



Errata for UltraSPARC-IIIi

Part No. 820-4009-10

1/9/08

Sun Microsystems, Inc.
www.sun.com

Submit comments about this document at: <http://www.sun.com/hwdocs/feedback>

Copyright 2007 Sun Microsystems, Inc., 4150 Network Circle, Santa Clara, California 95054, U.S.A. All rights reserved.

Sun Microsystems, Inc. has intellectual property rights relating to technology that is described in this document. In particular, and without limitation, these intellectual property rights may include one or more of the U.S. patents listed at <http://www.sun.com/patents> and one or more additional patents or pending patent applications in the U.S. and in other countries.

This document and the product to which it pertains are distributed under licenses restricting their use, copying, distribution, and decompilation. No part of the product or of this document may be reproduced in any form by any means without prior written authorization of Sun and its licensors, if any.

Third-party software, including font technology, is copyrighted and licensed from Sun suppliers.

Parts of the product may be derived from Berkeley BSD systems, licensed from the University of California. UNIX is a registered trademark in the U.S. and in other countries, exclusively licensed through X/Open Company, Ltd.

Sun, Sun Microsystems, the Sun logo, Java, docs.sun.com, Sun Blade, SunVTS, SunSolve, SunService, Sun Fire, and Solaris are trademarks or registered trademarks of Sun Microsystems, Inc. in the U.S. and in other countries.

All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. in the U.S. and in other countries. Products bearing SPARC trademarks are based upon an architecture developed by Sun Microsystems, Inc.

The OPEN LOOK and Sun™ Graphical User Interface was developed by Sun Microsystems, Inc. for its users and licensees. Sun acknowledges the pioneering efforts of Xerox in researching and developing the concept of visual or graphical user interfaces for the computer industry. Sun holds a non-exclusive license from Xerox to the Xerox Graphical User Interface, which license also covers Sun's licensees who implement OPEN LOOK GUIs and otherwise comply with Sun's written license agreements.

U.S. Government Rights—Commercial use. Government users are subject to the Sun Microsystems, Inc. standard license agreement and applicable provisions of the FAR and its supplements.

DOCUMENTATION IS PROVIDED "AS IS" AND ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, ARE DISCLAIMED, EXCEPT TO THE EXTENT THAT SUCH DISCLAIMERS ARE HELD TO BE LEGALLY INVALID.

Copyright 2007 Sun Microsystems, Inc., 4150 Network Circle, Santa Clara, Californie 95054, États-Unis. Tous droits réservés.

Sun Microsystems, Inc. possède les droits de propriété intellectuels relatifs à la technologie décrite dans ce document. En particulier, et sans limitation, ces droits de propriété intellectuels peuvent inclure un ou plusieurs des brevets américains listés sur le site <http://www.sun.com/patents>, un ou les plusieurs brevets supplémentaires ainsi que les demandes de brevet en attente aux les États-Unis et dans d'autres pays.

Ce document et le produit auquel il se rapporte sont protégés par un copyright et distribués sous licences, celles-ci en restreignent l'utilisation, la copie, la distribution, et la décompilation. Aucune partie de ce produit ou document ne peut être reproduite sous aucune forme, par quelque moyen que ce soit, sans l'autorisation préalable et écrite de Sun et de ses bailleurs de licence, s'il y en a.

Tout logiciel tiers, sa technologie relative aux polices de caractères, comprise, est protégé par un copyright et licencié par des fournisseurs de Sun.

Des parties de ce produit peuvent dériver des systèmes Berkeley BSD licenciés par l'Université de Californie. UNIX est une marque déposée aux États-Unis et dans d'autres pays, licenciée exclusivement par X/Open Company, Ltd.

Sun, Sun Microsystems, le logo Sun, Java, docs.sun.com, Sun Blade, SunVTS, SunSolve, SunService, Sun Fire, et Solaris sont des marques de fabrique ou des marques déposées de Sun Microsystems, Inc. aux États-Unis et dans d'autres pays.

Toutes les marques SPARC sont utilisées sous licence et sont des marques de fabrique ou des marques déposées de SPARC International, Inc. aux États-Unis et dans d'autres pays. Les produits portant les marques SPARC sont basés sur une architecture développée par Sun Microsystems, Inc.

L'interface utilisateur graphique OPEN LOOK et Sun™ a été développée par Sun Microsystems, Inc. pour ses utilisateurs et licenciés. Sun reconnaît les efforts de pionniers de Xerox dans la recherche et le développement du concept des interfaces utilisateur visuelles ou graphiques pour l'industrie informatique. Sun détient une licence non exclusive de Xerox sur l'interface utilisateur graphique Xerox, cette licence couvrant également les licenciés de Sun implémentant les interfaces utilisateur graphiques OPEN LOOK et se conforment en outre aux licences écrites de Sun.

