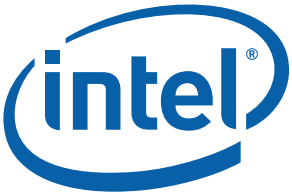


Intel® Xeon® Processor 7500 Series

Specification Update

March 2015

Reference Number: 323344-022

Intel technologies features and benefits depend on system configuration and may require enabled hardware, software, or service activation. Learn more at intel.com, or from the OEM or retailer.

No computer system can be absolutely secure. Intel does not assume any liability for lost or stolen data or systems or any damages resulting from such losses.

You may not use or facilitate the use of this document in connection with any infringement or other legal analysis concerning Intel products described herein. You agree to grant Intel a non-exclusive, royalty-free license to any patent claim thereafter drafted which includes subject matter disclosed herein.

No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document.

The products described 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.

This document contains information on products, services and/or processes in development. All information provided here is subject to change without notice. Contact your Intel representative to obtain the latest Intel product specifications and roadmaps.

Intel disclaims all express and implied warranties, including without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement, as well as any warranty arising from course of performance, course of dealing, or usage in trade.

Intel is a sponsor and member of the Benchmark XPRT Development Community, and was the major developer of the XPRT family of benchmarks. Principled Technologies is the publisher of the XPRT family of benchmarks. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases.

Copies of documents which have an order number and are referenced in this document may be obtained by calling 1-800-548-4725 or by visiting www.intel.com/design/literature.htm.

Intel, the Intel logo, and Xeon are trademarks of Intel Corporation in the U.S. and/or other countries.

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

Contents

§

Revision History

Revision	Description	Date
-001	Public Release	March 2010
-002	Added Specification Change BA1; Added Errata BA91, BA92, BA93, BA94, BA95	June 2010
-003	Added Specification Change BA2; Added Errata BA96, BA97, BA98	July 2010
-004	Removed Specification Change BA1; Added Errata BA99, BA100, BA101, BA102, BA103	September 2010
-005	Added Errata BA104, BA105, BA106, BA107, BA108, BA109, BA110	November 2010
-006	Added Errata BA111, BA112, BA113, BA114, BA115, BA116, Updated Erratum BA65	February 2011
-007	Added erratum BA117 and BA118	July 2011
-008	BA113 Workaround, SDM volume reference correction	October 2011
-009	Added errata BA119 through BA125	February 2012
-010	Added erratum BA126	April 2012
-011	Added errata BA127 through BA131	May 2012
-012	Added erratum BA132	June 2012
-013	Added erratum BA133	September 2012
-014	Added documentation change DC1	January 2013
-015	Added erratum BA134	May 2013
-016	Added errata BA135 and BA136	June 2013
-017	Added errata BA137	July 2013
-018	Added errata BA138	August 2013
-019	Added errata BA139	March 2014
-020	Removed erratum BA83 Replaced erratum BA69 with BA140	November 2014
-021	Updated erratum BA134	February 2015
-022	Added erratum BA141	March 2015

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 errata, 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
Intel® Xeon® Processor 7500 Series Datasheet Volume 1	323340
Intel® Xeon® Processor 7500 Series Datasheet Volume 2	323341

Related Documents

Document Title	Document Number / Location
Intel® 64 and IA-32 Architectures Software Developer's Manual • Volume 1: Basic Architecture • Volume 2A: Instruction Set Reference Manual A-M • Volume 2B: Instruction Set Reference Manual N-Z • Volume 3A: System Programming Guide • Volume 3B: System Programming Guide	http://www.intel.com/products/processor/manuals/index.htm
Intel® 64 and IA-32 Intel Architecture Optimization Reference Manual	248966
Intel® 64 and IA-32 Architectures Software Developer's Manual Documentation Changes	252046

Nomenclature

Errata are design defects or errors. These may cause the Intel® Xeon® Processor 7500 Series behavior to deviate from published specifications. Hardware and software designed to be used with any given stepping must assume that all errata documented for that stepping are present on all devices.

S-Spec Number is a five-digit code used to identify products. Products are differentiated by their unique characteristics,e.g., 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.

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: Errata remain in the specification update throughout the product's lifecycle, or until a particular stepping is no longer commercially available. Under these circumstances, errata removed from the specification update are archived and available upon request. 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, and so forth).

Summary Tables of Changes

The following tables indicate the errata, specification changes, specification clarifications, or documentation changes which apply to the Intel® Xeon® Processor 7500 series product. 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. These tables uses the following notations:

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 table row indicates this erratum is either new or modified from the previous version of the document.

Each Specification Update item is prefixed with a capital letter to distinguish the product. The key below details the letters that are used in Intel's microprocessor Specification Updates:

Table 1. Errata Table (Sheet 1 of 5)

Number	Stepping	Status	Description
	D-0		
BA1.	X	No Fix	Intel® Interconnect BIST (Intel® IBIST) Does Not Work in Intel® QPI in Slow Mode
BA2.	X	No Fix	Retraining Parameter Negotiation is Not Implemented for Intel® QPI
BA3.	X	No Fix	Intel® IBIST Slave Ignores Loop Count Values Sent by Master on Intel® QPI
BA4.	X	No Fix	System Hangs when Skipping Stop Req2 and Start Req1 Messages in Quiesce/Lock Sequence
BA5.	X	No Fix	Integrated Memory Controller Signals Spurious CMCI when Home Agent Failover Count Saturation Occurs
BA6.	X	No Fix	Memory Controller Does Not Set S Bit for Uncorrectable Error Followed by Software Recoverable Error
BA7.	X	No Fix	Running SDDC+1b with NB CRC Error May Result in Incorrect Data
BA8.	X	No Fix	MCi_STATUS S Bit Not Set for LLC Software Recoverable Errors
BA9.	X	No Fix	Correctable SB CRC Error May be Propagated to an Uncorrected ECC Error
BA10.	X	No Fix	Memory Controller Patrol Scrub Ceases to Function with CRC Errors and the IMT31 Reclaim Feature Enabled
BA11.	X	No Fix	False Patrol Read Error Logging May Occur with Patrol Scrub Enabled and with SB CRC Errors
BA12.	X	No Fix	Electrically Idle Intel SMI and Intel® QPI Lanes May Deliver Data that May Look Like Deskew Headers
BA13.	X	No Fix	A Sequence of Instruction Fetches and Snoops to Locked Cache Lines May Cause Processor to Hang
BA14.	X	No Fix	Writing to Unimplemented Bits of UU_CR_U_MSR_PMON_EVNT_SEL MSR does Not Result in #GP Fault
BA15.	X	No Fix	Mixed Rank Size Memory Configurations May Cause a Missing Refresh Event
BA16.	X	No Fix	Mirror Slave May Deliver Incorrect Data when a Read to the Mirror Master Completes Before the Write-back from the IOH
BA17.	X	No Fix	UU_CR_U_MSR_PMON_GLOBAL_OVF_CTL MSR Does Not Follow RW1C Access Method
BA18.	X	No Fix	HNID Field is Incorrect for CMP Messages From PrefetchHint
BA19.	X	No Fix	Processor Internal Initialization Code May become Damaged Under Certain Conditions
BA20.	X	No Fix	MCi_Status Overflow Bit May Be Incorrectly Set on a Single Instance of a DTLB Error
BA21.	X	No Fix	Debug Exception Flags DR6.B0-B3 Flags May be Incorrect for Disabled Breakpoints
BA22.	X	No Fix	MONITOR or CLFLUSH on the Local XAPIC's Address Space Results in Hang
BA23.	X	No Fix	Corruption of CS Segment Register During RSM While Transitioning From Real Mode to Protected Mode
BA24.	X	No Fix	The Processor May Report a #TS Instead of a #GP Fault
BA25.	X	No Fix	REP MOVS/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
BA26.	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
BA27.	X	No Fix	Performance Monitor SSE Retired Instructions May Return Incorrect Values
BA28.	X	No Fix	Premature Execution of a Load Operation Prior to Exception Handler Invocation
BA29.	X	No Fix	MOV To/From Debug Registers Causes Debug Exception
BA30.	X	No Fix	Incorrect Address Computed For Last Byte of FXSAVE/FXRSTOR Image Leads to Partial Memory Update
BA31.	X	No Fix	Values for LBR/BTS/BTM will be Incorrect after an Exit from SMM

Table 1. Errata Table (Sheet 2 of 5)

Number	Stepping	Status	Description
	D-0		
BA32.	X	No Fix	Single Step Interrupts with Floating Point Exception Pending May Be Mishandled
BA33.	X	No Fix	Fault on ENTER Instruction May Result in Unexpected Values on Stack Frame
BA34.	X	No Fix	IRET under Certain Conditions May Cause an Unexpected Alignment Check Exception
BA35.	X	No Fix	General Protection Fault (#GP) for Instructions Greater than 15 Bytes May be Preempted
BA36.	X	No Fix	General Protection (#GP) Fault May Not Be Signaled on Data Segment Limit Violation above 4-G Limit
BA37.	X	No Fix	LBR, BTS, BTM May Report a Wrong Address when an Exception/Interrupt Occurs in 64-bit Mode
BA38.	X	No Fix	A VM Exit on MWAIT May Incorrectly Report the Monitoring Hardware as Armed
BA39.	X	No Fix	Performance Monitor Event SEGMENT_REG_LOADS Counts Inaccurately
BA40.	X	No Fix	#GP on Segment Selector Descriptor that Straddles Canonical Boundary May Not Provide Correct Exception Error Code
BA41.	X	No Fix	Improper Parity Error Signaled in the IQ Following Reset When a Code Breakpoint is Set on a #GP Instruction
BA42.	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
BA43.	X	No Fix	IA32_MPERF Counter Stops Counting During On-Demand TM1
BA44.	X	No Fix	Synchronous Reset of IA32_APERF/IA32_MPERF Counters on Overflow Does Not Work
BA45.	X	No Fix	Disabling Thermal Monitor While Processor is Hot, Then Re-enabling, May Result in Stuck Core Operating Ratio
BA46.	X	No Fix	OVER Bit for IA32_MCi_STATUS Register May Get Set on Specific Internal Error
BA47.	X	No Fix	Writing the Local Vector Table (LVT) when an Interrupt is Pending May Cause an Unexpected Interrupt
BA48.	X	No Fix	Reading Reserved APIC Registers May Not Signal an APIC Error
BA49.	X	No Fix	Faulting MMX Instruction May Incorrectly Update x87 FPU Tag Word
BA50.	X	No Fix	Certain Store Parity Errors May Not Log Correct Address in IA32_MCi_ADDR
BA51.	X	No Fix	xAPIC Timer May Decrement Too Quickly Following an Automatic Reload While in Periodic Mode
BA52.	X	No Fix	Certain Undefined Opcodes Crossing a Segment Limit May Result in #UD Instead of #GP Exception
BA53.	X	No Fix	Reported Memory Type May Not Be Used to Access the VMCS and Referenced Data Structures
BA54.	X	No Fix	MOVNTDQA From WC Memory May Pass Earlier Locked Instructions
BA55.	X	No Fix	Changing the Memory Type for an In-Use Page Translation May Lead to Memory-Ordering Violations
BA56.	X	No Fix	Delivery of Certain Events Immediately Following a VM Exit May Push a Corrupted RIP Onto The Stack
BA57.	X	No Fix	The Combination of a Bus Lock and a Data Access That is Split Across Page Boundaries May Lead to Processor Livelock
BA58.	X	No Fix	An Unexpected Page Fault May Occur Following the Unmapping and Remapping of a Page
BA59.	X	No Fix	Two xAPIC Timer Event Interrupts May Unexpectedly Occur
BA60.	X	No Fix	FREEZE_WHILE_SMM Does Not Prevent Event From Pending PEBS During SMM
BA61.	X	No Fix	PEBS Records For Load Latency Monitoring May Contain an Incorrect Linear Address
BA62.	X	No Fix	PEBS Field “Data Linear Address” is Not Sign Extended to 64 Bits
BA63.	X	No Fix	APIC Error “Received Illegal Vector” May be Lost

Table 1. Errata Table (Sheet 3 of 5)

