AMD BIOS Development Guide

Publication#**19720** Rev**: C** Amendment**/0**
Issue Date: **August 1995**

Contents

TABLES

FIGURES

1 Introduction

Purpose

The purpose of this document is to identify BIOS modifications required to support standard Am486® microprocessors, Enhanced Am486 microprocessors, and $Am5_X86$™ microprocessors. There may be more than one way to implement the functionality detailed in this document; this information provides implementation examples.

Audience

The reader should have a detailed familiarity with x86 architecture and programming requirements, the Enhanced Am486 microprocessor, and the $Am5_X86$ microprocessor. The data provided in this document is designed to assist third party vendors to prepare BIOS to support the Enhanced Am486 processor and $Am5_X86$ processor families.

Recommended Documentation

Table 1-1 lists additional reference documentation for writing software for AMD microprocessors.

TABLE 1-1. Reference Documentation

Document Title	Document #
Am486 Microprocessor Software User's Manual	18497A
Standard Am486DX2 Microprocessor data sheet	19200D
Standard Am486DX4 Microprocessor data sheet	19160D
Enhanced Am486 Microprocessor Family data sheet	19225B
$Am5_X86$ Microprocessor Family data sheet	19751B

2 General

Standard AM486 Microprocessor

At RESET, the standard Am486DX4-100 processor places a value of 043xh in the DX register. This value is different from that used by the i486DX4-100 CPU. Use the detection algorithm presented in Figure 3-1 on page 4 to assure proper detection of AMD microprocessors.

Am486Plus Microprocessor vs. Enhanced Am486 Microprocessor

Early documentation released under Non-Disclosure Agreements referred to an Am486Plus microprocessor. The Am486Plus designation was an internal code name. The released product is named the Enhanced Am486 microprocessor. Use the detection algorithm presented in Figure 3-2 on page 6 to assure proper detection of Enhanced AMD microprocessors.

Enhanced Am486 Microprocessor vs. $Am5_X86$ Microprocessor

The Enhanced Am486 and $Am5_X86$ microprocessors can be easily differentiated by evaluating clock speed. $Am5_X86$ processors operate at clock speeds of 133 MHz and greater. Enhanced Am486 microprocessors operate at clock speeds of 120 MHz or less.

3 CPU Identification Algorithms

The following sections describe the CPU identification algorithms for the various microprocessors.

