AMD

Mobile AMD-K6® Processor BIOS Design Guide

Application Note

Publication # **23015**	Rev: **A**	Amendment/**0**
Issue Date: **May 2000**		

Contents

List of Figures

List of Tables

Revision History

Date	Rev	Description
May 2000	A	Initial release.

Application Note

Mobile AMD-K6® Processor BIOS Design Guide

Introduction

This document highlights the BIOS modifications required to fully support the Mobile AMD-K6® processors. This document is a supplement to the *AMD K86™ Family BIOS and Software Tools Developers Guide*, order# 21062 and the *AMD Processor Recognition Application Note*, order# 20734. Unless otherwise noted, the information in this application note pertains to all processors in the Mobile AMD-K6 family, which includes:

- Mobile AMD-K6 processor Model 7
- Mobile AMD-K6-2 and Mobile AMD-K6-2-P processors Model 8
- Mobile AMD-K6-III-P processor Model 9
- Mobile AMD-K6-2+ and Mobile AMD-K6-III+ processors Model D

There can be more than one way to implement the functionality detailed in this document, and the information provided is for demonstration purposes.

All referenced Mobile AMD-K6 processor documents can be found on the AMD website at http://www.amd.com/K6/k6docs.

Audience

It is assumed that the reader possesses the proper knowledge of the K86 processors, the x86 architecture, and the programming requirements to understand the information presented in this document.

Processor Models and Steppings

There are four models within the Mobile AMD-K6 family of processors—Models 7, 8, 9, and D. For most models, feature and function detection can be determined by reading the standard and extended feature bits by executing the CPUID instruction. However, for certain models, it is necessary to check the stepping—by executing the CPUID instruction—to determine specific function support. Table 1 shows the features of each model and stepping of the Mobile AMD-K6 processor family.

Table 1. Features of the Mobile AMD-K6® Processor Family

Processor	Model	Process (in microns)	Number of MSRs[1]	3DNow!™ Instructions	3DNow! Extensions	L2 Cache
Mobile AMD-K6®	7	0.25	6			
Mobile AMD-K6®-2 Mobile AMD-K6®-2-P	8/[F:8]	0.25	10[2]	Yes		
Mobile AMD-K6®-III-P	9/[3:0]	0.25	11[3]	Yes		256 Kbytes
Mobile AMD-K6®-2+	D/[7:4]	0.18	11[3]	Yes	Yes	128 Kbtyes
Mobile AMD-K6®-III+	D/[3:0]	0.18	11[3]	Yes	Yes	256 Kbytes

Notes:

1. *Refer to "Model-Specific Registers (MSRs)" on page 12 for more information.*
2. *Model 8/[F:8] defines the bits and fields in the WHCR and EFER MSRs differently from the Model 7.*
3. *This model implements the same ten MSRs as the Model 8/[F:8]. With the exception of bit 4 (L2D) in the EFER register, the bits and fields within these ten MSRs are defined identically.*

The following descriptions provide more detailed information on the Mobile AMD-K6 processor family members, and the models and steppings that comprise each member:

- **Mobile AMD-K6 processor**
  - **Model 7** is the first processor manufactured in the 0.25-micron process. The Model 7 supports six Model-Specific Registers (MSRs).
- **Mobile AMD-K6-2 and Mobile AMD-K6-2-P processors—** Some important features supported by this processor include the 3DNow!™ instruction set and support for a 100-MHz processor bus.
  - **Model 8/[F:8]** is any of eight possible model/steppings—Models 8/8, 8/9, 8/A, 8/B, 8/C, 8/D, 8/E, or 8/F. Model 8/[F:8] is manufactured in the 0.25-micron process. Model 8/[F:8] implements the same six MSRs as the Model 7, but the bits and fields within two of these MSRs—WHCR and EFER—are not defined identically. Also, Model 8/[F:8] supports the SYSCALL/SYSRET Target Address Register (STAR) MSR and three additional MSRs, for a total of ten MSRs.
- **Mobile AMD-K6-III**-P **processor—**In addition to supporting the 3DNow! instruction set and a 100-MHz processor bus, this processor contains a 256-Kbyte backside L2 cache.
  - Model 9/[3:0] is any of four possible model/steppings—Models 9/0, 9/1, 9/2, or 9/3. Model 9/[3:0] is manufactured in the 0.25-micron process. Model 9/[3:0] implements the same ten MSRs as the Model 8/[F:8]. With the exception of bit 4 (L2D) in the EFER register, the bits and fields within these ten MSRs are defined identically. Also, Model 9/[3:0] supports one additional MSR for a total of eleven MSRs.
- **Mobile AMD-K6-2+ processor—**In addition to supporting the 3DNow! instruction set and a 100-MHz processor bus, this processor contains a 128-Kbyte backside L2 cache.
  - **Model D/[7:4]** is any of four possible model/steppings—Models D/4, D/5, D/6, or D/7. Model D/[7:4] is manufactured in the 0.18-micron process. Model D/[7:4] supports the 3DNow! instruction extensions and implements the same eleven MSRs as the Model 9/[3:0]. Also, Model D/[7:4] supports the Enhanced Power Management Register (EPMR) MSR for a total of twelve MSRs.

- **Mobile AMD-K6-III+ processor—**In addition to supporting the 3DNow! instruction set and a 100-MHz processor bus, this processor contains a 256-Kbyte backside L2 cache.
  - **Model D/[3:0]** is any of four possible model/steppings—Models D/0, D/1, D/2, or D/3. Model D/[3:0] is manufactured in the 0.18-micron process. Model D/[3:0] supports the 3DNow! instruction extensions and implements the same eleven MSRs as the Model 9/[3:0]. Also, Model D/[3:0] supports the EPMR MSR for a total of twelve MSRs.

Table 7 on page 13 summarizes the MSR differences between the models and steppings of the AMD-K6 family of processors.

BIOS Consideration Checklist

CPUID

- Use the CPUID instruction to properly identify the processor. For information on the CPUID instruction, see "CPUID Instruction Overview" on page 47.
- Determine the processor model, stepping, and features using functions 0000_0001h and 8000_0001h of the CPUID instruction.
- Display BIOS boot strings:
  - *Mobile AMD-K6(tm)/XXX* for Model 7
  - *Mobile AMD-K6(tm)-2/XXX* for Model 8/[F:8]
  - *Mobile AMD-K6(tm)-III/XXX* for Model 9/[3:0]
  - *Mobile AMD-K6(tm)-2+/XXX* for Model D/[7:4]
  - *Mobile AMD-K6(tm)-III+/XXX* for Model D/[3:0]

  For more information, see "CPUID Identification Algorithms" on page 8.

CPU Speed Detection

- Use speed detection algorithms that do not rely on repetitive instruction sequences.
- 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 *CPU Speed Determination Program* available on the AMD website at http://www.amd.com/products/cpg/bin/.
- Display the recommended BIOS boot string as shown in Table 5, “Summary of Mobile AMD-K6® Processor Models and BIOS Boot String,” on page 9.

Model-Specific Registers (MSRs)

- Only access MSRs implemented in the processor.
- Enable Write Allocation by programming the Write Handling Control Register (WHCR). See "Write Handling Control Register (WHCR)" on page 15 and the *Implementation of Write Allocate in the K86™ Processors Application Note*, order# 21326 for more information.

Note: *The WHCR register as defined in Model 7 is implemented differently in the Models 8/[F:8], 9, and D.*

- For the Mobile AMD-K6-2 and Mobile AMD-K6-2-P Model 8/[F:8], Mobile AMD-K6-III-P, Mobile AMD-K6-2+, and Mobile AMD-K6-III+ processors, utilize the information provided in the Processor State Observability Register (PSOR) to display the correct processor bus frequency.

Cache Testing

- The Mobile AMD-K6 family of processors does not contain MSRs to allow for testing of the L1 cache. However, the Mobile AMD-K6-III-P, Mobile AMD-K6-2+ and Mobile AMD-K6-III+ processors do contain an MSR that allows testing of their L2 cache. This MSR is called L2AAR and it is described in "Level-2 Cache Array Access Register (L2AAR)" on page 34.

SMM Issues

- The System Management Mode (SMM) functionality of the processor is the same as the Pentium® processor.
- Implement the processor SMM state-save area in a similar manner as Pentium processors except for the IDT Base and possibly Pentium processor-reserved areas. See "System Management Mode (SMM)" on page 11 for more information.

Register States After RESET and INIT

After the processor has completed its initialization following the recognition of an asserted RESET or INIT signal, the states of all architecture registers and MSRs are compatible with those of Pentium processors. Differences are listed in Table 2 through Table 4.

Table 2. State of the Mobile AMD-K6® Processor Model 7 After RESET

Register	RESET State	Notes
EDX	0000_057Sh	1
EFER	0000_0000_0000_0000h	
STAR	0000_0000_0000_0000h	
WHCR	0000_0000_0000_0000h	

Note:
1. *"S" represents the Stepping.*

Table 3. State of the Mobile AMD-K6®-2 and Mobile AMD-K6-2-P Processors Model 8/[F:8] After RESET

Register	RESET State	Notes
EDX	0000_058Sh	1
EFER	0000_0000_0000_0002h	
PFIR	0000_0000_0000_0000h	
PSOR	0000_0000_0000_01SBh	1, 2
STAR	0000_0000_0000_0000h	
UWCCR	0000_0000_0000_0000h	
WHCR	0000_0000_0000_0000h	

Notes:
1. *"S" represents the Stepping.*
2. *"B" represents PSOR[3:0], where PSOR[3] equals 0, and PSOR[2:0] is equal to the value of the BF[2:0] signals sampled during the falling transition of RESET.*

Table 4. State of the Mobile AMD-K6®-III-P, Mobile AMD-K6®-2+ and Mobile AMD-K6®-III+ Processors After RESET

Register	RESET State	Notes
EDX	0000_05MSh	1
EFER	0000_0000_0000_0002h	2
L2AAR	0000_0000_0000_0000h	
PFIR	0000_0000_0000_0000h	
PSOR	0000_0000_0000_00SBh	1, 3
STAR	0000_0000_0000_0000h	
UWCCR	0000_0000_0000_0000h	
WHCR	0000_0000_0000_0000h	
EPMR	0000_0000_0000_0000h	4

Notes:

1. *"M" represents the Model and "S" represents the Stepping.*
2. *Because EFER[4] equals 0 after RESET, the L2 cache is enabled by default after RESET.*
3. *"B" represents PSOR[3:0], where PSOR[3] equals 0, and PSOR[2:0] is equal to the value of the BF[2:0] signals sampled during the falling transition of RESET.*
4. *Applies only to Mobile-AMD-K6-2+ and Mobile-AMD-K6-III+ Processors, Model D.*

State of the Processor After INIT

The assertion of INIT causes the processor to empty its pipelines, initialize most of its internal state, and branch to address FFFF_FFF0h—the same instruction execution starting point used after RESET. Unlike RESET, the processor preserves the contents of its caches, the floating-point state, the SMM base, MSRs, and the CD and NW bits of the CR0 register.

The edge-sensitive interrupts FLUSH# and SMI# are sampled and preserved during the INIT process and are handled accordingly after the initialization is complete. However, the processor resets any pending NMI interrupt upon sampling INIT asserted.

INIT can be used as an accelerator for 80286 code that requires a reset to exit from Protected mode back to Real mode.

CPUID Identification Algorithms

The CPUID instruction provides information about the processor (vendor, type, name, etc.) and its capabilities (features). After detecting the processor and its capabilities, software can be accurately tuned to the system for maximum performance and benefit to users. For example, game software can test the performance level available from a particular processor by detecting the type of processor. If the performance level is high enough, the software can enable additional capabilities or more advanced algorithms. Another example involves testing if the processor supports 3DNow! instructions. If the software finds this functionality present when it checks the feature bits, it can utilize these more powerful extensions for better performance on new multimedia software.

For more detailed information, see "CPUID Instruction Overview" on page 47.

Table 5 on page 9 shows the recommended BIOS boot strings for the Mobile AMD-K6 processor. The recommended boot strings are:

- *Mobile AMD-K6(tm)/XXX* for Model 7
- *Mobile AMD-K6(tm)-2/XXX* for Model 8/[F:8]
- *Mobile AMD-K6(tm)-III/XXX* for Model 9/[3:0]
- *Mobile AMD-K6(tm)-2+/XXX* for Model D/[7:4]
- *Mobile AMD-K6(tm)-III+/XXX* for Model D/[3:0]

The value for XXX is determined by calculating the core frequency of the processor. 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.

In addition to displaying the recommended boot string, BIOS code that normally indicates the presence of an L2 cache within the Summary/Configuration screen should change all occurrences of “L2” to “External.”

Note: *Table 5 contains information intended to prepare the infrastructure for potential future products. Although these products may not be announced, BIOS software should be prepared to support these options. The boot string for the Model D processors is determined by the size of the L2 cache.*

Table 5. Summary of Mobile AMD-K6® Processor Models and BIOS Boot String

Models	Steppings	CPU Speed (MHz)	CPU Bus Speed (MHz)	Recommended BIOS Boot-String Display
7	All	200	66	Mobile AMD-K6(tm)/200
		233	66	Mobile AMD-K6(tm)/233
		266	66	Mobile AMD-K6(tm)/266
		300	66	Mobile AMD-K6(tm)/300
8	F:8	266	66	Mobile AMD-K6(tm)-2/266
		300	66/100	Mobile AMD-K6(tm)-2/300
		333	66/95	Mobile AMD-K6(tm)-2/333
		350	100	Mobile AMD-K6(tm)-2/350
		366	66	Mobile AMD-K6(tm)-2/366
		380	95	Mobile AMD-K6(tm)-2/380
		400	66/100	Mobile AMD-K6(tm)-2/400
		433	96.2	Mobile AMD-K6(tm)-2/433
		450	100	Mobile AMD-K6(tm)-2/450
		475	95	Mobile AMD-K6(tm)-2/475
		500	100	Mobile AMD-K6(tm)-2/500
9	3:0	350	100	Mobile AMD-K6(tm)-III/350
		366	66	Mobile AMD-K6(tm)-III/400
		380	95	Mobile AMD-K6(tm)-III/400
		400	66/100	Mobile AMD-K6(tm)-III/400
		450	100	Mobile AMD-K6(tm)-III/450
D	7:4	450	100	Mobile AMD-K6(tm)-2+/450
		475	95	Mobile AMD-K6(tm)-2+/475
		500	100	Mobile AMD-K6(tm)-2+/500
	3:0	450	100	Mobile AMD-K6(tm)-III+/450
		475	95	Mobile AMD-K6(tm)-III+/475
		500	100	Mobile AMD-K6(tm)-III+/500

Figure 1 on page 10 shows a flow chart for the CPUID instruction. Use this chart to implement a CPUID algorithm. For more information on the CPUID instruction or processor recognition, see "Mobile AMD Processor Recognition" on page 47.

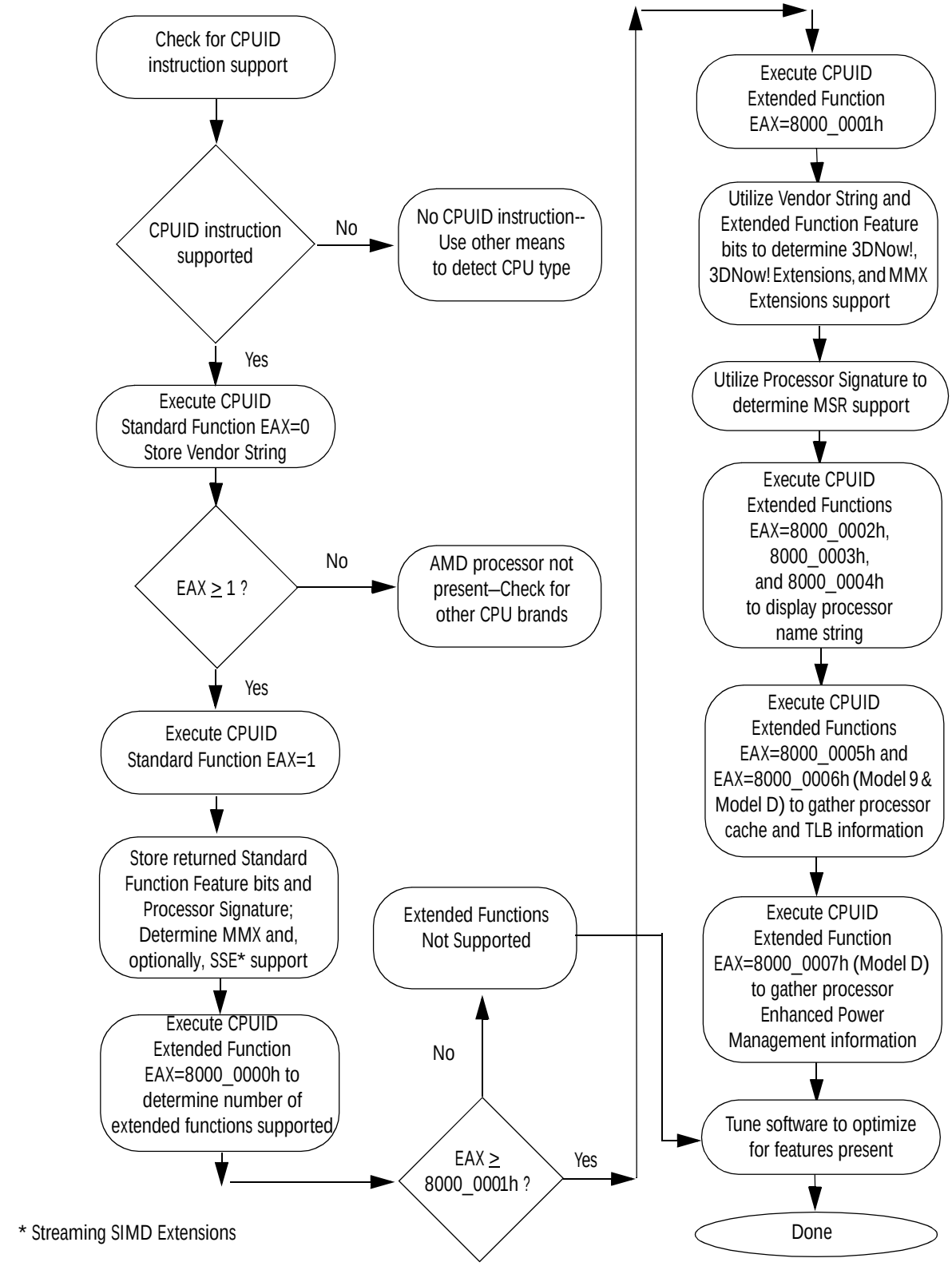


Figure 1. CPUID Instruction Flow Chart

Built-In Self-Test (BIST)

For all models of the Mobile AMD-K6 processor, BIST is run unconditionally following the falling transition of RESET. The results of the test are contained in the general purpose register EAX. If EAX contains 0000_0000h, then BIST was successful. If the contents of EAX are non-zero, the BIST failed. The internal resources tested during BIST include the following:

- L1 instruction and data caches
- L2 unified cache (Model 9 and Model D only)
- Instruction and Data Translation Lookaside Buffers (TLBs)

System Management Mode (SMM)

This section documents the SMM differences between specified models of the Mobile AMD-K6 processor and the Pentium processor. For more information on SMM implementation in the K86 processors, see the *AMD K86™ Family BIOS and Software Tools Developers Guide*, order# 21062.