Number	Stepping	Status	Description
	D-0		
BA64.	X	No Fix	CPUID Incorrectly Indicates the UnHalted Reference Cycle Architectural Event is Supported
BA65.	X	No Fix	DR6.B0-B3 May Not Report All Breakpoints Matched When a MOV/POP SS is Followed by a Store Instruction
BA66.	X	No Fix	An Uncorrectable Error Logged in IA32_CR_MC2_STATUS May also Result in a System Hang
BA67.	X	No Fix	IA32_PERF_GLOBAL_CTRL MSR May be Incorrectly Initialized
BA68.	X	No Fix	Processors with SMT May Hang on P-State Transition or ACPI Clock Modulation Throttling
BA69.	X	No Fix	Replaced with BA140
BA70.	X	No Fix	Sleeping Cores May Not be Woken Up on Logical Cluster Mode Broadcast IPI Using Destination Field Instead of Shorthand
BA71.	X	No Fix	Faulting Executions of FXRSTOR May Update State Inconsistently
BA72.	X	No Fix	Performance Monitor Counters May Count Incorrectly
BA73.	X	No Fix	Performance Monitor Event Offcore_response_0 (B7H) Does Not Count NT Stores to Local DRAM Correctly
BA74.	X	No Fix	EFLAGS Discrepancy on Page Faults and on EPT-Induced VM Exits after a Translation Change
BA75.	X	No Fix	LER and LBR MSRs May Be Incorrectly Updated During a Task Switch
BA76.	X	No Fix	Back to Back Uncorrected Machine Check Errors May Overwrite IA32_MC3_STATUS.MSCOD
BA77.	X	No Fix	Corrected Errors With a Yellow Error Indication May be Overwritten by Other Corrected Errors
BA78.	X	No Fix	A String Instruction that Remaps a Page May Encounter an Unexpected Page Fault
BA79.	X	No Fix	Performance Monitor Events DCACHE_CACHE_LD and DCACHE_CACHE_ST May Overcount
BA80.	X	No Fix	Rapid Core C3 Transition May Cause Unpredictable System Behavior
BA81.	X	No Fix	Performance Monitor Events INSTR_RETIRED and MEM_INST_RETIRED May Count Inaccurately
BA82.	X	No Fix	A Page Fault May Not be Generated When the PS bit is set to “1” in a PML4E or PDPTE
BA83.	X	No Fix	Erratum removed
BA84.	X	No Fix	Storage of PEBS Record Delayed Following Execution of MOV SS or STI
BA85.	X	No Fix	Performance Monitoring Events STORE_BLOCKS.NOT_STA and STORE_BLOCKS.STA May Not Count Events Correctly
BA86.	X	No Fix	VMX-Preemption Timer Does Not Count Down at the Rate Specified
BA87.	X	No Fix	VM Exits Due to LIDT/LGDT/SIDT/SGDT Do Not Report Correct Operand Size
BA88.	X	No Fix	Multiple Performance Monitor Interrupts are Possible on Overflow of IA32_FIXED_CTR2
BA89.	X	No Fix	VM Exits Due to “NMI-Window Exiting” May Be Delayed by One Instruction
BA90.	X	No Fix	LBRs May Not be Initialized During Power-On Reset of the Processor
BA94.	X	No Fix	Single Bit Error Will Become an Uncorrectable Error if it Occurs on Specific Bits Under SDDC
BA92.	X	No Fix	Power Controller Performance Monitor Counters May Not Operate Correctly
BA93.	X	No Fix	Intel® QPI HNID Field is Incorrect for CMP Messages From PrefetchHint
BA94.	X	No Fix	System Configuration Controller Misaligned Error May Result in a System Hang
BA95.	X	No Fix	Recoverable Errors Signaled From Intel® QPI or Intel® SMI Port to the System Configuration Controller May Get Lost if the Ports are Disabled
BA96.	X	No Fix	Performance Monitor Events for Hardware Prefetches Which Miss The L1 Data Cache May be Over Counted
BA97.	X	No Fix	VM Exit May Incorrectly Clear IA32_PERF_GLOBAL_CTRL [34:32]
BA98.	X	No Fix	Direct Connect Flash ROM May Become Overwritten

Table 1. Errata Table (Sheet 4 of 5)

Number	Stepping	Status	Description
	D-0		
BA99.	X	No Fix	Memory Controller Does Not Handle MCA Overwrite Rules for Software Recoverable Errors Correctly
BA100.	X	No Fix	Intel® QPI REUTPHRDS Register Only Records Bad Lanes
BA101.	X	No Fix	Write-1-to-clear Does Not Work for Two Intel ® IBIST Registers when Clock Gating is Enabled
BA102.	X	No Fix	Memory Aliasing of Code Pages May Cause Unpredictable System Behavior
BA103.	X	No Fix	Performance Monitor Event EPT.EPDPE_MISS May be Counted While EPT is Disabled
BA104.	X	No Fix	PECI Command GET_TEMP does not take into account Uncore Temperature
BA105.	X	No Fix	Intel QPI Lane May Be Dropped During Full Frequency Deskew Phase of Training
BA106.	X	No Fix	EOI Transaction May Not be Sent if Software Enters Core C6 During an Interrupt Service Routine
BA107.	X	No Fix	Intel® QPI and SMI Links Do Not Meet the VTx-cm-ac-pin Specification And May Cause Unexpected In-band Resets
BA108.	X	No Fix	System Quiesce Events Initiated While Power Events are In Progress May Cause System Hangs
BA109.	X	No Fix	Uncorrected Memory Error Detected by a Memory Patrol Scrub With SMI Generated by Other Memory Controllers May Cause MCE/SMI Race Condition
BA110.	X	No Fix	An Intel QPI Link Layer Retry Quickly Followed by an Intel QPI Physical Layer Reset May Cause an MCE
BA111.	X	No Fix	A Transient UECC Error on Memory Reads on Systems With Mirrored Memory May Assert MCE
BA112.	X	No Fix	PerfMon Overflow Status Can Not be Cleared After Certain Conditions Have Occurred
BA113.	X	No Fix	An Unexpected Page Fault or EPT Violation May Occur After Another Logical Processor Creates a Valid Translation for a Page
BA114.	X	No Fix	Intel® QPI and Intel® SMI Drift Buffer Alarms May be Observed on the Processor
BA115.	X	No Fix	L1 Data Cache Errors May be Logged With Level Set to 1 Instead of 0
BA116.	X	No Fix	Stack Pushes May Not Occur Properly for Events Delivered Immediately After VM Entry to 16-Bit Software
BA117.	X	No Fix	VM Entries That Return From SMM Using VMLAUNCH May Not Update The Launch State of the VMCS
BA118.	X	No Fix	VM Entry May Clear Bytes 81H-83H on Virtual-APIC Page When “Use TPR Shadow” Is 0
BA119.	X	No Fix	System Hangs Possible Due to ECC Correctable Errors with EPT and DCU 16KB Mode Enabled
BA120.	X	No Fix	Concurrent Updates to a Segment Descriptor May be Lost
BA121.	X	No Fix	INVLPG Following INVEPT or INVVPID May Fail to Flush All Translations for a Large Page
BA122.	X	No Fix	A 2 MB Page Split Lock Accesses Combined With Complex Internal Events May Cause Unpredictable System Behavior
BA123.	X	No Fix	Changes to Reserved Bits of Some Nonarchitectural MSRs May Cause Unpredictable System Behavior
BA124.	X	No Fix	IO_SMI Indication in SMRAM State Save Area May be Set Incorrectly
BA125.	X	No Fix	L1 Cache Uncorrected Errors May be Recorded as Correctable in 16K Mode
BA126.	X	No Fix	Writing an Illegal Vector to the IA32_X2APIC_SELF_IPI MSR Will Hang the Processor
BA127.	X	No Fix	Successive Fixed Counter Overflows May be Discarded
BA128.	X	No Fix	VM Exits Due to “NMI-Window Exiting” May Not Occur Following a VM Entry to the Shutdown State
BA129.	X	No Fix	Execution of INVVPID Outside 64-Bit Mode Cannot Invalidate Translations For 64-Bit Linear Addresses

Table 1. Errata Table (Sheet 5 of 5)

Number	Stepping	Status	Description
	D-0		
BA130.	X	No Fix	A Combination of Data Accesses That Are Split Across Cacheline Boundaries May Lead to a Processor Hang
BA131.	X	No Fix	A Load May Appear to be Ordered Before an Earlier Locked Instruction
BA132.	X	No Fix	A Machine Check Occurring During VM Entry May Cause Unpredictable Behavior
BA133.	X	No Fix	MCI_ADDR May be Incorrect For Cache Parity Errors
BA134.	X	No Fix	The Corrected Error Count Overflow Bit in IA32_ MC0_STATUS is Not Updated When The UC Bit is Set
BA135.	X	No Fix	The Upper 32 Bits of CR3 May be Incorrectly Used With 32-Bit Paging
BA136.	X	No Fix	EPT Violations May Report Bits 11:0 of Guest Linear Address Incorrectly
BA137.	X	No Fix	SMRAM State-Save Area Above the 4-GB Boundary May Cause Unpredictable System Behavior
BA138.	X	No Fix	Virtual-APIC Page Accesses With 32-Bit PAE Paging May Cause a System Crash
BA139.	X	No Fix	VM Exit May Set IA32_EFER.NXE When IA32_MISC_ENABLE Bit 34 is Set to 1
BA140.	X	No Fix	Performance Monitor Counter MEM_INST_RETIRED.STORES May Count Higher than Expected
BA141.	X	No Fix	Pending x87 FPU Exceptions (#MF) May be Signaled Earlier Than Expected

Specification Changes

Number	Specification Changes
SCh1.	Removed
SCh2.	CCorrection to Datasheet Introduction Cache Table Entry

Specification Clarifications

Number	Specification Clarifications
	None for this revision of this specification update.

Document Changes

Number	Document Changes
DC1.	On-Demand Clock Modulation Feature Clarification

Identification Information

Component Identification via Programming Interface

The Intel® Xeon Processor 7500 Series stepping can be identified by the following register contents:

Reserved	Extended Family[1]	Extended Model[2]	Reserved	Processor Type[3]	Family Code[4]	Model Number[5]	Stepping ID[6]
31:28	27:20	19:16	15:14	13:12	11:8	7:4	3:0
	00000000b	0010b		00b	0110	1111b	0000b

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® Pro, Pentium® 4, Intel® Core™ processor family or Intel® Core™ i7 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 Processor Type, specified in bits [13:12] indicates whether the processor is an original OEM processor, an OverDrive processor, or a dual processor (capable of being used in a dual processor system).
4. 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.
5. 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.
6. The Stepping ID in bits [3:0] indicates the revision number of that model. See Table 2 for the processor stepping ID number in the CPUID information.

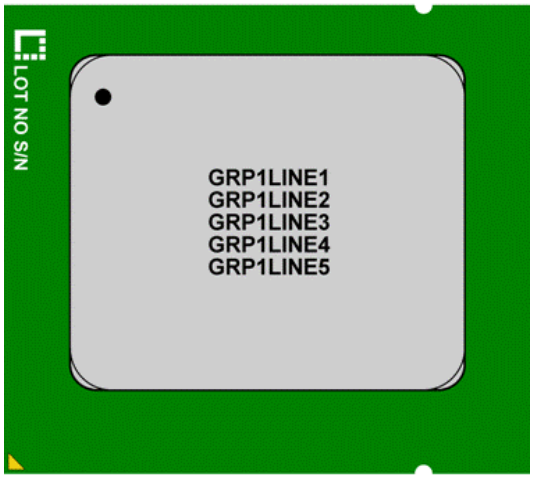
When EAX is initialized to a value of '1', the CPUID instruction returns the *Extended Family, Extended Model, Processor Type, Family Code, Model Number and Stepping ID* value in the EAX register. Note that the EDX processor signature value after reset is equivalent to the processor signature output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

Intel® Xeon® Processor 7500 Series stepping can be identified by the following component markings:

Figure 1. Processor Top-Side Marking (Example)



Legend: Mark Text (Production Mark):
GRP1LINE1:INTEL{M}{C}'YY PROC#
GRP1LINE2:Intel® Xeon
GRP1LINE3:SSPEC XXXXX
GRP1LINE4:SPEED/CACHE/INTC
GRP1LINE5:{FPO} {e4}

YY = Year
PROC# = Processor Number
xxxxx = Country of Origin
INTC = Interconnect Speed (Intel® QPI)
Factory Information

Table 2. Intel® Xeon® Processor 7500 Series Identification

S-Spec Number	Step	CPUID	Core Frequency (GHz) / Intel® QuickPath Interconnect (GT/s) / DDR3 (MHz)	Number of Cores	Cache Size (MB)	Series
SLBRD	D-0	000206E6h	2.266 Ghz/6.40 GT/s/6.40 GT/s	8	24 MB	X7560
SLBRB	D-0	000206E6h	2.000 Ghz/6.40 GT/s/6.40 GT/s	8	18 MB	X6550
SLBRH	D-0	000206E6h	1.866 Ghz/ 5.86 GT/s/5.86 GT/s	6	18 MB	L7545
SLBRJ	D-0	000206E6h	1.866 Ghz/ 5.86 GT/s/5.86 GT/s	6	12 MB	E7530
SLBRK	D-0	000206E6h	1.866 Ghz/ 4.80 GT/s/4.80 GT/s	4	18 MB	E7520
SLBRE	D-0	000206E6h	2.000 Ghz/6.40 GT/s/6.40 GT/s	8	18 MB	X7550
SLBRC	D-0	000206E6h	2.000 Ghz/6.40 GT/s/6.40 GT/s	6	18 MB	E6540
SLBRL	D-0	000206E6h	1.733 Ghz/4.80 GT/s/4.80 GT/s	4	12 MB	E6510
SLBRM	D-0	000206E6h	2.666 Ghz/ 5.86 GT/s/5.86 GT/s	6	18 MB	X7542
SLBRG	D-0	000206E6h	2.000 Ghz/6.40 GT/s/6.40 GT/s	6	18 MB	E7540
SLBRF	D-0	000206E6h	1.866 Ghz/ 5.86 GT/s/5.86 GT/s	8	24 MB	L7555

Mixing Processor Within MP Platforms

Intel supports multiprocessor (MP) configurations consisting of processors:

1. From the same power optimization segment.
2. That support the same maximum Intel® QuickPath Interconnect (Intel® QPI) and DDR3 memory speeds.
3. That share symmetry across physical packages with respect to the number of logical processors per package, number of cores per package, number of Intel® QPI interfaces, and cache topology.

4. That have identical Extended Family, Extended Model, Processor Type, Family Code, and Model Number as indicated by the function 1 of the CPUID instruction.

