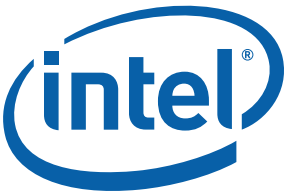


Intel® Core™ i7 Processor Extreme Edition Series and Intel® Core™ i7 Processor

Specification Update

November 2008

Reference Number: 320836-001

Contents

§

Revision History

Doc ID	Revision	Description	Date
320836	-001	Initial Release	November 2008

Preface

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

Information types defined in Nomenclature are consolidated into the specification update and are no longer published in other documents.

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

Affected Documents

Document Title	Document Number/ Location
Intel® Core™ i7 Processor Extreme Edition Series and Intel® Core™ i7 Processor Datasheet - Volume 1	320834
Intel® Core™ i7 Processor Extreme Edition Series and Intel® Core™ i7 Processor Datasheet - Volume 2	320835

Related Documents

Document Title	Document Number/ Location
AP-485, Intel® Processor Identification and the CPUID Instruction	http://www.intel.com/design/processor/applnots/241618.htm
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* *Intel® 64 and IA-32 Intel Architecture Optimization Reference Manual*	http://www.intel.com/products/processor/manuals/index.htm
Intel® 64 and IA-32 Architectures Software Developer's Manual Documentation Changes	http://www.intel.com/design/processor/specupdt/252046.htm
ACPI Specifications	www.acpi.info

Nomenclature

Errata are design defects or errors. These may cause the Intel® Core™ i7 Processor 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® Core™ i7 Processor product. Intel may fix some of the errata in a future stepping of the component, and account for the other outstanding issues through documentation or specification changes as noted. These tables uses the following notations:

Codes Used in Summary Tables

Stepping

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

Page

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

Status

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

Row

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

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

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 and Intel® Core™ i7 Extreme Edition

AAL = Intel® Pentium Dual-Core Processor E5000 Series

Errata (Sheet 1 of 3)

Number	Steppings	Status	ERRATA
	C-0		
AAJ1	X	No Fix	MCi_Status Overflow Bit May Be Incorrectly Set on a Single Instance of a DTLB Error
AAJ2	X	No Fix	Debug Exception Flags DR6.B0-B3 Flags May be Incorrect for Disabled Breakpoints
AAJ3	X	No Fix	MONITOR or CLFLUSH on the Local XAPIC's Address Space Results in Hang
AAJ4	X	No Fix	Corruption of CS Segment Register During RSM While Transitioning From Real Mode to Protected Mode
AAJ5	X	No Fix	The Processor May Report a #TS Instead of a #GP Fault
AAJ6	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
AAJ7	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
AAJ8	X	No Fix	Performance Monitor SSE Retired Instructions May Return Incorrect Values
AAJ9	X	No Fix	Premature Execution of a Load Operation Prior to Exception Handler Invocation
AAJ10	X	No Fix	MOV To/From Debug Registers Causes Debug Exception
AAJ11	X	No Fix	Incorrect Address Computed For Last Byte of FXSAVE/FXRSTOR Image Leads to Partial Memory Update
AAJ12	X	No Fix	Values for LBR/BTS/BTM will be Incorrect after an Exit from SMM
AAJ13	X	No Fix	Single Step Interrupts with Floating Point Exception Pending May Be Mishandled
AAJ14	X	No Fix	Fault on ENTER Instruction May Result in Unexpected Values on Stack Frame
AAJ15	X	No Fix	IRET under Certain Conditions May Cause an Unexpected Alignment Check Exception
AAJ16	X	No Fix	General Protection Fault (#GP) for Instructions Greater than 15 Bytes May be Preempted
AAJ17	X	No Fix	General Protection (#GP) Fault May Not Be Signaled on Data Segment Limit Violation above 4-G Limit
AAJ18	X	No Fix	LBR, BTS, BTM May Report a Wrong Address when an Exception/Interrupt Occurs in 64-bit Mode
AAJ19	X	No Fix	Performance Monitoring Events for Read Miss to Level 3 Cache Fill Occupancy Counter may be Incorrect
AAJ20	X	No Fix	A VM Exit on MWAIT May Incorrectly Report the Monitoring Hardware as Armed
AAJ21	X	No Fix	Use of Memory Aliasing With Inconsistent Memory Type May Cause Unpredictable System Behavior
AAJ22	X	No Fix	Delivery Status of the LINT0 Register of the Local Vector Table May be Lost
AAJ23	X	No Fix	Performance Monitor Event SEGMENT_REG_LOADS Counts Inaccurately
AAJ24	X	No Fix	#GP on Segment Selector Descriptor that Straddles Canonical Boundary May Not Provide Correct Exception Error Code
AAJ25	X	No Fix	Improper Parity Error Signaled in the IQ Following Reset When a Code Breakpoint is Set on a #GP Instruction
AAJ26	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 3)