LA DOCUMENTATION EST FOURNIE "EN L'ÉTAT" ET TOUTES AUTRES CONDITIONS, DÉCLARATIONS ET GARANTIES EXPRESSES OU TACITES SONT FORMELLEMENT EXCLUES DANS LA LIMITE DE LA LOI APPLICABLE, Y COMPRIS NOTAMMENT TOUTE GARANTIE IMPLICITE RELATIVE À LA QUALITÉ MARCHANDE, À L'APTITUDE À UNE UTILISATION PARTICULIÈRE OU À L'ABSENCE DE CONTREFAÇON.





1. Errata Table

Table 1: UltraSPARC-IIIi Errata

Errata #	Version 2.3	Version 2.4	Version 3.2	Version 3.4	See ...
1	✓	✓	✓	✓	page 5
2	✓	✓	✓	✓	page 6
3	✓	✓	✓	✓	page 6
4	✓	✓	✓	✓	page 7
5	✓	✓	✓	✓	page 8
6	✓	✓	✓	✓	page 9
7	✓	✓	✓	✓	page 10
8	✓	✓	✓	✓	page 11
9	✓	✓	✓	✓	page 12
10	✓	✓	Not applicable	Not applicable	page 12
11	✓	Not applicable	Not applicable	Not applicable	page 13
12	✓	✓	Not applicable	Not applicable	page 14
13	✓	✓	Not applicable	Not applicable	page 15
14	✓	Not applicable	Not applicable	Not applicable	page 16
15	✓	Not applicable	Not applicable	Not applicable	page 17
16	✓	✓	Not applicable	Not applicable	page 18
17	✓	✓	Not applicable	Not applicable	page 19
18	✓	✓	✓	✓	page 19
19	✓	✓	✓	✓	page 20
20	✓	✓	Not applicable	Not applicable	page 23
21	✓	✓	Not applicable	Not applicable	page 25



Table 1: UltraSPARC-III Errata (Continued)

Errata #	Version 2.3	Version 2.4	Version 3.2	Version 3.4	See ...
22	✓	✓	✓	Not applicable	page 26
23	✓	✓	✓	✓	page 27



2. Errata Descriptions and Workarounds

Erratum #1: A diagnostic read of a fully associative Translation Lookaside Buffer (TLB) translation table entry may return incorrect data.

Applicability:

UltraSPARC-III Versions 2.3, 2.4, 3.2, and 3.4.

Description:

This problem happens under following conditions.

- **Data TLBs:** Any memory access instruction that misses the Data TLB and is followed by a diagnostic read access (Idxa from ASI_DTLB_DATA_ACCESS_REG, ASI=0x5d) from the fully associative TLBs and the target TTE has page size is set to 64 KB, 512 KB, or 4 MB.
- **Instruction TLBs:** Any instruction that misses the Instruction TLB and is followed by a diagnostic read access (Idxa from ASI_ITLB_DATA_ACCESS_REG, ASI=0x55) from the fully associative TLBs and the target TTE has page size set to 64 KB, 512 KB, or 4 MB.

Impact:

The data returned from the Translation Table Entry (TLB) will be incorrect.

Workaround:

This problem can be overcome by reading the fully associative TLB TTE twice, back to back. The first access may return incorrect data if the above conditions are met; however, the second access will return correct data.

Status:

This bug will not be fixed in future releases of the silicon.



Erratum #2: An incorrect context number is placed in the Data Memory Management Unit (DMMU) Tag Access register after a *data_access_exception* trap.

Applicability:

UltraSPARC-III Versions 2.3, 2.4, 3.2, and 3.4.

Description:

When a floating point load on either ALU pipe (A0/A1) takes a *data_access_exception* trap, the wrong context number may be saved in the context field of the Data Memory Management Unit (DMMU) Tag Access register (ASI 0x58, VA=0x30).

Impact:

The incorrect context number is provided to the *data_access_exception* trap handler.

Workaround:

Use the CT field (context ID) of the DMMU Synchronous Fault Status register (SFSR, ASI 0x58, VA=0x18). The *data_access_exception* trap handler should refer to either the primary, secondary, or nucleus context registers to convert the 2-bit context ID into a 13-bit context number.

Status:

This bug will not be fixed in future releases of the silicon.

Erratum #3: A read to a non-existent Jalapeno Bus (JBus) Agent ID can cause the CPU to hang if ESTATE.PERREN is disabled.

Applicability:

UltraSPARC-III Versions 2.3, 2.4, 3.2, and 3.4.

**Description:**

If ESTATE.PERREN is programmed to 0, prefetch read requests and other read requests can interact in the JBus Unit (JBU) to cause a CPU hang condition. When ESTATE.PERREN is enabled, reads to non-existent JBus Agent IDs will normally result in an unmapped error in the Asynchronous Fault Status Register (AFSR.TO) and a deferred trap and later a hardware time-out (AFSR.JETO) that will reset the CPU.

Impact:

A CPU hangs when ESTATE.PERREN is 0.

There should be no reason for software to disable fatal reset traps on JBus protocol errors by writing ESTATE.PERREN to 0. When these traps are disabled, errors which can corrupt data and system state (AFSR.JETO, AFSR.JEIC, AFSR.JEIS, AFSR.JEIT) will not cause a fatal reset and will result in unpredictable behavior. Software should enable this bit under all conditions.