Note: Connected processors must operate with the same Intel® QPI and core frequency.

While Intel does nothing to prevent processors from operating together, some combinations may not be supported due to limited validation, which may result in uncharacterized errata. Coupling this fact with the large number of Intel® Xeon® Processor 7500 Series attributes, the following population rules and stepping matrix have been developed to define supported configurations.

1. Processors must be of the same power-optimization segment. This ensures processors include the same maximum Intel® QPI and cache sizes.
2. Processors must operate at the same core frequency. Note: Processors within the same power-optimization segment supporting different maximum core frequencies (for example, a 2.26 GHz / 130 W and 2.00 GHz / 130 W) can be operated within a system. However, both must operated at the highest frequency rating commonly supported. Mixing components operating at different internal clock frequencies is not supported and will not be validated by Intel.
3. Processors must share symmetry across physical packages with respect to the number of logical processors per package, number of Intel® QPI interfaces, and cache topology.
4. Mixing dissimilar steppings is only supported with processors that have identical Extended Family, Extended Model, Processor type, Family Code, and Model Number as indicated by the function 1 of the CPUID instruction. Mixing processors of different steppings but the same model (as per CPUID instruction) is supported. Details regarding the CPUID instruction are provided in the *AP-487,* Intel® Processor Identification and the CPUID Instruction application note and *Intel® 64 and IA-32 Architectures Software Developer's Manual, Volume 2A.*
5. After ANDing the feature flag and extended feature flag from the installed processors, any processor whose set of feature flags exactly matches the ANDed feature flags can be selected by the BIOS as the BSP. If no processor exactly matches the ANDed feature flag values, then the processors with the numerically lower CPUID should be selected as the BSP.
6. Intel requires that the processor microcode update be loaded on each processor operating within the system. Any processor that does not have the proper microcode update loaded is considered by Intel to be operating out of specification.
7. The workarounds identified in this, and subsequent specification updates, must properly applied to each processor in the system. Certain errata are specific to the multiprocessor environment. Errata for all processor steppings will affect system performance if not properly worked around.
8. Customers are fully responsible for the validation of their system configurations.

Errata

BA1. Intel® Interconnect BIST (Intel® IBIST) Does Not Work in Intel® QPI in Slow Mode

Problem: The Intel® IBIST (Intel® Interconnect Built-in Self Test) does not work in the Intel® QPI (Intel® QuickPath Interconnect) slow mode and only works at operational speed.

Implication: The Intel IBIST does not work in Intel® QPI slow mode.

Workaround: Do not run the Intel IBIST in slow mode.

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

BA2. Retraining Parameter Negotiation is Not Implemented for Intel® QPI

Problem: The Intel® QPI specification states that the physical layer initialization process needs to negotiate retraining parameters with a remote agent. The protocol is that agents should first exchange their respective retraining interval and duration as part of the link initialization flow. Then, each agent should compare the local and remote values and choose common values by selecting the shortest interval and the longest duration. This erratum is conveying that the described negotiation feature is not implemented in the processor.

Implication: The processor does not perform the hardware-based retraining parameter negotiation.

Workaround: BIOS will need to perform the necessary computations to determine the proper parameters and program them into the processor.

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

BA3. Intel® IBIST Slave Ignores Loop Count Values Sent by Master on Intel® QPI

Problem: During Intel IBIST (Interconnect Built-in Self Test) loopback, one agent is the master agent while the other is the slave agent on Intel® QPI. The slave should extract an Intel IBIST loop count from the training sequence sent by the master, and use this count to time its stay in the Loopback.Pattern state before returning to Loopback.Marker state. While the processor is operating as a slave, it does not extract this loop count and times its stay in the Loopback.Pattern state based on its locally programmed loop count.

Implication: When operating as an Intel IBIST slave, the processor ignores the loop count values sent by the master.

Workaround: Software needs to program the same loop count into the master and the slave.

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

BA4. System Hangs when Skipping Stop Req2 and Start Req1 Messages in Quiesce/Lock Sequence

Problem: Quiesce master skipping the StopReq2 and the StartReq1 Intel® QPI messages in the lock sequence will result in a system hang.

Implication: Due to this erratum, quiesce master lock flows with no StopReq2 and StartReq1 messages will cause a system hang.

Workaround: StopReq2 and StartReq1 messages should not be considered optional by quiesce master and must be sent to the processor as part of any lock flow.

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

BA5. Integrated Memory Controller Signals Spurious CMCI when Home Agent Failover Count Saturation Occurs

Problem: When home agent failover count saturation occurs, the memory controller signals a spurious CMCI (Corrected Machine Check Interrupt) without logging an error. Failover count saturation is not an error and a CMCI should not be issued.

Implication: Due to this erratum, software receives a CMCI with no error logged.

Workaround: None identified.

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

BA6. Memory Controller Does Not Set S Bit for Uncorrectable Error Followed by Software Recoverable Error

Problem: If an uncorrectable memory controller error is followed by a software recoverable error, the memory controller will not set the S (Signaling flag) bit of the MCi_STATUS to indicate that a software recoverable error occurred.

Implication: Due to this erratum, the MCi_STATUS of the memory controller will have the fields Valid=1, UC=1, PCC=0, OVER=1 and S=0 xlogged. When the MCA handler comes in, it ignores the MCi_STATUS since S=0; and the MCA is treated as a spurious MCA.

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*.

BA7. Running SDDC+1b with NB CRC Error May Result in Incorrect Data

Problem: When a read with a single bit error is made to a rank with SDDC+1b (Single Device Data Correction plus an additional single bit ECC error), along with the NB (North Bound) CRC error, it could result in incorrect data.

Implication: Due to this erratum, incorrect data may be returned when SDDC+1b is used with NB CRC error. This only has been observed in a synthetic validation environment.

Workaround: Disable SDDC+1b ECC correction by setting the register field M_PCSR_ECC_CORR_CTL.en_sgl_on_kill = 0 (Device:0x05; Function:0x4; Offset:0xB8; bit 6), so that every single-bit error to a rank with SDDC will become an uncorrectable error.

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

BA8. MCi_STATUS S Bit Not Set for LLC Software Recoverable Errors

Problem: When an explicit LLC (Last Level Cache) write-back software recoverable error is detected while there is already a poison error in the MCi_STATUS register, a machine check is signaled but the MCi_STATUS.S (Signaling Flag) bit is not set. In this case the MCi_STATUS.PCC (Processor Context Corrupt) bit and the S bit are both 0. As a result, the machine check handler assumes this to be a spurious error.

Implication: If there is already a poison error in the MCi_STATUS register and an LLC recoverable error is then logged the MCA handler may assume this to be a spurious error.

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*.

BA9. Correctable SB CRC Error May be Propagated to an Uncorrected ECC Error

Problem: Due to the processor not having a mechanism to detect incorrect alert frames, correctable SB (South Bound) CRC Error may be propagated to an uncorrected ECC error.

Implication: An incorrect alert frame will not be detected by the processor. In most cases there is no issue, due to the memory buffer issuing a series of alert frames. In a specific case

where a SB Intel® Scalable Memory Interconnect (Intel® SMI) CRC error (transient or persistent) is detected and the NB (North Bound) Alert frame responding to this error is also corrupted by an error, the original packet may not be reissued. However, since the memory controller uses two Intel SMI channels in lockstep for each cache line access, on a future read if one channel was affected by this issue the other would return valid data. Due to this erratum, the correctable SB CRC error may get propagated to be a detected but uncorrected ECC error. Intel has not observed this erratum on any commercially available system.

Workaround: None identified.

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

BA10. Memory Controller Patrol Scrub Ceases to Function with CRC Errors and the IMT31 Reclaim Feature Enabled

Problem: The processor does not fully implement the protocol in the Memory Controller-Home Agent for sharing the IMT31 (In-flight Memory Table) entry resulting in a patrol scrub deadlock. This issue can occur whenever the Error Flow State is invoked in response to CRC errors or hardware injected periodic ZQCAL (ZQ Calibration).

Implication: Patrol scrub may not function with CRC errors and the IMT31 reclaim feature enabled.

Workaround: A BIOS workaround has been identified. Please refer to the latest version of the memory reference code.

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

BA11. False Patrol Read Error Logging May Occur with Patrol Scrub Enabled and with SB CRC Errors

Problem: When there is a link level CRC on the SB (South Bound) Intel SMI lanes and patrol scrubbing is enabled, patrol scrubs may detect false uncorrectable ECC errors (logged as patrol read errors). The data at this location is not corrupted and poison bit is not set. The probability of this happening increases with a higher patrol rate or a higher SB CRC error rate.

Implication: Under poison mode, a patrol read error is a software recoverable error. If enabled system software should perform its programmed recovery action. This erratum has only been observed in a synthetic testing environment. Intel has not observed this erratum with any commercially available system.

Workaround: None identified. Performing patrol scrub through memory no more than once in 12 hours can minimize the occurrence of this erratum.

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

BA12. Electrically Idle Intel SMI and Intel® QPI Lanes May Deliver Data that May Look Like Deskew Headers

Problem: Intel SMI or Intel® QPI lanes that are not physically connected on the board, or have become unconnected, may result in a deskew failure.

Implication: Improper deskew headers may be observed if the Intel SMI or Intel® QPI lane of a port is not physically connected.

Workaround: None identified.

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

BA13. A Sequence of Instruction Fetches and Snoops to Locked Cache Lines May Cause Processor to Hang

Problem: During a sequence of instruction fetches with specific address relationships to other system traffic a snoop beat pattern that includes snoops to locked cache lines may become established which could cause the processor to hang.

Implication: The processor may hang under a set of conditions involving instruction fetches, and snoops to locked cache lines.

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*.

BA14. Writing to Unimplemented Bits of UU_CR_U_MSR_PMON_EVNT_SEL MSR does Not Result in #GP Fault

Problem: The bits [31:23,17,15:8] in UU_CR_U_MSR_PMON_EVNT_SEL MSR (C10H) are not implemented on the processor and are marked as reserved. Due to this erratum writing 1's to these bits does not generate a #GP (General Protection Fault) as expected.

Implication: Writing 1's to the unimplemented bits in UU_CR_U_MSR_PMON_EVNT_SEL MSR does not result in a #GP fault.

Workaround: None identified.

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

BA15. Mixed Rank Size Memory Configurations May Cause a Missing Refresh Event

Problem: When using DIMMs of different rank sizes on the same memory channel, a refresh may be missed when a write command to a memory rank is blocked by sustained reads to another memory rank. This erratum has been seen only in a synthetic testing environment. Intel has not observed this erratum with any commercially available software.

Implication: A missing refresh may cause the refresh rate to be lower than the programmed value.

Workaround: A BIOS workaround has been identified. Please refer to the latest version of the memory reference code.

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

BA16. Mirror Slave May Deliver Incorrect Data when a Read to the Mirror Master Completes Before the Write-back from the IOH

Problem: A read from the mirror master may complete before the write-back from the IOH completes. This will result in the IOH write-data not being immediately visible and can lead to the IOH write-data never becoming visible. In the case of a RdInvOwn transaction, the reading caching agent will take ownership which can then overwrite the IOH data. This is especially visible in false-sharing of cache lines which involve the IOH.

Implication: Due to this erratum, correct data is not delivered by the mirror slave. This erratum only occurs during mirror failover.

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*.

BA17. UU_CR_U_MSR_PMON_GLOBAL_OVF_CTL MSR Does Not Follow RW1C Access Method

Problem: The UU_CR_U_MSR_PMON_GLOBAL_OVF_CTL MSR (C02H) is access type RW1C (Read Write 1 Clear) and when written with 1's should clear the corresponding bit in the UU_CR_U_MSR_PMON_ GLOBAL_STATUS MSR (C01H). Due to this erratum, a read of the UU_CR_U_MSR_PMON_GLOBAL_OVF_CTL MSR does not return zeros however a read of the UU_CR_U_MSR_PMON_ GLOBAL_STATUS MSR will show appropriate clearing.

Implication: The UU_CR_U_MSR_PMON_GLOBAL_OVF_CTL MSR does not return zeros on a read.

Workaround: None identified.

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

BA18. HNID Field is Incorrect for CMP Messages From PrefetchHint

Problem: The HNID (Home Node ID) field used in the PMON (Performance Monitoring) match/mask is incorrect for CMP (complete) messages from PrefetchHint. The same incorrect HNID is logged in the event of an error condition on a CMP for a PrefetchHint in the caching agent. The logging of the RNID (Requester Node ID) in the error logs for NDR (Non Data Response) messages is incorrect and impacts the caching agent system bound errors.

Implication: There are two implications:

1. Incorrect HNID is filtered or matched for CMP's using the caching agent PMON match/mask.
2. The incorrect RNID will be logged only for errors on NDR messages.

Workaround: There are two potential workarounds:

1. The Intel® QPI performance monitor match/mask can be used to count different types of CMP messages from the caching agent.
2. Ignore the RNID field for NDR system bound messages.

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

BA19. Processor Internal Initialization Code May become Damaged Under Certain Conditions

Problem: Under certain platform conditions, it is possible for the processor's internal initialization code to become damaged or erased over the course of power cycling VIO and VCACHE multiple times. At least four such power cycles are required to encounter this erratum.

Implication: If this erratum occurs, the processor internal initialization code will become damaged or erased and the processor will no longer be functional.

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*.

BA20. 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*.

BA21. 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*.

BA22. 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*.

BA23. 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*.

