AMD Duron™ Processor Model 3 Revision Guide

Publication # **23865** Rev: **K**
Issue Date: **October 2003**

Revision History

Date	Rev	Description
October 2003	K	Revised erratum #24.
June 2003	J	Added erratum #24.
December 2002	I	Added errata #21–23.
July 2002	H	Added errata #20.
February 2002	G	Added errata #17.
April 2001	F	Removed OPN information. Added errata items 13, 14, and 16.

AMD Duron™ Processor Model 3 Revision Guide

The purpose of the *AMD Duron™ Processor Model 3 Revision Guide* is to communicate updated product information on the AMD Duron™ processor model 3 to designers of computer systems and software developers. This guide consists of three sections:

- **Product Errata:** This section, which starts on page 5, provides a detailed description of product errata, including potential effects on system operation and suggested workarounds. An erratum is defined as a deviation from the product's specification. A product errata may cause the behavior of the AMD Duron processor model 3 to deviate from the published specifications.
- **Revision Determination:** This section, which starts on page 20, shows the AMD Duron processor model 3 identification numbers returned by the CPUID instruction for each revision of the processor.
- **Technical and Documentation Support:** This section, which starts on page 21, provides a listing of available technical support resources. It also lists corrections, modifications, and clarifications to listed documents. Please refer to the data sheets listed in this section for product marking information.

Revision Guide Policy

Occasionally AMD identifies deviations from or changes to the specification of the AMD Duron processor model 3. These changes are documented in the *AMD Duron™ Processor Model 3 Revision Guide* as errata. Descriptions are written to assist system and software designers in using the AMD Duron processor model 3 and corrections to AMD's documentation on the AMD Duron processor model 3 are included. This release documents currently characterized product errata, but is not intended to document all errata that may be found in the processor.

1 Product Errata

This section documents AMD Duron processor model 3 product errata. The errata are divided into categories to assist referencing particular errata. A unique tracking number for each erratum has been assigned within this document for user convenience in tracking the errata within specific revision levels. Table 1 cross-references the revisions of the processor to each erratum. An "X" indicates that the erratum applies to the stepping. The absence of an "X" indicates that the erratum does not apply to the stepping.

Note: *There can be missing errata numbers. Errata that have been resolved from early revisions of the processor have been deleted, and errata that have been reconsidered may have been deleted or renumbered.*

Table 1. Cross-Reference of Product Revision to Errata

Errata Numbers and Description	Revision Numbers	
	A0	A2
3 CPUID Instruction Reports Incorrect L2 Cache Size	X	
5 MCA Bus Unit Control Register MSR 408H Returns Incorrect Information	X	X
10 Resistance Value of the ZN and ZP Pins	X	
11 PLL Overshoot on Wake-Up from Disconnect Causes Auto-Compensation Circuit to Fail	X	X
13 Instruction Execution Deadlock	X	X
14 Processors with Half-Frequency Multipliers May Hang Upon Wake-up from Disconnect	X	X
16 INVLPG Instruction Does Not Flush Entire Four-Megabyte Page Properly with Certain Linear Addresses	X	X
17 Code Modifications that Coincide with Level 2 Instruction TLB Translations May Escape Detection Resulting in Stale Code Execution	X	X
20 A Speculative SMC Store Followed by an Actual SMC Store May Cause One-Time Stale Execution	X	X
21 Real Mode RDPMC with Illegal ECX May Cause Unpredictable Operation	X	X
22 Using Task Gates With Breakpoints Enabled May Cause Unexpected Faults	X	X
23 Single Step Across I/O SMI Skips One Debug Trap	X	X
24 Software Prefetches May Report A Page Fault	X	X

3 CPUID Instruction Reports Incorrect L2 Cache Size

Products Affected. A0

Normal Specified Operation. The CPUID instruction should report a 64-Kbyte L2 Cache using extended function 8000_0006h.

Non-conformance. The CPUID instruction reports an incorrect L2 cache size as 1 Kbytes.

Potential Effect on System. System software that relies on the CPUID instruction for L2 cache information may not behave as expected.

Suggested Workaround. For Revision A0, system software should assume a 64-Kbyte L2 cache size. See Technical Note TN13, *AMD Duron™ Processor Model 3 Rev. A0: CPUID Reporting of L2 Cache Size*.

Resolution Status. Fixed in a future revision.

5 MCA Bus Unit Control Register MSR 408H Returns Incorrect Information

Products Affected. A0, A2

Normal Specified Operation. System reads to MSR 408h, MCA Bus Unit Control Register MC2_CTL, should return correct information—the lower 32 bits in EAX and all zeros for the upper 32 bits in EDX.

