

# Intel® Xeon® Processor 5500 Series

Specification Update

---

*October 2009*



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

Intel products are not intended for use in medical, life saving, life sustaining, critical control or safety systems, or in nuclear facility applications.

Intel may make changes to specifications and product descriptions at any time, without notice.

Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them.

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

Hyper-Threading Technology requires a computer system with a processor supporting HT Technology and an HT Technology-enabled chipset, BIOS and operating system. Performance will vary depending on the specific hardware and software you use. For more information including details on which processors support HT Technology, see [http://www.intel.com/products/ht/hyperthreading\\_more.htm](http://www.intel.com/products/ht/hyperthreading_more.htm).

Enabling Execute Disable Bit functionality requires a PC with a processor with Execute Disable Bit capability and a supporting operating system. Check with your PC manufacturer on whether your system delivers Execute Disable Bit functionality.

64-bit computing on Intel architecture requires a computer system with a processor, chipset, BIOS, operating system, device drivers and applications enabled for Intel® 64 architecture. Performance will vary depending on your hardware and software configurations. Consult with your system vendor for more information.

Intel® Virtualization Technology requires a computer system with an enabled Intel® processor, BIOS, virtual machine monitor (VMM) and, for some uses, certain computer system software enabled for it. Functionality, performance or other benefits will vary depending on hardware and software configurations and may require a BIOS update. Software applications may not be compatible with all operating systems. Please check with your application vendor.

Intel® Turbo Boost Technology requires a PC with a processor with Intel Turbo Boost Technology capability. Intel Turbo Boost Technology performance varies depending on hardware, software and overall system configuration. Check with your PC manufacturer on whether your system delivers Intel Turbo Boost Technology. For more information, see [www.intel.com](http://www.intel.com).

The may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request. Contact your local Intel sales office or your distributor to obtain the latest specifications and before placing your product order.

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

Intel, Intel Core, Celeron, Pentium, Intel Xeon, Enhanced Intel SpeedStep technology, Intel Atom, and the Intel logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

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

Copyright © 2009, Intel Corporation. All Rights Reserved.



## Contents

---

<b>Revision History</b> .....	5
<b>Preface</b> .....	6
<b>Summary Tables of Changes</b> .....	8
<b>Identification Information</b> .....	16
<b>Errata</b> .....	20
<b>Specification Changes</b> .....	53
<b>Specification Clarifications</b> .....	54
<b>Documentation Changes</b> .....	55

§





# Revision History

---

Revision	Version	Description	Date
397809	0.10	Added Errata AAK121 and AAK122. Added Document Change AAK1 and Document Change AAK2. Updated Microcode Update Table.	April 2009
321324	-001	Public Release	March 2009
321324	-002	Added Errata AAK102 and AAK103	April 2009
321324	-003	Added Errata AAK104 and AAK105	May 2009
321324	-004	Added Errata AAK106 through AAK110	June 2009
321324	-005	Added Errata AAK111 through AAK115	July 2009
321324	-006	Added Errata AAK116 through AAK121 Added SLBGE, SLBKC and SLBGF to CPU Identification Table	September 2009
321324	-007	Added Errata AAK122 through AAK127	October 2009



# 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
<i>Intel® Xeon® Processor 5500 Series Datasheet Volume 1 and 2</i>	321321 & 321322

## Related Documents

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



## Nomenclature

**Errata** are design defects or errors. These may cause the Intel® Xeon® Processor 5500 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, etc.).



# Summary Tables of Changes

---

The following tables indicate the errata, specification changes, specification clarifications, or documentation changes which apply to the Intel® Xeon® Processor 5500 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 use 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:

- A = Dual-Core Intel® Xeon® processor 7000 sequence
- C = Intel® Celeron® processor
- D = Dual-Core Intel® Xeon® processor 2.80 GHz
- E = Intel® Pentium® III processor
- F = Intel® Pentium® processor Extreme Edition and Intel® Pentium® D processor
- I = Dual-Core Intel® Xeon® processor 5000 series
- J = 64-bit Intel® Xeon® processor MP with 1MB L2 cache
- K = Mobile Intel® Pentium® III processor



- L = Intel® Celeron® D processor
- M = Mobile Intel® Celeron® processor
- N = Intel® Pentium® 4 processor
- O = Intel® Xeon® processor MP
- P = Intel® Xeon® processor
- Q = Mobile Intel® Pentium® 4 processor supporting Hyper-Threading technology on 90-nm process technology
- R = Intel® Pentium® 4 processor on 90 nm process
- S = 64-bit Intel® Xeon® processor with 800 MHz system bus (1 MB and 2 MB L2 cache versions)
- T = Mobile Intel® Pentium® 4 processor-M
- U = 64-bit Intel® Xeon® processor MP with up to 8MB L3 cache
- V = Mobile Intel® Celeron® processor on .13 micron process in Micro-FCPGA package
- W = Intel® Celeron® M processor
- X = Intel® Pentium® M processor on 90nm process with 2-MB L2 cache and Intel® processor A100 and A110 with 512-KB L2 cache
- Y = Intel® Pentium® M processor
- Z = Mobile Intel® Pentium® 4 processor with 533 MHz system bus
- AA = Intel® Pentium® D processor 900 sequence and Intel® Pentium® processor Extreme Edition 955, 965
- AB = Intel® Pentium® 4 processor 6x1 sequence
- AC = Intel® Celeron® processor in 478 pin package
- AD = Intel® Celeron® D processor on 65nm process
- AE = Intel® Core™ Duo processor and Intel® Core™ Solo processor on 65nm process
- AF = Dual-Core Intel® Xeon® processor LV
- AG = Dual-Core Intel® Xeon® processor 5100 series
- AH = Intel® Core™2 Duo/Solo Processor for Intel® Centrino® Duo Processor Technology
- AI = Intel® Core™2 Extreme processor X6800 and Intel® Core™2 Duo desktop processor E6000 and E4000 sequence
- AJ = Quad-Core Intel® Xeon® processor 5300 series
- AK = Intel® Core™2 Extreme quad-core processor QX6000 sequence and Intel® Core™2 Quad processor Q6000 sequence
- AL = Dual-Core Intel® Xeon® processor 7100 series
- AM = Intel® Celeron® processor 400 sequence
- AN = Intel® Pentium® dual-core processor
- AO = Quad-Core Intel® Xeon® processor 3200 series
- AP = Dual-Core Intel® Xeon® processor 3000 series
- AQ = Intel® Pentium® dual-core desktop processor E2000 sequence
- AR = Intel® Celeron® processor 500 series
- AS = Intel® Xeon® processor 7200, 7300 series

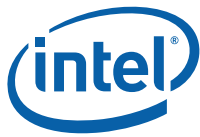


- AU = Intel® Celeron® Dual Core processor T1400
- AV = Intel® Core™2 Extreme processor QX9650 and Intel® Core™2 Quad processor Q9000 series
- AW = Intel® Core™ 2 Duo processor E8000 series
- AX = Quad-Core Intel® Xeon® processor 5400 series
- AY = Dual-Core Intel® Xeon® processor 5200 series
- AZ = Intel® Core™2 Duo processor and Intel® Core™2 Extreme processor on 45-nm process
- AAA = Quad-Core Intel® Xeon® processor 3300 series
- AAB = Dual-Core Intel® Xeon® E3110 processor
- AAC = Intel® Celeron® dual-core processor E1000 series
- AAD = Intel® Core™2 Extreme processor QX9775
- AAE = Intel® Atom™ processor Z5xx series
- AAF = Intel® Atom™ processor 200 series
- AAG = Intel® Atom™ processor N series
- AAH = Intel® Atom™ processor 300 series
- AAI = Intel® Xeon® Processor 7400 Series
- AAJ = Intel® Core™ i7 processor and Intel® Core™ i7 Extreme Edition processor
- AAK = Intel® Xeon® Processor 5500 Series
- AAL = Intel® Pentium Dual-Core Processor E5000 Series