Number	Steppings	Status	ERRATA
	C-0		
AAJ27	X	No Fix	IA32_MPERF Counter Stops Counting During On-Demand TM1
AAJ28	X	No Fix	Intel® QuickPath Memory Controller tTHROT_OPREF Timings May be Violated During Self Refresh Entry
AAJ29	X	No Fix	Processor May Over Count Correctable Cache MESI State Errors
AAJ30	X	No Fix	Synchronous Reset of IA32_APERF/IA32_MPERF Counters on Overflow Does Not Work
AAJ31	X	No FIx	Disabling Thermal Monitor While Processor is Hot, Then Re-enabling, May Result in Stuck Core Operating Ratio
AAJ32	X	Plan Fix	The PECI Throttling Counter May Not be Accurate
AAJ33	X	No Fix	PECI Does Not Support PCI Configuration Reads/Writes to Misaligned Addresses
AAJ34	X	No Fix	OVER Bit for IA32_MCi_STATUS Register May Get Set on Specific Internal Error
AAJ35	X	No Fix	Writing the Local Vector Table (LVT) when an Interrupt is Pending May Cause an Unexpected Interrupt
AAJ36	X	Plan Fix	A Processor Core May Not Wake Up from S1 State
AAJ37	X	Plan Fix	Reading Reserved APIC Registers May Not Signal an APIC Error
AAJ38	X	Plan Fix	A Logical Processor Receiving a SIPI After a VM Entry Into WFS State May Become Unresponsive
AAJ39	X	No Fix	Memory Controller May Deliver Incorrect Data When Memory Ranks Are In Power-Down
AAJ40	X	No Fix	Faulting MMX Instruction May Incorrectly Update x87 FPU Tag Word
AAJ41	X	Plan Fix	A Floating-Point Store Instruction May Cause an Unexpected x87 FPU Floating-Point Error (#MF)
AAJ42	X	Plan Fix	Incorrect TLB Translation May Occur After Exit From C6
AAJ43	X	Plan Fix	USB 1.1 ISOCH Audio Glitches with Intel® QuickPath Interconnect Locks and Deep C-States
AAJ44	X	Plan Fix	Stack Pointer May Become Incorrect In Loops With Unbalanced Push and Pop Operations
AAJ45	X	No Fix	A P-state Change While Another Core is in C6 May Prevent Further C-state and P-state Transitions
AAJ46	X	Plan Fix	Certain Store Parity Errors May Not Log Correct Address in IA32_MCi_ADDR
AAJ47	X	No Fix	xAPIC Timer May Decrement Too Quickly Following an Automatic Reload While in Periodic Mode
AAJ48	X	No Fix	Certain Undefined Opcodes Crossing a Segment Limit May Result in #UD Instead of #GP Exception
AAJ49	X	No Fix	Indication of A20M Support is Inverted
AAJ50	X	No Fix	Reported Memory Type May Not Be Used to Access the VMCS and Referenced Data Structures
AAJ51	X	Plan Fix	After VM Entry, Instructions May Incorrectly Operate as if CS.D=0
AAJ52	X	Plan Fix	Spurious Machine Check Error May Occur When Logical Processor is Woken Up
AAJ53	X	No Fix	B0-B3 Bits in DR6 For Non-Enabled Breakpoints May be Incorrectly Set
AAJ54	X	No Fix	Core C6 May Clear Previously Logged TLB Errors

Errata (Sheet 3 of 3)

