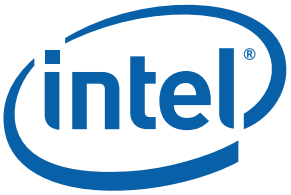

Intel® Core™ i7-900 Desktop Processor Extreme Edition Series and Intel® Core™ i7-900 Desktop Processor Series on 32-nm Process

Specification Update

September 2015

Document Number: 323254-024US

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

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

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

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

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

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

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

This document contains information on products in the design phase of development. Do not finalize a design with this information. Revised information will be published when the product is available. Verify with your local sales office that you have the latest datasheet before finalizing a design.

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 http://www.intel.com/technology/turboboost

Intel® 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/info/hyperthreading.

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.

No computer system can provide absolute security under all conditions. Intel® Trusted Execution Technology (Intel® TXT) requires a computer system with Intel® Virtualization Technology, an Intel TXT-enabled processor, chipset, BIOS, Authenticated Code Modules and an Intel TXT-compatible measured launched environment (MLE). The MLE could consist of a virtual machine monitor, an OS or an application. In addition, Intel TXT requires the system to contain a TPM v1.2, as defined by the Trusted Computing Group and specific software for some uses. For more information, see http://www.intel.com/technology/security/

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 for details.

Intel, Intel Core, Pentium, 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.

© 2015 Intel Corporation

Contents

§ §

Revision History

Revision	Description	Date
001	• Initial Release	March 2010
002	• Added new erratum BC83	May 2010
003	• Added new Erratum BC84, BC85, BC86 • Added new processor sku i7-970	July 2010
004	• Added new Erratum BC87 • Added new Erratum BC88 • Updated Erratum BC38	September 2010 October 2010
005	• Added new Errata BC90 and BC91 • Deleted Errata BC89	December 2010
006	• Added new Errata BC92and BC93	January 2011
007	• Added new processor sku i7-990X • Added new Errata BC94 to BC98	February 2011
008	• Added new Errata BC99 and BC100	May 2011
009	• Added new processor Sku i7-980 and i7-980X	June 2011
010	• Added new erratum BC101	September 2011
011	• Updated Erratum BC93	October 2011
012	• Added new erratum BC102	March 2012
013	• Added new Errata BC103-BC110	June 2012
014	• Added new Erratum BC111	December 2012
015	• Added Documentation Change BC1	January 2013
016	• Added new erratum BC112	May 2013
017	• Added new errata BC113-115	June 2013
018	• Added new errata BC116	August 2013
019	• No errata added or deleted • Document standardization	October 2013
020	• No errata added or deleted • Document standardization	December 2013
021	• Updated link to access Intel® 64 and IA-32 Architecture Software Developer's Manual Documentation Changes.	July 2014
022	• Removed Erratum BC71 • Updated Erratum BC45	November 2014
023	• Updated Erratum BC112	February 2015
024	• Added new Errata BC117	September 2015

§ §

Preface

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

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

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

Affected Documents

Document Title	Document Number/Location
Intel® Core™ i7-900 Desktop Processor Extreme Edition Series on 32-nm Process Datasheet, Volume 1	323252-003
Intel® Core™ i7-900 Desktop Processor Extreme Edition Series on 32-nm Process Datasheet, Volume 2	323253-002

Related Documents

Document Title	Location	Notes
Intel® 64 and IA-32 Architectures Software Developer's Manual *Volume 1: Basic Architecture* *Volume 2A: Instruction Set Reference, A-M* *Volume 2B: Instruction Set Reference, N-Z* *Volume 3A: System Programming Guide, Part 1* *Volume 3B: System Programming Guide, Part 2*	http://www.intel.com/content/www/us/en/processors/architectures-software-developer-manuals.html	1
Intel® 64 and IA-32 Architectures Software Developer's Manual Documentation Changes	http://www.intel.com/content/www/us/en/processors/architectures-software-developer-manuals.html	
Intel® 64 and IA-32 Architectures Optimization Reference Manual	http://www.intel.com/Assets/en_US/PDF/manual/248966.pdf	

Note:

1. Documentation changes for Intel® 64 and IA-32 Architecture Software Developer's Manual volumes 1, 2A, 2B, 3A, and 3B, and bug fixes are posted in the *Intel® 64 and IA-32 Architecture Software Developer's Manual Documentation Changes*.

Nomenclature

Errata are design defects or errors. These may cause the 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 such as, core speed, L2 cache size, package type, etc. as described in the processor identification information table. Read all notes associated with each S-Spec number.

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

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

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

Note: Errata remain in the specification update throughout the product's lifecycle, or until a particular stepping is no longer commercially available. Under these circumstances, errata removed from the specification update are archived and available upon request. Specification changes, specification clarifications and documentation changes are removed from the specification update when the appropriate changes are made to the appropriate product specification or user documentation (datasheets, manuals, and so on).

§ §

Summary Table of Changes

The following tables indicate the errata, specification changes, specification clarifications, or documentation changes which apply to the processor. Intel may fix some of the errata in a future stepping of the component, and account for the other outstanding issues through documentation or specification changes as noted. 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 a table row indicates this erratum is either new or modified from the previous version of the document.

Errata (Sheet 1 of 5)

