Revision Guide for AMD Family 15h Models 70h-7Fh Processors

Publication # **55370**	Revision: **3.00**
Issue Date: **July 2016**	

Advanced Micro Devices

List of Figures

List of Tables

Revision History

July 2016	3.00	Initial public release.

Overview

The purpose of the *Revision Guide for AMD Family 15h Models 70h-7Fh Processors* is to communicate updated product information to designers of computer systems and software developers. This revision guide includes information on the following products:

- AMD A-Series APU with Radeon™ Graphics
- AMD E-Series APU with Radeon™ Graphics
- AMD G-Series APU with Radeon™ Graphics

Feature support varies by brands and OPNs. To determine the features supported by your processor, contact your customer representative.

This guide consists of these major sections:

- Processor Identification shows how to determine the processor revision and workaround requirements, and to construct, program, and display the processor name string.
- Product Errata 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, and as such may cause the behavior of the processor to deviate from the published specifications.
- Documentation Support provides a listing of available technical support resources.

Revision Guide Policy

Occasionally, AMD identifies product errata that cause the processor to deviate from published specifications. Descriptions of identified product errata are designed to assist system and software designers in using the processors described in this revision guide. This revision guide may be updated periodically.

Conventions

Numbering

- **Binary numbers**. Binary numbers are indicated by appending a "b" at the end, e.g., 0110b.
- **Decimal numbers**. Unless specified otherwise, all numbers are decimal. This rule does not apply to the register mnemonics.
- **Hexadecimal numbers**. Hexadecimal numbers are indicated by appending an "h" to the end, e.g., 45F8h.
- **Underscores in numbers**. Underscores are used to break up numbers to make them more readable. They do not imply any operation. e.g., 0110_1100b.
- **Undefined digit**. An undefined digit, in any radix, is notated as a lower case "x".

Register References and Mnemonics

In order to define errata workarounds it is sometimes necessary to reference processor registers. References to registers in this document use a mnemonic notation consistent with that defined in the *BIOS and Kernel Developer's Guide (BKDG) for AMD Family 15h Models 70h-7Fh Processors,* order# 55072. Each mnemonic is a concatenation of the register-space indicator and the offset of the register. The mnemonics for the various register spaces are as follows:

- IOXXX: x86-defined input and output address space registers; XXX specifies the byte address of the I/O register in hex (this may be 2 or 3 digits). This space includes the I/O-Space Configuration Address Register (IOCF8) and the I/O-Space Configuration Data Port (IOCFC) to access configuration registers.
- DZFYxXXX: PCI-defined configuration space at bus 0; Z specifies the PCI device address in hex; XXX specifies the byte address of the configuration register (this may be 2 or 3 digits) in hex; Y specifies the function number. For example, D18F3x40 specifies the register at bus 0, device 18h, function 3, address 40h. Some registers in D18F2xXXX have a _dct[1:0] mnemonic suffix, which indicates there is one instance per DRAM controller (DCT). The DCT instance is selected by DCT Configuration Select[DctCfgSel] (D18F1x10C[0]).
- DZFYxXXX_xZZZZZ: Port access through the PCI-defined configuration space at bus 0; Z specifies the PCI device address in hex; XXX specifies the byte address of the data port configuration register (this may be 2 or 3 digits) in hex; Y specifies the function number; ZZZZZ specifies the port address (this may be 2 to 7 digits) in hex. For example, D18F2x9C_x1C specifies the port 1Ch register accessed using the data port register at bus 0, device 18h, function 2, address 9Ch. Refer to the *BKDG* for access properties. Some registers in D18F2xXXX_xZZZZZ have a _dct[1:0] mnemonic suffix, which indicates there is one instance per DRAM controller (DCT). The DCT instance is selected by DCT Configuration Select[DctCfgSel] (D18F1x10C[0]).
- APICXXX: APIC memory-mapped registers; XXX is the byte address offset from the base address in hex (this may be 2 or 3 digits). The base address for this space is specified by the APIC Base Address Register (APIC_BAR) at MSR0000_001B.
- CPUID FnXXXX_XXXX_RRR_xYYY: processor capability information returned by the CPUID instruction where the CPUID function is XXXX_XXXX (in hex) and the ECX input is YYY (if specified). When a register is specified by RRR, the reference is to the data returned in that register. For example, CPUID Fn8000_0001_EAX refers to the data in the EAX register after executing CPUID instruction function 8000_0001h.
- MSRXXXX_XXXX: model specific registers; XXXX_XXXX is the MSR number in hex. This space is accessed through x86-defined RDMSR and WRMSR instructions.
- PMCxXXX[Y]: performance monitor events; XXX is the hexadecimal event counter number programmed into MSRC001_020[A,8,6,4,2,0][EventSelect] (PERF_CTL[5:0] bits 7:0). Y, when specified, signifies the unit mask programmed into MSRC001_020[A,8,6,4,2,0][UnitMask] (PERF_CTL[5:0] bits 15:8).