Number	Steppings	Status	ERRATA
	C-0		
AAJ55	X	Plan Fix	Processor May Hang When Two Logical Processors Are in Specific Low Power States
AAJ56	X	Plan Fix	MOVNTDQA From WC Memory May Pass Earlier Locked Instructions
AAJ57	X	No Fix	Performance Monitor Event MISALIGN_MEM_REF May Over Count
AAJ58	X	No Fix	Changing the Memory Type for an In-Use Page Translation May Lead to Memory-Ordering Violations
AAJ59	X	Plan Fix	Writes to IA32_CR_PAT or IA32_EFER MSR May Cause an Incorrect ITLB Translation
AAJ60	X	Plan Fix	The "Virtualize APIC Accesses" VM-Execution Control May be Ignored
AAJ61	X	Plan Fix	C6 Transitions May Cause Spurious Updates to the xAPIC Error Status Register
AAJ62	X	Plan Fix	Critical ISOCH Traffic May Cause Unpredictable System Behavior When Write Major Mode Enabled
AAJ63	X	Plan Fix	Running with Write Major Mode Disabled May Lead to a System Hang
AAJ64	X	No Fix	Memory Controller Address Parity Error Injection Does Not Work Correctly
AAJ65	X	No Fix	Memory Controller Opportunistic Refreshes Might be Missed
AAJ66	X	Plan Fix	Delivery of Certain Events Immediately Following a VM Exit May Push a Corrupted RIP Onto The Stack
AAJ67	X	Plan Fix	The Combination of a Bus Lock and a Data Access That is Split Across Page Boundaries May Lead to Processor Livelock
AAJ68	X	Plan Fix	CPUID Instruction Returns Incorrect Brand String
AAJ69	X	Plan Fix	An Unexpected Page Fault or EPT Violation May Occur Following the Unmapping and Re-mapping of a Page
AAJ70	X	No Fix	Infinite Stream of Interrupts May Occur if an ExtINT Delivery Mode Interrupt is Received while All Cores in C6
AAJ71	X	No Fix	Two xAPIC Timer Event Interrupts May Unexpectedly Occur
AAJ72	X	No Fix	EOI Transaction May Not be Sent if Software Enters Core C6 During an Interrupt Service Routine
AAJ73	X	No Fix	FREEZE_WHILE_SMM Does Not Prevent Event From Pending PEBS During SMM
AAJ74	X	No Fix	PEBS Records For Load Latency Monitoring May Contain an Incorrect Linear Address
AAJ75	X	No Fix	PEBS Field “Data Linear Address” is Not Sign Extended to 64 Bits
AAJ76	X	Plan Fix	Core C6 May Not Operate Correctly in the Presence of Bus Locks
AAJ77	X	No Fix	Intel® Turbo Boost Technology May be Limited Immediately After Package C-state Exit with QPI L1 Mode Disabled

Specification Changes

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

Specification Clarifications

Number	SPECIFICATION CLARIFICATIONS
AAJ1	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® Core™ i7 Processor stepping can be identified by the following register contents:

Reserved	Extended Family[1]	Extended Model[2]	Reserved	Processor Type[3]	Family Code[4]	Model Number[5]	Stepping ID[6]
31:28	**27:20**	**19:16**	**15:14**	**13:12**	**11:8**	**7:4**	**3:0**
	00000000b	0001b		00b	0110	1010b	0011b

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, or Intel® Core™ processor family.
2. The Extended Model, bits [19:16] in conjunction with the Model Number, specified in bits [7:4], are used to identify the model of the processor within the processor's family.
3. The 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.

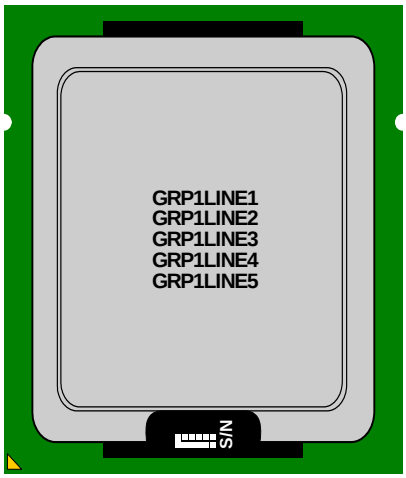
When EAX is initialized to a value of '1', the CPUID instruction returns the *Extended Family, Extended Model, Processor Type, Family Code, Model Number and Stepping ID* value in the EAX register. Note that the EDX processor signature value after reset is equivalent to the processor signature output 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® Core™ i7 Processor stepping can be identified by the following component markings:

Figure 1. Processor Top-side Markings (Example)



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

XXXXX = Country of Origin

Table 1. Processor Identification

S-Spec Number	Stepping	CPUID	Core Frequency (GHz) / Intel QuickPath Interconnect (GT/s) / DDR3 (MHz)	Available bins of Intel® Turbo Boost Technology[2]	Cache Size (MB)	Notes
SLBCJ	C-0	0x000106A4	3.20 / 6.40/ 1066	1/1/1/2	8	1
SLBCK	C-0	0x000106A4	2.93 / 4.80/ 1066	1/1/1/2	8	
SLBCH	C-0	0x000106A4	2.66 / 4.80/ 1066	1/1/1/2	8	

1. Although these units are factory-configured for 1333 MHz integrated memory controller frequency, Intel does not support operation beyond 1066 MHz; however, this processor has additional support to override the integrated memory controller frequency.
2. 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.