State-Save Map Differences

The SMM implemented in the Mobile AMD-K6 processor differs from the SMM implemented in the Pentium processor in one way. The IDT Base location in the Mobile AMD-K6 processor is located at offset FF90h. Pentium has the IDT Base located at offset FF94h.

I/O Trap Dword Differences

The I/O trap dword is located at offset FFA4h. This state-save area, which is reserved in Pentium, contains information

regarding an I/O instruction that may have been trapped by an SMI# assertion.

Table 6. Mobile AMD-K6® Processor I/O Trap Dword Configuration at Offset FFA4h

Bits 31–16	Bits 15–4	Bit 3	Bit 2	Bit 1	Bit 0
I/O Port Address	Reserved	Rep String Operation	I/O String Operation	Valid I/O Instruction	Input or Output

Model-Specific Registers (MSRs)

All models and steppings of the Mobile AMD-K6 processor family support the following four standard MSRs, and the bits and fields within each of these MSRs are defined identically:

- Machine-Check Address Register (MCAR)—ECX = 00h
- Machine-Check Type Register (MCTR)—ECX = 01h
- Test Register 12 (TR12)—ECX = 0Eh
- Time Stamp Counter (TSC)—ECX = 10h

All models and steppings of the Mobile AMD-K6 processor family also support the following two MSRs, but the fields within each of these MSRs are defined differently as shown in Table 7 (an 'X' indicates support for a register or field):

- Extended Feature Enable Register (EFER)—ECX = C000_0080h
- Write Handling Control Register (WHCR)—ECX = C000_0082h

All models and steppings of the Mobile AMD-K6 processor family, with the exception of the Model 7, support the following MSRs:

- SYSCALL/SYSRET Target Address Register (STAR)—ECX = C000_0081h
- UC/WC Cacheability Control Register (UWCCR)—ECX = C000_0085h
- Processor State Observability Register (PSOR)—ECX = C000_0087h
- Page Flush/Invalidate Register (PFIR)—ECX = C000_0088h

Model D of the Mobile AMD-K6 processor family supports one additional MSR:

- Enhanced Power Management Register (EPMR)—ECX = C000_0086h

Table 7. Summary of MSR Differences Within the Mobile AMD-K6® Family

Model	Stepping	Standard MSRs[1]	EFER[2]				WHCR[3]		STAR	UWCCR	PSOR	PFIR	L2AAR	EPMR
			L2D	EWBEC	DPE	SCE	508 MB	4092 MB						
7	All	X				X	X							
8	F:8	X		X	X	X		X	X	X	X	X		
9	3:0	X	X	X	X	X		X	X	X	X	X	X	
D	All	X	X	X	X	X		X	X	X	X	X	X	X

Notes:

1. *There are four MSRs that every model and stepping of the Mobile AMD-K6 family of processors support identically–MCAR, MCTR, TR12, and TSC.*
2. *L2D, EWBEC, and DPE are bits/fields supported in EFER for the indicated models/steppings. All models/steppings support the System Call Extension (SCE) bit in EFER, even if the corresponding SYSCALL and SYSRET instructions and the STAR register are not supported.*
3. *Indicates whether the WAELIM field supports 508 Mbytes or 4092 Mbytes of memory. The location of the WAE15M bit and the WAELIM field within the WHCR register differs between the models/steppings that support 508 Mbytes of memory and those that support 4092 Mbytes of memory.*

Standard MSRs

This section describes the four standard MSRs that every model and stepping of the Mobile AMD-K6 family of processors support identically.

Machine-Check Address Register (MCAR) and Machine-Check Type Register (MCTR)

The processor does not support the generation of a machine check exception, but does provide a 64-bit Machine Check Address Register (MCAR) and a 64-bit Machine Check Type Register (MCTR) for software compatibility. Because the processor does not support machine check exceptions, the contents of the MCAR and MCTR are only affected by the WRMSR instruction and by RESET being sampled asserted (where all bits in each register are reset to 0).

The processor also provides the Machine Check Exception (MCE) bit in Control Register 4 (CR4, bit 6) as a read-write bit. However, the state of this bit has no effect on the operation of the processor.

Test Register 12 (TR12)

The processor provides the 64-bit Test Register 12 (TR12), but only the Cache Inhibit (CI) bit (bit 3 of TR12) is supported. All other bits in TR12 have no effect on the processor's operation. The I/O Trap Restart function (bit 9 of TR12) is always enabled on the Mobile AMD-K6.

Time Stamp Counter (TSC)

With each processor clock cycle, the processor increments a 64-bit time stamp counter (TSC) MSR. The counter can be written or read using the WRMSR or RDMSR instructions when the ECX register contains the value 10h and CPL = 0. The counter can also be read using the RDTSC instruction, but the required privilege level for this instruction is determined by the Time Stamp Disable (TSD) bit in CR4. With either of these instructions, the EDX and EAX registers hold the upper and lower dwords of the 64-bit value to be written to or read from the TSC, as follows:

- *EDX*—Upper 32 bits of TSC
- *EAX*—Lower 32 bits of TSC

The TSC can be loaded with any arbitrary value. This feature is compatible with the Pentium processor.

Mobile AMD-K6® Processor Model 7

The Mobile AMD-K6 processor Model 7 provides the following MSRs. The first four MSRs are described in "Standard MSRs" on page 13. The contents of ECX selects the MSR to be addressed by the RDMSR and WRMSR instructions.

- Machine-Check Address Register (MCAR)—ECX = 00h
- Machine-Check Type Register (MCTR)—ECX = 01h
- Test Register 12 (TR12)—ECX = 0Eh
- Time Stamp Counter (TSC)—ECX = 10h
- Extended Feature Enable Register (EFER)—ECX = C000_0080h
- Write Handling Control Register (WHCR)—ECX = C000_0082h

Extended Feature Enable Register (EFER)

The Extended Feature Enable Register (EFER) contains the control bits that enable the extended features of the Mobile AMD-K6 processor. Figure 2 shows the format of the EFER register, and Table 8 defines the function of each bit of the EFER register. The EFER register is MSR C000_0080h.

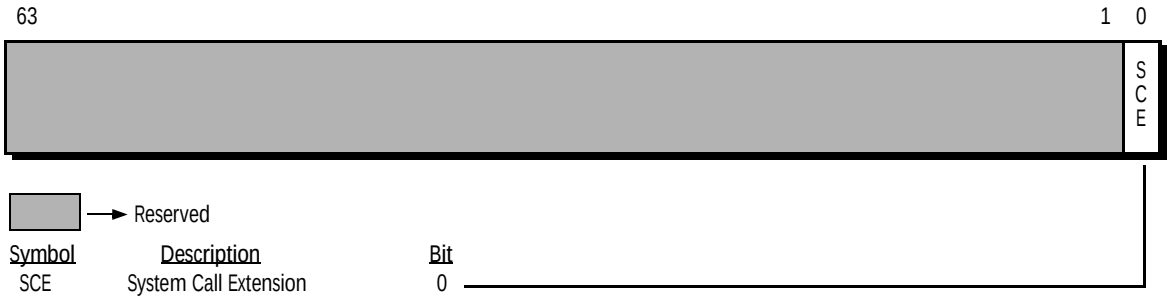


Figure 2. Extended Feature Enable Register (EFER)—MSR C000_0080h (Model 7)

Table 8. Extended Feature Enable Register (EFER) Definition (Model 7)

Bit	Description	R/W	Function
63–1	Reserved	R	Writing a 1 to any reserved bit causes a general protection fault to occur. All reserved bits are always read as 0.
0	System Call Extension (SCE)	R/W	SCE must be set to 1 to enable the usage of the SYSCALL and SYSRET instructions.

Note: *The Mobile AMD-K6 processor Model 7 provides the SCE bit in the EFER register, but this bit does not affect processor operation because the SYSCALL and SYSRET instructions and the STAR register are not supported in this model.*

Write Handling Control Register (WHCR)

The processor contains a split level-1 (L1) 64-Kbyte writeback cache organized as a separate 32-Kbyte instruction cache and a 32-Kbyte data cache with two-way set associativity. The cache line size is 32 bytes and lines are read from memory using an efficient pipelined burst read cycle. Further performance gains are achieved by the implementation of a write allocation scheme.

A write allocate, if enabled, occurs when the processor has a pending memory write cycle to a cacheable line and the line does not currently reside in the L1 cache. For more information on write allocate, see the *Implementation of Write Allocate in the K86™ Processors Application Note*, order# 21326, and the “Cache

Organization" chapter of the *Mobile AMD-K6® Processor Data Sheet*, order# 20695.

This section describes two programmable mechanisms used by the processor to determine when to perform write allocate. When either of these mechanisms indicates that a pending write is to a cacheable area of memory, a write allocate is performed.

Before enabling write allocate or changing memory cacheability/writeability, the BIOS must writeback and invalidate the internal cache by using the WBINVD instruction. In addition, write allocate should be enabled only after performing any memory sizing or typing algorithms.

The Write Handling Control Register (WHCR) is an MSR that contains three fields—the WCDE bit, the Write Allocate Enable Limit (WAELIM) field, and the Write Allocate Enable 15-to-16-Mbyte (WAE15M) bit (see Figure 3).

Symbol	Description	Bits
WCDE	Always program to 0	8
WAELIM	Write Allocate Enable Limit	7–1
WAE15M	Write Allocate Enable 15-to-16-Mbyte	0

***Note**: Hardware RESET initializes this MSR to all zeros.*

Figure 3. Write Handling Control Register (WHCR)—MSR C000_0082h (Model 7)

WCDE. For proper functionality, always program bit 8 of WHCR to 0. See "Pipelining Support" on page 58 for more information on WCDE.

Write Allocate Enable Limit. The WAELIM field is 7 bits wide. This field, multiplied by 4 Mbytes, defines an upper memory limit. Any pending write cycle that misses the L1 cache and that addresses memory below this limit causes the processor to perform a write allocate (assuming the address is not within a range where write allocates are disallowed). Write allocate is disabled for memory accesses at and above this limit unless the processor determines a pending write cycle is cacheable by means of one of the other write allocate mechanisms—"Write

to a Cacheable Page" and "Write to a Sector" (for more information, see the "Cache Organization" chapter of the *Mobile AMD-K6® Processor Data Sheet*, order# 20695). The maximum value of this limit is $((2^7-1) \cdot 4$ Mbytes) = 508 Mbytes. When all the bits in this field are set to 0, all memory is above this limit and the write allocate mechanism is disabled (even if all bits in the WAELIM field are set to 0, write allocates can still occur due to the "Write to a Cacheable Page" and "Write to a Sector" mechanisms).

Once the BIOS determines the amount of RAM installed in the system, this number should also be used to program the WAELIM field. For example, a system with 32 Mbytes of RAM would program the WAELIM field with the value 0001000b. This value (8), when multiplied by 4 Mbytes, yields 32 Mbytes as the write allocate limit.

Write Allocate Enable 15-to-16-Mbyte. The WAE15M bit is used to enable write allocations for the memory write cycles that address the 1 Mbyte of memory between 15 Mbytes and 16 Mbytes. This bit must be set to 1 to allow write allocates in this memory area. This sub-mechanism of the WAELIM provides a memory hole to prevent write allocates. This memory hole is provided to account for a small number of uncommon memory-mapped I/O adapters that use this particular memory address space. If the system contains one of these peripherals, the bit should be set to 0 (even if the WAE15M bit is set to 0, write allocates can still occur between 15 Mbytes and 16 Mbytes due to the "Write to a Cacheable Page" and "Write to a Sector" mechanisms). The WAE15M bit is ignored if the value in the WAELIM field is set to less than 16 Mbytes.

By definition, write allocations are not performed in the memory area between 640 Kbytes and 1 Mbyte unless the processor determines a pending write cycle is cacheable by means of "Write to a Cacheable Page" or "Write to a Sector." It is not safe to perform write allocations between 640 Kbytes and 1 Mbyte (000A_0000h to 000F_FFFFh) because it is considered a noncacheable region of memory.

Mobile AMD-K6®-2 and Mobile AMD-K6-2-P Processors Model 8/[F:8]

The Mobile AMD-K6-2 and Mobile AMD-K6-2-P processors Model 8 provide the following ten MSRs. The first four MSRs are described in "Standard MSRs" on page 13. The contents of ECX selects the MSR to be addressed by the RDMSR and WRMSR instructions.

- Machine-Check Address Register (MCAR)—ECX = 00h
- Machine-Check Type Register (MCTR)—ECX = 01h
- Test Register 12 (TR12)—ECX = 0Eh
- Time Stamp Counter (TSC)—ECX = 10h
- Extended Feature Enable Register (EFER)—ECX = C000_0080h
- Write Handling Control Register (WHCR)—ECX = C000_0082h
- SYSCALL/SYSRET Target Address Register (STAR)—ECX = C000_0081h
- UC/WC Cacheability Control Register (UWCCR)—ECX = C000_0085h
- Processor State Observability Register (PSOR)—ECX = C000_0087h
- Page Flush/Invalidate Register (PFIR)—ECX = C000_0088h

Extended Feature Enable Register (EFER)

The Extended Feature Enable Register (EFER) contains the control bits that enable the extended features of the processor. Figure 4 shows the format of the EFER register, and Table 9 defines the function of each bit of the EFER register. The EFER register is MSR C000_0080h.

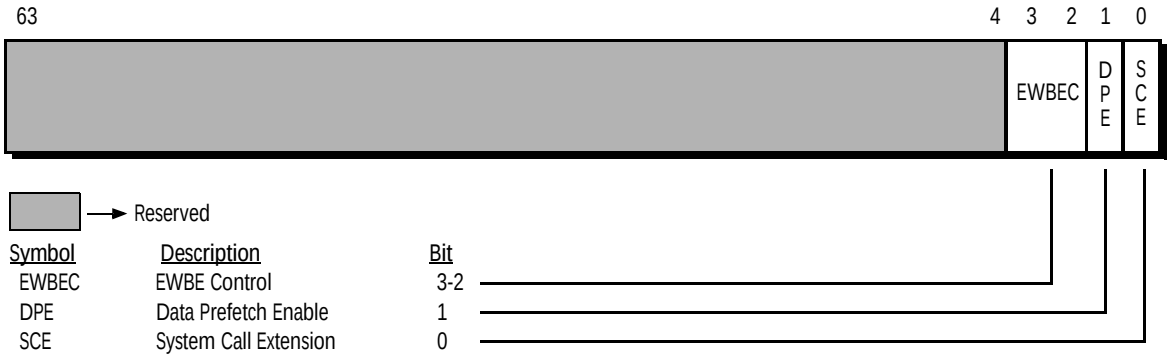


Figure 4. Extended Feature Enable Register (EFER)—MSR C000_0080h (Model 8)

Table 9. Extended Feature Enable Register (EFER) Definition (Model 8)

Bit	Description	R/W	Function
63–4	Reserved	R	Writing a 1 to any reserved bit causes a general protection fault to occur. All reserved bits are always read as 0.
3-2	EWBE Control (EWBEC)	R/W	This 2-bit field controls the behavior of the processor with respect to the ordering of write cycles and the EWBE# signal. EFER[3] and EFER[2] are Global EWBE Disable (GEWBED) and Speculative EWBE Disable (SEWBED), respectively.
1	Data Prefetch Enable (DPE)	R/W	DPE must be set to 1 to enable data prefetching (this is the default setting following reset). If enabled, cache misses initiated by a memory read within a 32-byte cache line are conditionally followed by cache-line fetches of the other line in the 64-byte sector.
0	System Call Extension (SCE)	R/W	SCE must be set to 1 to enable the usage of the SYSCALL and SYSRET instructions.

