BIOS and Kernel Developer's Guide for AMD NPT Family 0Fh Processors

Publication # **32559**	Revision: **3.16**
Issue Date: **November 2009**	

Contents

List of Figures

List of Tables

Revision History

Date	Rev.	Description
November 2009	3.16	Updated Table 2 "Packages". Updated 3.6.2 "DRAM Controller". Updated 4.5.20 "DRAM Controller Miscellaneous Register". Added Table 42 "SO-DIMM Address Timings and Drive Strengths for ASB1 Package". Updated Table 57 "Valid MC2_ADDR Bits". Updated 10.5.1.1 "P-state Recognition Algorithm". Updated Table 78 "Configuration Register Settings for Power Management".

Date	Rev.	Description
September 2008	3.12	Updated Table 1 "Related Documents", Table 2 "Packages", and Table 3 "DIMM Support Per Package". Added 2.3 "Using L2 Cache as General Storage During Boot". Updated 3.6.6 "On-Line Spare" and 3.7.3.2 "TDIE max and TCONTROL for Socket S1g1 and ASB1 Processors". Clarified 4.3.9 "HyperTransport™ Transaction Control Register". Updated Table 8 "DRAM CS Base Address and DRAM CS Mask Registers". Corrected Table 11 "DRAM address mapping 128-bit interface". Updated 4.5.10 "DRAM Timing Low Register". Clarified 4.5.17 "Address Timing Control Register". Updated 4.5.20 "DRAM Controller Miscellaneous Register". Updated 4.6.4.1 "MCA NB Control Register". Clarified 4.6.4.2 "MCA NB Configuration Register". Updated Table 23 "Northbridge Error Codes". Updated 4.6.10 "XBAR Flow Control Buffers" and added Table 33 "2P Dual Coherent Link Data Buffer Allocation". Clarified 5.1.1.9 "tRTP (Internal Read to Precharge Command Delay)". Added Table 42 "SO-DIMM Address Timings and Drive Strengths for ASB1 Package". Added Table 43 "Unbuffered DIMM Address Timings and Drive Strengths for ASB1 Package". Corrected 6.3.2.5 "MCi_MISC—Machine Check Miscellaneous Registers". Clarified 6.4.1.1 "MC0_CTL—DC Machine Check Control Register". Clarified 6.5 "Initializing the Machine Check Mechanism". Updated 10.9.1 "Chipset Configuration for Power Management". Updated Table 78 "Configuration Register Settings for Power Management". Corrected 12 "CPUID Function Registers" Fn8000_0006_ECX. Clarified: 14.2.1.3 "HWCR Register".

Date	Rev.	Description
July 2007	3.08	Updated Table 4 “DRAM Interface Speed vs. CPU Core Clock Multiplier”. Updated 3.7.3 “Thermal Diode”. Added 3.7.3.1 “TCASE max and TCONTROL max for Socket F (1207) and Socket AM2 Processors”. Added 3.7.3.2 “TDIE max and TCONTROL for Socket S1g1 and ASB1 Processors”. Updated 4.5.11 “DRAM Timing High Register”. Updated 4.5.17 “Address Timing Control Register”. Corrected 4.6.4.2 “MCA NB Configuration Register”. Updated 4.6.4.7 “Watchdog Timer Errors”. Updated 4.6.10 “XBAR Flow Control Buffers”. Clarified 5.1.1.19 “DIMM ODT”. Updated Table 38. Updated Table 39 “SO-DIMM ODT Settings”. Updated Table 40 “Four Bank Activate Window Values”. Updated Table 45 “SO-DIMM Address Timings and Drive Strengths for S1g1 Package”. Corrected Table 47 “Registered DIMM Address Timings and Drive Strengths for F(1207) Package 8-DIMM System”. Corrected 5.1.5.1 “DQS Receiver Enable Training”. Corrected Table 67 “ACPI-State Support by System Class”. Updated Table 69 “Low FID Frequency Table (< 1600 MHz)”. Clarified 10.3 “Throttling”. Clarified 10.5.7.2.2, 10.6.2.1.3 and 10.8 for PLL lock time. Updated Table 78 “Configuration Register Settings for Power Management”. Updated 13.16 “Cache Initialization For General Storage During Boot”. Corrected 14.2.1.3 “HWCR Register”. Updated 14.2.9.1 “FIDVID_CTL Register”. Clarified 14.2.11 “Interrupt Pending Message Register”. Added 14.2.12 “VM_CR Register”. Added 14.2.13 “VM_HSAVE_PA Register”.

Date	Rev.	Description
December 2006	3.04	Corrected 4.5.20 "DRAM Controller Miscellaneous Register". Clarified 4.6.22 "HyperTransport™ FIFO Read Pointer Optimization Register". Updated 4.6.23 "Thermtrip Status Register". Updated Table 45 "SO-DIMM Address Timings and Drive Strengths for S1g1 Package". Updated Table 69 "Low FID Frequency Table (< 1600 MHz)" for 100 MHz frequency steps. Updated Table 70 "High FID Frequency Table (>= 1600 MHz)" for 100 MHz frequency steps. Updated Figure 10. Added Figure 11. Updated Figure 12. Added Figure 13. Added Figure 14. Added 10.7 "ACPI 3.0 Processor P-State Objects". Updated Table 78 "Configuration Register Settings for Power Management". Updated 12 "CPUID Function Registers" Fn8000_0001_ECX, Fn8000_0007_EDX, and Fn8000_000A_EDX. Updated Table 83 FID Code Translations with odd FID codes.
May 2006	3.00	Initial Public Release

1 Introduction and Overview

The *BIOS and Kernel Developer's Guide for AMD NPT Family 0Fh Processors* is intended for programmers involved in the development of low-level BIOS (basic input/output system) functions, drivers, and operating system kernel modules. It assumes previous experience in microprocessor programming, as well as fundamental knowledge of legacy x86 and AMD64 microprocessor architecture. The reader should also have previous experience in BIOS or OS kernel design issues, as related to microprocessor systems, and a familiarity with various platform technologies, such as DDR, HyperTransport™ technology, and JTAG.

1.1 About This Guide

This guide covers the *implementation-specific* features of AMD NPT Family 0Fh Processors, as opposed to architectural features. AMD64 architectural features include the AMD64 technology, general-purpose, multimedia, and x87 floating-point registers, as well as other programmer-visible features defined to be constant across all processors. A subset of implementation-specific features are not defined by the processor architectural specifications. These implementation-specific features may differ in various details from one implementation to the next.

Note: *The term* processor *in this document refers to AMD NPT processor architecture. NPT, or New Platform Technology, refers to the technology developed to support AMD processors such as AMD F (1207), AMD AM2, AMD ASB1, and AMD S1g1 processors.*

The implementation-specific features covered in the following chapters include:

- Model-specific registers
- Processor initialization
- Integrated memory system configuration
- HyperTransport technology fabric initialization
- Performance monitoring and special debug features
- DRAM configuration
- Machine check error codes
- Thermal and power management

1.2 Related Documents

The references listed in Table 1 may prove invaluable towards a complete understanding of the subject matter in this volume.

Table 1. Related Documents

Title	Order#
AMD64 Architecture Programmer's Manual, Volume 1, *Application Programming*	24592
AMD64 Architecture Programmer's Manual, Volume 2, *System Programming*	24593
AMD64 Architecture Programmer's Manual, Volume 3, *General-Purpose and System Instructions* *AMD64 Architecture Programmer's Manual*, Volume 4, *128-Bit Media Instructions* *AMD64 Architecture Programmer's Manual*, Volume 5, *64-Bit Media and x87 Floating-Point Instructions*	24594 (three-volume kit)
AMD Socket F (1207) Processor Functional Data Sheet	31118
AMD Socket AM2 Processor Functional Data Sheet	31117
AMD Socket S1g1 Processor Functional Data Sheet	31731
AMD ASB1 Processor Functional Data Sheet	45584
AMD NPT Processor Electrical Data Sheet	31119
AMD NPT Family 0Fh Server and Workstation Processor Power and Thermal Data Sheet	33953
AMD NPT Family 0Fh Desktop Processor Power and Thermal Data Sheet	33954
AMD NPT Family 0Fh Mobile Processor Power and Thermal Data Sheet	33952
AMD Athlon™ 64 and AMD Opteron™ Processors Thermal Design Guide	26633
AMD Athlon™ 64 and AMD Opteron™ Processors Thermal Test Kit Instruction Manual	26956
Thermal Design and Testing Guide for AMD NPT Family 0Fh Processors in Mobile Systems	32769
CPUID Guide for AMD Athlon™ 64 and AMD Opteron™ Processors	25481
Revision Guide for the AMD NPT Family 0F Processors	33610
HyperTransport™ I/O Link Specification, Rev. 1.03	http://www.hypertransport.org

1.3 Conventions and Definitions

Some of the following definitions assume a knowledge of the legacy x86 architecture. See Table 1 for documents that include information on the legacy x86 architecture.

Additional definitions are provided in the glossary at the end of this book, beginning on page 435. That glossary includes the most important terminology of the AMD64 architecture.

1.3.1 Notation

1011b. A binary value. In this example, a 4-bit value is shown.

F0EAh. A hexadecimal value. In this example, a 2-byte value is shown.

[1,2]. A range that includes the left-most value (in this case, 1) but excludes the right-most value (in this case, 2).

7–4. A bit range, from bit 7 to 4, inclusive. The high-order bit is shown first.

#GP(0). Notation indicating a general-protection exception (#GP) with error code of 0.

^ . Logical exclusive-OR operator; sometimes used as "raised to the power of" as well, as indicated by the context in which it is used.

CR0–CR4. A register range, from register CR0 through CR4, inclusive, with the low-order register first.

CR0.PE = 1. Notation indicating that the PE bit of the CR0 register has a value of 1.

DS:rSI. The contents of a memory location whose segment is DS and whose byte address is located in the rSI register.

EFER.LME = 0. Notation indicating that the LME bit of the EFER register has a value of 0.

FF /0. Notation indicating that FF is the first byte of an opcode and a sub-opcode field in the MODRM byte has a value of 0.

DramEn. Notation indicating that the bit is an enable ("En" or "EN" is part of the name) and that a 1 specifies that the function is enabled.

MemClkDis. Notation indicating that the bit is a disable ("Dis" or "DIS" is part of the name) and that a 1 specifies that the function is disabled.

1.4 Register Differences in Revisions of AMD NPT Family 0Fh Processors

Some changes to the register set are introduced with different silicon revisions. Refer to the *Revision Guide for the AMD NPT Family 0F Processors*, order# 33610 for information about how to identify different processor revisions. The following summarizes register changes after the initial revision.

Revision F (differences from Revision E DDR1 Family 0Fh Processors)

- Function 3, Offset E4h, ThermtpSense (bit 3) renamed ThermtpSense0.
- Function 3, Offset E4h, TjOffset (bits 28-24), CurTmp (bits 23-16), ThermSenseSel (bit 6), ThermtpSense1 (bit 4) and ThermSenseCoreSel (bit2) added.
- Function 3, Offset B0h, On-Line Spare Control Register added.
- Function 2 Offset 40h, 44h, 48h, 4Ch, 50h, 54h, 58h, 5Ch, DRAM CS Base Registers modified.
- Function 2 Offset 60h, 64h, 68h, 6Ch, DRAM CS Mask Registers modified.
- Function 2 Offset 70h, 74h, 78h, 7Ch, DRAM CS Mask Registers removed.

- Function 2 Offset 80h, DRAM Bank Address Mapping Register modified.
- Function 2 Offset 88h, DRAM Timing Low Register modified.
- Function 2 Offset 8Ch, DRAM Timing High Register modified.
- Function 2 Offset 90h, DRAM Configuration Low Register modified.
- Function 2 Offset 8Ch, DRAM Timing High Register modified.
- Function 2 Offset 94h, DRAM Configuration High Register modified.
- Function 2 Offset78h, DRAM Control Register added.
- Function 2 Offset 7Ch, DRAM Initialization Register added.
- Function 2 Offset 98h, DRAM Controller Additional Data Offset Register added.
- Function 2 Offset 9Ch, DRAM Controller Additional Data Register added.
- Function 2 Offset A0h, DRAM Controller Miscellaneous Register added.
- Function 2 Offset 98h, DRAM Delay Line Register removed.
- Function 2 Offset 9Ch, Scratch Register removed.
- MSR C001_0114 (VM_CR) added.
- MSR C001_0117 (VM_HSAVE_PA) added.
- MSR C001_0041h (FIDVID_CTL), NewVID (bit13) added.
- MSR C001_0042h (FIDVID_STATUS), AltVidOffset (bits 60-57), PstateStep (bit 56), MaxVID (bit 53), StartVID (bit 45), CurrVID (bit 37), and MaxRampVID (bit 28) added.
- MSR C001_0042h (FIDVID_STATUS) MinVID (61-56) removed.
- APIC 30h (APIC_VER), ExtApicSpace (bit 31) added.
- APIC 400h (EXT_APIC_FEAT) added.
- APIC 500h (THRCNT_INT0_LVT) added.
- MSR C001_0060h (BIST Results) added.
- Function 0 Offset 6Ch, BiosRstDet (bits 9-8) added.
- MSR C001_0010h (SYSCFG), Tom2ForceMemTypeWB (bit 22) added.
- Function 3 Offset A0h, Power Control Miscellaneous Register added.
- Function 3 Offset FCh, CPUID Family Model Register added.
- MSR 0413h (MC4_MISC—DRAM Errors Threshold) added.
- Function 0, Offset 68h, InstallStateS (bit 23) added.

- Function 3, Offset 44h, SyncOnDramAdrParErrEn (bit 30), DisMstAbtCpuErrRsp (bit 29) and DisTgtAbtCpuErrRsp (bit 28) added.
- Function 3, Offset D8h, GfxMode (bit 27) and AltVidTriEn (bit 26) added.
- Function 3 Offset 40h, DramParEn (bit 18) added.
- MSR C001_0055h (Interrupt Pending), C1eOnCmpHalt (bit 28) and SmiOnCmpHalt (bit 27) added.
- CPUID Fn[8000_0001]_EBX BrandId (bits 15-12) added.
- CPUID Fn[0000_0001]_ECX CMPXCHG16B (bit 13) added.
- CPUID Fn[8000_0001]_ECX LockMovCr0 (bit 4), ExtApicSpace (bit 3), and SVM (Bit 2) added.
- CPUID Fn[8000_0001]_EDX RDTSCP (bit 27) added.
- CPUID Fn[8000_000A] added.

Revision G

- CPUID Fn[8000_0001]_ECX 3DNowPrefetch (bit 8) added.
- CPUID Fn[8000_0007]_EDX 100MhzSteps (bit 6) added.
- CPUID Fn[8000_000A]_EDX LBR (bit 1) added.
- Revision G2: Function 2 Offset 9Ch, Index 04h, AtcDllMaxPhases (bit 28) added.
- Function 3 Offset E4h, CurTmp (bits 15-14) added.

2 Processor Initialization and Configuration

Each AMD NPT Family 0Fh Processor based system has one processor with its HyperTransport™ link connected to a HyperTransport™ I/O hub. When a reset signal is applied, this processor is initialized as the bootstrap processor (BSP). In a multiple processor system, any other processors are initialized as application processors (APs). The BSP begins executing code from the reset vector (0xFFFFFFF0), while each of the APs waits for its Request Disable bit to clear to 0. An AP does not fetch code until its Request Disable bit is cleared. Both BSP and AP operate in 16-bit Real mode after reset. The BSP has the Boot Strap Processor bit set in its APIC_BASE (001Bh) MSR, and each AP has this bit cleared.

The processor node is addressed by its Node ID on the HyperTransport link and can be accessed with a device number in the PCI configuration space on Bus 0. The Node ID 0 is mapped to Device 24, the Node ID 1 is mapped to Device 25, and so on. The BSP is initialized with Node ID 0 after reset, and all APs are initialized with Node ID 7.

The BSP is ready to access its Northbridge and memory controller after its routing table is enabled. The APs are not accessible from the BSP until the links and the routing table are configured.

The initial processor states are described in the "Processor Initialization State" section of the *AMD64 Architecture Programmer's Manual:* Volume 2, *System Programming*.

2.1 Bootstrap Processor Initialization

The BSP is responsible for the execution of the BIOS Power-On Self Test (POST) and initialization of APs. The BSP must perform the following tasks:

- AP detection and routing table initialization
- Noncoherent HyperTransport device initialization
- Link width and frequency initialization
- Memory controller initialization on all processor nodes
- Address map table initialization

2.1.1 Detecting AP and Initializing Routing Table

The AMD Opteron™ processor has three HyperTransport links; therefore, an AMD Opteron™ system can have from one to eight processors. The BSP must detect all APs in the system, starting from its adjacent processor. Since the APs are initialized at Node 7, the BSP must set entry 7 of its Routing Table before the AP can be accessed. The adjacent AP will respond to a PCI configuration

read if it is present. A new Node ID can be assigned to the AP after the routing information is updated in the routing table. Chapter 9, "HyperTransport™ Technology Configuration and Enumeration" explains the steps for enumerating and initializing routing tables in one-processor (UP), two-processor (DP), and *n* processor systems.

The AMD Athlon™ 64 processor has one HyperTransport link. Because one link must be connected to the HyperTransport I/O hub, an AMD Athlon™ 64 system can only be used in a uniprocessor system.

2.1.2 Initializing Noncoherent HyperTransport™ Technology Devices

Once all APs have been detected and the routing tables and link control registers have been initialized, the BSP must initialize noncoherent HyperTransport technology devices.

A noncoherent HyperTransport technology device has either one or two links. A tunnel device has two links and a terminal device has one link. The HyperTransport I/O hub is a typical example of a terminal noncoherent HyperTransport device. The noncoherent HyperTransport device is identified by its Unit ID in a noncoherent HyperTransport link and can be accessed with a device number in the PCI configuration space. The device number in PCI space is the same as the Unit ID assigned to the noncoherent HyperTransport device.

After reset, the Unit ID of each noncoherent HyperTransport technology device is initialized with a value of 0. When a PCI configuration read is performed from the BSP through a noncoherent HyperTransport link, the noncoherent HyperTransport device connected to the port responds. A new non-zero Unit ID value must be assigned to this device. The BIOS continues this process until no device responds at Device 0. The bus number range must be set to the noncoherent HyperTransport link on the BSP and in the address map table prior to the detection. The link control and status registers of a noncoherent device are implemented in the capability register block.

The noncoherent HyperTransport technology devices connected to a port on the AP node can be detected in the same way. Again, the bus number range must be set to the noncoherent HyperTransport link on this AP and in the address map table. The noncoherent device can be detected with the starting bus number, Device 0 on the PCI configuration space.

A noncoherent HyperTransport technology device may use more than one Unit ID. The new ID assigned to a device is its starting ID, the Base Unit ID (BaseUID). The next logic device in this noncoherent HyperTransport device can be identified with BaseUID + 1, and so on. The Unit ID Count field in the Capabilities register indicates how many Unit IDs this device uses.

2.1.3 Initializing Link Width and Frequency

The HyperTransport link width and frequency are initialized between the adjacent coherent and/or noncoherent HyperTransport technology devices during the reset sequence. After AP and noncoherent HyperTransport device detection, the link width and frequency can be changed based on the capability of the adjacent devices or the implementation of the system. See "HyperTransport™ Link Frequency Selection" on page 366 for more information. To make the new link width and

frequency take effect, an LDTSTOP_L needs to be asserted or a warm reset must be performed. The BIOS must ensure that LDTSTOP_L assertion for link width and frequency changes does not occur within 200 µs of reset. LDTSTOP_L must be asserted for a minimum of 2 µs for link width and frequency changes.

2.1.4 Initializing the Memory Controller on All Processor Nodes

The BSP is responsible for configuring the memory controller on all processor nodes and for initializing the DRAM map table. Chapter 5, "DRAM Configuration," describes the functionality of the memory controller and the steps required to initialize memory.

After memory configuration, the memory address MSRs must be set accordingly. The Top of Memory MSR is used for the top of DRAM below 4 Gbytes, and TOP_MEM2 is set to the top of DRAM above 4 Gbytes.

2.1.5 Initializing the Address Map Table

Each processor node has an address map table. This table contains the address map for the DRAM area, the address map for PCI memory spaces (MMIO), the address map for PCI I/O spaces, and the bus number range for each noncoherent HyperTransport link. The address map table must be duplicated on all processors in a system.

2.2 Application Processor Initialization

The application processor (AP) waits until its request disable bit is cleared to 0. The BIOS may clear this bit for an AP as soon as the AP node detection is completed and the routing tables are initialized, or just before the AP register initialization.

When the request disable bit is cleared, the corresponding AP starts to fetch code from the reset vector (0xFFFFFFF0). The BSP bit in AP APIC_BASE register is cleared at reset, so it can be used to terminate the AP execution after the initial APIC initialization.

The AP register initialization is the same as that of other AMD x86 processor families, such as the AMD Athlon™ processors.

2.3 Using L2 Cache as General Storage During Boot

Prior to initializing the DRAM controller for system memory, BIOS may use the L2 cache of each core as general storage. BIOS manages the mapping of the L2 storage such that cacheable accesses do not cause L2 victims.

The L2 cache as storage is described as follows:

- Each core has its own L2 cache.
- The L2 size, L2 associativity, and L2 line size is determined by reading CPUID Fn8000_0006 L2 Cache Identifiers ECX[L2Size, L2Assoc, L2LineSize]. (Note that L2WayNum is defined to be the number of ways indicated by the L2Assoc code.)
  - The L2 cache is viewed as (L2Size/L2LineSize) cache lines of storage, organized as L2WayNum ways, each way being (L2Size/L2WayNum) in size.
  - For each of the following values of L2Size, the following values are defined:
    - L2Size=128KB: L2Tag=PhysAddr[39:13], L2WayIndex=PhysAddr[12:6].
    - L2Size=256KB: L2Tag=PhysAddr[39:14], L2WayIndex=PhysAddr[13:6].
    - L2Size=512KB: L2Tag=PhysAddr[39:15], L2WayIndex=PhysAddr[14:6].
    - L2Size=1MB: L2Tag=PhysAddr[39:16], L2WayIndex=PhysAddr[15:6].
  - PhysAddr[5:0] addresses the L2LineSize number of bytes of storage associated with the cache line.
  - The L2 cache, when allocating a line at L2WayIndex:
    - Picks an invalid way before picking a valid way.
    - Prioritizes the picking of invalid ways such that way 0 is the highest priority and L2WayNum-1 is the lowest priority.
- In order to prevent victimizing L2 data, no more than L2WayNum cache lines may have the same L2WayIndex.
  - Software does not need to know which ways the L2WayNum lines are allocated to for any given value of L2WayIndex, only that invalid ways will be selected for allocation before valid ways will be selected for allocation.

It is recommended that BIOS:

- Assume a simpler allocation of L2 cache memory, being L2WayNum size-aligned blocks of memory,each being L2Size/L2WayNum bytes.
- Assume the minimum L2Size for all configurations.

The following memory types are supported as follows:

- WP-IO: BIOS ROM may be assigned the write-protect IO memory type and may be accessed read-only as data and fetched as instructions.
  - BIOS initializes a location in the L2 cache, mapped as write-protect IO, with 1 load of any size or an instruction fetch to any location within the L2LineSize cache line.
- WB-DRAM: General storage may be assigned the write-back DRAM memory type and may be accessed as read-write data, but not accessed by instruction fetch.
  - BIOS initializes a location in the L2 cache, mapped as write-back DRAM, with 1 read to at least 1 byte of the L2LineSize cache line. BIOS may store to a line only after it has been allocated by a load.
  - Fills, sent to the disabled memory controller, return undefined data.
  - All of memory space that is not accessed as WB-DRAM space must be marked as UC memory type.

Performance monitor event EventSelect 07Fh [L2 Fill/Writeback], subevent bit 1, titled "L2 Writebacks to system", can be used to indicate whether L2 dirty data was victimized and sent to the disabled memory controller.

The following requirements must be satisfied prior to using the cache as general storage:

- Paging must be disabled.
- MSRC001_1022[8]=1.
- MSRC001_1022[13]=1.
- MSRC001_0015[INVD_WBINVD]=0.
- CLFLUSH, INVD, and WBINVD must not be used.
- The BIOS must not use 3DNow!™, SSE, or MMX™ instructions, with the exception of the following list: MOVD, MOVQ, MOVDQA, MOVQ2DQ, MOVDQ2Q.
- The BIOS must not enable exceptions, page-faults, and other interrupts.
- BIOS must not use software prefetches.

When the BIOS is done using the cache as general storage the following steps are followed:

1. An INVD instruction should be executed on each core that used cache as general storage.
2. If DRAM is initialized and there is data in the cache that needs to get moved to main memory, CLFLUSH or WBINVD may be used instead of INVD, but software must ensure that needed data in main memory is not overwritten.
3. Restore the following configuration state: MSRC001_1022[8]=0, MSRC001_1022[13]=0, MSRC001_0015[INVD_WBINVD].

2.4 BIOS Requirement for 64-Bit Operation

In general, the BIOS need not be aware of 64-bit mode in POST. There are no additional requirements to support 64-bit mode. The operating system determines whether to enable 64-bit mode after the BIOS invokes the operating system loader in legacy mode. There are two areas that need additional explanation:

- Sizing and testing memory above 4 Gbytes
- Using BIOS callback when in 64-bit mode

2.4.1 Sizing and Testing Memory above 4 Gbytes

AMD NPT Family 0Fh Processors have extended physical address extension (PAE) to support a 40-bit address space. Thus, the BIOS can set up a 32-bit page table that allows it to size and test all physical memory in the system.

2.4.2 Using BIOS Callbacks in 64-Bit Mode

A BIOS callback is a mechanism that allows an operating system to call a service routine in the BIOS. BIOS callbacks on an AMD NPT Family 0Fh Processor platform in 64-bit mode are not supported by AMD64 operating systems. An operating system loader must be invoked in legacy mode with paging disabled, so that the loader can use 16/32-bit BIOS callbacks.

ACPI code and operating system drivers are not affected by AMD64 operating systems that do not use a BIOS callback in 64-bit mode. ACPI is coded in ASL, which is not affected by the operating system mode. AMD64 operating system drivers are not allowed to use ROM callbacks.

3 Functional Description

3.1 Overview

The processor is designed to support performance desktop and workstation applications. It provides single or dual core capability, up to three high-performance HyperTransport™ links, as well as a single 128-bit high-performance DDR2 SDRAM memory controller. A block diagram of the processor is shown in Figure 1.

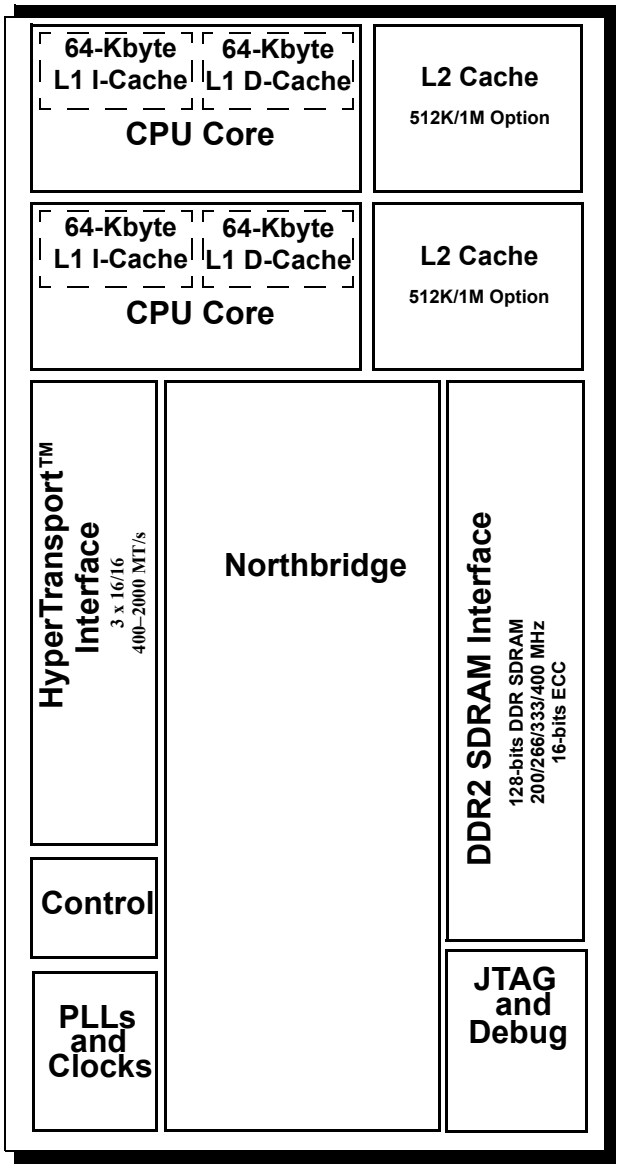


Figure 1. Processor Block Diagram

3.2 Instruction Set Support

The processor supports the standard x86-instruction set defined in the *AMD64 Architecture Programmer's Manual, volumes 3–5, order# 24594*. The processor also supports the following extensions to the standard x86 instruction set, which are described in the same volume set:

- AMD64 instructions
- MMX® and 3DNow!™ technology instructions
- SSE, and SSE2, and SSE3 instructions

3.3 Supported Packages

The processor is implemented in the following packages.

Package	Memory I/Fs	Links	Notes
ASB1	Unbuffered DIMMs; one or two channels (by brand)	One 16-bit	Targets the small form factor desktop, mobile and embedded market.
S1g1	Unbuffered DIMMs; two channels	One 16-bit	Targets the mobile market.
AM2	Unbuffered DIMMs; two channels	One 16-bit	Targets the desktop market.
F(1207)	Registered DIMMs; two channels	Three 16-bit	Targets the server and workstation market.

Table 2. Packages

3.4 Address Space

The processor supports 48 address bits of virtual memory space (256 terabyte) as indicated by CPUID Fn8000_0008_EAX and 40 address bits of physical memory space (1 terabyte).

3.5 Internal Cache Structures

The processor implements internal caching structures as described in the following sections.

3.5.1 Level 1 Caches

The L1 data cache (L1 D-Cache) contains 64 Kbytes of storage organized as two-way set associative. The L1 data cache is protected with ECC. Two simultaneous 64-bit operations (load, store or combination) are supported. The L1 instruction cache (L1 I-Cache) contains 64 Kbytes of storage organized as 2-way associative. The L1 Instruction Cache is protected with parity.

3.5.2 Level 2 Cache

The L2 cache contains both instruction and data stream information. It is organized as 16-way set-associative. The L2 cache data and tag store is protected with ECC. When a given cache line in the L2

cache contains instruction stream information, the ECC bits associated with the given line are used to store predecode and branch prediction information.

3.6 Northbridge

The Northbridge logic in the processor refers to the HyperTransport™ technology interface, the memory controller, and their respective interfaces to the CPU core. These interfaces are described in more detail in the following sections.

3.6.1 HyperTransport™ Technology Overview

The processor includes up to three 16-bit HyperTransport™ technology interfaces capable of operating up to 2000 mega-transfers per second (MT/s) with a resulting bandwidth of up to 8.0 Gbytes/s (4.0 Gbytes/s in each direction). Refer to Table 2 on page 38 for product-specific details on HyperTransport™ interfaces. The processor supports HyperTransport™ technology synchronous clocking mode. Refer to the *HyperTransport™ I/O Link Specification* (www.hypertransport.org) for details of link operation.

Note: *The processor's HyperTransport links operate in Synchronous (Sync) mode as described in the HyperTransport I/O Link Specification. Sync mode operation requires the transmit and receive clocks for all links to be derived from the same time base. If different clock generators are used to drive the reference clock to the devices on both sides of the link, then the following requirements must be satisfied to ensure proper link operation:*

1. *All clock generators must use the same reference clock source.*
2. *Spread spectrum clocking must not be enabled in any of the clock generators unless the frequency deviations are synchronized between the outputs of all clock generators.*

3.6.1.1 Link Initialization

The *HyperTransport™ I/O Link Specification* details the negotiation that occurs at power-on to determine the widths and rates used with the link. See Chapter 9, "HyperTransport™ Technology Configuration and Enumeration" on page 289 for information about link initialization and setup of routing tables.

3.6.1.2 HyperTransport™ Technology Transfer Speeds

The HyperTransport™ link of the processor is capable of operating at 400, 800, 1200, 1600, and 2000 MT/s. The link transfer rate is determined during the software configuration of the system, as specified in the *HyperTransport™ I/O Link Specification*.

Processors with multiple HyperTransport™ links are capable of operating the links at different frequencies. There are three supported link frequency groups:

- 800 MHz, 400 MHz, 200 MHz

- 1000 MHz
- 600 MHz

Processors can be configured with link frequencies from up to two link frequency groups.

3.6.1.3 System Address Map

System software must not map memory in the reserved HyperTransport address regions. The *HyperTransport™ I/O Link Specification* details the address map available to system hosts and devices. Downstream host accesses to reserved HyperTransport address regions result in a page fault. Upstream system device accesses to reserved HyperTransport address regions result in undefined operation.

3.6.2 DRAM Controller

The processor's DRAM controller provides a programmable interface to a variety of standard DDR2 SDRAM DIMM configurations. The following features are supported:

- Self-Refresh mode
- Unbuffered and registered DIMMs. The package type determines the number of logical DIMMs supported and what type of DRAM is supported.

Table 3. DIMM Support Per Package

Package	Number of DIMMs per Channel			
	Registered DIMMs	4 Rank Registered DIMMs	Unbuffered DIMMs	SO-DIMMs
F(1207)	4	2	0	0
AM2	0	0	2	1
S1g1	0	0	0	1
ASB1	0	0	1	1

- The controller provides programmable control of DRAM timing parameters to support the following memory speeds:
  - 200-MHz (DDR2-400) PC2-3200 DIMMs
  - 266-MHz (DDR2-533) PC2-4200 DIMMs
  - 333-MHz (DDR2-667) PC2-5300 DIMMs
  - 400-MHz (DDR2-800) PC2-6400 DIMMs
- DRAM devices that are 4, 8 and 16 bits wide.
- DIMM sizes from 128 Mbytes to 16 Gbyte.

- Interleaving memory within DIMMs.
- Stacked registered DIMMs.
- ECC checking with double-bit detect with single-bit correct in ASB1, AM2 and F(1207) packages.
- May be configured for 32-byte or 64-byte burst length.
- Programmable page-policy:
  - Supports up to sixteen open pages total across all chip-selects.
  - Statically idle open-page time.
  - Optional dynamic precharge control based on page-hit/miss history.

For programming information and specific details of these features, see 4.5 DRAM Controller page 95.

3.6.2.1 DRAM Operation

At power on reset, the M[B, A]_CLK_H/L and M[B, A]_RESET_L (RDIMMs only) pins are driven low while the processor PLLs are ramping. Clocks are driven on the M[B, A]_CLK_H/L pins only after BIOS programs the appropriate clock ratio value in the memory controller configuration registers. The actual DRAM frequency may vary for some speeds based on the CPU clock multiplier and other configuration options, as shown in Table 4 on page 41 (the memory controller automatically adjusts refresh counters at all speeds as required to meet the device refresh specifications).

Table 4. DRAM Interface Speed vs. CPU Core Clock Multiplier

Multiplier	Core Frequency	DRAM Frequency			
		200 MHz	266 MHz	333 MHz	400 MHz
4	800 MHz	160.00	160.00	160.00	160.00
4.5	900 MHz	180.00	180.00	180.00	180.00
5	1000 MHz	200.00	200.00	200.00	200.00
5.5	1100 MHz	183.33	220.00	220.00	220.00
6	1200 MHz	200.00	240.00	240.00	240.00
6.5	1300 MHz	185.71	260.00	260.00	260.00
7	1400 MHz	200.00	233.33	280.00	280.00
7.5	1500 MHz	187.50	250.00	300.00	300.00

Table 4. DRAM Interface Speed vs. CPU Core Clock Multiplier

Multiplier	Core Frequency	DRAM Frequency			
		200 MHz	266 MHz	333 MHz	400 MHz
8	1600 MHz	200.00	266.67	320.00	320.00
8.5	1700 MHz	188.89	242.86	283.33	340.00
9	1800 MHz	200.00	257.14	300.00	360.00
9.5	1900 MHz	190.00	237.50	316.67	380.00
10	2000 MHz	200.00	250	333.33	400.00
10.5	2100 MHz	190.9	262.50	300.00	350.00
11	2200 MHz	200.00	244.44	314.29	366.67
11.5	2300 MHz	191.67	255.56	328.57	383.33
12	2400 MHz	200.00	266.67	300.00	400.00
12.5	2500 MHz	192.3	250.00	312.50	357.14
13	2600 MHz	200.00	260.00	325.00	371.43
13.5	2700 MHz	192.86	245.46	300.00	385.71
14	2800 MHz	200.00	254.55	311.11	400.00
14.5	2900 MHz	193.33	263.64	322.22	362.50
15	3000 MHz	200.00	250.00	333.33	375.00
15.5	3100 MHz	193.75	258.33	310.00	387.50
Note: *Fractional (half-step)multipliers are supported in revision G and later revisions.*					

The use of 2T timing allows support of many DIMM combinations at the higher DDR speeds. The 2T timing feature causes commands and addresses to be driven for two clock cycles and qualified with an associated chip select on the second clock cycle, allowing an extra clock of setup to accommodate heavy DIMM loading (such as double-rank DIMMs). The use of 2T timing is only supported for unbuffered DIMMs.

The controller supports programmable timing and refresh. Auto-refresh is supported and is staggered by t_{RFC} across chip-selects to reduce system noise. Unpopulated DIMM slots are not refreshed.

3.6.2.2 DRAM Power Management

The memory controller supports self-refresh mode to accommodate various power management states such as ACPI C3, S1, and S3 states. The M[B,A]_RESET_L pin is provided for resetting the registers on registered DDR2 SDRAM DIMMs as required for the S3 (Suspend to RAM) power management state.

3.6.3 Main Memory Hardware Scrubbing

The memory controller scrubs the main memory arrays to prevent the build up of soft errors. Any correctable or non-correctable errors are logged to the machine check logs and can be programmed to invoke the machine check interrupt. The scrubbing function can be used in three modes as described in the following sections.

3.6.3.1 Sequential Scrubbing

In this mode, the scrubber sequentially proceeds through main memory, performing a read-write cycle or a read-modify-write cycle if a correctable error is found. The scrubber scrubs one cache line on each scrub interval that is programmable from 40 ns to 84 ms.

3.6.3.2 Source Correction Scrubbing

In this mode, the scrubber is directed to scrub any cache line that is the source of any corrected error during normal accesses. During normal operation when source correction scrubbing is disabled, single-bit errors are corrected on the fly and the corrected data is passed without updating the source memory location. When source scrubbing is enabled the scrubber also corrects the source memory location.

3.6.3.3 Sequential Plus Source Correction Scrubbing

When both sequential and source correction scrubbing are enabled, the scrubber sequentially proceeds through main memory. If a correctable error is detected during normal operation, the scrubber is redirected to the location of the error, and after it corrects that location in main memory it resumes sequential scrubbing at the previous location.

3.6.4 Memory Interleaving Modes

Interleaving is defined as the spreading contiguous physical address space over multiple DIMM banks, as opposed to each DIMM owning a single contiguous address space. This is accomplished by using lower-order address bits to select between DIMMs. The processor supports two different types of interleaving modes:

• CS: interleaving between the DIMM banks of a channel. This is controlled though DRAM CS Base Address Registers (Function 2: Offsets 40h-5Ch).

• Node: interleaving between DIMMs of different processor nodes. This is controlled though DRAM Base i Registers (Function 1: Offsets 40h, 48h, 50h, 58h, 60h, 68h, 70h, 78h) and DRAM Limit i Registers (Function 1: Offsets 44h, 4Ch, 54h, 5Ch, 64h, 6Ch, 74h, 7Ch).

Any combination of these interleaving modes may be enabled concurrently.

3.6.5 Memory Hoisting

Typically, memory mapped I/O space is mapped from MMIOLowAddr to 4GB-1, and DRAM memory is mapped from address 0 to TOP_MEM, an address that is less or equal to MMIOLowAddr-1. DRAM memory can also be mapped from 4GB to TOM2, an address greater than 4GB. Memory hoisting is defined as reclaiming the DRAM space that would naturally reside in the MMIO hole just below the 4G address level.

Memory that resides in the MMIO is repositioned above the 4G level by programming the DRAM Hole Address Register and the DramLimit properly.

The memory hoisting offset field, DramHoleOffset, is programmed as shown in the following equation:

DramHoleOffset[31:24] = ((100h - DramHoleBase[31:24]) + DramBase[31:24]);

The DramLimit field must be programmed to include the size of the DRAM hole as shown in the following equation:

DramLimit = 4GB + Hole Size;

3.6.5.1 Non-Node-Interleave Mode

For a system that is configured in the non-node-interleave mode, the DRAM Hole Address Register is enabled only on the node that has its DRAM space overlapped with the MMIO hole if the calculated DramHoleOffset is not equal to 4GB. The DramLimit of the current node must be adjusted by adding the size of the MMIO hole. The DramBase and DramLimit registers for the following nodes must also adjusted by adding the size of the MMIO hole.

Example. A two processor system with 4 GB DRAM on each processor and a 1 GB MMIO hole.

Processor 0 DramBase[39:24] = 0000h

Processor 0 DramLimit[39:24] = 013Fh (5GB -1; DramLimit + Hole Size)

Processor 0 DramHoleBase[31:24] = C0h (3GB; Starting address of MMIO hole)

Processor 0 DramHoleOffset [31:24] = 40h (1GB)

Processor 0 DramHoleValid = 1(Memory hoisting enabled)

Processor 1 DramBase[39:24] = 0140h (5GB)

Processor 1 DramLimit[39:24] = 023Fh (9GB -1)

Processor 1 DramHoleBase[31:24] = 00h (Memory hoisting is not enabled on node 1)

Processor 1 DramHoleOffset [31:24] = 00b (Memory hoisting is not enabled on node 1)

Processor 1 DramHoleValid = 0 (Memory hoisting disabled)

If the calculated DramHoleOffset is equal to 4GB, meaning the DramBase of the current node is aligned to the starting address of the MMIO hole, the DRAM Hole Address Register must not be

enabled. Both the DramBase and DramLimit of current node, and for the following nodes, must be adjusted by adding the size of the MMIO hole.

Example. A two processor system with 2GB memory on each processor and a 2GB MMIO hole.

Processor 0 DramBase[39:24] = 0000h

Processor 0 DramLimit[39:24] = 007Fh (2GB - 1)

Processor 0 DramHoleBase[31:24] = 00h

Processor 0 DramHoleOffset[31:24] = 00h (calculated DramHoleOffset = 4GB)

Processor 0 DramHoleValid = 0 (Memory hoisting disabled)

Processor 1 DramBase[39:24] = 0100h (4GB; DramBase + hole size)

Processor 1 DramLimit[39:24] = 017Fh (6GB - 1; DramLimit + hole size)

Processor 1 DramHoleBase[31:24] = 00h

Processor 1 DramHoleOffset[31:24] = 00h

Processor 1 DramHoleValid = 0 (Memory hoisting disabled)

3.6.5.2 Node-Interleave Mode

For a system with the node-interleaving enabled, the DRAM Hole Address Register is enabled on every node with the same setting in its DramHoleBase and DramHoleOffset fields. The DramLimit of every node in the DRAM address map must be adjusted by adding the size of the MMIO hole.

Example. A two processor system with 2GB memory on each processor and a 2GB MMIO hole with the node-interleave mode enabled.

Processor 0 DramBase[39:24] = 0000h

Processor 0 DramLimit[39:24] = 017Fh (6GB - 1; DramLimit + hole size)

Processor 0 DramHoleBase[31:24] = 80h (2G; Starting address of MMIO hole)

Processor 0 DramHoleOffset[31:24] = 80h(Calculated DramHoleOffset = 2GB)

Processor 0 DramHoleValid = 1

Processor 1 DramBase[39:24] = 0000h (Same as processor 0)

Processor 1 DramLimit[39:24] = 017Fh (Same as processor 0)

Processor 1 DramHoleBase[31:24] = 80h (Same as processor 0)

Processor 1 DramHoleOffset[31:24] = 80h (Same as processor 0)

Processor 1 DramHoleValid = 1

3.6.6 On-Line Spare

On-line spare is a RAS mechanism supported in F(1207), ASB1, and AM2 packages that allows the system to reserve one rank of one logical DIMM to be used as a spare rank. System software reserves

a spare rank by setting the spare bit (Function 2 Offsets 40h, 44h, 48h, 4Ch, 50h, 54h, 58h, 5Ch bit 1) in one of the CS Base address registers. The spare rank must be greater than or equal to the size of all other ranks in the system. On-line spare should never be used in a UMA system.

The system can switch to the spare rank when system software determines that one of the ranks in the system is no longer functioning properly and needs to be replaced. The system software initiates the swap to the spare rank by writing the chip select number of the bad rank to BadDramCS (Function 3 Offset B0h bits 6:4) and setting the swap enable SwapEn (Function 3 Offset B0h bit 0).

3.6.6.1 On-Line Spare and CS Interleaving

The on-line spare feature can only be used with 2 way and 4 way CS interleaving under the following conditions.

- All ranks of each DIMM present must be of the same size and configuration.
- Only the following populations are supported:
  - 2 DIMMs per channel (2 way CS interleaving)
    - One single rank DIMM and one dual rank DIMM. Any rank can be used as the spare rank.
    - Both DIMMs are dual rank. Any rank can be used as the spare rank. One rank must be marked as bad since only two ranks can be active.
  - 3 DIMMs per channel (4 way CS interleaving)
    - Two dual rank DIMMs and one single rank DIMM. Any rank can be used as the spare rank.
    - All DIMMs are dual rank. Any rank can be used as the spare rank. One rank must be marked as bad since only four ranks can be active.
  - 4 DIMMs per channel (4 way CS interleaving)
    - One dual rank DIMMs and three single rank DIMMs. Any rank can be used as the spare rank.
    - Two dual rank DIMMs and two single rank DIMMs. Any rank can be used as the spare rank. One rank must be marked as bad since only four ranks can be active.

3.7 Power Management

The processor provides the following power management features designed to be compliant with the Advanced Configuration and Power Interface (ACPI) Specification and HyperTransport™ technology: For more details see Chapter 10, "Power and Thermal Management" on page 293

- Halt state with associated programmable power savings
- STPCLK/Stop Grant protocol capable of supporting eight distinct versions of Stop Grant
- LDTSTOP_L signal support
- Memory controller and host bridge power management
- Processor Performance state (P-state) transition support

- Voltage plane isolation based upon PWROK signal
- Low-power state while RESET_L signal is asserted
- Hardware Thermal Control
- Software Thermal Control

3.7.1 PWROK

When PWROK is deasserted, the processor performs the following steps:

- Isolates it's VDDIO- and VTT-powered logic from all other internal logic to prevent leakage current paths between power planes.
- Tristates all DDR SDRAM I/O pins except for the M[B,A]_CKE and M[B,A]_RESET_L outputs, which are driven Low.
- Drives it's VID[5:0] outputs to the value that selects the startup core voltage level.

3.7.1.1 RESET_L and M[B,A]_RESET_L

When RESET_L is asserted, the processor performs the following steps:

- The processor core is held in a low-power state.
- The M[B,A]_CKE and M[B,A]_RESET_L outputs are forced low.

After RESET_L is deasserted, BIOS must program the appropriate clock divisor in the memory controller configuration registers, causing the M[B,A]_CLK_H/L clocks to be driven.

3.7.2 Thermal Sensor

Each processor core contains two thermal sensors. The temperature calculated by the thermal sensor is used by HTC, STC, THERMTRIP_L, and the PROCHOT_L signal.

3.7.2.1 Software Thermal Control

Software Thermal Control (STC) is a software mechanism used to reduce power consumption. When the die temperature becomes greater than the high temperature threshold, or when the die temperature becomes less than the low temperature threshold after the high temperature threshold was exceeded, an interrupt can be generated. There are two ways to generate an interrupt: (1) via APIC Thermal Local Vector Table Register that allows software to define the interrupt type, or (2) via a special bus cycle, and the I/O hub. Inside the interrupt handler software may take an action that will result in power consumption reduction. For example, it can turn the fan on/off, enable normal StpClk throttling, or do a FID/VID change.

3.7.2.2 Hardware Thermal Control

Hardware Thermal Control (HTC) is a hardware mechanism used to reduce processor power consumption. The processor enters HTC active state when the die temperature is greater than the HTC temperature limit or when PROCHOT_L is asserted. While it is in HTC active state, processor core clock grid (GCLK) is periodically ramped down for a specified period of time, and consequently processor performance is reduced. Deterministic power savings are accomplished by unconditional GCLK ramping down. After the temperature drops below the HTC temperature limit (plus a specified number of degrees of hysteresis), the processor automatically exits the HTC-active state if PROCHOT_L is not asserted. The processor can enter the HTC-active state only when PWROK and RESET_L are both high. The processor can not enter the HTC-active state while in the C3 ACPI state. Special bus cycles can be generated when HTC active state is entered or exited. The Northbridge clock grid (BCLK) is not ramped down when HTC is active.

3.7.2.3 PROCHOT_L

The PROCHOT_L pin is asserted as an input to force the processor into the HTC-active state or becomes asserted as an output to indicate when the die temperature is greater than the HTC temperature limit.

As in input to the processor, PROCHOT is ignored if THERMTRIP_L, PWROK or RESET_L is low or if HtcEn (Function 3 Offset 64h) is zero. As an output, it is low to reflect the HTC-active state.

PROCHOT_L will be recognized when it's state has been stable high or low for greater than 10ns, however to get any effective power reduction from PROCHOT_L assertion, the assertion time must allow for several throttling periods to elapse. Each throttling period is approximately 6us. See "BIOS Requirements for PROCHOT_L Throttling" on page 298 for how to enable PROCHOT_L based throttling.

3.7.2.4 THERMTRIP_L

The processor provides a hardware-enforced thermal protection mechanism. When the processor's die temperature exceeds a specified temperature, the processor is designed to stop its internal clocks and assert the THERMTRIP_L output.

THERMTRIP_L assertion is only valid when PWROK is asserted and RESET_L is deasserted.

THERMTRIP_L assertion indicates the processor die temperature has exceeded normal operating parameters. PWROK must be deasserted in response to a THERMTRIP_L assertion to enable proper processor operation.

Once asserted THERMTRIP_L remains asserted until RESET_L is asserted.

If the processor's die temperature still exceeds the thermal trip point when RESET_L is deasserted, THERMTRIP_L will immediately be reasserted and the processor's internal clocks stop.

3.7.3 Thermal Diode

An on-die thermal diode is provided as a tool for thermal management. An external sensor is necessary to measure the temperature of the thermal diode.

Hardware thermal solutions for lidded processors should be designed and validated against the case temperature specification per the methodology specified in *AMD Athlon™ 64 and AMD Opteron™ Processors Thermal Design Guide*, order# 26633. Hardware thermal solutions for lidless processors should be designed against the die temperature specification in *Thermal Design and Testing Guide for AMD NPT Family 0Fh Processors in Mobile Systems*, order# 32769.

System thermal management (e.g. fan control) for lidded processors using the thermal diode should be designed against the case temperature specification. See "TCASE max and TCONTROL max for Socket F (1207) and Socket AM2 Processors" on page 49 for more information.

3.7.3.1 T_{CASE} max and $T_{CONTROL}$ max for Socket F (1207) and Socket AM2 Processors

T_{CASE} max is the maximum case temperature specification for a lidded processor. Thermal solutions should be designed to this specification. The thermal test procedure is defined in *AMD Athlon™ 64 and AMD Opteron™ Processors Thermal Test Kit Instruction Manual,* order# 26956.

T_{CASE} max is a physical temperature specification in degrees Celsius that can be measured at the center of the lid with a thermocouple. The correct method for measuring case temperature is discussed in the *AMD Athlon™ 64 and AMD Opteron™ Processors Thermal Design Guide, order# 26633*. The case temperature specification is provided in the power and thermal data sheet (*AMD NPT Family 0Fh Server and Workstation Processor Power and Thermal Data Sheet*, order# 33953 and *AMD NPT Family 0Fh Desktop Processor Power and Thermal Data Sheet*, order# 33954).

$T_{CONTROL}$ max represents the maximum allowed $T_{CONTROL}$ value for the processor to be within its functional temperature specification. The value for $T_{CONTROL}$ max is provided in the power and thermal data sheet.

$T_{CONTROL}$ (processor control temperature) is a non physical temperature on an arbitrary scale that can be used for system thermal management policies. $T_{CONTROL}$ does not equal the physical temperature of the die, lid, or any other temperature. Under low power conditions, $T_{CONTROL}$ can be less than the ambient temperature. Negative $T_{CONTROL}$ values are undefined.

$T_{CONTROL}$= dual sourcing current temperature measurement + T_{OFFSET}

T_{OFFSET} (thermal diode temperature offset) is supplied as a read only field in the Thermtrip Status Register.

System thermal management should be designed to prevent the case temperature from being exceeded even in transient situations. For example if the processor is in an ACPI C1 Halt state with a low fan speed and a high power application is started, the fan speed policy must ensure that the processor $T_{CONTROL}$ never exceeds the $T_{CONTROL}$ max limit. This requires increasing the fan speed

before reaching $T_{CONTROL}$ max. Additionally, the accuracy of the thermal diode monitor, typically ±1 C to 5 C, must be considered when setting thermal trip points.

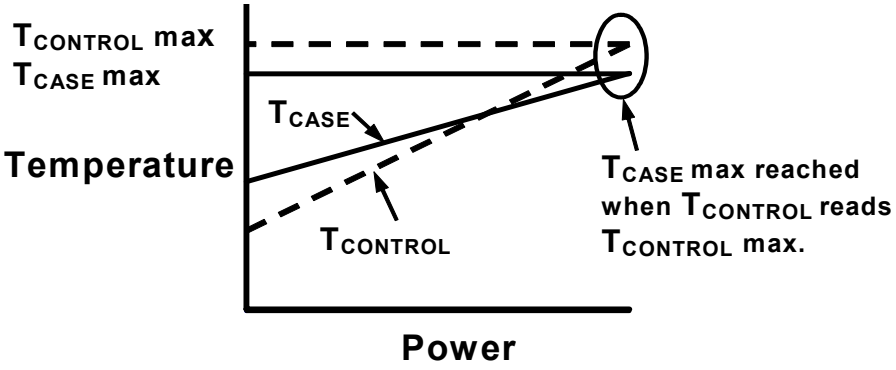


Figure 2. T_{CASE} Max and $T_{CONTROL}$ Max Relationship

3.7.3.2 T_{DIE} max and $T_{CONTROL}$ for Socket S1g1 and ASB1 Processors

T_{DIE} max is the maximum die temperature specification for a lidless processor. Thermal solutions should be designed to this specification. T_{DIE} max is a physical temperature specification in degrees Celsius that can be measured at the die with a thermocouple. The correct method for measuring die temperature is discussed in the *Thermal Design and Testing Guide for AMD NPT Family 0Fh Processors in Mobile Systems*, order# 32769.

$T_{CONTROL}$ = dual sourcing current temperature measurement + T_{OFFSET}

T_{OFFSET} (thermal diode temperature offset) is supplied as a read only field in the Thermtrip Status Register. For lidless processors, this is the ideality factor which calibrates the monitor reading to the actual die temperature.

System thermal management should be designed to prevent the processor control temperature from exceeding the maximum die temperature even in transient situations. For example if the processor is in an ACPI C1 Halt state with a low fan speed and a high power application is started, the fan speed policy must ensure that the processor $T_{CONTROL}$ never exceeds the T_{DIE} max limit. This requires increasing the fan speed before reaching T_{DIE} max. Additionally, the accuracy of the thermal diode monitor, typically ±1 C to 5 C, must be considered when setting thermal trip points.

4 Memory System Configuration

4.1 Configuration Space Accesses

The AMD NPT Family 0Fh Processors implement configuration space as defined in the PCI Local Bus Specification, Rev. 2.2, and the HyperTransport™ I/O Link Specification, Rev. 1.03.

Both coherent HyperTransport links and the compatibility bus (the bus to the HyperTransport I/O hub) are accessed through Bus 0. Configuration accesses to Bus 0, devices 0 to 23, are transmitted to the compatibility noncoherent HyperTransport bus. Coherent HyperTransport device configuration space is accessed by using Bus 0, devices 24 to 31, where Device 24 corresponds to Node 0 and Device 31 corresponds to Node 7.

All configuration cycles are type 1 on coherent HyperTransport links. When transmitted down a noncoherent HyperTransport chain by the host bridge, configuration cycles are translated to type 0 if they target the noncoherent HyperTransport bus immediately behind the host bridge. If they target a subordinate bus, they remain as type 1.

Accesses to memory system configuration registers are controlled through the Configuration Address register (see "Configuration Address Register" on page 51) and the Configuration Data register (see "Configuration Data Register" on page 52), which can be accessed through I/O reads/writes to addresses 0CF8h and 0CFCh, respectively.

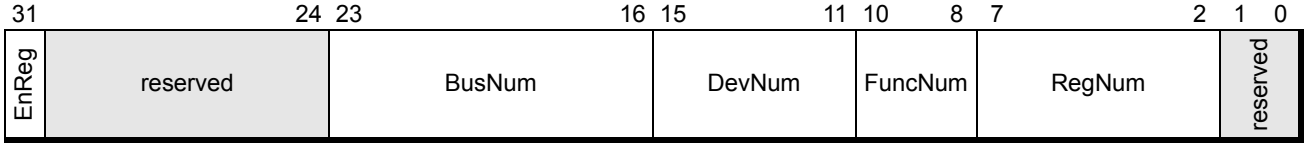
4.1.1 Configuration Address Register

Writes to the Configuration Address register specify the target of a configuration access in terms of the bus, device, function, and register. Access to the Configuration Address register must be full doubleword reads or writes.

To access one of the memory system configuration registers defined in this chapter, the bus number should be 0, the device number should be the target processor node number plus 24, and the function and register number should be set based on the register function and offset as specified in this chapter.

When the Enable bit of the Configuration Address register is set, reads and writes to the Configuration Data register access the register specified in the Configuration Address register.

Configuration Address Register **0CF8h (doubleword)**



Bit	Mnemonic	Function	R/W
31	EnReg	Enable Register	R/W
30–24	reserved		R
23–16	BusNum	Bus Number	R/W
15–11	DevNum	Device Number	R/W
10–8	FuncNum	Function Number	R/W
7–2	RegNum	Register Number	R/W
1–0	reserved		R

Field Descriptions

Register Number (RegNum)—Bits 7–2. Specifies the doubleword offset of the configuration address.

Function Number (FuncNum)—Bits 10–8. Specifies the function number of the configuration address.

Device Number (DevNum)—Bits 15–11. Specifies the device number of the configuration address.

Bus Number (BusNum)—Bits 23–16. Specifies the bus number of the configuration address.

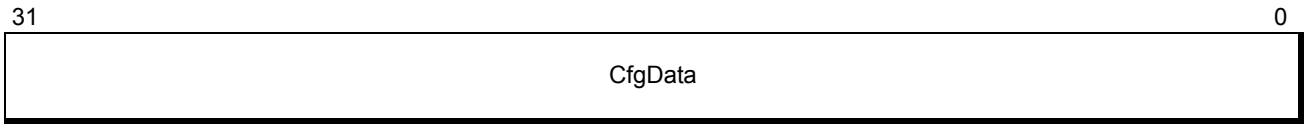
Enable (EnReg)—Bit 31. When this bit is set, aligned doubleword accesses to the Configuration Data Register result in configuration-space transactions.

0 = Configuration transactions disabled.
1 = Configuration transactions enabled.

4.1.2 Configuration Data Register

Accesses to the Configuration Data Register are translated to configuration-space transactions at the address specified by the Configuration Address Register.

Configuration Data Register 0CFCh (Doubleword)



Bits	Mnemonic	Function	R/W
31–0	CfgData	Configuration Data	R/W

Field Descriptions

Configuration Data (CfgData)—Bits 31–0.

4.2 Memory System Configuration Registers

The AMD NPT Family 0Fh Processor implement memory system configuration registers in the "Northbridge" block of the processor. These registers are mapped into the coherent HyperTransport technology configuration space (Bus 0, device numbers 24–31). The number of devices implemented is equal to the number of processor nodes in the system. Configuration space Device 24 corresponds to Node 0 and is normally the bootstrap processor (BSP). Configuration space device numbers 25 through 31 correspond to nodes 1 through 7, which may or may not be implemented.

The processor implements configuration registers in PCI configuration space using the following four headers:

- Function 0: HyperTransport technology configuration (see page 53)
- Function 1: Address map configuration (see page 80)
- Function 2: DRAM configuration (see page 95)
- Function 3: Miscellaneous configuration (see page 136)

Each of these functions are detailed in the following sections.

Note: *Contents of some fields in the headers and HyperTransport technology capability blocks are maintained through a warm reset. If not specified otherwise, a field is initialized by a warm reset.*

4.3 Function 0—HyperTransport™ Technology Configuration

Table 5. Function 0 Configuration Registers

Offset	Register Name				Reset	Access	Description
00h	Device ID		Vendor ID (AMD)		1100_1022h	RO	page 55
04h	Status		Command[1]		0010_0000h	RO	page 56
08h	Base Class Code	Subclass Code	Program Interface	Revision ID	0600_0000h	RO	page 56
0Ch	BIST	Header Type	Latency Timer	Cache Line Size	0080_0000h	RO	page 57
10h	Base Address 0				0000_0000h	RO	

Notes:
1. *The unimplemented registers in the standard PCI configuration space are implemented as read-only and return 0 if read.*
2. *Reads and writes to unimplemented registers in the extended PCI configuration space will result in unpredictable behavior.*

Table 5. Function 0 Configuration Registers (Continued)

Offset	Register Name				Reset	Access	Description
14h	Base Address 1				0000_0000h	RO	
18h	Base Address 2				0000_0000h	RO	
1Ch	Base Address 3				0000_0000h	RO	
20h	Base Address 4				0000_0000h	RO	
24h	Base Address 5				0000_0000h	RO	
28h	Card Bus CIS Pointer				0000_0000h	RO	
2Ch	Sub-System ID		Sub-System Vendor ID		0000_0000h	RO	
30h	ROM Base Address				0000_0000h	RO	
34h	Capabilities Pointer				page 57	RO	page 57
38h	reserved				0000_0000h	RO	
3Ch	Max Latency	Min GNT	Int Pin	Int Line	0000_0000h	RO	
40h	Routing Table Node 0				0001_0101h	RW	page 58
44h	Routing Table Node 1				0001_0101h	RW	page 58
48h	Routing Table Node 2				0001_0101h	RW	page 58
4Ch	Routing Table Node 3				0001_0101h	RW	page 58
50h	Routing Table Node 4				0001_0101h	RW	page 58
54h	Routing Table Node 5				0001_0101h	RW	page 58
58h	Routing Table Node 6				0001_0101h	RW	page 58
5Ch	Routing Table Node 7				0001_0101h	RW	page 58
60h	Node ID				0000_0000h	RW	page 60
64h	Unit ID				0000_00E4h	RW	page 61
68h	HyperTransport™ Transaction Control				0F00_0000h	RW	page 62
6Ch	HyperTransport™ Initialization Control				page 67	RW	page 67
80h	LDT0 Capabilities				page 68	RO	page 68
84h	LDT0 Link Control				0011_0000h	RW	page 70
88h	LDT0 Frequency/Revision				page 74	RW	page 74
8Ch	LDT0 Feature Capability				0000_0002h	RO	page 75
90h	LDT0 Buffer Count				page 76	RW	page 76
94h	LDT0 Bus Number				0000_0000h	RW	page 78
98h	LDT0 Type				page 78	RO	page 78
A0h	LDT1 Capabilities				page 68	RO	page 68
A4h	LDT1 Link Control				0011_0000h	RW	page 70

Notes:

1. *The unimplemented registers in the standard PCI configuration space are implemented as read-only and return 0 if read.*
2. *Reads and writes to unimplemented registers in the extended PCI configuration space will result in unpredictable behavior.*

Table 5. Function 0 Configuration Registers (Continued)

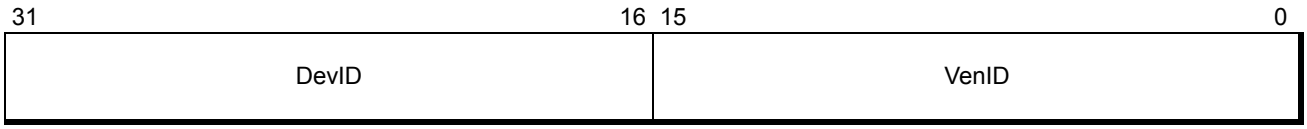
Offset	Register Name	Reset	Access	Description
A8h	LDT1 Frequency/Revision	page 74	RW	page 74
ACh	LDT1 Feature Capability	0000_0002h	RO	page 75
B0h	LDT1 Buffer Count	page 76	RW	page 76
B4h	LDT1 Bus Number	0000_0000h	RW	page 78
B8h	LDT1 Type	page 78	RO	page 78
C0h	LDT2 Capabilities	page 68	RO	page 68
C4h	LDT2 Link Control	0011_0000h	RW	page 70
C8h	LDT2 Frequency/Revision	page 74	RW	page 74
CCh	LDT2 Feature Capability	0000_0002h	RO	page 75
D0h	LDT2 Buffer Count	page 76	RW	page 76
D4h	LDT2 Bus Number	0000_0000h	RW	page 78
D8h	LDT2 Type	page 78	RO	page 78

Notes:
1. *The unimplemented registers in the standard PCI configuration space are implemented as read-only and return 0 if read.*
2. *Reads and writes to unimplemented registers in the extended PCI configuration space will result in unpredictable behavior.*

4.3.1 Device/Vendor ID Register

This register specifies the device and vendor IDs for the Function 0 registers and is part of the standard PCI configuration header.

Device/Vendor ID Register **Function 0: Offset 00h**



Bits	Mnemonic	Function	R/W	Reset
31–16	DevID	Device ID	R	1100h
15–0	VenID	Vendor ID	R	1022h

Field Descriptions

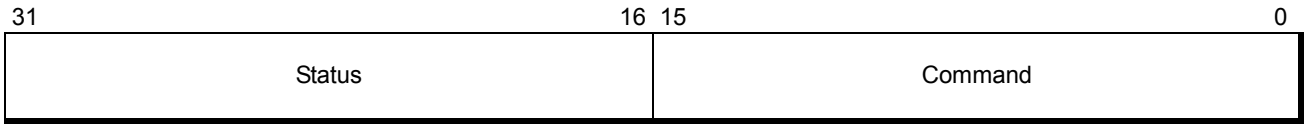
Vendor ID (VenID)—Bits 15–0. This read-only value is defined as 1022h for AMD.

Device ID (DevID)—Bits 31–16. This read-only value is defined as 1100h for the HyperTransport technology configuration function.

4.3.2 Status/Command Register

This register contains status and command information for the Function 0 registers and is part of the standard PCI configuration header.

Status/Command Register — Function 0: Offset 04h



Bits	Mnemonic	Function	R/W	Reset
31–16		Status	R	0010h
15–0		Command	R	0000h

Field Descriptions

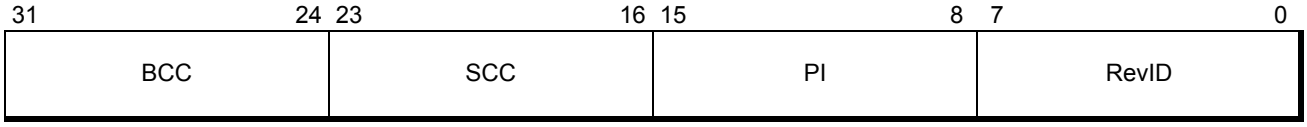
Command—Bits 15–0. This read-only value is defined as 0000h.

Status—Bits 31–16. This read-only value is defined as 0010h to indicate that the processor has a capabilities list containing configuration information specific to HyperTransport technology.

4.3.3 Class Code/Revision ID Register

This register specifies the class code and revision for the Function 0 registers and is part of the standard PCI configuration header.

Class Code/Revision ID Register — Function 0: Offset 08h



Bits	Mnemonic	Function	R/W	Reset
31–24	BCC	Base Class Code	R	06h
23–16	SCC	Subclass Code	R	00h
15–8	PI	Programming Interface	R	00h
7–0	RevID	Revision ID	R	00h

Field Descriptions

Revision ID (RevID)—Bits 7–0.

Programming Interface (PI)—Bits 15–8. This read-only value is defined as 00h.

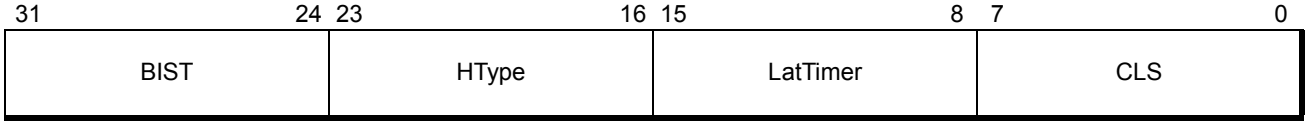
Sub Class Code (SCC)—Bits 23–16. This read-only value is defined as 00h.

Base Class Code (BCC)—Bits 31–24. This read-only value is defined as 06h for a host bridge device.

4.3.4 Header Type Register

This register specifies the header type for the Function 0 registers and is part of the standard PCI configuration header.

Header Type Register **Function 0: Offset 0Ch**



Bits	Mnemonic	Function	R/W	Reset
31–24	BIST	BIST	R	00h
23–16	HType	Header Type	R	80h
15–8	LatTimer	Latency Timer	R	00h
7–0	CLS	Cache Line Size	R	00h

Field Descriptions

CacheLineSize (CLS)—Bits 7–0. This read-only value is defined as 00h.

LatencyTimer (LatTimer)—Bits 15–8. This read-only value is defined as 00h.

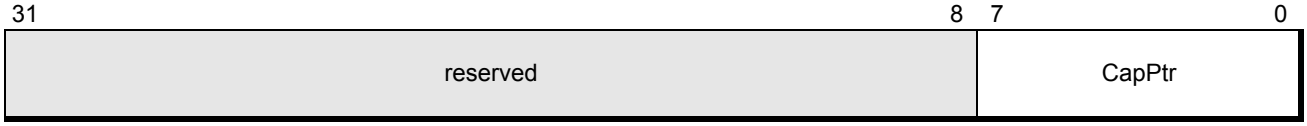
HeaderType (HType)—Bits 23–16. This read-only value is defined as 80h to indicate that multiple functions are present in the configuration header and that the header layout corresponds to a device header as opposed to a bridge header.

BIST—Bits 31–24. This read-only value is defined as 00h.

4.3.5 Capabilities Pointer Register

This register contains a capabilities pointer to a linked capabilities list of registers specific to HyperTransport technology, with one set for each HyperTransport link. It is part of the standard PCI configuration header.

Capabilities Pointer Register **Function 0: Offset 34h**



Bits	Mnemonic	Function	R/W	Reset
31–8	reserved		R	0
7–0	CapPtr	Capability Pointer	R	

Field Descriptions

Capability Pointer (CapPtr)—Bits 7–0. Points to the first available HyperTransport technology capabilities block. Depending on the specific HyperTransport link configuration physically present on the processor, this will be either 80h (LDT0), A0h (LDT1) or C0h (LDT2).

4.3.6 Routing Table Node *i* Registers

The routing table contains three entries—one for requests RQRoute[7:0], one for responses RPRoute[7:0], and one for broadcasts BCRoute[7:0].

A maximum of eight nodes with three links per node is supported in an AMD Opteron™ system.

These table entries can be read and written, and its contents are not maintained through a warm reset. At reset, all table entries are initialized to the value 01h, indicating that packets should be accepted by this node. Care must be exercised when changing table entries after reset to ensure that connectivity is not lost during the process.

4.3.6.1 Request Routing Table

Each node contains a routing table for use with directed requests, with one entry for each of the eight possible destination Node IDs. The value in each entry indicates which outgoing link is used for request packets directed to that particular destination node (DstNode). A 1 in a given bit position indicates that the request is routed through the corresponding output link. Bit 0, when set to 1, indicates that the request must be accepted by this node.

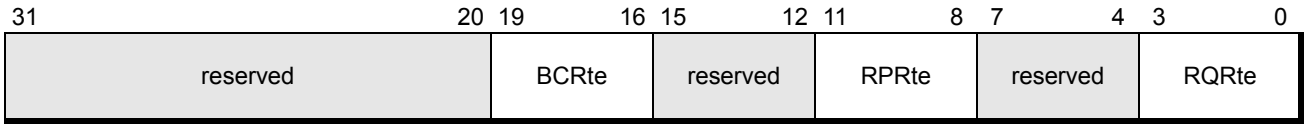
4.3.6.2 Response Routing Table

Each node contains a routing table for use with responses, with one entry for each of the eight possible destination node IDs. The value in each entry indicates which outgoing link is used for response packets directed to that particular destination node (DstNode). A 1 in a given bit position indicates that the response is routed through the corresponding output link. Bit 0, when set to 1, indicates that the response must be accepted by this node.

4.3.6.3 Broadcast Routing Table

Each node contains a routing table for use with broadcast and probe requests with one entry for each of the eight possible source node IDs. Each entry contains a single bit for each of the outbound links from the node. The packet is forwarded on all links with their corresponding links set to a 1. Bit 0 when set to 1 indicates that the broadcast must be accepted by this node. The algorithm to generate the routing table follows the backbone first rule.

Routing Table Node 0–7 Registers — Function 0: Offset 40h, 44h, 48h, 4Ch, 50h, 54h, 58h, 5Ch



Bits	Mnemonic	Function	R/W	Reset
31–20	reserved		R	000h
19–16	BCRte	Broadcast Route	R/W	1h
15–12	reserved		R	0h
11–8	RPRte	Response Route	R/W	1h
7–4	reserved		R	0h
3–0	RQRte	Request Route	R/W	1h

Field Descriptions

Request Route (RQRte)—Bits 3–0. The Request Route field defines the node or link to which a request packet is forwarded. Request packets are routed to only one destination and the table index is based on the destination Node ID.

Bit [0] = Route to this node
Bit [1] = Route to Link 0
Bit [2] = Route to Link 1
Bit [3] = Route to Link 2

Response Route (RPRte)—Bits 11–8. The Response Route field defines the node or link to which a response packet is forwarded. Response packets are routed to only one destination and the table index is based on the destination Node ID.

Bit [0] = Route to this node
Bit [1] = Route to Link 0
Bit [2] = Route to Link 1
Bit [3] = Route to Link 2

Broadcast Route (BCRte)—Bits 19–16. The Broadcast Route field defines the node or link(s) to which a broadcast packet is forwarded. Broadcasts may be routed to more than one destination. The Node ID that indexes into the table is the source Node ID.

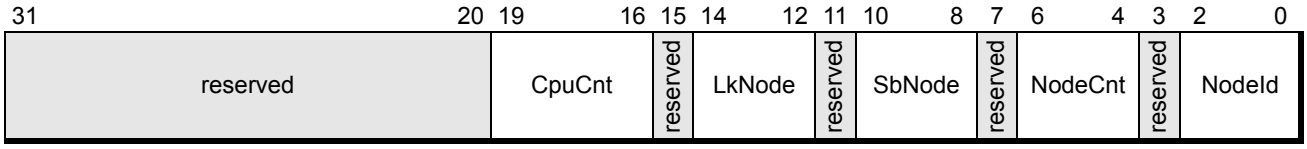
Bit [0] = Route to this node
Bit [1] = Route to Link 0
Bit [2] = Route to Link 1
Bit [3] = Route to Link 2

4.3.7 Node ID Register

The Node ID register contains node ID, node count and CPU core count information (a node can have more than one CPU core).

It is expected that system configuration software will program the Node ID as part of coherent HyperTransport link configuration. Correct system operation depends on an assignment of distinct node ID values not exceeding NodeCnt[2:0]. For example, the Node IDs in a 4-node system must be 0, 1, 2, and 3; an example of an incorrect Node ID assignment in this system is 0, 1, 3, and 4. The Node ID will also be used as the initial APIC ID for the local APIC.

Node ID Register **Function 0: Offset 60h**



Bits	Mnemonic	Function	R/W	Reset
31–20	reserved		R	0
19–16	CPUCnt	CPU Count	R/W	0
15	reserved		R	0
14–12	LkNode	Lock Controller Node ID	R/W	0
11	reserved		R	0
10–8	SbNode	HyperTransport™ I/O Hub Node ID	R/W	0
7	reserved		R	0
6–4	NodeCnt	Node Count	R/W	0
3	reserved		R	0
2–0	NodeID	This Node ID	R/W	

Field Descriptions

Node ID (NodeID)—Bits 2–0. Defines the Node ID of this node. It resets to 0 for the boot strap processor (BSP) and to 7h for all other nodes.

Node Count (NodeCnt)—Bits 6–4. Specifies the number of coherent nodes in the system. Note that the hardware allows only values to be programmed into this field that are consistent with the multiprocessor capabilities of the device, as specified in "Northbridge Capabilities Register" on page 190. This field will not be updated if there is an attempt to write a value that is inconsistent with the specified multiprocessor capabilities.

000b = 1 node
001b = 2 nodes
010b = 3 nodes
011b = 4 nodes
101b = 6 nodes

111b = 8 nodes

HyperTransport I/O Hub Node ID (SbNode)—Bits 10–8. Defines the Node ID of the node that owns the HyperTransport link to the HyperTransport I/O hub.

Lock Controller Node ID (LkNode)—Bits 14–12. Defines the Node ID of the node that contains the Lock Controller. The lock controller node (LkNode) must be the same as the HyperTransport I/O hub node (SbNode).

CPU Count (CPUCnt)—Bits 19–16. Defines the number of CPU cores in the system.

0000b = 1 CPU

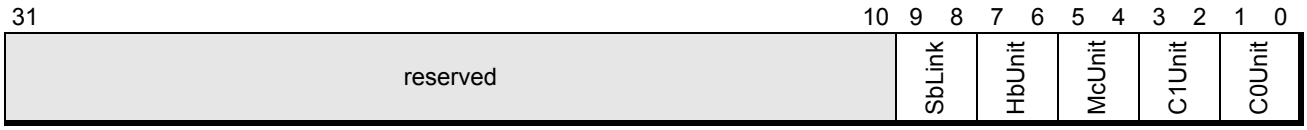
...

1111b = 16 CPUs

4.3.8 Unit ID Register

The Unit ID register specifies the Unit ID of each functional unit on the processor. Under normal operation, there is no need for software to change the default Unit ID assignments. This register also contains a pointer to the HyperTransport I/O hub link.

Unit ID Register **Function 0: Offset 64h**



Bits	Mnemonic	Function	R/W	Reset
31–10	reserved		R	0
9–8	SbLink	HyperTransport™ I/O Hub Link ID	R/W	00b
7–6	HbUnit	Host Bridge Unit ID	R/W	11b
5–4	McUnit	Memory Controller Unit ID	R/W	10b
3–2	C1Unit	CPU1 Unit ID	R/W	01b
1–0	C0Unit	CPU0 Unit ID	R/W	00b

Field Descriptions

CPU0 Unit ID (C0Unit)—Bits 1–0. Defines the Unit ID of CPU0 core.

CPU1 Unit ID (C1Unit)—Bits 3–2. Defines the Unit ID of CPU1 core, if it is implemented.

Memory Controller Unit ID (McUnit)—Bits 5–4. Defines the Unit ID of the memory controller.

Host Bridge Unit ID (HbUnit)—Bits 7–6. Defines the Unit ID of the host bridge. The host bridge is used to bridge between coherent and noncoherent HyperTransport technology domains.

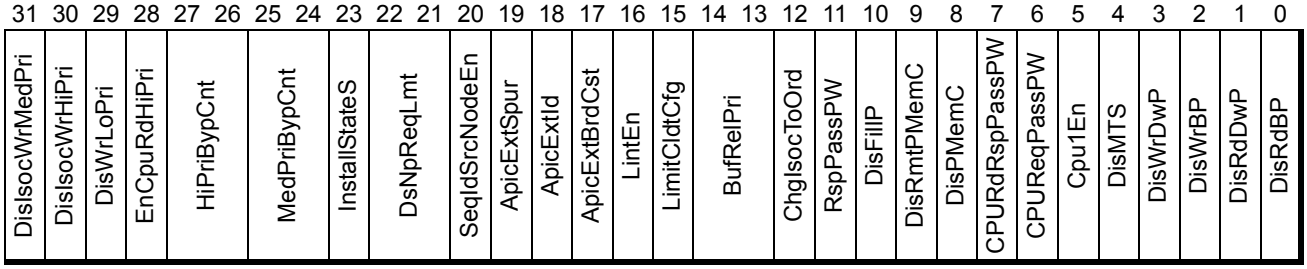
HyperTransport I/O Hub Link ID (SbLink)—Bits 9–8. Defines the link to which the HyperTransport I/O hub is connected. It is only used by the node which owns the HyperTransport I/O hub, as indicated in the Node ID register.

00b = LDT0 (default)
01b = LDT1
10b = LDT2
11b = Undefined

4.3.9 HyperTransport™ Transaction Control Register

The HyperTransport Transaction Control register configures the high-level HyperTransport protocols. It can be used to optimize system performance or change HyperTransport technology transaction protocols.

HyperTransport™ Transaction Control Register — Function 0: Offset 68h



Bits	Mnemonic	Function	R/W	Reset
31	DisIsocWrMedPri	Disable medium priority isochronous writes	R/W	0
30	DisIsocWrHiPri	Disable high priority isochronous writes	R/W	0
29	DisWrLoPri	Disable low priority writes	R/W	0
28	EnCpuRdHiPri	Enable high priority CPU reads	R/W	0
27–26	HiPriBypCnt	High-priority bypass count	R/W	11b
25–24	MedPriBypCnt	Medium-priority bypass count	R/W	11b
23	InstallStateS	Read Block Cache Install State Shared	R/W	0
22–21	DsNpReqLmt	Downstream non-posted request limit	R/W	00b
20	SeqIdSrcNodeEn	Sequence ID source node enable	R/W	0
19	ApicExtSpur	APIC extended spurious vector enable	R/W	0
18	ApicExtId	APIC extended ID enable	R/W	0
17	ApicExtBrdCst	APIC extended broadcast enable	R/W	0
16	LintEn	Local interrupt conversion enable	R/W	0
15	LimitCldtCfg	Limit coherent HyperTransport™ configuration space range	R/W	0
14–13	BufRelPri	Buffer release priority select	R/W	00b
12	ChgIsocToOrd	Change ISOC to Ordered	R/W	0
11	RspPassPW	Response PassPW	R/W	0
10	DisFillP	Disable fill probe	R/W	0

Bits	Mnemonic	Function	R/W	Reset
9	DisRmtPMemC	Disable remote probe memory cancel	R/W	0
8	DisPMemC	Disable probe memory cancel	R/W	0
7	CPURdRspPassPW	CPU Read response PassPW	R/W	0
6	CPUReqPassPW	CPU request PassPW	R/W	0
5	Cpu1En	CPU1 enable	R/W	0
4	DisMTS	Disable memory controller target start	R/W	0
3	DisWrDwP	Disable write doubleword probes	R/W	0
2	DisWrBP	Disable write byte probes	R/W	0
1	DisRdDwP	Disable read doubleword probe	R/W	0
0	DisRdBP	Disable read byte probe	R/W	0

Field Descriptions

Disable Read Byte Probe (DisRdBP)—Bit 0. This bit determines if probes are issued for CPU-generated RdSized byte (must be 0 for multi-CPU core systems; recommended to be 1 for uniprocessor, single CPU core systems)

0 = Probes issued
1 = Probes not issued

Disable Read Doubleword Probe (DisRdDwP)—Bit 1. This bit determines if probes are issued for CPU-generated RdSized Doubleword (must be 0 for multi-CPU core systems; recommended to be 1 for uniprocessor, single CPU core systems)

0 = Probes issued
1 = Probes not issued

Disable Write Byte Probes (DisWrBP)—Bit 2. This bit determines if probes are issued for CPU-generated WrSized byte (must be 0 for multi-CPU core systems; recommended to be 1 for uniprocessor, single CPU core systems)

0 = Probes issued
1 = Probes not issued

Disable Write Doubleword Probes (DisWrDwP)—Bit 3. This bit determines if probes are issued for CPU-generated WrSized Doubleword (must be 0 for multi-CPU core systems; recommended to be 1 for uniprocessor, single CPU core systems)

0 = Probes issued
1 = Probes not issued

Disable Memory Controller Target Start (DisMTS)—Bit 4. Disables use of TgtStart. TgtStart is used to improve scheduling of back-to-back ordered transactions by indicating when the first transaction is received and ordered at the memory controller.

0 = TgtStart packets are generated
1 = TgtStart packets are not generated

CPU1 Enable (Cpu1En)—Bit 5. Enables a second CPU core, if it is implemented on the node. When this bit is set the second CPU core begins fetching code. This bit should only be set for nodes which have two CPU cores.

0 = Second CPU core disabled or not present
1 = Second CPU core enabled

CPU Request PassPW (CPUReqPassPW)—Bit 6. Allows CPU-generated requests to pass posted writes.

0 = CPU requests do not pass posted writes
1 = CPU requests pass posted writes

CPU Read Response PassPW (CPURdRspPassPW)—Bit 7. Allows RdResponses to CPU-generated reads to pass posted writes.

0 = CPU responses do not pass posted writes
1 = CPU responses pass posted writes

Disable Probe Memory Cancel (DisPMemC)—Bit 8. Disables generation of the MemCancel command when a probe hits a dirty cache block. MemCancels are used to attempt to save DRAM and/or HyperTransport™ technology bandwidth associated with the transfer of stale DRAM data.

0 = Probes may generate MemCancels
1 = Probes may not generate MemCancels

Disable Remote Probe Memory Cancel (DisRmtPMemC)—Bit 9. Disables generation of the MemCancel command when a probe hits a dirty cache block unless the probed cache is on the same node as the memory controller. MemCancels are used to attempt to save DRAM and/or HyperTransport technology bandwidth associated with the transfer of stale DRAM data.

0 = Probes hitting dirty blocks generate memory cancel packets, regardless of the location of the probed cache
1 = Only probed caches on the same node as the target memory controller generate memory cancel packets

Disable Fill Probe (DisFillP)—Bit 10. Disables probes for CPU-generated fills (must be 0 for multiprocessor or dual-core processor systems; recommended to be 1 for uniprocessor with a single core).

0 = Probes issued for cache fills
1 = Probes not issued for cache fills

Response PassPW (RspPassPW)—Bit 11. Causes the host bridge to set the PassPW bit in all downstream responses.

This technically breaks the PCI ordering rules but it is not expected to be an issue in the downstream direction. Setting this bit improves the latency of upstream requests by allowing the downstream responses to pass posted writes.

0 = Downstream response PassPW based on original request
1 = Downstream response PassPW set to 1

Change ISOC to Ordered (ChgIsocToOrd)—Bit 12. Causes bit 1 of RdSz/WrSz commands to be treated as an ordered bit in coherent HyperTransport technology devices, instead of as an isochronous bit. Setting this bit will disable prioritization of isochronous requests, but will enable tracking of ordered commands across coherent HyperTransport links. This bit should only be set if the performance of non-ordered peer-to-peer traffic across coherent HyperTransport links is optimized, or to disable isochronous prioritization.

0 = Bit 1 of coherent HyperTransport technology sized command used for isochronous prioritization
1 = Bit 1 of coherent HyperTransport technology sized command used for ordering

Buffer Release Priority Select (BufRelPri)—Bits 14–13. Selects the number of HyperTransport technology doublewords sent with a buffer release pending before the buffer release is inserted into the command/data stream of a busy link.

00b = 64 (default)
01b = 16
10b = 8
11b = 2

This should be set to a value of 10 (8 HyperTransport packets) to maximize HyperTransport technology bandwidth.

Limit Coherent HyperTransport Configuration Space Range (LimitCldtCfg)—Bit 15. Limits the extent of the coherent HyperTransport technology configuration space based on the number of nodes in the system. When this bit is set, configuration accesses that normally map to coherent HyperTransport space will be sent to noncoherent HyperTransport links instead, if these accesses attempt to access a non-existent node, as specified through the node count in the Node ID register. This bit should be set by BIOS once coherent HyperTransport fabric initialization is complete. Failure to do so will result in PCI configuration accesses to non-existent nodes being sent into the coherent HyperTransport routing fabric, causing the system to hang.

0 = No coherent HyperTransport configuration space restrictions
1 = Limit coherent HyperTransport configuration space based on number of nodes

Local Interrupt Conversion Enable (LintEn)—Bit 16. Enables the conversion of broadcast ExtInt/NMI HyperTransport technology interrupts to LINT0/1 before delivering to the local APIC. This conversion only takes place if the local APIC is hardware enabled.

0 = ExtInt/NMI interrupts unaffected
1 = ExtInt/NMI broadcast interrupts converted to LINT0/1

APIC Extended Broadcast Enable (ApicExtBrdCst)—Bit 17. Enables extended APIC broadcast functionality.

0 = APIC broadcast is 0Fh
1 = APIC broadcast is FFh

APIC Extended ID Enable (ApicExtId)—Bit 18. Enables extended APIC ID functionality.

0 = APIC ID is 4 bits
1 = APIC ID is 8 bits

APIC Extended Spurious Vector Enable (ApicExtSpur)—Bit 19. Enables extended APIC spurious vector functionality.

0 = Lower 4 bits of spurious vector are read-only 1111b
1 = Lower 4 bits of spurious vector are writable

Sequence ID Source Node Enable (SeqIdSrcNodeEn)—Bit 20. Causes the source node of requests to be used in the SeqID field of downstream noncoherent HyperTransport technology request packets. This may be desirable for some debug HyperTransport packet tracing applications, to match downstream packets with their originating CPU. For normal operation, this bit should be cleared, since this use of SeqID for source node determination may cause problems with packet ordering.

Downstream non-posted request limit (DsNpReqLmt)—Bits 22–21. Sets the number of downstream non-posted requests issued by CPU(s) which may be outstanding on the noncoherent HyperTransport links attached to this node. Non-posted requests from CPU(s) will be throttled such that they do not exceed the programmed value in this field.

00b = No limit
01b = Limited to 1
10b = Limited to 4
11b = Limited to 8

Read Block Cache Install State Shared (InstallStateS)—Bit 23. This bit controls the state that a cache line is installed at when the cache performs a read block. See "Register Differences in Revisions of AMD NPT Family 0Fh Processors" on page 27 for revision information about this field.

0 = Install state is dependent on probe results.
1 = Install state shared.

Medium-Priority Bypass Count (MedPriBypCnt)—Bits 25–24. The maximum number of times a medium priority access can pass a low priority access before medium priority mode is disabled for one access.

High-Priority Bypass Count (HiPriBypCnt)—Bits 27–26. The maximum number of times a high priority access can pass a medium or low priority access before high priority mode is disabled for one access.

Enable High Priority CPU Reads (EnCpuRdHiPri)—Bit 28. Enables CPU reads to be treated as high priority. CPU reads are treated as medium priority by default. Setting EnCpuRdHiPri changes their priority to high.

Disable Low Priority Writes (DisWrLoPri)—Bit 29. Disables non-isochronous writes from being treated as low priority. Non-isochronous writes are treated as low priority by default. Setting DisWrLoPri changes their priority to medium.

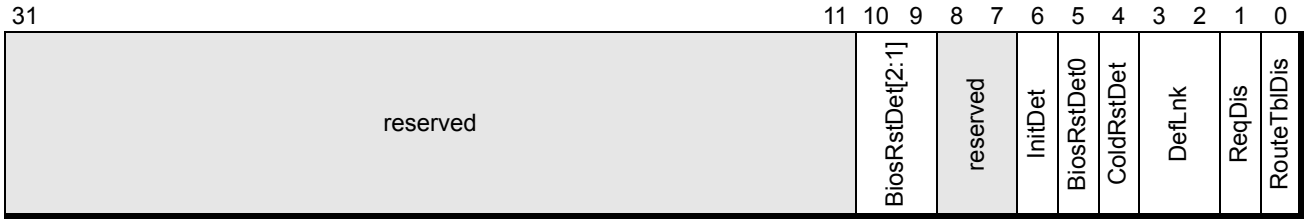
Disable High Priority Isochronous Writes (DisIsocWrHiPri)—Bit 30. Disables isochronous writes from being treated as high priority. Isochronous writes are treated as high priority by default. Setting DisIsocWrHiPri changes their priority to medium (if DisIsocWrMedPri is clear) or low (if DisIsocWrMedPri is set).

Disable Medium Priority Isochronous Writes (DisIsocWrMedPri)—Bit 31. Disables isochronous writes from being treated as medium priority. Isochronous writes are treated as high priority by default. Setting DisIsocWrMedPri along with DisIsocWrHiPri changes their priority to low.

4.3.10 HyperTransport™ Initialization Control Register

The HyperTransport Initialization Control Register controls the routing and request generation that follows device initialization. This register also contains a set of scratchpad read/write bits that are cleared by different initialization conditions and that may be used to distinguish between various types of initialization events. Note that the Request Disable and Routing Table Disable bits must not be cleared until the routing tables have been initialized.

HyperTransport™ Initialization Control Register **Function 0: Offset 6Ch**



Bits	Mnemonic	Function	R/W	Reset
31–11	reserved		R	0
10–9	BiosRstDet[2:1]	BIOS Reset Detect[2:1]		
8–7	reserved		R	0
6	InitDet	INIT Detect	R/W	0
5	BiosRstDet0	BIOS Reset Detect[0]	R/W	0
4	ColdRstDet	Cold Reset Detect	R/W	0
3–2	DefLnk	Default Link	R	
1	ReqDis	Request Disable	R/W	
0	RouteTblDis	Routing Table Disable	R/W	1

Field Descriptions

Routing Table Disable (RouteTblDis)—Bit 0. This bit determines whether the routing tables are used or whether default configuration access routing is used. It resets to 1, thereby routing requests to the Configuration Special Register (CSR) block and routing responses based on DefLnk. Once the routing tables have been set up this bit should be cleared.

0 = Packets are routed according to the routing tables

1 = Packets are routed according to the default link field (DefLnk)

Request Disable (ReqDis)—Bit 1. This bit determines if the node is allowed to generate request packets. It resets to 0 for the BSP and to 1 for all other processors. This bit should be cleared by system initialization firmware once the system has been initialized from the BSP. This bit is set by hardware and cleared by software.

0 = Request packets may be generated
1 = Request packets may not be generated

Default Link (DefLnk)—Bits 3–2. This field is used after a reset and before the routing tables are initialized. It is used by hardware to determine which link to route responses to and may be used by software as part of system connectivity discovery.

The default link field is loaded each time an incoming request is received, with the link ID of the link on which the packet arrived. It is read-only from software. It is only used to route packets during initialization, when the Routing Table Disable bit is set to 1, and only one outstanding request is active in the system at a time. During this interval, the value in the Default Link field is used to route responses, so that responses are always sent out on the link from which the last request was received. The register is updated as soon as a request is received, so reads of this register from the fabric return the link ID of the link on which the read command arrived. Reads from the CPU on the local node cause this field to get loaded with 11b.

00b = LDT0
01b = LDT1
10b = LDT2
11b = CPU on same node

Cold Reset Detect (ColdRstDet)—Bit 4. This bit may be used to distinguish between a cold versus a warm reset event by setting the bit to 1 before an initialization event is generated. This bit is cleared by a cold reset but not by a warm reset.

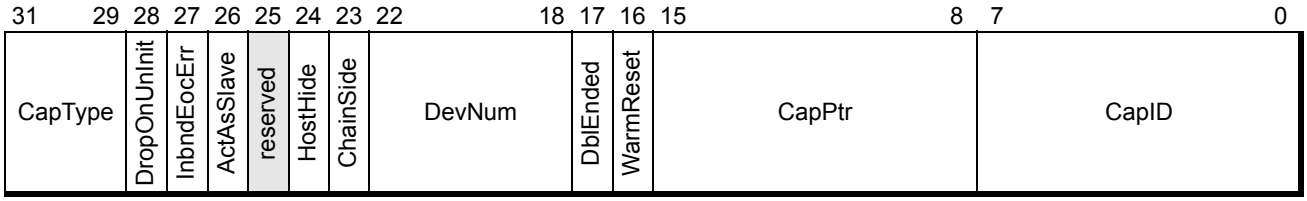
BIOS Reset Detect (BiosRstDet)—Bits 10–9,5. These bits may be used to distinguish between a reset event generated by the BIOS versus a reset event generated for any other reason by setting one or more of the bits to 1 before initiating a BIOS-generated reset event. These bits are cleared by a cold reset but not by a warm reset. See "Register Differences in Revisions of AMD NPT Family 0Fh Processors" on page 27 for revision information about this field.

INIT Detect (InitDet)—Bit 6. This bit may be used to distinguish between an INIT and a warm/cold reset by setting the bit to 1 before an initialization event is generated. This bit is cleared by a warm or cold reset but not by an INIT.

4.3.11 LDT*i* Capabilities Register

These registers define the capabilities of the LDT links. They also contain a capabilities pointer that points to the next item in the linked capabilities list.

LDT0, LDT1, LDT2 Capabilities Registers — Function 0: Offset 80h, A0h, C0h



Bits	Mnemonic	Function	R/W	Reset
31–29	CapType	Capability Type	R	001b
28	DropOnUnInit	Drop on Uninitialized Link	R	0
27	InbndEocErr	Inbound End-of-Chain Error	R	0
26	ActAsSlave	Act As Slave	R	0
25	reserved		R	0
24	HostHide	Host Hide	R	1
23	ChainSide	Chain Side	R	0
22–18	DevNum	Device Number	R	0
17	DblEnded	Double Ended	R	0
16	WarmReset	Warm Reset	R	1
15–8	CapPtr	Capability Pointer	R	
7–0	CapID	Capability ID (Always Reads 08h)	R	08h

Field Descriptions

Capability ID (CapID)—Bits 7–0. A capability ID of 08h indicates HyperTransport technology capability.

Capability Pointer (CapPtr)—Bits 15–8. Contains a pointer to the next capability in the list. Depending on the specific HyperTransport link configuration physically present on the processor, this will be either A0h (LDT1), C0h (LDT2), or 00h, if it is the last one.

Warm Reset (WarmReset)—Bit 16. Allows a reset sequence initiated by the HyperTransport technology control register to be either warm or cold. Since resets initiated through the HyperTransport control register are not supported, this field is read-only 1.

Double Ended (DblEnded)—Bit 17. Indicates that there is another bridge at the far end of the noncoherent HyperTransport chain. Since double-hosted chains are not supported, this field is read-only 0.

Device Number (DevNum)—Bits 22–18. The device number of configuration accesses which Northbridge responds to when accessed from the noncoherent HyperTransport technology chain. Since double-hosted chains are not supported, this field is read-only 0.

Chain Side (ChainSide)—Bit 23. This bit indicates which side of the host bridge is being accessed. Since double-hosted chains are not supported, this bit is read-only 0.

Host Hide (HostHide)—Bit 24. This bit causes the memory system configuration space to be inaccessible from the noncoherent HyperTransport technology chain. Since double-hosted chains are not supported, this bit is read-only 1.

Act As Slave (ActAsSlave)—Bit 26. Since this function is not currently implemented, this bit is read-only 0.

Inbound End-of-Chain Error (InbndEocErr)—Bit 27. Since this function is not currently implemented, this bit is read-only 0.

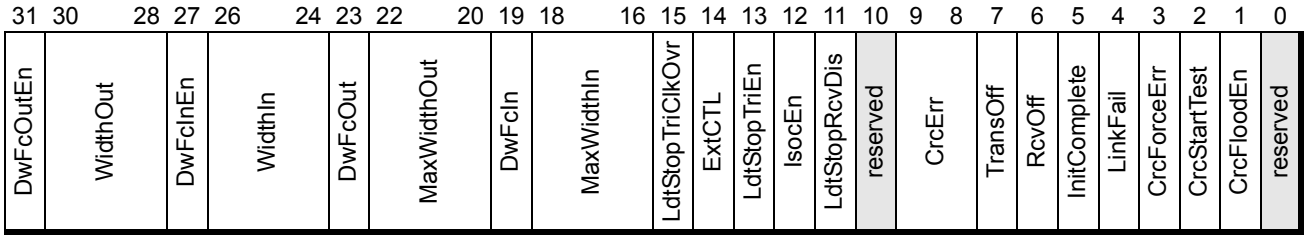
Drop on Uninitialized Link (DropOnUnInit)—Bit 28. Since this function is not currently implemented, this bit is read-only 0.

Capability Type (CapType)—Bits 31–29. The code 001b designates a host interface HyperTransport technology capability.

4.3.12 LDT*i* Link Control Registers

These registers control LDT link operation and log link errors.

LDT0, LDT1, LDT2 Link Control Registers — Function 0: Offset 84h, A4h, C4h



Bits	Mnemonic	Function	R/W	Reset
31	DwFcOutEn	Doubleword Flow Control Out Enable	R	0
30–28	WidthOut	Link Width Out	R/W	0
27	DwFcInEn	Doubleword Flow Control In Enable	R	0
26–24	WidthIn	Link Width In	R/W	0
23	DwFcOut	Doubleword Flow Control Out	R	0
22–20	MaxWidthOut	Max Link Width Out	R	001b
19	DwFcIn	Doubleword Flow Control In	R	0
18–16	MaxWidthIn	Max Link Width In	R	001b
15	LdtStopTriClkOvr	HyperTransport™ Stop Tristate Clock Override	R/W	0
14	ExtCTL	Extended CTL Time	R/W	0
13	LdtStopTriEn	HyperTransport Stop Tristate Enable	R/W	0
12	IsocEn	Isochronous Enable	R	0
11	LdtStopRcvDis	HyperTransport™ Stop Receiver Disable	R/W	0
10	reserved		R	0
9–8	CrcErr	CRC_Error	R/W	0
7	TransOff	Transmitter Off	R/W	0

Bits	Mnemonic	Function	R/W	Reset
6	RcvOff	Receiver Off	R/W	0
5	InitComplete	Initialization Complete	R	0
4	LinkFail	Link Failure	R/W	0
3	CrcForceErr	CRC Force Error	R/W	0
2	CrcStartTest	CRC Start Test	R	0
1	CrcFloodEn	CRC Flood Enable	R/W	0
0	reserved		R	0

Field Descriptions

CRC Flood Enable (CrcFloodEn)—Bit 1. If cleared to 0, this bit prevents CRC errors both from generating sync packets and causing the system to come down, and from setting the LinkFail bit. However, CRC checking logic still runs on all lanes enabled by LinkWidthIn, and detected errors still set the CRC Error bits to 1. Its reset value is 0.

0 = Do not generate sync packets on CRC error (default)
1 = Generate sync packets on CRC error

CRC Start Test (CrcStartTest)—Bit 2. Since this function is not currently implemented, this bit is read-only 0.

CRC Force Error (CrcForceErr)—Bit 3. When this bit is set, a bad CRC is generated on all transmitting lanes as enabled by LinkWidthOut. The covered data is not affected. This bit is readable and writable by software and resets to 0.

0 = Do not generate a bad CRC (default)
1 = Generate a bad CRC

Link Failure (LinkFail)—Bit 4. The LinkFail bit is set to 1 when a CRC error or a sync packet is detected after link initialization or if the link is not connected. See "Machine Check Architecture (MCA) Registers" on page 140 for more details. This bit is maintained through a warm reset and is cleared to 0 on a cold reset. LinkFail is set to 1 by hardware in the event of a link error that results in a sync flood. If this bit is set after a warm reset, BIOS must attempt to clear the bit by writing a 1 to determine if the link is not connected or if a sync flood occurred. It can be cleared by software by writing a 1. This bit becomes valid after Function 0, Offset 98h, B8h, or D8h, LinkConPend bit is cleared by hardware.

0 = No link failure detected
1 = Link failure detected

Initialization Complete (InitComplete)—Bit 5. This read-only bit is reset to 0, and set to 1 by hardware when low-level link initialization is successfully complete. If there is no device on the other end of the link, or if that device is unable to properly perform the low-level link initialization protocol, the bit is not set to 1. Software must not attempt to generate any packets across the link until this bit is set to 1. CRC checking in the hardware does not begin until initialization is complete.

0 = Initialization not complete

1 = Initialization is complete

Receiver Off (RcvOff)—Bit 6. This bit disables the receiver from accepting any further packets. This bit resets to 0, and is set by software writing a 1 to the bit. This bit cannot be cleared by software—a write of 0 to this bit position has no effect. If the HyperTransport link is not connected, then this bit is set by hardware.

0 = Receiver on
1 = Receiver off

Transmitter Off (TransOff)—Bit 7. This bit provides a mechanism to shut off a link transmitter for power savings or EMI reduction. When set to 1, no output signals toggle on the link. This bit resets to 0, and is set by software writing a 1 to the bit. This bit cannot be cleared by software—a write of 0 to this bit position has no effect. If the HyperTransport link is not connected, then this bit is set by hardware.

0 = Transmitter on
1 = Transmitter off

CRC_Error (CrcErr)—Bits 9–8. These bits are set to 1 by hardware when a CRC error is detected on an incoming link. Errors are detected and reported on a per byte lane basis where bit 8 corresponds to the least significant byte lane. Two bits are required to cover the maximum HyperTransport link width of 16 bits. Error bits for unimplemented (as specified by MaxWidthIn) or unused byte lanes return 0 when read.

These bits are maintained through a warm reset and is cleared to 0 on a cold reset. They are readable from software and are individually cleared to 0 by software by writing a 1.

00b = No error
Bit [0] =1 Error on Byte Lane 0
Bit [1] =1 Error on Byte Lane 1

HyperTransport Stop Receiver Disable (LdtStopRcvDis)—Bit 11. This bit enables power savings by controlling the state of the HyperTransport receiver. When this bit is set for any HyperTransport link, the HyperTransport receiver is disabled when LDTSTOP_L is asserted.

Isochronous Enable (IsocEn)—Bit 12. Since this function is not currently implemented, this bit is read-only 0.

HyperTransport Stop Tristate Enable (LdtStopTriEn)—Bit 13. This bit controls whether the transmitter tristates the link during the disconnected state of an LDTSTOP_L sequence. When the bit is set the transmitter tristates the link. When the bit is clear, it continues to drive the link. When this bit is set and LdtStopTriClkOvr is clear, the delay specified in the ReConDel (Function 3, Offset D4h) field is applied before restarting the link. The bit is cleared by a cold reset and its value is maintained through a warm reset.

0 = Driven during disconnect of an LDTSTOP_L
1 = Tristated during disconnect of an LDTSTOP_L

Extended CTL Time (ExtCTL)—Bit 14. When this bit is set, CTL will be asserted for 50µs instead of 16 bit times during the link initialization sequence. This bit should be set if extended CTL

is required by the device at the other end of the HyperTransport link, as indicated by the Extended CTL Required bit in the Feature Capability register. The bit is cleared by cold reset and its value is maintained through warm reset.

HyperTransport Stop Tristate Clock Override (LdtStopTriClkOvr)—Bit 15. This bit prevents the HyperTransport clock from being tristated when LDTSTOP_L is asserted and LdtStopTriEn is set.

Max Link Width In (MaxWidthIn)—Bits 18–16. This field contains three bits that indicate the physical width of the incoming side of the HyperTransport link implemented by this device. For AMD NPT Family 0Fh Processors, this field is set to 000 or 001 to indicate an 8-bit or 16-bit link.

Doubleword Flow Control In (DwFcIn)—Bit 19. This bit is read-only 0 to indicate that this link does not support doubleword flow control.

Max Link Width Out (MaxWidthOut)—Bits 22–20. This field contains three bits that indicate the physical width of the outgoing side of the HyperTransport link implemented by this device. It is read-only. For AMD NPT Family 0Fh Processors, this field is set to 000 or 001 to indicate an 8-bit or 16-bit link.

Doubleword Flow Control Out (DwFcOut)—Bit 23. This bit is read-only 0 to indicate that this link does not support doubleword flow control.

Link Width In (WidthIn)—Bits 26–24. This field is similar to the WidthOut field, except that it controls the utilized width of the incoming side of the links implemented by this device.

The hardware will only allow values to be programmed into this field which are consistent with the width capabilities of the link as specified by MaxWidthIn. Attempts to write values inconsistent with the capabilities will result in this field not being updated.

Doubleword Flow Control In Enable (DwFcInEn)—Bit 27. Since this function is not currently implemented, this bit is read-only 0.

Link Width Out (WidthOut)—Bits 30–28. This field controls the utilized width (which may not exceed the physical width) of the outgoing side of the HyperTransport link. Software can read/write this field, and its value is maintained through a warm reset. After cold reset, this field is initialized by hardware based on the results of the link width negotiation sequence described in the HyperTransport technology specification. Based on sizing the devices at both ends of the link, software may then write a different value into the register. The chain must pass through warm reset or a LDTSTOP_L disconnect sequence for the new width values to be reflected on the link.

The WidthOut CSR in the link transmitter must match the WidthIn CSR in the link receiver of the device on the other side of the link. The LinkWidthIn and LinkWidthOut registers within the same device are not required to have matching values.

The hardware will only allow values to be programmed into this field which are consistent with the width capabilities of the link as specified by MaxWidthOut. Attempts to write values inconsistent with the capabilities will result in this field not being updated

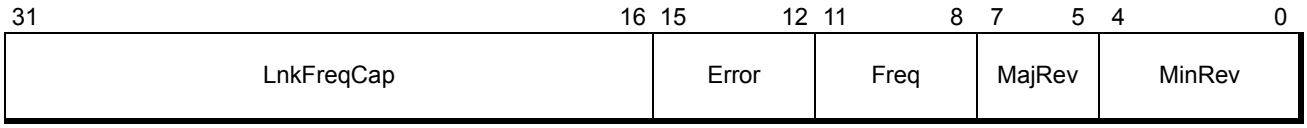
000b = 8-bit
001b = 16-bit
010b = reserved
011b = 32-bit
100b = 2-bit
101b = 4-bit
110b = reserved
111b = Link physically not connected

Doubleword Flow Control Out Enable (DwFcOutEn)—Bit 31. Since this function is not currently implemented, this bit is read-only 0.

4.3.13 LDT*i* Frequency/Revision Registers

These registers control the link frequency, specify the link frequency capabilities, and describe the HyperTransport technology specification level to which the specific link conforms.

LDT0, LDT1, LDT2 Frequency/Revision Registers Function 0: Offset 88h, A8h, C8h



Bits	Mnemonic	Function	R/W	Reset
31–16	LnkFreqCap	Link Frequency Capability	R	
15–12	Error	Error	R	0
11–8	Freq	Link Frequency	R/W	0
7–5	MajRev	Major Revision	R	001b
4–0	MinRev	Minor Revision	R	00010b

Field Descriptions

Minor Revision (MinRev)—Bits 4–0. Contains the minor revision of the HyperTransport technology specification to which the device conforms.

Major Revision (MajRev)—Bits 7–5. Contains the major revision of the HyperTransport technology specification level to which the device conforms.

Link Frequency (Freq)—Bits 11–8. Specifies the maximum operating frequency of the link's transmitter clock. The encoding of this field is shown below.

The link frequency field (Freq) is cleared by cold reset. Software can write a nonzero value to this register, and that value takes effect on the next warm reset or LDTSTOP_L disconnect sequence.

The hardware will only allow values to be programmed into this field which are consistent with the frequency capabilities of the link as specified by LinkFreqCap. Attempts to write values inconsistent with the capabilities will result in this field not being updated.

It is possible to program this field for a higher frequency than the maximum allowed by the processor. Refer to the processor data sheet for the maximum operating frequency allowed for a given processor implementation. See 3.6.1.2 "HyperTransport™ Technology Transfer Speeds" on page 39 for processor specific link frequency restrictions.

0000b = 200 MHz
0001b = reserved
0010b = 400 MHz
0011b = reserved
0100b = 600 MHz
0101b = 800 MHz
0110b = 1000 MHz
0111b = reserved
1000–1110b = reserved
1111b = 100 MHz

Error (Error)—Bits 15–12. This function is not currently implemented.

Link Frequency Capability (LnkFreqCap)—Bits 31–16. Indicates the clock frequency capabilities of the link. Each bit corresponds to one of the 16 possible link frequency encodings.

While the frequency capabilities of different AMD NPT Family 0Fh Processors may vary, the 200-MHz and 100-MHz frequencies are supported by the design.

4.3.14 LDT*i* Feature Capability Registers

These registers identify features that are supported by the specific link.

LDT0, LDT1, LDT2 Feature Capability Registers — Function 0: Offset 8Ch, ACh, CCh

Bits	Mnemonic	Function	R/W	Reset
31–9	reserved		R	0
8	ExtRegSet	Extended Register Set	R	0

Bits	Mnemonic	Function	R/W	Reset
7–4	reserved		R	0
3	ExtCTLRqd	Extended CTL Required	R	0
2	CrcTstMode	CRC Test Mode	R	0
1	LdtStopMode	HyperTransport Stop Mode	R	1
0	IsocMode	Isochronous Flow Control Mode	R	0

Field Descriptions

Isochronous Flow Control Mode (IsocMode)—Bit 0. This is a read-only bit that reflects whether isochronous flow control operation is supported. This bit is set to 0, indicating no support for this feature.

HyperTransport Stop Mode (LdtStopMode)—Bit 1. This is a read-only bit that indicates whether the LDTSTOP_L protocol is supported. This bit is set to 1, indicating support for this feature.

CRC Test Mode (CrcTstMode)—Bit 2. This is a read-only bit that indicates whether CRC test mode is supported. This bit is set to 0, indicating no support for this feature.

Extended CTL Required (ExtCTLRqd)—Bit 3. This is a read-only bit that indicates whether extended CTL is required. This bit is set to 0, indicating that extended CTL is not required.

