Intel® Xeon™ Processor Specification Update

Release Date: November 2001

Order Number: 249678-008

The Intel® Xeon™ processor may contain design defects or errors known as errata that may cause the product to deviate from published specifications. Current characterized errata are documented in this Specification Update.

CONTENTS

REVISION HISTORY

Date of Revision	Version	Description
May 2001	-001	Initial Release.
June 2001	-002	Added errata P27-P28.
June 2001	-003	Updated erratum P25.
July 2001	-004	Added RCPPS, RCPSS, RSQRTPS and RSQRTSS instruction specification clarification.
August 2001	-005	Added errata P29-P32. Added Unused outputs specification clarifications.
September 2001	-006	Added errata P33-34. Production mark update to include 2D matrix
October 2001	-007	Added 2 GHz frequency, D0 stepping and FC-BGA package information Added DP Platform Population Matrix Updated Summary of Errata table with applicable “Fixed” & “No Fix” errata plans Updated Errata P32
November 2001	-008	Added Erratum P35 Updated Summary of Errata Tale with latest errata status Added Documentation Changes P1-P5

PREFACE

This document is an update to the specifications contained in the following related documents:

- *Intel® Xeon™ Processor at 1.40 GHz, 1.50 GHz, 1.70 and 2 GHz* datasheet (Order Number 249665)
- *IA-32 Intel® Architecture Software Developer's Manual, Volumes 1, 2 and 3* (Order Numbers 245470, 245471, and 245472, respectively)

It is intended for hardware system manufacturers and software developers of applications, operating systems, or tools. It contains S-Specs, Errata, Documentation Changes, Specification Clarifications and Specification Changes.

Nomenclature

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

Errata are design defects or errors. Errata may cause Intel® Xeon™ processor's 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.

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

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 the next release of the specifications.

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

Specification Update for the Intel® Xeon™ Processor

GENERAL INFORMAITON

INTEL® XEON™ PROCESSOR MARKINGS (31 mm OLGA or FC-BGA)

Figure 1. Top Side Processor Marking

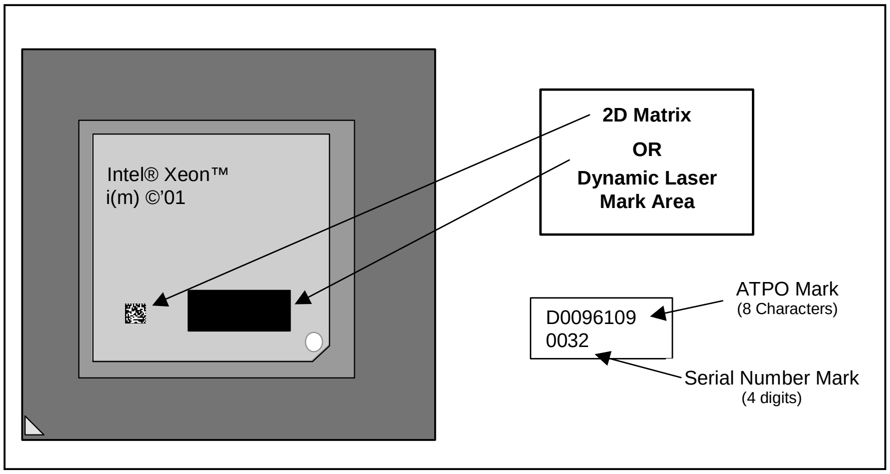
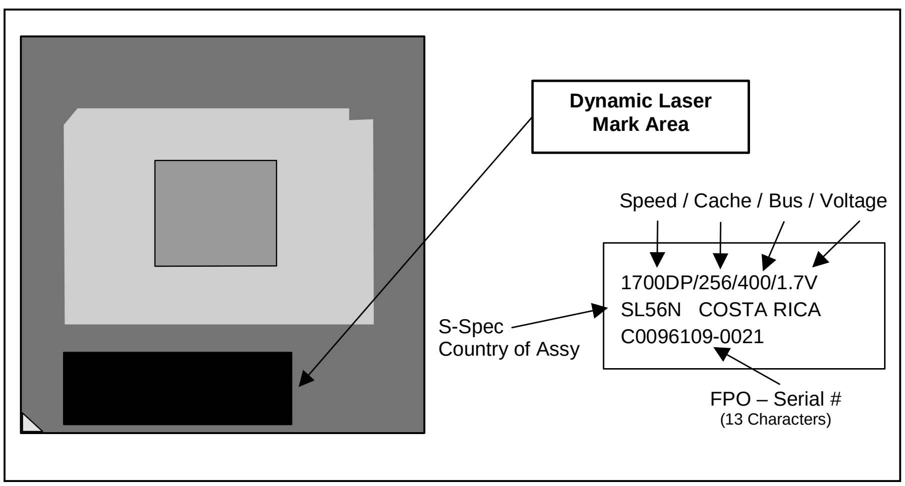


Figure 2. Bottom Side Processor Marking



IDENTIFICATION INFORMATION

The Intel® Xeon™ processor can be identified by the following values:

Family[1]	Model[2]	Brand ID[3]
1111	0000	00001110
1111	0001	00001110

NOTES:

1. The Family 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.
2. The Model corresponds to bits [7:4] of the EDX register after RESET, bits [7:4] of the EAX register after the CPUID 2. instruction is executed with a 1 in the EAX register, and the model field of the Device ID register accessible through Boundary Scan.
3. The Brand ID corresponds to bits [7:0] of the EBX register after the CPUID instruction is executed with a 1 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. Please refer to the Intel Processor Identification and the CPUID Instruction Application Note (AP-485) for further information on the CPUID instruction.

Intel® Xeon™ Processor Identification and Package Information

S-Spec Number	Core Stepping	CPUID	Speed Core/System Bus (GHz/MHz)	L2 Size (Kbytes)	Processor Interposer Revision	Package and Revision	Notes
SL4WX	C1	0F0Ah	1.40/400	256K	B0	31 mm OLGA rev 2.0	1
SL56G	C1	0F0Ah	1.40/400	256K	B0	31 mm OLGA rev 2.0	1, 2
SL4WY	C1	0F0Ah	1.50/400	256K	B0	31 mm OLGA rev 2.0	1
SL4ZT	C1	0F0Ah	1.50/400	256K	B0	31 mm OLGA rev 2.0	1, 2
SL56N	C1	0F0Ah	1.70/400	256K	B0	31 mm OLGA rev 2.0	1
SL56H	C1	0F0Ah	1.70/400	256K	B0	31 mm OLGA rev 2.0	1, 2
SL5TE	D0	0F12h	1.50/400	256K	C0	31 mm FC-BGA rev 3.0	1, 3
SL5U6	D0	0F12h	1.50/400	256K	C0	31 mm FC-BGA rev 3.0	1, 2, 3
SL5TD	D0	0F12h	1.70/400	256K	C0	31 mm BC-BGA rev 3.0	1, 3
SL5U7	D0	0F12h	1.70/400	256K	C0	31 mm BC-BGA rev 3.0	1, 2, 3
SL5TH	D0	0F12h	2/400	256K	C0	31 mm FC-BGA rev 3.0	1, 3
SL548	D0	0F12h	2/400	256K	C0	31 mm FC-BGA rev 3.0	1, 2, 3

NOTES:

1. These parts require the inputs from A20M#, IGNNE#, LINT[1]/NMI and LINT[0]/INTR pins during RESET to set the correct core to bus frequency ratio.
2. These parts are Intel boxed processors.
3. FC-BGA packaging maintains form, fit, and functionality when compared to OLGA packaging. Users may notice a color change.