Errata Number	Steppings	Status	ERRATA
	B-1		
BC1	X	No Fix	The Processor may Report a #TS Instead of a #GP Fault
BC2	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
BC3	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
BC4	X	No Fix	Performance Monitor SSE Retired Instructions May Return Incorrect Values
BC5	X	No Fix	Premature Execution of a Load Operation Prior to Exception Handler Invocation
BC6	X	No Fix	MOV To/From Debug Registers Causes Debug Exception
BC7	X	No Fix	Incorrect Address Computed For Last Byte of FXSAVE/FXRSTOR Image Leads to Partial Memory Update
BC8	X	No Fix	Values for LBR/BTS/BTM will be Incorrect after an Exit from SMM
BC9	X	No Fix	Single Step Interrupts with Floating Point Exception Pending May Be Mishandled
BC10	X	No Fix	Fault on ENTER Instruction May Result in Unexpected Values on Stack Frame
BC11	X	No Fix	IRET under Certain Conditions May Cause an Unexpected Alignment Check Exception
BC12	X	No Fix	General Protection Fault (#GP) for Instructions Greater than 15 Bytes May be Preempted
BC13	X	No Fix	General Protection (#GP) Fault May Not Be Signaled on Data Segment Limit Violation above 4-G Limit
BC14	X	No Fix	LBR, BTS, BTM May Report a Wrong Address when an Exception/Interrupt Occurs in 64-bit Mode
BC15	X	No Fix	MCi_Status Overflow Bit May Be Incorrectly Set on a Single Instance of a DTLB Error
BC16	X	No Fix	Debug Exception Flags DR6.B0-B3 Flags May be Incorrect for Disabled Breakpoints
BC17	X	No Fix	MONITOR or CLFLUSH on the Local XAPIC's Address Space Results in Hang
BC18	X	No Fix	Corruption of CS Segment Register During RSM While Transitioning From Real Mode to Protected Mode
BC19	X	No Fix	A VM Exit on MWAIT May Incorrectly Report the Monitoring Hardware as Armed
BC20	X	No Fix	Performance Monitor Event SEGMENT_REG_LOADS Counts Inaccurately
BC21	X	No Fix	#GP on Segment Selector Descriptor that Straddles Canonical Boundary May Not Provide Correct Exception Error Code
BC22	X	No Fix	Improper Parity Error Signaled in the IQ Following Reset When a Code Breakpoint is Set on a #GP Instruction
BC23	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
BC24	X	No Fix	IA32_MPERF Counter Stops Counting During On-Demand TM1
BC25	X	No Fix	The Memory Controller tTHROT_OPREF Timings May be Violated During Self Refresh Entry
BC26	X	No Fix	Synchronous Reset of IA32_APERF/IA32_MPERF Counters on Overflow Does Not Work

Errata (Sheet 2 of 5)

Errata Number	Steppings	Status	ERRATA
	B-1		
BC27	X	No Fix	Disabling Thermal Monitor While Processor is Hot, Then Re-enabling, May Result in Stuck Core Operating Ratio
BC28	X	No Fix	Writing the Local Vector Table (LVT) when an Interrupt is Pending May Cause an Unexpected Interrupt
BC29	X	No FIx	Faulting MMX Instruction May Incorrectly Update x87 FPU Tag Word
BC30	X	No Fix	xAPIC Timer May Decrement Too Quickly Following an Automatic Reload While in Periodic Mode
BC31	X	No Fix	Reported Memory Type May Not Be Used to Access the VMCS and Referenced Data Structures
BC32	X	No Fix	B0-B3 Bits in DR6 For Non-Enabled Breakpoints May be Incorrectly Set
BC33	X	No Fix	Core C6 May Clear Previously Logged TLB Errors
BC34	X	No Fix	Changing the Memory Type for an In-Use Page Translation May Lead to Memory-Ordering Violations
BC35	X	No Fix	A String Instruction that Re-maps a Page May Encounter an Unexpected Page Fault
BC36	X	No Fix	Infinite Stream of Interrupts May Occur if an ExtINT Delivery Mode Interrupt is Received while All Cores in C6
BC37	X	No Fix	Two xAPIC Timer Event Interrupts May Unexpectedly Occur
BC38	X	No Fix	EOI Transaction May Not be Sent if Software Enters Core C6 During an Interrupt Service Routine
BC39	X	No Fix	FREEZE_WHILE_SMM Does Not Prevent Event From Pending PEBS During SMM
BC40	X	No Fix	APIC Error “Received Illegal Vector” May be Lost
BC41	X	No Fix	DR6 May Contain Incorrect Information When the First Instruction After a MOV SS,r/m or POP SS is a Store
BC42	X	No Fix	An Uncorrectable Error Logged in IA32_CR_MC2_STATUS May also Result in a System Hang
BC43	X	No Fix	IA32_PERF_GLOBAL_CTRL MSR May be Incorrectly Initialized
BC44	X	No Fix	ECC Errors Can Not be Injected on Back-to-Back Writes
BC45	X	No Fix	Performance Monitor Counter MEM_INST_RETIRED.STORES May Count Higher than Expected
BC46	X	No Fix	Sleeping Cores May Not be Woken Up on Logical Cluster Mode Broadcast IPI Using Destination Field Instead of Shorthand
BC47	X	No Fix	Faulting Executions of FXRSTOR May Update State Inconsistently
BC48	X	No Fix	Memory Aliasing of Code Pages May Cause Unpredictable System Behavior
BC49	X	No Fix	Performance Monitor Counters May Count Incorrectly
BC50	X	No Fix	Simultaneous Accesses to the Processor via JTAG and PECI May Cause Unexpected Behavior
BC51	X	No Fix	Performance Monitor Event Offcore_response_0 (B7H) Does Not Count NT Stores to Local DRAM Correctly
BC52	X	No Fix	EFLAGS Discrepancy on Page Faults and on EPT-Induced VM Exits after a Translation Change
BC53	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

Errata (Sheet 3 of 5)