- NBPMCxXXX[Y]: northbridge performance monitor events; XXX is the hexadecimal event counter number programmed into MSRC001_024[6,4,2,0][EventSelect] (NB_PERF_CTL[3:0] bits 7:0). Y, when specified, signifies the unit mask programmed into MSRC001_024[6,4,2,0][UnitMask] (NB_PERF_CTL[3:0] bits 15:8).

Many register references use the notation "[]" to identify a range of registers. For example, D18F2x[1,0][4C:40] is a shorthand notation for D18F2x40, D18F2x44, D18F2x48, D18F2x4C, D18F2x140, D18F2x144, D18F2x148, and D18F2x14C.

Arithmetic and Logical Operators

In this document, formulas follow some Verilog conventions as shown in Table 1.

Table 1. Arithmetic and Logic Operators

Operator	Definition
{}	Curly brackets are used to indicate a group of bits that are concatenated together. Each set of bits is separated by a comma. E.g., {Addr[3:2], Xlate[3:0]} represents a 6-bit value; the two MSBs are Addr[3:2] and the four LSBs are Xlate[3:0].
|	Bitwise OR operator. E.g. (01b | 10b == 11b).
||	Logical OR operator. E.g. (01b || 10b == 1b); logical treats multibit operand as 1 if >=1 and produces a 1-bit result.
&	Bitwise AND operator. E.g. (01b & 10b == 00b).
&&	Logical AND operator. E.g. (01b && 10b == 1b); logical treats multibit operand as 1 if >=1 and produces a 1-bit result.
^	Bitwise exclusive-OR operator; sometimes used as "raised to the power of" as well, as indicated by the context in which it is used. E.g. (01b ^ 10b == 11b). E.g. (2^2 == 4).
~	Bitwise NOT operator (also known as one's complement). E.g. (~10b == 01b).
!	Logical NOT operator. E.g. (!10b == 0b); logical treats multibit operand as 1 if >=1 and produces a 1-bit result.
==	Logical "is equal to" operator.
!=	Logical "is not equal to" operator.
<=	Less than or equal operator.
>=	Greater than or equal operator.
*	Arithmetic multiplication operator.
/	Arithmetic division operator.
<<	Shift left first operand by the number of bits specified by the 2nd operand. E.g. (01b << 01b == 10b).
>>	Shift right first operand by the number of bits specified by the 2nd operand. E.g. (10b >> 01b == 01b).

Processor Identification

This section shows how to determine the processor revision.

Revision Determination

A processor revision is identified using a unique value that is returned in the EAX register after executing the CPUID instruction function 0000_0001h (CPUID Fn0000_0001_EAX). Figure 1 shows the format of the value from CPUID Fn0000_0001_EAX. In some cases, two or more processor revisions may exist within a stepping of a processor family and are identified by a unique value in D0F0xBC_xD823_1008 SMUSBI Errata Status Register (see D0F0xBC_xD823_1008 SMUSBI Errata Status Register).