Workaround:

Software should not disable ESTATE.PERREN. In addition, software should not do any probing with reads to invalid addresses that will result in a fatal reset. If address probing needs to be done, stores should be used. Stores will cause AFSR.TO to be set, but will not hang the CPU.

The best alternative for software that is to find the existent memory space and JBus agents is to read the JBUS_CONFIG register, which contains the existing Agent IDs in bits [50:44]. For each bit that is set, that particular agent exists and cacheable space exists (e.g., if JBUS_CONFIG[45]==1, cacheable space with PA[40:36]==0xb00001 exists).

Status:

This bug will not be fixed in future releases of the silicon.

Erratum #4: Diagnostic loads/stores to Address Space Identifier (ASI) 0x4e (ASI_ECACHE_TAG or ASI_ECACHE_FLUSH) during normal operation can break coherency or hang the CPU.**Applicability:**

UltraSPARC-III Versions 2.3, 2.4, 3.2, and 3.4.

**Description:**

The following operations can result in a CPU hang or coherency corruption if done during normal operation:

- An attempt to invalidate an L2 tag through the use of ASI 0x4e.
- Loads and stores to ASI 0x4e (L2 tag access or L2-cache flush) that interact with snoops, copybacks, other stores, and cache victim lines.

Impact:

CPU can hang or lose coherency when intermixing normal L2 tag behavior with ASI 0x4e loads and stores.

Workaround:

All L2 tag diagnostic accesses or flushes should be surrounded with Memory Barrier Instructions (MEMBARs) and run out of non-cacheable space to protect the software from running into this case.

Status:

This bug will not be fixed in future releases of the silicon.

Erratum #5: Clarification of the UltraSPARC-III User Manual description of the Prefetch Cache-related Data Cache Unit (DCU) Control Register bits.**Applicability:**

UltraSPARC-III Versions 2.3, 2.4, 3.2, and 3.4.

Description:

The Joint Programming Specification (JPS1) supplement text "PE=0 causes all prefetches to be handled as P-cache misses" is misleading.

A more detailed description and behavior of the Pcache Data Cache Unit (DCU) control registers bits is as follows:

- **PE Prefetch Cache Enable** (bit 45): If cleared, no hardware prefetches will be generated and all software prefetches will be treated as No Operation (NOP). No Floating Point (FP) load miss data (32B) will be installed from the L2-cache into the P-cache and all P-cache references for FP-loads are treated as a P-cache miss.



- **HPE Prefetch Cache Hardware Prefetch Enable** (bit 44): If cleared, the P-cache does not generate any hardware prefetch requests to the L2-cache and no FP-load miss data (32B) is installed from L2-cache to the P-cache.
- **SPE Software Prefetch Enable** (bit 43): If cleared, software prefetch instructions do not generate a request to the pipeline,. They will be treated as NOPs.

Usage:

- For hardware prefetch generation, both HPE and PE should be turned ON.
- For software prefetch generation, both SPE and PE should be turned ON.

Status:

This erratum updates the documentation.

Erratum #6: In a Multi-processor (MP) system, replacing a load with a store while it is executing can cause a hang

Applicability:

UltraSPARC-IIIi Versions 2.3, 2.4, 3.2, and 3.4.

Description:

There were two cases discovered which exhibit this hang problem.

- The first is a load that misses the D-cache is recirculating and it is subsequently changed into a store. In this case, the Integer Execution Unit (IEU) Store Queue (STQ) counter hits an illegal state and the machine hangs.
- The second case is similar to the first, where the load is recirculating waiting for data from memory. If the load is changed into an ALUop, the IEU STQ counter is not hitting the illegal state, but if the next memory reference is a store in the instruction stream, the same problem can be seen.

Impact:

When a load that missed the Data Cache (D-cache) is recirculating and it is changed into a store, the system hangs.

**Workaround:**

Do not write self-modifying code that changes a possibly-executing load into a store or changes that load into an ALUop such that the next memory reference will be a store. It's best to leave the load alone until one is sure that it will not be executed.

The generic workaround is for the system software to ensure that when memory is mapped by an Instruction Translation Look-Aside Buffer (iTLB) entry in a processor that no other processors have writable Data Translation Look-Aside Buffer (dTLB) entries for the same memory. There are complexity and performance effects with this approach, however (e.g., slowing down TLB misses and scalability due to more demaps).

Status:

This bug will not be fixed in future releases of the silicon.

Erratum #7: It is not legal to relax the bstore/bstore* and MEMBAR #Sync rule when Physical Address PA[12:5] match. Order is required.**Applicability:**

UltraSPARC-IIIi Versions 2.3, 2.4, 3.2, and 3.4.

Description:

The Memory Barrier Instruction (MEMBAR) rules table (i.e., table 8-3 in the UltraSPARC-IIIi Users Manual) shows that neither a bstore followed by bstore commit case nor a bstore followed by the bstore case require a MEMBAR #Sync when there is a Virtual Address VA[12:5] match. This is incorrect. The MEMBAR #Sync is required.

Impact:

The failure mode is “wrong data”.

Status:

This erratum updates the documentation.



