# AMD-K6™

## MMX™ Enhanced Processor

## Revision Guide

Publication # **21266** Rev: **E** Amendment/**0**
Issue Date: **August 1997**

# AMD-K6™ MMX™ Enhanced Processor Revision Guide

The purpose of the *AMD-K6™ MMX™ Enhanced Processor Revision Guide* is to communicate updated product information on the AMD-K6 processor to designers of computer systems and software developers. This guide consists of four major sections:

- **Product Marking Identification:** This section provides product types, product revisions, OPNs (Ordering Part Numbers), and product marking information.
- **Product Errata:** This section provides a detailed description of product errata, including potential effects on system operation and suggested workarounds. An erratum is defined as a deviation from the product's specification.
- **Specification Changes/Clarifications:** This section provides changes, additions, and clarifications to product specifications.
- **Technical and Documentation Support:** This section provides a listing of available technical support resources. It also lists corrections, modifications, and clarifications to listed documents.

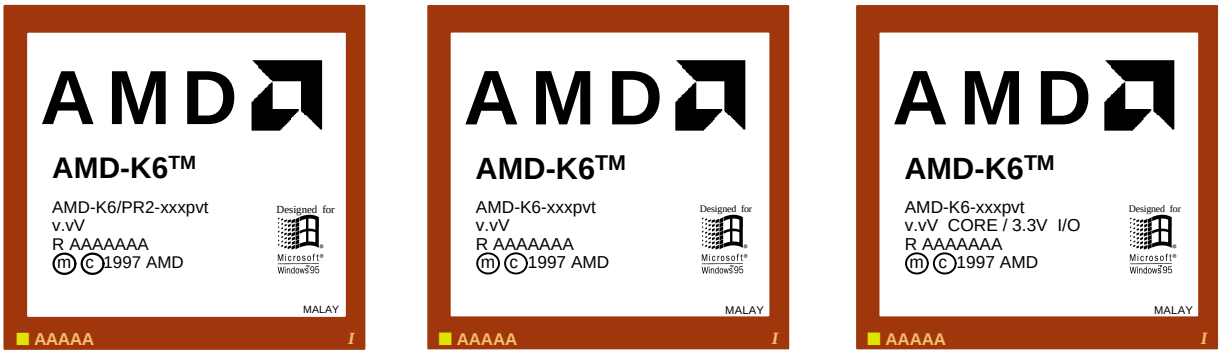
## Revision Guide Policy

At times, AMD identifies deviations or changes to the specification of the AMD-K6 MMX enhanced processor. These are documented in the *AMD-K6 MMX Enhanced Processor Revision Guide* as errata or specification changes/clarifications and are available to anyone who requests the information. The descriptions are written to assist system and software designers in using the AMD-K6 processor. In addition, any corrections to AMD's published documentation on the AMD-K6 processor are included. The errata and specification changes are the result of extensive testing and validation that is done for all AMD products. AMD works closely with system and software designers to ensure the appropriate workarounds or changes are implemented to avoid impact to PC users.

The *AMD-K6 MMX Enhanced Processor Revision Guide* is made publicly available to all who are interested during the third week of each month. All issues that have been resolved and communicated to AMD's customers are included in this release.

# 1 Product Marking Identification

## 1.1 Production Marking




**Ceramic Pin Grid Array (CPGA)**

**(Packages Not Drawn to Scale)**

---

xxxpvt = OPN, where:

- xxx = Operating Frequency
- p = Package Type
  - A = 321-pin PGA
- v = Operating Voltage
  - N = 3.1-3.3V Core/3.135-3.6V I/O
  - L = 2.755-3.045V Core/3.135-3.6V I/O
- t = Case Temperature
  - R = 70°C

v.vV = Core Voltage, where:

- 2.9V = 2.9V Component
- 3.2V = 3.2V Component

R AAAAAAA = Revision, where:

- R = Revision
  - B = Revision B
  - C = Revision C
  - etc.
- AAAAAAA = Internally-Defined

# 2 Product Errata

This section documents AMD-K6 processor product errata. The errata are divided into categories to assist referencing particular errata. A unique tracking number for each erratum has been assigned within this document for user convenience in tracking the errata within specific revision levels. Table 1 cross-references the revisions of the processor to each erratum. An "X" indicates that the erratum applies to the stepping. The absence of an "X" indicates that the erratum does not apply to the stepping. Shading within the table indicates an addition or modification from the previous release of this document.