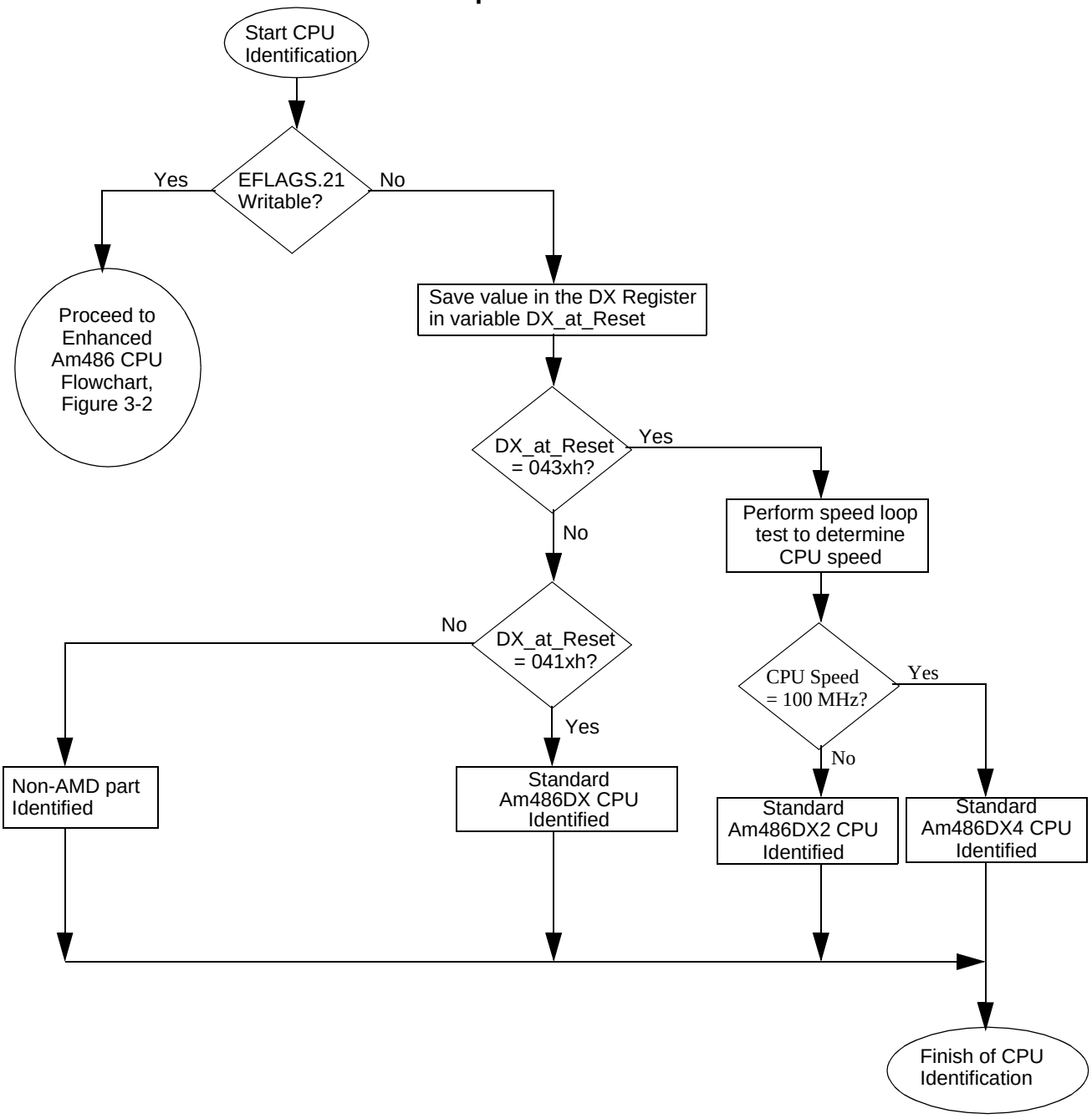
Standard Am486 Microprocessor

Table 3-1 summarizes the CPU IDs found in the DX register after RESET for the Standard AM486 Microprocessor..

TABLE 3-1. Summary of CPU IDs for the Standard AM486 Microprocessor

AMD CPU Type	CPU ID in DX at Reset
Standard Am486DX	041xh
Standard Am486DX2	043xh
Standard Am486DX4	043xh

FIGURE 3-1. Standard Am486 Microprocessor Identification Procedure



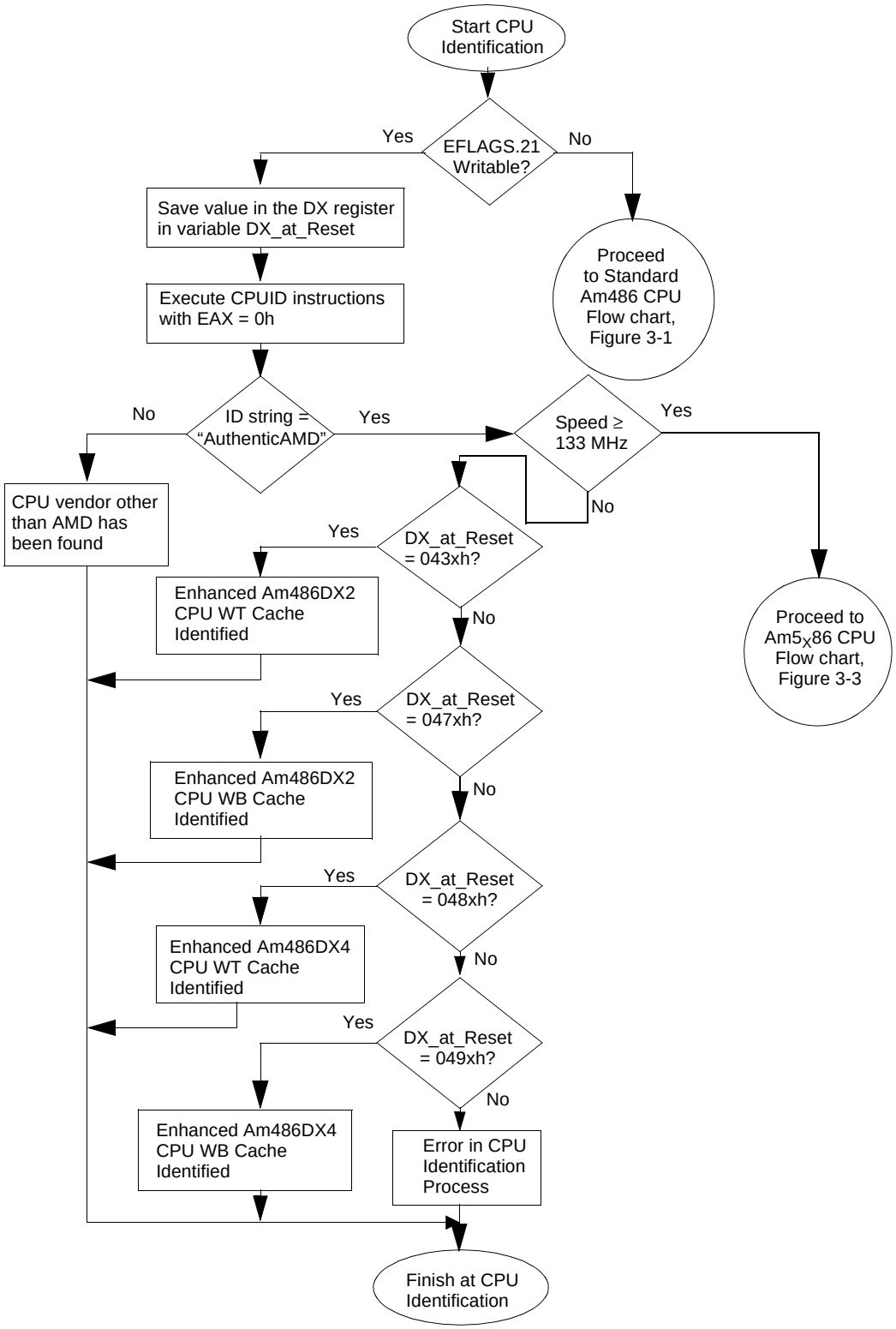
Enhanced Am486 Microprocessor

Table 3-2 summarizes the CPU IDs found in the DX register after RESET for the Enhanced AM486 Microprocessors.

