.Reset DMA bit (Offset 84; Bit 5). CHANCMD is in the region described by CB\_BAR(Device 4; function 0-7; Offset 10H) on the processor's internal IO bus (as defined in the IIOBUSNO register).

**Implication:** An incorrect data transfer may occur on the active legacy DMA channels.

**Workaround:** Software must suspend all legacy DMA channels before suspending an active DMA XOR channel (channel 0 or 1).

**Status:** For the steppings affected, see the Summary Tables of Changes.



### **BS86.      DWORD-Aligned DMA XOR Descriptors With Fencing And Multi-Channel Operation May Cause a Channel Hang**

**Problem:** DMA XOR descriptors with DWORD aligned sources and fencing enabled may result in a XOR channel hang until the next platform reset. XOR DMA fencing is set by software in Descriptor Control.Fence (XOR base descriptor, bit 4)

**Implication:** An XOR DMA descriptor with non cacheline aligned sources may hang until the next platform reset.

**Workaround:** Do not enable fencing on XOR descriptors. Fencing can be enabled on legacy descriptors. It is recommended that a NULL legacy descriptor must be paired with each XOR descriptor. Software can use fencing of the legacy NULL descriptor to track full completion of its associated XOR descriptor

**Status:** For the steppings affected, see the Summary Tables of Changes.

### **BS87.      Storage of PEBS Record Delayed Following Execution of MOV SS or STI**

**Problem:** When a performance monitoring counter is configured for PEBS (Precise Event Based Sampling), overflow of the counter results in storage of a PEBS record in the PEBS buffer. The information in the PEBS record represents the state of the next instruction to be executed following the counter overflow. Due to this erratum, if the counter overflow occurs after execution of either MOV SS or STI, storage of the PEBS record is delayed by one instruction.

**Implication:** When this erratum occurs, software may observe storage of the PEBS record being delayed by one instruction following execution of MOV SS or STI. The state information in the PEBS record will also reflect the one instruction delay.

**Workaround:** None identified.

**Status:** For the steppings affected, see the Summary Tables of Changes.

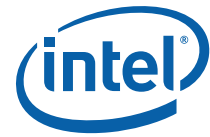
### **BS88.      Instruction Fetch May Cause Machine Check if Page Size and Memory Type Was Changed Without Invalidation**

**Problem:** This erratum may cause a machine-check error (IA32\_MCi\_STATUS.MCACOD=0150H) on the fetch of an instruction that crosses a 4-KByte address boundary. It applies only if (1) the 4-KByte linear region on which the instruction begins is originally translated using a 4-KByte page with the WB memory type; (2) the paging structures are later modified so that linear region is translated using a large page (2-MByte, 4-MByte, or 1-GByte) with the UC memory type; and (3) the instruction fetch occurs after the paging-structure modification but before software invalidates any TLB entries for the linear region.

**Implication:** Due to this erratum an unexpected machine check with error code 0150H may occur, possibly resulting in a shutdown. Intel has not observed this erratum with any commercially available software.

**Workaround:** Software should not write to a paging-structure entry in a way that would change, for any linear address, both the page size and the memory type. It can instead use the following algorithm: first clear the P flag in the relevant paging-structure entry (e.g., PDE); then invalidate any translations for the affected linear addresses; and then modify the relevant paging-structure entry to set the P flag and establish the new page size and memory type.

**Status:** For the steppings affected, see the Summary Tables of Changes.



### **BS89. PCIe\* Adaptive Equalization May Not Train to the Optimal Settings**

**Problem:** In the case of the PCIe equalization procedure for 8 GT/s, the Downstream Port's (e.g. the processor's) TXEQ (transmitter equalization settings) can be fine tuned for each Lane during a process called Adaptive Equalization Phase 3. Due to this erratum, the processor may not direct the end-agent to the optimal TXEQ settings.

**Implication:** The PCIe link may not be as robust as possible potentially leading to a higher bit error rate than expected.