## Errata (Sheet 1 of 5)

Number	Stepping	Status	ERRATA
	D-0		
AAK1	X	No Fix	MCi_Status Overflow Bit May Be Incorrectly Set on a Single Instance of a DTLB Error
AAK2	X	No Fix	Debug Exception Flags DR6.B0-B3 Flags May be Incorrect for Disabled Breakpoints
AAK3	X	No Fix	MONITOR or CLFLUSH on the Local XAPIC's Address Space Results in Hang
AAK4	X	No Fix	Corruption of CS Segment Register During RSM While Transitioning From Real Mode to Protected Mode
AAK5	X	No Fix	The Processor May Report a #TS Instead of a #GP Fault
AAK6	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
AAK7	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
AAK8	X	No Fix	Performance Monitor SSE Retired Instructions May Return Incorrect Values
AAK9	X	No Fix	Premature Execution of a Load Operation Prior to Exception Handler Invocation
AAK10	X	No Fix	MOV To/From Debug Registers Causes Debug Exception
AAK11	X	No Fix	Incorrect Address Computed For Last Byte of FXSAVE/FXRSTOR Image Leads to Partial Memory Update
AAK12	X	No Fix	Values for LBR/BTS/BTM will be Incorrect after an Exit from SMM
AAK13	X	No Fix	Single Step Interrupts with Floating Point Exception Pending May Be Mishandled
AAK14	X	No Fix	Fault on ENTER Instruction May Result in Unexpected Values on Stack Frame
AAK15	X	No Fix	IRET under Certain Conditions May Cause an Unexpected Alignment Check Exception
AAK16	X	No Fix	General Protection Fault (#GP) for Instructions Greater than 15 Bytes May be Preempted
AAK17	X	No Fix	General Protection (#GP) Fault May Not Be Signaled on Data Segment Limit Violation above 4-G Limit
AAK18	X	No Fix	LBR, BTS, BTM May Report a Wrong Address when an Exception/Interrupt Occurs in 64-bit Mode
AAK19	X	No Fix	Performance Monitoring Events for Read Miss to Level 3 Cache Fill Occupancy Counter may be Incorrect
AAK20	X	No Fix	A VM Exit on MWAIT May Incorrectly Report the Monitoring Hardware as Armed
AAK21	X	No Fix	Memory Aliasing of Code Pages May Cause Unpredictable System Behavior
AAK22	X	No Fix	Delivery Status of the LINT0 Register of the Local Vector Table May be Lost
AAK23	X	No Fix	Performance Monitor Event SEGMENT_REG_LOADS Counts Inaccurately
AAK24	X	No Fix	#GP on Segment Selector Descriptor that Straddles Canonical Boundary May Not Provide Correct Exception Error Code
AAK25	X	No Fix	Improper Parity Error Signaled in the IQ Following Reset When a Code Breakpoint is Set on a #GP Instruction
AAK26	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



## Errata (Sheet 2 of 5)

Number	Stepping	Status	ERRATA
	D-0		
AAK27	X	No Fix	IA32_MPERF Counter Stops Counting During On-Demand TM1
AAK28	X	No Fix	Intel® QuickPath Memory Controller May Hang If Uncorrectable ECC Errors Occur on Both Channels in Mirror Channel Mode
AAK29	X	No Fix	Simultaneous Correctable ECC Errors on Different Memory Channels With Patrol Scrubbing Enabled May Result in Incorrect Information Being Logged
AAK30	X	No Fix	Intel® QuickPath Memory Controller tTHROT_OPREF Timings May be Violated During Self Refresh Entry
AAK31	X	No Fix	Processor May Over Count Correctable Cache MESI State Errors
AAK32	X	No Fix	Synchronous Reset of IA32_APERF/IA32_MPERF Counters on Overflow Does Not Work
AAK33	X	No Fix	Disabling Thermal Monitor While Processor is Hot, Then Re-enabling, May Result in Stuck Core Operating Ratio
AAK34	X	No Fix	PECI Does Not Support PCI Configuration Reads/Writes to Misaligned Addresses
AAK35	X	No Fix	OVER Bit for IA32_MCi_STATUS Register May Get Set on Specific Internal Error
AAK36	X	No Fix	Writing the Local Vector Table (LVT) when an Interrupt is Pending May Cause an Unexpected Interrupt
AAK37	X	No Fix	Reading Reserved APIC Registers May Not Signal an APIC Error
AAK38	X	No Fix	Memory Controller May Deliver Incorrect Data When Memory Ranks Are In Power-Down
AAK39	X	No Fix	Faulting MMX Instruction May Incorrectly Update x87 FPU Tag Word
AAK40	X	No Fix	A P-state Change While Another Core is in C6 May Prevent Further C-state and P-state Transitions
AAK41	X	No Fix	Certain Store Parity Errors May Not Log Correct Address in IA32_MCi_ADDR
AAK42	X	No Fix	xAPIC Timer May Decrement Too Quickly Following an Automatic Reload While in Periodic Mode
AAK43	X	No Fix	Certain Undefined Opcodes Crossing a Segment Limit May Result in #UD Instead of #GP Exception
AAK44	X	No Fix	Indication of A20M Support is Inverted
AAK45	X	No Fix	Reported Memory Type May Not Be Used to Access the VMCS and Referenced Data Structures
AAK46	X	No Fix	B0-B3 Bits in DR6 For Non-Enabled Breakpoints May be Incorrectly Set
AAK47	X	No Fix	Core C6 May Clear Previously Logged TLB Errors
AAK48	X	No Fix	Processor May Hang When Two Logical Processors Are in Specific Low Power States
AAK49	X	No Fix	MOVNTDQA From WC Memory May Pass Earlier Locked Instructions
AAK50	X	No Fix	Performance Monitor Event MISALIGN_MEM_REF May Over Count
AAK51	X	No Fix	Changing the Memory Type for an In-Use Page Translation May Lead to Memory-Ordering Violations
AAK52	X	No Fix	In Memory Lockstep Mode Per DIMM Correctable ECC Errors Are Logged Incorrectly
AAK53	X	No Fix	Running with Write Major Mode Disabled May Lead to a System Hang
AAK54	X	No Fix	Memory Controller Address Parity Error Injection Does Not Work Correctly



## Errata (Sheet 3 of 5)

Number	Stepping	Status	ERRATA
	D-0		
AAK55	X	No Fix	Memory Controller Opportunistic Refreshes Might be Missed
AAK56	X	No Fix	Delivery of Certain Events Immediately Following a VM Exit May Push a Corrupted RIP Onto The Stack
AAK57	X	No Fix	The Combination of a Bus Lock and a Data Access That is Split Across Page Boundaries May Lead to Processor Livelock
AAK58	X	No Fix	CPUID Instruction Returns Incorrect Brand String
AAK59	X	No Fix	An Unexpected Page Fault May Occur Following the Unmapping and Re-mapping of a Page
AAK60	X	No Fix	Infinite Stream of Interrupts May Occur if an ExtINT Delivery Mode Interrupt is Received while All Cores in C6
AAK61	X	No Fix	Two xAPIC Timer Event Interrupts May Unexpectedly Occur
AAK62	X	No Fix	EOI Transaction May Not be Sent if Software Enters Core C6 During an Interrupt Service Routine
AAK63	X	No Fix	FREEZE_WHILE_SMM Does Not Prevent Event From Pending PEBS During SMM
AAK64	X	No Fix	PEBS Records For Load Latency Monitoring May Contain an Incorrect Linear Address
AAK65	X	No Fix	PEBS Field "Data Linear Address" is Not Sign Extended to 64 Bits
AAK66	X	No Fix	Core C6 May Not Operate Correctly in the Presence of Bus Locks
AAK67	X	No Fix	USB 1.1 ISOCH Audio Glitches With QPI L1 and Package C6 in a DP Configuration
AAK68	X	No Fix	Intel® Turbo Boost Technology May be Limited Immediately After Package C-state Exit with QPI L1 Mode Disabled
AAK69	X	No Fix	APIC Error "Received Illegal Vector" May be Lost
AAK70	X	No Fix	CPUID Incorrectly Indicates the UnHalted Reference Cycle Architectural Event is Supported
AAK71	X	No Fix	DR6 May Contain Incorrect Information When the First Instruction After a MOV SS,r/m or POP SS is a Store
AAK72	X	No Fix	An Uncorrectable Error Logged in IA32_CR_MC2_STATUS May also Result in a System Hang
AAK73	X	No Fix	IA32_PERF_GLOBAL_CTRL MSR May be Incorrectly Initialized
AAK74	X	No Fix	ECC Errors Can Not be Injected on Back-to-Back Writes
AAK75	X	No Fix	Performance Monitor Interrupts Generated From Uncore Fixed Counters (394H) May be Ignored
AAK76	X	No Fix	Processors with SMT May Hang on P-State Transition or ACPI Clock Modulation Throttling
AAK77	X	No Fix	Performance Monitor Counter INST_RETIRED.STORES May Count Higher than Expected
AAK78	X	No Fix	Sleeping Cores May Not be Woken Up on Logical Cluster Mode Broadcast IPI Using Destination Field Instead of Shorthand
AAK79	X	No Fix	Faulting Executions of FXRSTOR May Update State Inconsistently
AAK80	X	No Fix	Failing DIMM ID May be Incorrect in the 2DPC Configuration When Mirroring is Enabled
AAK81	X	No Fix	ISSUEONCE Bit in MC_SCRUB_CONTROL Register Does Not Work Correctly