EWBE Control. The Mobile AMD-K6-2 and Mobile AMD-K6-2-P processors Model 8 contain an 8-byte write merge buffer that allows the processor to conditionally combine data from multiple noncacheable write cycles into this merge buffer. The merge buffer operates in conjunction with the Memory Type Range Registers (MTRRs). Refer to "UC/WC Cacheability Control Register (UWCCR)" on page 24 for a description of the MTRRs.

Merging multiple write cycles into a single write cycle reduces processor bus utilization and processor stalls, thereby increasing the overall system performance.

The presence of the merge buffer creates the potential to perform out-of-order write cycles relative to the processor's cache. In general, the ordering of write cycles that are driven externally on the system bus and those that hit the processor's cache can be controlled by the EWBE# signal. If EWBE# is sampled negated, the processor delays the commitment of write cycles to cache lines in the modified state or exclusive state in the processor's cache. Therefore, the system logic can enforce strong ordering by negating EWBE# until the external write cycle is complete, thereby ensuring that a subsequent write cycle that hits the cache does not complete ahead of the external write cycle.

However, the addition of the write merge buffer introduces the potential for out-of-order write cycles to occur between writes to the merge buffer and writes to the processor's cache. Because these writes occur entirely within the processor and are not sent out to the processor bus, the system logic is not able to enforce strong ordering with the EWBE# signal.

The EWBE control (EWBEC) bits provide a mechanism for enforcing three different levels of write ordering in the presence of the write merge buffer:

- EFER[3] is defined as the Global EWBE Disable (GEWBED). When GEWBED equals 1, the processor does not attempt to enforce any write ordering internally or externally (the EWBE# signal is ignored). This is the maximum performance setting.
- EFER[2] is defined as the Speculative EWBE Disable (SEWBED). SEWBED only affects the processor when GEWBED equals 0. If GEWBED equals 0 and SEWBED equals 1, the processor enforces strong ordering for all internal write cycles with the exception of write cycles addressed to a range of memory defined as uncacheable (UC) or write-combining (WC) by the MTRRs. In addition, the processor samples the EWBE# signal. If EWBE# is sampled negated, the processor delays the commitment of write cycles to processor cache lines in the modified state or exclusive state until EWBE# is sampled asserted.

  This setting provides performance comparable to, but slightly less than, the performance obtained when GEWBED equals 1 because some degree of write ordering is maintained.

- If GEWBED equals 0 and SEWBED equals 0, the processor enforces strong ordering for all internal and external write cycles. In this setting, the processor assumes, or *speculates*, that strong order must be maintained between writes to the merge buffer and writes that hit the processor's cache. Once the merge buffer is written out to the processor's bus, the EWBE# signal is sampled. If EWBE# is sampled negated, the processor delays the commitment of write cycles to processor cache lines in the modified state or exclusive state until EWBE# is sampled asserted.

  This setting is the default after RESET and provides the lowest performance of the three settings because full write ordering is maintained.

Table 10 summarizes the three settings of the EWBEC field for the EFER register, along with the effect of write ordering and performance.

Table 10. EWBEC Settings

EFER[3] (GEWBED)	EFER[2] (SEWBED)	Write Ordering	Performance
1	0 or 1	None	Best
0	1	All except UC/WC	Close-to-Best
0	0	All	Slowest

Enforcing complete write ordering in a uniprocessor system is usually not necessary. To achieve the highest level of performance while still maintaining support for the EWBE# signal, AMD recommends that the BIOS set EFER[3:2] to 01b (close-to-best performance). Many uniprocessor systems do not support the EWBE# signal, in which case AMD recommends that the BIOS set EFER[3:2] to 10b or 11b (best performance).

Write Handling Control Register (WHCR)

The Mobile AMD-K6-2 and Mobile AMD-K6-2-P processors Model 8 contain a split level-1 (L1) 64-Kbyte writeback cache organized as a separate 32-Kbyte instruction cache and a 32-Kbyte data cache with two-way set associativity. The cache line size is 32 bytes, and lines are read from memory using an efficient pipelined burst read cycle. Further performance gains are achieved by the implementation of a write allocation scheme.

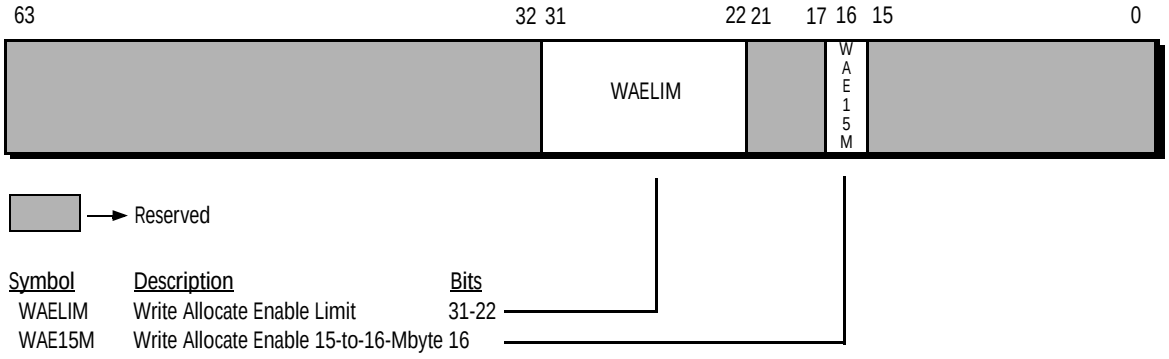
A write allocate, if enabled, occurs when the processor has a pending memory write cycle to a cacheable line and the line does not currently reside in the L1 cache. For more information on write allocate, see the *Implementation of Write Allocate in the K86™ Processors Application Note*, order# 21326 and the Cache Organization section of the appropriate Mobile AMD-K6 processor data sheet.

This section describes two programmable mechanisms used by the Mobile AMD-K6-2 and Mobile AMD-K6-2-P processors Model 8 to determine when to perform write allocate. When either of these mechanisms indicates that a pending write is to a cacheable area of memory, a write allocate is performed.

Before enabling write allocate or changing memory cacheability, the BIOS must write back and invalidate the internal cache by using the WBINVD instruction. In addition, write allocate should be enabled only after performing any memory sizing or typing algorithms.

The Write Handling Control Register (WHCR) is a MSR that contains two fields—the Write Allocate Enable Limit (WAELIM) field, and the Write Allocate Enable 15-to-16-Mbyte (WAE15M) bit (see Figure 5).

Note: *The WHCR register as defined in the Model 7 has changed in the Model 8.*



Note*: Hardware RESET initializes this MSR to all zeros.*

Figure 5. Write Handling Control Register (WHCR)—MSR C000_0082h (Models 8, 9, and D)

Write Allocate Enable Limit. The WAELIM field is 10 bits wide. This field, multiplied by 4 Mbytes, defines an upper memory limit. Any pending write cycle that misses the L1 cache and that

addresses memory below this limit causes the processor to perform a write allocate (assuming the address is not within a range where write allocates are disallowed). Write allocate is disabled for memory accesses at and above this limit unless the processor determines a pending write cycle is cacheable by means of one of the other write allocate mechanisms—"Write to a Cacheable Page" and "Write to a Sector" (for more information, see the "Cache Organization" chapter in the appropriate Mobile AMD-K6 processor data sheet). The maximum value of this limit is $((2^{10}-1) \cdot 4$ Mbytes) = 4092 Mbytes. When all the bits in this field are set to 0, all memory is above this limit and the write allocate mechanism is disabled (even if all bits in the WAELIM field are set to 0, write allocates can still occur due to the "Write to a Cacheable Page" and "Write to a Sector" mechanisms).

Once the BIOS determines the amount of RAM installed in the system, this number should also be used to program the WAELIM field. For example, a system with 32 Mbytes of RAM would program the WAELIM field with the value 00_0000_1000b. This value (8), when multiplied by 4 Mbytes, yields 32 Mbytes as the write allocate limit.

Write Allocate Enable 15-to-16-Mbyte. The WAE15M bit is used to enable write allocations for the memory write cycles that address the 1 Mbyte of memory between 15 Mbytes and 16 Mbytes. This bit must be set to 1 to allow write allocates in this memory area. This sub-mechanism of the WAELIM provides a memory hole to prevent write allocates. This memory hole is provided to account for a small number of uncommon memory-mapped I/O adapters that use this particular memory address space. If the system contains one of these peripherals, the bit should be set to 0 (even if the WAE15M bit is set to 0, write allocates can still occur between 15 Mbytes and 16 Mbytes due to the "Write to a Cacheable Page" and "Write to a Sector" mechanisms). The WAE15M bit is ignored if the value in the WAELIM field is set to less than 16 Mbytes.

By definition, write allocations are not performed in the memory area between 640 Kbytes and 1 Mbyte unless the processor determines a pending write cycle is cacheable by means of "Write to a Cacheable Page" or "Write to a Sector." It is not safe to perform write allocations between 640 Kbytes and 1 Mbyte (000A_0000h to 000F_FFFFh) because it is considered a noncacheable region of memory. Additionally, if a memory

region is defined as write-combinable or uncacheable by a MTRR, write allocates are not performed in that region.

SYSCALL/SYSRET Target Address Register (STAR)

All Mobile AMD-K6 processors, Models 8 and above, implement the STAR register. This register contains the target EIP address used by the SYSCALL instruction and the 16-bit code and stack segment selector bases used by the SYSCALL and SYSRET instructions. Figure 6 shows the format of the STAR register, and Table 11 defines the function of each field of the STAR register. The STAR register is MSR C000_0081h. For more information about SYSCALL/SYSRET, see the *Mobile AMD-K6® Processor SYSCALL and SYSRET Instruction Specification Application Note*, order# 21086.

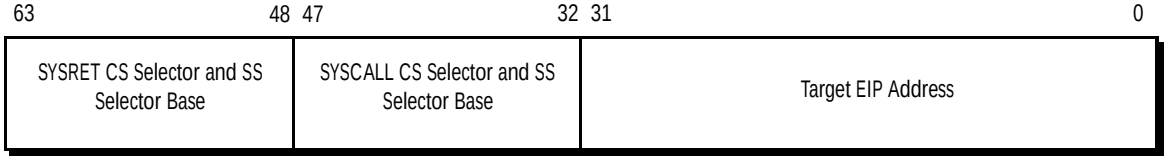


Figure 6. SYSCALL/SYSRET Target Address Register (STAR)—MSR C000_0081h (Models 8, 9, and D)

Table 11. SYSCALL/SYSRET Target Address Register (STAR) Definition (Models 8, 9, and D)

Bit	Description	R/W	Function
63–48	SYSRET CS and SS Selector Base	R/W	During the SYSRET instruction, this field is copied into the CS register and the contents of this field, plus 1000b, are copied into the SS register.
47–32	SYSCALL CS and SS Selector Base	R/W	During the SYSCALL instruction, this field is copied into the CS register and the contents of this field, plus 1000b, are copied into the SS register.
31–0	Target EIP Address	R/W	During the SYSCALL instruction, this address is copied into the EIP and points to the new starting address.

UC/WC Cacheability Control Register (UWCCR)

The Mobile AMD-K6-2 and Mobile AMD-K6-2-P processors Model 8 provide two variable-range Memory Type Range Registers (MTRRs)—MTRR0 and MTRR1—that each specify a range of memory. Each range can be defined as one of the following memory types:

- Uncacheable (UC) memory—Memory read cycles are sourced directly from the specified memory address and the processor does not allocate a cache line. Memory write

cycles are targeted at the specified memory address and a write allocation does not occur.

- Write-Combining (WC) memory—Memory read cycles are sourced directly from the specified memory address and the processor does not allocate a cache line. The processor conditionally combines data from multiple noncacheable write cycles that are addressed within this range into a merge buffer. Merging multiple write cycles into a single write cycle reduces processor bus utilization and processor stalls, thereby increasing the overall system performance. This memory type is applicable for linear video frame buffers.

Note: *The MTRRs defined in this document are not software compatible to the MTRRs defined by the Pentium Pro and Pentium II processors.*

The programmer accesses the MTRRs by addressing the 64-bit MSR known as the UC/WC Cacheability Control Register (UWCCR). The MSR address of the UWCCR is C000_0085h. Following reset, all bits in the UWCCR register are set to 0. MTRR0 (lower 32 bits of the UWCCR register) defines the size and memory type of range 0 and MTRR1 (upper 32 bits) defines the size and memory type of range 1 (see Figure 7).

Prior to programming write-combining or uncacheable areas of memory in the UWCCR, the BIOS must disable the processor's cache, then flush the cache. This can be achieved by setting the CD bit in CR0 to 1 and executing the WBINVD instruction. Following the programming of the UWCCR, the processor's cache must be enabled by setting the CD bit in CR0 to 0.

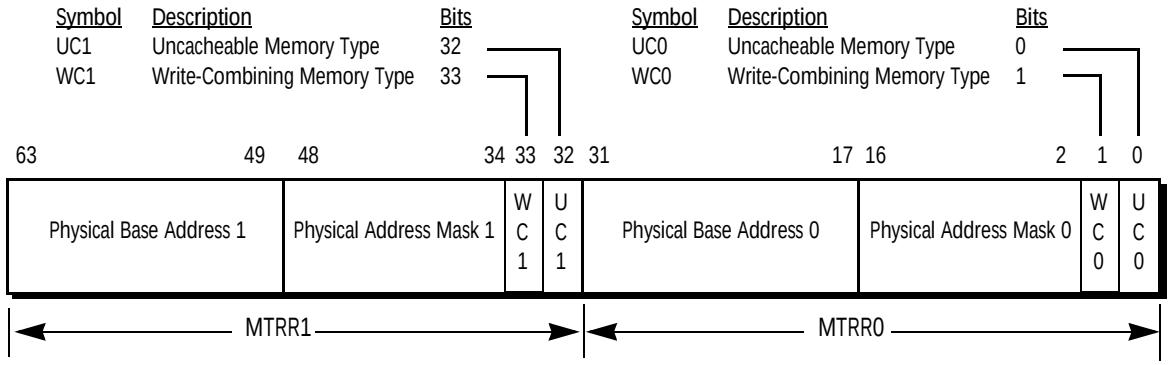


Figure 7. UC/WC Cacheability Control Register (UWCCR)—MSR C000_0085h (Models 8, 9, and D)

Physical Base Address n (n=0, 1). This address is the 15 most-significant bits of the physical base address of the memory range. The least-significant 17 bits of the base address are not needed because the base address is by definition always aligned on a 128-Kbyte boundary.

Physical Address Mask n (n=0, 1). This value is the 15 most-significant bits of a physical address mask that is used to define the size of the memory range. This mask is logically ANDed with both the physical base address field of the UWCCR register and the physical address generated by the processor. If the results of the two AND operations are equal, then the generated physical address is considered within the range. That is, if:

Mask & Physical Base Address = Mask & Physical Address Generated

then the physical address generated by the processor is in the range.

WCn (n=0, 1). When set to 1, this memory range is defined as write-combinable (refer to Table 12). Write-combinable memory is uncacheable.

UCn (n=0, 1). When set to 1, this memory range is defined as uncacheable (refer to Table 12).

Table 12. WC/UC Memory Type

WCn	UCn	Memory Type
0	0	No effect on cacheability or write-combining
1	0	Write-combining memory range (uncacheable)
0 or 1	1	Uncacheable memory range

Memory-Range Restrictions. The following rules regarding the address alignment and size of each range must be adhered to when programming the physical base address and physical address mask fields of the UWCCR register:

- The minimum size of each range is 128 Kbytes.
- The physical base address must be aligned on a 128-Kbyte boundary.
- The physical base address must be *range-size aligned*. For example, if the size of the range is 1 Mbyte, then the

physical base address must be aligned on a 1-Mbyte boundary.

- All bits set to 1 in the physical address mask must be contiguous. Likewise, all bits set to 0 in the physical address mask must be contiguous. For example:

  111_1111_1100_0000b is a valid physical address mask

  111_1111_1101_0000b is invalid

Table 13 lists the valid physical address masks and the resulting range sizes that can be programmed in the UWCCR register.

Table 13. Valid Masks and Range Sizes

Masks	Size
111_1111_1111_1111b	128 Kbytes
111_1111_1111_1110b	256 Kbytes
111_1111_1111_1100b	512 Kbytes
111_1111_1111_1000b	1 Mbyte
111_1111_1111_0000b	2 Mbytes
111_1111_1110_0000b	4 Mbytes
111_1111_1100_0000b	8 Mbytes
111_1111_1000_0000b	16 Mbytes
111_1111_0000_0000b	32 Mbytes
111_1110_0000_0000b	64 Mbytes
111_1100_0000_0000b	128 Mbytes
111_1000_0000_0000b	256 Mbytes
111_0000_0000_0000b	512 Mbytes
110_0000_0000_0000b	1 Gbyte
100_0000_0000_0000b	2 Gbytes
000_0000_0000_0000b	4 Gbytes

Example. Suppose that the range of memory from 16 Mbytes to 32 Mbytes is uncacheable, and the 8-Mbyte range of memory on

top of 1 Gbyte is write-combinable. Range 0 is defined as the uncacheable range, and range 1 is defined as the write-combining range.

Extracting the 15 most-significant bits of the 32-bit physical base address that corresponds to 16 Mbytes (0100_0000h) yields a physical base address 0 field of 000_0000_1000_0000b. Because the uncacheable range size is 16 Mbytes, the physical mask value 0 field is 111_1111_1000_0000b, according to Table 13 . Bit 1 of the UWCCR register (WC0) is set to 0 and bit 0 of the UWCCR register is set to 1 (UC0).