TABLE 3-2. Summary of CPU IDs for Enhanced Am486 Microprocessors

AMD CPU Type	CPU ID in DX at Reset	Comments
Enhanced Am486DX2 CPU	043xh	Write Through Cache
	047xh	Write-Back Cache
Enhanced Am486DX4 CPU	048xh	Write-Through Cache
	049xh	Write-Back Cache

FIGURE 3-2. Enhanced Am486 Microprocessor Identification Procedure



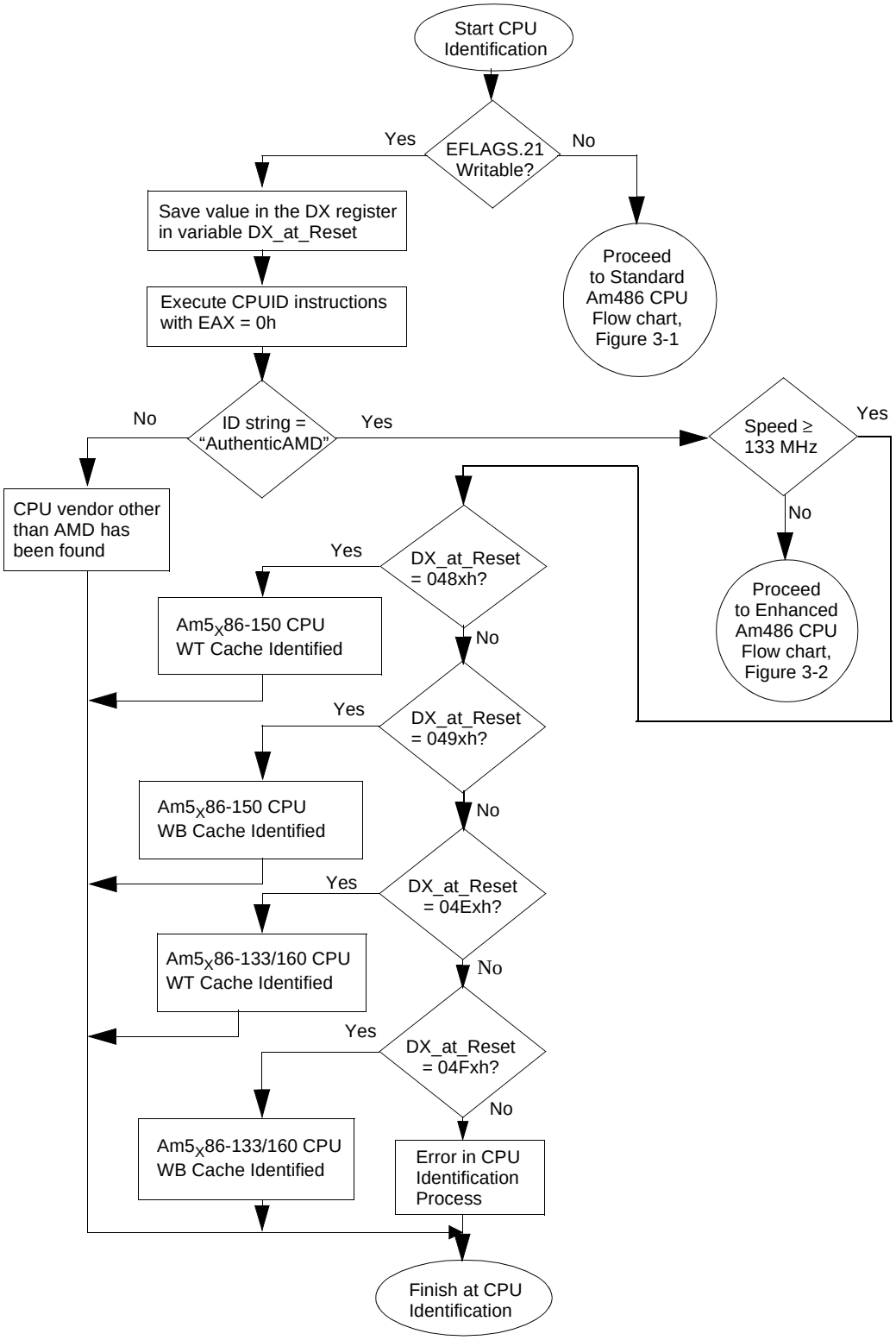
Am5$_X$86 Microprocessor

Table 3-3summarizes the CPU IDs found in the DX register after RESET for the Am5$_X$86 microprocessors.

TABLE 3-3. Summary of CPU IDs for the Am5$_X$86 Microprocessors

AMD CPU Type	CPU ID in DX at Reset	Comments
Am5$_X$86 CPU at 133 MHz or 160 MHz	04Exh	Write-Through Cache
	04Fxh	Write-Back Cache
Am5$_X$86 CPU at 150 MHz	048xh	Write-Through Cache
	049xh	Write-Back Cache

FIGURE 3-3. Am5$_X$86 Microprocessor Identification Procedure



4 CPUID Instruction for the Enhanced Am486 and $Am5_X86$ Microprocessors

Enhanced Am486 and $Am5_X86$ microprocessors support the CPUID instruction. To utilize the CPUID instruction, a pseudo-code algorithm is provided in Figure 4-1 to aid code development.