BA24. 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*.

BA25. REP MOVS/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

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 MOVS or REP STOS as fast strings. Due to this erratum fast string REP MOVS/REP STOS 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 MOVS or REP STOS instruction that will execute with fast strings enabled.

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

BA26. 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 (for example, NMI (Non-Maskable Interrupt), Debug break(#DB), Machine Check (#MC), and so forth). If the RSM attempts to return to a noncanonical address, the address pushed onto the stack for this #GP fault may not match the noncanonical 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*.

BA27. Performance Monitor SSE Retired Instructions May Return Incorrect Values

Problem: Performance Monitoring counter SIMD_INST_RETIRED (Event: C7H) is used to track retired SSE instructions. Due to this erratum, the processor may also count other types of instructions resulting in higher than expected values.

Implication: Performance Monitoring counter SIMD_INST_RETIRED may report count higher than expected.

Workaround: None identified.

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

BA28. Premature Execution of a Load Operation Prior to Exception Handler Invocation

Problem: If any of the below circumstances occur, it is possible that the load portion of the instruction will have executed before the exception handler is entered.

- If an instruction that performs a memory load causes a code segment limit violation.
- If a waiting X87 floating-point (FP) instruction or MMX™ technology (MMX) instruction that performs a memory load has a floating-point exception pending.
- If an MMX or SSE/SSE2/SSE3/SSSE3 extensions (SSE) instruction that performs a memory load and has either CR0.EM=1 (Emulation bit set), or a floating-point Top-of-Stack (FP TOS) not equal to 0, or a DNA exception pending.

Implication: In normal code execution where the target of the load operation is to write back memory there is no impact from the load being prematurely executed, or from the restart and subsequent re-execution of that instruction by the exception handler. If the

target of the load is to uncached memory that has a system side-effect, restarting the instruction may cause unexpected system behavior due to the repetition of the side-effect. Particularly, while CR0.TS [bit 3] is set, a MOVD/MOVQ with MMX/XMM register operands may issue a memory load before getting the DNA exception.

Workaround: Code which performs loads from memory that has side-effects can effectively workaround this behavior by using simple integer-based load instructions when accessing side-effect memory and by ensuring that all code is written such that a code segment limit violation cannot occur as a part of reading from side-effect memory.

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

BA29. 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 (that is, 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*.

BA30. Incorrect Address Computed For Last Byte of FXSAVE/FXRSTOR Image Leads to Partial Memory Update

Problem: A partial memory state save of the 512-byte FXSAVE image or a partial memory state restore of the FXRSTOR image may occur if a memory address exceeds the 64KB limit while the processor is operating in 16-bit mode or if a memory address exceeds the 4GB limit while the processor is operating in 32-bit mode.

Implication: FXSAVE/FXRSTOR will incur a #GP fault due to the memory limit violation as expected but the memory state may be only partially saved or restored.

Workaround: Software should avoid memory accesses that wrap around the respective 16-bit and 32-bit mode memory limits.

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

BA31. 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. This issue would only occur when one of the 2 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*.

BA32. 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*.

BA33. 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 *Intel® 64 and IA-32 Architectures Software Developer's Manual, Volume 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*.

BA34. 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*.

BA35. 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 (for example, 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*.

BA36. General Protection (#GP) Fault May Not Be Signaled on Data Segment Limit Violation above 4-G Limit

Problem: In 32-bit mode, memory accesses to flat data segments (base = 00000000h) that occur above the 4G limit (0ffffffffh) may not signal a #GP fault.

Implication: When such memory accesses occur in 32-bit mode, the system may not issue a #GP fault.

Workaround: Software should ensure that memory accesses in 32-bit mode do not occur above the 4G limit (0ffffffffh).

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

BA37. 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 (0x00007FFFFFFFFFFF) 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*.

BA38. A VM Exit on MWAIT May Incorrectly Report the Monitoring Hardware as Armed

Problem: A processor write to the address range armed by the MONITOR instruction may not immediately trigger the monitoring hardware. Consequently, a VM exit on a later MWAIT may incorrectly report the monitoring hardware as armed, when it should be reported as unarmed due to the write occurring prior to the MWAIT.

Implication: If a write to the range armed by the MONITOR instruction occurs between the MONITOR and the MWAIT, the MWAIT instruction may start executing before the monitoring hardware is triggered. If the MWAIT instruction causes a VM exit, this could cause its exit qualification to incorrectly report 0x1. In the recommended usage model for MONITOR/MWAIT, there is no write to the range armed by the MONITOR instruction between the MONITOR and the MWAIT.

Workaround: Software should never write to the address range armed by the MONITOR instruction between the MONITOR and the subsequent MWAIT.

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

BA39. Performance Monitor Event SEGMENT_REG_LOADS Counts Inaccurately

Problem: The performance monitor event SEGMENT_REG_LOADS (Event 06H) counts instructions that load new values into segment registers. The value of the count may be inaccurate.

Implication: The performance monitor event SEGMENT_REG_LOADS may reflect a count higher or lower than the actual number of events.

Workaround: None identified.

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

BA40. #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*.

BA41. Improper Parity Error Signaled in the IQ Following Reset When a Code Breakpoint is Set on a #GP Instruction

Problem: While coming out of cold reset or exiting from C6, if the processor encounters an instruction longer than 15 bytes (which causes a #GP) and a code breakpoint is enabled on that instruction, an IQ (Instruction Queue) parity error may be incorrectly logged resulting in an MCE (Machine Check Exception).

Implication: When this erratum occurs, an MCE may be incorrectly signaled.

Workaround: None identified.

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

BA42. 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 *Intel® 64 and IA-32 Intel® Architectures 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*.

BA43. IA32_MPERF Counter Stops Counting During On-Demand TM1

Problem: According to the *Intel® 64 and IA-32 Architectures Software Developer's Manual* Volume 3A: System Programming Guide, the ratio of IA32_MPERF (MSR E7H) to IA32_APERF (MSR E8H) should reflect actual performance while TM1 or on-demand throttling is activated. Due to this erratum, IA32_MPERF MSR stops counting while TM1 or on-demand throttling is activated, and the ratio of the two will indicate higher processor performance than actual.

Implication: The incorrect ratio of IA32_APERF/IA32_MPERF can mislead software P-state (performance state) management algorithms under the conditions described above. It is possible for the Operating System to observe higher processor utilization than actual, which could lead the OS into raising the P-state. During TM1 activation, the OS P-state request is irrelevant and while on-demand throttling is enabled, it is expected that the OS will not be changing the P-state. This erratum should result in no practical implication to software.

Workaround: None identified.

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

BA44. Synchronous Reset of IA32_APERF/IA32_MPERF Counters on Overflow Does Not Work

Problem: When either the IA32_MPERF or IA32_APERF MSR (E7H, E8H) increments to its maximum value of 0xFFFF_FFFF_FFFF_FFFF, both MSRs are supposed to synchronously reset to 0x0 on the next clock. This synchronous reset does not work. Instead, both MSRs increment and overflow independently.

Implication: Software can not rely on synchronous reset of the IA32_APERF/IA32_MPERF registers.

Workaround: None identified.

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

BA45. Disabling Thermal Monitor While Processor is Hot, Then Re-enabling, May Result in Stuck Core Operating Ratio

Problem: If a processor is at its TCC (Thermal Control Circuit) activation temperature and then Thermal Monitor is disabled by a write to IA32_MISC_ENABLES MSR (1A0H) bit [3], a subsequent re--enable of Thermal Monitor will result in an artificial ceiling on the maximum core P-state. The ceiling is based on the core frequency at the time of Thermal Monitor disable. This condition will only correct itself once the processor reaches its TCC activation temperature again.

Implication: Since Intel requires that Thermal Monitor be enabled in order to be operating within specification, this erratum should never be seen during normal operation.

Workaround: Software should not disable Thermal Monitor during processor operation.

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

BA46. OVER Bit for IA32_MCi_STATUS Register May Get Set on Specific Internal Error

Problem: If a specific type of internal unclassified error is detected, as identified by IA32_MCi_STATUS.MCACOD=0x0405, the IA32_MCi_ STATUS.OVER (overflow) bit [62] may be erroneously set.

Implication: The OVER bit of the MCi_STATUS register may be incorrectly set for a specific internal unclassified error.

Workaround: None identified.

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

BA47. Writing the Local Vector Table (LVT) when an Interrupt is Pending May Cause an Unexpected Interrupt

Problem: If a local interrupt is pending when the LVT entry is written, an interrupt may be taken on the new interrupt vector even if the mask bit is set.

Implication: An interrupt may immediately be generated with the new vector when a LVT entry is written, even if the new LVT entry has the mask bit set. If there is no Interrupt Service Routine (ISR) set up for that vector the system will GP fault. If the ISR does not do an

End of Interrupt (EOI) the bit for the vector will be left set in the in-service register and mask all interrupts at the same or lower priority.

Workaround: Any vector programmed into an LVT entry must have an ISR associated with it, even if that vector was programmed as masked. This ISR routine must do an EOI to clear any unexpected interrupts that may occur. The ISR associated with the spurious vector does not generate an EOI, therefore the spurious vector should not be used when writing the LVT.

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

BA48. Reading Reserved APIC Registers May Not Signal an APIC Error

Problem: Reads of reserved APIC registers in xAPIC compatibility mode should signal an APIC error with the Illegal Register Address bit [11] set in the Error Status Register (offset 0x280). Due to the erratum, the error is neither logged nor signaled.

Implication: A reserved APIC register access error interrupt may not be logged or signaled, even though the APIC error interrupt is enabled, on a read of a reserved APIC register.

Workaround: None identified.

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

BA49. 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 (that is, 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*.

BA50. Certain Store Parity Errors May Not Log Correct Address in IA32_MCi_ADDR

Problem: When store parity errors in the Level 0 hierarchy (as defined in the LL subfield of the IA32_MCi_STATUS MSR) occur, it is possible that the address of the error will not be logged in IA32_MCi_ADDR MSR. The error itself will be logged properly.

Implication: The address in IA32_MCi_ADDR may be incorrect after certain store parity errors occur.

Workaround: None identified.

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

BA51. xAPIC Timer May Decrement Too Quickly Following an Automatic Reload While in Periodic Mode

Problem: When the xAPIC Timer is automatically reloaded by counting down to zero in periodic mode, the xAPIC Timer may slip in its synchronization with the external clock. The xAPIC timer may be shortened by up to one xAPIC timer tick.

Implication: When the xAPIC Timer is automatically reloaded by counting down to zero in periodic mode, the xAPIC Timer may slip in its synchronization with the external clock. The xAPIC timer may be shortened by up to one xAPIC timer tick.

Workaround: None identified.

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

BA52. Certain Undefined Opcodes Crossing a Segment Limit May Result in #UD Instead of #GP Exception

Problem: Processor may take a #UD (Invalid Opcode) exception instead of a #GP (General Protection) exception when certain undefined opcodes (opcodes 0F 01 D0 - 0F 01 D5) extend beyond the segment limit.

Implication: Due to this erratum, processor may not take a #GP exception in this situation.

Workaround: None identified.

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

BA53. 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*.

BA54. MOVNTDQA From WC Memory May Pass Earlier Locked Instructions

Problem: An execution of MOVNTDQA that loads from WC (write combining) memory may appear to pass an earlier locked instruction to a different cache line.

Implication: Software that expects a lock to fence subsequent MOVNTDQA instructions may not operate properly. If the software does not rely on locked instructions to fence the subsequent execution of MOVNTDQA then this erratum does not apply.

Workaround: Software that requires a locked instruction to fence subsequent executions of MOVNTDQA should insert an LFENCE instruction before the first execution of MOVNTDQA following the locked instruction. If there is already a fencing or serializing instruction between the locked instruction and the MOVNTDQA, then an additional LFENCE is not necessary.

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

BA55. 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*.

BA56. Delivery of Certain Events Immediately Following a VM Exit May Push a Corrupted RIP Onto The Stack

Problem: If any of the following events is delivered immediately following a VM exit to 64-bit mode from outside 64-bit mode, bits 63:32 of the RIP value pushed on the stack may be cleared to 0:

1. A nonmaskable interrupt (NMI);
2. A machine-check exception (#MC);
3. A page fault (#PF) during instruction fetch; or
4. A general-protection exception (#GP) due to an attempt to decode an instruction whose length is greater than 15 bytes.

Implication: Unexpected behavior may occur due to the incorrect value of the RIP on the stack. Specifically, return from the event handler via IRET may encounter an unexpected page fault or may begin fetching from an unexpected code address.

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*.

BA57. The Combination of a Bus Lock and a Data Access That is Split Across Page Boundaries May Lead to Processor Livelock

Problem: Under certain complex microarchitectural conditions, the coincidence of a bus lock initiated by one logical processor of a Hyper-threading enabled processor core and data accesses that are split across page boundaries, initiated on the other logical processor on the same core, may lead to processor livelock.

Implication: Due to this erratum, a livelock may occur that can only be terminated by a processor reset. 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*.

BA58. An Unexpected Page Fault May Occur Following the Unmapping and Remapping of a Page

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

- The paging structures initially specify a valid translation for the page.
- Software modifies the paging structures so that there is no valid translation for the page (for example, by clearing to 0 the present bit in one of the paging-structure entries used to translate the page).
- Software later modifies the paging structures so that the translation is again a valid translation for the page (for example, by setting to 1 the bit that was cleared earlier).
- A subsequent instruction loads from a linear address on the page.
- Software did not invalidate TLB entries for the page between the first modification of the paging structures and the load from the linear address.