Erratum #8: The UltraSPARC-IIIi has only a one-deep queue for servicing noncacheable reads with RDERR returns.

Applicability:

UltraSPARC-IIIi Versions 2.3, 2.4, 3.2, and 3.4.

Description:

The UltraSPARC-IIIi does not have any accessible noncacheable space and any noncacheable read directed at a CPU should be responded with an unmapped RDERR packet on the JBus. Noncacheable reads, however, are not set up with the same flow control as normal reads, and there is just a one-entry queue for noncacheable read responses.

If there is a RDERR packet waiting to go out on the JBus and another noncacheable read is received, the ADTYPE of the first will be overwritten by the second, causing two RDERR return packets to be sent to the same device. If the sources of the two noncacheable reads are different devices, this may hang the system.

If there are multiple noncacheable reads directed at a single CPU and the RDERR response to the first noncacheable read cannot be serviced due to the outgoing queue begin full, the second noncacheable read will overwrite the first. The first read will not get a response and will eventually time-out. The second will be responded to with an unmapped RDERR packet. In this case, a JBus read is dropped, and a hardware time-out is the most likely consequence.

Impact:

If multiple noncacheable reads are directed to a single CPU, a RDERR packet can be sent returned to the wrong device. It is also possible for a noncacheable read directed at the CPU to be dropped, causing a system time-out and a fatal reset if enabled.

Workaround:

None. Software must not probe devices using noncacheable reads, since this can causes a possible hang situation or a hardware time-out and system reset.

Status:

This bug will not be fixed in future releases of the silicon.



Erratum #9: Diagnostic Address Space Identifier (ASI) writes to the L2 tag with bit VA[31]=1 will hang the CPU.

Applicability:

UltraSPARC-III Versions 2.3, 2.4, 3.2, and 3.4.

Description:

The ASI register address for both L2-cache flushes and diagnostic L2 tag accesses is 0x4E. For flushes, VA[31] should be one for the ASI read and for diagnostic accesses, VA[31] should be zero for the ASI read/write. If software attempts a diagnostic L2 tag write with VA[31]=1, the L2-cache unit thinks that it is doing a flush operation and after the tag write, will wait for the fill data to come from the JBU, which never happens, thereby hanging the CPU.

Impact:

The system hangs during Address Space Identifier (ASI) writes the L2 tag with Virtual Address VA[31]=1.

Workaround:

Software must not write ASI 0x4E with VA[31]=1.

Status:

This bug will not be fixed in future releases of the silicon.

Erratum #10: The mixture of fdiv and fsqrt instructions with a pdist instruction cause an incorrect Floating Point (FP) register result.

Applicability:

UltraSPARC-III Versions 2.3 and 2.4.

Description:

The sequence of code that generates the problem is unlikely by nature. There is a fdiv or fsqrt instruction that has a write-after-write (WAW) hazard with a subsequent FP or graphics operation. The second FP operation will overwrite the result of the fdiv or sqrt operation before it can be used, making the sequence unlikely. There is a second fdiv or sqrt instruction that is followed by



a pdist instruction that has both a read-after-write (RAW) and WAW hazard with the rd of the pdist instruction. pdist is a special instruction that uses both rs1 and rs2 as source operands and an integer in the rd as a source operand, and then stores the results into rd. Also unlikely is a pdist that would normally use the result of an fdiv or fsqrt instruction as its source operand.

What happens is that the WAW hazard before the first fdiv or fsqrt instruction and the subsequent FP or GR operation is used incorrectly as a WAW to affect the state of the later, and second, fdiv or fsqrt instruction, making it unable to bypass its result to the following instructions (in this case, the pdist instruction). When the pdist instruction executes, the rd register that it wants to use in its computation appears to be available in the FP register file, instead of still being computed by the fdiv or fsqrt instruction. Thus, the pdist instruction executes at the same time as the fdiv or fsqrt instruction does (which is asynchronous to the pipeline) and uses the old rd from the FPRF incorrectly.

Impact:

The result after a Pixel Component Distance (pdist) instruction execution is either incorrect or the pdist result appears to be dropped completely.

Workaround:

Ensure that the failing code sequence does not exist in current software routines. Check that no code has a pdist instruction following an fdiv or fsqrt instruction.

Status:

This bug will not be fixed in future releases of the silicon.

Erratum #11: The memory controller corrupts read data in the presence of reads to out-of-range addresses while in Energy Star (E-star) mode.

Applicability:

UltraSPARC-III Versions 2.3, 2.4, and 3.2.

Description:

When in Energy Star (E-star) mode and the software is generating out-of-range accesses to memory (as is the case during as L2 flush), data corruption results in a RDERR, internal processor error (IERR), and Timeout (TO).

**Impact:**

When a CPU is in E-star mode, the memory controller can return RDERR responses to valid memory access in the presence of out-of-range accesses. Typically, the out-of-range memory accesses are generated by the operating system when flushing the L2-cache in the presence of ECC errors. During the flushing, other CPUs in the system can be accessing the memory of the CPU that is having the L2 flushed.

Workaround:

Do not to allow out-of-range accesses during E-star mode (i.e., do not allow L2-cache flushing). There is no bit to disable address range checking in the MCU.