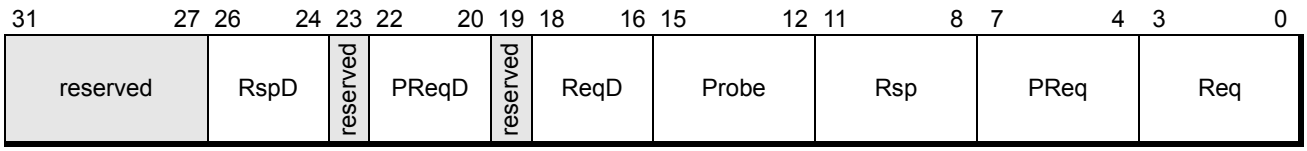
Extended Register Set (ExtRegSet)—Bit 8. This is a read-only bit that indicates whether the Enumeration Scratchpad, Error Handling and Memory Base/Limit Upper registers are supported. This bit is set to 0, indicating no support for this feature.

4.3.15 LDT*i* Buffer Count Registers

These registers specify the number of command and data buffers for each virtual channel available for use by the transmitter at the other end of the specific HyperTransport™ technology link. See "XBAR Flow Control Buffers" on page 164 for more information on command and data buffers.

Note: *The reset values for each of the LDTn Buffer Count registers depend on the link connection type (coherent HyperTransport or noncoherent HyperTransport technology). Because hardware attempts to choose optimal settings, this register should not, in general, need to be changed.*

LDT0, LDT1, LDT2 Buffer Count Registers **Function 0: Offset 90h, B0h, D0h**



Bits	Mnemonic	Function	R/W	Coherent Link Reset	Noncoherent Link Reset
31–27	reserved		R	0	0
26–24	RspD	Response Data Buffer Count	R/W	100b	010b
23	reserved		R	0	0
22–20	PReqD	Posted Request Data Buffer Count	R/W	001b	101b
19	reserved		R	0	0
18–16	ReqD	Request Data Buffer Count	R/W	011b	001b
15–12	Probe	Probe Buffer Count	R/W	5h	0h
11–8	Rsp	Response Buffer Count	R/W	6h	4h
7–4	PReq	Posted Request Buffer Count	R/W	1h	5h
3–0	Req	Request Buffer Count	R/W	3h	6h

Field Descriptions

Request Buffer Count (Req)—Bits 3–0. Defines the number of request command buffers available for use by the transmitter at the other end of the link. See the register layout for the default reset settings for coherent and noncoherent HyperTransport technology links.

Posted Request Buffer Count (PReq)—Bits 7–4. Defines the number of Posted request command buffers available for use by the transmitter at the other end of the link. See the register layout for the default reset settings for coherent and noncoherent HyperTransport technology links.

Response Buffer Count (Rsp)—Bits 11–8. Defines the number of response buffers available for use by the transmitter at the other end of the link. See the register layout for the default reset settings for coherent and noncoherent HyperTransport technology links.

Probe Buffer Count (Probe)—Bits 15–12. Defines the number of probe buffers available for use by the transmitter at the other end of the link. This field must be 0 for a noncoherent HyperTransport link. See the register layout for the default reset settings for coherent and noncoherent HyperTransport technology links.

Request Data Buffer Count (ReqD)—Bits 18–16. Defines the number of request data buffers available for use by the transmitter at the other end of the link. See the register layout for the default reset settings for coherent and noncoherent HyperTransport technology links.

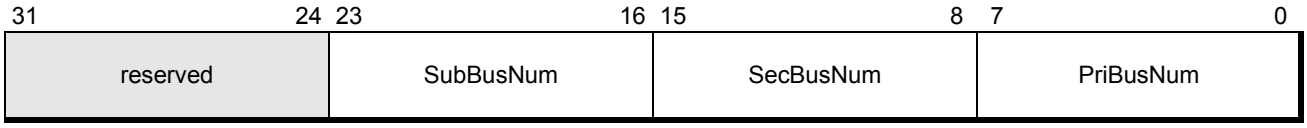
Posted Request Data Buffer Count (PReqD)—Bits 22–20. Defines the number of posted request data buffers available for use by the transmitter at the other end of the link. See the register layout for the default reset settings for coherent and noncoherent HyperTransport technology links.

Response Data Buffer Count (RspD)—Bits 26–24. Defines the number of response data buffers available for use by the transmitter at the other end of the link. See the register layout for the default reset settings for coherent and noncoherent HyperTransport technology links.

4.3.16 LDT*i* Bus Number Registers

If the specific link (LDT0, LDT1, or LDT2) is noncoherent HyperTransport technology, this register specifies the bus numbers downstream (behind) the host bridge. If the link is coherent HyperTransport technology, this register has no meaning.

LDT0, LDT1, LDT2 Bus Number Registers **Function 0: Offset 94h, B4h, D4h**



Bits	Mnemonic	Function	R/W	Reset
31–24	reserved		R	0
23–16	SubBusNum	Subordinate Bus Number	R/W	0
15–8	SecBusNum	Secondary Bus Number	R/W	0
7–0	PriBusNum	Primary Bus Number	R	0

Field Descriptions

Primary Bus Number (PriBusNum)—Bits 7–0. Defines the primary bus number. Because the primary bus is the coherent HyperTransport technology fabric, this field always reads 0.

Secondary Bus Number (SecBusNum)—Bits 15–8. Defines the secondary bus number.

The Secondary Bus Number register is used to record the bus number of the bus segment to which the secondary interface of the host bridge is connected. Configuration software programs the value in this register. If this link contains the HyperTransport I/O hub, the secondary bus number must be programmed to 0.

Subordinate Bus Number (SubBusNum)—Bits 23–16. Defines the subordinate bus number.

The Subordinate Bus Number register is used to record the bus number of the highest numbered bus segment that is behind (or subordinate to) the host bridge. Configuration software programs the value in this register.

4.3.17 LDT*i* Type Registers

These registers designate the type of HyperTransport link attached to the specific link. The bits are set by hardware after link initialization.

LDT0, LDT1, LDT2 Type Registers — Function 0: Offset 98h, B8h, D8h

Bits	Mnemonic	Function	R/W	Reset
31–5	reserved		R	0
4	LinkConPend	Link Connect Pending	R	
3	UniP-cLDT	UniP-cLDT	R	
2	NC	Non Coherent	R	
1	InitComplete	Initialization Complete	R	
0	LinkCon	Link Connected	R	

Field Descriptions

Link Connected (LinkCon)—Bit 0. Indicates that the link is connected. It is valid once the LinkConPend bit is clear.

0 = Not connected
1 = Connected

Initialization Complete (InitComplete)—Bit 1. Set to 1 to indicate that the initialization of the link has completed. (It is a duplicate of Bit 5 in HyperTransport Link Control). The NC and UniP-cLDT bits are invalid until link initialization is complete.

0 = Initialization not complete
1 = Initialization is complete.

Non Coherent (NC)—Bit 2. Defines the link type (coherent versus noncoherent).

0 = Coherent HyperTransport technology
1 = Noncoherent HyperTransport technology

UniP-cLDT (UniP-cLDT)—Bit 3. Further qualifies the NC link type bit. A 1 indicates that this link is a uniprocessor coherent HyperTransport link connected to an external Northbridge.

0 = Normal coherent HyperTransport or noncoherent HyperTransport link
1 = Uniprocessor coherent HyperTransport link to external Northbridge

Link Connect Pending (LinkConPend)—Bit 4. Qualifies the LinkCon bit and is set to 1 when hardware is attempting to determine whether a link is connected or not.

0 = Link connection determination complete
1 = Link connection still being determined

4.4 Function 1—Address Map

The address map defines the address spaces assigned to DRAM, memory-mapped I/O, PCI I/O, and configuration accesses and also specifies destination information for each to facilitate routing of each access to the appropriate target. Table 6 lists each Function 1 configuration register.

Table 6. Function 1 Configuration Registers

Offset	Register Name				Reset	Access	Description
00h	Device ID		Vendor ID (AMD)		1101_1022h	RO	page 55
04h	Status[1]		Command		0000_0000h	RO	
08h	Base Class Code	Subclass Code	Program Interface	Revision ID	0600_0000h	RO	page 82
0Ch	BIST	Header Type	Latency Timer	Cache Line Size	0080_0000	RO	page 83
10h	Base Address 0				0000_0000h	RO	
14h	Base Address 1				0000_0000h	RO	
18h	Base Address 2				0000_0000h	RO	
1Ch	Base Address 3				0000_0000h	RO	
20h	Base Address 4				0000_0000h	RO	
24h	Base Address 5				0000_0000h	RO	
28h	Card Bus CIS Pointer				0000_0000h	RO	
2Ch	Sub-System ID		Sub-System Vendor ID		0000_0000h	RO	
30h	ROM Base Address				0000_0000h	RO	
34h	Capabilities				0000_0000h	RO	
38h	reserved				0000_0000h	RO	
3Ch	Max Latency	Min GNT	Int Pin	Int Line	0000_0000h	RO	
40h	DRAM Base 0					RW	page 85
44h	DRAM Limit 0					RW	page 86
48h	DRAM Base 1					RW	page 85
4Ch	DRAM Limit 1					RW	page 86
50h	DRAM Base 2					RW	page 85
54h	DRAM Limit 2					RW	page 86
58h	DRAM Base 3					RW	page 85
5Ch	DRAM Limit 3					RW	page 86

Notes:

1. *The unimplemented registers in the standard PCI configuration space are implemented as read-only and return 0 if read.*
2. *Reads and writes to unimplemented registers in the extended PCI configuration space will result in unpredictable behavior.*

Table 6. Function 1 Configuration Registers (Continued)

Offset	Register Name	Reset	Access	Description
60h	DRAM Base 4		RW	page 85
64h	DRAM Limit 4		RW	page 86
68h	DRAM Base 5		RW	page 85
6Ch	DRAM Limit 5		RW	page 86
70h	DRAM Base 6		RW	page 85
74h	DRAM Limit 6		RW	page 86
78h	DRAM Base 7		RW	page 85
7Ch	DRAM Limit 7		RW	page 86
80h	Memory-Mapped I/O Base 0		RW	page 88
84h	Memory-Mapped I/O Limit 0		RW	page 89
88h	Memory-Mapped I/O Base 1		RW	page 88
8Ch	Memory-Mapped I/O Limit 1		RW	page 89
90h	Memory-Mapped I/O Base 2		RW	page 88
94h	Memory-Mapped I/O Limit 2		RW	page 89
98h	Memory-Mapped I/O Base 3		RW	page 88
9Ch	Memory-Mapped I/O Limit 3		RW	page 89
A0h	Memory-Mapped I/O Base 4		RW	page 88
A4h	Memory-Mapped I/O Limit 4		RW	page 89
A8h	Memory-Mapped I/O Base 5		RW	page 88
ACh	Memory-Mapped I/O Limit 5		RW	page 89
B0h	Memory-Mapped I/O Base 6		RW	page 88
B4h	Memory-Mapped I/O Limit 6		RW	page 89
B8h	Memory-Mapped I/O Base 7		RW	page 88
BCh	Memory-Mapped I/O Limit 7		RW	page 89
C0h	PCI I/O Base 0		RW	page 90
C4h	PCI I/O Limit 0		RW	page 91
C8h	PCI I/O Base 1		RW	page 90
CCh	PCI I/O Limit 1		RW	page 91
D0h	PCI I/O Base 2		RW	page 90
D4h	PCI I/O Limit 2		RW	page 91
D8h	PCI I/O Base 3		RW	page 90
DCh	PCI I/O Limit 3		RW	page 91

Notes:

1. *The unimplemented registers in the standard PCI configuration space are implemented as read-only and return 0 if read.*
2. *Reads and writes to unimplemented registers in the extended PCI configuration space will result in unpredictable behavior.*

Table 6. Function 1 Configuration Registers (Continued)

Offset	Register Name	Reset	Access	Description
E0h	Configuration Base and Limit 0		RW	page 92
E4h	Configuration Base and Limit 1		RW	page 92
E8h	Configuration Base and Limit 2		RW	page 92
ECh	Configuration Base and Limit 3		RW	page 92
F0h	DRAM Hole Address	0000_0000	R/W	page 93

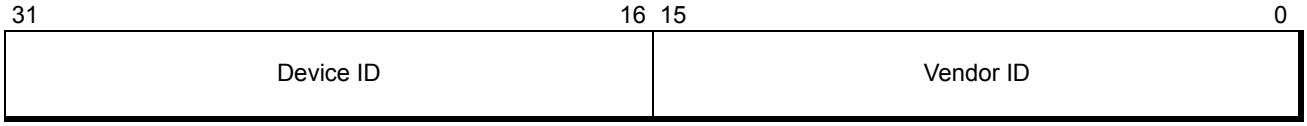
Notes:

1. *The unimplemented registers in the standard PCI configuration space are implemented as read-only and return 0 if read.*
2. *Reads and writes to unimplemented registers in the extended PCI configuration space will result in unpredictable behavior.*

4.4.1 Device/Vendor ID Register

This register specifies the device and vendor IDs for the Function 1 registers and is part of the standard PCI configuration header.

Device/Vendor ID Register **Function 1: Offset 00h**



Bits	Mnemonic	Function	R/W	Reset
31–16	DevID	Device ID	R	1101h
15–0	VenID	Vendor ID	R	1022h

Field Descriptions

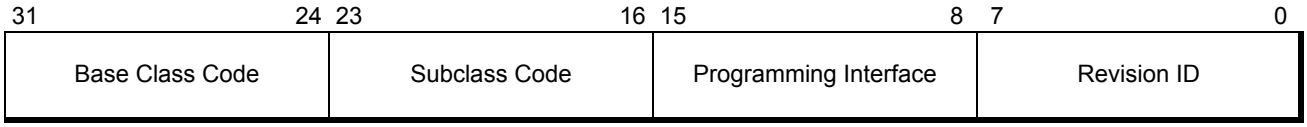
Vendor ID (VenID)—Bits 15–0. This read-only value is defined as 1022h for AMD.

Device ID (DevID)—Bits 31–16. This read-only value is defined as 1101h for the HyperTransport technology configuration function.

4.4.2 Class Code/Revision ID Register

This register specifies the class code and revision for the Function 1 registers and is part of the standard PCI configuration header.

Class Code/Revision ID Register — Function 1: Offset 08h



Bits	Mnemonic	Function	R/W	Reset
31–24	BCC	Base Class Code	R	06h
23–16	SCC	Subclass Code	R	00h
15–8	PI	Programming Interface	R	00h
7–0	RevID	Revision ID	R	00h

Field Descriptions

Revision ID (RevID)—Bits 7–0.

Programming Interface (PI)—Bits 15–8. This read-only value is defined as 00h.

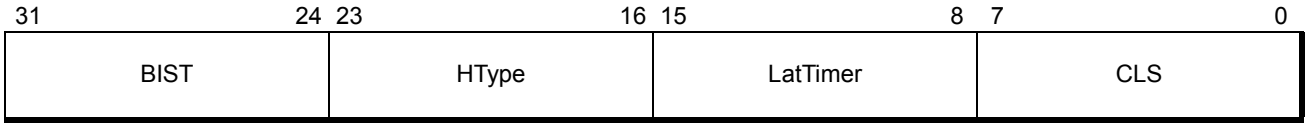
Sub Class Code (SCC)—Bits 23–16. This read-only value is defined as 00h.

Base Class Code (BCC)—Bits 31–24. This read-only value is defined as 06h for a host bridge device.

4.4.3 Header Type Register

This register specifies the header type for the Function 1 registers and is part of the standard PCI configuration header.

Header Type Register — Function 1: Offset 0Ch



Bits	Mnemonic	Function	R/W	Reset
31–24	BIST	BIST	R	00h
23–16	HType	Header Type	R	80h
15–8	LatTimer	Latency Timer	R	00h
7–0	CLS	Cache Line Size	R	00h

Field Descriptions

CacheLineSize (CLS)—Bits 7–0. This read-only value is defined as 00h.

LatencyTimer (LatTimer)—Bits 15–8. This read-only value is defined as 00h.

HeaderType (HType)—Bits 23–16. This read-only value is defined as 80h to indicate that multiple functions are present in the configuration header and that the header layout corresponds to a device header as opposed to a bridge header. See "Register Differences in Revisions of AMD NPT Family 0Fh Processors" on page 27 for revision information about this field.

BIST—Bits 31–24. This read-only value is defined as 00h.

4.4.4 DRAM Address Map

These registers define sections of the memory address map for which accesses should be routed to DRAM. DRAM regions must not overlap each other. For addresses within the specified range of a base/limit pair, requests are routed to the memory controller on the node specified by the destination Node ID.

System addresses are considered to be within the defined range if they are greater than or equal to the base and less than or equal to the limit. For the purposes of this comparison, the lower unspecified bits of the base are assumed to be 0s and the lower unspecified bits of the limit are assumed to be 1s.

An address that maps to both DRAM and memory-mapped I/O will be routed to MMIO.

Programming of the DRAM address maps must be consistent with the Top Of Memory and Memory Type Range registers (see Chapter 14, "Processor Configuration Registers"). Accesses from the CPU can only access the DRAM address maps if the corresponding CPU memory type is DRAM. For accesses from I/O devices, the lookup is based on address only.

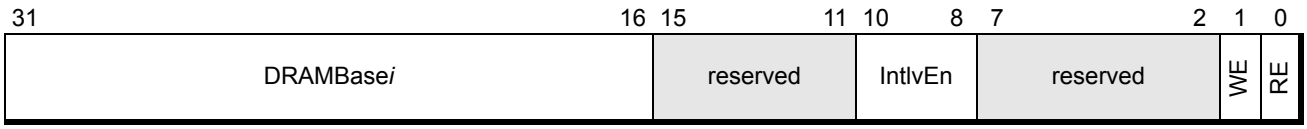
Each base/limit set of DRAM address maps is associated with a particular Node ID (see "Node ID Register" on page 60). The DRAM Base/Limit 0 registers specify the DRAM attached to node 0. Similarly, DRAM on nodes 1–7 is described by base/limit registers 1–7. Note that the destination NodeId field must still be written to ensure correct operation.

When node interleaving is enabled, each node's DRAM limit must be set to the Top Of Memory and each node's DRAM base must be set to 0. The node to which an address is routed to when nodes are interleaved is defined by IntlvEn (Function 1, Offset 40h, 48h, etc.) and IntlvSel (Function 1, Offset 44h, 4Ch, etc.). Each node must be configured with the same amount of DRAM when node interleaving is enabled.

Address routed to the DRAM controller (InputAddr) is calculated from the system address (SysAddr) in the following way:
DramAddr[39:0] = {SysAddr[39:24] - DRAMBase[39:24], SysAddr[23:0]},
InputAddr[35:0] = {DramAddr[35:12], DramAddr[11:0]} when node memory is not interleaved,
InputAddr[35:0] = {DramAddr[36:13], DramAddr[11:0]} when 2 nodes are interleaved,
InputAddr[35:0] = {DramAddr[37:14], DramAddr[11:0]} when 4 nodes are interleaved,
InputAddr[35:0] = {DramAddr[38:15], DramAddr[11:0]} when 8 nodes are interleaved.

4.4.4.1 DRAM Base *i* Registers

DRAM Base 0–7 Registers Function 1: Offset 40h, 48h, 50h, 58h, 60h, 68h, 70h, 78h



Bits	Mnemonic	Function	R/W	Reset
31–16	DRAMBase*i*	DRAM Base Address *i* Bits 39–24	R/W	X
15–11	reserved		R	0
10–8	IntlvEn	Interleave Enable	R/W	X
7–2	reserved		R	0
1	WE	Write Enable	R/W	0
0	RE	Read Enable	R/W	0

"X" in the Reset column indicates that the field initializes to an undefined state after reset.

Field Descriptions

Read Enable (RE)—Bit 0. This bit enables reads to the defined address space.

0 = Disabled
1 = Enabled

Write Enable (WE)—Bit 1. This bit enables writes to the defined address space.

0 = Disabled
1 = Enabled

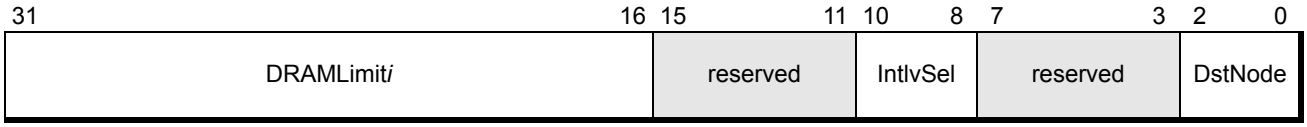
Interleave Enable (IntlvEn)—Bits 10–8. This field enables interleaving on a 4-Kbyte boundary between memory on different nodes. The bits are encoded as follows:

000b = No interleave
001b = Interleave on A[12] (2 nodes)
010b = reserved
011b = Interleave on A[12] and A[13] (4 nodes)
100b = reserved
101b = reserved
110b = reserved
111b = Interleave on A[12] and A[13] and A[14] (8 nodes)

DRAM Base Address *i* Bits 39–24 (DRAMBase*i*)—Bits 31–16. This field defines the upper address bits of a 40-bit address that defines the start of DRAM region *i* (where *i* = 0, 1, . . . 7).

4.4.4.2 DRAM Limit *i* Registers

DRAM Limit 0–7 Registers Function 1: Offset 44h, 4Ch, 54h, 5Ch, 64h, 6Ch, 74h, 7Ch



Bits	Mnemonic	Function	R/W	Reset
31–16	DRAMLimit*i*	DRAM Limit Address *i* (39–24)	R/W	X
15–11	reserved		R	0
10–8	IntlvSel	Interleave Select	R/W	X
7–3	reserved		R	0
2–0	DstNode	Destination Node ID	R/W	X

"X" in the Reset column indicates that the field initializes to an undefined state after reset.

Field Descriptions

Destination Node ID (DstNode)—Bits 2–0. This field specifies the node that a packet is sent to if it is within the address range. The value of this field must correspond to the number of the register pair, and it should be set as follows:

000b for Function 1, Offset 44h register
001b for Function 1, Offset 4Ch register
...
111b for Function 1, Offset 7Ch register

Interleave Select (IntlvSel)—Bits 10–8. This field specifies the values of address bits A[14:12] to use with the Interleave Enable field (IntlvEn[2:0]) to determine which 4-Kbyte blocks are routed to this region (See Figure 3).

IntlvSel[0] corresponds to A[12]
IntlvSel[1] corresponds to A[13]
IntlvSel[2] corresponds to A[14]

DRAM Limit Address *i* (39–24) (DRAMLimit*i*)—Bits 31–16. This field defines the upper address bits of a 40-bit address that defines the end of DRAM region n (where $i = 0,1, \ldots 7$).

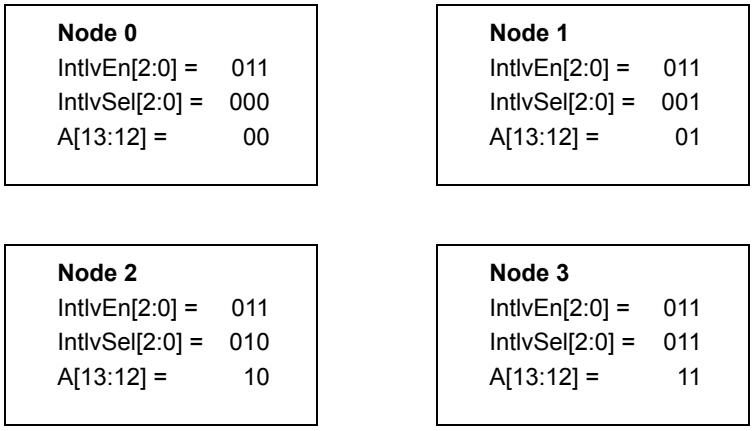


Figure 3. Interleave Example (IntlvEn Relation to IntlvSel)

4.4.5 Memory-Mapped I/O Address Map Registers

These registers define sections of the memory address map for which accesses should be routed to memory-mapped I/O. MMIO regions must not overlap each other. For addresses within the specified range of a base/limit pair, requests are routed to the noncoherent HyperTransport link specified by the destination Node ID and destination Link ID.

Addresses are considered to be within the defined range if they are greater than or equal to the base and less than or equal to the limit. For the purposes of this comparison, the lower unspecified bits of the base are assumed to be 0s and the lower unspecified bits of the limit are assumed to be 1s.

An address that maps to both DRAM and memory-mapped I/O is routed to MMIO.

Programming of the MMIO address maps must be consistent with the Top Of Memory and Memory Type Range registers (see Chapter 14, "Processor Configuration Registers"). In particular, accesses from the CPU can only hit in the MMIO address maps if the corresponding CPU memory type is of type I/O. For accesses from I/O devices, the lookup is based on address only.

4.4.5.1 Extended Configuration Space Access

Chipset devices may support PCI-defined extended configuration space through an MMIO range. Typically, requests to the MMIO range for extended configuration space are required to use the non-posted channel. Therefore, the Non-Posted bit should be set for this MMIO range. Instructions used to read extended configuration space must be of the following form:

mov EAX/AX/AL, <any_address_mode>;

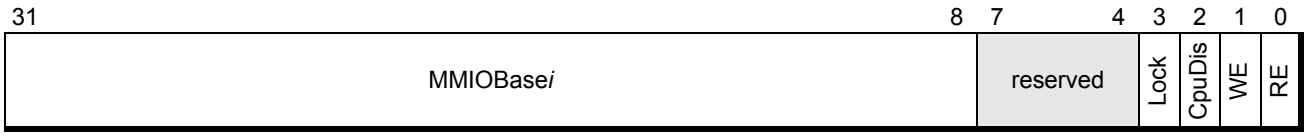
Instructions used to write extended configuration space must be of the following form:

mov <any_address_mode>, EAX/AX/AL;

In addition, all such accesses are required not to cross any naturally aligned doubleword boundary. Access to extended configuration registers that do not meet these requirements result in undefined behavior. Extended configuration space is normally specified to be the uncacheable memory type.

4.4.5.2 Memory-Mapped I/O Base *i* Registers

Memory-Mapped I/O Base 0–7 Registers **Function 1: Offset 80h, 88h, 90h, 98h, A0h, A8h, B0h, B8h**



Bits	Mnemonic	Function	R/W	Reset
31–8	MMIOBase*i*	Memory-Mapped I/O Base Address *i* (39–16)	R/W	X
7–4	reserved		R	0
3	Lock	Lock	R/W	X
2	CpuDis	CPU Disable	R/W	X
1	WE	Write Enable	R/W	0
0	RE	Read Enable	R/W	0

"X" in the Reset column indicates that the field initializes to an undefined state after reset.

Field Descriptions

Read Enable (RE)—Bit 0. This bit enables reads to the defined address space.

0 = Disabled
1 = Enabled

Write Enable (WE)—Bit 1. This bit enables writes to the defined address space.

0 = Disabled
1 = Enabled

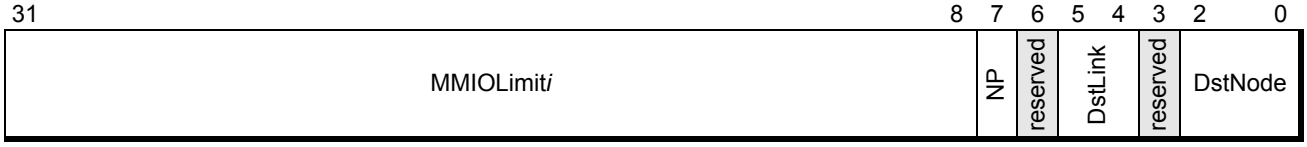
CPU Disable (CpuDis)—Bit 2. When set, this bit causes this MMIO range to apply to I/O requests only, not to CPU requests.

Lock (Lock)—Bit 3. Setting this bit, along with either the WE or RE bits, makes the base/limit registers read-only.

Memory-Mapped I/O Base Address *i* (39–16) (MMIOBase*i*)—Bit 31–8. This field defines the upper address bits of a 40-bit address that defines the start of memory-mapped I/O region n (where n = 0,1, . . . 7).

4.4.5.3 Memory-Mapped I/O Limit *i* Registers

Memory-Mapped I/O Limit *i* Registers **Function 1: Offset 84h, 8Ch, 94h, 9Ch, A4h, ACh, B4h, BCh**



Bits	Mnemonic	Function	R/W	Reset
31–8	MMIOLimit*i*	Memory-Mapped I/O Limit Address *i*	R/W	X
7	NP	Non-Posted	R/W	X
6	reserved		R	0
5–4	DstLink	Destination Link ID	R/W	X
3	reserved		R	0
2–0	DstNode	Destination Node ID	R/W	X

"X" in the Reset column indicates that the field initializes to an undefined state after reset.

Field Descriptions

Destination Node ID (DstNode)—Bits 2–0. This field specifies the node that a packet is sent to if it is within the address range. The bits are encoded as follows:

000b = Node 0
001b = Node 1
...
111b = Node 7

Destination Link ID (DstLink)—Bits 5–4. This field specifies the HyperTransport link number to send the packet to once the packet reaches the destination node.

00b = Link 0
01b = Link 1
10b = Link 2
11b = reserved

Non-Posted (NP)—Bit 7. When set to 1, this bit forces CPU writes to this memory-mapped I/O region to be non-posted. This may be used to force writes to be non-posted for MMIO regions which map to the legacy ISA/LPC bus, or in conjunction with the DsNpReqLmt field in the HyperTransport Transaction Control register (page 62) in order to allow downstream CPU requests to be counted and thereby limited to a specified number. This latter use of the NP bit may be used to avoid loop deadlock in systems that implement a reflection region in an I/O device that reflects downstream accesses back upstream. See the *HyperTransport™ I/O Link Specification* summary of deadlock scenarios for more information.

0 = CPU writes may be posted
1 = CPU writes must be non-posted

Memory-Mapped I/O Limit Address *i* (39–16) (MMIOLimit*i*)—Bits 31–8. This field defines the upper address bits of a 40-bit address that defines the end of memory-mapped I/O region n (where n = 0,1, . . . 7).

4.4.6 PCI I/O Address Map Registers

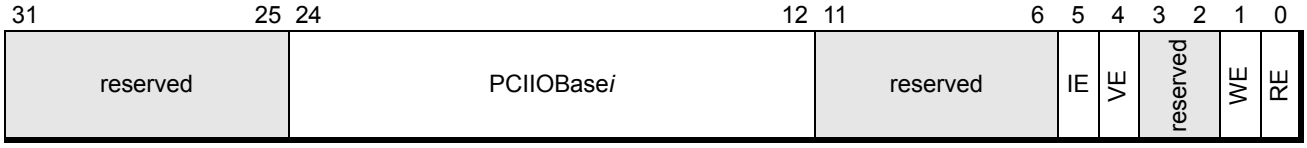
These registers define sections of the PCI I/O address map for which accesses should be routed to each noncoherent HyperTransport technology chain. PCI I/O regions must not overlap each other. For PCI I/O addresses within the specified range of a base/limit pair, requests are routed to the noncoherent HyperTransport link specified by the destination Node ID and destination Link ID.

Addresses are considered to be within the defined range if they are greater than or equal to the base and less than or equal to the limit. For the purposes of this comparison, the lower unspecified bits of the base are assumed to be 0s and the lower unspecified bits of the limit are assumed to be 1s.

PCI I/O accesses are generated from x86 IN/OUT instructions.

4.4.6.1 PCI I/O Base *i* Registers

PCI I/O Base 0–3 Registers **Function 1: Offset C0h, C8h, D0h, D8h**



Bits	Mnemonic	Function	R/W	Reset
31–25	reserved		R	0
24–12	PCIIOBase*i*	PCI I/O Base Address *i*	R/W	X
11–6	reserved		R	0
5	IE	ISA Enable	R/W	X
4	VE	VGA Enable	R/W	X
3–2	reserved		R	0
1	WE	Write Enable	R/W	0
0	RE	Read Enable	R/W	0

"X" in the Reset column indicates that the field initializes to an undefined state after reset.

Field Descriptions

Read Enable (RE)—Bit 0. This bit enables reads to the defined address space.

0 = Disabled
1 = Enabled

Write Enable (WE)—Bit 1. This bit enables writes to the defined address space.

0 = Disabled

1 = Enabled

VGA Enable (VE)—Bit 4. Forces addresses in the first 64 Kbytes of PCI I/O space and where A[9:0] is in the range 3B0–3BBh or 3C0–3DFh to match against this base/limit pair independent of the base/limit addresses (see the *PCI-to-PCI Bridge Architecture* specification for a description of this function). A WE or RE bit must also be set to enable this feature.

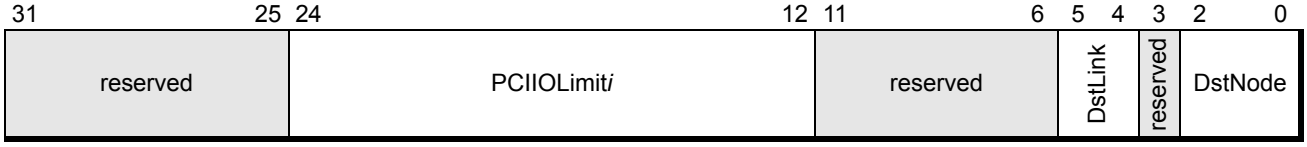
The MMIO 000A_0000h to 000B_FFFFh matching function normally associated with the VGA enable bit in PCI is NOT included in the VE bit. To enable this behavior, an MMIO register pair must be used.

ISA Enable (IE)—Bit 5. Blocks addresses in the first 64 Kbytes of PCI I/O space and in the last 768 bytes in each 1-Kbyte block from matching against this base/limit pair (see the *PCI-to-PCI Bridge Architecture* specification for a description of this function). A WE or RE bit must also be set to enable this feature.

PCI I/O Base Address *i* (PCIIOBase*i*)—Bits 24–12. This field defines the start of PCI I/O region n (where n = 0, 1, 2,3).

4.4.6.2 PCI I/O Limit *i* Registers

PCI I/O Limit 0–3 Registers **Function 1: Offset C4h, CCh, D4h, DCh**



Bits	Mnemonic	Function	R/W	Reset
31–25	reserved		R	0
24–12	PCIIOLimit*i*	PCI I/O Limit Address *i*	R/W	X
11–6	reserved		R	0
5–4	DstLink	Destination Link ID	R/W	X
3	reserved		R	0
2–0	DstNode	Destination Node ID	R/W	X

"X" in the Reset column indicates that the field initializes to an undefined state after reset.

Field Descriptions

Destination Node ID (DstNode)—Bits 2–0. This field specifies the node to which a packet is sent if it is within the address range. The bits are encoded as follows:

000b = Node 0
001b = Node 1
...
111b = Node 7

Destination Link ID (DstLink)—Bits 5–4. This field specifies the link to send the packet to once the packet has reached the destination node.

00b = Link 0
01b = Link 1
10b = Link 2
11b = reserved

PCI I/O Limit Address *i* (PCI I/OLimit*i*)—Bits 24–12. This field defines the end of PCI I/O region

4.4.7 Configuration Map Registers

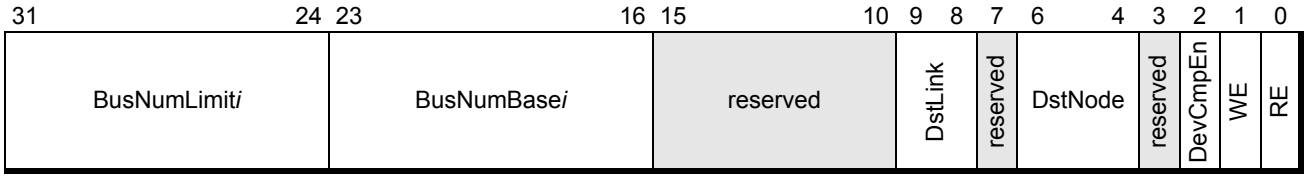
These registers define sections of the configuration bus number map for which configuration accesses should be routed to each noncoherent HyperTransport technology chain. Configuration regions must not overlap each other. For configuration accesses with bus numbers within the specified range of a base/limit pair, requests are routed to the noncoherent HyperTransport link specified by the destination Node ID and destination Link ID.

Bus numbers are considered to be within the defined range if they are greater than or equal to the base and less than or equal to the limit.

In device number compare mode (see DevCmpEn bit), the base/limit fields are interpreted as device number base/limit values for Bus 0 configuration accesses. This may be used to support configurations where Bus 0 is split between two independent noncoherent HyperTransport technology chains.

Configuration accesses to Bus 0, device 24–31, are assumed to be targeting coherent HyperTransport technology devices and are routed to the corresponding coherent HyperTransport node irrespective of the values in these registers (see Chapter 4, "Memory System Configuration"). Note that the range of device numbers affected may be modified based on the number of nodes present (see "HyperTransport™ Transaction Control Register" on page 62).

Configuration Base and Limit 0–3 Registers Function 1: Offset E0h, E4h, E8h, ECh



Bits	Mnemonic	Function	R/W	Reset
31–24	BusNumLimit*i*	Bus Number Limit *i*	R/W	X
23–16	BusNumBase*i*	Bus Number Base *i*	R/W	X
15–10	reserved		R	0
9–8	DstLink	Destination Link ID	R/W	X
7	reserved		R	0
6–4	DstNode	Destination Node ID	R/W	X

Bits	Mnemonic	Function	R/W	Reset
3	reserved		R	0
2	DevCmpEn	Device Number Compare Enable	R/W	X
1	WE	Write Enable	R/W	0
0	RE	Read Enable	R/W	0

"X" in the Reset column indicates that the field initializes to an undefined state after reset.

Field Descriptions

Read Enable (RE)—Bit 0. This bit enables reads to the defined address space.

0 = Disabled
1 = Enabled

Write Enable (WE)—Bit 1. This bit enables writes to the defined address space.

0 = Disabled
1 = Enabled

Device Number Compare Enable (DevCmpEn)—Bit 2. When this bit is set, this register defines a device number range rather than a bus number range. To match, configuration cycles must be to bus number 0 and have device numbers between BusNumBase and BusNumLimit. This is used to enable multiple noncoherent HyperTransport chains to be configured as Bus 0.

Destination Node ID (DstNode)—Bits 6–4. This field specifies the node that a packet is sent to if it is within the address range. The bits are encoded as follows:

000b = Node 0
001b = Node 1
...
111b = Node 7

Destination Link ID (DstLink)—Bits 9–8. This field specifies the link to send the packet to once it has reached the destination node and unit.

00b = Link 0
01b = Link 1
10b = Link 2
11b = reserved

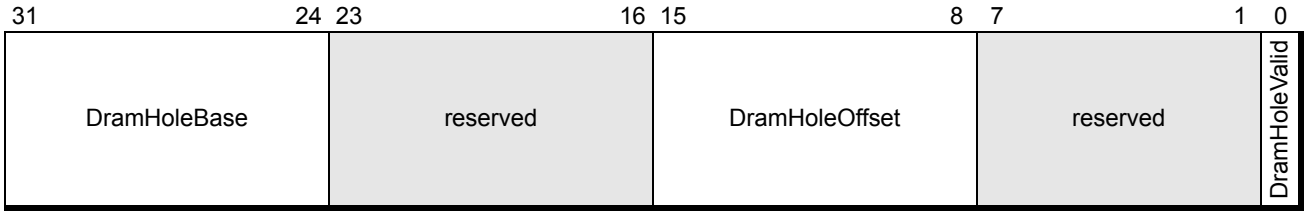
Bus Number Base *i* (BusNumBase*i*)—Bits 23–16. This field defines the lowest bus number in configuration region *i* (where *i* = 0, 1, 2, 3).

Bus Number Limit *i* (BusNumLimit*i*)—Bits 31–24. This bit field defines the highest bus number in configuration region *i* (where *i* = 0, 1, 2, 3).

4.4.8 DRAM Hole Address Register

This register is used to control memory hoisting. See Section 3.6.5 on page 44 for more information on memory hoisting.

DRAM Hole Address Register — Function 1: Offset F0h



Bits	Mnemonic	Function	R/W	Reset
31–24	DramHoleBase	DRAM Hole Base Address	R/W	0
23–16	reserved		R	0
15–8	DramHoleOffset	DRAM Hole Offset Address	R/W	0
7–1	reserved		R	0
0	DramHoleValid	DRAM Hole Valid	R/W	0

Field Descriptions

DRAM Hole Valid (DramHoleValid)—Bit 0. This bit should be set in the processor node(s) that own the DRAM address space that is hoisted above the 4GB address level. If node interleaving is enabled, then this should be set in all processors.

0 = Memory hoisting is not enabled.
1 = Memory hoisting is enabled in the processor node.

DRAM Hole Offset Bits 31-24 (DramHoleOffset)—Bits 15–8. When memory hoisting is enabled, this value is subtracted from the physical address of certain transactions before being passed to the DRAM controller.

DRAM Hole Base Bits 31-24 (DramHoleBase)—Bit 31–24. This specifies the base address of the I/O hole, below the 4G address level, that is used in memory hoisting. Normally, DramHoleBase >= TOP_MEM[31:24].

4.5 Function 2—DRAM Controller

Function 2 configuration registers are listed in Table 7.

Table 7. Function 2 Configuration Registers

Offset	Register Name				Reset	Access	Description
00h	Device ID		Vendor ID (AMD)		1102_1022h	RO	page 96
04h	Status[1]		Command		0000_0000h	RO	
08h	Base Class Code	Subclass Code	Program Interface	Revision ID	0600_0000h	RO	page 97
0Ch	BIST	Header Type	Latency Timer	Cache Line Size	0080_0000h	RO	
10h	Base Address 0				0000_0000h	RO	
14h	Base Address 1				0000_0000h	RO	
18h	Base Address 2				0000_0000h	RO	
1Ch	Base Address 3				0000_0000h	RO	
20h	Base Address 4				0000_0000h	RO	
24h	Base Address 5				0000_0000h	RO	
28h	Card Bus CIS Pointer				0000_0000h	RO	
2Ch	Sub-System ID		Sub-System Vendor ID		0000_0000h	RO	
30h	ROM Base Address				0000_0000h	RO	
34h	Capabilities				0000_0000h	RO	
38h	reserved				0000_0000h	RO	
3Ch	Max Latency	Min GNT	Int Pin	Int Line	0000_0000h	RO	
40h	DRAM CS Base 0				0000_0000h	RW	page 98
44h	DRAM CS Base 1				0000_0000h	RW	page 98
48h	DRAM CS Base 2				0000_0000h	RW	page 98
4Ch	DRAM CS Base 3				0000_0000h	RW	page 98
50h	DRAM CS Base 4				0000_0000h	RW	page 98
54h	DRAM CS Base 5				0000_0000h	RW	page 98
58h	DRAM CS Base 6				0000_0000h	RW	page 98
5Ch	DRAM CS Base 7				0000_0000h	RW	page 98
60h	DRAM CS Mask 0				0000_0000h	RW	page 103

Notes:

1. *The unimplemented registers in the standard PCI configuration space are implemented as read-only and return 0 if read.*
2. *Reads and writes to unimplemented registers in the extended PCI configuration space will result in unpredictable behavior.*

Table 7. Function 2 Configuration Registers

Offset	Register Name	Reset	Access	Description
64h	DRAM CS Mask 1	0000_0000h	RW	page 103
68h	DRAM CS Mask 2	0000_0000h	RW	page 103
6Ch	DRAM CS Mask 3	0000_0000h	RW	page 103
78h	DRAM Control	0000_0006h	RW	page 103
7Ch	DRAM Initialization	0000_0000h	RW	page 104
80h	DRAM Bank Address Mapping	0000_0000h	RW	page 106
88h	DRAM Timing Low	0000_0000h	RW	page 112
8Ch	DRAM Timing High	0000_0000h	RW	page 114
90h	DRAM Configuration Low	0000_0000h	RW	page 117
94h	DRAM Configuration High	0000_0000h	RW	page 117
98h	DRAM Controller Additional Data Offset	0000_0000h	RW	page 122
9Ch	DRAM Controller Additional Data	0000_0000h	RW	page 124
A0h	DRAM Controller Miscellaneous	0000_0000	R/W	page 134

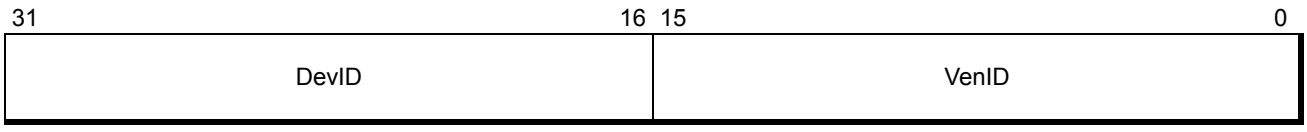
Notes:

1. *The unimplemented registers in the standard PCI configuration space are implemented as read-only and return 0 if read.*
2. *Reads and writes to unimplemented registers in the extended PCI configuration space will result in unpredictable behavior.*

4.5.1 Device/Vendor ID Register

This register specifies the device and vendor IDs for the Function 2 registers and is part of the standard PCI configuration header.

Device/Vendor ID Register — Function 2: Offset 00h



Bits	Mnemonic	Function	R/W	Reset
31–16	DevID	Device ID	R	1102h
15–0	VenID	Vendor ID	R	1022h

Field Descriptions

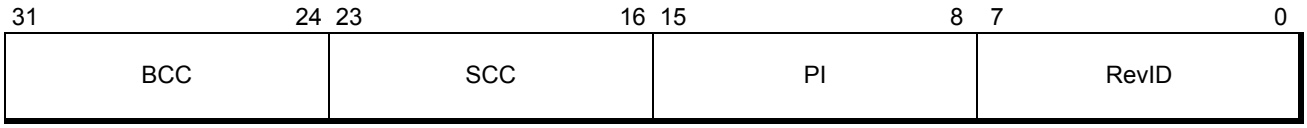
Vendor ID (VenID)—Bits 15–0. This read-only value is defined as 1022h for AMD.

Device ID (DevID)—Bits 31–16. This read-only value is defined as 1102h for the HyperTransport technology configuration function.

4.5.2 Class Code/Revision ID Register

This register specifies the class code and revision for the Function 2 registers and is part of the standard PCI configuration header.

Class Code/Revision ID Register — Function 2: Offset 08h



Bits	Mnemonic	Function	R/W	Reset
31–24	BCC	Base Class Code	R	06h
23–16	SCC	Subclass Code	R	00h
15–8	PI	Programming Interface	R	00h
7–0	RevID	Revision ID	R	00h

Field Descriptions

Revision ID (RevID)—Bits 7–0.

Programming Interface (PI)—Bits 15–8. This read-only value is defined as 00h.

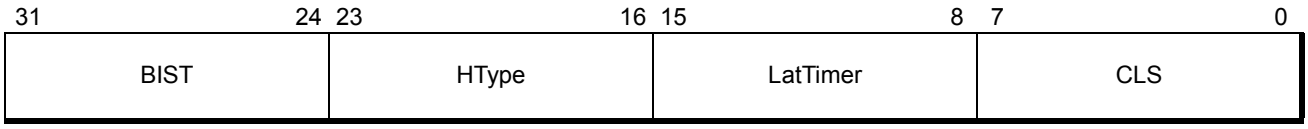
Sub Class Code (SCC)—Bits 23–16. This read-only value is defined as 00h.

Base Class Code (BCC)—Bits 31–24. This read-only value is defined as 06h for a host bridge device.

4.5.3 Header Type Register

This register specifies the header type for the Function 2 registers and is part of the standard PCI configuration header.

Header Type Register — Function 2: Offset 0Ch



Bits	Mnemonic	Function	R/W	Reset
31–24	BIST	BIST	R	00h
23–16	HType	Header Type	R	80h
15–8	LatTimer	Latency Timer	R	00h
7–0	CLS	Cache Line Size	R	00h

Field Descriptions

CacheLineSize (CLS)—Bits 7–0. This read-only value is defined as 00h.

LatencyTimer (LatTimer)—Bits 15–8. This read-only value is defined as 00h.

HeaderType (Htype)—Bits 23–16. This read-only value is defined as 80h to indicate that multiple functions are present in the configuration header and that the header layout corresponds to a device header as opposed to a bridge header.

BIST—Bits 31–24. This read-only value is defined as 00h.

4.5.4 DRAM CS Base Address Registers

These registers define DRAM chip select (CS) address mapping. They are programmed based on data in the Serial Presence Detect (SPD) ROM on each DIMM.

Function 1 address map registers define to which node to route a DRAM request. Function 2 address map registers define what DRAM chip select to access on that particular node. See "DRAM Address Map" on page 84. for more information on addresses routed to the DRAM controller (InputAddr).

The memory size of each chip select bank is defined by a DRAM CS Base Address Register and a DRAM CS Mask register. The chip selects are formed as follows, using the field names from the DRAM CS Base Address Registers and DRAM CS Mask Registers, where

"`,`" means "concatenate"
"`/`" means "not"
"==" means "equals"
"`&`" means "and"

ChipSelect[i] is asserted if the following is true:

```
CSBE[i] &
( { (InputAddr[36:27] & /AddrMaskHi[i][36:27]),
    ( InputAddr[21:13] & /AddrMaskLo[i][21:13]) } ==
  { (BaseAddrHi[i][36:27] & /AddrMaskHi[i][36:27]),
    ( BaseAddrLo[i][21:13] & /AddrMaskLo[i][21:13]) } ) );
```

There are eight DRAM CS Base Address and four DRAM CS Mask register pairs. Table 8 and Table 9 show the mapping between the DRAM CS Base and Mask registers and the logical DIMM, the chip select, the CKE pins and the ODT pins.

Table 8. DRAM CS Base Address and DRAM CS Mask Registers

Registers	Logical DIMM[1]			Chip Select	CKE	ODT
	N	R4	S4			
DRAM CS Base 0 DRAM CS Mask 0	0	0	0	M[B,A]0_CS_L[0]	M[B,A]_CKE[0]	M[B,A]0_ODT[0]
DRAM CS Base 1 DRAM CS Mask 0	0	0	0	M[B,A]0_CS_L[1]	M[B,A]_CKE[1]	M[B,A]1_ODT[0][4] M[B,A]0_ODT[1][5]
DRAM CS Base 2 DRAM CS Mask 1	1	1	0	M[B,A]1_CS_L[0][3] M[B,A]0_CS_L[2][5]	M[B,A]_CKE[0]	M[B,A]1_ODT[0][3] M[B,A]0_ODT[0][4],[5]
DRAM CS Base 3 DRAM CS Mask 1	1	1	0	M[B,A]1_CS_L[1][3] M[B,A]0_CS_L[3][5]	M[B,A]_CKE[1]	M[B,A]1_ODT[0][3] M[B,A]0_ODT[1][5]
DRAM CS Base 4 DRAM CS Mask 2	2[2]	0		M[B,A]2_CS_L[0]	M[B,A]_CKE[0]	M[B,A]2_ODT[0]
DRAM CS Base 5 DRAM CS Mask 2	2[2]	0		M[B,A]2_CS_L[1]	M[B,A]_CKE[1]	M[B,A]2_ODT[0]
DRAM CS Base 6 DRAM CS Mask 3	3[2]	1		M[B,A]3_CS_L[0]	M[B,A]_CKE[0]	M[B,A]3_ODT[0]
DRAM CS Base 7 DRAM CS Mask 3	3[2]	1		M[B,A]3_CS_L[1]	M[B,A]_CKE[1]	M[B,A]3_ODT[0]

Notes:

1. *N=Normal.*
   R4=Four-rank registered DIMM only (Function 2 Offset 94h bit 18 = 1).
   S4=Four-rank SO-DIMM only (Function 2 Offset 94h bit 17 = 1); only 4 chip selects are supported.
2. *Logical DIMM numbers 2 and 3 are not supported in the ASB1, S1g1, and AM2 packages.*
3. *F(1207) , or unbuffered DIMM on AM2 and ASB1 Packages.*
4. *SO-DIMM using an AM2 or ASB1 Package.*
5. *S1g1 Package.*

Table 9. DRAM CS Base Address and DRAM CS Mask Registers Mismatched DIMM Support[1]

Registers	Logical DIMM[2]		Chip Select	CKE	ODT
	N	S4			
DRAM CS Base 0 DRAM CS Mask 0	0	0	MA0_CS_L[0]	MA_CKE[0]	MA0_ODT[0]
DRAM CS Base 1 DRAM CS Mask 0	0	0	MA0_CS_L[1]	MA_CKE[1]	MA0_ODT[0][3] MA1_ODT[0][4] MA0_ODT[1][5]
DRAM CS Base 2 DRAM CS Mask 1	1	0	MA1_CS_L[0][3] MA0_CS_L[2][5]	MA_CKE[0]	MA1_ODT[0][3] MA0_ODT[0][4],[5]
DRAM CS Base 3 DRAM CS Mask 1	1	0	MA1_CS_L[1][3] MA0_CS_L[3][5]	MA_CKE[1]	MA1_ODT[0][3] MA0_ODT[1][5]
DRAM CS Base 4 DRAM CS Mask 2	2	1	MB0_CS_L[0]	MB_CKE[0]	MB0_ODT[0]
DRAM CS Base 5 DRAM CS Mask 2	2	1	MB0_CS_L[1]	MB_CKE[1]	MB0_ODT[0][3] MB1_ODT[0][4] MB0_ODT[1][5]
DRAM CS Base 6 DRAM CS Mask 3	3	1	MB1_CS_L[0][3] MB0_CS_L[2][5]	MB_CKE[0]	MB1_ODT[0][3] MB0_ODT[0][4],[5]
DRAM CS Base 7 DRAM CS Mask 3	3	1	MB1_CS_L[1][3] MB0_CS_L[3][5]	MB_CKE[1]	MB1_ODT[0][3] MB0_ODT[1][5]

Notes:

1. *Mod64Mux (Function 2 Offset A0h bit 4) = 1.*
2. *N=Normal.*
   S4=Four-rank SO-DIMM only (Function 2 Offset 94h bit 17 = 1); only 4 chip selects are supported.
3. *AM2 Packages.*
4. *SO-DIMM using an AM2 Package.*
5. *S1g1 Package.*

DRAM memory can be assigned to chip select banks in two ways: non-interleaving, when contiguous addresses are assigned to each chip select bank, and interleaving, when non-contiguous addresses are assigned to each chip select bank.

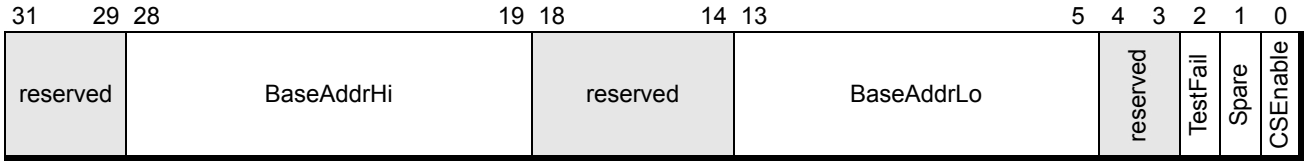
Non-interleaving mode can always be used. The BIOS must assign the largest DIMM chip-select range to the lowest address. As addresses increase, the chip select size must remain constant or decrease. This is necessary to keep DIMM chip select banks on aligned address boundaries as chip-select banks with different depths are added. The masking does not work otherwise.

Memory interleaving mode can be used if the memory system is composed of a single type and size DRAM, and if the number of chip selects is a power of two. DRAM controller swaps low order address bits with high order address bits during decoding. As a result, some low order address bits are

used to determine chip select, and some high order address bits are used to determine DIMM row. It allows that a switch between DIMMs occurs before a page conflict within a DIMM. The one cycle turnaround saves time needed to close/open new pages. Algorithm for programming DRAM CS Base Address and DRAM CS Mask registers in interleaving mode are described in section "DRAM Address Mapping in Interleaving Mode" on page 108.

The BIOS sets up the base address and base mask registers after determining the number of populated DIMMs, how many chip-select banks they have, and their sizes. A single base address is initialized for each chip select. A single mask is initialized for each chip select pair. Hardware decodes addresses and determines to which DIMM chip-select range the address maps, as a function of the base address and mask.

DRAM CS Base Address Registers — Function 2: Offset 40h, 44h, 48h, 4Ch, 50h, 54h, 58h, 5Ch



Bits	Mnemonic	Function	R/W	Reset
31–29	reserved		R	0
28–19	BaseAddrHi	Base Address (36–27)	R/W	0
18–14	reserved		R	0
13–5	BaseAddrLo	Base Address (21–13)	R/W	0
4–3	reserved		R	0
2	TestFail	Memory Test Failed	R/W	0
1	Spare	Spare Rank	R/W	0
0	CSEnable	Chip-Select Bank Enable	R/W	0

Field Descriptions

Chip-Select Bank Enable (CSEnable)—Bit 0. This bit enables access to the defined address space. The TestFail and Spare bits should never be set if this bit is set.

Spare Rank (Spare)—Bit 1. This bit identifies the chip select associated with the spare rank. The TestFail and CSEnable bits should never be set if this bit is set. This bit must be set prior to DRAM initialization.

Memory Test Failed (TestFail)—Bit 2. This bit is set by BIOS to indicate that a rank is present but the memory is bad. The CSEnable and Spare bits should never be set if this bit is set. This bit must be set prior to DRAM initialization.

Base Address (21–13) (BaseAddrLo)—Bits 13–5. This field is only used for memory interleaving to avoid page conflicts. A new chip select bank is accessed before accessing a new row in the old chip select bank, delaying or avoiding a page conflict.

If contiguous addresses from address 0 are accessed, the controller would first access a row (e.g., row 0) in bank 0, chip select 0. As the address increases, the controller would access different columns in the following order for four bank DIMMs:

row 0/bank 0/chip select 0 and then
row 0/bank 1/chip select 0 and then
row 0/bank 2/chip select 0 and then
row 0/bank 3/chip select 0 and then
row 0/bank 0/chip select 1.

As the address increases, the controller would access different columns in the following order for eight bank DIMMs:

row 0/bank 0/chip select 0 and then
row 0/bank 1/chip select 0 and then
row 0/bank 2/chip select 0 and then
row 0/bank 3/chip select 0 and then
row 0/bank 4/chip select 0 and then
row 0/bank 5/chip select 0 and then
row 0/bank 6/chip select 0 and then
row 0/bank 7/chip select 0 and then
row 0/bank 0/chip select 1.

A chip select bank does not have a contiguous region of memory assigned to it. Memory is interleaved between two DIMMs, four DIMMs, or eight DIMMs.

Base Address (36–27) (BaseAddrHi)—Bits 28–19. These bits decode 128-Mbyte blocks of memory. In the non-interleaving mode (BaseAddrLo has a value of 0), physical addresses map to DRAM addresses in the following way:

Col—Lowest order physical address bits
Bank—Second lowest order physical address bits (page miss is better than page conflict)
Row—Second highest order physical address bits (page conflict before accessing new chip)
Chip Select—Highest order physical address bits

In the non-interleaving mode, contiguous addresses from 0 would first access a row (e.g., row 0) in bank 0, chip select 0. As the address increases, the next access is to a different column and eventually a different chip select bank, e.g.:

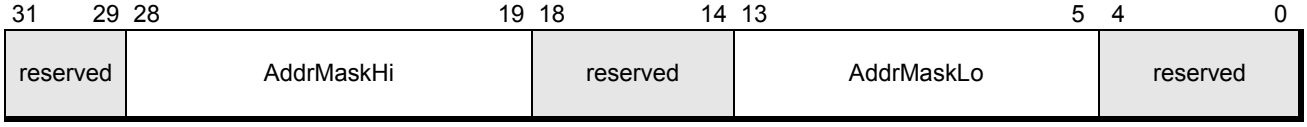
row 0/bank 0/chip select 0 and then
row 0/bank 1/chip select 0 and then
row 0/bank 2/chip select 0 and then
row 0/bank 3/chip select 0 and then
row 1/bank 0/chip select 0

Memory controller accesses all rows and banks in a single chip select bank before accessing a new chip select bank in the non-interleaving mode.

4.5.5 DRAM CS Mask Registers

The purpose of this register is to exclude the corresponding address bits from the comparison with the DRAM Base address register.

DRAM CS Mask Registers **Function 2: Offset 60h, 64h, 68h, 6Ch**



Bits	Mnemonic	Function	R/W	Reset
31–29	reserved		R	0
28–19	AddrMaskHi	Address Mask (36–27)	R/W	0
18–14	reserved		R	0
13–5	AddrMaskLo	Address Mask (21–13)	R/W	0
4–0	reserved		R	0

Field Descriptions

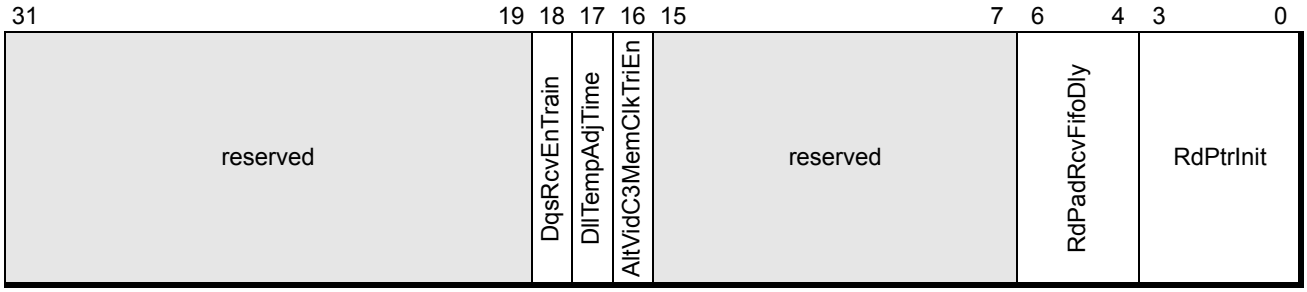
Address Mask (21–13) (AddrMaskLo)—Bits 13–5. This field specifies the addresses to be excluded for the memory interleaving mode described in the Base Address bit definitions.

Address Mask (36–27) (AddrMaskHi)—Bits 28–19. This field defines the top Address Mask bits. The bits with an address mask of 1 are excluded from the address comparison. This allows the memory block size to be larger than 128 Mbytes. If Address Mask bit 19 is set to 1, the memory block size is 256 Mbytes.

4.5.6 DRAM Control Register

This register is used to control the behavior of the DRAM controller. See "Register Differences in Revisions of AMD NPT Family 0Fh Processors" on page 27 for revision information about this Register.

DRAM Control Register **Function 2: Offset 78h**



Bits	Mnemonic	Function	R/W	Reset
31–19	reserved		R	0
18	DqsRcvEnTrain	DQS Receiver Enable Training Mode	R/W	0
17	DllTempAdjTime	DLL Temperature Adjust Cycle Time	R/W	0
16	AltVidC3MemClkTriEn	AltVID Memory Clock Tristate Enable	R/W	0
15–7	reserved		R	0
6–4	RdPadRcvFifoDly	Read Delay from Pad Receive FIFO	R/W	7h
3–0	RdPtrInit	Read Pointer Initial Value	R/W	6h

Field Descriptions

Read Pointer Initial Value (RdPtrInit)—Bits 3–0. This field specifies the value of the read pointer for the pad transmit FIFO when the read pointer is reinitialized. Writing to this register reinitializes the read and write pointers in the pad transmit FIFO. Values of 3h-0h are reserved and should never programmed into this field.

Read Delay from Pad Receive FIFO (RdPadRcvFifoDly)—Bits 6–4. This field specifies the delay from the DQS receiver enable to the first data being read from the pad receive FIFO.

000b = 0.5 Memory Clocks
001b = 1 Memory Clock
010b = 1.5 Memory Clocks
011b = 2 Memory Clocks
100b = 2.5 Memory Clocks
101b = 3 Memory Clock
110b = 3.5 Memory Clocks
111b = 4 Memory Clocks

AltVID Memory Clock Tristate Enable (AltVidC3MemClkTriEn)—Bit 16. Enables the DDR memory clocks to be tristated when alternate VID mode is enabled. This bit has no effect if the DisNbClkRamp bit (Function 3, Offset 88h) is set.

DLL Temperature Adjust Cycle Time (DllTempAdjTime)—Bit 17. This bit selects the DLL temperature adjust Cycle Time.

0 = 5ms
1 = 1ms

DQS Receiver Enable Training Mode (DqsRcvEnTrain)—Bit 18. This bit enables the DQS receiver enable training mode.

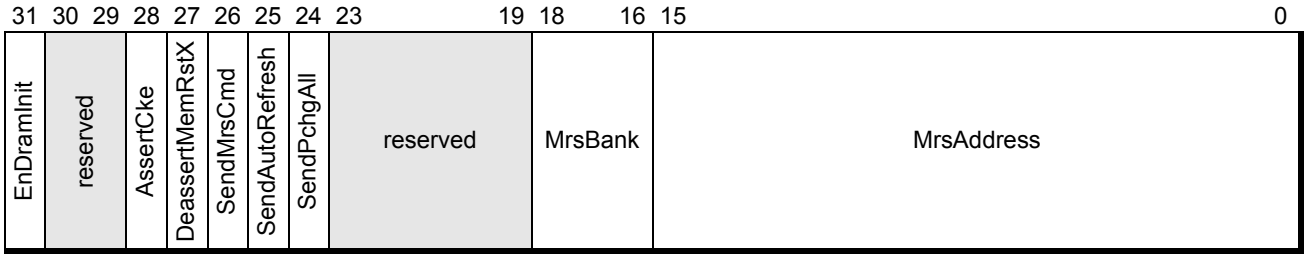
0 = Normal DQS receiver enable operation.
1 = DQS receiver enable training mode.

4.5.7 DRAM Initialization Register

This register is used by BIOS to control the DRAM initialization sequence. The BIOS is responsible for not changing the command specified until the controller has completed the command. See

"Register Differences in Revisions of AMD NPT Family 0Fh Processors" on page 27 for revision information about this Register.

DRAM Initialization Register — Function 2: Offset 7Ch



Bits	Mnemonic	Function	R/W	Reset
31	EnDramInit	Enable DRAM Initialization	R/W	0
30–29	reserved		R	0
28	AssertCke	Assert CKE	R/W	0
27	DeassertMemRstX	Deassert Memory Reset	R/W	0
26	SendMrsCmd	Send MSR/EMSR Command	R/W	0
25	SendAutoRefresh	Send Autorefresh Command	R/W	0
24	SendPchgAll	Send Precharge All Command	R/W	0
23–19	reserved		R	0
18–16	MrsBank	Bank Address for MRS/EMRS Commands	R/W	0
15–0	MrsAddress	Address for MRS/EMRS Commands	R/W	0

Field Descriptions

Address for MRS/EMRS Commands (MrsAddress)—Bits 15–0. This field specifies the data driven on the DRAM address pins 15-0 for MRS and EMRS commands.

Bank Address for MRS/EMRS Commands (MrsBank)—Bits 18–16. This field specifies the data driven on the DRAM bank pins for MRS and EMRS commands.

Send Precharge All Command (SendPchgAll)—Bit 24. Setting this bit causes the DRAM controller to send a precharge all command. This bit is cleared by the hardware after the command completes.

Send Auto Refresh Command (SendAutoRefresh)—Bit 25. Setting this bit causes the DRAM controller to send an auto refresh command. This bit is cleared by the hardware after the command completes.

Send MRS/EMRS Command (SendMrsCmd)—Bit 26. Setting this bit causes the DRAM controller to send the MRS or EMRS command defined by the MrsAddress and MrsBank fields. This bit is cleared by the hardware after the command completes.

De-assert Memory Reset (DeassertMemRstX)—Bit 27. Setting this bit causes the DRAM controller to de-assert the memory reset pin. This bit cannot be used to assert the memory reset pin.

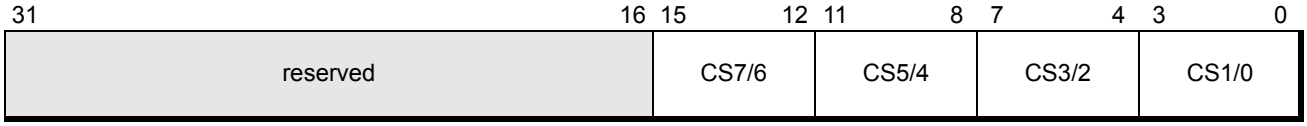
Assert CKE (AssertCke)—Bit 28. Setting this bit causes the DRAM controller to assert the CKE pins. This bit cannot be used to de-assert the CKE pins.

Enable DRAM Initialization (EnDramInit)—Bit 31. Setting this bit puts the DRAM controller in a BIOS controlled DRAM initialization mode. Setting this bit causes the DRAM controller to assert memory reset for registered DIMMs and de-assert CKE for all DIMMs. BIOS must clear this bit after DRAM initialization is complete.

4.5.8 DRAM Bank Address Mapping Register

Each DIMM may have one or two chip-select banks. The address mode covers the whole DIMM, regardless of whether there is one or two chip-select banks.

DRAM CS Address Mapping Register **Function 2: Offset 80h**



Bits	Mnemonic	Function	R/W	Reset
31–16	reserved		R	
15–12	CS7/6	CS7/6	R/W	0
11–8	CS5/4	CS5/4	R/W	0
7–4	CS3/2	CS3/2	R/W	0
3–0	CS1/0	CS1/0	R/W	0

Field Descriptions

Chip Select 1/0 (CS1/0)—Bits 3–0. This field specifies the memory module size. This field is programmed the same regardless of whether the DRAM interface is 64-bits or 128-bits wide. This field describes the type of DIMMs that compose the chip select. The bits are encoded as shown in Table 10 and Table 11.

Chip Select 3/2 (CS3/2)—Bits 7–4. This field specifies the memory module size. The bits are encoded as shown in Table 10 and Table 11.

Chip Select 5/4 (CS5/4)—Bits 11–8. This field specifies the memory module size. The bits are encoded as shown in Table 10 and Table 11.

Chip Select 7/6 (CS7/6)—Bits 15–12. This field specifies the memory module size. The bits are encoded as shown in Table 10 and Table 11.

4.5.8.1 DRAM Address Mapping

The address line mapping with a 64-bit DRAM interface is shown in Table 10 and with a 128-bit DRAM interface in Table 11. Column A10 presents the precharge all signal (PC).

Table 10: DRAM address mapping 64-bit interface

Bits	CS Size	Device size, width	Bank			Address																
			2	1	0		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0000b	128 MB	256Mb, x16	x	13	12	Row	x	x	x	17	16	15	14	26	25	24	23	22	21	20	19	18
						Col	x	x	x	x	x	AP	x	11	10	9	8	7	6	5	4	3
0001b	256 MB	256Mb, x8	x	14	13	Row	x	x	x	17	16	15	27	26	25	24	23	22	21	20	19	18
		512Mb, x16				Col	x	x	x	x	x	AP	12	11	10	9	8	7	6	5	4	3
0010b	512 MB	512Mb, x8	x	14	13	Row	x	x	17	16	15	28	27	26	25	24	23	22	21	20	19	18
						Col	x	x	x	x	x	AP	12	11	10	9	8	7	6	5	4	3
0011b	512 MB	256Mb, x4	x	15	14	Row	x	x	x	17	16	28	27	26	25	24	23	22	21	20	19	18
						Col	x	x	x	x	13	AP	12	11	10	9	8	7	6	5	4	3
0100b	512 MB	1G, x16	15	14	13	Row	x	x	x	17	16	28	27	26	25	24	23	22	21	20	19	18
						Col	x	x	x	x	x	AP	12	11	10	9	8	7	6	5	4	3
0101b	1 GB	1G, x8	15	14	13	Row	x	x	17	16	29	28	27	26	25	24	23	22	21	20	19	18
		2G, x16				Col	x	x	x	x	x	AP	12	11	10	9	8	7	6	5	4	3
0110b	1 GB	512Mb, x4	x	15	14	Row	x	x	17	16	29	28	27	26	25	24	23	22	21	20	19	18
						Col	x	x	x	x	13	AP	12	11	10	9	8	7	6	5	4	3
0111b	2 GB	2Gb, x8	15	14	13	Row	x	17	16	30	29	28	27	26	25	24	23	22	21	20	19	18
		4Gb, x16				Col	x	x	x	x	x	AP	12	11	10	9	8	7	6	5	4	3
1000b	2 GB	1Gb, x4	16	15	14	Row	x	x	17	30	29	28	27	26	25	24	23	22	21	20	19	18
						Col	x	x	x	x	13	AP	12	11	10	9	8	7	6	5	4	3
1001b	4 GB	2Gb, x4	16	15	14	Row	x	17	31	30	29	28	27	26	25	24	23	22	21	20	19	18
						Col	x	x	x	x	13	AP	12	11	10	9	8	7	6	5	4	3
1010b	4 GB	4Gb, x8	15	14	13	Row	17	16	31	30	29	28	27	26	25	24	23	22	21	20	19	18
						Col	x	x	x	x	x	AP	12	11	10	9	8	7	6	5	4	3
1011b	8 GB	4Gb, x4	16	15	14	Row	17	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18
						Col	x	x	x	x	13	AP	12	11	10	9	8	7	6	5	4	3

Table 11: DRAM address mapping 128-bit interface

Bits	CS Size	Device size, width	Bank			Address																
			2	1	0		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0000b	256 MB	256Mb, x16	x	14	13	Row	x	x	x	18	17	16	15	27	26	25	24	23	22	21	20	19
						Col	x	x	x	x	x	AP	x	12	11	10	9	8	7	6	5	4
0001b	512 MB	256Mb, x8	x	15	14	Row	x	x	x	18	17	16	28	27	26	25	24	23	22	21	20	19
		512Mb, x16				Col	x	x	x	x	x	AP	13	12	11	10	9	8	7	6	5	4
0010b	1 GB	512Mb, x8	x	15	14	Row	x	x	18	17	16	29	28	27	26	25	24	23	22	21	20	19
						Col	x	x	x	x	x	AP	13	12	11	10	9	8	7	6	5	4

Table 11: DRAM address mapping 128-bit interface

Bits	CS Size	Device size, width	Bank			Address															
			2	1	0		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0011b	1 GB	256Mb, x4	x	16	15	Row	x	x	x	18	17	29	28	27	26	25	24	23	22	21	20	19
						Col	x	x	x	x	14	AP	13	12	11	10	9	8	7	6	5	4
0100b	1 GB	1G, x16	16	15	14	Row	x	x	x	18	17	29	28	27	26	25	24	23	22	21	20	19
						Col	x	x	x	x	x	AP	13	12	11	10	9	8	7	6	5	4
0101b	2 GB	1G, x8 2G, x16	16	15	14	Row	x	x	18	17	30	29	28	27	26	25	24	23	22	21	20	19
						Col	x	x	x	x	x	AP	13	12	11	10	9	8	7	6	5	4
0110b	2 GB	512Mb, x4	x	16	15	Row	x	x	18	17	30	29	28	27	26	25	24	23	22	21	20	19
						Col	x	x	x	x	14	AP	13	12	11	10	9	8	7	6	5	4
0111b	4 GB	2Gb, x8 4Gb, x16	16	15	14	Row	x	18	17	31	30	29	28	27	26	25	24	23	22	21	20	19
						Col	x	x	x	x	x	AP	13	12	11	10	9	8	7	6	5	4
1000b	4 GB	1Gb, x4	17	16	15	Row	x	x	18	31	30	29	28	27	26	25	24	23	22	21	20	19
						Col	x	x	x	x	14	AP	13	12	11	10	9	8	7	6	5	4
1001b	8 GB	2Gb, x4	17	16	15	Row	x	18	32	31	30	29	28	27	26	25	24	23	22	21	20	19
						Col	x	x	x	x	14	AP	13	12	11	10	9	8	7	6	5	4
1010b	8 GB	4Gb, x8	16	15	14	Row	18	17	32	31	30	29	28	27	26	25	24	23	22	21	20	19
						Col	x	x	x	x	x	AP	13	12	11	10	9	8	7	6	5	4
1011b	16 GB	4Gb, x4	17	16	15	Row	18	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19
						Col	x	x	x	x	14	AP	13	12	11	10	9	8	7	6	5	4

4.5.8.2 DRAM Address Mapping in Interleaving Mode

In memory interleaving mode all DIMM chip-select ranges are the same size and type, and the number of chip selects is a power of two. A BIOS algorithm for programming DRAM CS Base Address and DRAM CS Mask registers in memory interleaving mode is:

4. Program all DRAM CS Base Address and DRAM CS Mask registers using contiguous mapping.
5. For each enabled chip select, swap BaseAddrHi bits with BaseAddrLo bits, as defined Table 12 and Table 13.
6. For each enabled chip select, swap AddrMaskHi bits with AddrMaskLo bits, as defined in Table 12 and Table 13.

Table 12. Swapped physical address lines for interleaving with 64-bit interface

Chip Select Mode	Chip Select Size	Swapped Base Address and Address Mask bits		
		8 way interleaving	4 way interleaving	2 way interleaving
0000b	128-MB	[29:27] and [16:14]	[28:27] and [15:14]	[27] and [14]
0001b	256-MB	[30:28] and [17:15]	[29:28] and [16:15]	[28] and [15]
0010b	512-MB	[31:29] and [17:15]	[30:29] and [16:15]	[29] and [15]

Table 12. Swapped physical address lines for interleaving with 64-bit interface

Chip Select Mode	Chip Select Size	Swapped Base Address and Address Mask bits		
		8 way interleaving	4 way interleaving	2 way interleaving
0011b	512-MB	[31:29] and [18:16]	[30:29] and [17:16]	[29] and [16]
0100b	512-MB	[31:29] and [18:16]	[30:29] and [17:16]	[29] and [16]
0101b	1-GB	[32:30] and [18:16]	[31:30] and [17:16]	[30] and [16]
0110b	1-GB	[32:30] and [18:16]	[31:30] and [17:16]	[30] and [16]
0111b	2-GB	[33:31] and [18:16]	[32:31] and [17:16]	[31] and [16]
1000b	2-GB	[33:31] and [19:17]	[32:31] and [18:17]	[31] and [17]
1001b	4-GB	[34:32] and [19:17]	[33:32] and [18:17]	[32] and [17]
1010b	4-GB	[34:32] and [18:16]	[33:32] and [17:16]	[32] and [16]
1011b	8-GB	[35:33] and [19:17]	[34:33] and [18:17]	[33] and [17]

Table 13. Swapped physical address lines for interleaving with 128-bit interface

Chip Select Mode	Chip Select Size	Swapped Base Address and Address Mask bits		
		8 way interleaving	4 way interleaving	2 way interleaving
0000b	256-MB	[30:28] and [17:15]	[29:28] and [16:15]	[28] and [15]
0001b	512-MB	[31:29] and [18:16]	[30:29] and [17:16]	[29] and [16]
0010b	1-GB	[32:30] and [18:16]	[31:30] and [17:16]	[30] and [16]
0011b	1-GB	[32:30] and [19:17]	[31:30] and [18:17]	[30] and [17]
0100b	1-GB	[32:30] and [19:17]	[31:30] and [18:17]	[30] and [17]
0101b	2-GB	[33:31] and [19:17]	[32:31] and [18:17]	[31] and [17]
0110b	2-GB	[33:31] and [19:17]	[32:31] and [18:17]	[31] and [17]
0111b	4-GB	[34:32] and [19:17]	[33:32] and [18:17]	[32] and [17]
1000b	4-GB	[34:32] and [20:18]	[33:32] and [19:18]	[32] and [18]
1001b	8-GB	[35:33] and [20:18]	[34:33] and [19:18]	[33] and [18]
1010b	8-GB	[35:33] and [19:17]	[34:33] and [18:17]	[33] and [17]
1011b	16-GB	[36:34] and [20:18]	[35:34] and [19:18]	[34] and [18]

Example. DRAM memory consists of two 2-sided DIMMs with 256 MBytes on each side. A 64-bit interface to DRAM is used.

1. Register settings for contiguous memory mapping are:

   Function 2, Offset 80h = 0000_0011h // CS0/1 = 256 MB; CS2/3 = 256 MB

   Function 2, Offset 40h = 0000_0001h // 0 MB base

   Function 2, Offset 44h = 0010_0001h // 256 MB base = 0 MB + 256 MB

   Function 2, Offset 48h = 0020_0001h // 512 MB base = 256 MB + 256 MB

Function 2, Offset 4Ch = 0030_0001h // 768 MB base = 512 MB + 256 MB

Function 2, Offset 60h = 0008_3FF0h // CS0/CS1 = 256 MB

Function 2, Offset 64h = 0008_3FF0h // CS2/CS3 = 256 MB

2. Base Address bits to be swapped are defined in Table 12 (64-bit interface), 256MByte row (chip select size), 4 way interleaving column (4 chip selects are used). Base Address bits [29:28] are defined with BaseAddrHi[21:20]. Base Address bits [16:15] are defined with BaseAddrLo[8:7].

   Function 2, Offset 40h=0000_0001h

   Function 2, Offset 44h=0000_0081h

   Function 2, Offset 48h=0000_0101h

   Function 2, Offset 4Ch=0000_0181h

3. Address Mask bits to be swapped are the same as the Base Address bits defined in the previous step. Address Mask bits [29:28] are defined with AddrMaskHi[21:20]. Address Mask bits [16:15] are defined with AddrMaskLo[8:7].

   Function 2, Offset 60h=0038_3E70h

   Function 2, Offset 64h=0038_3E70h

4.5.9 Address to Node/Chip Select Translation

The system address map consists of DRAM memory and memory mapped I/O space. A system address is mapped to a node with DRAM base registers (Function 1, Offsets 40h, 48h, etc.) and DRAM limit registers (Function 1, Offsets 44h, 4Ch, etc.). A system address (SysAddr) that is within the address range for a node, and is not a memory mapped I/O address, is normalized (InputAddr) and forwarded to the DRAM controller on that node (see 4.4.4 for details on SysAddr to InputAddr translation). A DRAM address (InputAddr) is mapped to DRAM chip selects with DRAM CS base address registers (Function 2, Offsets 40h, 44h, etc.) and DRAM CS mask registers (Function 2, Offsets 60h, 64h, etc.) (see 4.5.4 and 4.5.5).

The following algorithm is designed to be used to determine the node and the chip select for a system address that maps to DRAM. SystemAddr is a 32 bit input variable that is equal to bits 39-8 of the system address. CSFound, NodeID, and CS are output variables. If CSFound is equal to 1, than NodeID and CS outputs are equal to the node and the chip select that corresponds to the input address.

If the On-line Spare feature is enabled BIOS assigns one of the chip-selects, (constant) SPARE_RANK, to be the spare rank in the event of a DIMM failure precondition. If the DIMM failure precondition occurs and the data of the failing rank is copied over, the spare rank decodes to the same system address range as the failing rank (BadDramCs).

```
(int,int,int) TranslateSysAddrToCS((uint32)SystemAddr){
int    CSFound, NodeID, CS, F1Offset, F2Offset, F2MaskOffset, Ilog;
uint32 IntlvEn, IntlvSel;
uint32 DramBase, DramLimit, DramEn;
uint32 HoleOffset, HoleEn;
uint32 CSBase, CSLimit, CSMask, CSEn;
```

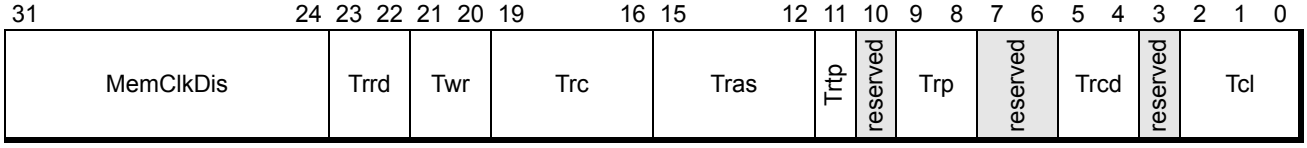
```
uint32 InputAddr, Temp;
uint32 OnlineSpareCTL;
int    SwapDone, BadDramCs;

CSFound = 0;
for(NodeID = 0; NodeID < 8; NodeID++){
    F1Offset = 0x40 + (NodeID << 3);
    DramBase = Get_PCI(bus0, dev24 + NodeID, func1, F1Offset);
    DramEn = DramBase & 0x00000003;
    IntlvEn = (DramBase & 0x00000700) >> 8;
    DramBase = DramBase & 0xFFFF0000;
    DramLimit = Get_PCI(bus0, dev24 + NodeID, func1, F1Offset + 4);
    IntlvSel = (DramLimit & 0x00000700) >> 8;
    DramLimit = DramLimit | 0x0000FFFF;
    HoleEn = Get_PCI(bus0, dev24 + NodeID, func1, 0xF0);
    HoleOffset = (HoleEn & 0x0000FF00) << 8;
    HoleEn = (HoleEn &0x00000001);
    if(DramEn && DramBase <= SystemAddr && SystemAddr <= DramLimit){
        If(HoleEn && SystemAddr > 0x00FFFFFF)
            InputAddr = SystemAddr - HoleOffset;
        Else
            InputAddr = SystemAddr - DramBase;
        if(IntlvEn){
            if(IntlvSel == ((SystemAddr >> 4) & IntlvEn)){
                if(IntlvEn == 1) Ilog = 1;
                else if(IntlvEn == 3) Ilog = 2;
                else if(IntlvEn == 7) Ilog = 3;
                else break;
                Temp = (InputAddr >> (4 + Ilog)) << 4;
                InputAddr = (Temp | (SystemAddr & 0x0000000F));
            }
            else continue;
        }
        for(CS = 0; CS < 8; CS++){
            F2Offset = 0x40 + (CS << 2);
            if ((CS % 2) == 0){
                F2MaskOffset = 0x60 + (CS << 1);
            }
            else{
                F2MaskOffset = 0x60 + ((CS-1) << 1);
            }
            CSBase = Get_PCI(bus0, dev24 + NodeID, func2, F2Offset);
            CSEn = CSBase & 0x00000001;
            CSBase = CSBase & 0x1FF83FE0;
            CSMask = Get_PCI(bus0, dev24 + NodeID, func2, F2MaskOffset);
            CSMask = (CSMask | 0x0007C01F) & 0x1FFFFFFF;
            if(CSEn && ((InputAddr & ~CSMask) == (CSBase & ~CSMask))){
                CSFound = 1;
                OnlineSpareCTL = Get_PCI(bus0, dev24 + NodeID, func3, 0xB0);
                SwapDone = (OnlineSpareCTL >> 1) & 0x00000001;
                BadDramCs = (OnlineSpareCTL >> 4) & 0x00000007;
```

```
                if(SwapDone && CS == BadDramCs) CS=SPARE_RANK;
                break;
            }
        }
    }
    if(CSFound) break;
}
return(CSFound,NodeID,CS);
}
```

4.5.10 DRAM Timing Low Register

This register contains the normal timing parameters specified in a DRAM data sheet.

DRAM Timing Low Register — Function 2: Offset 88h



Bits	Mnemonic	Function	R/W	Reset
31–24	MemClkDis	MEMCLK Disable	R/W	FFh
23–22	Trrd	Active-to-active (RAS#-to-RAS#) Delay	R/W	0
21–20	Twr	Write Recovery Time	R/W	0
19–16	Trc	Row Cycle Time	R/W	0
15–12	Tras	Minimum RAS# Active Time	R/W	0
11	Trtp	Read CAS# to Precharge Time	R/W	0
10	reserved		R	0
9–8	Trp	Row Precharge Time	R/W	0
7–6	reserved		R	0
5–4	Trcd	RAS#-active to CAS#-read/write Delay	R/W	0
3	reserved		R	0
2–0	Tcl	CAS# Latency	R/W	0

Field Descriptions

CAS# Latency (Tcl)—Bits 2–0. Specifies the CAS#-to-read-data-valid.

000b = reserved
001b = reserved
010b = CL=3
011b = CL=4
100b = CL=5
101b = CL=6
11xb = reserved

RAS# Active to CAS# read/write Delay (Trcd)—Bits 5–4. Specifies the RAS# to CAS# delay for a read/write command to the same bank.

00b = 3 Clocks
01b = 4 Clocks
10b = 5 Clocks
11b = 6 Clocks

Row Precharge Time (Trp)—Bits 9–8. Specifies the row precharge time (Precharge-to-Active or Auto-Refresh of the same bank.)

00b = 3 Clocks
01b = 4 Clocks
10b = 5 Clocks
11b = 6 Clocks

Read to Precharge Time (Trtp)—Bit 11. Specifies the read CAS# to precharge time command bus separation by the controller. This field should not be confused with tRTP, which is the internal DRAM timing as is specified by the DRAM data sheet and also SPD byte 38. The recommended programming of this field varies based on DRAM speed.

0b = 2 Clocks for Burst Length of 32 Bytes (DDR400, DDR533)
4 Clocks for Burst Length of 64 Bytes (DDR400, DDR533)
1b = 3 Clocks for Burst Length of 32 Bytes (DDR667, DDR800)
5 Clocks for Burst Length of 64 Bytes (DDR667, DDR800)

Minimum RAS# Active Time (Tras)—Bits 15–12. Specifies the minimum RAS# active time.

0000b = reserved
0001b = reserved
0010b = 5 bus clocks
...
1111b = 18 bus clocks

Row Cycle Time (Trc)—Bits 19–16. RAS#-active to RAS#-active or auto refresh of the same bank. These bits are encoded as follows:

0000b = 11 bus clocks
0001b = 12 bus clocks
. . .
1110b = 25 bus clocks
1111b = 26 bus clocks

Write Recovery Time (Twr)—Bit 21–20. Measures when the last write datum is safely registered by the DRAM. It measures from the last data to precharge (writes can go back-to-back).

00b = 3 bus clocks
01b = 4 bus clocks
10b = 5 bus clocks
11b = 6 bus clocks

Active-to-active (RAS#-to-RAS#) Delay (Trrd)—Bits 23–22. Specifies the active-to-active delay (RAS#-to-RAS#) of different banks.

00b = 2 bus clocks
01b = 3 bus clocks
10b = 4 bus clocks
11b = 5 bus clocks

MEMCLK disable (MemClkDis)—Bits 31–24. These bits disable the MEMCLK outputs for DRAM channel A. BIOS should leave all bits corresponding to unused clocks set to reduce power consumption. These bits are mapped to packages as follows:

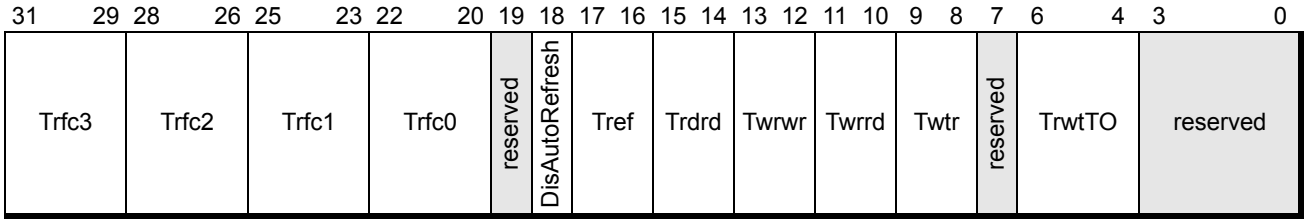
:

Bit	F(1207) Package	AM2 Package	S1g1 Package	ASB1 Package
0	N/A	MA1_CLK1	N/A	N/A
1	N/A	MA0_CLK1	MA0_CLK1	MA0_CLK1
2	MA3_CLK	N/A	N/A	N/A
3	MA2_CLK	N/A	N/A	N/A
4	MA1_CLK	MA1_CLK0	N/A	N/A
5	MA0_CLK	MA0_CLK0	MA0_CLK0	MA0_CLK0
6	N/A	MA1_CLK2	N/A	N/A
7	N/A	MA0_CLK2	MA0_CLK2	MA0_CLK2

4.5.11 DRAM Timing High Register

This register contains the normal timing parameters specified in a DRAM data sheet.

DRAM Timing High Register — Function 2: Offset 8Ch



\

Bits	Mnemonic	Function	R/W	Reset
31–29	Trfc3	Auto-Refresh Row Cycle Time for Logical DIMM3	R/W	0
28–26	Trfc2	Auto-Refresh Row Cycle Time for Logical DIMM2	R/W	0
25–23	Trfc1	Auto-Refresh Row Cycle Time for Logical DIMM1	R/W	0
22–20	Trfc0	Auto-Refresh Row Cycle Time for Logical DIMM0	R/W	0
19	reserved		R	0
18	DisAutoRefresh	Disable Automatic Refresh	R/W	0
17–16	Tref	Refresh Rate	R/W	0

Bits	Mnemonic	Function	R/W	Reset
15–14	Trdrd	Read to Read Timing	R/W	0
13–12	Twrwr	Write to Write Timing	R/W	0
11–10	Twrrd	Write to Read DIMM Termination Turnaround	R/W	0
9–8	Twtr	Internal DRAM write to Read Command Delay	R/W	0
7	reserved		R	0
6–4	TrwtTO	Read-to-Write Turnaround for Data, DQS Contention	R/W	0
3–0	reserved		R	0

Field Descriptions

Read-to-Write Turnaround for Data, DQS Contention (TrwtTO)—Bits 6–4. This specifies the minimum number of cycles from the last clock of virtual CAS (defined as the clock in which CAS is asserted for the burst, N, plus the burst length, BL, minus 1; so the last clock of virtual CAS = N + BL - 1) of a first read operation to the clock in which CAS is asserted for a following write operation. Time may need to be inserted to insure there is no bus contention on bidirectional pins.

000b = 2 bus clocks
001b = 3 bus clocks
010b = 4 bus clocks
011b = 5 bus clocks
100b = 6 bus clocks
101b = 7 bus clocks
110b = 8 bus clocks
111b = 9 bus clocks

Internal DRAM Write-to-Read Command Delay (Twtr)—Bit 9–8. These bits specify the minimum write-to-read delay when both access the same chip select.

00b = reserved
01b = 1 bus clocks
10b = 2 bus clocks
11b = 3 bus clocks

Write to Read DIMM Termination Turnaround (Twrrd)—Bits 11–10. These bits specify the minimum write-to-read delay when accessing two different DIMMs.

00b = 0 bus clocks
01b = 1 bus clocks
10b = 2 bus clocks
11b = 3 bus clocks

Write to Write Timing (Twrwr)—Bits 13–12. This specifies the minimum number of cycles from the last clock of virtual CAS (defined as the clock in which CAS is asserted for the burst, N, plus the burst length, BL, minus 1; so the last clock of virtual CAS = N + BL - 1) of the first write-burst operation to the clock in which CAS is asserted for a following write-burst

operation that changes the terminator that is enabled. Time must be inserted between consecutive writes to account for termination timing on DDR2 devices.

00b = 1 bus clocks (0 idle cycles on the bus)
01b = 2 bus clocks (1 idle cycle on the bus)
10b = 3 bus clocks (2 idle cycles on the bus)
11b = reserved

Read to Read Timing (Trdrd)—Bits 15–14. This specifies the minimum number of cycles from the last clock of virtual CAS (defined as the clock in which CAS is asserted for the burst, N, plus the burst length, BL, minus 1; so the last clock of virtual CAS = N + BL - 1) of the first read-burst operation to the clock in which CAS is asserted for a following read-burst operation that is to a different chip select than the first read-burst operation. Time must be inserted between consecutive reads to different chip selects to account for (1) turn-around timing and (2) termination timing.

00b = 2 bus clocks (1 idle cycle on the bus)
01b = 3 bus clocks (2 idle cycle on the bus)
10b = 4 bus clocks (3 idle cycle on the bus)
11b = 5 bus clocks (4 idle cycle on the bus)

Refresh Rate (Tref)—Bits 17–16. This specifies the time required to refresh all DRAM devices.

00b = Undefined behavior.
01b = Reserved
10b = Refresh interval of 7.8 microseconds
11b = Refresh interval of 3.9 microseconds

Disable Automatic Refresh (DisAutoRefresh)—Bit 18. Setting this bit disables automatic refresh to the DIMMs. BIOS should normally leave this bit cleared. This function is sometimes useful during electrical characterization.

Auto-Refresh Row Cycle Time for Logical DIMM0 (Trfc0)—Bits 22–20. This specifies the minimum time from an auto-refresh command to an activate command or another auto refresh command. The recommended programming of this register varies based on DRAM density and speed.

000b = 75 ns (all speeds, 256 Mbit)
001b = 105 ns all speeds, (512 Mbit)
010b = 127.5 ns (all speeds, 1 Gbit)
011b = 195 ns (all speeds, 2 Gbit)
100b = 327.5 ns (all speeds, 4 Gbit)
101b = reserved
110b = reserved
111b = reserved

Auto-Refresh Row Cycle Time for Logical DIMM1 (Trfc1)—Bits 25–23. See Trfc0.

Auto-Refresh Row Cycle Time for Logical DIMM2 (Trfc2)—Bits 28–26. See Trfc0.

Auto-Refresh Row Cycle Time for Logical DIMM3 (Trfc3)—Bits 31–29. See Trfc0.

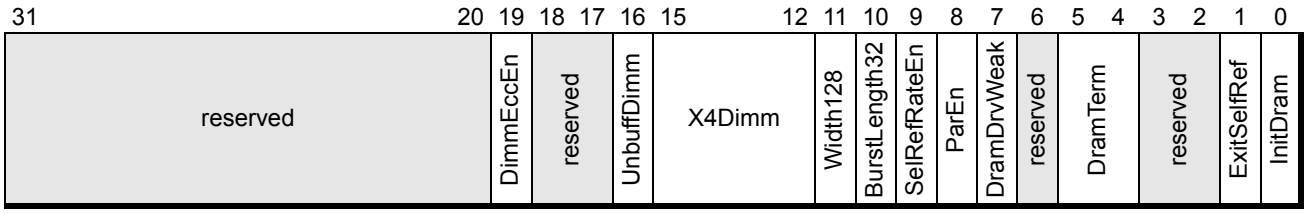
4.5.12 DRAM Configuration Registers

The following registers contain all the bits to configure the DRAMs for proper operation.

4.5.12.1 DRAM Configuration Low Register

This register controls drive strength, refresh, and initialization.

DRAM Configuration Low Register **Function 2: Offset 90h**



Bits	Mnemonic	Function	R/W	Reset
31–20	reserved		R	0
19	DimmEccEn	DIMM ECC Enable	R/W	0
18–17	reserved		R	0
16	UnbuffDimm	Unbuffered DIMM	R/W	0
15–12	X4Dimm	x4 DIMMs	R/W	0
11	Width128	Width of DRAM Interface in 128-bit Mode	R/W	0
10	BurstLength32	DRAM Burst Length Set for 32 Bytes	R/W	0
9	SelRefRateEn	Faster Self Refresh Rate Enable	R/W	0
8	ParEn	Parity Enable	R/W	0
7	DramDrvWeak	DRAM Drivers Weak Mode	R/W	0
6	reserved		R	0
5–4	DramTerm	DRAM Termination	R/W	0
3–2	reserved		R	0
1	ExitSelfRef	Exit Self Refresh Command	R/W	0
0	InitDram	Initialize DRAM	R/W	0

Field Descriptions

Initialize DRAM (InitDram)—Bit 0. Writing a 1 to this bit causes the DRAM controller to execute the DRAM initialization sequence described by the JEDEC specification. This command should be executed by BIOS when booting from an unpowered state (ACPI S4, S5 or G3; not S3 or suspend to RAM), after the DRAM controller configuration registers are properly initialized. This bit is read as a 1 while the DRAM initialization sequence is executing; it is read as 0 at all other times. The updated values of the other changed fields in this register are

used in the initialization sequence when InitDram is written to 1 (they do not need to be set up correctly before writing InitDram to 1).

Exit Self Refresh Command (ExitSelfRef)—Bit 1. Writing a 1 to this bit causes the DRAM controller to bring the DRAMs out of self refresh mode. This command should be executed by BIOS when returning from the suspend to RAM state, after the DRAM controller configuration registers are properly initialized. This bit is read as a 1 while the exit-self-refresh command is executing; it is read as 0 at all other times.

DRAM Termination (DramTerm)—Bits 4–5. This specifies the programming of the DRAM termination value (Rtt) when the EMRS command is issued during DRAM initialization.

00b = On die termination disabled.
01b = 75 ohms.
10b = 150 ohms.
11b = 50 ohms.

DRAM Drivers Weak Mode (DramDrvWeak)—Bit 7. This specifies the programming of the DRAM data drive strength mode when the EMRS command is issued during DRAM initialization.

0 = Normal drive strength mode.
1 = Weak drive strength mode.

Parity Enable (ParEn)—Bit 8. This bit enables the DRAM command and address parity error input, M[B,A]ERR_L. Clearing this bit disables the input into the error reporting logic of the processor.

Faster Self Refresh Rate Enable (SelfRefRateEn)—Bit 9. This bit enables high temperature (two times normal) self refresh rate. This bit is reflected in the EMRS(2) command to the DRAM devices.

DRAM Burst Length Set for 32 Bytes (BurstLength32)—Bit 10. This specifies the burst length of DRAM accesses and, as a result, the number of data bytes exchanged in each access.

0 = 64-byte mode.
1 = 32-byte mode.

32-byte mode may be preferred in platforms that include graphics controllers that generate a lot of 32-byte system memory accesses. 32-byte mode is not supported when the DRAM interface is 128 bits wide; so this bit interacts with Width128 as follows:

BurstLength32	Width128	Description
0	0	8-beat burst length; 64-byte accesses
0	1	4-beat burst length; 64-byte accesses
1	0	4-beat burst length; 32-byte accesses
1	1	4-beat burst length; 64-byte accesses

Width of DRAM Interface (Width128)—Bit 11. This controls the width of the DRAM interface.

0 = The controller DRAM interface is 64-bits wide
1 = The controller DRAM interface is 128-bits wide.

X4 DIMMs (X4Dimm)—Bits 15–12. Indicates whether each of the four logical DIMMs are x4 or not.

0 = DIMM is not x4
1 = x4 DIMM present

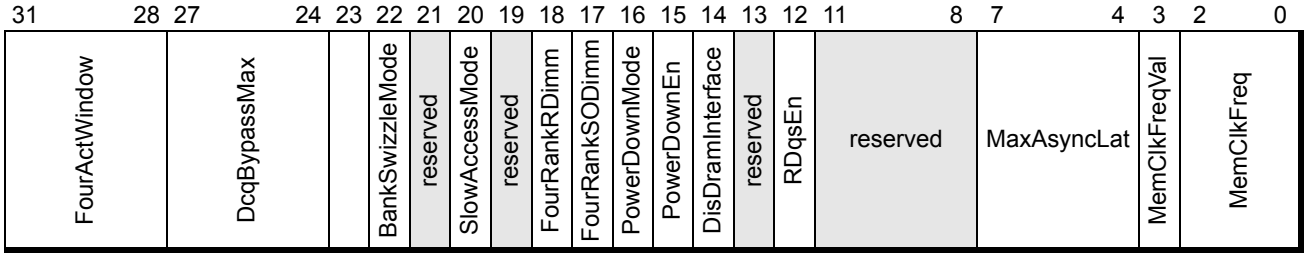
Unbuffered DIMMs (UnBuffDimm)—Bit 16. When set, the DRAM controller is only connected to unbuffered DIMMs.

0 = Buffered DIMMs
1 = Unbuffered DIMMs

DIMM ECC Enable (DimmEccEn)—Bit 19. ECC checking is capable of being enabled for all DIMMs on the DRAM controller (through Function 3 Offset 44h [EccEn]). This bit should not be set unless all populated DIMMs support ECC check bits.

4.5.12.2 DRAM Configuration High Register

DRAM Configuration High Register **Function 2: Offset 94h**



Bits	Mnemonic	Function	R/W	Reset
31–28	FourActWindow	Four Bank Activate Window	R/W	0
27–24	DcqBypassMax	DRAM Controller Queue Bypass Maximum	R/W	0
22	BankSwizzleMode	Bank Swizzle Mode	R/W	0
21	reserved		R	0
20	SlowAccessMode	Slow Access Mode(2T Mode)	R/W	0
19	reserved		R	0
18	FourRankRDimm	Four Rank Registered DIMM	R/W	0
17	FourRankSODimm	Four Rank SO-DIMM	R/W	0
16	PowerDownMode	Power Down Mode	R/W	0
15	PowerDownEn	Power Down Mode Enable	R/W	0
14	DisDramInterface	Disable the DRAM Interface	R/W	0
13	reserved		R	0
12	RDqsEn	Read DQS Enable	R/W	0
11–8	reserved		R	0

Bits	Mnemonic	Function	R/W	Reset
7–4	MaxAsyncLat	Maximum Asynchronous Latency	R/W	
3	MemClkFreqVal	Memory Clock Frequency Valid	R/W	0
2–0	MemClkFreq	Memory Clock Frequency	R/W	0

Field Descriptions

Memory Clock Frequency (MemClkFreq)—Bits 2–0. This field specifies the frequency of the DRAM interface (MEMCLK).

000b = 200 MHz
001b = 266 MHz
010b = 333 MHz
011b = 400 MHz
1xxb = reserved

Memory Clock Frequency Valid (MemClkFreqVal)—Bit 3. System BIOS should set this bit when setting up MemClkFreq to the proper value. This indicates to the DRAM controller that it may start driving MEMCLK at the proper frequency. BIOS must wait a minimum 20 MEMCLKs after setting this bit before initializing DRAM by setting InitDram (Function 2, Offset 90h), ExitSelfRef (Function 2, Offset 90h) or EnDramInit (Function 2, Offset 7Ch)

Maximum Asynchronous Latency (MaxAsyncLat)—Bits 7–4. This field should be programmed by system BIOS to specify the maximum round trip latency in the system from the processor to the DRAM devices and back. The DRAM controller uses this to help determine when incoming DRAM read data can be safely transferred to the core clock domain. See 5.1.2.5 on page -200 for more information on how to program this field.

0000b = 0 ns
...
1111b = 15 ns

Read DQS Enable (RDqsEn)—Bit 12. When this bit is set, bit A11 is set when the EMRS(1) command is sent during DRAM initialization and the DRAM controller uses the DM pins as read DQS pins and disables data masking. This bit should only be set if x8 registered DIMMs are present in the system.

0 = DM pins function as data mask pins.
1 = DM pins function as read DQS pins.

Disable the DRAM Interface (DisDramInterface)—Bit 14. When this bit is set, the DRAM controller is disabled and the DRAM interface is placed into a low power state. This bit is normally set if there are no DIMMs connected to the DCT. To maximize power savings when this bit is set, all of the MemClkDis bits should also be set.

0 = Enabled (default)
1 = Disabled

Power Down Mode Enable (PowerDownEn)—Bit 15. When in power down mode, if all pages of the DRAMs associated with a CKE pin are closed, then these parts are placed in power down mode. Only pre-charge power down mode is supported, not active power down mode.

0 = Disabled (default)
1 = Enabled

Power Down Mode (PowerDownMode)—Bit 16. This bit is initialized based on the type of system being implemented. For non-mobile systems, power down mode should be set to channel CKE control.

A DIMM or a group of DIMMs enters power down mode by deasserting the corresponding clock enable signal when the DRAM controller detects that there are no transactions scheduled to any of the DIMMs connected to the clock enable signal. A DIMM or a group of DIMMs exits power down mode by asserting the corresponding clock enable signal when a transaction is scheduled to any DIMM connected to the clock enable signal.

0 = Channel CKE control. The DRAM channel is placed in power down when all chip selects associated with the channel are idle.

1 = Chip Select CKE control. A chip select or pair of chip selects is placed in power down when no transactions are pending for the chip select(s).

Four Rank SO-DIMM (FourRankSODimm)—Bit 17. This bit is set by BIOS to indicate that a four rank SO-DIMM is present.

Four Rank Registered DIMM (FourRankRDimm)—Bit 18. This bit is set by BIOS to indicate that a four rank registered DIMM is present.

Slow Access Mode(2T Mode) (SlowAccessMode)—Bit 20. This bit controls when 2T timing is used. 2T mode may be needed in order to meet electrical requirements of certain DIMM speed and loading configurations.

0 = DRAM address and control signals are driven for one MEMCLK cycle.
1 = One additional MEMCLK of setup time is provided on all DRAM address and control signals except CS, CKE, and ODT; i.e., these signals are driven for two MEMCLK cycles rather than one.

Bank Swizzle Mode (BankSwizzleMode)—Bit 22. This bit remaps the DRAM device bank address bits as a function of normalized physical address bits. Each of the bank address bits, as specified in Table 10 and Table 11, are remapped as follows:

	64-Bit Interface	128-Bit Interface
Bank Address 0	BA0 ^ A17 ^ A22	BA0 ^ A18 ^ A23
Bank Address 1	BA1 ^ A18 ^ A23	BA1 ^ A19 ^ A24
Bank Address 2	BA2 ^ A19 ^ A24	BA2 ^ A20^ A25

For example, encoding 06h of Table 10 would be remapped from bank[1:0]={A15, A14} to the following for a 64-bit DCT: Bank[1:0] = {A15 ^ A18 ^ A23, A14^ A17 ^ A22}.

0 = Disabled (default)
1 = Enabled

Note: *^ is the logical exclusive-OR operator.*

DRAM Controller Queue Bypass Maximum (DcqBypassMax)—Bits 27–24. The DRAM controller arbiter normally allows transactions to pass other transactions in order to optimize DRAM bandwidth. This field specifies the maximum number of times that the oldest memory-access request in the DRAM controller queue may be bypassed before the arbiter decision is overridden and the oldest memory-access request is serviced instead.

0000b = No bypass; the oldest request is never bypassed.
0001b = The oldest request may be bypassed no more than 1 time.
... ...
1111b = The oldest request may be bypassed no more than 15 times.

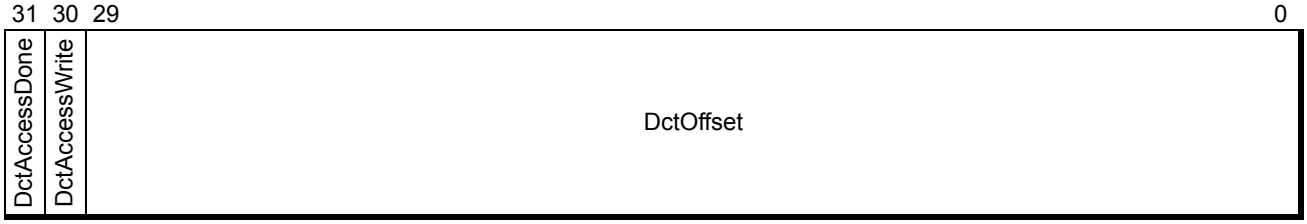
Four Bank Activate Window (FourActWindow)—Bits 31–28. This specifies the rolling tFAW window during which no more than 4 banks in a 8-bank device are activated, per JEDEC DDR2 specification. For example, if this is set to 10 clocks and an activate command is issued in clock N, then no more than three further activate commands may be issued in clocks N+1 through N+9. To program this field, BIOS must convert the tFAW parameter into MEMCLK cycles by dividing the highest tFAW (in ns) across all DIMMs connected to the channel by the lowest period (highest frequency) of MEMCLK (in ns) over all P-states and rounding up to the next integer.

0000b = No tFAW window restriction.
0001b = 8 MEMCLK cycles
0010b = 9 MEMCLK cycles
....
1101b = 20 MEMCLK cycles
111xb = Reserved

4.5.13 DRAM Controller Additional Data Offset Register

This register is used to select the DRAM controller register that is accessed when reading or writing Function 2 Offset 9Ch.

Dram Controller Additional Data Offset Register **Function 2: Offset 98h**



Bits	Mnemonic	Function	R/W	Reset
31	DctAccessDone	DRAM Controller Access Done	R	1
30	DctAccessWrite	DRAM Controller Read/Write Select	W	0
29–0	DctOffset	DRAM Controller Offset	R/W	0

Field Descriptions

Dram Controller Offset (DctOffset)—Bits 29–0. This field is used to select the DRAM controller register that is accessed when reading or writing Function 2 Offset 9Ch.

Dram Controller Read/Write Select (DctAccessWrite)—Bit 30. This bit indicates if the access to Function 2 Offset 9Ch is a read or write.

0b = Read access.
1b = Write access.

Dram Controller Access Done (DctAccessDone)—Bit 31. This bit indicates if the access to Function 2 Offset 9Ch is complete.

0b = Access in progress.
1b = No access is progress.

Table 14. DctOffset Field Encodings

DctOffset Channel A	DctOffset Channel B	Register
0000_0000h	0000_0020h	Output Driver Compensation Control
0000_0001h	0000_0021h	Write Data Timing Low
0000_0002h	0000_0022h	Write Data Timing High
0000_0003h	0000_0023h	Write ECC Timing
0000_0004h	0000_0024h	Address/Command Timing Control
0000_0005h	0000_0025h	Read DQS Timing Control Low
0000_0006h	0000_0026h	Read DQS Timing Control High
0000_0007h	0000_0027h	Read DQS ECC Timing Control
0000_0010h	0000_0030h	DQS Receiver Enable Timing Control Chip Selects M[B:A]0_CS_L[1:0]
0000_0011h	0000_0031h	Reserved
0000_0012h	0000_0032h	Reserved
0000_0013h	0000_0033h	DQS Receiver Enable Timing Control Chip Selects M[B:A]1_CS_L[1:0] Note:
0000_0014h	0000_0034h	Reserved
0000_0015h	0000_0035h	Reserved
0000_0016h	0000_0036h	DQS Receiver Enable Timing Control Chip Selects M[B:A]2_CS_L[1:0]

Note: *For S1g1 Package this register controls the DQS Receiver Enable Timing for Chip Selects M[B:A]0_CS_L[3:2]*

Table 14. DctOffset Field Encodings

DctOffset Channel A	DctOffset Channel B	Register
0000_0017h	0000_0037h	Reserved
0000_0018h	0000_0038h	Reserved
0000_0019h	0000_0039h	DQS Receiver Enable Timing Control Chip Selects M[B:A]3_CS_L[1:0]
0000_001Ah	0000_003Ah	Reserved
0000_001Bh	0000_003Bh	Reserved
Note: *For S1g1 Package this register controls the DQS Receiver Enable Timing for Chip Selects M[B:A]0_CS_L[3:2]*		

4.5.14 DRAM Controller Additional Data Port Register

This register is used to access the DRAM controller register that is selected by Function 2 Offset 98h. Access to these registers is accomplished as follows:

- Reads:
  - Write the register number to Function 2 Offset 98h with Function 2 Offset 98h (DctAccessWrite)=0.
  - Poll Function 2 Offset 98h (DctAccessDone) until it is high.
  - Read the register contents from Function 2 Offset 9Ch.
- Writes:
  - Write the register data to Function 2 Offset 9Ch.
  - Write the register number to Function 2 Offset 98h (DctOffset) with Function 2 Offset 98h (DctAccessWrite)=1.
  - Poll Function 2 Offset 98h (DctAccessDone) until it is high to ensure that the contents of the write have been delivered to the phy.