Errata Number	Steppings	Status	ERRATA
	B-1		
BC54	X	No Fix	Package C3/C6 Transitions When Memory 2x Refresh is Enabled May Result in a System Hang
BC55	X	No Fix	Back to Back Uncorrected Machine Check Errors May Overwrite IA32_MC3_STATUS.MSCOD
BC56	X	No Fix	Corrected Errors With a Yellow Error Indication May be Overwritten by Other Corrected Errors
BC57	X	No Fix	Performance Monitor Events DCACHE_CACHE_LD and DCACHE_CACHE_ST May Overcount
BC58	X	No Fix	Performance Monitor Events INSTR_RETIRED and MEM_INST_RETIRED May Count Inaccurately
BC59	X	No Fix	A Page Fault May Not be Generated When the PS bit is set to "1" in a PML4E or PDPTE
BC60	X	No Fix	Uncacheable Access to a Monitored Address Range May Prevent Future Triggering of the Monitor Hardware
BC61	X	No Fix	BIST Results May be Additionally Reported After a GETSEC[WAKEUP] or INIT-SIPI Sequence
BC62	X	No Fix	Pending x87 FPU Exceptions (#MF) May be Signaled Earlier Than Expected
BC63	X	No Fix	VM Exits Due to “NMI-Window Exiting” May Be Delayed by One Instruction
BC64	X	No Fix	Multiple Performance Monitor Interrupts are Possible on Overflow of IA32_FIXED_CTR2
BC65	X	No Fix	C-State Autodemotion May be too Aggressive Under Certain Configurations and Workloads
BC66	X	No Fix	LBRs May Not be Initialized During Power-On Reset of the Processor
BC67	X	No Fix	Multiple Performance Monitor Interrupts are Possible on Overflow of Fixed Counter 0
BC68	X	No Fix	VM Exits Due to LIDR/LGDT/SIDT/SGDT Do Not Report Correct Operand Size
BC69	X	No Fix	Performance Monitoring Events STORE_BLOCKS.NOT_STA and STORE_BLOCKS.STA May Not Count Events Correctly
BC70	X	No Fix	Storage of PEBS Record Delayed Following Execution of MOV SS or STI
BC71	X	No Fix	<Erratum Removed>
BC72	X	No Fix	The PECI Bus May be Tri-stated After System Reset
BC73	X	No Fix	LER MSRs May Be Unreliable
BC74	X	No Fix	APIC Timer CCR May Report 0 in Periodic Mode
BC75	X	No Fix	LBR, BTM or BTS Records May have Incorrect Branch From Information After an EIST Transition, T-states, C1E, or Adaptive Thermal Throttling
BC76	X	No Fix	PEBS Records Not Created For FP-Assists Events
BC77	X	No Fix	MSR_TURBO_RATIO_LIMIT MSR May Return Intel® Turbo Boost Technology Core Ratio Multipliers for Non-Existent Core Configurations
BC78	X	No Fix	L1 Cache Uncorrected Errors May be Recorded as Correctable in 16K Mode
BC79	X	No Fix	Extra APIC Timer Interrupt May Occur During a Write to the Divide Configuration Register
BC80	X	No Fix	PECI Reads of Machine Check MSRs in the Processor Core May Not Function Correctly

Errata (Sheet 4 of 5)

Errata Number	Steppings	Status	ERRATA
	B-1		
BC81	X	No Fix	The Combination of a Page-Split Lock Access And Data Accesses That Are Split Across Cacheline Boundaries May Lead to Processor Livelock
BC82	X	No Fix	Package C6 Transitions May Cause Memory Bit Errors to be Observed
BC83	X	No Fix	Sensitivity in Clocking Circuitry May Cause Unpredictable System Behavior
BC84	X	No Fix	Accesses to a VMCS May Not Operate Correctly If CR0.CD is Set on Any Logical Processsor of a Core
BC85	X	No Fix	Performance Monitor Events for Hardware Prefetches Which Miss TheL1 Data Cache May be Over Counted
BC86	X	No Fix	VM Exit May Incorrectly Clear IA32_PERF_GLOBAL_CTRL [34:32]
BC87	X	No Fix	Package C6 Transitions May Result in Single and Multi-Bit Memory Errors
BC88	X	No Fix	VM Entry May Omit Consistency Checks Related to Bit 14 (BS) of the Pending Debug Exception Field in Guest-State Area of the VMCS
BC89		No Fix	*BC89 Erratum Removed in Rev -008, November 2010 Edition*
BC90	X	No Fix	Execution of VMPTRLD May Corrupt Memory If Current-VMCS Pointer is Invalid
BC91	X	No Fix	PerfMon Overflow Status Can Not be Cleared After Certain Conditions Have Occurred
BC92	X	No Fix	L1 Data Cache Errors May be Logged With Level Set to 1 Instead of 0
BC93	X	No Fix	An Unexpected Page Fault or EPT Violation May Occur After Another Logical Processor Creates a Valid Translation for a Page
BC94	X	No Fix	PerfMon Event LOAD_HIT_PRE.SW_PREFETCH May Overcount
BC95	X	No Fix	Successive Fixed Counter Overflows May be Discarded
BC96	X	No Fix	#GP May be Signaled When Invalid VEX Prefix Precedes Conditional Branch Instructions
BC97	X	No Fix	A Logical Processor May Wake From Shutdown State When Branch-Trace Messages or Branch-Trace Stores Are Enabled
BC98	X	No Fix	Task Switch to a TSS With an Inaccessible LDTR Descriptor May Cause Unexpected Faults
BC99	X	No Fix	Unexpected Load May Occur on Execution of Certain Opcodes
BC100	X	No Fix	EOI-Broadcast Suppression May Not Function Properly if Enabled or Disabled While an Interrupt is in Service
BC101	X	No Fix	A First Level Data Cache Parity Error May Result in Unexpected Behavior
BC102	X	No Fix	An Event May Intervene Before a System Management Interrupt That Results from IN or INS
BC103	X	No Fix	Successive Fixed Counter Overflows May be Discarded
BC104	X	No Fix	VM Exits Due to “NMI-Window Exiting” May Not Occur Following a VM Entry to the Shutdown State
BC105	X	No Fix	Execution of INVVPID Outside 64-Bit Mode Cannot Invalidate Translations For 64-Bit Linear Addresses
BC106	X	No Fix	A Combination of Data Accesses That Are Split Across Cacheline Boundaries May Lead to a Processor Hang
BC107	X	No Fix	Package C6 C-State Exit May Result in Uncorrectable Memory Errors
BC108	X	No Fix	VMRESUME May Omit Check of Revision Identifier of Linked VMCS
BC109	X	No Fix	An INIT Signal May Cause Unpredictable Behavior After a VMXOFF
BC110	X	No Fix	APIC Timer Interrupts May be Lost During Core C3
BC111	X	No Fix	MCI_ADDR May be Incorrect For Cache Parity Errors

Errata (Sheet 5 of 5)

Errata Number	Steppings	Status	ERRATA
	B-1		
BC112	X	No Fix	The Corrected Error Count Overflow Bit in IA32_ MC0_STATUS is Not Updated When the UC Bit is Set
BC113	X	No Fix	The Upper 32 Bits of CR3 May be Incorrectly Used With 32-Bit Paging
BC114	X	No Fix	EPT Violations May Report Bits 11:0 of Guest Linear Address Incorrectly
BC115	X	No Fix	IA32_VMX_VMCS_ENUM MSR (48AH) Does Not Properly Report The Highest Index Value Used For VMCS Encoding
BC116	X	No Fix	Virtual-APIC Page Accesses With 32-Bit PAE Paging May Cause a System Crash
BC117	X	No Fix	An IRET Instruction That Results in a Task Switch Does Not Serialize the Processor

Specification Changes

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

Specification Clarifications

Number	SPECIFICATION CLARIFICATIONS
—	None for this revision of this specification update.