Non-conformance. A read to the Machine Check Architecture (MCA) Bus Unit Control MSR 408h (MC2_CTL) returns incorrect information in EDX. It returns the information stored in the upper 32 bits of the BU Status MSR 409h (MC2_STATUS) instead.

Potential Effect on System. If the system reads this MCA control register, the upper 32 bits, which are reported in EDX, can contain unexpected data.

Suggested Workaround. This register is only implemented as a 32-bit register. Ignore the upper 32 bits which are reported in EDX.

Resolution Status. No fix planned.

10 Resistance Value of the ZN and ZP Pins

Products Affected. A0

Normal Specified Operation. The ZN and ZP pins are specified such that the AMD system bus output drivers autocompensate to whatever resistance value is applied between ZN and VDD and ZP and VSS.

Non-conformance. The AMD system bus driver impedance is approximately 20 ohms higher than the applied resistor value.

Potential Effect on System. The AMD system bus signal quality for signals driven to the northbridge may be adversely affected.

Suggested Workaround. All motherboards should have ZN/ZP resistors set to approximately 20 ohms less than the characteristic board impedance.

Resolution Status. Fixed in future revisions.

Note: *When future revisions of the processor are used with motherboards implementing the workaround, the expected output impedance of the driver will continue to yield good signal quality. (For example, with a motherboard setting of 40 ohms for ZN and ZP, an old part that drives with a 60-ohm driver works properly. A revised part with a true 40-ohm driver also works properly.)*

11 PLL Overshoot on Wake-Up from Disconnect Causes Auto-Compensation Circuit to Fail

Products Affected. A0, A2

Normal Specified Operation. The AMD Duron processor model 3 PLL should return to the normal operating frequency when reconnecting to the system bus after a disconnect where the PLL was reduced to a lower operating frequency.

Non-conformance. The AMD Duron processor model 3 PLL can exceed the normal operating frequency when reconnecting to the system bus after a disconnect, causing a failure to maintain sufficient system bus I/O drive strength levels in the driver compensation circuit. The compensation circuit attempts to correct the drive strength, but if there is not sufficient time to perform this function, the system bus cannot operate properly.

Potential Effect on System. The system hangs.

Suggested Workaround. The event can be avoided through BIOS manipulation of the reconnect timing using the CLK_CTRL MSR. (See *AMD Athlon™ and AMD Duron™ Processor CLK_CTRL MSR Settings*, order# 24478, for exact values.)

The time for the PLL to overshoot can be greatly shortened to reduce it to a very small time period that does not enable the failure, and the time between the rampup and the reconnect to the system bus can be increased such that a failure in the compensation circuit has enough time to recover before reconnecting to the bus.

Resolution Status. Fixed in future revisions.

13 Instruction Execution Deadlock

Products Affected. A0, A2

Normal Specified Operation. Legitimate instruction sequences should execute as specified.

Non-conformance. Under rare and unlikely conditions, the load-store unit, instruction scheduler and effective address generation unit interact in such a way that a deadlock occurs, preventing further instruction execution.

Potential Effect on System. The system hangs. No instructions complete and the processor will not respond to interrupts.

Suggested Workaround. None.

Resolution Status. Fix planned for a future revision.

14 Processors with Half-Frequency Multipliers May Hang Upon Wake-up from Disconnect

Products Affected. A0, A2

Normal Specified Operation. The processor should reconnect to the system bus upon wake-up after a disconnect while in the C2 and C3 ACPI low-power states.

Non-conformance. The processor uses a special circuit to wake up from a low-power state and reconnect to the system bus when the nominal operating frequency is generated with a half-frequency multiplier. This circuit is rarely observed to glitch when coming out of the C2 and C3 low-power states.

Potential Effect on System. The system will hang.

Suggested Workaround. Do not use the C2 or C3 ACPI states on processors that run at a nominal operating frequency generated with a half-frequency multiplier. This can be accomplished by having the BIOS not declare C2 or C3 support to the operating system in the Fixed ACPI Description Table (FADT).

Resolution Status. Fix planned for a future revision.

16 INVLPG Instruction Does Not Flush Entire Four-Megabyte Page Properly with Certain Linear Addresses

Products Affected. A0, A2

Normal Specified Operation. After executing an INVLPG instruction the TLB should not contain any translations for any part of the page frame associated with the designated logical address.

Non-conformance. When the logical address designated by the INVLPG instruction is mapped by a 4-MB page mapping and LA[21] is equal to one, it is possible that the TLB will still retain translations after the instruction has finished executing.

Potential Effect on System. The residual data in the TLB can result in unexpected data access to stale or invalid pages of memory.

Suggested Workaround. When using the INVLPG instruction in association with a page that is mapped via a 4-MB page translation, always clear bit 21.