**Workaround:** A BIOS code change has been identified and may be implemented as a workaround for this erratum.

**Status:** For the steppings affected, see the Summary Tables of Changes.

### **BS90. The VT-d Queued Invalidation Status Write May Fail**

**Problem:** Intel® Virtualization Technology for Directed I/O (Intel® VT-d) queued invalidation operations issue a status write to modify a semaphore. Due to this erratum, the status write may fail.

**Implication:** When using queued invalidation operations, a failed status write can result in unpredictable system behavior.

**Workaround:** If operating without queued invalidations, interrupt re-mapping, and X2APIC features is feasible, then VT-d invalidations should be performed using the VT-d register facility (c.f., VTD0\_CTXCMD [offset 028h], VTD1\_CTXCMD [offset 1028h], VTD0\_INVADDRREG [offset 0200h] and VTD0\_IOTLBINV [offset 0208h], VTD1\_INVADDRREG [offset 1200h] and VTD1\_IOTLBINV [offset 1208h] in the VT-d register region with a base address specified through the VTBAR register at 0:5:0, offset 0180h). If those operational limitations are not feasible, disable VT-d through BIOS facilities. This will prevent the use of Intel VT-d, including X2APIC and TXT facilities that are dependent on Intel VT-d.

**Status:** For the affected steppings, see the Summary Tables of Changes.

### **BS91. Executing The GETSEC Instruction While Throttling May Result in a Processor Hang**

**Problem:** If the processor throttles due to either high temperature thermal conditions or due to an explicit operating system throttling request (TT1) while executing GETSEC[SENTER] or GETSEC[SEXIT] instructions, then under certain circumstances, the processor may hang. Intel has not been observed this erratum with any commercially available software.

**Implication:** Possible hang during execution of GETSEC instruction.

**Workaround:** None Identified.

**Status:** For the steppings affected, see the Summary Tables of Changes.

### **BS92. Platform Idle Power Higher May be Higher Than Expected**

**Problem:** The processor may not place the associated DRAM subsystem in the lowest allowed power state during Package C3 and Package C6 states. This may cause the platform idle power to be higher than expected.

**Implication:** Platform average power and idle power may be higher than expected.

**Workaround:** It is possible for the BIOS to contain a workaround for this erratum.

**Status:** For the steppings affected, see the Summary Tables of Changes.



**BS93.      **PECI Transactions during an S-State Transition May Result in a Platform Cold Reset****

**Problem:** Due to this erratum, a PECI transaction during an S-state transition may result in an unexpected platform cold reset rather than an S-state transition.

**Implication:** Use of PECI transactions during an S-state transition can result in a platform reset that terminates transitioning to the desired S-state.

**Workaround:** It is possible for the BIOS to contain a workaround for this erratum.

**Status:** For the steppings affected, see the Summary Tables of Changes.

**BS94.      **Complex Platform Conditions during a Transition to S4 or S5 State May Result in an Internal Timeout Error****

**Problem:** Due to this erratum, the BIOS sequencing associated with S4 (sometimes known as "Hibernate") and S5 (also known as "Soft Off"), when undertaken with certain complex platform conditions, can result in an internal timeout error as indicated by IA32\_MCI\_STATUS.MCACOD of 0000\_0100\_0000\_0000 and IERR assertion. This internal timeout error stops the platform S-state sequencing before platform power down occurs. Certain platforms may have logic that, upon detection of the failure to reach power down, initiates a cold reset sequence.

**Implication:** S4 state or S5 state may not be reliably entered; the platform may not reach the very low power condition.

**Workaround:** It is possible for the BIOS to contain a workaround for this erratum.

**Status:** For the steppings affected, see the Summary Tables of Changes.



# Specification Changes

---

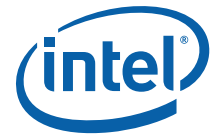
There are no Specification Changes at this time.



# Specification Clarifications

---

There are no Specification Clarifications at this time.



# Documentation Changes

---

There are no Documentation Changes at this time.

§ §