Documentation Changes

Number	DOCUMENTATION CHANGES
BC1	On-Demand Clock Modulation Feature Clarification

§ §

Identification Information

Component Identification

The Intel® Core™ i7-900 Desktop Processor Extreme Edition Series and Intel® Core™ i7-900 Desktop Processor Series on 32-nm Process stepping can be identified by the following register contents.

Table 1. Intel® Core™ i7-900 Desktop Processor Extreme Edition Series and Intel® Core™ i7-900 Desktop Processor Series on 32-nm Process Signature/Version

Reserved	Extended Family[1]	Extended Model[2]	Reserved	Processor Type[3]	Family Code[4]	Model Number[5]	Stepping ID[6]
31:28	27:20	19:16	15:14	13:12	11:8	7:4	3:0
	00000000b	0010b		00b	0110	1100b	XXXXb

Notes:

1. The Extended Family, bits [27:20] are used in conjunction with the Family Code, specific 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 family.
3. The Processor Type, specified in bits [13:12] indicates whether the processor is an original OEM processor, an OverDrive processor, or a dual processor (capable of being used in a dual processor system).
4. The Family Code corresponds to bits [11:8] of the EDX register after RESET, bits [11:8] of the EAX register after the CPUID instruction is executed with a 1 in the EAX register, and the generation field of the Device ID register accessible through Boundary Scan.
5. The Model Number corresponds to bits [7:4] of the EDX register after RESET, bits [7:4] of the EAX register after the CPUID instruction is executed with a 1 in the EAX register, and the model field of the Device ID register accessible through Boundary Scan.
6. The Stepping ID in bits [3:0] indicates the revision number of that model. See Table 2 for the processor stepping ID number in the CPUID information.

When EAX is initialized to a value of '1', the CPUID instruction returns the Extended Family, Extended Model, Processor Type, Family Code, Model Number, and Stepping ID 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

The Intel® Core™ i7-900 Desktop Processor Extreme Edition Series and Intel® Core™ i7-900 Desktop Processor Series on 32-nm Process can be identified by the following component markings.

Figure 1. Processor Top-side Markings (Example)

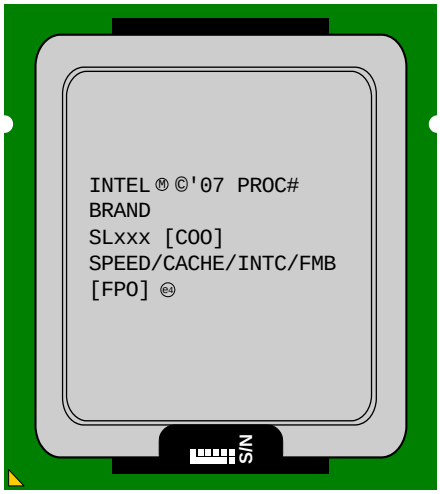

Table 2. Intel® Core™ i7-900 Desktop Processor Extreme Edition Series and Intel® Core™ i7-900 Desktop Processor Series on 32-nm Process Identification

S-Spec Number	Processor Number	Stepping	Processor Signature	Core Frequency (GHz) / Intel QuickPath Interconnect (GT/s) / DDR3 (MHz)	Max Intel® Turbo Boost Technology Frequency (GHz)[3]	Shared L3 Cache Size (MB)	Notes
SLBYU	i7980	B-1	0x000206C2	3.33/6.40/1066	6 core: 3.46 5 core: 3.46 4 core: 3.46 3 core: 3.46 2 core: 3.60 1 core: 3.60	12	1,2
SLBUZ	i7-980X	B-1	0x000206C2	3.33 / 6.40 / 1066	6 core: 3.46 5 core: 3.46 4 core: 3.46 3 core: 3.46 2 core: 3.60 1 core: 3.60	12	1,2
SLBVF	i7-970	B-1	0x000206C2	3.20/4.80/1066	6 core: 3.33 5 core: 3.33 4 core: 3.33 3 core: 3.33 2 core: 3.46 1 core: 3.46	12	1,2
SLBVZ	i7-990X	B-1	0x000206C2	3.46/6.40/1066	6 core: 3.60 5 core: 3.60 4 core: 3.60 3 core: 3.60 2 core: 3.73 1 core: 3.73	12	1,2

Notes:

1. CPUID is 0000206Csh, where 's' is the stepping number.
2. The core frequency reported in the processor brand string is rounded to 2 decimal digits. (For example, core frequency of 3.3333, is reported as @3.33 in brand string. Core frequency of 3.1666, repeating 6, is reported as @ 3.16 in brand string.)
3. This column indicated maximum Intel® Turbo Boost Technology frequency (GHz) for 6, 5, 4, 3, 2, and 1 cores active, respectively.

§ §

Errata

BC1. 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 Table of Changes.

BC2. 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 Table of Changes.

BC3. 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 Table of Changes.

BC4. 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 Table of Changes.

BC5. 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 Table of Changes.

BC6. 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 Table of Changes.

BC7. 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 Table of Changes.

BC8. 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 Table of Changes.

BC9. 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 Table of Changes.

BC10. 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 Table of Changes.

BC11. 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 Table of Changes.

BC12. 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 Table of Changes.

BC13. 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 Table of Changes.

BC14. 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 Table of Changes.

BC15. 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 Table of Changes.

BC16. 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 Table of Changes.

BC17. 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 Table of Changes.

BC18. 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 Table of Changes.

BC19. 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 Table of Changes.

BC20. 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 Table of Changes.

BC21. #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 Table of Changes.

BC22. 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 Table of Changes.

BC23. 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 Table of Changes.

BC24. 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 Table of Changes.

BC25. The 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 Table of Changes.

BC26. 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 Table of Changes.

BC27. 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 Table of Changes.

BC28. 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 Table of Changes.