## Errata (Sheet 4 of 5)

Number	Stepping	Status	ERRATA
	D-0		
AAK82	X	No Fix	Unexpected QPI Link Behavior May Occur When a CRC Error Happens During L0s
AAK83	X	No Fix	Performance Monitor Event EPT.EPDPPE_MISS May be Counted While EPT is Disabled
AAK84	X	No Fix	Intel® QuickPath Memory Controller Operating In Lockstep Mode May Deliver Incorrect Data on Exiting Package C3 or C6 State
AAK85	X	No Fix	Performance Monitor Counters May Count Incorrectly
AAK86	X	No Fix	Memory Thermal Throttling May Not Work as Expected in Lockstep Channel Mode
AAK87	X	No Fix	Processor Forward Progress Mechanism Interacting With Certain MSR/CSR Writes May Cause Unpredictable System Behavior
AAK88	X	No Fix	Processor May Incorrectly Demote Processor C6 State to a C3 State
AAK89	X	No Fix	Performance Monitor Event Offcore_response_0 (B7H) Does Not Count NT Stores to Local DRAM Correctly
AAK90	X	No Fix	EFLAGS Discrepancy on Page Faults and on EPT-Induced VM Exits after a Translation Change
AAK91	X	No Fix	Persistent Stream of Correctable Memory ECC Errors May Disable the Scrubbing Engine
AAK92	X	No Fix	System May Hang if MC_CHANNEL_{0,1,2}_MC_DIMM_INIT_CMD.DO_ZQCL Commands Are Not Issued in Increasing Populated DDR3 Rank Order
AAK93	X	No Fix	LER and LBR MSRs May Be Incorrectly Updated During a Task Switch
AAK94	X	No Fix	Back to Back Uncorrected Machine Check Errors May Overwrite IA32_MC3_STATUS.MSCOD
AAK95	X	No Fix	Memory Intensive Workloads with Core C6 Transitions May Cause System Hang
AAK96	X	No Fix	Corrected Errors With a Yellow Error Indication May be Overwritten by Other Corrected Errors
AAK97	X	No Fix	PSI# Signal May Incorrectly be Left Asserted
AAK98	X	No Fix	A Patrol Scrub Operation Conflicting with a Memory Self Refresh Request May Result in a System Hang
AAK99	X	No Fix	A String Instruction that Re-maps a Page May Encounter an Unexpected Page Fault
AAK100	X	No Fix	Lockstep Memory Configurations May Not Operate Correctly With Package C3/C6 States Enabled
AAK101	X	No Fix	Memory ECC Errors May be Observed When a UC Partial Write is Followed by a UC Read to the Same Location
AAK102	X	No Fix	Performance Monitor Events DCACHE_CACHE_LD and DCACHE_CACHE_ST May Overcount
AAK103	X	No Fix	Rapid Core C3/C6 Transition May Cause Unpredictable System Behavior
AAK104	X	No Fix	Performance Monitor Events INSTR_RETIRED and MEM_INST_RETIRED May Count Inaccurately
AAK105	X	No Fix	A Page Fault May Not be Generated When the PS bit is set to "1" in a PML4E or PDPTE
AAK106	X	Plan Fix	QPI Link in a DP Configuration May Hang Due to Back-Back CRC Errors
AAK107	X	Plan Fix	tRP Timing Violations May be Observed Near a Self Refresh Entry
AAK108	X	Plan Fix	Concurrent Updates to a Segment Descriptor May be Lost
AAK109	X	Plan Fix	PMIs May be Lost During Core C6 Transitions



## Errata (Sheet 5 of 5)

Number	Stepping	Status	ERRATA
	D-0		
AAK110	X	No Fix	Uncacheable Access to a Monitored Address Range May Prevent Future Triggering of the Monitor Hardware
AAK111	X	No Fix	Memory Controller Clock Circuits May Show a Temperature Sensitive Dependence on Power-On Conditions
AAK112	X	No Fix	BIST Results May be Additionally Reported After a GETSEC[WAKEUP] or INIT-SIPI Sequence
AAK113	X	No Fix	Enabling Demand/Patrol Scrubs Along With WMM May Cause Unpredictable System Behavior
AAK114	X	No Fix	Pending x87 FPU Exceptions (#MF) May be Signaled Earlier Than Expected
AAK115	X	No Fix	VM Exits Due to "NMI-Window Exiting" May Be Delayed by One Instruction
AAK116	X	Plan Fix	VM Exits Due to EPT Violations Do Not Record Information About Pre-IRET NMI Blocking
AAK117	X	No Fix	Multiple Performance Monitor Interrupts are Possible on Overflow of IA32_FIXED_CTR2
AAK118	X	No Fix	LBRs May Not be Initialized During Power-On Reset of the Processor
AAK119	X	No Fix	Unexpected Interrupts May Occur on C6 Exit If Using APIC Timer to Generate Interrupts
AAK120	X	No Fix	BTM or BTS Records May have Incorrect Branch From Information After an EIST Transition, T-states, C1E, or Adaptive Thermal Throttling
AAK121	X	No Fix	VMX-Preemption Timer Does Not Count Down at the Rate Specified
AAK122	X	No Fix	INVLPG Following INVEPT or INVVPID May Fail to Flush All Translations for a Large Page
AAK123	X	No Fix	Performance Monitoring Events STORE_BLOCKS.NOT_STA and STORE_BLOCKS.STA May Not Count Events Correctly
AAK124	X	No Fix	Storage of PEBS Record Delayed Following Execution of MOV SS or STI
AAK125	X	No Fix	Performance Monitoring Event FP_MMX_TRANS_TO_MMX May Not Count Some Transitions
AAK126	X	No Fix	The PECl Bus May be Tri-stated After System Reset
AAK127	X	No Fix	MSRs May Be Unreliable

## Specification Changes

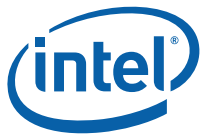
Number	SPECIFICATION CHANGES
	None for this revision of this specification update.

## Specification Clarifications

Number	SPECIFICATION CLARIFICATIONS
AAK1	Clarification of TRANSLATION LOOKASIDE BUFFERS (TLBS) Invalidation

## Documentation Changes

Number	DOCUMENTATION CHANGES
	None for this revision of this specification update



# Identification Information

## Component Identification via Programming Interface

The Intel® Xeon® Processor 5500 Series stepping can be identified by the following register contents:

Reserved	Extended Family <sup>1</sup>	Extended Model <sup>2</sup>	Reserved	Processor Type <sup>3</sup>	Family Code <sup>4</sup>	Model Number <sup>5</sup>	Stepping ID <sup>6</sup>
31:28	27:20	19:16	15:14	13:12	11:8	7:4	3:0
	00000000b	0001b		00b	0110	1010b	xxxxb

**Note:**

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 1](#) for the processor stepping ID number in the CPUID information.