MIXED STEPPINGS IN DP SYSTEMS

Intel Corporation fully supports mixed steppings of Intel® Xeon™ processors. The following list and processor matrix describes the requirements to support mixed steppings:

- Mixed steppings are only supported with processors that have identical family numbers as indicated by the CPUID instruction. The Intel® Xeon™ processor is available with two different Model numbers as indicated by the CPUID. Please refer to the "DP Platform Population Matrix for the Intel® Xeon™ Processor" for details regarding inclusion of processors with mixed CPUID/Core steppings.
- While Intel has done nothing to specifically prevent processors operating at differing frequencies from functioning within a multiprocessor system, there may be uncharacterized errata that exist in such configurations. Intel does not support such configurations. In mixed stepping systems, all processors must operate at identical frequencies (i.e., the highest frequency rating commonly supported by all processors).
- While there are no known issues associated with the mixing of processors with differing cache sizes in a multiprocessor system, and Intel has done nothing to specifically prevent such system configurations from operating, Intel does not support such configurations since there may be uncharacterized errata that exist. In mixed stepping systems, all processors must be of the same cache size.
- While Intel believes that certain customers may wish to perform validation of system configurations with mixed frequency or cache sizes, and that those efforts are an acceptable option to our customers, customers would be fully responsible for the validation of such configurations.
- Intel requires that the proper microcode update be loaded on each processor operating in a multiprocessor system. Any processor that does not have the proper microcode update loaded is considered by Intel to be operating out of specification.
- The workarounds identified in this and following specification updates must be properly applied to each processor in the system. Certain errata are specific to the multiprocessor environment and are identified in the *Mixed Stepping Processor Matrix* found at the end of this section. Errata for all processor steppings will affect system performance if not properly worked around. Also see the Intel® Xeon™ Processor Identification and Package Information section for additional details on which processors are affected by specific errata.
- In mixed stepping systems, the processor with the lowest feature-set, as determined by the CPUID Feature Bytes, must be the Bootstrap Processor (BSP). In the event of a tie in feature-set, the tie should be resolved by selecting the BSP as the processor with the lowest stepping as determined by the CPUID instruction.

In the following processor matrix, "NI" indicates that there are currently no known issues associated with mixing these steppings. A number indicates that a known issue has been identified as listed in the table following the matrix. A dual processor system using mixed processor steppings must assure that errata are addressed appropriately for each processor.

DP Platform Population Matrix for the Intel® Xeon™ Processor

CPUID/Core Stepping	0F0Ah/C1	0F12h/D0
0F0Ah/C1	NI	Note 1
0F12h/D0	Note 1	NI

NOTES:

1. Some of these processors are affected by errata, which may affect the features an MP system is able to support. See the *Intel® Xeon™ Processor Identification and Package Information* table for details on which processors are affected by these errata.

SUMMARY OF CHANGES

The following table indicates the Errata, Documentation Changes, Specification Clarifications, or Specification Changes that apply to Intel® Xeon™ processors. Intel intends to fix some of the errata in a future stepping of the component, and to account for the other outstanding issues through documentation or specification changes as noted. This table uses the following notations:

CODES USED IN SUMMARY TABLE

X:	Erratum, Documentation Change, Specification Clarification, or Specification Change applies to the given processor stepping.
(No mark) or (blank box):	This item is fixed in or does not apply to the given stepping.
Fix:	Intel intends to fix this erratum in a future stepping of the component.
Fixed:	This erratum has been previously fixed.
NoFix:	There are no plans to fix this erratum.
Doc:	Intel intends to update the appropriate documentation in a future revision.
PKG:	This column refers to errata on the Intel Xeon™ processor substrate.
AP:	APIC related erratum.
Shaded	This item is either new or modified from the previous version of the document.

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

A = Intel ® Pentium ® II processor

B = Mobile Intel ® Pentium ® II processor

C = Intel ® Celeron® processor

D = Intel ® Pentium ® II Xeon™ processor

E = Intel ® Pentium ® III processor

G = Intel ® Pentium ® III Xeon™ processor

H = Mobile Intel ® Celeron® processor at 466 MHz 433 MHz, 400 MHz, 366 MHz, 333 MHz, 300 MHz, and 266 MHz

K = Mobile Intel ® Pentium ® III processor

M = Mobile Intel ® Celeron® processor at 500 MHz, 450 MHz, and 400A MHz

N = Intel ® Pentium ® 4 processor

P = Intel ® Xeon™ processor

The Specification Updates for the Pentium ® processor, Pentium ® Pro processor, and other Intel products do not use this convention.