Extracting the 15 most-significant bits of the 32-bit physical base address that corresponds to 1 Gbyte (4000_0000h) yields a physical base address 1 field of 010_0000_0000_0000b. Because the write-combining range size is 8 Mbytes, the physical mask value 1 field is 111_1111_1100_0000b, according to Table 13 . Bit 33 of the UWCCR register (WC1) is set to 1 and bit 32 of the UWCCR register is set to 0 (UC1).

Processor State Observability Register (PSOR)

The Mobile AMD-K6-2 and Mobile AMD-K6-2-P processors Model 8 provide the Processor State Observability Register (PSOR) (see Figure 8).

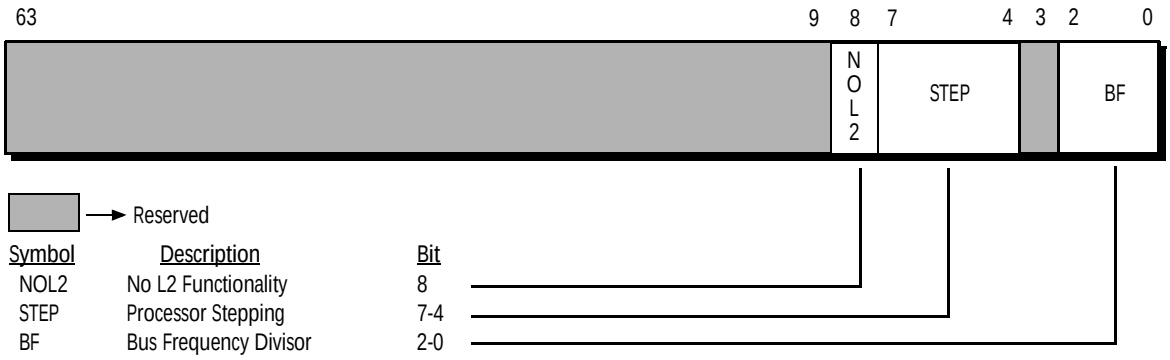


Figure 8. Processor State Observability Register (PSOR)—MSR C000_0087h (Models 8 and 9)

NOL2. This read-only bit indicates whether the processor contains an L2 cache. This bit is always set to 1 for Model 8.

STEP. This read-only field contains the stepping ID. This is identical to the value returned by CPUID standard function 1 in EAX[3:0].

BF. This read-only field contains the value of the BF signals sampled by the processor during the falling transition of RESET, which allows the BIOS to determine the frequency of the host bus. The core frequency must first be determined using the Time Stamp Counter method (See "Time Stamp Counter (TSC)" on page 14). The core frequency is then divided by the processor-clock to bus-clock ratio as determined by the BF field of the PSOR register (see Table 14). The result is the frequency of the processor bus.

Table 14. Processor-to-Bus Clock Ratios (Models 8 and 9)

State of BF[2:0]	Processor-to-Bus Clock Ratio
100b	2.5x
101b	3.0x
110b	6.0x*
111b	3.5x
000b	4.5x
001b	5.0x
010b	4.0x
011b	5.5x

Note:

* *The 2.0x ratio that was supported on Model 7 is no longer supported on Model 8 or Model 9. Instead, a ratio of 6.0x is selected when BF[2:0] equals 110b.*

Page Flush/Invalidate Register (PFIR)

The Mobile AMD-K6-2 and Mobile AMD-K6-2-P processors Model 8 contain the Page Flush/Invalidate Register (PFIR) (see Figure 9) that allows cache invalidation and optional flushing of a specific 4-Kbyte page from the linear address space. The total amount of L1 cache in the Model 8 is 64 Kbytes. Using this register can result in a much lower cycle count for flushing particular pages versus flushing the entire cache. When the PFIR is written to (using the WRMSR instruction), the invalidation and, optionally, the flushing begins.

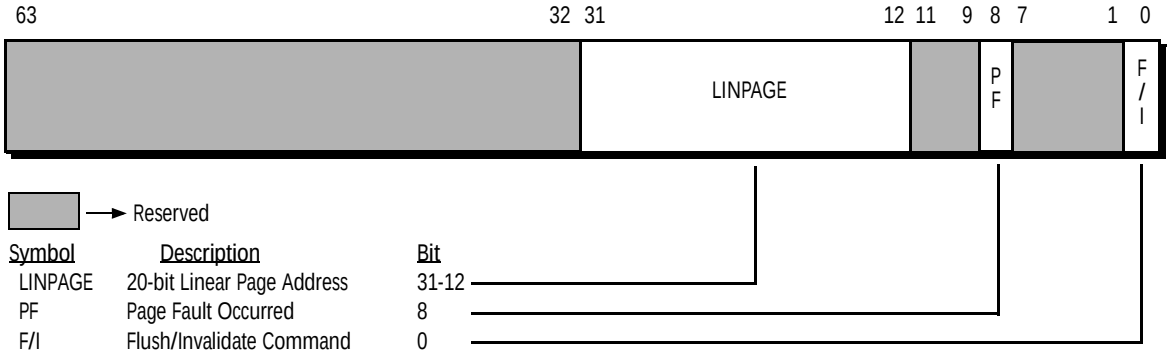


Figure 9. Page Flush/Invalidate Register (PFIR)—MSR C000_0088h (Models 8, 9, and D)

LINPAGE. This 20-bit field must be written with bits 31:12 of the linear address of the 4-Kbyte page that is to be invalidated and optionally flushed from the L1 cache.

PF. If an attempt to invalidate or flush a page results in a page fault, the processor sets the PF bit to 1, and the invalidate or flush operation is not performed (even though invalidate operations do not normally generate page faults). In this case, an actual page fault exception is not generated. If the PF bit equals 0 after an invalidate or flush operation, then the operation executed successfully. The PF bit must be read after every write to the PFIR register to determine if the invalidate or flush operation executed successfully.

F/I. This bit is used to control the type of action that occurs to the specified linear page. If a 0 is written to this bit, the operation is a flush, in which case all cache lines in the modified state within the specified page are written back to memory, after which the entire page is invalidated. If a 1 is written to this bit, the operation is an invalidation, in which case the entire page is invalidated without the occurrence of any writebacks.

Mobile AMD-K6®-III-P Processor Model 9

The Mobile AMD-K6-III-P processor Model 9 provides the following eleven MSRs. The first four MSRs are described in "Standard MSRs" on page 13. The contents of ECX selects the MSR to be addressed by the RDMSR and WRMSR instructions.

- Machine-Check Address Register (MCAR)—ECX = 00h
- Machine-Check Type Register (MCTR)—ECX = 01h
- Test Register 12 (TR12)—ECX = 0Eh
- Time Stamp Counter (TSC)—ECX = 10h
- Extended Feature Enable Register (EFER)—ECX = C000_0080h
- Write Handling Control Register (WHCR)—ECX = C000_0082h
- SYSCALL/SYSRET Target Address Register (STAR)—ECX = C000_0081h
- UC/WC Cacheability Control Register (UWCCR)—ECX = C000_0085h
- Processor State Observability Register (PSOR)—ECX = C000_0087h
- Page Flush/Invalidate Register (PFIR)—ECX = C000_0088h
- Level-2 Cache Array Access Register (L2AAR)—ECX = C000_0089h

Extended Feature Enable Register (EFER)

Bits 3:0 of the EFER register in the Mobile AMD-K6-III-P processor Model 9 are identical to the implementation of these bits in the Model 8. See “Extended Feature Enable Register (EFER)” on page 18. The L2 Disable bit (L2D)—EFER[4]—is an addition to the EFER register in the Model 9.

Figure 10 shows the format of the EFER register, and Table 15 defines the function of each bit of the EFER register. The EFER register is MSR C000_0080h.

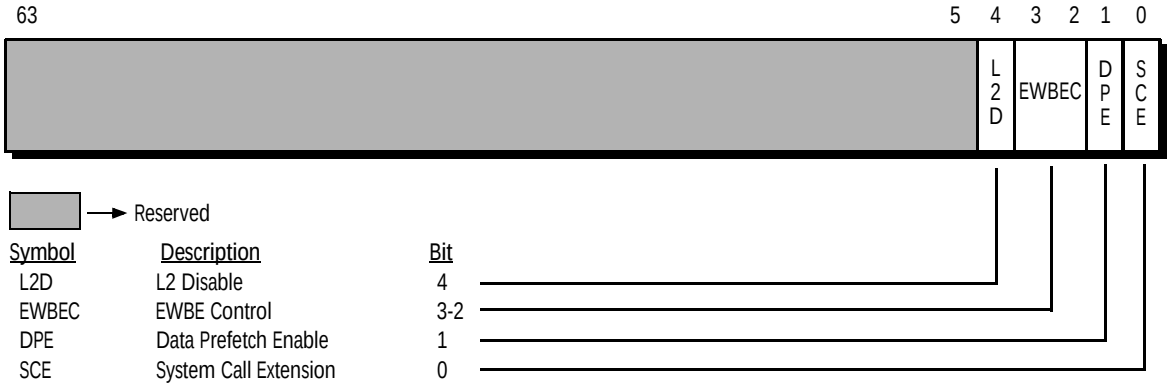


Figure 10. Extended Feature Enable Register (EFER)—MSR C000_0080h (Models 9 and D)

Table 15. Extended Feature Enable Register (EFER) Definition (Models 9 and D)

Bit	Description	R/W	Function
63–5	Reserved	R	Writing a 1 to any reserved bit causes a general protection fault to occur. All reserved bits are always read as 0.
4	L2 Disable (L2D)	R/W	If L2D is set to 1, the L2 cache is completely disabled. This bit is provided for debug and testing purposes. For normal operation and maximum performance, this bit must be set to 0 (this is the default setting following reset).
3-2	EWBE Control (EWBEC)	R/W	This 2-bit field controls the behavior of the processor with respect to the ordering of write cycles and the EWBE# signal. EFER[3] and EFER[2] are Global EWBE Disable (GEWBED) and Speculative EWBE Disable (SEWBED), respectively.
1	Data Prefetch Enable (DPE)	R/W	DPE must be set to 1 to enable data prefetching (this is the default setting following reset). If enabled, cache misses initiated by a memory read within a 32-byte cache line are conditionally followed by cache-line fetches of the other line in the 64-byte sector.
0	System Call Extension (SCE)	R/W	SCE must be set to 1 to enable the usage of the SYSCALL and SYSRET instructions.

Note: *Setting L2D to 1 does not guarantee cache coherency. To ensure coherency, the processor's caches must be disabled (by setting the CD bit of the CR0 register to 1), then flushed prior to setting L2D to 1.*

Write Handling Control Register (WHCR)

The Mobile AMD-K6-III-P processor contains a split level-1 (L1) 64-Kbyte writeback cache organized as a separate 32-Kbyte instruction cache and a 32-Kbyte data cache with two-way set associativity. The cache line size is 32 bytes, and lines are read

from memory using an efficient pipelined burst read cycle. In addition, the Mobile AMD-K6-III-P processor also contains a 256-Kbyte, 4-way set associative, unified level-2 (L2) cache. Further performance gains are achieved by the implementation of a write allocation scheme.

The WHCR register in the Mobile AMD-K6-III-P processor Model 9 is identical to the implementation of this register in the Model 8. See "Write Handling Control Register (WHCR)" on page 21.

Note: *The WHCR register as defined in the Model 7 has changed in the Model 9.*

SYSCALL/SYSRET Target Address Register (STAR)

The STAR register in the Mobile AMD-K6-III-P processor Model 9 is identical to the implementation of this register in the Model 8. See "SYSCALL/SYSRET Target Address Register (STAR)" on page 24.

UC/WC Cacheability Control Register (UWCCR)

The UWCCR register in the Mobile AMD-K6-III-P processor Model 9 is identical to the implementation of this register in the Model 8. See "UC/WC Cacheability Control Register (UWCCR)" on page 24.

Processor State Observability Register (PSOR)

The Mobile AMD-K6-III-P processor Model 9 provides the Processor State Observability Register (PSOR) (see Figure 11).

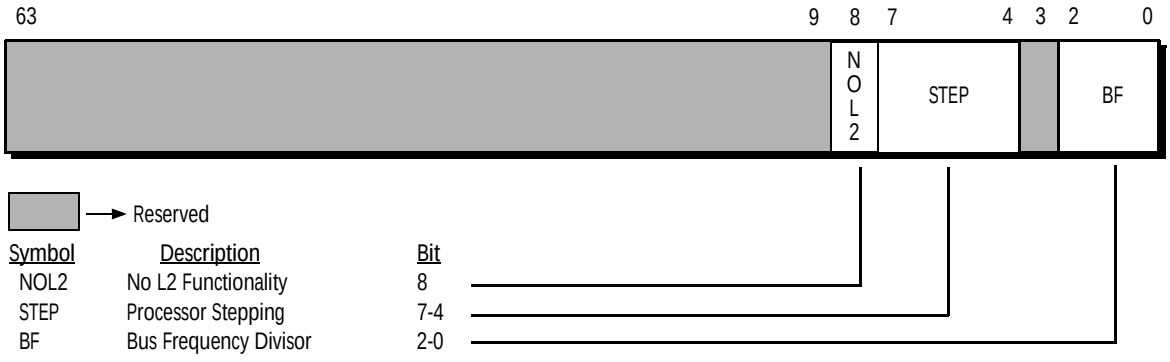


Figure 11. Processor State Observability Register (PSOR)—MSR C000_0087h (Model 9)

NOL2. This read-only bit indicates whether the processor contains an L2 cache. This bit is always set to 0 for Model 9.

STEP. This read-only field contains the stepping ID. This is identical to the value returned by CPUID standard function 1 in EAX[3:0].

BF. This read-only field contains the value of the BF signals sampled by the processor during the falling transition of RESET, which allows the BIOS to determine the frequency of the host bus. The core frequency must first be determined using the Time Stamp Counter method (See "Time Stamp Counter (TSC)" on page 14). The core frequency is then divided by the processor-clock to bus-clock ratio as determined by the BF field of the PSOR register (see Table 14 on page 29). The result is the frequency of the processor bus.

Page Flush/Invalidate Register (PFIR)

The PFIR register in the Mobile AMD-K6-III-P processor Model 9 is identical to the implementation of this register in the Model 8. See “Page Flush/Invalidate Register (PFIR)” on page 29. The invalidate and flush operations affect the processor’s L1 and L2 caches on the Model 9.

Level-2 Cache Array Access Register (L2AAR)

The Mobile AMD-K6-III-P processor Model 9 provides the L2AAR register that allows for direct access to the L2 cache and L2 tag arrays. The 256-Kbyte L2 cache in the Mobile AMD-K6-III-P processor is organized as shown in Figure 12:

- Four 64-Kbyte ways
- Each way contains 1024 sectors
- Each set contains four 64-byte sectors (one sector in each way)
- Each sector contains two 32-byte cache lines
- Each cache line contains four 8-byte octets
- Each octet contains an upper and lower dword (4 bytes)

Each line within a sector contains its own MESI state bits, and associated with each sector is a tag and LRU (Least Recently Used) information.

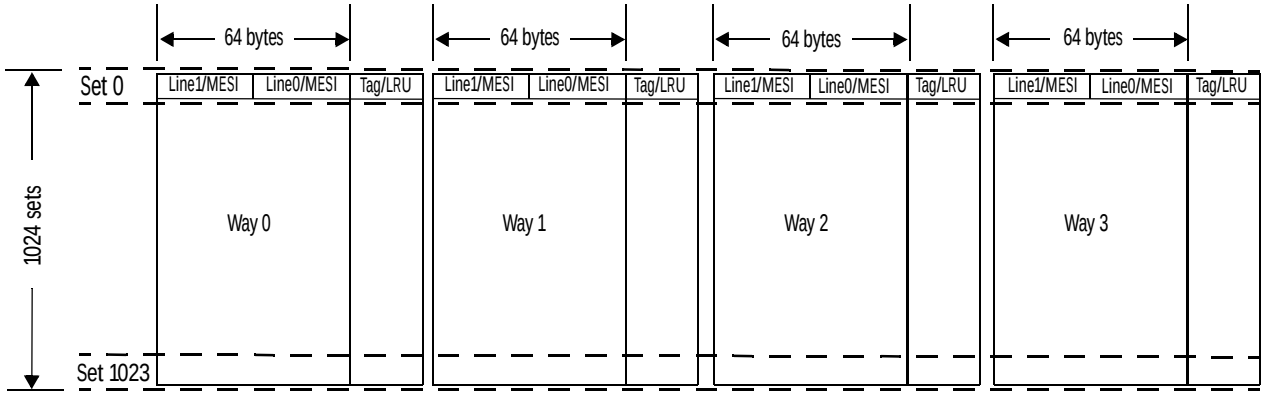


Figure 12. L2 Cache Organization

Figure 13 shows the L2 cache sector and line organization. If bit 5 of the address of a cache line equals 1, then this cache line is stored in Line 1 of a sector. Similarly, if bit 5 of the address of a cache line equals 0, then this cache line is stored in Line 0 of a sector.