**Table 1. Cross-Reference of Product Revision to Errata**

| Erratum Number | Description | Rev B |
|---|---|---|
| | **Test and Debug** | |
| 2.1.1 | Built-In Self-Test (BIST) | X |
| | **System Bus** | |
| 2.2.1 | Drive Strength Selection | X |
| 2.2.2 | HLDA Assertion Delayed by One Clock | X |
| | **Interrupts and Exceptions** | |
| 2.3.1 | Interruption of REP MOVS Instruction | X |
| 2.3.2 | Memory Accesses Using Null Selectors | X |
| | **Instructions** | |
| 2.4.1 | LGDT Instruction | X |
| | **DC Specifications** | |
| 2.5.1 | Input Leakage Current Bias with Pullup ($I_{IL}$) | X |
| | **Self-Modifying Code** | |
| 2.6.1 | Misaligned Write to Displacement of Transfer Control Instruction | X |
| 2.6.2 | Re-execution of Instructions Due to Self-Modifying Code | X |
| | **Numeric Processing** | |
| 2.7.1 | Numeric Multiply With Extended Precision | X |
| | Shading indicates additions or modifications from the previous release of this document | |

## 2.1 Test and Debug

### 2.1.1 Built-In Self-Test (BIST)

***Products Affected.*** B stepping

***Normal Specified Operation.*** Following the falling transition of RESET, the processor unconditionally runs its BIST.

***Non-conformance.*** BIST is not supported.

***Potential Effect on System.*** The L1 caches are not tested after RESET.

***Suggested Workaround.*** None.

***Resolution Status.*** This erratum will be corrected in a future stepping of the AMD-K6 processor.

## 2.2 System Bus

### 2.2.1 Drive Strength Selection

***Products Affected.*** B stepping

***Normal Specified Operation.*** The processor samples the BRDYC# input during the falling transition of RESET to configure the drive strength of A[20:3], ADS#, HITM#, and W/R#. If BRDYC# is sampled asserted during the fall of RESET, these particular outputs are configured using a higher drive strength than the standard drive strength. If BRDYC# is sampled negated during the fall of RESET, the standard drive strength is selected for these particular outputs.

***Non-conformance.*** If BRDYC# is sampled asserted during the fall of RESET, all output drivers are configured using the higher drive strength.

***Potential Effect on System.*** Using the higher strength drive configuration can affect signal quality by resulting in additional overshoot or undershoot.

***Suggested Workaround.*** Use the standard drive configuration. With few exceptions, most board designs require the standard drive configuration.

***Resolution Status.*** This erratum will be corrected in a future stepping of the AMD-K6 processor.

### 2.2.2 HLDA Assertion Delayed by One Clock

***Products Affected.*** B stepping

***Normal Specified Operation.*** If BOFF# and HOLD are sampled asserted on the same clock edge that negates ADS#, the processor asserts HLDA one clock edge after HOLD is sampled asserted.

***Non-conformance.*** If BOFF# and HOLD are sampled asserted on the same clock edge that negates ADS#, the processor asserts HLDA two clock edges after HOLD is sampled asserted.