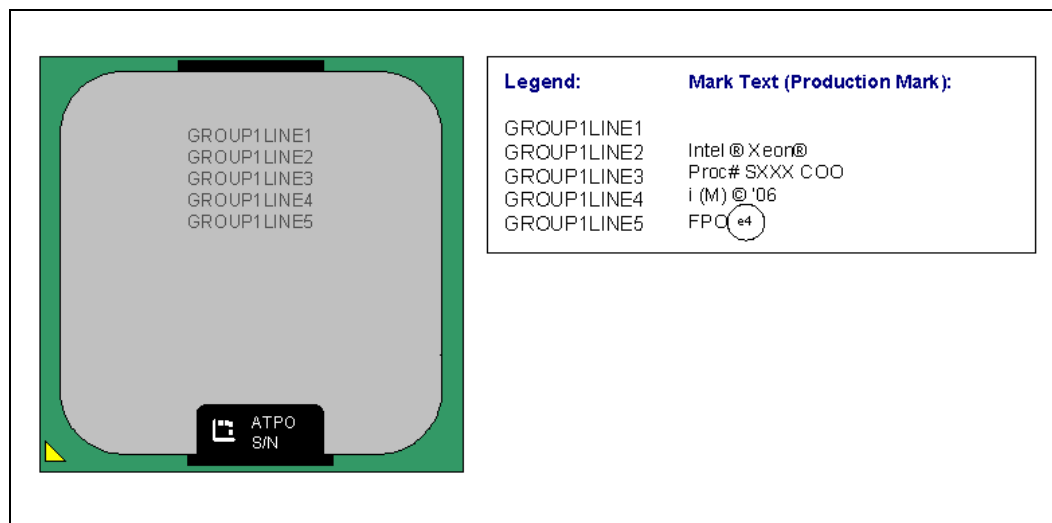
When EAX is initialized to a value of '1', the CPUID instruction returns the *Extended Family, Extended Model, Processor Type, Family Code, Model Number and Stepping ID* value in the EAX register. Note that the EDX processor signature value after reset is equivalent to the processor signature output 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 5500 Series stepping can be identified by the following component markings:

**Figure 1. Processor Top-side Markings (Example)**



**Table 1. Intel® Xeon® Processor 5500 Series Identification**

S-Spec Number	Stepping	CPUID	Core Frequency (GHz) / Intel® QuickPath Interconnect (GT/s) / DDR3 (MHz)	Available bins of Intel® Turbo Boost Technology	Cache Size (MB)	Notes
SLBF3	D-0	0x000106A5	2.93 / 6.40 / 1333	2/2/3/3	8	1, 2
SLBF4	D-0	0x000106A5	2.80 / 6.40 / 1333	2/2/3/3	8	1, 3
SLBF5	D-0	0x000106A5	2.66 / 6.40 / 1333	2/2/3/3	8	1, 4
SLBF6	D-0	0x000106A5	2.53 / 5.86 / 1066	1/1/1/2	8	1, 5
SLBF7	D-0	0x000106A5	2.40 / 5.86 / 1066	1/1/2/2	8	1, 6
SLBFD	D-0	0x000106A5	2.26 / 5.86 / 1066	1/1/2/2	8	1, 7
SLBF8	D-0	0x000106A5	2.13 / 4.80 / 800	n/a	4	1, 8
SLBF9	D-0	0x000106A5	2.00 / 4.80 / 800	n/a	4	1, 9
SLBEZ	D-0	0x000106A5	1.86 / 4.80 / 800	n/a	4	1, 10
SLBFA	D-0	0x000106A5	2.26 / 5.86 / 1066	1/1/2/2	8	1, 11
SLBFH	D-0	0x000106A5	2.13 / 4.80 / 800	n/a	4	1, 12
SLBF2	D-0	0x000106A5	3.20 / 6.40 / 1333	1/1/1/2	8	1, 13
SLBGK	D-0	0x000106A5	2.00 / 5.86 / 1066	na/na/2/3	8	1, 18
SLBFW	D-0	0x000106A5	2.13 / 5.86 / 1066	1/1/2/2	8	1, 17
SLBGE	D-0	0x000106A5	3.33 / 6.40 / 1333	1/1/1/2	8	1, 19
SLBKC	D-0	0x000106A5	2.26 / 4.80 / 800	n/a	8	1, 20
SLBGF	D-0	0x000106A5	2.40 / 5.86 / 1066	1/1/2/2	8	1, 21

Notes:



1. CPUID is 0000106Ash, where 's' is the stepping number.
2. This is an Intel® Xeon Processor X5570 with 95W TDP (Thermal Design Power).
3. This is an Intel® Xeon Processor X5560 with 95W TDP (Thermal Design Power).
4. This is an Intel® Xeon Processor X5550 with 95W TDP (Thermal Design Power).
5. This is an Intel® Xeon Processor E5540 with 80W TDP (Thermal Design Power).
6. This is an Intel® Xeon Processor E5530 with 80W TDP (Thermal Design Power).
7. This is an Intel® Xeon Processor E5520 with 80W TDP (Thermal Design Power).
8. This is an Intel® Xeon Processor E5506 with 80W TDP (Thermal Design Power).
9. This is an Intel® Xeon Processor E5504 with 80W TDP (Thermal Design Power).
10. This is an Intel® Xeon Processor E5502 with 80W TDP (Thermal Design Power).
11. This is an Intel® Xeon Processor L5520 with 60W TDP (Thermal Design Power).
12. This is an Intel® Xeon Processor L5506 with 60W TDP (Thermal Design Power).
13. This is an Intel® Xeon Processor W5580 with 130W TDP (Thermal Design Power).
14. Column indicates the number of frequency bins (133.33 MHz) of Intel® Turbo Boost Technology that are available for 4, 3, 2, or 1 cores active respectively.
15. This is an Intel® Xeon Processor 5500 Series with 60W TDP and elevated case temperature spec.
16. This is an Intel® Xeon Processor 5500 Series with 38W TDP and elevated case temperature spec.
17. This is an Intel® Xeon Processor L5518 with 60W TDP and elevated case temperature spec.
18. This is an Intel® Xeon Processor L5508 with 38W TDP and elevated case temperature spec.
19. This is an Intel® Xeon Processor W5590 with 130W TDP (Thermal Design Power).
20. This is an Intel® Xeon Processor E5507 with 80W TDP (Thermal Design Power).
21. This is an Intel® Xeon Processor L5530 with 60W TDP (Thermal Design Power).

## Mixing Processor Within DP Platforms

Intel supports dual processor (DP) configurations consisting of processors:

1. From the same power optimization segment
2. That support the same maximum Intel® QuickPath® Interconnect 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® QuickPath® 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:*

Processors must operate with the same Intel® QuickPath® Interconnect, DDR3 memory 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® 5500 series processor attributes, the following population rules and stepping matrix have been developed to clearly define supported configurations.

1. Processors must be of the same power-optimization segment. This insures processors include the same maximum Intel® QuickPath® interconnect and DDR3 operating speeds 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 (e.g a 2.93GHz / 95 W and 2.66GHz / 95W) can be operated within a system. However, both must operate 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® QuickPath® 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 AND'ing the feature flag and extended feature flag from the installed processors, any processor whose set of feature flags exactly matches the AND'ed feature flags can be selected by the BIOS as the BSP. If no processor exactly matches the AND'ed 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

---

## **AAK1. 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*.

## **AAK2. 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*.

## **AAK3. 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*.

## **AAK4. 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 CRO 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 CRO 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*.

#### **AAK5. 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*.

#### **AAK6. REP MOVSB/STOSB 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 MOVSB or REP STOSB as fast strings. Due to this erratum fast string REP MOVSB/REP STOSB 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 MOVSB or REP STOSB instruction that will execute with fast strings enabled.

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

#### **AAK7. Code Segment Limit/Canonical Faults on RSM May be Serviced before Higher Priority Interrupts/Exceptions and May Push the Wrong Address Onto the Stack**

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

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



**Workaround:** None identified.

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

### **AAK8. 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*.

### **AAK9. 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 work around 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*.

### **AAK10. MOV To/From Debug Registers Causes Debug Exception**

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

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

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



did occur in V86 mode, the exception may be directed to the general-protection exception handler.

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

#### **AAK11. 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*.

#### **AAK12. Values for LBR/BTS/BTM will be Incorrect after an Exit from SMM**

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

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

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

**Workaround:** None identified.

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

#### **AAK13. 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*.

#### **AAK14. 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*.

#### **AAK15. 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*.

#### **AAK16. General Protection Fault (#GP) for Instructions Greater than 15 Bytes May be Preempted**

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

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

**Workaround:** None identified.

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

#### **AAK17. 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 (0fffffffh) 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 (0fffffffh).