Status:

This bug will not be fixed in future releases of the silicon.

Erratum #12: An illegal L2 install state is logged as an Asynchronous Fault Status Register System Interface Protocol Error (AFSR.JEIS) when flushing the L2-cache.**Applicability:**

UltraSPARC-III Versions 2.3 and 2.4.

Description:

When the Memory Controller Unit (MCU) returns a read error because the address from an L2-cache flush is out-of-range, the logic in the JBU considers this a fast time-out and does not wait for all J_PACK snoops to return from the Jalapeno Bus (JBus). The fill is sent to the L2-cache along with a time-out signal and the ECU will install the line in the L2 in the invalid state.

Because the JBU logic detects valid install state for fills to the L2, and the request for a flush is RDS, the install state is detected as illegal because invalid is normally an illegal install state for RDS. An illegal install state is flagged (AFSR.JEIS). Installing in invalid state should be expected for the L2 flush case.

**Impact:**

In the presence of multiple reads outstanding in the JBU while doing L2-cache flushing, a system reset occurs (if enabled) and the Asynchronous Fault Status Register System Interface Protocol Error (AFSR.JEIS) is logged.

Workaround:

When doing an L2-cache flushing or any reads that access an out-of-range address in the system, the bit that enables fatal resets for the JEIS should be cleared (e.g., ESTATE.PERREN, ASI=0x4b, bit 21). After finishing, clear AFSR.JEIS and re-enable the above bit for fatal resets on JBus protocol errors.

Status:

This bug will not be fixed in future releases of the silicon.

Erratum #13: The Asynchronous Fault Status Register AFSR.J_ARB_WIN field can be incorrect in a 4-way system when Jalapeno Bus (JBus) errors are logged.**Applicability:**

UltraSPARC-III Versions 2.3 and 2.4.

Description:

The AFSR.J_ARB_WIN field is intended to log the JBus driver at the time an error is detected on the JBus. This logic works properly for JBus systems without the JBus repeater chip and in most cases with the repeater chip.

A corner case exists: If an error is sent across the JBus repeater chip at the same time that a JBus arbitration grant is lost from the agent that is across the repeater to an agent on a local segment to the CPU that is logging the error, an incorrect driver ID will be logged.

Impact:

The Asynchronous Fault Status Register AFSR.J_ARB_WIN field shows a different value than what is expected when a Timeout (TO) or Bus Error From System Bus (BERR) error is logged when sent across the Jalapeno Bus (JBus) repeater chip.

**Workaround:**

None. On systems that use the JBus repeater chip on the JBus (i.e., typically 4-socket systems), the AFSR.J_ARB_WIN field cannot be guaranteed to be correct in all cases when JBus errors are detected.

Status:

This bug will not be fixed in future releases of the silicon.

Erratum #14: L2 Cache ECC errors can be caused by data corruption in the Memory Controller Unit (MCU).**Applicability:**

UltraSPARC-III Versions 2.3, 2.4, and 3.2.

Description:

Under corner-case conditions, data can be mishandled by the Memory Data Buffer within the Memory Interface Unit (MCU). This mishandling results in data corruption for a local read.

Impact:

The L2-cache has data corrupted in one 16 byte word of a cacheline (usually the last 16 bytes). The corruption can be detected as either an L2-cache ECC error or not detected at all. The data and check bits will be completely corrupted. The data and check bits will appear as an AND of the correct data and another piece of data recently read from memory.

Workaround:

There is only one complete workaround:

1. Use a memory ratio below 10:1, since it is not possible to see this problem for memory ratios 8:1 and 9:1.

The following workarounds reduce the frequency of the bug, but do not eliminate the possibility of seeing it:

2. Use an ASI_MCU_CTL_REG2.AREN setting where $AREN + (CASL * MCLK \text{ ratio})$ is an even number.
3. Disable the Early Access feature (i.e., clear bit 29 of ASI_MCU_CTL_REG1).



4. Set the Memory Data Buffer size to 1 entry (i.e., clear bit 28 of ASI_MCU_CTL_REG1).

Workarounds 3 and 4 will work for any combination of ratios. Workaround 3 has smaller performance impact.

Status:

This bug will not be fixed in future releases of the silicon.

Erratum #15: In a 4-way system, a live-lock case on the JBus causes a Memory Controller Unit Time Out (MCU_TO), which results in an AFSR.IERR logging and a system reset.

Applicability:

UltraSPARC-III Versions 2.3, 2.4, and 3.2.

Description:

With a specific kind of JBus traffic and CPU traffic in a 4-way system, a performance issue is created during internal arbitration in the JBus Transaction Issue Unit (TIU). The performance issue becomes a functional issue when the Memory Read Queue times out after it is unable to issue a request for 8,000 cycles. The end result is an internal error and a fatal reset if enabled.

Impact:

An AFSR.IERR is logged and the system resets when there is a stream of local block stores (causing INV transactions) and high JBus utilization with many outstanding reads. A cyclical pattern appears on the JBus, which prevents data from returning from memory.