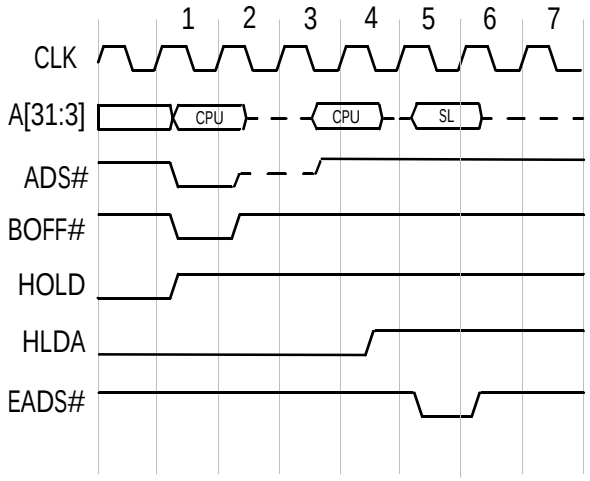
***Potential Effect on System.*** There are three potential effects of this erratum to consider:

- If the system logic asserts BOFF# for a duration of one clock, anticipates the assertion of HLDA in clock 3 (see Figure 1)—which is the normal specified operation—and drives the address bus and EADS# for an inquire cycle in clock 3, then the processor will not sample EADS# asserted. In addition, address bus contention will occur in clock 3.
- If the system logic asserts BOFF# for a duration of two clocks, anticipates the assertion of HLDA in clock 3, and drives the address bus and EADS# for an inquire cycle in clock 3, then the processor will not sample EADS# asserted. (No address bus contention occurs in this case.)
- If the system logic asserts BOFF# for a duration of one clock, anticipates the assertion of HLDA in clock 3, and drives the address bus and EADS# for an inquire cycle in clock 4, then address bus contention may occur in clock 4. (The processor will sample EADS# asserted in this case.)

If the processor does not sample EADS# asserted during an inquire cycle, cache/memory incoherency will occur. Address bus contention can affect the reliability of the processor and the system logic.

***Suggested Workaround.*** The system logic must sample the assertion of HLDA before asserting EADS# and driving the address bus for an inquire cycle—as shown in clock 5 of Figure 1.

***Resolution Status.*** This erratum will be corrected in a future stepping of the AMD-K6 processor.




CPU: Processor is driving A[31:3]
SL: System Logic is driving A[31:3]

**Figure 1. AMD-K6™ Processor Assertion of HLDA Due to Simultaneous BOFF#/HOLD Assertion**

## 2.3 Interrupts and Exceptions

### 2.3.1 Interruption of REP MOVS Instruction

***Products Affected.*** B stepping

***Normal Specified Operation.*** A move string instruction (MOVS) with a REP prefix can be interrupted before the start of its execution by the occurrence of a hardware interrupt—such as NMI, INTR, STPCLK#, or SMI#.

***Non-conformance.*** If a REP MOVS instruction is interrupted after it is decoded, but before the execution of the first iteration of the instruction—specifically, during the initial check for ECX equal to 0—and all of the conditions itemized below are true, then the incorrect address size and/or operand segment register is used for the one iteration of the REP MOVS instruction that is executed before the interrupt is recognized. Following is a list of the conditions that must apply to the REP MOVS instruction for this erratum to occur:

- An address size override prefix is used
- The initial loop count loaded in ECX equals 6
- The interrupt is received internally on exactly one specific clock just as the processor is checking if the initial loop count in ECX equals 0
- The next few instructions—between one and four—that immediately follow the REP MOVS instruction reside in the processor's instruction cache, and one of them is a vector-decoded instruction (the complex x86 instructions are typically vector decoded)
- The effective address size or operand segment register of the vector-decoded instruction differs from that of the REP MOVS instruction
- Certain other relative internal pipeline timing conditions must occur

***Potential Effect on System.*** The effect on software is unpredictable.

***Suggested Workaround.*** None.

***Resolution Status.*** This erratum is corrected in a future stepping of the AMD-K6 processor.

### 2.3.2 Memory Accesses Using Null Selectors

***Products Affected.*** B stepping

***Normal Specified Operation.*** An attempt to access memory with a segment register that contains a null selector causes a general protection fault to occur.