FIGURE 4-1. CPUID Instruction Pseudo-Code

```
begin
{
if EFLAGS.21 is writable then
    {
    if vendor string report desired
        {
        load EAX with 0h
        execute CPUID instruction (opcode = 0fh a2h)
        if Result:
            EBX = 'Auth'
            EDX = 'enti'
            ECX = 'cAMD'
            Part is AMD.
        }
        else
            Part is non-AMD.
    if part description is desired
        {
        load EAX with 1
        execute CPUID instruction (opcode = 0Fh A2h)
        Result:
        EAX[3–0] = stepping ID (contact AMD for specifics)
        EAX[7–4] = Model
            Am486DX2-WT -> 3
            Am486DX2-WB -> 7
            Am486DX4-WT -> 8
            Am486DX4-WB -> 9
            Am5X86-WT (150 MHz) -> 8
            Am5X86-WB (150 MHz) -> 9
            Am5X86-WT (133 and 160 MHz) -> E
            Am5X86-WB (133 and 160 MHz) -> F
        EAX[11–8] = Family
            Am486 CPU -> 4
            Am5X86 CPU -> 4
        EAX[15–12] = 0000
```

```
        EAX[31–16] = reserved
        EBX = 00000000h
        ECX = 00000000h
        EDX = 00000001h (bit 0==1 indicates FPU present)
        }
    }
else
    CPUID is not supported
}end
```

Note: *The standard Am486 microprocessor does not support the CPUID instruction.*

5 Cache

The Enhanced Am486 and $Am5_X86$ processors support both write-back and write-through cache modes. To select between the different cache modes, set the WB/$\overline{WT}$ pin on the microprocessor to the proper logic state at RESET (see Table 5-1). The processor responds with the proper CPU ID in the DX register after RESET so that the BIOS can detect the selected cache operating mode. The CPUID instruction also reports the proper configuration.

TABLE 5-1. Cache Mode Selection Using WB/$\overline{WT}$

WB/$\overline{WT}$ Input at RESET	Cache Mode Selected
V_{SS} or Floating	write-through
V_{CC}	write-back

Figure 5-1 shows a code example that can be used to test the cache size. Figure 5-2 is a flow diagram that illustrates how this code detects the cache size.

If the EXT bit in TR5 = 0, bit 11 in the TAG field in TR4 will behave as follows:

- For 8-Kbyte cache, the bit is read/write.
- For 16-Kbyte cache, the bit is read only and always zero during a cache look-up. The bit is read/write otherwise.

FIGURE 5-1. Cache Size Detection Code Example

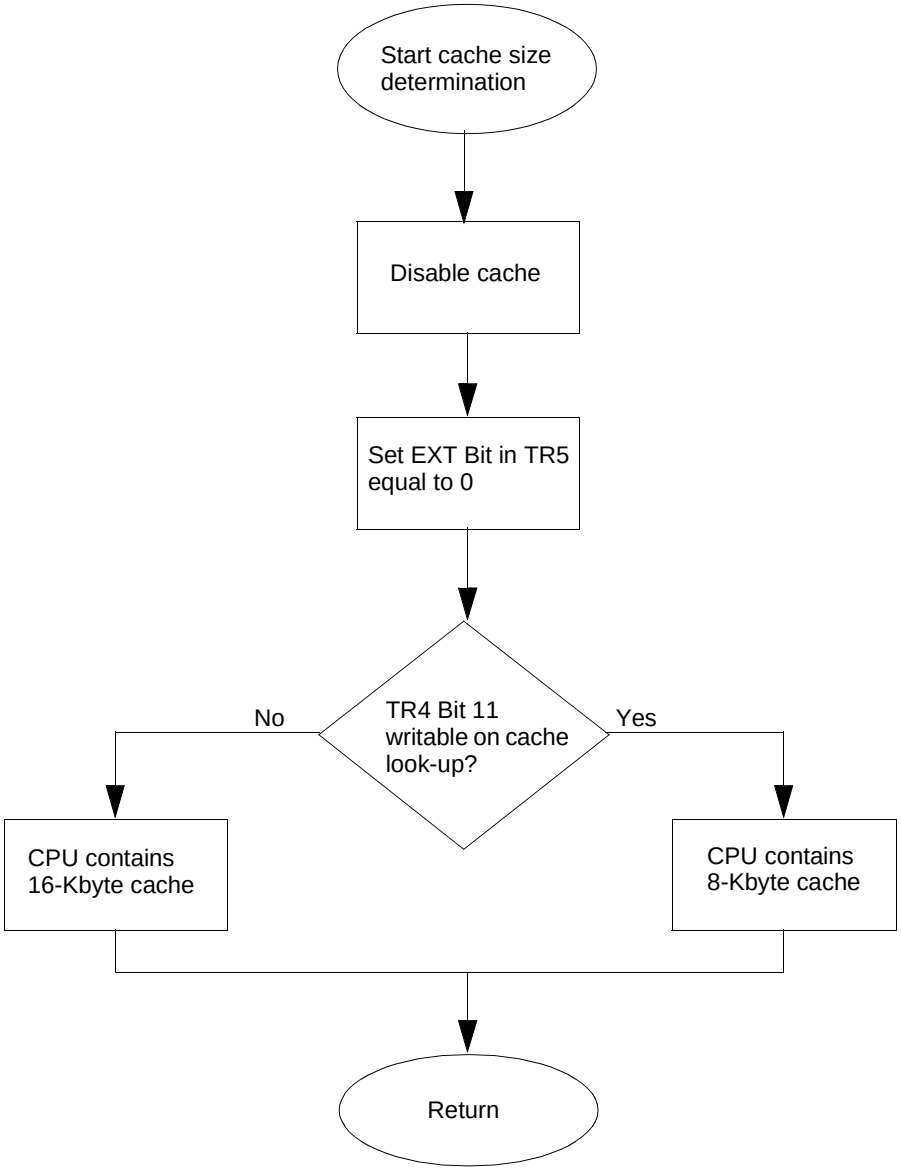
```
disable_cache
    mov  ebx,00000002h      ;perform cache read with EXT==0
    mov  tr5,ebx
    mov  eax,tr4            ;read tr4 to get valid tag bits
    xor  eax,00000800h      ;toggle bit 11 of tr4
    mov  tr4,eax            ;write new value to tr4
    mov  ebx,00000001h      ;perform cache write with new tr4
    mov  tr5,ebx
    push  eax
    mov  ebx,00000002h      ;cache read with EXT==0
    mov  tr5,ebx
    mov  eax,tr4            ;read tr4 for valid tag bits
    pop  ebx
    and  bh,00001000b       ;mask for bit 11
    and  ah,00001000b       ;mask for bit 11
```

```
    cmp  ah,bh
    jne  16k_cache
    jmp  8k_cache

16k_cache:
    ;16-Kbyte cache is detected

8k_cache:
    ;8-Kbyte cache is detected
```