In this case, the load bye the later instruction may cause a page fault that indicates that there is no translation for the page.

Implication: Software may see an unexpected page fault that indicates that there is no translation for the page.

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*.

BA59. Two xAPIC Timer Event Interrupts May Unexpectedly Occur

Problem: If an xAPIC timer event is enabled and while counting down the current count reaches 1 at the same time that the processor thread begins a transition to a low power C-state, the xAPIC may generate two interrupts instead of the expected one when the processor returns to C0.

Implication: Due to this erratum, two interrupts may unexpectedly be generated by an xAPIC timer event.

Workaround: None identified.

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

BA60. 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*.

BA61. PEBS Records For Load Latency Monitoring May Contain an Incorrect Linear Address

Problem: The load latency performance monitoring feature stores information about a load into a record in the PEBS (Precise event-based sampling) buffer in the DS save area. This information includes the Data Source Encoding, Latency Value, and Data Linear Address of the load causing the performance counter to overflow. Under certain conditions it is possible for the linear address to be incorrect.

Implication: The linear address reported by the load latency performance monitoring feature for PEBS may be incorrect.

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*.

BA62. PEBS Field “Data Linear Address” is Not Sign Extended to 64 Bits

Problem: The Data Linear Address field of the PEBS (Precise Event-Based Sampling) record is not correctly sign extended to 64 bits and may appear as a noncanonical address when observed in the PEBS record.

Implication: The PEBS Data Linear Address field may not have the sign bit correctly extended to bits [63:48].

Workaround: None identified.

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*.

BA63. 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*.

BA64. CPUID Incorrectly Indicates the UnHalted Reference Cycle Architectural Event is Supported

Problem: The architectural performance monitoring event for UnHalted Reference Cycles (3CH, Umask 01H) is not supported on the processor. The CPUID instruction, when executed with EAX = 0AH, should return bit 2 of EBX as 1 to indicate that this event is not supported. Due to this erratum, CPUID will improperly return bit 2 as 0.

Implication: Software relying on the CPUID instruction to determine support of the UnHalted Reference Cycles event will incorrectly assume the event is available.

Workaround: None identified.

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

BA65. DR6.B0-B3 May Not Report All Breakpoints Matched When a MOV/POP SS is Followed by a Store Instruction

Problem: Normally, data breakpoints matches that occur on a MOV SS, r/m or POP SS will not cause a debug exception immediately after MOV/POP SS but will be delayed until the instruction boundary following the next instruction is reached. After the debug exception occurs, DR6.B0-B3 bits will contain information about data breakpoints matched during the MOV/POP SS as well as breakpoints detected by the following instruction. Due to this erratum, DR6.B0-B3 bits may not contain information about data breakpoints matched during the MOV/POP SS when the following instruction is a store instruction.

Implication: When this erratum occurs, DR6 may not contain information about all breakpoints matched. This erratum will not be observed under the recommended usage of the MOV SS,r/m or POP SS instructions (that is, following them only with an instruction that writes (E/R)SP).

Workaround: None identified.

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

BA66. 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*.

BA67. IA32_PERF_GLOBAL_CTRL MSR May be Incorrectly Initialized

Problem: The IA32_PERF_GLOBAL_CTRL MSR (38FH) bits [34:32] may be incorrectly set to 7H after reset; the correct value should be 0H.

Implication: The IA32_PERF_GLOBAL_CTRL MSR bits [34:32] may be incorrect after reset (EN_FIXED_CTR{0, 1, 2} may be enabled).

Workaround: None identified.

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

BA68. Processors with SMT May Hang on P-State Transition or ACPI Clock Modulation Throttling

Problem: When SMT is enabled, it is possible that a P-state transition or ACPI clock modulation throttling may hang and log a machine check error with IA32_MCi_STATUS.MCACOD = 0x0150. This hang condition requires a specific sequence of instructions coincident with the P-state or ACPI event.

Implication: When this erratum occurs, the processor will unexpectedly hang. 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*.

BA69. Replaced with BA140

BA70. Sleeping Cores May Not be Woken Up on Logical Cluster Mode Broadcast IPI Using Destination Field Instead of Shorthand

Problem: If software sends a logical cluster broadcast IPI using a destination shorthand of 00B (No Shorthand) and writes the cluster portion of the Destination Field of the Interrupt Command Register to all ones while not using all 1s in the mask portion of the Destination Field, target cores in a sleep state that are identified by the mask portion of the Destination Field may not be woken up. This erratum does not occur if the destination shorthand is set to 10B (All Including Self) or 11B (All Excluding Self).

Implication: When this erratum occurs, cores which are in a sleep state may not wake up to handle the broadcast IPI. Intel has not observed this erratum with any commercially available software.

Workaround: Use destination shorthand of 10B or 11B to send broadcast IPIs.

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

BA71. Faulting Executions of FXRSTOR May Update State Inconsistently

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

Implication: Software that relies on x87 state or SSE state following a faulting execution of FXRSTOR may behave inconsistently.

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

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

BA72. Performance Monitor Counters May Count Incorrectly

Problem: Under certain circumstances, a general purpose performance counter, IA32_PMC0-4 (C1H – C4H), may count at core frequency or not count at all instead of counting the programmed event.

Implication: The Performance Monitor Counter IA32_PMCx may not properly count the programmed event. Due to the requirements of the workaround there may be an interruption in the counting of a previously programmed event during the programming of a new event.

Workaround: Before programming the performance event select registers, IA32_PERFEVTSELx MSR (186H – 189H), the internal monitoring hardware must be cleared. This is accomplished by first disabling, saving valid events and clearing from the select registers, then programming three event values 0x4300D2, 0x4300B1 and 0x4300B5 into the IA32_PERFEVTSELx MSRs, and finally continuing with new event programming and restoring previous programming if necessary. Each performance counter, IA32_PMCx, must have its corresponding IA32_PREFEVTSELx MSR programmed with at least one of the event values and must be enabled in IA32_PERF_GLOBAL_CTRL MSR (38FH) bits [3:0]. All three values must be written to either the same or different IA32_PERFEVTSELx MSRs before programming the performance counters. Note that the performance counter will not increment when its IA32_PERFEVTSELx MSR has a value of 0x4300D2, 0x4300B1 or 0x4300B5 because those values have a zero UMASK field (bits [15:8]).

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

BA73. Performance Monitor Event Offcore_response_0 (B7H) Does Not Count NT Stores to Local DRAM Correctly

Problem: When a IA32_PERFEVTSELx MSR is programmed to count the Offcore_response_0 event (Event:B7H), selections in the OFFCORE_RSP_0 MSR (1A6H) determine what is counted. The following two selections do not provide accurate counts when counting NT (Non-Temporal) Stores:

- OFFCORE_RSP_0 MSR bit [14] is set to 1 (LOCAL_DRAM) and bit [7] is set to 1 (OTHER): NT Stores to Local DRAM are not counted when they should have been.
- OFFCORE_RSP_0 MSR bit [9] is set to (OTHER_CORE_HIT_SNOOP) and bit [7] is set to 1 (OTHER): NT Stores to Local DRAM are counted when they should not have been.

Implication: The counter for the Offcore_response_0 event may be incorrect for NT stores.

Workaround: None identified.

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

BA74. 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 nonwritable 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*.

BA75. LER and LBR MSRs May Be Incorrectly Updated During a Task Switch

Problem: LER (Last Exception Record) and LBR (Last Brand Record) MSRs (MSR_LER_FROM_LIP (1DDH), MSR_LER_TO_LIP (1DEH) and MSR_LASTBRANCH{0:15}_FROM_IP (680H – 68FH)) may contain incorrect values after a fault or trap that does a task switch.

Implication: After a task switch the value of the LER and LBR MSRs may be updated to point to incorrect instructions.

Workaround: None identified.

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

BA76. Back to Back Uncorrected Machine Check Errors May Overwrite IA32_MC3_STATUS.MSCOD

Problem: When back-to-back uncorrected machine check errors occur that would both be logged in the IA32_MC3_STATUS MSR (40CH), the IA32_MC3_STATUS.MSCOD (bits [31:16]) field may reflect the status of the most recent error and not the first error. The rest of the IA32_MC3_STATUS MSR contains the information from the first error.

Implication: Software should not rely on the value of IA32_MC3_STATUS.MSCOD if IA32_MC3_STATUS.OVER (bit [62]) is set.

Workaround: None identified.

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

BA77. Corrected Errors With a Yellow Error Indication May be Overwritten by Other Corrected Errors

Problem: A corrected cache hierarchy data or tag error that is reported with IA32_MCi_STATUS.MCACOD (bits [15:0]) with value of 000x_0001_xxxx_xx01 (where x stands for zero or one) and a yellow threshold-based error status indication (bits [54:53] equal to 10B) may be overwritten by a corrected error with a no tracking indication (00B) or green indication (01B).

Implication: Corrected errors with a yellow threshold-based error status indication may be overwritten by a corrected error without a yellow indication.

Workaround: None identified.

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

BA78. A String Instruction that Remaps a Page May Encounter an Unexpected Page Fault

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

- The paging structures initially specify a valid translation for the page.
- Software modifies the paging structures so that there is no valid translation for the page (for example, by clearing to 0 the present bit in one of the paging-structure entries used to translate the page).
- An iteration of a string instruction modifies the paging structures so that the translation is again a valid translation for the page (for example, by setting to 1 the bit that was cleared earlier).
- A later iteration of the same string instruction loads from a linear address on the page.
- Software did not invalidate TLB entries for the page between the first modification of the paging structures and the string instruction. In this case, the load in the later iteration may cause a page fault that indicates that there is no translation for the page (for example, with bit 0 clear in the page-fault error code, indicating that the fault was caused by a not-present page).

Implication: Software may see an unexpected page fault that indicates that there is no translation for the page. Intel has not observed this erratum with any commercially available software or system.

Workaround: Software should not update the paging structures with a string instruction that accesses pages mapped the modified paging structures.

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

BA79. Performance Monitor Events DCACHE_CACHE_LD and DCACHE_CACHE_ST May Overcount

Problem: The performance monitor events DCACHE_CACHE_LD (Event 40H) and DCACHE_CACHE_ST (Event 41h) count cacheable loads and stores that hit the L1 cache. Due to this erratum, in addition to counting the completed loads and stores, the counter will incorrectly count speculative loads and stores that were aborted prior to completion.

Implication: The performance monitor events DCACHE_CACHE_LD and DCACHE_CACHE_ST may reflect a count higher than the actual number of events.

Workaround: None identified.

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

BA80. Rapid Core C3 Transition May Cause Unpredictable System Behavior

Problem: Under a complex set of internal conditions, cores rapidly performing C3 transitions in a system with Intel® Hyper-Threading Technology enabled may cause a machine check error (IA32_MCi_STATUS.MCACOD = 0x0106), system hang or unpredictable system behavior.

Implication: This erratum may cause a machine check error, system hang or unpredictable system behavior.

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*.

BA81. Performance Monitor Events INSTR_RETIRED and MEM_INST_RETIRED May Count Inaccurately

Problem: The performance monitor event INSTR_RETIRED (Event C0H) should count the number of instructions retired, and MEM_INST_ RETIRED (Event 0BH) should count the number of load or store instructions retired. However, due to this erratum, they may undercount.

Implication: The performance monitor event INSTR_RETIRED and MEM_INST_RETIRED may reflect a count lower than the actual number of events.

Workaround: None identified.

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

BA82. A Page Fault May Not be Generated When the PS bit is set to “1” in a PML4E or PDPTE

Problem: On processors supporting Intel® 64 architecture, the PS bit (Page Size, bit 7) is reserved in PML4Es and PDPTEs. If the translation of the linear address of a memory access encounters a PML4E or a PDPTE with PS set to 1, a page fault should occur. Due to this erratum, PS of such an entry is ignored and no page fault will occur due to its being set.

Implication: Software may not operate properly if it relies on the processor to deliver page faults when reserved bits are set in paging-structure entries.

Workaround: Software should not set bit 7 in any PML4E or PDPTE that has Present Bit (Bit 0) set to “1”.

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

BA83. Erratum removed

BA84. 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*.

BA85. Performance Monitoring Events STORE_BLOCKS.NOT_STA and STORE_BLOCKS.STA May Not Count Events Correctly

Problem: Performance Monitor Events STORE_BLOCKS.NOT_STA and STORE_BLOCKS.STA should only increment the count when a load is blocked by a store. Due to this erratum, the count will be incremented whenever a load hits a store, whether it is blocked or can forward. In addition this event does not count for specific threads correctly.

Implication: If Intel Hyper-Threading Technology is disabled, the Performance Monitor events STORE_BLOCKS.NOT_STA and STORE_BLOCKS.STA may indicate a higher occurrence of loads blocked by stores than have actually occurred. If Intel Hyper-Threading Technology is enabled, the counts of loads blocked by stores may be unpredictable and they could be higher or lower than the correct count.

Workaround: None identified.

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

BA86. VMX-Preemption Timer Does Not Count Down at the Rate Specified

Problem: The VMX-preemption timer should count down by 1 every time a specific bit in the TSC (Time Stamp Counter) changes. (This specific bit is indicated by IA32_VMX_MISC bits [4:0] (0x485h) and has a value of 5 on the affected processors.) Due to this erratum, the VMX-preemption timer may instead count down at a different rate and may do so only intermittently.