All software access to this register must be doubleword access. Byte and word access to this register are not supported. The MemClkFreqVal bit (Function 2, Offset 94h) must be set before attempting to read or write the DRAM Controller Additional Data registers.

DRAM Controller Additional Data Port Register **Function 2: Offset 9Ch**

Bits	Mnemonic	Function	R/W	Reset
31–0	DctOffsetData	DRAM Controller Offset Data	R/W	0

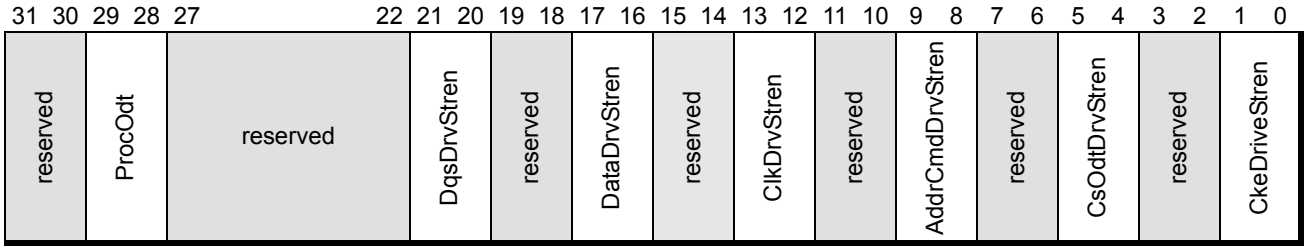
Field Descriptions

Dram Controller Offset Data (DctOffsetData)—Bits 31–0. This field is used to access the DRAM controller register that is selected by Function 2 Offset 98h.

4.5.15 Output Driver Compensation Control Register

This register is used to program the drive strength and termination resistance for the DRAM channel.

Output Driver Compensation Control Register **Function 2: Offset 9Ch**
Index 00h, 20h



Bits	Mnemonic	Function	R/W	Reset
31–30	reserved		R	0
29–28	ProcOdt	Processor On-die Termination	R/W	0
27–22	reserved		R	0
21–20	DqsDrvStren	DQS Drive Strength	R/W	01b
19–18	reserved		R	0
17–16	DataDrvStren	Data Drive Strength	R/W	01b
15–14	reserved		R	0
13–12	ClkDrvStren	Clock Drive Strength	R/W	01b
11–10	reserved		R	0
9–8	AddrCmdDrvStren	Address/Command Drive Strength	R/W	10b
7–6	reserved		R	0
5–4	CsOdtDrvStrength	CS/ODT Drive Strength	R/W	10b
3–2	reserved		R	0
1–0	CkeDreStrength	CKE Drive Strength	R/W	10b

Field Descriptions

CKE Drive Strength (CkeDrvStren)—Bits 1–0. This field is used to select the drive strength of the CKE pins.

00b = 1.0x.
01b = 1.25x
10b = 1.5x (Default)
11b = 2.0x

CS/ODT Drive Strength (CsOdtDrvStren)—Bits 5–4. This field is used to select the drive strength of the CS and ODT pins.

00b = 1.0x.
01b = 1.25x
10b = 1.5x (Default)
11b = 2.0x

Address/Command Drive Strength (AddrCmdDrvStren)—Bits 9–8. This field is used to select the drive strength of the address, RAS, CAS, WE, bank and parity pins.

00b = 1.0x.
01b = 1.25x
10b = 1.5x (Default)
11b = 2.0x

MEMCLK Drive Strength (ClkDrvStren)—Bits 13–12. This field is used to select the drive strength of the MEMCLK pins.

00b = 0.75x.
01b = 1.0x (Default)
10b = 1.25x
11b = 1.5x

Data Drive Strength (DataDrvStren)—Bits 17–16. This field is used to select the drive strength of the Data pins.

00b = 0.75x.
01b = 1.0x (Default)
10b = 1.25x
11b = 1.5x

DQS Drive Strength (DqsDrvStren)—Bits 21–20. This field is used to select the drive strength of the DQS pins.

00b = 0.75x.
01b = 1.0x (Default)
10b = 1.25x
11b = 1.5x

Note: *The DM[8:0] and DQS[17:9] functions share pins on the DIMM connector. The function selection is applied based on whether the DIMM is populated with by-4 (x4) DRAM devices, in which case the DQS[17:9] function is applied, or not (x8 or x16 DRAM devices), in which case the DM[8:0] function is applied. However, the DM function is associated with the data pin group and should therefore be controlled DataDrvStren. While the processor supports concurrent population of x4 and non-x4 DIMMs, the determination as to which field controls the drive strength of these pins is applied statically based on these rules:*
- If all DIMMs of a channel are populated with non-x4 devices, DataDrvStren is applied.
- If any DIMMs of a channel is populated with x4 devices, DqsDrvStren is applied.

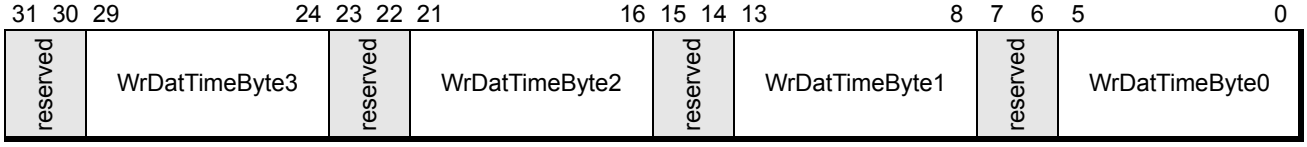
Processor On-die Termination (ProcOdt)—Bits 29–28. This field is used to select the resistance of the on-die termination resistors.

00b = 300 ohms +/- 20%
01b = 150 ohms +/- 20%.
10b = 75 ohms +/- 20%.
11b = Reserved.

4.5.16 Write Data Timing Control Registers

These registers control the timing of write data with respect to DQS.

Write Data Timing Low Timing Control Register — Function 2: Offset 9Ch Index 01h, 21h



Bits	Mnemonic	Function	R/W	Reset
31–30	reserved		R	0
29–24	WrDatTimeByte3	Write Data Byte 3 Timing Control	R/W	17h
23–22			R	0
21–16	WrDatTimeByte2	Write Data Byte 2 Timing Control	R/W	17h
15–14	reserved		R	0
13–8	WrDatTimeByte1	Write Data Byte 1 Timing Control	R/W	17h
7–6	reserved		R	0
5–0	WrDatTimeByte0	Write Data Byte 0 Timing Control	R/W	17h

Field Descriptions

Write Data Byte 0 Timing Control (WrDatTimeByte0)—Bits 5–0. These bits select how much delay is added to byte 0 of the data with respect to DQS.

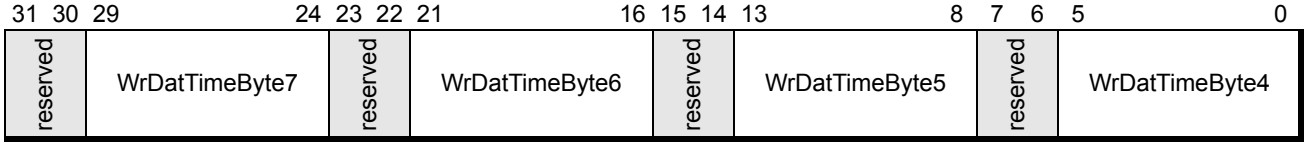
000000b = no delay
000001b = 1/96 MEMCLK delay
000010b = 2/96 MEMCLK delay
...
101111b = 47/96 MEMCLK delay
11xxxxb = reserved

Write Data Byte 1 Timing Control (WrDatTimeByte1)—Bits 13–8. These bits select how much delay is added to byte 1 of the data with respect to DQS. See WrDatTimeByte0 for bit definition.

Write Data Byte 2 Timing Control (WrDatTimeByte2)—Bits 21–16. These bits select how much delay is added to byte 2 of the data with respect to DQS. See WrDatTimeByte0 for bit definition.

Write Data Byte 3 Timing Control (WrDatTimeByte3)—Bits 29–24. These bits select how much delay is added to byte 3 of the data with respect to DQS. See WrDatTimeByte0 for bit definition.

Write Data Timing High Timing Control Register — Function 2: Offset 9Ch Index 02h, 22h



Bits	Mnemonic	Function	R/W	Reset
31–30	reserved		R	0
29–24	WrDatTimeByte7	Write Data Byte 7 Timing Control	R/W	17h
23–22			R	0
21–16	WrDatTimeByte6	Write Data Byte 6 Timing Control	R/W	17h
15–14	reserved		R	0
13–8	WrDatTimeByte5	Write Data Byte 5 Timing Control	R/W	17h
7–6	reserved		R	0
5–0	WrDatTimeByte4	Write Data Byte 4 Timing Control	R/W	17h

Field Descriptions

Write Data Byte 4 Timing Control (WrDatTimeByte4)—Bits 5–0. These bits select how much delay is added to byte 4 of the data with respect to DQS.

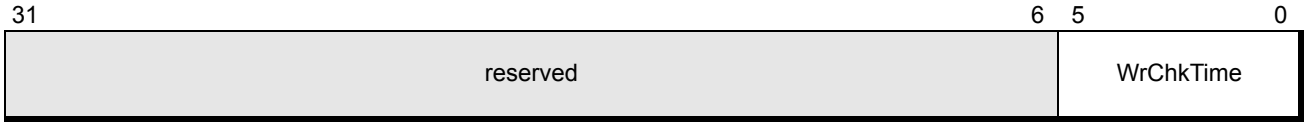
000000b = no delay
000001b = 1/96 MEMCLK delay
000010b = 2/96 MEMCLK delay
...
101111b = 47/96 MEMCLK delay
11xxxxb = reserved

Write Data Byte 5 Timing Control (WrDatTimeByte5)—Bits 13–8. These bits select how much delay is added to byte 5 of the data with respect to DQS. See WrDatTimeByte4 for bit definition.

Write Data Byte 6 Timing Control (WrDatTimeByte6)—Bits 21–16. These bits select how much delay is added to byte 6 of the data with respect to DQS. See WrDatTimeByte4 for bit definition.

Write Data Byte 7 Timing Control (WrDatTimeByte7)—Bits 29–24. These bits select how much delay is added to byte 7 of the data with respect to DQS. See WrDatTimeByte4 for bit definition

Write Data ECC Timing Control Register **Function 2: Offset 9Ch Index 03h, 23h**



Bits	Mnemonic	Function	R/W	Reset
31–6	reserved		R	0
5–0	WrChkTime	Write Data ECC Timing Control	R/W	17h

Field Descriptions

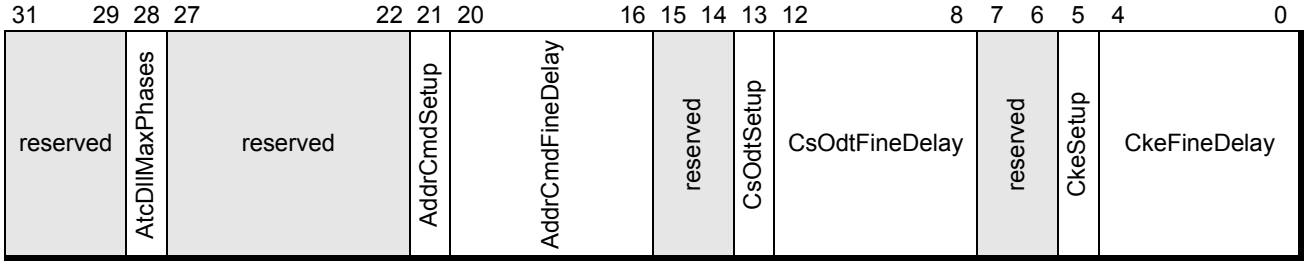
Write Data ECC Timing Control (WrChkTime)—Bits 5–0. These bits select how much delay is added to the ECC with respect to DQS.

000000b = no delay
000001b = 1/96 MEMCLK delay
000010b = 2/96 MEMCLK delay
...
101111b = 47/96 MEMCLK delay
11xxxxb = reserved

4.5.17 Address Timing Control Register

This register controls the time of the address, command, chip select, ODT and clock enable pins with respect to MEMCLK.

Address Timing Control Register **Function 2: Offset 9Ch Index 04h, 24h**



Bits	Mnemonic	Function	R/W	Reset
31–29	reserved		R	0
28	AtcDllMaxPhases	Address Timing Control DLL Phases, index 04h Reserved, index 24h	R/W	0
27–22	reserved		R	0
21	AddrCmdSetup	Address/Command Setup Time	R/W	0
20–16	AddrCmdFineDelay	Address/Command Fine Delay	R/W	0
15–14	reserved		R	0
13	CsOdtSetup	CS/ODT Setup Time	R/W	0
12–8	CsOdtFineDelay	CS/ODT Fine Delay	R/W	0
7–6	reserved		R	0
5	CkeSetup	CKE Setup Time	R/W	0
4–0	CkeFineDelay	CKE Fine Delay	R/W	0

Field Descriptions

CKE Fine Delay (CkeFineDelay)—Bits 4–0. These bits control how long the CKE pins are delayed from the default setup time.

00000b = no delay
00001b = 1/64 MEMCLK delay
00010b = 2/64 MEMCLK delay
...
11111b = 31/64 MEMCLK delay

CKE Setup Time (CkeSetup)—Bit 5. This bit selects the default setup time for the CKE pins versus MEMCLK.

0 = 1/2 MEMCLK
1 = 1 MEMCLK

CS/ODT Fine Delay (CsOdtFineDelay)—Bits 12–8. These bits control how long the CS and ODT pins are delayed from the default setup time.

00000b = no delay
00001b = 1/64 MEMCLK delay
00010b = 2/64 MEMCLK delay
...
11111b = 31/64 MEMCLK delay

CS/ODT Setup Time (CsOdtSetup)—Bit 13. This bit selects the default setup time for the CS and ODT pins versus MEMCLK.

0 = 1/2 MEMCLK
1 = 1 MEMCLK)

Address/Command Fine Delay (AddrCmdFineDelay)—Bits 20–16. These bits control how long the address and command pins are delayed from the default setup time.

00000b = no delay
00001b = 1/64 MEMCLK delay
00010b = 2/64 MEMCLK delay
...
11111b = 31/64 MEMCLK delay

Address/Command Setup Time (AddrCmdSetup)—Bit 21. This bit selects the default setup time for the address and command pins versus MEMCLK.

0 = 1/2 MEMCLK (1 1/2 MEMCLK for 2T timing)
1 = 1 MEMCLK(2 MEMCLKs for 2T timing)

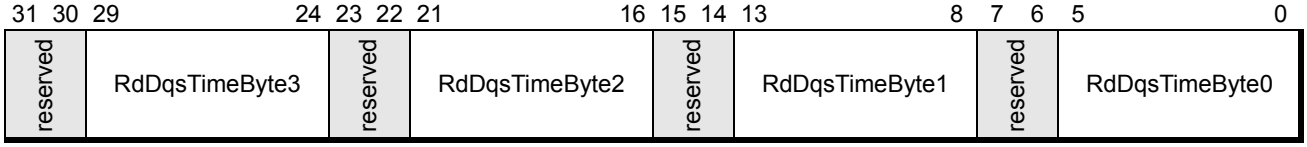
Address Timing Control DLL Phases(AtcDllMaxPhases)—Bit 28. This bit selects the number of phases for the 8 DLLs controlling the timing of the address-command (AddrCmd, CsOdt, Cke) interface group. This bit exists only in index 04h and controls the DLL phases for both channel A and B. BIOS should program this bit prior to DRAM training. It is recommended that BIOS set this bit to 1b. See "Register Differences in Revisions of AMD NPT Family 0Fh Processors" on page 27 for revision information about this field.

0 = 32 phases per bit-time
1 = 64 phases per bit-time

4.5.18 Read DQS Timing Control Registers

These registers control how much the read DQS is delayed with respect to the data.

Read DQS Timing Low Timing Control Register — Function 2: Offset 9Ch Index 05h, 25h



Bits	Mnemonic	Function	R/W	Reset
31–30	reserved		R	0
29–24	RdDqsTimeByte3	Read DQS Byte 3 Timing Control	R/W	17h
23–22				
21–16	RdDqsTimeByte2	Read DQS Byte 2 Timing Control	R/W	17h
15–14	reserved		R	0
13–8	RdDqsTimeByte1	Read DQS Byte 1 Timing Control	R/W	17h
7–6	reserved		R	0
5–0	RdDqsTimeByte0	Read DQS Byte 0 Timing Control	R/W	17h

Field Descriptions

Read DQS Byte 0 Timing Control (RdDqsTimeByte0)—Bits 5–0. These bits select how much delay is added to the DQS associated with byte 0 with respect to the data.

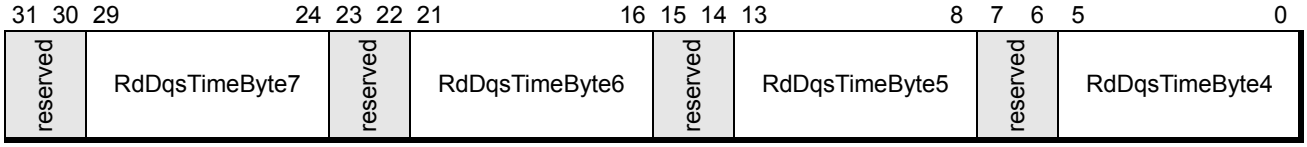
000000b = no delay
000001b = 1/96 MEMCLK delay
000010b = 2/96 MEMCLK delay
...
101111b = 47/96 MEMCLK delay
11xxxxb = reserved

Read DQS Byte 1 Timing Control (RdDqsTimeByte1)—Bits 13–8. These bits select how much delay is added to the DQS associated with byte 1 with respect to the data. See RdDqsTimeByte0 for bit definition.

Read DQS Byte 2 Timing Control (RdDqsTimeByte2)—Bits 21–16. These bits select how much delay is added to the DQS associated with byte 2 with respect to the data. See RdDqsTimeByte0 for bit definition.

Read DQS Byte 3 Timing Control (RdDqsTimeByte3)—Bits 29–24. These bits select how much delay is added to the DQS associated with byte 3 with respect to the data. See RdDqsTimeByte0 for bit definition.

Read DQS Timing High Timing Control Register — Function 2: Offset 9Ch Index 06h, 26h



Bits	Mnemonic	Function	R/W	Reset
31–30	reserved		R	0
29–24	RdDqsTimeByte7	Read DQS Byte 7 Timing Control	R/W	17h
23–22				0
21–16	RdDqsTimeByte6	Read DQS Byte 6 Timing Control	R/W	17h
15–14	reserved		R	0
13–8	RdDqsTimeByte5	Read DQS Byte 5 Timing Control	R/W	17h
7–6	reserved		R	0
5–0	RdDqsTimeByte4	Read DQS Byte 4 Timing Control	R/W	17h

Field Descriptions

Read DQS Byte 4 Timing Control (RdDqsTimeByte4)—Bits 5–0. These bits select how much delay is added to the DQS associated with byte 4 with respect to the data.

000000b = no delay

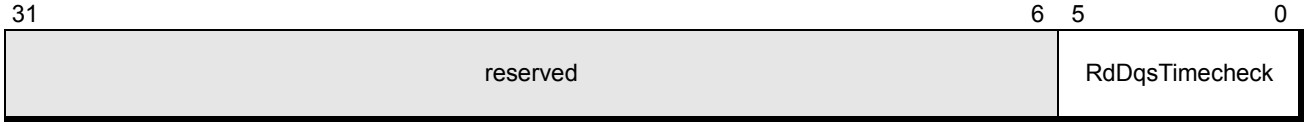
000001b = 1/96 MEMCLK delay
000010b = 2/96 MEMCLK delay
...
101111b = 47/96 MEMCLK delay
11xxxxb = reserved

Read DQS Byte 5 Timing Control (RdDqsTimeByte5)—Bits 13–8. These bits select how much delay is added to the DQS associated with byte 5 with respect to the data. See RdDqsTimeByte4 for bit definition.

Read DQS Byte 6 Timing Control (RdDqsTimeByte6)—Bits 21–16. These bits select how much delay is added to the DQS associated with byte 6 with respect to the data. See RdDqsTimeByte4 for bit definition.

Read DQS Byte 7 Timing Control (RdDqsTimeByte7)—Bits 29–24. These bits select how much delay is added to the DQS associated with byte 7 with respect to the data. See RdDqsTimeByte4 for bit definition.

Read DQS ECC Timing Control Register — Function 2: Offset 9Ch Index 07h, 27h



Bits	Mnemonic	Function	R/W	Reset
31–6	reserved		R	0
5–0	RdDqsTimeCheck	Read DQS ECC Byte Timing Control	R/W	17h

Field Descriptions

Read DQS ECC Byte Timing Control (RdDqsTimeCheck)—Bits 5–0. These bits select how much delay is added to the DQS associated with the ECC byte with respect to the data.

000000b = no delay
000001b = 1/96 MEMCLK delay
000010b = 2/96 MEMCLK delay
...
101111b = 47/96 MEMCLK delay
11xxxxb = reserved

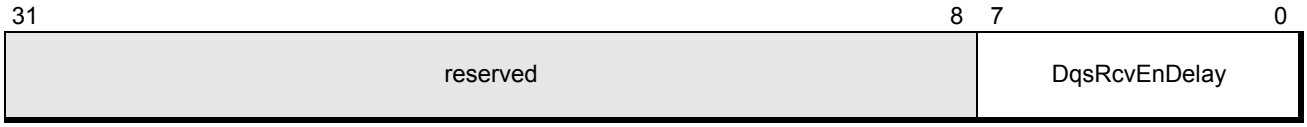
4.5.19 DQS Receiver Enable Timing Control Registers

These registers specify the delay in picoseconds from the assertion of MEMCLK at the CPU to the driving of the read preamble by the DIMM as perceived by the CPU. Writing to the DQS receiver

enable timing causes the read and write pointers in the DRAM controller FIFOs to be reset. The DQS receiver enable delays programmed must meet the following restrictions:

- The maximum delay between two DIMMs that comprise a logical 128 bit DIMM = 1 MEMCLK.
- The delays for unused DIMMs should be programmed to 00h.

DQS Receiver Enable Timing Control Register — Function 2: Offset 9Ch
Index 10h, 13h, 16h, 19h, 30h, 33h, 36h, 39h



Bits	Mnemonic	Function	R/W	Reset
31–8		reserved	R	0
7–0	DqsRcvEnDelay	DQS Receiver Enable Delay	R/W	0

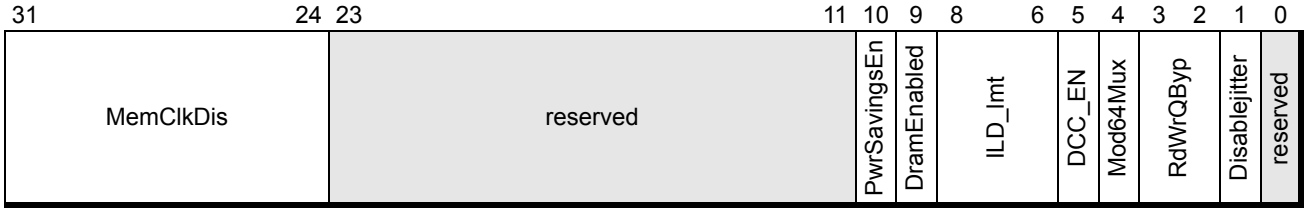
Field Descriptions

DQS Receiver Enable Delay (DqsRcvEnDelay)—Bits 7–0. These bits specify the delay in 50 picosecond increments from the rising edge of MEMCLK at the CPU to the driving of the read preamble by the DIMM as perceived at the CPU.

00h = 0 ps
01h = 50 ps
02h = 100 ps
...
AEh = 8.70 ns
AFh–FFh = Reserved

4.5.20 DRAM Controller Miscellaneous Register

DRAM Controller Miscellaneous Register — Function 2: Offset A0h



Bits	Mnemonic	Function	R/W	Reset
31–24	MemClkDis	Memory Clock Disable	R/W	FFh
23–11	reserved		R	0
10	PwrSavingsEn	Dram Power Savings Enable	R/W	0

Bits	Mnemonic	Function	R/W	Reset
9	DramEnabled	Dram enabled	R	0
8–6	ILD_lmt	Idle Cycle Limit	R/W	0
5	DCC_EN	Dynamic Idle Cycle Counter Enable	R/W	0
4	Mod64Mux	Mismatched DIMM Support Enable	R/W	0
3–2	RdWrQByp	Read/Write Queue Bypass Count	R/W	0
1	DisableJitter	Disable Jitter	R/W	0
0	reserved		R	X

Field Descriptions

Disable Jitter (DisableJitter)—Bit 1. When set the DDR compensation circuit will not change values unless the change is more than one step from the current value.

Read/Write Queue Bypass Count (RdWrQByp)—Bits 2–3. Specifies the number of times the oldest operation in the DCI read/write queue can be bypassed before the arbiter is overridden and the oldest operation is chosen.

00b = 2
01b = 4
10b = 8
11b = 16

Mismatched DIMM Support Enable (Mod64BitMux)—Bit 4. This bit enables support for mismatched DIMMs when using 128-bit DRAM interface. When this bit is set, the Width128 bit has no effect. This bit only applies to the AM2 and S1g1 packages. This mode cannot be used with on-line spare.

Dynamic Idle Cycle Counter Enable (DCC_EN)—Bit 5. When set to 1, indicates that each entry in the page table dynamically adjusts the idle cycle limit based on Page Conflict/Page Miss (PC/PM) traffic.

Idle Cycle Limit (ILD_lmt)—Bits 8–6. Specifies the number of MemCLKs before forcibly closing (precharging) an open page. If DCC_EN (Function 2, Offset A0h) has a value of 0, the static counters are loaded with the ILD_lmt and decremented each clock. If DCC_EN has a value of 1, the dynamic counters are loaded with the ILD_lmt and modified as follows:

Increment—When a Page Miss (PM) page hits on an invalid entry in the Page Table. The presumption is that in the past, that page table entry was occupied by the very same page that has a Page Miss. Had the old page been kept open longer, it would have been a Page Hit. Increment the Idle Cycle Limit count to increase the probability of getting a future Page Hit.

Decrement—When a Page Conflict (PC) arrives and hits on an idle entry (obviously an open page). This Page Conflict can be avoided if the open page is closed earlier. Decrement the Idle Cycle Limit count to increase the probability of avoiding a future Page Conflict.

000b = 0 cycles
001b = 4 cycles

010b = 8 cycles
011b = 16 cycles
100b = 32 cycles
101b = 64 cycles
110b = 128 cycles
111b = 256 cycles

DRAM Enabled (DramEnabled)—Bit 9. When set, this bit indicates that DRAM is enabled. This bit is set by hardware after DRAM initialization or on an exit from self refresh. The DRAM controller is initialized after the hardware-controlled initialization process (initiated by Function 2: Offset 90h[DramInit]) completes or when the BIOS-controlled initialization process completes (Function 2: Offset 7Ch [EnDramInit] is written from 1 to 0).

DRAM Power Savings Enable (PwrSavingsEn)—Bit 10. When set, the DRAM DLLs and the on die DRAM termination are disabled when the DRAM is placed in self refresh.

MEMCLK disable (MemClkDis)—Bits 31–24. These bits disable the MEMCLK outputs for DRAM channel B. BIOS should leave all bits corresponding to unused clocks set to reduce power consumption. These bits are mapped to packages as follows:

Bit	F(1207) Package	AM2 Package	S1g1 Package	ASB1 Package
0	N/A	MB1_CLK1	N/A	N/A
1	N/A	MB0_CLK1	MB0_CLK1	MB0_CLK1
2	MB3_CLK	N/A	N/A	N/A
3	MB2_CLK	N/A	N/A	N/A
4	MB1_CLK	MB1_CLK0	N/A	N/A
5	MB0_CLK	MB0_CLK0	MB0_CLK0	MB0_CLK0
6	N/A	MB1_CLK2	N/A	N/A
7	N/A	MB0_CLK2	MB0_CLK2	MB0_CLK2
Note: Channel B is not supported on all processor brands in ASB1 package. Refer to the Processor Functional Data Sheet.				

4.6 Function 3—Miscellaneous Control

This function is a collection of the remaining memory system configuration registers. As listed in Table 15, Function 3 includes the following registers:

- Machine check architecture for Northbridge errors
- DRAM scrubber
- Buffer counts for Northbridge flow control

- Power management
- GART
- Clocking/FIFO control
- Thermtrip status
- Northbridge capabilities

Table 15. Function 3 Configuration Registers

Offset	Register Name				Reset	Access	Description
00h	Device ID		Vendor ID (AMD)		1103_1022h	RO	page 138
04h	Status[1]		Command		0000_0000h	RO	
08h	Base Class Code	Subclass Code	Program Interface	Revision ID	0600_0000h	RO	page 139
0Ch	BIST	Header Type	Latency Timer	Cache Line Size	0080_0000h	RO	page 140
10h	Base Address 0				0000_0000h	RO	
14h	Base Address 1				0000_0000h	RO	
18h	Base Address 2				0000_0000h	RO	
1Ch	Base Address 3				0000_0000h	RO	
20h	Base Address 4				0000_0000h	RO	
24h	Base Address 5				0000_0000h	RO	
28h	Card Bus CIS Pointer				0000_0000h	RO	
2Ch	Sub System ID		Sub system Vendor ID		0000_0000h	RO	
30h	ROM Base Address				0000_0000h	RO	
34h	Capabilities				0000_0000h	RO	
38h	reserved				0000_0000h	RO	
3Ch	Max Latency	Min GNT	Int Pin	Int Line	0000_0000h	RO	
40h	MCA Northbridge Control				0000_0000h	RW	page 141
44h	MCA Northbridge Configuration				0000_0000h	RW	page 143
48h	MCA Northbridge Status Low				page 147	RW	page 147
4Ch	MCA Northbridge Status High				page 150	RW	page 150
50h	MCA Northbridge Address Low				page 153	RW	page 153
54h	MCA Northbridge Address High				page 154	RW	page 154

Notes:

1. *The unimplemented registers in the standard PCI configuration space are implemented as read-only and return 0 if read.*
2. *Reads and writes to unimplemented registers in the extended PCI configuration space will result in unpredictable behavior.*

Table 15. Function 3 Configuration Registers (Continued)

Offset	Register Name	Reset	Access	Description
58h	Scrub Control	0000_0000h	RW	page 159
5Ch	DRAM Scrub Address Low	page 161	RW	page 161
60h	DRAM Scrub Address High	page 162	RW	page 162
64h	HTC	page 162	RW	page 162
68h	Thermal Control	page 163	RW	page 163
70h	SRI-to-XBAR Buffer Counts	5102_0111h	RW	page 170
74h	XBAR-to-SRI Buffer Counts	5000_8011h	RW	page 172
78h	MCT-to-XBAR Buffer Counts	0800_3800h	RW	page 171
7Ch	Free List Buffer Counts	0000_221Bh	RW	page 172
80h	Power Management Control Low	0000_0000h	RW	page 175
84h	Power Management Control High	0000_0000h	RW	page 176
90h	GART Aperture Control	0000_0000h	RW	page 177
94h	GART Aperture Base	page 178	RW	page 178
98h	GART Table Base	page 179	RW	page 179
9Ch	GART Cache Control	0000_0000h	RW	page 179
A0h	Power Control Miscellaneous	0000_0000h	RW	page 180
B0h	On-Line Spare Control Register	0000_0000h	R/W	page 181
D4h	Clock Power/Timing Low	page 182	RW	page 182
D8h	Clock Power/Timing High	page 184	RW	page 184
DCh	HyperTransport™ Read Pointer Optimization	page 186	RW	page 186
E4h	Thermtrip Status	0000_0000h	RO	page 188
E8h	Northbridge Capabilities	0000_0000h	RO	page 190
FCh	CPUID Family Model	page 192	R	page 192

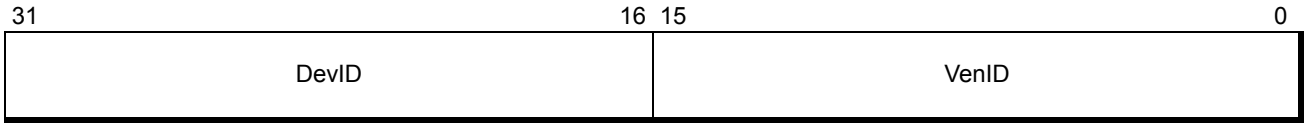
Notes:

1. *The unimplemented registers in the standard PCI configuration space are implemented as read-only and return 0 if read.*
2. *Reads and writes to unimplemented registers in the extended PCI configuration space will result in unpredictable behavior.*

4.6.1 Device/Vendor ID Register

This register specifies the device and vendor IDs for the Function 3 registers and is part of the standard PCI configuration header.

Device/Vendor ID Register — Function 3: Offset 00h



Bits	Mnemonic	Function	R/W	Reset
31–16	DevID	Device ID	R	1103h
15–0	VenID	Vendor ID	R	1022h

Field Descriptions

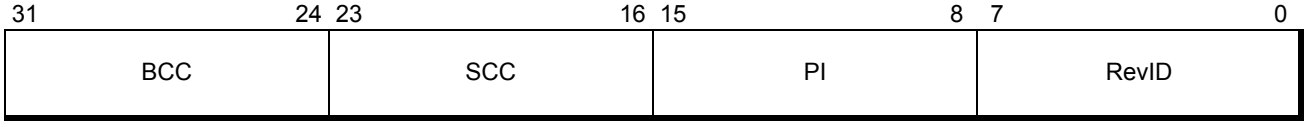
Vendor ID (VenID)—Bits 15–0. This read-only value is defined as 1022h for AMD.

Device ID (DevID)—Bits 31–16. This read-only value is defined as 1103h for the HyperTransport technology configuration function.

4.6.2 Class Code/Revision ID Register

This register specifies the class code and revision for the Function 3 registers and is part of the standard PCI configuration header.

Class Code/Revision ID Register — Function 3: Offset 08h



Bits	Mnemonic	Function	R/W	Reset
31–24	BCC	Base Class Code	R	06h
23–16	SCC	Subclass Code	R	00h
15–8	PI	Programming Interface	R	00h
7–0	RevID	Revision ID	R	00h

Field Descriptions

Revision ID (RevID)—Bits 7–0.

Programming Interface (PI)—Bits 15–8. This read-only value is defined as 00h.

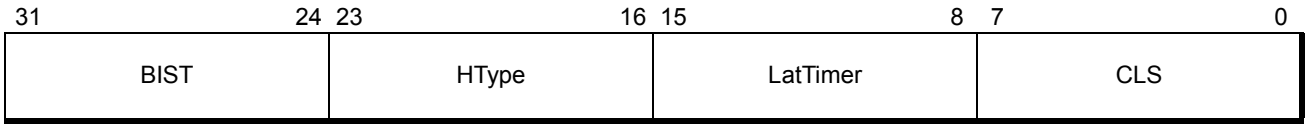
Sub Class Code (SCC)—Bits 23–16. This read-only value is defined as 00h.

Base Class Code (BCC)—Bits 31–24. This read-only value is defined as 06h for a host bridge device.

4.6.3 Header Type Register

This register specifies the header type for the Function 3 registers and is part of the standard PCI configuration header.

Header Type Register **Function 3: Offset 0Ch**



Bits	Mnemonic	Function	R/W	Reset
31–24	BIST	BIST	R	00h
23–16	HType	Header Type	R	80h
15–8	LatTimer	Latency Timer	R	00h
7–0	CLS	Cache Line Size	R	00h

Field Descriptions

CacheLineSize (CLS)—Bits 7–0. This read-only value is defined as 00h.

LatencyTimer (LatTimer)—Bits 15–8. This read-only value is defined as 00h.

HeaderType (Htype)—Bits 23–16. This read-only value is defined as 80h to indicate that multiple functions are present in the configuration header and that the header layout corresponds to a device header as opposed to a bridge header.

BIST—Bits 31–24. This read-only value is defined as 00h.

4.6.4 Machine Check Architecture (MCA) Registers

These registers are used to configure the Machine Check Architecture (MCA) functions of the on-chip Northbridge (NB) hardware and to provide a method for the Northbridge hardware to report errors in a way compatible with MCA. All of the Northbridge MCA registers, except for the MCA NB Configuration register, are accessible through the MCA-defined MSR method, as well as through PCI configuration space. The MCA NB Configuration register is accessible only through PCI configuration space.

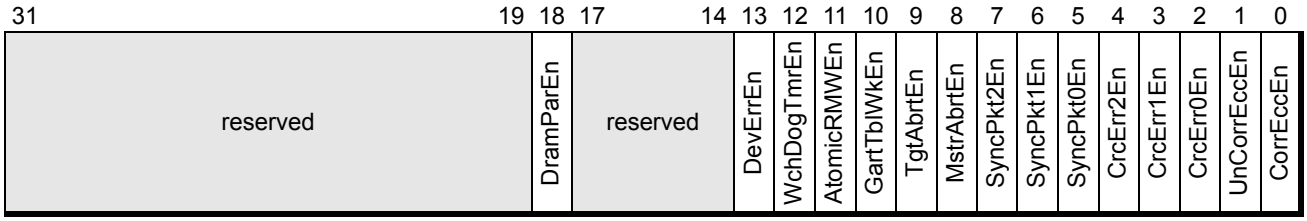
The MCA NB Control register enables MCA reporting of each error checked by the Northbridge. The global MCA error enables must also be set through the global MCA MSRs. The error enables in this register only affect error reporting through MCA. Actions which the Northbridge may take in addition to MCA reporting are enabled through the MCA NB Configuration register.

The MCA NB Status registers and MCA NB Address registers provide status and address information regarding an error which the Northbridge has logged. The values of error-logging registers are

maintained through a warm reset allowing software to identify an error source that resulted in system-wide initialization.

4.6.4.1 MCA NB Control Register

MCA NB Control Register **Function 3: Offset 40h**



Bits	Mnemonic	Function	R/W	Reset
31–19	reserved		R	0
18	DramParEn	DRAM Parity Error Reporting enable	R/W	0
17–14	reserved		R	0
13	DevErrEn	DEV Error Reporting Enable	R/W	0
12	WchDogTmrEn	Watchdog Timer Error Reporting Enable	R/W	0
11	AtomicRMWEn	Atomic Read-Modify-Write Error Reporting Enable	R/W	0
10	GartTblWkEn	GART Table Walk Error Reporting Enable	R/W	0
9	TgtAbrtEn	Target Abort Error Reporting Enable	R/W	0
8	MstrAbrtEn	Master Abort Error Reporting Enable	R/W	0
7	SyncPkt2En	HyperTransport™ Link 2 Sync Packet Error Reporting Enable	R/W	0
6	SyncPkt1En	HyperTransport Link 1 Sync Packet Error Reporting Enable	R/W	0
5	SyncPkt0En	HyperTransport Link 0 Sync Packet Error Reporting Enable	R/W	0
4	CrcErr2En	HyperTransport Link 2 CRC Error Reporting Enable	R/W	0
3	CrcErr1En	HyperTransport Link 1 CRC Error Reporting Enable	R/W	0
2	CrcErr0En	HyperTransport Link 0 CRC Error Reporting Enable	R/W	0
1	UnCorrEccEn	Uncorrectable ECC Error Reporting Enable	R/W	0
0	CorrEccEn	Correctable ECC Error Reporting Enable	R/W	0

Field Descriptions

Correctable ECC Error Reporting Enable (CorrEccEn)—Bit 0. Enables MCA reporting of correctable ECC errors which are detected in the Northbridge. Since correctable errors do not result in an MCA exception, the effect of this bit is limited to the setting of the error enable bit in the MCA NB Status High register.

Uncorrectable ECC Error Reporting Enable (UnCorrEccEn)—Bit 1. Enables MCA reporting of uncorrectable ECC errors which are detected in the Northbridge. Note that in some cases data

may be forwarded to the CPU core prior to checking ECC in which case the check takes place in one of the other error reporting banks.

HyperTransport Link 0 CRC Error Reporting Enable (CrcErr0En)—Bit 2. Enables MCA reporting of CRC errors detected on HyperTransport link 0. The Northbridge floods its outgoing HyperTransport links with sync packets after detecting a CRC error on an incoming link independent of the state of this bit.

HyperTransport Link 1 CRC Error Reporting Enable (CrcErr1En)—Bit 3. Enables MCA reporting of CRC errors detected on HyperTransport link 1. The Northbridge floods its outgoing HyperTransport links with sync packets after detecting a CRC error on an incoming link independent of the state of this bit.

HyperTransport Link 2 CRC Error Reporting Enable (CrcErr2En)—Bit 4. Enables MCA reporting of CRC errors detected on HyperTransport link 2. The Northbridge floods its outgoing HyperTransport links with sync packets after detecting a CRC error on an incoming link independent of the state of this bit.

HyperTransport Link 0 Sync Packet Error Reporting Enable (SyncPkt0En)—Bit 5. Enables MCA reporting of HyperTransport sync error packets detected on HyperTransport link 0. The Northbridge floods its outgoing HyperTransport links with sync packets after detecting a sync packet on an incoming link independent of the state of this bit.

HyperTransport Link 1 Sync Packet Error Reporting Enable (SyncPkt1En)—Bit 6. Enables MCA reporting of HyperTransport sync error packets detected on HyperTransport link 1. The Northbridge floods its outgoing HyperTransport links with sync packets after detecting a sync packet on an incoming link independent of the state of this bit.

HyperTransport Link 2 Sync Packet Error Reporting Enable (SyncPkt2En)—Bit 7. Enables MCA reporting of HyperTransport sync error packets detected on HyperTransport link 2. The Northbridge floods its outgoing HyperTransport links with sync packets after detecting a sync packet on an incoming link independent of the state of this bit.

Master Abort Error Reporting Enable (MstrAbrtEn)—Bit 8. Enables MCA reporting of master aborts (HyperTransport packets that return an error status with the Non Existent Address bit set). The Northbridge returns an error response back to the requestor with any associated data all 1s independent of the state of this bit.

Target Abort Error Reporting Enable (TgtAbrtEn)—Bit 9. Enables MCA reporting of target aborts (HyperTransport technology packets that return an error status with the Non Existent Address bit clear). The Northbridge returns an error response back to the requestor with any associated data all 1s independent of the state of this bit.

GART Table Walk Error Reporting Enable (GartTblWkEn)—Bit 10. Enables MCA reporting of GART cache table walks which encounter a GART PTE entry which is invalid.

Atomic Read-Modify-Write Error Reporting Enable (AtomicRMWEn)—Bit 11. Enables MCA reporting of atomic read-modify-write (RMW) HyperTransport technology commands

received from an noncoherent HyperTransport link which do not target DRAM. Atomic RMW commands are not supported by AMD NPT Family 0Fh Processors. Atomic RMW commands to memory-mapped I/O are not supported in HyperTransport technology. An atomic RMW command that targets MMIO results in a HyperTransport error response being generated back to the requesting I/O device. The generation of the HyperTransport error response is not affected by this bit.

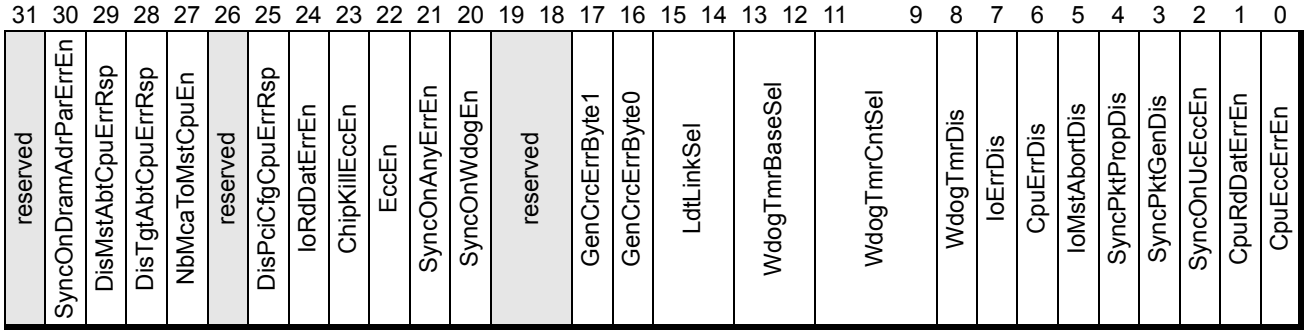
Watchdog Timer Error Reporting Enable (WchDogTmrEn)—Bit 12. Enables MCA reporting of watchdog timer errors. The watchdog timer checks for Northbridge system accesses for which a response is expected and where no response is received. See the MCA NB Configuration register for information regarding configuration of the watchdog timer duration. Note that this bit does not affect operation of the watchdog timer in terms of its ability to complete an access that would otherwise cause a system hang. This bit only affects whether such errors are reported through MCA.

DEV Error Reporting Enable (DevErrEn)—Bit 13. Enables MCA reporting of SVM DEV errors. This bit only affects whether such errors are reported through MCA.

DRAM Parity Error Reporting Enable (DramParEn)—Bit 18. Enables MCA reporting of DRAM address parity errors. See "Register Differences in Revisions of AMD NPT Family 0Fh Processors" on page 27 for revision information about this field.

4.6.4.2 MCA NB Configuration Register

MCA NB Configuration Register **Function 3: Offset 44h**



Bits	Mnemonic	Function	R/W	Reset
31	reserved		R	0
30	SyncOnDramAdrParErrEn	Sync Flood on DRAM Address Parity Error Enable	R/W	0
29	DisMstAbtCpuErrRsp	Master Abort CPU Error Response Disable	R/W	0
28	DisTgtAbtCpuErrRsp	Target Abort CPU Error Response Disable	R/W	0
27	NbMcaToMstCpuEn	Northbridge MCA to CPU 0 Enable	R/W	0
26	reserved		R	0
25	DisPciCfgCpuErrRsp	PCI configuration CPU Error Response Disable	R/W	0

Bits	Mnemonic	Function	R/W	Reset
24	IoRdDatErrEn	I/O Read Data Error Log Enable	R/W	0
23	ChipKillEccEn	Chip-Kill ECC Mode Enable	R/W	0
22	EccEn	ECC Enable	R/W	0
21	SyncOnAnyErrEn	Sync Flood On Any Error Enable	R/W	0
20	SyncOnWdogEn	Sync Flood on Watchdog Timer Error Enable	R/W	0
19–18	reserved		R	0
17	GenCrcErrByte1	Generate CRC Error on Byte Lane 1	R/W	0
16	GenCrcErrByte0	Generate CRC Error on Byte Lane 0	R/W	0
15–14	LdtLinkSel	HyperTransport™ Link Select for CRC Error Generation	R/W	0
13–12	WdogTmrBaseSel	Watchdog Timer Time Base Select	R/W	0
11–9	WdogTmrCntSel	Watchdog Timer Count Select	R/W	0
8	WdogTmrDis	Watchdog Timer Disable	R/W	0
7	IoErrDis	I/O Error Response Disable	R/W	0
6	CpuErrDis	CPU Error Response Disable	R/W	0
5	IoMstAbortDis	I/O Master Abort Error Response Disable	R/W	0
4	SyncPktPropDis	Sync Packet Propagation Disable	R/W	0
3	SyncPktGenDis	Sync Packet Generation Disable	R/W	0
2	SyncOnUcEccEn	Sync Flood on Uncorrectable ECC Error Enable	R/W	0
1	CpuRdDatErrEn	CPU Read Data Error Log Enable	R/W	0
0	CpuEccErrEn	CPU ECC Error Log Enable	R/W	0

Field Descriptions

CPU ECC Error Log Enable (CpuEccErrEn)—Bit 0. Enables reporting of ECC errors for data destined for the CPU on this node. This bit should be clear if ECC error logging is enabled for the remaining error reporting blocks in the CPU. Logging the same error in more than one block may cause a single error event to be treated as a multiple error event and cause the CPU to enter shutdown.

CPU Read Data Error Log Enable (CpuRdDatErrEn)—Bit 1. Enables reporting of read data errors (master aborts and target aborts) for data destined for the CPU on this node. This bit should be clear if read data error logging is enabled for the remaining error reporting blocks in the CPU. Logging the same error in more than one block may cause a single error event to be treated as a multiple error event and cause the CPU to enter shutdown.

Sync Flood on Uncorrectable ECC Error Enable (SyncOnUcEccEn)—Bit 2. Enables flooding of all HyperTransport links with sync packets on detection of an uncorrectable ECC error.

Sync Packet Generation Disable (SyncPktGenDis)—Bit 3. Disables flooding of all outgoing HyperTransport links with sync packets when a CRC error is detected on an incoming link. By default, sync packet generation for CRC errors is controlled through the HyperTransport technology LDTn Link Control registers (see page 70).

Sync Packet Propagation Disable (SyncPktPropDis)—Bit 4. Disables flooding of all outgoing HyperTransport links with sync packets when a sync packet is detected on an incoming link. Sync packets are propagated by default.

I/O Master Abort Error Response Disable (IoMstAbortDis)—Bit 5. Disables setting the NXA bit in HyperTransport response packets to I/O devices on detection of a master abort HyperTransport technology error condition.

CPU Error Response Disable (CpuErrDis)—Bit 6. Disables generation of a read data error response to the CPU core on detection of a target or master abort HyperTransport technology error condition.

I/O Error Response Disable (IoErrDis)—Bit 7. Disables setting the Error bit in HyperTransport response packets to I/O devices on detection of a target or master abort HyperTransport technology error condition.

Watchdog Timer Disable (WdogTmrDis)—Bit 8. Disables the watchdog timer. The watchdog timer is enabled by default and checks for Northbridge system accesses for which a response is expected and where no response is received. If such a condition is detected the outstanding access is completed by generating an error response back to the requestor. An MCA error may also be generated if enabled in the MCA NB Control register.

Watchdog Timer Count Select (WdogTmrCntSel)—Bit 11–9. Selects the count used by the watchdog timer. The counter selected by WdogTmrCntSel determines the maximum count value in the time base selected by WdogTmrBaseSel.

000b = 4095 (default)
001b = 2047
010b = 1023
011b = 511
100b = 255
101b = 127
110b = 63
111b = 31

Watchdog Timer Time Base Select (WdogTmrBaseSel)—Bit 13–12. Selects the time base used by the watchdog timer. The counter selected by WdogTmrCntSel determines the maximum count value in the time base selected by WdogTmrBaseSel.

00b = 1 ms (default)
01b = 1 μs
10b = 5 ns
11b = reserved

HyperTransport Link Select for CRC Error Generation (LdtLinkSel)—Bit 15–14. Selects the HyperTransport link to be used for CRC error injection through GenCrcErrByte1/GenCrcErrByte0.

00b = HyperTransport link 0

01b = HyperTransport link 1
10b = HyperTransport link 2
11b = undefined

Generate CRC Error on Byte Lane 0 (GenCrcErrByte0)—Bit 16. Causes a CRC error to be injected on Byte Lane 0 of the HyperTransport link specified by LdtLinkSel. The data carried by the link is unaffected. This bit is cleared after the error has been generated.

Generate CRC Error on Byte Lane 1 (GenCrcErrByte1)—Bit 17. Causes a CRC error to be injected on Byte Lane 1 of the HyperTransport link specified by LdtLinkSel. The data carried by the link is unaffected. This bit is cleared after the error has been generated.

Sync Flood on Watchdog Timer Error Enable (SyncOnWdogEn)—Bit 20. Enables flooding of all HyperTransport links with sync packets on detection of a watchdog timer error.

Sync Flood On Any Error Enable (SyncOnAnyErrEn)—Bit 21. Enables flooding of all HyperTransport links with sync packets on detection of any NB MCA error that is uncorrectable. Note: Individual errors cannot be disabled from generating sync packets by clearing bits in the MCA NB Control Register. Some individual errors may be disabled from generating sync packets by disabling their error sources in the MCA NB Configuration Register.

ECC Enable (EccEn)—Bit 22. Enables ECC check/correct mode. This bit must be set in order for any ECC checking/correcting to be enabled for any processor block. If set, ECC is checked and correctable errors are corrected irrespective of whether machine check ECC reporting is enabled. See "ECC and Chip Kill Error Checking and Correction" on page 157 for more details.

The hardware only allows values to be programmed into this field which are consistent with the ECC capabilities of the device as specified in the Northbridge Capabilities register (page 190). Attempts to write values inconsistent with the capabilities results in this field not being updated.

Chip-Kill ECC Mode Enable (ChipKillEccEn)—Bit 23. Enables chip-kill ECC mode. Setting this bit causes ECC checking to be based on a 128/16 data/ECC rather than on a 64/8 data/ECC. Chip-kill ECC can only be enabled in 128-bit DRAM data width mode. It allows correction of an error affecting a single symbol rather than the single bit correction available in 64/8 ECC checking. When this bit is set, EccEn must be set. See "ECC and Chip Kill Error Checking and Correction" on page 157 for more details.

The hardware only allows values to be programmed into this field which are consistent with the ECC and Chip Kill ECC capabilities of the device as specified in the Northbridge Capabilities register (page 190). Attempts to write values inconsistent with the capabilities results in this field not being updated.

I/O Read Data Error Log Enable (IoRdDatErrEn)—Bit 24. Enables reporting of read data errors (master aborts and target aborts) for data destined for I/O devices. If this bit is clear (default

state), master and target aborts for transactions from I/O devices are not logged by MCA, although error responses may still be generated to the requesting I/O device.

PCI Configuration CPU Error Response Disable (DisPciCfgCpuErrRsp)—Bit 25. Disables master abort and target abort reporting through the CPU error-reporting banks for PCI configuration accesses. It is recommended that this bit be set in order to avoid MCA exceptions being generated from master aborts for PCI configuration accesses (which can be common during device enumeration).

Northbridge MCA to CPU 0 Enable (NbMcaToMstCpuEn)—Bit 27. Enables reporting of all MCA errors to CPU 0 including errors from CPU 1. In addition, when this bit is set, reading MC4_CTL, MC4_ADDRESS and MC4_STATUS through MSR accesses from CPU 1 returns zeros and writes that are not all zeros cause a GP fault. This bit is only valid for dual core processors.

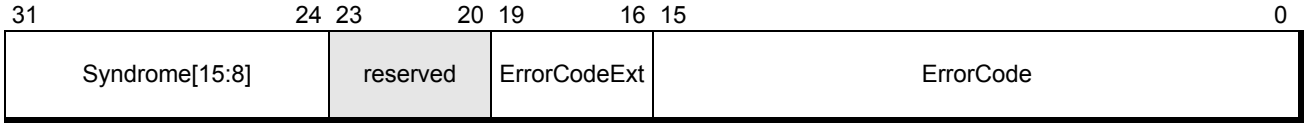
Target Abort CPU Error Response Disable ((DisTgtAbtCpuErrRsp)—Bit 28. Disables target abort reporting through the CPU error-reporting banks. See "Register Differences in Revisions of AMD NPT Family 0Fh Processors" on page 27 for revision information about this field.

Master Abort CPU Error Response Disable (DisMstAbtCpuErrRsp)—Bit 29. Disables master abort reporting through the CPU error-reporting banks. See "Register Differences in Revisions of AMD NPT Family 0Fh Processors" on page 27 for revision information about this field.

Sync Flood on DRAM Address Parity Error Enable (SyncOnDramAdrParErrEn)—Bit 30. Enables flooding of all HyperTransport links with sync packets on detection of a DRAM Address Parity Error. See "Register Differences in Revisions of AMD NPT Family 0Fh Processors" on page 27 for revision information about this field.

4.6.4.3 MCA NB Status Low Register

MCA NB Status Low Register **Function 3: Offset 48h**



Bits	Mnemonic	Function	R/W	Reset
31–24	Syndrome[15:8]	Syndrome Bits 15–8 for Chip Kill ECC Mode	R/W	X
23–20	reserved		R	0
19–16	ErrorCodeExt	Extended Error Code	R/W	X
15–0	ErrorCode	Error Code	R/W	X

"X" in the Reset column indicates that the field initializes to an undefined state after reset.

Field Descriptions

Error Code (ErrorCode)—Bits 15–0. Logs an error code when an error is detected. See Table 23 on page 150 for the error codes. Table 16 on page 148 describes the possible error code formats.

Extended Error Code (ErrorCodeExt)—Bits 19–16. Logs an extended error code when an error is detected. See Table 23 on page 150 for the extended error codes.

Syndrome Bits 15–8 for Chip Kill ECC Mode (Syndrome[15–8])—Bits 31–24. Logs the upper eight syndrome bits when an ECC error is detected. Only valid for ECC errors in Chip-Kill ECC mode.

Table 16. Error Code Field Formats

Error Value	Error Type	Description
0000 0000 0001 TTLL	TLB errors	Errors in the GART TLB cache. TT = Transaction Type (Table 17 on page 148) LL = Cache Level (Table 18 on page 148)
0000 0001 RRRR TTLL	Memory errors	Errors in the cache hierarchy (not applicable for Northbridge errors) RRRR = Memory Transaction Type (Table 19 on page 149) TT = Transaction Type (Table 17 on page 148) LL = Cache Level (Table 18 on page 148)
0000 1PPT RRRR IILL	Bus errors	General bus errors including errors in the HyperTransport™ link or DRAM. PP = Participation Processor (Table 20 on page 149) T = Time-out (Table 21 on page 149) RRRR = Memory Transaction Type (Table 19 on page 149) II = Memory or I/O (Table 22 on page 149) LL = Cache Level (Table 18 on page 148)

Table 17. Transaction Type Bits (TT)

00b	Instruction
01b	Data
10b	Generic
11b	reserved

Table 18. Cache Level Bits (LL)

00b	Level 0 (L0)
01b	Level 1 (L1)
10b	Level 2 (L2)
11b	Generic (LG)

Table 19. Memory Transaction Type Bits (RRRR)

0000b	Generic error (GEN)
0001b	Generic read (RD)
0010b	Generic write (WR)
0011b	Data read (DRD)
0100b	Data write (DWR)
0101b	Instruction fetch (IRD)
0110b	Prefetch
0111b	Evict
1000b	Snoop (probe)

Table 20. Participation Processor Bits (PP)

00b	Local node originated the request (SRC)
01b	Local node responded to the request (RES)
10b	Local node observed error as 3rd party (OBS)
11b	Generic

Table 21. Time-Out Bit (T)

0	Request did not time out
1	Request timed out (TIMOUT)

Table 22. Memory or I/O Bits (II)

00b	DRAM Memory access (MEM)
01b	reserved
10b	MMIO or I/O Space access
11b	Generic (GEN)

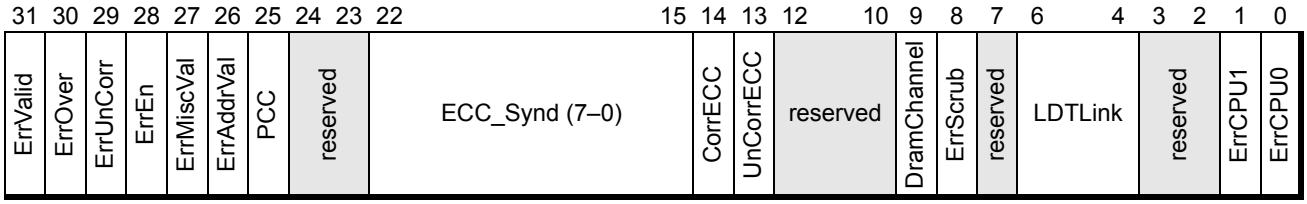
Table 23 on page 150 lists the errors reported by the Northbridge, along with the error codes for each error. The error code fields are defined in the table above. See the NB Control register for more information on errors detected by the Northbridge.

Table 23. Northbridge Error Codes

Error Type	Extended Error Code	Error Code						
		Type	PP	T	RRRR	II	LL	TT
ECC error	0000	BUS	SRC/RES	0	RD/WR	MEM	LG	-
CRC error	0001	BUS	OBS	0	GEN	GEN	LG	-
Sync error	0010	BUS	OBS	0	GEN	GEN	LG	-
Mst Abort	0011	BUS	SRC/OBS	0	RD/WR	MEM/IO	LG	-
Tgt Abort	0100	BUS	SRC/OBS	0	RD/WR	MEM/IO	LG	-
GART error	0101	TLB	-	-	-	-	LG	GEN
RMW error	0110	BUS	OBS	0	GEN	IO	LG	-
Wdog error	0111	BUS	GEN	1	GEN	GEN	LG	-
ChipKill ECC error	1000	BUS	SRC/RES	0	RD/WR	MEM	LG	-
DEV Error	1001	BUS	SRC/OBS	0	RD/WR	MEM/IO	LG	-
DRAM Parity Error	1101	BUS	OBS	0	GEN	MEM	LG	-

4.6.4.4 MCA NB Status High Register

MCA NB Status High Register **Function 3: Offset 4Ch**



Bits	Mnemonic	Function	R/W	Reset
31	ErrValid	Error Valid	R/W	X
30	ErrOver	Error Overflow	R/W	X
29	ErrUnCorr	Error Uncorrected	R/W	X
28	ErrEn	Error Enable	R/W	X
27	ErrMiscVal	Miscellaneous Error Register Valid	R	X
26	ErrAddrVal	Error Address Valid	R/W	X
25	PCC	Processor Context Corrupt	R/W	X
24–23	reserved		R	0
22–15	ECC_Synd (7–0)	Syndrome Bits (7–0) for ECC Errors	R/W	X
14	CorrECC	Correctable ECC Error	R/W	X
13	UnCorrECC	Uncorrectable ECC Error	R/W	X
12–10	reserved		R	0

Bits	Mnemonic	Function	R/W	Reset
9	DramChannel	DRAM Error Channel	R/W	X
8	ErrScrub	Error Found by DRAM Scrubber	R/W	X
7	reserved		R	0
6–4	LDTLink	HyperTransport™ Link Number	R/W	X
3–2	reserved		R	0
1	ErrCPU1	Error Associated with CPU 1	R/W	X
0	ErrCPU0	Error Associated with CPU 0	R/W	X

"X" in the Reset column indicates that the field initializes to an undefined state after reset.

Field Descriptions

Error Associated with CPU 0 (ErrCPU0)—Bit 0. If set to 1, this bit indicates that the error was associated with CPU0 core.

Error Associated with CPU 1 (ErrCPU1)—Bit 1. If set to 1, this bit indicates that the error was associated with CPU1 core.

HyperTransport Link Number (LDTLink[2:0])—Bits 6–4. For errors associated with a HyperTransport link (e.g., CRC errors), this field indicates which link was associated with the error.

LDTLink[0] = Error associated with HyperTransport link 0
LDTLink[1] = Error associated with HyperTransport link 1
LDTLink[2] = Error associated with HyperTransport link 2

Error Found by DRAM Scrubber (ErrScrub)—Bit 8. If set to 1, this bit indicates that the error was found by the DRAM scrubber.

DRAM Error Channel (Dram Channel)—Bit 9. This bit indicates the channel that the DRAM parity error occurred on.

0 = Error occurred on DRAM Channel A.
1 = Error occurred on DRAM Channel B.

Uncorrectable ECC Error (UnCorrECC)—Bit 13. If set to 1, this bit indicates that the error was an uncorrectable ECC error.

Correctable ECC Error (CorrECC)—Bit 14. If set to 1, this bit indicates that the error was a correctable ECC error.

Syndrome Bits 7–0 for ECC Errors (ECC_Synd[7:0])—Bits 22–15. Logs the lower eight syndrome bits when an ECC error is detected.

Processor Context Corrupt (PCC)—Bit 25. If set to 1, this bit indicates that the state of the processor may be corrupted by the error condition. Reliable restarting might not be possible.

0 = Processor not corrupted
1 = Processor may be corrupted

Error Address Valid (ErrAddrVal)—Bit 26. If set to 1, this bit indicates that the address saved in the address register is the address where the error occurred (i.e. it validates the address in the address register).

0 = Address register not valid
1 = Address register valid

Miscellaneous Error Register Valid (ErrMiscVal)—Bit 27. If set to 1, this bit indicates whether the Miscellaneous Error register contains valid information for this error. This bit is set when CntEn (MSR413) is set and the current error is an ECC error.

Error Enable (ErrEn)—Bit 28. If set to 1, this bit indicates that MCA error reporting is enabled in the MCA Control register.

0 = MCA error reporting not enabled
1 = MCA error reporting enabled

Error Uncorrected (ErrUnCorr)—Bit 29. If set to 1, this bit indicates that the error was not corrected by hardware.

0 = Error corrected
1 = Error not corrected

Error Overflow (ErrOver)—Bit 30. Set to 1 if the Northbridge detects an error with the valid bit of this register already set. Enabled errors are written over disabled errors, uncorrectable errors are written over correctable errors. Uncorrectable errors are not overwritten.

0 = No error overflow
1 = Error overflow

Error Valid (ErrValid)—Bit 31. If set to 1, this bit indicates that a valid error has been detected. This bit should be cleared to 0 by software after the register is read.

0 = No valid error detected
1 = Valid error detected

Table 24 lists the errors reported by Northbridge along with the status bits logged for each error. The status bits are defined in the table above. See the Northbridge Control register for more information on errors detected by Northbridge.

Table 24. Northbridge Error Status Bit Settings

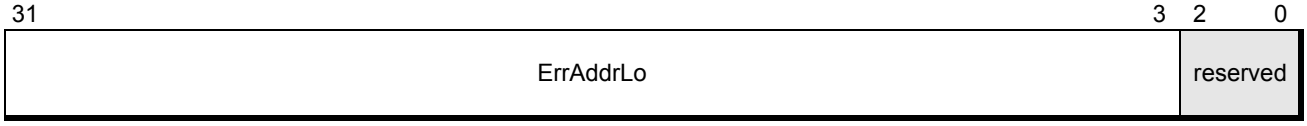
Error Type	ErrUnCorr	ErrAddr Val	PCC	ECC_Synd Valid	Corr ECC	UnCorr ECC	Err Scrub	LDTLink Valid	Err CPU
ECC error	If multi-bit	1	If multi-bit and src CPU	1	If single-bit	If multi-bit	1/0	0	1/0
CRC error	1	0	1	0	0	0	0	1	0
Sync error	1	0	1	0	0	0	0	1	0

Table 24. Northbridge Error Status Bit Settings (Continued)

Error Type	ErrUnCorr	ErrAddr Val	PCC	ECC_Synd Valid	Corr ECC	UnCorr ECC	Err Scrub	LDTLink Valid	Err CPU
Mst Abort	1	1	If src CPU	0	0	0	0	1	1/0
Tgt Abort	1	1	If src CPU	0	0	0	0	1	1/0
GART error	1	1	If src CPU	0	0	0	0	0	1/0
RMW error	1	1	0	0	0	0	0	1	0
Wdog error	1	0	1	0	0	0	0	0	X
Chip-Kill ECC error	If multi-symbol	1	If multi-symbol and src CPU	1	If single-symbol	If multi-symbol	1/0	0	1/0
DEV Error	1	1	0	0	0	0	0	1	X
DRAM Parity Error	1	0	1	0	0	0	0	0	X

4.6.4.5 MCA NB Address Low Register

MCA NB Address Low Register **Function 3: Offset 50h**



Bits	Mnemonic	Function	R/W	Reset
31–3	ErrAddrLo	Error Address Bits 31–3	R/W	X
2–0	reserved		R	0

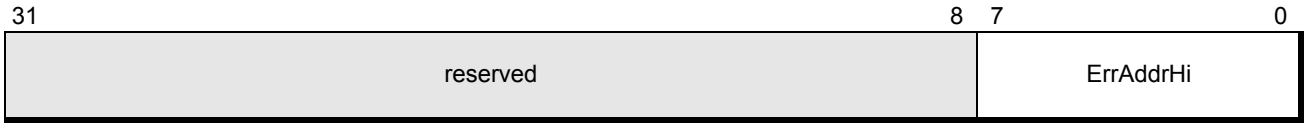
"X" in the Reset column indicates that the field initializes to an undefined state after reset.

Field Descriptions

Error Address Bits 31–3 (ErrAddrLo)—Bits 31–3. This field specifies bits 31–3 of the address associated with a machine check error.

4.6.4.6 MCA NB Address High Register

MCA NB Address High Register **Function 3: Offset 54h**



Bits	Mnemonic	Function	R/W	Reset
31–8	reserved		R	0
7–0	ErrAddrHi	Error Address Bits 39–32	R/W	X

"X" in the Reset column indicates that the field initializes to an undefined state after reset.

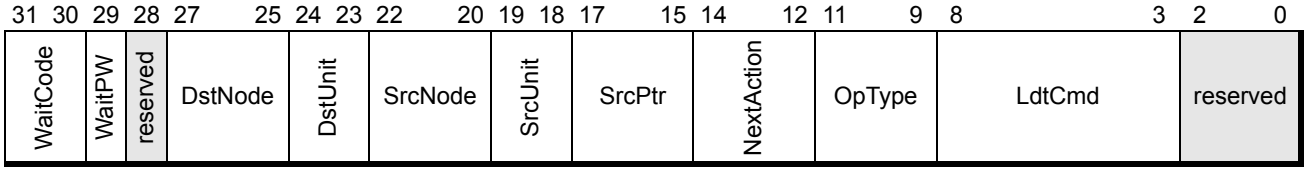
Field Descriptions

Error Address Bits 39–32 (ErrAddrHi)—Bits 7–0. This field specifies bits 39–32 of the address associated with a machine check error.

4.6.4.7 Watchdog Timer Errors

For watchdog timer errors the ErrAddrHi and ErrAddrLo fields in the MCA NB Address High/Low registers log the hardware state information of the request for which the response timed out.

MCA NB Address Low Register for Watchdog Timer Error **Function 3: Offset 50h**



Bit	Mnemonic	Function	R/W	Reset
31-30	WaitCode	Wait Code (Bits 1-0 of WaitCode)		
29	WaitPW	Wait For Posted Write		
28	reserved			
27–25	DstNode	Destination Node ID		
24–23	DstUnit	Destination Unit ID		
22–20	SrcNode	Source Node ID		
19–18	SrcUnit	Source Unit ID		
17–15	SrcPtr	Source Pointer		
14–12	NextAction	Next Action		
11–9	OpType	Operation Type		
8–3	LdtCmd	HyperTransport™ Command		
2–0	reserved			

Field Descriptions

HyperTransport Command (LdtCmd)—Bit 8–3. The initial command (in HyperTransport technology format) for the stalled operation.

Operation Type (OpType)—Bit 11–9. Indicates what type of operation is stalled.

000b = Normal operation (i.e., none of the other types listed)
001b = Bus lock
010b = Local APIC access
011b = Interrupt request
100b = System management request
101b = Interrupt broadcast
110b = Stop grant
111b = SMI ack

Next Action (NextAction)—Bit 14–12. Indicates what the stalled operation would have done next had it not become stalled.

000b = Complete
001b = reserved
010b = Send request
011b = Send second request (if any)
100b = Send final response back to requestor
101b = Send initial response (if any) back to requestor
110b = Send final response to the memory controller
111b = Send initial response (if any) to mem controller

Source Pointer (SrcPtr)—Bit 17–15. Source pointer of the stalled operation.

000b = Host bridge on local node
001b = CPU on local node
010b = Mem controller on local node
101b = HyperTransport link 0
110b = HyperTransport link 1
111b = HyperTransport link 2

Source Unit ID (SrcUnit)—Bit 19–18. Source Unit ID of the stalled operation.

Source Node ID (SrcNode)—Bit 22–20. Source Node ID of the stalled operation.

Destination Unit ID (DstUnit)—Bit 24–23. Destination Unit ID of the stalled operation.

Destination Node ID (DstNode)—Bit 27–25. Destination Node ID of the stalled operation.

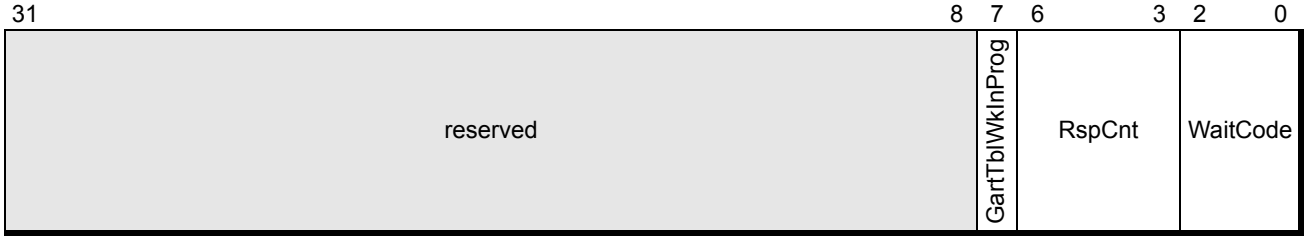
Wait For Posted Write (WaitPW)—Bit 29. When set, indicates that a response for the stalled operation is waiting for one or more prior posted writes to complete.

Wait Code (WaitCode)—Bit 31-30. WaitCode is a 5 bit field; WaitCode[1:0] is in Function 3, Offset 50h; WaitCode[4:2] is in Function 3, Offset 54h. WaitCode encodings are implementation

specific. They indicate the reason the hardware is waiting. All zeros mean that there is not a waiting condition.

00000b = No waiting condition
00100b = XBAR waiting for probe response from CPU0
00101b = XBAR waiting for probe response from CPU1
01000b = XBAR waiting because of GART/DEV miss
01001b = XBAR waiting for GART table walk state machine to free up
01010b = XBAR or CPU waiting for bus to unlock (locked instruction)
1x000b = SRI waiting for available XBAR upstream request buffer
1x001b = SRI waiting for available XBAR upstream posted request buffer
1x010b = SRI waiting for available XBAR downstream request buffer
1x011b = SRI waiting for available XBAR downstream posted request buffer
1x100b = SRI waiting for available XBAR upstream response buffer
1x101b = SRI waiting for available XBAR downstream response buffer
1x110b = SRI waiting for available XBAR display refresh buffer

MCA NB Address High Register for Watchdog Timer Error Function 3: Offset 54h



Bits	Mnemonic	Function	R/W	Reset
31–8	reserved			
7	GartTblWkInProg	GART Table Walk In Progress (Bit 39 of ErrAddr)		
6–3	RspCnt	System Response Count (Bits 38–35 of ErrAddr)		
2-0	WaitCode	Wait Code (Bits 4-2 of WaitCode)		

Field Descriptions

Wait Code (WaitCode)—Bit 2-0. See WaitCode definition in MCA NB Address Low Register for Watchdog Timer Error (Function 3, Offset 50h).

System Response Count (RspCnt)—Bits 6–3 (Bits 38–35 of ErrAddr). Number of outstanding system responses for which the stalled operation is waiting.

GART Table Walk In Progress (GartTblWkInProg)—Bit 7 (Bit 39 of ErrAddr). Indicates that a GART table walk was in progress at the time the error was logged.

4.6.5 ECC and Chip Kill Error Checking and Correction

The memory controller implements two ECC modes: normal ECC and Chip Kill ECC. These error checking and correction modes can only be used if all DIMMs are ECC capable.

4.6.5.1 Normal ECC

Normal ECC mode is a 64/8 SEC/DED (Single Error Correction/Double Error Detection) Hamming code. It can detect one bit error and correct it, it can detect two bit errors but it can not correct them, and it may detect more than two bit errors depending on the position of corrupted bits. Normal ECC mode is enabled when EccEn, Function 3, Offset 44h is set and ChipKillEccEn, Function 3, Offset 44h is clear.

Error address (Function 3, Offsets 50h and 54h) is valid and uniquely identifies the DIMM with a correctable or uncorrectable error for 64-bit and 128-bit data width configurations. In a 128-bit data width configuration, ECC is independently applied to the lower 64 and upper 64 data bits. In this configuration ECC may detect and correct two bit errors if one error happens on data bits 63-0 and one error happens on data bits 127-64. Status registers (Function 3, Offsets 48h and 4Ch) and error address registers contains information about the second detected error.

Bit position of a correctable error can be determined by matching Ecc_Synd, Function 3, Offset 4Ch with a value from Table 25. Bits 63-0 correspond to data bits, and bits 73-64 correspond to ECC check bits.

Table 25. ECC Syndromes

	n = 0	n = 1	n = 2	n = 3	n = 4	n = 5	n = 6	n = 7
Bit (0+n)	ce	cb	d3	d5	d6	d9	da	dc
Bit (8+n)	23	25	26	29	2a	2c	31	34
Bit (16+n)	0e	0b	13	15	16	19	1a	1c
Bit (24+n)	e3	e5	e6	e9	ea	ec	f1	f4
Bit (32+n)	4f	4a	52	54	57	58	5b	5d
Bit (40+n)	a2	a4	a7	a8	ab	ad	b0	b5
Bit (48+n)	8f	8a	92	94	97	98	9b	9d
Bit (56+n)	62	64	67	68	6b	6d	70	75
Bit (64+n)	01	02	04	08	10	20	40	80

4.6.5.2 Chip Kill

Chip Kill ECC mode is a 128/16 SSC/DSD (Single Symbol Correction/Double Symbol Detection) BCH code. A symbol is a group of 4 bits which are 4 bit aligned (bits 0–3 make symbol 0, bits 4–7 make symbol 1, etc.). A single symbol error is any bit error combination within one symbol (1 to 4 bits can be corrupted). Chip Kill ECC mode can detect one symbol error and correct it, it can detect two symbol errors but it can not correct them, and it may detect more than two symbol errors

depending on the position of corrupted symbols. Chip Kill ECC mode is enabled when EccEn and ChipKillEccEn, Function 3, Offset 44h are set.

Chip Kill mode can only be used in a 128-bit data width configuration.

A DIMM with a correctable error is uniquely identified with error address (Function 3, Offsets 50h and 54h) and Chip Kill syndrome (Syndrome, Function 3, Offset 48h and Ecc_Synd, Function 3, Offset 4Ch). Error address identifies two DIMMs and Chip Kill syndrome identifies one of them. Chip Kill syndromes for symbols 00h-0fh map to data bits 0-63; Chip Kill syndromes for symbols 20-21h map to ECC check bits for data bits 0-63; Chip Kill syndromes for symbols 10h-1fh map to data bits 64-127; Chip Kill syndromes for symbols 22-23h map to ECC check bits for data bits 64-127 (see Table 26). Corrupted bit positions within a symbol are determined from the Chip Kill syndrome column number in Table 26. Bits set in the column number identify corrupted bits within a symbol. For example, if 6913h is a Chip Kill syndrome, symbol 05h has an error, and bits 0 and 1 within that symbol are corrupted, since the syndrome is in column 3h. Symbol 05h maps to bits 23-20, so the corrupted bits are 20 and 21.

A DIMM with an uncorrectable error can not be uniquely identified. The error address is valid and maps to two DIMMs.

Chip Kill mode can be used with any device width. Chip Kill mode can correct all errors in an x4 device, since the device width is one symbol. Chip Kill mode can correct a subset of errors in an x8 or x16 device.

Table 26. Chip Kill ECC Syndromes

Symbol	1h	2h	3h	4h	5h	6h	7h	8h	9h	ah	bh	ch	dh	eh	fh
00h	e821	7c32	9413	bb44	5365	c776	2f57	dd88	35a9	a1ba	499b	66cc	8eed	1afe	f2df
01h	5d31	a612	fb23	9584	c8b5	3396	6ea7	eac8	b7f9	4cda	11eb	7f4c	227d	d95e	846f
02h	0001	0002	0003	0004	0005	0006	0007	0008	0009	000a	000b	000c	000d	000e	000f
03h	2021	3032	1013	4044	6065	7076	5057	8088	a0a9	b0ba	909b	c0cc	e0ed	f0fe	d0df
04h	5041	a082	f0c3	9054	c015	30d6	6097	e0a8	b0e9	402a	106b	70fc	20bd	d07e	803f
05h	be21	d732	6913	2144	9f65	f676	4857	3288	8ca9	e5ba	5b9b	13cc	aded	c4fe	7adf
06h	4951	8ea2	c7f3	5394	1ac5	dd36	9467	a1e8	e8b9	2f4a	661b	f27c	bb2d	7cde	358f
07h	74e1	9872	ec93	d6b4	a255	4ec6	3a27	6bd8	1f39	f3aa	874b	bd6c	c98d	251e	51ff
08h	15c1	2a42	3f83	cef4	db35	e4b6	f177	4758	5299	6d1a	78db	89ac	9c6d	a3ee	b62f
09h	3d01	1602	2b03	8504	b805	9306	ae07	ca08	f709	dc0a	e10b	4f0c	720d	590e	640f
0ah	9801	ec02	7403	6b04	f305	8706	1f07	bd08	2509	510a	c90b	d60c	4e0d	3a0e	a20f
0bh	d131	6212	b323	3884	e9b5	5a96	8ba7	1cc8	cdf9	7eda	afeb	244c	f57d	465e	976f
0ch	e1d1	7262	93b3	b834	59e5	ca56	2b87	dc18	3dc9	ae7a	4fab	642c	85fd	164e	f79f
0dh	6051	b0a2	d0f3	1094	70c5	a036	c067	20e8	40b9	904a	f01b	307c	502d	80de	e08f
0eh	a4c1	f842	5c83	e6f4	4235	1eb6	ba77	7b58	df99	831a	27db	9dac	396d	65ee	c12f

Table 26. Chip Kill ECC Syndromes

Symbol	1h	2h	3h	4h	5h	6h	7h	8h	9h	ah	bh	ch	dh	eh	fh
0fh	11c1	2242	3383	c8f4	d935	eab6	fb77	4c58	5d99	6e1a	7fdb	84ac	956d	a6ee	b72f
10h	45d1	8a62	cfb3	5e34	1be5	d456	9187	a718	e2c9	2d7a	68ab	f92c	bcfd	734e	369f
11h	63e1	b172	d293	14b4	7755	a5c6	c627	28d8	4b39	99aa	fa4b	3c6c	5f8d	8d1e	eeff
12h	b741	d982	6ec3	2254	9515	fbd6	4c97	33a8	84e9	ea2a	5d6b	11fc	a6bd	c87e	7f3f
13h	dd41	6682	bbc3	3554	e815	53d6	8e97	1aa8	c7e9	7c2a	a16b	2ffc	f2bd	497e	943f
14h	2bd1	3d62	16b3	4f34	64e5	7256	5987	8518	aec9	b87a	93ab	ca2c	e1fd	f74e	dc9f
15h	83c1	c142	4283	a4f4	2735	65b6	e677	f858	7b99	391a	badb	5cac	df6d	9dee	1e2f
16h	8fd1	c562	4ab3	a934	26e5	6c56	e387	fe18	71c9	3b7a	b4ab	572c	d8fd	924e	1d9f
17h	4791	89e2	ce73	5264	15f5	db86	9c17	a3b8	e429	2a5a	6dcb	f1dc	b64d	783e	3faf
18h	5781	a9c2	fe43	92a4	c525	3b66	6ce7	e3f8	b479	4a3a	1dbb	715c	26dd	d89e	8f1f
19h	bf41	d582	6ac3	2954	9615	fcd6	4397	3ea8	81e9	eb2a	546b	17fc	a8bd	c27e	7d3f
1ah	9391	e1e2	7273	6464	f7f5	8586	1617	b8b8	2b29	595a	cacb	dcdc	4f4d	3d3e	aeaf
1bh	cce1	4472	8893	fdb4	3155	b9c6	7527	56d8	9a39	12aa	de4b	ab6c	678d	ef1e	23ff
1ch	a761	f9b2	5ed3	e214	4575	1ba6	bcc7	7328	d449	8a9a	2dfb	913c	365d	688e	cfef
1dh	ff61	55b2	aad3	7914	8675	2ca6	d3c7	9e28	6149	cb9a	34fb	e73c	185d	b28e	4def
1eh	5451	a8a2	fcf3	9694	c2c5	3e36	6a67	ebe8	bfb9	434a	171b	7d7c	292d	d5de	818f
1fh	6fc1	b542	da83	19f4	7635	acb6	c377	2e58	4199	9b1a	f4db	37ac	586d	82ee	ed2f
20h	be01	d702	6903	2104	9f05	f606	4807	3208	8c09	e50a	5b0b	130c	ad0d	c40e	7a0f
21h	4101	8202	c303	5804	1905	da06	9b07	ac08	ed09	2e0a	6f0b	f40c	b50d	760e	370f
22h	c441	4882	8cc3	f654	3215	bed6	7a97	5ba8	9fe9	132a	d76b	adfc	69bd	e57e	213f
23h	7621	9b32	ed13	da44	ac65	4176	3757	6f88	19a9	f4ba	829b	b5cc	c3ed	2efe	58df

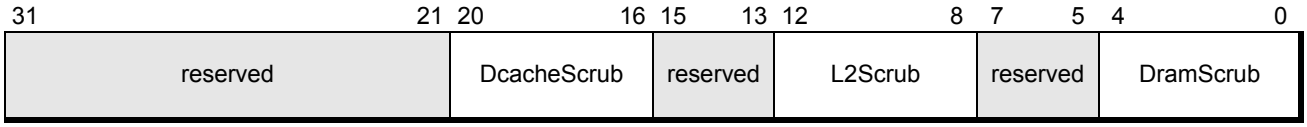
4.6.6 Scrub Control Register

This register specifies the scrub rate for sequential ECC scrubbing of DRAM, the L2 cache and the DCACHE. The scrub rate specifies the duration between successive scrub events. A value of 0 disables scrubbing.

Each scrub event corresponds to:

- 64 bytes for the DRAM scrubber.
- 64 bits for the data cache scrubber.
- One single L2 cache line tag address for the L2 scrubber.

Scrub Control Register — Function 3: Offset 58h



Bits	Mnemonic	Function	R/W	Reset
31–21	reserved		R	0
20–16	DcacheScrub	Data Cache Scrub Rate	R/W	0
15–13	reserved		R	0
12–8	L2Scrub	L2 Cache Scrub Rate	R/W	0
7–5	reserved		R	0
4–0	DramScrub	DRAM Scrub Rate	R/W	0

Field Descriptions

DRAM Scrub Rate (DramScrub)—Bits 4–0. Specifies the scrub rate for the DRAM. See Table 27. For example, if 256MByte memory is scrubbed every 12 hours, a 64-byte memory block should be scrubbed every 10ms, and scrubbing rate of 10.49ms should be selected.

L2 Cache Scrub Rate (L2Scrub)—Bits 12–8. Specifies the scrub rate for the L2 cache. See Table 27.

Data Cache Scrub Rate (DcacheScrub)—Bits 20–16. Specifies the scrub rate for the data cache. See Table 27.

Table 27. Scrub Rate Control Values

Scrub Rate Code	Scrub Rate	Scrub Rate Code	Scrub Rate
00000b	Do not scrub	01100b	81.9 μs
00001b	40.0 ns	01101b	163.8 μs
00010b	80.0 ns	01110b	327.7 μs
00011b	160.0 ns	01111b	655.4 μs
00100b	320.0 ns	10000b	1.31 ms
00101b	640.0 ns	10001b	2.62 ms
00110b	1.28 μs	10010b	5.24 ms
00111b	2.56 μs	10011b	10.49 ms
01000b	5.12 μs	10100b	20.97 ms
01001b	10.2 μs	10101b	42.00 ms
01010b	20.5 μs	10110b	84.00 ms
01011b	41.0 μs	All others	reserved

4.6.7 DRAM Scrub Address Registers

These registers specify the next address to be scrubbed by the DRAM scrubber. They should be initialized by the BIOS before the scrubber is enabled (the DRAM scrubber is enabled by writing a valid scrub rate to DramScrub field, Function 3, Offset 58h). They are then updated by the scrubber hardware as it scrubs successive 64-byte blocks of DRAM. Once the scrubber reaches the DRAM limit address for the node (see "DRAM Address Map" on page 84), it wraps to the DRAM base address.

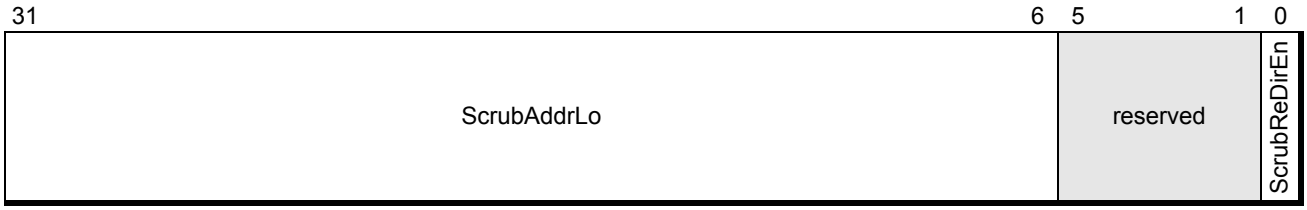
The initial scrubbing address specified by these registers must be between the base and limit address for the node, defined through the DRAM address maps (see "DRAM Address Map" on page 84). The recommended initial scrubbing address is the DRAM base address.

In addition to sequential DRAM scrubbing, the DRAM scrubber has a redirect mode for scrubbing DRAM locations accessed during normal operation. This is enabled by setting ScrubReDirEn (Function 3, Offset 5Ch). When a DRAM read is generated by any agent other than the DRAM scrubber, correctable ECC errors are corrected as the data is passed to the requestor, but the data in DRAM is not corrected if redirect scrubbing mode is disabled. In scrubber redirect mode, correctable errors detected during normal DRAM read accesses redirect the scrubber to the location of the error. After the scrubber corrects the location in DRAM, it resumes scrubbing from where it left off. DRAM scrub address registers are not modified by the redirect scrubbing mode. Sequential scrubbing and scrubber redirection can be enabled independently or together.

ECC errors detected by the scrubber are logged in the MCA registers (see "Machine Check Architecture Registers" on page 218).

4.6.7.1 DRAM Scrub Address Low Register

DRAM Scrub Address Low Register **Function 3: Offset 5Ch**



Bits	Mnemonic	Function	R/W	Reset
31–6	ScrubAddrLo	DRAM Scrub Address Bits 31–6	R/W	X
5–1	reserved		R	0
0	ScrubReDirEn	DRAM Scrubber Redirect Enable	R/W	0

"X" in the Reset column indicates that the field initializes to an undefined state after reset.

Field Descriptions

DRAM Scrubber Redirect Enable (ScrubReDirEn)—Bit 0. If this bit is set, the scrubber is redirected to correct errors found during normal operation.

DRAM Scrub Address Bits 31–6 (ScrubAddrLo)—Bits 31–6. This field specifies the low address bits 31–6 of the next 64-byte block to be scrubbed.

4.6.7.2 DRAM Scrub Address High Register

DRAM Scrub Address High Register **Function 3: Offset 60h**

Bits	Mnemonic	Function	R/W	Reset
31–8	reserved		R	0
7–0	ScrubAddrHi	DRAM Scrub Address Bits 39–32	R/W	X

"X" in the Reset column indicates that the field initializes to an undefined state after reset.

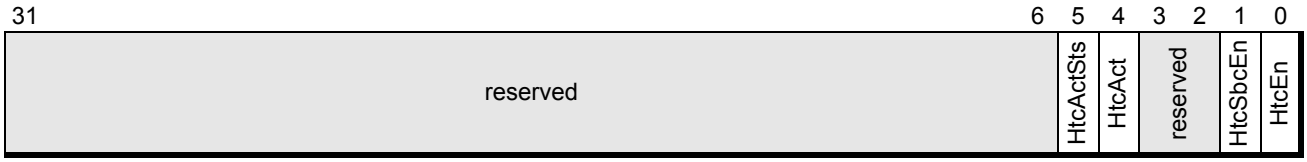
Field Descriptions

DRAM Scrub Address Bits 39–32 (ScrubAddrHi)—Bits 7–0. This field specifies the high address bits 39–32 of the next 64-byte block to be scrubbed.

4.6.8 HTC Register

This register is not accessible if HtcCap bit (Function 3, Offset E8h) is not set. See "Register Differences in Revisions of AMD NPT Family 0Fh Processors" on page 27 for revision information about this register.

HTC Register **Function 3: Offset 64h**



Bits	Mnemonic	Function	R/W	Reset
31–6	reserved		R	0
5	HtcActSts	HTC Active Status	R/W	0
4	HtcAct	HTC Active	R	0
3–2	reserved		R	0

Bits	Mnemonic	Function	R/W	Reset
1	HtcSbcEn	HTC Special Bus Cycle Enable	R/W	0
0	HtcEn	HTC Enable	R/W	0

Field Descriptions

HTC Enable (HtcEn)—Bit 0. HTC is enabled when this bit is set.

HTC Special Bus Cycle Enable (HtcSbcEn)—Bit 1. Enables special bus cycles generation when HTC active state is entered or exited. When this bit is set, HTC clock active must be a minimum of 1024 ns.

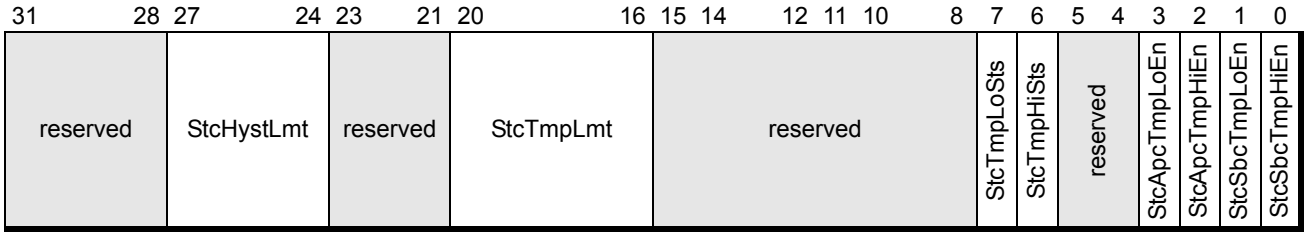
HTC Active (HtcAct)—Bit 4. When this bit is set, HTC is active.

HTC Active Status (HtcActSts)—Bit 5. This bit is set by hardware when HTC is active. It is cleared by writing a 1.

4.6.9 Thermal Control Register

This register is not accessible if HtcCap bit in Function 3, Offset E8h register is not set. See "Register Differences in Revisions of AMD NPT Family 0Fh Processors" on page 27 for revision information about this register.

Thermal Control Register — Function 3: Offset 68h



Bits	Mnemonic	Function	R/W	Reset
31-28	reserved		R	0
27-24	StcHystLmt	STC Hysteresis Limit	R/W	0
23–21	reserved		R	0
20-16	StcTmpLmt	STC Temperature Limit	R/W	0
15–8	reserved		R	0
7	StcTmpLoSts	STC Temperature Low Status	R/W	0
6	StcTmpHiSts	STC Temperature High Status	R/W	0
5–4	reserved		R	0
3	StcApcTmpLoEn	STC APIC Temperature Low Interrupt Enable	R/W	0
2	StcApcTmpHiEn	STC APIC Temperature High Interrupt Enable	R/W	0

Bits	Mnemonic	Function	R/W	Reset
1	StcSbcTmpLoEn	STC Special Bus Cycle Temperature Low Enable	R/W	0
0	StcSbcTmpHiEn	STC Special Bus Cycle Temperature High Enable	R/W	0

Field Descriptions

STC Special Bus Cycle Temperature High Enable (StcSbcTmpHiEn)—Bit 0. Enables STC special bus cycle generation when temperature is higher than the STC high temperature.

STC Special Bus Cycle Temperature Low Enable (StcSbcTmpLoEn)—Bit 1. Enables STC special bus cycle generation when temperature is lower than the STC low temperature.

STC APIC Temperature High Enable (StcApcTmpHiEn)—Bit 2. Enables STC APIC interrupt when temperature is higher than the STC high temperature.

STC APIC Temperature Low Enable (StcApcTmpLoEn)—Bit 3. Enables STC APIC interrupt when temperature is lower than the STC low temperature.

STC Temperature High Status (StcTmpHiSts)—Bit 6. This bit is set by hardware when die temperature is higher than the STC high temperature. It is cleared by writing a 1.

STC Temperature Low Status (StcTmpLoSts)—Bit 7. This bit is set by hardware when die temperature is lower than the STC low temperature. It is cleared by writing a 1.

STC Temperature Limit (StcTmpLmt)—Bits 20-16. Interrupt is generated or special bus cycle is sent when die temperature is greater than HtcTmpLmt.

00000b = 52C
00001b = 54C
...
11111b = 114C

STC Hysteresis Limit (StcHystLmt)—Bits 27-24. Interrupt is generated or special bus cycle is sent when die temperature is less than HtcTmpLmt minus HtcHystLmt.

0000b = 2C
0001b = 4C
...
1111b = 32C

4.6.10 XBAR Flow Control Buffers

The Northbridge interfaces with the CPU core, DRAM controller, and, through three HyperTransport links, to external chips.

The major Northbridge blocks are: System Request Interface (SRI), Memory Controller (MCT), and Cross Bar (XBAR). SRI interfaces with the CPU core and connects coherent HyperTransport links and noncoherent HyperTransport links. MCT maintains cache coherency and interfaces with the

DRAM. XBAR is a five port switch which routes the command packets between SRI, MCT, and the three HyperTransport links. Not all HyperTransport links have to be active.

The number of buffers available for each link at the XBAR input is shown in Table 28.

Table 28. XBAR Input Buffers

Link	Number of Command Buffers	Number of Data Buffers
HyperTransport™ Link 0-2	16 x 3	8 x 3
SRI	10	5
MCT	12	8
Total	70	37

XBAR command and data buffers are independent. There are 70 command buffers available, but a maximum of 64 can be used at any given time. Number of used data buffers is not restricted. The default allocation of command buffers when all HyperTransport links are present is shown in Table 29.

Table 29. Default XBAR Command Buffer Allocation

Link	Number of Command Buffers
HyperTransport™ Link 0-2	15 x 3
SRI	8
MCT	11
Total	64

The HyperTransport™ I/O Link Specification defines four virtual channels: requests, posted requests, responses, and probes. The default virtual channel command buffer allocation is shown in Table 30. At least one command buffer should be allocated to each used virtual channel to avoid deadlock. Command buffers do not need to be allocated for a virtual channel that is not used by a link. For example, MCT does not initiate any requests, so there is no need to allocate request or posted request command buffers for an MCT link. A link should not send transactions through a virtual channel that does not have at least one command buffer allocated.

Default allocation of SRI buffers can be found in "SRI-to-XBAR Buffer Count Register" on page 170 and "Free List Buffer Count Register" on page 172. In the SRI-to-Xbar Buffer Count Register (Function 3, Offset 70h), the virtual channels are subdivided into upstream (coherent HyperTransport) and downstream (noncoherent HyperTransport) directions, since SRI must send packets into both coherent HyperTransport and noncoherent HyperTransport links. Some buffers are allocated to a free list pool to use the available resources more efficiently. The buffers in the free list pool are shared by packets in multiple virtual channels. By default, no more than one buffer is allocated to a virtual channel and the rest are allocated to the free list pool in the Free List Buffer

Count Register (Function 3, Offset 7Ch). Any buffer increase to SRI can be added to the free list or allocated directly to a virtual channel. The total number of request command buffers allocated through the Free List Buffer Count Register and the SRI-to-Xbar Buffer Count Register cannot exceed 10.

Table 30. Default Virtual Channel Command Buffer Allocation

Link	Request	Posted Request	Response	Probe	Number of Command Buffers
Coherent HyperTransport™ Links	3	1	6	5	15
Non-coherent HyperTransport™ Links	6	5	4	0	15
SRI	2	3	3	0	8
MCT	0	0	8	3	11

Software can reallocate buffers by modifying buffer count registers (Function 0, Offsets 90h, B0h, D0h, Function 3, Offsets 70h, 78h, 7Ch). The new buffer counts take effect after a warm reset. Since hardware attempts to choose optimal settings, in general, these registers should not be changed.

The following information should be used for buffer reallocation:

1. The number of allocated command buffers for each link should not be greater than the number of available command buffers for that link (see Table 28). Sum of allocated command buffers for three HyperTransport links, SRI, and MCT should not be greater than 64.
2. The number of allocated data buffers for each link should not be greater than the number of available data buffers for that link (see Table 28). There is no restriction on the sum of allocated data buffers for three HyperTransport links, SRI, and MCT.
3. Each present HyperTransport link must have at least one command response buffer (Rsp) allocated, even if the link is designated as a request only link in the routing table.
4. If one HyperTransport link is not present, then other links can use all available buffers for each of them (MCT and HyperTransport links can use one additional buffer, and SRI can use two additional buffers).
5. If all links are present, some buffers from a noncoherent HyperTransport link can be reallocated to coherent HyperTransport links. Table 31 shows command buffer allocation after two response buffers from noncoherent HyperTransport link 0 are reallocated to coherent HyperTransport links 1 and 2.

Table 31. An Example of a Non Default Virtual Channel Command Buffer Allocation

Link	Request	Posted Request	Response	Probe	Number of Command Buffers
HyperTransport™ Link 0	6	5	2	0	13
HyperTransport™ Link 1	3	1	7	5	16
HyperTransport™ Link 2	3	1	7	5	16
SRI	2	3	3	0	8

Table 31. An Example of a Non Default Virtual Channel Command Buffer Allocation

Link	Request	Posted Request	Response	Probe	Number of Command Buffers
MCT	0	0	8	3	11

6. If there are no noncoherent HyperTransport links, one or more response buffers can be reallocated from MCT to HyperTransport links.
7. In a multiprocessor system, number of coherent HyperTransport response buffers should be increased first. The default buffer allocation is sufficient for a uniprocessor system.
8. If a two processor system is implemented with two coherent HyperTransport links between the processors, AMD recommends configuring one link for requests and the other link for responses using the command and data buffer allocations in Table 32 and Table 33.
   a. The request link may have zero response data buffers (RspD).
   b. The response link may have zero request and posted-request data buffers.

Table 32. 2P Dual Coherent Link XBAR Command Buffer Allocation

Link	Request (Req)	Posted Request (PReq)	Response (Rsp)	Probe (Prb)	Number of Command Buffers
Coherent HyperTransport™ Request Link[1,2]	6	3	1	6	16
Coherent HyperTransport™ Response Link[1]	1	0	15	0	16
Non-coherent HyperTransport™ Links[1]	6	5	2	0	13
SRI Upstream (Dev:3x70)	1	1	1	-	8
SRI Downstream (Dev:3x70)	1	1	-	-	
Free List (Dev:3x7C)	1	-	2	-	
MCT (Dev:3x78)	-	-	8	3	11

Note:
1. *Dev:0x90, Dev:0xB0, or Dev:0xD0.*
2. *Route broadcasts to the request link.*

Table 33. 2P Dual Coherent Link Data Buffer Allocation

Link	Request (ReqD)	Posted Request (PReqD)	Response (RspD)	Number of Data Buffers
Coherent HyperTransport™ Request Link[1]	4	3	1	8

Table 33. 2P Dual Coherent Link Data Buffer Allocation

Link	Request (ReqD)	Posted Request (PReqD)	Response (RspD)	Number of Data Buffers
Coherent HyperTransport™ Response Link[1]	1	1	6	8
Note: *Dev:0x90, Dev:0xB0, or Dev:0xD0.*				

9. AMD recommends using different buffer allocations for inner and outer nodes when an eight processor system is implemented using either a ladder (Figure 4) or a twisted ladder (Figure 5) configuration. An outer node is a node with a noncoherent HyperTransport link. Unconnected links are never considered noncoherent HyperTransport links when determining if the node is an inner or outer node. Table 34 and Table 35 show the recommended buffer allocations.

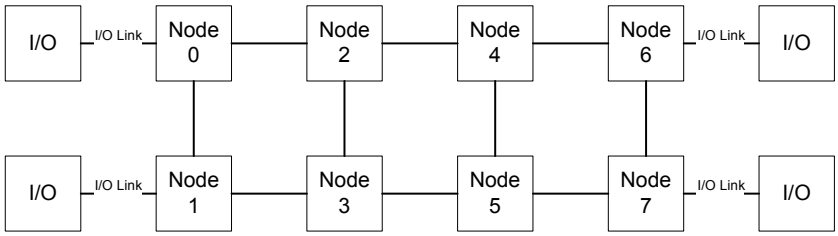


Figure 4. 8P Ladder Configuration

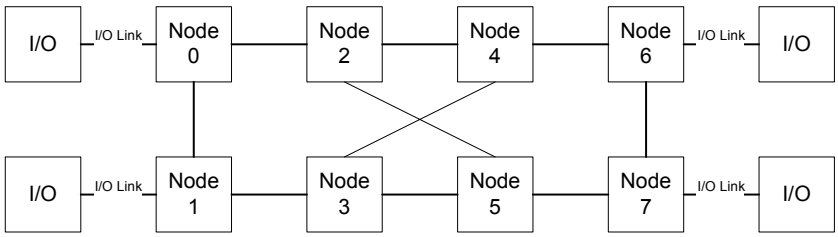


Figure 5. 8P Twisted Ladder Configuration

Table 34. 8P Outer Node Virtual Channel XBAR Command Buffer Allocation

Link	Request (Req)	Posted Request (PReq)	Response (Rsp)	Probe (Prb)	Number of Command Buffers
Coherent HyperTransport™ Links[1]	1	1	8	5	15
Non-coherent HyperTransport™ Links[1]	6	5	4	0	15

Table 34. 8P Outer Node Virtual Channel XBAR Command Buffer Allocation

Link	Request (Req)	Posted Request (PReq)	Response (Rsp)	Probe (Prb)	Number of Command Buffers
SRI Upstream (Dev:3x70)	1	1	1	-	8
SRI Downstream (Dev:3x70)	1	1	-	-	
Free List (Dev:3x7C)	1	-	2	-	
MCT (Dev:3x78)	-	-	8	3	11
Note: *Dev:0x90, Dev:0xB0, or Dev:0xD0.*					

Table 35. 8P Inner Node Virtual Channel XBAR Command Buffer Allocation

Link	Request (Req)	Posted Request (PReq)	Response (Rsp)	Probe (Prb)	Number of Command Buffers
Coherent HyperTransport™ Links (Dev:0x90, Dev:0xB0, Dev:0xD0)	3	1	6	5	15
SRI Upstream (Dev:3x70)	1	1	1	-	8
SRI Downstream (Dev:3x70)	1	1	-	-	
Free List (Dev:3x7C)	1	-	2	-	
MCT (Dev:3x78)	-	-	8	3	11

10. AMD recommends that UMA graphics systems use the flow control buffer allocation in Table 36 to meet the display refresh bandwidth requirements of typical UMA systems.
    - Note that some chipsets may require different values from those specified here.
    - For chipsets that require less display refresh bandwidth a single DispRefReq credit in SRI (Dev:3x74) may be reallocated to the Free Command List (Dev:3x7C) to increase overall system performance. Note that the optimal value needs to be determined by the developer and tested to ensure stability.
    - Contact the chipset vendor for specific recommendations.

Table 36. Recommended Flow Control Buffer Allocations in UMA systems

Dev:0x90 LDT		Dev:3x70 SRI-XBAR		Dev:3x74 XBAR-SRI		Dev:3x78 MCT-XBAR		Dev:3x7C Free List	
RspD	1	DPReq	1	DPReq	1	RspD	8	FRspD	2
PReqD	5	DReq	1	DReq	1	Prb	2	FRsp	1
ReqD	2	URspD	1	DispRefReq	7	Rsp	10	FReq	1
Probe	0	DispRefReq	3	Prb	4			FreeCmd	8[1]/7[2]

1. *Single Core Processor*
2. *Dual Core Processor*

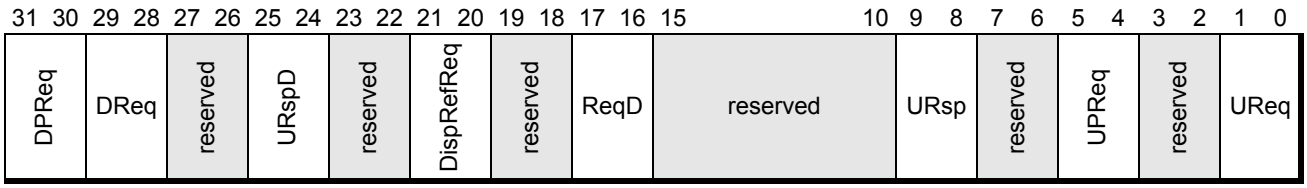
Table 36. Recommended Flow Control Buffer Allocations in UMA systems

Dev:0x90 LDT		Dev:3x70 SRI-XBAR		Dev:3x74 XBAR-SRI		Dev:3x78 MCT-XBAR		Dev:3x7C Free List	
Rsp	1	ReqD	2	UPReq	2				
PReq	5	URsp	1	UReq	1				
Req	10	UPReq	1						
		UReq	1						

1. Single Core Processor
2. Dual Core Processor

4.6.10.1 SRI-to-XBAR Buffer Count Register

SRI-to-XBAR Buffer Count Register **Function 3: Offset 70h**



Bits	Mnemonic	Function	R/W	Reset
31–30	DPReq	Downstream Posted Request Buffer Count	R/W	01b
29–28	DReq	Downstream Request Buffer Count	R/W	01b
27–26	reserved		R	0
25–24	URspD	Upstream Response Data Buffer Count	R/W	01b
23–22	reserved		R	0
21:20	DispRefReq	Display Refresh Request Buffer Count	R/W	0
19–18	reserved		R	0
17–16	ReqD	Request Data Buffer Count	R/W	10b
15–10	reserved		R	0
9–8	URsp	Upstream Response Buffer Count	R/W	01b
7–6	reserved		R	0
5–4	UPReq	Upstream Posted Request Buffer Count	R/W	01b
3–2	reserved		R	0
1–0	UReq	Upstream Request Buffer Count	R/W	01b

Field Descriptions

Upstream Request Buffer Count (UReq)—Bits 1–0. This field defines the number of upstream request buffers available in the XBAR for use by the SRI.

Upstream Posted Request Buffer Count (UPReq)—Bits 5–4. This field defines the number of upstream posted request buffers available in the XBAR for use by the SRI.

Upstream Response Buffer Count (URsp)—Bits 9–8. This field defines the number of upstream response buffers available in the XBAR for use by the SRI.

Request Data Buffer Count (ReqD)—Bits 17–16. This field defines the number of request data buffers available in the XBAR for use by the SRI.

Display Refresh Request Buffer Count (DispRefReq)—Bits 21-20. This field defines the number of display refresh request buffers available in the XBAR for use by the SRI.

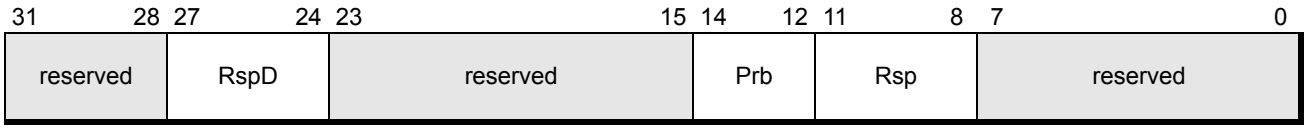
Upstream Response Data Buffer Count (URspD)—Bits 25–24. This field defines the number of upstream response data buffers available in the XBAR for use by the SRI.

Downstream Request Buffer Count (DReq)—Bits 29–28. This field defines the number of downstream request buffers available in the XBAR for use by the SRI.

Downstream Posted Request Buffer Count (DPReq)—Bits 31–30. This field defines the number of downstream posted request buffers available in the XBAR for use by the SRI.

4.6.10.2 MCT-to-XBAR Buffer Count Register

MCT-to-XBAR Buffer Count Register **Function 3: Offset 78h**



Bits	Mnemonic	Function	R/W	Reset
31–28	reserved		R	0
27–24	RspD	Response Data Buffer Count	R/W	8h
23–15	reserved		R	0
14–12	Prb	Probe Buffer Count	R/W	011b
11–8	Rsp	Response Buffer Count	R/W	8h
7–0	reserved		R	0

Field Descriptions

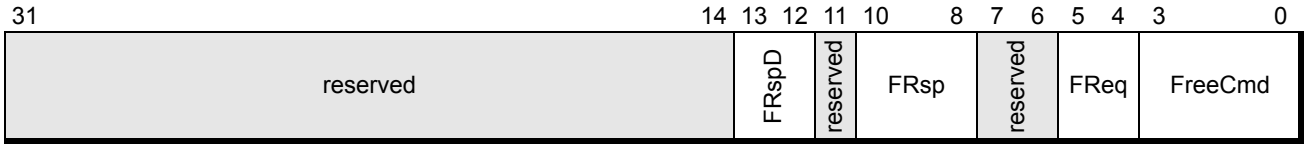
Response Buffer Count (Rsp)—Bits 11–8. This field defines the number of response buffers available in the XBAR for use by the MCT.

Probe Buffer Count (Prb)—Bits 14–12. This field defines the number of probe buffers available in the XBAR for use by the MCT.

Response Data Buffer Count (RspD)—Bits 27–24. This field defines the number of response data buffers available in the XBAR for use by the MCT.

4.6.10.3 Free List Buffer Count Register

Free List Buffer Count Register **Function 3: Offset 7Ch**



Bits	Mnemonic	Function	R/W	Reset
31–14	reserved		R	0
13–12	FRspD	SRI to XBAR Free Response Data Buffer Count	R/W	10b
11	reserved		R	0
10–8	FRsp	SRI to XBAR Free Response Buffer Count	R/W	010b
7–6	reserved		R	0
5–4	FReq	SRI to XBAR Free Request Buffer Count	R/W	01b
3–0	FreeCmd	SRI Free Command Buffer Count	R/W	Bh[1]/Ah[2]

1. *Single Core Processor*
2. *Dual Core Processor*

Field Descriptions

SRI Free Command Buffer Count (FreeCmd)—Bits 3–0. This field defines the number of free list request or posted request buffers available in the SRI for use by the XBAR or CPU.

SRI to XBAR Free Request Buffer Count (FReq)—Bits 5–4. This field defines the number of free list request buffers available in the XBAR for use by the SRI.

SRI to XBAR Free Response Buffer Count (FRsp)—Bits 10–8. This field defines the number of free list response buffers available in the XBAR for use by the SRI.

SRI to XBAR Free Response Data Buffer Count (FRspD)—Bits 13–12. This field defines the number of free list response data buffers available in the XBAR for use by the SRI.

4.6.11 XBAR-to-SRI Buffer Count Register

The Cross Bar Switch to System Request Interface (XBAR-to-SRI) Buffer Count register specifies the number of command buffers for each virtual channel available in the SRI for use by the XBAR. These buffers store commands for traffic routed to the SRI by the XBAR. The buffer counts take effect on the next warm reset.