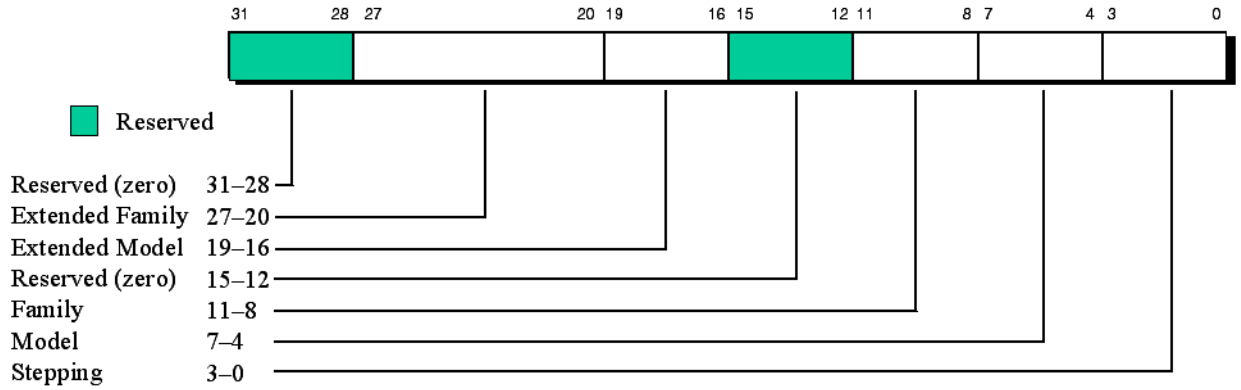

Figure 1. Format of CPUID Fn0000_0001_EAX

The following tables show the identification numbers from CPUID Fn0000_0001_EAX and D0F0xBC_xD823_1008 SMUSBI (if necessary) for each revision of the processor to each processor segment. "X" signifies that the revision has been used in the processor segment. "N/A" signifies that the revision has not been used in the processor segment.

D0F0xBC_xD823_1008 SMUSBI Errata Status Register

Communicating the status of an erratum within a stepping of a processor family is necessary in certain circumstances. D0F0xBC_xD823_1008 is used to communicate the status of such an erratum fix so that BIOS or system software can determine the necessity of applying the workaround. Under these circumstances, the erratum workaround references the specified bit to enable software to test for the presence of the erratum. The erratum may be specific to some steppings of the processor, and the specified bit may or may not be set on other unaffected revisions within the same family. Therefore, software should use the CPUID Fn00000_0001_EAX extended model, model, and stepping as the first criteria to identify the applicability of an erratum. Once defined, the definition of the status bit will persist within the family of processors.

Bits	Description
31:0	0000_0000h. Reserved.

Graphic Device IDs

Processors with an integrated AMD Radeon HD Graphics Processing Engine use a graphics device ID at D1F0x00[31:16] to further identify the processor.

Programming and Displaying the Processor Name String

This section, intended for BIOS programmers, describes how to program and display the 48-character processor name string that is returned by CPUID Fn8000_000[4:2]. The hardware or cold reset value of the processor name string is 48 ASCII NUL characters, so the BIOS must program the processor name string before any general purpose application or operating system software uses the extended functions that read the name string. It is common practice for the BIOS to display the processor name string and model number whenever it displays processor information during boot up.

Note: *Motherboards that do not program the proper processor name string and model number will not pass AMD validation and will not be posted on the AMD Recommended Motherboard Web site.*

The name string must be ASCII NUL terminated and the 48-character maximum includes that NUL character.

The processor name string is programmed by MSR writes to the six MSR addresses covered by the range MSRC001_00[35:30]h. Refer to the for the format of how the 48-character processor name string maps to the 48 bytes contained in the six 64-bit registers of MSRC001_00[35:30].

The processor name string is read by CPUID reads to a range of CPUID functions covered by CPUID Fn8000_000[4:2]. Refer to CPUID Fn8000_000[4:2] in the for the 48-character processor name string mapping to the 48 bytes contained in the twelve 32-bit registers of CPUID Fn8000_000[4:2].