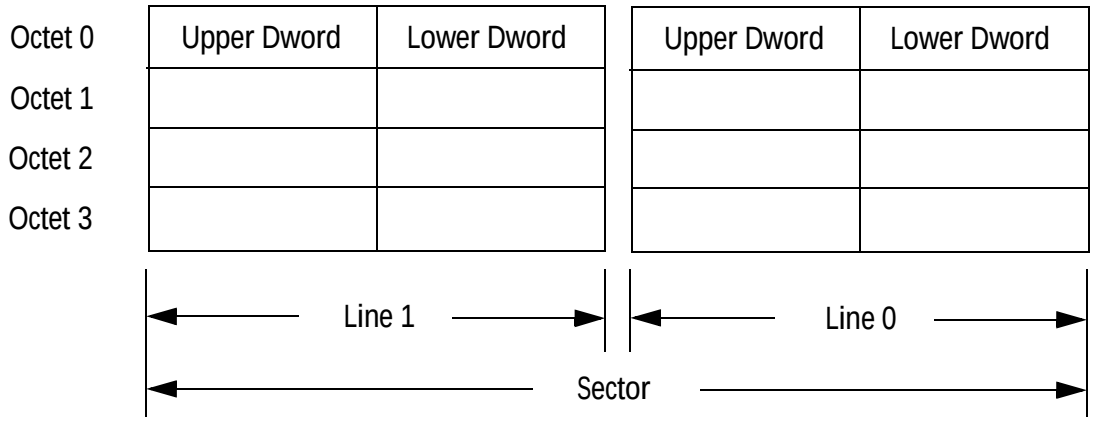


Figure 13. L2 Cache Sector and Line Organization

The L2AAR register is MSR C000_0089h. The operation that is performed on the L2 cache is a function of the instruction executed—RDMSR or WRMSR—and the contents of the EDX register. The EDX register specifies the location of the access, and whether the access is to the L2 cache data or tags (refer to Figure 14). Bit 20 of EDX (T/D) determines whether the access is to the L2 cache data or tag. Table 16 describes the operation that is performed based on the instruction and the T/D bit.

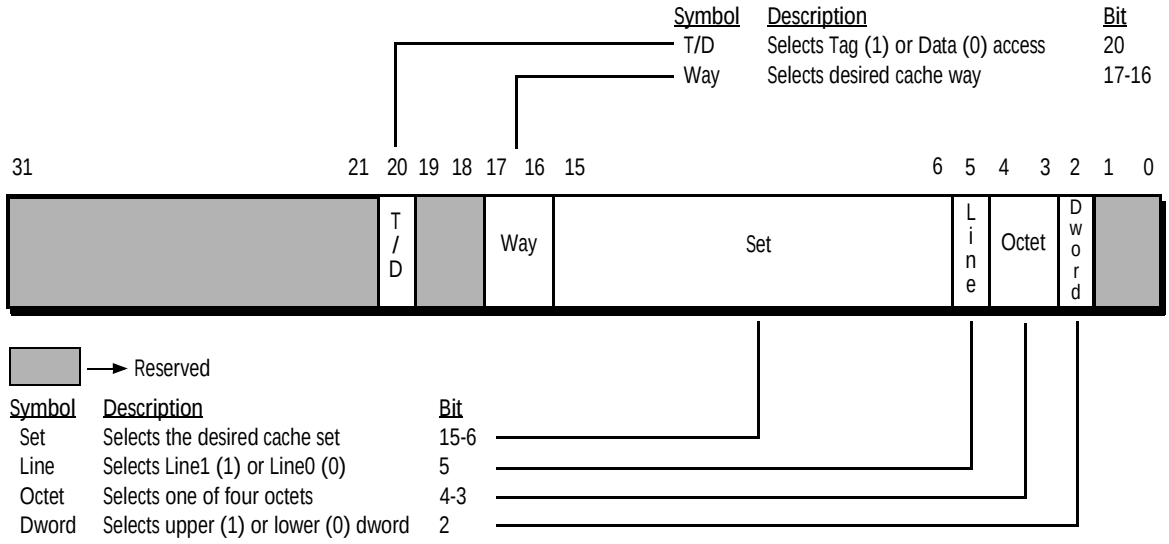


Figure 14. L2 Tag or Data Location - EDX

Table 16. Tag versus Data Selector

Instruction	T/D (EDX[20])	Operation
RDMSR	0	Read dword from L2 data array into EAX. Dword location is specified by EDX.
RDMSR	1	Read tag, line state and LRU information from L2 tag array into EAX. Location of tag is specified by EDX.
WRMSR	0	Write dword to the L2 data array using data in EAX. Dword location is specified by EDX.
WRMSR	1	Write tag, line state and LRU information into L2 tag array from EAX. Location of tag is specified by EDX.

When the L2AAR is read or written, EDX is left unchanged. This facilitates multiple accesses when testing the entire cache/tag array.

If the L2 cache data is read (as opposed to reading the tag information), the result (dword) is placed in EAX in the format as illustrated in Figure 15. Similarly, if the L2 cache data is written, the write data is taken from EAX.

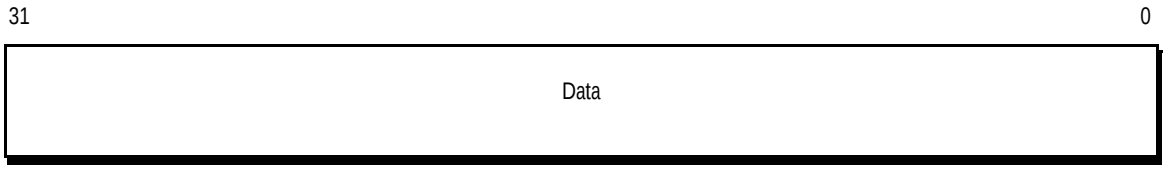


Figure 15. L2 Data - EAX

If the L2 tag is read (as opposed to reading the cache data), the result is placed in EAX in the format as illustrated in Figure 16. Similarly, if the L2 tag is written, the write data is taken from EAX.

When writing to the L2 tag, special consideration must be given to the least significant bit of the Tag field of the EAX register—EAX[15]. The length of the L2 tag required to support the 256-Kbyte L2 cache on the Model 9 is 16 bits, which corresponds to bits 31:16 of the EAX register. However, the processor provides a total of 17 bits for storing the L2 tag—that is, 16 bits for the tag (EAX[31:16]), plus an additional bit for internal purposes (EAX[15]). During normal operation, the processor ensures that this additional bit (bit 15) always corresponds to the set in which the tag resides. Note that bits 15:6 of the address determine the set, in which case bit 15 equal to 0 addresses sets 0 through 511, and bit 15 equal to 1 addresses sets 512 through 1023.

In order to set the full 17-bit L2 tag properly when using the L2AAR register, EAX[15] must likewise correspond to the set in which the tag is being written—that is, EAX[15] must be equal to EDX[15] (refer to Figure 14 and Figure 16).

It is important to note that this special consideration is only required if the processor will subsequently be expected to properly execute instructions or access data from the L2 cache following the setup of the L2 cache by means of the L2AAR register. If the intent of using the L2AAR register is solely to test or debug the L2 cache without the subsequent intent of executing instructions or accessing data from the L2 cache, then this consideration is not required.

When accessing the L2 tag, the Line, Octet, and Dword fields of the EDX register are ignored.

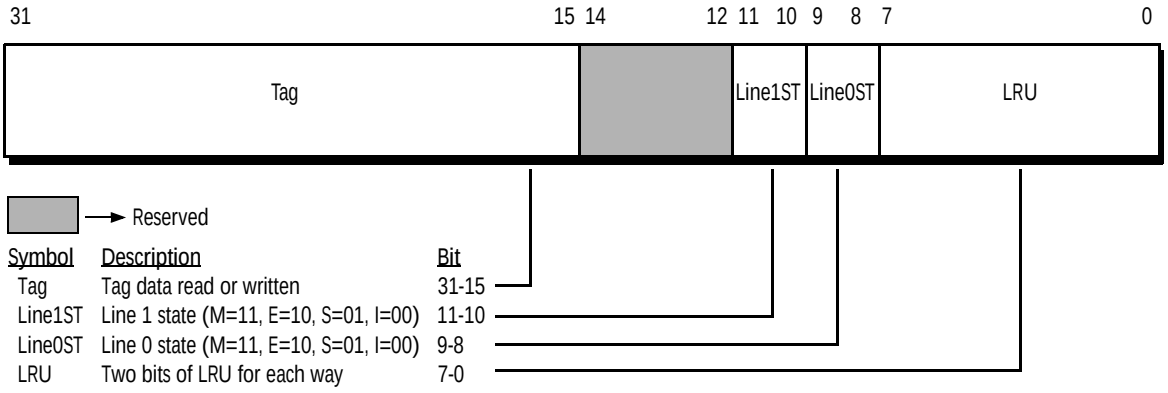


Figure 16. L2 Tag Information - EAX

LRU (Least Recently Used). For the 4-way set associative L2 cache, each way has a 2-bit LRU field for each sector. Values for the LRU field are 00b, 01b, 10b, and 11b, where 00b indicates that the sector is "most recently used," and 11b indicates that the sector is "least recently used" (see Figure 17). EAX[7:6] indicate LRU information for Way 0, EAX[5:4] for Way 1, EAX[3:2] for Way 2, and EAX[1:0] for Way 3.

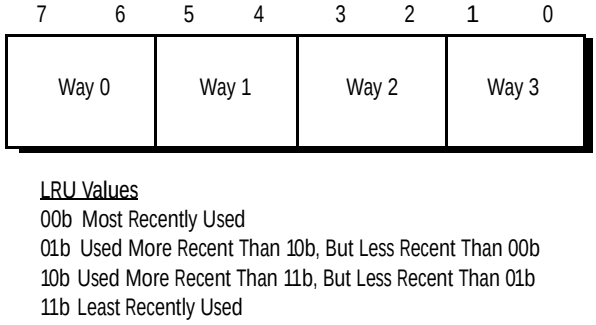


Figure 17. LRU Byte

Mobile AMD-K6-2+ and AMD-K6-III+ Processor Model D

The Mobile AMD-K6-2+ and Mobile AMD-K6-III+ processors Model D provide the following twelve MSRs. The first four MSRs are described in "Standard MSRs" on page 13. The contents of ECX selects the MSR to be addressed by the RDMSR and WRMSR instructions.

- Machine-Check Address Register (MCAR)—ECX = 00h
- Machine-Check Type Register (MCTR)—ECX = 01h
- Test Register 12 (TR12)—ECX = 0Eh
- Time Stamp Counter (TSC)—ECX = 10h
- Extended Feature Enable Register (EFER)—ECX = C000_0080h
- Write Handling Control Register (WHCR)—ECX = C000_0082h
- SYSCALL/SYSRET Target Address Register (STAR)—ECX = C000_0081h
- UC/WC Cacheability Control Register (UWCCR)—ECX = C000_0085h
- Processor State Observability Register (PSOR)—ECX = C000_0087h
- Page Flush/Invalidate Register (PFIR)—ECX = C000_0088h
- Level-2 Cache Array Access Register (L2AAR)—ECX = C000_0089h
- Enhanced Power Management Register (EPMR)—ECX = C000_0086h

Extended Feature Enable Register (EFER)

The EFER register in the Mobile AMD-K6-2+ and Mobile AMD-K6-III+ processors Model D are identical to the implementation of this register in the Model 9. See “Extended Feature Enable Register (EFER)” on page 31.

Write Handling Control Register (WHCR)

The Mobile AMD-K6-2+ and Mobile AMD-K6-III+ processors contain a split level-1 (L1) 64-Kbyte writeback cache organized as a separate 32-Kbyte instruction cache and a 32-Kbyte data cache with two-way set associativity. The cache line size is 32 bytes, and lines are read from memory using an efficient pipelined burst read cycle. In addition, the Mobile AMD-K6-2+ and Mobile AMD-K6-III+ processors Model D contain a 4-way set associative, 128-Kbyte or 256-Kbyte unified level-2 (L2)

cache respectively. Further performance gains are achieved by the implementation of a write allocation scheme.

The WHCR register in the Mobile AMD-K6-2+ and Mobile AMD-K6-III+ processors Model D are identical to the implementation of this register in the Model 8. See "Write Handling Control Register (WHCR)" on page 21.

Note: *The WHCR register as defined in the Model 7 has changed in the Model D.*

SYSCALL/SYSRET Target Address Register (STAR)

The STAR register in the Mobile AMD-K6-2+ and Mobile AMD-K6-III+ processors Model D are identical to the implementation of this register in the Model 8. See "SYSCALL/SYSRET Target Address Register (STAR)" on page 24.

UC/WC Cacheability Control Register (UWCCR)

The UWCCR register in the Mobile AMD-K6-2+ and Mobile AMD-K6-III+ processors Model D are identical to the implementation of this register in the Model 8. See "UC/WC Cacheability Control Register (UWCCR)" on page 24.

Processor State Observability Register (PSOR)

The Mobile AMD-K6-2+ and Mobile AMD-K6-III+ processors Model D provide the Processor State Observability Register (PSOR) (see Figure 18).

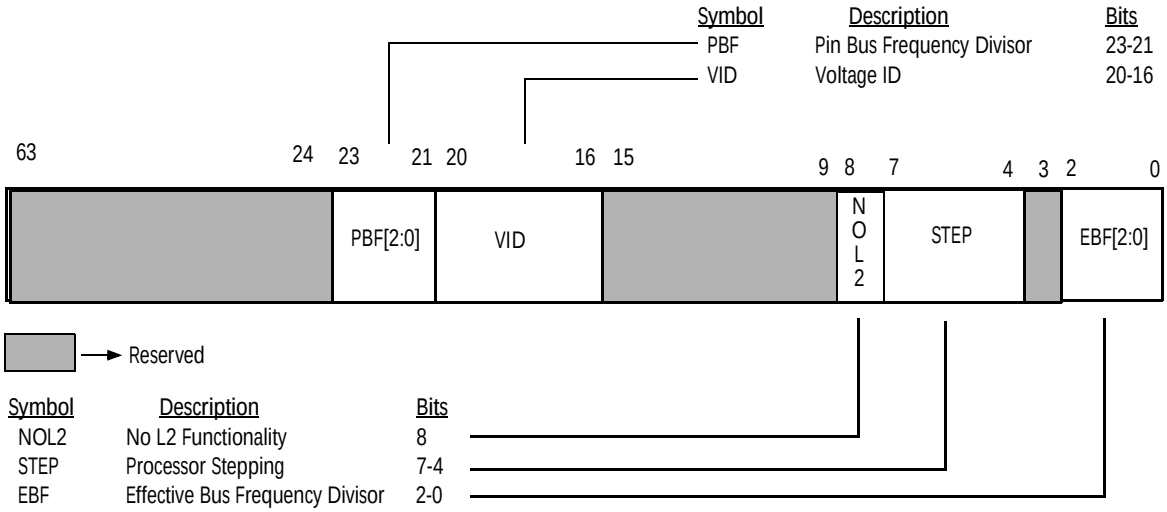


Figure 18. Processor State Observability Register (PSOR)—MSR C000_0087h (Model D)

PBF[2:0]. This read-only field contains the BF divisor values externally applied to the processor BF[2:0] pins. These input BF values are sampled by the processor during the falling transition of RESET.

Note: This BF divisor value may be different than the BF divisor value supplied to the processor's internal PLL.

VID. This read-only field contains the Voltage ID bits driven to the processor VID[4:0] pins upon entering the Enhanced Power Management (EPM) Stop Grant state. These bits are initialized to 01010b and driven on the VID[4:0] pins at RESET.

NOL2. This read-only bit indicates whether the processor contains an L2 cache. This bit is always set to 1 for Model D.

STEP. This read-only field contains the stepping ID. This is identical to the value returned by CPUID standard function 1 in EAX[3:0].

EBF[2:0]. This read-only field contains the effective value of the BF divisor supplied to the processor's internal PLL, which allows the BIOS to determine the frequency of the host bus. The core frequency must first be determined using the Time Stamp Counter method (See "Time Stamp Counter (TSC)" on page 14). The core frequency is then divided by the processor-to-bus clock ratio as determined by the EBF field of the PSOR register (see Table 17). The result is the frequency of the processor bus.

Table 17. Processor-to-Bus Clock Ratios (Model D)

State of EBF[2:0]	Processor-to-Bus Clock Ratio
100b	2.0x*
101b	3.0x
110b	6.0x
111b	3.5x
000b	4.5x
001b	5.0x
010b	4.0x

Note:
* *The 2.5x ratio that was supported on Models 8 and 9 is no longer supported on Model D. Instead, a ratio of 2.0x is selected when EBF[2:0] equals 100b.*

Table 17. Processor-to-Bus Clock Ratios (Model D) (continued)

State of EBF[2:0]	Processor-to-Bus Clock Ratio
011b	5.5x

Note:
* *The 2.5x ratio that was supported on Models 8 and 9 is no longer supported on Model D. Instead, a ratio of 2.0x is selected when EBF[2:0] equals 100b.*

Page Flush/Invalidate Register (PFIR)

The PFIR register in the Mobile AMD-K6-2+ and Mobile AMD-K6-III+ processors Model D are identical to the implementation of this register in the Model 8. See "Page Flush/Invalidate Register (PFIR)" on page 29. The invalidate and flush operations affect the processor's L1 and L2 caches on the Model D.

Level-2 Cache Array Access Register (L2AAR)

The Mobile AMD-K6-2+ and Mobile AMD-K6-III+ processors Model D provide the L2AAR register that allows for direct access to the L2 cache and L2 tag arrays. The L2AAR register in the Mobile AMD-K6-2+ and Mobile AMD-K6-III+ processors Model D are identical to the implementation of this register in the Model 9. See "Level-2 Cache Array Access Register (L2AAR)" on page 34. The Mobile AMD-K6-2+ processor's 128-Kbyte L2 cache is organized with four 32-Kbyte ways, each way containing 512 sectors. The Mobile AMD-K6-III+ processor's 256-Kbyte L2 cache is organized with four 64-Kbyte ways, each way containing 1024 sectors.