BC29. 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 Table of Changes.

BC30. 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 Table of Changes.

BC31. 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 Table of Changes.

BC32. 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 Table of Changes.

BC33. 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 Table of Changes.

BC34. Changing the Memory Type for an In-Use Page Translation May Lead to Memory-Ordering Violations

Problem: Under complex micro-architectural 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 Table of Changes.

BC35. A String Instruction that Re-maps a Page May Encounter an Unexpected Page Fault

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

- The paging structures initially specify a valid translation for the page.
- Software modifies the paging structures so that there is no valid translation for the page (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.

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 Table of Changes.

BC36. 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 Table of Changes.

BC37. 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 Table of Changes.

BC38. EOI Transaction May Not be Sent if Software Enters Core C6 During an Interrupt Service Routine

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

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

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

BC39. 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

- A performance counter overflowed before an SMI
- A PEBS record has not yet been generated because another count of the event has not occurred
- 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 Table of Changes.

BC40. 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 Table of Changes.

BC41. 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 Table of Changes.

BC42. 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 Table of Changes.

BC43. 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 Table of Changes.

BC44. 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 Table of Changes.

BC45. Performance Monitor Counter MEM_INST_RETIRED.STORES May Count Higher than Expected

Problem: Performance Monitoring counter MEM_INST_RETIRED.STORES (Event: 0BH, Umask: 02H is used to track retired instructions which contain a store operation. Due to this erratum, the processor may also count other types of instructions including WRMSR and MFENCE.

Implication: Performance Monitoring counter MEM_INST_RETIRED.STORES may report counts higher than expected.

Workaround: None identified.

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

BC46. 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 Table of Changes.

BC47. Faulting Executions of FXRSTOR May Update State Inconsistently

Problem: The state updated by a faulting FXRSTOR instruction may vary from one execution to another.November 2014

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 Table of Changes.

BC48. 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 Table of Changes.

BC49. Performance Monitor Counters May Count Incorrectly

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

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

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

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

BC50. Simultaneous Accesses to the Processor via JTAG and PECI May Cause Unexpected Behavior

Problem: JTAG commands that are executed at the same time as a PECI (Platform Environment Control Interface) access may cause unexpected behavior. In addition the PECI command may take longer to complete or may not complete.

Implication: The processor could be left in an unexpected state and any software or firmware doing PECI writes may time out.

Workaround: Ensure that PECI commands are not executed while using JTAG.

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

BC51. 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 Table of Changes.

BC52. 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 Table of Changes.

BC53. 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: 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 Table of Changes.

BC54. Package C3/C6 Transitions When Memory 2x Refresh is Enabled May Result in a System Hang

Problem: If ASR_PRESENT (MC_CHANNEL_{0,1,2}_REFRESH_THROTTLE_SUP PORT CSR function 0, offset 68H, bit [0], Auto Self Refresh Present) is clear which indicates that high temperature operation is not supported on the DRAM, the memory controller will not enter self-refresh if software has REF_2X_NOW (bit 4 of the MC_CLOSED_LOOP CSR, function 3, offset 84H) set. This scenario may cause the system to hang during C3/C6 entry.

Implication: Failure to enter self-refresh can delay C3/C6 power state transitions to the point that a system hang may result with CATERR being asserted. REF_2X_NOW is used to double the refresh rate when the DRAM is operating in extended temperature range. The ASR_PRESENT was intended to allow low power self refresh with DRAM that does not support automatic self refresh.

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 Table of Changes.

BC55. 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 Table of Changes.

BC56. 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 Table of Changes.

BC57. 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 Table of Changes.

BC58. 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 Table of Changes.

BC59. 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 Table of Changes.

BC60. 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 CR0.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: The 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 Table of Changes.

BC61. 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[WAKEIUP] 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 Table of Changes.

BC62. 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 be observed #MF being-signaled before pending interrupts are serviced.

Workaround: None identified.

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

BC63. 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 not blocking of events by STR. 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 virutalization 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 Table of Changes.

BC64. 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 Table of Changes.

BC65. C-State Autodemotion May be too Aggressive Under Certain Configurations and Workloads

Problem: The C-state autodemotion feature allows the processor to make intelligent power and performance tradeoffs regarding the OS-requested C-state. Under certain operating system and workload specific conditions, the C-state auto-demotion feature may be overly aggressive in demoting OS C-sate requests to a C-sate with higher power and lower exit latency.

Implication: This aggressive demotion can result in higher platform power under idle conditions.

Workaround: None identified

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

BC66. 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 Table of Changes.

BC67. Multiple Performance Monitor Interrupts are Possible on Overflow of Fixed Counter 0

Problem: The processor can be configured to issue a PMI (performance monitor interrupt) upon overflow of the IA32_FIXED_CTR0 MSR (309H). A single PMI should be observed on overflow of IA32_FIXED_CTR0, however multiple PMIs are observed when this erratum occurs. This erratum only occurs when IA32_FIXED_CTR0 overflows and the processor and counter are configured as follows:

- Intel® Hyper-Threading Technology is enabled
- IA32_FIXED_CTR0 local and global controls are enabled
- IA32_FIXED_CTR0 is set to count events only on its own thread (IA32_FIXED_CTR_CTRL MSR (38DH) bits[2] = '0)
- PMIs are enabled on IA32_FIXED_CTR0 (IA32_FIXED_CTR_CTRL MSR bit[3] = '1)
- Freeze_on_PMI feature is enabled (IA32_DEBUGCTL MSR (1D9H) bit[12] = '1)

Implication: When this erratum occurs there may be multiple PMIs observed when IA32_FIXED_CTR0 overflows.

Workaround: Disable the FREEZE_PERFMON_ON_PMI feature in IA32_DEBUGCTL MSR (1D9H) bit[12].

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

BC68. VM Exits Due to LIDR/LGDT/SIDT/SGDT Do Not Report Correct Operand Size

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

Implication: Virtual-machine monitors cannot rely on bit 11 of the VM-exit instruction information field to determine the operand size of the instruction causing the VM exit.

Workaround: Virtual-machine monitor software may decode the instruction to determine operand size.

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

BC69. 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 Table of Changes.

BC70. 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 Table of Changes.