Constructing the Processor Name String

This section describes how to construct the processor name string. BIOS forms the name string as follows:

1. If D18F5x198_x0 is 00000000h, then use a name string of "AMD Unprogrammed Engineering Sample" and skip the remaining steps.
2. Read {D18F5x198_x1, D18F5x198_x0} and write this value to MSRC001_0030.
3. Read {D18F5x198_x3, D18F5x198_x2} and write this value to MSRC001_0031.
4. Read {D18F5x198_x5, D18F5x198_x4} and write this value to MSRC001_0032.
5. Read {D18F5x198_x7, D18F5x198_x6} and write this value to MSRC001_0033.
6. Read {D18F5x198_x9, D18F5x198_x8} and write this value to MSRC001_0034.
7. Read {D18F5x198_xB, D18F5x198_xA} and write this value to MSRC001_0035.

Operating System Visible Workarounds

This section describes how to identify operating system visible workarounds.

MSRC001_0140 OS Visible Work-around MSR0 (OSVW_ID_Length)

This register, as defined in *AMD64 Architecture Programmer's Manual Volume 2: System Programming*, order# 24593, is used to specify the number of valid status bits within the OS Visible Work-around status registers.

The reset default value of this register is 0000_0000_0000_0000h.

BIOS shall program the OSVW_ID_Length to 0005h prior to hand-off to the OS.

Bits	Description
63:16	Reserved.
15:0	**OSVW_ID_Length: OS visible work-around ID length**. Read-write.

MSRC001_0141 OS Visible Work-around MSR1 (OSVW_Status)

This register, as defined in *AMD64 Architecture Programmer's Manual Volume 2: System Programming*, order# 24593, provides the status of the known OS visible errata. Known errata are assigned an OSVW_ID corresponding to the bit position within the valid status field.

Operating system software should use MSRC001_0140 to determine the valid length of the bit status field. For all valid status bits: 1=Hardware contains the erratum, and an OS software work-around is required or may be applied instead of a BIOS workaround. 0=Hardware has corrected the erratum, so an OS software work-around is not necessary.

The reset default value of this register is 0000_0000_0000_0000h.

Bits	Description
63:5	**OsvwStatusBits:** Reserved. OS visible work-around status bits. Read-write.
3	**OsvwId3:** Reserved, must be zero.
2	**OsvwId2:** Reserved, must be zero.
1	**OsvwId1:** Reserved, must be zero.
0	**OsvwId0:** Reserved, must be zero.

BIOS shall program the state of the valid status bits as shown in Table 2 prior to hand-off to the OS.

Table 2. Cross Reference of Product Revision to OSVW ID

CPUID Fn0000_0001_EAX (Mnemonic)	MSRC001_0141 Bits
00670F00h (ST-A0)	0000_0000_0000_0000h

Product Errata

This section documents product errata for the processors. A unique tracking number for each erratum has been assigned within this document for user convenience in tracking the errata within specific revision levels. This table cross-references the revisions of the part to each erratum. "No fix planned" indicates that no fix is planned for current or future revisions of the processor.

Note: There may be missing errata numbers. Errata that do not affect this product family do not appear. In addition, 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 3. Cross-Reference of Processor Revision to Errata

No.	Errata Description	CPUID Fn0000_0001_EAX
		0670F00h (ST-A0)
732	IOMMU Event Log Ordering Violation	No fix planned
733	IOMMU PPR Log Ordering Violation	No fix planned
800	IOMMU IO_PAGE_FAULT Events Are Not Correctly Suppressed When DTE.SA=1 and PTE.PR=0	No fix planned
801	IOMMU IO_PAGE_FAULT Event May Be Logged Instead of INVALID_DEVICE_REQUEST Event	No fix planned
813	D-bit May Not Be Set By IOMMU For Untranslated Guest PCIe® Atomic Requests	No fix planned
814	IOMMU Event Not Flagged when Guest PTE Reserved Bits Are Not Zero	No fix planned
870	IOMMU Always Masks PASID TLP Prefix On PPR Auto Response Regardless Of PRG Response PASID Required Register Bit Setting On PRI Endpoint Devices	No fix planned
902	LPC_SMI_L/AGPIO86 Pin On FP4 Package/Socket Does Not Support LPC SMI Function	No fix planned
903	IOMMU Mishandles Invalid COMPLETE_PPR_REQUEST Command	No fix planned
911	IOMMU Unnecessarily Updates Dirty Bit (D-bit) While Handling Non-supervisor DMA Write Request To Writable Supervisor-only Page	No fix planned
937	Unpredictable IOMMU IO_PAGE_FAULT Event Logging For PCIe® Atomic Requests To Protected Pages	No fix planned
965	Incorrect IOMMU IO_PAGE_FAULT Event Logging For Reserved Message Type Interrupt Requests With DTE.IG = 1	No fix planned