Enhanced Power Management Register (EPMR)

The Mobile AMD-K6-2+ and Mobile AMD-K6-III+ processors Model D are designed with Enhanced Power Management (EPM) features: Dynamic Bus Divisor control, and dynamic Voltage ID control.

The EPMR register of the Mobile AMD-K6-2+ and Mobile AMD-K6-III+ processors Model D (see Figure 19 on page 43) defines the base address for a 16-byte block of I/O address space. Enabling the EPMR allows software to access the EPM 16-byte I/O block, which contains bits for enabling, controlling, and monitoring the EPM features.

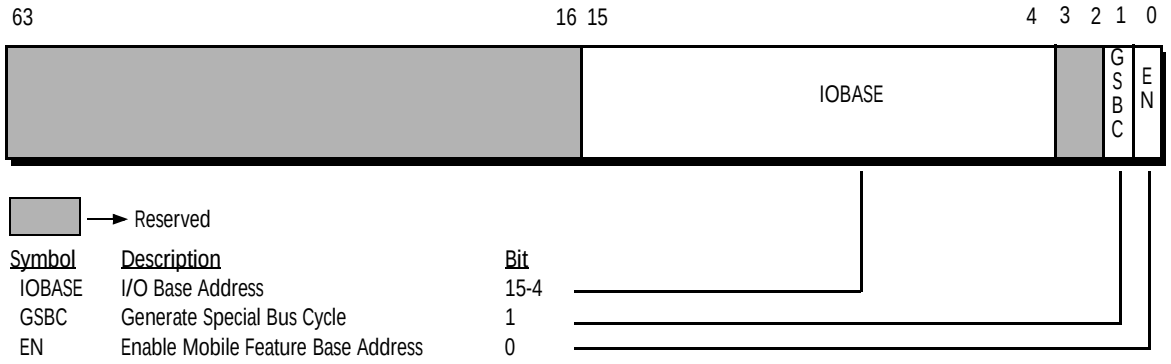


Figure 19. Enhanced Power Management Register (EPMR)—MSR C000_0086h (Model D)

Table 18. Enhanced Power Management Register (EPMR) Definition (Model D)

Bit	Description	R/W	Function
63–16	Reserved	R	All reserved bits are always read as 0.
15-4	I/O BASE Address (IOBASE)	R/W	IOBASE defines a base address for a 16-byte block of I/O address space accessible for enabling, controlling, and monitoring the EPM features.
3-2	Reserved	R	All reserved bits are always read as 0.
1	Generate Special Bus Cycle (GSBC)	R/W	This bit controls whether a special bus cycle is generated upon dword accesses within the EPM 16-byte block. If set to 1, an EPM special bus cycle is generated, where BE[7:0]# = BFh and A[4:3] = 00b.
0	Enable Mobile Feature Base Address (EN)	R/W	This bit controls access to the mapped I/O address space for the EPM features. Clearing this bit does not affect the state of bits defined in the EPM 16-byte I/O block.

Notes:
All bits default to 0 when RESET is asserted.

EPM 16-Byte I/O Block

The EPM 16-byte I/O block contains one 4-byte field—Bus Divisor and Voltage ID Control (BVC)—for enabling, controlling, and monitoring the EPM features (see Figure 20). All accesses to the EPM 16-byte I/O block must be aligned dword accesses. Except for the EPM special bus cycle, valid accesses to the EPM 16-byte block do not generate I/O bus cycles, while non-aligned and non-dword accesses are passed to the I/O bus.

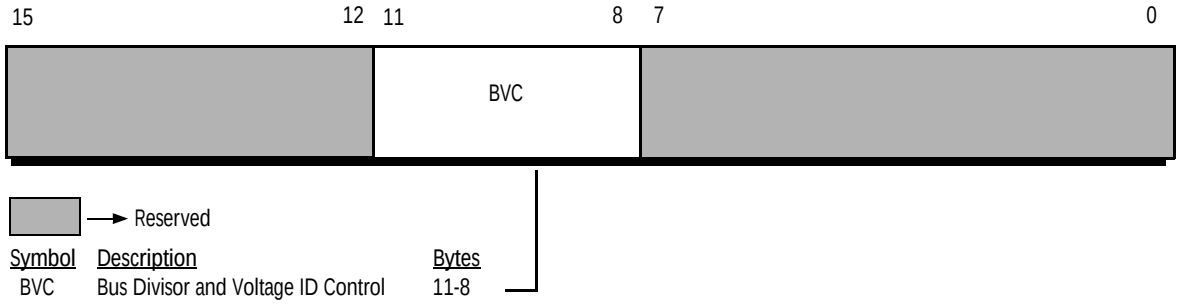


Figure 20. EPM 16-Byte I/O Block (Model D)

Table 19 defines the function of the byte-field within the EPM 16-byte I/O block mapped by the EPMR.

Table 19. EPM 16-Byte I/O Block Definition (Model D)

Byte	Description	R/W	Function
15-12	Reserved	R	All reserved bits are always read as 0.
11-8	Bus Divisor and Voltage ID Control (BVC)	R/W	The bit fields within the BVC bytes allow software to change the processor bus divisor and core voltage.
7-0	Reserved	R	All reserved bits are always read as 0.
Notes: *All bits default to 0 when RESET is asserted.*			

BVC. Figure 21 shows the format and Table 20 defines the function of each bit of the BVC field located within the EPM 16-byte I/O block mapped by the EPMR.

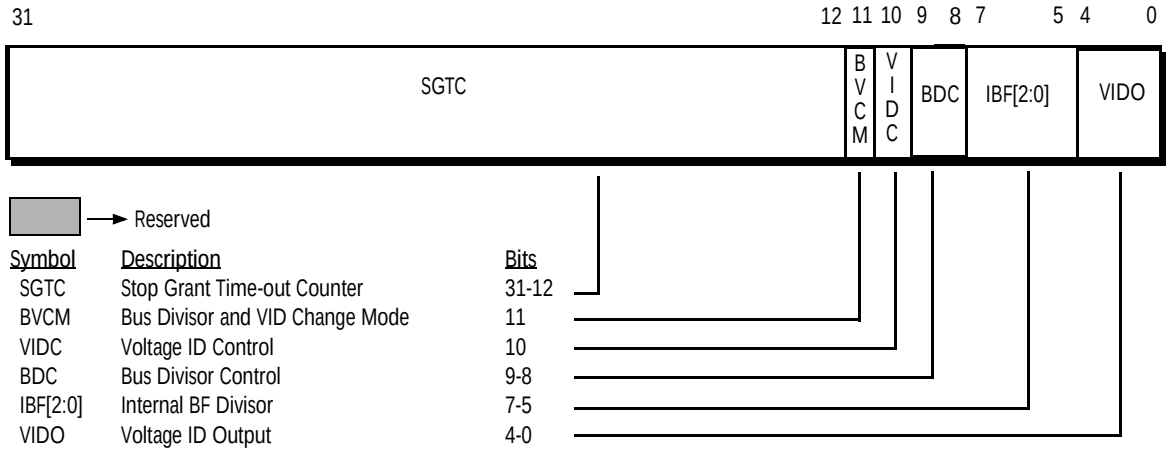


Figure 21. Bus Divisor and Voltage ID Control (BVC) Field (Model D)

Table 20. Bus Divisor and Voltage ID Control (BVC) Definition (Model D)

Bit	Description	R/W	Function
31-12	Stop Grant Time-out Counter (SGTC)	W	Writing a non-zero value to this field causes the processor to enter the EPM Stop Grant state internally. This 20-bit value is multiplied by 4096 to determines the duration of the EPM Stop Grant state, measured in processor bus clocks.
11	Bus Divisor and VID Change Mode (BVCM)	R/W	This bit controls the mode in which the bus-divisor and the voltage control bits are allowed to change. If BVCM=0, the Bus Divisor and Voltage ID changes take effect only upon entering the EPM Stop Grant state as a result of the SGTC field being programmed. BVCM=1 is reserved.
10	Voltage ID Control (VIDC)	R/W	This bit controls the mode of Voltage ID control. If VIDC=0, the processor VID[4:0] pins are unchanged upon entering the EPM Stop Grant state. If VIDC=1, the processor VID[4:0] pins are programmed to the VIDO value upon entering the EPM Stop Grant state. *BIOS should initialize this bit to 1 during the POST routine.*
9-8	Bus Divisor Control (BDC)	R/W	This 2-bit field controls the mode of Bus Divisor control. If BDC[1:0]=00b, the BF[2:0] pins are sampled at the falling edge of RESET. If BDC[1:0]=1xb, the IBF[2:0] field is sampled upon entering the EPM Stop Grant state. BDC[1:0]=01b is reserved. *BIOS should initialize these bits to 10b during the POST routine.*
7-5	Internal BF Divisor (IBF[2:0])	R/W	If BDC[1:0]=1xb, the processor EBF[2:0] field of the PSOR is programmed to the IBF[2:0] value upon entering the EPM Stop Grant state.
Notes: *All bits default to 0 when RESET is asserted, except the VIDO bits which default to 01010b.*			

Table 20. Bus Divisor and Voltage ID Control (BVC) Definition (Model D) (continued)

Bit	Description	R/W	Function
4-0	Voltage ID Output (VIDO)	R/W	This 5-bit value is driven out on the processor VID[4:0] pins upon entering the EPM Stop Grant state if the VIDC bit=1. These bits are initialized to 01010b and driven on the processor VID[4:0] pins at RESET.
Notes: *All bits default to 0 when RESET is asserted, except the VIDO bits which default to 01010b.*			

Note: *The EPM Stop Grant state is a low-power, clock-control state entered by writing a non-zero value to the SGTC field for the purpose of changing the internal EBF[2:0] field of the PSOR and VID[4:0] pin states. System-initiated inquire (snoop) cycles are not supported and must be prevented during the EPM Stop Grant clock control state.*

Mobile AMD Processor Recognition

Due to the increasing number of choices available in the x86 processor marketplace, the need for a simple way for hardware and software to identify the type of processor and its feature set has become critical. The CPUID instruction was added to the x86 instruction set for this purpose.

The CPUID instruction provides complete information about the processor (vendor, type, name, etc.) and its capabilities (features). After detecting the processor and its capabilities, software can be accurately tuned to the system for maximum performance and benefit to users. For example, game software can test the performance level available from a particular processor by detecting the type or speed of the processor. If the performance level is high enough, the software can enable additional capabilities or more advanced algorithms. Another example involves testing for the presence of 3DNow!™ or MMX™ instructions on the processor. If the software finds these features present when it checks the feature bits, it can utilize these more powerful extensions for dramatically better performance on new multimedia software. The current software development kit (SDK) contains code for determining the speed of AMD processors. The SDK is available at the following url:

http://www.amd.com/products/cpg/bin

CPUID Instruction Overview

Software operating at any privilege level can execute the CPUID instruction to identify the processor and its feature set. In addition, the CPUID instruction implements multiple functions, each providing different information about the processor, including the vendor, model number, revision (stepping), features, cache organization, and processor name. The multiple-function approach allows the CPUID instruction to return a complete picture about the type of processor and its capabilities—more detailed information than could be returned by a single function. In addition to gathering all the information by calling multiple functions, the CPUID

instruction provides the flexibility of making only one call to obtain the specific data requested.

The functions are divided into two types: **standard functions** and **extended functions**. Standard functions provide a simple method for software to access information common to all x86 processors. Extended functions provide information on extensions specific to a vendor's processor (for example, AMD's processors).

The flexibility of the CPUID instruction allows for the addition of new CPUID functions in future generations of processors. "Appendix A" on page 60 contains a detailed description of the CPUID instruction.

Testing for the CPUID Instruction

Beginning with the Mobile AMD-K6 processor Model 7, all Mobile AMD processors support the CPUID instruction. To use the CPUID instruction, software must first determine if the processor supports the CPUID instruction. CPUID support is determined in one of the following ways:

- Execute the CPUID instruction and check whether an illegal instruction exception occurs. If an exception occurs, the processor does not have CPUID support.
- Check if the ID bit (bit 21) of the EFLAGS register is writable. If the bit is writable (that is, it can be modified), the CPUID instruction is supported.

The operating system (OS) environment determines which approach is more appropriate. These techniques are described in the following sections.

Illegal Instruction Exception Method

This technique requires a way for a user program to detect and handle illegal instruction exceptions. Where such capabilities are present, this method represents a reliable way of detecting support for the CPUID instruction. The CPUID sample code starting on page 57 uses this approach.

EFLAGS ID-Bit Method

This technique retrieves the contents of EFLAGS using the PUSHFD instruction, toggles the ID bit, and uses the POPFD instruction to write the modified value of the ID bit into the EFLAGS register. It then retrieves the contents of EFLAGS using a second PUSHFD instruction and checks whether the

value of the ID bit differs from the original value. If the value has changed, the CPUID instruction is available for identifying the processor and its features. The following code sample demonstrates the way a program uses the PUSHFD and POPFD instructions to test the ID bit.

```
pushfd                  ; Save EFLAGS to stack
pop   eax               ; Store EFLAGS in EAX
mov   ebx, eax          ; Save in EBX for testing later
xor   eax, 00200000h    ; Switch bit 21
push  eax               ; Copy changed value to stack
popfd                   ; Save changed EAX to EFLAGS
pushfd                  ; Push EFLAGS to top of stack
pop   eax               ; Store EFLAGS in EAX
cmp   eax, ebx          ; See if bit 21 has changed
jz    NO_CPUID          ; If no change, no CPUID
```

A potential problem with this approach is that an interrupt or a trap (such as a debug trap) can occur between the POPFD and the following PUSHFD, and that the interrupt or trap handler code destroys the value of the ID bit. Where possible, the above code should be preceded by a CLI instruction and followed by an STI instruction, which ensures that no interrupts occur between the POPFD and the PUSHFD. However, traps can still occur, even if the code is preceded by a CLI instruction and followed by an STI instruction.

Using CPUID Functions

When software uses the CPUID instruction to identify a processor, it is important that it uses the instruction appropriately. The instruction has been defined to make it easy to identify the type and features of x86 processors manufactured by many different vendors.

The standard functions (EAX=0 and EAX=1) are the same for all processors. Having standard functions simplifies software's task of testing for and implementing features common to x86 processors. Software can test for these features and, as new x86 processors are released, benefit from these capabilities immediately.

Extended functions are specific to a vendor's processor. These functions provide additional information about AMD processors that software can use to identify enhanced features and functions. To test for extended functions, software checks

for a value of at least 8000_0001h in the EAX register returned by function 8000_0000h.

Within AMD's family of processors, different members can execute a different number of functions. Table 21 summarizes the CPUID functions currently implemented on AMD processors.

Table 21. Summary of CPUID Functions in Mobile AMD Processors

Standard Function	Extended Function	Description	Mobile AMD-K6® Processor (Model 7)	Mobile AMD-K6®-2 & Mobile AMD-K6-2-P Processors (Model 8)	Mobile AMD-K6®-III-P Processor (Model 9)	Mobile AMD-K6-2+ & Mobile AMD-K6-III+ Processors (Model D)
0	—	Vendor String and Largest Standard Function Value	X	X	X	X
1	—	Processor Signature and Standard Feature Bits	X	X	X	X
—	8000_0000h	Largest Extended Function Value	X	X	X	X
—	8000_0001h	Extended Processor Signature and Extended Feature Bits	X	X	X	X
—	8000_0002h	Processor Name	X	X	X	X
—	8000_0003h	Processor Name	X	X	X	X
—	8000_0004h	Processor Name	X	X	X	X
—	8000_0005h	L1 TLB* Cache Information	X	X	X	X
—	8000_0006h	L2 TLB Cache Information	—	—	X	X
—	8000_0007h	Enhanced Power Management	—	—	—	X

Note:

Future versions of these processors may implement additional functions.

"Appendix A" on page 60 contains detailed descriptions of the functions.

* TLB = translation lookaside buffer.

Identifying the Processor's Vendor

Software must execute the standard function EAX=0. The CPUID instruction returns a 12-character string that identifies the processor's vendor. The instruction also returns the largest

standard function input value defined for the CPUID instruction on the processor.

For AMD processors, function 0 returns a vendor string of "AuthenticAMD". This string informs the software to follow AMD's definition for subsequent CPUID functions and the registers returned for those functions.

Once the software identifies the processor's vendor, it knows the definition for all the functions supplied by the CPUID instruction. By using these functions, the software obtains the processor information needed to properly tune its functionality to the capabilities of the processor.

Testing For Extended Functions

Software must test for extended functions with function 8000_0000h. The EAX register returns the largest extended function input value defined for the CPUID instruction on the processor. If this value is at least 8000_0001h, extended functions are supported.

With one exception, the AMD extended feature flags include all the information provided in the standard feature flags as well as indicators for the additional AMD processor-specific feature enhancements. The duplication of standard feature bits within the extended feature bits can minimize the number of function calls required by software. The exception is bit 11, which indicates that the SYSENTER and SYSEXIT instructions are supported in the standard features and that the SYSCALL and SYSRET instructions are supported in the extended features.

Determining the Processor Signature

Standard function 1 (EAX=1) of the CPUID instruction returns the standard processor signature and feature bits. The standard processor signature is returned in the EAX register and provides information regarding the specific revision (stepping) and model of the processor and the instruction family level supported by the processor. The revision level can be used to determine if the processor supports specific features. However, it is not recommended that the revision level be used in this

manner unless this information is not available through the standard or extended feature bits.