BC71. <Erratum Removed>

BC72. 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.

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

BC73. LER MSRs May Be Unreliable

Problem: Due to certain internal processor events, updates to the LER (Last Exception Record) MSRs, MSR_LER_FROM_LIP (1DDH) and MSR_LER_TO_LIP (1DEH), may happen when no update was expected.

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

Workaround: None identified.

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

BC74. APIC Timer CCR May Report 0 in Periodic Mode

Problem: In periodic mode the APIC timer CCR (current-count register) is supposed to be automatically reloaded from the initial-count register when the count reaches 0, consequently software would never be able to observe a value of 0. Due to this erratum, software may read 0 from the CCR when the timer has counted down and is in the process of re-arming.

Implication: Due to this erratum, an unexpected value of 0 may be read from the APIC timer CCR when in periodic mode.

Workaround: None identified.

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

BC75. LBR, BTM or BTS Records May have Incorrect Branch From Information After an EIST Transition, T-states, C1E, or Adaptive Thermal Throttling

Problem: The “Form” 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 (Enchanced 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 Table of Changes.

BC76. PEBS Records Not Created For FP-Assists Events

Problem: When a performance monitor counter is configured to count FP_ASSISTS (Event: F7H) and to trigger PEBS (Precise Event Based Sampling), the processor does not create a PEBS record when the counter overflows.

Implication: FP_ASSISTS events cannot be used for PEBS.

Workaround: None identified.

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

BC77. MSR_TURBO_RATIO_LIMIT MSR May Return Intel® Turbo Boost Technology Core Ratio Multipliers for Non-Existent Core Configurations

Problem: MSR_TURBO_RATIO_LIMIT MSR (1ADH) is designed to describe the maximum Intel Turbo Boost Technology potential of the processor. On some processors, a non-zero Intel Turbo Boost Technology value will be returned for non-existent core configurations.

Implication: Due to this erratum, software using the MSR_TURBO_RATIO_LIMIT MSR to report Intel Turbo Boost Technology processor capabilities may report erroneous results.

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

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

BC78. L1 Cache Uncorrected Errors May be Recorded as Correctable in 16K Mode

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

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

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

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

BC79. Extra APIC Timer Interrupt May Occur During a Write to the Divide Configuration Register

Problem: If the APIC timer Divide Configuration Register (Offset 03E0H) is written at the same time that the APIC timer Current Count Register (Offset 0390H) reads 1H, it is possible that the APIC timer will deliver two interrupts.

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

Workaround: Software should reprogram the Divide Configuration Register only when the APIC timer interrupt is disarmed.

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

BC80. PECI Reads of Machine Check MSRs in the Processor Core May Not Function Correctly

Problem: PECI reads which target machine check MSRs in the processor core may either be directed to a different core than intended or report that the data is not available.

Implication: PECI reads of machine check MSRs in the processor core may return incorrect data or incorrectly report that data is not available for the requested core.

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

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

BC81. The Combination of a Page-Split Lock Access And Data Accesses That Are Split Across Cacheline Boundaries May Lead to Processor Livelock

Problem: Under certain complex micro-architectural conditions, the simultaneous occurrence of a page-split lock and several data accesses that are split cacheline boundaries 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 Table of Changes.

BC82. Package C6 Transitions May Cause Memory Bit Errors to be Observed

Problem: During Package C6 transitions, internal signaling noise may cause the DDRx_CKE signal to become asserted during self-refresh. These assertions may result in memory bit errors upon exiting from the package C6 state. Due to this erratum the DDRx_CKE signals can be driven during times in which the DDR3 JEDEC specification requires that they are idle.

Implication: DDRx_CKE signals can be driven during package C6 memory self-refresh creating an invalid memory DRAM state. A system hang, memory ECC errors or unpredictable system behavior may occur when exiting the package C6 state.

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

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

BC83. Sensitivity in Clocking Circuitry May Cause Unpredictable System Behavior

Problem: On a subset of processors the clocking circuitry may be sensitive to fluctuations in Vtt voltage during stressful testing and/or operating conditions and may cause unpredictable system behavior.

Implication: This erratum may result in 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 Table of Changes.

BC84. Accesses to a VMCS May Not Operate Correctly If CR0.CD is Set on Any Logical Processsor of a Core

Problem: The VMX (virtual-machine extensions) are controlled by the VMCS (virtual-machinecontrol structure). If CR0.CD is set on any logical processor of a core, operations using the VMCS may not function correctly. Such operations include the VMREAD and VMWRITE instructions as well as VM entries and VM exits.

Implication: If CR0.CD is set on either logical processor in a core, the VMWRITE instruction may not correctly update the VMCS and the VMREAD instruction may not return correct data. VM entries may not load state properly and may not establish VMX controls properly. VM exits may not save or load state properly.

Workaround: VMMs (Virtual-machine monitors) should ensure that CR0.CD is clear on all logical processors of a core before entering VMX operation on any logical processor. Software should not set CR0.CD on a logical processor if any logical processor of the same core is in VMX operation. VMM software should prevent guest software from setting CR0.CD by setting bit 30 in the CR0 guest/host mask field in every VMCS.

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

BC85. Performance Monitor Events for Hardware Prefetches Which Miss TheL1 Data Cache May be Over Counted

Problem: Hardware prefetches that miss the L1 data cache but cannot be processed immediately due to resource conflicts will count and then retry. This may lead to incorrectly incrementing the L1D_PREFETCH.MISS (event 4EH, umask 02H) event multiple times for a single miss.

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

Workaround: None identified.

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

BC86. VM Exit May Incorrectly Clear IA32_PERF_GLOBAL_CTRL [34:32]

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

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

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

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

BC87. Package C6 Transitions May Result in Single and Multi-Bit Memory Errors

Problem: On a subset of processors, during package C6 transitions, internal circuit marginality may cause DDR3 JEDEC specification violations. These violations may result in control and data signal errors upon exiting from package C6 state.

Implication: Certain memory control signals may be incorrectly driven during package C6 memory self-refresh. This can create an invalid memory DRAM state, system hang, reboot, memory ECC errors or unpredictable system behavior. For systems with ECC memory, correctable/uncorrectable ECC errors may be logged in the IA32_MC8_STATUS MSR (421H) with the uncorrectable errors resulting in a machine check exception.