Implication: The VMX-preemption timer may cause VM exits at a rate different from that expected by software.

Workaround: None identified.

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

BA87. VM Exits Due to LIDT/LGDT/SIDT/SGDT Do Not Report Correct Operand Size

Problem: When a VM exit occurs due to a LIDT, LGDT, SIDT, or SGDT instruction with a 32-bit operand, bit 11 of the VM-Exit Instruction-Information Field should be set to 1. Due to this erratum, this bit is instead cleared to 0 (indicating a 16-bit operand).

Implication: Virtual Machine Monitors cannot rely on bit 11 of the VM-Exit Instruction-Information Field to determine the operand size of the instruction causing the VM exit.

Workaround: Virtual Machine Monitor software may decode the instruction to determine operand size.

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

BA88. Multiple Performance Monitor Interrupts are Possible on Overflow of IA32_FIXED_CTR2

Problem: When multiple performance counters are set to generate interrupts on an overflow and more than one counter overflows at the same time, only one interrupt should be generated. However, if one of the counters set to generate an interrupt on overflow is the IA32_FIXED_CTR2 (MSR 30BH) counter, multiple interrupts may be generated when the IA32_FIXED_CTR2 overflows at the same time as any of the other performance counters.

Implication: Multiple counter overflow interrupts may be unexpectedly generated.

Workaround: None Identified.

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

BA89. 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 nonroot 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*.

BA90. LBRs May Not be Initialized During Power-On Reset of the Processor

Problem: If a second reset is initiated during the power-on processor reset cycle, the LBRs (Last Branch Records) may not be properly initialized.

Implication: Due to this erratum, debug software may not be able to rely on the LBRs out of power-on reset.

Workaround: Ensure that the processor has completed its power-on reset cycle prior to initiating a second reset.

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

BA91. Single Bit Error Will Become an Uncorrectable Error if it Occurs on Specific Bits Under SDDC

Problem: After a DRAM device is mapped out, SDDC+1 (Single Device Data Correction +1 bit correction) detects but does not correct a single bit error found in bit 0 of the cache line or a single bit error found in bit 0 of the cache line parity information, when the Intel® SMI (Intel Scalable Memory Interconnect) link is running at 5.86 GT/s and both of the DDR buses behind the Intel® 7500 Scalable Memory Controller on which the error occurs are populated. Instead, these single bit errors are escalated as uncorrectable errors.

Implication: When a single-bit error occurs on one of these two bits, the data is flagged as poisoned and an uncorrectable response is sent to the home agent. This erratum results in approximately a 0.19% loss of single-bit error correction coverage for SDDC+1.

Workaround: None identified.

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

BA92. Power Controller Performance Monitor Counters May Not Operate Correctly

Problem: The programmable power controller performance monitor counters MSR_W_PMON_CTR{0,1,2,3} MSRs (C91H, C93H, C95H, C97H) do not operate correctly in some scenarios.

These programmable counters have two common usage models: Polled and PMI (Performance Monitor Interrupt).

Polled: The polled usage model starts the counters from zero, and reads the values after some period. As long as the accumulated event count is less than 2^24, the counter value is correct. If the counter value exceeds 2^24, then in cases when the counter value modulo 2^24 equals 0 or 1 the value of the counter will be too low by 2^24.

PMI: The interrupt usage model loads the counter with a value of -N and takes a PMI after N events (when the counter rolls over to zero). The most common usage model triggers a PMI on the first (or second) occurrence of an event; this corresponds to loading a value of -1 or -2 into the counter. These values work correctly. Values of N from 3 to 2^24 will generate a PMI after N+2 events (two events later than requested). Values of N greater than 2^24 will in almost all cases also generate a PMI after N+2 events; the exceptions occur when (N mod 2^24) equals 1 or 2, in which case the PMI will be generated after N-(2^24-2) events.

Implication: Power controller performance monitor counter read values may be incorrect and/or PMIs may occur at the wrong time.

Workaround: Workarounds have been identified for the polled and PMI usage models.

Polled: If the counter value is less than 2^24, the value N is correct. If the counter value N is greater than 2^24, examine N. If the low order 24 bits of N = 0x000000 or 0x000001 (N mod 2^24 = 0 or 1), then add 2^24 to the N to get the correct count; in all other cases, the value of N is correct.

PMI: If the number of events N to be counted before triggering the PMI is:

(a) N = 1 or 2, then load -N into the counter

(b) (N mod 2^24) = 1 or 2 (but N <> 1 or 2), then load -N-(2^24-2)

(c) All other cases, load -N + 2

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

BA93. Intel® QPI HNID Field is Incorrect for CMP Messages From PrefetchHint

Problem: The HNID (Home Node ID) field used in the PMON (Performance Monitoring) match/mask is incorrect for CMP (complete) messages from PrefetchHint. The same incorrect HNID is logged in the event of an error condition on a CMP for a PrefetchHint in the caching agent. The logging of the RNID (Requester Node ID) in the error logs for NDR (Non Data Response) messages is incorrect and impacts the caching agent system bound errors.

Implication: There are three implications:

1. Incorrect HNID is filtered or matched for CMPs using the caching agent PMON match/mask.

2. Caching agent MCA logs will have incorrect HNID in the event of an error on a CMP due to a PrefetchHint. The mask/match and error logs will be unable to correctly distinguish CMPs from a PrefetchHint versus an Interrupt.
3. The incorrect RNID will be logged for errors on NDR messages.

Workaround: There are two potential workarounds:

1. The Intel® QPI performance monitor match/mask can be used to count different types of CMP messages from the caching agent.
2. Ignore the RNID field for NDR system bound messages.

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

BA94. System Configuration Controller Misaligned Error May Result in a System Hang

Problem: Under certain conditions the system configuration controller may not correctly handle NcRd (Non-Coherent Read) packets which may result in a misaligned uncorrectable error, Machine Check Exception or system hang.

Implication: The system configuration controller incorrectly signals an uncorrectable error resulting in a system hang.

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*.

BA95. Recoverable Errors Signaled From Intel® QPI or Intel® SMI Port to the System Configuration Controller May Get Lost if the Ports are Disabled

Problem: Each Intel® Quick-Path Interconnect (Intel® QPI) and Intel® Scalable Memory Interconnect (Intel SMI) physical layer port may be configured through its PBOXERRMASK register (Device:0x14, Function:0x2, Offset:0x68) to generate RAS (Reliability Accessibility Serviceability) recoverable error signals in any of the four situations: initialization failure, width reduction (Intel® QPI) or lane failover (Intel SMI), drift buffer alarm, or latency buffer roll-over. When generated, the error signal is sent to the system configuration controller where it is processed into a system management interrupt (SMI).

Under specific conditions, a RAS recoverable error signal is generated and logged in a physical layer port, but the interrupt is not generated. More specifically, the error signal is lost on the way from the port to the system configuration controller.

The problem arises when the error signal tries to pass through a port that has been disabled. Each physical layer port has its own internal clock generator. When a port is disabled, its clock generator is off, and the error signal cannot propagate through that port.

Implication: If any physical layer ports are configured to signal errors of the RAS recoverable type, then depending on the pattern of disabled ports, the errors may be logged properly in the physical layer port, but a matching system management interrupt may not occur. Fatal error signals are not affected; they will always be transmitted successfully

Workaround: Do not disable physical layer ports, or if they have been disabled then reenable them, such that ports that may generate RAS recoverable errors have paths to send their error signals to the system configuration controller.

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

BA96. Performance Monitor Events for Hardware Prefetches Which Miss The L1 Data Cache May be Over Counted

Problem: Hardware prefetches that miss the L1 data cache but cannot be processed immediately due to resource conflicts will count and then retry. This may lead to incorrectly

incrementing the L1D_PREFETCH.MISS (event 4EH, umask 02H) event multiple times for a single miss.

Implication: The count reported by the L1D_PREFETCH.MISS event may be higher than expected.

Workaround: None identified.

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

BA97. VM Exit May Incorrectly Clear IA32_PERF_GLOBAL_CTRL [34:32]

Problem: If the "load IA32_PERF_GLOBAL_CTRL" VM-exit control is 1, a VM exit should load the IA32_PERF_GLOBAL_CTRL MSR (38FH) from the IA32_PERF_GLOBAL_CTRL field in the guest-state area of the VMCS. Due to this erratum, such a VM exit may instead clear bits 34:32 of the MSR, loading only bits 31:0 from the VMCS.

Implication: All fixed-function performance counters will be disabled after an affected VM exit, even if the VM exit should have enabled them based on the IA32_PERF_GLOBAL_CTRL field in the guest-state area of the VMCS.

Workaround: A VM monitor that wants the fixed-function performance counters to be enabled after a VM exit may do one of two things: (1) clear the "load IA32_PERF_GLOBAL_CTRL" VM-exit control; or (2) include an entry for the IA32_PERF_GLOBAL_CTRL MSR in the VM-exit MSR-load list.

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

BA98. Direct Connect Flash ROM May Become Overwritten

Problem: Under certain platform conditions, it is possible for the DC (Direct Connect) Flash ROM contents to become overwritten when glitches during power cycling on the Flash ROM interface pins are interpreted as DC Flash ROM opcodes and alter the DC Flash ROM. This erratum is only possible over the course of multiple power cycles due to the programming requirements on this interface.

Implication: If this erratum occurs, the DC Flash ROM contents could be overwritten or erased and, depending on how the DC Flash ROM is used, it could prevent the system from booting.

Workaround: A platform update and 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*.

BA99. Memory Controller Does Not Handle MCA Overwrite Rules for Software Recoverable Errors Correctly

Problem: The following overwrite rules for software recoverable errors (MCACOD form: 000F 0000 1MMM CCCC) are handled incorrectly:

1. If the first logged error is a software recoverable error SRAO (Software Recoverable Action Optional), and the second logged error is a corrected error, the IA32_MCi_STATUS MSR ADDRV bit 58 and MSCOD bits [31:16] fields should not be overwritten, but are overwritten.
2. If the first logged error is a software recoverable error SRAO, and the second logged error is a fatal error, the second error is expected to overwrite the first error, but does not overwrite the IA32_MCi_STATUS MSR MISCV bit 59.
3. If the first logged error is UCNA (Uncorrected No Action) error and the second logged error is a software recoverable SRAO error the IA32_MCi_STATUS MSR MISCV bit 59 of the second error should not overwrite bit 59 of the first error. Due to this erratum, the IA32_MCi_STATUS MSR MISCV bit 59 of the second error overwrites the MISCV bit of the first error.
4. If the first logged error is a software recoverable SRAO error with and the second logged error is an UCNA error the IA32_MCi_STATUS MSR ADDRV bit 58 of the second error should not overwrite bit 58 of the first error. Due to this erratum, the

IA32_MCi_STATUS MSR ADDRV bit 58 of the second error overwrites the ADDRV bit of the first error.

Implication: The IA32_MCi_STATUS MSR overwrite rules are not handled correctly when the software recoverable error is the first error.

Workaround: None identified.

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

BA100. Intel® QPI REUTPHRDS Register Only Records Bad Lanes

Problem: The Intel® QuickPath Interconnect (Intel® QPI) physical layer includes the REUTPHRDS (Device:0x14; Function:0x0; Offset:0x78) register which has a bit for each Rx lane, and is supposed to indicate which Rx lanes are in use and which are not. The processor however, only marks a lane as unused if the lane was actually discovered to be bad. If a port is constrained to run at something less than full width (e.g. by setting the REUTPHWCI (Device: 0x14; Function:0x0; Offset:0x88) register, then certain quadrants will not be used, but the REUTPHRDS register may report all lanes to be in use.

Implication: Due to this erratum, the REUTPHRDS register indicates only those lanes that were found to be bad and does not indicate lanes that are unused.

Workaround: Information about the unused quadrants is available in the REUTPHLMS (Device: 0x14; Function: 0x0; Offset: 0x8c) register.

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

BA101. Write-1-to-clear Does Not Work for Two Intel ® IBIST Registers when Clock Gating is Enabled

Problem: The Intel® Interconnect Built-in Self Test (Intel® IBIST) registers P_PCSR_REUTPATLEC (Device:0x14; Function:0x0; Offset:0xC4) and P_PCSR_REUTPATRES (Device:0x14, Function:0x0, Offset:0xCC) should be cleared when ones are written to these registers, however this function does work if the port is in L0/Active state and clock gating is enabled.

Implication: Write-1-to-clear operation on P_PCSR_REUTPATLEC and P_PCSR_REUTPATRES Intel IBIST registers does not work if the port is in L0/Active state and clock gating is enabled. Normally, these registers are not used while in L0/Active.

Workaround: Use the Intel IBIST registers before entering L0/Active state.

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

BA102. Memory Aliasing of Code Pages May Cause Unpredictable System Behavior

Problem: The type of memory aliasing contributing to this erratum is the case where two different logical processors have the same code page mapped with two different memory types. Specifically, if one code page is mapped by one logical processor as write-back and by another as uncacheable and certain instruction fetch timing conditions occur, the system may experience unpredictable behavior.

Implication: The type of memory aliasing contributing to this erratum is the case where two different logical processors have the same code page mapped with two different memory types. Specifically, if one code page is mapped by one logical processor as write-back and by another as uncacheable and certain instruction fetch timing conditions occur, the system may experience unpredictable behavior.