Workaround:

The easiest way to overcome this case is to intersperse the local block stores (or writebacks) with reads that don't cause writebacks such that the string of INV transactions is interrupted, allowing read data to go out onto the bus from the Data Out Buffer (DOB). The following can break the live-lock from occurring:

- Other agents are not requesting the bus constantly.
- If AOK turns off a long time because of too many outstanding addresses.



- If DOK turns off.
- Any other traffic from the local CPU to Jalapeno Bus (JBus) other than INV transactions.
- Any foreign request that hits in the L2-cache.

Status:

This bug will not be fixed in future releases of the silicon.

Erratum #16: On a local DRAM Uncorrectable Error (UE), the two LSBs of the check bits from memory are flipped when installed in the L2.

Applicability:

UltraSPARC-IIIi Versions 2.3 and 2.4.

Description:

On other UltraSPARC-III-based processors, the L2 is installed with the data and check bits that were read from DRAM on an uncorrectable error (UE).

UltraSPARC-IIIi creates a signalling UE case, where new ECC check bits are generated for the data and then the lower two bits of the check bits are flipped. If the line is flushed (i.e., if the data is installed in M state), this causes a UE when read from the L2, with a syndrome of 0x003. This notifies software that this data is a result of a previous UE somewhere else in the system. However, in this case new ECC check bits are not generated and UltraSPARC-IIIi flips the two LSBs that were read from DRAM.

In the case of a DRAM UE, the data is normally installed in the L2 in the Invalid state, but can be installed in the Modified state in the case of a write.

Impact:

On a local DRAM uncorrectable error (AFSR.UE), the data and check bits installed in the L2 are not the data and check bits read from the DRAM. The two LSBs of the check bits are flipped.

Workaround:

None. Currently, software should panic on a UE and the kernel will scrub the L2 as well as the DRAM. This could become a problem if software is hardened to retry memory operations.

**Status:**

This bug will not be fixed in future releases of the silicon.

Erratum #17: An SDRAM out-of-range write (AFSR.UMS) may corrupt the following SDRAM read.**Applicability:**

UltraSPARC-III Versions 2.3 and 2.4.

Description:

When an out-of-range physical address (PA) SDRAM write is followed immediately by an in-range PA DRAM read having the same DEVICE, BANK, and ROW as the write (except for a difference in the out-of-range bits PA[31:28]), the read returns wrong data.

Impact:

If a write to memory with an out-of-range PA occurs (causing AFSR.UMS to be set), a memory read immediately following the read may return corrupted data.

Workaround:

Software should not perform any writes to out-of-range memory locations.

Status:

This bug will not be fixed in future releases of the silicon.

Erratum #18: In privileged mode, a store alternate using Address Space Identifier (ASI) 0x64 hangs the processor.**Applicability:**

UltraSPARC-III Versions 2.3, 2.4, 3.2, and 3.4.

**Description:**

In privileged mode, a store alternate operation using Address Space Identifier (ASI) 0x64 should be flagged as usage of an illegal ASI and result in a *data_access_exception* trap. However, this checking is not present and the store is allowed to execute. The store, however, is never acknowledged since ASI 0x64 doesn't exist.

Impact:

The machine hangs in privileged mode. User mode behavior is correct, so a malevolent user cannot hang the machine. ASI 0x64 behavior is as follows:

	user mode	privileged mode
ASI LOAD	<i>privileged_action</i> trap	<i>data_access_exception</i> trap
ASI STORE	<i>privileged_action</i> trap	CPU HANG

Workaround:

Privileged code should not issue store alternates to ASI 0x64.

Status:

This bug will not be fixed in future releases of the silicon.

Erratum #19: A delay slot involving Delayed Control Transfer Instructions (DCTIs) may not be properly executed, and subsequent execution of certain instructions may result in skipping the original instruction stream and instead executing a different instruction stream.

Applicability:

UltraSPARC-III Versions 2.3, 2.4, 3.2, and 3.4.



Description:

One example of a failing code sequence is as follows: The dcti-couple is represented here as back-to-back branches: branch_1 and branch_2. The last instruction, the add in the delay slot of a branch, is dropped on the nth iteration through this code, but the number n is not significant:

```
        ldsb      [%l1], %o6
        bcs,pt   %icc, .-0xC
        fmovs   %f4, %f17
        nop
        fsubs   %f17, %f5, %f17
        be,a,pt %xcc, .+0x4
        subcc   %o7, 0x2E4, %o7
        fcmps   %fcc0, %f17, %f6

branch_1: bn,pn   %icc, .+0x0
branch_2: fbue   .-0x18
d_slot_1: andn   %o7, %o6, %o7
branch_3: brlz,a,pt %o5, .-0x54
d_slot_2: movre %i0, -0x0C3, %i3
branch_4: brlez,pt %i1, .-0x138
failure:  add    %o0, 0x001, %o0 // add does not execute
```

The delay slot of a mispredicted Delayed Control Transfer Instruction (DCTI) may not be properly executed if the DCTI is last part of a dcti-couple or triple, or follows within several pipeline stages of a dcti-couple instruction pair.