Errata

AAJ1. 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*.

AAJ2. 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*.

AAJ3. 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*.

AAJ4. Corruption of CS Segment Register During RSM While Transitioning From Real Mode to Protected Mode

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

Implication: The corruption of the bottom two bits of the CS segment register will have no impact unless software explicitly examines the CS segment register between enabling protected mode and the first FAR JMP. *Intel® 64 and IA-32 Architectures Software Developer's Manual Volume 3A: System Programming Guide, Part 1*, in the section titled "Switching to Protected Mode" recommends the FAR JMP immediately follows the

write to CR0 to enable protected mode. Intel has not observed this erratum with any commercially available software.

Workaround: None identified.

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

AAJ5. 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*.

AAJ6. REP MOVS/STOS Executing with Fast Strings Enabled and Crossing Page Boundaries with Inconsistent Memory Types may use an Incorrect Data Size or Lead to Memory-Ordering Violations

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

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

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

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

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

AAJ7. 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*.

AAJ8. 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*.

AAJ9. Premature Execution of a Load Operation Prior to Exception Handler Invocation

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

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

Implication: In normal code execution where the target of the load operation is to write back memory there is no impact from the load being prematurely executed, or from the restart and subsequent re-execution of that instruction by the exception handler. If the target of the load is to uncached memory that has a system side-effect, restarting the instruction may cause unexpected system behavior due to the repetition of the side-effect. Particularly, while CR0.TS [bit 3] is set, a MOVD/MOVQ with MMX/XMM register operands may issue a memory load before getting the DNA exception.

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

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

AAJ10. 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*.

AAJ11. 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*.

AAJ12. 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*.

AAJ13. 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*.

AAJ14. Fault on ENTER Instruction May Result in Unexpected Values on Stack Frame

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

Implication: Data in the created stack frame may be altered following a fault on the ENTER instruction. Please refer to "Procedure Calls For Block-Structured Languages" in *IA-32*

Intel® Architecture Software Developer's Manual, Vol. 1, Basic Architecture, for information on the usage of the ENTER instructions. This erratum is not expected to occur in ring 3. Faults are usually processed in ring 0 and stack switch occurs when transferring to ring 0. Intel has not observed this erratum on any commercially available software.

Workaround: None identified.

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

AAJ15. 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*.

AAJ16. 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*.

AAJ17. General Protection (#GP) Fault May Not Be Signaled on Data Segment Limit Violation above 4-G Limit

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

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

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

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

AAJ18. LBR, BTS, BTM May Report a Wrong Address when an Exception/ Interrupt Occurs in 64-bit Mode

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

during a specific boundary condition where the exception/interrupt occurs right after the execution of an instruction at the lower canonical boundary (0x00007FFFFFFFFFFF) in 64-bit mode, the LBR return registers will save a wrong return address with bits 63 to 48 incorrectly sign extended to all 1's. Subsequent BTS and BTM operations which report the LBR will also be incorrect.

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

Workaround: None identified.

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

AAJ19. 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*.

AAJ20. 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*.

AAJ21. Use of Memory Aliasing With Inconsistent Memory Type May Cause Unpredictable System Behavior

Problem: When a code page is being accessed as uncacheable and write back, under certain instruction fetch and memory timing conditions, the system may have unpredictable behavior.

Implication: The aliasing of memory regions, a condition necessary for this erratum to occur, is documented as being unsupported in the *Intel 64 and IA-32 Intel® Architecture Software Developer's Manual*, Volume 3A, in the section titled Programming the PAT. However, if this erratum occurs the system may have unpredictable behavior. Intel has not observed this erratum with any commercially available software or system.

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*.

AAJ22. 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*.

AAJ23. 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 identifed.

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

AAJ24. #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*.

AAJ25. 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*.