All Mobile AMD-K6 processor models belong to instruction family 5 (as returned in EAX by function 1). All AMD Athlon processor models belong to instruction family 6 (as returned in EAX by function 1). Figure 22 on page 52 shows the contents of the EAX register obtained by function 1. Table 22 on page 52 summarizes the specific processor signature values returned for AMD processors.

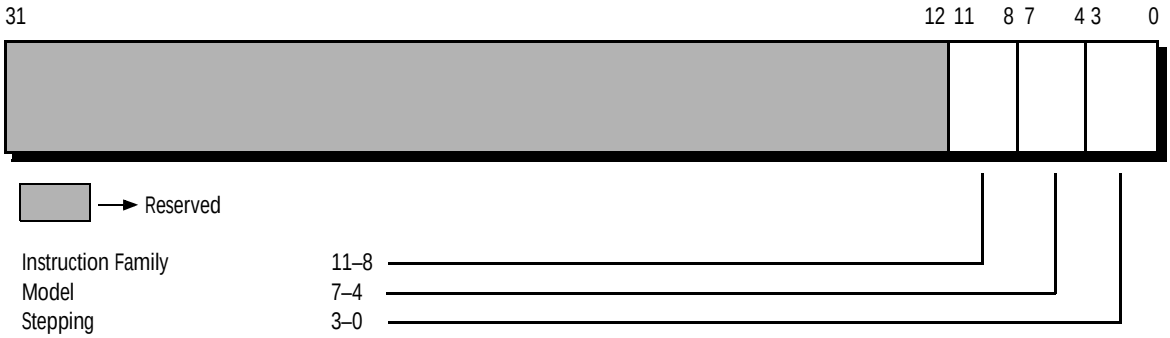


Figure 22. Contents of EAX Register Returned by Function 1

Table 22. Summary of Processor Signatures for Mobile AMD Processors

Processor	Instruction Family	Model	Stepping ID
Mobile AMD-K6® Processor (Model 7)	0101b (5h)	0111b (7h)	xxxx[1]
Mobile AMD-K6®-2 Processor (Model 8) Mobile AMD-K6®-2-P Processor (Model 8)	0101b (5h)	1000b (8h)	xxxx[1]
Mobile AMD-K6®-III-P Processor (Model 9)	0101b (5h)	1001b (9h)	xxxx[1]
Mobile AMD-K6®-2+ Processor (Model D) Mobile AMD-K6®-III+ Processor (Model D)	0101b (5h)	1101b (Dh)	xxxx[1]

Notes:
1. Contact your AMD representative for the latest stepping information.

Identifying Supported Features

The feature bits are returned in the EDX register for two CPUID functions—standard function 1 and extended function 8000_0001h. Each bit corresponds to a specific feature and

indicates if that feature is present on the processor. Table 23 summarizes the standard and extended feature bits.

Table 23. Summary of Standard and Extended Feature Bits

Bit[1]	Feature	Description	Standard[2]	Extended[2]
0	Floating-Point Unit	A floating-point unit is available.	1	1
1	Virtual Mode Extensions	Virtual mode extensions are available.	1	1
2	Debugging Extensions	I/O breakpoint debug extensions are supported.	1	1
3	PSE (Page Size Extensions)	4-Mbyte pages are supported.	1	1
4	Time Stamp Counter (with RDTSC and CR4 disable bit)	A time stamp counter is available in the processor, and the RDTSC instruction is supported.	1	1
5	K86™ Family of Processors' Model-Specific Registers (with RDMSR and WRMSR)	The K86 model-specific registers are available in the processor, and the RDMSR and WRMSR instructions are supported.	1	1
6	PAE (Page Address Extensions)	Page address extensions are supported using an 8-byte directory entry.	1	1
7	MCE (Machine Check Exception)	The machine check exception is supported.	1	1
8	CMPXCHG8B Instruction	The CMPXCHG8B instruction is supported.	1	1
9	APIC	A local APIC unit is available.	1	1
11	SYSENTER/SYSEXIT Instructions	The SYSENTER and SYSEXIT instructions are supported.	1	0
	SYSCALL and SYSRET Instructions	The SYSCALL and SYSRET instructions and associated extensions are supported.	0	1
12	MTRR (Memory Type Range Registers)	Memory type range registers are available.	1	1
13	Global Paging Extension	Global paging extensions are available.	1	1
14	MCA (Machine Check Architecture)	Machine check architecture is supported	1	1
15	Conditional Move Instructions	The conditional move instructions CMOV, FCMOV, and FCOMI are supported.	1	1
16	PAT (Page Attribute Table)	The Page attribute tables are supported.	1	1
17	PSE-36 (Page Size Extension)	Page size extensions for 36-bit addresses are supported using a 4-byte directory entry.	1	1
22	AMD Multimedia Instruction Extensions	AMD additions to the original MMX™ instruction set are supported.	0	1
23	MMX™ Instructions	The MMX instruction set is supported.	1	1
24	FXSAVE/FXRSTOR	Fast floating-point save and restore is supported.	1	1

Note:

1. *"Appendix A" on page 60 contains details on bit locations and values.*
2. *Bit definitions: 0 = No Support, 1 = Support*

Table 23. Summary of Standard and Extended Feature Bits (continued)

Bit[1]	Feature	Description	Standard[2]	Extended[2]
30	AMD 3DNow!™ Instruction Extensions	Extensions to the 3DNow! instruction set are supported.	0	1
31	3DNow! Instructions	3DNow! instructions are supported.	0	1

Note:
1. *"Appendix A" on page 60 contains details on bit locations and values.*
2. *Bit definitions: 0 = No Support, 1 = Support*

Before using any of the enhanced features added to the latest generation of processors, software should test each feature bit returned by functions 1 and 8000_0001h to identify the capabilities available on the processor. For example, software must test feature bit 23 to determine if the processor executes the MMX technology instructions. Attempting to execute an unavailable feature can cause errors and exceptions.

Bit 31, as returned by extended function 8000_0001h, designates the presence of 3DNow! technology. Other processor vendors have adopted this technology, so bit 31 is now considered an open standard. "Appendix A" on page 60, and "Appendix B" on page 69 contain details on bit locations and values.

Determining Instruction Set Support

It is preferable to use CPUID feature flags as much as possible, rather than deriving capabilities from vendor specifiers combined with CPUID model numbers.

The Mobile AMD-K6-2+ and Mobile AMD-K6-III+ processors add a new set of powerful extensions to the x86 instruction set—3DNow! extensions. See the *AMD Extensions to the 3DNow!™ and MMX™ Instruction Sets Manual*, order# 22466 for more information about these new instructions.

To simplify the detection of the new instructions and the original 3DNow! and MMX instructions, use the following algorithm. A code sample using the CPUID instruction to identify the processor and its features is available from AMD's website at http://www.amd.com/products/cpg/bin. There are other ways to implement detection besides the way shown in the sample.

CPUID Test

1. Establish that the processor has support for CPUID. See "Testing for the CPUID Instruction" on page 48.

Standard Function Test

2. Execute CPUID function 0, which returns the processor vendor string and the highest standard function supported. Save the vendor string for a later comparison. (See step 9.)
3. If step 2 indicates that the highest standard function is at least 1, execute CPUID function 1, which returns the standard feature flags in the EDX register.

MMX Test

4. If bit 23 of the standard feature flags is set to 1, MMX technology is supported. MMX instruction support is the basic minimum processor feature required to support other instruction extensions.

Optional SSE Test

5. Optionally, if bit 25 of the standard feature flags is set, the processor has streaming SIMD extensions (SSE) capabilities. Further qualification of SSE is done by checking for OS support. SSE support might be present in the processor, but not usable due to a lack of OS support for the additional architected registers.

Extended Functions Test

6. Execute CPUID extended function 8000_0000h. This function returns the highest extended function supported in EAX. If EAX=0, there is no support for extended functions.
7. If the highest extended function supported is at least 8000_0001h, execute CPUID function 8000_0001h. This function returns the extended feature flags in EDX.

3DNow! Test

8. If bit 31 of the extended feature flags is set to 1, the 3DNow! instructions are supported.

Vendor Check

9. If the previously saved vendor string (see step 2) contains “AuthenticAMD”, continue on to the next step.

3DNow! Extensions Test

10. If bit 30 of the extended feature flags is set to 1, the additions to the 3DNow! instruction set are supported.

MMX Extensions Test

11. If bit 22 of the extended feature flags is set to 1, the new multimedia enhancement instructions that augment the MMX instruction set are supported.

AMD Processor Signature (Extended Function)

Extended function 8000_0001h returns the Mobile AMD processor signature. The signature is returned in the EAX register and provides generation, model, and stepping information for AMD processors. Figure 23 shows the contents returned in the EAX register.

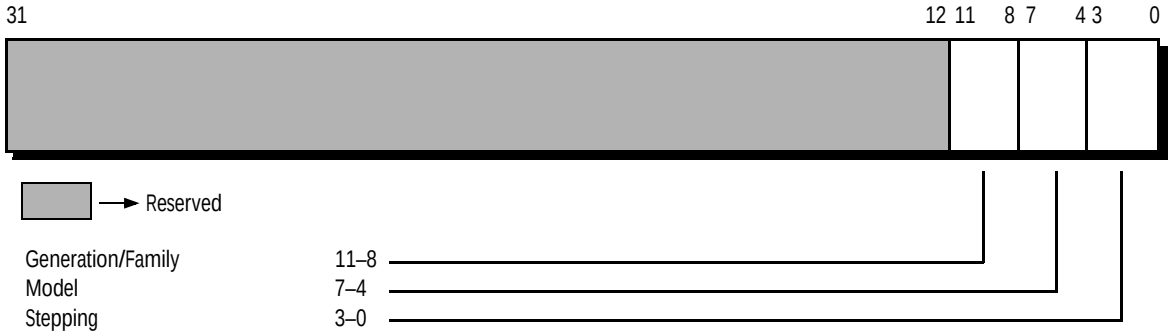


Figure 23. Contents of EAX Register Returned by Extended Function 8000_0001h

Displaying the Processor's Name

Extended functions 8000_0002h, 8000_0003h, and 8000_0004h return an ASCII string containing the name of the processor. These functions eliminate the need for software to search for the processor name in a lookup table, a process requiring a large block of memory and frequent updates. Instead, software can simply call these three functions to obtain the name string (48 ASCII characters in little-endian format) and display it on the screen. Although the name string can be up to 48 characters in length, shorter names have the remaining byte locations filled with the ASCII NULL character (00h). To simplify the display routines and avoid using screen space, software only needs to display characters until a NULL character is detected.

Displaying Cache Information

Extended functions 8000_0005h and 8000_0006h provide cache information for the processor. Some diagnostic software displays information about the system and the processor's configuration. It is common for this type of software to provide cache size and organization of information. Functions

8000_0005h and 8000_0006h provide a simple way for software to obtain information about the on-chip caches and Translation Lookaside Buffer (TLB) structures. The size and organization information is returned in the registers as described in "Appendix A" on page 60. Software can simply display these values, eliminating the need for large pieces of code to test the memory structures.

Determining Enhanced Power Management Information

Extended function 8000_0007h provides information regarding the processor's support for Enhanced Power Management (EPM) features. Based on the status of the EPM Flags, software can determine if the processor supports programmable Bus Frequency control and programmable Voltage ID control. A '1' for each bit indicates that the feature is supported, however, the feature must be enabled by software.

Sample Code

A code sample using the CPUID instruction to identify the processor and its features is available from AMD's website at http://www.amd.com/products/cpg/bin.

New Mobile AMD-K6® Processor Instructions

All models of the Mobile AMD-K6 processor implement the following new instruction set:

- MMX™ Instructions—57 new instructions for multimedia software. See the *AMD-K6® Processor Multimedia Technology Manual*, order# 20726 for more information.

All Mobile AMD-K6 processors, beginning with the Mobile AMD-K6-2 processor Model 8, implement the following new instructions:

- 3DNow!™ Instructions—21 new instructions for multimedia software. See the *3DNow!™ Technology Manual*, order# 21928 for more information.

- SYSCALL and SYSRET—See the *SYSCALL and SYSRET Instruction Specification Application Note*, order# 21086 for more information.

All Mobile AMD-K6-2+ and Mobile AMD-K6-III+ processors Model D implement the following new instructions:

- 3DNow!™ Instruction Extensions—5 new instructions for multimedia software. See the *AMD Extensions to the 3DNow!™ and MMX™ Instruction Sets Manual*, order# 22466 for more information.

Additional Considerations

Software Timing Dependencies Relative to Memory Controller Setup

Processors in the K86 family differ from other processors with regards to instruction latencies and the order or priority of processor bus cycles. Timing-dependent software that relies on the specific latencies of other processors should be re-tested for proper operation with the K86 processor. In addition, re-testing should be performed on components with variable timing (such as, memory modules, oscillators, and timers).

Particular attention should be paid to memory-setup subroutines that determine the type of DRAM in the system. Some chipsets may not tolerate a DRAM mode change (such as, EDO to SDRAM) on the same clock as a DRAM refresh cycle. For example some chipsets do not tolerate having its memory refresh enabled prior to changing memory mode types. Refresh should only be enabled after the memory type has been determined.

Note: *The BIOS for the K86 family of processors should enable the write allocate mechanisms only after performing any memory sizing or typing algorithms.*

Pipelining Support

All production models and steppings of the Mobile AMD-K6 processor support the WAELIM form of write allocate, which is the only form of write allocate that should be enabled. AMD does not recommend enabling the obsolete form of write

allocate (WCDE) because system performance can be degraded by doing so.

Early implementations of the Mobile AMD-K6 processor did not support the WHCR register and therefore did not support the WAELIM form of write allocate. WCDE was the only form of write allocate supported, which required the chipset to assert KEN# for cacheable memory write cycles. Because KEN# is sampled by the processor on the clock edge on which the first BRDY# or NA# is sampled asserted, some chipsets that supported the WCDE form of write allocate did not assert NA# during write cycles in order to prevent the processor from sampling KEN# before it was valid (in this case, BRDY# was used by the processor to sample KEN#). If NA# is not asserted during memory write cycles, then the processor does not fully take advantage of the potential performance gains that bus pipelining can achieve.

For proper functionality, always program the WCDE bit to 0 for Model 7. Models 8 and above do not support the WCDE bit.

Read-Only Memory

The processor's caches must be flushed prior to defining any area of memory as cacheable and read only. (The BIOS is typically "shadowed" into main memory and defined as cacheable and read only.) If the caches are not flushed, then a line that resides in the processor's cache that falls within a read-only area of memory can be written to, which would place the cache line in the modified state. If this modified line is subsequently replaced and written back to memory, then the system may hang (or other unpredictable effects may occur) because the writeback is directed to an area of memory defined as read only by the chipset.

Appendix A

CPUID

mnemonic	*opcode*	*description*
CPUID	0F A2h	Identify the processor and its feature set
Privilege:	none	
Registers Affected:	EAX, EBX, ECX, EDX	
Flags Affected:	none	
Exceptions Generated:	none	

The CPUID instruction is an application-level instruction that software executes to identify the processor and its feature set. This instruction offers multiple functions, each providing a different set of information about the processor. The CPUID instruction can be executed from any privilege level. Software can use the information returned by this instruction to tune its functionality for the specific processor and its features.

Not all processors implement the CPUID instruction. Therefore, software must test to determine if the instruction is present on the processor. If the ID bit (21) in the EFLAGS register is writeable, the CPUID instruction is implemented.

The CPUID instruction supports multiple functions. The information associated with each function is obtained by executing the CPUID instruction with the function number in the EAX register. Functions are divided into two types: standard functions and extended functions. Standard functions are found in the low function space, 0000_0000h–7FFF_FFFFh. In general, all x86 processors have the same standard function definitions.

Extended functions are defined specifically for processors supplied by the vendor listed in the vendor identification string. Extended functions are found in the high function space, 8000_0000h–8FFF_FFFFh. Because not all vendors have defined extended functions, software must test for their presence on the processor.

Standard Functions

Function 0 — Largest Standard Function Input Value and Vendor Identification String

Input: EAX = 0

Output: EAX = Largest function input value recognized by the CPUID instruction
EBX, EDX, ECX = Vendor identification string

This is a standard function found in all processors implementing the CPUID instruction. It returns two values. The first value is returned in the EAX register and indicates the largest standard function value recognized by the processor. The second value is the vendor identification string. This 12-character ASCII string is returned in the EBX, EDX, and ECX registers in little-endian format. AMD processors return a vendor identification string of “AuthenticAMD” as follows:

EBX				**EDX**				**ECX**			
h	t	u	A	i	t	n	e	D	M	A	c
68	74	75	41	69	74	6E	65	44	4D	41	63

Software uses the vendor identification string as follows:

- To identify the processor as an AMD processor
- To apply AMD’s definition of the CPUID instruction for all additional function calls

Function 1 — Processor Signature and Standard Feature Flags

Input: EAX = 1

Output: EAX = Processor Signature
EBX = Reserved
ECX = Reserved
EDX = Standard Feature Flags

Function 1 returns two values—the Processor Signature and the Standard Feature Flags. The processor signature is returned in the EAX register and identifies the specific processor by providing information on its type—instruction family, model, and revision (stepping). The information is formatted as follows:

- EAX[3–0] Stepping ID
- EAX[7–4] Model
- EAX[11–8] Instruction Family
- EAX[31–12] Reserved

The standard feature flags are returned in the EDX register and indicate the presence of specific features. In most cases, a "1" indicates the feature is present, and a "0" indicates the feature is not present. Table 24 contains a list of the currently defined standard feature flags for the Mobile AMD-K6 processors. Reserved bits will be used for new features as they are added.