FIGURE 5-2. Cache Size Determination Flowchart



6 Clock Mode Selection

The Enhanced Am486 and $Am5_X86$ processors sample the CLKMUL input signal at reset to determine the design operating frequency. If CLKMUL is High or floating, the CPU uses 3x clock mode. If CLKMUL is Low, the Enhanced Am486 CPU uses a 2x clock mode; the $Am5_X86$ CPU uses a 4x clock mode (see Table 6-1). An internal pull-up connects CLKMUL to V_{CC}. The processor responds with the proper CPU ID in the DX register after RESET so that the BIOS can detect the selected clock mode. The CPUID instruction also reports the clock mode configuration.

TABLE 6-1. Clock Mode Selection

CLKMUL Input State at RESET	Clock Mode Selected
V_{SS}	Enhanced Am486 CPU = 2x $Am5_X86$ CPU = 4x
V_{CC} or Floating	3x

Note: *The Enhanced Am486 microprocessor and the $Am5_X86$ microprocessor DO NOT SUPPORT a 2.5 times clock multiplier. Do not connect the BREQ signal to the CLKMUL input signal.*

7 Test Registers 4 and 5

The Cache Test Registers for the Enhanced Am486 and $Am5_X86$ microprocessors are the same test registers (TR3, TR4, and TR5) provided in earlier Am486 microprocessors. TR3 is the cache test data register. TR4, the cache test status register, and TR5, the cache test control register, operate together with TR3.

If WB/$\overline{WT}$ meets the necessary setup timing and is sampled Low on the falling edge of RESET, the processor is placed in write-through cache mode and the test register function is identical to the earlier Am486 microprocessors. If WB/$\overline{WT}$ meets the necessary setup timing and is sampled High on the falling edge of RESET, the processor is placed in write-back cache mode and TR4 and TR5 are modified to support the added write-back cache functionality. The following tables show the individual bit functions of these registers.

TABLE 7-1. Test Register 4 (TR4) Bit Definitions for 8-Kbyte Cache

Bit	31	30–29	28	27–26	25–24	23–22	21–20	19–16	15–11	10	9–7	6–3	2–0
EXT=0	Tag									Valid	LRU	Valid (rd)	NA
EXT=1	NA	Stn	Resv.	ST3	ST2	ST1	ST0	Reserved	NA	Valid	LRU	Valid (rd)	NA

Note: *Bit 11 of TR4 can be used to identify internal cache size. See Figures 5 and 6 for additional information.*

TABLE 7-2. Test Register 4 (TR4) Bit Definitions for 16-Kbyte Cache

Bit	31	30–29	28	27–26	25–24	23–22	21–20	19–16	15–12	11	10	9–7	6–3	2–0
EXT=0	Tag									0	Valid	LRU	Valid (rd)	NA
EXT=1	NA	Stn	Resv.	ST3	ST2	ST1	ST0	Reserved	NA	NA	Valid	LRU	Valid (rd)	NA

TABLE 7-3. Test Register 5 (TR5) Bit Definitions for 8-Kbyte Cache

Cache Type	31–20	19	18–17	16	15–11	10–4	3–2	1–0
WB	Not Used	EXT	Set State	Resv.	Not Used	Index	Entry	Control
WT	Not Used					Index	Entry	Control

TABLE 7-4. Test Register 5 (TR5) Bit Definitions for 16-Kbyte Cache

Cache Type	31–20	19	18–17	16	15–12	11–4	3–2	1–0
WB	Not Used	EXT	Set State	Resv.	Not Used	Index	Entry	Control
WT	Not Used					Index	Entry	Control

Note: *TR3 has the same functions in both write-through and write-back cache modes. These functions are identical to the TR3 register functions provided by earlier Am486 microprocessors.*

TR4 Definition

This section includes a detailed description of the bit fields defined for TR4.