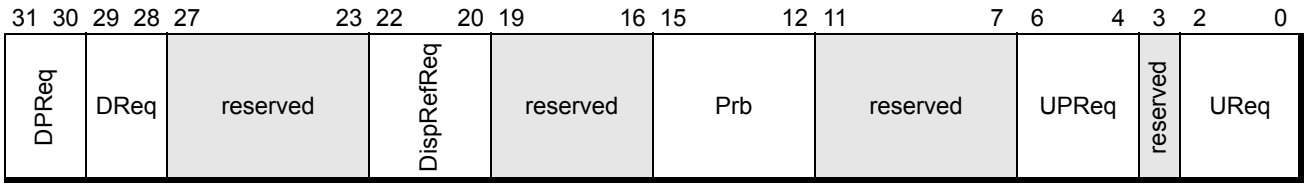
Since the XBAR must route packets in both the coherent HyperTransport technology fabric and down the noncoherent HyperTransport chains, the usual virtual channels are further subdivided into upstream (from noncoherent HyperTransport devices) and downstream (to noncoherent HyperTransport devices) directions. In order to make more efficient use of the available resources, some of the buffers are allocated onto a free list pool in which case these buffers can be used by

packets issued in either direction. See "Free List Buffer Count Register" on page 172. Any increase in the number of fixed allocated buffers in the XBAR-to-SRI Buffer Count register must be compensated by a corresponding reduction of the number of free list buffers specified in the Free List Buffer Count register.

When modifying buffer counts care must be taken to allocate a minimum number of buffers to each virtual channel to avoid deadlock. Requests and Posted requests in both directions require at least one buffer and Probes require at least two buffers to avoid deadlock. Since hardware attempts to choose optimal settings, this register should not in general need to be changed.

Note: *When the Probe Buffer Count is changed, care must be taken to avoid generation of system management events from the time the new value is written into this register until it takes effect (on the next warm reset).*

XBAR-to-SRI Buffer Count Register — Function 3: Offset 74h



Bits	Mnemonic	Function	R/W	Reset
31–30	DPReq	Downstream Posted Request Buffer Count	R/W	01b
29–28	DReq	Downstream Request Buffer Count	R/W	01b
27–23	reserved		R	0
22:20	DispRefReq	Display Refresh Request Buffer Count	R/W	0
19–16	reserved		R	0
15–12	Prb	Probe Buffer Count	R/W	8h
11–7	reserved		R	0
6–4	UPReq	Upstream Posted Request Buffer Count	R/W	001b
3	reserved		R	0
2–0	UReq	Upstream Request Buffer Count	R/W	001b

Field Descriptions

Upstream Request Buffer Count (UReq)—Bits 2–0. This field defines the number of upstream request buffers available in the SRI for use by the XBAR.

Upstream Posted Request Buffer Count (UPReq)—Bits 6–4. This field defines the number of upstream posted request buffers available in the SRI for use by the XBAR.

Probe Buffer Count (Prb)—Bits 15–12. This field defines the number of probe buffers available in the SRI for use by the XBAR.

Display Refresh Request Buffer Count (DispRefReq)—Bits 22-20. This field defines the number of display refresh request buffers available in the SRI for use by the XBAR.

Downstream Request Buffer Count (DReq)—Bits 29–28. This field defines the number of downstream request buffers available in the SRI for use by the XBAR.

Downstream Posted Request Buffer Count (DPReq)—Bits 31–30. This field defines the number of downstream posted request buffers available in the SRI for use by the XBAR.

4.6.12 Display Refresh Flow Control Buffers

The Northbridge has a separate logical path from HyperTransport links to the system memory for display refresh requests. This is necessary to support the bandwidth and latency requirements of display refresh requests in systems where the display-refresh frame buffer is located in the system memory.

A HyperTransport packet is defined as a display-refresh request packet when the read request has isochronous bit, PassPW bit, and RspPassPW bit set, and coherent bit and SeqID cleared. All display-refresh requests are marked as high priority requests and will have higher priority than other requests. In order to accomplish a dedicated logical path to system memory a minimum number of buffers have to be allocated in various queues.

DispRefReq (Function 3, Offset 70h) and DispRefReq (Function 3, Offset 74h) need to be set to allocate the buffers for display refresh requests. At least one buffer should be allocated for display refresh requests to avoid deadlock. Usually more than one buffer may be required to support the necessary bandwidth. The buffer count written to the registers takes effect after a warm reset. See "SRI-to-XBAR Buffer Count Register" on page 170 and "XBAR-to-SRI Buffer Count Register" on page 172 for more information on setting these registers.

4.6.13 Power Management Control Registers

These registers specify the processor response to STPCLK and HALT power management events. Each STPCLK request received contains a 3-bit System Management Action Field (SMAF) which is used as an index into the Power Management Control Low/High registers to select a Power Management Mode (PMM) value to use for the duration of the STPCLK event. A SMAF value of 000 selects PMM0, a SMAF value of 001 selects PMM1, and so on. HALT is hardwired to use PMM7.

4.6.13.1 Power Management Mode (PMM) Value

The PMM value contains the following fields:

Bit	Name	Function	R/W
6–4	ClkSel	Clock Divisor Select	R/W
3	AltVidEn	Alternate VID Change Enable	R/W
2	FidVidEn	FID/VID Change Enable	R/W
1	NBLowPwrEn	Northbridge Low Power Enable	R/W
0	CPULowPwrEn	CPU Low Power Enable	R/W

PMM Value Field Descriptions

Clock Divisor Select (ClkSel)—Bits 6–4. This field specifies the divisor to use when ramping down the CPU clock or the Northbridge clock.

000b = Divide by 8	100b = Divide by 128
001b = Divide by 16	101b = Divide by 256
010b = Divide by 32	110b = Divide by 512
011b = Divide by 64	111b = reserved

Alternate VID Change Enable (AltVidEn)—Bit 3. Enables a VID change to the value programmed in the AltVid field (Function 3, Offset D8h) after CPU clocks and Northbridge clocks are ramped down. This VID change only occurs if the processor is in the minimum P-state when the STPCLK event occurred. This bit should never be set for FID/VID changes or when UMA graphics are used.

0 = Alternate VID change disabled
1 = Alternate VID change enabled

FID/VID Change Enable (FidVidEn)—Bit 2. Enables a change in Frequency ID (FID) and/or Voltage ID (VID). The CPU and the Northbridge clocks must also be ramped down (see NBLowPwrEn and CPULowPwrEn).

0 = FID/VID change disabled
1 = FID/VID change enabled

Northbridge Low Power Enable (NBLowPwrEn)—Bit 1. Causes the Northbridge clock to ramp down according to the clock divisor specified in ClkSel and puts the DRAM in self-refresh mode. The CPU clock must also be ramped down (see CPULowPwrEn)

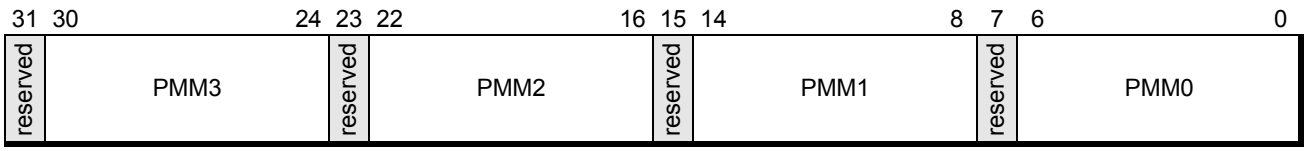
0 = Northbridge low power disabled
1 = Northbridge low power enabled

CPU Low Power Enable (CPULowPwrEn)—Bit 0. Causes the CPU clock to ramp down according to the clock divisor specified in ClkSel.

0 = CPU low power disabled
1 = CPU low power enabled

4.6.13.2 Power Management Control Low Register

Power Management Control Low Register **Function 3: Offset 80h**



Bits	Mnemonic	Function	R/W	Reset
31	reserved		R	0
30–24	PMM3	Power Management Mode 3	R/W	0
23	reserved		R	0
22–16	PMM2	Power Management Mode 2	R/W	0
15	reserved		R	0
14–8	PMM1	Power Management Mode 1	R/W	0
7	reserved		R	0
6–0	PMM0	Power Management Mode 0	R/W	0

Field Descriptions

Power Management Mode 0 (PMM0)—Bits 6–0. See "Power Management Mode (PMM) Value" on page 174 for a description of these bits.

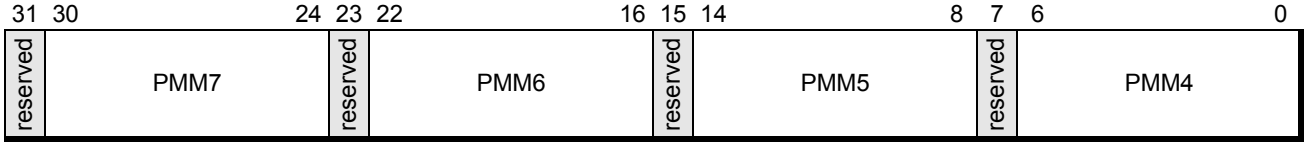
Power Management Mode 1 (PMM1)—Bits 14–8. See "Power Management Mode (PMM) Value" on page 174 for a description of these bits.

Power Management Mode 2 (PMM2)—Bits 22–16. See "Power Management Mode (PMM) Value" on page 174 for a description of these bits.

Power Management Mode 3 (PMM3)—Bits 30–24. See "Power Management Mode (PMM) Value" on page 174 for a description of these bits.

4.6.13.3 Power Management Control High Register

Power Management Control High Register **Function 3: Offset 84h**



Bits	Mnemonic	Function	R/W	Reset
31	reserved		R	0
30–24	PMM7	Power Management Mode 7	R/W	0
23	reserved		R	0
22–16	PMM6	Power Management Mode 6	R/W	0
15	reserved		R	0
14–8	PMM5	Power Management Mode 5	R/W	0
7	reserved		R	0
6–0	PMM4	Power Management Mode 4	R/W	0

Field Descriptions

Power Management Mode 4 (PMM4)—Bits 6–0. See "Power Management Mode (PMM) Value" on page 174 for a description of these bits.

Power Management Mode 5 (PMM5)—Bits 14–8. See "Power Management Mode (PMM) Value" on page 174 for a description of these bits.

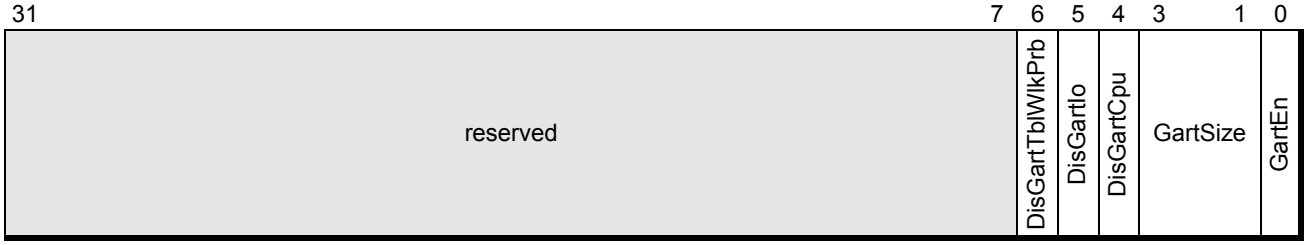
Power Management Mode 6 (PMM6)—Bits 22–16. See "Power Management Mode (PMM) Value" on page 174 for a description of these bits.

Power Management Mode 7 (PMM7)—Bits 30–24. See "Power Management Mode (PMM) Value" on page 174 for a description of these bits.

4.6.14 GART Aperture Control Register

This register contains the aperture size and enable for the graphics aperture relocation table (GART) mechanism.

GART Aperture Control Register **Function 3: Offset 90h**



Bits	Mnemonic	Function	R/W	Reset
31–7	reserved		R	0
6	DisGartTblWlkPrb	Disable GART Table Walk Probes	R/W	0
5	DisGartIo	Disable GART I/O Accesses	R/W	0
4	DisGartCpu	Disable GART CPU Accesses	R/W	0
3–1	GartSize	GART Size	R/W	0
0	GartEn	GART Enable	R/W	0

Field Descriptions

GART Enable (GartEn)—Bit 0. Enables GART address translation for accesses falling within the GART aperture. GartAperBaseAddr (Function 3, Offset 94h) and other related registers should be initialized before GartEn is set.

GART Size (GartSize)—Bits 3–1. Defines the virtual address space to be allocated to the GART.

000b = 32 Mbytes
001b = 64 Mbytes
010b = 128 Mbytes

011b = 256 Mbytes
100b = 512 Mbytes
101b = 1 Gbyte
110b = 2 Gbyte
111b = reserved

Disable GART CPU Accesses (DisGartCpu)—Bit 4. Disables requests from CPUs from accessing the GART.

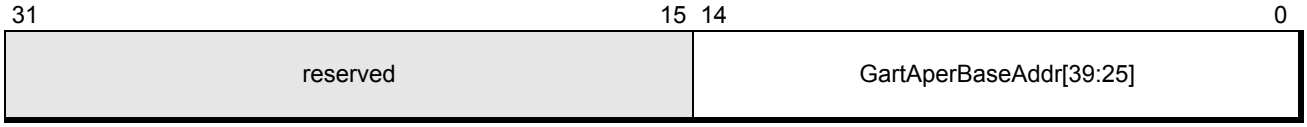
Disable GART I/O Accesses (DisGartIo)—Bit 5. Disables requests from I/O devices from accessing the GART.

Disable GART Table Walk Probes (DisGartTblWlkPrb)—Bit 6. Disables generation of probes for GART table walks. This bit may be set to improve performance in cases where the GART table entries are in address space which is marked uncacheable in processor MTRRs or page tables.

4.6.15 GART Aperture Base Register

This register contains the base address of the aperture for the graphics aperture relocation table (GART). The GART aperture base along with the GART aperture size define the GART aperture address window. BIOS can place the GART aperture below the 4-gigabyte level in address space in order to support legacy operating systems and legacy AGP cards (that do not support 64-bit address space).

GART Aperture Base Register **Function 3: Offset 94h**



Bits	Mnemonic	Function	R/W	Reset
31–15	reserved		R	0
14–0	GartAperBaseAddr[39:25]	GART Aperture Base Address Bits 39–25	R/W	X

"X" in the Reset column indicates that the field initializes to an undefined state after reset.

Field Descriptions

GART Aperture Base Address Bits 39–25 (GartAperBaseAddr[39:25])—Bits 14–0. These bits are used to compare to the incoming addresses to determine if they are in the aperture range. They are active based on the aperture size. The remaining address bits are assumed to be 0.

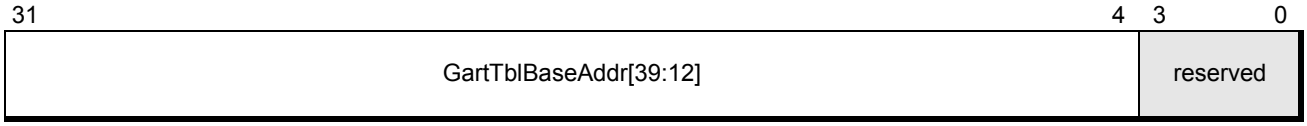
[39:25] = 32 Mbytes
[39:26] = 64 Mbytes
[39:27] = 128 Mbytes
[39:28] = 256 Mbytes

[39:29] = 512 Mbytes
[39:30] = 1 Gbytes
[39:31] = 2 Gbytes

4.6.16 GART Table Base Register

This register contains the base address of the translation table for the graphics aperture relocation table (GART). The GART table base points to the base address of a table of 32-bit GART page table entries (PTEs) that contain the physical addresses to use for an incoming address that falls within the GART aperture.

GART Table Base Register **Function 3: Offset 98h**



Bits	Mnemonic	Function	R/W	Reset
31–4	GartTblBaseAddr[39:12]	GART Table Base Address Bits 39–12	R/W	X
3–0	reserved		R	0

"X" in the Reset column indicates that the field initializes to an undefined state after reset.

Field Descriptions

GART Table Base Address Bits 39–12 (GartTblBaseAddr[39:12])—Bits 31–4. These bits point to the base of the table of GART PTEs to be used for GART address translation. Table 37 shows the organization of each 32-bit PTE in the GART table.

Table 37. GART PTE Organization

Bits	Description
0	Valid
1	Coherent
3–2	reserved
11–4	PhysAddr[39:32]
31–12	PhysAddr[31:12]

4.6.17 GART Cache Control Register

This register controls the GART cache, which is used to cache the most recently translated GART aperture accesses. The GART cache is enabled automatically whenever the GART aperture is enabled.

GART Cache Control Register — Function 3: Offset 9Ch

Bits	Mnemonic	Function	R/W	Reset
31–2	reserved		R	0
1	GartPteErr	GART PTE Error	R/WC	0
0	InvGart	Invalidate GART	R/W	0

Field Descriptions

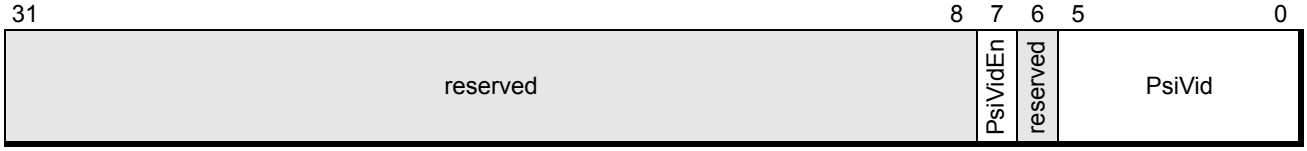
Invalidate GART (InvGart)—Bit 0. Setting this bit causes the GART cache to be invalidated. This bit is cleared by hardware when the invalidation is complete.

GART PTE Error (GartPteErr)—Bit 1. This bit is set when an invalid PTE is encountered during a table walk. Cleared by writing a 1.

4.6.18 Power Control Miscellaneous Register

This register specifies how the PSI_L signal is controlled. This signal may be used by the voltage regulator to improve efficiency while in reduced power states.

Power Control Miscellaneous Register — Function 3: Offset A0h



Bits	Mnemonic	Function	R/W	Reset
31–8	reserved		R	0
7	PsiVidEn	PsiVID Enable	R/W	0
6	reserved		R	0
5–0	PsiVid	PsiVID	R/W	0

Field Descriptions

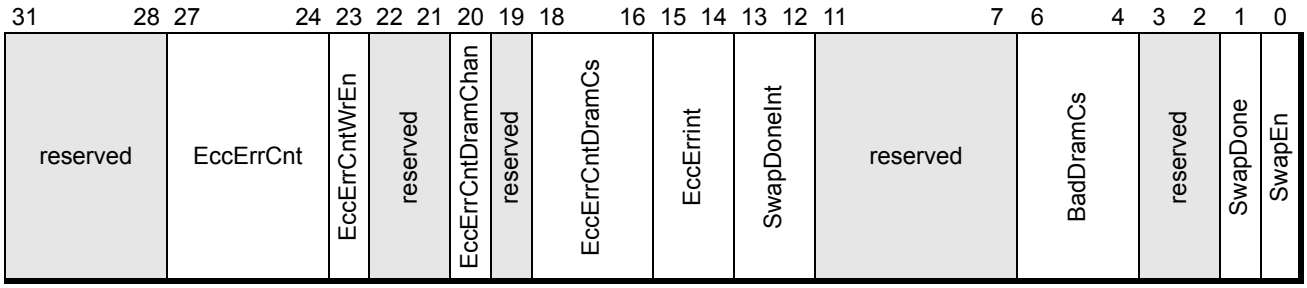
PsiVID (PsiVid)—Bits 5–0. When enabled by PsiVidEn, this field specifies the threshold value of VID code generated by the processor, which in turn determines the state of PSI_L. When the VID code generated by the processor is less than PsiVid (i.e., the VID code is specifying a higher voltage level than the PsiVid-specified voltage level), then PSI_L is high; when the VID code is greater than or equal to PsiVid, PSI_L is driven low.

PsiVID Enable (PsiVidEn)—Bit 7. This bit enables checking of PsiVid and the assertion of PSI_L.
0 = PSI_L disabled and drive high.
1 = PSI_L enable and controlled by PsiVid.

4.6.19 On-Line Spare Control Register

This register is used by system software to control the transition to the spare rank and to determine which rank has gone bad.

On-Line Spare Control Register — Function 3: Offset B0h



Bits	Mnemonic	Function	R/W	Reset
31–28	reserved		R	0
27–24	EccErrCnt	ECC Error Count	R/W	0
23	EccErCntWrEn	ECC Error Count Write Enable	R/W	0
22–21	reserved		R	0
20	EccErrCntDramChan	ECC Error Counter DRAM Channel	R/W	0
19	reserved		R	0
18–16	EccErrCntDramCs	ECC Error Counter DRAM Chip Select	R/W	0
15–14	EccErrInt	ECC Error Interrupt Type	R/W	0
13–12	SwapDoneInt	Swap Done Interrupt Type	R/W	0
11–7	reserved		R	0
6–4	BadDramCs	Bad DRAM Chip Select	R/W	0
3–2	reserved		R	0
1	SwapDone	Swap Done	R/W	0
0	SwapEn	Swap Enable	R/W	0

Field Descriptions

Swap Enable (SwapEn)—Bit 0. Setting this bit causes the hardware to copy the contents of the DRAM chip select identified by the BadDramCs field to the spare rank. The DRAM scrubber must be enabled with a scrub address range that encompasses the address of the bad chip select for the swap to occur. The scrub rate is accelerated automatically by hardware until the copy completes at which point the scrub rate returns to normal. The DRAM scrub address registers (Function 3: Offset 5Ch and 60h) should not be modified while the copy is taking place. During the copy, DRAM accesses (including accesses to the bad CS) proceed normally.

Once this bit is set it cannot be cleared by software. This bit is not cleared by a warm reset once the swap has completed.

Swap Done (SwapDone)—Bit 1. This bit is set when the hardware has completed copying the data to the spare rank. This bit can also be set by BIOS to immediately enable the swap to the spare rank after a suspend to RAM. Once this bit is set it cannot be cleared by software. This bit cannot be set by software if DRAM is enabled (Function 2 OffsetA0h bit 9 = 1). This bit is not cleared by a warm reset.

Bad Dram Chip Select (BadDramCs)—Bits 6–4. This field is programmed with the DRAM chip select to be replaced. This field cannot be written when SwapDone is set. This field is not cleared by a warm reset.

Swap Done Interrupt Type (SwapDoneInt)—Bits 13–12. This field specifies what type of interrupt generated when a swap is complete.

00b = No Interrupt
01b = Reserved
10b = SMI[1]
11b = Reserved

ECC Error Interrupt Type (EccErrInt)—Bits 15–14. This field specifies if an interrupt is generated when the EccErrCnt field for any chip select and channel transitions to 1111b.

00b = No Interrupt
01b = Reserved
10b = SMI[1]
11b = Reserved

1. *This interrupt will always be routed to cpu0 in a dual core processor.*

ECC Error Counter DRAM Chip Select (EccErrCntDramCs)—Bits 18–16. This field specifies the DRAM chip select for which ECC error count information is returned in EccErrCnt field.

ECC Error Counter DRAM Channel (EccErrCntDramChan)—Bit 20. This bit specifies the DRAM channel for which ECC error count information is returned in EccErrCnt field.

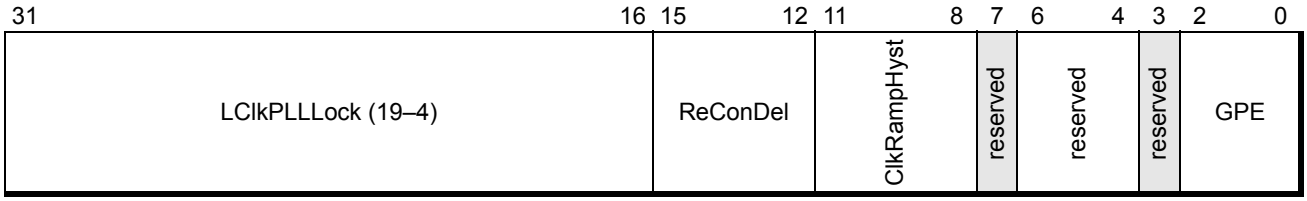
ECC Error Counter Write Enable (EccErrCntWrEn)—Bit 23. When set, this bit enables writes to the EccErrCnt field.

ECC Error Counter (EccErrCnt)—Bits 27–24. This field returns the number of ECC errors for the chip select selected by the EccErrCntDramCs and EccErrCntDramChan fields. This field can be written by software to clear the count. This field returns Fh if 15 or more correctable ECC errors have occurred.

4.6.20 Clock Power/Timing Low Register

This register controls the transition times between various voltage and frequency states. The value of this register is maintained through a warm reset and is initialized to 0 on a cold reset.

Clock Power/Timing Low Register — Function 3: Offset D4h



Bits	Mnemonic	Function	R/W	Reset
31–16	LClkPLLLock	HyperTransport™ CLK PLL Lock Counter Bits 19–4	R/W	0
15–12	ReConDel	Link Reconnect Delay	R/W	Ah
11-8	ClkRampHyst	Clock Ramp Hysteresis	R/W	0
7	reserved		R	0
6–4	reserved	SBZ	R/W	0
3	reserved		R	0
2–0	GPE	Good Phase Error	R/W	0

Field Descriptions

Good Phase Error (GPE)—Bits 2–0. This field defines the BCLK PLL time until good phase error. Counting occurs when at full frequency.

000b = reserved
001b = 200 system clocks (1us)
010b = 400 system clocks (2us)
011b = 600 system clocks (3us)
100b = 800 system clocks (4us)
101b = 1600 system clocks (8us)
110b = 3000 system clocks (15us)
111b = 20000 system clocks (100us)

Clock Ramp Hysteresis (ClkRampHyst)—Bits 11-8. A non-zero value in this field enables a hysteresis time which prevents the CPU clock grid from being ramped down after processing a probe. It avoids unnecessary changes of the CPU clock grid when the probe arrival rate is relatively low.

0000b= 0
0001b= 125 ns
0010b= 250 ns
0011b= 375 ns
0100b= 500 ns
0101b= 750 ns
0110b= 1000 ns
0111b= 2000 ns
1000b= 4 µs
1001b= 8 µs

1010b= 16 µs
1011b= 32 µs
1100b= Reserved
1101b= Reserved
1110b= Reserved
1111b= Reserved

Link Reconnect Delay (ReConDel)—Bits 12-15. A non-zero value in this field specifies the approximate delay, in microseconds, from the deassertion of LDTSTOP_L until the link initialization process is allowed to start. This delay is only applied if LdtStopTriEn = 1 and LdtStopTriClkOvr = 0 (Function 0, Offsets 84h, Ah4, C4h).

0000b= 0
0001b= 1 µs
0010b= 2 µs
0011b= 3 µs
0100b= 4 µs
0101b= 5 µs
0110b= 6 µs
0111b= 7 µs
1000b= 8 µs
1001b= 9 µs
1010b= 10 µs
1011b= Reserved
1100b= Reserved
1101b= Reserved
1110b= Reserved
1111b= Reserved

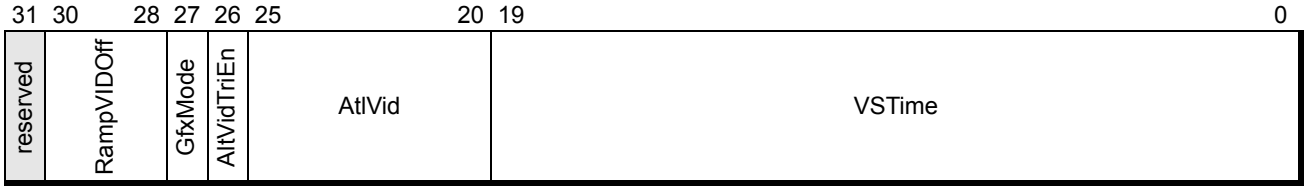
HyperTransport CLK PLL Lock Counter Bits 19–4 (LClkPLLLock)—Bits 31–16. This bit field indicates how long it takes for the slowest HyperTransport technology clock PLL to ramp to its new frequency and lock.

The 16 bits represent the most significant 16 bits of a 20-bit value. The number reflects the number of system bus clocks (5 ns) to wait. It is only necessary to wait a short amount of time in the event the frequency change is one that maintains the VCO frequency. In this case, the PLL will be re-locked quickly.

4.6.21 Clock Power/Timing High Register

This register controls the transition times between various voltage and frequency states. The value of this register is maintained through a warm reset and is initialized to 0 on a cold reset.

Clock Power/Timing High Register **Function 3: Offset D8h**



Bits	Mnemonic	Function	R/W	Reset
31	reserved		R	0
30–28	RampVIDOff	Ramp VID Offset	R/W	0
27	GfxMode	Graphics Mode	R/W	0
26	AltVidTriEn	AltVID HyperTransport™ Clock Tristate Enable	R/W	0
25–20	AltVid	Alternate VID	R/W	0
19–0	VSTime	Voltage Regulator Stabilization Time	R/W	0

Field Descriptions

Voltage Regulator Stabilization Time (VSTime)—Bits 19–0. This field indicates when the voltage regulator is stable at the Ramp VID before ramping up the clocks. The count is the number of 5-ns system clock cycles.

00000h = 0 ns
00001h = 5 ns
00002h = 10 ns
...
FFF36h = 5,241,870 ns
FFF37h = 5,241,875 ns
FFF38h - FFFFFh = Reserved

Alternate VID (AltVid)—Bits 25–20. This field specifies the alternate VID while in low power states when enabled through PMM fields in the Power Management Control registers. Switching to AltVID can only be initiated by a StpClk message from the I/O Hub.

If an attempt is made to write a VID value that corresponds to a voltage greater than the voltage that MaxVID corresponds to in the FIDVID_STATUS register then the MaxVID value is written instead. If an attempt is made to write a VID value that corresponds to a voltage lower than the voltage that MinVID corresponds to in the FIDVID_STATUS register then the MinVID value is written instead. See "Register Differences in Revisions of AMD NPT Family 0Fh Processors" on page 27 for revision information about this register.

AltVID HyperTransport Clock Tristate Enable (AltVidTriEn)—Bit 26. This bit enables the HyperTransport clock to be tristated when AltVID is active.

Graphics Mode (GfxMode)—Bit 27. This bit is set by BIOS to indicate the graphics mode used by the system. This bit must be set to 0 if LDTSTOP_L with SMAF 001b is used to change link width/freq at boot.See "Register Differences in Revisions of AMD NPT Family 0Fh Processors" on page 27 for revision information about this register.

0b = Discrete graphics mode
1b = UMA graphics mode

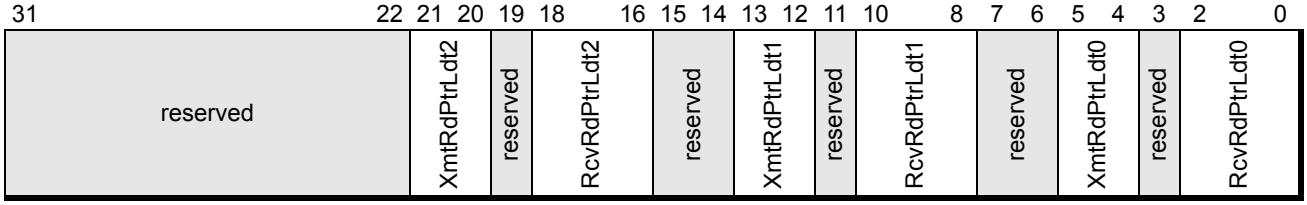
Ramp VID Offset (RampVIDOff)—Bits 30–28. Defines the amount of extra voltage required while the PLL is ramping for low power states. This "over-voltage" is applied only until the PLL has phase locked to within specifications.

000b = 0 mV
001b = 25 mV
010b = 50 mV
011b = 75 mV
100b = 100 mV
101b = 125 mV
110b = 150 mV
111b = 175 mV

4.6.22 HyperTransport™ FIFO Read Pointer Optimization Register

This register allows the separation of read/write pointers in the HyperTransport technology receive/transmit FIFOs to be changed from their default settings. The pointer separation written to this register takes effect after a warm reset. The value of this register is maintained through a warm reset and is initialized to 0 on a cold reset.

HyperTransport™ FIFO Read Pointer Optimization Register Function 3: Offset DCh



Bits	Mnemonic	Function	R/W	Reset
31–22	reserved		R	0
21–20	XmtRdPtrLdt2	Change Read Pointer For HyperTransport™ Link 2 Transmitter	R/W	0

Bits	Mnemonic	Function	R/W	Reset
19	reserved		R	0
18–16	RcvRdPtrLdt2	Change Read Pointer For HyperTransport Link 2 Receiver	R/W	0
15–14	reserved		R	0
13–12	XmtRdPtrLdt1	Change Read Pointer For HyperTransport Link 1 Transmitter	R/W	0
11	reserved		R	0
10–8	RcvRdPtrLdt1	Change Read Pointer For HyperTransport Link 1 Receiver	R/W	0
7–6	reserved		R	0
5–4	XmtRdPtrLdt0	Change Read Pointer For HyperTransport Link 0 Transmitter	R/W	0
3	reserved		R	0
2–0	RcvRdPtrLdt0	Change Read Pointer For HyperTransport Link 0 Receiver	R/W	0

Field Descriptions

Change Read Pointer For HyperTransport Link 0 Receiver (RcvRdPtrLdt0)—Bits 2–0. See RcvRdPtrLdt2 for values.

Change Read Pointer For HyperTransport Link 0 Transmitter (XmtRdPtrLdt0)—Bits 5–4. See XmtRdPtrLdt2 for values.

Change Read Pointer For HyperTransport Link 1 Receiver (RcvRdPtrLdt1)—Bits 10–8. See RcvRdPtrLdt2 for values.

Change Read Pointer For HyperTransport Link 1 Transmitter (XmtRdPtrLdt1)—Bits 13–12. See XmtRdPtrLdt2 for values.

Change Read Pointer For HyperTransport Link 2 Receiver (RcvRdPtrLdt2)—Bits 18–16. Moves the read pointer for the HyperTransport receive FIFO closer to the write pointer thereby reducing latency through the receiver.

000b = RdPtr assigned by hardware
001b = Move RdPtr closer to WrPtr by 1 HyperTransport clock period
010b = Move RdPtr closer to WrPtr by 2 HyperTransport clock periods
011b = Move RdPtr closer to WrPtr by 3 HyperTransport clock periods
100b = Move RdPtr closer to WrPtr by 4 HyperTransport clock periods
101b = Move RdPtr closer to WrPtr by 5 HyperTransport clock periods
110b = Move RdPtr closer to WrPtr by 6 HyperTransport clock periods
111b = Move RdPtr closer to WrPtr by 7 HyperTransport clock periods

The BIOS must set this field to 5 for all noncoherent HyperTransport links and to 5 or more for all coherent HyperTransport links. The BIOS must issue a warm reset for the settings to take affect and the warm reset must occur prior to an LDTSTOP_L initiated link frequency change. AMD recommends setting this field to 6 for all coherent HyperTransport links for

optimal performance, however the optimal value needs to be determined by the developer and tested to ensure stability.

Change Read Pointer For HyperTransport Link 2 Transmitter (XmtRdPtrLdt2)—Bits 21–20. Moves the read pointer for the HyperTransport technology transmit FIFO closer to the write pointer thereby reducing latency through the transmitter.

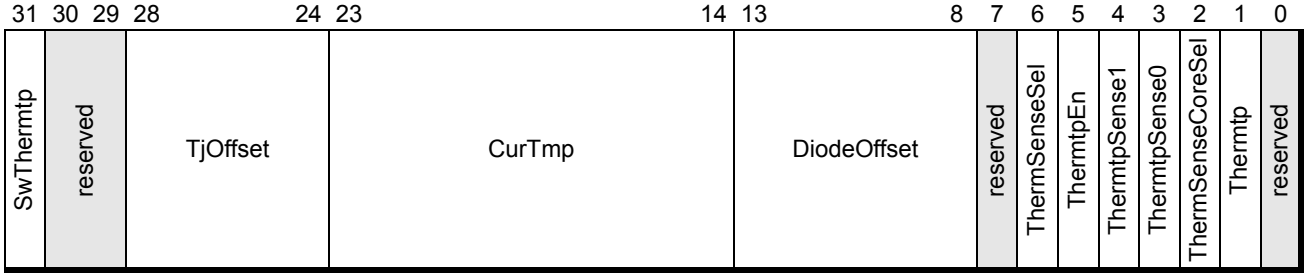
00b = RdPtr assigned by hardware
01b = Move RdPtr closer to WrPtr by 1 HyperTransport clock period
10b = Move RdPtr closer to WrPtr by 2 HyperTransport clock periods
11b = Move RdPtr closer to WrPtr by 3 HyperTransport clock periods

The BIOS must set this field to 2 for all coherent and noncoherent HyperTransport links and issue a warm reset prior to an LDTSTOP_L initiated link frequency change.

4.6.23 Thermtrip Status Register

The Thermtrip Status register provides status information regarding the THERMTRIP thermal sensor.

Thermtrip Status Register **Function 3: Offset E4h**



Bits	Mnemonic	Function	R/W	Reset
31	SwThermtp	Software Thermtrip	W	0
30–29	reserved		R	0
28-24	TjOffset	Tj Offset	R	0
23-14	CurTmp	Current Temperature	R	
13–8	DiodeOffset	Diode Offset	R	
7	reserved		R	0
6	ThermSenseSel	Thermal Sensor Select	R/W	0
5	ThermtpEn	Thermtrip Enabled	R	
4	ThermtpSense1	Thermtrip Sense 1	R	
3	ThermtpSense0	Thermtrip Sense 0	R	
2	ThermSenseCoreSel	Thermal Sensor Core Select	R/W	0
1	Thermtp	Thermtrip	R	
0	reserved		R	0

Field Descriptions

Thermtrip (Thermtp)—Bit 1. Set to 1 if a temperature sensor trip occurs and was enabled.

Thermal Sensor Core Select (ThermSenseCoreSel)—Bit 2. This bit selects the CPU whose temperature is reported in the CurTemp field. This bit only applies to dual core processors. For single core processors CPU0 Thermal Sensor is always selected. See "Register Differences in Revisions of AMD NPT Family 0Fh Processors" on page 27 for revision information about this register.

0b = CPU1 Thermal Sensor.
1b = CPU0 Thermal Sensor.

Thermtrip Sense 0 (ThermtpSense0)—Bit 3. Set to 1 if a temperature sensor trip occurs for CPU0 on a single core processor, or for CPU1 on a dual core processor. The value of this bit is maintained through warm reset.

Thermtrip Sense 1 (ThermtpSense1)—Bit 4. Set to 1 if a temperature sensor trip occurs for CPU0 on a dual core processor. The value of this bit is maintained through warm reset.

Thermtrip Enabled (ThermtpEn)—Bit 5. Indicates that the thermtrip temperature sensor is enabled. When this bit is set to 1, a THERMTRIP High event causes the hardware to shut down the PLL, assert the THERMTRIP output pin and set the ThermtpHi bit. The ThermtpSense bit is set for a THERMTRIP High event, irrespective of the state of ThermtpEn.

Thermal Sensor Select (ThermSenseSel)—Bit 6. This bit selects which sensor in the CPU whose temperature is reported in the CurTemp field. See "Register Differences in Revisions of AMD NPT Family 0Fh Processors" on page 27 for revision information about this register.

0b = CPU[0,1] Thermal Sensor 0.
1b = CPU[0,1] Thermal Sensor 1.

Diode Offset (DiodeOffset[5:0])—Bits 13–8. Thermal diode offset is used to correct the measurement made by an external temperature sensor. This diode offset supports temperature sensors using two sourcing currents only. Other sourcing current implementations are not compatible with the diode offset and are not supported by AMD. The allowable offset range is provided in the appropriate processor functional data sheet, and the maximum offset can vary for different processors. A correction to the offset may be needed for some temperature sensors. Contact the temperature sensor vendor to determine whether an offset correction is needed.

With the diode offset, the thermal diode can be used to ensure the processor is within its functional temperature limits. When the thermal diode measurement plus diode offset equals the maximum control temperature ($T_{CONTROL}$ max), the processor has reached its case temperature specification (T_{CASE} max). The maximum control temperature is provided in the appropriate power and thermal data sheet. The relationship between T_{CASE} max and $T_{CONTROL}$ max is described in more detail in section Section 3.7.3.1 on page 49.

This field defines a 6 bit signed value between the range of +10 C to -52 C. The diode offset value (in degrees celcius) should be added to the external temperature sensor reading.

000000b = undefined.
000001b = +10C
000010b = +9C
...
111110b = -51C
111111b = -52C

Current Temperature (CurTmp)—Bits 23-14. This field returns the current value of the internal thermal sensor. The value returned in this field is selected by ThermSenseCoreSel and ThermSenseSel. See "Register Differences in Revisions of AMD NPT Family 0Fh Processors" on page 27 for revision information about this field.

Revision F encodings bits 23-16 (ignore bits 15-14)
00h = -49C
01h = -48C
...
ffh = 206C
Revision G encodings bits 23-14
000h = -49.00C
001h = -48.75C
002h = -48.50C
003h = -48.25C
004h = -48.00C
...
0C4h = 0.00C
...
3ffh = 206.75C

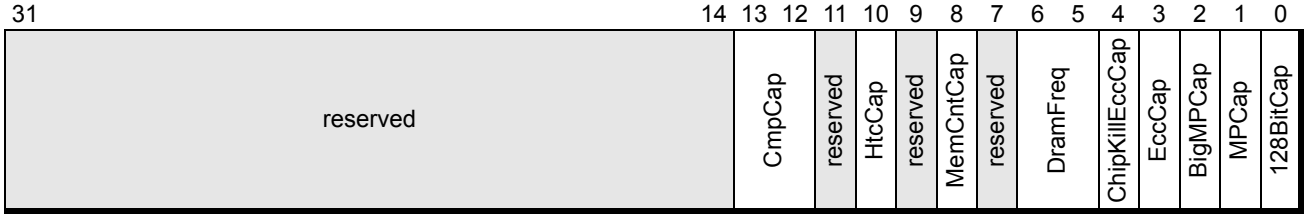
Tj Offset (TjOffset)—Bits 28-24. This field is the offset from CurTmp used to normalize to Tcontrol. Tcontrol = CurTmp - TjOffset*2 - 49. If CurTmp - (TjOffset*2) - 49 < 0 Tcontrol = 0. See "Register Differences in Revisions of AMD NPT Family 0Fh Processors" on page 27 for revision information about this field.

Software Thermtrip (SwThermtp)—Bit 31. Writing a 1 to this bit position induces a THERMTRIP event. This bit is write-only and returns 0 when read. This is a diagnostic bit, and it should be used for testing purposes only.

4.6.24 Northbridge Capabilities Register

The Northbridge Capabilities register indicates whether or not this Northbridge is capable of certain behavior.

Northbridge Capabilities Register — Function 3: Offset E8h



Bits	Mnemonic	Function	R/W	Reset
31–14	reserved		R	0
13–12	CmpCap	Dual-Core Capability	R	
11	reserved		R	0
10	HtcCap	HTC Capable	R	
9	reserved		R	0
8	MemCntCap	Memory Controller Capable	R	
7	reserved		R	0
6–5	DramFreq	Maximum DRAM Frequency	R	
4	ChipKillEccCap	Chip-Kill ECC Capable	R	
3	EccCap	ECC Capable	R	
2	BigMPCap	Big MP Capable	R	
1	MPCap	MP Capable	R	
0	128BitCap	128-Bit DRAM Capable	R	

Field Descriptions

128-Bit DRAM Capable (128BitCap)—Bit 0. This bit is set to 1 if the Northbridge is capable of supporting a 128-bit DRAM interface.

MP Capable (MPCap)—Bit 1. This bit is set to 1 if the Northbridge is capable of supporting multiprocessor systems.

Big MP Capable (BigMPCap)—Bit 2. This bit is set to 1 if the Northbridge is capable of supporting multiprocessor systems greater than DP.

ECC Capable (EccCap)—Bit 3. This bit is set to 1 if the Northbridge is capable of supporting ECC.

Chip-Kill ECC Capable (ChipKillEccCap)—Bit 4. This bit is set to 1 if the Northbridge is capable of supporting chip-kill ECC.

Maximum DRAM Frequency (DramFreq)—Bits 6–5. Indicates the maximum DRAM frequency supported.

00b = No limit
01b = 333 MHz
10b = 266 MHz
11b = 200 MHz

Memory Controller Capable (MemCntCap)—Bit 8. This bit is set to 1 if the Northbridge is capable of supporting an on-chip memory controller.

HTC Capable (HtcCap)—Bit 10. This bit is set to 1 if the Northbridge is capable of supporting HTC. See "Register Differences in Revisions of AMD NPT Family 0Fh Processors" on page 27 for revision information about this field.

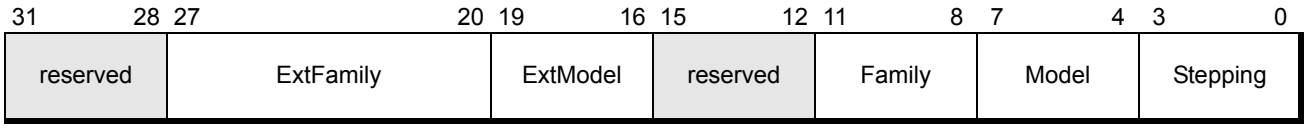
Dual-Core Capability (CmpCap)—Bits 13–12. Indicates the number of CPU cores present in the processor.

00b = Single core processor
01b = Dual core processor
10b = Reserved
11b = Reserved

4.6.25 CPUID Family Model Register

This register returns the same information that is returned in EAX by CPU function 0000_0001h.

CPUID Family Model Register **Function 3: Offset FCh**



Bits	Mnemonic	Function	R/W	Reset
31–28	reserved		R	0
27–20	ExtFamily	Extended Family	R	X
19–16	ExtModel	Extended Model	R	X
15–12	reserved		R	0
11–8	Family	Family	R	X
7–4	Model	Model	R	X
3–0	Stepping	Stepping	R	X

Field Descriptions

Stepping (Stepping)—Bits 3–0. These bits identify the processor revision.

Model (Model)—Bits 7–4. These bits identify the processor model number.

Family (Family)—Bits 11–8. These bits identify the processor instruction family.

Extended Model (ExtModel)—Bits 19–16. These bits provide additional model information if Family is F0h.

Extended Family (ExtFamily)—Bits 27–20. These bits provide additional family information if Family is F0h.

5 DRAM Configuration

5.1 Programming Interface

This section describes how to program various DRAM controller registers. There are two principal areas of configuration:

- Configuration state information obtained via the DIMM Serial Presence Detect (SPD) ROMs. Non-SPD memory sizing is not supported.
- All other configuration state information.

5.1.1 SPD ROM-Based Configuration

The SPD device is an EEPROM on the DIMM encoded by the DIMM manufacturer. The description of the EEPROM is usually provided on a data sheet for the DIMM itself along with data describing the memory devices used. The data describes configuration and speed characteristics of the DIMM and the SDRAM components mounted on the DIMM. The data sheet also contains the DIMM byte values that are encoded in the SPD on the DIMM.

BIOS acquires the values encoded in the SPD ROM through the I/O hub, which obtains the information through a secondary device connected to the I/O hub through the SMBus. This secondary device communicates with the DIMM by means of the I^2C bus.

The SPD ROM provides values for several DRAM timing parameters that are required by the DRAM controller. These parameters are:

- tCL: (CAS latency)
- tRC: Active-to-Active/Auto Refresh command period
- tRFC: Auto-Refresh-to-Active/Auto Refresh command period
- tRCD: Active-to-Read-or-Write delay
- tRRD: Active-Bank-A to-Active-Bank-B delay
- tRAS: Active-to-Precharge delay
- tRP: Precharge time
- tREF: Refresh interval
- tRTP: Internal Read to Precharge command delay
- tWTR: Internal Write to Read command delay
- tWR: Write recovery time

5.1.1.1 tCL (CAS Latency)

The number of memory clocks it takes a DRAM to return data after the read CAS_L is asserted depends on the memory clock frequency. The value that BIOS programs into the memory controller is a function of the target clock frequency. The target clock frequency is determined from the supported CAS latencies at given clock frequencies of each DIMM. A suggested algorithm is as follows:

1. Determine all CAS latencies supported by each installed DIMM, defined in SPD byte 18. One bit corresponds to each supported CAS latency. SPD byte 18 specifies CAS latencies with which devices on the DIMM can reliably operate. BIOS should not assume that a device can operate with either slower or faster CAS latency than those specified by the SPD. The DRAM controller is designed to support CAS latencies 2, 3,4, 5 and 6.
2. Determine the maximum clock frequency at each supported CAS latency for each DIMM. The minimum cycle time for the highest, the second highest, and the third highest supported CAS latencies are defined in SPD byte 9, 23, and 25, respectively. The minimum cycle time has 1/10ns granularity.
3. Determine the best CAS latency and clock frequency combination. Find the highest clock frequency supported by the slowest DIMM and determine the CAS latency at that operating frequency. It is necessary to choose the highest CAS latency supported by all the DIMMs at the target frequency.

5.1.1.2 tRCD (RAS-to-CAS Delay)

This parameter is defined in SPD byte 29 and it has 1/4ns granularity. BIOS should read and convert this value into a number of DRAM clocks using the target frequency obtained in "tCL (CAS Latency)" on page 196.

5.1.1.3 tRAS (Active-to-Precharge Delay)

This parameter is defined in SPD byte 30 and it has 1-ns granularity. BIOS should read and convert this value into a number of DRAM clocks using the target frequency obtained in "tCL (CAS Latency)" on page 196.

5.1.1.4 tRP (Precharge Command Period)

This parameter is defined in SPD byte 27 and it has 1/4-ns granularity. BIOS should read and convert this value into a number of DRAM clocks using the target frequency obtained in "tCL (CAS Latency)" on page 196.

5.1.1.5 tRC (Active-to-Active/Auto-Refresh Command Period)

This parameter is defined in SPD bytes 40 and 41 and it has 1/4-ns granularity. BIOS should read and convert this value into a number of DRAM clocks using the target frequency obtained in "tCL (CAS Latency)" on page 196.

5.1.1.6 tRRD (Active-to-Active of a Different Bank)

This parameter is defined in SPD byte 28 and it has 1/4-ns granularity. BIOS should read this value and convert into a number of DRAM clocks using the target frequency obtained in "tCL (CAS Latency)" on page 196.

5.1.1.7 tRFC (Auto-Refresh-to-Active/Auto-Refresh Command Period)

This parameter is defined in SPD bytes 40 and 42 and it has 1/4-ns granularity. BIOS should read and convert this value into a number of DRAM clocks using the target frequency obtained in "tCL (CAS Latency)" on page 196.

5.1.1.8 tREF (Refresh Rate)

This parameter is defined in SPD byte 12.

5.1.1.9 tRTP (Internal Read to Precharge Command Delay)

This parameter is defined in SPD byte 38 and it has 1/4-ns granularity. BIOS should read and convert this value into a number of DRAM clocks using the target frequency obtained in "tCL (CAS Latency)" on page 196. After the value is converted, the BIOS should round the value up and use this to program Trtp (Function 2, Offset 88h) using the JEDEC guidelines for calculating the controller command bus separation.Note: BIOS may use the recommended value for Trtp in Section 4.5.10 on page 112 and skip the calculation, since tRTP is specified in the DRAM data sheets for all supported speed grades.

5.1.1.10 tWR (Write Recovery)

This parameter is defined in SPD byte 36 and it has 1/4-ns granularity. BIOS should read and convert this value into a number of DRAM clocks using the target frequency obtained in "tCL (CAS Latency)" on page 196.

5.1.1.11 tWTR (Internal Write to Read Command Delay)

This parameter is defined in SPD byte 37 and it has 1/4-ns granularity. BIOS should read and convert this value into a number of DRAM clocks using the target frequency obtained in "tCL (CAS Latency)" on page 196.

5.1.1.12 Registered or Unbuffered DIMMs

SPD byte 20, bits 0 and 4 indicates whether the DIMM registers the address and commands or not. If either of these bits is set the DIMM registers the address and commands. Refer to the processor data sheet to determine the type of DIMMs supported. Systems with mixed unbuffered and registered DIMMs are not supported.

5.1.1.13 Four Rank SO-DIMMs

SPD byte 5, bits 2–0 indicate the number of ranks present on the DIMM. If these bits are 011b and SPD byte 20, bits 6–0 are 0001000b the DIMM is a four rank SO-DIMM.

5.1.1.14 Four Rank RDIMMs

SPD byte 5, bits 2–0 indicate the number of ranks present on the DIMM. If these bits are 011b and SPD byte 20, bits 6–0 are 000001b or 010000b the DIMM is a four rank RDIMM.

5.1.1.15 DIMM Chip Select Density

The size of a DIMM module chip-select range can be obtained directly from the SPD ROM.

Byte 31 indicates the size of the chip-select range or ranges on the DIMM (from 128 Mbyte to 8 Gbyte). See the SPD ROM specification for the encoding.

Note: *If the DIMM uses more than one chip select, then each chip-select range is the same size.*

5.1.1.16 DIMM ECC Enable

SPD byte 11 indicates whether the DIMM supports ECC bits. Bit 1 indicates that ECC is supported for all chip-select banks on the DIMM.

5.1.1.17 DIMM Address/Command Parity Enable

SPD byte 11 indicates whether the DIMM supports Address/Command parity. Bit 2 indicates that Address/Command parity is supported for all chip-select banks on the DIMM.

5.1.1.18 x4 DIMMs

The DRAM device width is largely inconsequential to the controller except when they are x4 devices. The width of the devices on the DIMM is determined by the value of SPD byte 13. If the DIMM implements two chip-select banks, then each chip-select bank uses devices of the same width. See the SPD ROM specification for the encoding.

5.1.1.19 DIMM ODT

SPD byte 22, bit 1 indicates support for 50 Ohm ODT. All DIMMs must support 75 Ohms and 150 Ohms. The following tables specify the recommended ODT settings for both the processor and the DIMMs on a per-channel basis.

Table 38. RDIMM and Unbuffered DIMM ODT Settings

Number of DIMMS	Processor ODT	DIMM ODT
1 at 200-333 MHz	75 Ohms	75 Ohms

Table 38. RDIMM and Unbuffered DIMM ODT Settings

Number of DIMMS	Processor ODT	DIMM ODT
2 or more at 200-333 MHz	150 Ohms	75 Ohms
1 at 400 MHz	75 Ohms	75 Ohms
2 at 400 MHz	150 Ohms	50 Ohms
3 or more at 400 MHz	150 Ohms	75 Ohms
Note: DIMMs on channel A and B must have same DIMM ODT value. The DIMM ODT recommendation for a system with 3 total unbuffered DIMMs at 400 MHz is 50 Ohms.		

Table 39. SO-DIMM ODT Settings

MEMCLK Frequency	Processor ODT	DIMM ODT
200-333 MHz	150 Ohms	150 Ohms
400 MHz	150 Ohms	150 Ohms

5.1.2 Non-SPD ROM-Based Configuration

Many other bit fields are also required by the DRAM controller configuration registers, but these values cannot be obtained from the SPD ROM. In many cases, these values are hardcoded into the BIOS, but in others they are functions of other bit values. This section describes how BIOS programs each non-SPD related field that is not hardcoded.

5.1.2.1 TrwtTO (Read-to-Write Turnaround for Data, DQS Contention)

This timing parameter ensures read-to-write data-bus turnaround. The setting for this parameter is a function of the asynchronous round-trip loop delay. This parameter should be programmed to 010b for all DIMM types and loadings.

5.1.2.2 Twrrd (Write to Read DIMM Termination Turn-around)

This timing parameter accounts for termination timing when a write is followed by a read to a different DIMM. The setting for this parameter is a function of the asynchronous round-trip loop delay. This parameter should be programmed to 00b for all DIMM types and loadings.

5.1.2.3 Trdrd (Read to Read Timing)

This timing parameter accounts for turn-around and termination timing when a read is followed by a read to a different chip select. The setting for this parameter is a function of the asynchronous round-trip loop delay. This parameter should be programmed to 00b for all DIMM types and loadings.

5.1.2.4 Twrwr (Write to Write Timing)

This timing parameter accounts for turn-around timing when a write is followed by a write to a different DIMM. This parameter should be programmed to 01b for all DIMM types and loadings.

5.1.2.5 Maximum Asynchronous Latency (MaxAsyncLat)

The maximum asynchronous latency is the maximum round-trip latency in the system from the processor to the DRAM devices and back. MaxAsyncLat is the sum of two components:

1. The worst case delay setting for DqsRcvEnDelay
2. A constant of 2ns associated with the delay internal to the processor.

If the sum of these two components is less than 4ns, MaxAsyncLat should be programmed to 4ns.

5.1.2.6 tFAW (Four Bank Activate Window)

No more than 4 banks may be activated in a rolling tFAW window. tFAW can be converted to clocks by dividing tFAW(ns) by tCK(ns) and rounding up to next integer value. As an example of the rolling window, if (tFAW/tCK) rounds up to 10 clocks, and an activate command is issued in clock N, no more than three further activate commands may be issued in clock N+1 through N+9.

Table 40. Four Bank Activate Window Values

Page Size	400 MHz	333 MHz	266 MHz	200 MHz
1K	14 Clocks	13 Clocks	10 Clocks	8 Clocks
2K	18 Clocks	17 Clocks	14 Clocks	10 Clocks

5.1.2.7 Pad Receive FIFO Delay

The RdPadRcvFifoDly (Function 2, Offset 78h) specifies how long after the DQS receiver is enabled before the read data is read out of the pad receive FIFO by the DRAM controller.

Table 41. RdPadRcvFifoDly Values

DRAM Frequency	Initial Value
400 MHz	4
333 MHz	4
266 MHz	4
200 MHz	4

5.1.2.8 Read Pointer Initialization (RdPtrInit)

The RdPtrInit (Function 2, Offset 78h) specifies the initial value of the DRAM FIFO read pointer. This field should be programmed to 6h for all DRAM frequencies.

5.1.2.9 Address Timing

The DRAM controller provides controls for programming the setup on the address pins, the CS/ODT pins and the CKE pins. Table 42, Table 43, Table 44, Table 45, Table 46, and Table 47 document the address timing settings on a per channel basis. The DIMMs on each channel are numbered from 1 to n where DIMM1 is the DIMM closest to the processor on that channel and DIMMn is the DIMM farthest from the processor on that channel. DIMMs must be populated from farthest slot to closest slot to the processor on a per channel basis. Populations that are not shown in these tables are not supported.

Table 42. SO-DIMM Address Timings and Drive Strengths for ASB1 Package

DRAM Speed[1]	DIMM[2]	Timing Mode	Address Timing Control Register	Output Driver Compensation Control Register[3]
DDR2-400	SRx16	1T	002F_2F00h	X011_1222h
	SRx8			
	DRx16			
	DRx8			
DDR2-533	SRx16	2T	003A_3900h	X011_1222h
	SRx8	1T	002F_2F00h	
	DRx16			
	DRx8	1T	002C_2C00h	
DDR2-667	SRx16	2T	003B_3400h	X011_1222h
	SRx8	1T	002A_2A00h	X011_1222h
	DRx16			
	DRx8	2T	0000_2800h	X011_1222h

1. *BIOS should program MemClkFreq (Function 2 Offset 94h) to 0h for all 1GHz CPUs.*
2. *SRx16 = Single Rank x16 DIMM*
   SRx8 = Single Rank x8 DIMM
   DRx16 = Dual Rank x16 DIMM
   DRx8 = Dual Rank x8 DIMM
3. *See Table 39 on page 199 for ProcOdt settings.*

Table 43. Unbuffered DIMM Address Timings and Drive Strengths for ASB1 Package

DRAM Speed	DIMM[1]	Timing Mode	Address Timing Control Register	Output Driver Compensation Control Register[2]
DDR2-400	Any	1T	002F_2F00h	X011_1222h
DDR2-533	SRx16	1T	002B_2F00h	X011_1222h
	DRx8			
	SRx8	1T	002F_2F00h	X011_1222h
DDR2-667	Any	1T	0020_2220h	X011_1222h

1. *SRx16 = Single Rank x16 DIMM*
   SRx8 = Single Rank x8 DIMM
   DRx8 = Dual Rank x8 DIMM
2. *See Table 38 on page 198 for ProcOdt settings.*

Table 44. Unbuffered DIMM Address Timings and Drive Strengths for AM2 Package

DRAM Speed	DIMM1[1]	DIMM2[1]	Timing Mode	Address Timing Control Register	Output Driver Compensation Control Register[2]
DDR2-400	-	Any	1T	002F_2F00h	X011_1222h
DDR2-400	Any	Any	2T	002F_2F00h	X011_1322h
DDR2-533	-	SRx16	1T	002B_2F00h	X011_1222h
	-	DRx8			
DDR2-533	-	SRx8	1T	002F_2F00h	X011_1222h
DDR2-533	SRx16	SRx16	2T	002F_2F00h	X011_1322h
	SRx16	SRx8			
	SRx8	SRx16			
DDR2-533	SRx8	SRx8	2T	0000_2F00h	X011_1322h
DDR2-533	DRx8	DRx8	2T	0034_2F00h	X011_1322h
DDR2-533	DRx8	SRx16	2T	0038_2F00h	X011_1322h
	SRx16	DRx8			
DDR2-533	DRx8	SRx8	2T	0037_2F00h	X011_1322h
	SRx8	DRx8			
DDR2-667	-	Any	1T	0020_2220h	X011_1222h
DDR2-667	SRx16	SRx16	2T	0020_2220h	X011_1322h
	SRx16	SRx8			
	SRx8	SRx16			
DDR2-667	SRx8	SRx8	2T	0030_2220h	X011_1322h
DDR2-667	DRx8	DRx8	2T	002B_2220h	X011_1322h
DDR2-667	DRx8	SRx16	2T	002C_2220h	X011_1322h
	SRx16	DRx8			
DDR2-667	DRx8	SRx8	2T	002A_2220h	X011_1322h
	SRx8	DRx8			
DDR2-800	-	Any	2T	0020_2520h	X011_3222h
DDR2-800	Any	Any	2T	0020_2520h	X011_3322h

1. *SRx16 = Single Rank x16 DIMM*
   SRx8 = Single Rank x8 DIMM
   DRx16 = Dual Rank x16 DIMM
   DRx8 = Dual Rank x8 DIMM
2. *See Table 38 on page 198 for ProcOdt settings.*

Table 45. SO-DIMM Address Timings and Drive Strengths for S1g1 Package

DRAM Speed	DIMM[1]	Timing Mode	Address Timing Control Register	Output Driver Compensation Control Register[2]
DDR2-400	SRx16	1T	002F_2F00h	X011_1222h
DDR2-400	SRx8	1T	002F_2F00h	X011_1222h
	DRx16			
	DRx8			
DDR2-533	SRx16	1T	002F_2F00h	X011_1222h
	SRx8			
	DRx16			
DDR2-533	DRx8	1T	002C_2C00h	X011_1222h
DDR2-667	SRx16	1T	0027_2700h	X011_1222h
DDR2-667	SRx8	1T	002A_2A00h	X011_1222h
	DRx16			
DDR2-667	DRx8	2T	0000_2800h	X011_1222h
DDR2-800	SRx16	1T	0029_2900h	X011_1222h
DDR2-800	SRx8	1T	002A_2A00h	X011_1222h
	DRx16			
DDR2-800	DRx8	2T	0000_2A00h	X011_1222h

1. *SRx16 = Single Rank x16 DIMM*
   SRx8 = Single Rank x8 DIMM
   DRx16 = Dual Rank x16 DIMM
   DRx8 = Dual Rank x8 DIMM
2. *See Table 39 on page 199 for ProcOdt settings.*

Table 46. Registered DIMM Address Timings and Drive Strengths for F(1207) Package 4-DIMM System

DRAM Speed	DIMM1[1]	DIMM2[1]	Timing Mode	Address Timing Control Register	Output Driver Compensation Control Register[2]
DDR2-400	-	SR or DR	1T	002F_0000h	X011_1222h
DDR2-400	-	QR	1T	002F_2700h	X011_1222h

1. *SR = Single Rank DIMM*
   DR = Dual Rank DIMM
   QR = Quad Rank DIMM
   any = SR, DR, or QR
2. *See Table 38 on page 198 for ProcOdt settings.*
3. *Only SR and DR supported.*

Table 46. Registered DIMM Address Timings and Drive Strengths for F(1207) Package 4-DIMM System

DRAM Speed	DIMM1[1]	DIMM2[1]	Timing Mode	Address Timing Control Register	Output Driver Compensation Control Register[2]
DDR2-400	SR or DR	SR or DR	1T	002F_0000h	X011_1222h
DDR2-400	QR	QR	1T	002F_2700h	X011_1222h
DDR2-533	-	any	1T	002F_2700h	X011_1222h
DDR2-533	any	any	1T	002F_2700h	X011_1222h
DDR2-667[3]	-	SR	1T	002F_0000h	X011_1222h
DDR2-667[3]	-	DR	1T	002F_2F00h	X011_1222h
DDR2-667[3]	SR	SR	1T	002F_0000h	X011_1222h
DDR2-667[3]	SR	DR	1T	002F_2F00h	X011_1222h
DDR2-667[3]	DR	SR	1T	002F_2F00h	X011_1222h
DDR2-667[3]	DR	DR	1T	002F_2F00h	X011_1222h

1. *SR = Single Rank DIMM*
   DR = Dual Rank DIMM
   QR = Quad Rank DIMM
   any = SR, DR, or QR
2. *See Table 38 on page 198 for ProcOdt settings.*
3. *Only SR and DR supported.*

Table 47. Registered DIMM Address Timings and Drive Strengths for F(1207) Package 8-DIMM System

DRAM Speed	DIMM1[1]	DIMM2[1]	DIMM3[1]	DIMM4[1]	Timing Mode	Address Timing Control Register	Output Driver Compensation Control Register[2]
DDR2-400	-	-	-	any	1T	002F_0000h	X011_1222h
DDR2-400	-	-	any	any	1T	002F_0000h	X011_1222h
DDR2-400	-	any	any	any	1T	002F_0000h	X011_1222h
DDR2-400	any	any	any	any	1T	002F_0000h	X011_1222h
DDR2-533	-	-	-	SR	1T	0032_0000h	X011_1222h
DDR2-533	-	-	-	DR	1T	002F_0000h	X011_1222h
DDR2-533	-	-	any	any	1T	0032_0000h	X011_1222h
DDR2-533	-	any	any	any	1T	0032_0000h	X011_1222h
DDR2-533	any	any	any	any	1T	0032_0000h	X011_1222h

1. *any = SR or DR.*
2. *See Table 38 on page 198 for ProcOdt settings.*

Table 47. Registered DIMM Address Timings and Drive Strengths for F(1207) Package 8-DIMM System

DRAM Speed	DIMM1[1]	DIMM2[1]	DIMM3[1]	DIMM4[1]	Timing Mode	Address Timing Control Register	Output Driver Compensation Control Register[2]
DDR2-667	-	-	-	SR	1T	0000_0000h	X011_1222h
DDR2-667	-	-	-	DRx8	1T	002F_0000h	X011_1222h
DDR2-667	-	-	-	DRx4	1T	002F_2F00h	X011_1222h
DDR2-667	-	-	SR or DRx8	SR or DRx8	1T	002F_0000h	X011_1222h
DDR2-667	-	-	any	DRx4	1T	002F_2F00h	X011_1222h
DDR2-667	-	-	DRx4	any	1T	002F_2F00h	X011_1222h

1. *any = SR or DR.*
2. *See Table 38 on page 198 for ProcOdt settings.*

5.1.3 Mismatched DIMM Support

The Mod64BitMux bit (Function 2: Offset A0h), can be used to configure the 128-bit interface as two 64-bit interfaces for AM2 and S1g1 packages. This mode allows mismatched DIMMs to be installed in the upper and lower DIMM slots without reducing any of the DRAM capacity. When operating in this mode the DRAM interface never operates as a 128-bit interface and the two 64-bit interfaces are not accessed simultaneously.

5.1.4 DRAM Initialization

There are two mechanisms that BIOS can use to initialize DRAM hardware controlled initialization and BIOS controlled initialization. Hardware controlled initialization is initiated by BIOS, but hardware controls the initialization sequence. BIOS controlled initialization allows the BIOS to control the initialization sequence.

5.1.4.1 Hardware Controlled DRAM Initialization

Some DRAM configuration fields must be programmed in a specific order. BIOS must set MemClkFreqVal (Function 2 Offset 94h) and wait a minimum of 20 MEMCLKs before setting InitDram (Function 2 Offset 90h). BIOS must set MemClkFreqVal (Function 2 Offset 94h) and wait a minimum of 10us for unbuffered DIMMs and 100us for registered Dimms before setting ExitSelfRef (Function 2 Offset 90h). BIOS must set InitDram last to start DRAM initialization after a cold reset not associated with suspend to RAM mode. BIOS must set ExitSelfRef last to exit self-refresh mode after a cold reset associated with suspend to RAM mode. BIOS should not assert LDTSTOP_L to change HyperTransport™ link width and frequency while DramConfigRegLo[0] or DramConfigRegLo[1] are set.

5.1.4.2 BIOS Controlled DRAM Initialization

BIOS can directly control the DRAM initialization sequence using the DRAM Initialization register (Function 2 Offset 7Ch). BIOS must set MemClkFreqVal (Function 2 Offset 94h) and wait a minimum of 200us before setting EnDramInit (Function 2 Offset 7Ch). BIOS must set the EnDramInit bit last to start DRAM initialization after a cold reset not associated with suspend to RAM mode. BIOS should then complete the initialization sequence specified in the JEDEC DDR2 specification. For registered DIMMs, BIOS should follow the recommendations for reset usage in the JEDEC RDIMM specification during the initialization sequence. After completing the initialization sequence, BIOS must clear the EnDramInit (Function 2 Offset 7Ch) bit. BIOS should not assert LDTSTOP_L to change HyperTransport™ link width and frequency while EnDramInit (Function 2 Offset 7Ch) is set.

5.1.5 DRAM Training

DRAM training is used to determine when to enable the DQS receivers on reads and to position the DQS strobe in the center of the data eye for both reads and writes. DRAM training must be performed by BIOS after initializing the DRAM controller to ensure proper DRAM operation. Prior to performing DRAM training, the BIOS must transition the processor frequency to 2 GHz and the processor voltage to MaxVID - RVO. If the maximum frequency of the processor is less than 2 GHz, the BIOS must transition the processor to the maximum operating frequency and processor voltage to MaxVID - RVO before performing DRAM training. After DRAM training is complete, the BIOS should transition server and desktop processors the maximum P-state and mobile parts to the minimum P-state.

5.1.5.1 DQS Receiver Enable Training

DQS receiver enable training determines when to enable the DQS receiver for each chip select pair in use. DQS receiver enable training is done in two phases. The first phase is used to detect when the DIMM starts driving the read preamble. The second phase is used to determine effective width of the read preamble. The first phase of DQS receiver enable training is performed using the following procedure:

- Program the DQS Write Timing Control Registers (Function 2: Offset 9Ch, indexes 01h-03h and 21h-23h) to 00h for all bytes.
- Program the Read DQS Timing Control Registers (Function 2: Offset 9Ch, indexes 05h-07h and 25h-27h) to 2Fh for all bytes.
- Set the DqsRcvEnTrain bit.
- Select two non-consecutive cacheline aligned test addresses on each chip select present. If BASE is the base address of the target chip select, then assigning the first test address to BASE and the second test address to BASE+2MB, targets two non-consecutive addresses on the same chip select regardless of the logical DIMM width, DRAM device configuration, and DRAM bank interleaving mode.

- Write a cache line where each byte is 55h to the first test address for each rank.
- Write a cache line where each byte is AAh to the second test address for each rank.
- For each channel
  - For each rank
    - For each DQS receiver enable setting
      - Write the current DQS receiver enable setting to the DQS Receiver Enable Timing Control Register for the current rank.
      - Program the MaxAsyncLat field with the current DQS receiver enable setting plus 2ns. See Section 5.1.2.5 on page 200 for further information on MaxAsyncLat requirements.
      - Read first test address for the rank and compare the first data beat (either 64 or 128 bit depending on the DRAM interface width) with the value written.
      - Reset the read pointer in the DRAM controller FIFO by writing the current DQS receiver enable setting to the DQS Receiver Enable Timing Control Register.
      - Read second test address for the rank and compare the first data beat with the value written.
      - Mark the DQS receiver enable setting as a pass if both reads passed.
- For each chip select pair
  - Program the DQS Receiver Enable Timing Control Register for the chip select pair with the first DQS receiver enable setting that passes for both ranks plus 300ps.
  - Save the first DQS receiver enable setting that passes for both ranks.
- Program the MaxAsyncLat field with the largest DQS receiver enable setting plus 2ns. See Section 5.1.2.5 on page 200 for further information on MaxAsyncLat requirements.
- Clear the DqsRcvEnTrain bit.

The second phase of DQS receiver enable training is performed after DQS position training is complete using the following procedure:

- Select two test addresses on each chip select present.
- Program the starting DQS Receiver Enable Timing Control Register for the chip select pair with the first DQS receiver enable setting that passes for both ranks in phase one plus 300ps.
- Write a cache line with the following data pattern to the first test address for each rank:

  1234_5678_8765_4321h
  2345_6789_9876_5432h
  5938_5824_3049_6724h
  2449_0795_9993_8733h
  4038_5642_3846_5245h
  2943_2163_0506_7894h

1234_9045_9872_3467h
1238_7634_3458_7623h

- Write a cache line with the following data pattern to the second test address for each rank:

  1234_5678_8765_4321h
  2345_6789_9876_5432h
  5938_5824_3049_6724h
  2449_0795_9993_8733h
  4038_5642_3846_5245h
  2943_2163_0506_7894h
  1234_9045_9872_3467h
  1238_7634_3458_7623h

- For each channel
  - For each rank
    - For each DQS receiver enable setting
      - Write the current DQS receiver enable setting to the DQS Receiver Enable Timing Control Register for the current rank.
      - Program the MaxAsyncLat field with the current DQS receiver enable setting plus 2ns. See Section 5.1.2.5 on page 200 for further information on MaxAsyncLat requirements.
      - Read first test address for the rank and compare the data read with the expected value.
      - Reset the read pointer in the DRAM controller FIFO by writing the current DQS receiver enable setting to the DQS Receiver Enable Timing Control Register.
      - Read second test address for the rank and compare the data read with the expected value.
      - Mark the DQS receiver enable setting as a pass if all three reads passed.
- For each chip select pair
  - Center the DQS receiver enable setting between the first DQS receiver enable setting that passed the first phase of training for both ranks and the last DQS receiver enable setting that passed the second phase of training for both ranks.
- Program the MaxAsyncLat field with the largest DQS Receiver Enable setting plus 2ns. See Section 5.1.2.5 on page 200 for further information on MaxAsyncLat requirements.

5.1.5.2 DQS Position Training

DQS position training is used to place the DQS strobe in the center of the data eye. Determining the correct DQS position for both reads and writes must be performed using a two dimensional search of the read and write position settings.

- Select three test addresses on each rank present.

- For each channel
  - For each byte
    - For each rank
      - For each write DQS position (Write DQS Position Loop)
        - Write the current write DQS position to the DQS Write Timing Control Register byte for the current byte.
        - Write the DRAM training pattern to the first test address for the rank.
        - For each read DQS position
          - Write the current read DQS position to the Read DQS Timing Control Register byte for the current byte.
          - Read the DRAM training pattern from the first test address three times.
          - If the training pattern is read correctly all three times mark read position for the byte as a pass.
        - If the read position contains three or more consecutive passing positions exit write DQS position loop.
      - Write the Read DQS Timing Control Register byte for the current byte with the middle position of the passing region.
      - For each write DQS position
        - Write the current write DQS position to the DQS Write Timing Control Register byte for the current byte.
        - Write 0's to the three test addresses for the rank.
        - Write the DRAM training pattern to the three test addresses for the rank.
        - Read the DRAM training pattern from the three test addresses.
        - If the training pattern is read correctly from each test address mark read position for the byte as a pass.
    - Compare the passing regions for the current byte for each rank to determine a mutual passing region for all ranks.
    - Write the Read DQS Timing Control Register byte for the current byte with the middle position of the mutual passing region for reads.
    - Write the DQS Write Timing Control Register byte for the current byte with the middle position of the mutual passing region for writes.

5.2 DDR2 DRAM Configuration

This section shows in a quick reference format how each DRAM controller configuration register should be programmed.

BaseAddrReg[7:0][31:0] = See "DRAM CS Base Address Registers" on page 98.

BaseMaskReg[7:0][31:0] = See "DRAM CS Mask Registers" on page 103.

BankAddrReg[31:0] = See "DRAM Bank Address Mapping Register" on page 106.

DramTimingRegLo[31:24] = MemClkDis[7:0] = See "DRAM Timing Low Register" on page 112.

DramTimingRegLo[23:22] = tRRD[1:0] = See "tRRD (Active-to-Active of a Different Bank)" on page 197.

DramTimingRegLo[21:20] = tWR[1:0] = See "tWR (Write Recovery)" on page 197.

DramTimingRegLo[19:16] = tRC[3:0] = See "tRC (Active-to-Active/Auto-Refresh Command Period)" on page 196.

DramTimingRegLo[15:12] = tRAS[3:0] = See "tRAS (Active-to-Precharge Delay)" on page 196.

DramTimingRegLo[11] = tRTP= See "tRTP (Internal Read to Precharge Command Delay)" on page 197.

DramTimingRegLo[9:8] = tRP[1:0] = See "tRP (Precharge Command Period)" on page 196.

DramTimingRegLo[5:4] = tRCD[1:0] = See "tRCD (RAS-to-CAS Delay)" on page 196.

DramTimingRegLo[2:0] = tCL[2:0] = See "tCL (CAS Latency)" on page 196.

DramTimingRegHi[31:29] = tRFC3[3:0] = See "tRFC (Auto-Refresh-to-Active/Auto-Refresh Command Period)" on page 197.

DramTimingRegHi[28:26] = tRFC2[3:0] = See "tRFC (Auto-Refresh-to-Active/Auto-Refresh Command Period)" on page 197.

DramTimingRegHi[25:23] = tRFC1[3:0] = See "tRFC (Auto-Refresh-to-Active/Auto-Refresh Command Period)" on page 197.

DramTimingRegHi[22:20] = tRFC0[3:0] = See "tRFC (Auto-Refresh-to-Active/Auto-Refresh Command Period)" on page 197.

DramTimingRegHi[17:16] = tREF[1:0] = See "tREF (Refresh Rate)" on page 197.

DramTimingRegHi[15:14] = Trdrd[1:0] = See "Trdrd (Read to Read Timing)" on page 199.

DramTimingRegHi[13:12] = Twrwr[1:0] = See "Twrwr (Write to Write Timing)" on page 200.

DramTimingRegHi[11:10] = Twrrd[1:0] = See "Twrrd (Write to Read DIMM Termination Turn-around)" on page 199.

DramTimingRegHi[9:8] = tWTR[1:0] = See "tWTR (Internal Write to Read Command Delay)" on page 197.

DramTimingRegHi[0] = TrwtTO = See "TrwtTO (Read-to-Write Turnaround for Data, DQS Contention)" on page 199.

DramConfigRegLo[19] = DimmEccEn = See "DIMM ECC Enable,".

DramConfigRegLo[16] = UnBuffDimm = See "Registered or Unbuffered DIMMs" on page 197.

DramConfigRegLo[15:12] = x4DIMMs[3:0] = See "x4 DIMMs" on page 198.

DramConfigRegLo[11] = Width128 = 0h[1]

DramConfigRegLo[10] = BurstLength32 = 0h[2]

DramConfigRegLo[9] = SelRefRateEn = 0h

DramConfigRegLo[8] = ParEn = See "DIMM Address/Command Parity Enable" on page 198

DramConfigRegLo[7] = DramDrvWeak= 0

DramConfigRegLo[5:4] = DramTerm[1:0] = See "DIMM ODT" on page 198

DramConfigRegLo[1] = ExitSelfRef = 0h

DramConfigRegLo[0] = InitDram = 1h

DramConfigRegHi[31:28] = FourActWindow[3:0] = See "tFAW (Four Bank Activate Window)" on page 200.

DramConfigRegHi[27:24] = DqsBypassMax[3:0] = 4h

DramConfigRegHi[22] = BankSwizzleMode = 0h

DramConfigRegHi[20] = SlowAccessMode = 0h

DramConfigRegHi[16] = PowerDownMode = 0/1h[3]

DramConfigRegHi[15] = PowerDownEn = 1h

DramConfigRegHi[14] = DisDramInterface = 0h[4]

DramConfigRegHi[6:4] = MaxAsyncLat[2:0] = See "Maximum Asynchronous Latency (MaxAsyncLat)" on page 200.

DramConfigRegHi[3] = MemClkFreqVal = 1h

DramConfigRegHi[2:0] = MemClkFreq[2:0] = See "tCL (CAS Latency)" on page 196.

DramMisc[8:6] = ILD_lmt[2:0] = 3h

DramMisc[5] = DCC_EN = 1h

DramMisc[3:2] = RdWrQByp = 2h

ScrubControl[31:0] = DramScrubAddrHi[31:0] = DramScrubAddrLo[31:0] = 0000_0000h

Notes:

1. *This bit should be cleared for 64-bit interfaces and set for 128-bit interfaces. The value of this bit is a function of the system design and cannot be determined via the SPD ROM.*

2. *This bit should be set for 64-bit interfaces (when DramConfigRegLo[11] is cleared) on a platform with a graphics core (AGP or UMA) that issues 32-byte requests. This bit is ignored for 128-bit interfaces (when DramConfigRegLo[11] is set).*
3. *This bit should be set for mobile systems and cleared for server and desktop systems.*
4. *In systems that do not use the DRAM controller, it is preferable to disable DRAM input receivers and output drivers and to leave input receivers unconnected. This bit should be set in such systems.*

6 Machine Check Architecture

The AMD NPT Family 0Fh Processor machine check mechanism allows the processor to detect and report a variety of hardware (or machine) errors found when reading and writing data, probing, cache-line fills and writebacks. These include parity errors associated with caches and TLBs, ECC errors associated with caches and DRAM, as well as system bus errors associated with reading and writing to the external bus.

Software can enable the processor to report machine check errors through the machine check exception (See "#MC—Machine Check Exception" in *AMD64 Architecture Programmer's Manual*, Volume 2: System Programming). Most machine check exceptions do not allow reliable restarting of the interrupted programs. However, error conditions are logged in a set of model-specific registers (MSRs) that can be used by system software to determine the possible source of a hardware problem.

6.1 Determining Machine Check Support

The availability of machine check registers and support of the machine check exception is implementation dependent. System software executes the CPUID instruction to determine whether a processor implements these features. After CPUID is executed, the values of the machine check architecture (MCA) bit and the machine check exception (MCE) bit loaded in the EDX register indicate whether the processor implements the machine check registers and the machine check exception, respectively. See "Processor Feature Identification" in *AMD64 Architecture Programmer's Manual*, Volume 2: System Programming, and "CPUID" in *AMD64 Architecture Programmer's Manual*, Volume 3: General Purpose and System Instructions, for further information on the level of machine check support.

Once system software determines that the machine check registers are available, it must determine the extent of processor support for the machine check mechanism. This is accomplished by reading the machine check capabilities register (MCG_CAP). See "Machine Check Global Capabilities Register" in *AMD64 Architecture Programmer's Manual*, Volume 2: System Programming, for more information on the interpretation of the MCG_CAP contents.

6.2 Machine Check Errors

Machine check errors are either recoverable or irrecoverable. Recoverable errors are those that the processor can correct and, thus, do not raise the machine check exception (#MC). However, the error is still logged in the machine check MSRs and it is the responsibility of the system software to periodically poll the machine check MSRs to determine if recoverable errors have occurred. If a recoverable error has been logged in the machine check MSRs, a second recoverable error can overwrite it.

Irrecoverable, or fatal, machine check errors cannot be corrected by the processor. If machine check is enabled, they raise the machine check exception (#MC). If an irrecoverable error is logged in the machine check MSRs, a second recoverable or irrecoverable MCA error does not overwrite it but does set a bit that indicates overflow.

In the case of both recoverable and irrecoverable MCA errors, the contents of the machine check MSRs are maintained though a warm reset. This is helpful since in some cases it may not be possible to invoke the machine check handler due to the error and this allows the BIOS or other system boot software to recover and report the information associated with the error.

6.2.1 Sources of Machine Check Errors

The processor can detect errors from the following hardware blocks within the processor. Each block forms an error reporting bank for the purpose of reporting machine check errors.

- Data cache unit (DC)—Includes the cache structures that hold data and tags, the data TLBs, and the data cache probing logic.
- Instruction cache unit (IC)—Includes the instruction cache structures that hold instructions and tags, the instruction TLBs, and the instruction cache probing logic.
- Bus unit (BU)—Includes the system bus interface to the Northbridge and the level 2 cache.
- Load/store unit (LS)—Includes logic used to manage loads and stores.
- Northbridge unit (NB)—Includes the Northbridge and DRAM controller.

A scrubber is associated with the data cache in DC, the L2 cache tag array in bus unit, and the DRAM in the Northbridge. A scrubber is a hardware widget that periodically wakes up during idle cache cycles and inspects the next line of the array with which it is associated to look for errors. If it finds a single bit ECC error, the scrubber corrects the error and prevents a regular access from encountering the same error. This is important in the data cache, since all ECC errors encountered during regular operation in the data cache are fatal. In the bus unit and the Northbridge, this scrubber function also saves the additional time it would take to fix single-bit ECC errors when they are encountered during normal operation.

Even though 64-bits are read from the data cache during normal operation, a byte load uses one byte, a word load uses two bytes, and a double word load uses 4 bytes. If a data cache ECC error is detected in a byte that is not being used by a load, it is corrected like an error detected by the data cache scrubber. This feature is referred to as "piggyback scrubbing".

Table 48 on page 217 shows the various sources of machine check errors that can be encountered in the processor. ECC check on the linefill data for IC data is done by the DC unit. However, the ECC error is reported by IC which is the unit responsible for receiving the data. The data cache is protected by ECC; however single bit ECC errors encountered by normal load and store accesses are not corrected. When the data cache scrubber encounters a single-bit ECC error, it corrects it.

Table 48. Sources of Machine Check Errors

Unit	Error Source	Type of Check	Recoverable
DC	System line fill into the data cache	ECC	Yes—single bit error No—multiple bit error
	L2 cache line fill into the data cache		
	Cache data—data array	ECC	Yes—if single bit ECC error is detected by the scrubber No—any other case different than single bit ECC error detected by the scrubber
	Cache data—snoop tag array	Parity	No
	Cache data—tag array		No
	L1 Data TLB—physical and virtual arrays		No
	L2 Data TLB—physical and virtual arrays		No
IC	System line fill into the instruction cache	ECC	Yes—single bit error No—multiple bit error
	L2 cache line fill into the instruction cache		
	Instruction cache—data array	Parity	Yes[1]
	Instruction cache—tag array		Yes[1]
	Instruction cache—snoop tag array		No
	L1 Instruction TLB—physical and virtual arrays		Yes[1]
	L2 Instruction TLB—physical and virtual arrays		Yes[1]
	System address out of range		No, but detection is precise
BU	L2 cache data array	ECC	Yes—single bit error No—multiple bit error
	L2 cache—tag array (during scrubs)		
	L2 cache—tag array	Parity	No. Single and multiple bit errors in an L2 cache tag can also be detected, but not corrected.
	System Address Out-of-Range	Read Data	No
LS	System Address Out-of-Range	Read Data	No. Loads are detected precisely and stores are detected imprecisely.
NB	See "Machine Check Architecture (MCA) Registers" on page 140 for information on Northbridge machine check errors.		

Note: *Instruction cache lines are invalidated and refetched.*

6.3 Machine Check Architecture Registers

The processor architecture defines a set of model-specific registers (MSRs) to support the MCA mechanism. They are used to configure the MCA functions and provide a way for the hardware to report errors in a manner compatible with the machine check architecture. These registers include the global MCA registers used to set up machine checks and additional banks of MSRs for recording errors that are detected by the hardware blocks listed in "Sources of Machine Check Errors" on page 216. They can be read and written using the RDMSR and WRMSR instructions. The following is a complete list of MCA MSRs.

- Global status and control registers
  - Machine check capabilities MSR (MCG_CAP)
  - Processor status MSR (MCG_STATUS)
  - Exception reporting control MSR (MCG_CTL)
- Each error reporting bank (associated with a specific hardware block listed in "Sources of Machine Check Errors" on page 216) contains the following registers:
  - Error reporting control register (MC*i*_CTL)
  - Error reporting control register mask (MC*i*_CTL_MASK)
  - Error reporting status register (MC*i*_STATUS)
  - Error reporting address register (MC*i*_ADDR)
  - Machine check miscellaneous error information register (MC*i*_MISC)

The *i* in each register name corresponds to the number of a supported register bank. Each error-reporting register bank is associated with a specific processor unit (or group of processor units). The number of error-reporting register banks is implementation-specific.

Software reads the MCG_CAP register to determine the number of supported register banks. AMD NPT Family 0Fh Processors support five banks. The first error-reporting register (MC0_CTL) always starts with MSR address 400h, followed by MC0_STATUS (401h), MC0_ADDR (402h), and MC0_MISC (403h). Error-reporting-register MSR addresses are assigned sequentially through the remaining supported register banks. Using this information, software can access all error-reporting registers in an implementation-independent manner.

The global machine check registers as well as the generic form of each register in the error reporting banks are now described. The specifics of each error reporting bank are described in "Error Reporting Banks" on page 225.

The MCi_STATUS_WREN bit in MSR C001_0015h can be used to help debug Machine Check exception handlers. When the MCi_STATUS_WREN bit is set, privileged software can write non-zero values to the MCG_STATUS, MCi_STATUS, MCi_ADDR, and if implemented, MCi_MISC MSRs without generating exceptions, and then simulate a machine check using the INT 18 instruction. Setting a reserved bit in these MSRs does not generate an exception when this mode is enabled. However, setting a reserved bit may result in undefined behavior.

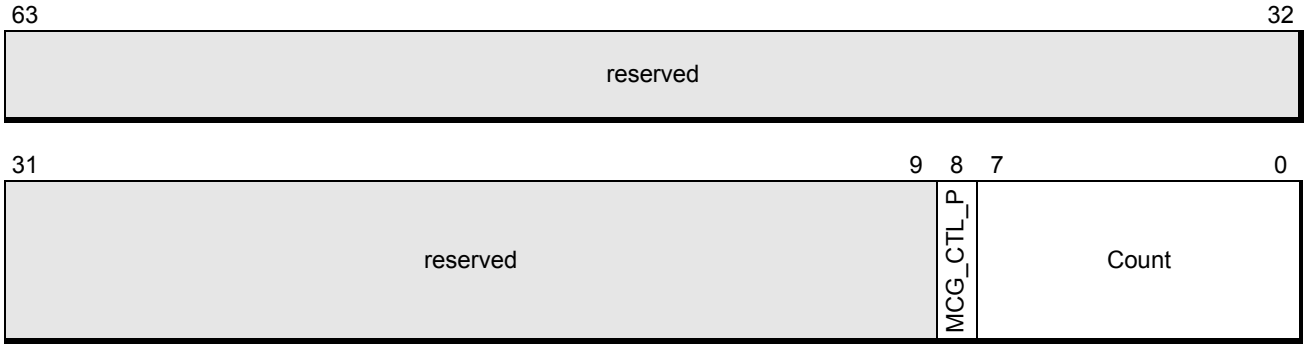
6.3.1 Global Machine Check Model-Specific Registers (MSRs)

The global MSRs supported by the machine check mechanism include the MCG_CAP, the MCG_STATUS and the MCG_CTL registers.

6.3.1.1 MCG_CAP—Global Machine Check Capabilities Register

This read-only register indicates the capabilities of the processor machine check architecture. Attempting to write to this register results in a #GP exception. See A*MD64 Architecture Programmer's Manual, Volume 2: System Programming*, for more details.

MCG_CAP Register **MSR 0179h**



Bit	Name	Function	R/W	Reset
63–9	reserved			0
8	MCG_CTL_P	MCG_CTL Register Present	R	1
7–0	Count	Count	R	05h

Field Descriptions

Count (Count)—Bits 7–0. Number of error-reporting banks supported by the processor implementation.

MCG_CTL Register Present (MCG_CTL_P)—Bit 8. Indicates if the MCG_CTL register is present. When the bit is set to 1, the register is supported. When the bit is cleared to 0, the register is unsupported.

6.3.1.2 MCG_STATUS—Global Machine Check Processor Status Register

This register contains basic information about the processor state after a machine check error is detected. See *AMD64 Architecture Programmer's Manual*, Volume 2: System Programming, for more details.

MCG_STATUS Register — MSR 017Ah

Bit	Name	Function	R/W	Reset
63–3	reserved			0
2	MCIP	Machine Check in Progress	R/W	0
1	EIPV	Error IP Valid Flag	R/W	0
0	RIPV	Restart IP Valid Flag	R/W	0

Field Descriptions

Restart IP Valid Flag (RIPV)—Bit 0. When set, indicates if program execution can be restarted at EIP pushed on stack.

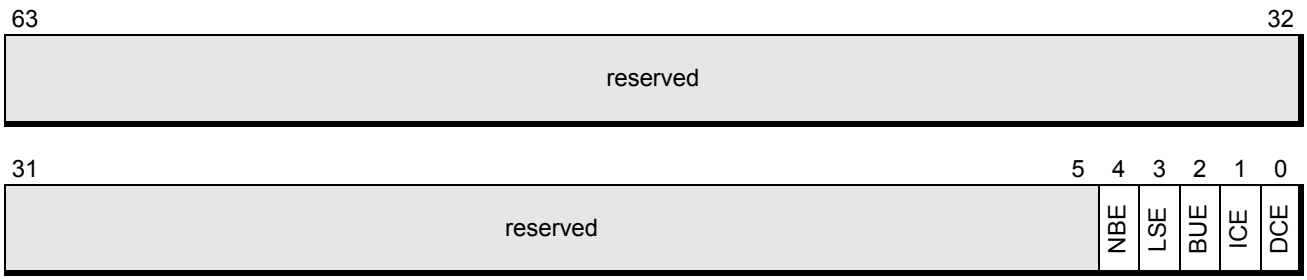
Error IP Valid Flag (EIPV)—Bit 1. When set, indicates if the EIP pushed on stack is that of the instruction which caused the detection of the machine check error.

Machine Check in Progress (MCIP)—Bit 2. When set, indicates that a machine check is in progress.

6.3.1.3 MCG_CTL—Global Machine Check Exception Reporting Control Register

This register contains a global bit for each error-checking unit in the processor. Each bit enables the reporting, through the MCA interface, of errors detected by that particular unit. See *AMD64 Architecture Programmer's Manual*, Volume 2: System Programming, for more details.

MCG_CTL Register — MSR 017Bh



Bit	Name	Function	R/W	Reset
63–5	reserved			0
4	NBE	NB Register Bank Enable	R/W	0
3	LSE	LS Register Bank Enable	R/W	0
2	BUE	BU Register Bank Enable	R/W	0
1	ICE	IC Register Bank Enable	R/W	0
0	DCE	DC Register Bank Enable	R/W	0

Field Descriptions

DC Register Bank Enable (DCE)—Bit 0. When set, indicates that the Data Cache register bank is enabled.

IC Register Bank Enable (ICE)—Bit 1. When set, indicates that the Instruction Cache register bank is enabled.

BU Register Bank Enable (BUE)—Bit 2. When set, indicates that the Bus Unit register bank is enabled.

LS Register Bank Enable (LSE)—Bit 3. When set, indicates that the Load/Store register bank is enabled.

NB Register Bank Enable (NBE)—Bit 4. When set, indicates that the Northbridge register bank is enabled.

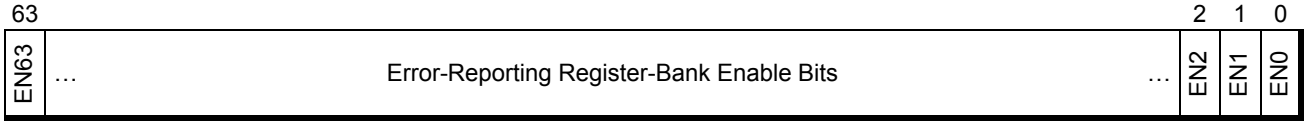
6.3.2 Error Reporting Bank Machine Check MSRs

The registers in each error reporting bank include MC*i*_CTL, MC*i*_CTL_MASK, MC*i*_STATUS and MC*i*_ADDR.

6.3.2.1 MC*i*_CTL—Machine Check Control Registers

The machine check control registers (MC*i*_CTL) contain an enable bit for each error source within an error-reporting register bank. Setting an enable bit to 1 enables error-reporting for the specific feature controlled by the bit, and clearing the bit to 0 disables error reporting for the feature. These registers should generally be set to either all zeros or all ones. As described in Section 5.4.2.2, each enable bit can be masked by the MC*i*_CTL_MASK register.

MC*i*_CTL Registers **MSRs 0400h, 0404h, 0408h, 040Ch, 0410h**



Bit	Name	Function	R/W	Reset
63–0	EN63–EN0	Enables	R/W	0

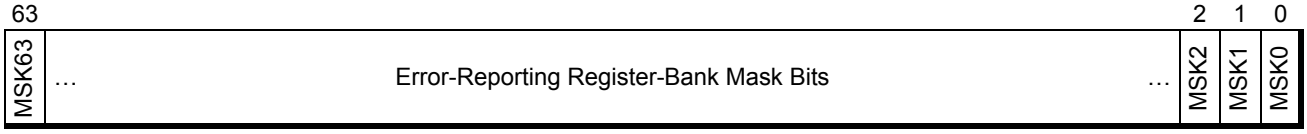
Field Descriptions

Enables (EN63–EN0)—Bits 63–0. If set, error reporting for the specific feature controlled by the bit is enabled. Not all these bits may be implemented in each bank.

6.3.2.2 MC*i*_CTL_MASK—Machine Check Control Mask Registers

The machine check control register masks (MC*i*_CTL_MASK) provide another level of control in enabling and disabling specific error reporting features. Each bit set to 1 in the MC*i*_CTL_MASK register inhibits the setting of the corresponding enable bit in the MC*i*_CTL register that it is associated with. This register is typically programmed by BIOS and not by the Kernel software.

MC*i*_CTL_MASK Registers **MSR C001_0044h, C001_0045h, C001_0046h, C001_0047h, C001_0048h**



Bit	Name	Function	R/W	Reset
63–0	MSK63–MSK0	Control Register Masks	R/W	0

Field Descriptions

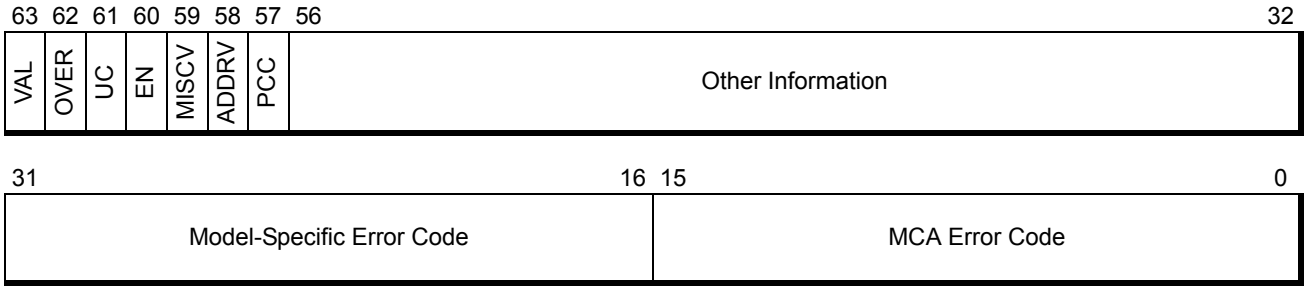
Control Register Masks (MSK63–MSK0)—Bits 63–0. If set, the corresponding bit in the MC*i*_CTL register is forced to 0 when written.

0 = Masking not enabled
1 = Masking enabled

6.3.2.3 MC*i*_STATUS—Machine Check Status Registers

The machine check status register (MC*i*_STATUS) describes the error detected by its unit. It is written by the processor and should be cleared to 0 by software; writing any other value to the register causes a general protection (GP#) exception. When a machine check error occurs, the processor loads an error code into bits [15:0] of the appropriate MCi_STATUS register. Errors are classified as either system bus, cache, or TLB errors. All the status registers in each bank use the same general logging format shown below.

MC*i*_STATUS Registers — MSRs 0401h, 0405h, 0409h, 040Dh, 0411h



Bit	Name	Function	R/W	Reset
63	VAL	Valid	R/W	X
62	OVER	Status Register Overflow	R/W	X
61	UC	Uncorrected Error indication	R/W	X
60	EN	Error Condition Enabled	R/W	X
59	MISCV	Miscellaneous-Error Register Valid	R/W	X
58	ADDRV	Error-Address Register Valid	R/W	X
57	PCC	Processor-Context Corrupt	R/W	X
56–32		Other Information	R/W	X
31–16		Model-Specific Error Code	R/W	X
15–0		MCA Error Code	R/W	X

Field Descriptions

MCA Error Code—Bits 15–0. This field holds an error code when an error is detected.

Model-Specific Error Code—Bits 31–16. This field encodes model-specific information about the error.

Other Information—Bits 56–32. This field holds model-specific error information.

Processor-Context Corrupt (PCC)—Bit 57. If set to 1, this bit indicates that the state of the processor may be corrupted by the error condition. Reliable restarting might not be possible.

0 = Processor not corrupted
1 = Processor may be corrupted

Error-Address Register Valid (ADDRV)—Bit 58. If set to 1, this bit indicates that the address saved in the address register is the address where the error occurred.

0 = Address register not valid
1 = Address register valid

Miscellaneous-Error Register Valid (MISCV)—Bit 59. If set to 1, this bit indicates whether the Miscellaneous Error register contains valid information for this error.

Error Condition Enabled (EN)—Bit 60. If set to 1, this bit indicates that MCA error reporting is enabled for this error in the MCA Control register.

0 = Error checking not enabled
1 = Error checking enabled

Uncorrected Error indication (UC)—Bit 61. If set to 1, this bit indicates that the error was not corrected by hardware.

0 = Error corrected
1 = Error not corrected

Status Register Overflow (OVER)—Bit 62. Set to 1 if the unit detects an error but the valid bit of this register already set. Enabled errors are written over disabled errors, uncorrectable errors are written over correctable errors. Uncorrectable errors are not overwritten.

0 = No error overflow
1 = Error overflow

Valid (VAL)—Bit 63. If set to 1, this bit indicates that a valid error has been detected by the unit. This bit should be cleared to 0 by software after the register is read.

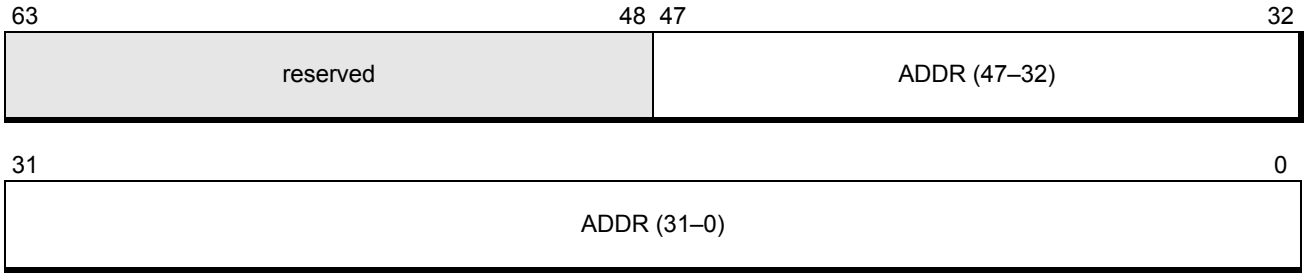
0 = No valid error detected
1 = Valid error detected

6.3.2.4 MC*i*_ADDR—Machine Check Address Registers

Each error-reporting register bank includes a machine check address register (MC*i*_ADDR) that the processor uses to report the instruction memory address or data memory address responsible for the machine check error. The contents of this register are valid only if the ADDRV bit in the corresponding MC*i*_STATUS register is set to 1.

The address can be up to 48 bits wide. In reality, not all address bits may be valid and the address field of the MC*i*_ADDR varies in width. The address field can hold either a virtual (linear) or physical address, depending on the type or error. Some error types cause the processor to load valid values into a subset of the address bits. The bit ranges depend on the values in the MC*i*_STATUS register, i.e., on the type of error detected by the unit, which can be determined by examining the MCA error code contained in the MC*i*_STATUS register.

MC*i*_ADDR Registers **MSRs 0402h, 0406h, 040Ah, 040Eh, 0412h**



Bit	Name	Function	R/W	Reset
63–48	reserved			0
47–0	ADDR	Address	R/W	0

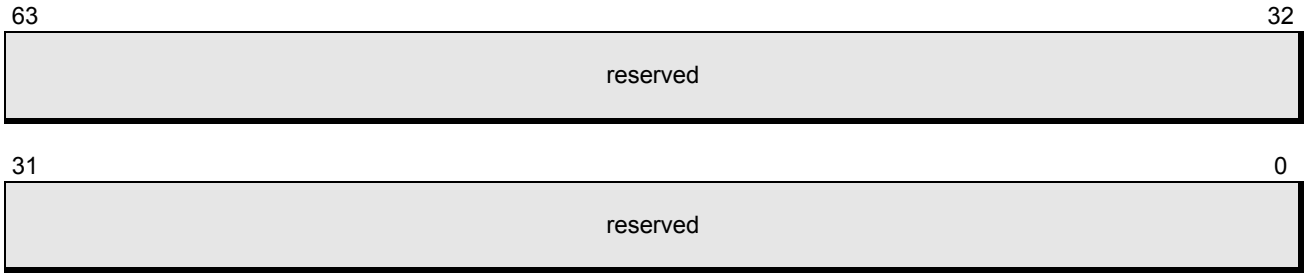
Field Descriptions

Address (ADDR)—Bits 47–0. Not all these bits may be valid. The valid bits depend on the type of error registered in the corresponding MC*i*_STATUS register.

6.3.2.5 MC*i*_MISC—Machine Check Miscellaneous Registers

Each error-reporting register bank includes a machine check address register (MC*i*_MISC) that the processor uses to report additional information about specific errors. MC0_MISC, MC1_MISC, MC2_MISC, and MC3_MISC are read only zero and provide no additional information. MC4_MISC is described in "MC4_MISC—DRAM Errors Threshold Register" on page 241.

MC*i*_MISC Registers **MSRs 0403h, 0407h, 040Bh, 040Fh, 0413h**



Bit	Name	Function	R/W	Reset
63–0	reserved			0

6.4 Error Reporting Banks

AMD NPT Family 0Fh Processors have five error-reporting banks—DC, IC, BU, LS, and NB. Each error reporting bank includes the following registers:

- Machine check control register (MC*i*_CTL).
- Error reporting control register mask (MC*i*_CTL_MASK).
- Machine check status register (MC*i*_STATUS).
- Machine check address register (MC*i*_ADDR).

The general format of these registers was described in "Error Reporting Bank Machine Check MSRs" on page 221. This section describes the specifics of each register as it relates to each error reporting bank.

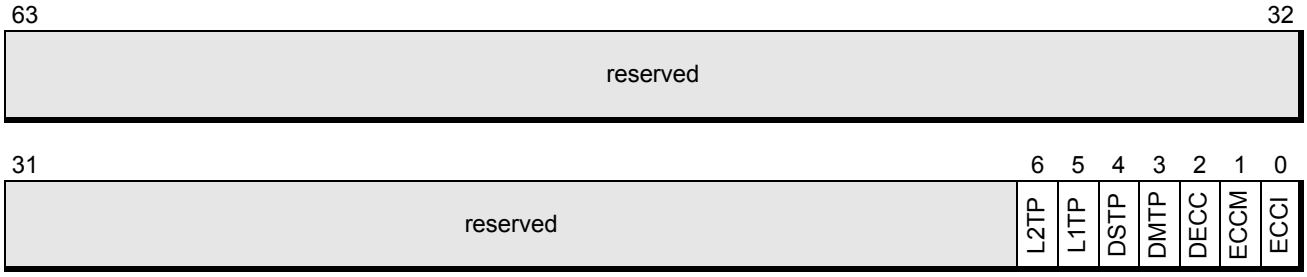
6.4.1 Data Cache (DC)

The data cache unit includes the level 1 data cache that holds data and tags, as well as two levels of TLBs and data cache probing logic.

6.4.1.1 MC0_CTL—DC Machine Check Control Register

This register enables the reporting, via the MCA interface, of a variety of errors detected by the Data Cache (DC) processor unit. For an error to be reported, both the global DC enable, MCG_CTL[DCE], and the corresponding local enable shown below must be set. If the local enable is off, the error is logged in MC0_STATUS, but not reported via the machine check exception. If the global DC enable is off, no error is logged or reported. BIOS is recommended to set all bits in this register to a value of 1, to accomodate some legacy operating systems which do not properly configure this register.

MC0_CTL Register **MSR 0400h**



Bit	Name	Function	R/W	Reset
63–7	reserved			0
6	L2TP	L2 TLB Parity Errors	R/W	0
5	L1TP	L1 TLB Parity Errors	R/W	0
4	DSTP	Snoop Tag Array Parity Errors	R/W	0
3	DMTP	Main Tag Array Parity Errors	R/W	0
2	DECC	Data Array ECC Errors	R/W	0
1	ECCM	Multi-bit ECC Data Errors	R/W	0
0	ECCI	Single-bit ECC Data Errors	R/W	0

Field Descriptions

Single-bit ECC Data Errors (ECCI)—Bit 0. Report single-bit ECC data errors during data cache line fills or TLB reloads from the internal L2 or the system.

Multi-bit ECC Data Errors (ECCM)—Bit 1. Report multi-bit ECC data errors during data cache line fills or TLB reloads from the internal L2 or the system.

Data Array ECC Errors (DECC)—Bit 2. Report data cache data array ECC errors.

Main Tag Array Parity Errors (DMTP)—Bit 3. Report data cache main tag array parity errors.

Snoop Tag Array Parity Errors (DSTP)—Bit 4. Report data cache snoop tag array parity errors.

L1 TLB Parity Errors (L1TP)—Bit 5. Report data cache L1 TLB parity errors.

L2 TLB Parity Errors (L2TP)—Bit 6. Report data cache L2 TLB parity errors.

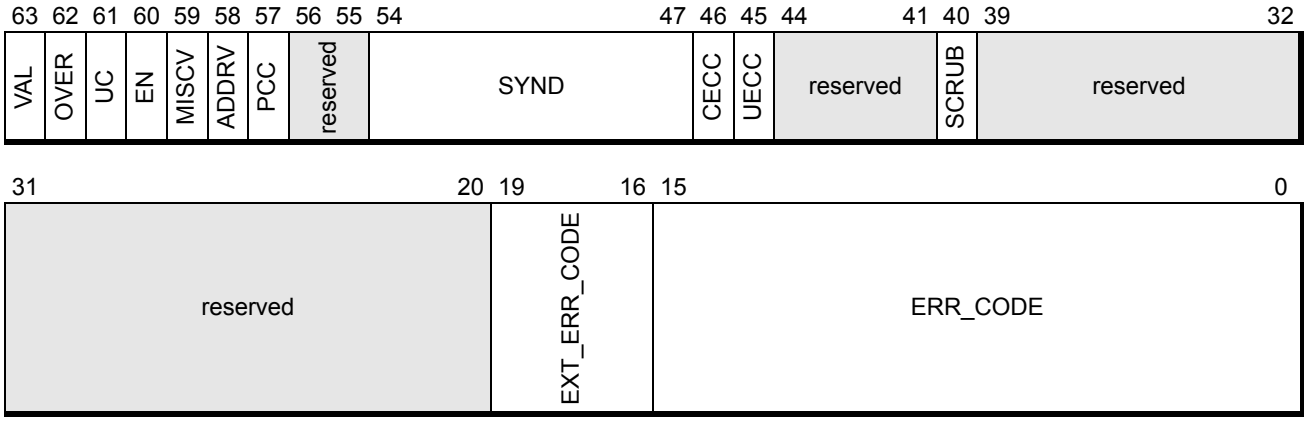
6.4.1.2 MC0_CTL_MASK—DC Machine Check Control Mask Register

The MC0_CTL_MASK register, at address MSR C001_0044h, can be used to mask the enabling of individual error reporting features in DC. Any bit set to 1 in this register inhibits writing 1s to the corresponding bits of the MC0_CTL register. See "MCi_CTL_MASK—Machine Check Control Mask Registers" on page 222 for a description of this register.

6.4.1.3 MC0_STATUS—DC Machine Check Status Register

The MC0_STATUS MSR describes the error that was detected by DC. The machine check mechanism writes the status register bits when an error is detected and sets the register valid bit (bit 63) to 1 to indicate that the status information is valid. This register is written even if error reporting for the detected error is not enabled. However, error reporting must be enabled for the error to result in a machine check exception. Software is responsible for clearing this register.

MC0_STATUS Register **MSR 0401h**



Bit	Name	Function	R/W	Reset
63	VAL	Valid.	R/W	X
62	OVER	Second error detected	R/W	X
61	UC	Error not corrected	R/W	X
60	EN	Error reporting enabled	R/W	X
59	MISCV	Additional info in MC*i*_MISC	R/W	X
58	ADDRV	Error address in MC*i*_ADDR	R/W	X
57	PCC	Processor state corrupted by error	R/W	X
56–55	reserved			0
54–47	SYND	ECC Syndrome (7–0)	R/W	X
46	CECC	Correctable ECC error	R/W	X

Bit	Name	Function	R/W	Reset
45	UECC	Uncorrectable ECC error	R/W	X
44–41	reserved		R/W	0
40	SCRUB	Error detected on a scrub	R/W	X
39–20	reserved			0
19–16	EXT_ERR_CODE	Extended error code	R/W	X
15–0	ERR_CODE	Error subsection	R/W	X

Field Descriptions

Error Subsection (ERR_CODE)—Bits 15–0. Indicates in which subsection the error was detected and what transaction initiated it. See Table 16 on page 148 for the format of the error code and Tables 17 through 22 on pages 148–149 for descriptions of the subfields.