Table 24. Standard Feature Flag Descriptions

Bit	Feature[1]	Mobile AMD-K6 Processor (Model 7)	Mobile AMD-K6-2 & Mobile AMD-K6-2-P Processors (Model 8)	AMD-K6-III-P Processor (Model 9)	Mobile AMD-K6-2+ & Mobile AMD-K6-III+ Processors (Model D)
0	Floating-Point Unit	1	1	1	1
1	Virtual Mode Extensions	1	1	1	1
2	Debugging Extensions	1	1	1	1
3	Page Size Extensions (4-Mbyte pages)	1	1	1	1
4	Time Stamp Counter (with RDTSC and CR4 disable bit)	1	1	1	1
5	K86™ Family of Processors' Model-Specific Registers (with RDMSR and WRMSR)	1	1	1	1
6	PAE (Page Address Extensions)	0	0	0	0
7	Machine Check Exception	1	1	1	1
8	CMPXCHG8B Instruction	1	1	1	1
9	APIC	0	0	0	0
10	*Reserved on all AMD processors*	0	0	0	0
11	SYSENTER/SYSEXIT	0	0	0	0
12	Memory Type Range Registers	0	0	0	0
13	Global Paging Extension	1[2]	1[2]	1	1
14	Machine Check Architecture	0	0	0	0
15	Conditional Move Instruction	0	0	0	0
16	PAT (Page Attribute Table)	0	0	0	0
17	PSE-36 (Page Size Extensions)	0	0	0	0
18–22	*Reserved on all AMD processors*	0	0	0	0
23	MMX™ Instructions	1	1	1	1
24	FXSAVE/FXRSTOR	0	0	0	0
25–31	*Reserved on all AMD processors*	0	0	0	0

Notes:

1. *Bit definitions: 0 = No Support, 1 = Support.*
2. *See Table 32 on page 69 for more information about Global Paging Extensions in the Mobile AMD-K6-2 processor Model 8.*

Extended Functions

Function 8000_0000h — Largest Extended Function Input Value

Input: EAX = 8000_0000h

Output: EAX = Largest function input value recognized by the CPUID instruction
EBX = Reserved
ECX = Reserved
EDX = Reserved

Function 8000_0000h returns a value in the EAX register that indicates the largest extended function value recognized by the processor.

Function 8000_0001h — AMD Processor Signature and Extended Feature Flags

Input: EAX = 8000_0001h

Output: EAX = AMD Processor Signature
EBX = Reserved
ECX = Reserved
EDX = Extended Feature Flags

Function 8000_0001h returns two values—the AMD Processor Signature and the Extended Feature Flags. The AMD processor signature is returned in the EAX register and identifies the specific processor by providing information regarding its type—generation/family, model, and revision (stepping). The information is formatted as follows:

- EAX[3–0] Stepping ID
- EAX[7–4] Model
- EAX[11–8] Generation/Family
- EAX[31–12] Reserved

The extended feature flags are returned in the EDX register and indicate the presence of specific features found in AMD processors. In most cases, a ‘1’ indicates the feature is present, and a ‘0’ indicates the feature is not present. Table 25 on page 64 contains a list of the currently defined extended feature flags for the Mobile AMD-K6 processors. Reserved bits will be used for new features as they are added.

Table 25. Extended Feature Flag Descriptions

Bit	Feature[1]	Mobile AMD-K6 Processor (Model 7)	Mobile AMD-K6-2 & Mobile AMD-K6-2-P Processors (Model 8)	AMD-K6-III-P Processor (Model 9)	Mobile AMD-K6-2+ & Mobile AMD-K6-III+ Processors (Model D)
0	Floating-Point Unit	1	1	1	1
1	Virtual Mode Extensions	1	1	1	1
2	Debugging Extensions	1	1	1	1
3	Page Size Extensions (4-Mbyte Pages)	1	1	1	1
4	Time Stamp Counter (with RDTSC and CR4 disable bit)	1	1	1	1
5	K86™ Family of Processors' Model-Specific Registers (with RDMSR and WRMSR)	1	1	1	1
6	PAE (Page Address Extensions)	0	0	0	0
7	Machine Check Exception	1	1	1	1
8	CMPXCHG8B Instruction	1	1	1	1
9	APIC	0	0	0	0
10	*Reserved on all AMD processors*	0	0	0	0
11	SYSCALL and SYSRET Instructions	1	1	1	1
12	Memory Type Range Registers	0	0	0	0
13	Global Paging Extension	1	1	1	1
14	Machine Check Architecture	0	0	0	0
15	Conditional Move Instruction	0	0	0	0
16	PAT (Page Attribute Table)	0	0	0	0
17	PSE-36 (Page Size Extensions)	0	0	0	0
18–21	*Reserved on all AMD processors*	0	0	0	0
22	AMD MMX™ Instruction Extensions	0	0	0	0
23	MMX Instructions	1	1	1	1
24	FXSAVE/FXRSTOR	0	0	0	0
25–29	*Reserved on all AMD processors*	0	0	0	0
30	AMD 3DNow!™ Instruction Extensions	0	0	0	1
31	3DNow! Instructions	1	1	1	1

Notes:
1. *Bit definitions: 0 = No Support, 1 = Support.*

Functions 8000_0002h, 8000_0003h, and 8000_0004h — Processor Name String

Input: EAX = 8000_0002h, 8000_0003h, or 8000_0004h

Output: EAX = Processor Name String
EBX = Processor Name String
ECX = Processor Name String
EDX = Processor Name String

Functions 8000_0002h, 8000_0003h, and 8000_0004h each return part of the processor name string in the EAX, EBX, ECX, and EDX registers. These three functions use the four registers to return an ASCII string of up to 48 characters in little endian format. For example, function 8000_0002h returns the first 16 characters of the processor name. The first character resides in the least significant byte of EAX, and the last character (of this group of 16) resides in the most significant byte of EDX. The NULL character (ASCII 00h) is used to indicate the end of the processor name string. This feature is useful for processor names that require fewer than 48 characters.

Function 8000_0005h — L1 Cache Information

Input: EAX = 8000_0005h

Output: EAX = Reserved
EBX = TLB Information
ECX = L1 Data Cache Information
EDX = L1 Instruction Cache Information

Function 8000_0005h returns information about the processor's on-chip L1 caches and associated TLBs. Tables 26, 27, and 28 provide the format for the information returned by the 8000_0005h function.

Table 26. EBX Format Returned by Function 8000_0005h

	Data TLB		Instruction TLB	
	Associativity*	# Entries	Associativity*	# Entries
EBX	Bits 31–24	Bits 23–16	Bits 15–8	Bits 7–0

Note:
* **See "Associativity for L1 Caches and L1 TLBs" on page 68 for more information.*

Table 27. ECX Format Returned by Function 8000_0005h

	L1 Data Cache			
	Size (Kbytes)	**Associativity***	**Lines per Tag**	**Line Size (bytes)**
ECX	Bits 31–24	Bits 23–16	Bits 15–8	Bits 7–0

Note:
* **See "Associativity for L1 Caches and L1 TLBs" on page 68 for more information.*

Table 28. EDX Format Returned by Function 8000_0005h

	L1 Instruction Cache			
	Size (Kbytes)	**Associativity***	**Lines per Tag**	**Line Size (bytes)**
EDX	Bits 31–24	Bits 23–16	Bits 15–8	Bits 7–0

Note:
* **See "Associativity for L1 Caches and L1 TLBs" on page 68 for more information.*

Function 8000_0006h — L2 Cache Information

This function is only available on Mobile AMD-K6 processors Models 9 and D.

Input: EAX = 8000_0006h

Output: EAX = Reserved
EBX = Reserved
ECX = L2 Unified Cache Information
EDX = Reserved

Function 8000_0006h returns information about the processor's L2 cache. "ECX Format Returned by Function 8000_0006h" on page 66 provides the format for the information returned by the 8000_0006h function.

Table 29. ECX Format Returned by Function 8000_0006h

	L2 Cache			
	Size (Kbytes)	**Associativity***	**Lines per Tag**	**Line Size (bytes)**
ECX	Bits 31–16	Bits 15–12	Bits 11–8	Bits 7–0

Note:
* *See "Associativity for L2 Cache" on page 68 for more information.*

Function 8000_0007h — Enhanced Power Management (EPM) Information

This function is available on Mobile AMD-K6-2+ and Mobile AMD-K6-III+ processors, Model D.

Input: EAX = 8000_0007h

Output: EAX = Reserved
EBX = Reserved
ECX = Reserved
EDX = EPM Flags

Function 8000_0007h returns information about the processor's enhanced power management support. Table 30 provides the format for the information returned by the 8000_0007h function.

Table 30. EDX Format Returned by Function 8000_0007h

	Enhanced Power Management			
	Reserved	**Voltage ID Control**	**Bus Divisor Control**	**Reserved**
EDX	Bits 31–3	Bit 2	Bit 1	Bit 0

Note:
A '1' indicates the feature is present, however the feature must still be enabled by software.

Associativity Field Definitions

This section describes the values returned in the associativity fields.

Associativity for L1 Caches and L1 TLBs

The associativity fields for the L1 data cache, L1 instruction cache, L1 data TLB, and L1 instruction TLB are all 8 bits wide. Except for 00h (Reserved) and FFh (Full), the number returned in the associativity field represents the actual number of ways, with a range of 01h through FEh. For example, a returned value of 02h indicates 2-way associativity and a returned value of 04h indicates 4-way associativity.

Associativity for L2 Cache

The associativity field for the L2 cache is 4 bits wide. Table 31 shows the value returned in the associativity field.

Table 31. Associativity Values For L2 Cache

Bits 15–12	Associativity
0000b	L2 off
0001b	Direct mapped
0010b	2-way
0011b	Reserved
0100b	4-way
0101b	Reserved
0110b	8-way
0111b	Reserved
1000b	16-way
1001b	Reserved
1010b	Reserved
1011b	Reserved
1100b	Reserved
1101b	Reserved
1110b	Reserved
1111b	Full

Appendix B

Values Returned By the CPUID Instruction

Table 32 contains all the values returned for Mobile AMD-K6 processors by the CPUID instruction.

Table 32. Values Returned By Mobile AMD-K6® Processors

Function Register	Mobile AMD-K6 Processor (Model 7)	Mobile AMD-K6-2 & Mobile AMD-K6-2-P Processors (Model 8)	Mobile AMD-K6-III-P Processor (Model 9)	Mobile AMD-K6-2+ & Mobile AMD-K6-III+ Processors (Model D)
Function: 0				
EAX	0000_0001h	0000_0001h	0000_0001h	0000_0001h
EBX	6874_7541h	6874_7541h	6874_7541h	6874_7541h
ECX	444D_4163h	444D_4163h	444D_4163h	444D_4163h
EDX	6974_6E65h	6974_6E65h	6974_6E65h	6974_6E65h
Function: 1				
EAX	0000_057Xh	0000_058Xh	0000_059Xh	0000_05DXh
EBX	Reserved	Reserved	Reserved	Reserved
ECX	Reserved	Reserved	Reserved	Reserved
EDX	0080_01BFh	0080_21BFh	0080_21BFh	0080_21BFh
Function: 8000_0000h				
EAX	8000_0005h	8000_0005h	8000_0006h	8000_0007h
EBX	Reserved	Reserved	Reserved	Reserved
ECX	Reserved	Reserved	Reserved	Reserved
EDX	Reserved	Reserved	Reserved	Reserved
Function: 8000_0001h				
EAX	0000_067Xh	0000_068Xh	0000_069Xh	0000_06DXh
EBX	Reserved	Reserved	Reserved	Reserved
ECX	Reserved	Reserved	Reserved	Reserved
EDX	0080_05BFh	8080_29BFh	8080_29BFh	C080_29BFh

1. *Extended functions 8000_0002h, 8000_0003h, and 8000_0004h each return part of the processor name string. Some Mobile AMD-K6-2 processors may have the following name string: function 8000_0002h, ECX = 322D_296Dh and EDX = 6F72_5020h, and function 8000_0003h, EAX = 7373_6563h and EBX = 0000_726Fh.*
2. *Extended functions 8000_0002h, 8000_0003h, and 8000_0004h each return part of the processor name string. Some Mobile AMD-K6-III-P processors may have the following name string: function 8000_0002h, ECX = 492D_296Dh and EDX = 5020_4949h, and function 8000_0003h, EAX = 6563_6F72h and EBX = 726F_7373h.*
3. *Extended function 8000_0006h returns the processor L2 cache information. For the Mobile AMD-K6-2+ processor Model D, ECX = 0080_4220h. For the Mobile AMD-K6-III+ processor Model D, ECX = 0100_4220h.*

Table 32. Values Returned By Mobile AMD-K6® Processors (continued)

Function Register	Mobile AMD-K6 Processor (Model 7)	Mobile AMD-K6-2 & Mobile AMD-K6-2-P Processors (Model 8)	Mobile AMD-K6-III-P Processor (Model 9)	Mobile AMD-K6-2+ & Mobile AMD-K6-III+ Processors (Model D)
Function: 8000_0002h				
EAX	2D44_4D41h	2D44_4D41h	2D44_4D41h	2D44_4D41h
EBX	6D74_364Bh	7428_364Bh	7428_364Bh	7428_364Bh
ECX	202F_7720h	3320_296Dh[1]	3320_296Dh[2]	492D_296Dh
EDX	746C_756Dh	7270_2044h[1]	5020_2B44h[2]	6020_4949h
Function: 8000_0003h				
EAX	6465_6D69h	7365_636F[1]h	6563_6F72h[2]	6563_6F72h
EBX	6520_6169h	0072_6F73[1]h	726F_7373h[2]	726F_7373h
ECX	6E65_7478h	0000_0000h	0000_0000h	0000_0000h
EDX	6E6F_6973h	0000_0000h	0000_0000h	0000_0000h
Function: 8000_0004h				
EAX	0000_0073h	0000_0000h	0000_0000h	0000_0000h
EBX	0000_0000h	0000_0000h	0000_0000h	0000_0000h
ECX	0000_0000h	0000_0000h	0000_0000h	0000_0000h
EDX	0000_0000h	0000_0000h	0000_0000h	0000_0000h
Function: 8000_0005h				
EAX	Reserved	Reserved	Reserved	Reserved
EBX	0280_0140h	0280_0140h	0280_0140h	0280_0140h
ECX	2002_0220h	2002_0220h	2002_0220h	2002_0220h
EDX	2002_0220h	2002_0220h	2002_0220h	2002_0220h
Function: 8000_0006h				
EAX	Undefined	Undefined	Reserved	Reserved
EBX	Undefined	Undefined	Reserved	Reserved
ECX	Undefined	Undefined	0100_4220h	0080_4220h[3]
EDX	Undefined	Undefined	Reserved	Reserved

1. *Extended functions 8000_0002h, 8000_0003h, and 8000_0004h each return part of the processor name string. Some Mobile AMD-K6-2 processors may have the following name string: function 8000_0002h, ECX = 322D_296Dh and EDX = 6F72_5020h, and function 8000_0003h, EAX = 7373_6563h and EBX = 0000_726Fh.*
2. *Extended functions 8000_0002h, 8000_0003h, and 8000_0004h each return part of the processor name string. Some Mobile AMD-K6-III-P processors may have the following name string: function 8000_0002h, ECX = 492D_296Dh and EDX = 5020_4949h, and function 8000_0003h, EAX = 6563_6F72h and EBX = 726F_7373h.*
3. *Extended function 8000_0006h returns the processor L2 cache information. For the Mobile AMD-K6-2+ processor Model D, ECX = 0080_4220h. For the Mobile AMD-K6-III+ processor Model D, ECX = 0100_4220h.*

Table 32. Values Returned By Mobile AMD-K6® Processors (continued)

Function Register	Mobile AMD-K6 Processor (Model 7)	Mobile AMD-K6-2 & Mobile AMD-K6-2-P Processors (Model 8)	Mobile AMD-K6-III-P Processor (Model 9)	Mobile AMD-K6-2+ & Mobile AMD-K6-III+ Processors (Model D)
Function: 8000_0007h				
EAX	Undefined	Undefined	Undefined	Reserved
EBX	Undefined	Undefined	Undefined	Reserved
ECX	Undefined	Undefined	Undefined	Reserved
EDX	Undefined	Undefined	Undefined	0000_0007h

1. *Extended functions 8000_0002h, 8000_0003h, and 8000_0004h each return part of the processor name string. Some Mobile AMD-K6-2 processors may have the following name string: function 8000_0002h, ECX = 322D_296Dh and EDX = 6F72_5020h, and function 8000_0003h, EAX = 7373_6563h and EBX = 0000_726Fh.*
2. *Extended functions 8000_0002h, 8000_0003h, and 8000_0004h each return part of the processor name string. Some Mobile AMD-K6-III-P processors may have the following name string: function 8000_0002h, ECX = 492D_296Dh and EDX = 5020_4949h, and function 8000_0003h, EAX = 6563_6F72h and EBX = 726F_7373h.*
3. *Extended function 8000_0006h returns the processor L2 cache information. For the Mobile AMD-K6-2+ processor Model D, ECX = 0080_4220h. For the Mobile AMD-K6-III+ processor Model D, ECX = 0100_4220h.*