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

#### **AAK18. LBR, BTS, BTM May Report a Wrong Address when an Exception/Interrupt Occurs in 64-bit Mode**

**Problem:** An exception/interrupt event should be transparent to the LBR (Last Branch Record), BTS (Branch Trace Store) and BTM (Branch Trace Message) mechanisms. However,



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

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

**Workaround:** None identified.

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

### **AAK19. Performance Monitoring Events for Read Miss to Level 3 Cache Fill Occupancy Counter may be Incorrect**

**Problem:** Whenever an Level 3 cache fill conflicts with another request's address, the miss to fill occupancy counter, UNC\_GQ\_ALLOC.RT\_LLC\_MISS (Event 02H), will provide erroneous results.

**Implication:** The Performance Monitoring UNC\_GQ\_ALLOC.RT\_LLC\_MISS event may count a value higher than expected. The extent to which the value is higher than expected is determined by the frequency of the L3 address conflict.

**Workaround:** None identified.

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

### **AAK20. 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*.

### **AAK21. 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*.

### **AAK22. Delivery Status of the LINT0 Register of the Local Vector Table May be Lost**

**Problem:** The Delivery Status bit of the LINT0 Register of the Local Vector Table will not be restored after a transition out of C6 under the following conditions

- LINT0 is programmed as level-triggered
- The delivery mode is set to either Fixed or ExtINT
- There is a pending interrupt which is masked with the interrupt enable flag (IF)

**Implication:** Due to this erratum, the Delivery Status bit of the LINT0 Register will unexpectedly not be set. 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*.

### **AAK23. 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*.

### **AAK24. #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*.

### **AAK25. 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*.



#### **AAK26. 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*.

#### **AAK27. 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*.

#### **AAK28. Intel® QuickPath Memory Controller May Hang If Uncorrectable ECC Errors Occur on Both Channels in Mirror Channel Mode**

**Problem:** If an uncorrectable ECC error occurs on the mirrored channel before an uncorrectable ECC error on the other channel can be resolved, the Intel QuickPath Memory Controller will hang without an uncorrectable ECC error being logged.

**Implication:** The processor may hang and not report the error when uncorrectable ECC errors occur in close proximity on both channels in a mirrored channel pair. No uncorrectable ECC error will be logged in the machine check banks.

**Workaround:** None identified.



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

**AAK29. Simultaneous Correctable ECC Errors on Different Memory Channels With Patrol Scrubbing Enabled May Result in Incorrect Information Being Logged**

**Problem:** When a correctable patrol scrub ECC error occurs simultaneously with a correctable system read ECC error on different memory channels, IA32\_MCi\_STATUS and IA32\_MCi\_MISC should log the system read error. Due to this erratum IA32\_MCi\_MISC may incorrectly contain the patrol scrub error information and the IA32\_MCi\_ADDR may not be correct.

**Implication:** IA32\_MCi\_MISC and IA32\_MCi\_STATUS information may be inconsistent. IA32\_MCi\_ADDR may be incorrect.

**Workaround:** None identified.

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

**AAK30. Intel® QuickPath Memory Controller tTHROT\_OPREF Timings May be Violated During Self Refresh Entry**

**Problem:** During self refresh entry, the memory controller may issue more refreshes than permitted by tTHROT\_OPREF (bits 29:19 in MC\_CHANNEL\_{0,1,2}\_REFRESH\_TIMING CSR).

**Implication:** The intention of tTHROT\_OPREF is to limit current. Since current supply conditions near self refresh entry are not critical, there is no measurable impact due to this erratum.

**Workaround:** None identified.

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

**AAK31. Processor May Over Count Correctable Cache MESI State Errors**

**Problem:** Under a specific set of conditions, correctable Level 2 cache hierarchy MESI state errors may be counted more than once per occurrence of a correctable error.

**Implication:** Correctable Level 2 cache hierarchy MESI state errors may be reported in the MCI\_STATUS register at a rate higher than their actual occurrence.

**Workaround:** None identified.

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

**AAK32. 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*.

**AAK33. 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*.

#### **AAK34. PECl Does Not Support PCI Configuration Reads/Writes to Misaligned Addresses**

**Problem:** The PECl (Platform Environment Control Interface) specification allows for partial reads from or writes to misaligned addresses within the PCI configuration space. However, the PECl client does not properly interpret addresses that are Dword (4 byte) misaligned and may read or write incorrect data.

**Implication:** Due to this erratum, writes to or reads from Dword misaligned addresses could result in unintended side effects and unpredictable behavior.

**Workaround:** PECl host controllers may issue byte, word and Dword reads and writes as long as they are aligned to Dword addresses.

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

#### **AAK35. 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*.

#### **AAK36. 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*.



### **AAK37. 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*.

### **AAK38. Memory Controller May Deliver Incorrect Data When Memory Ranks Are In Power-Down**

**Problem:** When one or more memory ranks are in Power-Down (as controlled by MC\_CHANNEL\_{0,1,2}\_CKE\_TIMING CSR parameters), certain memory access patterns may result in incorrect data.

**Implication:** Due this erratum, incorrect data may result.

**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*.

### **AAK39. Faulting MMX Instruction May Incorrectly Update x87 FPU Tag Word**

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

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

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

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

**Workaround:** None identified.

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

### **AAK40. A P-state Change While Another Core is in C6 May Prevent Further C-state and P-state Transitions**

**Problem:** Under a specific set of conditions, when one core is in C6 and another core transitions from P<sub>n</sub> to a non-P<sub>n</sub> ratio, further C-state and P-state changes may be blocked.

**Implication:** The processor may stop responding to additional requests for deeper sleep state or ratio changes.

**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*.

### **AAK41. 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*.

#### **AAK42. 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*.

#### **AAK43. 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*.

#### **AAK44. Indication of A20M Support is Inverted**

**Problem:** The value read back from VLW\_CAPABILITY MSR (1F0H) bit [1] (A20M support) is inverted. Therefore, reading back a '1' (which should indicate A20M is supported) actually indicates A20M is not supported, and vice versa.

**Implication:** Software relying on this bit to determine whether A20M feature is supported by the processor will read back the opposite value of what is supported.

**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*.

#### **AAK45. 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*.



#### **AAK46. B0-B3 Bits in DR6 For Non-Enabled Breakpoints May be Incorrectly Set**

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

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

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

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

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

**Status:** For the steppings affected, see the [Summary Tables of Changes](#).

#### **AAK47. Core C6 May Clear Previously Logged TLB Errors**

**Problem:** Following an exit from core C6, previously logged TLB (Translation Lookaside Buffer) errors in IA32\_MCI\_STATUS may be cleared.

**Implication:** Due to this erratum, TLB errors logged in the associated machine check bank prior to core C6 entry may be cleared. Provided machine check exceptions are enabled, the machine check exception handler can log any uncorrectable TLB errors prior to core C6 entry. The TLB marks all detected errors as uncorrectable.

**Workaround:** As long as machine check exceptions are enabled, the machine check exception handler can log the TLB error prior to core C6 entry. This will ensure the error is logged before it is cleared.

**Status:** For the steppings affected, see the [Summary Tables of Changes](#).

#### **AAK48. Processor May Hang When Two Logical Processors Are in Specific Low Power States**

**Problem:** When two logical processors in a physical core have entered the C1 and C6 idle states respectively, it is possible that the processor may hang and log a machine check error with IA32\_MCI\_STATUS.MCACOD = 0x0106. The error does not occur when either core has entered C3 or when both logical processors enter the same idle state.

**Implication:** Due to this erratum, a hang may occur and a machine check may be logged while two logical processors are in a low power state.

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

**Status:** For the steppings affected, see the [Summary Tables of Changes](#).

#### **AAK49. 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*.

#### **AAK50. Performance Monitor Event MISALIGN\_MEM\_REF May Over Count**

Problem: The MISALIGN\_MEM\_REF Performance Monitoring (Event 05H) may over count memory misalignment events, possibly by orders of magnitude.

Implication: Software relying on MISALIGN\_MEM\_REF to count cache line splits for optimization purposes may read excessive number of memory misalignment events.

Workaround: None identified.

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

#### **AAK51. 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*.

#### **AAK52. In Memory Lockstep Mode Per DIMM Correctable ECC Errors Are Logged Incorrectly**

Problem: In lockstep mode, if a correctable ECC error occurs on DIMMx of one DDR3 channel, the corrected error count, MC\_COR\_ECC\_CNT {0, 1, 2, 3} will be incremented on both channels for that DIMM, instead of just the channel and DIMM on which the error has occurred.