Resolution Status. Fix planned for a future revision.

17 Code Modifications that Coincide with Level 2 Instruction TLB Translations May Escape Detection Resulting in Stale Code Execution

Products Affected. A0, A2

Normal Specified Operation. Self-modifying code sequences should be correctly detected and handled in a manner that results in correct canonical results; stale code should not be executed.

Non-conformance. If the following events occur within a single clock cycle and the pipeline is not flushed by a coincidental event, then the processor will execute the stale code once instead of the newly generated code.

1. A write operation corresponding to the code self-modification has generated an instruction cache invalidation
2. The processor is fetching the line that is being invalidated
3. The fetch misses in the Level 1 instruction TLB
4. The fetch hits in the Level 2 instruction TLB

In this scenario the processor will properly invalidate the instruction cache line but will not mark the appropriate bit within the instruction buffer used to track instructions within the pipeline. This causes the pipeline to not be flushed and leads to execution of invalid instructions. Subsequent attempts to execute the overwritten instruction will miss in the instruction cache and the correct instruction data will be fetched from the L2 and subsequently executed.

Potential Effect on System. Stale code will be executed resulting in unpredictable system behavior.

Suggested Workaround. Consult with your platform vendor for a BIOS that works around this erratum.

Resolution Status. No fix planned.

20 A Speculative SMC Store Followed by an Actual SMC Store May Cause One-Time Stale Execution

Products Affected. A0, A2

Normal Specified Operation. Self-modifying code sequences should be correctly detected and handled in a manner consistent with canonical results; stale code should not be executed.

Non-conformance. The following scenario can result in a one-time execution of stale instructions:

1. A speculative store instruction initiates a request (R) to modify a 64-byte cache line with address A, which currently resides within the L1 instruction cache.
2. The speculative store instruction is ultimately not executed because of a branch misprediction. However, the store R is still in flight attempting to bring the line into the data cache in the modified state.
3. The instruction cache, which fetches instructions 16 bytes at a time, is redirected by the branch into the cache line with address A and fetches a portion of the line into the instruction buffer.
4. R then invalidates the instruction cache line with address A and brings the line into the L1 data cache, marking it as modified. However, the instruction buffer, which also contains some bytes from address A, is not invalidated.
5. The instruction fetch mechanism attempts to read the next 16-byte chunk of code and must issue a request to bring the 64-byte line back into the instruction cache.
6. This instruction cache request for address A hits on the modified line now in the L1 cache, and evicts it from the data cache to the L2.
7. A second store instruction (S) from the instruction buffer is issued into the execution units. S is a self-modifying code reference to another instruction that currently exists in the 64-byte cache block at address A and is also in the instruction buffer.
8. The execution of S detects that an instruction request to fetch address A is in flight. However, the store request is given priority. Since it now hits in the L2 and the L2 state is modified, it assumes that the line cannot be in the instruction cache or the instruction buffer.

Potential Effect on System. The processor will execute stale code instructions.

Suggested Workaround. None. This failure has only been observed in internally generated synthetic code.

Resolution Status. No fix planned.

21 Real Mode RDPMC with Illegal ECX May Cause Unpredictable Operation

Products Affected. A0, A2

Normal Specified Operation. Illegal values of ECX (that is, ECX>3) for the RDPMC (Read Performance Monitor Counter) instruction cause the processor to take a general protection exception.

Non-conformance. If the RDPMC is executed in real mode with a specific illegal value of ECX=4, then the processor may incorrectly enter the GP fault handler as if it were in 32-bit real mode.

Potential Effect on System. Incorrect instruction decode leading to unpredictable system failure.

Suggested Workaround. When in real mode, restrict use of the RDPMC instruction to legal counter values (0-3). This circumstance is not expected to occur in normal operation and has only been detected in a simulation environment.

Resolution Status. No fix planned.

22 Using Task Gates With Breakpoints Enabled May Cause Unexpected Faults

Products Affected. A0, A2

Normal Specified Operation. Task gates should correctly use the TSS selector out of the task gate for CALL and JMP instructions.

Non-conformance. When a task gate is used by a CALL or JMP instruction and any debug breakpoint is enabled through the DR7.LE or GE bits, the processor may, under certain timing scenarios, incorrectly use the new TSS base[15:0] contained in the new TSS as a selector. This will most likely cause a GP fault with an error code of the new TSS base.

Potential Effect on System. System failure.

Suggested Workaround. When using software that uses task gates with CALL or JMP instructions, do not enable breakpoints.

Resolution Status. No fix planned.