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

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

BC88. VM Entry May Omit Consistency Checks Related to Bit 14 (BS) of the Pending Debug Exception Field in Guest-State Area of the VMCS

Problem: Section "Checks on Guest Non-Register State" of Volume 3B specifies consistency checks that VM entry should perform for bit 14 (BS, indicating a pending single-step exception) of the pending debug exception field in guest-state area of the VMCS. These checks enforce the consistency of that bit with other fields in the guest-state area. Due to this erratum, VM entry may fail to perform these checks.

Implication: A logical processor may enter VMX non-root operation with a pending single-step debug exception that not consistent other register state; this may result in unexpected behavior. Intel has not observed this erratum with any commercially available software

Workaround: When using VMWRITE to write to a field in the guest-state area, software should ensure that the value written is consistent with the state of other guest-state fields.

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

BC89. BC89 Erratum Removed in Rev -008, November 2010 Edition

BC90. Execution of VMPTRLD May Corrupt Memory If Current-VMCS Pointer is Invalid

Problem: If the VMCLEAR instruction is executed with a pointer to the current-VMCS (virtual-machine control structure), the current-VMCS pointer becomes invalid as expected. A subsequent execution of the VMPTRLD (Load Pointer to Virtual-Machine Control Structure) instruction may erroneously overwrite the four bytes at physical address 0000008FH.

Implication: Due to this erratum, the four bytes in system memory at physical address 0000008FH may be corrupted.

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

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

BC91. PerfMon Overflow Status Can Not be Cleared After Certain Conditions Have Occurred

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

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

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

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

BC92. L1 Data Cache Errors May be Logged With Level Set to 1 Instead of 0

Problem: When an L1 Data Cache error is logged in IA32_MCi_STATUS[15:0], which is the MCA Error Code Field, with a cache error type of the format 0000 0001 RRRR TTLL, the LL field may be incorrectly encoded as 01b instead of 00b.

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

Workaround: None identified.

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

BC93. An Unexpected Page Fault or EPT Violation May Occur After Another Logical Processor Creates a Valid Translation for a Page

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

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

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

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

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

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

BC94. PerfMon Event LOAD_HIT_PRE.SW_PREFETCH May Overcount

Problem: PerfMon event LOAD_HIT_PRE.SW_PREFETCH (event 4CH, umask 01H) should count load instructions hitting an ongoing software cache fill request initiated by a preceding software prefetch instruction. Due to this erratum, this event may also count when there is a preceding ongoing cache fill request initiated by a locking instruction.

Implication: PerfMon event LOAD_HIT_PRE.SW_PREFETCH may overcount.

Workaround: None identified.

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

BC95. Successive Fixed Counter Overflows May be Discarded

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

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

Workaround: Software can avoid this by:

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

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

BC96. #GP May be Signaled When Invalid VEX Prefix Precedes Conditional Branch Instructions

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

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

Workaround: None identified.

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

BC97. A Logical Processor May Wake From Shutdown State When Branch-Trace Messages or Branch-Trace Stores Are Enabled

Problem: Normally, a logical processor that entered the shutdown state will remain in that state until a break event (NMI, SMI, INIT) occurs. Due to this erratum, if CR4.MCE (Machine Check Enable) is 0 and a branch-trace message or branch-trace store is pending at the time of a machine check, the processor may not remain in shutdown state. In addition, if the processor was in VMX non-root operation when it improperly woke from shutdown state, a subsequent VM exit may save a value of 2 into the activity-state field in the VMCS (indicating shutdown) even though the VM exit did not occur while in shutdown state.

Implication: This erratum may result in unexpected system behavior. If a VM exit saved a value of 2 into the activity-state field in the VMCS, the next VM entry will take the processor to shutdown state.

Workaround: Software should ensure that CR4.MCE is set whenever IA32_DEBUGCTL MSR (60EH) TR bit [6] is set.

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

BC98. Task Switch to a TSS With an Inaccessible LDTR Descriptor May Cause Unexpected Faults

Problem: A task switch may load the LDTR (Local Descriptor Table Register) with an incorrect segment descriptor if the LDT (Local Descriptor Table) segment selector in the new TSS specifies an inaccessible location in the GDT (Global Descriptor Table).

Implication: Future accesses to the LDT may result in unpredictable system behavior.

Workaround: Operating system code should ensure that segment selectors used during task switches to the GDT specify offsets within the limit of the GDT and that the GDT is fully paged into memory.

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

BC99. Unexpected Load May Occur on Execution of Certain Opcodes

Problem: If software executes an instruction with an opcode of the form 66 0F 38 8x (where x is in the range 0 to 6), the processor may unexpectedly perform a load operation (the data loaded is not used). The load occurs even if the instruction causes a VM exit or a fault (including an invalid-opcode exception). If the VMXON instruction has been executed successfully, the load is from the physical address in the VMXON pointer plus 408H; otherwise, it is from physical address 407H. The affected opcodes include the INVEPT and INVVPID instructions as well as five invalid opcodes.

Implication: This erratum may cause incorrect side effects if the load accesses a memory-mapped I/O device. 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 Table of Changes.

BC100. EOI-Broadcast Suppression May Not Function Properly if Enabled or Disabled While an Interrupt is in Service

Problem: If a processor supports EOI-broadcast suppression, a write to the local APIC's EOI register does not generate a broadcast EOI (even if the interrupt is level-triggered) if bit 12 of the local APIC's SVR (Spurious-Interrupt Vector Register) is set at the time of the write. Due to this erratum, the local APIC decides whether to generate a broadcast EOI based on the value that bit 12 of the SVR had at the time at which the most recent interrupt was delivered or the time of the most recent write to the EOI register (whichever is later).

Implication: If software modifies bit 12 of SVR while servicing an interrupt, the next write to the EOI register may not use the new bit value.

Workaround: Software should not modify bit 12 of SVR while servicing a level-triggered interrupt.

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

BC101. A First Level Data Cache Parity Error May Result in Unexpected Behavior

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

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

Workaround: None identified.