Summary of Errata

NO.	C1	D0	Plans	ERRATA
P1	X	X	NoFix	UC Code in same line as WriteBack (WB) data may lead to data corruption
P2	X	X	NoFix	Transaction is not retried after BINIT#
P3	X	X	NoFix	Invalid opcode 0FFFH requires a ModRM byte
P4	X	X	NoFix	When in No-Fill Mode (CR0.CD=1) the memory type of large (PSE-4M and PAE-2M) pages are wrongly forced to uncacheable
P5	X	X	NoFix	Processor may hang due to Speculative Page Walks to Non-Existent System Memory
P6	X	X	NoFix	Writing a performance counter may result in an incorrect counter value
P7	X	X	NoFix	Performance Counter May Contain Incorrect Value After Being Stopped
P8	X		Fixed	REP MOV instruction with overlapping source and destination may result in data corruption
P9	X	X	NoFix	Memory type of the load lock different from its corresponding store unlock
P10	X	X	NoFix	Machine check architecture error reporting and recovery may not work as expected
P11	X	X	NoFix	Debug mechanisms may not function as expected
P12	X		Fixed	Processor may live-lock if PDEs or PTEs are in UC space
P13	X		Fixed	Thermal status log bit may not be set when the Thermal Control Circuit is Active
P14	X	X	NoFix	Processor may timeout waiting for a device to respond after 0.67 seconds
P15	X	X	NoFix	Cascading of performance counters does not work correctly when forced overflow is enabled
P16	X	X	NoFix	EMON event counting of X87 loads may not work as expected
P17	X	X	NoFix	Simultaneous code breakpoint and uncorrectable error results in a processor hang
P18	X	X	NoFix	Software controlled clock modulation using a 12.5% or 25% duty cycle may cause the processor to hang
P19	X		Fixed	RFO with ECC error may result in incorrect data
P20	X		Fixed	Speculative page fault may cause livelock
P21	X		Fixed	PAT index MSB may be calculated incorrectly
P22	X	X	NoFix	System bus interrupt messages without data and which receive a HardFailure response may hang the processor
P23	X	X	NoFix	SQRTPD and SQRTSD may return QNaN indefinite instead of negative zero
P24	X	X	NoFix	Bus Invalidate Line Request that returns unexpected data may result in L1 cache corruption
P25	X		Fixed	Multi-processor boot protocol may not complete with an IOQ depth of one
P26	X	X	NoFix	Processor flags #PF instead of #AC on an unlocked CMPXC8B instruction
P27	X	X	NoFix	Incorrect data may be returned when page tables are located in Write Combining (WC) memory
P28	X	X	NoFix	FSW may not be completely restored after page fault on FRSTOR or FLDENV instructions

Summary of Errata

NO.	C1	D0	Plans	ERRATA
P29	X	X	NoFix	Write Combining (WC) load may result in an unintended address on system bus
P30	X	X	No Fix	Processor issues inconsistent transaction size attributes for locked operation
P31	X	X	NoFix	Multiple accesses to the same S-state L2 cache line and ECC error combination may result in loss of cache coherency
P32	X	X	No Fix	IA32_MC0_ADDR and IA32_MC0_MISC registers will contain invalid or stale data following a data, address or response parity error
P33	X	X	No Fix	When the processor is in the System Management Mode (SMM), Debug Registers may be fully writeable
P34	X	X	No Fix	Associated counting logic must be configured when using Event Selection Control (ESCR) MSR
P35	X	X	No Fix	Livelock may occur when bus parking is disabled

Summary of Documentation Changes

NO.	C1	D0	PKG	Plans	Documentation Changes
P1	X	X		Fix	Machine Check Exception detected when BINIT# drive enabled
P2	X	X		Fix	The encoding of “Immediate to Register” of “AND” instruction is missing
P3	X	X		Fix	The ‘reg’ field of CMPXCHG8B instruction encoding must be 001
P4	X	X		Fix	SCAS/SCASB/SCASW/SCASD encoding operand is incorrect
P5	X	X		Fix	XCHG encoding operand (1-byte form) does not have a w-bit, hence the reg size is implied. The AL register is not a valid option for this 1-byte encoding.

Summary of Specification Clarifications

NO.	C1	D0	PKG	Plans	Specification Clarifications
					There are no specification clarifications

Summary of Specification Changes

NO.	C1	PKG	Plans	Specification Changes
				There are no specification changes

ERRATA

P1. *UC Code in Same Line as WriteBack (WB) Data may Lead to Data Corruption*

Problem: This erratum occurs when both code (being accessed as UC or WC) and data (being accessed as WB) are placed in the same cache line. The UC fetch will cause the processor to self-snoop and generate an implicit writeback. The data supplied by this implicit writeback may be corrupted due to the way the processor is currently handling self-modifying code.

Implication: UC code located in the same cache line as WB data may lead to data corruption.

Workaround: UC or WC code should not be located in the same 64 byte cache line as any location that is being stored to with WB data.

Status: For the steppings affected see the Summary of Changes at the beginning of this section.

P2. *Transaction Is Not Retried After BINIT#*

Problem: If the first transaction of a locked sequence receives a HITM# and DEFER# during the snoop phase it should be retried and the locked sequence restarted. However, if BINIT# is also asserted during this transaction, the transaction will not be retried.

Implication: When this erratum occurs, locked transactions will not be retried.

Workaround: None identified

Status: For the steppings affected see the Summary of Changes at the beginning of this section.

P3. *Invalid Opcode 0FFFh Requires A ModRM Byte*

Problem: Some invalid opcodes require a ModRM byte and other following bytes, while others do not. The invalid opcode 0FFFh did not require a ModRM in previous generation microprocessors such as Pentium® II or Pentium III processors, but it is required in the Intel® Xeon™ processor

Implication: The use of an invalid opcode 0FFFh without the ModRM byte may result in a page or limit fault on the Intel Xeon processor.

Workaround: To avoid this erratum use ModRM byte with invalid 0FFFh opcode.

Status: For the steppings affected see the Summary of Changes at the beginning of this section.

P4. *When in No-Fill Mode (CR0.CD=1) the Memory Type of Large (PSE-4M and PAE-2M) Pages are Wrongly Forced to Uncacheable*

Problem: When the processor is operating in No-Fill Mode (CR0.CD=1), the page miss hardware incorrectly forces the memory type of large (PSE-4M and PAE-2M) pages to UC memory type regardless of the MTRR settings. By forcing the memory type of these pages to UC, load operations, which should hit valid data in the L1 cache, are forced to load the data from system memory. Some applications will lose the performance advantage associated with the caching permitted by other memory types.

Implication: This erratum may result in some performance degradation when using no-fill mode with large pages.

Workaround: None identified

Status: For the steppings affected see the Summary of Changes at the beginning of this section.

P5. *Processor May Hang Due to Speculative Page Walks to NonExistent System Memory*

Problem: A load operation issued speculatively by the processor that misses the Data Translation Lookaside Buffer (DTLB) results in a page walk. A branch instruction older than the load retires so that this load operation is now in the mispredicted branch path. Due to an internal boundary condition, in some instances the load is not canceled before the page walk is issued.

The Page Miss Handler (PMH) starts a speculative page-walk for the Load and issues a cacheable load of the Page Directory Entry (PDE). This PDE load returns data that points to a page table entry in uncacheable (UC) memory. The PMH issues the PTE Load to UC space, which is issued on the system bus. No response comes back for this load PTE operation since the address is pointing to system memory, which does not exist.

This load to non-existent system memory causes the processor to hang because other bus requests are queued up behind this UC PTE load, which never gets a response. If the load was accessing valid system memory, the speculative page-walk would successfully complete and the processor would continue to make forward progress.

Implication: Processor may hang due to speculative page walks to non-existent system memory

Workaround: Page directories and page tables in UC memory space must point to system memory that exists.

Status: For the steppings affected see the Summary of Changes at the beginning of this section.