Cross-Reference of Errata to Package Type

This table cross-references the errata to each package type. "X" signifies that the erratum applies to the package type. An empty cell signifies that the erratum does not apply. An erratum may not apply to a package type due to a specific characteristic of the erratum, or it may be due to the affected silicon revision(s) not being used in this package.

Table 4. Cross-Reference of Errata to Package Type

Errata	Package	
	FP4	FT4
732	X	X
733	X	X
800	X	X
801	X	X
813	X	X
814	X	X
870	X	X
882	X	X
902	X	
903	X	X
911	X	X
937	X	X
965	X	X

Cross-Reference of Errata to Processor Segments

This table cross-references the errata to each processor segment. "X" signifies that the erratum applies to the processor segment. An empty cell signifies that the erratum does not apply. An erratum may not apply to a processor segment due to a specific characteristic of the erratum, or it may be due to the affected silicon revision(s) not being used in this processor segment.

Table 5. Cross-Reference of Errata to Processor Segments

Errata	Processor Segment		
	AMD A-Series APU	AMD E-Series APU	AMD G-Series APU
732	X	X	X
733	X	X	X
800	X	X	X
801	X	X	X
813	X	X	X
814	X	X	X
870	X	X	X
882	X	X	X
902	X	X	X
903	X	X	X
911	X	X	X
937	X	X	X
965	X	X	X

732 IOMMU Event Log Ordering Violation

Description

The processor IOMMU does not maintain producer-consumer ordering between the IOMMU event log DMA writes and IOMMU MMIO register read completions. The processor core may read stale or uninitialized event logs from memory when a read response from the event log tail pointer register passes the corresponding event log DMA write. A series or burst of event log DMA writes would normally be necessary for this ordering violation to be observed.

Potential Effect on System

Software may process an event log before it has been completely written, possibly resulting in the operating system or hypervisor taking improper corrective actions.

Suggested Workaround

The IOMMU driver of the hypervisor or operating system should initialize the event log buffer to all zeros and write event log entries to zero after they are processed. If software subsequently observes an all zero event log entry, it should re-read the buffer until a non-zero event log is returned. It is recommended that software detects that the log buffer has not been written by checking for an EventCode (bits 63:60) that is equal to 0000b.

Fix Planned

No fix planned

733 IOMMU PPR Log Ordering Violation

Description

The processor IOMMU does not maintain producer-consumer ordering between the IOMMU peripheral page service request (PPR) log DMA writes and IOMMU MMIO register read completions. The processor core may read stale or uninitialized PPR logs from memory when a read response from the PPR log tail pointer register passes the corresponding PPR log DMA write. A series or burst of PPR log DMA writes would normally be necessary for this ordering violation to be observed.

This erratum only applies in systems where a device is performing Address Translation Service (ATS) requests.

Potential Effect on System

Software may process a PPR log before it has been completely written, possibly resulting in the IOMMU software not properly processing a page service request. This may result in unpredictable IOMMU behavior.

Suggested Workaround

The IOMMU driver of the hypervisor or operating system should initialize the PPR log buffer to all zeros and write PPR log entries to zero after they are processed. If software subsequently observes an all zero PPR log entry, it should re-read the buffer until a non-zero PPR log is returned. It is recommended that software detects that the log buffer has not been written by checking for a PPRCode (bits 63:60) that is equal to 0000b.

Fix Planned

No fix planned

800 IOMMU IO_PAGE_FAULT Events Are Not Correctly Suppressed When DTE.SA=1 and PTE.PR=0

Description

IO_PAGE_FAULT events may incorrectly be logged when DTE.SA=1 and PTE.PR=0.