Tag (bits 31–11 for 8-Kbyte cache, or bits 31–12 for 16-Kbyte cache)

Read/Write, always available in write-through mode. Available only when EXT=0 in TR5 in write-back mode. For a cache write, this is the tag that specifies the address in memory. On a cache look-up, this is tag for the selected entry in the cache.

STn (bits 30–29)

Read Only, available only in write-back mode when Ext=1 in TR5. STn returns the status of the set (ST3, ST2, ST1, or ST0) specified by the TR5 Set State field (bits 18–17) during cache look-ups. Returned values are:

00 = invalid
01 = exclusive
10 = modified
11 = shared

ST3 (bits 27–26) Read Only, available only in write-back mode when Ext=1 in TR5. ST3 returns the status of Set 3 during cache look-ups. Returned values are:

00 = invalid
01 = exclusive
10 = modified
11 = shared

ST2 (bits 25–24) Read Only, available only in write-back mode when Ext=1 in TR5. ST2 returns the status of Set 2 during cache look-ups. Returned values are:

00 = invalid
01 = exclusive
10 = modified
11 = shared

ST1 (bits 23–22) Read Only, available only in write-back mode when Ext=1 in TR5. ST1 returns the status of Set 1 during cache look-ups. Returned values are:

00 = invalid
01 = exclusive
10 = modified
11 = shared

ST0 (bits 21–20) Read Only, available only in write-back mode when Ext=1 in TR5. ST0 returns the status of Set 0 during cache look-ups. Returned values are:

00 = invalid
01 = exclusive
10 = modified
11 = shared

Cache Size Bit (bit 11) Read/Write for 8-Kbyte cache if Ext=0. Read only for 16-Kbyte cache if Ext=0 during a cache look-up (the bit is read/write at all other times). The bit can be used to determine cache size. See Figures 5 and 6 for more information.

Valid (bit 10) Read/Write, independent of the Ext bit in TR5. This is the Valid bit for the accessed entry. On a cache look-up, Valid is a copy of one of the bits reported in bits 6–3. On a cache write in write-through mode, Valid becomes the new valid bit for the selected entry and set. In write-back mode, writing to the Valid bit has no effect and is ignored; the Set State bit locations in TR5 are used to set the Valid bit for the selected entry and set.

LRU (bits 9–7)

Read Only, independent of the Ext bit in TR5. On a cache look-up, these are the three LRU bits of the accessed set. On a cache write, these bits are ignored; the LRU bits in the cache are updated by the pseudo-LRU cache replacement algorithm. Write operations to these locations have no effect on the device.

Valid (bits 6–3)

Read Only, independent of the Ext bit in TR5. On a cache look-up, these are the four Valid bits of the accessed set. In write-back mode, these valid bits are set if a cache set is in the exclusive, modified, or shared state. Write operations to these locations have no effect on the device.

TR5 Definition

This section includes a detailed description of the bit fields in the TR4.

Ext (bit 19)

Read/Write, available only in write-back mode. Ext, or extension, determines which bit fields are defined for TR4: the address TAG field, or the STn and ST3–ST0 status bit fields. In write-through mode, the Ext bit is not accessible. The following describes the two states of Ext:

Ext = 0, bits 31–11 of TR4 contain the TAG address
Ext = 1, bits 30–29 of TR4 contain STn, bits 27–20 contain ST3–ST0

Set State (bits 18–17)

Read/Write, available only in write-back mode. The Set State field is used to change the MESI state of the set specified by the Index and Entry bits. The state is set by writing one of the following combinations to this field:

00 = invalid
01 = exclusive
10 = modified
11 = shared

Index (8-Kbyte Cache = bits 10–4; 16-Kbyte Cache = bits 11–4)

Read/Write, independent of write-through or write-back mode. Index selects one of the 128 sets.

Entry (bits 3–2)

Read/Write, independent of write-through or write-back mode. Entry selects between one of the four entries in the set addressed by the Set Select during a cache read or write. During cache fill buffer writes or cache read buffer reads, the value in the Entry field selects one of the four double-words in a cache line.

Control (bits 1–0)

Read/Write, independent of write-through or write-back mode. The control bits determine which operation to be performed. The following is a definition of the control operations:

00 = Write to cache fill buffer, or read from cache read buffer.
01 = Perform cache write.
10 = Perform cache read.
11 = Flush the cache (mark all entries invalid).

Using TR4 and TR5 for Cache Testing

The following sections provide examples of testing the cache using TR4 and TR5.

Example 1: Reading the Cache (Write-back Mode Only)

1. Disable caching by setting the CD bit in the CR0 register.
2. In TR5, load 0 into the Ext field (bit 19), the required index into the Index field (bits 10–4), the required entry value into the Entry field (bits 3–2), and 10 into the Control field (bits 1–0). Loading the values into TR5 triggers the cache read. The cache read loads the TR4 register with the TAG for the read entry, and the LRU and Valid bits for the entire set that was read. The cache read loads 128 data bits into the cache read buffer. The entire buffer can be read by placing each of the four binary combinations in the Entry field and setting the Control field in TR5 to 00 (binary). Read each doubleword from the cache read buffer through TR3.
3. Reading the Set State fields in TR4 during write-back mode is accomplished by setting the Ext field in TR5 to 1 and re-reading TR4.