P6. *Writing a Performance Counter May Result in an Incorrect Counter Value*

Problem: Accessing a performance counter also enables the counter input so that writing one half of the counter can cause the other half to increment. When a performance counter is written and the event counter for the event being monitored is non-zero, the performance counter will be incremented by the value on that event counter. Because the upper eight bits of the performance counter are not written at the same time as the lower 32 bits, the increment due to the non-zero event counter may cause a carry to the upper bits such that the performance counter contains a value higher than what was written. The worst case error caused by this can be about 4 billion counts.

Implication: When this erratum occurs, the performance counter will contain a different value from that which was written.

Workaround: If the performance counter is set to select a null event and the CCCR for that counter has its compare bit set to zero, before the performance counter is written, this erratum will not occur.

Status: For the steppings affected see the Summary of Changes at the beginning of this section.

P7. *Performance Counter May Contain Incorrect Value After Being Stopped*

Problem: If a performance counter is stopped on the precise internal clock cycle where the intermediate carry from the lower 32 bits of the counter to the upper eight bits occurs, the intermediate carry is lost.

Implication: When this erratum occurs the performance counter may contain a value about 4 billion (232) less than it should.

Workaround: Since this erratum does not occur if the performance counters are read when running, a possible workaround is to read the counter before stopping it.

Status: For the steppings affected see the Summary of Changes at the beginning of this section.

P8. *REP MOV Instruction with Overlapping Source and Destination may Result in Data Corruption*

Problem: When fast strings are enabled and a REP MOV instruction is used to move a string and the source and destination strings overlap by 56 bytes or less, data corruption may occur.

Implication: When this erratum occurs, data corruption may occur.

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

Status: For the steppings affected see the Summary of Changes at the beginning of this section.

P9. *Memory Type of the Load Lock Different from its Corresponding Store Unlock*

Problem: The Intel® Xeon™ processor employs a use-once protocol to ensure that a processor in a multiprocessor system may access data that is loaded into its cache on a Read-for-Ownership operation at least once before it is snooped out by another processor. This protocol is necessary to avoid a dual processor livelock scenario where no processor in the system can gain ownership of a line and modify it before that data is snooped out by another processor. In the case of this erratum, the use-once protocol incorrectly activates for split load lock instructions. A load lock operation accesses data that splits across a page boundary with both pages of WB memory type. The use-once protocol activates and the memory type for the split halves get forced to UC. Since use-once does not apply to stores, the store unlock instructions go out as WB memory type. The full sequence on the Bus is: locked partial read (UC), partial read (UC), partial write (WB), locked partial write (WB). The Use-once protocol should not be applied to Load locks.

Implication: When this erratum occurs, the memory type of the load lock will be different than the memory type of the store unlock operation. This behavior (Load Locks and Store Unlocks having different memory types) does not however introduce any functional failures such as system hangs or memory corruption.

Workaround: None identified

Status: For the steppings affected see the Summary of Changes at the beginning of this section.

P10. *Machine Check Architecture Error Reporting and Recovery May Not Work as Expected*

Problem: When the processor detects errors it should attempt to report and/or recover from the error. In the situations described below, the processor does not report and/or recover from the error(s) as intended.

- When a transaction is deferred during the snoop phase and subsequently receives a Hard Failure response, the transaction should be removed from the bus queue so that the processor may proceed. Instead, the transaction is not properly removed from the bus queue, the bus queue is blocked, and the processor will hang.
- When a hardware prefetch results in an uncorrectable tag error in the L2 cache, MC0_STATUS.UNCOR and MC0_STATUS.PCC are set but no Machine Check Exception (MCE) is signaled. No data loss or corruption occurs because the data being prefetched has not been used. If the data location with the uncorrectable tag error is subsequently accessed, an MCE will occur. However, upon this MCE, or any other subsequent MCE, the information for that error will not be logged because MC0_STATUS.UNCOR has already been set and the MCA status registers will not contain information about the error which caused the MCE assertion but instead will contain information about the prefetch error event.
- When the reporting of errors is disabled for Machine Check Architecture (MCA) Bank 2 by setting all MC2_CTL register bits to 0, uncorrectable errors should be logged in the IA32_MC2_STATUS register but no machine-check exception should be generated. Uncorrectable loads on bank 2, which would normally be logged in the IA32_MC2_STATUS register, are not logged.
- When one half of a 64 byte instruction fetch from the L2 cache has an uncorrectable error and the other 32 byte half of the same fetch from the L2 cache has a correctable error, the processor will attempt to correct the correctable error but cannot proceed due to the uncorrectable error. When this occurs the processor will hang.

- When an L1 cache parity error occurs, the cache controller logic should write the physical address of the data memory location that produced that error into the IA32_MC1_ADDR REGISTER (MC1_ADDR). In some instances of a parity error on a load operation that hits the L1 cache, however, the cache controller logic may write the physical address from a subsequent load or store operation into the IA32_MC1_ADDR register.
- The local xAPIC has an Error Status Register, which records all errors it detects. Bit 6 of this register, the Receive Illegal Vector bit, is set when the local xAPIC detects an illegal vector in a message that it received. When an illegal vector error is received on the same internal clock that the error status register is being written due to a previous error, bit 6 does not get set and illegal vector errors are not flagged.
- When an error exists in the tag field of a cache line such that a request for ownership (RFO) issued by the processor hits multiple tag fields in the L2 cache (the correct tag and the tag with the error) and the accessed data also has a correctable error, the processor will correctly log the multiple tag match error but will hang when attempting to execute the machine check exception handler.
- If a memory access receives a machine check error on both 64 byte halves of a 128-byte L2 cache sector, the IA32_MC0_STATUS register records this event as multiple errors, i.e., the valid error bit and the overflow error bit are both set indicating that a machine check error occurred while the results of a previous error were in the error-reporting bank. The IA32_MC1_STATUS register should also record this event as multiple errors but instead records this event as only one correctable error.
- The overflow bit should be set to indicate when more than one error has occurred. The overflow bit being set indicates that more than one error has occurred. Because of this erratum, if any further errors occur, the MCA overflow bit will not be updated, thereby incorrectly indicating only one error has been received.
- If an I/O instruction (IN, INS, REP INS, OUT, OUTS, or REP OUTS) is being executed, and if the data for this instruction becomes corrupted, the processor will signal a Machine Check Exception (MCE). If the instruction is directed at a device that is powered down, the processor may also receive an assertion of SMI#. Since MCEs have higher priority, the processor will call the MCE handler, and the SMI# assertion will remain pending. However, while attempting to execute the first instruction of the MCE handler, the SMI# will be recognized and the processor will attempt to execute the SMM handler. If the SMM handler is successfully completed, it will attempt to restart the I/O instruction, but will not have the correct machine state due to the call to the MCE handler. This can lead to failure of the restart and shutdown of the processor.
- If PWRGOOD is de-asserted during a RESET# assertion causing internal glitches, the MCA registers may latch invalid information.
- If RESET# is asserted, then de-asserted, and reasserted, before the processor has cleared the MCA registers, then the information in the MCA registers may not be reliable, regardless of the state or state transitions of PWRGOOD.
- If MCERR# is asserted by one processor and observed by another processor, the observing processor does not log the assertion of MCERR#. The Machine Check Exception (MCE) handler called upon assertion of MCERR# will not have any way to determine the cause of the MCE.
- The Overflow Error bit (bit 62) in the IA32_MC0_STATUS register indicates, when set, that a machine check error occurred while the results of a previous error were still in the error reporting bank (i.e. The Valid bit was set when the new error occurred). If an uncorrectable error is logged in the error-reporting bank and another error occurs, the overflow bit will not be set.
- The MCA Error Code field of the IA32_MC0_STATUS register gets written by a different mechanism than the rest of the register. For uncorrectable errors, the other fields in the IA32_MC0_STATUS register are only updated by the first error. Any further errors that are detected will update the MCA Error Code field without updating the rest of the register, thereby leaving the IA32_MC0_STATUS register with stale information.
- When a speculative load operation hits the L2 cache and receives a correctable error, the IA32_MC1_Status Register may be updated with incorrect information. The IA32_MC1_Status Register should not be updated for speculative loads.
- The processor should only log the address for L1 parity errors in the IA32_MC1_Status register if a valid address is available. If a valid address is not available, the Address Valid bit in the IA32_MC1_Status register should not be set. In instances where an L1