23 Single Step Across I/O SMI Skips One Debug Trap

Products Affected. A0, A2

Normal Specified Operation. When single stepping (with EFLAGS.TF) across an IN or OUT instruction that detects an SMI, the processor correctly defers taking the debug trap and instead enters SMM. Upon RSM (without I/O restart), the processor should immediately enter the debug trap handler.

Non-conformance. Under this scenario, the processor does not enter the debug trap handler but instead returns to the instruction following the I/O instruction.

Potential Effect on System. When using the single step debug mode, following an I/O operation that detects an SMI, one instruction may appear to be skipped.

Suggested Workaround. None required as this is a debug limitation only. If a workaround is desired, modify the SMM handler to detect this case and enter the debug handler directly.

Resolution Status. No fix planned.

24 Software Prefetches May Report A Page Fault

Products Affected. A0, A2

Normal Specified Operation. Software prefetches should not report page faults if they encounter them.

Non-conformance. Software prefetch instructions are defined to ignore page faults. Under highly specific and detailed internal circumstances, a prefetch instruction may report a page fault if both of the following conditions are true:

- The target address of the prefetch would cause a page fault if the address was accessed by an actual memory load or store instruction under the current privilege mode;
- The prefetch instruction is followed in execution-order by an actual or speculative byte-sized memory access of the same modify-intent to the same address.

PREFETCH and PREFETCHNTA/0/1/2 have the same modify-intent as a memory load access. PREFETCHW has the same modify-intent as a memory store access.

The page fault exception error code bits for the faulting prefetch will be identical to that for a byte-sized memory access of the same-modify intent to the same address.

Note that some misaligned accesses can be broken up by the processor into multiple accesses where at least one of the accesses is a byte-sized access.

If the target address of the subsequent memory access of the same modify-intent is aligned and not byte-sized, this errata does not occur and no workaround is needed.

Potential Effect on System. An unexpected page fault may occur infrequently on a prefetch instruction.

Suggested Workaround. Two workarounds are described for this erratum.

Kernel Workaround

The Operating System kernel can work around the erratum by allowing the page fault handler to satisfy the page fault to an "accessible" page regardless of whether the fault was due to a load, store, or prefetch operation. If the faulting instruction is permitted access to the page, return to it as usual. (An "accessible" page is one for which memory accesses are allowed under the current privilege mode once the page is resident in memory).

If the faulting instruction is trying to access an "inaccessible" page, scan the instruction stream bytes at the faulting Instruction Pointer to determine if the instruction is a prefetch. (An "inaccessible" page is one for which memory accesses are not allowed under the current privilege mode.) If the faulting instruction is a prefetch instruction, simply return back to it; the internal hardware conditions that caused the prefetch to fault should be removed and operation should continue normally. If it is not a prefetch instruction, generate the appropriate memory access control violation as appropriate. The performance impact of doing the scan is small because the actual errata is infrequent and does not produce an excessive number of page faults that affect system performance.

General Workaround

If the page-fault handler for a kernel can be patched as described in the preceding kernel workaround, no further action by software is required. The following general workarounds should only be considered for kernels where the page-fault handler can not be patched and a prefetch instruction could end up targeting an address in an "inaccessible" page.

Because the actual errata is infrequent, it does not produce an excessive number of page faults that affect system performance. Therefore a page fault from a prefetch instruction for an address within an "accessible" page does not require any general workaround.

Software can minimize the occurrence of the errata by issuing only one prefetch instruction per cache-line (a naturally-aligned 64-byte quantity) and ensuring one of the following:

- In many cases, if a particular target address of a prefetch is known to encounter this errata, simply change the prefetch to target the next byte.
- Avoid prefetching inaccessible memory locations, when possible.
- In the general case, ensure that the address used by the prefetch is offset into the middle of an aligned quadword near the end of the cache-line. For example, if the address desired to be prefetched is "ADDR", use an offset of 0x33 to compute the address used by the actual prefetch instruction as: "(ADDR & ~0x3f) + 0x33".

Resolution Status. No fix planned.

2 Revision Determination

Table 2 shows the AMD Duron processor model 3 identification number returned by the CPUID instruction for each revision of the processor.

Table 2. CPUID Values for the Revisions of the AMD Duron™ Processor Model 3

Revision	CPUID
A0	630
A2	631

3 Technical and Documentation Support

The following documents provide additional information regarding the operation of the AMD Duron processor model 3. Please refer to the data sheets listed in this section for product marking information.

- *AMD Duron™ Processor Model 3 Data Sheet*, order# 23802
- *AMD-751™ System Controller Data Sheet*, order# 21910
- *AMD-756™ Peripheral Bus Controller Data Sheet*, order# 22548

For the latest updates, refer to www.amd.com and download the appropriate files.