Potential Effect on System

This behavior has been observed in simulation. To date, no detrimental effects from this behavior have been observed in system.

Suggested Workaround

IOMMU driver software may need to ignore extraneous IO_PAGE_FAULT event logs.

Fix Planned

No fix planned

801 IOMMU IO_PAGE_FAULT Event May Be Logged Instead of INVALID_DEVICE_REQUEST Event

Description

An IO_PAGE_FAULT event may be logged instead of an INVALID_DEVICE_REQUEST event for untranslated guest requests to the IOMMU with AT=0 and PASID TLP prefix.

Potential Effect on System

This behavior has been observed in simulation. To date, no detrimental system impact has been observed.

Suggested Workaround

IOMMU driver software may need to treat IO_PAGE_FAULT events as INVALID_DEVICE_REQUEST events.

Fix Planned

No fix planned

813 D-bit May Not Be Set By IOMMU For Untranslated Guest PCIe® Atomic Requests

Description

Under specific and detailed pipeline conditions, the IOMMU may not set the D-bit properly when handling untranslated guest PCIe® atomic requests.

Potential Effect on System

If a page that should be marked dirty is paged out without IOMMU setting the D-bit, software will incorrectly revert back to the stored page, resulting in unpredictable system behavior.

This behavior has been observed in simulation. To date, no detrimental effects from this behavior have been observed in system. No PCIe device that will issue untranslated guest PCIe atomic requests have been encountered.

Suggested Workaround

PCIe devices should be programmed not to issue untranslated guest PCIe atomic requests.

Fix Planned

No fix planned

814 IOMMU Event Not Flagged when Guest PTE Reserved Bits Are Not Zero

Description

An IOMMU event is not flagged when a translation table walk encounters a guest page table entry (PTE) with non-zero values programmed for MBZ reserved bits 20:13 for a 2 MByte PTE, or bits 30:13 for a 1 GByte PTE. The reserved bits are incorrectly ignored by the IOMMU.

Potential Effect on System

None expected in the absence of IOMMU software programming errors.

This behavior has been observed in simulation. To date, no detrimental effects from this behavior have been observed in system.

Suggested Workaround

None.

Fix Planned

No fix planned

870 IOMMU Always Masks PASID TLP Prefix On PPR Auto Response Regardless Of PRG Response PASID Required Register Bit Setting On PRI Endpoint Devices

Description

If the IOMMU PPR Auto Response feature is enabled and PPR Log overflow occurs, IOMMU generates PPR responses automatically. However, IOMMU always masks the PASID TLP Prefix in the PPR auto response regardless of the PRG Response PASID Required bit setting by the PRI endpoint devices; it fails to generate PPR response with PASID TLP Prefix even if the PRI request contains a valid PASID TLP Prefix and the originating endpoint device has the PRG Response PASID Required bit set.

Potential Effect on System

None expected in the absence of PPR Log overflow events. If PRI endpoint devices stay within their quota and PPRLOG is sized appropriately, PPR Log overflow should not happen.

In the unlikely event of PPR Log overflow, PRI endpoint devices expecting the PASID TLP Prefix in the PPR auto response and not getting it may cause unpredictable system behavior.

Suggested Workaround

IOMMU PPR Auto Response feature should not be used on systems with PRI endpoint devices with PRG Response PASID Required bit = 1 that do not initiate auto retry on incomplete responses.

Fix Planned

No fix planned

882 Down-cored Processor May Experience Incorrect Machine Check Error (MCE) Reporting via SB-RMI

Description

Dual core or software down-cored processors may experience incorrect logical core number mapping. This may lead to erroneous MCE status reporting.

Potential Effect on System

MCE alert event may be incorrectly reported via SB-RMI interface.

Suggested Workaround

SB-RMI should not be used on systems with dual core processors or four core processors that may be software down-cored.

Fix Planned

902 LPC_SMI_L/AGPIO86 Pin On FP4 Package/Socket Does Not Support LPC SMI Function

Description

LPC SMI function is not supported by the processor LPC_SMI_L/AGPIO86 pin on FP4 package or socket.

Potential Effect on System

LPC SMI device will not function if connected to LPC_SMI_L/AGPIO86 pin.