parity error occurs and the address is not available because the linear to physical address translation is not complete or an internal resource conflict has occurred, the Address Valid bit is incorrectly set.

- The processor may hang when an instruction code fetch receives a hard failure response from the system bus. This occurs because the bus control logic does not return data to the core, leaving the processor empty. IA32_MC0_STATUS MSR does indicate that a hard fail response occurred.
- The processor may hang when the following events occur and the machine check exception is enabled, CR4.MCE=1. A processor that has it's STPCLK# pin asserted will internally enter the Stop Grant State and finally issue a Stop Grant Acknowledge special cycle to the bus. If an uncorrectable error is generated during the Stop Grant process it is possible for the Stop Grant special cycle to be issued to the bus before the processor vectors to the machine check handler. Once the chipset receives its last Stop Grant special cycle it is allowed to ignore any bus activity from the processors. As a result, processor accesses to the machine check handler may not be acknowledged, resulting in a processor hang.

Implication: The processor is unable to correctly report and/or recover from certain errors.

Workaround: None identified

Status: For the steppings affected see the Summary of Changes at the beginning of this section.

P11. *Debug Mechanisms May Not Function as Expected*

Problem: Certain debug mechanisms may not function as expected on the processor. The cases are as follows:

- When the following conditions occur: 1) An FLD instruction signals a stack overflow or underflow; 2) the FLD instruction splits a page-boundary or a 64 byte cache line boundary; 3) the instruction matches a Debug Register on the high page or cache line respectively; 4) the FLD has a stack fault and a memory fault on a split access, the processor will only signal the stack fault and the debug exception will not be taken.
- When a data breakpoint is set on the ninth and/or tenth byte(s) of a floating-point store using the Extended Real data type, and an unmasked floating-point exception occurs on the store, the break point will not be captured.
- When any instruction has multiple debug register matches, and any one of those debug registers is enabled in DR7, all of the matches should be reported in DR6 when the processor goes to the debug handler. This is not true during a REP instruction. As an example, during execution of a REP MOVSW instruction the first iteration a load matches DR0 and DR2 and sets DR6 as FFFF0FF5h. On a subsequent iteration of the instruction, a load matches only DR0. The DR6 register is expected to still contain FFFF0FF5h, but the processor will update DR6 to FFFF0FF1h.
- When the memory type is set to UC (Uncacheable) and a physical address code breakpoint is set, the processor will not break at the physical address code breakpoint and the machine will continue execution beyond the breakpoint.

Implication: Certain debug mechanisms do not function as expected on the processor.

Workaround: None identified

Status: For the steppings affected see the Summary of Changes at the beginning of this section.

P12. *Processor May Live-lock if PDEs or PTEs are in UC Space*

Problem: The processor may livelock under the following boundary conditions:

- The Page-Directory Entries (PDEs) or Page-Table Entries (PTEs) are in uncacheable (UC) space
- An instruction fetch misses the ITLB resulting in a page walk
- This instruction fetch is immediately followed by a store that splits a page boundary

Implication: When this erratum occurs, the processor will livelock. This erratum was found using random instruction testing and has not been observed with commercial software.

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

Status: For the steppings affected see the Summary of Changes at the beginning of this section.

P13. *Thermal Status Log Bit May Not be Set when the Thermal Control Circuit is Active*

Problem: Bit 1 of the IA32_THERM_STATUS register (Thermal Status Log) is a sticky bit designed to be set to '1' if the thermal control circuit (TCC) has been active since either the previous processor reset or software cleared this bit. If TCC is active and the Thermal Status Log bit is cleared by a processor reset or by software, it will remain clear (set to ‘0’) as long as the TCC remains active. Once TCC deactivates, the next activation of the TCC will set the Thermal Status Log bit.

Implication: When this erratum occurs, the Thermal Status Log bit will be cleared (set to ‘0’) although the thermal control circuit is active.

Workaround: None identified

Status: For the steppings affected see the Summary of Changes at the beginning of this section.

P14. *Processor May Timeout Waiting For A Device To Respond After 0.67 Seconds*

Problem: The PCI 2.1 target initial latency specification allows two seconds for a device to respond during initialization-time. The processor may timeout after only approximately 0.67 seconds. When the processor times out it will hang with IERR# asserted. PCI devices that take longer than 0.67 seconds to initialize may not be initialized properly.

Implication: System may hang with IERR# asserted.

Workaround: None identified

Status: For the steppings affected see the Summary of Changes at the beginning of this section.

P15. *Cascading Of Performance Counters Does Not Work Correctly When Forced Overflow is Enabled*

Problem: The performance counters are organized into pairs. When the CASCADE bit of the Counter Configuration Control Register (CCCR) is set, a counter that overflows will continue to count in the other counter of the pair. The FORCE_OVF bit forces the counters to overflow on every non-zero increment. When the FORCE_OVF bit is set, the counter overflow bit will be set but the counter no longer cascades.

Implication: The performance counters do not cascade when the FORCE_OVF bit is set.

Workaround: None identified

Status: For the steppings affected see the Summary of Changes at the beginning of this section.