***Non-conformance.*** On entering System Management Mode (SMM), the processor saves its state into the SMM state-save area. The processor returns from SMM when it executes the RSM (resume) instruction from within the SMM service routine. If any of the segment registers contains a null selector on entering SMM, then after executing the RSM instruction, the internal state bit associated with the corresponding segment descriptor is erroneously set to indicate that this segment is readable. In this case, the processor does not generate a general protection fault when attempting to read memory using a null selector, but instead allows the memory read cycle to occur without generating an exception.

***Potential Effect on System.*** Unless an application is errant, an application generally does not access segments using null selectors. However, an application that depends on the processor to generate a general protection fault if a null selector is used will not execute successfully.

This erratum has not been observed to adversely affect a system. It was detected by design inspection.

***Suggested Workaround.*** None.

***Resolution Status.*** This erratum will be corrected in a future stepping of the AMD-K6 processor.

## 2.4 Instructions

### 2.4.1 LGDT Instruction

***Products Affected.*** B stepping

***Normal Specified Operation.*** While reading the memory operand of the Load Global Descriptor Table Register (LGDT) instruction, a general protection fault, a stack segment fault, or a page fault can occur.

***Non-conformance.*** If only the limit value portion of the memory operand of the LGDT instruction faults—that is, the base address portion of the memory operand must not fault for this erratum to occur—then one of two possible scenarios can occur:

- If the assertion of SMI# is recognized any time after the execution of the LGDT instruction, but before the next successfully-executed LGDT instruction, then the base address portion of the GDTR that is stored in the SMM state-save area is incorrect. During the execution of the RSM instruction, the GDTR that is restored will be corrupted.
- If an SGDT instruction is executed after the LGDT instruction—without an intervening successfully-executed LGDT instruction—then the SGDT instruction will store an incorrect base address.

***Potential Effect on System.*** Either of these scenarios can lead to unpredictable operating system behavior.

***Suggested Workaround.*** None. However, the LGDT instruction is privileged and only executed by the operating system. Operating systems typically either avoid faults on the LGDT instruction, or they align the memory operand of the LGDT instruction such that if a fault were to occur, both the base address and the limit value will fault.

This erratum has not been observed to adversely affect a system. It was detected by design inspection.

***Resolution Status.*** This erratum will be corrected in a future stepping of the AMD-K6 processor.

## 2.5 DC Specifications

### 2.5.1 Input Leakage Current Bias with Pullup ($I_{IL}$)

***Products Affected.*** B stepping

***Normal Specified Operation.*** The *Input Leakage Current Bias With Pullup* specification is –400 µA.

***Non-conformance.*** The *Input Leakage Current Bias With Pullup* is –450 µA.

***Potential Effect on System.*** None.

***Suggested Workaround.*** None.

***Resolution Status.*** This erratum will be corrected in a future stepping of the AMD-K6 processor.

## 2.6 Self-Modifying Code

### 2.6.1 Misaligned Write to Displacement of Transfer Control Instruction

***Products Affected.*** B stepping

***Normal Specified Operation.*** If an instruction causes a memory write to the displacement part of a transfer control instruction, and the transfer control instruction is subsequently executed, the processor ensures that the transfer control uses the modified displacement.

***Non-conformance.*** If:

- The target of the transfer control instruction is sequentially fetched and stored in the processor's L1 cache
- The transfer control instruction is speculatively executed causing the processor to initiate the allocation of an entry in the Branch Target Cache (BTC)
- A data read of the target of the transfer control instruction occurs (this has the effect of invalidating the previously-fetched target of the transfer control instruction that resides in the L1 cache, which in turn delays the completion of the BTC allocation)
- A misaligned memory write to the displacement part of the transfer control instruction occurs (self-modifying code)
- The processor fetches the original target of the transfer control instruction in order to complete the pending BTC allocation
- The processor refetches the transfer control instruction with the new displacement as a result of detecting the self-modifying code sequence
- Certain other relative internal pipeline timing conditions must occur

then: the processor erroneously hits the BTC and executes the instruction at the original displacement of the transfer control rather than the instruction at the modified displacement.