Symptoms of the failure may include:

- The execution of a delay slot instruction from an older or a younger DCTI.
- Non-execution of the real delay slot.
- Skipping the instruction stream starting at a subsequent refetched instruction (e.g., a mispredicted or Jump and Link Instruction (JMPL) or return from subroutine (RET) delay slot, recirculating Load Integer Instruction (LD), following FLUSH or certain Write Privileged or Write State Register) or trapping instruction, and instead executing the instruction stream at PC=0x80 greater than the desired instruction.

Spurious increment of the PC by 0x80 may also occur in the value saved by RDPC, TPC or TnPC on trap, or the return address of the CALL or JMPL.

The following conditions are necessary for delay slot failure to occur:

- The DCTI instruction is mispredicted.



- The Delay slot that will not be handled properly reaches I-stage before its DCTI reaches E-stage (delay slot must be in same cache line as DCTI or hit in icache).
- There must be at least two older DCTI instructions in C-stage or earlier that are not JMPL/RET and not in delay slot of an earlier DCTI when mishandled delay slot instruction reaches I-stage.
- There must be an older dcti-couple in C-stage or earlier when the mispredicted DCTI reaches I-stage.
- Older unresolved DCTIs than the one with mishandled delay must all be predicted correctly.

The erroneous PC increment of 0x80 additionally requires:

- The mispredicted branch is actually not taken.
- The PC[6] of DCTIs delay slot and PC[6] of the next sequential instruction are both 0, and PC[6] of the falsely requeued delay slot is 1.

Impact:

A delay slot of a dcti-couple or a DCTI closely following a dcti-couple that reaches issue stage with older unresolved (C-stage or earlier) DCTIs may not be properly executed, and subsequent execution of either a refetched instruction or trapping instructions may result in skipping the original instruction stream and instead executing the instruction stream starting at PC=0x80 greater than the original instruction's PC.

Workaround:

Avoid DCTI couples, as per the V8 specification of unpredictable results from SPARC V8 manual: If the first instruction of a DCTI couple is a conditional branch, the targets of the DCTI are within the same address space as the DCTI couple, but are otherwise unpredictable. Given the relative rarity of dcti-couples, this problem is not viewed as particularly severe.

Status:

This bug will not be fixed in future releases of the silicon.



Erratum #20: A Read-after-Write address checking failure for load in the delay slot of a mispredicted Delayed Control Transfer Instruction (DCTI) can result in stale Data Cache (D Cache) data.

Applicability:

UltraSPARC-III Versions 2.3 and 2.4.

Description:

In one special circumstance, the read-after-write (RAW) checking fails for currently executing loads existing stores pending in the Store Queue, resulting in stale data being installed in the Data Cache (D Cache). From a program point of view, it will appear as if a store operation completed normally to External Cache (E Cache) and Memory, but did not update the D Cache.

This situation can only manifest itself when there is a load that has a RAW hazard in the delay slot of a Delayed Control Transfer Instruction (DCTI) (e.g., conditional branch) that is mispredicted. There is a distinction in RAW checking where some loads are bypassable and some are not. A read-after-write load that is bypassable can obtain the data directly from the Store Queue. A read-after-write load that is not bypassable must wait until the youngest store (there can be more than one) that touches the D Cache line in question has left the Store Queue. Here is an example sequence and the conditions that need to be met for this bug to occur.

```
ST A    << Hits in D Cache.
..
ST B    << Miss D Cache
..
..
BR X    << Must be mis-predicted
LD C    << Same 32 Byte line as B (Non-bypassable RAW)
LD A    << Line A invalidated in D Cache before this LD
        (Bypassable RAW)
..
X:
```

In this case of the branch being mispredicted taken, but actually taken, the execution pipe trace may look like this. When a branch is mispredicted, the delay slot is always cancelled and re-executed (unless it was annulled).

```
BR X    R E C M W X T D          < Mis-predicted
```



```

LD C      R E (c) (m) (w) (x) (t) (d)      < Cancelled in C
LD A      R E (c) (m) (w) (x) (t) (d)      < Cancelled in C
..
LD C      R E C M W X
          < Delay slot reqeud from MisPred Q

```

The conditions are as follows:

- Branch is mispredicted and LD C and LD A were cancelled due to a misprediction in the C Stage.
- LD A data is raw bypassable from the previous store; the store data has been retired into W Cache and is waiting to update D Cache; and the line A has been invalidated in D Cache.

Impact:

When the above conditions are met, the Store Queue fails to detect the RAW condition from ST B and fetches line C (same line as B) and installs it into the D Cache. The ST B in the Store Queue is not aware of this action since it was a D Cache miss when it was entered onto the Store Queue (and, if working properly, a store that misses the D Cache should never become a D Cache hit). When Store B exits the Store Queue, it only updates the data in W Cache, not the line in the D Cache.