Example 2: Writing the Cache

1. Disable the cache by setting the CD bit in the CR0 register.
2. In TR5, load 0 into the Ext field (bit 19), the required entry value into the Entry field (bits 3–2), and 00 into the Control field (bits 1–0).
3. Load the TR3 register with the data to write to the cache fill buffer. The cache fill buffer write is triggered by loading TR3.
4. Repeat steps 2 and 3 for the remaining three doublewords in the cache fill buffer.
5. In TR4, load the required values into TAG field (bits 31–11) and the Valid field (bit 10). In write-back mode, the Valid bit is ignored since the Set State field in TR5 is used in place of the TR4 Valid bit. The other bits in TR4 (9–0) have no effect on the cache write.

6. In TR5, load 0 into the Ext field (bit 19), the required value into the Set State field (bits 18–17) (write-back mode only), the required index into the Index field (bits 10–4), the required entry value into the Entry field (bits 3–2), and 01 into the Control field (bits 1–0). Loading the values into TR5 triggers the cache write. In write-through mode, the Set State field is ignored, and the Valid bit (bit 10) in TR4 is used instead to define the state of the specified set.

Example 3: Flushing the Cache

The cache flush mechanism functions the same way both in write-back and write-through modes. Load 11 into the Control field (bits 1–0) of TR5. All other fields are ignored, except for Ext in write-back mode. The cache flush is triggered by loading the value into TR5. All of the LRU bits, Valid bits, and Set State bits are cleared.

8 SMM Support

Standard Am486 Microprocessor

The standard Am486 microprocessor does not provide SMM support.

Enhanced Am486 and Am5$_X$86 Microprocessors

When an $\overline{\text{SMI}}$ signal is recognized on an instruction execution boundary, the processor waits for all stores to complete. The processor then saves its register state to SMRAM and begins to execute the SMM handler.

The following is a summary of the key features in the SMM environment:

- Real mode style addressing
- 4-Gbyte limit checking
- IF flag is cleared; INTR is not recognized
- NMI is disabled
- TF flag in EFLAGS is cleared; single step traces are disabled
- DR7 is cleared; debug traps are disabled
- The RSM instruction restores the state of the CPU prior to entering SMM
- Default 16-bit opcode, register, and stack use

CPU Registers

Tables 8-1 and 8-2 highlight the default register values when entering SMM.

TABLE 8-1. SMM Initial CPU Register Settings

Register	SMM Initial State
General Purpose Registers	unmodified
EFLAGS	00000002h
CR0	Bits 0, 2, 3, and 31 cleared (PE, EM, TS, and PG); remainder is unmodified
DR6	undefined
DR7	Bit 12 retains its value at the time the processor entered SMM mode
GDTR, LDTR, IDTR, TSSR	unmodified
EIP	00008000h

TABLE 8-2. Segment Register Initial States in SMM

Segment Register	Selector	Base	Attributes	Limit
CS	3000h	30000h	16-bit, expand up	4 Gbytes
DS	0000h	00000000h	16-bit, expand up	4 Gbytes
ES	0000h	00000000h	16-bit, expand up	4 Gbytes
FS	0000h	00000000h	16-bit, expand up	4 Gbytes
GS	0000h	00000000h	16-bit, expand up	4 Gbytes
SS	0000h	00000000h	16-bit, expand up	4 Gbytes

Exceptions and Interrupts within SMM

This feature is compatible with the industry standard 2-pin SMM (see documents # 19225 and 19751).

System Management Mode Revision

The SMM Revision Identifier specifies the version of SMM and the extensions that are available on the processor. Table 8-3 defines the bits associated with this register. A 1 present in either the I/O Trap Extension or the SMM Base Relocation indicates that this feature is available for use.

TABLE 8-3. SMM Revision Identifier

31–18	17	16	15–0
Reserved	I/O Trap Extension	SMM Base Relocation	SMM Revision Level
0	1	1	0000h

Auto Halt Restart

The Auto Halt Restart slot at register offset 7F02h in SMRAM indicates to the SMM handler that the SMI interrupted the CPU during a HALT state. Bit position 0 of 7F02h will be set to a one in this condition. Figure 8-1 shows the pseudo-code required to implement this feature in the BIOS.

FIGURE 8-1. Auto Halt Restart Implementation Pseudo-Code

```
begin
{
if EFLAGS.21 is writable then                ;should be done during ID process
    {
    if HLT instruction needs to be restarted then
        {
        if SMI during halt state then        ;bit 0 of offset 7F02h = 1
            set HLT restart slot to 00FFh    ;offset 7F00h in state save map
        }
    }
else
    SMM features are not supported
}
end
```

Auto Halt Power Down

This feature is described in documents # 19225 and 19751.

Relocatable SMI Handler

The address space used as SMRAM can be modified by changing the SMBASE register before exiting an SMI handler routine. SMBASE can be changed to any 32-Kbyte aligned value. Values that are not 32-Kbyte aligned will cause the CPU to enter the Shutdown state when executing the RSM instruction. SMBASE is set to the default value of 30000h on reset, but is not changed an SMM handler. All subsequent SMI requests will initiate a state save at the new SMBASE.