Workaround: Code pages should not be mapped with uncacheable and cacheable memory types at the same time.

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

BA103. Performance Monitor Event EPT.EPDPE_MISS May be Counted While EPT is Disabled

Problem: Performance monitor event EPT.EPDPE_MISS (Event: 4FH, Umask: 08H) is used to count Page Directory Pointer table misses while EPT (extended page tables) is enabled. Due to this erratum, the processor will count Page Directory Pointer table misses regardless of whether EPT is enabled or not.

Implication: Due to this erratum, performance monitor event EPT.EPDPE_MISS may report counts higher than expected.

Workaround: Software should ensure this event is only enabled while in EPT mode.

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

BA104. PECI Command GET_TEMP does not take into account Uncore Temperature

Problem: The PECI Command GET_TEMP returns the highest core temperature and its reported values are used to determine the temperature of the processor in order to regulate fan speed. In some instances the Uncore temperature is higher than the core temperature and in these instances the GET_TEMP command may return a lower temperature than the processor is actually experiencing. Note that the uncore temperature may tend to be higher than the core temperature only under very unique workloads, for example, if there is intensive traffic directed to the processor socket when the cores are in a deep C-state.

Implication: The PECI Command GET_TEMP may report a temperature that is lower than the actual temperature.

Workaround: It is possible for the BIOS to contain a workaround for this erratum. This workaround adds a PECI GET_UNCORE_TEMP command to allow software to read the uncore temperature in order to make the comparison (core vs. uncore) to determine the highest temperature in the processor.

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

BA105. Intel QPI Lane May Be Dropped During Full Frequency Deskew Phase of Training

Problem: A random Intel QPI Lane may be dropped during the lane deskew phase while the Intel QPI Bus is training at full frequency.

Implication: When there are multiple resets after the Intel QPI Bus has reached full speed operation there is a small chance that a lane could be dropped during the deskew phase of training. In the case of a lane being dropped this will be detected and a retry will be done until the link is established and the lane is retrained.

Workaround: None identified.

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

BA106. EOI Transaction May Not be Sent if Software Enters Core C6 During an Interrupt Service Routine

Problem: If core C6 is entered after the start of an interrupt service routine but before a write to the APIC EOI (End of Interrupt) register, and the core is woken up by an event other than a fixed interrupt source the core may drop the EOI transaction the next time APIC EOI register is written and further interrupts from the same or lower priority level will be blocked.

Implication: EOI transactions may be lost and interrupts may be blocked when core C6 is used during interrupt service routines.

Workaround: Software should check the ISR register and if any interrupts are in service only enter C1.

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

BA107. Intel® QPI and SMI Links Do Not Meet the $VT_{x\text{-}cm\text{-}ac\text{-}pin}$ Specification And May Cause Unexpected In-band Resets

Problem: The processor does not meet the Intel® QuickPath Interconnect (Intel® QPI) and Intel® Scalable Memory Interconnect (Intel® SMI) $VT_{x\text{-}cm\text{-}ac\text{-}pin}$ specification that is documented in the Intel Xeon Processor 7500 Series Datasheet, Volume 1. The datasheet documents the $VT_{x\text{-}cm\text{-}ac\text{-}pin}$ to be -0.0375 and +0.0375 of $VT_{x\text{-}diff\text{-}pp\text{-}pin}$ for the Intel® QPI and SMI links running at all speeds. The processor's $VT_{x\text{-}cm\text{-}ac\text{-}pin}$ is between the range of -0.050 and +0.050 of $VT_{x\text{-}diff\text{-}pp\text{-}pin}$.

Implication: When the processor is the transmitter and the Intel 7500 chipset is the receiver, Intel has observed unexpected Intel® QPI link behavior while the Intel® QPI link is running at 6.4 GT/s under specific system-level conditions. The behavior has been observed particularly on system designs with high attenuation on the Intel® QPI channel. Intel® QPI links may observe unexpected IBR (in-band resets) from the chipset.

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*.

BA108. System Quiesce Events Initiated While Power Events are In Progress May Cause System Hangs

Problem: BIOS initiation of a system quiesce flow via the QUIESCE_CONTROL2 MSR (51H) and exit of a system quiesce flow QUIESCE_CONTROL1 MSR (50H) via may conflict with a power event on the BSP (boot strap processor) core. Due to this conflict, the BSP core, which BIOS code runs on, may have one thread take the power event and the other thread not take the power event, resulting in a system hang.

Implication: As a result of this erratum, the system may hang after BIOS initiates a system quiesce flow.

Workaround: A BIOS code change has been identified and may be implemented to fix this erratum.

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

BA109. Uncorrected Memory Error Detected by a Memory Patrol Scrub With SMI Generated by Other Memory Controllers May Cause MCE/SMI Race Condition

Problem: BIOS may configure a SMI to be signaled when the patrol scrub engine has reached the end of scrubbing a memory range. If the SMI is generated while an uncorrected error is detected by another memory patrol scrub engine, it may result MCE/SMI race condition which may lead to system shut down.

Implication: Due to this erratum, the system may shut down.

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*.

BA110. An Intel QPI Link Layer Retry Quickly Followed by an Intel QPI Physical Layer Reset May Cause an MCE

Problem: While an Intel® QuickPath Interconnect (Intel® QPI) link is processing a link level retry requested by a remote QPI agent (due to link CRC errors), if an Intel QPI phy layer reset is triggered and aligns with a specific retry stage, a packet may get dropped and cause time out error with MCA error code, IA32_MCi_Status [15:0] encoded as a Bus and Interconnect Error with Timeout [bit 8] = 1, Cache Hierarchy Error, or Internal Timer error.

Implication: Due to this erratum, a fatal MCE may be signaled with MCA error code, IA32_MCi_Status [15:0] encoded as a Bus and Interconnect Error with Timeout [bit 8] = 1, Cache Hierarchy Error, or Internal Timer error.

Workaround: None identified

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

BA111. A Transient UECC Error on Memory Reads on Systems With Mirrored Memory May Assert MCE

Problem: In a system with mirrored memory, where multiple agents share a cache line with multiple read / write requests outstanding to that line, a transient UECC (Uncorrectable ECC) error on the mirrored master may be improperly handled resulting in the assertion of the MCE (Machine Check Error). This erratum only occurs with specific timing conflicts on the outstanding requests. Intel has not observed this erratum with any commercially available system.

Implication: As a result of this erratum, the processor home-agent may MCE instead of properly returning data from the mirror slave.

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*.

BA112. PerfMon Overflow Status Can Not be Cleared After Certain Conditions Have Occurred

Problem: Under very specific timing conditions, if software tries to disable a PerfMon counter through MSR IA32_PERF_GLOBAL_CTRL (0x38F) or through the per-counter event-select (e.g. MSR 0x186) and the counter reached its overflow state very close to that time, then due to this erratum the overflow status indication in MSR IA32_PERF_GLOBAL_STAT (0x38E) may be left set with no way for software to clear it.

Implication: Due to this erratum, software may be unable to clear the PerfMon counter overflow status indication.

Workaround: Software may avoid this erratum by clearing the PerfMon counter value prior to disabling it and then clearing the overflow status indication bit.

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

BA113. 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 (for example, 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 (for example, 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 (for example, 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 3A 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*.

BA114. Intel® QPI and Intel® SMI Drift Buffer Alarms May be Observed on the Processor

Problem: False drift buffer alarms may be observed in the PR{0-3}_CR_P_PCSR_REUTPHPLS (Device 14H,15H,16H,17H; Function 0; Offset 90H) and PZ{0-1}_CR_P_PCSR_REUTPHPLS (Device 18H, 19H; Function 0; Offset 90H) registers of the Intel® QuickPath Interconnect (Intel® QPI) and Intel® Scalable Memory Interconnect (Intel® SMI) of the processor, without CRCs logged on the port/link.

Implication: Unexpected Intel® QPI and Intel® SMI drift buffer alarms flagged by the processor.

Workaround: Drift buffer alarms should be ignored. PR{0-3}_CR_P_PCSR_REUTPHPLS.DriftBuffAlarm bit 4 and PZ{0-1}_CR_P_PCSR_REUTPHPLS.DriftBuffAlarm bit 4 are no longer supported in the Intel® Xeon® Processor 7500 series External Design Specification

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

BA115. 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*.

BA116. Stack Pushes May Not Occur Properly for Events Delivered Immediately After VM Entry to 16-Bit Software

Problem: The stack pushes for an event delivered after VM entry and before execution of an instruction in VMX nonroot operation may not occur properly. The erratum applies only if the VM entry establishes IA32_EFER.LMA = 0 and CS.D = 0 and only if the event handler is also invoked with CS.D = 0.

Implication: This erratum affects events that are pending upon completion of VM entry and that do not cause VM exits. Examples include debug exceptions, interrupts, and general-protection faults generated in virtual-8086 mode by the mode’s virtual interrupt mechanism. The erratum applies only if the VM entry is not to IA-32e mode and is to 16-bit operation, and only if the relevant handler uses 16-bit operation. The incorrect stack pushes resulting from the erratum may cause incorrect guest operation. 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*.

BA117. 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: None identified.

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

BA118. VM Entry May Clear Bytes 81H-83H on Virtual-APIC Page When "Use TPR Shadow" Is 0

Problem: VM entry should not clear bytes 81H-83H on the virtual-APIC page if the "use TPR shadow" VM-execution control is 0. Due to this erratum, VM entry will do so if the "virtualize x2APIC mode" VM-execution control is 1.

Implication: VM entries with the 0-setting of the "use TPR shadow" VM-execution control and the 1-setting of the "virtualize x2APIC mode" VM-execution control cause any nonzero data at bytes 81H-83H on the virtual-APIC page to be lost. Note that this combination of settings is not allowed; any such VM entry will fail after clearing these bytes.

Workaround: Software should always set the "use TPR shadow" VM-execution control to 1 whenever it sets that "virtualize x2APIC mode" VM-execution control to 1.

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

BA119. System Hangs Possible Due to ECC Correctable Errors with EPT and DCU 16KB Mode Enabled

Problem: Under a complex set of internal conditions, a system hang may occur in the presence of L2 ECC correctable errors that set bit [54:53] = 0x2 in IA32_MCi_STATUS for a system with EPT (Extended Page Tables) and DCU (Data Cache Unit) 16 KB mode (which is a 16 KB, 4-way, ECC-enabled mode) enabled.

Implication: This erratum may cause a system hang.

Workaround: Do not enable DCU 16KB mode in a system that uses EPT.

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

BA120. Concurrent Updates to a Segment Descriptor May be Lost

Problem: If a logical processor attempts to set the accessed bit in a code or data segment descriptor while another logical processor is modifying the same descriptor, both modifications of the descriptor may be lost.

Implication: Due to this erratum, updates to segment descriptors may not be preserved. Intel has not observed this erratum with any commercially available software or system.

Workaround: None identified.

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

BA121. INVLPG Following INVEPT or INVVPID May Fail to Flush All Translations for a Large Page

Problem: This erratum applies if the address of the memory operand of an INVEPT or INVVPID instruction resides on a page larger than 4KBytes and either (1) that page includes the low 1 MBytes of physical memory; or (2) the physical address of the memory operand matches an MTRR that covers less than 4 MBytes. A subsequent execution of INVLPG that targets the large page and that occurs before the next VM-entry instruction may fail to flush all TLB entries for the page. Such entries may persist in the TLB until the next VM-entry instruction.

Implication: Accesses to the large page between INVLPG and the next VM-entry instruction may incorrectly use translations that are inconsistent with the in-memory page tables.

Workaround: None identified.

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

BA122. A 2 MB Page Split Lock Accesses Combined With Complex Internal Events May Cause Unpredictable System Behavior

Problem: A 2 MB Page Split Lock (a locked access that spans two 2 MB large pages) coincident with additional requests that have particular address relationships in combination with a timing sensitive sequence of complex internal conditions may cause unpredictable system behavior.

Implication: This erratum may cause unpredictable system behavior. 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*.

BA123. Changes to Reserved Bits of Some Nonarchitectural MSRs May Cause Unpredictable System Behavior

Problem: Under normal circumstances, an operation fails if it attempts to modify a reserved bit of a model-specific register (MSR). Due to this erratum and for some nonarchitectural MSRs, such an attempt may cause unpredictable system behavior.

Implication: Unpredictable system behavior may occur if software attempts to modify reserved bits of some nonarchitectural MSRs. (Note that documentation of the WRMSR instruction states that “Undefined or reserved bits in an MSR should be set to values previously read.”)

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*.

BA124. 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 processor 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
- An event where an I/O read sets the IO_SMI bit but another interrupt is taken before the recognition of the SMI event
- A REP INS instruction
- An I/O read that redirects to MWAIT

Workaround: SMM handlers may get false IO_SMI indication.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*.

BA125. L1 Cache Uncorrected Errors May be Recorded as Correctable in 16K Mode

Problem: When the L1 Cache is operating in 16K redundant parity mode and a parity error occurs on both halves of the duplicated cache on the same cacheline, an uncorrectable error should be logged. Due to this erratum, the uncorrectable error will be recorded as correctable, however a machine check exception will be appropriately taken in this case.

Implication: Due to this erratum, the IA32_MCi_STATUS.UC bit will incorrectly contain a value of 0 indicating a correctable error.

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*.

BA126. Writing an Illegal Vector to the IA32_X2APIC_SELF_IPI MSR Will Hang the Processor

Problem: Writing an illegal vector (0 to 15) to the IA32_X2APIC_SELF_IPI MSR while the local APIC is in x2APIC mode will cause the processor to hang.