Implication: Neither MC\_COR\_ECC\_CNT {0, 1, 2, 3} nor IA32\_MCi\_MISC Channel/DIMM fields can be used to determine which channel and DIMM the error occurred on.

Workaround: Information logged into IA32\_MCi\_MISC syndrome field can be used to decode and identify the actual failing DIMM.

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

#### **AAK53. Running with Write Major Mode Disabled May Lead to a System Hang**

Problem: With write major mode disabled, reads will be favored over writes and under certain circumstances this can lead to a system hang.

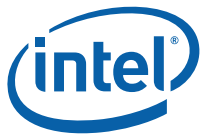
Implication: Due to this erratum a system hang may occur.

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

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

#### **AAK54. Memory Controller Address Parity Error Injection Does Not Work Correctly**

Problem: When MC\_CHANNEL\_{0,1,2}\_ECC\_ERROR\_INJECT.INJECT\_ADDR\_PARITY bit [4] = 1 an error may be injected on any command on the channel and not just RD or WR CAS commands that match MC\_CHANNEL\_{0,1,2}\_ADDR\_MATCH.



**Implication:** Address parity error injection cannot be used to reliably target a DIMM or memory location within a channel. When the address parity errors occur, the IA32\_MCI\_MISC register reflects the DIMM ID of the DIMM that detected error and not necessarily the DIMM that was targeted by the error injection settings.

**Workaround:** None identified.

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

### **AAK55. Memory Controller Opportunistic Refreshes Might be Missed**

**Problem:** If a system meets all 3 conditions below, opportunistic refresh capability might be degraded.

1. 2x refresh enabled and opportunistic refreshes enabled through tTHROT\_OPPREFRESH field in the MC\_CHANNEL\_{0,1,2}\_REFRESH\_TIMING
2. DDR3-800 DIMMS or DDR3-1066 DIMMS with tREFI value programmed more than 5% lower than the nominal value
3. More than 2 DIMMs populated

**Implication:** Due to this erratum, a corner condition can cause a persistent degradation of opportunistic refresh capability.

**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*.

### **AAK56. 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 non-maskable 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*.

### **AAK57. 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 micro-architectural 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*.



### **AAK58. CPUID Instruction Returns Incorrect Brand String**

**Problem:** When a CPUID instruction is executed with EAX = 80000002H, 80000003H and 80000004H, the return values will contain the brand string with an additional zero between the processor number and the @ symbol. (For example: Intel® Xeon® CPU nnn0 @ x.xx GHz where nnn is a processor number and x.xx is the frequency)

**Implication:** When this erratum occurs, the processor will report the incorrect brand string.

**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*.

### **AAK59. An Unexpected Page Fault May Occur Following the Unmapping and Re-mapping 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 (e.g., 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 (e.g., 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 by 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*.

### **AAK60. Infinite Stream of Interrupts May Occur if an ExtINT Delivery Mode Interrupt is Received while All Cores in C6**

**Problem:** If all logical processors in a core are in C6, an ExtINT delivery mode interrupt is pending in the xAPIC and interrupts are blocked with EFLAGS.IF=0, the interrupt will be processed after C6 wakeup and after interrupts are re-enabled (EFLAGS.IF=1). However, the pending interrupt event will not be cleared.

**Implication:** Due to this erratum, an infinite stream of interrupts will occur on the core servicing the external interrupt. Intel has not observed this erratum with any commercially available software/system.

**Workaround:** None identified.

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

### **AAK61. 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*.

**AAK62. 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 register, the core may not send an EOI transaction (if needed) and further interrupts from the same priority level or lower may be blocked.

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

**AAK63. 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*.

**AAK64. 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*.



#### **AAK65. 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 non-canonical 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:** It is possible for the BIOS to contain a workaround for this erratum.

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

#### **AAK66. Core C6 May Not Operate Correctly in the Presence of Bus Locks**

**Problem:** The processor state may be incorrect after core C6 exit if system bus locks are in progress at the time of core C6 entry.

**Implication:** The processor may begin fetching from the wrong address or have incorrect state after an exit from core C6.

**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*.

#### **AAK67. USB 1.1 ISOCH Audio Glitches With QPI L1 and Package C6 in a DP Configuration**

**Problem:** In a DP (dual-processor) configuration, when packages are entering package C6, if an QPI (Intel® QuickPath Interconnect) link master initiates the transition from any other state to L1 while the other QPI link master exits from L1, a partial serialization of QPI L1 exit latencies occurs which may cause audio glitches.

**Implication:** USB 1.1 Audio devices may have audio glitches.

**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*.

#### **AAK68. Intel® Turbo Boost Technology May be Limited Immediately After Package C-state Exit with QPI L1 Mode Disabled**

**Problem:** If the processor is resident in package C3 or C6 for greater than 100ms and QPI (Intel® QuickPath Interconnect) link L1 mode is disabled, it is possible for Turbo Boost input parameters to be incorrect. As a result, on exit from the package C-state the processor may not enter Turbo Boost for up to 2 ms.

**Implication:** Turbo Boost may be limited after exiting a package C-state (C3 and C6) that lasted longer than 100 ms.

**Workaround:** None identified.

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

#### **AAK69. 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*.

#### **AAK70. 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*.

#### **AAK71. DR6 May Contain Incorrect Information When the First Instruction After a MOV SS,r/m or POP SS is a Store**

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

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

**Workaround:** None identified.

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

#### **AAK72. 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*.

#### **AAK73. 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*.



#### **AAK74. ECC Errors Can Not be Injected on Back-to-Back Writes**

**Problem:** ECC errors should be injected on every write that matches the address set in the MC\_CHANNEL\_{0,1,2}\_ADDR\_MATCH CSRs. Due to this erratum if there are two back-to-back writes that match MC\_CHANNEL\_{0,1,2}\_ADDR\_MATCH, the 2nd write will not have the error injected.

**Implication:** The 2nd back-to-back write that matches MC\_CHANNEL\_{0,1,2}\_ADDR\_MATCH will not have the ECC error properly injected. Setting MC\_CHANNEL\_{0,1,2}\_ADDR\_MATCH to a specific address will reduce the chance of being impacted by this erratum.

**Workaround:** Only injecting errors to specific address should reduce the chance on being impacted by this erratum.

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

#### **AAK75. Performance Monitor Interrupts Generated From Uncore Fixed Counters (394H) May be Ignored**

**Problem:** Performance monitor interrupts (PMI's) from Uncore fixed counters are ignored when Uncore general performance monitor counters 3BOH-3BFH are not programmed.

**Implication:** This erratum blocks a usage model in which each of the cores can sample its own performance monitor events synchronously based on single interrupt from the Uncore.

**Workaround:** Program any one of the Uncore general performance monitor counters with a valid performance monitor event and enable the event by setting the local enable bit in the corresponding performance monitor event select MSR. For the usage model where no counting is desired, program that Uncore general performance counter's global enable bit to be zero.

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

#### **AAK76. 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*.

#### **AAK77. Performance Monitor Counter INST\_RETIRED.STORES May Count Higher than Expected**

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

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

**Workaround:** None identified.

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



#### **AAK78. 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*.

#### **AAK79. 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*.

#### **AAK80. Failing DIMM ID May be Incorrect in the 2DPC Configuration When Mirroring is Enabled**

**Problem:** When redundancy is lost in the 2DPC (2 DIMMs Per Channel) configuration, MC\_SMI\_SPARE\_DIMM\_ERROR\_STATUS CSR bits [13:12] (REDUNDANCY\_LOSS\_FAILING\_DIMM) may indicate the incorrect failing DIMM ID. The 2DPC configuration is indicated when MC\_CHANNEL\_{0,1}\_DIMM\_INIT\_PARAMS CSR bit [24] (THREE\_DIMMS\_PRESENT) is 0.

**Implication:** The failing DIMM ID may be reported incorrectly in the 2DPC configuration when mirroring is enabled. The 3DPC configuration is not affected.

**Workaround:** Only use the value in bit [13] to determine the failing DIMM ID in the non-3DPC configurations when mirroring is enabled. This workaround will show correct results for both the 1DPC and 2DPC configurations.

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

#### **AAK81. ISSUEONCE Bit in MC\_SCRUB\_CONTROL Register Does Not Work Correctly**

**Problem:** When ISSUEONCE (bit [25]) in the MC\_SCRUB\_CONTROL register (Device 3, Function 2, Offset 4CH) is set, the memory controller should issue one patrol scrub. Due to this erratum, scrubbing requests continue to be issued.