AAJ26. An Enabled Debug Breakpoint or Single Step Trap May Be Taken after MOV SS/POP SS Instruction if it is Followed by an Instruction That Signals a Floating Point Exception

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

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

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

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

AAJ27. 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*.

AAJ28. 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*.

AAJ29. 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*.

AAJ30. 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*.

AAJ31. 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*.

AAJ32. The PECI Throttling Counter May Not be Accurate

Problem: Under certain throttling circumstances, the PECI (Platform Environment Control Interface) throttling counter may not be accurate. If the throttling counter is zero, then the counter accurately reflects that throttling never occurred.

Implication: If the PECI throttle counter is non-zero, it may not be accurate.

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*.

AAJ33. PECI Does Not Support PCI Configuration Reads/Writes to Misaligned Addresses

Problem: The PECI (Platform Environment Control Interface) specification allows for partial reads from or writes to misaligned addresses within the PCI configuration space. However, the PECI 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: PECI 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*.

AAJ34. 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*.

AAJ35. 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*.

AAJ36. A Processor Core May Not Wake Up from S1 State

Problem: If there is an interrupt pended at the same time as the package is entering S1 and one of the cores in the package is entering C3, it is possible that the core entering C3 may not wake up from the S1 state.

Implication: Due to this erratum, the processor core may not wake up from S1 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*.

AAJ37. 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*.

AAJ38. A Logical Processor Receiving a SIPI After a VM Entry Into WFS State May Become Unresponsive

Problem: A logical processor may become unresponsive after receiving a SIPI (Start-up Interprocessor Interrupt) following a VM Entry into a WFS (Wait-for-SIPI) state.

Implication: The logical processor that receives a SIPI while in the WFS state may stop responding.

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*.

AAJ39. 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*.

AAJ40. 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*.

AAJ41. A Floating-Point Store Instruction May Cause an Unexpected x87 FPU Floating-Point Error (#MF)

Problem: If a floating-point store instruction (FST or FSTP) causes an inexact-result exception (#P) and such exceptions are unmasked, the next “waiting” x87 FPU instruction or WAIT/FWAIT instruction will incur an x87 FPU Floating-Point Error (#MF). Due to this erratum, the #MF may occur prematurely and prevent the floating-point store instruction from executing. This may occur when the logical processor is in VMX non-root operation and either (1) the “use EPT” VM-execution control is 1; or (2) the “virtual APIC accesses” VM-execution control is 1 and the store is to the APIC-access page.

Implication: Due to this erratum, a floating-point store instruction may cause a #MF that should be held pending until the next “waiting” x87 FPU instruction or WAIT/FWAIT instruction.

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*.

AAJ42. Incorrect TLB Translation May Occur After Exit From C6

Problem: Under certain conditions when C6 and two logical processors on the same core are enabled on a processor, an instruction fetch occurring after a logical processor exits from C6 may incorrectly use the translation lookaside buffer (TLB) address mapping belonging to the other logical processor in the processor core.

Implication: When this erratum occurs, unpredictable behavior 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*.

AAJ43. USB 1.1 ISOCH Audio Glitches with Intel® QuickPath Interconnect Locks and Deep C-States

Problem: An interrupt directed at a Core in C3 or C6 that collides with an Intel® QuickPath Interconnect Lock sequence may delay ISOCH transactions to DRAM long enough to underrun USB 1.1 buffers.

Implication: USB 1.1 Audio devices may have audio glitches.

Workaround: None identified.

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

AAJ44. Stack Pointer May Become Incorrect In Loops With Unbalanced Push and Pop Operations

Problem: If a loop has an unbalanced number of Push and Pop operations, under a specific set of conditions, it is possible that the stack pointer (SP/ESP/RSP) may become incorrect.

Implication: When this erratum occurs, unpredictable behavior may result. Intel has not observed this erratum with any commercially available software.

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*.

AAJ45. 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 Pn to a non-Pn 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*.

AAJ46. 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*.

AAJ47. 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*.

AAJ48. 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*.

AAJ49. 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*.

AAJ50. 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*.

AAJ51. After VM Entry, Instructions May Incorrectly Operate as if CS.D=0

Problem: If bit 13 (L) and bit 14 (D/B) of the guest CS access rights field in the VMCS are both 1 and VM entry takes the processor out of IA-32e mode, instructions executed after VM entry may operate as if CS.D=0.