Implication: When this erratum occurs, the processor will hang.

Workaround: Software should not write an illegal vector to the IA32_X2APIC_SELF_IPI MSR while the local APIC is in X2APIC mode. Virtual-machine monitors should not allow guest software to write to the IA32_X2APIC_SELF_IPI MSR.

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

BA127. Successive Fixed Counter Overflows May be Discarded

Problem: Under specific internal conditions, when using Freeze PerfMon on PMI feature (bit 12 in IA32_DEBUGCTL.Freeze_PerfMon_on_PMI, MSR 1D9H), if two or more PerfMon Fixed Counters overflow very closely to each other, the overflow may be mishandled for some of them. This means that the counter's overflow status bit (in MSR_PERF_GLOBAL_STATUS, MSR 38EH) may not be updated properly; additionally, PMI interrupt may be missed if software programs a counter in Sampling-Mode (PMI bit is set on counter configuration).

Implication: Successive Fixed Counter overflows may be discarded when Freeze PerfMon on PMI is used.

Workaround: Software can avoid this by doing the following:

1. Avoid using Freeze PerfMon on PMI bit.
2. Enable only one fixed counter at a time when using Freeze PerfMon on PMI.

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

BA128. VM Exits Due to "NMI-Window Exiting" May Not Occur Following a VM Entry to the Shutdown State

Problem: If VM entry is made with the "virtual NMIs" and "NMI-window exiting", VM-execution controls set to 1, and if there is no virtual-NMI blocking after VM entry, a VM exit with exit reason "NMI window" should occur immediately after VM entry unless the VM entry put the logical processor in the wait-for SIPI state. Due to this erratum, such VM exits do not occur if the VM entry put the processor in the shutdown state.

Implication: A VMM may fail to deliver a virtual NMI to a virtual machine in the shutdown state.

Workaround: Before performing a VM entry to the shutdown state, software should check whether the "virtual NMIs" and "NMI-window exiting" VM-execution controls are both 1. If they are, software should clear "NMI-window exiting" and inject an NMI as part of VM entry.

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

BA129. Execution of INVVPID Outside 64-Bit Mode Cannot Invalidate Translations For 64-Bit Linear Addresses

Problem: Executions of the INVVPID instruction outside 64-bit mode with the INVVPID type "individual-address invalidation" ignore bits 63:32 of the linear address in the INVVPID descriptor and invalidate translations for bits 31:0 of the linear address.

Implication: The INVVPID instruction may fail to invalidate translations for linear addresses that set bits in the range 63:32. Because this erratum applies only to executions outside 64-bit mode, it applies only to attempts by a 32-bit virtual-machine monitor (VMM) to invalidate translations for a 64-bit guest. 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*.

BA130. A Combination of Data Accesses That Are Split Across Cacheline Boundaries May Lead to a Processor Hang

Problem: Under certain complex micro-architectural conditions, closely spaced data accesses that are split across cacheline boundaries may lead to a processor hang.

Implication: Due to this erratum, the processor may hang. This erratum has not been observed with any general purpose operating systems.

Workaround: None identified.

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

BA131. A Load May Appear to be Ordered Before an Earlier Locked Instruction

Problem: Under certain timing conditions involving multiple cores, a cacheable load may appear to be ordered before an earlier cacheable locked instruction that accesses a different location.

Implication: Locked instructions and subsequent loads may not occur in the expected order when run on multiple cores. In some circumstances this could lead to unpredictable system behavior.

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*.

BA132. A Machine Check Occurring During VM Entry May Cause Unpredictable Behavior

Problem: A machine check occurring during VM entry may cause the VM entry to fail. Due to this erratum, such a VM entry failure may be followed by unpredictable behavior, including a processor hang.

Implication: This erratum may result in a system hang. 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*.

BA133. MCI_ADDR May be Incorrect For Cache Parity Errors

Problem: In cases when a WBINVD instruction evicts a line containing an address or data parity error (MCACOD of 0x124, and MSCOD of 0x10), the address of this error should be logged in the MCi_ADDR register. Due to this erratum, the logged address may be incorrect, even though MCi_Status.ADDRV (bit 63) is set.

Implication: The address reported in MCi_ADDR may not be correct for cases of a parity error found during WBINVD execution.

Workaround: None identified.

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

BA134. The Corrected Error Count Overflow Bit in IA32_ MC0_STATUS is Not Updated When The UC Bit is Set

Problem: After a UC (uncorrected) error is logged in the IA32_MC0_STATUS MSR (401H), corrected errors will continue to be counted in the lower 14 bits (bits 51:38) of the Corrected Error Count. Due to this erratum, the sticky count overflow bit (bit 52) of the Corrected Error Count will not get updated when the UC bit (bit 61) is set to 1.

Implication: The Corrected Error Count Overflow indication will be lost if the overflow occurs after an uncorrectable error has been logged.

Workaround: None identified.

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

BA135. The Upper 32 Bits of CR3 May be Incorrectly Used With 32-Bit Paging

Problem: When 32-bit paging is in use, the processor should use a page directory located at the 32-bit physical address specified in bits 31:12 of CR3; the upper 32 bits of CR3 should be ignored. Due to this erratum, the processor will use a page directory located at the 64-bit physical address specified in bits 63:12 of CR3.

Implication: The processor may use an unexpected page directory or, if EPT (Extended Page Tables) is in use, cause an unexpected EPT violation. This erratum applies only if software enters 64-bit mode, loads CR3 with a 64-bit value, and then returns to 32-bit paging without changing CR3. Intel has not observed this erratum with any commercially available software.

Workaround: Software that has executed in 64-bit mode should reload CR3 with a 32-bit value before returning to 32-bit paging.

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

BA136. EPT Violations May Report Bits 11:0 of Guest Linear Address Incorrectly

Problem: If a memory access to a linear address requires the processor to update an accessed or dirty flag in a paging-structure entry and if that update causes an EPT violation, the processor should store the linear address into the “guest linear address” field in the VMCS. Due to this erratum, the processor may store an incorrect value into bits 11:0 of this field. (The processor correctly stores the guest-physical address of the paging-structure entry into the “guest-physical address” field in the VMCS.)

Implication: Software may not be easily able to determine the page offset of the original memory access that caused the EPT violation. Intel has not observed this erratum to impact the operation of any commercially available software.

Workaround: Software requiring the page offset of the original memory access address can derive it by simulating the effective address computation of the instruction that caused the EPT violation.

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

BA137. SMRAM State-Save Area Above the 4-GB Boundary May Cause Unpredictable System Behavior

Problem: If BIOS uses the RSM instruction to load the SMBASE register with a value that would cause any part of the SMRAM state-save area to have an address above 4-GBytes, subsequent transitions into and out of SMM (system-management mode) might save and restore processor state from incorrect addresses.

Implication: This erratum may cause unpredictable system behavior. Intel has not observed this erratum with any commercially available system.

Workaround: Ensure that the SMRAM state-save area is located entirely below the 4-GB address boundary.

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

BA138. Virtual-APIC Page Accesses With 32-Bit PAE Paging May Cause a System Crash

Problem: If a logical processor has EPT (Extended Page Tables) enabled, is using 32-bit PAE paging, and accesses the virtual-APIC page then a complex sequence of internal

processor micro-architectural events may cause an incorrect address translation or machine check on either logical processor.

Implication: This erratum may result in unexpected faults, an uncorrectable TLB error logged in IA32_MCi_STATUS.MCACOD (bits [15:0]) with a value of 0000_0000_0001_xxxxb (where x stands for 0 or 1), a guest or hypervisor crash, or other unpredictable system behavior.

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*.

BA139. VM Exit May Set IA32_EFER.NXE When IA32_MISC_ENABLE Bit 34 is Set to 1

Problem: When "XD Bit Disable" in the IA32_MISC_ENABLE MSR (1A0H) bit 34 is set to 1, it should not be possible to enable the "execute disable" feature by setting IA32_EFER.NXE. Due to this erratum, a VM exit that occurs with the 1-setting of the "load IA32_EFER" VM-exit control may set IA32_EFER.NXE even if IA32_MISC_ENABLE bit 34 is set to 1. This erratum can occur only if IA32_MISC_ENABLE bit 34 was set by guest software in VMX non-root operation.

Implication: Software in VMX root operation may execute with the "execute disable" feature enabled despite the fact that the feature should be disabled by the IA32_MISC_ENABLE MSR. Intel has not observed this erratum with any commercially available software.

Workaround: A virtual-machine monitor should not allow guest software to write to the IA32_MISC_ENABLE MSR.

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

BA140. Performance Monitor Counter MEM_INST_RETIRED.STORES May Count Higher than Expected

Problem: Performance Monitoring counter MEM_INST_RETIRED.STORES (Event: 0BH, Umask: 02H) 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 MEM_INST_RETIRED.STORES may report counts higher than expected.

Workaround: None identified.

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

BA141. 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*.

Specification Changes

The Specification Changes listed in this section apply to the following documents:

- *Intel® Xeon® Processor 7500 Series Datasheet, Volumes 1 and 2*
- *Intel® 64 and IA-32 Architectures Software Developer's Manual, Volume 1: Basic Architecture*
- *Intel® 64 and IA-32 Architectures Software Developer's Manual, Volume 2A: Instruction Set Reference Manual A-M*
- *Intel® 64 and IA-32 Architectures Software Developer's Manual, Volume 2B: Instruction Set Reference Manual N-Z*
- *Intel® 64 and IA-32 Architectures Software Developer's Manual, Volume 3A: System Programming Guide*
- *Intel® 64 and IA-32 Architectures Software Developer's Manual, Volume 3B: System Programming Guide*

SCh1. Removed

SCh2. Correction to Datasheet Introduction Cache Table Entry

In the *Intel® Xeon® Processor 7500 Series Datasheet, Volume 1* under section 1, stated:

Instruction Cache (L1) = 32 KB/core (I) and 16 KB/core (D)

is changing to:

Instruction Cache (L1) = 32 KB/core (I) and 32 KB/core (D)

Specification Clarifications

The Specification Clarifications listed in this section may apply to the following documents:

- *Intel® Xeon® Processor 7500 Series Datasheet, Volumes 1 and 2*.
- *Intel® 64 and IA-32 Architectures Software Developer's Manual, Volume 1: Basic Architecture*.
- *Intel® 64 and IA-32 Architectures Software Developer's Manual, Volume 2A: Instruction Set Reference Manual A-M*.
- *Intel® 64 and IA-32 Architectures Software Developer's Manual, Volume 2B: Instruction Set Reference Manual N-Z*.
- *Intel® 64 and IA-32 Architectures Software Developer's Manual, Volume 3A: System Programming Guide*.
- *Intel® 64 and IA-32 Architectures Software Developer's Manual, Volume 3B: System Programming Guide*.

There are no new Specification Clarifications in this Specification Update revision.

Documentation Changes

The Documentation Changes listed in this section apply to the following documents:

- *Intel® 64 and IA-32 Architectures Software Developer's Manual, Volume 1: Basic Architecture*.
- *Intel® 64 and IA-32 Architectures Software Developer's Manual, Volume 2A: Instruction Set Reference Manual A-M*.
- *Intel® 64 and IA-32 Architectures Software Developer's Manual, Volume 2B: Instruction Set Reference Manual N-Z*.
- *Intel® 64 and IA-32 Architectures Software Developer's Manual, Volume 3A: System Programming Guide*.
- *Intel® 64 and IA-32 Architectures Software Developer's Manual, Volume 3B: System Programming Guide*.

All Documentation Changes will be incorporated into a future version of the appropriate Processor documentation.

Note: Documentation changes for *Intel® 64 and IA-32 Architecture Software Developer's Manual* volumes 1, 2A, 2B, 3A, and 3B will be posted in a separate document, *Intel® 64 and IA-32 Architecture Software Developer's Manual Documentation Changes*. Follow the link below to become familiar with this file.

http://developer.intel.com/products/processor/manuals/index.htm

DC1. On-Demand Clock Modulation Feature Clarification

Software Controlled Clock Modulation section of the *Intel® 64 and IA-32 Architectures Software Developer's Manual, Volume 3B: System Programming Guide* will be modified to differentiate on-demand clock modulation feature on different processors. The clarification will state:

For Intel® Hyper-Threading Technology enabled processors, the IA32_CLOCK_MODULATION register is duplicated for each logical processor. In order for the on-demand clock modulation feature to work properly, the feature must be enabled on all the logical processors within a physical processor. If the programmed duty cycle is not identical for all the logical processors, the processor clock will modulate to the highest duty cycle programmed for processors with any of the following CPUID *DisplayFamily_DisplayModel* signatures (listed in Table 3). For all other processors, if the programmed duty cycle is not identical for all logical processors in the same core, the processor will modulate at the lowest programmed duty cycle.

For multiple processor cores in a physical package, each core can modulate to a programmed duty cycle independently.

For the P6 family processors, on-demand clock modulation was implemented through the chipset, which controlled clock modulation through the processor's STPCLK# pin.

Table 3. CPUID DisplayFamily_DisplayModel Signatures for Legacy Processors That Resolve to Higher Performance Setting of Conflicting Duty Cycle Requests

06_1A	06_1E	06_25	06_27	06_2E	06_35	0F_xx
06_1C	06_1F	06_26	06_2C	06_2F	06_36	

§