The SMBASE slot in the SMM state save area indicates and changes the SMI vector location and the SMRAM save area. When bit 17 of the SMM Revision Identifier is set, then this feature exists and the SMRAM base and consequently the jump vector are as indicated by the SMM Base slot. Figure 8-2 shows the pseudo-code required to implement this feature in the BIOS.

FIGURE 8-2. Relocatable SMI Handler Implementation Pseudo-Code

```
begin
{
if EFLAGS.21 is writable then
    {
    if SMI Handler is to be Relocated then
        {
        set SMBASE slot to required value;offset fef8h in state map
        resume
        }
    else
        {
        SMI Handler execution to begin at relocation area.
        resume
        }
    }
else
    SMM is not supported
}
end
```

I/O Trap Restart

The I/O instruction restart slot gives the SMM handler the option of causing the RSM instruction to automatically re-execute an interrupted I/O instruction. When the RSM instruction is executed and the I/O instruction restart slot contains the value 0FFh, the CPU automatically re-executes the I/O instruction that the $\overline{\text{SMI}}$ signal trapped. The CPU automatically initializes the I/O instruction restart slot to 00h during SMM entry. The I/O instruction restart slot should be written only when a valid I/O instruction has occurred; this can be checked via the I/O Trap Configuration Word bit 1 (offset 0FF04h in the State Save Map). The bit definitions for the I/O Trap Word are in Table 8-4. If the I/O instruction restart slot is set on an invalid I/O instruction the processor operation is unpredictable.

TABLE 8-4. I/O Trap Word Configuration

31–16	15–2	1	0
I/O Address	Reserved	Valid I/O Instruction	Read/Write

If the system executes back-to-back SMI requests, the second SMM handler must not set the I/O instruction restart slot. the second $\overline{\text{SMI}}$ signal will not have the I/O Trap Word set. Figure 8-3 shows the pseudo-code required to implement this feature in the BIOS.

FIGURE 8-3. I/O Trap Restart Implementation Pseudo-Code

```
begin
{
if EFLAGS.21 is writable then
    {
    if I/O instruction needs to be restarted then
        {
        if valid I/O instruction then
            set I/O restart slot to 00ffh;offset 7f00h in state save map
        }
    }
else
    SMM features are not supported
}
end
```

State Save Information

When the $\overline{\text{SMI}}$ signal is recognized, Enhanced Am486 and $Am5_X86$ microprocessors will save the CPU state to the state save area specified in Table 8-5. If the SMI has been relocated, then the state dump will begin at CS Base + [8000h + 7FFFh]. The default CS Base is 3000h.

TABLE 8-5. Enhanced AM486 and $Am5_X86$ Microprocessor State Save Map

CPU Registers	Address
CR0	FFFCh
CR3	FFF8h
EFLAGS	FFF4h
EIP	FFF0h
EDI	FFECh
ESI	FFE8h
EBP	FFE4h
ESP	FFE0h
EBX	FFDCh
EDX	FFD8h
ECX	FFD4h
EAX	FFD0h
DR6 (FFFFCFF3h)	FFCCh
DR7	FFC8h
TR	FFC4h

TABLE 8-5. Enhanced AM486 and Am5$_X$86 Microprocessor State Save Map (continued)

CPU Registers	Address
LDTR	FFC0h
GS	FFBCh
FS	FFB8h
DS	FFB4h
SS	FFB0h
CS	FFACh
ES	FFA8h
TSS ATR	FFA4h
TSS BASE	FFA0h
TSS LIMIT	FF9Ch
IDT ATR	FF98h
IDT BASE	FF94h
IDT LIMIT	FF90h
GDT ATR	FF8Ch
GDT BASE	FF88h
GDT LIMIT	FF84h
LDT ATR	FF80h
LDT BASE	FF7Ch
LDT LIMIT	FF78h
GS ATR	FF74h
GS BASE	FF70h
GS LIMIT	FF6Ch
FS ATR	FF68h
FS BASE	FF64h
FS LIMIT	FF60h
DS ATR	FF5Ch
DS BASE	FF58h
DS LIMIT	FF54h
SS ATR	FF50h
SS BASE	FF4Ch
SS LIMIT	FF48h
CS ATR	FF44h

TABLE 8-5. Enhanced AM486 and Am5$_X$86 Microprocessor State Save Map (continued)

CPU Registers	Address
CS BASE	FF40h
CS LIMIT	FF3Ch
ES ATR	FF38h
ES BASE	FF34h
ES LIMIT	FF30h
NO	FF2Ch
NO	FF28h
EIP_PREV	FF10h
NO	FF0Ch
NO	FF08h
IO TRAP WORD	FF04h
HALT AUTO RSTRT/IO RSTRT	FF00h
SMM REV ID 20030000h	FEFCh
STATE DUMP BASE	FEF8h
CR2	FEF4h
DR0	FEF0h
DR1	FEECh
DR2	FEE8h
DR3	FEE4h

9 SRESET

Standard Am486 Microprocessor

The standard Am486 microprocessor does not provide support for SRESET.

Enhanced Am486 Microprocessor

This feature is described in document # 19225.

Am5$_X$86 Microprocessor

The feature is described in document # 19751.