Implication: Instructions executed after VM entry may use the wrong operation size. Intel has not observed this erratum with any commercially available system.

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*.

AAJ52. Spurious Machine Check Error May Occur When Logical Processor is Woken Up

Problem: The first time a logical processor is woken up after power on (including resume from system sleep states) an Internal Parity Error may be detected and logged when no real parity error occured.

Implication: When this erratum occurs, a spurious Internal Parity Error may be logged. However, no machine check exception will be signaled in this case.

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*.

AAJ53. 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*.

AAJ54. 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*.

AAJ55. 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*.

AAJ56. 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*.

AAJ57. 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*.

AAJ58. 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*.

AAJ59. Writes to IA32_CR_PAT or IA32_EFER MSR May Cause an Incorrect ITLB Translation

Problem: Under certain conditions, writes to IA32_CR_PAT (277H) or IA32_EFER (C0000080H) MSRs may result in an incorrect ITLB (instruction translation lookaside buffer) translation.

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

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

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

AAJ60. The "Virtualize APIC Accesses" VM-Execution Control May be Ignored

Problem: If a VM exit occurs while the “virtualize APIC accesses” and “enable VPID” VM-execution controls are both 1 and the VM-exit MSR-store count is not 0, the logical processor may operate as if the “virtualize APIC accesses” VM-execution control was 0 following a subsequent VM entry.

Implication: This erratum may prevent VMM software from virtualizing memory-mapped APIC accesses if it is using VPIDs (virtual-processor identifiers) and is saving MSRs on VM exits.

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*.

Status:

AAJ61. C6 Transitions May Cause Spurious Updates to the xAPIC Error Status Register

Problem: If any of the LVT entries are not initialized, reads from xAPIC Error Status Register following a C6 transition may report a spurious illegal vector received.

Implication: Due to this erratum, reads to xAPIC Error Status Register may report illegal vector received when none was actually received.

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*.

AAJ62. Critical ISOCH Traffic May Cause Unpredictable System Behavior When Write Major Mode Enabled

Problem: Under a specific set of conditions, critical ISOCH (isochronous) traffic may cause unpredictable system behavior with write major mode enabled.

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

Workaround: Write major mode must be disabled in the BIOS by writing the write major mode threshold value to its maximum value of 1FH in ISOCHEXITTRESHOLD bits [19:15], ISOCHENTRYTHRESHOLD bits [14:10], WMENTRYTHRESHOLD bits [9:5], and WMEXITTHRESHOLD bits [4:0] of the MC_CHANNEL_{0,1,2}_WAQ_PARAMS register.

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

AAJ63. 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.

AAJ64. 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.

AAJ65. 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.

AAJ66. 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.

AAJ67. 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.

AAJ68. CPUID Instruction Returns Incorrect Brand String

Problem: When a CPUID instruction is executed with EAX = 80000002H, 80000003H and 80000004H, the return values contain the brand string Intel(R) Core(TM) CPU when it should have Intel(R) Core(TM) i7 CPU. In addition, the processor number will be

incorrect. The return value will be will have have an additional zero between the processor number and the @ symbol (for example: Intel(R) Core(TM) 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: None identified.

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

Status:

AAJ69. An Unexpected Page Fault or EPT Violation May Occur Following the Unmapping and Re-mapping of a Page

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

- The paging structures initially specify 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).
- There is a subsequent load 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 may cause a page fault or EPT violation that indicates that there is no translation for the page (e.g., with bit 0 clear in the page-fault error code, indicating that the fault was caused by a not-present page).

Implication: Software may see an unexpected page fault or EPT violation 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.

AAJ70. 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.

AAJ71. 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.

AAJ72. 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.

AAJ73. 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.

AAJ74. 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.

AAJ75. 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: None identified.

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

AAJ76. 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.

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.

AAJ77. 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.

Specification Changes

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

- *Intel® Core™ i7 Processor Datasheet*
- *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® Core™ i7 Processor Datasheet*
- *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*

AAJ1. 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.

In rare instances, improper TLB invalidation may result in unpredictable system behavior, such as system hangs or incorrect data. Developers of operating systems should take this documentation into account when designing TLB invalidation algorithms. For the processors affected, Intel has provided a recommended update to system and BIOS vendors to incorporate into their BIOS to resolve this issue.

Documentation Changes

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

- *Intel® Core™ i7 Processor Datasheet*
- *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

There are no new Documentation Changes in this Specification Update revision.

§