**Implication:** ISSUEONCE bit in MC\_SCRUB\_CONTROL register does not work correctly.

**Workaround:** None identified.

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



#### **AAK82. Unexpected QPI Link Behavior May Occur When a CRC Error Happens During L0s**

**Problem:** When a QPI (Intel® QuickPath Interconnect) agent requests L0s entry while a CRC (Cyclic Redundancy Check) error occurs during this flit or on the flit just before it, the requesting QPI agent may enter L0s and turn its drivers off. During this time noise on the link may be interpreted as a QPI command by the remote QPI agent, and may result in unexpected behavior.

**Implication:** Unexpected QPI link behavior may occur when CRC error happens on or just before L0s entry request.

**Workaround:** Disable L0s.

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

#### **AAK83. 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*.

#### **AAK84. Intel® QuickPath Memory Controller Operating In Lockstep Mode May Deliver Incorrect Data on Exiting Package C3 or C6 State**

**Problem:** If a processor exits from package C3 or C6 state, while operating in lockstep memory channel mode [e.g. Bit 1 set in MC\_RAS\_ENABLES (Device 3; Function 2; Offset 50h)], Intel® QuickPath Memory Controller memory accesses may result in incorrect data.

**Implication:** Occurrence of this erratum may result in loss of lockstep between memory channels, generation of an uncorrectable ECC DRAM machine check error and 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*.

#### **AAK85. 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 MSR 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](#).

#### **AAK86. Memory Thermal Throttling May Not Work as Expected in Lockstep Channel Mode**

**Problem:** Thermal Throttling on a channel that is in lockstep mode affects all channels in order to maintain lockstep requirements. If throttling parameters are modified at different times during runtime, throttling on one channel is likely to be out of phase with throttling on other channels. Throttling which is out of phase will result in more throttling than anticipated. If the throttling duty cycle exceeds 50%, certain phase relationships can result in persistent memory traffic blockage.

**Implication:** Runtime modification of throttling parameters may result in a system hang.

**Workaround:** Since Thermal Throttling on one channel affects all channels while in lockstep mode, throttling should only be applied to one channel.

**Status:** For the steppings affected, see the [Summary Tables of Changes](#).

#### **AAK87. Processor Forward Progress Mechanism Interacting With Certain MSR/CSR Writes May Cause Unpredictable System Behavior**

**Problem:** Under specific internal conditions, a mechanism within the processor to ensure forward progress may interact with writes to a limited set of MSRs/CSRs and consequently may lead to unpredictable system behavior.

**Implication:** This erratum may cause 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](#).

#### **AAK88. Processor May Incorrectly Demote Processor C6 State to a C3 State**

**Problem:** The auto demotion feature on the processor demotes processor C6 C-state requests to C3 in a more aggressive manner than expected, leading to low C6 residency.

**Implication:** Due to this erratum, the system may exhibit higher than expected idle power due to low C6 residency.

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

**Status:** For the steppings affected, see the [Summary Tables of Changes](#).

#### **AAK89. 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*.

#### **AAK90. EFLAGS Discrepancy on Page Faults and on EPT-Induced VM Exits after a Translation Change**

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

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

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

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

#### **AAK91. Persistent Stream of Correctable Memory ECC Errors May Disable the Scrubbing Engine**

**Problem:** Under a persistent stream of correctable memory ECC errors, a demand scrub conflicting with a patrol scrub operation may result in the disabling of the scrubbing engine. Intel has only observed this erratum in an artificial system environment in which ECC errors are being continuously injected.

**Implication:** When this erratum occurs, demand and patrol scrubbing are disabled until the next system reset.

**Workaround:** It is possible for the BIOS to contain a workaround for this erratum. The workaround detects a persistent stream of correctable memory ECC errors and disables demand scrubbing only while keeping patrol scrubbing enabled.

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

#### **AAK92. System May Hang if MC\_CHANNEL\_{0,1,2}\_MC\_DIMM\_INIT\_CMD.DO\_ZQCL Commands Are Not Issued in Increasing Populated DDR3 Rank Order**

**Problem:** ZQCL commands are used during initialization to calibrate DDR3 termination. A ZQCL command can be issued by writing 1 to the MC\_CHANNEL\_{0,1,2}\_MC\_DIMM\_INIT\_CMD.DO\_ZQCL (Device 4,5,6, Function 0, Offset 15, bit[15]) field and it targets the DDR3 rank specified in the RANK field (bits[7:5]) of the same register. If the ZQCL commands are not issued in increasing populated rank order then ZQ calibration may not complete, causing the system to hang.

**Implication:** Due to this erratum the system may hang if writes to the MC\_CHANNEL\_{0,1,2}\_MC\_DIMM\_INIT\_CMD.DO\_ZQCL field are not in increasing populated DDR3 rank order.



**Workaround:** It is possible for Intel provided BIOS reference code to contain a workaround for this erratum.

**Status:** For the steppings affected, see the [Summary Tables of Changes](#).

### **AAK93. LER and LBR MSR May Be Incorrectly Updated During a Task Switch**

**Problem:** LER (Last Exception Record) and LBR (Last Branch 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](#).

### **AAK94. 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](#).

### **AAK95. Memory Intensive Workloads with Core C6 Transitions May Cause System Hang**

**Problem:** Under a complex set of internal conditions, a system running a high cache stress and I/O workload combined with the presence of frequent core C6 transitions may result in a system hang.

**Implication:** Due to this erratum, the system may 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](#).

### **AAK96. 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](#).

### **AAK97. PSI# Signal May Incorrectly be Left Asserted**

**Problem:** When some of the cores in the processor are in C3/C6 state, the PSI# (Power Status Indicator) signal may incorrectly be left asserted when another core makes a frequency change request without changing the operating voltage. Since this erratum results in a



possible maximum core current greater than the PSI# threshold of 20A, PSI# should have been de-asserted.

**Implication:** Due to this erratum, platform voltage regulator tolerances may be exceeded and a subsequent system reset may occur.

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

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

#### **AAK98. A Patrol Scrub Operation Conflicting with a Memory Self Refresh Request May Result in a System Hang**

**Problem:** A memory patrol scrub operation during a memory self refresh request may result in a machine check being logged in IA32\_MC7\_STATUS.MCACOD (bits [15:0]) with value 0000\_0000\_0000\_0001 and a subsequent system hang.

**Implication:** Due to this erratum, the system may encounter a machine check and subsequently 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*.

#### **AAK99. A String Instruction that Re-maps a Page May Encounter an Unexpected Page Fault**

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 (e.g., 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 (e.g., 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.

**Problem:** 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 (e.g., 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*.

**Status:**

#### **AAK100. Lockstep Memory Configurations May Not Operate Correctly With Package C3/C6 States Enabled**

**Problem:** In a system with package C3/C6 states enabled, memory lockstep configurations may not operate correctly.



**Implication:** Lockstep capability may be lost and uncorrectable memory ECC errors may be erroneously reported.

**Workaround:** It is possible for the BIOS to contain a workaround for this erratum. Package C3/C6 states will be disabled if memory lockstep is enabled. There may be an increase in power consumption due to this workaround.

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

#### **AAK101. Memory ECC Errors May be Observed When a UC Partial Write is Followed by a UC Read to the Same Location**

**Problem:** Memory ECC errors may be observed when an uncacheable partial write is closely followed by an uncacheable read to the same location. This erratum will only occur if the BIOS is configured to use Closed\_Page memory mode (Bit 0 of MC\_CONTROL register Device 3 Function 0 Offset 48H set to 1).

**Implication:** Correctable and Uncorrectable ECC errors may be logged in the IA32\_MCi\_Status registers.

**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*.

#### **AAK102. 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*.

#### **AAK103. Rapid Core C3/C6 Transition May Cause Unpredictable System Behavior**

**Problem:** Under a complex set of internal conditions, cores rapidly performing C3/C6 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*.

#### **AAK104. 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*.

#### **AAK105. 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*.

#### **AAK106. QPI Link in a DP Configuration May Hang Due to Back-Back CRC Errors**

**Problem:** In a DP (dual-processor) configuration, under certain conditions, if two CRC (Cyclic Redundancy Check) errors occur in very close proximity on the same QPI (Intel® QuickPath Interconnect) link, LLR (Link Level Retry) signal LLR.Ack may never be returned after LLR.Req is sent. This may cause the LLR buffers on both agents to become completely full and the QPI link hard stalled.