The branch is mispredicted and not-taken.

```

0x5f16c554 : ld      [%i1 + 0xa74], %o3
0x5f16c558 : sth     %l6, [%g2 + 0x3c]
0x5f16c55c : st      %l3, [%g2 + 4]
A 0x5f16c560 : st      %f20, [%g2 + 0x14]
          <<< RAW bypassable Store (E).
B 0x5f16c564 : sth     %o5, [%i1 - 0x300]
C 0x5f16c568 : st      %f20, [%g2 + 0x2c] <<< RAW Hazard (D)
0x5f16c56c : addcc  %l1, %l0, %l6
0x5f16c570 : stb     %o4, [%g2 + 0x27] <<< RAW Hazard (D)
0x5f16c574 : addcc  %l4, %l0, %o1
0x5f16c578 : sth     %l7, [%g2 + 0x26] <<< RAW Hazard (D)
0x5f16c57c : st      %f20, [%g2 + 0x3c] <<< RAW Hazard (D)
Br 0x5f16c580 : be     0x5f16c588 <<< MISPRED,NOT-TAKEN
D 0x5f16c584 : ldsb   [%g2 + 0x37], %l7 <<< RAW check fail
E 0x5f16c588 : ld      [%g2 + 0x14], %f21 /
0x5f16c590 : andcc  %l1, %l4, %l3

```


**Workaround:**

Turn off the RAW bypass enable (DCU.RE) bit in D-Cache Control Register (ASI=0x45, VA=0x0). This situation will not occur if RAW bypassing is disabled.

Status:

This bug will not be fixed in future releases of the silicon.

Erratum #21: In Energy Star (E-star) 1/2 mode or full speed mode, the Jalapeno Bus (JBus) IO (JIO) ASIC is held off of the bus by a stream of blocks stores between CPUs, resulting in a PCI Bus Error From System Bus (BERR).**Applicability:**

UltraSPARC-IIIi Versions 2.3 and 2.4.

Description:

In the failing condition, CPU1 is doing block stores to CPU0's memory, and because the system is in 1/2 E-star mode where the Memory Controller Unit (MCU) services one memory request at a time, the data FIFOs fill up, causing DOK_off to assert on JBus after every store from CPU1. CPU1 is able to get back on JBus once CPU0 asserts DOK_on faster than JIO can, and thus send another store to CPU0, causing DOK_off to assert from CPU0. This live-lock situation continues, preventing JIO from arbitrating for JBus until the DMA request times out and causes a BERR.

In Version 3.2 of UltraSPARC-IIIi, a bit is added to the JBUS_CONFIG register that when programmed will cause the JBus Unit Data In Buffer (DIB) to have a low-water mark of 2 entries (as opposed to 3). This enables two agents to drive data packets to the CPU after DOK_on is asserted. There may be a slight performance impact with this bit enabled for certain applications.

The bit to enable for this feature is bit 60 of the JBUS_CONFIG register, named DOK_mode. Setting the bit to a 1 will enable the feature.

**Impact:**

In Energy Star (E-star) 1/2 mode, the Jbus IO ASIC (JIO) DMA does not make it to the Jalapeno Bus (JBus) for a very long time (i.e., greater than 5 ms), causing a PCI Bus Error From System Bus (BERR). This can also happen in full-speed mode with high SDRAM memory clock ratios.

Workaround:

None.

Status:

This bug will not be fixed in future releases of the silicon.

Erratum #22: In Energy Star (E-star) 1/2 mode, the Jalapeno Bus (JBus) IO ASIC (JIO) is held off of the JBus by a stream of software prefetches, causing PCI Bus Error From System Bus (BERR).**Applicability:**

UltraSPARC-III Versions 2.3, 2.4, and 3.2.

Description:

In this failure, prefetch reads are saturating the DRAM interface. The Memory Controller Unit (MCU) normally gives priority to reads over writes, unless writes are sufficiently queued up. The high watermark of the memory write queue (MWQ) is ignored, causing writes never to be drained. The DMA writes back up, and eventually cause the PCI bus error (BERR).

Impact:

In Energy Star (E-star) 1/2 mode, JIO DMA does not make it to Jalapeno Bus (JBus) for a very long time (i.e., greater than 5 ms), causing PCI timeout and Bus Error From System Bus (BERR).

Workaround:

Disabling software prefetches during E-star mode reduces the number of outstanding reads in the system and thus, avoids the read backup in the memory controller in E-star 1/2 mode.

**Status:**

This bug will not be fixed in future releases of the silicon.

Erratum #23: An Address Space Identifier (ASI) load returns corrupted data following an L2 ASI flush.**Applicability:**

UltraSPARC-IIIi Versions 2.3, 2.4, 3.2, and 3.4.

Description:

A flush request to ASI 0x4E may win arbitration more than once in the L2-cache controller due to an incorrect blocking condition in the arbiter. This occurs when a fill from the JBU lines up next to the ASI load or store in the L2-cache control. An immediately following load to 0x4E may return incorrect data.

Impact:

After an ASI 0x4E L2 flush, a following Address Space Identifier (ASI) load returns incorrect data.

Workaround:

Following L2 ASI flush routines, do not issue an ASI load immediately afterward. Alternatively, if an ASI load following the L2 flush routines is a requirement, issue consecutive ASI loads and use the data from the second load.

Status:

This bug will not be fixed in future releases of the silicon.