Extended Error Code (EXT_ERR_CODE)—Bits 19–16. Contains an extended error code.

DC/IC:

0000b = TLB parity error in physical array
0001b = TLB parity error in virtual array (multi-match error)

BU:

0000b = Bus or cache data array error
0010b = Cache tag array error

LS: Reserved

Table 49. DC Error Codes

Error Type	Extended Error Code	Error Code						
		Type	PP	T	RRRR	II	LL	TT
System Line Fill	0000	BUS	SRC	0	DRD	MEM/IO	LG	-
L2 Cache Line Fill	0000	Memory	-	-	DRD	-	L2	Data
Data Load/ Store/ Victim/ Snoop	0000	Memory	-	-	DRD/DWR/Evict/Snoop	-	L1	Data
Data Scrub	0000	Memory	-	-	GEN	-	L1	Data
Tag Snoop/ Victim	0000	Memory	-	-	Snoop/ Evict	-	L1	Data
Tag Load/ Store	0000	Memory	-	-	DRD/DWR	-	L1	Data
L1 TLB	0000	TLB	-	-	-	-	L1	Data

Table 49. DC Error Codes

Error Type	Extended Error Code	Error Code						
		Type	PP	T	RRRR	II	LL	TT
L1 TLB Multimatch	0001	TLB	-	-	-	-	L1	Data
L2 TLB	0000	TLB	-	-	-	-	L2	Data
L2 TLB Multimatch	0001	TLB	-	-	-	-	L2	Data

Error Detected on a Scrub (SCRUB)—Bit 40.

DC: If set, indicates that the error was detected on a scrub.

IC/BU/LS: Reserved.

Uncorrectable ECC Error (UECC)—Bit 45. If set, indicates that the error is an uncorrectable ECC error.

Correctable ECC Error (CECC)—Bit 46. If set, indicates that the error is a correctable ECC error.

ECC Syndrome Bits 7–0 (SYND)—Bits 54–47.

DC: The lower 8 syndrome bits when an ECC error is detected.

IC/BU/LS: Reserved.

Processor State Corrupted By Error (PCC)—Bit 57. If set, indicates that the processor state may have been corrupted by the error condition.

Error address in MC*i*_ADDR (ADDRV)—Bit 58. If set, indicates that the address saved in the corresponding MC*i*_ADDR register is the address where the error occurred.

Additional Info in MC*i*_MISC (MISCV)—Bit 59. If set, indicates that the MC*i*_MISC register contains additional info. This bit is always set to 0 on an AMD NPT Family 0Fh Processor.

Error Reporting Enabled (EN)—Bit 60. If set, indicates that error reporting was enabled for this error in the corresponding MC*i*_CTL register.

Error Not Corrected (UC)—61. If set, it indicates that the error was not corrected.

Second Error Detected (OVER)—Bit 62. Set if a second error is detected while the VAL bit of this register is already set.

Valid (VAL)—Bit 63. Set if the information in this register is valid.

Table 50. DC Error Status Bit Settings

Error Type	UC	ADDRV	PCC	ECC_Synd Valid	CECC	UECC	SCRUB
System Line Fill	If multi-bit	1	If multi-bit	Y	If single-bit	If multi-bit	0
L2 Cache Line Fill	If multi-bit	1	If multi-bit	Y	If single-bit	If multi-bit	0
Data Load/ Store/ Victim/ Snoop	1	1/0	1	Y	If single-bit	If multi-bit	0
Data Scrub	If multi-bit	1	0	Y	If single-bit	If multi-bit	1
Tag Snoop/ Victim	1	1/0	1	N	0	0	0
Tag Load/ Store	1	1	1	N	0	0	0
L1 TLB	1	1	1	N	0	0	0
L1 TLB Multimatch	1	1	1	N	0	0	0
L2 TLB	1	1	1	N	0	0	0
L2 TLB Multimatch	1	1	1	N	0		0

6.4.1.4 MC0_ADDR—DC Machine Check Address Register (MSR 0402h)

The contents of this register are valid only if the ADDRV bit in the MC0_STATUS register is set to 1. As shown in "MCi_ADDR—Machine Check Address Registers" on page 224, the address can be up to 48 bits wide. In reality, not all address bits may be valid and the address field of the MC0_ADDR varies in width. The bit ranges depend on the values in the MC0_STATUS register (i.e., the type of error detected) as shown in Table 51.

Table 51. Valid MC0_ADDR Bits

Error Source	Access Type	Valid Address Bits
Snoop Tag Array	snoop	physical[11–6]
	victim	
Data Tag Array	load	physical[39–3]
	store	

Table 51. Valid MC0_ADDR Bits

Error Source	Access Type	Valid Address Bits
Data Array	victim	physical[11–6]
	snoop	
	load	physical[39–3]
	store	
	scrub	physical[11–3]
L1 Data TLB	load, store	linear[47–12]
L2 Data TLB		
L2 Cache Data	line-fill	physical[39–6]
System Data		

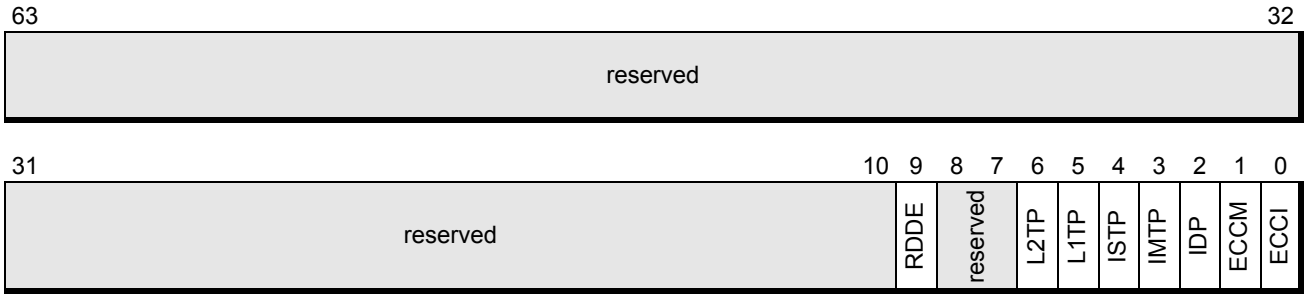
6.4.2 Instruction Cache (IC)

The IC unit includes the level 1 instruction cache that holds instruction data and tags, as well as two levels of TLBs and instruction cache probing logic.

6.4.2.1 MC1_CTL—IC Machine Check Control Register

This register enables the reporting, via the MCA interface, of a variety of errors detected by the Instruction Cache (IC) processor unit. For an error to be reported, both the global IC enable, MCG_CTL[ICE], and the corresponding local enable shown below must be set. If the local enable is off, the error is logged in MC1_STATUS, but not reported via the machine check exception. If the global IC enable is off, no error is logged or reported.

MC1_CTL Register **MSR 0404h**



Bit	Name	Function	R/W	Reset
63–10	reserved			0
9	RDDE	Read Data Errors	R/W	0
8–7	reserved			0
6	L2TP	L2 TLB Parity Errors	R/W	0
5	L1TP	L1 TLB Parity Errors	R/W	0
4	ISTP	Snoop tag array parity errors	R/W	0

Bit	Name	Function	R/W	Reset
3	IMTP	Main tag array parity errors	R/W	0
2	IDP	Data array parity errors	R/W	0
1	ECCM	Multi-bit ECC data errors	R/W	0
0	ECCI	Single-bit ECC data errors	R/W	0

Field Descriptions

Single-bit ECC Data Errors (ECCI)—Bit 0. Report single-bit ECC data errors during instruction cache line fills or TLB reloads from the internal L2 or the system.

Multi-bit ECC Data Errors (ECCM)—Bit 1. Report multi-bit ECC data errors during instruction cache line fills or TLB reloads from the internal L2 or the system.

Data Array Parity Errors (IDP)—Bit 2. Report instruction cache data array parity errors.

Main Tag Array Parity Errors (IMTP)—Bit 3. Report instruction cache main tag array parity errors.

Snoop Tag Array Parity Errors (ISTP)—Bit 4. Report instruction cache snoop tag array parity errors.

L1 TLB Parity Errors (L1TP)—Bit 5. Report instruction cache L1 TLB parity errors.

L2 TLB Parity Errors (L2TP)—Bit 6. Report instruction cache L2 TLB parity errors.

Read Data Errors (RDDE)—Bit 9. Report system read data errors for an instruction cache fetch if MC2_CTL[S_RDE_ALL] = 1.

6.4.2.2 MC1_CTL_MASK—IC Machine Check Control Mask Register (MSR C001_0045h)

The MC1_CTL_MASK register can be used to mask the enabling of individual error reporting features in IC. Any bit set to 1 in this register inhibits writing 1s to the corresponding bits of the MC1_CTL register. See "MCi_CTL_MASK—Machine Check Control Mask Registers" on page 222 for a description of this register.

6.4.2.3 MC1_STATUS—IC Machine Check Status Register (MSR 0405h)

The MC1_STATUS MSR describes the error that was detected by IC and is very similar to MC0_STATUS. See "MC0_STATUS—DC Machine Check Status Register" on page 227 for a description of the fields in this register.

Table 52. IC Error Codes

Error Type	Extended Error Code	Error Code						
		Type	PP	T	RRRR	II	LL	TT
System Line Fill	0000	BUS	SRC	0	IRD	MEM	LG	-
L2 Cache Line Fill	0000	Memory	-	-	IRD	-	L2	Instruction
IC Data Load	0000	Memory	-	-	IRD	-	L1	Instruction
Tag Snoop/ Victim	0000	Memory	-	-	Snoop/ Evict	-	L1	Instruction
Tag Load/ Victim	0000	Memory	-	-	IRD/Evict	-	L1	Instruction
L1 TLB	0000	TLB	-	-	-	-	L1	Instruction
L1 TLB Multimatch	0001	TLB	-	-	-	-	L1	Instruction
L2 TLB	0000	TLB	-	-	-	-	L2	Instruction
L2 TLB Multimatch	0001	TLB	-	-	-	-	L2	Instruction
System Data Read Error	0000	BUS	SRC	0	IRD	MEM	LG	-

Table 53. IC Error Status Bit Settings

Error Type	UC	ADDRV	PCC	Syndrome	CECC	UECC	SCRUB
System Line Fill	If multi-bit	1	If multi-bit	N	If single-bit	If multi-bit	0
L2 Cache Line Fill	If multi-bit	1	If multi-bit	N	If single-bit	If multi-bit	0
IC Data Load	0	1	0	N	0	0	0
Tag Snoop/ Victim	1	1/0	1	N	0	0	0
Tag Load/ Victim	0	1/0	0	N	0	0	0
L1 TLB	0	1	0	N	0	0	0
L1 TLB Multimatch	0	1	0	N	0	0	0
L2 TLB	0	1	0	N	0	0	0

Table 53. IC Error Status Bit Settings (Continued)

Error Type	UC	ADDRV	PCC	Syndrome	CECC	UECC	SCRUB
L2 TLB Multimatch	0	1	0	N	0		0
System Data Read Error	1	0	0	N	0	0	0

6.4.2.4 MC1_ADDR—IC Machine Check Address Register (MSR 0406h)

The contents of this register are valid only if the ADDRV bit in the MC1_STATUS register is set to 1. The address can be up to 48 bits wide. In reality, not all address bits may be valid and the address field of the MC1_ADDR varies in width. The bits ranges depend on the values in the MC1_STATUS register (i.e., the type of error detected) and this is shown in Table 54.

Table 54. Valid MC1_ADDR Bits

Error Source	Access Type	Valid Address Bits
Snoop Tag Array	Snoop	Physical[39–6]
	Victim	None
Instruction Tag Array	Code Read	Linear[47–6]
	Victim	None
Instruction Data Array	Code Read	Linear[47–4]
L1 TLB	Code Read	Linear[47–12] for 4-Kbyte page Linear[47–20] for 2-Mbyte page
L2 TLB	Code Read	Linear[47–12] for 4-Kbyte page
L2 Cache Data	Line-fill	Physical[39–6]
System Data		
System Address Out of Range		None

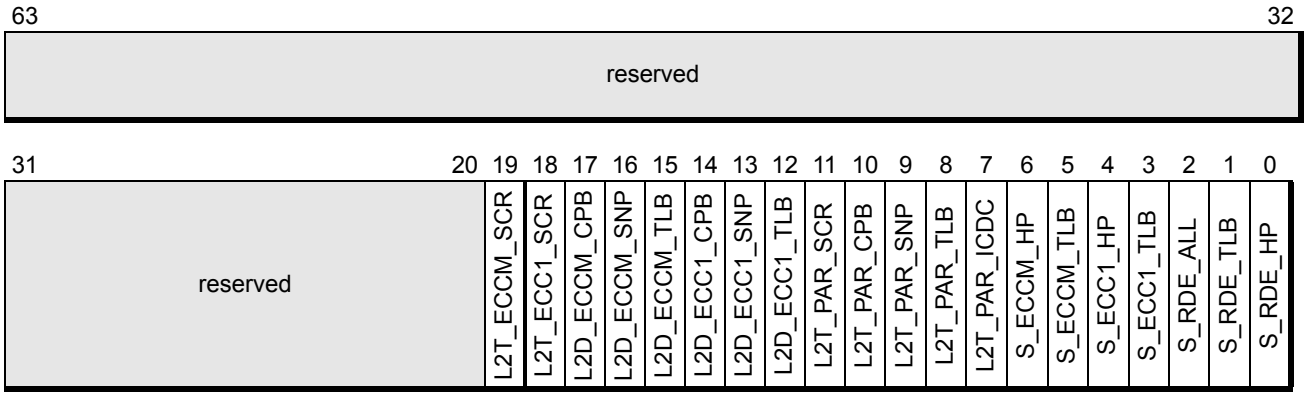
6.4.3 Bus Unit (BU)

The bus unit consists of the system bus interface logic and the L2 cache.

6.4.3.1 MC2_CTL—BU Machine Check Control Register

This register enables the reporting, via the MCA interface, of a variety of errors detected in the Bus processor unit (BU). For an error to be reported, both the global BU enable, MCG_CTL[BUE], and the corresponding local enable shown below must be set. If the local enable is off, the error is logged in MC2_STATUS, but not reported via the machine check exception. If the global BU enable is off, no error is logged or reported.

MC2_CTL Register — MSR 0408h



Bit	Name	Function	R/W	Reset
63–20	reserved			0
19	L2T_ECCM_SCR	L2 tag array multi-bit ECC Scrub	R/W	0
18	L2T_ECC1_SCR	L2 tag array 1-bit ECC Scrub	R/W	0
17	L2D_ECCM_CPB	L2 data array multi-bit ECC copyback	R/W	0
16	L2D_ECCM_SNP	L2 data array multi-bit ECC snoop	R/W	0
15	L2D_ECCM_TLB	L2 data array multi-bit ECC TLB reload	R/W	0
14	L2D_ECC1_CPB	L2 data array 1-bit ECC copyback	R/W	0
13	L2D_ECC1_SNP	L2 data array 1-bit ECC snoop	R/W	0
12	L2D_ECC1_TLB	L2 data array 1-bit ECC TLB reload	R/W	0
11	L2T_PAR_SCR	L2 tag array parity scrub	R/W	0
10	L2T_PAR_CPB	L2 tag array parity copyback	R/W	0
9	L2T_PAR_SNP	L2 tag array parity snoop	R/W	0
8	L2T_PAR_TLB	L2 tag array parity TLB reload	R/W	0
7	L2T_PAR_ICDC	L2 tag array parity IC or DC fetch	R/W	0
6	S_ECCM_HP	System data multi-bit ECC hardware prefetch	R/W	0
5	S_ECCM_TLB	System data multi-bit ECC TLB reload	R/W	0
4	S_ECC1_HP	System data 1-bit ECC hardware prefetch	R/W	0
3	S_ECC1_TLB	System data 1-bit ECC TLB reload	R/W	0
2	S_RDE_ALL	All system read data	R/W	0
1	S_RDE_TLB	System read data TLB reload	R/W	0
0	S_RDE_HP	System read data hardware prefetch	R/W	0

Field Descriptions

System Read Data Hardware Prefetch (S_RDE_HP)—Bit 0. Report system read data errors for a hardware prefetch.

System Read Data TLB Reload (S_RDE_TLB)—Bit 1. Report system read data errors for a TLB reload.

All System Read Data (S_RDE_ALL)—Bit 2. Report system read data errors for any operation including a DC/IC fetch, TLB reload or hardware prefetch.

System Data 1-bit ECC TLB Reload (S_ECC1_TLB)—Bit 3. Report system data 1-bit ECC errors for a TLB reload.

System Data 1-bit ECC Hardware Prefetch (S_ECC1_HP)—Bit 4. Report system data 1-bit ECC errors for a hardware prefetch.

System Data Multi-bit ECC TLB Reload (S_ECCM_TLB)—Bit 5. Report system data multi-bit ECC errors for a TLB reload.

System Data Multi-bit ECC Hardware Prefetch (S_ECCM_HP)—Bit 6. Report system data multi-bit ECC errors for a hardware prefetch.

L2 Tag Array Parity IC or DC Fetch (L2T_PAR_ICDC)—Bit 7. Report L2 tag array parity errors for an IC or DC fetch.

L2 Tag Array Parity TLB Reload (L2T_PAR_TLB)—Bit 8. Report L2 tag array parity errors for a TLB reload.

L2 Tag Array Parity Snoop (L2T_PAR_SNP)—Bit 9. Report L2 tag array parity errors for a snoop.

L2 Tag Array Parity Copyback (L2T_PAR_CPB)—Bit 10. Report L2 tag array parity errors for a copyback.

L2 Tag Array Parity Scrub (L2T_PAR_SCR)—Bit 11. Report L2 tag array parity errors for a scrub.

L2 Data Array 1-bit ECC TLB Reload (L2D_ECC1_TLB)—Bit 12. Report L2 data array 1-bit ECC errors for a TLB reload.

L2 Data Array 1-bit ECC Snoop (L2D_ECC1_SNP)—Bit 13. Report L2 data array 1-bit ECC errors for a snoop.

L2 Data Array 1-bit ECC Copyback (L2D_ECC1_CPB)—Bit 14. Report L2 data array 1-bit ECC errors for a copyback.

L2 Data Array Multi-bit ECC TLB Reload (L2D_ECCM_TLB)—Bit 15. Report L2 data array multi-bit ECC errors for a TLB reload.

L2 Data Array Multi-bit ECC Snoop (L2D_ECCM_SNP)—Bit 16. Report L2 data array multi-bit ECC errors for a snoop.

L2 Data Array Multi-bit ECC Copyback (L2D_ECCM_CPB)—Bit 17. Report L2 data array multi-bit ECC errors for a copyback.

L2 Tag Array 1-bit ECC Scrub (L2T_ECC1_SCR)—Bit 18. Report L2 tag array 1-bit ECC errors for a scrub.

L2 Tag Array Multi-bit ECC Scrub (L2T_ECCM_SCR)—Bit 19. Report L2 tag array multi-bit ECC errors for a scrub.

6.4.3.2 MC2_CTL_MASK—BU Machine Check Control Mask Register (MSR C001_0046h)

The MC2_CTL_MASK register can be used to mask the enabling of individual error reporting features in BU. Any bit set to 1 in this register inhibits writing 1s to the corresponding bits of the MC2_CTL register. See "MCi_CTL_MASK—Machine Check Control Mask Registers" on page 222 for a description of this register.

6.4.3.3 MC2_STATUS—BU Machine Check Status Register (MSR 0409h)

The MC2_STATUS MSR describes the error that was detected by BU and is very similar to MC0_STATUS. See "MC0_STATUS—DC Machine Check Status Register" on page 227 for a description of the fields in this register.

Table 55. BU Error Codes

Error Type	Access Type	Extended Error Code	Error Code						
			Type	PP	T	RRRR	II	LL	TT
Tag Parity	Instruction Fetch	0010	Memory	-	-	IRD	-	L2	Instruction
	Data Fetch	0010	Memory	-	-	DRD	-	L2	Data
	TLB/ Snoop/ Evict	0010	Memory	-	-	RD/ Snoop/ Evict	-	L2	Generic
	Scrub	0010	Memory	-	-	GEN	-	L2	Generic
Tag ECC	Scrub	0010	Memory	-	-	GEN	-	L2	Instruction
System Data Read Error	TLB	0000	BUS	SRC	0	RD	MEM	LG	-
	Hardware Prefetch	0000	BUS	SRC	0	Prefetch	MEM/IO	LG	-
System Data ECC	TLB	0000	BUS	SRC	0	RD	MEM/IO	LG	-
	Hardware Prefetch	0000	BUS	SRC	0	Prefetch	MEM	LG	-
Data ECC	TLB	0000	Memory	-	-	RD	-	L2	Generic
Data Copyback	Snoop/ Evict	0000	Memory	-	-	Snoop/ Evict	-	L2	Generic

Table 56. BU Error Status Bit Settings

Error Type	Access Type	UC	ADDRV	PCC	CECC	UECC	SCRUB
Tag Parity	Instruction Fetch	1	1	1	0	0	0
	Data Fetch	1	1	1	0	0	0
	TLB/ Snoop/ Evict	1	1	1	0	0	0
	Scrub	1	1	0	0	0	1
Tag ECC	Scrub	If multi-bit	1	0	If single-bit	If multi-bit	1
System Data Read Error	TLB	1	1	0	0	0	0
	Hardware Prefetch	1	0	0	0	0	0
System Data ECC	TLB	If multi-bit	1	If multi-bit	If single-bit	If multi-bit	0
	Hardware Prefetch	If multi-bit	0	If multi-bit	If single-bit	If multi-bit	0
Data ECC	TLB	If multi-bit	1	If multi-bit	If single-bit	If multi-bit	0
Data Copyback	Snoop/ Evict	If multi-bit	1	If multi-bit	If single-bit	If multi-bit	0

6.4.3.4 MC2_ADDR—BU Machine Check Address Register (MSR 040Ah)

The contents of this register are valid only if the ADDRV bit in the MC2_STATUS register is set to 1. The address can be up to 48 bits wide. In reality, not all address bits may be valid and the address field of the MC2_ADDR varies in width. The bit ranges depend on the values in the MC2_STATUS register (i.e., the type of error detected) and this is shown in Table 57.

Table 57. Valid MC2_ADDR Bits

Error Source	Access Type	Valid Address Bits
L2 Cache—Data Array	Any except evict	Physical[39–6]
	Evict	Physical[15–6] for 1-Mbyte L2 Physical[14–6] for 512-Kbyte L2 Physical[13–6] for 256-Kbyte L2 Physical[12–6] for 128-Kbyte L2

Table 57. Valid MC2_ADDR Bits (Continued)

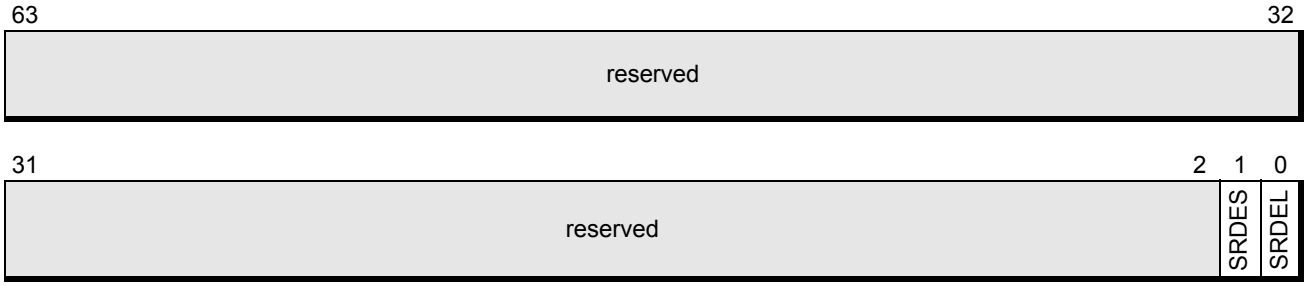
Error Source	Access Type	Valid Address Bits
L2 Cache—Tag Array	Any	MC2_ADDR[3–0] contains encoded cache way Physical[15–6] for 1-Mbyte L2 Physical[14–6] for 512-Kbyte L2 Physical[13–6] for 256-Kbyte L2 Physical[12–6] for 128-Kbyte L2
System Address Out of Range	Any	Physical[39–6]

6.4.4 Load Store Unit (LS)

6.4.4.1 MC3_CTL—LS Machine Check Control Register

This register enables the reporting, via the MCA interface, of two types of errors detected by the Load/Store (LS) processor unit. For an error to be reported, both the global LS enable, MCG_CTL[LSE], and the corresponding local enable shown below must be set. If the local enable is off, the error is logged in MC3_STATUS, but not reported via the machine check exception. If the global LS enable is off, no error is logged or reported.

MC3_CTL Register **MSR 040Ch**



Bit	Name	Function	R/W	Reset
63–3	reserved			0
2	reserved		R/W	0
1	S_RDE_S	Read Data Errors on Store	R/W	0
0	S_RDE_L	Read Data Errors on Load	R/W	0

Field Descriptions

Read Data Errors on Load (S_RDE_L)—Bit 0. Report system read data errors on a load if MC2_CTL[S_RDE_ALL] = 1.

Read Data Errors on Store (S_RDE_S)—Bit 1. Report system read data errors on a store if MC2_CTL[S_RDE_ALL] = 1.

6.4.4.2 MC3_CTL_MASK—LS Machine Check Control Mask Register (MSR C001_0047h)

The MC3_CTL_MASK register can be used to mask the enabling of individual error reporting features in LS. Any bit set to 1 in this register inhibits writing 1s to the corresponding bits of the MC3_CTL register. See "MCi_CTL_MASK—Machine Check Control Mask Registers" on page 222 for a description of this register.

6.4.4.3 MC3_STATUS—LS Machine Check Status Register (MSR 040Dh)

The MC3_STATUS MSR describes the error that was detected by LS and is very similar to MC0_STATUS. See "MC0_STATUS—DC Machine Check Status Register" on page 227 for a description of the fields in this register.

Table 58. LS Error Codes

Error Type	Extended Error Code	Error Code						
		Type	PP	T	RRRR	II	LL	TT
Read Data on Store	0000	BUS	SRC	0	DWR	MEM	LG	-
Read Data on Load	0000	BUS	SRC	0	DRD	MEM/IO	LG	-

Table 59. LS Error Status Bit Settings

Error Type	UC	ADDRV	PCC	Syndrome	CECC	UECC	SCRUB
Read Data on Store	1	1/0	1/0	N	0	0	0
Read Data on Load	1	1/0	1/0	N	0	0	0

6.4.4.4 MC3_ADDR—LS Machine Check Address Register (MSR 040Eh)

The contents of this register are valid only if the ADDRV bit in the MC3_STATUS register is set to 1. The only type of error recorded by the LS machine check mechanism is a "system address out of range" or read data error for which MC3_ADDR[39:0] store the physical address.

6.4.5 Northbridge (NB)

The Northbridge and DRAM memory controller are included in this block.

6.4.5.1 MC4_CTL—NB Machine Check Control Register (MSR 0410h)

This register enables the reporting, via the MCA interface, of the errors detected by the North Bridge (NB) processor unit. "MCA NB Control Register" (Function 3, Offset 40h) maps to MC4_CTL[31:0]. See "MCA NB Control Register" on page 141 for detailed information on this register. For an error to be reported, both the global NB enable, MCG_CTL[NBE], and the corresponding local enable must be set. If the local enable is off, the error is logged in MC4_STATUS, but not reported via the machine check exception. If the global NB enable is off, no error is logged or reported.

6.4.5.2 MC4_CTL_MASK—NB Machine Check Control Mask Register (MSR C001_0048h)

The MC4_CTL_MASK register can be used to mask the enabling of individual error reporting features in the Northbridge. Any bit set to 1 in this register inhibits writing1s to the corresponding bits of the MC4_CTL register. See "MCi_CTL_MASK—Machine Check Control Mask Registers" on page 222 for a description of this register.

6.4.5.3 MC4_STATUS—NB Machine Check Status Register (MSR 0411h)

The MC4_STATUS MSR describes the error that was detected by the Northbridge. "MCA NB Status Low Register" (Function 3, Offset 48h) maps to MC4_STATUS MSR[31:0] and "MCA NB Status High Register" (Function 3, Offset 4Ch) maps to MC4_STATUS MSR[63:32]. See "MCA NB Status Low Register" on page 147 and "MCA NB Status High Register" on page 150 for more detailed information.

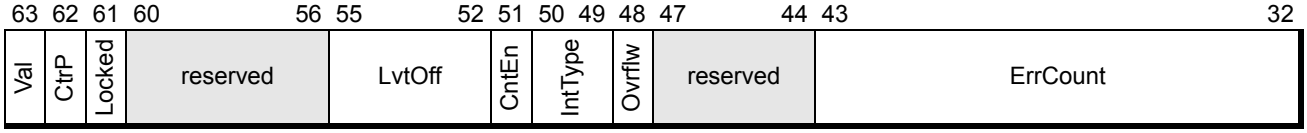
6.4.5.4 MC4_ADDR—NB Machine Check Address Register (MSR 0412h)

The contents of this register are valid only if the ErrAddrVal bit in the MC4_STATUS register is set to 1. "MCA NB Address Low Register" (Function 3, Offset 50h) maps to MC4_ADDR[31:0] and "MCA NB Address High Register" (Function 3, Offset 54h) maps to MC4_ADDR[63:32]. See "MCA NB Address Low Register" on page 153 and "MCA NB Address High Register" on page 154 for a description of this register.

6.4.5.5 MC4_MISC—DRAM Errors Threshold Register

This register is used to count correctable and uncorrectable ECC errors that occur during DRAM read operations.

MC4_MISC—DRAM Errors Threshold Register **MSR 0413h**



Bits	Mnemonic	Function	R/W	Reset
63	Val	Valid	R	1
62	CtrP	Counter Present	R	1
61	Locked	Locked	R	0
60–56	reserved		R	0
55–52	LvtOff	LVT Offset	R	0
51	CntEn	Counter Enable	R/W	0
50–49	IntType	Interrupt Type	R/W	X
48	Ovrflw	Overflow	R/W	X
47–44	reserved		R	0
43–32	ErrCount	Error Count	R/W	X
31–0	reserved		R	0

Field Descriptions

Error Count (ErrCount)—Bits 43–32. This field is written by software to set the starting value of the error counter. This field is incremented by hardware when errors are logged. When this counter overflows, it stays at FFFh (no rollover). To set the threshold value, software should subtract the desired error count (the number of errors necessary in order for an interrupt to be taken) from FFFh and write the result into this field. This field is read only if the Locked bit (MSR 0413h) is set.

Overflow (Ovrflw)—Bit 48. This bit is set by hardware when ErrCount transitions from FFEh to FFFh. When this bit is set, the interrupt selected by the IntType field is generated. This bit can be cleared by writing a zero to the bit. This bit is read only if the Locked bit (MSR 0413h) is set.

Interrupt Type (IntType)—Bits 50–49. This field determines the type of interrupt signaled when the counter overflows. This field is read only if the Locked bit (MSR 0413h) is set.

00b = No Interrupt.
01b = APIC based interrupt.
10b = SMI.
11b = Reserved.

Counter Enable (CntEn)—Bit 51. When this bit is set, DRAM ECC error counting is enabled. This field is read only if the Locked bit (MSR 0413h) is set.

LVT Offset (LvtOff)—Bits 55–52. This field is shifted left by four bits and added to 500h to determine the address of the LVT entry in the APIC registers.

Locked (Locked)—Bit 61. This bit is set by BIOS to indicate that this register is not available for OS use. When this bit is set, write to this register are silently ignored. BIOS should set this bit if IntType is set to SMI. This bit is read/write if the MCi_STATUS_WREN bit (MSR C001_0015h bit 18) is set.

Counter Present (CtrP)—Bit 62. This bit indicates the presence of the ErrCount field in the MISC register. This bit is read/write if the MCi_STATUS_WREN bit (MSR C001_0015h bit 18) is set.

Valid (Valid)—Bit 63. This bit indicates the presence of a valid CtrP field in the MISC register. This bit is read/write if the MCi_STATUS_WREN bit (MSR C001_0015h bit 18) is set.

6.5 Initializing the Machine Check Mechanism

Following is the general process system software should follow to initialize the machine check mechanism:

1. Execute the CPUID and verify that the processor supports the machine check exception (MCE) and MCA. MCE is supported when EDX bit 7 is set to 1, and MCA is supported when EDX bit 14 set to 1. Software should not proceed with initializing the machine check mechanism if MCE is not supported.
2. If MCA is supported, system software should take the following steps:
   a. Check to see if the MCG_CTL_P bit in the MCG_CAP register is set to 1. If it is, then the MCG_CTL register is supported by the processor. When this register is supported, software should set its enable bits to 1 for the machine check features it uses. Software can load MCG_CTL with all 1s to enable all machine check features.
   b. Read the COUNT field from the MCG_CAP register to determine the number of error-reporting register banks supported by the processor. This is set to 5 in AMD NPT Family 0Fh Processors since there are five blocks—DC, IC, BU, LS, and NB. For each error-reporting register bank, software should set the enable bits to 1 in the MC*i*_CTL register for the error types it wants the processor to report. Software can load each MC*i*_CTL register bit with a 1 to enable all error-reporting mechanisms.

      The error-reporting register banks are numbered from 0 to one less than the value found in the MCG_CAP.COUNT field. For example, when the COUNT field indicates that 5 register banks are supported, they are numbered 0 to 4.
   c. For each error-reporting register bank, software should clear all status fields in the MC*i*_STATUS register by writing all 0s to the register.

      It is possible that valid error status is reported in the MC*i*_STATUS registers at the time software clears them. The status can reflect fatal errors recorded before a processor reset or errors recorded during the system power-up and boot process. Prior to clearing the MC*i*_STATUS registers, software should examine their contents and log any errors found.

See "Machine Check Architecture (MCA)" on page 371 for BIOS specific initialization requirements.

3. As a final step in the initialization process, system software should enable the machine check exception by setting CR4.MCE (bit 6) to 1.

6.6 Using Machine Check Features

System software can detect and handle machine check errors using two methods:

- Software can periodically examine the machine check status registers for reported errors, and log any errors found.
- Software can enable the machine check exception (#MC). When an uncorrectable error occurs, the processor immediately transfers control to the machine check exception handler. In this case, system software provides a machine check exception handler that, at a minimum, logs detected errors. The exception handler can be designed for a specific processor implementation or can be generalized to work on multiple implementations.

6.6.1 Handling Machine Check Exceptions

The processor uses the interrupt control-transfer mechanism to invoke an exception handler after a machine check exception occurs. This requires system software to initialize the interrupt-descriptor table (IDT) with either an interrupt gate or a trap gate that references the interrupt handler.

At a minimum, the machine check exception handler must be capable of logging errors for later examination. Because most machine check errors are not recoverable, the ability to log errors can be sufficient for implementation of the handler. More thorough exception-handler implementations can analyze errors to determine if each error is recoverable. If a recoverable error is identified, the exception handler can attempt to correct the error and restart the interrupted program.

Machine check exception handlers that attempt to correct recoverable errors must be thorough in their analysis and the corrective actions they take. The following guidelines should be used when writing such a handler:

- All status registers in the error-reporting register banks must be examined to identify the cause or causes of the machine check exception. Read the COUNT field from MCG_CAP to determine the number of status registers supported by the processor. The status registers are numbered from 0 to one less than the value found in the MCG_CAP.COUNT field. For example, if the COUNT field indicates five status registers are supported, they are numbered MC0_STATUS to MC4_STATUS.
- Check the valid bit in each status register (MC*i*_STATUS.VAL). The MC*i*_STATUS register does not need to be examined when its valid bit is clear.

- Check the valid MC*i*_STATUS registers to see if error recovery is possible. Error recovery is not possible when:
  - The processor-context corrupt bit (MC*i*_STATUS.PCC) is set to 1.
  - The error-overflow status bit (MC*i*_STATUS. OVER) is set to 1. This bit indicates that more than one machine check error has occurred, but only one error is reported by the status register.

If error recovery is not possible, the handler should log the error information and return to the operating system.

- Check the MC*i*_STATUS.UC bit to see if the processor corrected the error. If UC=1, the processor did not correct the error, and the exception handler must correct the error prior to restarting the interrupted program. If the handler cannot correct the error, it should log the error information and return to the operating system.
- When identifying the error condition, portable exception handlers should examine only the MCA error-code field of the MC*i*_STATUS register. See the error codes in tables 17 through 22 for information on interpreting this field.

If the MCG_STATUS.RIPV bit is set to 1, the interrupted program can be restarted reliably at the instruction-pointer address pushed onto the exception-handler stack. If RIPV = 0, the interrupted program cannot be restarted reliably, although it may be possible to restart it for debugging purposes.

- When logging errors, particularly those that are not recoverable, check the MCG_STATUS.EIPV bit to see if the instruction-pointer address pushed onto the exception-handler stack is related to the machine check error. If EIPV = 0, the address is not guaranteed to be related to the error.
- Prior to exiting the machine check handler, be sure to clear MCG_STATUS.MCIP to 0. MCIP indicates that a machine check exception occurred. If this bit is set when another machine check exception occurs, the processor enters the shutdown state.
- When an exception handler is able to, at a minimum, successfully log an error condition, clear the MC*i*_STATUS registers prior to exiting the machine check handler. Software is responsible for clearing at least the MC*i*_STATUS.VAL bit.
- Additional machine check exception-handler portability can be added by having the handler use the CPUID instruction to identify the processor and its capabilities. Implementation-specific software can be added to the machine check exception handler based on the processor information reported by CPUID.

6.6.1.1 Reporting Correctable Machine Check Errors

Machine check exceptions do not occur if the error is correctable by the processor. If system software wishes to log and report correctable machine check errors, a system-service routine must be provided to check the contents of the machine check status registers for correctable errors. The service routine can be invoked by system software on a periodic basis, or it can be manually invoked by the user as needed.

Assuming that the processor supports the machine check registers, a service routine that reports correctable errors should perform the following:

1. Examine each status register (MC*i*_STATUS) in the error-reporting register banks. For each MC*i*_STATUS register with a set valid bit (VAL = 1), the service routine should:
   a. Save the contents of the MC*i*_STATUS register.
   b. Save the contents of the corresponding MC*i*_ADDR register if MC*i*_STATUS.ADDRV = 1.
   c. Save the contents of the corresponding MC*i*_MISC register if MC*i*_STATUS.MISCV = 1.
   d. Check to see if MCG_STATUS.MCIP = 1, indicating that the machine check exception handler is in progress. If this is the case, then the machine check exception handler has called the service routine to log the errors. In this situation, the error-logging service routine should determine whether or not the interrupted program is restartable, and report the determination back to the exception handler. The program is *not restartable* if either of the following is true:
      – MC*i*_STATUS.PCC = 1, indicating the processor context is corrupted.
      – MCG_STATUS.RIPV = 0, indicating the interrupted program cannot be restarted reliably at the instruction-pointer address pushed onto the exception-handler stack.
2. Once the information found in the error-reporting register banks is saved, the MC*i*_STATUS register should be cleared to 0. This allows the processor to properly report any subsequent errors in the MC*i*_STATUS registers.
3. In multiprocessor configurations, the service routine can save the processor node identifier. This can help locate a failing multiprocessor-system component, which can then be isolated from the rest of the system.

7 Advanced Programmable Interrupt Controller (APIC)

All AMD NPT Family 0Fh Processor-based systems must support 64-bit operating systems. As a result, all platforms must support APIC and declare it to the operating system in the appropriate tables.

The local APIC contains logic to receive interrupts from a variety of sources and to send interrupts to other local APICs, as well as registers to control its behavior and report status.

Interrupts can be received from:

- HyperTransport™ I/O devices, including the I/O hub (I/O APICs)
- Other local APICs (inter-processor interrupts)
- APIC timer
- Performance counters
- Legacy local interrupts from the I/O hub (INTR and NMI)
- APIC internal errors

The APIC timer, performance counters, local interrupts, and internal errors are all considered local interrupt sources, and their routing is controlled by local vector table entries. These entries assign a message type and vector to each interrupt, allow them to be masked, and track the status of the interrupt.

I/O and inter-processor interrupts have their message type and vector assigned at the source and are unaltered by the local APIC. They carry a destination field and a mode bit that together determine which local APICs accepts them. The destination mode (DM) bit specifies if the interrupt request packet should be handled in Physical or Logical destination mode.

- In physical destination mode, if the destination field is FFh, the interrupt is a broadcast and is accepted by all local APICs. Otherwise, the interrupt is only accepted by the local APIC whose APIC ID matches the destination field of the interrupt. Physical mode allows up to 255 APICs to be addressed individually.

- In logical destination mode, a local APIC accepts interrupts selected by the logical destination register (LOG_DEST) and the destination field of the interrupt using either cluster or flat format, as configured by the destination format register.
  - If flat destinations are in use, bits 7–0 of the LOG_DEST field are checked against bits 7–0 of the arriving interrupt's Destination Field. If any bit position is set in both fields, this APIC is a valid destination. Flat format allows up to 8 APICs to be addressed individually.
  - If cluster destinations are in use, bits 7–4 of the LOG_DEST field are checked against bits 7–4 of the arriving interrupt's destination field, identifying the cluster. If all of bits 7–4 match, then bits 3–0 of the LOG_DEST and the interrupt destination are checked for any bit positions that are set in both fields, choosing processors within the cluster. If both conditions are met, this APIC is a valid destination. Cluster format allows 15 clusters of 4 APICs each to be addressed.
  - In both flat and cluster formats, if the destination field is FFh, the interrupt is a broadcast and is accepted by all local APICs.

7.1 Interrupt Delivery

SMI, NMI, INIT, Startup, and External interrupts are classified as non-vectored interrupts.

When an APIC accepts a non-vectored interrupt, it is handled directly by the processor instead of being queued in the APIC. When an APIC accepts a fixed or lowest priority interrupt, it sets the bit in the interrupt request register (IRR) corresponding to the vector in the interrupt. (For local interrupt sources, this comes from the vector field in that interrupt's local vector table entry.) If a subsequent interrupt with the same vector arrives when the corresponding IRR bit is already set, the two interrupts are collapsed into one. Vectors 0–15 are reserved.

7.2 Vectored Interrupt Handling

The task priority register (TPR_PRI) and processor priority register (PROC_PRI) each contain an 8-bit priority, divided into a main priority (bits 7–4) and a priority sub-class (3–0). The task priority is assigned by software to set a threshold priority at which the processor is interrupted.

To calculate processor priority, an 8-bit ISR vector is set based on the highest bit set in the in-service register (ISR). Like the TPR_PRI and PROC_PRI, this vector is divided into a main priority (7–4) and priority sub-class (3–0). The main priority of the TPR_PRI and ISR are compared and the highest is the PROC_PRI, as follows:

```
If (TPR_PRI[7:4] >= ISRVect[7:4])PROC_PRI = TPR_PRI
Else                             PROC_PRI = {ISRVect[7:4],0h}
```

The resulting processor priority is used to determine if any accepted interrupts (indicated by IRR bits) are high enough priority to be serviced by the processor. When the processor is ready to service an interrupt, the highest bit in the IRR is cleared, and the corresponding bit is set in the ISR. The

corresponding trigger mode register (TMR) bit is set if the interrupt is level-triggered and cleared if edge-triggered.

When the processor has completed service for an interrupt, it performs a write to the end of interrupt (EOI) register, clearing the highest ISR bit and causing the next-highest interrupt to be serviced. If the corresponding TMR bit is set, EOI messages is sent to all APICs to complete service of the interrupt at the source.

7.3 Spurious Interrupts

In the event that the task priority is set to or above the level of the interrupt to be serviced, the local APIC delivers a spurious interrupt vector to the processor, as specified by the spurious interrupt vector register. The ISR is not changed and no EOI occurs.

7.3.1 Spurious Interrupts Caused by Timer Tick Interrupt

A typical interrupt is asserted until it is serviced. Interrupt is deasserted when software clears the interrupt status bit within the interrupt service routine. Timer tick interrupt is an exception, since it is deasserted regardless of whether it is serviced or not. A BIOS programs the 8254 Programmable Timer to generate an 18.2 Hz square wave. This square wave represents the timer tick interrupt (PITIRQ) and in some interrupt configurations it is connected to the 8259 Programmable Interrupt Controller (PIC) IRQ0 input.

The processor is not always able to service interrupts immediately (i.e. when interrupts are masked by clearing EFLAGS.IM or when the processor is in debug mode). The method of interrupt delivery to the processor (wire or messages) determines system behavior for the timer tick interrupt connected to the PIC after the processor has not been able to service it for some time. The PIC output INTR is identical to PITIRQ when interrupts are not serviced.

If interrupts are delivered to the processor using a wire, and the processor is not able to service the timer tick interrupt for some time, timer tick interrupts asserted during that time are lost. The following cases are possible when the processor is ready to service interrupts:

- INTR is deasserted and it is not serviced by the processor. This happens almost 50 percent of the time.
- INTR is asserted and it is serviced by the processor. This happens almost 50 percent of the time.
- INTR is asserted, and it is detected by the processor. The processor sends the interrupt acknowledge cycle, but when the PIC receives it, INTR is deasserted, and the PIC sends a spurious interrupt vector. This happens a very small percentage of the time.

The probability of spurious interrupts, when interrupts are delivered to the processor using a wire, is very low. An example of a configuration that uses wire to deliver interrupts to the processor has PITIRQ connected to PIC IRQ0, and PIC INTR connected to LINT0 of an AMD Athlon™ processor.

If interrupts are delivered to the processor using messages, and the processor is not able to service the timer tick interrupt for an extended period of time, the INTR caused by the first timer tick interrupt asserted during that time is delivered to the local APIC in ExtInt mode and latched, and the subsequent timer tick interrupts are lost. The following cases are possible when the processor is ready to service interrupts:

- An ExtInt interrupt is pending, and INTR is asserted. This results in timer tick interrupt servicing. This occurs 50 percent of the time.
- An ExtInt interrupt is pending, and INTR is deasserted. The processor sends the interrupt acknowledge cycle, but when the PIC receives it, INTR is deasserted, and the PIC sends a spurious interrupt vector. This occurs 50 percent of the time.

There is a 50 percent probability of spurious interrupts when interrupts are delivered to the processor using messages. An example of an AMD NPT Family 0Fh Processor configuration that uses messages to deliver interrupts to the processor has PITIRQ connected to PIC IRQ0, PIC INTR connected to IOAPIC IRQ0, and interrupts are delivered to the processor by HyperTransport link messages. HyperTransport link messages are always used to deliver PIC interrupts to the processor in a system with AMD NPT Family 0Fh Processors. An example of an AMD Athlon™ processor configuration that uses messages to deliver interrupts to the processor has PITIRQ connected to PIC IRQ0, PIC INTR connected to IOAPIC IRQ0, and interrupts are delivered to the processor by APIC 3-wire messages.

7.4 Lowest-Priority Arbitration

Fixed, remote read, and non-vectored interrupts are accepted by their destination APICs without arbitration.

Delivery of lowest-priority interrupts requires all APICs to arbitrate to determine which one accepts the interrupt. If focus processor checking is enabled (bit 9 of the spurious interrupt vector register cleared), then the focus processor for an interrupt always accept the interrupt. A processor is the focus of an interrupt if it is already servicing that interrupt (corresponding ISR bit is set) or if it already has a pending request for that interrupt (corresponding IRR bit is set). If there is no focus processor for an interrupt, or focus processor checking is disabled, then each APIC calculates an arbitration priority value, and the one with the lowest result accepts the interrupt.

To calculate arbitration priority, an 8-bit IRR Vector is set based on the highest bit set in the IRR. Like the ISR, TPR_PRI, and PROC_PRI, this vector is divided into a main priority (7–4) and priority sub-class (3–0). The main priority of the TPR_PRI, ISR, and IRR are compared and the highest is the new ARB_PRI, as follows:

```
If   (TPR_PRI[7:4] >= IRRVect[7:4] and
      TPR_PRI[7:4] >  ISRVect[7:4])    ARB_PRI = TPR_PRI
Else if (IRRVect[7:4] > ISRVect[7:4]) ARB_PRI = {IRRVect[7:4],0h}
Else                                  ARB_PRI = {ISRVect[7:4],0h}
```

7.5 Inter-Processor Interrupts

In order to redirect an interrupt to another processor, originate an interrupt to another processor, or allow a processor to interrupt itself, the interrupt command register (ICR) provides a mechanism for generating interrupts. A write to the low doubleword of the ICR causes an interrupt to be generated, with the properties specified by the ICR low and ICR high fields.

7.6 APIC Timer Operation

The local APIC contains a 32-bit timer, controlled by the Timer LVT entry, initial count, and divide configuration registers. The processor bus clock is divided by the value in the timer divide configuration register to obtain a time base for the timer. When the timer initial count register is written, the value is copied into the timer current count register. The current count register is decremented at the rate of the divided clock. When the count reaches 0, a timer interrupt is generated with the vector specified in the Timer LVT entry. If the Timer LVT entry specifies periodic operation, the current count register is reloaded with the initial count value, and it continues to decrement at the rate of the divided clock. If the mask bit in the timer LVT entry is set, timer interrupts are not generated.

7.7 State at Reset

At power-up or reset, all registers take on the values listed in their descriptions. SMI, NMI, INIT, Startup, remote read, and LINT interrupts may be accepted.

When the APIC is software-disabled, pending interrupts in the ISR and IRR are held, but further fixed, lowest-priority, and ExtInt interrupts are not accepted. All LVT entry mask bits are set and cannot be cleared.

When a processor accepts an INIT interrupt, the APIC is reset as at power-up, with the exception that APIC_ID is unaffected.

7.8 Register Summary

All APIC registers are accessed with memory reads and writes of (APIC_BASE+Offset). APIC_BASE is MSR 001Bh and defaults to 00_FEE0_0000h.

The APIC registers and their offsets are listed in Table 60.

Table 60. APIC Register Summary

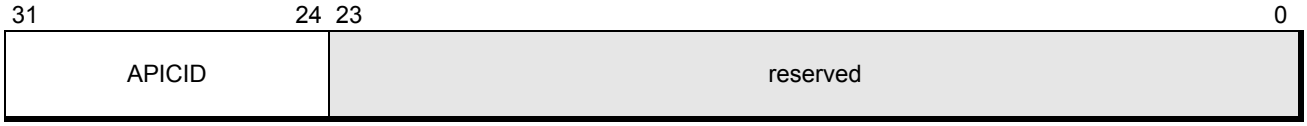
Offset (Hex)	Mnemonic	Name
20	APIC_ID	APIC ID Register
30	APIC_VER	APIC Version Register
80	TPR_PRI	Task Priority Register
90	ARB_PRI	Arbitration Priority Register
A0	PROC_PRI	Processor Priority Register
B0	EOI	End Of Interrupt Register
C0	RMT_RD	Remote Read Register
D0	LOG_DEST	Logical Destination Register
E0	DEST_FORMAT	Destination Format Register
F0	SPUR_INTR_VEC	Spurious Interrupt Vector Register
100-170	ISR	In-Service Registers
180-1F0	TMR	Trigger Mode Registers
200-270	IRR	Interrupt Request Registers
280	ERR_STAT	Error Status Register
300	ICRLO	Interrupt Command Register Low (bits 31–0)
310	ICRHI	Interrupt Command Register High (bits 63–32)
320	TIMER_LVT	Timer Local Vector Table Entry
330	THERMAL_LVT	Thermal Local Vector Table Entry
340	PERF_CNT_LVT	Performance Counter Local Vector Table Entry
350	LINT0_LVT	Local Interrupt 0 Local Vector Table Entry
360	LINT1_LVT	Local Interrupt 1 Local Vector Table Entry
370	ERROR_LVT	Error Local Vector Table Entry
380	INIT_CNT	Timer Initial Count Register
390	CURR_CNT	Timer Current Count Register

Table 60. APIC Register Summary (Continued)

3E0	TIMER_DVD_CFG	Timer Divide Configuration Register
400	EXT_APIC_FEAT	Extended APIC Feature Register
500	THRCNT_INT0_LVT	Threshold Count Interrupt 0 Local Vector Table Entry

7.8.1 APIC ID Register

APIC_ID Register — Offset 20h



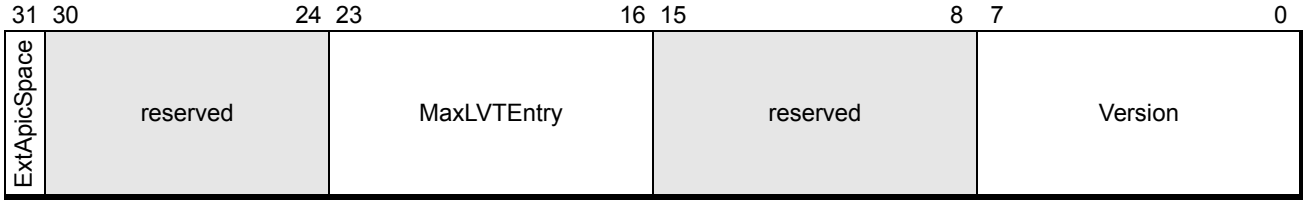
Bit	Name	Function	R/W	Reset
31–24	APICID	APIC Identification	R/W	Node ID
23–0	reserved		R/O	00_0000h

Field Descriptions

APIC Identification (APICID)—Bits 31–24. Software must ensure that all APICs are assigned unique APIC IDs. When both ApicExtId and ApicExtBrdCst in the HyperTransport™ Transaction Control Register are set, all 8 bits of APIC ID are used. When either ApicExtID or ApicExtBrdCst is clear, only bits 3–0 of APIC ID are used, and bits 7–4 are reserved. Node ID is initially 00h if this is the boot strap processor or 07h for all other nodes.

7.8.2 APIC Version Register

APIC_VER Register — Offset 30h



Bit	Name	Function	R/W	Reset
31	ExtApicSpace	Extended APIC Register Space Present	R/O	1h
30–24	reserved		R/O	00h
23–16	MaxLVTEntry	Maximum LVT Entry	R/O	04h
15–8	reserved		R/O	00h
7–0	Version	Version	R/O	10h

Field Descriptions

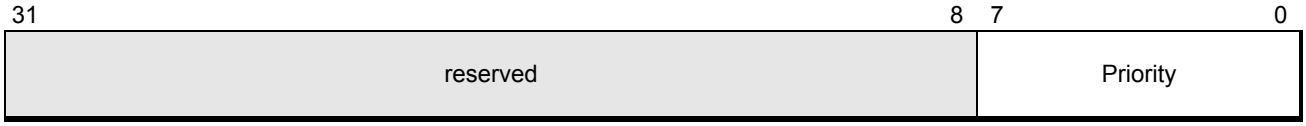
Extended APIC Register Space Present—Bit 31. This field indicates the presence extended APIC register space starting at 400h.

Max LVT Entry—Bits 23–16. This field indicates the number of entries in the Local Vector Table minus one.

Version—Bits 7–0. This field indicates the version number of this APIC implementation.

7.8.3 Task Priority Register

TPR_PRI Register — Offset 80h



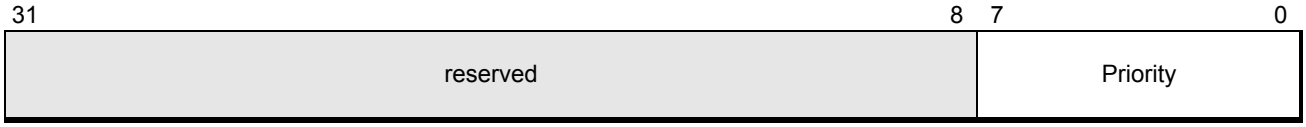
Bit	Name	Function	R/W	Reset
31–8	reserved		R/O	0000_00h
7–0	Priority	Priority	R/W	00h

Field Descriptions

Priority—Bits 7–0. This field is assigned by software to set a threshold priority at which the processor is interrupted.

7.8.4 Arbitration Priority Register

ARB_PRI Register — Offset 90h



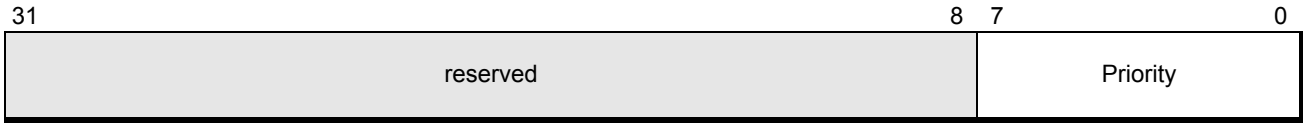
Bit	Name	Function	R/W	Reset
31–8	reserved		R/O	0000_00h
7–0	Priority	Priority	R/O	00h

Field Descriptions

Priority—Bits 7–0. This field indicates the processor's current priority, for a task being serviced, an interrupt being serviced, or an interrupt that is pending, and is used to arbitrate between processors to determine which accepts a lowest-priority interrupt request.

7.8.5 Processor Priority Register

PROC_PRI Register — Offset A0h



Bit	Name	Function	R/W	Reset
31–8	reserved		R/O	0000_00h
7–0	Priority	Priority	R/O	00h

Field Descriptions

Priority—Bits 7–0. This field indicates the processor's current priority servicing a task or interrupt, and is used to determine if any pending interrupts should be serviced.

7.8.6 End of Interrupt Register

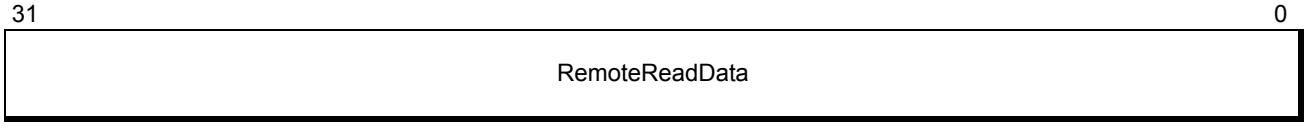
This register is written by the software interrupt handler to indicate that servicing of the current interrupt is complete.

EOI Register — Offset B0h

Bit	Name	Function	R/W	Reset
31–0	reserved		W/O	XXXX_XXXXh

7.8.7 Remote Read Register

RMT_RD Register — Offset C0h



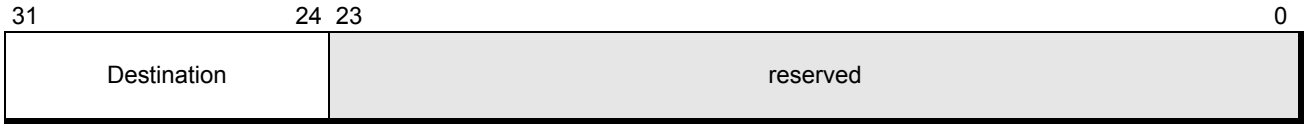
Bit	Name	Function	R/W	Reset
31–0	RemoteReadData	Remote Read Data	R/O	0000_0000h

Field Descriptions

Remote Read Data (RemoteReadData)—Bits 31–0. This field contains the data resulting from a valid completion of a Remote Read Inter-Processor Interrupt.

7.8.8 Logical Destination Register

LOG_DEST Register — Offset D0h



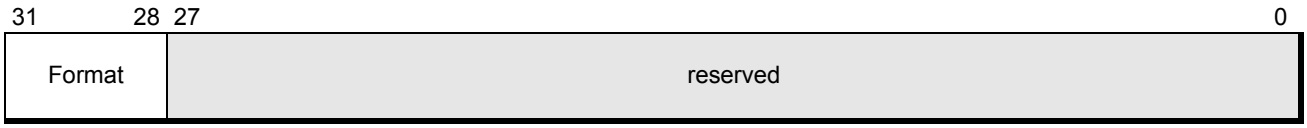
Bit	Name	Function	R/W	Reset
31–24	Destination	Destination	R/W	00h
23–0	reserved		R/O	00_0000h

Field Descriptions

Destination (Destination)—Bits 31–24. This field contains this APIC's destination identification, and is used to determine which interrupts should be accepted.

7.8.9 Destination Format Register

DEST_FORMAT Register — Offset E0h



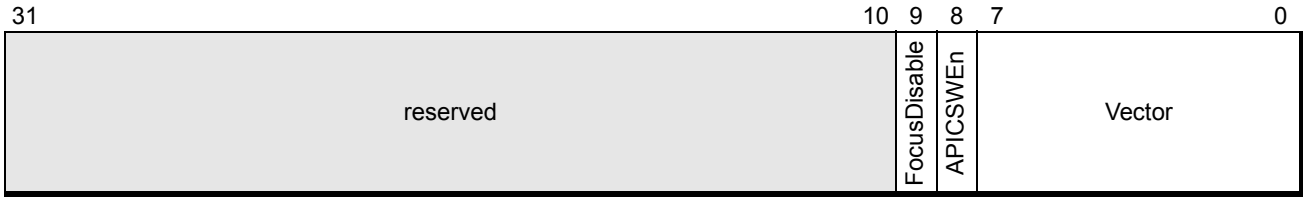
Bit	Name	Function	R/W	Reset
31–28	Format	Format	R/W	Fh
27–0	reserved		R/O	FFF_FFFFh

Field Descriptions

Format (Format)—Bits 31–28. 0h and Fh are the only allowed values. This field controls which format to use when accepting interrupts with a logical destination mode.
0h = Cluster destinations are used.
Fh = Flat destinations are used.

7.8.10 Spurious Interrupt Vector Register

SPUR_INTR_VEC Register — Offset F0h



Bit	Name	Function	R/W	Reset
31–10	reserved		R/O	0000_00h
9	FocusDisable	Focus Disable	R/W	0
8	APICSWEn	APIC Software Enable	R/W	0
7–0	Vector	Vector	R/W	FFh

Field Descriptions

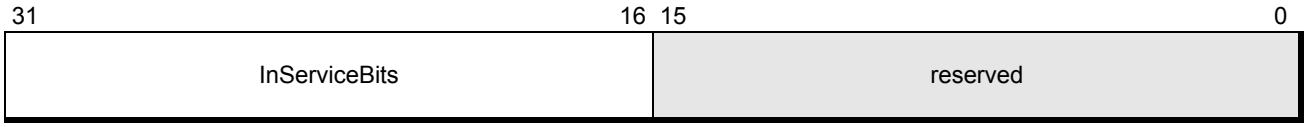
Focus Disable (FocusDisable)—Bit 9. This bit disables focus processor checking during lowest-priority arbitrated interrupts.

APIC Software Enable (APICSWEn)—Bit 8. When APICSWEnable is cleared, SMI, NMI, INIT, Startup, Remote Read, and LINT interrupts may be accepted, pending interrupts in the ISR and IRR are held, but further Fixed, Lowest-Priority, and ExtInt interrupts are not accepted. All LVT entry Mask bits are set and cannot be cleared.

Vector—Bits 7–0. When ApicExtSpur in the HyperTransport™ Transaction Control Register is set, bits 3–0 of Vector are writable. When ApicExtSpur is clear, bits 3–0 are read-only 1111b. This field contains the vector that is sent to the processor in the event of a spurious interrupt.

7.8.11 In-Service Registers

ISR Register — Offset 100h



Bit	Name	Function	R/W	Reset
31–16	InServiceBits	In-Service Bits	R/O	0000h
15–0	reserved		R/O	0000h

Field Descriptions

In-Service Bits (InServiceBits)—Bits 31–16. These bits are set when the corresponding interrupt is being serviced by the CPU.

ISR Registers — Offsets 170h, 160h, 150h, 140h, 130h, 120h, 110h

Bit	Name	Function	R/W	Reset
31–0	InServiceBits	In-Service Bits	R/O	0000_0000h

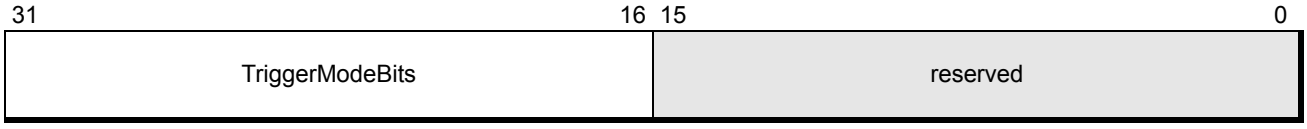
Field Descriptions

In-Service Bits (InServiceBits)—Bits 31–0. These bits are set when the corresponding interrupt is being serviced by the CPU. Interrupts are mapped as follows:

ISR	Interrupt number
Offset 110h	63–32
Offset 120h	95–64
Offset 130h	127–96
Offset 140h	159–128
Offset 150h	191–160
Offset 160h	223–192
Offset 170h	255–224

7.8.12 Trigger Mode Registers

TMR Register — Offset 180h

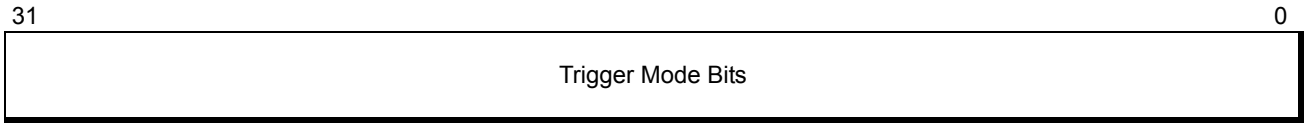


Bit	Name	Function	R/W	Reset
31–16	TriggerModeBits	Trigger Mode Bits	R/O	0000h
15–0	reserved		R/O	0000h

Field Descriptions

Trigger Mode Bits (TriggerModeBits)—Bits 31–16. The Trigger Mode bit for each corresponding interrupt is updated when an interrupt enters servicing. It is 0 for edge-triggered interrupts and 1 for level-triggered interrupts.

TMR Registers — Offsets 1F0h, 1E0h, 1D0h, 1C0h, 1B0h, 1A0h, 190h



Bit	Name	Function	R/W	Reset
31–0	TriggerModeBits	Trigger Mode Bits	R/O	0000_0000h

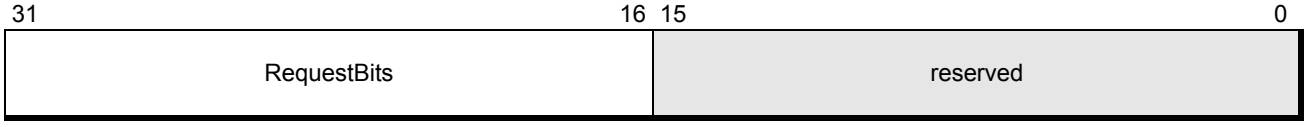
Field Descriptions

Trigger Mode Bits (TriggerModeBits)—Bits 31–0. The Trigger Mode bit for each corresponding interrupt is updated when an interrupt enters servicing. It is 0 for edge-triggered interrupts and 1 for level-triggered interrupts. Interrupts are mapped as follows:

TMR	Interrupt number
Offset 190h	63–32
Offset 1A0h	95–64
Offset 1B0h	127–96
Offset 1C0h	159–128
Offset 1D0h	191–160
Offset 1E0h	223–192
Offset 1F0h	255–224

7.8.13 Interrupt Request Registers

IRR Register — Offset 200h



Bit	Name	Function	R/W	Reset
31–16	RequestBits	Interrupt Requests Bits	R/O	0000h
15–0	reserved		R/O	0000h

Field Descriptions

Interrupt Request Bits (RequestBits)—Bits 31–16. Request bits are set when the corresponding interrupt is accepted by the APIC.

IRR Registers **Offsets 270h, 260h, 250h, 240h, 230h, 220h, 210h**

Bit	Name	Function	R/W	Reset
31–0	RequestBits	Interrupt Requests Bits	R/O	0000_0000h

Field Descriptions

Interrupt Request Bits (RequestBits)—Bits 31–0. Request bits are set when the corresponding interrupt is accepted by the APIC. Interrupts are mapped as follows:

IRR	Interrupt number
Offset 210h	63–32
Offset 220h	95–64
Offset 230h	127–96
Offset 240h	159–128
Offset 250h	191–160
Offset 260h	223–192
Offset 270h	255–224

7.8.14 Error Status Register

This register must be written to trigger an update before it can be read. Each write causes the internal error state to be loaded into this register, clearing the internal error state. A second write before another error occurs causes this register to be cleared.

ERR_STAT Register **Offset 280h**

Bit	Name	Function	R/W	Reset
31–8	reserved		R/O	0000_00h
7	IllegalRegAddr	Illegal Register Address	W/R	0
6	RcvdIllegalVector	Received Illegal Vector	W/R	0
5	SentIllegalVector	Sent Illegal Vector	W/R	0
4	reserved		R/O	0
3	RcvAcceptError	Receive Accept Error	W/R	0
2	SendAcceptError	Send Accept Error	W/R	0
1–0	reserved		R/O	00b

Field Descriptions

Illegal Register Address (IllegalRegAddr)—Bit 7. This bit indicates that an access to a nonexistent register location within this APIC was attempted.

Received Illegal Vector (RcvdIllegalVector)—Bit 6. This bit indicates that this APIC received a message with an illegal Vector (00h to 0Fh for fixed and lowest priority interrupts).

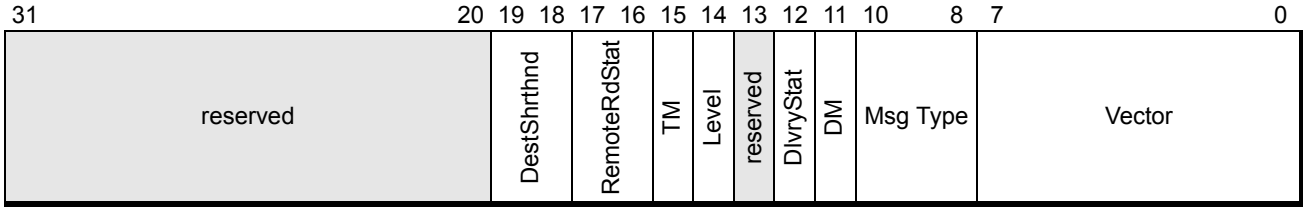
Sent Illegal Vector (SentIllegalVector)—Bit 5. This bit indicates that this APIC attempted to send a message with an illegal Vector (00h to 0Fh for fixed and lowest priority interrupts).

Receive Accept Error (RcvAcceptError)—Bit 3. This bit indicates that a message received by this APIC was not accepted by this or any other APIC.

Send Accept Error (SendAcceptError)—Bit 2. This bit indicates that a message sent by this APIC was not accepted by any APIC.

7.8.15 Interrupt Command Register Low

ICRLO Register — Offset 300h



Bit	Name	Function	R/W	Reset
31–20	reserved		R/O	000h
19–18	DestShrthnd	Destination Shorthand	R/W	00b
17–16	RemoteRdStat	Remote Read Status	R/O	00b
15	TM	Trigger Mode	R/W	0
14	Level	Level	R/W	0
13	reserved		R/O	0

Bit	Name	Function	R/W	Reset
12	DlvryStat	Delivery Status	R/O	0
11	DM	Destination Mode	R/W	0
10–8	MsgType	Message Type	R/W	000b
7–0	Vector	Vector	R/W	00h

Field Descriptions

Destination Shorthand (DestShrthnd)—Bits 19–18. This field provides a quick way to specify a destination for a message. If All Including Self or All Excluding Self are used, then DM is ignored and physical is automatically used.

00b =Destination Field
01b =Self
10b =All Including Self
11b =All Excluding Self (Note that this sends a message with a destination encoding of all 1's, so if lowest priority is used, the message could end up being reflected back to this APIC.)

Remote Read Status (RemoteRdStat)—Bits 17–16. The encoding for this field is as follows:

00b =Read was Invalid
01b =Delivery Pending
10b =Delivery Done and access was valid
11b =Reserved

Trigger Mode (TM)—Bit 15. This bit can be 0 for Edge or 1 for Level.

Level (Level)—Bit 14. This bit can be 0 for deasserted or 1 or asserted.

Delivery Status (DlvryStat)—Bit 12. This bit is set to indicate that the interrupt has not yet been accepted by the destination CPU(s).

Destination Mode (DM)—Bit 11. This bit can be 0 for Physical or 1 for Logical.**Message Type (MsgType)**—Bits 10–8. The encoding for this field is as follows:

000b = Fixed
001b = Lowest Priority
010b = SMI
011b = Remote Read
100b = NMI
101b = INIT
110b = Startup
111b = External Interrupt

Note: *Not all combinations of ICR fields are valid. Only those combinations listed in Table 61 are valid.*

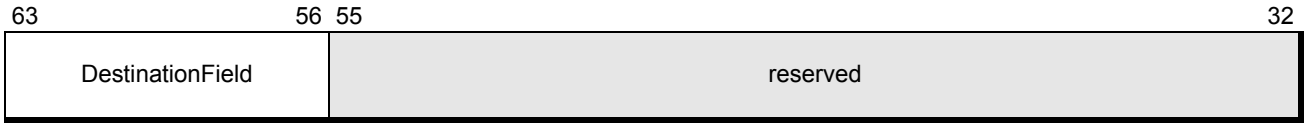
Table 61. Valid Combinations of ICR Fields

Message Type (MsgType)	Trigger Mode (TM)	Level (Lvl)	Destination Shorthand (DestShrthnd)
Fixed	Edge	X	X
	Level	Assert	X
Lowest Priority, SMI, NMI, INIT	Edge	X	Dest or All Excluding Self
	Level	Assert	Dest or All Excluding Self
Startup	X	X	Dest or All Excluding Self
Note: *X indicates a "don't care".*			

Vector (Vector)—Bits 7–0. This field contains the vector that is sent for this interrupt source.

7.8.16 Interrupt Command Register High

ICRHI Register — Offset 310h



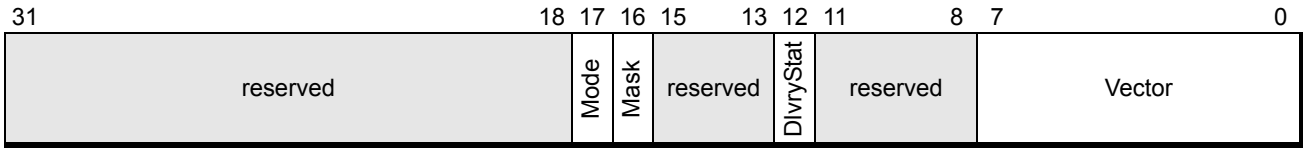
Bit	Name	Function	R/W	Reset
63–56	DestinationField	Destination Field	R/W	00h
55–32	reserved		R/O	00_0000h

Field Descriptions

Destination Field (DestinationField)—Bits 63–56. This field contains the destination encoding used when the destination shorthand is 00b.

7.8.17 Timer Local Vector Table Entry

TIMER_LVT Register — Offset 320h



Bit	Name	Function	R/W	Reset
31–18	reserved		R/O	0000h
17	Mode	Mode	R/W	0
16	Mask	Mask	R/W	1
15–13	reserved		R/O	000b
12	DlvryStat	Delivery Status	R/O	0
11–8	reserved		R/O	0h
7–0	Vector	Vector	R/W	00h

Field Descriptions

Mode (Mode)—Bit 17. This bit is 0 for One-shot and 1 for Periodic.

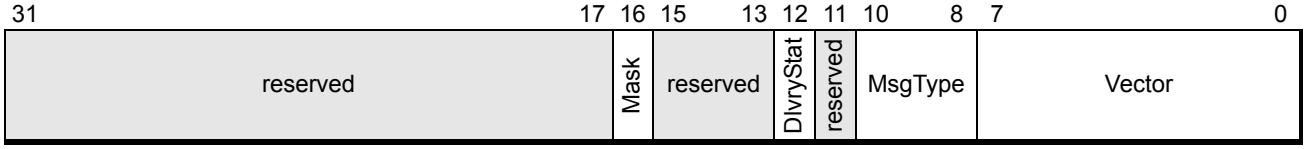
Mask (Mask)—Bit 16. If this bit is set, this LVT Entry does not generate interrupts.

Delivery Status (DlvryStat)—Bit 12. This bit is set to indicate that the interrupt has not yet been accepted by the CPU.

Vector (Vector)—Bits 7–0. This field contains the vector that is sent for this interrupt source.

7.8.18 Thermal Local Vector Table Entry

THERMAL_LVT Register — Offset 330h



Bit	Name	Function	R/W	Reset
31–17	reserved		R/O	0000h
16	Mask	Mask	R/W	1
15–13	reserved		R/O	000b
12	DlvryStat	Delivery Status	R/O	0
11	reserved		R/O	0
10–8	MsgType	Message Type	R/W	000b
7–0	Vector	Vector	R/W	00h

Field Descriptions

Mask (Mask)—Bit 16. If this bit is set, this LVT Entry does not generate interrupts.

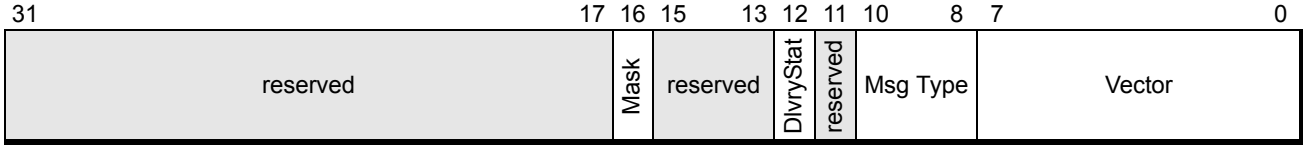
Delivery Status (DlvryStat)—Bit 12. This bit is set to indicate that the interrupt has not yet been accepted by the CPU.

Message Type (Msg Type)—Bits 10–8. Only Message Types 000b (Fixed), 010b (SMI) and 100b (NMI) are legal.

Vector (Vector)—Bits 7–0. This field contains the vector that is sent for this interrupt source.

7.8.19 Performance Counter Local Vector Table Entry

PERF_CNT_LVT Register — Offset 340h



Bit	Name	Function	R/W	Reset
31–17	reserved		R/O	0000h
16	Mask	Mask	R/W	1
15–13	reserved		R/O	000b
12	DlvryStat	Delivery Status	R/O	0
11	reserved		R/O	0
10–8	MsgType	Message Type	R/W	000b
7–0	Vector	Vector	R/W	00h

Field Descriptions

Mask (Mask)—Bit 16. If this bit is set, this LVT Entry does not generate interrupts.

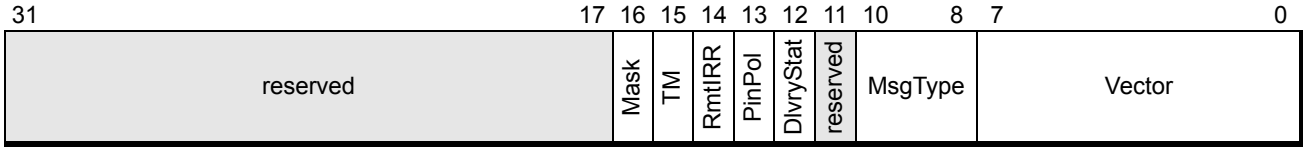
Delivery Status (DlvryStat)—Bit 12. This bit is set to indicate that the interrupt has not yet been accepted by the CPU.

Message Type (MsgType)—Bits 10–8. Message Types 000b (Fixed), 010b (SMI), and 100b (NMI) are legal.

Vector (Vector)—Bits 7–0. This field contains the vector that is sent for this interrupt source.

7.8.20 Local Interrupt 0 (Legacy INTR) Local Vector Table Entry Register

LINT0_LVT Register — Offset 350h



Bit	Name	Function	R/W	Reset
31–17	reserved		R/O	0000h
16	Mask	Mask	R/W	1
15	TM	Trigger Mode	R/W	0
14	RmtIRR	Remote IRR	R/O	0
13	PinPol	Pin Polarity	R/W	0
12	DlvryStat	Delivery Status	R/O	0
11	reserved		R/O	0
10–8	MsgType	Message Type	R/W	000b
7–0	Vector	Vector	R/W	00h

Field Descriptions

Mask (Mask)—Bit 16. If this bit is set, this LVT Entry does not generate interrupts.

Trigger Mode (TM)—Bit 15. This bit is set for level-triggered interrupts and clear for edge-triggered interrupts.

Remote IRR (RmtIRR)—Bit 14. If trigger mode is level, remote IRR is set when the interrupt has begun service. Remote IRR is cleared when the EOI has occurred.

Pin Polarity (PinPol)—Bit 13. This bit is not used because LINT interrupts are delivered by HyperTransport™ messages instead of individual pins.

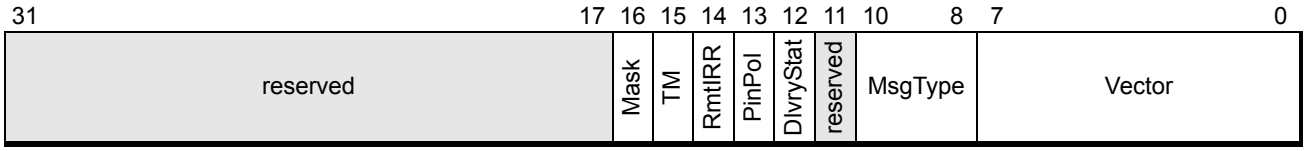
Delivery Status (DlvryStat)—Bit 12. This bit is set to indicate that the interrupt has not yet been accepted by the CPU.

Message Type (MsgType)—Bits 10–8. Only Message Types 000b (Fixed), 100b (NMI), and 111b (External Interrupt) are legal.

Vector (Vector)—Bits 7–0. This field contains the vector that is sent for this interrupt source.

7.8.21 Local Interrupt 1 (Legacy NMI) Local Vector Table Entry

LINT1_LVT Register — Offset 360h



Bit	Name	Function	R/W	Reset
31–17	reserved		R/O	0000h
16	Mask	Mask	R/W	1
15	TM	Trigger Mode	R/W	0
14	RmtIRR	Remote IRR	R/O	0

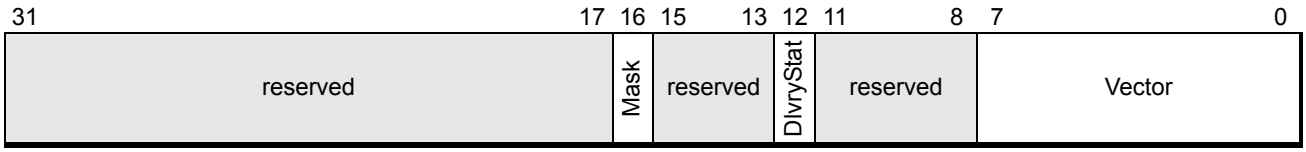
Bit	Name	Function	R/W	Reset
13	PinPol	Pin Polarity	R/W	0
12	DlvryStat	Delivery Status	R/O	0
11	reserved		R/O	0
10–8	MsgType	Message Type	R/W	000b
7–0	Vector	Vector	R/W	00h

Field Descriptions

These fields are the same as those for Local Interrupt 0.

7.8.22 Error Local Vector Table Entry

ERROR_LVT Register — Offset 370h



Bit	Name	Function	R/W	Reset
31–17	reserved		R/O	0000h
16	Mask	Mask	R/W	1
15–13	reserved		R/O	000b
12	DlvryStat	Delivery Status	R/O	0
11—8	reserved		R/O	0h
7–0	Vector	Vector	R/W	00h

Field Descriptions

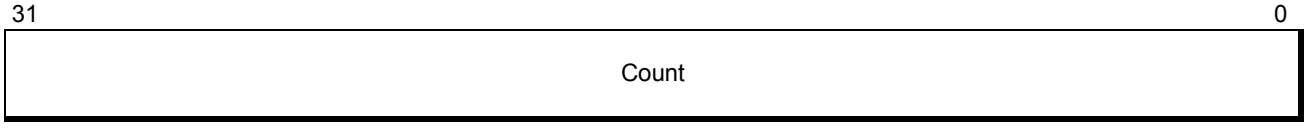
Mask (Mask)—Bit 16. If this bit is set, this LVT Entry does not generate interrupts.

Delivery Status (DlvryStat)—Bit 12. This bit is set to indicate that the interrupt has not yet been accepted by the CPU.

Vector (Vector)—Bits 7–0. This field contains the vector that is sent for this interrupt source.

7.8.23 Timer Initial Count Register

INIT_CNT Register — Offset 380h



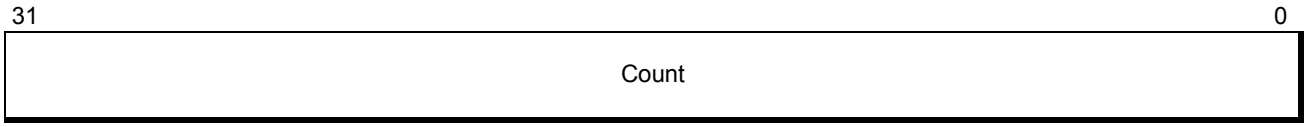
Bit	Name	Function	R/W	Reset
31–0	Count	Initial Count Value	R/W	0000_0000h

Field Descriptions

Count (Count)—Bits 31–0. This field contains the value copied into the current count register when the timer is loaded or reloaded.

7.8.24 Timer Current Count Register

CURR_CNT Register — Offset 390h



Bit	Name	Function	R/W	Reset
31–0	Count	Current Count Value	R/O	0000_0000h

Field Descriptions

Count (Count)—Bits 31–0. This field contains the current value of the counter.

7.8.25 Timer Divide Configuration Register

TIMER_DVD_CFG Register — Offset 3E0h

Bit	Name	Function	R/W	Reset
31–4	reserved		R/O	0000_00h
3	Div[3]	Div[3]	R/W	0
2	reserved		R/O	0
1–0	Div[1–0]	Div[1–0]	R/W	00b

Field Descriptions

Div[3] and Div[1–0]—Bits 3 and 1–0. The Div bits are encoded as follows:

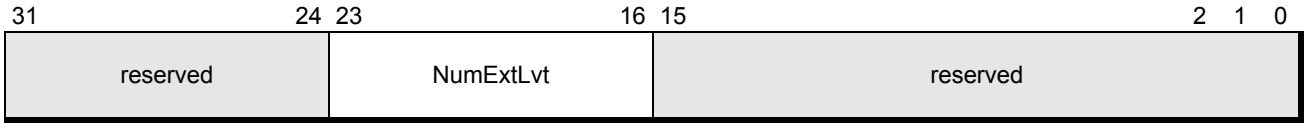
Div[3]	Div[1–0]	Resulting Timer Divide

0	00	2
0	01	4
0	10	8
0	11	16
1	00	32
1	01	64
1	10	128
1	11	1

7.8.26 Extended APIC Feature Register

See "Register Differences in Revisions of AMD NPT Family 0Fh Processors" on page 27 for revision information about this register.

EXT_APIC_FEAT Register — Offset 400h



Bit	Name	Function	R/W	Reset
31–24	reserved		R/O	0
23–16	NumExtLvt	Number of Extended LVTs Supported	R/O	1h
15–0	reserved		R/O	0

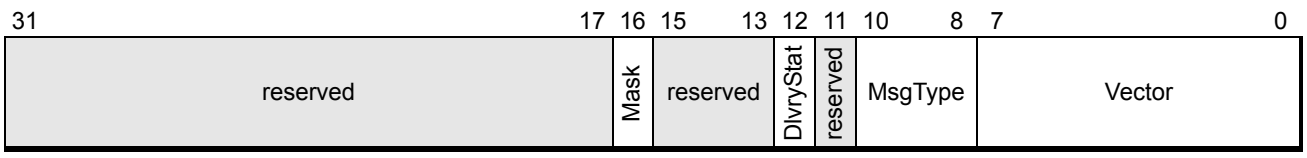
Field Descriptions

Number of Extended LVTs Supported (NumExtLvt)—Bits 23–16. This field indicates the number of extended LVT entries supported.

7.8.27 Threshold Count Interrupt 0 Local Vector Table Entry

This register enables and controls interrupts caused by the DRAM ECC threshold counter overflow in MC4_MISC (see "MC4_MISC—DRAM Errors Threshold Register" on page 241). See "Register Differences in Revisions of AMD NPT Family 0Fh Processors" on page 27 for revision information about this register.

THRCNT_INT0_LVT Register — Offset 500h



Bit	Name	Function	R/W	Reset
31–17	reserved		R/O	0000h
16	Mask	Mask	R/W	1
15–13	reserved		R/O	000b
12	DlvryStat	Delivery Status	R/O	0
11	reserved		R/O	0h
10–8	Msgtype	Message Type	R/W	0h
7–0	Vector	Vector	R/W	00h

Field Descriptions

Mask (Mask)—Bit 16. If this bit is set, this LVT Entry does not generate interrupts.

Delivery Status (DlvryStat)—Bit 12. This bit is set to indicate that the interrupt has not yet been accepted by the CPU.

Message Type (Msg Type)—Bits 10–8. Only Message Types 000b (Fixed), 010b (SMI), 100b (NMI) and 111b (ExtInt) are legal.

Vector (Vector)—Bits 7–0. This field contains the vector that is sent for this interrupt source.

8 System Management Mode (SMM)

System management mode (SMM) is an operating mode entered when system management interrupt SMI is asserted. The SMI is handled by a dedicated interrupt handler pointed to by processor model specific registers. SMM is used for system control activities such as power management. These activities are transparent to conventional operating systems (such as MS-DOS and Windows® operating system). SMM is used by the BIOS and specialized low level device drivers. The code and data for SMM are stored in the SMM memory area, which may be isolated from the main memory accesses using special processor functions.

This chapter describes the SMM state save area, entry into and exit from SMM, exceptions and interrupts in SMM, and memory allocation and addressing in SMM.

8.1 SMM Overview

The processor enters SMM after the system logic asserts the SMI interrupt and the processor recognizes SMI active. The processor may be programmed to send a special bus cycle to the system, indicating that it is entering SMM mode. The processor saves its state into the SMM memory state save area and jumps to the SMM service routine. The processor returns from SMM by executing the RSM instruction in the SMM service routine. The processor restores its state from the SMM state save area and resumes execution of the instruction following the point where it entered SMM. The processor may be programmed to send a special bus cycle to the system, indicating that it is exiting SMM mode.

The system logic asserting the SMI interrupt may be any I/O device connected to a noncoherent HyperTransport™ link. If more than one device exists in the system then the BIOS must take special care to ensure that all processors have entered SMM prior to accessing shared I/O resources and that all processor's SMI interrupt status bits are synchronized. See Section 8.13 on page 286 for more information on multiple system sources of SMIs.

The processor core also enters SMM after the Northbridge block of the processor asserts the core specific SMI interrupt and the processor core recognizes SMI active. This is a processor core-local SMI and is not recognized by other processor cores in the system. The BIOS must take special care to ensure that other processor cores have entered SMM prior to accessing shared I/O resources. See Section 8.13 on page 286 for more information on processor core local SMIs.

8.2 Operating Mode and Default Register Values

The software environment after entering SMM mode has the following characteristics:

- Addressing and operation in Real mode.
- 4-Gbyte segment limits.

- Default 16-bit operand, address, and stack sizes (instruction prefixes can override these defaults).
- Control transfers that do not override the default operand size truncate the EIP to 16 bits.
- Far jumps or calls cannot transfer control to a segment with a base address requiring more than 20 bits, like in Real mode segment-base addressing, unless a change is made into protected mode
- A20M# is disabled. A20M# assertion or deassertion have no effect on SMM mode.
- Interrupt vectors use the Real mode interrupt vector table.
- The IF flag in EFLAGS is cleared (INTR is not recognized).
- The TF flag in EFLAGS is cleared.
- The NMI and INIT interrupts are masked.
- Debug register DR7 is cleared (debug traps are disabled).

Figure 6 on page 273 shows the default map of the SMM memory area. It consists of a 64-Kbyte area, between 0003_0000h and 0003_FFFFh. The top 32 Kbytes (0003_8000–0003_FFFFh) must be populated with RAM. The default code-segment (CS) base address for the area (called the SMM_BASE address) is 0003_0000h. The top 512 bytes (0003_FE00–0003_FFFFh) are the SMM state save area. The default entry point for the SMM service routine is 0003_8000h.

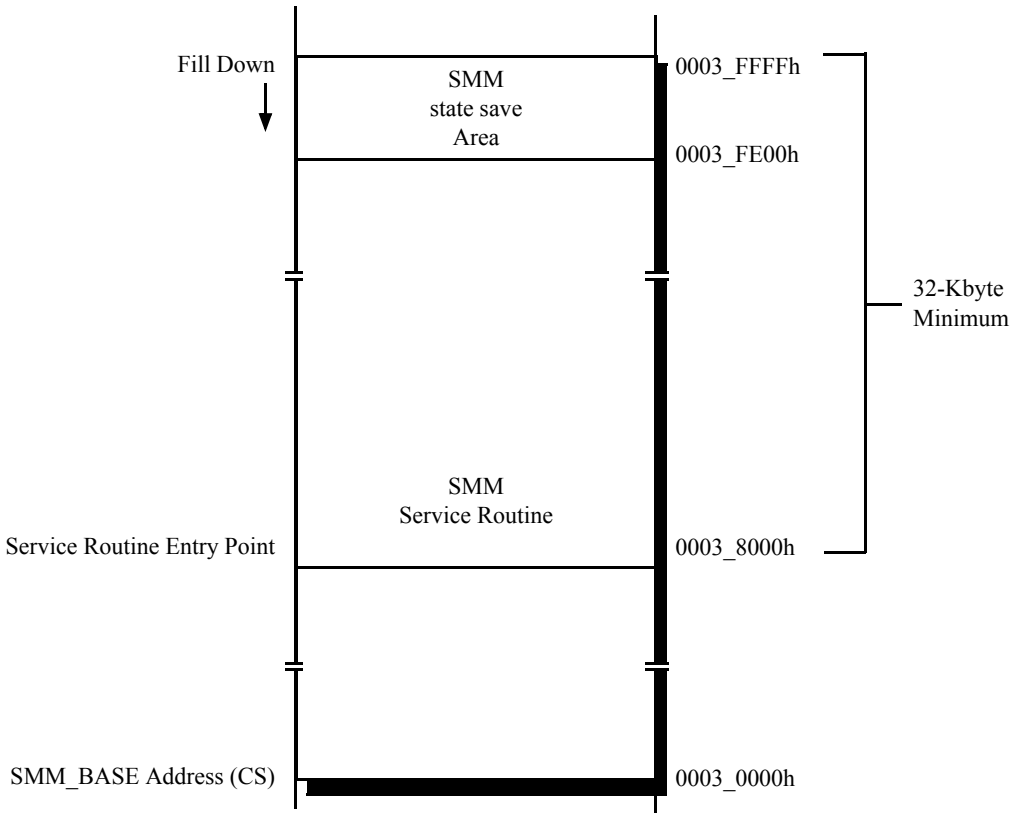


Figure 6. Default SMM Memory Map

8.3 SMM State Save Definition

Table 62. SMM Save State (Offset FE00–FFFFh)

Offset (Hex) from SMM_BASE[Note:]	Contents		Size	Allowable Access
FE00h	ES	Selector	Word	Read-Only
FE02h		Attributes	Word	
FE04h		Limit	Doubleword	
FE08h		Base	Quadword	
FE10h	CS	Selector	Word	Read-Only
FE12h		Attributes	Word	
FE14h		Limit	Doubleword	
FE18h		Base	Quadword	

Note: *The offset for the SMM-revision identifier is compatible with previous implementations.*

Table 62. SMM Save State (Offset FE00–FFFFh) (Continued)

Offset (Hex) from SMM_BASE[Note:]	Contents		Size	Allowable Access
FE20h	SS	Selector	Word	Read-Only
FE22h		Attributes	Word	
FE24h		Limit	Doubleword	
FE28h		Base	Quadword	
FE30h	DS	Selector	Word	Read-Only
FE32h		Attributes	Word	
FE34h		Limit	Doubleword	
FE38h		Base	Quadword	
FE40h	FS	Selector	Word	Read-Only
FE42h		Attributes	Word	
FE44h		Limit	Doubleword	
FE48h		Base	Quadword	
FE50h	GS	Selector	Word	Read-Only
FE52h		Attributes	Word	
FE54h		Limit	Doubleword	
FE58h		Base	Quadword	
FE60h–FE61h	GDTR	reserved	2 Bytes	Read-Only
FE62h		reserved	Word	
FE64h		Limit	Word	
FE66h–FE67h		reserved	2 Bytes	
FE68h		Base	Quadword	
FE70h	LDTR	Selector	Word	Read-Only
FE72h		Attributes	Word	
FE74h		Limit	Doubleword	
FE78h		Base	Quadword	
FE80h–FEB1h	IDTR	reserved	2 Bytes	Read-Only
FE82h		reserved	Word	
FE84h		Limit	Word	
FEB6h–FEB7h		reserved	2 Bytes	
FE88h		Base	Quadword	
FE90h	TR	Selector	Word	Read-Only
FE92h		Attributes	Word	
FE94h		Limit	Doubleword	
FE98h		Base	Quadword	

Note: *The offset for the SMM-revision identifier is compatible with previous implementations.*

Table 62. SMM Save State (Offset FE00–FFFFh) (Continued)

Offset (Hex) from SMM_BASE[Note:]	Contents	Size	Allowable Access
FEA0h–FEBFh	reserved	32 Bytes	—
FEC0h–FEC3h	SMM I/O Trap	Doubleword	Read-Only
FEC4h–FEC7h	reserved	4 Bytes	—
FEC8h	I/O Instruction Restart	Byte	Read/Write
FEC9h	Auto-Halt Restart	Byte	
FECAh	NMI Mask	Byte	
FECABh—FECFh	reserved	6 Bytes	—
FED0h	EFER	Quadword	Read-Only
FED8h—FEFBh	reserved	36 Bytes	—
FEFCh	SMM-Revision Identifier[1]	Doubleword	Read-Only
FF00h	SMM_BASE	Doubleword	Read/Write
FF04h—FF47h	reserved	68 Bytes	—
FF48h	CR4	Quadword	Read-Only
FF50h	CR3	Quadword	
FF58h	CR0	Quadword	
FF60h	DR7	Quadword	Read-Only
FF68h	DR6	Quadword	
FF70h	RFLAGS	Quadword	Read/Write
FF78h	RIP	Quadword	Read/Write
FF80h	R15	Quadword	
FF88h	R14	Quadword	
FF90h	R13	Quadword	
FF98h	R12	Quadword	
FFA0h	R11	Quadword	
FFA8h	R10	Quadword	
FFB0h	R9	Quadword	
FFB8h	R8	Quadword	

Note: *The offset for the SMM-revision identifier is compatible with previous implementations.*

Table 62. SMM Save State (Offset FE00–FFFFh) (Continued)

Offset (Hex) from SMM_BASE[Note:]	Contents	Size	Allowable Access
FFC0h	RDI	Quadword	Read/Write
FFC8h	RSI	Quadword	
FFD0h	RBP	Quadword	
FFD8h	RSP	Quadword	
FFE0h	RBX	Quadword	
FFE8h	RDX	Quadword	
FFF0h	RCX	Quadword	
FFF8h	RAX	Quadword	
Note: *The offset for the SMM-revision identifier is compatible with previous implementations.*			

After recognizing the SMI assertion, the processor saves almost the entire integer processor state to memory. Exceptions are: the floating point state, the model specific registers, and CR2. Any processor state not saved in the save state must be saved and restored by the SMM handler. With the AMD NPT Family 0Fh Processor extension of register size to 64-bits, the SMM save state has changed considerably compared to the legacy 32-bit version. One exception is the SMM Revision Identifier, which is located in the same position as before, for proper SMM identification.

Note: *The save state is always be 64-bit, regardless of the operating mode (32-bit or 64-bit).*

8.4 SMM Initial State

After storing the save state, the processor starts executing the handler at address SMM_BASE + 08000h. It starts with the entry state listed in Table 63 on page 277.

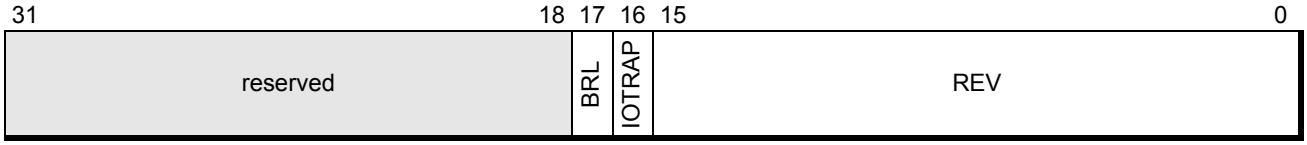
Table 63. SMM Entry State

Register	Initial Contents		
	Selector	Base	Limit
CS	SMM_BASE[19:4]	SMM_BASE[31:0]	4 Gbytes
DS	0000h	0000_0000h	4 Gbytes
ES	0000h	0000_0000h	4 Gbytes
FS	0000h	0000_0000h	4 Gbytes
GS	0000h	0000_0000h	4 Gbytes
SS	0000h	0000_0000h	4 Gbytes
General-Purpose Registers	Unmodified		
EFLAGS	0000_0002h		
EIP	0000_8000h		
CR0	Bits 0, 2, 3, and 31 cleared (PE, EM, TS, and PG); remainder is unmodified		
CR4	0000_0000h		
GDTR	Unmodified		
LDTR	Unmodified		
IDTR	Unmodified		
TR	Unmodified		
DR7	0000_0400h		
DR6	Undefined		

8.5 SMM-Revision Identifier

The SMM-Revision Identifier specifies the version of SMM and the processor extensions.

SMM-Revision Identifier **Offset FEFCh**



Bit	Name	Function	R/W	Reset
31–18	reserved	SBZ	R	
17	BRL	SMM Base Relocation Supported	R	1
16	IOTRAP	SMM I/O Trap Supported	R	1
15–0	REV	SMM Revision Level	R	0064h

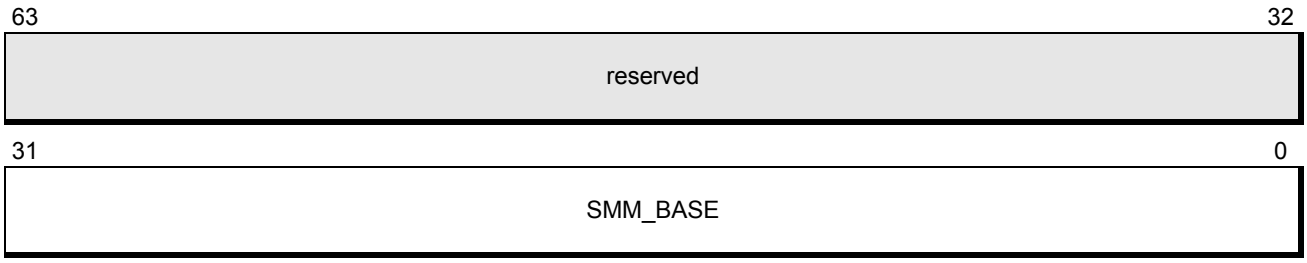
8.6 SMM Base Address

SMM_BASE register holds the base of the system management memory region, and its default value is 30000h. The value of this register is stored in the save state on entry into SMM and it is restored on returning from SMM. The first SMI handler instruction is fetched at SMM_BASE + 8000h. The 16 bit code segment selector is formed on entering SMM from SMM_BASE[19–4]. SMM_BASE[31–20] and SMM_BASE[3–0] have to be 0; if not, the CS base will not be equal to the (CS selector << 4), and attempts to load the CS_BASE into other descriptors will not work properly, since the selector cannot properly represent the base. If there is a far branch in the SMM handler, it will only be able to address the lower 1M of memory and will not be able to restore the SMM base to a value above 1M, unless it first switches to protected mode.

The SMM base address can be changed in two ways:

- The SMM base address, at offset FF00h in the SMM state save area, can be changed by the SMM service routine. The RSM instruction updates the SMM_BASE with the new value.
- The SMM_BASE is mapped to MSR C001_0111h and WRMSR instruction can be used to update it.

SMM_BASE Address **Offset FF00h**
MSR C001_0111h



Bit	Name	Function	R/W
63–32	reserved	RAZ	
31–0	SMM_BASE	System management mode base	R/W

8.7 Auto Halt Restart

During entry into SMM, the auto halt restart byte at offset FEC9h in the SMM state save area indicates whether SMM was entered from the Halt state. Before returning from SMM, the halt restart byte bit can be written by the SMM service routine to specify whether the return from SMM should take the processor back to the Halt state or to the instruction-execution state specified by the SMM state save area (the instruction after the HLT).

If the return from SMM takes the processor back to the Halt state, the HLT instruction is not refetched and reexecuted, but the Halt special bus cycle is sent to the system on the bus and the processor enters the HLT state.

Auto Halt Restart — Offset FEC9h

Bit	Name	Function	R/W
7–1	reserved	SBZ	R/W
0	HLT	HLT Restart Byte	R/W

Field Description

HLT Restart Byte (HLT)—Bit 0. On entry:

0 = Entered from normal x86 instruction boundary
1 = Entered from HLT State

After entry and before returning:

0 = Return to SMM State
1 = Return to HLT State

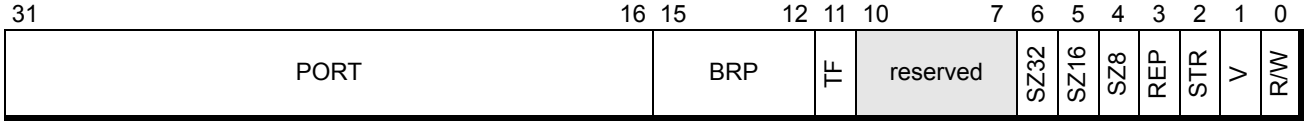
8.8 SMM I/O Trap and I/O Restart

If the assertion of SMI is recognized on the boundary of an I/O instruction, the I/O trap doubleword at offset FEC0h in the SMM state save area contains information about the executed I/O instruction. If the Valid bit is equal to 0, not all information is valid.

This is used in the I/O restart implementation. When an I/O access is done to a device that may be unavailable, the system can assert SMI and trap the I/O instruction. The system can then determine which device is not responding by using the I/O port information in the I/O Trap doubleword. It can then figure out why the device is not responding, fix the device, and re-execute the I/O instruction. It can reconstruct and then re-execute the I/O instruction in the SMM handler by using the SMM I/O trap information. More likely, it can use the SMM I/O restart mechanism to cause the processor to restart the I/O instruction immediately after the RSM.

8.8.1 SMM I/O Trap

SMM I/O Trap **Offset FEC0h**



Bit	Name	Function	R/W
31–16	PORT	Trapped I/O Port	R
15–12	BRP	I/O Breakpoint Matches	R
11	TF	EFLAGS TF Value	R
10–7	reserved	RAZ	R
6	SZ32	Port Access was 32-bit	R
5	SZ16	Port Access was 16-bit	R
4	SZ8	Port Access was 8-bit	R
3	REP	Repeated Port Access	R
2	STR	String Based Port Access	R
1	V	I/O Trap Word Valid Bit	R
0	R/W	Port Access Type 0 = Write (OUT instruction) 1 = Read (IN instruction)	R

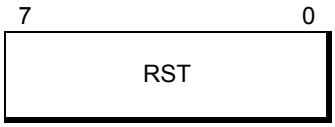
8.8.2 SMM I/O Restart Byte

The processor initializes the I/O trap restart slot to 00h on entry into SMM. If SMM is entered as a result of a trapped I/O instruction, the processor indicates the validity of the I/O instruction by setting or clearing bit 1 of the I/O trap doubleword at offset FEC0h in the SMM state save area. The SMM service routine should test bit 1 of the I/O trap doubleword to determine if a valid I/O instruction was being executed when entering SMM and before writing the I/O trap restart slot. If the I/O instruction is valid, the SMM service routine can safely rewrite the I/O trap restart slot with the value FFh, causing the processor to re-execute the trapped I/O instruction when the RSM instruction is executed. If the I/O instruction is invalid, writing the I/O trap restart slot has undefined results.

If a second SMI is asserted and a valid I/O instruction is trapped by the first SMM handler, the CPU services the second SMI prior to re-executing the trapped I/O instruction. The second entry into SMM does not have bit 1 of the I/O trap doubleword set, and the second SMM service routine must not rewrite the I/O trap restart slot.

During a simultaneous SMI I/O instruction trap and debug breakpoint trap, the processor first responds to the SMI and postpones recognizing the debug exception until after the RSM instruction returns from SMM. If the debug registers DR3–DR0 are used while in SMM, they must be saved and restored by the SMM handler. The processor automatically saves and restores DR7 and DR6. If the I/O trap restart slot in the SMM state save area contains the value FFh when the RSM instruction is executed, the debug trap does not occur until after the I/O instruction is re-executed.

SMM I/O Restart Byte **Offset FEC8h**



Bit	Name	Function	R/W
7–0	RST	I/O Restart Byte 00h = Do not restart FFh = Restart I/O instruction	R/W

8.9 Exceptions and Interrupts in SMM

When SMM is entered, the processor masks INTR, NMI, SMI, INIT, and A20M interrupts. The processor clears the IF flag to disable INTR interrupts. To enable INTR interrupts within SMM, the SMM handler must set the IF flag to 1. A20M is disabled so that address bit 20 is always generated externally when in SMM.

Generating an INTR interrupt is a method for unmasking NMI interrupts in SMM. The processor recognizes the assertion of NMI within SMM immediately after the completion of an IRET instruction. The NMI can be enabled by using an INTR interrupt. Once NMI is recognized within SMM, NMI recognition remains enabled until SMM is exited, at which point NMI masking is restored to the state it was in before entering SMM.

Because the IF flag is cleared when entering SMM, the HLT instruction should not be executed in SMM without first setting the IF bit to 1. Setting this bit to 1 enables the processor to exit the Halt state by means of an INTR interrupt.

The processor still responds to the DBREQ and STPCLK interrupts as well as to all exceptions that might be caused by the SMM handler.

8.10 NMI Mask

During entry into SMM, the NMI Mask byte at offset FECAh in the SMM state save area indicates whether NMI was masked when SMM was entered.

NMI Mask **Offset FECAh**

Bit	Name	Function	R/W
7–1	reserved	SBZ	R/W
0	NMI	NMI Mask	R/W

Field Description

NMI Mask (NMIM)—Bit 0. On entry:

0 = NMI not masked
1 = NMI Masked

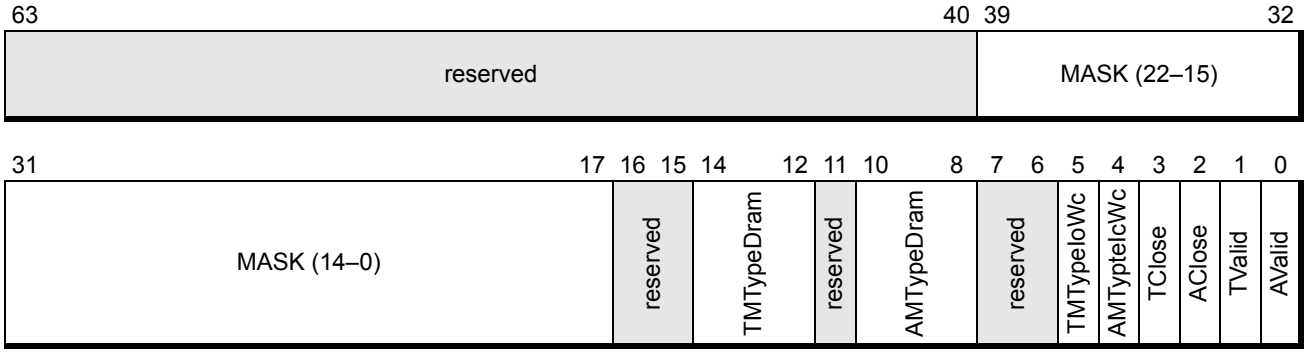
8.11 Protected SMM and ASeg/TSeg

System management memory, SMRAM, is defined by two ranges. The ASeg range is located at a fixed range from A0000h–BFFFFh. The TSeg range is located at a variable base. The size of the TSeg range is controlled by a variable mask. SMRAM provides a safe location for system management code and data that is not readily accessible by applications. The SMM interrupt handler can be located in one of these two ranges if protection is needed, or it can be located outside these ranges in the rest of memory.

8.11.1 SMM_MASK Register

This register holds the mask of the TSeg range as well as configuration information for both the fixed ASeg range and the variable TSeg range.

SMM_MASK Register **MSR C001_0113h**



Bit	Name	Function	R/W
63–40	reserved	RAZ	
39–17	MASK	SMRAM TSeg Range Mask	R/W
16–15	reserved	RAZ	
14–12	TMTypeDram	SMRAM TSeg Range Memory Type	R/W
11	reserved	RAZ	

Bit	Name	Function	R/W
10–8	AMTypeDram	SMRAM ASeg Range Memory Type	R/W
7–6	reserved	RAZ	
5	TMTypeIoWc	Non-SMRAM TSeg Range Memory Type	R/W
4	AMTypteIcWc	Non-SMRAM ASeg Range Memory Type	R/W
3	TClose	Send TSeg Range Data Accesses to Non-SMRAM	R/W
2	AClose	Send ASeg Range Data Accesses to Non-SMRAM	R/W
1	TValid	Enable TSeg SMRAM Range	R/W
0	AValid	Enable ASeg SMRAM Range	R/W

SMRAM TSeg Range Mask (MASK)—Bits 39–17. Mask of the SMRAM TSeg Range.

SMRAM TSeg Range Memory Type (TMTypeDram)—Bits 14–12. Memory Type for the SMRAM TSeg Range.

SMRAM ASeg Range Memory Type (AMTypeDram)—Bits 10–8. Memory Type for the SMRAM ASeg Range.

Non-SMRAM TSeg Range Memory Type (TMTypeIoWc)—Bit 5. Memory Type for the non-SMRAM TSeg Range.

Non-SMRAM ASeg Range Memory Type (AMTypteIcWc)—Bit 4. Memory Type for the non-SMRAM ASeg Range.

Send TSeg Range Data Accesses to Non-SMRAM (TClose)—Bit 3. Send Data Accesses to the TSeg Range to Non-SMRAM.

Send ASeg Range Data Accesses to Non-SMRAM (AClose)—Bit 2. Send Data Accesses to the ASeg Range to Non-SMRAM.