***Potential Effect on System.*** This erratum can lead to unpredictable system behavior.

***Suggested Workaround.*** None.

***Resolution Status.*** This erratum will be corrected in a future stepping of the AMD-K6 processor.

### 2.6.2 Re-execution of Instructions Due to Self-Modifying Code

***Products Affected.*** B stepping

***Normal Specified Operation.*** If the processor detects a potential self-modifying code condition, the processor performs specific internal actions to ensure that instruction execution occurs in the correct manner.

***Non-conformance.*** If:

- The target of a transfer control instruction is fetched and loaded into the processor's Branch Target Cache (BTC)
- This transfer control instruction is speculatively executed and hits in the BTC
- An instruction (Instruction "A") is executed that causes the processor to detect a potential self-modifying code condition relative to the transfer control target that resides in the BTC
- Instruction "A" is followed by a register-modifying instruction(s) in the form of:
  - A long-decoded instruction, or;
  - One or two short-decoded instructions
- The transfer control instruction that was speculatively executed follows the register-modifying instruction(s) within approximately 1–9 instructions
- Certain other relative internal pipeline timing conditions must occur

then: the long-decoded instruction, or the short-decoded instruction(s), is executed twice.

***Potential Effect on System.*** If software is not affected by the re-execution of the register-modifying instruction(s)—for instance, loading immediate data into a general purpose register—then this erratum has no effect on the system. However, if any of the instructions that are re-executed change the state of the system from the state that would be achieved by the normal specified operation—for instance, incrementing a register by one—then this erratum can lead to unpredictable system behavior.

***Suggested Workaround.*** None.

***Resolution Status.*** This erratum will be corrected in a future stepping of the AMD-K6 processor.

## 2.7 Numeric Processing

### 2.7.1 Numeric Multiply With Extended Precision

***Products Affected.*** B stepping

***Normal Specified Operation.*** The numeric multiply instructions—FIMUL and FMUL, and FMULP—multiply the source and destination operands and store the results in the destination operand.

***Non-conformance.*** If the source and destination operands meet the conditions described below, then the result of the multiply operation is erroneously computed as zero. In addition, the processor generates a numeric underflow exception in these particular cases.

Patterns 1-6, which are referenced in the descriptions below, are defined as follows:

- Pattern 1: D34D_34D3_4D34_D34Dh
- Pattern 2: A000_0000_0000_0000h
- Pattern 3: B5AD_6B5A_D6B5_AD6Bh
- Pattern 4: C000_0000_0000_0000h
- Pattern 5: A78A_78A7_8A78_A78Ah
- Pattern 6: D000_0000_0000_0000h

Following are the specific forms and dependencies of the numeric multiply instruction that are required to erroneously cause the result to be computed as zero:

- The numeric processor is configured for extended precision.
- FIMUL mem16—The significand of ST(0) equals pattern 1 and the 16-bit integer in memory is $\pm 5x2^n$, where (n = 0 to 12); the significand of ST(0) equals pattern 3 and the 16-bit integer in memory is $\pm 3x2^n$, where (n = 0 to 13); the significand of ST(0) equals pattern 5 and the 16-bit integer in memory is $\pm 13x2^n$, where (n = 0 to 11).
- FIMUL mem32—The significand of ST(0) equals pattern 1 and the 32-bit integer in memory is $\pm 5x2^n$, where (n = 0 to 28); the significand of ST(0) equals pattern 3 and the 32-bit integer in memory is $\pm 3x2^n$, where (n = 0 to 29); the significand of ST(0) equals pattern 5 and the 32-bit integer in memory is $\pm 13x2^n$, where (n = 0 to 27).
- FMUL mem32real—The significand of ST(0) equals pattern 1 and the single-precision operand in memory is $\pm 5x2^n$, where (n = –129 to +124); the significand of ST(0) equals pattern 3 and the single-precision operand in memory is $\pm 3x2^n$, where (n = –129 to +125); the significand of ST(0) equals pattern 5 and the single-precision operand in memory is $\pm 13x2^n$, where (n = –129 to +123).
- FMUL mem64real—The significand of ST(0) equals pattern 1 and the double-precision operand in memory is $\pm 5x2^n$, where (n = –1074 to +1029); the significand of ST(0) equals pattern 3 and the double-precision operand in memory is $\pm 3x2^n$, where (n = –1074 to +1030); the significand of ST(0) equals pattern 5 and the double-precision operand in memory is $\pm 13x2^n$, where (n = –1074 to +1028).
- FMUL(P) ST(i),ST—The significand of ST(i) equals pattern 1 and the significand of ST(0) is pattern 2; the significand of ST(i) equals pattern 3 and the significand of ST(0) is pattern 4; the significand of ST(i) equals pattern 5 and the significand of ST(0) is pattern 6.
- FMUL ST,ST(i)—The significand of ST(0) equals pattern 1 and the significand of ST(i) is pattern 2; the significand of ST(0) equals pattern 3 and the significand of ST(i) is pattern 4; the significand of ST(0) equals pattern 5 and the significand of ST(i) is pattern 6, then the result of the multiply stored in ST(0) is 0.

***Potential Effect on System.*** The computation by the numeric multiply instruction using the operand-pairs known to cause this erratum, with extended precision invoked, will be incorrect. This erratum has not been observed to occur in integer, single-precision, double-precision, or software implementations of extended precision applications, which represent the majority of the applications on the market. AMD has identified and tested commercially-available applications which specifically make use of the numeric processor in the AMD-K6. No failures have been observed. AMD has tried to force failures by choosing conditions and operands known to aggravate the erratum, and no failures were observed.

***Suggested Workaround.*** AMD will provide a software detection program that works in conjunction with Microsoft operating systems and applications to detect and workaround occurrences of this erratum.

***Resolution Status.*** This erratum will be corrected in a future stepping of the AMD-K6 processor.

# 3 Specification Changes/Clarifications

This section documents AMD-K6 processor specification changes and clarifications. The changes/clarifications are divided into categories to assist referencing particular changes. A unique tracking number for each change/clarification has been assigned within this document for user convenience in tracking the specification change/clarification within specific revision levels. Table 2 cross-references the revisions of the processor to each specification change/clarification. An "X" indicates that the specification change/clarification applies to the stepping. The absence of an "X" indicates the specification change/clarification does not apply to the stepping.

**Table 2. Cross-Reference of Product Revision to Specification Change/Clarification**

| Change Number | Description | Rev B | Future Rev |
|---|---|---|---|
| | **Interrupts and Exceptions** | | |
| 3.1.1 | Recognition of External Hardware Interrupts During I/O Read Cycle | | X |
| | **Instructions** | | |
| 3.2.1 | SYSCALL and SYSRET | | X |
| | **DC Specifications** | | |
| 3.3.1 | Maximum Stop Grant and Stop Clock Power Dissipation | | X |
| | Shading indicates additions or modifications from the previous release of this document | | |

## 3.1 Interrupts and Exceptions

### 3.1.1 Recognition of External Hardware Interrupts During I/O Read Cycle

***New Specification Applies to:*** A future stepping

***Previous Operation.*** For I/O Reads, the AMD-K6 processor waits for preceding instructions to complete before executing the I/O Read instruction. However, there is no serialization for succeeding instructions. This means that succeeding instructions can be executed in parallel with the I/O Read. As a result, external interrupts may not be recognized and serviced before succeeding instructions are completed.