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

BC102. An Event May Intervene Before a System Management Interrupt That Results from IN or INS

Problem: If an I/O instruction (IN, INS, OUT, or OUTS) results in an SMI (system-management interrupt), the processor will set the IO_SMI bit at offset 7FA4H in SMRAM. This interrupt should be delivered immediately after execution of the I/O instruction so that the software handling the SMI can cause the I/O instruction to be re-executed. Due to this erratum, it is possible for another event (e.g., a nonmaskable interrupt) to be delivered before the SMI that follows the execution of an IN or INS instruction.

Implication: If software handling an affected SMI uses I/O instruction restart, the handler for the intervening event will not be executed.

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

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

BC103. Successive Fixed Counter Overflows May be Discarded

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

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

Workaround: Software can avoid this by:

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

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

BC104. VM Exits Due to "NMI-Window Exiting" May Not Occur Following a VM Entry to the Shutdown State

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

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

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

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

BC105. Execution of INVVPID Outside 64-Bit Mode Cannot Invalidate Translations For 64-Bit Linear Addresses

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

Implication: The INVVPID instruction may fail to invalidate translations for linear addresses that set bits in the range 63:32. Because this erratum applies only to executions outside 64-bit mode, it applies only to attempts by a 32-bit virtual-machine monitor (VMM) to invalidate translations for a 64-bit guest. Intel has not observed this erratum with any commercially available software.

Workaround: None identified.

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

BC106. A Combination of Data Accesses That Are Split Across Cacheline Boundaries May Lead to a Processor Hang

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

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

Workaround: None identified.

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

BC107. Package C6 C-State Exit May Result in Uncorrectable Memory Errors

Problem: On a subset of processors, during package C6 C-State transitions, internal circuit marginality may cause uncorrectable memory errors to occur on exiting Package C6 C-State. The errors will be logged in IA32_MCi_STATUS MSR with MCACOD = 0x009F.

Implication: Uncorrectable memory errors may cause a system reboot/shutdown.

Workaround: A BIOS workaround has been identified. Please refer to the latest version of BIOS Memory Reference Code and release notes.

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

BC108. VMRESUME May Omit Check of Revision Identifier of Linked VMCS

Problem: If the VMCS link pointer is valid in the VMCS, VM entry instructions should check that the 32 bits referenced by that pointer contains the processor's VMCS revision identifier and fail if it does not. Due to this erratum, VMRESUME may omit this check and thus not cause VM entry to fail in some cases.

Implication: The revision identifier of the linked VMCS may not be checked. Intel has not observed this erratum with any commercially available software.

Workaround: None identified.

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

BC109. An INIT Signal May Cause Unpredictable Behavior After a VMXOFF

Problem: If software has used VMX-preemption timer and subsequently leaves VMX operation using VMXOFF, then an INIT signal may result in unpredictable behavior.

Implication: This erratum may lead to unpredictable behavior, including a system hang. 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 Table of Changes.

BC110. APIC Timer Interrupts May be Lost During Core C3

Problem: APIC timer interrupts intended to awaken from core C3 may be lost under certain timing conditions.

Implication: Due to this erratum, a lost timer interrupt may cause the system to hang.

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

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

BC111. MCI_ADDR May be Incorrect For Cache Parity Errors

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

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

Workaround: None identified.

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

BC112. The Corrected Error Count Overflow Bit in IA32_ MC0_STATUS is Not Updated When the UC Bit is Set

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

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

Workaround: None identified.

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

BC113. The Upper 32 Bits of CR3 May be Incorrectly Used With 32-Bit Paging

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

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

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

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

BC114. EPT Violations May Report Bits 11:0 of Guest Linear Address Incorrectly

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

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

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

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

BC115. IA32_VMX_VMCS_ENUM MSR (48AH) Does Not Properly Report The Highest Index Value Used For VMCS Encoding

Problem: IA32_VMX_VMCS_ENUM MSR (48AH) bits 9:1 report the highest index value used for any VMCS encoding. Due to this erratum, the value 21 is returned in bits 9:1 although there is a VMCS field whose encoding uses the index value 23.

Implication: Software that uses the value reported in IA32_VMX_VMCS_ENUM[9:1] to read and write all VMCS fields may omit one field.

Workaround: None identified.

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

BC116. Virtual-APIC Page Accesses With 32-Bit PAE Paging May Cause a System Crash

Problem: If a logical processor has EPT (Extended Page Tables) enabled, is using 32-bit PAE paging, and accesses the virtual-APIC page then a complex sequence of internal processor micro-architectural events may cause an incorrect address translation or machine check on either logical processor.

Implication: This erratum may result in unexpected faults, an uncorrectable TLB error logged in IA32_MCi_STATUS.MCACOD (bits [15:0]) with a value of 0000_0000_0001_xxxxb

(where x stands for 0 or 1), a guest or hypervisor crash, or other unpredictable system behavior.

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

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

BC117. An IRET Instruction That Results in a Task Switch Does Not Serialize the Processor

Problem: An IRET instruction that results in a task switch by returning from a nested task does not serialize the processor (contrary to the Software Developer's Manual Vol. 3 section titled "Serializing Instructions").

Implication: Software which depends on the serialization property of IRET during task switching may not behave as expected. Intel has not observed this erratum to impact the operation of any commercially available software.

Workaround: None identified. Software can execute an MFENCE instruction immediately prior to the IRET instruction if serialization is needed.

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

§ §

Specification Changes

There are no specification changes in this revision of the specification update.

§ §

Specification Clarifications

There are no specification clarifications in this revision of the specification update.

§ §

Documentation Changes

BC1. On-Demand Clock Modulation Feature Clarification

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

For Intel® Hyper-Threading Technology enabled processors, the IA32_CLOCK_MODULATION register is duplicated for each logical processor. In order for the On-demand clock modulation feature to work properly, the feature must be enabled on all the logical processors within a physical processor. If the programmed duty cycle is not identical for all the logical processors, the processor clock will modulate to the highest duty cycle programmed for processors if the CPUID DisplayFamily_DisplayModel signatures is listed in Table 14-2. For all other processors, if the programmed duty cycle is not identical for all logical processors in the same core, the processor will modulate at the lowest programmed duty cycle.

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

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

Table 14-2. CPUID Signatures for Legacy Processors That Resolve to Higher Performance Setting of Conflicting Duty Cycle Requests

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

§ §