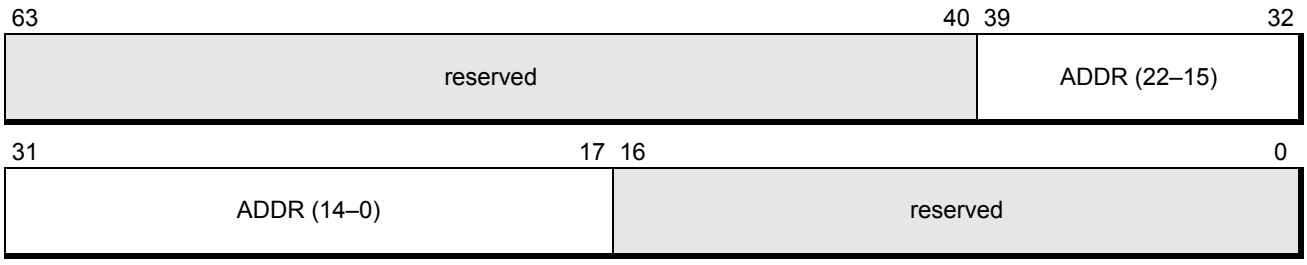
Enable TSeg SMRAM Range (TValid)—Bit 1. Enables the TSeg SMRAM Range.

Enable ASeg SMRAM Range (AValid)—Bit 0. Enables the ASeg SMRAM Range.

8.11.2 SMM_ADDR Register

This register holds the base of the variable TSeg range.

SMM_ADDR Register **MSR C001_0112h**



Bit	Name	Function	R/W
63–40	reserved	RAZ	
39–17	ADDR	Base of the SMRAM TSeg Range	R/W
16–0	reserved	RAZ	

8.11.3 SMM ASeg

The SMM ASeg address range is A0000–BFFFFh. The ASeg is enabled by the AValid bit in the SMM_MASK MSR.

When the ASeg is enabled, the MTRR memory maps are not used for the addresses in A0000–BFFFFh memory range. Instead, settings in the SMM_MASK register and whether the processor is in SMM or not determine how those addresses are handled. If not in SMM, the addresses are mapped to I/O space and do not access DRAM. The memory type used is either UC or WC, based on the AMTypeIoWc bit in the SMM_MASK register. When inside SMM, the accesses to the ASeg memory range are directed to DRAM.

The memory type is determined using the normal memory type encoding and is specified in AMTypeDram. Table 64 lists these types.

Table 64. SMM ASeg-Enabled Memory Types

		Out of SMM	
Address Range	**In SMM**	**AMTypeIoWc=0**	**AMTypeIoWc=1**
0–9FFFh	Normal MTRR	Normal MTRR	Normal MTRR
A0000–BFFFFh	DRAM, AMTypeDram	I/O, UC	I/O, WC
C0000h–Max	Normal MTRR	Normal MTRR	Normal MTRR

The SMM save state and code can be cached. After leaving SMM, the same addresses bypasses the caches and DRAM and accesses the I/O memory subsystem. This protects the SMM memory from corruption by programs outside of SMM.

8.11.4 SMM TSeg

The TSeg is similar to the ASeg except it provides more programmability and therefore allows more space to be allocated in a different area of memory for the SMM handler. It is enabled by the TValid bit in the SMM_MASK register. It uses the SMM_ADDR register to specify its base. It can be used with ASeg or by itself. When it is used with ASeg, the two spaces should not overlap. If they do overlap, the memory types should be consistent in both the ASeg map and the TSeg map for the overlapping addresses. Otherwise, undefined behavior occurs.

The SMM TSeg begins at the address specified in SMM_ADDR concatenated with 0s in the lower 17 bits, forcing the TSeg to begin on a 128-Kbyte boundary. The ending of TSeg is specified by the SMM_Mask[39:17]. The SMM_Mask and SMM_ADDR function as the variable-range MTRRs.

Each address generated by the processor is ANDed with the SMM_MASK and compared to the SMM_ADDR ANDed with the mask. If the values are equal, the address is within the TSeg range. For example, suppose you wish to create a TSeg starting at the 1-Mbyte address boundary and extending 256 Kbytes. The SMM_ADDR register would be set to 0010_0000h and the SMM_MASK to FFFC_0002h. This makes the TSeg range from 0010_0000–0013_FFFFh.

The TSeg memory type table is similar to the ASeg memory type table. When not in SMM and TSeg is enabled, the addresses within the TSeg range are directed to I/O space with either a UC memory type or a WC memory type, based on the TMTypeIoWc bit in the SMM_MASK register. When within SMM, the accesses are directed to DRAM with a memory type specified by TMTypeDram.

Table 65. SMM TSeg-Enabled Memory Types

Address Range	In SMM	Out of SMM	
		TMTypeIoWc=0	TMTypeIoWc=1
<TSeg Range	Normal MTRR	Normal MTRR	Normal MTRR
TSeg Range	DRAM, TMTypeDram	I/O, UC	I/O, WC
> TSeg Range	Normal MTRR	Normal MTRR	Normal MTRR

8.11.5 Closing SMM

Sometimes within SMM code with ASeg or TSeg enabled, there is a requirement to access the I/O space at the same address as the current SMM segment. That is typically only accessible outside of SMM. To accomplish this function, the Aclose and Tclose bits from SMM_MASK register are used. When the Aclose bit is set, data cache accesses to the ASeg that would normally go to DRAM are redirected to I/O, with the memory type specified by AMTypeIoWc.

The same function applies to the TSeg. Instruction cache accesses and Page Directory/Table accesses still access the SMM code in DRAM. When the SMM handler is done accessing the I/O space, it must clear the appropriate close bit. Failure to do so and then issuing an RSM will probably cause the processor to enter shutdown, as the save state is read from I/O space.

8.11.6 Locking SMM

The SMM registers can be locked by setting the SMMLOCK (HWCR, bit 0). Once set, the SMM_BASE, the SMM_ADDR, all but the two close bits of SMM_MASK and the RSMSPCYCDIS, SMISPCYCDIS, and SMMLOCK bits of HWCR are locked and cannot be changed. The only way to unlock the SMM registers is to assert reset. This provides security to the SMM mechanism. The BIOS can lock the SMM environment after setting it up so that it can not be tampered with.

8.12 SMM Special Cycles

Special cycles can be initiated on entry and exit from SMM to acknowledge to the system that these transitions are occurring. This allows secure spaces external to the processor to be secured within the SMM environment as well. It also helps with synchronizing the SMI interrupt across multiple processor systems. These special cycles can be disabled by setting the SMISPCYCDIS to 1 to disable the entry special cycle and setting the RSMSPCYCDIS bit to 1 to disable the exit special cycle.

8.13 MP-SMM Spring Boarding

When processor core-local SMI sources are enabled, or when more than one device in the system is enabled to signal SMI, BIOS must take special care to ensure that all cores have entered SMM prior to accessing shared I/O resources and all processor's SMI interrupt status bits are synchronized. The act of synchronizing cores into SMM is called spring boarding. MP-SMM spring boarding applies to dual-core uniprocessor systems (CMP-SMM) as well as symmetric multiprocessing systems (SMP-SMM).

An ACPI-compliant I/O hub is required for MP-SMM spring boarding. Depending on the I/O hub design, BIOS may have to set additional end-of-SMI bits to trigger an SMI from within SMM.

The software requirements for MP-SMM spring boarding are as follows.

- A binary semaphore located in SMRAM, accessible by all processors. For the purpose of this discussion, the semaphore is called CheckSpringBoard. CheckSpringBoard is initialized to zero.
- Two semaphores located in SMRAM, accessible by all processors. For the purpose of this discussion, the semaphores are called NotInSMM and WaitInSMM. NotInSMM and WaitInSMM are initialized to a value equal to the number of processor cores in the system (NumCPUs).

The following BIOS algorithm describes spring boarding and is optimized to reduce unnecessary SMI activity. This algorithm must be made part of the SMM instruction sequence for each processor core in the system.

1. Attempt to obtain ownership of the CheckSpringBoard semaphore with a read-modify-write instruction. If ownership was obtained then do the following, else proceed to step 2:
   - Check all enabled SMI status bits in the I/O hub.
     Let "Status"=enable1&status1 | enable2&status2 | enable3&status3…enable n&status n.
   - If "Status"=0 then perform the following sub-actions.
     - Trigger an SMI broadcast assertion from the I/O hub by writing to the software SMI command port.
     - Resume from SMM with the RSM instruction.

```
//Example:
InLineASM{
    BTS CheckSpringBoard,0    ;Try to obtain ownership of semaphore
```

```
    JC Step_2:
    CALL CheckIOHUB_SMIEVT  ;proc returns ZF=1 for no events
    JNZ Step_2:
    CALL Do_SpringBoard     ;Trigger SMI and then RSM
Step_2:
}
```

2. Decrement the NotInSMM variable. Wait for NotInSMM=0. *See Note 1*.
3. Execute the core-local event SMI handler. Using a third semaphore (not described here), synchronize processor core execution at the end of the task. After all processor cores have executed, proceed to step 4. The following is a brief description of the task for each processor core:
   – Check all enabled processor-core-local SMI status bits in the core's private or MSR address space. Handle the event if possible, or pass information necessary to handle the event to a mailbox for the boot strap processor to handle.
   – An exclusive mailbox must exist for each processor core for each core local event.
   – On-line spare events should be handled in this task by the individual core for optimal performance. Assign one core of a dual core processor to handle On-line spare. These events may be optionally handled by the BSP just as other global events.
   – Wait for all processor cores to complete this task at least once.
4. If the current processor core executing instructions is not the boot strap processor then jump to step 5. If the core executing instructions is the boot strap processor then jump to the modified main SMI handler task, described below.
   – Check all enabled SMI status bits in the I/O hub. Check mailboxes for event status.
   – For each event, handle the event and clear the corresponding status bit.
   – Repeat until all enabled SMI status bits are clear and no mailbox events remain.
   – Set NotInSMM=NumCPUs. (Jump to step 5.)
5. Decrement the WaitInSMM variable. Wait for WaitInSMM=0. *See Note 2.*
6. Increment the WaitInSMM variable. Wait for WaitInSMM=NumCPUs.
7. If the current processor core executing instructions is the boot strap processor then reset CheckSpringBoard to zero.
8. Resume from SMM with the RSM instruction.

Note:

1. *To support a secure startup by the secure loader the BIOS must provide a timeout escape from the otherwise endless loop. The timeout value should be large enough to account for the latency of all processor cores entering SMM. The maximum SMM entrance latency is defined by the platform's I/O sub-system, not the processor. AMD recommends a value of twice the watchdog timer count. See "MCA NB Configuration Register" for more information on the watchdog time-out value.*
   If a time-out occurs in the wait loop, the BIOS (the last core to decrement NotInSMM)

should record the number of cores that have not entered SMM and all cores must fall out of the loop.

2. *If a time-out occurs in the wait loop in step 2, the BIOS must not wait for WaitInSMM=0. Instead it must wait for WaitInSMM="the number of cores recorded in step 2".*

9 HyperTransport™ Technology Configuration and Enumeration

An AMD NPT Family 0Fh Processor is a node connected to other nodes with coherent HyperTransport™ links. Function 0 HyperTransport technology configuration register values for each node must be initialized. A node is identified in the Node ID register (Function 0, Offset 60h) with a number from 0 to 7. Routing tables are used by the node to decide whether to accept a packet and/or forward it through any or all of its HyperTransport links.

Three examples of coherent HyperTransport technology initialization sequence are shown in this chapter: 1-node initialization, 2-node initialization and generic initialization that can enumerate any number of nodes. This chapter also provides information on how to determine if a given routing table is valid.

9.1 Initial Configuration Steps

For coherent HyperTransport technology initialization, the following steps are required before enumeration:

1. Coherent HyperTransport Technology initialization is only performed by Node 0 (BSP). All the other nodes (APs) should bypass the HyperTransport initialization process.
2. Clearing RoutTblDis (Function 0, Offset 6Ch) enables the routing table for Node 0, and allows access to the memory controller on Node 0.
3. Node 0 is by definition connected to the HyperTransport I/O Hub, which means that Node 0 owns the compatibility chain. The link number that connects to HyperTransport I/O Hub should be written to SbLink (Function 0, Offset 64h).
4. Count coherent HyperTransport links on Node 0.

To perform the step 4 above, it is necessary to detect whether each HyperTransport link on a node is connected, and if the connected link type is coherent or noncoherent. The AMD Opteron™ processor supports up to three links, and the AMD Athlon™ 64 processor supports one link. For each HyperTransport link, the following steps are performed to detect the link connection status and type:

1. Check whether LinkConPend (Function 0, Offset 98h, D8h, B8h) is set.
2. If the LinkConPend is clear, check whether LinkCon (Function 0, Offset 98h, D8h, B8h) is set. If the Link Connected bit is set, the testing port is connected to other node.
3. Check whether InitComplete (Function 0, Offset 98h, D8h, B8h) is set.
4. If InitComplete is set, check NC (Function 0, Offset 98h, D8h, B8h) bit.
   The bit value 0 indicates a coherent HyperTransport link, while value 1 indicates a noncoherent link.

5. Record the number of coherent and noncoherent ports and their respective port numbers.

9.2 One-Node Coherent HyperTransport™ Technology Initialization

The number of nodes that exist in the system is determined on page 289. The following configuration steps should be executed in single (one-node) processor systems:

1. Set LimitCldtCfg (Function 0, Offset 68h) to enable limiting the extent of coherent HyperTransport configuration space based on the number of nodes in the system.
2. Disable read/write/fill probes in Function 0, Offset 68h for single-core one-node systems, since single processor systems do not need probes. For dual-core one-node systems, enable read/write/fill probes in Function 0, Offset 68h.

9.3 Two-Node Coherent HyperTransport™ Technology Initialization

For two-node systems, Node ID and Routing Tables need to be written for both nodes, as follows.

1. Initialize Node 0 Routing Table rows 0 and 1. Based on the coherent HyperTransport link number on Node 0.
2. Verify if Node 0-to-Node 1 link is established by reading Vendor ID/Device ID of Node 1.
3. Initialize Node 1 Routing Table row 0 and 1 using the coherent HyperTransport link number on Node 1.
4. Initialize the node ID of Node 1 by programming NodeId (Function 0, Offset 60h).
5. Write CPU Count and Node Count to CpuCnt and NodeCnt (Function 0, Offset 60h).
6. Set LimitCldtCfg (Function 0, Offset 68h) for Node 0 and Node 1.
7. Clear RouteTblDis (Function 0, Offset 6Ch) for Node 1.

9.4 Generic HyperTransport™ Technology Configuration

For multi-node systems, Node ID and Routing Tables need to be written for all nodes, as follows. This algorithm assumes the BIOS knows the topology of the nodes and how each link between nodes is routed on the system board.

1. Initialize Node 0 Routing Table rows 0 through N based on the coherent HyperTransport link numbers on Node 0.

2. For each node n in the system (starting with the nodes connected to Node 0 and working outward to the nodes furthest from Node 0) repeat steps a though e.
   a. Verify if Node 0-to-Node n link is established by reading Vendor ID/Device ID of Node n.
      – If the link node n is not established, disable the Routing Table on each node and go back to step 1 and build the Routing Tables for each node based on the new configuration.
   b. Poll the ReqDis bit (Function 0, Offset 6Ch) on Node n until set.
   c. Initialize Node n Routing Table row 0 through N. Using the coherent HyperTransport link numbers on Node n.
   d. Initialize the node ID of Node n by programming NodeId (Function 0, Offset 60h).
   e. Clear RouteTblDis (Function 0, Offset 6Ch) for Node 1.
3. Write CPU Count and Node Count to CpuCnt and NodeCnt (Function 0, Offset 60h) of each Node.
4. Set LimitCldtCfg (Function 0, Offset 68h) bit for each Node.

9.5 Rules for Valid Routing Tables

This section provides rules for determining if a given routing table is valid. A routing table is valid if and only if the routing table is deadlock free and probes are delivered only once.

A routing table is deadlock-free if it contains no Open Paths and no two-hop cycles.

An Open Path is a routing path between nodes that passes through one or more nodes that contains a subpath that is not a routing path in the routing table. For example if the routing path between Nodes 0 and 2 in Figure 7 was Node 0->Node 1->Node 3->Node 2 and the routing path between Nodes 3 and 2 was not Node 3->Node 2 then the routing path between Nodes 0 and 2 would be open because the subpath Node 3->Node 2 is not a path in the routing table.

A 2-hop cycle is a group of two hop routing paths (routing paths between two nodes that pass through a third node) such that the first and second nodes in the each two hop routing path are also the second and third nodes in a two hop routing path in the group.

9.5.1 2-Hop Cycle Example

Consider the four node configuration shown in Figure 7. A 2-hop cycle would occur in the configuration if the routing table was configured with the following routing paths:

- The routing path from Node 0 to Node 3 is: Node 0->Node 1->Node 3.
- The routing path from Node 1 to Node 2 is: Node 1->Node 3->Node 2.
- The routing path from Node 2 to Node 1 is: Node 2->Node 0->Node 1.
- The routing path from Node 3 to Node 0 is: Node 3->Node 2->Node 0.

To break this cycle at least one but no more than three of these routing paths must be modified to use a different intermediate node. Reconfiguring the routing paths as follows eliminates the 2-hop cycle.

- The routing path from Node 0 to Node 3 is: Node 0->Node 1->Node 3.
- The routing path from Node 1 to Node 2 is: Node 1->Node 3 to Node 2.
- The routing path from Node 2 to Node 1 is: Node 2->Node 0->Node 1.
- The routing path from Node 3 to Node 0 is: Node 3->Node 1->Node 0.

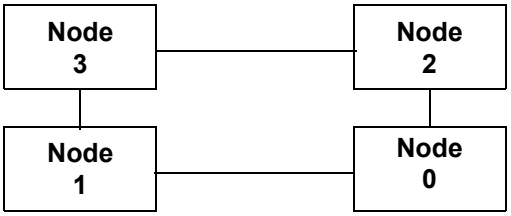


Figure 7. A Four-Node Configuration

10 Power and Thermal Management

AMD NPT Family 0Fh Processors support ACPI-compliant power management for all classes of systems from notebook PCs to multiprocessor servers. Table 66 lists various levels of power management. Table 67 on page 294 lists ACPI-state support by system class. This chapter covers processor-level power management and processor-specific aspects of system-level power management.

Table 66. Power Management Categories

Power Management Category	Comments
System Level	Coarse level power management applied to the entire system, typically after a predefined period of inactivity, or in response to a user action such as pressing the system power button. • System Sleep States (ACPI-defined S-states)
Processor Level	• Processor Power States (ACPI-defined C-states) • Processor Performance States (ACPI-defined P-states) • Software Transparent Power Management (Hardware enforced throttling).
Device Level	• Device Power States (ACPI-defined D-states) • Device Performance States • Software Transparent Power Management.
Bus Level	• Bus Power States • Software Transparent Power Management
Sub-system Level	• Typically not used in PC systems, and beyond the scope of this document.

Table 67. ACPI-State Support by System Class

ACPI State Support	Mobile Systems	Uniprocessor Desktop PCs	Multiprocessor Systems
G0/S0/C0: Working	Yes	Yes	Yes
G0/S0/C0: Processor performance state (P-state) transitions under OS control.	Yes	Yes	Yes
G0/S0/C0: Thermal clock throttling (I/O hub hardware enforced)	Yes	Yes	Yes
G0/S0/C0: Thermal clock throttling (operating system enforced)	Yes	Yes	No
G0/S0/C1: Halt	Yes	Yes	Yes
G0/S0/C2: Stop Grant Caches snoopable	Yes[1,2]	Revision F:No Revision G: Yes[1,2]	No
G0/S0/C3: Stop Grant Caches not snoopable	Yes[1,2]	Revision F:No Revision G: Yes[1,2]	No
G1/S1: Stand By (Powered On Suspend)	Yes	Yes	Yes
G1/S3: Stand By (Suspend to RAM)	Yes	Yes	Yes
G1/S4: Hibernate (Suspend to Disk)	Yes	Yes	Yes
G2/S5: Shut Down, Turn Off (Soft Off)	Yes	Yes	Yes
G3 Mechanical Off	Yes	Yes	Yes

Notes:

1. *See "Advanced Programmable Interrupt Controller (APIC)" on page 247.*
2. *Hardware can reach these states when both cores of a dual core processor are halted (C1E), but the operating system does not directly initiate C2 or C3 for dual core processors.*

10.1 Stop Grant

The processor and chipset use HyperTransport™ technology STPCLK and Stop Grant system management messages to sequence into all power management states except for Halt. These messages carry a 3-bit System Management Action Field (SMAF) to differentiate the various reasons for placing the processor into the Stop Grant state. Table 68 maps Stop Grant SMAF codes to ACPI states and actions that the BIOS is required to configure the processor to take for each ACPI state.

Table 68. Required SMAF Code to Stop Grant Mapping

Reason for STPCLK Message	SMAF Field of STPCLK and Stop Grant Messages	Mechanism that Forces the I/O Hub to Send the STPCLK Message	Processors Programmed Response after Entering the Stop Grant State (See Table 78 Power Management Control Registers.)
C2 Stop Grant Caches Snoopable	000b	Sent in response to a read of the ACPI-defined P_LVL2 register.	Ramp the CPU clock grid down when no probes need to be serviced. (CPU Low Power Enable)
C3 Stop Grant Caches not Snoopable. Used only by mobile systems.[2]	001b	Sent either in response to a read of the ACPI-defined P_LVL3 register or in response to a write to the Link Frequency Change and Resize LDTSTOP_L Command[1] register in the I/O hub.	Ramp the CPU clock grid frequency down. (CPU Low Power Enable) After LDTSTOP_L is asserted, place memory into self-refresh and ramp down the processor host bridge and memory controller clock grid. (Northbridge Low Power Enable)
VID/FID Change Or Link Width/ Frequency changes	010b	Sent either in response to a VID/FID Change special cycle from the processor or in response to a write to the Link Frequency Change and Resize LDTSTOP_L Command register.	After LDTSTOP_L assertion, place memory into self-refresh, ramp the processor clock grids down, then drive new FID values to PLL. (CPU Low Power Enable, Northbridge Low Power Enable, FID/VID Change Enable)
S1 Sleep state	011b	Sent in response to writing the S1 value to the SLP_TYP[2:0] field and setting the SLP_EN bit of the ACPI-defined PM1 control register in the I/O hub.	Same response as C3.
S3 Sleep state	100b	Sent in response to writing the S3 value to the SLP_TYP[2:0] field of the PM1 control register.	Same response as C3. Additionally, after LDTSTOP_L has been asserted, the I/O hub powers off the main power planes.
Throttling[3,4]	101b	Occurs based upon hardware initiated thermal throttling or ACPI-controlled throttling.	Ramp down the CPU grid by the programmed amount. (CPU Low Power Enable)

1. *GfxMode (Function3, Offset D8h) must be 0 when using this SMAF code to change the link Width/Frequency.*
2. *This SMAF code may be used in desktop and server systems to change link Width/Frequency. NBLowPwrEn and CpuLowPwrE must be 1 when using this SMAF code to change the link Width/Frequency.*
3. *AMD recommends using PROCHOT_L for thermal throttling and not implementing stop clock based throttling.*
4. *STPCLK throttling without ramping the CPU clock grid is supported. This will result in less power savings.*

Table 68. Required SMAF Code to Stop Grant Mapping (Continued)

Reason for STPCLK Message	SMAF Field of STPCLK and Stop Grant Messages	Mechanism that Forces the I/O Hub to Send the STPCLK Message	Processors Programmed Response after Entering the Stop Grant State (See Table 78 Power Management Control Registers.)
S4/S5	110b	Sent in response to writing the S4 or S5 value to the SLP_TYP[2:0] field of the PM1 control register.	Same response as S3 for processor. Additionally, all power is removed from processor during S4 and S5.
Reserved for use by processor	111b	No I/O hub STPCLK message uses this SMAF code. The power management register field corresponding to this SMAF code is used by the processor in response to executing the Halt instruction.	Same response as C2, except a Halt special cycle is broadcast.

1. *GfxMode (Function3, Offset D8h) must be 0 when using this SMAF code to change the link Width/Frequency.*
2. *This SMAF code may be used in desktop and server systems to change link Width/Frequency. NBLowPwrEn and CpuLowPwrE must be 1 when using this SMAF code to change the link Width/Frequency.*
3. *AMD recommends using PROCHOT_L for thermal throttling and not implementing stop clock based throttling.*
4. *STPCLK throttling without ramping the CPU clock grid is supported. This will result in less power savings.*

10.2 C-States

As Table 66 on page 293 indicates, the processor supports ACPI-defined processor power states, which are referred to as C-states. Table 67 on page 294 indicates which C-states are supported by system class. The operating system places the processor into C-states when the processor is idle an operating-system-determined percentage of the time.

10.2.1 C1 Halt State

C1 is the Halt state. C1 is entered after the HLT instruction has been executed. The BIOS configures the processor to enter a low-power state during C1, in which the CPU core clock grid is ramped down from its operating frequency. See Table 78 on page 328.

10.2.2 C2 and C3

C2 is the Stop Grant state in which the processor caches can be snooped. When the processor is in the Stop Grant state and there is no probe activity to the processor's caches, the CPU core clock grid is ramped down from its operating frequency. For more information, see Table 78 on page 328.

C3 is the Stop Grant state in which the processor's caches cannot be snooped. The chipset asserts LDTSTOP_L during the C3 state. The chipset may require BIOS to enable LDTSTOP_L assertion for the C3 state. When the processor is in the Stop Grant state, the CPU core clock grid is ramped

down from its operating frequency. After LDTSTOP_L assertion, the processor's system memory is placed into self-refresh mode and the host bridge/memory controller clock grid is ramped down from their operational frequency. For more information, see Table 78 on page 328.

10.2.3 C3 and AltVID

For mobile processors that use discrete graphics, the hardware can be programmed to drive an alternate VID code during the C3 processor power state. The purpose of the AltVID is to reduce the processor voltage to the minimum operational level while the processor is in the C3 state. AltVID settings can be determined by subtracting the AltVidOffset (MSRC001_0042) from the startup voltage.

10.2.4 C1 Enhanced (C1E) Halt State

C1E is a Stop Grant state like C3 supported by dual core mobile processors. The difference between C1E and C3 is that transition into C1E is not initiated by the operating system. The C1E state is initiated by an I/O transaction to the chipset when both cores transition into the C1 state. The chipset will either generate an SMI when receiving the I/O transaction.

10.2.4.1 BIOS Requirements for C1E

Chipsets that support an SMI command port can place the processor in the C1E state by generating an SMI when a write to the port is received. BIOS configures the CPU to generate the I/O write as follows:

- Program the Interrupt Pending Message register (MSR C001_0055h) in both cores with the address of the SMI command port in the chipset and an the eight bit message data that indicates to the SMM handler that the SMI was generated by both cores entering halt.
- Set the SmiOnCmpHalt bit and clear the IORd bit in MSR C001_0055h in both cores

The BIOS must not report C2 or C3 support to the OS. The BIOS must not enable C1E when the platform is not in ACPI power management mode.

10.2.4.1.1 C1E SMM Handler Requirements

The SMM handler in the BSP is responsible for handling the C1E code for both cores. The SMM handler on the AP should handshake with the BSP to determine when to exit and what state to exit into.

- The SMM handler should read the SMI command port to determine if the SMI was initiated because both cores went into halt.
- The SMM handler should read the Auto-Halt Restart byte for both cores to determine if both cores were in halt when entering SMM.
  - If the both cores were not in halt, the SMM handler should exit SMM.

- Set bit 1 (BM_RLD) of the ACPI PM1 Control register to enable the chipset to bring the cores out of C1E when bus master activity occurs.
- The SMM handler should then read the BM_STS bit to determine if bus master activity is occurring.
  - If the BM_STS bit is set, the SMM handler should clear the bit, look at the ACPI timer, log the time, and put both cores into C2 by reading the P_LVL2 register.
  - If the BM_STS bit is clear, the SMM handler should read the ACPI timer to see how long it has been since bus master activity last occurred.
  - If the BM_STS bit has been clear for less than 20ms, then the SMM handler should exit SMM mode, and put both cores into the C2 state.
  - If the BM_STS bit has been clear for more than 30ms, then the SMM handler should:
    - Set bit 0 (ARB_DIS) of the ACPI PM2 Control register to disable the system arbiter.
    - Read the P_LVL3 register in the chipset to force both cores into the C3 state.
- When C1E has been entered, the next timer tick interrupt or bus master activity should cause the chipset to de-assert LDTSTOP_L and issue a STPCLK deassertion message. The SMM handler should then exit both cores back to normal code execution.

10.3 Throttling

The BIOS must declare a DUTY_WIDTH value of zero in the Fixed ACPI Description table if the system does not support the _PSV object as part of a processor ACPI thermal zone.

Chipsets must not be configured to assert LDTSTOP_L during STPCLK throttling.

10.3.1 BIOS Requirements for PROCHOT_L Throttling

If PROCHOT_L throttling is being used by the system and HTC is not being used, the BIOS must program the HTC Register (Function 3, offset 64h) as follows to configure the processor for PROCHOT_L throttling.

- Program HtcEn=1b.
- Program HtcClkAct and HtcClkInact to 3 us (111b).
- Program HtcTmpLmt to 1_1111b.

10.4 Processor ACPI Thermal Zone

This section will be part of a future revision of this document.

10.5 Processor Performance States

Processor performance states (P-states) are valid operating combinations of processor core voltage and frequency. The hardware supporting P-state transitions in AMD processors is referred to as AMD PowerNow!™ technology for mobile systems and AMD Cool'n'Quiet™ technology for desktop systems. In this document all descriptions of AMD PowerNow!™ technology, software and driver apply to AMD Cool'n'Quiet™ technology, software, and driver.

- For operating systems without native support for P-state transitions (legacy operating systems), AMD PowerNow! software is used to perform P-state transitions. In this case, a BIOS generated Performance State Block (PSB) is used by AMD PowerNow! software to determine what P-states are supported by the processor in the system.
- For operating systems that have native support for P-state transitions, a processor-specific driver is used by the operating system to make P-state transitions. In this case, BIOS-provided ACPI-defined P-state objects inform the operating system that P-state transitions are supported by the system, which P-states are supported, and when given P-states are available for use by the operating system.

The valid P-states for each processor are defined using the P-state recognition algorithm. The BIOS must only use P-states defined for a given processor when building the ACPI P-state objects and PSB for use by higher-level software. BIOS is not required to implement all of the valid P-states for a processor.

MSR C001_0041h (FIDVID_CTL) and MSR C001_0042h (FIDVID_STATUS) are used for initiating P-state transitions and verifying that they are complete.

Chipsets provide support for P-state transitions. Refer to the chipset documentation for information on chipset configuration and P-state-related considerations.

10.5.1 BIOS Requirements for P-State Transitions

P-state transitions can be used only if they are supported by the processor and by the system.

BIOS is required to:

- Have a valid set of P-states for the processor, based on the P-state recognition algorithm.
- Take chipset or system limitations into account before providing the ACPI-defined P-state objects described in 10.6 on page 315, or the PSB described in 10.8 on page 325 of this document.
- BIOS is required to provide both:
  - ACPI-defined P-state objects for operating systems that support native P-state control.
  - a PSB in support of AMD PowerNow! software with legacy operating systems.

The BIOS must not provide the ACPI P-state objects or the PSB if:

- The chipset or system does not support P-state transitions.

- The BIOS does not have a set of valid P-states

10.5.1.1 P-state Recognition Algorithm

If CPUID extended function 8000_0007h returned EDX[2:1] = 11b, the BIOS must correlate the processor in the system to a valid set of P-states using this algorithm.

1. Determine Maximum (P[Max] or P[0]) and Minimum (P[Min]) P-states.
   - If MaxFID = 10_1010b and MaxVID != 00_0000b
     - P[Max] FID = start FID + 00_1010b (Start Frequency + 1GHz)
     - P[Max] VID = MaxVID + 00_0010b (Maximum Voltage - 50mV)
     - P[Min] FID = 00_0010b
     - P[Min] VID = start VID
   - else
     - P[Max] FID = MaxFID
     - P[Max] VID = MaxVID + 00_0010b
     - P[Min] FID = start FID
     - P[Min] VID = start VID
   - If P[Max] = P[Min] then BIOS should treat these as a single P-state.
2. Generate intermediate P-states based on the following rules:
   - No intermediate P-states are supported if IntPstateSup = 0b.
   - No intermediate P-state can have a VID > P[Min] VID.
   - No intermediate P-state can have a VID < P[Max] VID.
   - No intermediate P-state can have a FID < P[Min] FID + 00_1000b.
   - No intermediate P-state can have a FID >= P[Max] FID.
   - The highest intermediate P-state (P[1]) must have an even FID.
     - If P[Max] FID is odd: P[1] FID = P[Max] FID - 00_0001b
     - else: P[1] FID = P[Max] FID - 00_0010b
   - The P[1] VID is based on P[Max] VID and P[Min] VID.
     - If P[Min] VID - P[Max] VID <= 2^PstateStep: P[1] VID = P[Max] VID
     - Else if P[Max] FID is odd: P[1] VID = P[Max] VID + 00_0001b
     - else: P[1] VID = P[Max] VID + 2^PstateStep
   - The FID/VID for the current P-state (P[N]) is based on the FID/VID of the next highest P-state (P[N-1]).
     - If P[N-1] VID + 2^PstateStep >= P[Min]: P[N] = P[N-1]
     - else: P[N] = P[N-1] VID + 2^PstateStep
     - P[N] FID = P[N-1] FID - 00_0010b

3. Calculate the power for each P-state according to the following formula:
   - Power = (PwrLmt * P[N] Frequency * (P[N] Voltage^2))/(P[0] Frequency* (P[0] Voltage^2)).
     - The PwrLmt (CPUID Fn[8000_0001]_EBX[8:6,14]) field encodes the TDP for the processor. The TDP associated with the PwrLmt encoding can be determined from the appropriate Power and Thermal Datasheet.
     - If the BIOS does not contain a TDP for the PwrLmt encoding, the BIOS should increment the PwrLmt value by one until a valid TDP is found. This TDP should be used to calculate the power. If no valid TDP is found the BIOS must disable P-state support.

10.5.2 BIOS Requirements for P-State Transitions in a Multiprocessor System

In a multiprocessor system, P-state transitions can only be used if they are supported by all processors.

In a multiprocessor system, the BIOS is required to:

- Have a valid set of P-states for all processor. BIOS should follow the steps outlined in 10.5.1.1 on page 300.
- Take chipset or system limitations into account before providing the ACPI-defined P-state objects.
- Provide the same number of P-states for all processors, as per Section 8.3.3 of the ACPI 2.0 specification.
- Ensure that all processor performance states in the _PSS object have identical frequency and power-consumption parameters as per Section 8.3.3 of the ACPI 2.0 specification.
- Transition all processors to a voltage that is equal to or greater than the recommended voltage for the current frequency (valid voltage and frequency combinations are determined using the P-state recognition algorithm).
- Transition all processors to the maximum frequency of the slowest processor.
- Ensure that the frequency of each processor is one of the valid P-state frequencies.
- Copy the CurrVID (MSR C001_0042h, bits 36-32) value to NewVID (MSR C001_0041h, bits 12-8) for each processor.
- Copy the CurrFID (MSR C001_0042h, bits 5-0) value to NewFID (MSR C001_0041h, bits 5-0) for each processor.
- Set StpGntTOCnt (MSR C001_0041h, bits 51-32) to a value of 1h for each processor.
- Set the Clock Power/Timing High register (Function 3, Offset D8h) to a value of 2000_2710h for each processor.

In a multiprocessor system, the BIOS must not provide the ACPI P-state objects if:

- Any of the processors cannot be identified.
- The chipset or system does not support P-state transitions.
- The BIOS does not have a set of valid P-states.

In a multiprocessor system, the BIOS should follow the P-state transitioning algorithm for each processor as defined in 10.5.6 on page 303.

10.5.3 BIOS Requirements for P-State Transitions in Systems Using Dual Core Processors

- For systems with a single processor and unbuffered DIMMs or SODIMMs, BIOS configures the processor in the same way as described in Section 10.5.1 with the exception that one set of the _PCT, _PSS, and _PPC ACPI objects must be declared for each processor core.
- For systems with a multiple processors, or systems with Registered DIMMs, BIOS declares one set of _PCT, _PSS, and _PPC ACPI objects per processor core, and configures the processor as described in Section 10.5.2.

10.5.4 BIOS-Initiated P-State Transitions

All processors power up in the minimum P-state. BIOS can transition processors from the minimum P-state to a higher P-state during the POST routine. If BIOS performs a P-state transition it must follow the P-state transition algorithm defined in 10.5.7 on page 303. If the BIOS performs P-state transitions during POST on dual core processor, BIOS must enable core 1 by setting the Cpu1En bit (Function 0, Offset 68h) before initiating the P-state change.If the BIOS performs P-state transitions during POST in a multiprocessor system, BIOS must enable all processors by setting the ReqDis bit (Function 0, Offset 6Ch) before initiating the P-state change.

The BIOS must never initiate P-state transitions after passing control to the operating system.

10.5.5 BIOS Support for Operating System/CPU Driver-Initiated P-State Transitions

Operating systems that have native control for processor P-states exclusively use the presence of BIOS provided ACPI 2.0 defined processor P-state objects to determine if processor P-states are supported in a system. If the ACPI objects do not exist, then the processor driver is not loaded.

In operating systems that do not have native support for processor P-state transitions, an AMD defined PSB provided by the BIOS informs AMD PowerNow! technology software that P-state transitions are supported.

Once it has been determined that a processor and system support P-state transitions, the BIOS must provide both:

- ACPI-2.0-defined processor P-state objects for operating systems with native P-state support. See Section 10.6 on page 315.
- AMD-defined PSB for use with AMD PowerNow! software and operating systems that do not have native P-state support. See Section 10.8 on page 325.

10.5.6 Processor Driver Requirements

The processor driver that supports native operating system policy and control of P-state objects is required to use the ACPI 2.0 P-state objects as defined in this document. The processor driver is not permitted to use the legacy PSB tables described in section 10.8 on page 325.

10.5.7 P-State Transition Sequence

This section describes the P-state transition algorithm for AMD NPT Family 0Fh Processors.

10.5.7.1 FID to VCO Frequency Relationship

Table 69 and Table 70 provide the core frequency-to-VCO frequency relationship for AMD NPT Family 0Fh Processors. These processors do not support frequency transitions in VCO frequency steps greater than 200 MHz. The processor driver, and the AMD PowerNow! driver use these tables when making P-state transitions.

Only one P-state is allowed in Table 69. The one P-state consists of a frequency from the "Minimum Core Frequency" column in Table 69 and is the minimum P-State supported by the processor.

Table 69 additionally defines Portal Core Frequencies. The "Portal Core Frequencies in Table 70" column of Table 69 defines:

- The core frequencies in Table 70 to which direct transitions can be made from the minimum core frequency in Table 69.
- The core frequencies in Table 70 that support transitions to the minimum core frequency in Table 69.

The processor supports direct transitions between the minimum core frequency in Table 69 and any of the core frequencies from Table 70 listed in the Portal Core Frequencies column associated with the minimum core frequency.

If the minimum P-state from the low table (Table 69) has a VCO frequency > 1600 MHz, then the VCO/core frequency of all P- states in the high table (Table 70) must be >= the VCO frequency of the minimum P-state minus 200 MHz.

Example 1:

Given a processor with a minimum core frequency of 800 MHz and a maximum core frequency of 2000 MHz, to transition from 800 MHz to 2000 MHz, the processor driver will (in the order specified):

1. Transition the processor from the 800 MHz core frequency to 1800 MHz core frequency. 1800 MHz is a portal frequency to and from 800 MHz as defined in Table 69, because the VCO frequency change from an 800 MHz core frequency to an 1800 MHz core frequency is <= 200 MHz.
2. Transition the processor core frequency from 1800 MHz to 2000 MHz (VCO frequency change is <= 200 MHz according to Table 70).

Note: *To simplify this example, the voltage transitions and isochronous relief times are not described in this example.*

Example 2:

Given a processor with a minimum core frequency of 1400 MHz and a maximum core frequency of 3400 MHz, to transition from 1400 MHz to 3400 MHz, the processor driver will (in the order specified):

1. Transition the processor from the core frequency of 1400 MHz to a core frequency of 3000 MHz. 3000 MHz is a portal frequency to and from 1400 MHz as defined in Table 69, because the VCO frequency change from 1400 MHz core frequency to 3000 MHz core frequency is <= 200 MHz.
2. Transition the processor core frequency from 3000 MHz to 3200 MHz (VCO frequency change is <= 200 MHz per Table 70).
3. Transition the processor core frequency from 3200 MHz to 3400 MHz (VCO frequency change is <= 200 MHz per Table 70).

Note: *Note: to simplify this example, the voltage transitions and isochronous relief times are not described in this example.*

Table 69. Low FID Frequency Table (< 1600 MHz)

FID[5:0]	Minimum Core Frequency MHz	VCO Frequency MHz	Portal Core Frequencies in Table 70
000000b	800	1600	1600, 1700, 1800
000001b	900	1800	1600, 1700, 1800, 1900, 2000
000010b	1000	2000	1800, 1900, 2000, 2100, 2200
000011b	1100	2200	2000, 2100, 2200, 2300, 2400
000100b	1200	2400	2200, 2300, 2400, 2500, 2600
000101b	1300	2600	2400, 2500, 2600, 2700, 2800
000110b	1400	2800	2600, 2700, 2800, 2900, 3000
000111b	1500	3000	2800, 2900, 3000, 3100, 3200
Note: *Odd FID values are supported in revision G and later revisions.*			

Table 70. High FID Frequency Table (>= 1600 MHz)

FID[5:0]	Core Frequency MHz	VCO Frequency MHz
001000b	1600	1600
001001b	1700	1700
001010b	1800	1800
001011b	1900	1900
001100b	2000	2000
001101b	2100	2100
001110b	2200	2200
001111b	2300	2300
010000b	2400	2400
010001b	2500	2500
010010b	2600	2600
010011b	2700	2700
010100b	2800	2800
010101b	2900	2900
010110b	3000	3000
010111b	3100	3100
011000b	3200	3200
011001b	3300	3300
011010b	3400	3400
011011b	3500	3500
011100b	3600	3600
011101b	3700	3700
011110b	3800	3800
011111b	3900	3900
100000b	4000	4000
100001b	4100	4100
100010b	4200	4200
100011b	4300	4300
100100b	4400	4400
100101b	4500	4500
100110b	4600	4600
100111b	4700	4700
101000b	4800	4800

Note: *Odd FID values are supported in revision G and later revisions.*

Table 70. High FID Frequency Table (>= 1600 MHz) (Continued)

FID[5:0]	Core Frequency MHz	VCO Frequency MHz
101001b	4900	4900
101010b	5000	5000
Note: *Odd FID values are supported in revision G and later revisions.*		

10.5.7.2 P-state Transition Algorithm

The P-state transition algorithm has three phases. During phase 1 the processor voltage is transitioned to the level required to support frequency transitions. During phase 2 the processor frequency is transitioned to frequency associated with the OS-requested P-state. During phase 3 the processor voltage is transitioned to the voltage associated with the OS-requested P-state.

Figure 8 shows the high-level P-state transition flow. Figure 9 is a high-level timing diagram depicting voltage and frequency stepping associated with each phase of a P-state transition. Figure 10 shows the core voltage transition flow for phase 1 of a P-state transition. Figure 12 shows the core frequency transition flow for phase 2 of a P-state transition. Figure 15 shows the core voltage transition flow for phase 3 of a P-state transition.

The algorithm shown in Figure 8 through Figure 15 describes the 3-phases of a P-state transition in the context of Windows® XP and the associated processor driver, but the algorithm also applies to the AMD PowerNow! driver and any other software that performs P-state transitions (such as a debug tool).

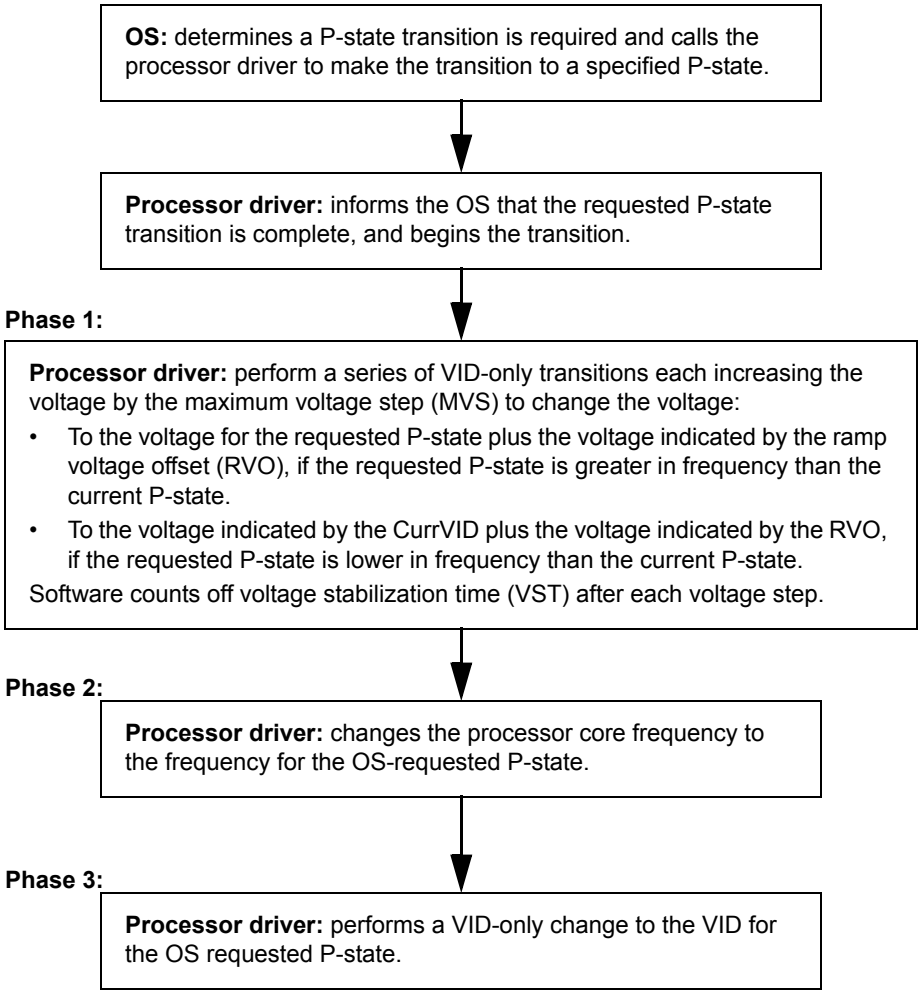


Figure 8. High-Level P-state Transition Flow

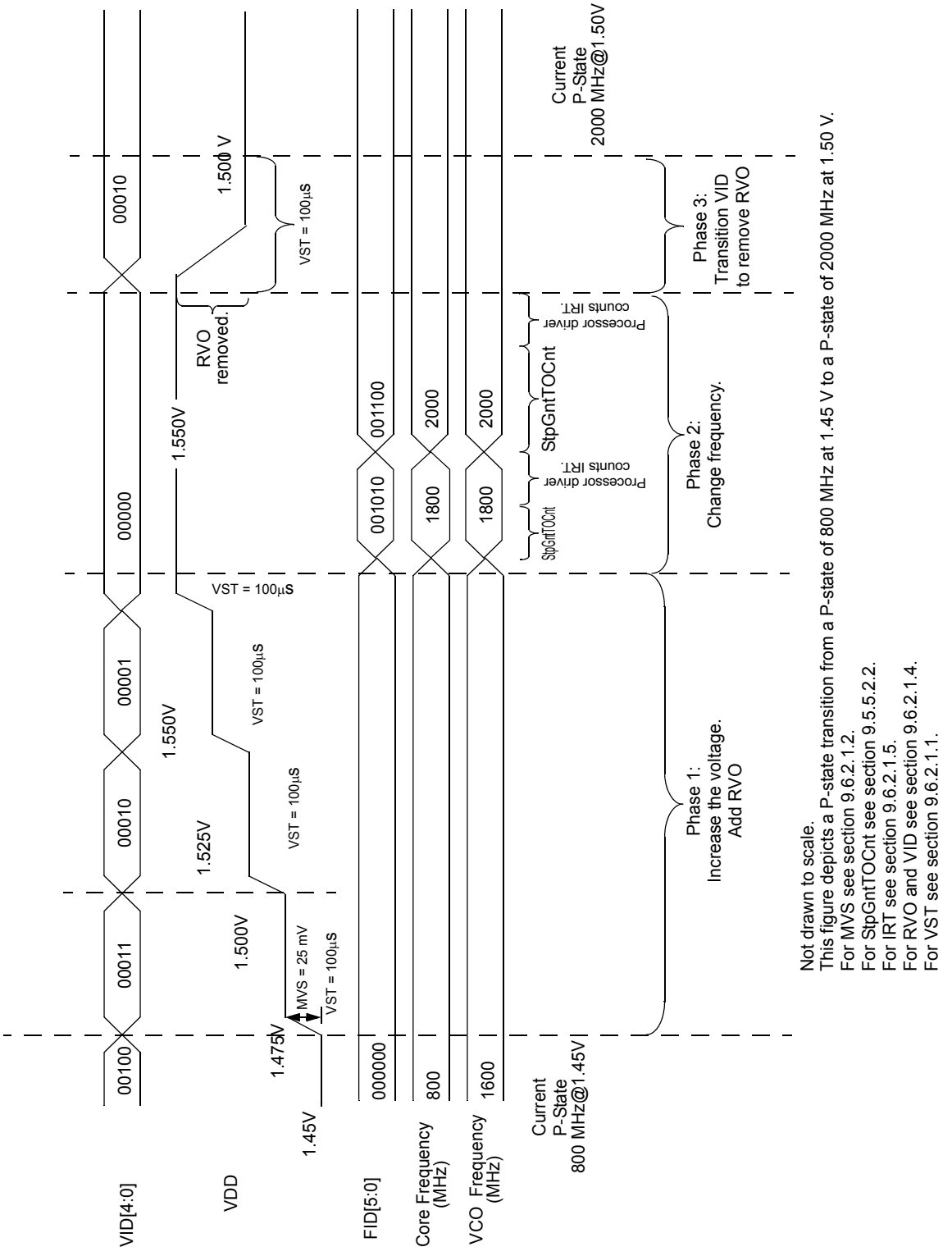


Figure 9. Example P-State Transition Timing Diagram

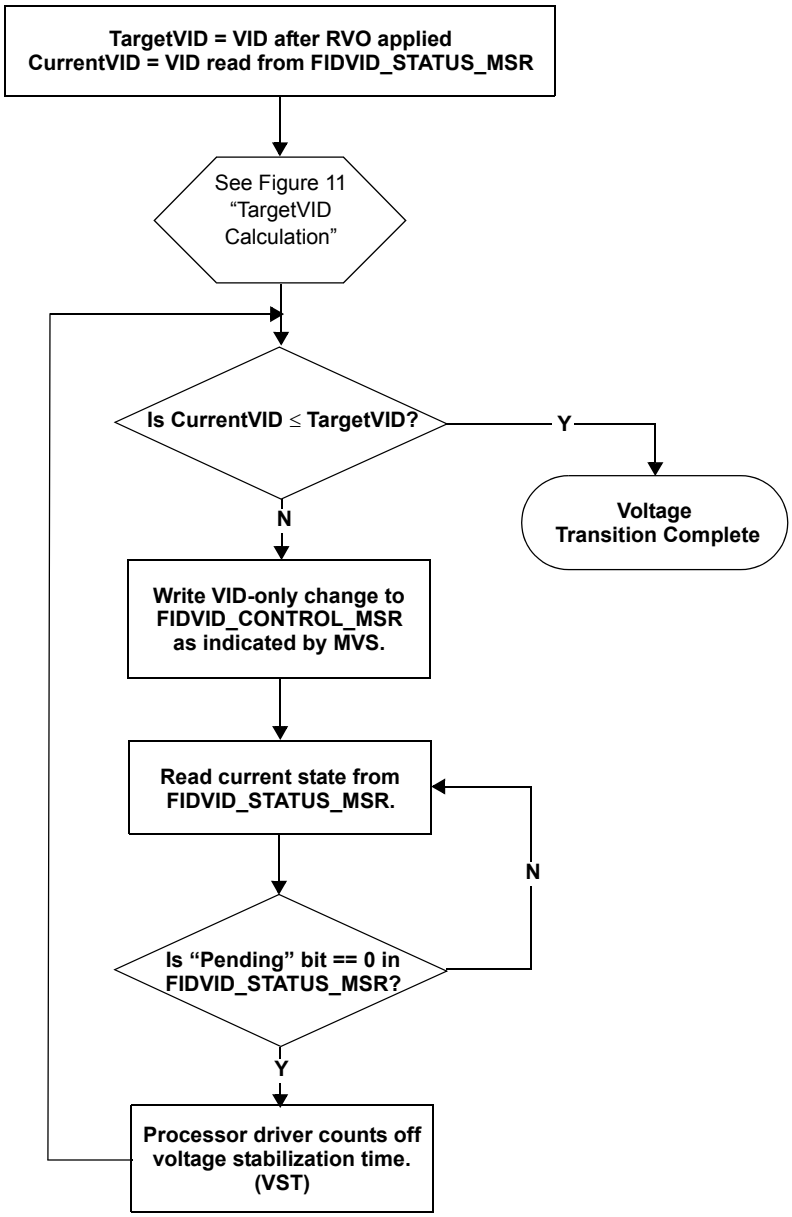


Figure 10. Phase 1: Core Voltage Transition Flow

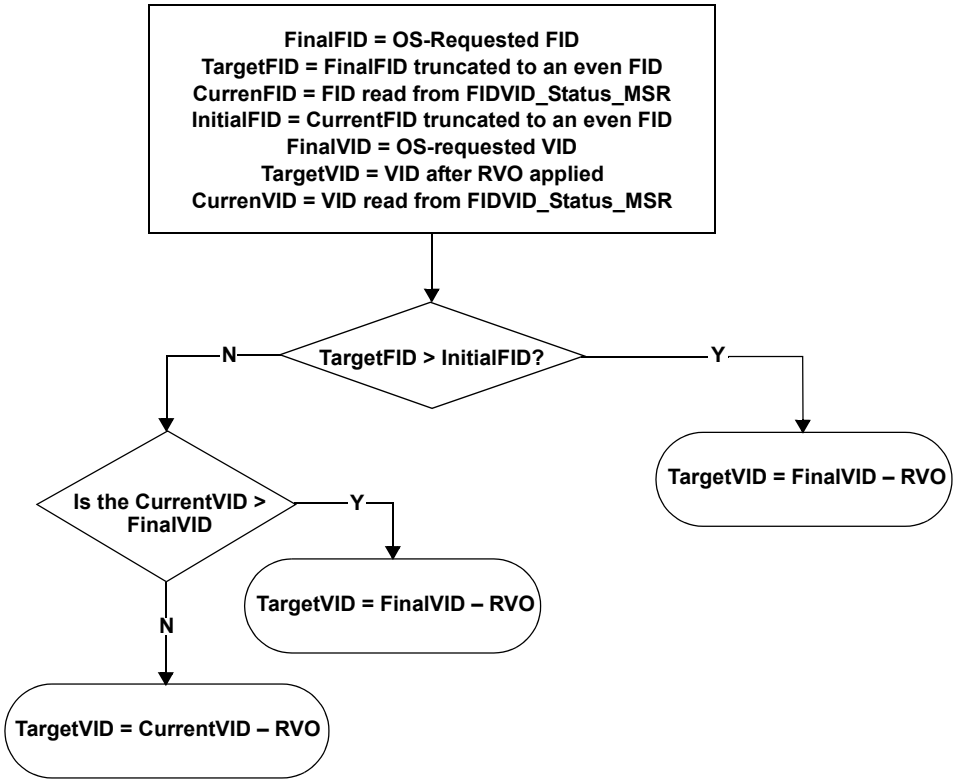


Figure 11. TargetVID Calculation

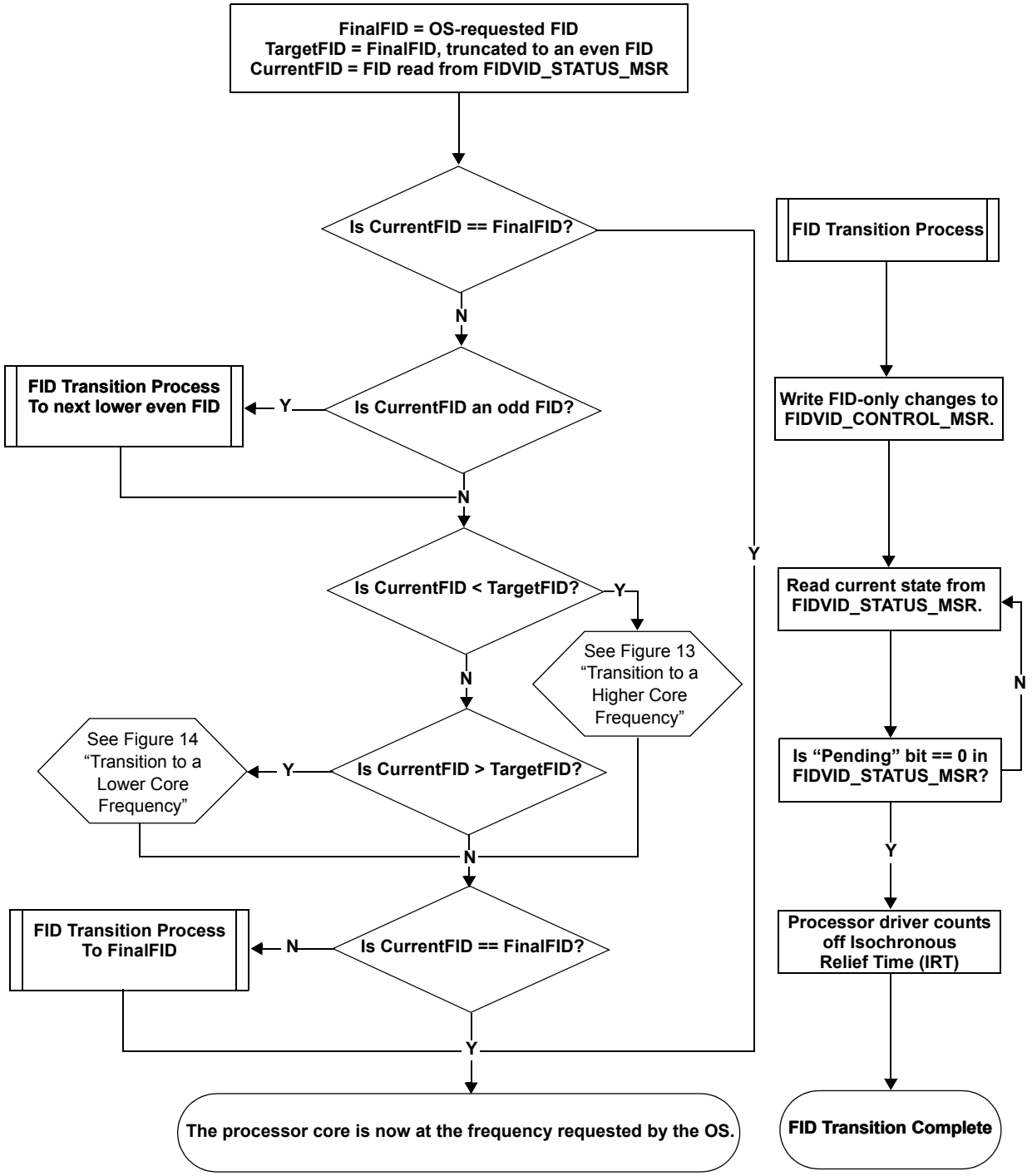


Figure 12. Phase 2: Core Frequency Transition Flow

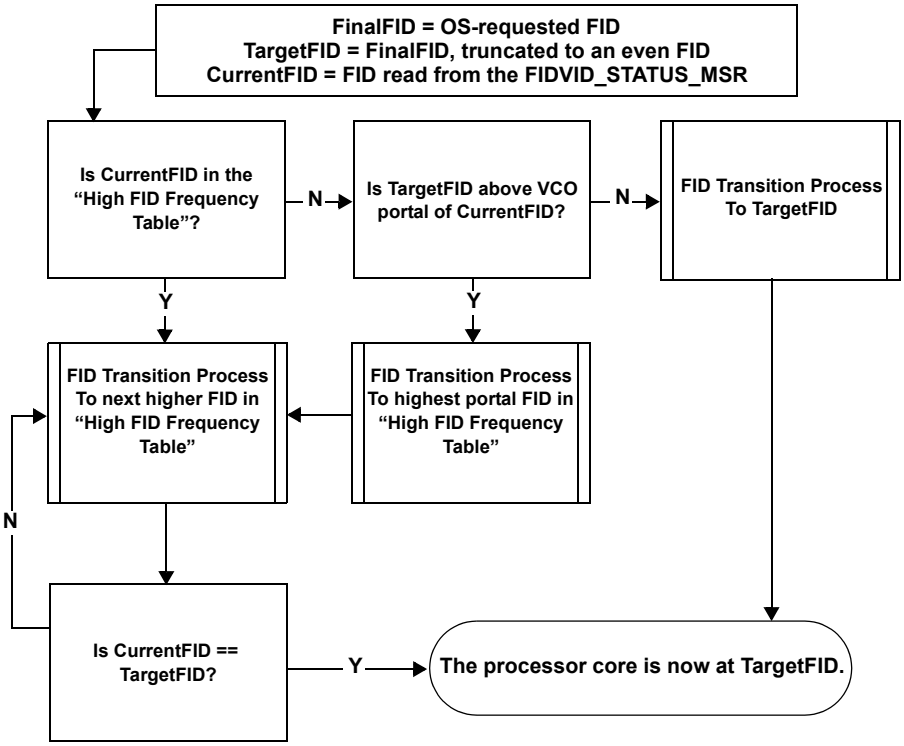


Figure 13. Transition to a Higher Core Frequency

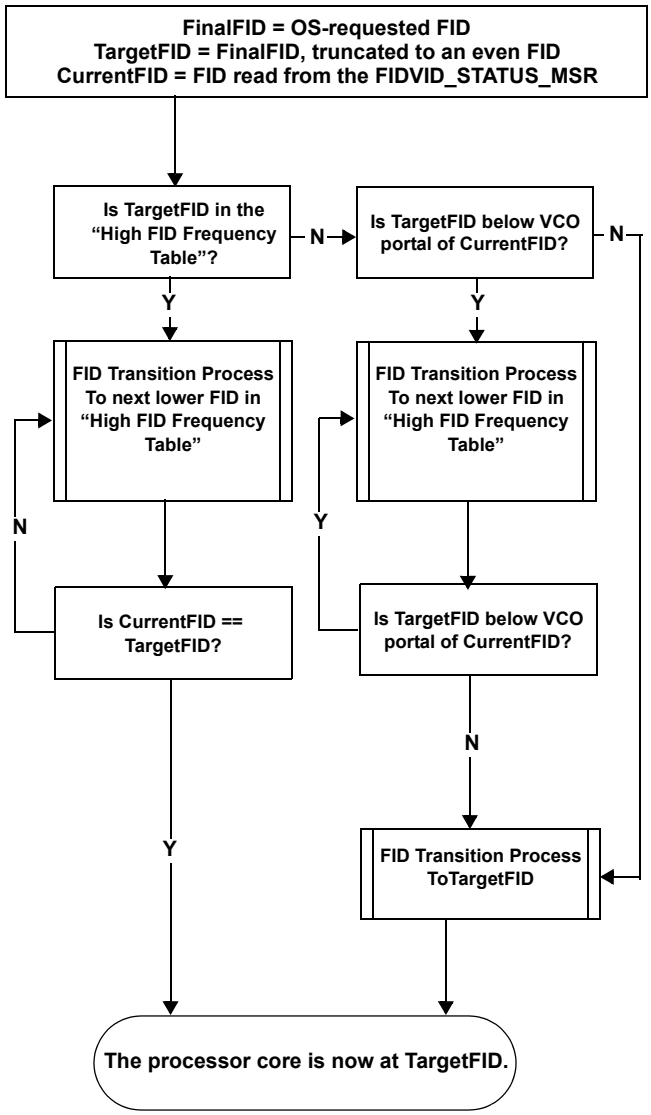


Figure 14. Transition to a Lower Core Frequency

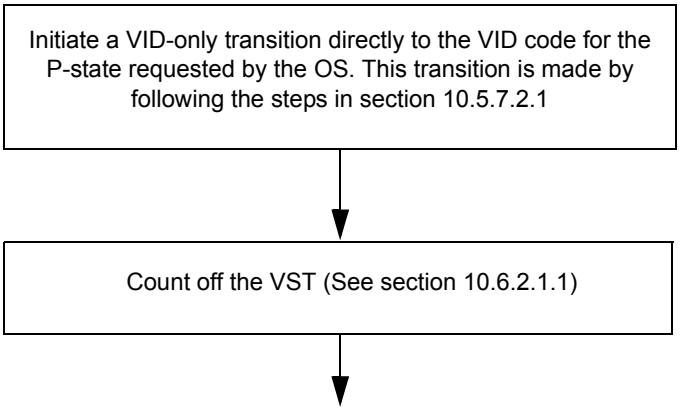


Figure 15. Phase 3: Core Voltage Transition Flow

10.5.7.2.1 Changing the VID

Note: *Software must hold the FID constant when changing the VID.*

To change the processor voltage:

1. Write the following values to FIDVID_CTL (MSR C001_0041h):
   – NewVID field (bits 13–8) with the VID code associated with the target voltage
   – NewFID field (bits 5–0) with the CurrFID value indicated in the FIDVID_STATUS MSR
   – InitFidVid bit (bit 16) set to 1. Setting this bit initiates the VID change.
   – Clear all other bits to 0.
2. Loop on reading the FidVidPending bit (bit 31) of FIDVID_STATUS (MSR C001_0042h) until the bit returns 0. The FidVidPending bit stays set to 1 until the new VID code has been driven to the voltage regulator.

10.5.7.2.2 Changing the FID

AMD NPT Family 0Fh Processors support direct FID transitions only between FIDs that represent a VCO frequency change less then or equal to 200 MHz. To change between FIDs that represent a VCO frequency change greater than 200 MHz, software must make multiple transitions each less than or equal to a 200 MHz change in VCO frequency. For information about allowable FID transitions, see Tables 69 and 70 on page 305.

Note: *Software must hold the VID constant when changing the FID.*

To change the core frequency:

1. Write the following values to FIDVID_CTL (MSR C001_0041h):
   – NewVID field (bits 13–8) with the CurrVID value reported in the FIDVID_STATUS MSR.
   – NewFID field (bits 5–0) with the FID code associated with the target frequency.
   – StpGntTOCnt field (bits 51–32) with a value corresponding to the processor PLL lock time. The PLL lock time is specified in section 14.2.9.1 on page 423. PLL lock time is communicated to the processor driver through the PLL_LOCK_TIME field of the _PSS object. PLL lock time is in microseconds. To translate the PLL lock time to an StpGntTOCnt value, multiply PLL_LOCK_TIME by 1000 to get nanoseconds, then divide by 5 which is the clock period of the counter in ns. Therefore, StpGntTOCnt value = PLL_LOCK_TIME ∗1000/5. For example, a PLL lock time of 2 µs results in an StpGntTOCnt value of 400 decimal and 190h.
   – InitFidVid bit (bit 16) set to 1. Setting this bit initiates the P-state transition.
   – Clear all other bits to 0.
2. Loop on reading the FidVidPending bit (bit 31) of FIDVID_STATUS (MSR C001_0042h) until the bit returns 0. The FidVidPending bit stays set to 1 until the new FID code is in effect.

Figure 12 on page 311 illustrates the transition flow for core frequencies, including the requirement to wait the isochronous relief time between FID steps.

10.5.7.3 Dual Core P-state Transition Requirements

For dual core processors, it is the processor driver's responsibility to coordinate operating systems P-state transition requests for the individual cores. The cores in a dual core processor share a common clock grid and voltage plane, and therefore do not support independent P-states even though the operating system views the cores as independent processors, and a set of ACPI P-state objects is declared to the operating system for each core. The processor driver enforces a "highest requested P-state" policy. Both cores will be in the highest P-state requested for either core. The processor driver keeps track of P-state transition requests and only reduces the P-state of either core after the operating system has requested that the P-state of both cores be reduced. When the operating system requests that the P-state of either core be increased, the processor driver increases the P-state of both cores.

10.6 ACPI 2.0 Processor P-State Objects

For systems that support P-state transitions, the BIOS must provide the following ACPI 2.0 P-state objects to support operating systems that provide native support for processor P-state transitions. These subsections discuss the specific implementation of these objects. In multiprocessor systems that support P-state transitions, the BIOS must provide the ACPI 2.0 P-state objects defined in this section for each processor. For dual core processors, one set of ACPI P-state objects is declared for each core.

10.6.1 _PCT (Performance Control)

The _PCT object is generically described in section 8.3.3.1 of the ACPI 2.0 specification. The _PCT, _PSS, and _PPC objects (defined in the next sections) are all placed after the Processor object in the _PR name space for operating systems that have native support for processor P-states. This section provides BIOS specifics regarding the use of _PCT.

The _PCT object specifies the control register and status register used to initiate and verify P-state transitions. The processor's P-state transition protocol is abstracted from the operating system by the processor driver, so this object does not declare the MSRs used for FID_Change protocol, but rather declares these two registers to be *functional fixed hardware*. Functional fixed hardware means that the mechanism is specific to the processor, and the processor driver knows how to initiate and verify P-state transitions because the processor driver is written by the processor vendor or written based upon documentation supplied to the operating system vendor by the processor vendor.

The following is a sample _PCT object. Sections referenced are in the ACPI 2.0 specification and ACPI 2.0 Errata Document, revision 1.4.

```
Name (_PCT, Package (2)         // Performance Control Object
        {
//            ResourceTemplate () {Register(FFixedHW, 0, 0, 0)}   // Control
         Buffer () {
                0x82,            // B0-Generic Register Descriptor (Sec. 6.4.3.7)
                 0xC,0,           // B1:2-length (from _ASI through _ADR fields)
                 0x7F,            // B3-Address space ID, _ASI, SystemIO
                 0,               // B4-Register Bit Width, _RBW
                 0,               // B5-Register Bit Offset, _RBO
                 0,               // B6-Reserved. Must be 0.
                 0,0,0,0,0,0,0,0, // B7:14-register address, _ADR (64bits)
                 0x79,0},         // B15:16-End Tag (Section 6.4.2.8)

//            ResourceTemplate () {Register(FFixedHW, 0, 0, 0)}      // Status
         Buffer () {
                 0x82,            // B0-Generic Register Descriptor (Section
6.4.3.7)
                 0xC,0,           // B1:2-length (2 bytes)
                 0x7F,            // B3-Address space ID, _ASI, SystemIO
                 0,               // B4-Register Bit Width, _RBW
                 0,               // B5-Register Bit Offset, _RBO
                 0,               // B6-Reserved. Must be 0.
                 0,0,0,0,0,0,0,0, // B7:14-register address, _ADR (64bits)
                 0x79,0},         // B15:16-End Tag (Section 6.4.2.8)
     }) // End of _PCT object
```

10.6.2 _PSS (Performance-Supported States)

The _PSS object is generically described in Section 8.3.3.2 of the ACPI 2.0 specification. This object follows the _PCT object in the _PR name space in support of operating systems with native support for processor P-states. This section provides the BIOS specifics regarding use of _PSS.

The _PSS object specifies the performance states that are supported by the system.

Based upon the maximum voltage and frequency supported by the processor and system requirements, the BIOS builds the _PSS object to specify the performance states supported by the system.

The BIOS creates the _PSS object based on the valid P-states determined by the BIOS. The BIOS is required to maintain a set of Performance State Tables (PSTs) sheet for all processors capable of P-state transitions. Only the PST entries that correspond to the processor found in the specific system by the BIOS during the POST routine are used by the BIOS to create the _PSS object for that system. The _PSS object can have as many performance states as the BIOS determines are supported by the processor. For example, the _PSS object could provide performance state definitions P0–P2:

```
P0: 2400 MHz at 1.20 V
P1: 1600 MHz at 1.10 V
P2:  800 MHz at 1.00 V
```

The generic _PSS object description has the following format:

```
Name (_PSS, Package()
{          // Field Name Field Type
Package () // Performance State 0 Definition – P0
{
CoreFreq,             // DWordConst
Power,                // DWordConst
TransitionLatency,    // DWordConst
BusMasterLatency,     // DWordConst
Control,              // DWordConst
Status                // DWordConst
},
Package ()            // Performance State n Definition – Pn
{
CoreFreq,             // DWordConst
Power,                // DWordConst
TransitionLatency,    // DWordConst
BusMasterLatency,     // DWordConst
Control,              // DWordConst
Status                // DWordConst
}
} )                   // End of _PSS object
```

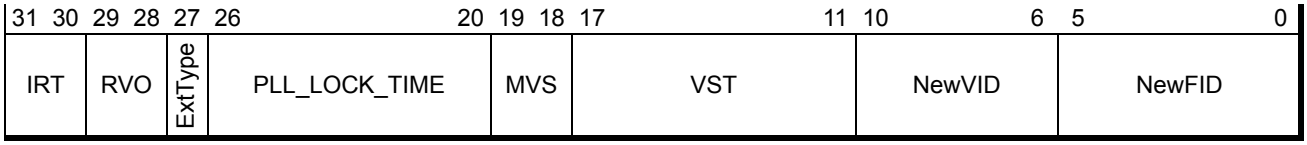
Each performance state entry contains six data fields as follows:

- ***CoreFreq*** is the core CPU operating frequency (in MHz).

- ***Power*** is the typical power dissipation (in milliWatts). The BIOS must fill out this field for the each performance state using the power calculated when determining which P-states are supported.
- ***TransitionLatency*** is the worst-case latency, in microseconds, that the CPU is unavailable during a transition from any performance state to this performance state. During a P-state transition, the CPU is unavailable for no more than 6 µs for each frequency step. While the total P-state transition could take up to 2 ms, the operating system is not blocked during the transition. Therefore the recommended value for this field is 100 µs.
- ***BusMasterLatency*** is the worst-case latency, in microseconds, that Bus Masters are prevented from accessing memory during a transition from any performance state to this performance state. This value is estimated at 7 µs.
- ***Control*** is the value to be written to the _PSS Control Field in order to initiate a transition to the performance state. The BIOS must fill this out as described on page 318.
- **Status** is the value that the processor driver can compare to a value read from the _PSS Status Field to ensure that the transition to the performance state was successful. The BIOS must fill this out as described on page 322.

10.6.2.1 _PSS Control Field

The control field of each performance state definition within the _PSS object contains the information that the processor driver uses. This includes NewVID and NewFID values that must be written into the FIDVID_CTL register (MSR C001_0041) to initiate transitions between P-states. _PSS Control field has the following format:



Bits	Field	Description
31–30	IRT	Isochronous Relief Time
29–28	RVO	Ramp Voltage Offset
27	ExtType	Extended Interface = 1.
26–20	PLL_LOCK_TIME	Phase-Locked Loop Time
19–18	MVS	Maximum Voltage Step
17–11	VST	Voltage Stabilization Time
10–6	NewVID	The NewVID[4:0] value associated with the OS-requested P-state. CurrVID[5] of the _PSS Status field specifies the OS-requested NewVID[5] value.
5–0	NewFID	The FID value associated with the OS-requested P-state

The following sections describe the sub-fields of the _PSS control field.

10.6.2.1.1 Voltage Stabilization Time

The Voltage Stabilization Time (VST) defines the number of microseconds (in 20 µs increments) that are required for the voltage to increase by the amount specified by the MVS field when the VID outputs of the processor transition. The processor driver counts VST between VID steps that increase the processor's core voltage, not between steps that decrease it. Table 71 shows the actual stabilization time for several VST values.

Table 71. Sample VST Values

VST (Bits 17–11)	Stabilization Time
7'h00	0 µs
7'h01	20 µs
7'h02	40 µs (BIOS default)
...	
7'h64	2000 µs
…	
7'h7e	2520 µs
7'h7f	2540 µs

10.6.2.1.2 Maximum Voltage Step

The Maximum Voltage Step (MVS) defines the maximum voltage increment the processor driver can use when changing from a lower voltage to a higher voltage. For the processor driver, when increasing core voltage, the next VID = CurrVID – 2^{MVS}. Table 72 on page 320 shows the values and increments for this field.

For example, if the voltage is being increased, and

- the current VID is 00100 (1.45 V),
- the OS-requested target VID is 00000 (1.55 V),
- and MVS is 0 (which selects 25mV voltage steps).

Then, the processor driver calculates the first VID to which to transition as follows:

Next_VID = CurrVID - 2^{MVS} = 00100 - 2^{0} = 00100 - 1 = 00011 = 1.475 V

Next the processor driver transitions the VID to select 1.475 V.

After the VST is counted off, the processor driver calculates the next VID to which to transition as follows:

Next_VID = CurrVID - 2^{MVS} = 00011 - 2^{0} = 00010 = 1.500 V

This process iterates until the CurrVID is equal to the OS-requested target VID minus the VID associated with the ramp voltage offset. After each VID transition the processor driver counts off the voltage stabilization time before calculating the Next_VID.

Table 72. MVS Values

MVS (Bits 19–18)	Increment
00	25 mV (BIOS default. This is the required value that must be used.)
01	50 mV (Reserved for future use)
10	100 mV (Reserved for future use)
11	200 mV (Reserved for future use)

10.6.2.1.3 PLL Lock Time

The 7-bit binary PLL_LOCK_TIME value defines the time required for the processor PLLs to lock in microseconds. The PLL lock time is specified in section 14.2.9.1 on page 423.

10.6.2.1.4 Ramp Voltage Offset

The Ramp Voltage Offset (RVO) is the offset voltage applied to the VID requested VDD supply level (VID_VDD) voltage when performing P-state transitions. The RVO is specified in the processor data sheet. Table 73 lists the RVO values and their corresponding offset voltages.

Table 73. RVO Values

RVO (Bits 29–28)	Offset Voltage	Description
00	0 mV	Target P-state VID needs no adjustment for RVO. (BIOS default for Multiprocessor Systems only. This is the required value.)
01	25 mV	Decrement the target P-state VID by 1 to add the RVO.
10	50 mV	Decrement the target P-state VID by 2 to add the RVO. (BIOS default. This is the required value.)
11	75 mV	Decrement the target P-state VID by 3 to add the RVO.

For the processor driver, RVO is in VID increments. To increase the processor voltage the processor driver must subtract the RVO field from the VID. The MVS field dictates the increments in which RVO can be subtracted from VID.

For example, consider the case when the VID is 00100b (1.45 V), RVO is 10b (50 mV), MVS is 00b (25 mV), and VST is 7'h05 (100 μs). Based upon these parameters, the voltage must be increased by 50 mV in two steps of 25 mV with a 100 μs delay after each VID change. The steps to apply the RVO are:

1. The VID code is reduced to 00011b. This transitions the voltage from 1.45 V to 1.475 V.
2. The processor driver waits 100 μs as specified by VST.
3. The VID code is reduced to 00010b. This transitions the voltage from 1.475 V to 1.5 V.

4. The processor driver waits 100 μs as specified by VST.

Table 74 lists the VID codes and their corresponding voltages for AMD NPT Family 0Fh Processors.

Table 74. VID Code Voltages

VID[5:0]	Voltage	VID[5:0]	Voltage	VID[5:0]	Voltage	VID[5:0]	Voltage
00_0000b	1.550 V	01_0000b	1.150 V	10_0000b	0.7625 V	11_0000b	0.5625 V
00_0001b	1.525 V	01_0001b	1.125 V	10_0001b	0.7500 V	11_0001b	0.5500 V
00_0010b	1.500 V	01_0010b	1.100 V	10_0010b	0.7375 V	11_0010b	0.5375 V
00_0011b	1.475 V	01_0011b	1.075 V	10_0011b	0.7250 V	11_0011b	0.5250 V
00_0100b	1.450 V	01_0100b	1.050 V	10_0100b	0.7125 V	11_0100b	0.5125 V
00_0101b	1.425 V	01_0101b	1.025 V	10_0101b	0.7000 V	11_0101b	0.5000 V
00_0110b	1.400 V	01_0110b	1.000 V	10_0110b	0.6875 V	11_0110b	0.4875 V
00_0111b	1.375 V	01_0111b	0.975 V	10_0111b	0.6750 V	11_0111b	0.4750 V
00_1000b	1.350 V	01_1000b	0.950 V	10_1000b	0.6625 V	11_1000b	0.4625 V
00_1001b	1.325 V	01_1001b	0.925 V	10_1001b	0.6500 V	11_1001b	0.4500 V
00_1010b	1.300 V	01_1010b	0.900 V	10_1010b	0.6375 V	11_1010b	0.4375 V
00_1011b	1.275 V	01_1011b	0.875 V	10_1011b	0.6250 V	11_1011b	0.4250 V
00_1100b	1.250 V	01_1100b	0.850 V	10_1100b	0.6125 V	11_1100b	0.4125 V
00_1101b	1.225 V	01_1101b	0.825 V	10_1101b	0.6000 V	11_1101b	0.4000 V
00_1110b	1.200 V	01_1110b	0.800 V	10_1110b	0.5875 V	11_1110b	0.3875 V
00_1111b	1.175 V	01_1111b	0.775 V	10_1111b	0.5750 V	11_1111b	0.3750 V

10.6.2.1.5 Isochronous Relief Time

The Isochronous Relief Time (IRT) is the amount of time the processor driver must count after each FID change step in a given P-state transition. During each FID change step, system memory is inaccessible to bus masters. While the processor driver is counting IRT between FID change steps, bus masters have access to system memory. Table 75 lists the IRT values and their corresponding times.

Table 75. IRT Values

IRT (Bits 31–30)	Time
00	10 μs
01	20 μs
10	40 μs
11	80 μs (BIOS default)

10.6.2.2 _PSS Status Field

The status field of each performance state definition within the _PSS object contains the information required by the processor driver to verify that the transition has completed based upon a read of the FIDVID_STATUS MSR. The _PSS Status field is 32 bits, and the BIOS must map the FIDVID_STATUS MSR CurrVID and CurrFID information to the _PSS status field as shown in Table 76 on page 322.

Table 76. _PSS Status Field

Bit	Name	Description
5–0	CurrFID	The FID value associated with the OS-requested P-state
11–6	CurrVID	The VID value associated with the OS-requested P-state
28–12	Reserved	These bits must be 0 and are ignored by the processor driver.
31–29	Version	Version = 0

The CurrFID and CurrVID fields of the FIDVID_STATUS MSR are valid only when the FidVidPending bit is 0.

10.6.3 _PPC (Performance Present Capabilities)

The _PPC object is used to limit the maximum P-state that the operating system can use at any given time. This object follows the _PSS object in the _PR name space for the operating systems that support the ACPI 2.0 processor P-state objects.

_PPC is generically described in Section 8.3.3.3 of the ACPI 2.0 specification.

This section provides BIOS specifics regarding use of _PPC.

If the OEM and BIOS vendor do not implement a _PPC that limits the maximum P-state under certain conditions, then the BIOS must always return a value of 0 for the _PPC object. When a value of 0 is returned for the _PPC object, all P-states specified by the _PSS object are available.

Operating systems with native support for processor P-state transitions and ACPI Thermal zones make use of reduced P-states for passive cooling before making use of "throttling" with the stop grant state. This is the case even when the _PPC object returns a value of 0.

10.6.3.1 Using _PPC to Limit the Maximum Processor P-State When battery Powered

Processor performance exceeds the demands of most typical notebook PC applications. However, there are basically idle scenarios where applications run instructions that incorrectly cause the operating system to conclude that the processor is 100% utilized. This causes the operating system to force the processor to its maximum performance state, wasting power with no benefit to the user. Also, static power consumption of the processor increases when the processor is operating in a P-state above the minimum voltage level supported by the processor.

Therefore, it increases the system's battery-powered runtime to limit the maximum performance state available when battery-powered.

10.6.3.2 Implementation Overview

An ACPI-compliant General-Purpose Event (GPE) input of the chipset (typically the I/O Hub) is dedicated to causing an SCI whenever the notebook power source changes. This input is referred to as AC available (ACAV). The system can control ACAV directly based upon the presence of an AC adapter, or the embedded controller (EC) in the system can control ACAV.

Whenever the AC adapter is inserted or removed from the system, the chipset logic associated with ACAV sets the status bit associated with ACAV, and if enabled causes an SCI to be asserted. Note: the control method associated with ACAV status could be implemented in the EC. A chipset GPE dedicated for AC adapter status is used for this description.

The BIOS provides the following ACPI ASL control methods and objects:

- _PPC (Performance Present Capabilities)
- _PSR (Power Source) is described in Section 11.3.1 of the ACPI 2.0 specification. _PSR returns:
  - 0x00000000 if the system is running on battery power.
  - 0x00000001 if the system is running from an AC adapter.
- _SB.AC is a BIOS-defined device object for the AC adapter.
- _L12 is the control method associated with GPE 12.

The flow associated with AC adapter insertion or removal is:

- AC adapter is inserted or removed
- The GPE Status bit associated with ACAV is asserted (for this example, GPE 12)
- An SCI (system control interrupt is issued)
- The OS/ACPI driver determines that GPE 12 was asserted and runs the associated control method (_L12)
- _L12
  - Issues a Notify(_PR.CPU0, 0x80) which forces the OS to re-evaluate the _PPC object.
    - _PPC reads the state of the ACAV pin and returns:

      0 if the system is AC-powered. All P-states are available.

      n if the system is battery-powered, where n is the highest P-state is supported when battery-powered. Only P-states n and lower are available. For example, if a processor supports 3 P-states:
      P0 = 2400 MHz at 1.2 V
      P1 = 1600 MHz at 1.1 V
      P2 = 800 MHz at 1.0 V

Then n = 2 limits the maximum P-state to 800 MHz at 1.0 V when the system is battery-powered.

  - OS takes into account the available P-states and if necessary performs a P-state transition.
- Issues a Notify(_SB.AC, 0x80) where _SB.AC is the AC adapter device object.
  - OS runs _PSR to determine if the current power source is the AC adapter. PSR reads the state of the ACAV pin and returns:

    1 if the system is AC-powered.

    0 if the system is battery-powered.

- Normal operation continues.

10.6.4 PSTATE_CNT

PSTATE_CNT is an entry in the Fixed ACPI Description Table (FADT). A PSTATE_CNT entry of 0 informs the operating system that the BIOS in System Management Mode (SMM) does not perform P-state transitions.

The BIOS must report a value of 0 in the PSTATE_CNT field of the FADT. The only BIOS-controlled P-state transition, if any, must be performed near the beginning of the POST routine before control is passed to the operating system. All subsequent transitions are made by system software not the BIOS. System software is either the AMD PowerNow! technology software (for Microsoft® operating systems prior to the Windows® XP operating system) or the operating system and associated processor driver (for the Windows XP operating system and subsequent Microsoft operating systems).

Note: *SMM is not used to perform P-state transitions.*

10.6.5 CST_CNT

CST_CNT is an entry in the FADT. If non-zero, this field contains the value the operating system writes to the SMI_CMD register to indicate operating system support for the _CST object and C States Changed notification.

The BIOS must report a value of 0 in the CST_CNT field of the FADT. AMD platforms only support ACPI 1.0b processor power state functionality. Processor power states are referred to as "C" states by the ACPI specifications.

10.7 ACPI 3.0 Processor P-State Objects

For systems that support P-state transitions, the BIOS must provide the following ACPI 3.0 P-state objects to support operating systems that provide native support for processor P-state transitions.

The _PSD (P-State Dependency) object is generically described in section 8.4.4.5 of the ACPI 3.0b specification. The _PSD object is placed after the Processor object in the _PR name space for

operating systems that have native support for processor P-states. This section provides BIOS specifics regarding the use of _PSD.

The _PSD object provides P-state control information for logical processor dependencies to OSPM. For example, in a dual-core processor, each core may not operate at a unique voltage. With the _PSD object, a P-state transition on one processor core will cause OSPM to inititate a P-state transition to the same performance state on the remaining dependent core.

The _PSD object should not be included for single-core processors. A _PSD object must be included for each dual-core processor in the system.

```
Name (_PSD, Package(0x1)
{          // Field Name Field Type
Package (0x5)
{
0x5,            // 5 entries
0x0,            // Revision 0
Domain,         // node or socket number, starting with zero
0xfd,           // OSPM Coordinate, Initiate on any Proc
0x2,            // 2 cores per node/socket
}
} )             // End of _PSD object
```

Domain is the logical processor enumeration, starting with a value of zero, and corresponding with a processor node/socket.

10.8 BIOS Support for AMD PowerNow!™ Software with Legacy Operating Systems

If BIOS determines it has a processor and system that support P-state transitions, it provides a PSB that is comprised of a header and the processor specific Performance State Table (PST) that matches the processor in the system. The PSB must be 16-byte aligned in memory.

If the BIOS cannot determine that the system and processor support P-state transitions as defined in 10.5.1 on page 299, then the BIOS must not provide a PSB. Changing the PSB Signature field to all 0s is an effective way of removing the PSB.

Table 77 defines the PSB and PST structures.

Table 77. Performance State Block Structure

Name	Length (Bytes)	Field Purpose
PSB Header		
Signature	10	Start of the PSB - Always set this field to "AMDK7PNOW!"

Table 77. Performance State Block Structure (Continued)

Name	Length (Bytes)	Field Purpose
TableVersion	1	The version of the table as defined by AMD. Set to 14h for version 1.4. If AMD PowerNow!™ software does not recognize the TableVersion, it does not attempt to make P-state transitions.
Flags	1	All bits are reserved and written to zero.
VST	2	Voltage Stabilization Time. The amount of time required for the processor's core voltage regulator to increase the processor's core voltage the increment specified by MVS. VST is specified in 20 μs increments. The default is 05h (100 μs). All other values are reserved.
Reserved1	1	Bits 1–0: Ramp Voltage Offset (RVO) has the same definition as for the _PSS ACPI object. Bits 3–2: Isochronous Relief Time (IRT) has the same definition as for the _PSS ACPI object. Bits 5–4: Maximum Voltage Step (MVS) has the same definition as for the _PSS ACPI object. Bits 7–6: Number of available P-states when the system is battery powered has the following definition: 00b = all P-states are available 01b = only the minimum P-state is available 10b = 2 lowest P-states are available 11b = 3 lowest P-states are available.
NumPST	1	The total number of PSTs in the PSB. This value must be set to 01h.
PST Header		
CPUID	4	CPUID (EAX of CPUID extended function 8000_0001h) of the processor that this PST belongs to. This provides processor family, extended family, model, extended model, and stepping (revision) fields.
PLL Lock Time	1	Processor PLL Lock time in microseconds. For example, PLL_LOCK_TIME of 2 μs is represented as 00000010b.The PLL Lock time is specified in section 14.2.9.1 on page 423.
MaxFID	1	MaxFID[5:0] as reported by the MaxFID field of the FIDVID_STATUS MSR.
MaxVID	1	MaxVID[5:0] as reported by the MaxVID field of the FIDVID_STATUS MSR.
NumPStates	1	The number of FID, VID combinations in the PST. This field must be >= 1h.
FID	1	FID[5:0] code corresponding to the frequency of the lowest performance state that the processor supports. This is defined in the processor data sheet.
VID	1	VID[5:0] code corresponding to the voltage of the lowest performance state that the processor supports. This is defined in the processor data sheet.
----------	---------	FID, VID pairs are concatenated here so that the total number of pairs is equal to the NumPstates field. Each subsequent FID, VID pair is appended in ascending order (i.e., the last FID,VID pair corresponds to the maximum performances state supported by the processor).

10.9 System Configuration for Power Management

BIOS configuration of the system for power management is described in the sections that follow.

10.9.1 Chipset Configuration for Power Management

Chipset configuration is covered in chipset related documentation. The BIOS is required to:

- Ensure that the SMAF code to STPCLK and Stop Grant mapping for all components based on HyperTransport technology is configured as described in Table 68 on page 295.
- For systems using AM2, ASB1, or S1g1 package processors, the HyperTransport links are tristated in response to LDTSTOP_L assertion.
- Chipset power management features are appropriately enabled based upon system class.
- LDTSTOP_L assertion for FID changes during P-state transitions:
  - Must be at least 2µs. Note: for UMA chipsets 2µs is ideal and may be required in some cases. Note: Links need to be 8-bits wide or wider and 400 MHz or greater.
  - Must be less than 6µs for chipsets that support isochronous streams (e.g. UMA graphics).
- LDTSTOP_L assertion for link width/frequency changes during initialization is a minimum of 2 µs for all processors. Note: For Links operating at 200 MHz no DMA traffic can be in progress.
- LDTSTOP_L assertion during FID_Change is 10 µs for systems using registered DIMMs.
- LDTSTOP_L assertion during C3 is a minimum of 1 µs for all processors.

10.9.2 Processor Configuration for Power Management

BIOS configures the processor to enable power management during the POST routine. Table 78 on page 328 lists processor memory system configuration register settings that the BIOS must initialize.

Table 78. Configuration Register Settings for Power Management

Functionality	Register	Mobile Setting	UP Server[1]/ Desktop[1] Setting	MP Server Setting
Enable HyperTransport™ links to be tristated in response to LDTSTOP_L assertions.	LDT0 Link Control (function 0: offset 84h[13]) (LdtStopTriEn) LDT1 Link Control (function 0: offset A4h[13]) (LdtStopTriEn) LDT2 Link Control (function 0: offset C4h[13]) (LdtStopTriEn)	1b	0b	0b
Enable HyperTransport™ Receivers to be tristated in response to LDTSTOP_L assertion in mobile systems.	LDT0 Link Control (function 0: offset 84h[11]) (LdtStopRcvDis) LDT1 Link Control (function 0: offset A4h[11]) (LdtStopRcvDis) LDT2 Link Control (function 0: offset C4h[11]) (LdtStopRcvDis)	1b	0b	0b
Do not enable overriding tristating of HyperTransport™ clocks in response to LDTSTOP_L assertion except for mobile systems with UMA Graphics.	LDT0 Link Control (function 0: offset 84h[15]) (LdtStopTriClkOvr) LDT1 Link Control (function 0: offset A4h[15]) (LdtStopTriClkOvr) LDT2 Link Control (function 0: offset C4h[15]) (LdtStopTriClkOvr)	Discrete graphics[2] 0b UMA graphics 1b	0b	0b

Notes:

1. *Single processor branded server with unbuffered DIMMs only, or any dekstop brand (with unbuffered DIMMs).*
2. *Optionally, a value of 1b can be used with discrete graphics. This reduces processor reconnect delay and increases power consumption.*
3. *Optionally, a ClkSel value of 3h for SMAF 1 (a divider of 64), can be used with discrete graphics. This reduces processor clock ramp delay and increases power consumption.*
4. *Optionally, a value of 2113_2113h can be used. This significantly reduces processor power during halt and may reduce system performance. To ensure that system performance loss is kept to a minimum in this state, ClkRampHys (Function 3, Offset D4h) should be programmed to 2 us (0111b.)*
5. *For mobile systems that enable AltVID, VSTIME should be programmed to account for the time it takes to ramp the voltage from AltVID to the VID for the 800MHz P-State.*
   *VSTIME = ((800 MHz VID) - AltVID) * 2us/mV*
6. *Only for Dual-Core Revision G processors in AM2 or ASB1 package with unbuffered DIMMs and SMI initiated C1E, or, single-core Revision G processors in ASB1 package. All others use Revision F SMAF 0 and 1 values. Do not enable C1E with SO-DIMMs. Do not enable AltVID with these processors. If chipset does not support SMI inititated C1E then use Revision F SMAF 0 and 1 values.*

Table 78. Configuration Register Settings for Power Management (Continued)

Functionality	Register	Mobile Setting	UP Server[1]/ Desktop[1] Setting	MP Server Setting
Map STPCLK actions to SMAF codes[3]: SMAF 0 = C2 SMAF 1 = C3 SMAF 2 = VID/FID Change SMAF 3 = S1	Power Management Control Low (function 3: offset 80h)	Discrete graphics 6B07_6B61h UMA graphics 3307_3331	Revision F processors 2307_0000h Revision G processors with Discrete graphics[6] 2307_6361h Revision G processors with UMA graphics[6] 2307_3331h	2307_0000h
Map STPCLK actions to SMAF codes: SMAF 4 = S3 SMAF 5 = Throttling SMAF 6 = S4/S5 SMAF 7 is not used for STPCLK, the corresponding field is used for HALT.	Power Management Control High (function 3: offset 84h)	Discrete graphics 6113_3113h UMA graphics 3113_3113h	2113_2113h	0013_2113h[4]
BCLK and LCLK PLL frequency lock	Clock Power/Timing Low (function 3: offset D4h)	000D_A701h		04E2_A706h

Notes:

1. *Single processor branded server with unbuffered DIMMs only, or any dekstop brand (with unbuffered DIMMs).*
2. *Optionally, a value of 1b can be used with discrete graphics. This reduces processor reconnect delay and increases power consumption.*
3. *Optionally, a ClkSel value of 3h for SMAF 1 (a divider of 64), can be used with discrete graphics. This reduces processor clock ramp delay and increases power consumption.*
4. *Optionally, a value of 2113_2113h can be used. This significantly reduces processor power during halt and may reduce system performance. To ensure that system performance loss is kept to a minimum in this state, ClkRampHys (Function 3, Offset D4h) should be programmed to 2 us (0111b.)*
5. *For mobile systems that enable AltVID, VSTIME should be programmed to account for the time it takes to ramp the voltage from AltVID to the VID for the 800MHz P-State.*
   *VSTIME = ((800 MHz VID) - AltVID) * 2us/mV*
6. *Only for Dual-Core Revision G processors in AM2 or ASB1 package with unbuffered DIMMs and SMI initiated C1E, or, single-core Revision G processors in ASB1 package. All others use Revision F SMAF 0 and 1 values. Do not enable C1E with SO-DIMMs. Do not enable AltVID with these processors. If chipset does not support SMI inititated C1E then use Revision F SMAF 0 and 1 values.*

Table 78. Configuration Register Settings for Power Management (Continued)

Functionality	Register	Mobile Setting	UP Server[1]/ Desktop[1] Setting	MP Server Setting
Sets "Ramp VID" (0 mV) and voltage regulator stabilization time (0 µs). The processor driver performs this function, not hardware.[5]	Clock Power/Timing High (function 3:offset D8h [30:28], [19:0]) (RampVIDOf) (VSTime)	000b[30:28] 0_0000h[19:0]	000b[30:28] 0_0000h[19:0]	010b[30:28] 0_2710h[19:0]
Set the AltVID value for mobile systems with discrete graphics.	Clock Power/Timing High (function 3:offset D8h [24:20]) (AltVid)	Discrete graphics P[Min] Voltage - AltVidOffset UMA graphics N/A	N/A	N/A
Enable PSIVID for mobile systems.	Power Control Miscellaneous (function 3: offset A0h [7]) (PsiVidEn)	1b	0b	0b
Set the PSIVID value for mobile systems.	Power Control Miscellaneous (function 3: offset A0h [5:0]) (PsiVid)	P[Min] VID	N/A	N/A
Do not enable tristating of memory clock except systems with discrete graphics.	DRAM Control (function 2: offset 78h[16])	UMA graphics 0b discrete graphics 1b	0b	0b
Enable DRAM power down mode.	DRAM Configuration Low (function 2: offset 94h[15])	1b	1b	1b

Notes:

1. *Single processor branded server with unbuffered DIMMs only, or any dekstop brand (with unbuffered DIMMs).*
2. *Optionally, a value of 1b can be used with discrete graphics. This reduces processor reconnect delay and increases power consumption.*
3. *Optionally, a ClkSel value of 3h for SMAF 1 (a divider of 64), can be used with discrete graphics. This reduces processor clock ramp delay and increases power consumption.*
4. *Optionally, a value of 2113_2113h can be used. This significantly reduces processor power during halt and may reduce system performance. To ensure that system performance loss is kept to a minimum in this state, ClkRampHys (Function 3, Offset D4h) should be programmed to 2 us (0111b.)*
5. *For mobile systems that enable AltVID, VSTIME should be programmed to account for the time it takes to ramp the voltage from AltVID to the VID for the 800MHz P-State.*
   *VSTIME = ((800 MHz VID) - AltVID) * 2us/mV*
6. *Only for Dual-Core Revision G processors in AM2 or ASB1 package with unbuffered DIMMs and SMI initiated C1E, or, single-core Revision G processors in ASB1 package. All others use Revision F SMAF 0 and 1 values. Do not enable C1E with SO-DIMMs. Do not enable AltVID with these processors. If chipset does not support SMI inititated C1E then use Revision F SMAF 0 and 1 values.*

Table 78. Configuration Register Settings for Power Management (Continued)

Functionality	Register	Mobile Setting	UP Server[1]/ Desktop[1] Setting	MP Server Setting
Enable DRAM controller power savings in C3.	DRAM Controller Miscellaneous (function 2: offset A0h [10]) (PwrSavingsEn)	1b	0b	0b

Notes:

1. *Single processor branded server with unbuffered DIMMs only, or any dekstop brand (with unbuffered DIMMs).*
2. *Optionally, a value of 1b can be used with discrete graphics. This reduces processor reconnect delay and increases power consumption.*
3. *Optionally, a ClkSel value of 3h for SMAF 1 (a divider of 64), can be used with discrete graphics. This reduces processor clock ramp delay and increases power consumption.*
4. *Optionally, a value of 2113_2113h can be used. This significantly reduces processor power during halt and may reduce system performance. To ensure that system performance loss is kept to a minimum in this state, ClkRampHys (Function 3, Offset D4h) should be programmed to 2 us (0111b.)*
5. *For mobile systems that enable AltVID, VSTIME should be programmed to account for the time it takes to ramp the voltage from AltVID to the VID for the 800MHz P-State.*
   *VSTIME = ((800 MHz VID) - AltVID) * 2us/mV*
6. *Only for Dual-Core Revision G processors in AM2 or ASB1 package with unbuffered DIMMs and SMI initiated C1E, or, single-core Revision G processors in ASB1 package. All others use Revision F SMAF 0 and 1 values. Do not enable C1E with SO-DIMMs. Do not enable AltVID with these processors. If chipset does not support SMI inititated C1E then use Revision F SMAF 0 and 1 values.*

10.9.2.1 Additional BIOS Support Requirements for S3

During the S3 state, power is removed from the processor's CPU core, host bridge, and memory controller. It is required that the BIOS restore the state of the processor's memory controller upon resume from S3 prior to having the memory controller bring system memory out of self-refresh mode. The system must provide non volatile storage for the memory controller configuration registers during the S3 state. For UP and DP systems, this storage is provided in the chipset. There are two opportunities for the BIOS to store the memory controller configuration to this nonvolatile memory.

- During POST as the memory controller is being initially configured—this is preferred, since it does not require SMM support.
- In SMM just prior to entry to the S3 state—BIOS uses an SMI trap on writes to the ACPI PM1 control register to get control of the system just prior to entry to S3. After storing the memory controller configuration registers, I/O instruction restart is used to place the system into the S3 state. This is not preferred because it relies on SMM.

The following processor registers must be stored in nonvolatile RAM for support of resume from S3:

- Function 1, offsets 40h–7Ch, F0h
- Function 2, offsets 40h–6Ch, 78h–94h, A0h
- Function 2, offset 98h indexes 0h–7h, 10h, 13h, 16h, 19h, 20h–27h, 30h, 33h, 36h, 39h
- Function 3, offset 44h and offset B0h if online spare is enabled
- MSRs 1Bh, 200h–20Fh, 250h, 258h, 259h, 268h-26Fh, 2FFh, C001_0015h–C001_001Ah, C001_001Dh, C001_001Fh, C001_0030h–C001_0035h, C001_0048h, C001_0055h, C001_1004h, C001_1005h, C001_100Dh

10.9.2.2 ACPI Tables/Objects

FADT entries for C-state and throttling support and throttling is listed in Table 79 on page 332. Some legacy operating systems do not use C3 if support for C2 is not declared.

Table 79. FADT Table Entries

Field	Mobile	UP Desktop	Multiprocessor
PSTATE_CNT	0	0	0
CST_CNT	0	0	0
P_LVL2_LAT	> 100 if C2 is not supported 1 if C2 is supported	> 100	> 100
P_LVL3_LAT	> 1000 if C3 is not supported 10 if C3 is supported	> 1000	> 1000
DUTY_WIDTH	0 if _PSV is not used 3 if _PSV is used	0 if _PSV is not used 3 if _PSV is used	0

11 Performance Monitoring

AMD NPT Family 0Fh Processors support the performance-monitoring features introduced in earlier processor implementations. These features allow the selection of events to be monitored, and include a set of corresponding counter registers that track the frequency of monitored events. Software tools can use these features to identify performance bottlenecks, such as sections of code that have high cache-miss rates or frequently mis-predicted branches. This information can then be used as a guide for improving or eliminating performance problems through software optimizations or hardware-design improvements.

For a general description of how to use these performance monitoring features, refer to the "Debug and Performance Resources" section in Volume 2: System Programming of the *AMD64 Architecture Programmers Manual*, order# 24593.

11.1 Performance Counters

The processor provides four 48-bit performance counters. Each counter can monitor a different event. The accuracy of the counters is not ensured. Some events are reserved. When a reserved event is selected, the results are undefined.

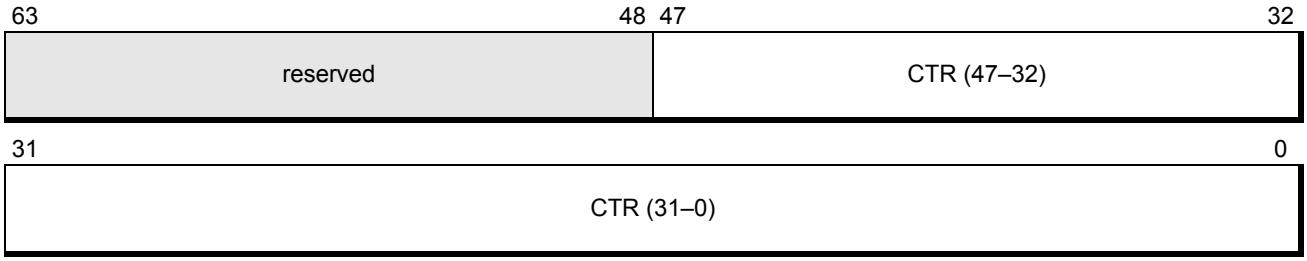
Performance counters are used to count specific processor events, such as data-cache misses, or the duration of events, such as the number of clocks it takes to return data from memory after a cache miss. During event counting, the processor increments the counter when it detects an occurrence of the event. During duration measurement, the processor counts the number of processor clocks it takes to complete an event. Each performance counter can be used to count one event, or measure the duration of one event at a time.

In addition to RDMSR instruction, the PerfCtr*n* registers can be read using a special *read performance-monitoring counter* instruction, RDPMC. The RDPMC instruction loads the contents of the PerfCtr*n* register specified by the ECX register, into the EDX register and the EAX register.

Writing the performance counters can be useful if software wants to count a specific number of events, and then trigger an interrupt when that count is reached. An interrupt can be triggered when a performance counter overflows. Software should use the WRMSR instruction to load the count as a two's-complement negative number into the performance counter. This causes the counter to overflow after counting the appropriate number of times.

The performance counters are not assured of producing identical measurements each time they are used to measure a particular instruction sequence, and they should not be used to take measurements of very small instruction sequences. The RDPMC instruction is not serializing, and it can be executed out-of-order with respect to other instructions around it. Even when bound by serializing instructions, the system environment at the time the instruction is executed can cause events to be counted before the counter value is loaded into EDX:EAX.

PerfCtr0–PerfCtr3 Registers C001_0004h, C001_0005h, C001_0006h, C001_0007h



Bit	Name	Function	R/W
63–48	Reserved	RAZ	R
47–0	CTR	Counter value	R

11.2 Performance Event-Select Registers

Performance Event-Select registers (PerfEvtSel*i*) are used to specify the events counted by the performance counters, and to control other aspects of their operation. Each performance counter supported by the implementation has a corresponding event-select register that controls its operation. This section shows the format of the PerfEvtSel*n* registers.

To accurately start counting with the write that enables the counter, disable the counter when changing the event and then enable the counter with a second MSR write.

The edge count mode increments the counter when a transition happens on the monitored event. If the event selected is changed without disabling the counter, an extra edge is falsely detected when the first event is a static 0 and the second event is a static one. To avoid this false edge detection, disable the counter when changing the event and then enable the counter with a second MSR write.

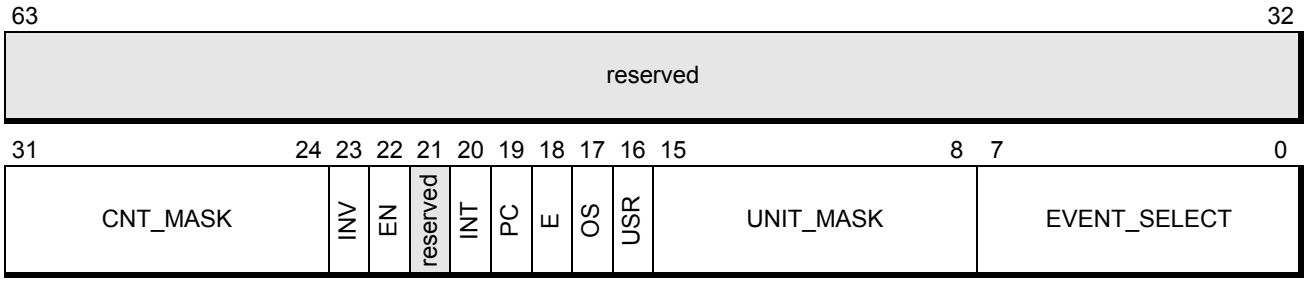
The EVENT_SELECT field specifies the event or event duration in a processor unit to be counted by the corresponding PerfCtr*i* register. The UNIT_MASK field is used to further specify or qualify a monitored event. Section 11.2.1 "Performance Monitor Events" shows the events and unit masks supported by the processor.

The performance counter registers can be used to track events in the Northbridge. Northbridge events include all memory controller events, crossbar events, and HyperTransport™ interface events as documented in 11.2.1.7, 11.2.1.8, and 11.2.1.9. When a Northbridge event is selected using one of the Performance Event-Select registers in either core of a dual core processor, a Northbridge event cannot be selected in the same Performance Event-Select register of the other core. Monitoring of Northbridge events should only be performed by one core in a dual core processor.

Care must be taken when measuring Northbridge or other non-processor-specific events under conditions where the processor may go into halt mode during the measurement period. For instance, one may wish to monitor DRAM traffic due to DMA activity from a disk or graphics adaptor. This entails running some event counter monitoring code on the processor, where such code accesses the counters at the beginning and end of the measurement period, or may even sample them periodically

throughout the measurement period. Such code typically gives up the processor during each measurement interval. If there is nothing else for the OS to run on that particular processor at that time, it may halt the processor until it is needed. Under these circumstances, the clock for the counter logic will be stopped, hence the counters will not count the events of interest. To prevent this, simply run a low-priority background process that will keep the processor busy during the period of interest.

PerfEvtSel0–PerfEvtSel3 Registers — C001_0000h, C001_0001h, C001_0002h, C001_0003h



)

Bit	Name	Function	R/W
63–32	reserved	RAZ	
31–24	CNT_MASK	Counter Mask	R/W
23	INV	Invert Counter Mask	R/W
22	EN	Enable Counter	R/W
21	reserved	SBZ	
20	INT	Enable APIC Interrupt	R/W
19	PC	Pin Control	R/W
18	E	Edge Detect	R/W
17	OS	Operating-System Mode	R/W
16	USR	User Mode	R/W
15–8	UNIT_MASK	Event Qualification	R/W
7–0	EVENT_MASK	Unit and Event Selection	R/W

Field Descriptions

Event Selection (EVENT_SELECT)—Bits 7–0. The EVENT_SELECT field specifies the event or event duration in a processor unit to be counted by the corresponding PerfCtr*i* register.

Event Qualification (UNIT_MASK)—Bits 15–8. Each UNIT_MASK bit further specifies or qualifies the event specified by EVENT_MASK. All events selected by UNIT_MASK are simultaneously monitored. Unless otherwise stated, the UNIT_MASK values shown may be combined (logically ORed) to select any desired combination of the sub-events for a given event. In some cases, certain combinations can result in misleading counts, or the UNIT_MASK value is an ordinal rather than a bit mask. These situations are described where applicable, or should be obvious from the event descriptions. For events where no UNIT_MASK table is shown, the UNIT_MASK is not applicable and may be set to zeros.

User Mode (USR)—Bit 16. Count events in user mode (at CPL > 0).

Operating-System Mode (OS)—Bit 17. Count events in operating-system mode (at CPL = 0).

Edge Detect (E)—Bit 18. 1=Edge detection. 0=Level detection.

Pin Control (PC)—Bit 19. With a value of 1, toggles the performance monitor pin PM*i* when PerfCtr*i* register overflows. With a value of 0, toggles the performance monitor pin PM*i* each time PerfCtr*i* register is incremented.

Enable APIC Interrupt (INT)—Bit 20. Enables APIC interrupt when PerfCtr*i* register overflows.

Enable Counter (EN)—Bit 22. Enables performance monitor counter.

Invert Counter Mask (INV)—Bit 23. See CNT_MASK.

Counter Mask (CNT_MASK)—Bits 31–24. a) When CNT_MASK is 0, the corresponding PerfCtr*i* register is incremented by the number of events occurring in a clock cycle. Maximum number of events in one cycle is 3. b) When CNT_MASK values are from 1 to 3 and INV is 0, the corresponding PerfCtr*i* register is incremented by 1, if the number of events occurring in a clock cycle is greater than or equal to CNT_MASK. When CNT_MASK values are from 1 to 3 and INV is 1, the corresponding PerfCtr*i* register is incremented by 1, if the number of events occurring in a clock cycle is less than CNT_MASK. c) CNT_MASK values from 4 to 255 are reserved.

11.2.1 Performance Monitor Events

Note: ***Speculative vs. Retired****: Several events may include speculative activity, meaning they may be associated with false-path instructions that are ultimately discarded due to a branch misprediction. Events associated with Retire reflect actual program execution. For events where the distinction may matter, these are explicitly labeled as one or the other.*

11.2.1.1 Floating Point Unit Events

Event Select: 00h - Dispatched FPU Operations

The number of operations (uops) dispatched to the FPU execution pipelines. This event reflects how busy the FPU pipelines are. This includes all operations done by x87, MMX® and SSE instructions, including moves. Each increment represents a one-cycle dispatch event; packed 128-bit SSE operations count as two ops; scalar operations count as one. Speculative. (See also event CBh).