**Implication:** In the unlikely event, if two CRC errors occur on the same QPI link in very close proximity, under certain conditions, the QPI link may hang. Intel has not observed this erratum with any commercial system.

**Workaround:** None Identified.

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

#### **AAK107. tRP Timing Violations May be Observed Near a Self Refresh Entry**

**Problem:** When entering package C3, C6 or S3 states, tRP violations may be observed near a self refresh (that is part of the C3, C6 or S3 entry).

**Implication:** tRP timing violation may occur on DRAM entry to self refresh while entering package C3, C6 or S3 states. Intel has not observed this erratum with any commercially available software. This condition has only been produced in simulation and affects a pre-charge to banks already pre-charged.

**Workaround:** None Identified.

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

#### **AAK108. 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*.



#### **AAK109. PMIs May be Lost During Core C6 Transitions**

**Problem:** If a performance monitoring counter overflows and causes a PMI (Performance Monitoring Interrupt) at the same time that the core is entering C6, then the PMI may be lost.

**Implication:** PMIs may be lost during a C6 transition.

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

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

#### **AAK110. Uncacheable Access to a Monitored Address Range May Prevent Future Triggering of the Monitor Hardware**

**Problem:** It is possible that an address range which is being monitored via the MONITOR instruction could be written without triggering the monitor hardware. A read from the monitored address range which is issued as uncacheable (for example having the CRO.CD bit set) may prevent subsequent writes from triggering the monitor hardware. A write to the monitored address range which is issued as uncacheable, may not trigger the monitor hardware and may prevent subsequent writes from triggering the monitor hardware.

**Implication:** MWAIT instruction will not exit the optimized power state and resume program flow if the monitor hardware is not triggered.

**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*.

#### **AAK111. Memory Controller Clock Circuits May Show a Temperature Sensitive Dependence on Power-On Conditions**

**Problem:** A large temperature delta between power-on and run time may affect memory controller clock circuits and subsequently could result in memory errors.

**Implication:** Correctable/Uncorrectable ECC errors may be observed on a system with memory ECC enabled. On systems that do not have memory ECC enabled, unpredictable system behavior may be observed.

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

#### **AAK112. BIST Results May be Additionally Reported After a GETSEC[WAKEUP] or INIT-SIPI Sequence**

**Problem:** BIST results should only be reported in EAX the first time a logical processor wakes up from the Wait-For-SIPI state. Due to this erratum, BIST results may be additionally reported after INIT-SIPI sequences and when waking up RLP's from the SENTER sleep state using the GETSEC[WAKEUP] command.

**Implication:** An INIT-SIPI sequence may show a non-zero value in EAX upon wakeup when a zero value is expected. RLP's waking up for the SENTER sleep state using the GETSEC[WAKEUP] command may show a different value in EAX upon wakeup than before going into the SENTER sleep state.

**Workaround:** If necessary software may save the value in EAX prior to launching into the secure environment and restore upon wakeup and/or clear EAX after the INIT-SIPI sequence.

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



### **AAK113. Enabling Demand/Patrol Scrubs Along With WMM May Cause Unpredictable System Behavior**

**Problem:** Under a specific set of conditions, enabling Demand scrubs (Bit 6, DEMAND\_SCRUB\_EN of MS\_SSRCONTROL CSR ADDR 48H) and/or Patrol Scrubs (Bits 1:0, SSR\_MODE of MS\_SSRCONTROL CSR ADDR 48H) along with WMM (Write Major Mode) may cause unpredictable system behavior.

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

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

### **AAK114. 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*.

### **AAK115. VM Exits Due to "NMI-Window Exiting" May Be Delayed by One Instruction**

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

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

**Workaround:** None Identified.

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

### **AAK116. VM Exits Due to EPT Violations Do Not Record Information About Pre-IRET NMI Blocking**

**Problem:** With certain settings of the VM-execution controls VM exits due to EPT violations set bit 12 of the exit qualification if the EPT violation was a result of an execution of the IRET instruction that commenced with non-maskable interrupts (NMIs) blocked. Due to this erratum, such VM exits will instead clear this bit.

**Implication:** Due to this erratum, a virtual-machine monitor that relies on the proper setting of bit 12 of the exit qualification may deliver NMIs to guest software prematurely.

**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*



#### **AAK117. 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*

#### **AAK118. 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*

#### **AAK119. Unexpected Interrupts May Occur on C6 Exit If Using APIC Timer to Generate Interrupts**

**Problem:** During a complex set of conditions, if the APIC timer is being used to generate interrupts, unexpected interrupts not related to the APIC timer may be signaled when a core exits the C6 power state. The APIC timer stops counting in C6 and as such isn't typically used to generate interrupts when the C6 core power state is enabled.

**Implication:** Unexpected interrupt vectors could be sent from the APIC to a logical processor. 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*

#### **AAK120. BTM or BTS Records May have Incorrect Branch From Information After an EIST Transition, T-states, C1E, or Adaptive Thermal Throttling**

**Problem:** The "From" address associated with the LBR (Last Branch Record), BTM (Branch Trace Message) or BTS (Branch Trace Store) may be incorrect for the first branch after an EIST (Enhanced Intel® SpeedStep Technology) transition, T-states, C1E (C1 Enhanced), or Adaptive Thermal Throttling.

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

**Workaround:** None identified.

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

#### **AAK121. 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](#).

### **AAK122. 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](#).

### **AAK123. 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](#).

### **AAK124. 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](#).



#### **AAK125. Performance Monitoring Event FP\_MMX\_TRANS\_TO\_MMX May Not Count Some Transitions**

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

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

**Workaround:** None identified.

**Status:** For the steppings affected, see the [Summary Tables of Changes](#)

#### **AAK126. The PECI Bus May be Tri-stated After System Reset**

**Problem:** During power-up, the processor may improperly assert the PECI (Platform Environment Control Interface) pin. This condition is cleared as soon as Bus Clock starts toggling. However, if the PECI host (also referred to as the master or originator) incorrectly determines this asserted state as another PECI host initiating a transaction, it may release control of the bus resulting in a permanent tri-state condition.

**Implication:** Due to this erratum, the PECI host may incorrectly determine that it is not the bus master and consequently PECI commands initiated by the PECI software layer may receive incorrect/invalid responses.

**Workaround:** To workaround this erratum the PECI host should pull the PECI bus low to initiate a PECI transaction. For platforms that route the PECI bus to the PCH, Intel has implemented a workaround in the PCH firmware for this erratum. For platforms that route the PECI bus to another device, please consult your third party vendor to understand the possible implications of this erratum.

**Status:** For the steppings affected, see the [Summary Tables of Changes](#)

#### **AAK127. MSRs May Be Unreliable**

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

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

**Workaround:** None identified.

**Status:** For the steppings affected, see the [Summary Tables of Changes](#)



## Specification Changes

---

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

- *Intel® Xeon® Processor 5500 Series Datasheet Volume 1 &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 Changes in this Specification Update revision.



# Specification Clarifications

---

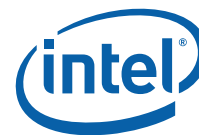
The Specification Clarifications listed in this section may apply to the following documents:

- *Intel® Xeon® Processor 5500 Series Datasheet Volume 1 & 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*

## **AAK1. Clarification of TRANSLATION LOOKASIDE BUFFERS (TLBS) Invalidation**

Section 10.9 INVALIDATING THE TRANSLATION LOOKASIDE BUFFERS (TLBS) of the Intel® 64 and IA-32 Architectures Software Developer's Manual, Volume 3A: System Programming Guide will be modified to include the presence of page table structure caches, such as the page directory cache, which Intel processors implement. This information is needed to aid operating systems in managing page table structure invalidations properly.

Intel will update the Intel® 64 and IA-32 Architectures Software Developer's Manual, Volume 3A: System Programming Guide in the coming months. Until that time, an application note, TLBs, Paging-Structure Caches, and Their Invalidation (<http://www.intel.com/products/processor/manuals/index.htm>), is available which provides more information on the paging structure caches and TLB invalidation.



## Documentation Changes

---

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

- Intel® Xeon® Processor 5500 Series *Datasheet Volume 1 & 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*

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>