Suggested Workaround

Connect LPC device SMI pin to any one of the pins according to the information in the section named "Enabling LPC_SMI Function" in the BIOS and Kernel Developers Guide (BKDG) for AMD Family 15h Models 60h-6Fh, PID # 50742, revision 3.01 or newer.Connect LPC device SMI pin to any one of the pins according to the information in the section named "Enabling LPC_SMI Function" in the BIOS and Kernel Developers Guide (BKDG) for AMD Family 15h Models 70h-7Fh, PID # 55072, revision 1.01 or newer.

Fix Planned

No fix planned

903 IOMMU Mishandles Invalid COMPLETE_PPR_REQUEST Command

Description

When IOMMU encounters an invalid COMPLETE_PPR_REQUEST command, it will log the Invalid PPR Request event correctly with RX=1; however, it will not fetch new commands even after software recovers from the INVALID_PPR_REQUEST event and re-enables the command buffer. Invalid COMPLETE_PPR_REQUEST command can be caused by the following conditions:

- out of range DeviceID value,
- out of range PASID value,
- GN=1 when guest translation is not enabled.

Potential Effect on System

Unpredictable system behavior.

Suggested Workaround

Software should not issue an invalid COMPLETE_PPR_REQUEST command.

Fix Planned

No fix planned

911 IOMMU Unnecessarily Updates Dirty Bit (D-bit) While Handling Non-supervisor DMA Write Request To Writable Supervisor-only Page

Description

IOMMU incorrectly sets the D-bit in the guest page table when it encounters a DMA write request without supervisor privilege to a writable supervisor-only page.

Potential Effect on System

No functional issue is expected; the non-permitted DMA write request is aborted without any memory content modification. However, the affected pages may be unnecessarily written out to the pagefile by software.

Suggested Workaround

None

Fix Planned

No fix planned

937 Unpredictable IOMMU IO_PAGE_FAULT Event Logging For PCIe® Atomic Requests To Protected Pages

Description

IOMMU has unpredictable IO_PAGE_FAULT event logging when it encounters a PCIe® atomic request accessing protected pages.

The following might occur when IOMMU encounters PCIe atomic requests accessing protected pages;

- when DTE.SA = 1, IO_PAGE_FAULT might be logged when it should be suppressed,
- when DTE.SA = 0, IO_PAGE_FAULT might be suppressed incorrectly.

Potential Effect on System

Unpredictable event logging behavior. PCIe atomic requests to protected pages are aborted as expected.

Suggested Workaround

None.

Fix Planned

No fix planned

965 Incorrect IOMMU IO_PAGE_FAULT Event Logging For Reserved Message Type Interrupt Requests With DTE.IG = 1

Description

IOMMU will log an IO_PAGE_FAULT event when it encounters any reserved message type interrupt request regardless of DTE.IG setting. Even when DTE.IG = 1, IO_PAGE_FAULT event is logged when it should be suppressed.

Potential Effect on System

Software may encounter unexpected IO_PAGE_FAULT event logging. Reserved message type interrupt requests are aborted as expected.

Suggested Workaround

Software may ignore IO_PAGE_FAULT event log entries for reserved message type interrupt requests from devices with DTE.IG = 1.

Fix Planned

No fix planned

Documentation Support

The following documents provide additional information regarding the operation of the processor:

- *AMD64 Architecture Programmer's Manual Volume 1: Application Programming*, order# 24592
- *AMD64 Architecture Programmer's Manual Volume 2: System Programming*, order# 24593
- *AMD64 Architecture Programmer's Manual Volume 3: General-Purpose and System Instructions*, order# 24594
- *AMD64 Architecture Programmer's Manual Volume 4: 128-Bit and 256-Bit Media Instructions*, order# 26568
- *AMD64 Architecture Programmer's Manual Volume 5: 64-Bit Media and x87 Floating-Point Instructions*, order# 26569
- *BIOS and Kernel Developer's Guide for AMD Family 15h Models 70h-7Fh Processors*, order# 55072.

See the AMD Web site at www.amd.com for the latest updates to documents.