Note: Since this event includes non-numeric operations it is not suitable for measuring MFLOPs.

UNIT_MASK	
01h	Add pipe ops
02h	Multiply pipe ops

UNIT_MASK	
04h	Store pipe ops
08h	Add pipe load ops
10h	Multiply pipe load ops
20h	Store pipe load ops

Event Select: 01h - Cycles with no FPU Ops Retired

The number of cycles in which no FPU operations were retired. Invert this (set the Invert control bit in the event select MSR) to count cycles in which at least one FPU operation was retired.

Event Select: 02h - Dispatched Fast Flag FPU Operations

The number of FPU operations that use the fast flag interface (e.g. FCOMI, COMISS, COMISD, UCOMISS, UCOMISD). Speculative.

11.2.1.2 Load/Store Unit and TLB Events

Event Select: 20h - Segment Register Loads

The number of segment register loads performed.

UNIT_MASK	
01h	ES
02h	CS
04h	SS
08h	DS
10h	FS
20h	GS
40h	HS

Event Select: 21h - Pipeline restart due to self-modifying code

The number of pipeline restarts that were caused by self-modifying code (a store that hits any instruction that's been fetched for execution beyond the instruction doing the store).

Event Select: 22h - Pipeline restart due to probe hit

The number of pipeline restarts caused by an invalidating probe hitting on a speculative out-of-order load.

Event Select: 23h - LS Buffer 2 Full

The number of cycles that the LS2 buffer is full. This buffer holds stores waiting to retire as well as requests that missed the data cache and are waiting on a refill. This condition will stall further data cache accesses, although such stalls may be overlapped by independent instruction execution.

Event Select: 24h - Locked Operations

This event covers locked operations performed and their execution time. The execution time represented by the cycle counts is typically overlapped to a large extent with other instructions. The non-speculative *cycles* event is suitable for event-based profiling of lock operations that tend to miss in the cache.

UNIT_MASK	
01h	The number of locked instructions executed
02h	The number of cycles spent in speculative phase
04h	The number of cycles spent in non-speculative phase (including cache miss penalty)

Event Select: 65h - Memory Requests by Type

These events reflect accesses to uncachable (UC) or write-combining (WC) memory regions (as defined by MTRR or PAT settings) and Streaming Store activity to WB memory. Both the WC and Streaming Store events reflect Write Combining buffer flushes, not individual store instructions. WC buffer flushes which typically consist of one 64-byte write to the system for each flush (assuming software typically fills a buffer before it gets flushed). A partially-filled buffer will require two or more smaller writes to the system. The WC event reflects flushes of WC buffers that were filled by stores to WC memory or streaming stores to WB memory. The Streaming Store event reflects only flushes due to streaming stores (which are typically only to WB memory). The difference between counts of these two events reflects the true amount of write events to WC memory.

UNIT_MASK	
01h	Requests to non-cacheable (UC) memory
02h	Requests to write-combining (WC) memory or WC buffer flushes to WB memory
80h	Streaming store (SS) requests

11.2.1.3 Data Cache Events

Event Select: 40h - Data Cache Accesses

The number of accesses to the data cache for load and store references. This may include certain microcode scratchpad accesses, although these are generally rare. Each increment represents an eight-byte access, although the instruction may only be accessing a portion of that. Speculative.

Event Select: 41h - Data Cache Misses

The number of data cache references which missed in the data cache. Speculative.

Except in the case of streaming stores, only the first miss for a given line is included - access attempts by other instructions while the refill is still pending are not included in this event. So in the absence of streaming stores, each event reflects one 64-byte cache line refill, and counts of this event are the same as, or very close to, the combined count for event 42h.

Streaming stores however will cause this event for every such store, since the target memory is not refilled into the cache. Hence this event should not be used as an indication of data cache refill activity - event 42h should be used for such measurements. (See event 65h for an indication of streaming store activity.) A large difference between events 41h (with all UNIT_MASK bits set) and 42h would be due mainly to streaming store activity.

Event Select: 42h - Data Cache Refills from L2 or System

The number of data cache refills satisfied from the L2 cache (and/or the system), per the UNIT_MASK. UNIT_MASK bits 4:1 allow a breakdown of refills from the L2 by coherency state. UNIT_MASK bit 0 reflects refills which missed in the L2, and provides the same measure as the combined sub-events of event 43h. Each increment reflects a 64-byte transfer. Speculative.

UNIT_MASK	
01h	Refill from System
02h	Shared-state line from L2
04h	Exclusive-state line from L2
08h	Owned-state line from L2
10h	Modified-state line from L2

Event Select: 43h - Data Cache Refills from System

The number of L1 cache refills satisfied from the system (system memory or another cache), as opposed to the L2. The UNIT_MASK selects lines in one or more specific coherency states. Each increment reflects a 64-byte transfer. Speculative.

UNIT_MASK	
01h	Invalid
02h	Shared
04h	Exclusive
08h	Owned
10h	Modified

Event Select: 44h - Data Cache Lines Evicted

The number of L1 data cache lines written to the L2 cache or system memory, having been displaced by L1 refills. The UNIT_MASK may be used to count only victims in specific coherency states. Each increment represents a 64-byte transfer. Speculative.

In most cases, L1 victims are moved to the L2 cache, displacing an older cache line there. Lines brought into the data cache by PrefetchNTA instructions, however, are evicted directly to system memory (if dirty) or invalidated (if clean). There is no provision for measuring this component by itself. The Invalid case (UNIT_MASK value 01h) reflects the replacement of lines that would have been invalidated by probes for write operations from another processor or DMA activity.

UNIT_MASK	
01h	Invalid
02h	Shared
04h	Exclusive
08h	Owned
10h	Modified

Event Select: 45h - L1 DTLB Miss and L2 DTLB Hit

The number of data cache accesses that miss in the L1 DTLB and hit in the L2 DTLB. Speculative.

Event Select: 46h - L1 DTLB and L2 DTLB Miss

The number of data cache accesses that miss in both the L1 and L2 DTLBs. Speculative.

Event Select: 47h - Misaligned Accesses

The number of data cache accesses that are misaligned. These are accesses which cross an eight-byte boundary. They incur an extra cache access (reflected in event 40h), and an extra cycle of latency on reads. Speculative.

Event Select: 48h - Microarchitectural Late Cancel of an Access

Event Select: 49h - Microarchitectural Early Cancel of an Access

Event Select: 4Ah - Single-bit ECC Errors Recorded by Scrubber

The number of single-bit errors corrected by either of the error detection/correction mechanisms in the data cache.

UNIT_MASK	
01h	Scrubber error
02h	Piggyback scrubber errors

Event Select: 4Bh - Prefetch Instructions Dispatched

The number of prefetch instructions dispatched by the decoder. Speculative. Such instructions may or may not cause a cache line transfer. Any Dcache and L2 accesses, hits and misses by prefetch instructions are included in these types of events.

UNIT_MASK	
01h	Load (Prefetch, PrefetchT0/T1/T2)
02h	Store (PrefetchW)
04h	NTA (PrefetchNTA)

Event Select: 4Ch - DCACHE Misses by Locked Instructions

The number of data cache misses incurred by locked instructions. (The total number of locked instructions may be obtained from event 24h.)

Such misses may be satisfied from the L2 or system memory, but there is no provision for distinguishing between the two. When used for event-based profiling, this event will tend to occur very close to the offending instructions. (See also event 24h.) This event is also included in the basic Dcache miss event (41h).

UNIT_MASK	
02h	Data cache misses by locked instructions

11.2.1.4 L2 Cache and System Interface Events

Event Select: 67h - Data Prefetcher

These events reflect requests made by the data prefetcher. UNIT_MASK bit 1 counts total prefetch requests, while bit 0 counts requests where the target block is found in the L2 or data cache. The difference between the two represents actual data read (in units of 64-byte cache lines) from the system by the prefetcher. This is also included in the count of event 7Fh, UNIT_MASK bit 0 (combined with other L2 fill events).

UNIT_MASK	
01h	Cancelled prefetches
02h	Prefetch attempts

Event Select: 6Ch - System Read Responses by Coherency State

The number of responses from the system for cache refill requests. The UNIT_MASK may be used to select specific cache coherency states. Each increment represents one 64-byte cache line transferred from the system (DRAM or another cache, including another core on the same node) to the data cache, instruction cache or L2 cache (for data prefetcher and TLB table walks). Modified-state

responses may be for Dcache store miss refills, PrefetchW software prefetches, hardware prefetches for a store-miss stream, or Change-to-Dirty requests that get a dirty (Owned) probe hit in another cache. Exclusive responses may be for any Icache refill, Dcache load miss refill, other software prefetches, hardware prefetches for a load-miss stream, or TLB table walks that miss in the L2 cache; Shared responses may be for any of those that hit a clean line in another cache.

UNIT_MASK	
01h	Exclusive
02h	Modified
04h	Shared

Event Select: 6Dh - Quadwords Written to System

The number of quadword (8-byte) data transfers from the processor to the system. These may be part of a 64-byte cache line writeback or a 64-byte dirty probe hit response, each of which would cause eight increments; or a partial or complete Write Combining buffer flush (Sized Write), which could cause from one to eight increments.

UNIT_MASK	
01h	Quadword write transfer

Event Select: 7Dh - Requests to L2 Cache

The number of requests to the L2 cache for Icache or Dcache fills, or page table lookups for the TLB. These events reflect only read requests to the L2; writes to the L2 are indicated by event 7Fh. These include some amount of retries associated with address or resource conflicts. Such retries tend to occur more as the L2 gets busier, and in certain extreme cases (such as large block moves that overflow the L2) these extra requests can dominate the event count.

These extra requests are not a direct indication of performance impact - they simply reflect opportunistic accesses that don't complete. But because of this, they are not a good indication of actual cache line movement. The Icache and Dcache miss and refill events (81h, 82h, 83h, 41h, 42h, 43h) provide a more accurate indication of this, and are the preferred way to measure such traffic.

UNIT_MASK	
01h	IC fill
02h	DC fill
04h	TLB fill (page table walks)
08h	Tag snoop request
10h	Cancelled request

Event Select: 7Eh - L2 Cache Misses

The number of requests that miss in the L2 cache. This may include some amount of speculative activity, as well as some amount of retried requests as described in event 7Dh. The IC-fill-miss and DC-fill-miss events tend to mirror the Icache and Dcache refill-from-system events (83h and 43h, respectively), and tend to include more speculative activity than those events.

UNIT_MASK	
01h	IC fill
02h	DC fill (includes possible replays, whereas event 41h does not)
04h	TLB page table walk

Event Select: 7Fh - L2 Fill/Writeback

The number of lines written into the L2 cache due to victim writebacks from the Icache or Dcache, TLB page table walks and the hardware data prefetcher (UNIT_MASK bit 0); or writebacks of dirty lines from the L2 to the system (UNIT_MASK bit 1). Each increment represents a 64-byte cache line transfer.

Note: *Victim writebacks from the Dcache may be measured separately using event 44h. However this is not quite the same as the Dcache component of event 7Fh, the main difference being PrefetchNTA lines. When these are evicted from the Dcache due to replacement, they are written out to system memory (if dirty) or simply invalidated (if clean), rather than being moved to the L2 cache.*

UNIT_MASK	
01h	L2 fills (victims from L1 caches, TLB page table walks and data prefetches)
02h	L2 Writebacks to system.

11.2.1.5 Instruction Cache Events

All instruction cache events are speculative.

Event Select: 80h - Instruction Cache Fetches

The number of instruction cache accesses by the instruction fetcher. Each access is an aligned 16 byte read, from which a varying number of instructions may be decoded.

Event Select: 81h - Instruction Cache Misses

The number of instruction fetches that miss in the instruction cache. This is typically equal to or very close to the sum of events 82h and 83h. Each miss results in a 64-byte cache line refill.

Event Select: 82h - Instruction Cache Refills from L2

The number of instruction cache refills satisfied from the L2 cache. Each increment represents one 64-byte cache line transfer.

Event Select: 83h - Instruction Cache Refills from System

The number of instruction cache refills from system memory (or another cache). Each increment represents one 64-byte cache line transfer.

Event Select: 84h - L1 ITLB Miss, L2 ITLB Hit

The number of instruction fetches that miss in the L1 ITLB but hit in the L2 ITLB.

Event Select: 85h - L1 ITLB Miss, L2 ITLB Miss

The number of instruction fetches that miss in both the L1 and L2 ITLBs.

Event Select: 86h - Pipeline Restart Due to Instruction Stream Probe

The number of pipeline restarts caused by invalidating probes that hit on the instruction stream currently being executed. This would happen if the active instruction stream was being modified by another processor in an MP system - typically a highly unlikely event.

Event Select: 87h - Instruction Fetch Stall

The number of cycles the instruction fetcher is stalled. This may be for a variety of reasons such as branch predictor updates, unconditional branch bubbles, far jumps and cache misses, among others. May be overlapped by instruction dispatch stalls or instruction execution, such that these stalls don't necessarily impact performance.

Event Select: 88h - Return Stack Hits

The number of near return instructions (RET or RET Iw) that get their return address from the return address stack (i.e. where the stack has not gone empty). This may include cases where the address is incorrect (return mispredicts). This may also include speculatively executed false-path returns. Return mispredicts are typically caused by the return address stack underflowing, however they may also be caused by an imbalance in calls vs. returns, such as doing a call but then popping the return address off the stack.

Note: *This event cannot be reliably compared with events C9h and CAh (such as to calculate percentage of return mispredicts due to an empty return address stack), since it may include speculatively executed false-path returns that are not included in those retire-time events.*

Event Select: 89h - Return Stack Overflows

The number of (near) call instructions that cause the return address stack to overflow. When this happens, the oldest entry is discarded. This count may include speculatively executed calls.

11.2.1.6 Execution Unit Events

Event Select: 26h - Retired CLFLUSH Instructions

The number of CLFLUSH instructions retired.

Event Select: 27h - Retired CPUID Instructions

The number of CPUID instructions retired.

Event Select: 76h - CPU Clocks not Halted

The number of clocks that the CPU is not in a halted state (due to STPCLK or a HALT instruction). Note: this event allows system idle time to be automatically factored out from IPC (or CPI) measurements, providing the OS halts the CPU when going idle. If the OS goes into an idle loop rather than halting, such calculations will be influenced by the IPC of the idle loop.

Event Select: C0h - Retired Instructions

The number of instructions retired (execution completed and architectural state updated). This count includes exceptions and interrupts - each exception or interrupt is counted as one instruction.

Event Select: C1h - Retired uops

The number of micro-ops retired. This includes all processor activity (instructions, exceptions, interrupts, microcode assists, etc.).

Event Select: C2h - Retired Branch Instructions

The number of branch instructions retired. This includes all types of architectural control flow changes, including exceptions and interrupts.

Event Select: C3h - Retired Mispredicted Branch Instructions

The number of branch instructions retired, of any type, that were not correctly predicted. This includes those for which prediction is not attempted (far control transfers, exceptions and interrupts).

Event Select: C4h - Retired Taken Branch Instructions

The number of taken branches that were retired. This includes all types of architectural control flow changes, including exceptions and interrupts.

Event Select: C5h - Retired Taken Branch Instructions Mispredicted

The number of retired taken branch instructions that were mispredicted.

Event Select: C6h - Retired Far Control Transfers

The number of far control transfers retired including far call/jump/return, IRET, SYSCALL and SYSRET, plus exceptions and interrupts. Far control transfers are not subject to branch prediction.

Event Select: C7h - Retired Branch Resyncs

The number of resync branches. These reflect pipeline restarts due to certain microcode assists and events such as writes to the active instruction stream, among other things. Each occurrence reflects a restart penalty similar to a branch mispredict. Relatively rare.

Event Select: C8h - Retired Near Returns

The number of near return instructions (RET or RET Iw) retired.

Event Select: C9h - Retired Near Returns Mispredicted

The number of near returns retired that were not correctly predicted by the return address predictor. Each such mispredict incurs the same penalty as a mispredicted conditional branch instruction.

Event Select: CAh - Retired Indirect Branches Mispredicted

The number of indirect branch instructions retired where the target address was not correctly predicted.

Event Select: CBh - Retired MMX®/FP Instructions

The number of MMX®, SSE or X87 instructions retired. The UNIT_MASK allows the selection of the individual classes of instructions as given in the table. Each increment represents one complete instruction.

Note: *Since this event includes non-numeric instructions it is not suitable for measuring MFLOPS.*

UNIT_MASK	
01h	x87 instructions
02h	MMX® and 3DNow!™ instructions
04h	Packed SSE and SSE2 instructions
08h	Scalar SSE and SSE2 instructions

Event Select: CCh - Retired Fastpath Double Op Instructions

UNIT_MASK	
01h	With low op in position 0
02h	With low op in position 1
04h	With low op in position 2

Event Select: CDh - Interrupts-Masked Cycles

The number of processor cycles where interrupts are masked (EFLAGS.IF = 0). Using edge-counting with this event will give the number of times IF is cleared; dividing the cycle-count value by this value gives the average length of time that interrupts are disabled on each instance. Compare the edge count with event CFh to determine how often interrupts are disabled for interrupt handling vs. other reasons (e.g. critical sections).

Event Select: CEh - Interrupts-Masked Cycles with Interrupt Pending

The number of processor cycles where interrupts are masked (EFLAGS.IF = 0) and an interrupt is pending. Using edge-counting with this event and comparing the resulting count with the edge count for event CDh gives the proportion of interrupts for which handling is delayed due to prior interrupts being serviced, critical sections, etc. The cycle count value gives the total amount of time for such delays. The cycle count divided by the edge count gives the average length of each such delay.

Event Select: CFh - Interrupts Taken

The number of hardware interrupts taken. This does not include software interrupts (INT n instruction).

Event Select: D0h - Decoder Empty

The number of processor cycles where the decoder has nothing to dispatch (typically waiting on an instruction fetch that missed the Icache, or for the target fetch after a branch mispredict).

Event Select: D1h - Dispatch Stalls

The number of processor cycles where the decoder is stalled for any reason (has one or more instructions ready but can't dispatch them due to resource limitations in execution). This is the combined effect of events D2h - DAh, some of which may overlap; this event reflects the net stall cycles. The more common stall conditions (events D5h, D6h, D7h, D8h, and to a lesser extent D2) may overlap considerably. The occurrence of these stalls is highly dependent on the nature of the code being executed (instruction mix, memory reference patterns, etc.).

Event Select: D2h - Dispatch Stall for Branch Abort to Retire

The number of processor cycles the decoder is stalled waiting for the pipe to drain after a mispredicted branch. This stall occurs if the corrected target instruction reaches the dispatch stage before the pipe has emptied. See also event D1h.

Event Select: D3h - Dispatch Stall for Serialization

The number of processor cycles the decoder is stalled due to a serializing operation, which waits for the execution pipeline to drain. Relatively rare; mainly associated with system instructions. See also event D1h.

Event Select: D4h - Dispatch Stall for Segment Load

The number of processor cycles the decoder is stalled due to a segment load instruction being encountered while execution of a previous segment load operation is still pending. Relatively rare except in 16-bit code. See also event D1h.

Event Select: D5h - Dispatch Stall for Reorder Buffer Full

The number of processor cycles the decoder is stalled because the reorder buffer is full. May occur simultaneously with certain other stall conditions; see event D1h.

Event Select: D6h - Dispatch Stall for Reservation Station Full

The number of processor cycles the decoder is stalled because a required integer unit reservation stations is full. May occur simultaneously with certain other stall conditions; see event D1h.

Event Select: D7h - Dispatch Stall for FPU Full

The number of processor cycles the decoder is stalled because the scheduler for the Floating Point Unit is full. This condition can be caused by a lack of parallelism in FP-intensive code, or by cache misses on FP operand loads (which could also show up as event D8h instead, depending on the nature of the instruction sequences). May occur simultaneously with certain other stall conditions; see event D1h

Event Select: D8h - Dispatch Stall for LS Full

The number of processor cycles the decoder is stalled because the Load/Store Unit is full. This generally occurs due to heavy cache miss activity. May occur simultaneously with certain other stall conditions; see event D1h.

Event Select: D9h - Dispatch Stall Waiting for All Quiet

The number of processor cycles the decoder is stalled waiting for all outstanding requests to the system to be resolved. Relatively rare; associated with certain system instructions and types of interrupts. May partially overlap certain other stall conditions; see event D1h.

Event Select: DAh - Dispatch Stall for Far Transfer or Resync to Retire

The number of processor cycles the decoder is stalled waiting for the execution pipeline to drain before dispatching the target instructions of a far control transfer or a Resync (an instruction stream restart associated with certain microcode assists). Relatively rare; does not overlap with other stall conditions. See also event D1h.

Event Select: DBh - FPU Exceptions

The number of floating point unit exceptions for microcode assists. The UNIT_MASK may be used to isolate specific types of exceptions.

UNIT_MASK	
01h	x87 reclass microfaults
02h	SSE retype microfaults
04h	SSE reclass microfaults
08h	SSE and x87 microtraps

Event Select: DCh - DR0 Breakpoint Matches

The number of matches on the address in breakpoint register DR0, per the breakpoint type specified in DR7. The breakpoint does not have to be enabled. Each instruction breakpoint match incurs an overhead of about 120 cycles; load/store breakpoint matches do not incur any overhead.

Event Select: DDh - DR1 Breakpoint Matches

The number of matches on the address in breakpoint register DR1. See notes for event DCh.

Event Select: DEh - DR2 Breakpoint Matches

The number of matches on the address in breakpoint register DR2. See notes for event DCh.

Event Select: DFh - DR3 Breakpoint Matches

The number of matches on the address in breakpoint register DR3. See notes for event DCh.

11.2.1.7 Memory Controller Events

Event Select: E0h - DRAM Accesses

The number of memory accesses performed by the local DRAM controller. The UNIT_MASK may be used to isolate the different DRAM page access cases. Page miss cases incur an extra latency to open a page; page conflict cases incur both a page-close as well as page-open penalties. These penalties may be overlapped by DRAM accesses for other requests and don't necessarily represent lost DRAM bandwidth. The associated penalties are as follows:

Page miss: Trcd (DRAM RAS-to-CAS delay)

Page conflict: Trp + Trcd (DRAM row-precharge time plus RAS-to-CAS delay)

Each DRAM access represents one 64-byte block of data transferred if the DRAM is configured for 64-byte granularity, or one 32-byte block if the DRAM is configured for 32-byte granularity. (The latter is only applicable to single-channel DRAM systems, which may be configured either way.)

UNIT_MASK	
01h	Page hit
02h	Page Miss
04h	Page Conflict

Event Select: E1 - Memory Controller Page Table Overflows

The number of page table overflows in the local DRAM controller. This table maintains information about which DRAM pages are open. An overflow occurs when a request for a new page arrives when the maximum number of pages are already open. Each occurrence reflects an access latency penalty equivalent to a page conflict.

Event Select: E3 - Memory Controller Turnarounds

The number of turnarounds on the local DRAM data bus. The UNIT_MASK may be used to isolate the different cases. These represent lost DRAM bandwidth, which may be calculated as follows (in bytes per occurrence):

DIMM turnaround: DRAM_width_in_bytes * 2 edges_per_memclk * 2

R/W turnaround: DRAM_width_in_bytes * 2 edges_per_memclk * 1

R/W turnaround: DRAM_width_in_bytes * 2 edges_per_memclk * (Tcl-1)

where DRAM_width_in_bytes is 8 or 16 (for single- or dual-channel systems), and Tcl is the CAS latency of the DRAM in memory system clock cycles (where the memory clock for DDR-400, or PC3200 DIMMS, for example, would be 200 MHz).

UNIT_MASK	
01h	DIMM (chip select) turnaround
02h	Read to write turnaround
04h	Write to read turnaround

Event Select: E4h - Memory Controller Bypass Counter Saturation

UNIT_MASK	
01h	Memory controller high priority bypass
02h	Memory controller low priority bypass
04h	DRAM controller interface bypass
08h	DRAM controller queue bypass

Event Select: E5 - Sized Blocks

Sized Read/Write activity: The Sized Read/Write events reflect 32- or 64-byte transfers (as opposed to other sizes which could be anywhere between 1 and 64 bytes), from either the processor or the Hostbridge (on any node in an MP system). Such accesses from the processor would be due only to write combining buffer flushes, where 32-byte accesses would reflect flushes of partially-filled buffers. Event 65h provides a count of sized write requests associated with WC buffer flushes; comparing that with counts for these events (providing there is very little Hostbridge activity at the same time) will give an indication of how efficiently the write combining buffers are being used. Event 65h may also be useful in factoring out WC flushes when comparing these events with the Upstream Requests component of event ECh.

UNIT_MASK	
04h	32-byte Sized Writes
08h	64-byte Sized Writes
10h	32-byte Sized Reads
20h	64-byte Sized Reads

Event Select: E8h - Thermal Status and ECC Errors

UNIT_MASK	
01h	Number of clocks CPU is active when HTC is active
02h	Number of clocks CPU clock is inactive when HTC is active
04h	Number of clocks when die temperature is higher than the software high temperature threshold
08h	Number of clocks when high temperature threshold was exceeded
80h	Number of correctable and Uncorrectable DRAM ECC errors

Event Select: E9h - CPU/IO Requests to Memory/IO

These events reflect request flow between units and nodes, as selected by the UNIT_MASK. The UNIT_MASK is divided into two fields: request type (CPU or I/O access to I/O or Memory) and source/target location (local vs. remote). One or more requests types must be enabled via bits 3:0, and

at least one source and one target location must be selected via bits 7:4. Each event reflects a request of the selected type(s) going from the selected source(s) to the selected target(s).

Not all possible paths are supported. The following table shows the UNIT_MASK values that are valid for each request type:

Source/Target	CPU to Mem	CPU to I/O	I/O to Mem	I/O to I/O
Local -> Local	A8h	A4h	A2h	A1h
Local -> Remote	98h	94h	92h	91h
Remote -> Local	-	64h	-	61h
Remote -> Remote	-	-	-	-

Any of the mask values shown may be logically ORed to combine the events. For instance, local CPU requests to both local and remote nodes would be A8h | 98h = B8h. Any CPU to any I/O would be A4h | 94h | 64h = F4h (but remote CPU to remote I/O requests would not be included).

Note: *It is not possible to tell from these events how much data is going in which direction, as there is no distinction between reads and writes. Also, particularly for I/O, the requests may be for varying amounts of data, anywhere from one to sixty-four bytes. Event E5h provides an indication of 32- and 64-byte read and write transfers for such requests (although from the target point of view). For a direct measure of the amount and direction of data flowing between nodes, use events F6h, F7h and F8h.*

UNIT_MASK	
01h	I/O to I/O
02h	I/O to Mem
04h	CPU to I/O
08h	CPU to Mem
10h	To remote node
20h	To local node
40h	From remote node
80h	From local node

Event Select: EAh - Cache Block Commands

The number of requests made to the system for cache line transfers or coherency state changes, by request type. Each increment represents one cache line transfer, except for Change-to-Dirty. If a Change-to-Dirty request hits on a line in another processor's cache that's in the Owned state, it will

cause a cache line transfer, otherwise there is no data transfer associated with Change-to-Dirty requests.

UNIT_MASK	
01h	Victim Block (Writeback)
04h	Read Block (Dcache load miss refill)
08h	Read Block Shared (Icache refill)
10h	Read Block Modified (Dcache store miss refill)
20h	Change to Dirty (first store to clean block already in cache)

Event Select: EBh - Sized Commands

The number of Sized Read/Write commands handled by the System Request Interface (local processor and hostbridge interface to the system). These commands may originate from the processor or hostbridge. Typical uses of the various Sized Read/Write commands are given in the UNIT_MASK table. See also event E5h, which covers commonly-used block sizes for these requests, and event ECh, which provides a separate measure of Hostbridge accesses.

UNIT_MASK		Typical Usage
01h	NonPosted SzWr Byte (1-32 bytes)	Legacy or mapped I/O, typically 1-4 bytes
02h	NonPosted SzWr Dword (1-16 dwords)	Legacy or mapped I/O, typically 1 dword
04h	Posted SzWr Byte (1-32 bytes)	Sub-cache-line DMA writes, size varies; also flushes of partially-filled Write Combining buffer
08h	Posted SzWr Dword (1-16 dwords)	Block-oriented DMA writes, often cache-line sized; also processor Write Combining buffer flushes
10h	SzRd Byte (4 bytes)	Legacy or mapped I/O
20h	SzRd Dword (1-16 dwords)	Block-oriented DMA reads, typically cache-line size
40h	RdModWr	

Event Select: ECh - Probe Responses and Upstream Requests

This covers two unrelated sets of events: cache probe results, and requests received by the Hostbridge from devices on non-coherent links.

Probe results: These events reflect the results of probes sent from a memory controller to local caches. They provide an indication of the degree data and code is shared between processors (or moved between processors due to process migration). The dirty-hit events indicate the transfer of a 64-byte cache line to the requestor (for a read or cache refill) or the target memory (for a write). The system bandwidth used by these, in terms of bytes per unit of time, may be calculated as 64 times the event count, divided by the elapsed time. Sized writes to memory that cover a full cache line do not

incur this cache line transfer -- they simply invalidate the line and are reported as clean hits. Cache line transfers will occur for Change2Dirty requests that hit cache lines in the Owned state. (Such cache lines are counted as Modified-state refills for event 6Ch, System Read Responses.)

Upstream requests: The upstream read and write events reflect requests originating from a device on a local non-coherent HyperTransport™ link. The two read events allow display refresh traffic in a UMA system to be measured separately from other DMA activity. Display refresh traffic will typically be dominated by 64-byte transfers. Non-display-related DMA accesses may be anywhere from 1 to 64 bytes in size, but may be dominated by a particular size such as 32 or 64 bytes, depending on the nature of the devices. Event E5h can provide a measure of 32- and 64-byte accesses by the hostbridge (possibly combined with write combining buffer flush activity from the processor, although that can be factored out via event 65h).

UNIT_MASK	
01h	Probe miss
02h	Probe hit clean
04h	Probe hit dirty without memory cancel (probed by Sized Write or Change2Dirty)
08h	Probe hit dirty with memory cancel (probed by DMA read or cache refill request)
10h	Upstream display refresh reads
20h	Upstream non-display refresh reads
40h	Upstream writes

Event Select: EEh - GART Events

These events reflect GART activity, and in particular allow one to calculate the GART TLB miss ratio as GART_miss_count divided by GART_aperture_hit_count. GART aperture accesses are typically from I/O devices as opposed to the processor, and generally from a 3D graphics accelerator, but can be from other devices when the GART is used as an IO MMU).

UNIT_MASK	
01h	GART aperture hit on access from CPU
02h	GART aperture hit on access from I/O
04h	GART miss

11.2.1.8 Crossbar Events

11.2.1.9 HyperTransport™ Interface Events

Event Select: F6h - HyperTransport Link 0 Transmit Bandwidth
F7h - HyperTransport Link 1 Transmit Bandwidth
F8h - HyperTransport Link 2 Transmit Bandwidth

The number of dwords transmitted (or unused, in the case of Nops) on the outgoing side of the HyperTransport links. The sum of all four subevents (all four UNIT_MASK bits set) directly reflects the maximum transmission rate of the link. Link utilization may be calculated by dividing the combined Command, Data and Buffer Release count (UNIT_MASK 07h) by that value plus the Nop count (UNIT_MASK 08h). Bandwidth in terms of bytes per unit time for any one component or combination of components is calculated by multiplying the count by four and dividing by elapsed time.

The Data event provides a direct indication of the flow of data around the system. Translating this link-based view into a source/target node based view requires knowledge of the system layout (i.e. which links connect to which nodes).

UNIT_MASK	
01h	Command dword sent
02h	Data dword sent
04h	Buffer release dword sent
08h	Nop dword sent (idle)

12 CPUID Function Registers

Processor feature capabilities and configuration information are provided through the CPUID instruction. Different information is accessed by (1) setting EAX as an index to the registers to be read, (2) executing the CPUID instruction, and (3) reading the results in EAX, EBX, ECX, and EDX. The phrase CPUID function X or CPUID FnX refers to the CPUID instruction when EAX is preloaded with X. Undefined function numbers return 0's in all 4 registers.

The following provides AMD Family 0Fh specific details about CPUID. See the CPUID Specification for further information.

CPUID Fn[8000_0000,0000_0000] AMD Authentic Identifier

Register	Bits	Description
EAX	31:0	Function 0000_0000h returns the largest CPUID standard-function input value supported by the processor implementation: 0000_0005h. Function 8000_0000h returns the largest CPUID extended-function input value supported by the processor implementation: 8000_0019h.
EBX, ECX, EDX	31:0	The 12 8-bit ASCII character codes to create the string "AuthenticAMD". EBX=6874_7541h "h t u A", ECX=444D_4163h "D M A c", EDX=6974_6E65h "i t n e

CPUID Fn[8000_0001,0000_0001]_EAX Family, Model, Feature Identifiers

This register provides identical information to Function 3, Offset FCh.

Family is an 8-bit value and is defined as: Family[7:0] = ({0000b,BaseFamily[3:0]} + ExtendedFamily[7:0]). E.g. If BaseFamily[3:0]=Fh and ExtendedFamily[7:0]=01h, then Family[7:0]=10h. This document applies only to family 0Fh processors.

Model is an 8-bit value and is defined as: Model[7:0] = {ExtendedModel[3:0], BaseModel[3:0]}. E.g. If ExtendedModel[3:0]=Eh and BaseModel[3:0]=8h, then Model[7:0] = E8h. Model numbers vary with product.

Bits	Description
31:28	Reserved.
27:20	**ExtendedFamily**: 00h.
19:16	**ExtendedModel**.
15:12	Reserved.
11:8	**BaseFamily**: Fh
7:4	**BaseModel**.
3:0	**Stepping**: processor stepping (revision) for a specific model.

CPUID Fn[0000_0001]_EBX LocalApicId, LogicalProcessorCount, CLFlush, 8BitBrandId

Bits	Description
31:24	**LocalApicId**: Provides the initial APICID (APIC Offset 20h) value in the three LSBs. After NodeId (Function 0: Offset 60h) has been initialized, changes to APICID do not effect the value of this register.
23:16	**LogicalProcessorCount**: If CPUID Fn[8000_0001, 0000_0001]_EDX[HTT] = 1, then this field indicates the number of CPU cores in the processor as CPUID Fn8000_0008[NC] + 1. Otherwise, this field is reserved.
15:8	**CLFlush**: CLFLUSH size in quadwords = 08h.
7:0	**8BitBrandId**: 8 bit brand ID = 00h. Indicates that the brand ID is in CPUID Fn8000_0001_EBX.

CPUID Fn[8000_0001]_EBX BrandId Identifier

Bits	Description
31:16	Reserved.
15:0	**BrandId**: brand ID.

CPUID Fn0000_0001_ECX Feature Identifiers

These fields are encoded as 1=Feature is supported by the processor; 0=Feature is not supported by the processor.

Bits	Description
31:14	Reserved.
13	**CMPXCHG16B**: CMPXCHG16B instruction = 1.
12:1	Reserved.
0	**SSE3**: SSE3 extensions = 1.

CPUID Fn8000_0001_ECX Feature Identifiers

These fields are encoded as 1=Feature is supported by the processor; 0=Feature is not supported by the processor.

Bits	Description
31:9	Reserved.
8	**3DNowPrefetch**: Prefetch and PrefetchW instructions = 1. See "Register Differences in Revisions of AMD NPT Family 0Fh Processors" on page 27 for revision information about this field.
7:5	Reserved.
4	**LockMovCr0**: LOCK MOV CR0 means MOV CR8 = 1.
3	**ExtApicSpace**: extended APIC register space = 1.
2	**SVM**: Secure Virtual Mode feature (setting varies by product).

Bits	Description
1	**CmpLegacy**: core multi-processing legacy mode (setting varies by product).
0	**LahfSahf**: LAHF/SAHF instructions =1.

CPUID Fn[8000_0001, 0000_0001]_EDX Feature Identifiers

These fields are encoded as 1=Feature is supported by the processor; 0=Feature is not supported by the processor. The value returned in EDX may be identical or different for Fn0000_0001 and Fn8000_0001, as indicated.

Bits	Function	Description
31	0000_0001	Reserved.
	8000_0001	**3DNow**: 3DNow!™ instructions = 1.
30	0000_0001	Reserved.
	8000_0001	**3DNowExt**: AMD extensions to 3DNow!™ instructions = 1.
29	0000_0001	Reserved.
	8000_0001	**LM**: long mode (may vary by product).
28	0000_0001	**HTT**: hyper-threading technology. This bit qualifies the meaning of CPUID Fn0000_0001_EBX[LogicalProcessorCount].
	8000_0001	Reserved.
27	0000_0001	Reserved.
	8000_0001	**RDTSCP**: RDTSCP instruction = 1.
26	0000_0001	**SSE2**: SSE2 extensions = 1.
	8000_0001	Reserved.
25	0000_0001	**SSE**: SSE extensions = 1.
	8000_0001	**FFXSR**: FXSAVE and FXRSTOR instruction optimizations = 1.
24	both	**FXSR**: FXSAVE and FXRSTOR instructions = 1.
23	both	**MMX**: MMX® instructions = 1.
22	0000_0001	Reserved.
	8000_0001	**MmxExt**: AMD extensions to MMX® instructions = 1.
21	both	Reserved.
20	0000_0001	Reserved.
	8000_0001	**NX**: no-execute page protection = 1.
19	0000_0001	**CLFSH**: CLFLUSH instruction = 1.
	8000_0001	Reserved.
18	both	Reserved.
17	both	**PSE36**: page-size extensions = 1.
16	both	**PAT**: page attribute table = 1.

Bits	Function	Description
15	both	**CMOV**: conditional move instructions, CMOV, FCOMI, FCMOV = 1.
14	both	**MCA**: machine check architecture, MCG_CAP = 1.
13	both	**PGE**: page global extension, CR4.PGE = 1.
12	both	**MTRR**: memory-type range registers = 1.
11	0000_0001	**SysEnterSysExit**: SYSENTER and SYSEXIT instructions = 1.
	8000_0001	**SysCallSysRet**: SYSCALL and SYSRET instructions = 1.
10	both	Reserved.
9	both	**APIC**: advanced programmable interrupt controller (APIC) exists and is enabled. This bit reflects the state of the APIC Base Address register (APIC_BAR) MSR 0000_001B[ApicEn].
8	both	**CMPXCHG8B**: CMPXCHG8B instruction = 1.
7	both	**MCE**: machine check exception, CR4.MCE = 1.
6	both	**PAE**: physical-address extensions (PAE) = 1.
5	both	**MSR**: AMD model-specific registers (MSRs), with RDMSR and WRMSR instructions = 1.
4	both	**TSE**: time stamp counter, RDTSC/RDTSCP instructions, CR4.TSD = 1.
3	both	**PSE**: page-size extensions (4 MB pages) = 1.
2	both	**DE**: debugging extensions, IO breakpoints, CR4.DE = 1.
1	both	**VME**: virtual-mode enhancements = 1.
0	both	**FPU**: x87 floating point unit on-chip = 1.

CPUID Fn8000_000[4, 3, 2] Processor Name String Identifier

These return the ASCII string corresponding to the processor name, stored in MSR C001_00[35:30]. The MSRs are mapped to these registers as follows:
Function 8000_0002: {EDX, ECX, EBX, EAX} == {MSRC001_0031, MSRC001_0030};
Function 8000_0003: {EDX, ECX, EBX, EAX} == {MSRC001_0033, MSRC001_0032};
Function 8000_0004: {EDX, ECX, EBX, EAX} == {MSRC001_0035, MSRC001_0034};

CPUID Fn8000_0005 TLB and L1 Cache Identifiers

This provides the processor's first level cache and TLB characteristics for each CPU core. The associativity fields returned are encoded as follows:

00h Reserved.
01h Direct mapped.
02h - FEh Specifies the associativity; e.g., 04h would indicate a 4-way associativity.
FFh Fully associative.

Register	Bits	Description
EAX	31:24	Data TLB associativity for 2 MB and 4 MB pages = FFh.

Register	Bits	Description
EAX	23:16	Data TLB number of entries for 2 MB and 4 MB pages = 8. The value returned is for the number of entries available for the 2 MB page size; 4 MB pages require two 2 MB entries, so the number of entries available for the 4 MB page size is one-half the returned value.
EAX	15:8	Instruction TLB associativity for 2 MB and 4 MB pages = FFh.
EAX	7:0	Instruction TLB number of entries for 2 MB and 4 MB pages = 8. The value returned is for the number of entries available for the 2 MB page size; 4 MB pages require two 2 MB entries, so the number of entries available for the 4 MB page size is one-half the returned value.
EBX	31:24	Data TLB associativity for 4 KB pages = FFh.
EBX	23:16	Data TLB number of entries for 4 KB pages = 32.
EBX	15:8	Instruction TLB associativity for 4 KB pages = FFh.
EBX	7:0	Instruction TLB number of entries for 4 KB pages = 32.
ECX	31:24	L1 data cache size in KB = 64.
ECX	23:16	L1 data cache associativity = 2.
ECX	15:8	L1 data cache lines per tag = 1.
ECX	7:0	L1 data cache line size in bytes = 64.
EDX	31:24	L1 instruction cache size in KB = 64.
EDX	23:16	L1 instruction cache associativity = 2.
EDX	15:8	L1 instruction cache lines per tag = 1.
EDX	7:0	L1 instruction cache line size in bytes = 64.

CPUID Fn8000_0006 L2 Cache Identifiers

This provides the processor's second level cache and TLB characteristics for each CPU core.

The presence of a unified L2 TLB is indicated by a value of 0000h in the upper 16 bits of the EAX and EBX registers. The unified L2 TLB information is contained in the lower 16 bits of these registers.

The associativity fields are encoded as follows:

0h: The L2 cache or TLB is disabled.
1h: Direct mapped.
2h: 2-way associative.
4h: 4-way associative.
6h: 8-way associative.
8h: 16-way associative.
Fh: Fully associative.

All other encodings are reserved.

Register	Bits	Description
EAX	31:0	The L2 TLB has no entries for 2 MByte and 4 MByte pages. 0000_000h
EBX	31:24	L2 data TLB associativity for 4 KB pages = 4.
EBX	23:16	L2 data TLB number of entries for 4 KB pages = 512.
EBX	15:8	L2 instruction TLB associativity for 4 KB pages = 4.
EBX	7:0	L2 instruction TLB number of entries for 4 KB pages = 512.
ECX	31:16	L2 cache size in KB (varies with product). May be one of 128, 256, 512, or 1024.
ECX	15:12	L2 cache associativity = 8.
ECX	11:8	L2 cache lines per tag = 1.
ECX	7:0	L2 cache line size in bytes = 64.
EDX	31:0	Reserved.

CPUID Fn8000_0007 Advanced Power Management Information

This provides advanced power management feature identifiers. Unless otherwise specified, these bits are encoded as 1=Feature is supported by the processor; 0=Feature is not supported by the processor.

Register	Bits	Description
EAX, EBX, ECX	31:0	Reserved.
EDX	31:7	Reserved.
EDX	6	**100MhzSteps**: 100 Mhz multiplier Control = 1. See "Register Differences in Revisions of AMD NPT Family 0Fh Processors" on page 27 for revision information about this field.
EDX	5	**STC**: software thermal control (STC) is supported (support may vary by product).
EDX	4	**TM**: hardware thermal control (HTC) is supported (support may vary by product).
EDX	3	**TTP**: THERMTRIP is supported = 1.
EDX	2	**VID**: Voltage ID control is supported.

Register	Bits	Description
EDX	1	**FID**: Frequency ID control is supported.
EDX	0	**TS**: Temperature sensor = 1.

CPUID Fn8000_0008 Address Size And Physical Core Count Information

This provides information about the number of physical CPU cores and the maximum physical and linear address width supported by the processor.

Register	Bits	Description
EAX	31:16	Reserved.
EAX	15:8	**LinAddrSize**: Maximum linear byte address size in bits. If the processor supports long mode (see CPUID Fn[8000_0001, 0000_0001]_EDX[LM]) then this is 30h; else this is 20h.
EAX	7:0	**PhysAddrSize**: Maximum physical byte address size in bits = 28h.
EBX	31:0	Reserved.
ECX	31:8	Reserved.
ECX	7:0	**NC**: number of physical CPU cores - 1. The number of CPU cores in the processor is NC+1 (e.g., if NC=0, then there is one CPU core).
EDX	3:0	Reserved.

CPUID Fn8000_0009 Reserved

CPUID Fn8000_000A SVM Revision and Feature Identification

This provides SVM revision and feature information. If CPUID Fn8000_0001_ECX[SVM]=0 then CPUID Fn8000_000A is reserved.

Register	Bits	Description
EAX	31:8	Reserved.
EAX	7:0	**SvmRev**: SVM revision = 01h.
EBX	31:0	**NASID**: number of address space identifiers (ASID) = 40h.
ECX	31:0	Reserved.
EDX	31:2	Reserved.
EDX	1	**LbrVirtulization**: LBR Virtulazation = 1. See "Register Differences in Revisions of AMD NPT Family 0Fh Processors" on page 27 for revision information about this field.
EDX	0	Reserved.

13 BIOS Checklist

The checklist in this chapter reviews information that is described elsewhere and is required by BIOS developers to properly incorporate AMD NPT Family 0Fh Processors into systems.

13.1 CPUID

Use the CPUID instruction to properly identify the processor.

- Determine the processor type, stepping, and available features by using functions 0000_0001h and 8000_0001h of the CPUID instruction.
- Boot-up display - The processor name should be displayed according to the processor name string defined in *AMD Revision Guide for NPT Processors*, order# 33610

For more information on the CPUID instruction, see *Chapter 12, "CPUID Function Registers"*, the *AMD Processor Recognition Application Note,* order# 20734 and the *CPUID Guide for the AMD Athlon™ 64 and AMD Opteron™ Processors,* order# 25481.

13.2 CPU Speed Detection

The BIOS can use the time-stamp counter (TSC) to clock a timed operation and compare the result to the Real-Time Clock (RTC) to determine the operating frequency. See the example of frequency-determination assembler code available on the AMD web site.

13.3 HyperTransport™ Link Detection

After any reset, the BIOS should check every HyperTransport™ link on the processor to detect whether the link is connected by executing the following steps:

1. Wait until the LinkConPend bit in Function 0, Offset 98h, B8h, or D8h is cleared.
2. Check LinkCon bit in Function 0, Offset 98h, B8h, or D8h. If LinkCon bit has a value of 0, the link is not connected. If LinkCon bit has a value of 1, the link is connected.

After a warm reset, if the link is connected, the BIOS should additionally check LinkFail bit in Function 0, Offset 84h, A4h, or C4h to determine if sync flood packets were detected by the link receivers before the warm reset assertion. The LinkFail bit should then be cleared.

13.4 HyperTransport™ Link Frequency Selection

The BIOS should determine the set of frequencies that a processor is capable of for every HyperTransport™ link connected to the processor. Each frequency in the set must be defined by the corresponding link frequency capability bit in the LnkFreqCap field, Function 0, Offsets 88h, A8h, C8h, and it must be equal to or less than the maximum frequency values specified in the processor data sheets.

If a HyperTransport™ link is non-coherent, then the BIOS should initialize the HyperTransport™ link frequency (Freq field, Function 0, Offsets 88h, A8h, C8h) with the highest value from the set of frequencies of which the processor is capable and which is less than or equal to the maximum frequency values specified in the chipset data sheets, and the maximum frequency supported by the platform.

If a HyperTransport™ link is coherent, then the BIOS should initialize the HyperTransport™ link frequencies (Freq field, Function 0, Offsets 88h, A8h, C8h) of both processors with the highest common frequency value from the sets of frequencies of which each processor is capable that is less than or equal to the maximum frequency supported by the platform.

13.5 Multiprocessing Capability Detection

The multiprocessing capability of the AMD Opteron™ processor is determined by the MPCap and BigMPCap bits in the Northbridge Capability Register (Function 3, Offset E8h).

Multiprocessing Capability	MPCap	BigMPCap
UP Capable	0	0
DP Capable	1	0
MP Capable	1	1

During POST, the BIOS checks the multiprocessing capability of AMD Opteron™ processors, and configures the system accordingly.

MP setup by the BIOS is also required in a UP where dual-cores are present.

Multiprocessing capability detection is not required in a UP system.

All processors must be DP capable or MP capable in a DP system. If any processor is only UP capable, the BIOS must configure the BSP as a UP processor, and must not initialize the AP.

All processors must be MP capable in an MP system. If any processor is not MP capable, the BIOS must configure the BSP as a UP processor, and must not initialize APs.

If all processors do not have adequate multiprocessing capability for a DP or an MP system, the BIOS must display the following message:

```
********** Warning: non-MP Processor **********
```

```
The processor(s) installed in your system are not multiprocessing capable. Now
your system will halt.
```

If all processors have adequate multiprocessing capability for a DP or an MP system, but have different model numbers or operate at different frequencies, the BIOS can do one of the following:

- configure the BSP as a UP processor, not initialize APs, and display a message indicating that processors with different model numbers or maximum frequencies have been installed, or
- implement the algorithm defined in "Processors With Different Revisions or Maximum Frequencies in Multiprocessor Systems".

13.6 Setup for Dual Core Processors

To ensure that dual core processors work correctly in existing operating systems, the BIOS must follow several steps. These steps must only be performed on a dual core processor when both cores are enabled.

1. Re-direct all MC4 accesses and error logging to core 0. This is done by setting the NbMcaToMstCpuEn bit (Function 3, Offset 44h).

   Setting this bit is required to ensure that current operating systems do not become confused with conflicting programming of this shared resource between two cores.
2. Ensure that the core number is the least significant bit of the initial APIC ID. This is done by setting the InitApicIdCpuIdLo bit (MSR C001_001Fh, bit 54) in each core.

   Setting this bit is required because current operating systems expect the core ID bits to be in the least significant bit positions of the APIC ID. The operating system makes scheduling decisions based on this information.
3. The firmware must properly report dual-core processors in a number of tables required by the Advanced Configuration and Power Interface (ACPI) specification:
   - The Static Resource Affinity Table (SRAT) must contain a "Processor Local APIC/SAPIC Affinity Structure" for each core rather than for each processor. Cores within the same processor must have the same proximity domain.
   - The CPU objects in the Differentiated System Description Table (DSDT) must contain the correct number of cores.
   - The Multiple APIC Description Table (MADT) must contain a "Processor Local APIC" record and an "ACPI Processor object" for each core rather than for each processor.
4. Each core, rather than each processor, should have a processor entry in the MP configuration table as defined in the MultiProcessor Specification ver, 1.4.
5. Each core, rather than each processor, should have a unique SMM Base Address specified in MSR (C001_0111h).

6. Software can always calculate the number of physical processors in a system by taking the total number of processor entries/records in the MADT or MP configuration tables and dividing by the number of cores per processor returned by CPUID Fn8000_0008.
7. The order of the processor entries/records in the MADT and the MP configuration table are not specified by AMD. However, some firmware vendors may choose to work around the licensing restrictions of some legacy operating systems by listing the entries/records of the first core of each processor (core 0) before listing the entries/records of the remaining cores:

   processor 0, core 0
   ...
   processor N, core 0
   processor 0, core 1
   ...
   processor N, core 1

13.7 APIC ID Assignment Requirements

The arbitration protocol used when sending data over the original serial APIC bus required that IOAPIC IDs be unique and not overlap the processor APIC IDs (i.e. mapped into the same linear ID space such as APIC IDs from 0 to N and IOAPIC IDs from N+1 to N+M). A number of operating systems may still have this requirement even though AMD NPT processors do not require it. Most of these operating systems also support numbering the IOAPIC IDs first before numbering the processor APIC IDs. Firmware should support this model for the broadest operating system support.

If the total number of IOAPICs plus the total number of processor APICs (number of processor * number of cores per processor) is less than 16:

- APIC_ID[node=i] = i for a single-core system
- APIC_ID[node=i, core=j] = i*2 + j for a dual-core system
- APIC_ID for IOAPIC starts from the next available number after the last APIC_ID assigned to the processor.

If the total number of processor APICs is equal to or greater than 16:

- APIC_ID for IOAPIC starts from 0
- APIC_ID[node=i] = i + IOAPIC_OFFSET for a single-core system
- APIC_ID[node=i, core=j] = (i*2 + j) + IOAPIC_OFFSET for a dual-core system.
- IOAPIC_OFFSET = total number of IOAPIC, or total number of IOAPIC + 1 if it is an odd number.

13.8 Processors With Different Revisions or Maximum Frequencies in Multiprocessor Systems

AMD Opteron™ processors with different revisions or maximum frequencies can be mixed in a multiprocessor system. Refer to the *Revision Guide for AMD NPT Family 0Fh Processors,* order# 33610 for information about processors that can be mixed in a multiprocessor system.

The following requirements must be met when processors of different revisions or maximum frequencies are mixed:

1. The system must be configured as a DP or an MP system. See section "Multiprocessing Capability Detection" on page 366 for more information.
2. All processors must properly initialize the Freq field in the corresponding Function 0, Offset 88h, A8h, or C8h register. See "HyperTransport™ Link Frequency Selection" on page 366 for more information.
3. The BIOS must set the FID of all processors in the system to the same value. That value should be the highest common FID for all processors in the system. The set of FID values of which each processor is capable is determined as follows:

```
Execute CPUID instruction with EAX=80000007h;
if(EDX[1] == 1) // Processor supports FID changes.
    Read MSR C001_0042h; // Get FIDVID Status register.
    MaxFID = EAX[21:16];
    // MaxFID is the maximum FID of which the processor is capable.
    // The processor is capable of all FIDs from MaxFID down to the FID that
    // represents 1600 MHz, and if any FID is a portal core frequency, then
    // the corresponding minimum core frequency is also a FID of which the
    // processor is capable. See Table 69.
else // Processor does not supports FID changes.
    Read MSR C001_0015h; // Get HWCR register.
    START_FID = EAX[29:24];
    // This is the only FID of which the processor is capable.
```

   The BIOS should determine whether a highest common FID value for all processors in the system exists. If such a value exists, then FID values for all processors must be set to that value. See Chapter 10, "Power and Thermal Management" for more information on FID changes. If there is not a common FID value for all processors in the system, then the BIOS should not change the FID on any processor, and it should display a warning message.

   Power management registers defined in Table 78 must be programmed before changing the processor FID value.
4. The BIOS must follow P-state change rules specified in Chapter 10, "Power and Thermal Management", however, VID changes do not have to be made.
5. All processors must have the same L2 cache size, returned in ECX[31:16] when a CPUID instruction is executed with EAX=80000006h. If all processors in the system do not have the

same L2 cache size, then the BIOS should display a warning message. See *CPUID Guide for the AMD Athlon™ 64 and AMD Opteron™ Processors,* order# 25481 for more information.

6. The BIOS must ensure that all CPUs present the same feature set to the operating system. For instance, if a CPU in the system does not support SSE3 instructions, then all CPUs in the system must be configured to not report the SSE3 capability to OS or application software.

   To accomplish this, BIOS must first determine the feature set that is common to all CPUs in the system. Then, for all CPUs, BIOS must disable any features that are not supported by all CPUs. The following registers contain the feature set bits:

   - CPUIDFeatures (MSR C001_1004h)
   - CPUIDExtFeatures (MSR C001_1005h)

   The bits in these registers identify individual features. A value of 1 indicates that the feature is supported and a value of 0 indicates that the feature is not supported. All of these bits are read/write, thus, if a bit has a value of 1, it can be written to a 0 to make it appear to the OS that the feature is not supported.
7. The errata, workarounds, and microcode updates must be properly applied to each processor in the system. See *Revision Guide for AMD NPT Family 0F Processors*, order# 33610 for more information.
8. If S3 power state is implemented in the system, all steps listed above must be executed on a resume from S3.

13.9 Model-Specific Registers (MSRs)

Access only the MSRs that are implemented in the processor.

- Follow the processor MSR programming sequence described in this document. This includes setting the HWCR, SYSCFG, MTRRs, IORRs, clock control MSR, Top of Memory, and other registers.

13.9.1 Software Considerations for Accessing Northbridge MSRs in Dual Core Processors

MSRs that control northbridge functions are shared between both cores in a dual core processor. If control of northbridge functions is shared between software on both core, software must ensure that only one core at a time is allowed to access the shared MSR.

Table 80. Northbridge MSRs

Address	Register Name	Description
0410h	MC4_CTL	page 241
0411h	MC4_STATUS	page 241

Table 80. Northbridge MSRs (Continued)

Address	Register Name	Description
0412h	MC4_ADDR	page 241
0413h	MC4_MISC	page 241
C001_001Fh	NB_CFG	page 413
C001_003Eh	HTC	page 415
C001_0041h	FIDVID_CTL	page 423
C001_0042h	FIDVID_STATUS	page 425
C001_0048h	MCi_CTL_MASK	page 241

13.10 Machine Check Architecture (MCA)

The processor supports a machine check architecture that allows reporting and handling of system errors through a consistent interface.

- The global MCA Status register must be cleared (all machine check features disabled).
- MC0_CTL must be set to all 1s because some do not set these bits.
- In addition, the local MCA status and address registers for the following five reporting blocks must also be cleared (all error and parity bits disabled):
  - BU—Bus Unit
  - IC—Instruction Cache Unit
  - DC—Data Cache Unit
  - LS—Load Store Unit
  - NB—Northbridge Unit

After reset, the BIOS needs to determine if the cause of the reset was a power-on or soft reset.

- When power-on reset occurs, the BIOS must clear the MCA registers.
- Otherwise, the BIOS should check for any MCA information that may be related to the soft reset.

For more information on MCA, see Chapter 6, "Machine Check Architecture."

13.10.1 GART Table Walk Error Reporting

The GART table walk error reporting function is enabled by setting bit 10 of the MC4_CTL register (MSR 410h). This results in a machine check exception when an AGP aperture access has no matching translation in the GART. This error is typically caused by a software graphics driver that improperly reserves or allocates aperture pages in the GART, resulting in benign visual artifacts

which are often undetected on other platforms. Setting MC4_CTL[10] allows software developers to debug this error; the resulting benign machine check errors can, however, confuse an end user. For this reason, AMD recommends that the BIOS developers disable this function by setting bit 10 of MC4_CTL_MASK register (MSR C001_0048h) to a value of 1. This bit must be set before MC4_CTL[10] bit is set. AMD also recommends adding a setup option to the BIOS setup menu. The following should be displayed in the setup option:

```
Gart Table Walk Error MC reporting:     Disabled/Enabled.
```

The default setting is disabled. The device driver developer may enable this function for implementation and testing purposes. Also, a help message should be added with this setup option. An example of the help message is:

```
This option should remain disabled for normal operation.
```

13.11 GART Register Restoration After S3 Resume

After a resume from S3, GartEn (Function 3, Offset 90h) must be restored after all GART related and AGP device registers are restored.

13.12 Register Settings in UMA Systems

AMD recommends that UMA graphics systems use the following register field values:

Register and Field Position	Field Name	Value
Function 0, Offset 68h, [28]	EnCpuRdHiPri, enable high priority CPU reads	0b
Function 0, Offset 68h, [27:26]	HiPriBypCnt, high-priority bypass count	11b
Function 0, Offset 68h, [11]	RspPassPW, response pass posted writes	1b
Function 2, Offset A0h, [5]	DCC_EN, dynamic idle cycle counter enable	1b
Function 2, Offset 94h, [27:24]	DcqBypMax, bypass maximum	111b
Function 2, Offset 90h, [10]	BurstLength32, enable 32-byte granularity	1b
Function 2, OffsetA0h, [3:2]	RdWrQByp, read/write queue bypass count	11b
MSR C001_001Fh, [36]	DisDatMsk, disable data mask	0b
MSR C001_001Fh, [9]	DisUseRefFreeBuf, enable display refresh to use free list buffers	0b

Refer to Table 36 for information on recommended flow control buffer allocation in UMA graphics systems.

13.13 Memory Map

13.13.1 I/O and Memory Type and Range Registers (IORRs, MTRRs)

The memory-type and range registers control the access and cacheability of memory regions in the processor.

- Follow the processor MTRR and IORR programming sequence that is described in this document.
- To initialize MTTRs in BIOS, set the default memory cache type for all addresses to uncacheable, then use a variable range MTRR to set all physical memory as cacheable (writeback), and finally use the fixed-range MTRRs to handle the special cases in the 640-Kbyte–1-Mbyte region. MTRRs should be modified while cache is disabled. Modifying MTRRs while cache is enabled results in undefined behavior.
- The GART driver (miniport) should program the AGP aperture.
- The BIOS does not map the cacheability of the video frame buffer and AGP aperture space; it is done by the operating system, video, and AGP drivers.

13.13.2 Memory Map Registers (MMRs)

The processor contains several memory map registers to support the appropriate software view of memory.

- Program the correct values in the MSRs, such as Top of Memory (TOP_MEM).

13.14 Access Routing And Type Determination

13.14.1 Memory Access to the Physical Address Space

All memory accesses to the physical address space from a CPU core are sent to its associated northbridge (NB). All memory accesses from an I/O link are routed through the NB. An I/O link access to physical address space indicates to the NB the cache attribute (Coherent or NonCoherent).

A CPU core access to physical address space has two attributes that the CPU must determine before issuing the access to the NB: the cache attribute (e.g., WB, WC, UC) and the access destination (DRAM or MMIO).

13.14.1.1 Determining The Cache Attribute

1. The CPU translates the logical address to a physical address. In that process it determines the initial cache attribute based on the settings of the Page Table Entry PAT bits, the MTRR Default Memory Type Register (MSR 02FF), the Variable-Size MTRRs (MSRs 02[0F:00]), the Fixed-

Size MTRRs (MSRs 02[6F:68, 59, 58, 50]), and the SYSCFG bit Tom2ForceMemTypeWB (MSR C001_0010h bit 22).

2. The ASeg and TSeg SMM mechanisms are then checked in parallel to determine if the initial cache attribute should be overridden (see MSR C001_0112 and MSR C001_0113). If the address falls within an enabled ASeg/TSeg region, then the final cache attribute is determined as specified in MSR C001_0113.

This mechanism is managed by the BIOS and does not require any setup or changes by system software.

13.14.1.2 Determining The Access Destination for CPU Accesses

The access destination is based on the highest priority of the following ranges that the access falls in:

1. (Lowest priority) Compare against the top-of memory registers (MSR C001_001A and MSR C001_001D).
2. The Fixed-Size MTRRs (MSRs 02[6F:68, 59, 58, 50]).
3. The IORRs (MSRs C001_00[18, 16] and MSRs C001_00[19, 17]).
4. TSEG & ASEG (MSR C001_0112 and MSR C001_0113).
5. (Highest priority) NB AGP aperture range registers.

To determine the access destination, the following steps are taken:

1. The CPU compares the address against the Top Of Memory Register (MSR C001_001A), and the Top Of Memory 2 Register (MSR C001_001D), to determine if the default access destination is DRAM or MMIO space.
2. For addresses below 1M byte, the address is then compared against the appropriate Fixed MTRRs (MSRs 02[6F:68, 59, 58, 50]) to override the default access destination. Each fixed MTRR includes two bits, RdDram and WrDram, that determine the destination based on the access type.
3. The CPU then compares the address against the IORRs (MSRs C001_00[18, 16] and MSRs C001_00[19, 17]); if it matches, the default access destination is overridden as specified by the IORRs. BIOS can use the IORRs to create an I/O hole within a range of addresses that would normally be mapped to DRAM. It can also use the IORRs to re-assert a DRAM destination for a range of addresses that fall within a bigger I/O hole that overlays DRAM. Some key points to consider:
    a. Operating system software never needs to program IORRs to re-map addresses that naturally target DRAM; any such programming is done by the BIOS.
    b. The IORRs should not cover the range used for the AGP aperture if the GART logic in the NB is enabled.
    c. The IORRs should be programmed to cover the AGP aperture if the aperture/GART translation is handled by an I/O device (e.g., the chipset).

4. The ASeg and TSeg SMM mechanisms are then checked in parallel to determine if the destination should be overridden (see MSR C001_0112 and MSR C001_0113). If the address falls within an enabled ASeg/TSeg region, then the destination is determined as specified in MSR C001_0113.

This mechanism is managed by the BIOS and does not require any setup or changes by system software.

13.14.2 Northbridge Processing of Accesses To Physical Address Space

The physical address, its cacheability type, its access type and its access destination is presented to the processor NB for further processing. Processing of an access is performed by NB in the following manner:

1. Regardless of the access destination, if supplied, the physical address is checked against the NB's AGP-aperture range-registers, if enabled; if the address matches, the NB translates the physical address through the AGP GART. A match in the AGP aperture overrides any match to the NB's DRAM Base/Limit Registers, Function 1, Offsets [7C:40], and Memory Mapped I/O Base/Limit Registers, Function 1, Offsets [BC:80].
2. For accesses from I/O devices, the cacheability attribute from the GART entry (GartPteCoh bit) is applied; for accesses from a CPU, the attribute already applied by the CPU is used and the GartPteCoh bit is ignored. (System software should ensure that the cacheability attribute assigned to an AGP aperture matches the GartPteCoh bit in the matching GART entry.)
3. Accesses that do not match the AGP aperture and post-GART translated addresses are compared against the NB DRAM Base/Limit Registers, Function 1, Offsets [7C:40], and Memory Mapped I/O Base/Limit Registers, Function 1, Offsets [BC:80], to determine the destination link of the access.
4. For accesses from a CPU, the access destination determines whether to compare the address against the DRAM Base/Limit Registers, Function 1, Offsets [7C:40] or the Memory Mapped I/O Base/Limit Registers, Function 1, Offsets [BC:80]. For accesses from an I/O device both sets of registers are checked; if an access matches against both, the MMIO registers take precedence.

13.15 Error Handling

Error detection and signaling capabilities of all system components: processors, chipsets, BIOS and operating system must be considered when selecting error handling mechanisms.

13.15.1 Error Handling Mechanisms

Sync flooding and machine check exception handling mechanisms are described.

13.15.1.1 Sync Flooding

Sync flooding is a method used to stop data propagation in the case of a serious error. The device that detects the error initiates sync flood, and all devices in the system that detect sync flood packets on its

HyperTransport™ receivers cease normal operation and start transmitting sync flood packets. Sync flood packets reach the I/O hub. If the I/O hub is programmed to generate a warm reset in the case of sync flood detection, the BIOS can analyze all error bits in the system that were not cleared by the warm reset and detect the error source. If the I/O hub is not programmed to generate a warm reset, the system hangs.

13.15.1.2 Machine Check Exceptions

Machine check exceptions are generated when uncorrectable error conditions and certain correctable error conditions are detected by AMD NPT processors. Detected error conditions are caused by the processor or an external I/O device. Machine check exception architecture can be programmed to generate sync flood instead of a machine check exception for some types of uncorrectable errors. See Chapter 6, "Machine Check Architecture" for more information.

13.15.2 Errors and Recommended Error Handling

13.15.2.1 CRC Errors on HyperTransport™ Links

AMD NPT Family 0Fh Processors can be programmed to generate sync flood when a CRC error is detected on a HyperTransport™ link. Sync flooding is enabled with Function 0, Offset 84h, A4h, or C4h, CRCFloodEn bit.

13.15.2.2 I/O Errors

Target aborts on reads and nonposted writes are indicated by ERROR=1 and NXA=0 in their response packets in the HyperTransport™ protocol. If detected by the processor, the recommended error handling mechanism is machine check exception. The BIOS should set Function 3, Offset 40h, TgtAbrtEn bit.

Master aborts on reads and nonposted writes are indicated by ERROR=1 and NXA=1 in their response packets in the HyperTransport™ protocol. If detected by the processor, the recommended error handling mechanism is machine check exception. The BIOS should set Function 3, Offset 40h, MstrAbrtEn bit.

It is also recommended to set DisPciCfgCpuErrRsp and CpuErrDis bits in Function 3, Offset 44h.

Error status is not returned to the processor for target and master aborts on posted writes from the processor to I/O devices, and the I/O device should respond. Error handling is chipset dependent in the case of target and master aborts on posted writes, PERR and SERR. Chipsets can implement sync flooding, interrupts, and other methods for error handling.

13.15.2.3 Machine Check Architecture Errors

It is recommended that all uncorrectable errors detected by the MCA banks in AMD NPT Family 0Fh Processors with the exception of uncorrectable ECC errors detected by the NB are handled by the machine check exceptions.

13.15.2.4 Uncorrectable ECC Errors Detected by the NB

In AMD Opteron™ multiprocessor system, sync flooding is a recommended response to uncorrectable ECC errors detected by the NB on reads from the DRAM. BIOS must take the following steps:

1. Initialize DRAM on all processors in the system (see Chapter 5, "DRAM Configuration").
2. Enable ECC checking in all DRAM controllers (Function 3, Offset 44h, bit EccEn), and initialize every DRAM location. ECC errors generated during DRAM initialization should be ignored.
3. Enable sync flooding on uncorrectable ECC errors (Function 3, Offset 44h, bit SyncOnUcEccEn).
4. Enable uncorrectable ECC reporting (Function 3, Offset 40h, bit UnCorrEccEn).
5. Enable the I/O hub to generate a warm reset when it detects sync flood packets on the HyperTransport™ link.
6. Determine the type of reset shortly after boot (Function 0, Offset 6Ch, ColdRstDet bit). If the reset was warm, check the LinkFail bit (Function 0, Offset 84h, A4h, or C4h) for every connected link (see "HyperTransport™ Link Detection" on page 365). If the LinkFail bit is set, sync flood packets were detected by the link before the warm reset. The LinkFail bit should be cleared by writing a 1, and the BIOS should determine the error source. Uncorrectable ECC errors detected on DRAM reads are reported in the MCA NB reporting bank. MCA NB status and address registers (Function 3, Offsets 48h, 4Ch, 50h, and 54h) should be analyzed to determine the DIMM where the error originated.

13.15.2.5 Watchdog Timeout Errors

In AMD Opteron™ and AMD Athlon™ 64 Processor systems sync flooding is the recommend response to watchdog timeout errors. BIOS must take the following steps:

1. Enable sync flooding on watchdog timeout errors (Function 3, Offset 44h, bit SyncOnWdogEn).
2. Enable watchdog timeout reporting (Function 3, Offset 40h, bit WchDogTmrEn).
3. Enable the I/O hub to generate a warm reset when it detects sync flood packets on the HyperTransport™ link.
4. Determine the type of reset shortly after boot (Function 0, Offset 6Ch, ColdRstDet bit). If the reset was warm, check the LinkFail bit (Function 0, Offset 84h, A4h, or C4h) for every connected link (see "HyperTransport™ Link Detection" on page 365). If the LinkFail bit is set, sync flood packets were detected by the link before the warm reset. The LinkFail bit should be cleared by writing a 1, and the BIOS should determine the error source. Watchdog timeout errors are reported in the MCA NB reporting bank. BIOS should not clear the MCi_Status registers when a watchdog timeout is detected.

13.15.2.6 Correctable ECC Errors Detected by the NB

System software may optionally detect and log large numbers of correctable ECC errors detected by the NB on reads from the DRAM, with minimal overhead. A programmable error threshold counter causes an interrupt when the counter overflows, indicating a measure of general DRAM health to the

system software. See "MC4_MISC—DRAM Errors Threshold Register" on page 241. Note, the counter also counts uncorrectable ECC errors detected by the NB on reads from the DRAM. The count threshold interrupt may be superceded by an MCA exception if MCA exceptions are enabled for the same errors. The system software must take the following steps to initialize thresholding:

1. Check the Locked bit (bit 61) in MC4_MISC to determine if the error thresholding mechanism is already in use.
2. Initialize the error threshold count and interrupt type in MC4_MISC.
3. If APIC interrupts are used to notify system software then program APIC register offset 500h accordingly. See "Threshold Count Interrupt 0 Local Vector Table Entry" on page 269.

13.16 Cache Initialization For General Storage During Boot

Prior to initializing the DRAM controller for system memory, the processor cache subsystem may be initialized for use by BIOS as general storage. The following steps should be taken to set up the cache in the boot core of the BSP to supply 64K bytes of memory, including a stack:

- Initialize the DRAM Base/Limit Registers (Function 1, Offsets [7C:40]) to a temporary value that allocates DRAM to each of the processors. BASE=The temporary base address of the DRAM range of the BSP.
- Clear all the MTRRs.
- Set MSRC001_0010 [MtrrFixDramEn, MtrrFixDramModEn, and MtrrVarDramEn].
- Enable cache through CR0.
- Temporarily disable cache fill probes on the BSP by programming DisFillP = 1 (Function 0, offset 68h).
- Use MTRRs to specify the write-back attribute (WB) to a 64 kilobyte portion of the memory map, starting at BASE.
- Read exactly 64 kilobytes of memory starting at BASE, using the REP MOV instruction.
- Restore DisFillP (Function 0, offset 68h).
- Set SS=BASE/16.
- Set SP=BASE+64K-4.

If DRAM training is used in conjunction with the cache for general storage then the storage size must be reduced and the address space of the storage chosen to meet the special requirements of the DRAM training software. The general requirement for DDR training is that 256 cache lines, corresponding to L1 cache tag indexes of 00h - FFh, are used and must be reserved for this purpose. See AMD64 Architecture Programmer's Manual, Volume 2: System Programming, for more details

on L1 cache organization and function. One possible implementation example that meets this requirement is as follows:

- A general storage size of 48KB is used.
- The address space for the storage is from C4000h to CFFFFh.

Note: *When using cache as general storage the BIOS must ensure that no data lines are evicted from the cache prior to the DRAM controller being initialized. The CLFLUSH and WBINVD must not be used before DRAM initialization when using cache as general storage.*

13.17 Cache Testing and Programming

- BIOS must correctly maintain the tag parity and data ECC enable bits.

13.18 Memory System Configuration Registers

Node configuration space contains registers that define RAM, PCI memory, and x86 I/O address paths.

- Address mapping registers are contained in function 1 of each node.
- Cold boot and reset sends code fetch and other addresses to the compatibility link of Node0. At cold boot and reset, these mapping registers are disabled.
- Once enabled, these registers distribute addresses according to register mappings, which include node and link routing. See Chapter 9, "HyperTransport™ Technology Configuration and Enumeration."
- Thereafter, carefully set registers to reflect current needs for code fetch and memory access, i.e., do not map BIOS E0000h and F0000h space to DRAM until BIOS exits execution from the ROM chip. Otherwise, the system will hang. Likewise, memory-mapped I/O space should not be mapped above 1 Mbyte until BIOS exits execution from the ROM chip, or else the system will hang.
- Memory-mapped I/O (MMIO) and PCI I/O mappings must include address space needed to move option ROM code to RAM, as well as the required run-time resources. This could require setting and resetting the MMIO mappings during enumeration and device initialization. Configuration space access of a device with or without an option ROM is defined by the HyperTransport technology configuration mapping.
- Multihost bus multiprocessor systems require that MMIO mappings direct relevant addresses to the buses of devices through the node/link path to the bus.
- Application processors and the bootstrap processor in multiprocessor systems must be mapped separately with paths to target address space.

13.19 Processor ID

ACPI 2.0 specifies that each processor is required to have a unique ProcessorID, that although arbitrary, must match its corresponding ACPI Processor ID field in the Processor Local APIC table.

13.20 XSDT Table

The Extended System Description Table (XSDT) table defined in ACPI 2.0 must be implemented in the BIOS to support 64-bit operating systems for AMD NPT Family 0Fh Processor based systems.

13.21 Detect Target Operating Mode Callback

The operating system notifies the BIOS what the expected operating mode is with the Detect Target Operating Mode callback (INT 15, function EC00h). Based on the target operating mode, the BIOS can enable or disable mode specific performance and functional optimizations that are not visible to system software.

This callback does not change the operating mode; it only declares the target mode to the BIOS. It should be executed only once by the BSP before the first transition into long mode.

The default operating mode assumed by the BIOS is Legacy Mode Target Only. If this is not the target operating mode, system software must execute this callback to change it before transitioning to long mode for the first time. If the target operating mode is Legacy Mode Target Only, the callback does not need to be executed.

The Detect Target Operating Mode callback inputs are stored in the AX and BL registers. AX has a value of EC00h, selecting the Detect Target Operating Mode function. One of the following values in the BL register selects the operating mode:

- 01h — Legacy Mode Target Only. All enabled processors will operate in legacy mode only.
- 02h — Long Mode Target Only. All enabled processor will switch into long mode once.
- 03h — Mixed Mode Target. Processors may switch between legacy mode and long mode, or the preferred mode for system software is unknown. This value instructs the BIOS to use settings that are valid in all modes.
- All other values are reserved.

The Detect Target Operating Mode callback outputs are stored in the AH register and CF (carry flag in the EFLAGS register), and the values of other registers are not modified. The following output values are possible:

- AH = 00h and CF = 0, if the callback is implemented and the value in BL is supported.
- AH = 00h and CF = 1, if the callback is implemented and the value in BL is reserved. This indicates an error; the target operating mode is set to Legacy Mode Target Only.

- AH = 86h and CF = 1, if the callback is not supported.

13.22 SMM Issues

The processor includes support for system management mode (SMM). The functionality of the AMD NPT Family 0Fh Processor SMM is a superset of the Pentium® processor functionality.

- AMD NPT Family 0Fh Processors implement the SMM remapping and control registers as model-specific registers on the CPU.
- Implement the SMM state-save area in a manner compatible with the description of SMM found in Chapter 8, "System Management Mode (SMM)."

Program the model-specific registers to set the SMM memory base, local address, destination address, memory type, size and control, etc. See Chapter 8, "System Management Mode (SMM)."

14 Processor Configuration Registers

This chapter includes descriptions of two types of model-specific registers, as follows:

- General model-specific registers (see page 383)
- AMD NPT Family 0Fh Processor model-specific registers (see page 406)

14.1 General Model-Specific Registers

Table 81 is a listing of the general model-specific registers supported by the processor, presented in ascending hexadecimal address order. Register descriptions follow the table, organized according to the following functions:

- System software (see page 385)
- Memory typing (see page 387)
- APIC (see page 404)
- Machine check architecture (see page 218)
- Software debug (see page 405)
- Performance monitoring (see page 406)

Table 81. General MSRs

Address	Register Name	Description
0010h	TSC	page 406
001Bh	APIC_BASE	page 404
002Ah	EBL_CR_POWERON	page 404
00FEh	MTRRcap	page 387
0174h	SYSENTER_CS	page 385
0175h	SYSENTER_ESP	page 386
0176h	SYSENTER_EIP	page 386
0179h	MCG_CAP	page 219
017Ah	MCG_STATUS	page 219
017Bh	MCG_CTL	page 220

Table 81. General MSRs (Continued)

Address	Register Name	Description
01D9h	DebugCtl	page 405
01DBh	LastBranchFromIP	page 405
01DCh	LastBranchToIP	page 405
01DDh	LastExceptionFromIP	page 405
01DEh	LastExceptionToIP	page 405
0200–020Eh Even	MTRRphysBase[7:0]	page 388
0201–020Fh Odd	MTRRphysMask[7:0]	page 388
0250h	MTRRfix64K_00000	page 389
0258h	MTRRfix16K_80000	page 390
0259h	MTRRfix16K_A0000	page 391
0268h	MTRRfix4K_C0000	page 393
0269h	MTRRfix4K_C8000	page 394
026Ah	MTRRfix4K_D0000	page 395
026Bh	MTRRfix4K_D8000	page 396
026Ch	MTRRfix4K_E0000	page 397
026Dh	MTRRfix4K_E8000	page 398
026Eh	MTRRfix4K_F0000	page 400
026Fh	MTRRfix4K_F8000	page 401
0277h	PAT	page 402
02FFh	MTRRdefType	page 403
0400h	MC0_CTL	page 226
0401h	MC0_STATUS	page 227
0402h	MC0_ADDR	page 230
0403h,	MC0_MISC	page 225
0404h	MC1_CTL	page 231
0405h	MC1_STATUS	page 232
0406h	MC1_ADDR	page 232
0407h	MC1_MISC	page 225

Table 81. General MSRs (Continued)

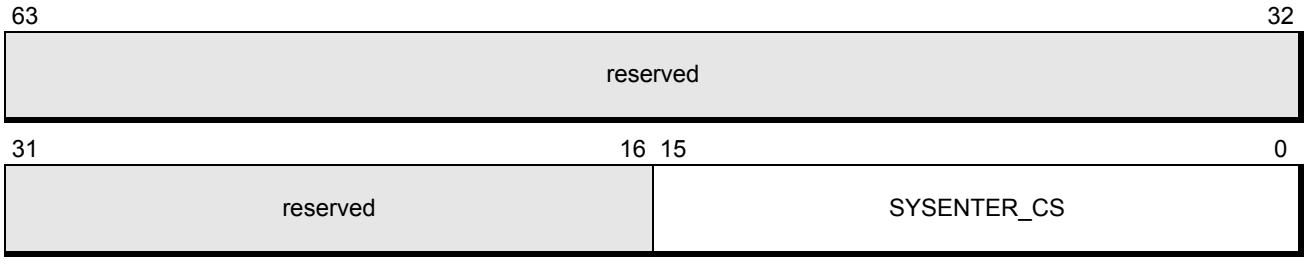
Address	Register Name	Description
0408h	MC2_CTL	page 234
0409h	MC2_STATUS	page 237
040Ah	MC2_ADDR	page 238
040Bh	MC2_MISC	page 225
040Ch	MC3_CTL	page 239
040Dh	MC3_STATUS	page 240
040Eh	MC3_ADDR	page 240
040Fh	MC3_MISC	page 225
0410h	MC4_CTL	page 241
0411h	MC4_STATUS	page 241
0412h	MC4_ADDR	page 241
0413h	MC4_MISC	page 241

14.1.1 System Software Registers

14.1.1.1 SYSENTER_CS Register

This register contains the code segment selector used by the SYSENTER and SYSEXIT instructions. See the SYSENTER and SYSEXIT section in Volume 2 of the *AMD64 Architecture Programmer's Manual* for more information.

SYSENTER_CS Register **MSR 0174h**



Bit	Mnemonic	Function	R/W	Reset
63–32	reserved	RAZ	R	0
31–16	reserved	SBZ	R/W	0
15–0	SYSENTER_CS	SYSENTER/SYSEXIT code segment selector	R/W	0

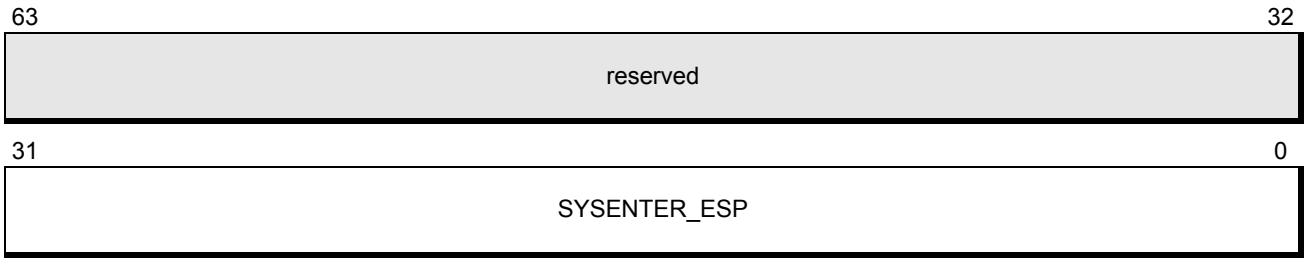
Field Descriptions

SYSENTER/SYSEXIT Code Segment Selector (SYSENTER_CS)—Bits 15–0.

14.1.1.2 SYSENTER_ESP Register

This register contains the stack pointer used by the SYSENTER and SYSEXIT instructions. See the SYSENTER and SYSEXIT section in Volume 2 of the *AMD64 Architecture Programmer's Manual* for more information.

SYSENTER_ESP Register **MSR 0175h**



Bit	Mnemonic	Function	R/W	Reset
63–32	reserved	RAZ	R	0
31–0	SYSENTER_ESP	SYSENTER/SYSEXIT stack pointer	R/W	0

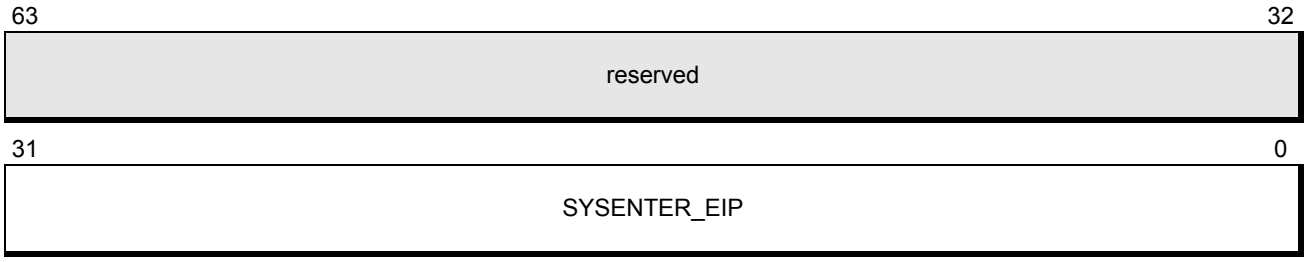
Field Descriptions

SYSENTER/SYSEXIT Stack Pointer (SYSENTER_ESP)—Bits 31–0.

14.1.1.3 SYSENTER_EIP Register

This register contains the instruction pointer used by the SYSENTER and SYSEXIT instructions. See the SYSENTER and SYSEXIT section in Volume 2 of the *AMD64 Architecture Programmer's Manual* for more information.

SYSENTER_EIP Register **MSR 0176h**



Bit	Mnemonic	Function	R/W	Reset
63–32	reserved	RAZ	R	0
31–0	SYSENTER_EIP	SYSENTER/SYSEXIT instruction pointer	R/W	0

Field Descriptions

SYSENTER/SYSEXIT Instruction Pointer (SYSENTER_EIP)—Bits 31–0.

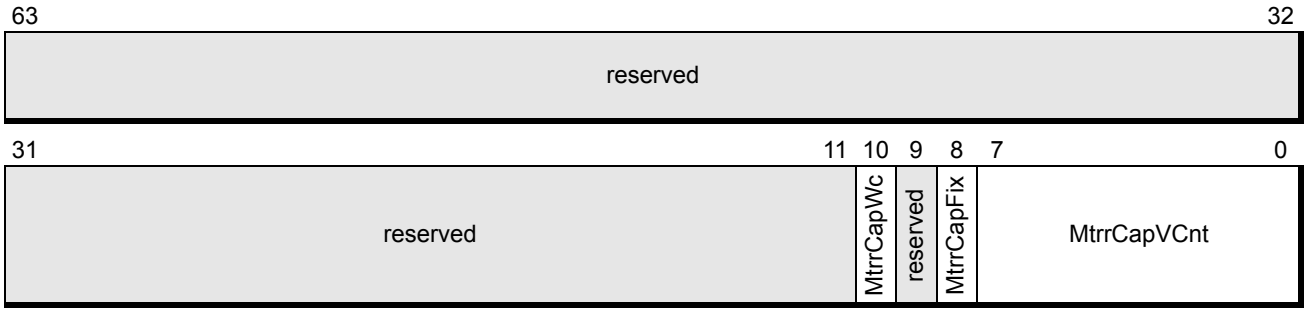
14.1.2 Memory Typing Registers

14.1.2.1 MTRRcap Register

This is a read-only register that returns information about the processors MTRR capabilities. See the "Using MTRRs" section in Volume 2 of the *AMD64 Architecture Programmer's Manual* for more information.

The MTRRcap register is a read-only status register. Attempting to modify this register results in a #GP(0).

MTRRcap Register **MSR 00FEh**



Bit	Mnemonic	Function	R/W
63–11	reserved	RAZ	R
10	MtrrCapWc	Write-combining memory type	R
9	reserved	RAZ	R
8	MtrrCapFix	Fixed range register	R
7–0	MtrrCapVCnt	Variable range registers count	R

Field Descriptions

Variable Range Registers Count (MtrrCapVCnt)—Bits 7–0. Indicates number of variable range registers.

Fixed Range Registers (MtrrCapFix)—Bit 8. Indicates fixed range register capability.

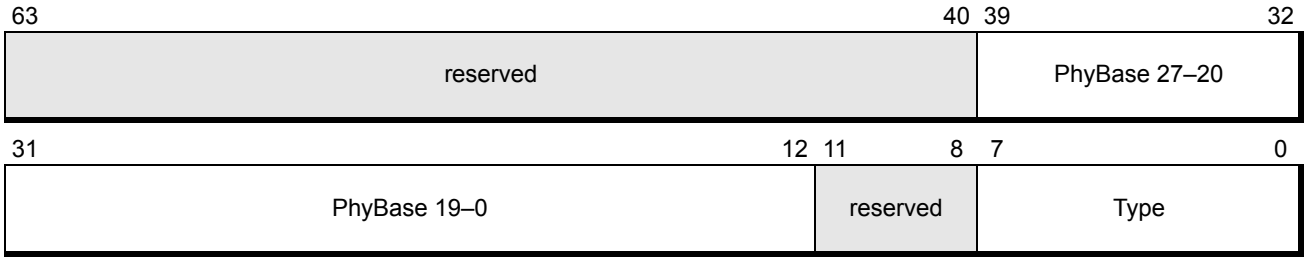
Write-Combining Memory Type (MtrrCapWc)—Bit 10. Indicates write-combining memory type capability.

14.1.2.2 MTRRphysBase*i* Registers

These registers define the base address and memory type for each of the variable MTRRs. See the "Memory Type Range Registers" section in Volume 2 of the *AMD64 Architecture Programmer's Manual* for more information.

Attempting to modify reserved bits in MTRRphysBase*i* results in a #GP(0).

MTRRphysBase0–7 Registers — **MSRs 0200h, 0202h, 0204h, 0206h, 0208h, 020Ah, 020Ch, 020Eh**



Bit	Mnemonic	Function	R/W	Reset
63–40	reserved	MBZ		0
39–12	PhyBase	Base address	R/W	U
11–8	reserved	MBZ		0
7–0	Type	Memory type	R/W	U

Field Descriptions

Memory Type (Type)—Bits 7–0. Specifies memory type for this memory range.

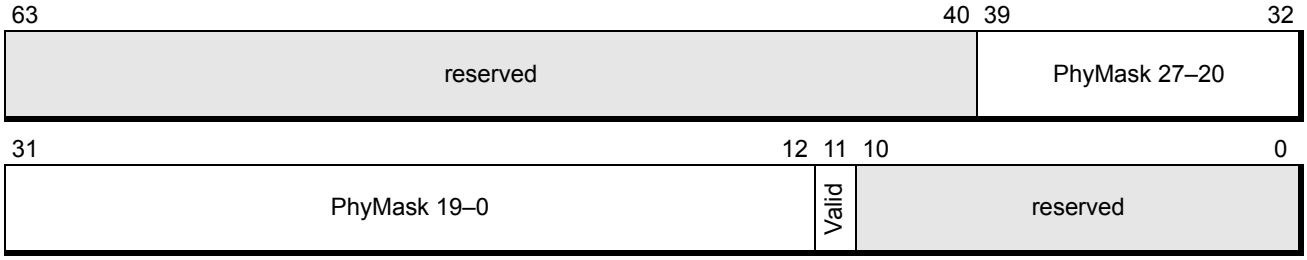
Base Address (PhysBase)—Bits 39–12. Specifies base address for this memory range

14.1.2.3 MTRRphysMask*i* Registers

These registers define the address mask and valid bit for each of the variable MTRRs. See the "Memory Type Range Registers" section in Volume 2 of the *AMD64 Architecture Programmer's Manual* for more information.

Attempting to modify reserved bits in MTRRphysMask*i* results in a #GP(0).

MTRRphysMask0–7 Registers — MSRs 0201h, 0203h, 0205, 0207h, 0209h, 020Bh, 020Dh, 020Fh



Bit	Mnemonic	Function	R/W	Reset
63–40	reserved	MBZ		0
39–12	PhysMask	Address mask	R/W	U
11	Valid	MTRR is valid	R/W	0
10–0	reserved	MBZ		0

Field Descriptions

MTRR is Valid (Valid)—Bit 11. Indicated MTRR is valid.

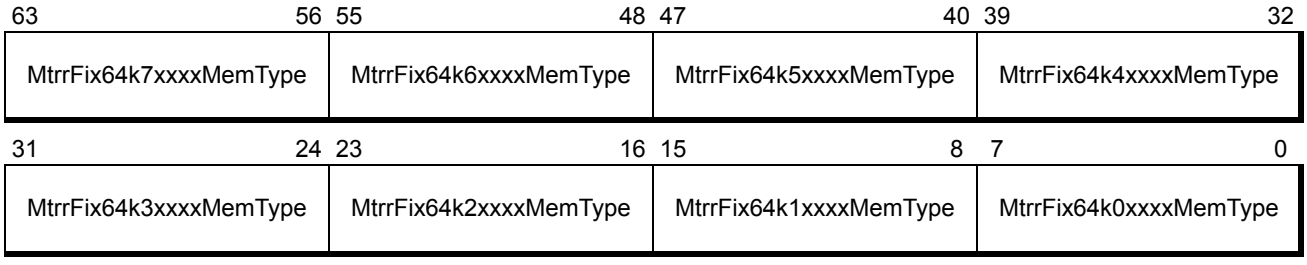
Address Mask (PhysMask)—Bits 39–12. Specifies the address mask for this memory range.

14.1.2.4 MTRRfix64K_00000 Register

This register controls the memory types for the first 512 Kbyte of physical memory. See the "Memory Type Range Registers" section in Volume 2 of the *AMD64 Architecture Programmer's Manual* for more information.

Setting any range to an undefined memory type results in a #GP(0).

MTRRfix64K_00000 Register — MSR 0250h



Bit	Mnemonic	Function	R/W	Reset
63–56	MtrrFix64k7xxxxMemType	Memory type address 70000–7FFFFh	R/W	U
55–48	MtrrFix64k6xxxxMemType	Memory type address 60000–6FFFFh	R/W	U
47–40	MtrrFix64k5xxxxMemType	Memory type address 50000–5FFFFh	R/W	U

* Memory types must be set to values consistent with system hardware.

Bit	Mnemonic	Function	R/W	Reset
39–32	MtrrFix64k4xxxxMemType	Memory type address 40000–4FFFFh	R/W	U
31–24	MtrrFix64k3xxxxMemType	Memory type address 30000–3FFFFh	R/W	U
23–16	MtrrFix64k2xxxxMemType	Memory type address 20000–2FFFFh	R/W	U
15–8	MtrrFix64k1xxxxMemType	Memory type address 10000–1FFFFh	R/W	U
7–0	MtrrFix64k0xxxxMemType	Memory type address 00000–0FFFFh	R/W	U

* Memory types must be set to values consistent with system hardware.

Field Descriptions

Memory Type Address 00000–0FFFFh (MtrrFix64k0xxxxMemType)—Bits 7–0. Memory type for physical address 00000–0FFFFh.

Memory Type Address 10000–1FFFFh (MtrrFix64k1xxxxMemType)—Bits 15–8. Memory type for physical address 10000–1FFFFh.

Memory Type Address 20000–2FFFFh (MtrrFix64k2xxxxMemType)—Bits 23–16. Memory type for physical address 20000–2FFFFh.

Memory Type Address 30000–3FFFFh (MtrrFix64k3xxxxMemType)—Bits 31–24. Memory type for physical address 30000–3FFFFh.

Memory Type Address 40000–4FFFFh (MtrrFix64k4xxxxMemType)—Bits 39–32. Memory type for physical address 40000–4FFFFh.

Memory Type Address 50000–5FFFFh (MtrrFix64k5xxxxMemType)—Bits 47–40. Memory type for physical address 50000–5FFFFh.

Memory Type Address 60000–6FFFFh (MtrrFix64k6xxxxMemType)—Bits 55–48. Memory type for physical address 60000–6FFFFh.

Memory Type Address 70000–7FFFFh (MtrrFix64k7xxxxMemType)—Bits 63–56. Memory type for physical address 70000–7FFFFh.

14.1.2.5 MTRRfix16K_80000 Register

This register controls the memory types for physical memory addresses 80000–9FFFFh. See the "Memory Type Range Registers" section in Volume 2 of the *AMD64 Architecture Programmer's Manual* for more information.

Setting any range to an undefined memory type results in a #GP(0).

MTRRfix16K_80000 Register **MSR 0258h**

63 56	55 48	47 40	39 32
MtrrFix16k9CxxxMemType	MtrrFix16k98xxxMemType	MtrrFix16k94xxxMemType	MtrrFix16k90xxxMemType

31 24	23 16	15 8	7 0
MtrrFix16k8CxxxMemType	MtrrFix16k88xxxMemType	MtrrFix16k84xxxMemType	MtrrFix16k80xxxMemType

Bit	Mnemonic	Function	R/W	Reset
63–56	MtrrFix16k9CxxxMemType	Memory type address 9C000–9FFFFh	R/W	U
55–48	MtrrFix16k98xxxMemType	Memory type address 98000–9BFFFh	R/W	U
47–40	MtrrFix16k94xxxMemType	Memory type address 94000–97FFFh	R/W	U
39–32	MtrrFix16k90xxxMemType	Memory type address 90000–93FFFh	R/W	U
31–24	MtrrFix16k8CxxxMemType	Memory type address 8C000–8FFFFh	R/W	U
23–16	MtrrFix16k88xxxMemType	Memory type address 88000–8BFFFh	R/W	U
15–8	MtrrFix16k84xxxMemType	Memory type address 84000–87FFFh	R/W	U
7–0	MtrrFix16k80xxxMemType	Memory type address 80000–83FFFh	R/W	U

* Memory types must be set to values consistent with system hardware.

Field Descriptions

Memory Type Address 80000–83FFFh (MtrrFix16k80xxxMemType)—Bits 7–0. Memory type for physical address 80000–83FFFh.

Memory Type Address 84000–87FFFh (MtrrFix16k84xxxMemType)—Bits 15–8. Memory type for physical address 84000–87FFFh.

Memory Type Address 88000–8BFFFh (MtrrFix16k88xxxMemType)—Bits 23–16. Memory type for physical address 88000–8BFFFh.

Memory Type Address 8C000–8FFFFh (MtrrFix16k8CxxxMemType)—Bits 31–24. Memory type for physical address 8C000–8FFFFh.

Memory Type Address 90000–93FFFh (MtrrFix16k90xxxMemType)—Bits 39–32. Memory type for physical address 90000–93FFFh.

Memory Type Address 94000–97FFFh (MtrrFix16k94xxxMemType)—Bits 47–40. Memory type for physical address 94000–97FFFh.

Memory Type Address 98000–9BFFFh (MtrrFix16k98xxxMemType)—Bits 55–48. Memory type for physical address 98000–9BFFFh.

Memory Type Address 9C000–9FFFFh (MtrrFix16k9CxxxMemType)—Bits 63–56. Memory type for physical address 9C000–9FFFFh.

14.1.2.6 MTRRfix16K_A0000 Register

This register controls the memory types for physical memory addresses A0000–BFFFFh. See the "Memory Type Range Registers" section in Volume 2 of the *AMD64 Architecture Programmer's Manual* for more information.

Setting any range to an undefined memory type results in a #GP(0).

MTRRfix16K_A0000 Register — MSR 0259h

63 – 56	55 – 48	47 – 40	39 – 32
MtrrFix16kBCxxxMemType	MtrrFix16kB8xxxMemType	MtrrFix16kB4xxxMemType	MtrrFix16kB0xxxMemType

31 – 24	23 – 16	15 – 8	7 – 0
MtrrFix16kACxxxMemType	MtrrFix16kA8xxxMemType	MtrrFix16kA4xxxMemType	MtrrFix16kA0xxxMemType

Bit	Mnemonic	Function	R/W	Reset
63–56	MtrrFix16kBCxxxMemType	Memory type address BC000–BFFFFh	R/W	U
55–48	MtrrFix16kB8xxxMemType	Memory type address B8000–BBFFFh	R/W	U
47–40	MtrrFix16kB4xxxMemType	Memory type address B4000–B7FFFh	R/W	U
39–32	MtrrFix16kB0xxxMemType	Memory type address B0000–B3FFFh	R/W	U
31–24	MtrrFix16kACxxxMemType	Memory type address AC000–AFFFFh	R/W	U
23–16	MtrrFix16kA8xxxMemType	Memory type address A8000–ABFFFh	R/W	U
15–8	MtrrFix16kA4xxxMemType	Memory type address A4000–A7FFFh	R/W	U
7–0	MtrrFix16kA0xxxMemType	Memory type address A0000–A3FFFh	R/W	U

* Memory types must be set to values consistent with system hardware.

Field Descriptions

Memory Type Address A0000–A3FFFh (MtrrFix16kA0xxxMemType)—Bits 7–0. Memory type for physical address A0000–A3FFFh.

Memory Type Address A4000–A7FFFh (MtrrFix16kA4xxxMemType)—Bits 15–8. Memory type for physical address A4000–A7FFFh.

Memory Type Address A8000–ABFFFh (MtrrFix16kA8xxxMemType)—Bits 23–16. Memory type for physical address A8000–ABFFFh.

Memory Type Address AC000–AFFFFh (MtrrFix16kACxxxMemType)—Bits 31–24. Memory type for physical address AC000–AFFFFh.

Memory Type Address B0000–B3FFFh (MtrrFix16kB0xxxMemType)—Bits 39–32. Memory type for physical address B0000–B3FFFh.

Memory Type Address B4000–B7FFFh (MtrrFix16kB4xxxMemType)—Bits 47–40. Memory type for physical address B4000–B7FFFh.

Memory Type Address B8000–BBFFFh (MtrrFix16kB8xxxMemType)—Bits 55–48. Memory type for physical address B8000–BBFFFh.

Memory Type Address BC000–BFFFFh (MtrrFix16kBCxxxMemType)—Bits 63–56. Memory type for physical address BC000–BFFFFh.

14.1.2.7 MTRRfix4K_C0000 Register

This register controls the memory types for physical memory addresses C0000–C7FFFh. See the "Memory Type Range Registers" section in Volume 2 of the *AMD64 Architecture Programmer's Manual* for more information.

Setting any range to an undefined memory type results in a #GP(0).

MTRRfix4K_C0000 Register **MSR 0268h**

63 56	55 48	47 40	39 32
MtrrFix4kC7xxxMemType	MtrrFix4kC6xxxMemType	MtrrFix4kC5xxxMemType	MtrrFix4kC4xxxMemType

31 24	23 16	15 8	7 0
MtrrFix4kC3xxxMemType	MtrrFix4kC2xxxMemType	MtrrFix4kC1xxxMemType	MtrrFix4kC0xxxMemType

Bit	Mnemonic	Function	R/W	Reset
63–56	MtrrFix4kC7xxxMemType	Memory type address C7000–C7FFFh	R/W	U
55–48	MtrrFix4kC6xxxMemType	Memory type address C6000–C6FFFh	R/W	U
47–40	MtrrFix4kC5xxxMemType	Memory type address C5000–C5FFFh	R/W	U
39–32	MtrrFix4kC4xxxMemType	Memory type address C4000–C4FFFh	R/W	U
31–24	MtrrFix4kC3xxxMemType	Memory type address C3000–C3FFFh	R/W	U
23–16	MtrrFix4kC2xxxMemType	Memory type address C2000–C2FFFh	R/W	U
15–8	MtrrFix4kC1xxxMemType	Memory type address C1000–C1FFFh	R/W	U
7–0	MtrrFix4kC0xxxMemType	Memory type address C0000–C0FFFh	R/W	U

* Memory types must be set to values consistent with system hardware.

Field Descriptions

Memory Type Address C0000–C0FFFh (MtrrFix4kC0xxxMemType)—Bits 7–0. Memory type for physical address C0000–C0FFFh.

Memory Type Address C1000–C1FFFh (MtrrFix4kC1xxxMemType)—Bits 15–8. Memory type for physical address C1000–C1FFFh.

Memory Type Address C2000–C2FFFh (MtrrFix4kC2xxxMemType)—Bits 23–16. Memory type for physical address C2000–C2FFFh.

Memory Type Address C3000–C3FFFh (MtrrFix4kC3xxxMemType)—Bits 31–24. Memory type for physical address C3000–C3FFFh.

Memory Type Address C40000–C4FFFh (MtrrFix4kC4xxxMemType)—Bits 39–32. Memory type for physical address C4000–C4FFFh.

Memory Type Address C5000–C5FFFh (MtrrFix4kC5xxxMemType)—Bits 47–40. Memory type for physical address C5000–C5FFFh.

Memory Type Address C6000–C6FFFh (MtrrFix4kC6xxxMemType)—Bits 55–48. Memory type for physical address C6000–C6FFFh.

Memory Type Address C7000–C7FFFh (MtrrFix4kC7xxxMemType)—Bits 63–56. Memory type for physical address C7000–C7FFFh.

14.1.2.8 MTRRfix4K_C8000 Register

This register controls the memory types for physical memory addresses C8000–CFFFFh. See the "Memory Type Range Registers" section in Volume 2 of the *AMD64 Architecture Programmer's Manual* for more information.

Setting any range to an undefined memory type results in a #GP(0).

MTRRfix4K_C8000 Register **MSR 0269h**

63–56	55–48	47–40	39–32
MtrrFix4kCFxxxMemType	MtrrFix4kCExxxMemType	MtrrFix4kCDxxxMemType	MtrrFix4kCCxxxMemType

31–24	23–16	15–8	7–0
MtrrFix4kCBxxxMemType	MtrrFix4kCAxxxMemType	MtrrFix4kC9xxxMemType	MtrrFix4kC8xxxMemType

Bit	Mnemonic	Function	R/W	Reset
63–56	MtrrFix4kCFxxxMemType	Memory type for address CF000–CFFFFh	R/W	U
55–48	MtrrFix4kCExxxMemType	Memory type for address CE000–CEFFFh	R/W	U
47–40	MtrrFix4kCDxxxMemType	Memory type for address CD000–CDFFFh	R/W	U
39–32	MtrrFix4kCCxxxMemType	Memory type for address CC000–CCFFFh	R/W	U
31–24	MtrrFix4kCBxxxMemType	Memory type for address CB000–CBFFFh	R/W	U
23–16	MtrrFix4kCAxxxMemType	Memory type for address CA000–CAFFFh	R/W	U
15–8	MtrrFix4kC9xxxMemType	Memory type for address C9000–C9FFFh	R/W	U
7–0	MtrrFix4kC8xxxMemType	Memory type for address C8000–C8FFFh	R/W	U

* Memory types must be set to values consistent with system hardware.

Field Descriptions

Memory Type Address C8000–C8FFFh (MtrrFix4kC8xxxMemType)—Bits 7–0. Memory type for physical address C8000–C8FFFh.

Memory Type Address C9000–C9FFFh (MtrrFix4kC9xxxMemType)—Bits 15–8. Memory type for physical address C9000–C9FFFh.

Memory Type Address CA000–CAFFFh (MtrrFix4kCAxxxMemType)—Bits 23–16. Memory type for physical address CA000–CAFFFh.

Memory Type Address CB000–CBFFFh (MtrrFix4kCBxxxMemType)—Bits 31–24. Memory type for physical address CB000–CBFFFh.

Memory Type Address CC0000–CCFFFh (MtrrFix4kCCxxxMemType)—Bits 39–32. Memory type for physical address CC000–CCFFFh.

Memory Type Address CD000–CDFFFh (MtrrFix4kCDxxxMemType)—Bits 47–40. Memory type for physical address CD000–CDFFFh.

Memory Type Address CE000–CEFFFh (MtrrFix4kCExxxMemType)—Bits 55–48. Memory type for physical address CE000–CEFFFh.

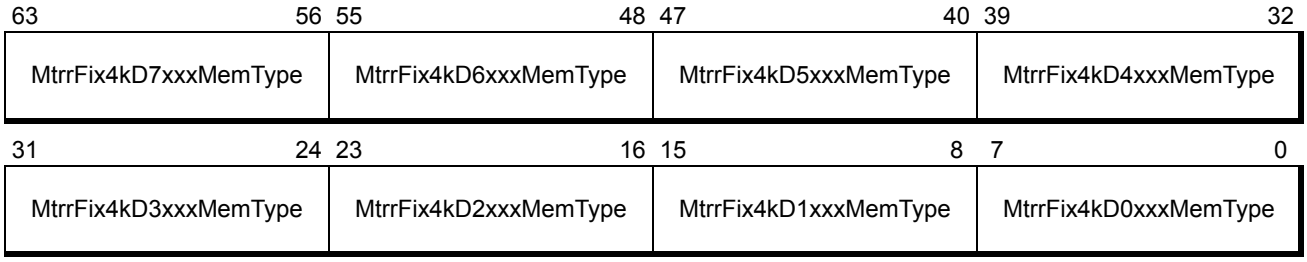
Memory Type Address CF000–CFFFFh (MtrrFix4kCFxxxMemType)—Bits 63–56. Memory type for physical address CF000–CFFFFh.

14.1.2.9 MTRRfix4K_D0000 Register

This register controls the memory types for physical memory addresses D0000–D7FFF. See the "Memory Type Range Registers" section in Volume 2 of the *AMD64 Architecture Programmer's Manual* for more information.

Setting any range to an undefined memory type results in a #GP(0).

MTRRfix4K_D0000 Register **MSR 026Ah**



Bit	Mnemonic	Function	R/W	Reset
63–56	MtrrFix4kD7xxxMemType	Memory type address D7000–D7FFFh	R/W	U
55–48	MtrrFix4kD6xxxMemType	Memory type address D6000–D6FFFh	R/W	U
47–40	MtrrFix4kD5xxxMemType	Memory type address D5000–D5FFFh	R/W	U
39–32	MtrrFix4kD4xxxMemType	Memory type address D4000–D4FFFh	R/W	U
31–24	MtrrFix4kD3xxxMemType	Memory type address D3000–D3FFFh	R/W	U
23–16	MtrrFix4kD2xxxMemType	Memory type address D2000–D2FFFh	R/W	U
15–8	MtrrFix4kD1xxxMemType	Memory type address D1000–D1FFFh	R/W	U
7–0	MtrrFix4kD0xxxMemType	Memory type address D0000–D0FFFh	R/W	U

* Memory types must be set to values consistent with system hardware.

Field Descriptions

Memory Type Address D0000–D0FFFh (MtrrFix4kD0xxxMemType)—Bits 7–0. Memory type for physical address D0000–D0FFFh.

Memory Type Address D1000–D1FFFh (MtrrFix4kD1xxxMemType)—Bits 15–8. Memory type for physical address D1000–D1FFFh.

Memory Type Address D2000–D2FFFh (MtrrFix4kD2xxxMemType)—Bits 23–16. Memory type for physical address D2000–D2FFFh.

Memory Type Address D3000–D3FFFh (MtrrFix4kD3xxxMemType)—Bits 31–24. Memory type for physical address D3000–D3FFFh.

Memory Type Address D40000–D4FFFh (MtrrFix4kD4xxxMemType)—Bits 39–32. Memory type for physical address D4000–D4FFFh.

Memory Type Address D5000–D5FFFh (MtrrFix4kD5xxxMemType)—Bits 47–40. Memory type for physical address D5000–D5FFFh.

Memory Type Address D6000–D6FFFh (MtrrFix4kD6xxxMemType)—Bits 55–48. Memory type for physical address D6000–D6FFFh.

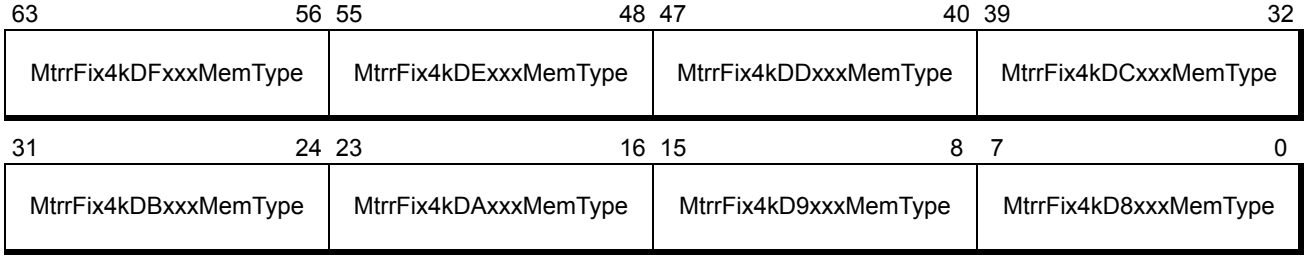
Memory Type Address D7000–D7FFFh (MtrrFix4kD7xxxMemType)—Bits 63–56. Memory type for physical address D7000–D7FFFh.

14.1.2.10 MTRRfix4K_D8000 Register

This register controls the memory types for physical memory addresses D8000–DFFFFh. See the "Memory Type Range Registers" section in Volume 2 of the *AMD64 Architecture Programmer's Manual* for more information.

Setting any range to an undefined memory type results in a #GP(0).

MTRRfix4K_D8000 Register **MSR 026Bh**



Bit	Mnemonic	Function	R/W	Reset
63–56	MtrrFix4kDFxxxMemType	Memory type address DF000–DFFFFh	R/W	U
55–48	MtrrFix4kDExxxMemType	Memory type address DE000–DEFFFh	R/W	U

* Memory types must be set to values consistent with system hardware.

Bit	Mnemonic	Function	R/W	Reset
47–40	MtrrFix4kDDxxxMemType	Memory type address DD000–DDFFFh	R/W	U
39–32	MtrrFix4kDCxxxMemType	Memory type address DC000–DCFFFh	R/W	U
31–24	MtrrFix4kDBxxxMemType	Memory type address DB000–DBFFFh	R/W	U
23–16	MtrrFix4kDAxxxMemType	Memory type address DA000–DAFFFh	R/W	U
15–8	MtrrFix4kD9xxxMemType	Memory type address D9000–D9FFFh	R/W	U
7–0	MtrrFix4kD8xxxMemType	Memory type address D8000–D8FFFh	R/W	U

* Memory types must be set to values consistent with system hardware.

Field Descriptions

Memory Type Address D8000–D8FFFh (MtrrFix4kD8xxxMemType)—Bits 7–0. Memory type for physical address D8000–D8FFFh.