P16. *EMON Event Counting of x87 Loads May Not Work as Expected*

Problem: If a performance counter is set to count x87 loads and floating-point exceptions are unmasked, the FPU Operand (Data) Pointer (FDP) may become corrupted.

Implication: When this erratum occurs, FPU Operand (Data) Pointer (FDP) may become corrupted.

Workaround: This erratum will not occur with floating point exceptions masked. If floating-point exceptions are unmasked, then performance counting of x87 loads should be disabled.

Status: For the steppings affected see the Summary of Changes at the beginning of this section.

P17. *Simultaneous Code Breakpoint and Uncorrectable Error Results in Processor Hang*

Problem: If an instruction fetch results in an uncorrectable error and there is also a debug breakpoint at this address, the processor will hang and the uncorrectable error will not be logged in the Machine Check registers.

Implication: When this erratum occurs the processor will hang.

Workaround: None identified

Status: For the steppings affected see the Summary of Changes at the beginning of this section.

P18. *Software Controlled Clock Modulation Using A 12.5% or 25% Duty Cycle May Cause The Processor To Hang*

Problem: Per the ACPI 1.0b specification, processor clock modulation may be controlled via a processor register (IA32_THERM_CONTROL). The On-Demand Clock Modulation Duty Cycle is controlled by bits 3:1. If these bits are set to a duty cycle of 12.5% or 25%, the processor may hang while attempting to execute a floating-point instruction. In this failure, the last instruction pointer (LIP) is pointing to a floating-point instruction whose instruction bytes are in UC space and which takes an exception 16 (floating point error exception). The processor stalls trying to fetch the bytes of the faulting floating-point instruction and those following it. This processor hang is caused by interactions between the thermal control circuit and floating-point event handler.

Implication: When the clock modulation is set to 12.5% or 25% duty cycle, the processor will go into a sleep state from which it fails to return.

Workaround: Use a duty cycle other than 12.5% or 25%.

Status: For the steppings affected see the Summary of Changes at the beginning of this section.

P19. *RFO With ECC Error May Result In Incorrect Data*

Problem: This erratum occurs as the result of the following conditions:

- A read for ownership (RFO) generates a correctable error.
- In the process of correcting the error, a locked RFO (LRFO) is issued that uses the same internal buffer as the previous RFO.
- Another processor issues a snoop to the same address as the LRFO.

An internal boundary condition exists which may prevent the LRFO from completing correctly causing the snoop to receive incorrect data. Intel has not been able to reproduce this erratum with commercial software.

Implication: When this erratum occurs, data corruption may result.

Workaround: None identified

Status: For the steppings affected see the Summary of Changes at the beginning of this section.

P20. *Speculative Page Fault May Cause Livelock*

Problem: If the processor detects a page fault, which is corrected before the operating system page fault handler can be called (e.g. a second processor or DMA activity modifies the page tables and the corrected page tables are left in a non-accessed or non-modified state) the processor may livelock. Intel has not been able to reproduce this erratum with commercial software.

Implication: This erratum occurs in systems where page tables are being modified by other processors. If this erratum is encountered, the processor will livelock resulting in a system hang or operating system failure.

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

Status: For the steppings affected see the Summary of Changes at the beginning of this section.

P21. *PAT Index MSB May Be Calculated Incorrectly*

Problem: When Mode B or Mode C paging support is enabled and all of the following events occur:

- A page walk returns the Page Directory Entry (PDE) for a large page from memory.
- A subsequent page walk returns the Page Table Entry (PTE) for a 4k page from memory and the Page Attribute Table (PAT) upper index bit in this PTE is set to 1b.

It is possible that the PAT upper index bit in the PTE is incorrectly ignored and assumed to be 0b. The result is that the memory type in the PAT that should have come from the corresponding PAT index [4-7] incorrectly comes from PAT index [0-3].

Implication: If an operating system has programmed the PAT in an asymmetrical fashion i.e. PAT[0-3] is different from PAT[4-7] then an incorrect memory type may be used.

Workaround: None identified

Status: For the steppings affected see the Summary of Changes at the beginning of this section.

P22. *System Bus Interrupt Messages Without Data And Which Receive A HardFailure Response May Hang The Processor*

Problem: When a system bus agent (processor or chipset) issues an interrupt transaction without data onto the system bus, and the transaction receives a HardFailure response, a potential processor hang can occur. The processor, which generates an inter-processor interrupt (IPI) that receives HardFailure response, will still log the MCA error event cause as HardFailure, even if the APIC causes a hang. Other processors, which are true targets of the IPI, will also hang on hardfailure-without-data, but will not record an MCA HardFailure event as a cause. If a HardFailure response occurs on a system bus interrupt message with data, the APIC will complete the operation so as not to hang the processor.

Implication: The processor may hang.

Workaround: None identified

Status: For the steppings affected see the Summary of Changes at the beginning of this section.

P23. *SQRTPD and SQRTSD May Return QNaN Indefinite Instead of Negative Zero*

Problem: When DAZ mode is enabled, and a SQRTPD or SQRTSD instruction has a negative denormal operand, the instruction will return a QNaN indefinite when the specified response should be zero.

Implication: When this erratum occurs, the instruction will return a QNaN indefinite when a zero is expected.

Workaround: Ensure that negative denormals are not used as operands to the SQRTPD or SQRTSD instructions when DAZ mode is enabled. Software could enable FTZ mode to ensure that negative denormals are not generated by computation prior to execution of the SQRTPD or SQRTSD instructions.

Status: For the steppings affected see the Summary of Changes at the beginning of this section.

P24. *Bus Invalidate Line Requests that Returns Unexpected Data May Result in L1 Cache Corruption*

Problem: When a Bus Invalidate Line (BIL) request receives unexpected data from a deferred reply, and a store operation write combines to the same address, there is a small window where the L1 cache is corrupt, and loads can retire with this corrupted data. This erratum occurs in the following scenario:

- A Read-For-Ownership (RFO) transaction is issued by the processor and hits a line in shared state in the L2 cache.
- The RFO is then issued on the system bus as a 0 length Read-Invalidate (BIL), since it doesn't need data, just ownership of the cache line.
- This transaction is deferred by the chipset.
- At some later point, the chipset sends a deferred reply for this transaction with an implicit write-back response. For this erratum to occur, no snoop of this cache line can be issued between the BIL and the deferred reply.
- The processor issues a write-combining store to the same cache line while data is returning to the processor. This store straddles an 8-byte boundary.

Due to an internal boundary condition, a time window exists where the L1 cache contains corrupt data, which could be accessed by a load.

Implication: The L1 cache may contain corrupted data. No known commercially available chipsets trigger the failure conditions.