***New Operation.*** For I/O Reads, the AMD-K6 processor waits for preceding instructions to complete before executing the I/O Read instruction and it serializes succeeding instructions. This means that the I/O Read instruction completes before any succeeding instructions are executed. Such serialization allows for external interrupts, asserted during the I/O cycle, to be recognized and serviced before any dependent instructions are executed.

***Implication.*** The previous and new operation has no implication for software and hardware that are designed to the Socket 7 specification, which states that IN instructions are not fully serialized.

## 3.2 Instructions

### 3.2.1 SYSCALL and SYSRET

***New Specification Applies to:*** B stepping

***Previous Operation.*** The AMD-K6 processor supports the SYSCALL and SYSRET Extensions, which provide a fast method for entering and exiting an operating system. Bit 10 of the Extended Feature Flags (Function 8000_0001h of the CPUID instruction) is set to 1 to indicate support for the SYSCALL and SYSRET Extensions. Bit 11 of the Extended Feature Flags is Reserved.

***New Operation.*** The AMD-K6 processor does not support the SYSCALL and SYSRET Extensions. Bit 10 of the Extended Feature Flags is Reserved. Bit 11 of the Extended Feature Flags indicates whether support for the SYSCALL and SYSRET Extensions exists—if bit 11 is set to 1, then the SYSCALL and SYSRET Extensions are supported; if bit 11 is set to 0, then the SYSCALL and SYSRET Extensions are not supported.

***Implication.*** Since no operating systems currently utilize these instructions, there is no implication to existing software.

## 3.3 DC Specifications

### 3.3.1 Maximum Stop Grant and Stop Clock Power Dissipation

***New Specification Applies to:*** B stepping

***Previous Operation.*** Table 3-1 contains the maximum power dissipation of the AMD-K6 processor in the Stop Grant and Stop Clock states.

**Table 3-1. Previous Specified Maximum Power Dissipation**

| Clock Control State | 2.9 V Component | | 3.2 V Component |
|---|---|---|---|
| | 166 MHz | 200 MHz | 233 MHz |
| Stop Grant / Halt (Maximum) | 450 mW | 525 mW | 745 mW |
| Stop Clock (Maximum) | 300 mW | 300 mW | 300 mW |

***New Operation.*** Table 3-2 contains the maximum power dissipation of the AMD-K6 processor in the Stop Grant and Stop Clock states.

**Table 3-2. New Specified Maximum Power Dissipation**

| Clock Control State | 2.9 V Component | | 3.2 V Component |
|---|---|---|---|
| | 166 MHz | 200 MHz | 233 MHz |
| Stop Grant / Halt (Maximum) | 1.45 W | 1.53 W | 1.75 W |
| Stop Clock (Maximum) | 1.0 W | 1.0 W | 1.0 W |

***Implication.*** There no implications for software and hardware in desktop applications.

# 4 Technical and Documentation Support

## 4.1 Documentation Support

The following documents provide additional information regarding the operation of the AMD-K6 processor:

- *AMD-K6™ MMX™ Enhanced Processor Data Sheet (order# 20695)*
- *AMD-K6™ MMX™ Enhanced Processor Multimedia Technology (order# 20726)*
- *AMD K86™ Family BIOS and Software Tools Developers Guide (order# 21062)*
- *AMD K86™ Family BIOS Design Application Note (order# 21329)*
- *AMD Processor Recognition Application Note (order# 20734)*
- *Implementation of Write Allocate in the K86 Processors (order# 21326)*
- *AMD-K6™ MMX™ Enhanced Processor Thermal Solution Design Application Note (order# 21085)*
- *AMD-K6™ MMX™ Enhanced Processor Power Supply Design Application Note (order# 21103)*
- *AMD-K6™ MMX™ Enhanced Processor I/O Model Application Note (order# 21084)*

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