Memory Type Address D9000–D9FFFh (MtrrFix4kD9xxxMemType)—Bits 15–8. Memory type for physical address D9000–D9FFFh.

Memory Type Address DA000–DAFFFh (MtrrFix4kDAxxxMemType)—Bits 23–16. Memory type for physical address DA000–DAFFFh.

Memory Type Address DB000–DBFFFh (MtrrFix4kDBxxxMemType)—Bits 31–24. Memory type for physical address DB000–DBFFFh.

Memory Type Address DC0000–DCFFFh (MtrrFix4kDCxxxMemType)—Bits 39–32. Memory type for physical address DC000–DCFFFh.

Memory Type Address DD000–DDFFFh (MtrrFix4kDDxxxMemType)—Bits 47–40. Memory type for physical address DD000–DDFFFh.

Memory Type Address DE000–DEFFFh (MtrrFix4kDExxxMemType)—Bits 55–48. Memory type for physical address DE000–DEFFFh.

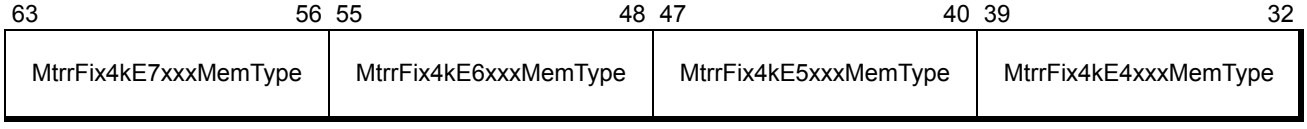
Memory Type Address DF000–DFFFFh (MtrrFix4kDFxxxMemType)—Bits 63–56. Memory type for physical address DF000–DFFFFh.

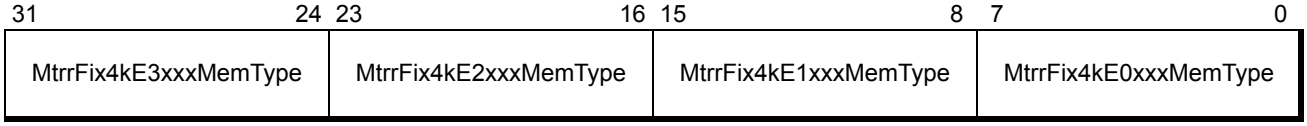
14.1.2.11 MTRRfix4K_E0000 Register

This register controls the memory types for physical memory addresses E0000–E7FFFh. See the "Memory Type Range Registers" section in Volume 2 of the *AMD64 Architecture Programmer's Manual* for more information.

Setting any range to an undefined memory type results in a #GP(0).

MTRRfix4K_E0000 Register **MSR 026Ch**





Bit	Mnemonic	Function	R/W	Reset
63–56	MtrrFix4kE7xxxMemType	Memory type for physical addr E7000–E7FFFh	R/W	U
55–48	MtrrFix4kE6xxxMemType	Memory type for physical addr E6000–E6FFFh	R/W	U
47–40	MtrrFix4kE5xxxMemType	Memory type for physical addr E5000–E5FFFh	R/W	U
39–32	MtrrFix4kE4xxxMemType	Memory type for physical addr E4000–E4FFFh	R/W	U
31–24	MtrrFix4kE3xxxMemType	Memory type for physical addr E3000–E3FFFh	R/W	U
23–16	MtrrFix4kE2xxxMemType	Memory type for physical addr E2000–E2FFFh	R/W	U
15–8	MtrrFix4kE1xxxMemType	Memory type for physical addr E1000–E1FFFh	R/W	U
7–0	MtrrFix4kE0xxxMemType	Memory type for physical addr E0000–E0FFFh	R/W	U

* Memory types must be set to values consistent with system hardware.

Field Descriptions

Memory Type Address E0000–E0FFFh (MtrrFix4kE0xxxMemType)—Bits 7–0. Memory type for physical address E0000–E0FFFh.

Memory Type Address E1000–E1FFFh (MtrrFix4kE1xxxMemType)—Bits 15–8. Memory type for physical address E1000–E1FFFh.

Memory Type Address E2000–E2FFFh (MtrrFix4kE2xxxMemType)—Bits 23–16. Memory type for physical address E2000–E2FFFh.

Memory Type Address E3000–E3FFFh (MtrrFix4kE3xxxMemType)—Bits 31–24. Memory type for physical address E3000–E3FFFh.

Memory Type Address E40000–E4FFFh (MtrrFix4kE4xxxMemType)—Bits 39–32. Memory type for physical address E4000–E4FFFh.

Memory Type Address E5000–E5FFFh (MtrrFix4kE5xxxMemType)—Bits 47–40. Memory type for physical address E5000–E5FFFh.

Memory Type Address E6000–E6FFFh (MtrrFix4kE6xxxMemType)—Bits 55–48. Memory type for physical address E6000–E6FFFh.

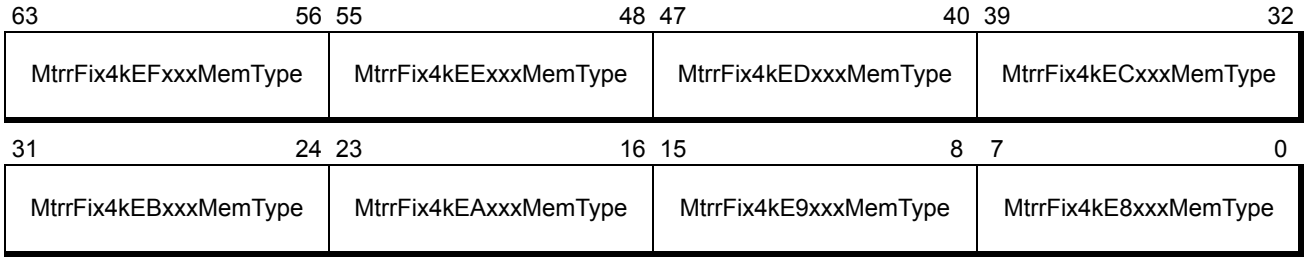
Memory Type Address E7000–E7FFFh (MtrrFix4kE7xxxMemType)—Bits 63–56. Memory type for physical address E7000–E7FFFh.

14.1.2.12 MTRRfix4K_E8000 Register

This register controls the memory types for physical memory addresses E8000–EFFFFh. See the "Memory Type Range Registers" section in Volume 2 of the *AMD64 Architecture Programmer's Manual* for more information.

Setting any range to an undefined memory type results in a #GP(0).

MTRRfix4K_E8000 Register — MSR 026Dh



Bit	Mnemonic	Function	R/W	Reset
63–56	MtrrFix4kEFxxxMemType	Memory type for physical addr EF000–EFFFFh	R/W	U
55–48	MtrrFix4kEExxxMemType	Memory type for physical addr EE000–EEFFFh	R/W	U
47–40	MtrrFix4kEDxxxMemType	Memory type for physical addr ED000–EDFFFh	R/W	U
39–32	MtrrFix4kECxxxMemType	Memory type for physical addr EC000–ECFFFh	R/W	U
31–24	MtrrFix4kEBxxxMemType	Memory type for physical addr EB000–EBFFFh	R/W	U
23–16	MtrrFix4kEAxxxMemType	Memory type for physical addr EA000–EAFFFh	R/W	U
15–8	MtrrFix4kE9xxxMemType	Memory type for physical addr E9000–E9FFFh	R/W	U
7–0	MtrrFix4kE8xxxMemType	Memory type for physical addr E8000–E8FFFh	R/W	U

* Memory types must be set to values consistent with system hardware.

Field Descriptions

Memory Type Address E8000–E8FFFh (MtrrFix4kE8xxxMemType)—Bits 7–0. Memory type for physical address E8000–E8FFFh.

Memory Type Address E9000–E9FFFh (MtrrFix4kE9xxxMemType)—Bits 15–8. Memory type for physical address E9000–E9FFFh.

Memory Type Address EA000–EAFFFh (MtrrFix4kEAxxxMemType)—Bits 23–16. Memory type for physical address EA000–EAFFFh.

Memory Type Address EB000–EBFFFh (MtrrFix4kEBxxxMemType)—Bits 31–24. Memory type for physical address EB000–EBFFFh.

Memory Type Address EC0000–ECFFFh (MtrrFix4kECxxxMemType)—Bits 39–32. Memory type for physical address EC000–ECFFFh.

Memory Type Address ED000–EDFFFh (MtrrFix4kEDxxxMemType)—Bits 47–40. Memory type for physical address ED000–EDFFFh.

Memory Type Address EE000–EEFFFh (MtrrFix4kEExxxMemType)—Bits 55–48. Memory type for physical address EE000–EEFFFh.

Memory Type Address EF000–EFFFFh (MtrrFix4kEFxxxMemType)—Bits 63–56. Memory type for physical address EF000–EFFFFh.

14.1.2.13 MTRRfix4K_F0000 Register

This register controls the memory types for physical memory addresses F0000–F7FFFh. See the "Memory Type Range Registers" section in Volume 2 of the *AMD64 Architecture Programmer's Manual* for more information.

Setting any range to an undefined memory type results in a #GP(0).

MTRRfix4K_F0000 Register **MSR 026Eh**

63 56	55 48	47 40	39 32
MtrrFix4kF7xxxMemType	MtrrFix4kF6xxxMemType	MtrrFix4kF5xxxMemType	MtrrFix4kF4xxxMemType

31 24	23 16	15 8	7 0
MtrrFix4kF3xxxMemType	MtrrFix4kF2xxxMemType	MtrrFix4kF1xxxMemType	MtrrFix4kF0xxxMemType

Bit	Mnemonic	Function	R/W	Reset
63–56	MtrrFix4kF7xxxMemType	Memory type address F7000–F7FFFh	R/W	U
55–48	MtrrFix4kF6xxxMemType	Memory type address F6000–F6FFFh	R/W	U
47–40	MtrrFix4kF5xxxMemType	Memory type address F5000–F5FFFh	R/W	U
39–32	MtrrFix4kF4xxxMemType	Memory type address F4000–F4FFFh	R/W	U
31–24	MtrrFix4kF3xxxMemType	Memory type address F3000–F3FFFh	R/W	U
23–16	MtrrFix4kF2xxxMemType	Memory type address F2000–F2FFFh	R/W	U
15–8	MtrrFix4kF1xxxMemType	Memory type address F1000–F1FFFh	R/W	U
7–0	MtrrFix4kF0xxxMemType	Memory type address F0000–F0FFFh	R/W	U

* Memory types must be set to values consistent with system hardware.

Field Descriptions

Memory Type Address F0000–F0FFFh (MtrrFix4kF0xxxMemType)—Bits 7–0. Memory type for physical address F0000–F0FFFh.

Memory Type Address F1000–F1FFFh (MtrrFix4kF1xxxMemType)—Bits 15–8. Memory type for physical address F1000–F1FFFh.

Memory Type Address F2000–F2FFFh (MtrrFix4kF2xxxMemType)—Bits 23–16. Memory type for physical address F2000–F2FFFh.

Memory Type Address F3000–F3FFFh (MtrrFix4kF3xxxMemType)—Bits 31–24. Memory type for physical address F3000–F3FFFh.

Memory Type Address F40000–F4FFFh (MtrrFix4kF4xxxMemType)—Bits 39–32. Memory type for physical address F4000–F4FFFh.

Memory Type Address F5000–F5FFFh (MtrrFix4kF5xxxMemType)—Bits 47–40. Memory type for physical address F5000–F5FFFh.

Memory Type Address F6000–F6FFFh (MtrrFix4kF6xxxMemType)—Bits 55–48. Memory type for physical address F6000–F6FFFh.

Memory Type Address F7000–F7FFFh (MtrrFix4kF7xxxMemType)—Bits 63–56. Memory type for physical address F7000–F7FFFh.

14.1.2.14 MTRRfix4K_F8000 Register

This register controls the memory types for physical memory addresses F8000–FFFFFh. See the "Memory Type Range Registers" section in Volume 2 of the *AMD64 Architecture Programmer's Manual* for more information.

Setting any range to an undefined memory type results in a #GP(0).

MTRRfix4K_F8000 Register **MSR 026Fh**

63–56	55–48	47–40	39–32
MtrrFix4kFFxxxMemType	MtrrFix4kFExxxMemType	MtrrFix4kFDxxxMemType	MtrrFix4kFCxxxMemType

31–24	23–16	15–8	7–0
MtrrFix4kFBxxxMemType	MtrrFix4kFAxxxMemType	MtrrFix4kF9xxxMemType	MtrrFix4kF8xxxMemType

Bit	Mnemonic	Function	R/W	Reset
63–56	MtrrFix4kFFxxxMemType	Memory type address FF000–FFFFFh	R/W	U
55–48	MtrrFix4kFExxxMemType	Memory type address FE000–FEFFFh	R/W	U
47–40	MtrrFix4kFDxxxMemType	Memory type address FD000–FDFFFh	R/W	U
39–32	MtrrFix4kFCxxxMemType	Memory type address FC000–FCFFFh	R/W	U
31–24	MtrrFix4kFBxxxMemType	Memory type address FB000–FBFFFh	R/W	U
23–16	MtrrFix4kFAxxxMemType	Memory type address FA000–FAFFFh	R/W	U
15–8	MtrrFix4kF9xxxMemType	Memory type address F9000–F9FFFh	R/W	U
7–0	MtrrFix4kF8xxxMemType	Memory type address F8000–F8FFFh	R/W	U

* Memory types must be set to values consistent with system hardware.

Field Descriptions

Memory Type Address F8000–F8FFFh (MtrrFix4kF8xxxMemType)—Bits 7–0. Memory type for physical address F8000–F8FFFh.

Memory Type Address F9000–F9FFFh (MtrrFix4kF9xxxMemType)—Bits 15–8. Memory type for physical address F9000–F9FFFh.

Memory Type Address FA000–FAFFFh (MtrrFix4kFAxxxMemType)—Bits 23–16. Memory type for physical address FA000–FAFFFh.

Memory Type Address FB000–FBFFFh (MtrrFix4kFBxxxMemType)—Bits 31–24. Memory type for physical address FB000–FBFFFh.

Memory Type Address FC0000–FCFFFh (MtrrFix4kFCxxxMemType)—Bits 39–32. Memory type for physical address FC000–FCFFFh.

Memory Type Address FD000–FDFFFh (MtrrFix4kFDxxxMemType)—Bits 47–40. Memory type for physical address FD000–FDFFFh.

Memory Type Address FE000–FEFFFh (MtrrFix4kFExxxMemType)—Bits 55–48. Memory type for physical address FE000–FEFFFh.

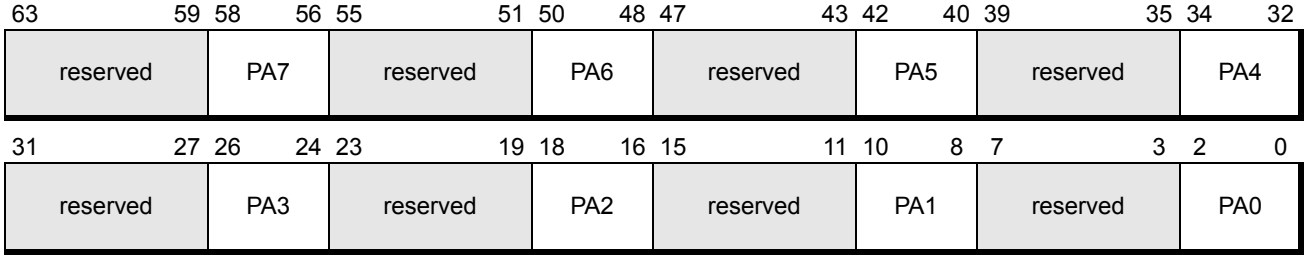
Memory Type Address FF000–FFFFFh (MtrrFix4kFFxxxMemType)—Bits 63–56. Memory type for physical address FF000–FFFFFh.

14.1.2.15 PAT Register

This register contains the eight page attribute fields used for specifying memory types for pages. See the "Page-Attribute Table Mechanism" section in Volume 2 of the *AMD64 Architecture Programmer's Manual* for more information.

Setting any range to an undefined memory type results in a #GP(0). Setting any reserved bits results in a #GP(0).

PAT Register **MSR 0277h**



Bit	Mnemonic	Function	R/W	Reset
63–59	reserved	MBZ		0
58–56	PA7	Memory type for Page Attribute index 7	R/W	0
55–51	reserved	MBZ		0
50–48	PA6	Memory type for Page Attribute index 6	R/W	7
47–43	reserved	MBZ		0
42–40	PA5	Memory type for Page Attribute index 5	R/W	4
39–35	reserved	MBZ		0
34–32	PA4	Memory type for Page Attribute index 4	R/W	6
31–27	reserved	MBZ		0

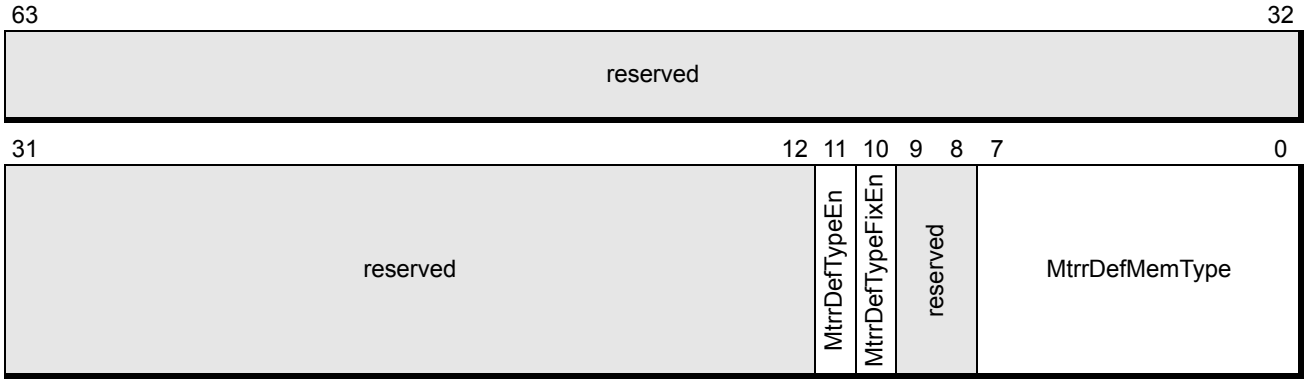
Bit	Mnemonic	Function	R/W	Reset
26–24	PA3	Memory type for Page Attribute index 3	R/W	0
23–19	reserved	MBZ		0
18–16	PA2	Memory type for Page Attribute index 2	R/W	7
15–11	reserved	MBZ		0
10–8	PA1	Memory type for Page Attribute index 1	R/W	4
7–3	reserved	MBZ		0
2–0	PA0	Memory type for Page Attribute index 0	R/W	6

14.1.2.16 MTRRdefType Register

This register enables the MTRRs and defines the default memory type for memory not within one of the MTRR ranges. See the "Memory Type Range Registers" section in Volume 2 of the *AMD64 Architecture Programmer's Manual* for more information.

Setting MtrrDefMemType to an undefined memory type results in a #GP(0). Setting any reserved bits results in a #GP(0).

MTRRdefType Register **MSR 02FFh**



Bit	Mnemonic	Function	R/W	Reset
63–12	reserved	MBZ		0
11	MtrrDefTypeEn	Enable MTRRs	R/W	0
10	MtrrDefTypeFixEn	Enable Fixed Range MTRRs	R/W	0
9–8	reserved			0
7–0	MtrrDefMemType	Default Memory Type	R/W	0

* Memory types must be set to values consistent with system hardware.

Field Descriptions

Default Memory Type (MtrrDefMemType)—Bit 7–0.

Enable Fixed Range MTRRs (MtrrDefTypeFixEn)—Bit 10.

Enable MTRRs (MtrrDefTypeEn)—Bit 11.

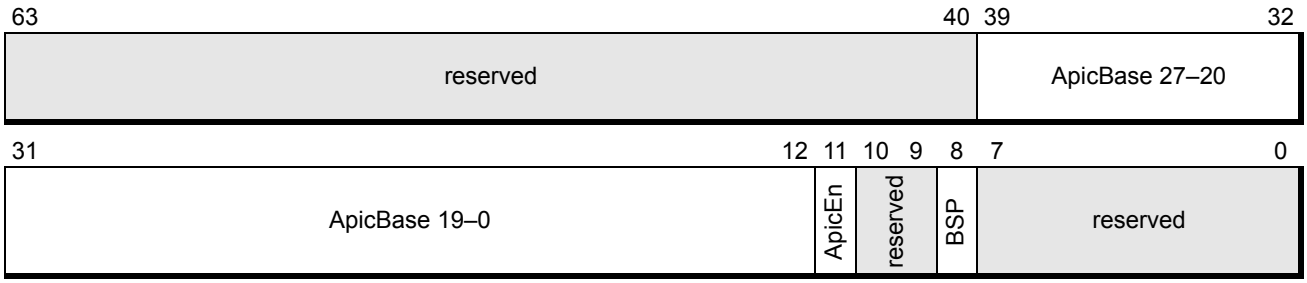
14.1.3 APIC Registers

14.1.3.1 APIC_BASE Register

This register enables APIC and defines the APIC base address. It also identifies the Boot Strap Processor (BSP).

Setting any reserved bits results in a #GP(0).

APIC_BASE Register **MSR 001Bh**



Bit	Mnemonic	Function	R/W	Reset
63–40	reserved	MBZ		0
39–12	ApicBase	APIC Base Address	R/W	00FEE00h
11	ApicEn	Enable APIC	R/W	0
10–9	reserved	MBZ		0
8	BSP	Boot Strap Processor	R/W	1 if uniprocessor or bsp of multiprocessor system
7–0	reserved	MBZ		0

* APIC configuration must be consistent with system hardware.

Field Descriptions

Boot Strap Processor (Bsp)—Bit 8.

Enable APIC (ApicEn)—Bit 11.

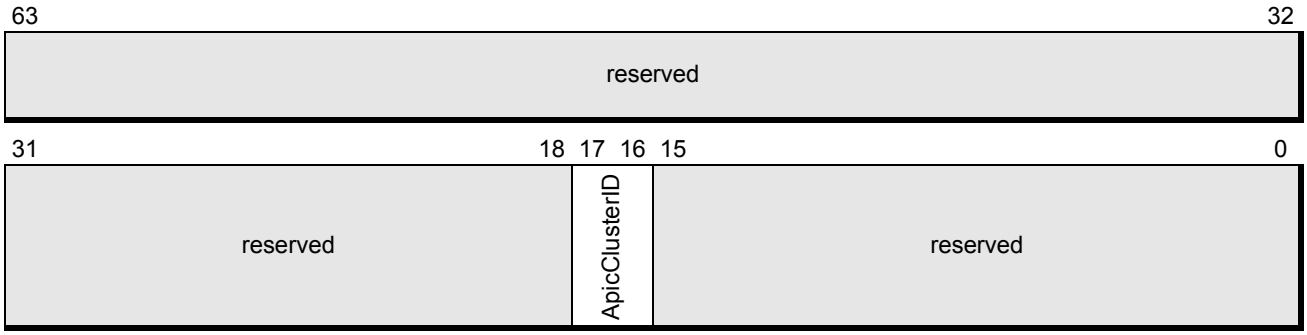
APIC Base Address (ApicBase)—Bits 39–12.

14.1.3.2 EBL_CR_POWERON Register

This read-only register contains the APIC cluster ID. See the "APIC Initialization" section in Volume 2 of the *AMD64 Architecture Programmer's Manual* for more information.

Attempting to write to this register results in a #GP(0).

EBL_CR_POWERON Register — MSR 002Ah



Bit	Mnemonic	Function	R/W
63–18	reserved	MBZ	
17–16	ApicClusterID	APIC Cluster ID	R
15–0	reserved	MBZ	

Field Descriptions

APIC Cluster ID (ApicClusterID)—Bits 17–16.

14.1.4 Software Debug Registers

AMD NPT Family 0Fh Processors incorporate extensive debug features. These include the following control, status, and control-transfer recording MSRs.

- DebugCtl Register (MSR 01D9h)—Provides additional debug controls over control-transfer recording and single stepping, as well as external-breakpoint reporting and trace messages.
- LastBranchFromIP Register (MSR 01DBh)—Loaded with the segment offset of the branch instruction.
- LastBranchToIP Register (MSR 01DCh)—Holds the target rIP of the last branch that occurred before an exception or interrupt.
- LastExceptionFromIP Register (MSR 01DDh)—Holds the source rIP of the last branch that occurred before the exception or interrupt.
- LastExceptionToIP Register (MSR 01DEh)—Holds the target rIP of the last branch that occurred before the exception or interrupt.

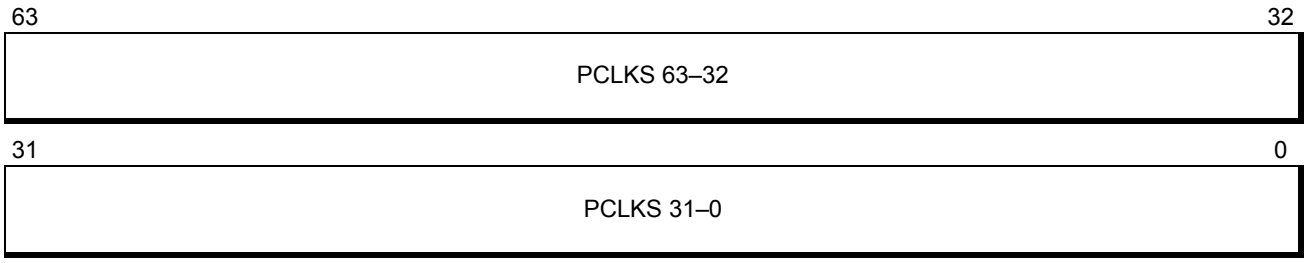
For more detailed information on the use of these MSRs, see the "Debug and Performance Resources" section in Volume 2 of the *AMD64 Architecture Programmer's Manual*.

14.1.5 Performance Monitoring Registers

14.1.5.1 TSC Register

The time-stamp counter (TSC) register maintains a running count of the number of internal processor clock cycles executed after a reset. It is incremented by 1 on each internal processor clock. When the TSC overflows its 64-bit range, it wraps around to 0. See the "Time-Stamp Counter" section in Volume 2 of the *AMD64 Architecture Programmer's Manual* for more information.

TSC Register **MSR 0010h**



Bit	Mnemonic	Function	R/W	Reset
63–0	PCLKS	Running Clock Cycle Count	R/W	0

Field Descriptions

Processor Clock Cycles (PCLKS)—Bits 63–0. Running count of number of internal processor clock cycles.

14.2 AMD NPT Family 0Fh Processor Model-Specific Registers

Table 82 is a listing of the model-specific registers supported by AMD NPT Family 0Fh Processors, presented in ascending hexadecimal address order. Register descriptions follow the table, organized according to the following functions:

- Features (see page 408)
- Identification (see page 415)
- Memory typing (see page 417)
- I/O range registers (see page 418)
- System call extension registers (see page 419)
- Segmentation (see page 422)

- System management (see page 271)
- Power management (see page 423)

Table 82. AMD NPT Family 0Fh Processor MSRs

Address	Register Name	Description
C000_0080h	EFER	page 408
C000_0081h	STAR	page 419
C000_0082h	LSTAR	page 420
C000_0083h	CSTAR	page 420
C000_0084h	SF_MASK	page 421
C000_0100h	FS.Base	page 422
C000_0101h	GS.Base	page 422
C000_0102h	KernelGSbase	page 423
C001_0000h–C001_0003h	PerfEvtSel*i*	page 334
C001_0004h–C001_0007h	PerfCtr*i*	page 333
C001_0010h	SYSCFG	page 409
C001_0015h	HWCR	page 411
C001_0016h, C001_0018h	IORRBase[1:0]	page 418
C001_0017h, C001_0019h	IORRMask[1:0]	page 419
C001_001Ah	TOP_MEM	page 417
C001_001Dh	TOP_MEM2	page 417
C001_001Eh	MANID	page 415
C001_001Fh	NB_CFG	page 413
C001_003Eh	HTC	page 415
C001_003Fh	Thermal Control	page 416
C001_0041h	FIDVID_CTL	page 423
C001_0042h	FIDVID_STATUS	page 425
C001_0044–C001_0048h	MCi_CTL_MASK	page 222
C001_0050–C001_0053h	IOTRAP_ADDR*i*	page 428
C001_0054h	IOTRAP_CTL	page 429

Table 82. AMD NPT Family 0Fh Processor MSRs (Continued)

C001_0055h	Interrupt Pending Message	page 431
C001_0111h	SMM_BASE	page 278
C001_0112h	SMM_ADDR	page 283
C001_0113h	SMM_MASK	page 282
C001_0114	VM_CR	page 433
C001_0117	VM_HSAVE_PA	page 434

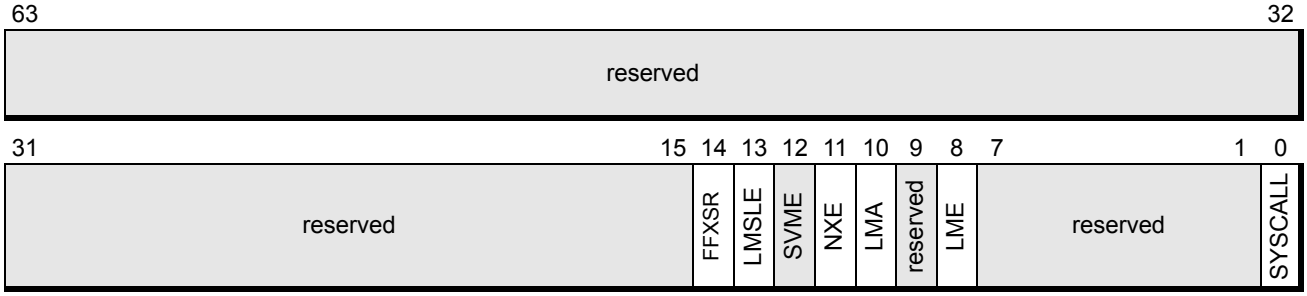
14.2.1 Feature Registers

14.2.1.1 EFER Register

This register controls which extended features are enabled. See the "Extended Feature Enable Register (EFER)" section in Volume 2 of the *AMD64 Architecture Programmer's Manual* for more information.

The LMA bit is a read-only status bit. When writing EFER, the processor signals a #GP(0) if an attempt is made to change LMA from its previous value. BIOS should not modify this register unless it needs to use one of the features enabled by this register.

EFER Register **MSR C000_0080h**



Bit	Mnemonic	Function	R/W	Reset
63–15	reserved	MBZ	R/W	0
14	FFXSR	Fast FXSAVE/FRSTOR Enable	R/W	0
13	LMSLE	Long Mode Segment Limit Enable	R/W	0
12	SVME	SVM Enable	R/W	0
11	NXE	No-Execute Page Enable	R/W	0
10	LMA	Long Mode Active	R	0
9	reserved	MBZ	R	0
8	LME	Long Mode Enable	R/W	0

Bit	Mnemonic	Function	R/W	Reset
7–1	reserved	RAZ	R	0
0	SYSCALL	System Call Extension Enable	R/W	0

Field Descriptions

System Call Extension Enable (SCE)—Bit 0. Enables the system call extension.

Long Mode Enable (LME)—Bit 8. Enables the long mode feature.

Long Mode Active (LMA)—Bit 10. Indicates the long mode feature is active.

No-Execute Page Enable (NXE)—Bit 11. Enables the no-execute page feature.

Long Mode Segment Limit Enable (LMSLE)—Bit 13. Enables the long mode segment limit check mechanism.

SVM Enable (SVME)—Bit 12. Enables SVM features.

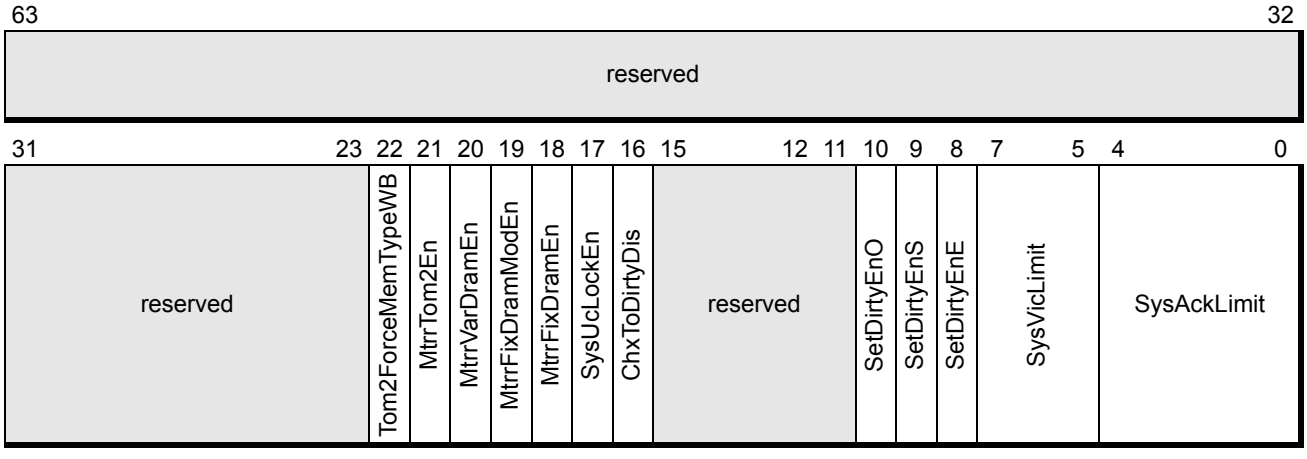
Fast FXSAVE/FRSTOR Enable (FFXSE)—Bit 14. Enables the fast FXSAVE/FRSTOR mechanism. A 64-bit operating system uses EDX bit 24 as returned by CPUID instruction standard function 1 to determine the presence of this feature before enabling it. This bit is set once by the operating system and its value is not changed afterwards.

14.2.1.2 SYSCFG Register

This register controls the system configuration.

The MtrrFixDramModEn bit should be set to 1 during BIOS initialization of the fixed MTRRs, then cleared to 0 for operation.

SYSCFG Register **MSR C001_0010h**



Bit	Mnemonic	Function	R/W	Reset	BIOS
63–21	reserved	RAZ	R	0	
22	Tom2ForceMemTypeWB	Top of Memory 2 Memory Type Write Back	R/W	0	0
21	MtrrTom2En	Top of Memory Address Register 2 Enable (reserved)	R/W	0	0
20	MtrrVarDramEn	Top of Memory Address Register and I/O Range Register Enable	R/W	0	1
19	MtrrFixDramModEn	RdDram and WrDram Bits Modification Enable	R/W	0	0
18	MtrrFixDramEn	Fixed RdDram and WrDram Attributes Enable	R/W	0	1
17	SysUcLockEn	System Interface Lock Command Enable	R/W	1	1
16	ChxToDirtyDis	Change to Dirty Command Disable	R/W	0	0
15–11	reserved	RAZ	R	0	
10	SetDirtyEnO	SharedToDirty Command for O->M State Transition Enable	R/W	1	1
9	SetDirtyEnS	SharedToDirty Command for S->M State Transition Enable	R/W	1	1
8	SetDirtyEnE	CleanToDirty Command for E->M State Transition Enable	R/W	0	0
7–5	SysVicLimit	Outstanding Victim Bus Command Limit	R/W	000b	000b
4–0	SysAckLimit	Outstanding Bus Command Limit	R/W	00001b	00001b

Field Descriptions

Outstanding Bus Command Limit (SysAckLimit)—Bit 4–0. Limits maximum number of outstanding bus commands.

Outstanding Victim Bus Command Limit (SysVicLimit)—Bit 7–5. Limits maximum number of outstanding victim bus commands.

CleanToDirty Command for E->M State Transition Enable (SetDirtyEnE)—Bit 8. Enables generating write probes when transitioning a cache line from Exclusive to Modified.

SharedToDirty Command for S->M State Transition Enable (SetDirtyEnS)—Bit 9. Enables generating write probes when transitioning a cache line from Shared to Modified.

SharedToDirty Command for O->M State Transition Enable (SetDirtyEnO)—Bit 10. Enables generating write probes when transitioning a cache line from Owned to Modified.

Change to Dirty Command Disable (ChxToDirtyDis)—Bit 16. Disables change to dirty commands, evicts line from DC instead.

System Interface Lock Command Enable (SysUcLockEn)—Bit 17. Enables lock commands on system interface.

Fixed RdDram and WrDram Attributes Enable (MtrrFixDramEn)—Bit 18. Enables fixed MTRR RdDram and WrDram attributes.

RdDram and WrDram Bits Modification Enable (MtrrFixDramModEn)—Bit 19. Enables modification of RdDram and WrDram bits in fixed MTRRS.

Top of Memory Address Register and I/O Range Register Enable (MtrrVarDramEn)—Bit 20. Enables use of top of memory address register and the I/O range registers.

Top of Memory Address Register 2 Enable (MtrrTom2En)—Bit 21. Enables use of top of memory address register 2 (reserved).

Top of Memory 2 Memory Type Write Back (Tom2ForceMemTypeWB)—Bit 22. When this bit is set, the default memory type of memory between 4GB and TOM2 is write back instead of the memory type defined by MtrrDefMemType (MSR 02FFh bits 7-0). MtrrDefTypeEn (MSR 02FFh bit 11) must be set to one for this bit to have any effect. Both MTRR's and PAT can be used to override this memory type. See "Register Differences in Revisions of AMD NPT Family 0Fh Processors" on page 27 for revision information about this field.

14.2.1.3 HWCR Register

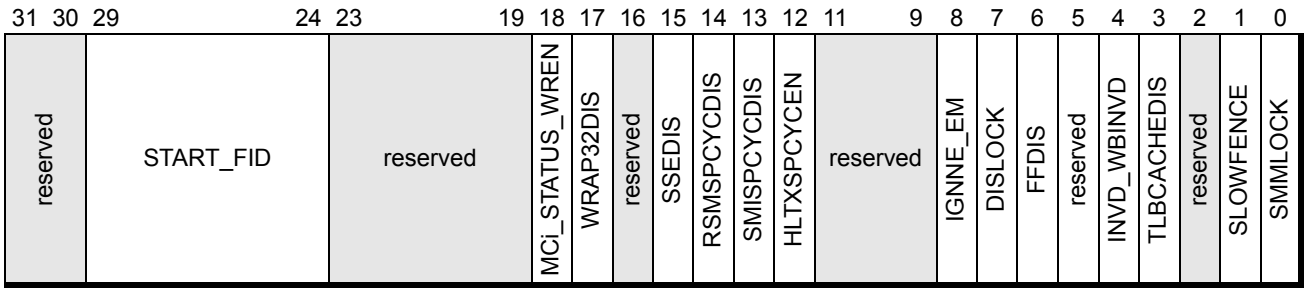
This register controls the hardware configuration.

Operating systems that maintain page tables in uncacheable memory (UC memory type) must set the TLBCACHEDIS bit to insure proper operation.

BIOS and SMM handlers may use the WRAP32DIS bit to access physical memory above 4 Gbytes without switching into 64-bit mode. By setting WRAP32DIS to a 1 in conjunction with setting the expanded FS or GS base registers, BIOS can size all of physical memory from legacy mode. To do so, BIOS would write a >32 bit base to the FS or GS base register using WRMSR. Then it would address ±2 Gbytes from one of those bases using normal memory reference instructions with a FS or GS override prefix. However, the INVLPG, FST, and SSE store instructions generate 32-bit addresses in legacy mode, regardless of the state of WRAP32DIS.

The MCi_STATUS_WREN bit can be used to help debug Machine Check exception handlers. When the MCi_STATUS_WREN bit is set, privileged software can write non-zero values to the MCG_STATUS, MCi_STATUS, MCi_ADDR, and if implemented, MCi_MISC MSRs without generating exceptions, and then simulate a machine check using the INT 18 instruction. Setting a reserved bit in these MSRs does not generate an exception when this mode is enabled. However, setting a reserved bit may result in undefined behavior.

HWCR Register **MSR C001_0015h**



Bit	Mnemonic	Function	R/W	Reset
63–32	reserved	RAZ	R	0
31–30	reserved	SBZ	R/W	0
29–24	START_FID	Startup FID Status	R	0
23–22	reserved	SBZ	R/W	0
21-20	reserved	RAZ	R	0
19	reserved	SBZ	R/W	0
18	MCi_STATUS_WREN	MCi Status Write Enable	R/W	0
17	WRAP32DIS	32-bit Address Wrap Disable	R/W	0
16	reserved			0
15	SSEDIS	SSE Instructions Disable	R/W	0
14	RSMSPCYCDIS	Special Bus Cycle On RSM Disable	R/W	0
13	SMISPCYCDIS	Special Bus Cycle On SMI Disable	R/W	0
12	HLTXSPCYCEN	Enable Special Bus Cycle On Exit From HLT	R/W	0
11–9	reserved	SBZ	R/W	0
8	IGNNE_EM	IGNNE Port Emulation Enable	R/W	0
7	DISLOCK	Disable x86 LOCK prefix functionality	R/W	0
6	FFDIS	TLB Flush Filter Disable	R/W	0
5	reserved	SBZ	R/W	0
4	INVD_WBINVD	INVD to WBINVD Conversion	R/W	0
3	TLBCACHEDIS	Cacheable Memory Disable	R/W	0
2	reserved	SBZ	R/W	0
1	SLOWFENCE	Slow SFENCE Enable	R/W	0
0	SMMLOCK	SMM Configuration Lock	R/W	0

Field Descriptions

SMM Code Lock (SMMLOCK)—Bit 0. Locks SMM configuration registers SMM_BASE, SMM_ADDR, SMM_MASK (all but SMM_MASK[3:2]), and HWCR[14:13, 0] by making them read-only.

Slow SFENCE Enable (SLOWFENCE)—Bit 1. Enable slow sfence.

Cachable Memory Disable (TLBCACHEDIS)—Bit 3. Disable performance improvement that assumes that the PML4, PDP, PDE and PTE entries are in cacheable memory. If page tables are uncachable, TLBCACHEDIS must be set to ensure correct functionality.

INVD to WBINVD Conversion (INVD_WBINVD)—Bit 4. Convert INVD to WBINVD. BIOS is recommended to not change the state of this bit.

TLB Flush Filter Disable (FFDIS)—Bit 6. Disable TLB flush filter.

Disable LOCK (DISLOCK)—Bit 7. Disable x86 LOCK prefix functionality.

IGNNE Port Emulation Enable (IGNNE_EM)—Bit 8. Enable emulation of IGNNE port.

Special Bus Cycle On HLT Exit Enable (HLTXSPCYCEN)—Bit 12. Enables special bus cycle generation on exit from HLT.

Special Bus Cycle On SMI Disable (SMISPCYCDIS)—Bit 13. Disables special bus cycle on SMI.

Special Bus Cycle On RSM Disable (RSMSPCYCDIS)—Bit 14. Disables special bus cycle on RSM.

SSE Instructions Disable (SSEDIS)—Bit 15. Disables SSE instructions.

32-bit Address Wrap Disable (WRAP32DIS)—Bit 17. Disable 32-bit address wrapping.

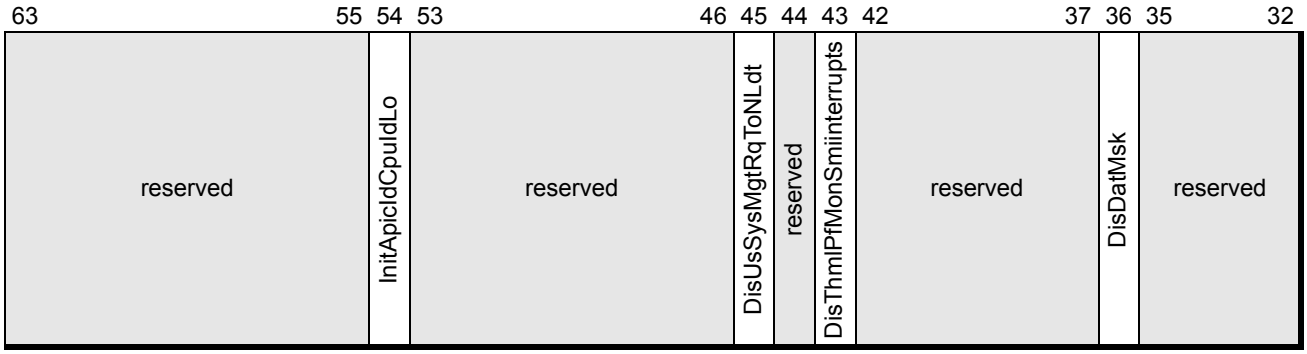
MCi_STATUS Write Enable (MCi_STATUS_WREN)—Bit 18. When set, writes by software to MCi_STATUS MSRs do not cause general protection faults. Such writes update all implemented bits in these registers. When clear, writing a non-zero pattern to these registers causes a general protection fault. See the description above for more details.

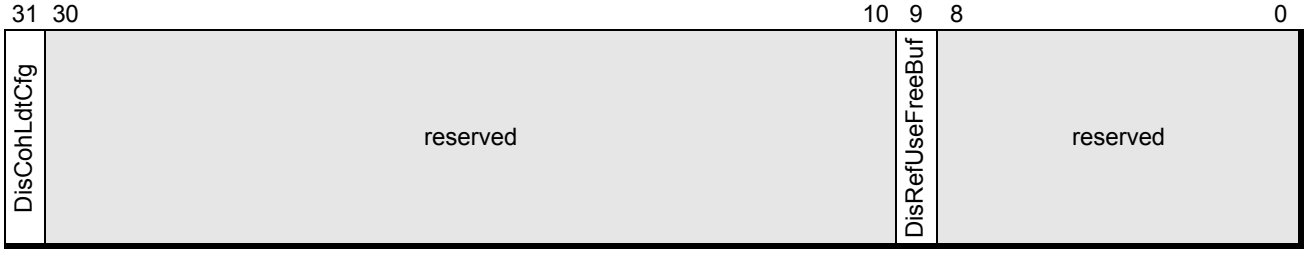
Startup FID Status (START_FID)—Bits 29–24. Status of the startup FID.

14.2.1.4 NB_CFG Register

Software must perform a read-modify-write to this register to change its value.

NB_CFG Register **MSR C001_001Fh**





Bit	Mnemonic	Function	R/W	Reset
63–55	reserved		R/W	0
54	InitApicIdCpuIdLo	Initial APIC ID CPU ID Low	R/W	0
53–46	reserved		R/W	0
45	DisUsSysMgtRqToNLdt	Disable Upstream System Management Re-broadcast	R/W	0
44	reserved		R/W	0
43	DisThmlPfMonSmiinterrupts	Disable Performance Monitor SMI	R/W	0
42–37	reserved		R/W	0
36	DisDatMsk	Disable Data Mask	R/W	0
35–32	reserved		R/W	0
31	DisCohLdtCfg	Disable Coherent HyperTransport Configuration Accesses	R/W	0
30–10	reserved		R/W	0
9	DisRefUseFreeBuf	Disable Display Refresh from Using Free List Buffers	R/W	0
8–0	reserved		R/W	0

Field Descriptions

Disable Display Refresh from Using Free List Buffers (DisRefUseFreeBuf)—Bit 9. Disables display refresh requests from using free list buffers.

Disable Coherent HyperTransport Configuration Accesses (DisCohLdtCfg)—Bit 31. Disables automatic routing of PCI configuration accesses to the processor configuration registers. When set, PCI configuration space accesses which fall within the hard-coded range reserved for AMD NPT Family 0Fh Processor registers (see "Memory System Configuration Registers" on page 53) are instead routed via the configuration address maps. This can be used to effectively hide the configuration registers from software if they are routed to the I/O hub, where they will then get a master abort. It can also be used to provide a means for an external chip to route processor configuration accesses according to some scheme other than the hard-coded version described in "Memory System Configuration Registers" on page 53. When used, this bit needs to be set on all processors in a system. PCI configuration accesses should not be generated if this bit is not set on all processors.

Disable Data Mask (DisDatMsk)—Bit 36. Disables DRAM data masking function. For all sized write requests a DRAM read is performed before writing the data.

Disable Performance Monitor SMI (DisThmlPfMonSmiIntr)—Bit 43. Disables SMI generation for Performance Monitor interrupts.

Disable Upstream System Management Rebroadcast (DisUsSysMgtRqToNLdt)—Bit 45. Disables re-broadcast of Upstream StpClk and Legacy Input System Management commands downstream.

Initial APIC ID CPU ID Low (InitApicIdCpuIdLo)—Bit 54. When this bit is set, CpuId and NodeId[2:0] bit field positions are swapped in the APICID.

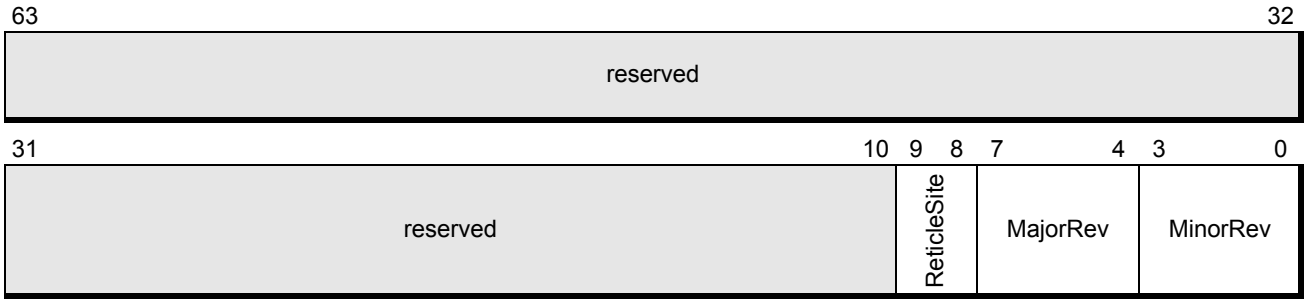
0: APICID = {CpuId, NodeId[2:0]}
1: APICID = {NodeId[2:0], CpuId}

14.2.2 Identification Registers

14.2.2.1 MANID Register

This status register holds the mask manufacturing identification number.

MANID Register **MSR C001_001Eh**



Bit	Mnemonic	Function	R/W
63–10	reserved		
9–8	ReticleSite	Reticle site	R
7–4	MajorRev	Major mask set revision number	R
3–0	MinorRev	Minor mask set revision number	R

Field Descriptions

Minor Mask Set Revision Number (MinorRev)—Bits 3–0.

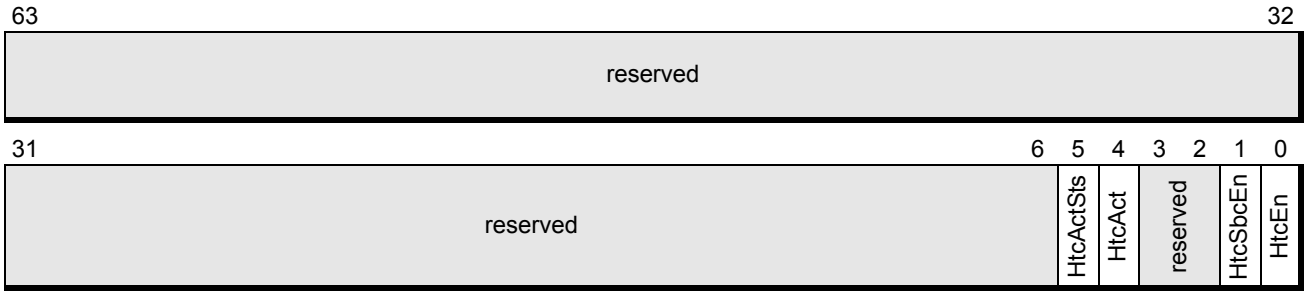
Major Mask Set Revision Number (MajorRev)—Bits 7–4.

Reticle Site (ReticleSite)—Bits 9–8.

14.2.3 HTC Register

See "HTC Register" for a complete description of this register.

HTC Register **MSR C001_003Eh**

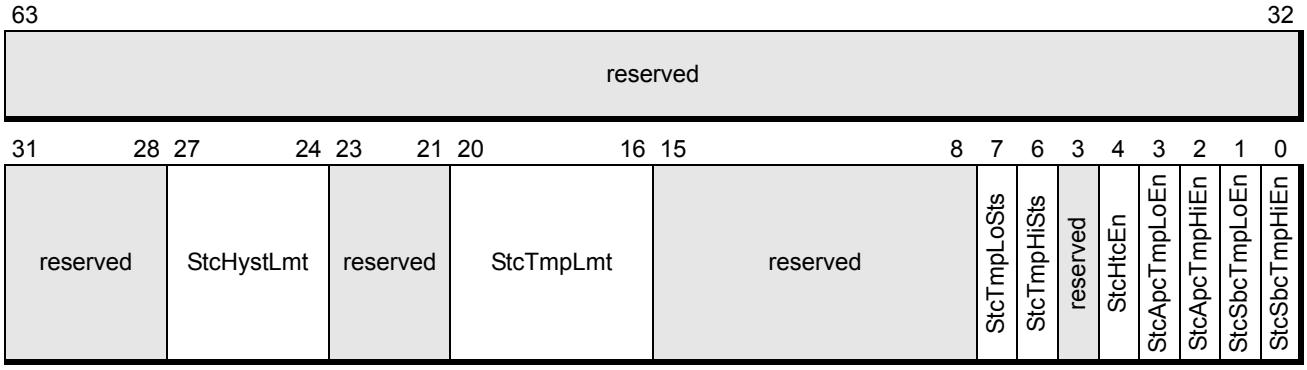


Bits	Mnemonic	Function	R/W	Reset
63–6	reserved		R	0
5	HtcActSts	HTC Active Status	R/W	0
4	HtcAct	HTC Active	R	0
3–2	reserved		R	0
1	HtcSbcEn	HTC Special Bus Cycle Enable	R/W	0
0	HtcEn	HTC Enable	R/W	0

14.2.4 Thermal Control Register

See "Thermal Control Register" for a complete description of this register.

Thermal Control Register **MSR C001_003Fh**



Bits	Mnemonic	Function	R/W	Reset
31-28	reserved		R	0
27-24	StcHystLmt	STC Hysteresis Limit	R/W	0
23–21	reserved		R	0
20-16	StcTmpLmt	STC Temperature Limit	R/W	0
15–8	reserved		R	0
7	StcTmpLoSts	STC Temperature Low Status	R/W	0
6	StcTmpHiSts	STC Temperature High Status	R/W	0
5	reserved		R	0

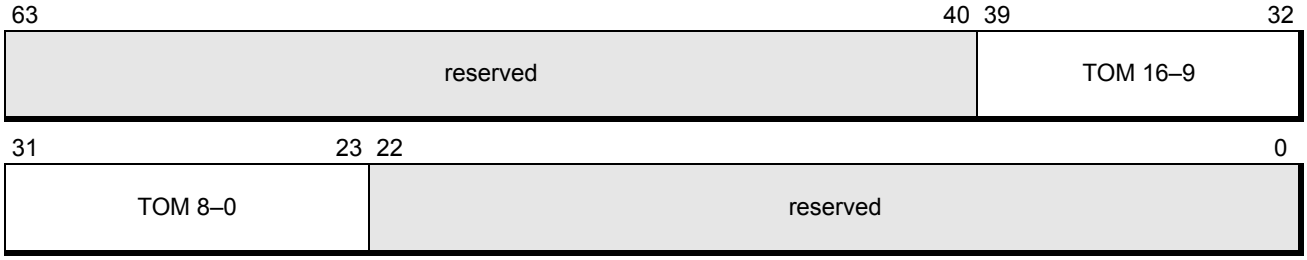
Bits	Mnemonic	Function	R/W	Reset
4	StcHtcEn	STC HTC Enable	R/W	0
3	StcApcTmpLoEn	STC APIC Temperature Low Interrupt Enable	R/W	0
2	StcApcTmpHiEn	STC APIC Temperature High Interrupt Enable	R/W	0
1	StcSbcTmpLoEn	STC Special Bus Cycle Temperature Low Enable	R/W	0
0	StcSbcTmpHiEn	STC Special Bus Cycle Temperature High Enable	R/W	0

14.2.5 Memory Typing Registers

14.2.5.1 TOP_MEM Register

This register holds the address of the top of memory. When enabled, this address indicates the first byte of I/O above DRAM.

TOP_MEM Register **MSR C001_001Ah**



Bit	Mnemonic	Function	R/W	Reset
63–40	reserved	RAZ		
39–23	TOM	Top of Memory	R/W	
22–0	reserved	RAZ		

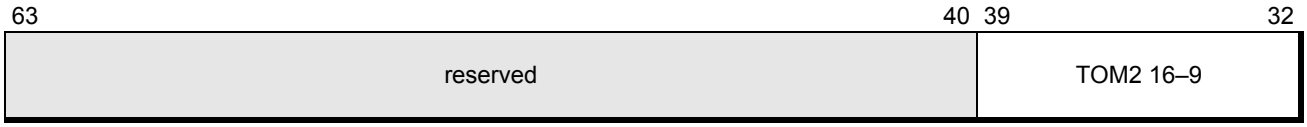
Field Descriptions

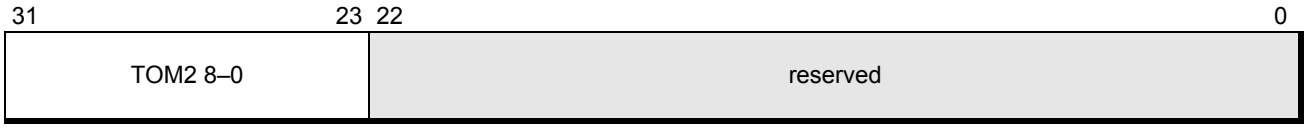
Top of Memory (TOM)—Bits 39–23. Address of top of memory (8-Mbyte granularity).

14.2.5.2 TOP_MEM2 Register

This register holds the second top of memory address. When enabled by setting the MtrrTom2En bit in the SYSCFG register, this address indicates the first byte of I/O above a second allocation of DRAM that starts at 4 Gbytes. Hence, the address in TOP_MEM2 should be set above 4 Gbytes.

TOP_MEM2 Register **MSR C001_001Dh**





Bit	Mnemonic	Function	R/W	Reset
63–40	reserved	RAZ		
39–23	TOM2	Second Top of Memory	R/W	
22–0	reserved	RAZ		

Field Descriptions

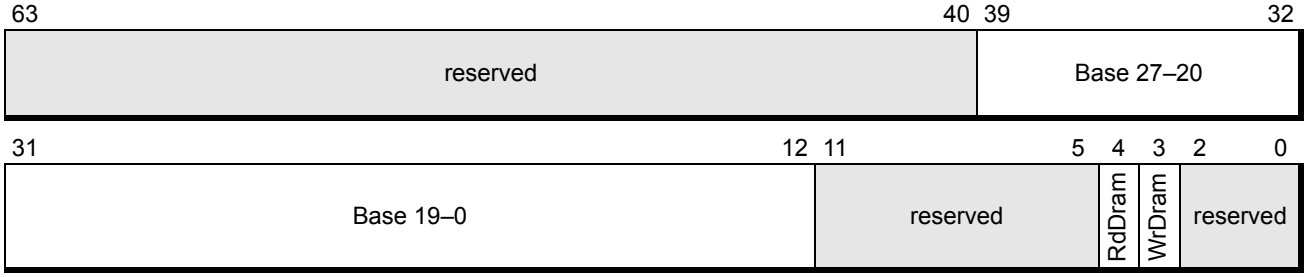
Second Top of Memory (TOM2)—Bits 39–23. Address of second top of memory (8-Mbyte granularity).

14.2.6 I/O Range Registers

14.2.6.1 IORRBase*i* Registers

These registers hold the bases of the variable I/O ranges.

IORRBase0–1 Registers **MSRs C001_0016h, C001_0018h**



Bit	Mnemonic	Function	R/W	Reset
63–40	reserved	RAZ		
39–12	Base	Base address	R/W	
11–5	reserved	RAZ		
4	RdDram	Read from DRAM	R/W	
3	WrDram	Write to DRAM	R/W	
2–0	reserved	RAZ		

Field Descriptions

Write to DRAM (WrDram)—Bit 3. If set, stores write to DRAM, otherwise I/O for this range.

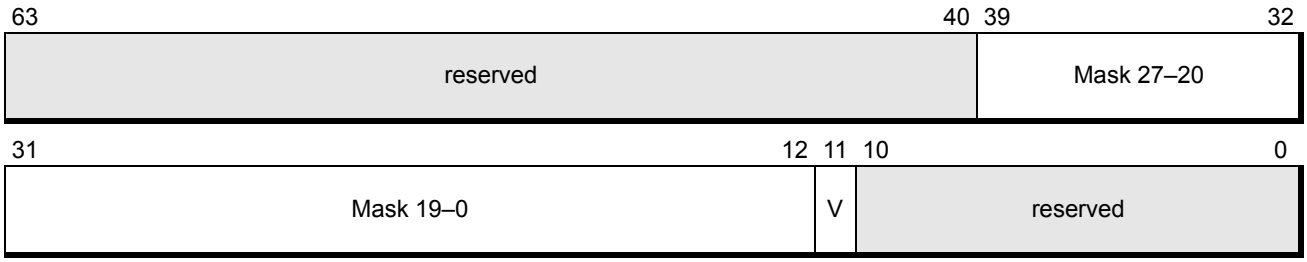
Read from DRAM (RdDram)—Bit 4. If set, fetches read from DRAM, otherwise from I/O for this range.

Base Address (Base)—Bits 39–12. Base address for this range.

14.2.6.2 IORRMask*i* Registers

These registers hold the masks of the variable I/O ranges.

IORRMask0–1 Registers **MSRs C001_0017h, C001_0019h**



Bit	Mnemonic	Function	R/W	Reset
63–40	reserved	RAZ		
39–12	Mask	Address mask	R/W	
11	V	Enables variable I/O range registers	R/W	
10–0	reserved	RAZ		

Field Descriptions

Variable I/O Range (V)—Bit 11. Enables variable I/O range register.

Address Mask (Mask)—Bits 39–12. Address mask.

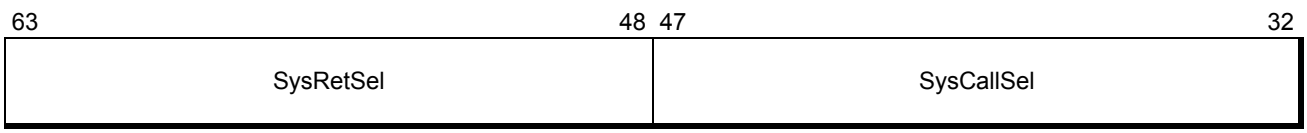
14.2.7 System Call Extension Registers

The system call extension is enabled by setting bit 0 in the EFER register. This feature adds the SYSCALL and SYSRET instructions which can be used in flat addressed operating systems as low latency system calls and returns. See the "SYSCALL and SYSRET" section in Volume 2 of the *AMD64 Architecture Programmer's Manual* for more information.

14.2.7.1 STAR Register

This register holds the target address used by the SYSCALL instruction and the code and stack segment selector bases used by the SYSCALL and SYSRET instructions.

STAR Register **MSR C000_0081h**



Bit	Mnemonic	Function	R/W	Reset
63–48	SysRetSel	SYSRET CS and SS	R/W	
47–32	SysCallSel	SYSCALL CS and SS	R/W	
31–0	Target	SYSCALL target address	R/W	

Field Descriptions

SYSCALL Target Address (Target)—Bits 31–0.

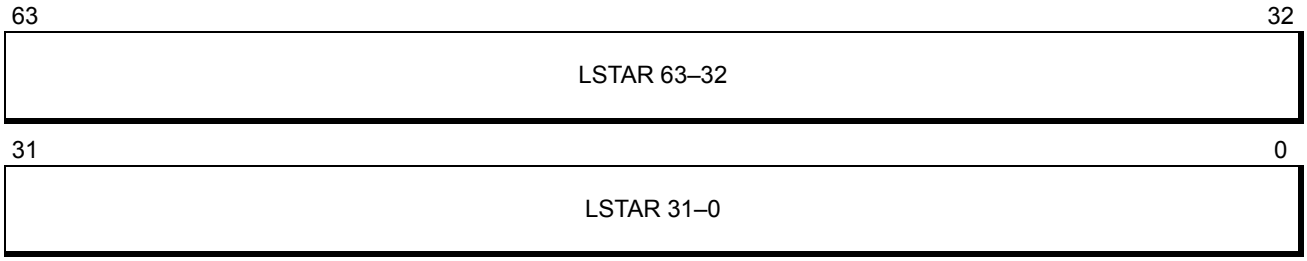
SYSCALL CS and SS (SysCallSel)—Bits 47–32.

SYSRET CS and SS (SysRetSel)—Bits 63–48.

14.2.7.2 LSTAR Register

The address stored in this register must be in canonical form (if not canonical, a #GP fault occurs).

LSTAR Register **MSR C000_0082h**



Bit	Mnemonic	Function	R/W	Reset
63–0	LSTAR	Long Mode Target Address	R/W	

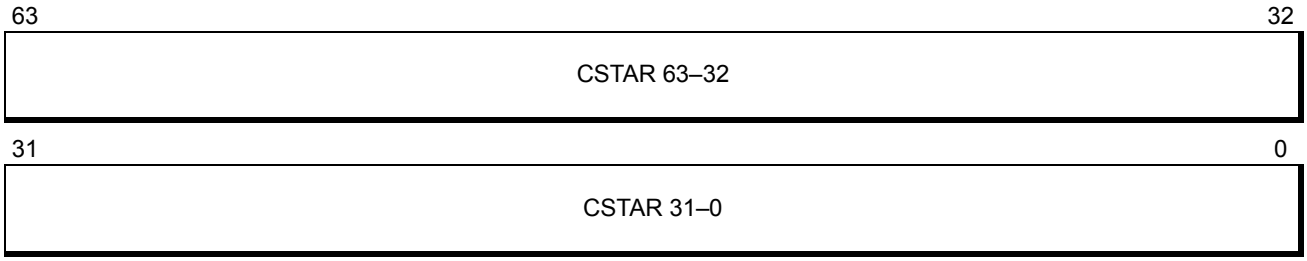
Field Descriptions

Long Mode Target Address (LSTAR)—Bits 63–0. Target address for 64-bit mode calling programs.

14.2.7.3 CSTAR Register

The address stored in this register must be in canonical form (if not canonical, a #GP fault will occur).

CSTAR Register — MSR C000_0083h



Bit	Mnemonic	Function	R/W	Reset
63–0	CSTAR	Compatibility mode target address	R/W	

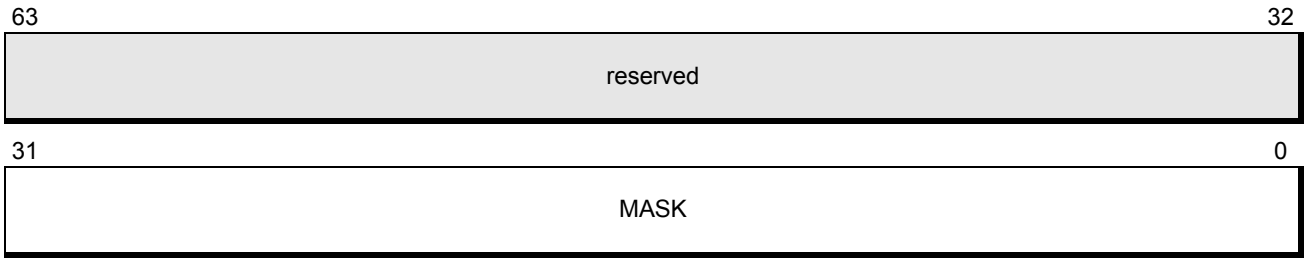
Field Descriptions

Compatibility Mode Target Address (CSTAR)—Bits 63–0. Target address for compatibility mode calling programs.

14.2.7.4 SF_MASK Register

This register holds the EFLAGS mask used by the SYSCALL instruction. Each bit in this mask clears the corresponding EFLAGS bit when executing the SYSCALL instruction.

SF_MASK Register — MSR C000_0084h



Bit	Mnemonic	Function	R/W	Reset
63–32	reserved	RAZ		
31–0	MASK	SYSCALL Flag Mask	R/W	

Field Descriptions

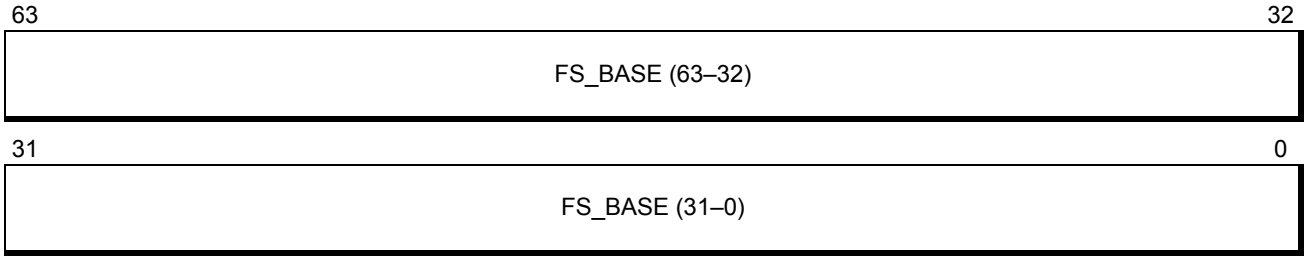
SYSCALL Flag Mask (MASK)—Bits 31–0.

14.2.8 Segmentation Registers

14.2.8.1 FS.Base Register

This register provides access to the expanded 64 bit FS segment base. The address stored in this register must be in canonical form (if not canonical, a #GP fault fill occur).

FS.Base Register **MSR C000_0100h**



Bit	Mnemonic	Function	R/W	Reset
63–0	FS_BASE	Expanded FS segment base	R/W	

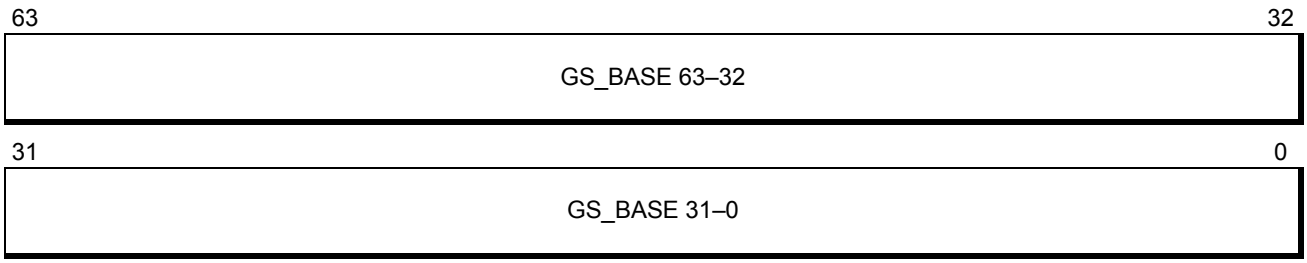
Field Descriptions

Expanded FS Segment Base (FS_BASE)—Bits 63–0.

14.2.8.2 GS.Base Register

This register provides access to the expanded 64 bit GS segment base. The address stored in this register must be in canonical form (if not canonical, a #GP fault fill occur).

GS.Base Register **MSR C000_0101h**



Bit	Mnemonic	Function	R/W	Reset
63–0	GS_BASE	Expanded GS segment base	R/W	

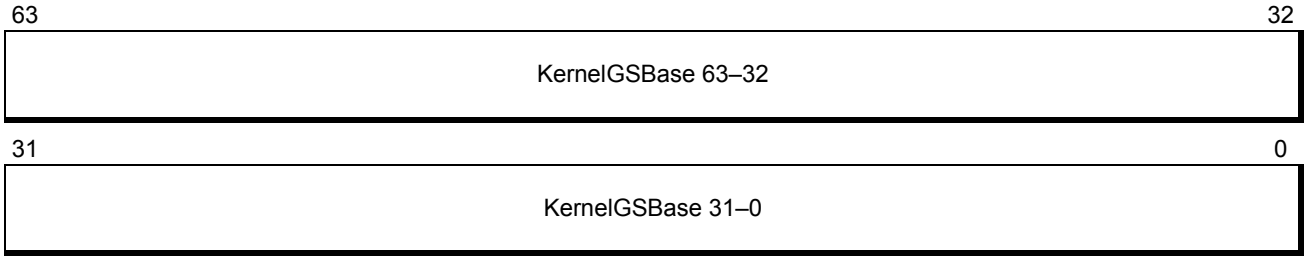
Field Descriptions

Expanded GS Segment Base (GS_BASE)—Bits 63–0.

14.2.8.3 KernelGSbase Register

This register holds the kernel data structure pointer which can be swapped with the GS_BASE register using the new 64-bit mode instruction SwapGS. See the "SWAPGS Instruction" section in Volume 2 of the *AMD64 Architecture Programmer's Manual* for more information. The address stored in this register must be in canonical form (if not canonical, a #GP fault occurs).

KernelGSbase Register **MSR C000_0102h**



Bit	Mnemonic	Function	R/W	Reset
63–0	KernelGSBase	Kernel data structure pointer	R/W	

Field Descriptions

Kernel Data Structure Pointer (KernelGSBase)—Bits 63–0.

14.2.9 Power Management Registers

Mobile processors contain AMD PowerNow!™ technology hardware that allows the processor operating voltage and frequency to be dynamically controlled for power management purposes. Use CPUID function 8000_0007h (Get Advanced Power Management Feature Flags) to determine the thermal and power management capabilities of the processor. Refer to "Processor Performance States" on page 299 regarding the use of the FIDVID_STATUS and FIDVID_CTL MSRs.

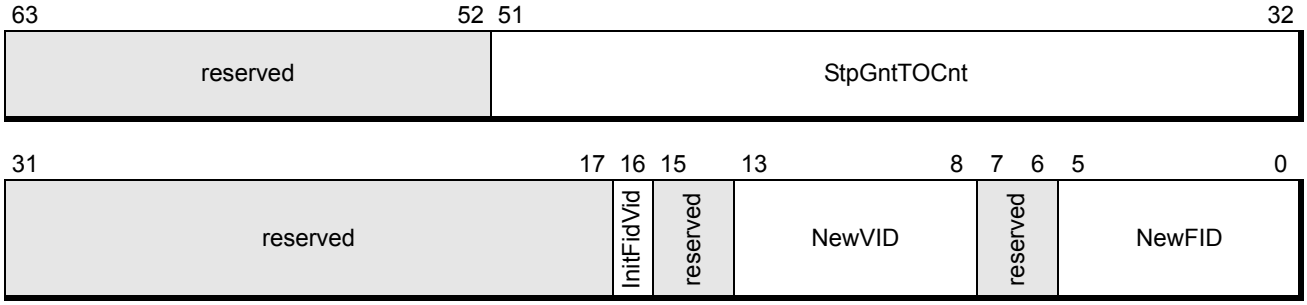
14.2.9.1 FIDVID_CTL Register

This register holds configuration information relating to power management control. Accessing this register is allowed if the device supports either FID control or VID control as indicated by CPUID. If neither FID control nor VID control is indicated, accessing the FIDVID_CTL register causes a GP# fault.

This register is used to specify new Frequency ID (FID) and/or Voltage ID (VID) codes to which to change on the next FID/VID change. This register is also used to control how long to wait following a FID/VID change before the PLL and voltage regulator are stable at the new FID/VID. This register is persistent through a warm reset.

The processor core clock PLL lock time is 2 µs for AMD NPT Family 0Fh Processors. This time refers to all components of PLL lock including frequency lock, phase lock, and settling time.

FIDVID_CTL Register — MSR C001_0041h



Bit	Mnemonic	Function	R/W	Reset
63–52	reserved	MBZ		0
51–32	StpGntTOCnt	Stop Grant Time-Out Count	R/W	0
31–17	reserved	MBZ		0
16	InitFidVid	Initiate FID/VID Change	W	0
15–14	reserved	MBZ		0
13–8	NewVID	New VID	R/W	0
7–6	reserved	MBZ	R	0
5–0	NewFID	New FID	R/W	0

Field Descriptions

Stop Grant Time-Out Count (StpGntTOCnt)—Bits 51-32. This field carries a count of system clock cycles (5 ns) that must elapse from the time a new FID is applied until the time that the PLL is stable at the new FID. The BIOS or processor driver must program this field with the PLL lock time, defined above, prior to a FID change.

Initiate FID/VID Change (InitFidVid)—Bits 16. Writing this bit to a 1 initiates a FID/VID change. In a multiprocessor system, only the bootstrap processor sees this bit written as a 1. Writing this bit to a 1 initiates the FID change special bus cycle. This is a write-only bit and always reads as 0. The status of the resulting FID/VID change can be determined from the FIDVID_STATUS register.

New VID (NewVID)—Bits 13–8. This field is the new VID to transition to. If an attempt is made to write a NewVID value that corresponds to a voltage greater than the voltage that MaxVID corresponds to in the FIDVID_STATUS register then the MaxVID value is written instead.See Table 74 on page 321 for VID values and their corresponding voltages. See "Register Differences in Revisions of AMD NPT Family 0Fh Processors" on page 27 for revision information about this field.

New FID (NewFID)—Bits 5–0. This field is the new FID to transition to. If an attempt is made to write a NewFID value greater than MaxFID in the FIDVID_STATUS register then the MaxFID value is written instead. See Table 83 for FID code translations.

Table 83. FID Code Translations

Reference Clock Multiplier	6-Bit FID Code	Reference Clock Multiplier	6-Bit FID Code
4x	00_0000b	15x	01_0110b
4.5x	00_0001b	15.5x	01_0111b
5x	00_0010b	16x	01_1000b
5.5x	00_0011b	16.5x	01_1001b
6x	00_0100b	17x	01_1010b
6.5x	00_0101b	17.5x	01_1011b
7x	00_0110b	18x	01_1100b
7.5x	00_0111b	18.5x	01_1101b
8x	00_1000b	19X	01_1110b
8.5x	00_1001b	19.5X	01_1111b
9x	00_1010b	20X	10_0000b
9.5x	00_1011b	20.5X	10_0001b
10x	00_1100b	21X	10_0010b
10.5x	00_1101b	21.5X	10_0011b
11x	00_1110b	22x	10_0100b
11.5x	00_1111b	22.5x	10_0101b
12x	01_0000b	23x	10_0110b
12.5x	01_0001b	23.5x	10_0111b
13x	01_0010b	24x	10_1000b
13.5x	01_0011b	24.5x	10_1001b
14x	01_0100b	25x	10_1010b
14.5x	01_0101b		

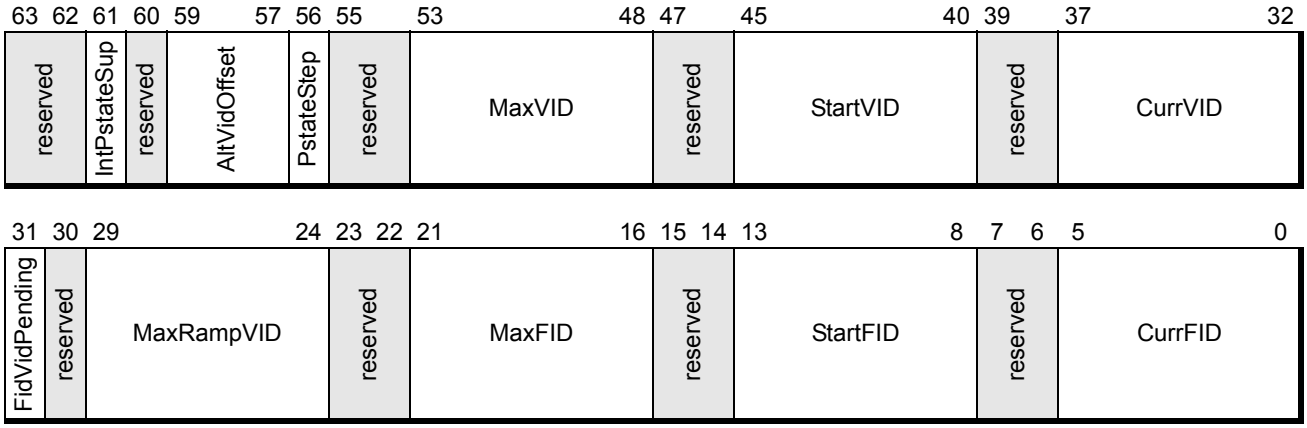
Note: *Encodings not listed are reserved. Using reserved encodings causes unpredictable behavior.*

14.2.9.2 FIDVID_STATUS Register

This register holds status of the current FID, the current VID, pending changes, and maximum values. Accessing this register is allowed if the device supports either FID control or VID control as indicated via CPUID. If neither FID control nor VID control is indicated, accessing the FIDVID_STATUS register causes a GP# fault.

See Table 83 on page 425 for a complete list of FID values and their corresponding reference clock multiplier values. See Table 74 on page 321 for VID values and their corresponding voltages.

FIDVID_STATUS Register — MSR C001_0042h



Bit	Mnemonic	Function	R/W	Reset
63–62	reserved	RAZ	R	0
61	IntPstateSup	Intermediate P-state support	R	X
60	reserved		R	X
59–57	AltVidOffset	AltVID Offset	R	X
56	PstateStep	P-state Voltage Step	R	X
55–54	reserved	RAZ	R	0
53–48	MaxVID	Max VID	R	0
47–46	reserved	RAZ	R	0
45–40	StartVID	Startup VID	R	0
39–38	reserved	RAZ	R	0
37–32	CurrVID	Current VID	R	0
31	FidVidPending	FID/VID Change Pending	R	0
30	reserved	RAZ	R	0
29–24	MaxRampVID	Max Ramp VID	R	0
23–22	reserved	RAZ	R	0
21–16	MaxFID	Max FID	R	0
15–14	reserved	RAZ	R	0
13–8	StartFID	Startup FID	R	0
7–6	reserved	RAZ	R	0
5–0	CurrFID	Current FID	R	0

Field Descriptions

Intermediate P-State Support (IntPstateSup)—Bit 61. This bit indicates if intermediate P-states are supported. See "Register Differences in Revisions of AMD NPT Family 0Fh Processors" on page 27 for revision information about this field.

0b = Intermediate P-states not supported.
1b = Intermediate P-states supported.

AltVID Offset (AltVidOffset)—Bits 59–57. This field specifies the amount of voltage subtracted from the start VID to determine the AltVID voltage. See "Register Differences in Revisions of AMD NPT Family 0Fh Processors" on page 27 for revision information about this field.

000b = AltVID not supported.
001b = -50mV.
010b = -100mV.
011b = -125mV.
100b = -150mV.
101b = -175mV.
110b = -200mV.
111b = -225mV.

P-State Voltage Step Size (PstateStep)—Bit 56. This bit indicates the voltage reduction associate with each 200MHz reduction in operating frequency. See "Register Differences in Revisions of AMD NPT Family 0Fh Processors" on page 27 for revision information about this field.

0b = 25mV reduction.
1b = 50mV reduction.

Max VID (MaxVID)—Bits 53–48. This field indicates the VID code associated with the maximum voltage software can select. See "Register Differences in Revisions of AMD NPT Family 0Fh Processors" on page 27 for revision information about this field.

Startup VID (StartVID)—Bits 45–40. This field indicates the startup VID. See "Register Differences in Revisions of AMD NPT Family 0Fh Processors" on page 27 for revision information about this field.

Current VID (CurrVID)—Bits 37–32. This field indicates the current VID. See "Register Differences in Revisions of AMD NPT Family 0Fh Processors" on page 27 for revision information about this field.

FID/VID Change Pending (FidVidPending)—Bit 31. This bit indicates that a FID/VID change is pending. It is set to 1 by hardware when the InitFidVid bit is set in the FIDVID_CTL register. It is cleared by hardware when the FID/VID change has completed.

Max Ramp VID (MaxRampVID)—Bit 29–24. This field indicates the max Ramp VID. See "Register Differences in Revisions of AMD NPT Family 0Fh Processors" on page 27 for revision information about this field.

Max FID (MaxFID)—Bit 21–16. This field indicates the max FID.

Startup FID (StartFID)—Bit 13–8. This field indicates the startup FID.

Current FID (CurrFID)—Bit 5–0. This field indicates the current FID.

14.2.10 I/O and Configuration Space Trapping to SMI

I/O and configuration space trapping to SMI is a mechanism for executing SMI handler if a specific I/O access to one of the specified addresses is detected. Access address and access type checking is done before I/O instruction execution. If the access address and access type match one of the specified I/O address and access types, the SMI handler is executed, the I/O instruction is not executed, and a breakpoint set on the I/O instruction is not taken. The I/O instruction can be executed inside the SMI handler. This mechanism has higher priority than the SMI interrupt. If the trap is not taken, check for SMI interrupt is done after the I/O instruction has been executed.

Configuration accesses are special I/O accesses. An I/O access is defined as a configuration access when I/O instruction address bits 31-0 are CFCh, CFDh, CFEh, or CFFh. The access address for a configuration space access is the current value of doubleword I/O register CF8h. The access address for an I/O access that is not a configuration access is equivalent to the I/O instruction address, bits 31–0.

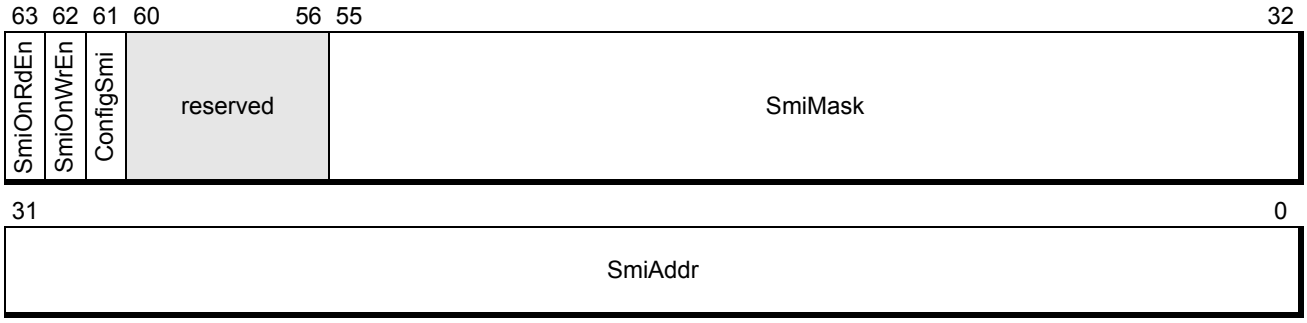
The access address is compared with SmiAddr, and the instruction access type is compared with the enabled access types defined by ConfigSMI, SmiOnRdEn, and SmiOnWrEn. Access address bits 23-0 can be masked with SmiMask. Fields SmiAddr, SmiMask, ConfigSMI, SmiOnRdEn, and SmiOnWrEn are defined in IOTRAP_ADDR*i* registers (i = 0,1,2,3).

An SMI handler executed as a result of I/O and configuration space trapping is by default executed only by the processor core executing the I/O instruction. The user has the ability to enable an SMI special bus cycle and RSM special bus cycle generation if SMI special bus cycles are enabled globally in the HWCR register.

I/O and configuration space trapping to SMI applies only to single I/O instructions; it does not apply to string and REP I/O instructions.

14.2.10.1 IOTRAP_ADDR*i* Registers

IOTRAP_ADDR*i* Registers **MSRs C001_0050h, C001_0051h, C001_0052h, C001_0053h**



Bit	Mnemonic	Function	R/W	Reset
63	SmiOnRdEn	Enable SMI on IO Read	R/W	0
62	SmiOnWrEn	Enable SMI on IO Write	R/W	0
61	ConfigSmi	Configuration Space SMI	R/W	0
60-56	reserved	RAZ	R	
55–32	SmiMask	SMI Mask	R/W	0
31–0	SmiAddr	SMI Address	R/W	0

Field Descriptions

Enable SMI on IO Read (SmiOnRdEn)—Bits 63. Enables SMI generation on a read access.

Enable SMI on IO Write (SmiOnWrEn)—Bit 62. Enables SMI generation on a write access.

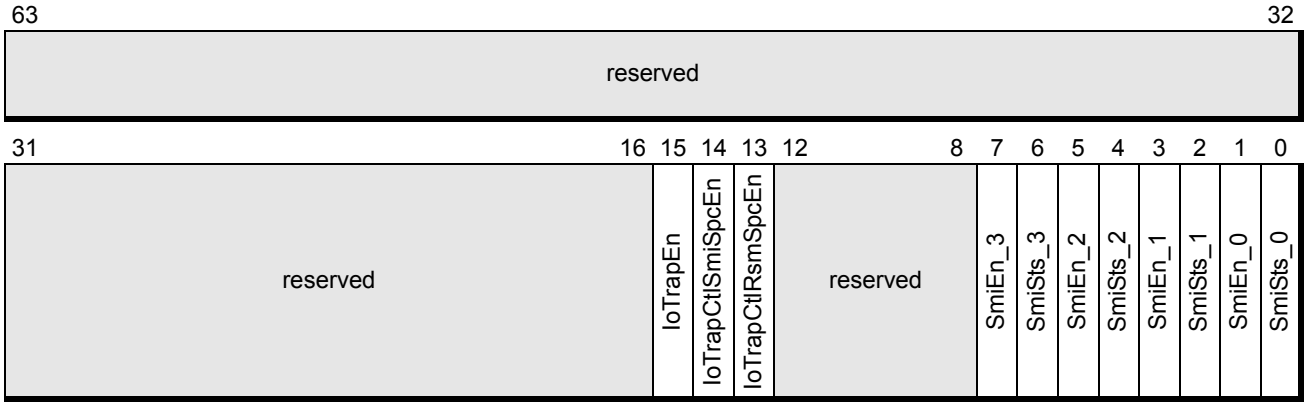
Configuration Space SMI (ConfigSmi)—Bit 61. 1 = configuration access (see "I/O and Configuration Space Trapping to SMI" on page 428 for more information on configuration access detection); 0 = I/O access different than configuration access.

SMI Mask (SmiMask)—Bits 55–32. SMI I/O trap mask. 0 = mask address bit; 1 = do not mask address bit. When the ConfigSmi bit is set bits 33 and 32 are ignored.

SMI Address (SmiAddr)—Bits 31–0. SMI I/O trap address. When the ConfigSmi bit set is the SMI address is internally doubleword aligned and bits 0 and 1 of SmiAddr are ignored.

14.2.10.2 IOTRAP_CTL Register

IOTRAP_CTL Register — MSR C001_0054h



Bit	Mnemonic	Function	R/W	Reset
63–32	reserved	RAZ	R	
31–16	reserved	MBZ	R	
15	IoTrapEn	IO Trap Enable	R/W	0
14	IoTrapCtlSmiSpcEn	IO Trap Control SMI Special Cycle Enable	R/W	0

Bit	Mnemonic	Function	R/W	Reset
13	IoTrapCtlRsmSpcEn	IO Trap Control RSM Special Cycle Enable	R/W	0
12-8	reserved	MBZ	R	
7	SmiEn_3	SMI Enable 3	R/W	0
6	SmiSts_3	SMI Status 3	R/W	0
5	SmiEn_2	SMI Enable 2	R/W	0
4	SmiSts_2	SMI Status 2	R/W	0
3	SmiEn_1	SMI Enable 1	R/W	0
2	SmiSts_1	SMI Status 1	R/W	0
1	SmiEn_0	SMI Enable 0	R/W	0
0	SmiSts_0	SMI Status 0	R/W	0

Field Descriptions

IO Trap Enable (IoTrapEn)—Bit 15. Enable I/O and configuration space trapping.

IO Trap Control SMI Special Cycle Enable (IoTrapCtlSmiSpcEn)—Bit 14. Enable SMI special bus cycle generation when SMI handler is entered as a result of I/O and configuration space trapping. If SMISPCYCDIS (MSR C001_0015h, bit 13) is set then this bit has no effect.

IO Trap Control RSM Special Cycle Enable (IoTrapCtlRsmSpcEn)—Bit 13. Enable RSM special bus cycle generation when SMI handler is entered as a result of I/O and configuration space trapping. If RSMSPCYCDIS (MSR C001_0015h, bit 14) is set, then this bit has no effect.

SMI Enable 3 (SmiEn_3)—Bit 7. Enable SMI generation if I/O access matches access specified in MSR C001_0053h.

SMI Status 3 (SmiSts_3)—Bit 6. Set if I/O access matched access specified in MSR C001_0053h.

SMI Enable 2 (SmiEn_2)—Bit 5. Enable SMI generation if I/O access matches access specified in MSR C001_0052h.

SMI Status 2 (SmiSts_2)—Bit 4. Set if I/O access matched access specified in MSR C001_0052h.

SMI Enable 1 (SmiEn_1)—Bit 3. Enable SMI generation if I/O access matches access specified in MSR C001_0051h.

SMI Status 1 (SmiSts_1)—Bit 2. Set if I/O access matched access specified in MSR C001_0051h.

SMI Enable 0 (SmiEn_0)—Bit 1. Enable SMI generation if I/O access matches access specified in MSR C001_0050h.

SMI Status 0 (SmiSts_0)—Bit 0. Set if I/O access matched access specified in MSR C001_0050h.

14.2.11 Interrupt Pending Message Register

This register is used to ensure that the system I/O Hub can wake the processor out of the stop grant state when there is a pending interrupt. If the interrupt pending message is not specified by this register, it is possible for the system I/O Hub to place the processor into the stop grant state while an interrupt is pending in the processor. This register must be setup for platforms that support the C2 or C3 power management states.

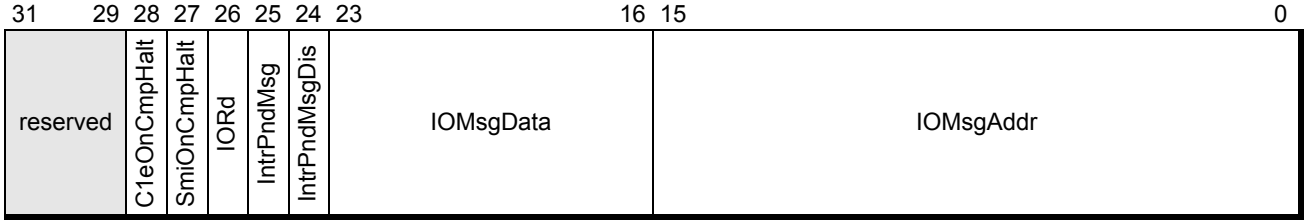
This register enables the processor to send a message to the I/O hub that results in the pending interrupt being serviced.The two message types are the I/O space message and the HyperTransport INT_PENDING message. The HyperTransport™ Technology INT_PENDING message is defined by the HyperTransport 1.05c specification.

If the processor or the I/O hub does not support the INT_PENDING HyperTransport Technology message, the I/O space message should be selected by IntPndMsg. The I/O space message should be chosen to avoid interference with the OSPM SMI command port value. If BIOS configures a write to the SMI command port for the interrupt pending message, then it is recommended that BIOS also disable all external asyncrhonous sources of STPCLK (STPCLK throttling, e.g.). A check for a pending interrupt is performed at the end of an I/O instruction. If there is a pending interrupt and STPCLK is asserted, the processor executes a byte-size read or write to I/O space defined with IORd, IOMsgAddr, and IOMsgData (used only for I/O writes) to generate an SMI. The SMI wakes up the processor so the original pending interrupt can be serviced. The SMI handler should not take any action if the SMI is generated by this mechanism. In order to prevent SMI generation with this mechanism in the SMI handler, IntrPndMsgDis bit should be set in the SMI handler before the first I/O instruction is executed, and it should be cleared prior to resuming from SMM.

If the processor and the I/O hub support the INT_PENDING HyperTransport Technology message, it should be selected by IntPndMsg. The check for a pending interrupt is performed when entering the stop grant state, if STPCLK is not initiated by Hardware Thermal Control (HTC) or Software Thermal Control (STC). If there is a pending interrupt, the processor sends an INT_PENDING HyperTransport message to the I/O hub.

The other purpose for this register is to generate messages to the I/O Hub that result in SMI interrupts when all the CPU cores in the processor are in the halt state. This MSR is not shared between CPU cores and must be programmed the same on both cores. If SmiOnCmpHalt is set, then whenever a CPU core enters the halt state while all other CPU cores of the processor are also in halt, then the processor generates an I/O cycle based on the programmed values in IORd, IOMsgData and IOMsgAddr. The expectation is that this I/O cycle targets an I/O Hub register that triggers an SMI back to the processor; the SMI handler may then be used to bring the processor into the C1 enhanced mode (C1E), a low-power halt mode.

Interrupt Pending Message Register **MSR C001_0055h**



Bit	Mnemonic	Function	R/W	Reset
63–29	reserved	RAZ	R	
28	C1eOnCmpHalt	C1E on Multi-core halt	R/W	0
27	SmiOnCmpHalt	SMI on Multi-core halt	R/W	0
26	IORd	IO Read	R/W	0
25	IntrPndMsg	Interrupt Pending Message	R/W	0
24	IntrPndMsgDis	Interrupt Pending Message Disable	R/W	0
23–16	IOMsgData	IO Message Data	R/W	0
15–0	IOMsgAddr	IO Message Address	R/W	0

Field Descriptions

C1E on Multi-core Halt (C1eOnCmpHalt)—Bit 28. 1=When all CPU cores of the processor have entered the halt state, the processor generates an I/O cycle as specified by IORd, IOMsgData, and IOMsgAddr. When this bit is set, the I/O cycle specified should be a read to the ACPI defined P_LVL3 register. When this bit is set SmiOnCmpHalt and IntPndMsg must be 0. This bit is not supported on single core processors.

SMI on Multi-core Halt (SmiOnCmpHalt)—Bit 27. 1=When all CPU cores of the processor have entered the halt state, the processor generates an I/O cycle as specified by IORd, IOMsgData, and IOMsgAddr. When this bit is set, the I/O cycle specified should be a write to the system SMI command port. When this bit is set C1eOnCmpHalt and IntPndMsg must be 0. This bit is not supported on single core processors.

IO Read (IORd)—Bit 26. 1 = I/O read; 0 = I/O write.

Interrupt Pending Message (IntrPndMsg)—Bit 25. Select an interrupt pending message type. 0 = HyperTransport INT_PENDING message; 1 = I/O space message. This bit should be set after IORd, IOMsgAddr and IOMsgData fields are programmed.

Interrupt Pending Message Disable (IntrPndMsgDis)—Bit 24. Disable generating an interrupt pending message defined with IntrPndMsg.

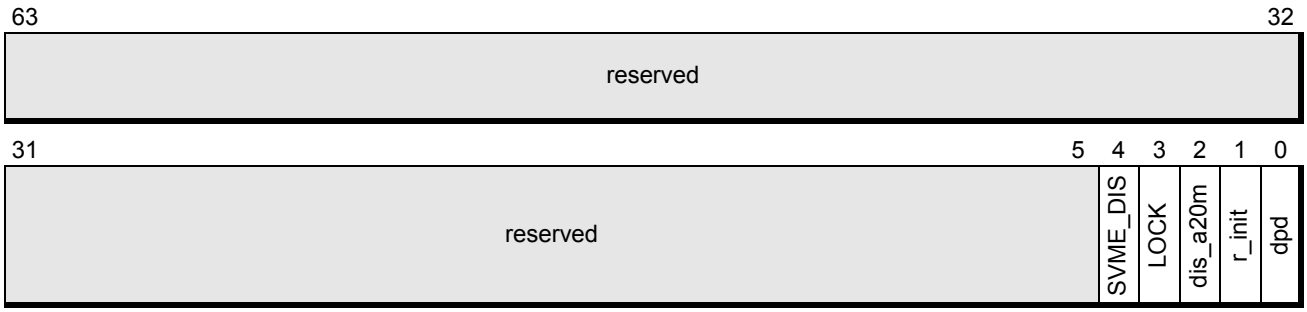
IO Message Data (IOMsgData)—Bits 23–16. I/O space message data. This field is only used when an I/O write is selected (IORd=0).

IO Message Address (IOMsgAddr)—Bits 15–0. I/O space message address.

14.2.12 VM_CR Register

This register provides security related controls. See "Register Differences in Revisions of AMD NPT Family 0Fh Processors" on page 27 for revision information about this register.

VM_CR Register **MSR C001_0114h**



Bit	Mnemonic	Function	R/W	Reset
63–5	reserved		R	0
4	SVME_DISABLE	Disable SVME	R/W	
3	LOCK	Lock SVM	R	
2	dis_a20m	Disable A20 Masking	R/W	0
1	r_init	Intercept INIT	R/W	0
0	dpd	Debug Port Disable	R/W	0

Field Descriptions

Debug Port Disable (dpd)—Bit 0. This bit controls if debug facilities such as JTAG and HDT have access to the processor state information. When this bit is set, all mechanism that could expose trusted code execution are disabled.

0 = Debug port enabled.
1 = Debug port disabled.

Intercept INIT (r_init)—Bit 1. This bit controls how INIT is delivered in host mode.

0 = INIT delivered normally.
1 = INIT translated into a SX interrupt.

Disable A20 Masking (dis_a20m)—Bit 2. When set, this bit disables A20 masking.

Lock SVM (LOCK)—Bit 3. When this bit is set writes to SVME_DISABLE (bit 4) are ignored.

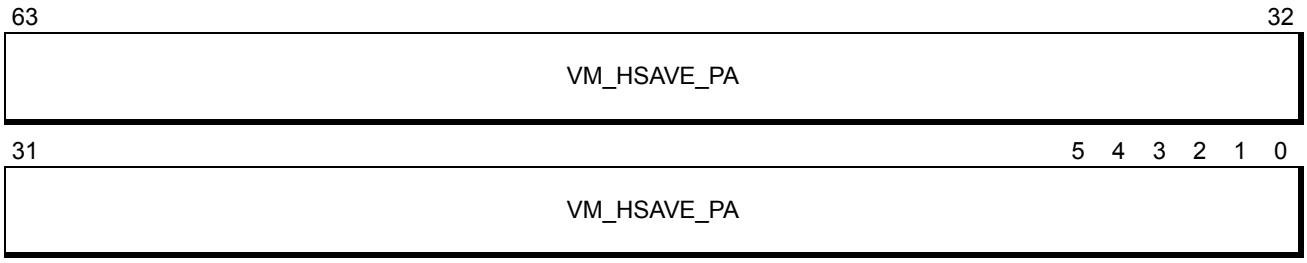
Disable SVME (SVME_DISABLE)—Bit 4. This bit determines if the EFER.SVME bit is writeable by hypervisors. If LOCK (bit 3) is set then this bit is read-only.

0 = SVM is allowed to be enabled using EFER.SVME.
1 = SVM is not allowed. The user must change a BIOS setting to enable SVM.

14.2.13 VM_HSAVE_PA Register

This register contains the physical address where VMRUN saves host state and where vm-exit restores host state from. Writing this register causes a GP exception if any of the lower 12 bits are not zero or if the address written is greater than FD_000_000h. See "Register Differences in Revisions of AMD NPT Family 0Fh Processors" on page 27 for revision information about this register.

VM_HSAVE_PA Register **MSR C001_0117h**



Bit	Mnemonic	Function	R/W	Reset
63–0	VM_HSAVE_PA	Physical Address of Host Save Area	R/w	0

Glossary

128-bit media instructions. Instructions that use the 128-bit XMM registers. These are a combination of SSE and SSE2 instruction sets.

64-bit media instructions. Instructions that use the 64-bit MMX® registers. These are primarily a combination of MMX® and 3DNow!™ instruction sets, with some additional instructions from the SSE and SSE2 instruction sets.

16-bit mode. Legacy mode or compatibility mode in which a 16-bit address size is active. See *legacy mode* and *compatibility mode*.

32-bit mode. Legacy mode or compatibility mode in which a 32-bit address size is active. See *legacy mode* and *compatibility mode*.

64-bit mode. A submode of *long mode*. In 64-bit mode, the default address size is 64 bits and new features, such as register extensions, are supported for system and application software.

absolute. Said of a displacement that references the base of a code segment rather than an instruction pointer. Contrast with *relative*.

AH–DH. The high 8-bit AH, BH, CH, and DH registers. Compare *AL–DL*.

AL–DL. The low 8-bit AL, BL, CL, and DL registers. Compare *AH–DH*.

AL–r15B. The low 8-bit AL, BL, CL, DL, SIL, DIL, BPL, SPL, and R8B–R15B registers, available in 64-bit mode.

biased exponent. The sum of a floating-point value's exponent and a constant bias for a particular floating-point data type. The bias makes the range of the biased exponent always positive, which allows reciprocation without overflow.

byte. Eight bits.

clear. To write a bit-value of 0. Compare *set*.

compatibility mode. A submode of *long mode*. In compatibility mode, the default address size is 32 bits and legacy 16-bit and 32-bit applications run without modification.

b. A bit, as in *1 Mb* for one megabit, or *lsb* for least-significant bit.

B. A byte, as in *1 MB* for one megabyte, or *LSB* for least-significant byte.

canonical address. AMD NPT Family 0Fh Processors implement 48-bit virtual addresses. A virtual address having all 1s or all 0s in bit 63 through the most significant implemented bit (bit 47 for AMD NPT Family 0Fh Processors) is said to be in canonical form. Accessing a virtual address that is not in canonical form results in a fault.

commit. To irreversibly write, in program order, an instruction's result to software-visible storage, such as a register (including flags), the data cache, an internal write buffer, or memory.

CPL. Current privilege level.

CR*n*. Control register number *n*.

CS. Code segment register.

direct. Referencing a memory location whose address is included in the instruction's syntax as an immediate operand. The address may be an absolute or relative address. Compare *indirect*.

dirty data. Data held in the processor's caches or internal buffers that is more recent than the copy held in main memory.

displacement. A signed value that is added to the base of a segment (absolute addressing) or an instruction pointer (relative addressing). Same as *offset*.

doubleword. Two words, or four bytes, or 32 bits.

double quadword. Eight words, or 16 bytes, or 128 bits. Also called *octword*.

DP. Two processors or dual processors.

eAX–eSP. The 16-bit AX, BX, CX, DX, DI, SI, BP, SP registers or the 32-bit EAX, EBX, ECX, EDX, EDI, ESI, EBP, ESP registers. Compare *rAX–rSP*.

EFER. Extended features enable register.

effective address size. The address size for the current instruction after accounting for the default address size and any address-size override prefix.

effective operand size. The operand size for the current instruction after accounting for the default operand size and any operand-size override prefix.

eFLAGS. 16-bit or 32-bit flags register. Compare *rFLAGS*.

EFLAGS. 32-bit (extended) flags register.

eIP. 16-bit or 32-bit instruction-pointer register. Compare *rIP*.

EIP. 32-bit (extended) instruction-pointer register.

element. See *vector*.

exception. An abnormal condition that occurs as the result of executing an instruction. The processor's response to an exception depends on the type of the exception. For all exceptions except 128-bit media SIMD floating-point exceptions and x87 floating-point exceptions, control is transferred to the handler (or service routine) for that exception, as defined by the exception's vector. For floating-point exceptions defined by the IEEE 754 standard, there are both masked and unmasked responses. When unmasked, the exception handler is called, and when masked a default response is provided instead of calling the handler.

FLAGS. 16-bit flags register.

flush. When referring to caches, (1) to writeback, if modified, and invalidate, as in "flush the cache line"; when referring to the processor pipeline, (2) to invalidate, as in "flush the pipeline"; or (3) to change a value, as in "flush to zero".

GDT. Global descriptor table.

GDTR. Global descriptor table register.

GPRs. General-purpose registers. For the 16-bit data size, these are AX, BX, CX, DX, DI, SI, BP, SP. For the 32-bit data size, these are EAX, EBX, ECX, EDX, EDI, ESI, EBP, ESP. For the 64-bit data size, these include RAX, RBX, RCX, RDX, RDI, RSI, RBP, RSP, and R8–R15.

IDT. Interrupt descriptor table.

IDTR. Interrupt descriptor table register.

IGN. Ignore. Field is ignored.

indirect. Referencing a memory location whose address is in a register or other memory location. The address may be an absolute or relative address. Compare *direct*.

IP. 16-bit instruction-pointer register.

IRB. The virtual-8086 mode interrupt-redirection bitmap.

IST. The long-mode interrupt-stack table.

IVT. The real-address mode interrupt-vector table.

LDT. Local descriptor table.

LDTR. Local descriptor table register.

legacy x86. The legacy x86 architecture. See "Related Documents" on page 25 for descriptions of the legacy x86 architecture.

legacy mode. An operating mode of the AMD64 architecture in which existing 16-bit and 32-bit applications and operating systems run without modification. A processor implementation of the AMD64 architecture can run in either *long mode* or *legacy mode*. Legacy mode has three sub-modes, *real mode*, *protected mode*, and *virtual-8086 mode*.

long mode. An operating mode unique to the AMD64 architecture. A processor implementation of the AMD64 architecture can run in either *long mode* or *legacy mode*. Long mode has two submodes, *64-bit mode* and *compatibility mode*.

lsb. least-significant bit.

LSB. least-significant byte.

main memory. Physical memory, such as RAM and ROM (but not cache memory) that is installed in a particular computer system.

mask. (1) A control bit that prevents the occurrence of a floating-point exception from invoking an exception handling routine. (2) A field of bits used for a control purpose.

MBZ. Must be 0. If software attempts to set an MBZ bit to 1, a general-protection exception (#GP) occurs.

memory. Unless otherwise specified, *main memory.*

ModRM. A byte following an instruction opcode that specifies address calculation based on mode (mod), register (r), and memory (m) variables.

moffset. A direct memory offset. I.e., a displacement that is added to the base of a code segment (for absolute addressing) or to an instruction pointer (for addressing relative to the instruction pointer, as in RIP-relative addressing).

MP. More than two processors.

msb. Most-significant bit.

MSB. Most-significant byte.

MSR. Model-specific register.

multimedia instructions. A combination of *128-bit media instructions* and *64-bit media instructions*.

octword. Same as *double quadword.*

offset. Same as *displacement*.

overflow. The condition in which a floating-point number is larger in magnitude than the largest, finite, positive or negative number that can be represented in the data-type format being used.

packed. See *vector.*

PAE. Physical-address extensions.

physical memory. Actual memory, consisting of *main memory* and cache.

probe. A check for an address in a processor's caches or internal buffers. *External probes* originate outside the processor, and *internal probes* originate within the processor.

processor. This term refers to AMD Athlon™ 64 processor architecture, AMD Opteron™ processor architecture, AMD Sempron™ processor architecture and AMD Turion™ 64 Mobile Technology processor architecture. This document covers all classes of these devices. term is used for to refer to packages that contain one or more processor cores. For details about differences between them, see the relevant processor data sheets and functional data sheets.

protected mode. A submode of *legacy mode*.

quadword. Four words, or eight bytes, or 64 bits.

r8–r15. The 8-bit R8B–R15B registers, or the 16-bit R8W–R15W registers, or the 32-bit R8D–R15D registers, or the 64-bit R8–R15 registers.

rAX–rSP. The 16-bit AX, BX, CX, DX, DI, SI, BP, SP registers, or the 32-bit EAX, EBX, ECX, EDX, EDI, ESI, EBP, ESP registers, or the 64-bit RAX, RBX, RCX, RDX, RDI, RSI, RBP, RSP registers. The "r" variable should be replaced by nothing for 16-bit size, "E" for 32-bit size, or "R" for 64-bit size.

RAX. 64-bit version of EAX register.

RAZ. Read as zero (0), regardless of what is written.

RBP. 64-bit version of EBP register.

RBX. 64-bit version of EBX register.

RCX. 64-bit version of ECX register.

RDI. 64-bit version of EDI register.

RDX. 64-bit version of EDX register.

real-address mode. See *real mode*.

real mode. A short name for *real-address mode,* a submode of *legacy mode*.

relative. Referencing with a displacement (also called offset) from an instruction pointer rather than the base of a code segment. Contrast with *absolute*.

reserved. Fields marked as reserved may be used at some future time. In order to preserve compatibility with future processors, reserved fields require special handling when read or written by software.

Reserved fields may be further qualified as MBZ, RAZ, SBZ or IGN (see definitions).

Software must not depend on the state of a reserved field, nor upon the ability of such fields to return state previously written.

If a reserved field is not marked with one of the above qualifiers, software shall not change the state of that field; it must reload that field with the same values returned from a prior read.

REX. An instruction prefix that specifies a 64-bit operand size and provides access to additional registers.

rFLAGS. 16-bit, 32-bit, or 64-bit flags register. Compare *RFLAGS*.

RFLAGS. 64-bit flags register. Compare *rFLAGS*.

rIP. 16-bit, 32-bit, or 64-bit instruction-pointer register. Compare *RIP*.

RIP-Relative Addressing. 4-bit instruction-pointer register. Addressing relative to the 64-bit RIP instruction pointer. Compare *moffset*.

RSI. 64-bit version of ESI register.

RSP. 64-bit version of ESP register.

SBZ. Should be zero. If software attempts to set an SBZ bit to 1, it results in undefined behavior.

set. To write a bit-value of 1. Compare *clear*.

SIB. A byte following an instruction opcode that specifies address calculation based on scale (S), index (I), and base (B).

SIMD. Single instruction, multiple data. See *vector*.

SP. Stack pointer register.

SS. Stack segment register.

SSE. Streaming SIMD extensions instruction set. See *128-bit media instructions* and *64-bit media instructions*.

SSE2. Extensions to the SSE instruction set. See *128-bit media instructions* and *64-bit media instructions*.

sticky bit. A bit that is set or cleared by hardware and that remains in that state until explicitly changed by software.

TOP. x87 top-of-stack pointer.

TPR. Task-priority register (CR8), a new register introduced in the AMD64 architecture to speed interrupt management.

TR. Task register.

TSS. Task state segment.

underflow. The condition in which a floating-point number is smaller in magnitude than the smallest non-zero, positive or negative number that can be represented in the data-type format being used.

UP. One processor or uniprocessor.

vector. (1) A set of integer or floating-point values, called *elements*, that are packed into a single operand. Most of the 128-bit and 64-bit media instructions use vectors as operands. Vectors are also called *packed* or *SIMD* (single-instruction multiple-data) operands. (2) An index into an interrupt descriptor table (IDT), used to for accessing exception handlers. Compare *exception*.

virtual-8086 mode. A submode of *legacy mode*.

word. Two bytes, or 16 bits.

x86. See *legacy x86*.

Index of Register Names

B

C

D

G

H

I

L

M

N

P

R

S

T

U

X