Workaround: The chipset could issue a BIL (snoop) to the deferred processor to eliminate the failure conditions.

Status: For the steppings affected see the Summary of Changes at the beginning of this section.

P25. *Multi-processor Boot Protocol may not Complete with an IOQ Depth of One*

Problem: When the In-Order Queue (IOQ) depth is managed by the chipset to be one entry deep, the system may hang during the multi-processor boot protocol. This hang occurs when the chipset drives BNR# in such a way that the processors are continually throttled off the bus then released to access the bus in alternating cycles which never allows the multi-processor boot protocol to complete execution.

Implication: The system may hang during the multi-processor boot protocol.

Workaround: If the chipset drives BNR# in such a way that the processors are continually throttled off the bus then released to access the bus in alternating cycles, do not use In-Order Queue de-pipelining.

Status: For the steppings affected see the Summary of Changes at the beginning of this section.

P26. *Processor Flags #PF Instead of #AC on an Unlocked CMPXC8B Instruction*

Problem: If a data page fault (#PF) and alignment check fault (#AC) both occur for an unlocked CMPXC8B instruction, then #PF will be flagged.

Implication: Software that depends #AC before #PF will be affected since #PF is flagged in this case.

Workaround: Remove the software's dependency on the fact that #AC has precedence over #PF. Alternately, if the reload is due to a not present page, reload the page in the page fault handler and then restart the faulting instruction.

Status: For the steppings affected see the Summary of Changes at the beginning of this section.

P27. *Incorrect Data may be Returned When Page Tables are Located in Write Combining (WC) Memory*

Problem: If page directories and/or page tables are located in Write Combining (WC) memory, speculative loads to cacheable memory may complete with incorrect data.

Implication: Cacheable loads to memory mapped using page tables located in write combining memory may return incorrect data. Intel has not been able to reproduce this erratum with commercially available software.

Workaround: Do not place page directories and/or page tables in WC memory.

Status: For the steppings affected see the Summary of Changes at the beginning of this section.

P28. *FSW May not be Completely Restored after Page Fault on FRSTOR or FLDENV Instructions*

Problem: If the FPU operating environment or FPU state (operating environment and register stack) being loaded by an FLDENV or FRSTOR instruction wraps around a 64-Kbyte or 4-Gbyte boundary and a page fault (#PF) or segment limit fault (#GP or #SS) occurs on the instruction near the wrap boundary, the upper byte of the FPU status word (FSW) might not be restored. If the fault handler does not restart program execution at the faulting instruction, stale data may exist in the FSW.

Implication: When this erratum occurs, stale data will exist in the FSW.

Workaround: Ensure that the FPU operating environment and FPU state do not cross 64-Kbyte or 4-Gbyte boundaries. Alternately, ensure that the page fault handler restarts program execution at the faulting instruction after correcting the paging problem.

Status: For the steppings affected see the Summary of Changes at the beginning of this section.

P29. *Write Combining (WC) Load May Result in an Unintended Address on System Bus*

Problem: When the processor performs a speculative write combining (WC) load, down the path of a mispredicted branch, and the address happens to match a valid UnCacheable (UC) address translation with the Data Translation Look-Aside Buffer, an unintended UnCacheable load operation may be sent out on the system bus.

Implication: When this erratum occurs, an unintended load may be sent on system bus. Intel has only encountered this erratum during pre-silicon simulation.

Workaround: It is possible for the BIOS to contain a workaround for this erratum for some steppings of the processor.

Status: For the steppings affected, see the Summary of Changes at the beginning of this section.

P30. *Processor Issues Inconsistent Transaction Size Attributes for Locked Operation*

Problem: When the processor is in the Page Address Extension (PAE) mode and detects the need to set the Access and/or Dirty bits in the page directory or page table entries, the processor sends an 8 byte load lock onto the system bus. A subsequent 8 byte store unlock is expected, but instead a 4 byte store unlock occurs. Correct data is provided since only the lower bytes change, however external logic monitoring the data transfer may be expecting an 8-byte store unlock.

Implication: No known commercially available chipset are affected by this erratum.

Workaround: None identified at this time.

Status: For the steppings affected see the Summary of Changes at the beginning of this section.

P31. *Multiple Accesses to the Same S-State L2 Cache Line and ECC Error Combination May Result in Loss of Cache Coherency*

Problem: When a Read For Ownership (RFO) cycle has a 64 bit address match with an outstanding read hit on a line in the L2 cache which is in the S-state AND that line contains an ECC error, the processor should recycle the RFO until the ECC error is handled. Due to this erratum, the processor does not recycle the RFO and attempt to service both the RFO and the read hit at the same time.

Implication: When this erratum occurs, cache may become incoherent.

Workaround: None identified at this time.

Status: For the steppings affected see the Summary of Changes at the beginning of this section.

P32. *IA32_MC0_ADDR and IA32_MC0_MISC Registers Will Contain Invalid or Stale Data Following a Data, Address, or Response Parity Error*

Problem: If the processor experiences a data, address, or response parity error, the ADDRV and MISCV bits of the IA32_MC0_STATUS register are set, but the IA32_MC0_ADDR and IA32_MC0_MISC registers are not loaded with data regarding the error.

Implication: When this erratum occurs, the IA32_MC0_ADDR and IA32_MC0_MISC registers will contain invalid or stale data.

Workaround: Ignore any information in the IA32_MC0_ADDR and IA32_MC0_MISC registers after a data or response parity error.

Status: For the steppings affected see the Summary of Changes at the beginning of this section.

P33. *When the Processor is in the System Management Mode (SMM), Debug Registers may be Fully Writeable*

Problem: When in System Management Mode (SMM), the processor executes code and stores data in the SMRAM space. When the processor is in this mode and writes are made to DR6 and DR7, the processor should block writes to the reserved bit locations. Due to this erratum, the processor may not block these writes. This may result in invalid data in the reserved bit locations.

Implication: Reserved bit locations within DR6 and DR7 may become invalid.

Workaround: Software may perform a read/modify/write when writing to DR6 and DR7 to ensure that the value in the reserved bits are maintained.

Status: For the steppings affected, see the *Summary of Changes* at the beginning of this section.

P34. *Associated Counting Logic must be Configured when Using Event Selection Control (ESCR) MSR*

Problem: ESCR MSRs allow software to select specific events to be counted, with each ESCR usually associated with a pair of performance counters. ESCRs may also be used to qualify the detection of at-retirement events that support precise-event-based sampling (PEBS). A number of performance metrics that support PEBS require a 2^{nd} ESCR to tag uops for the qualification of at-retirement events. (The first ESCR is required to program the at-retirement event.) Counting is enabled via counter configuration control registers (CCCR) while the event count is read from one of the associated counters. When counting logic is configured for the subset of at-retirement events that require a 2^{nd} ESCR to tag uops, at least one of the CCCRs in the same group of the 2^{nd} ESCR must be enabled.

Implication: If no CCCR/counter is enabled in a given group, the ESCR in that group that is programmed for tagging uops will have no effect. Hence a subset of performance metrics that require a 2^{nd} ESCR for tagging uops may result in 0 count.

Workaround: Ensure that at least one CCCR/counter in the same group as the tagging ESCR is enabled for those performance metrics that require 2 ESCRs and tagging uops for at-retirement counting.

Status: For the steppings affected, see the *Summary of Changes* at the beginning of this section.

P35. *Livelock may Occur when Bus Parking is Disabled*

Problem: A livelock may occur when processor bus parking is disabled, and when (1) the processor is the symmetric owner of the bus with one internal request pending, and (2) the processor observes the assertion of BPRI#. The processor assumes that it will assert ADS# and deasserts BREQ without issuing its pending request. Assertion of BPRI# coincident with the arbitration phase of the same processor that still has only one outstanding internal request will result in that processor being livelocked. Any change to the regular pattern of BPRI# assertion noted above or the arrival of a second internal transaction will release the processor from the livelock condition.

Implication: This erratum may result in a livelock.

Workaround: This erratum can be avoided by enabling bus parking.

Status: For the steppings affected, see the *Summary of Changes* at the beginning of this section.

DOCUMENTATION CHANGES

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

- *Intel® Xeon™ Processor at 1.40 GHz, 1.50 GHz, 1.70 and 2 GHz* datasheet (Order Number 249665)
- *IA-32 Intel® Architecture Software Developer's Manual, Volumes 1, 2 and 3* (Order Numbers 245470, 245471, and 245472, respectively)

All Documentation Changes will be incorporated into a future version of the appropriate Intel® Xeon™ processor documentation.

P1. Machine Check Exception Detected When BINIT# Drive Enabled

The last paragraph of section 13.7.1 in the *IA-32 Intel Architecture Software Developer's Manual, Volume 3: System Programming Guide* currently states:

13.7.1 Machine Check Exception Handler

The MCIP flag in the IA32_MCG_STATUS register indicates whether a machine-check exception was generated. Before returning from the machine-check exception handler, software should clear this flag so that it can be used reliably by an error logging utility. The MCIP flag also detects recursion. The machine-check architecture does not support recursion. When the processor detects machine-check recursion, it enters the shutdown state.

It should state:

13.7.1 Machine Check Exception Handler

The MCIP flag in the IA32_MCG_STATUS register indicates whether a machine-check exception was generated. Before returning from the machine-check exception handler, software should clear this flag so that it can be used reliably by an error logging utility. The MCIP flag also detects recursion. The machine-check architecture does not support recursion. When the processor detects machine-check recursion, it enters the shutdown state.

Note: For complete operation of the processors machine check capabilities it is essential that the system BIOS enable BINIT# drive and BINIT# observation. This allows the processor to use BINIT# to clear internal and potentially external blocking state and correctly report a wider range of machine check exceptions. For example, on an Intel® Pentium® III processor that is executing a locked CMPXCHG8B instruction and a machine check exception is seen on the initial data read, but the comparison operation fails, the processor unlocks the bus after completion of the locked sequence by asserting a BINIT# signal. Without BINIT# drive (UP environment) or BINIT# drive and observation (MP environment) enabled, the machine check error is logged, but the machine check exception is not taken (if MCE's are enabled).

P2. The Encoding of "Immediate to Register" of "AND" Instruction is Missing

The *IA-32 Intel Architecture Software Developer's Manual, Volume 2: Instruction Set Reference* Appendix B, table B-10 is missing the encoding of "Immediate to register" of "AND" instruction. The encoding of "Immediate to register" of "AND" instruction is as follows:

'1000 00sw 11100 reg : immediate data'

P3. The 'reg' Field of CMPXCHG8B Instruction Encoding Must be 001

The *IA-32 Intel Architecture Software Developer's Manual, Volume 2: Instruction Set Reference* Appendix B, table B-10 incorrectly documents opcode encoding format of CMPXCHG8B. It currently states:

CMPXCHG8B- Compare and Exchange 8 Bytes

Memory, register	0000 1111 : 1100 0111 : mod reg r/m

It should state:

CMPXCHG8B- Compare and Exchange 8 Bytes

Memory, register	0000 1111 : 1100 0111 : mod 001r/m

P4. SCAS/SCASB/SCASW/SCASD Encoding Operand is Incorrect

The *IA-32 Intel Architecture Software Developer's Manual, Volume 2: Instruction Set Reference* Appendix B, table B-10 incorrectly documents opcode encoding format of SCASx. It currently states:

SCAS/SCASB/SCASW/SCASD - Scan String 1101 111w

It should state:

SCAS/SCASB/SCASW/SCASD - Scan String 1010 111w

P5. XCHG Encoding Operand Information (1-byte form) Does Not have a w-Bit, hence the reg Size is Implied. The AL Register is Not a Valid Option for this 1-byte Encoding

The *IA-32 Intel Architecture Software Developer's Manual, Volume 2: Instruction Set Reference* Appendix B, table B-10 incorrectly documents opcode encoding format of XCHG. It currently states:

XCHG - Exchange Register/Memory with Register

register1 with register2	1000 011w : 11 reg1 reg2
AL, AX, or EAX with reg	1001 0 reg
memory with reg	1000 011w : mod reg r/m

It should state:

XCHG - Exchange Register/Memory with Register

register1 with register2	1000 011w : 11 reg1 reg2
AX or EAX with reg	1001 0 reg
memory with reg	1000 011w : mod reg r/m

SPECIFICATION CLARIFICATIONS

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

- *Intel® Xeon™ Processor at 1.40 GHz, 1.50 GHz, 1.70 and 2 GHz* datasheet (Order Number 249665)
- *IA-32 Intel® Architecture Software Developer's Manual, Volumes 1, 2 and 3* (Order Numbers 245470, 245471, and 245472, respectively)

All Specification Clarifications will be incorporated into a future version of the appropriate Intel® Xeon™ processor documentation.

There are no specification clarifications to report.

SPECIFICATION CHANGES

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

- *Intel® Xeon™ Processor at 1.40 GHz, 1.50 GHz, 1.70 and 2 GHz* datasheet (Order Number 249665)
- *IA-32 Intel® Architecture Software Developer's Manual, Volumes 1, 2 and 3* (Order Numbers 245470, 245471, and 245472, respectively)

All Specification Changes will be incorporated into a future version of the appropriate Intel® Xeon™ processor documentation.

There are no specification changes to report.