5x86 CPU BIOS WRITER'S GUIDE

REVISION 1.12

CYRIX CORPORATION

Cyrix Corporation
2703 North Central Expressway
Richardson, Texas 75080
United States of America

Order Number 94246-00

Revision History

REVISION	RELEASE DATE	DESCRIPTION OF CHANGES
1.0	8/31/94	First release (preliminary).
1.1	9/13/94	Revised I/O recovery time register bit definition.
1.2	9/30/94	Corrected Table 4-6 index assignments.
1.3	11/07/94	Revised Table 3-1 Device Identification Register contents.
1.4	1/26/94	Revised Table 3-1.
1.5	2/6/95	Changed M9 name to M1sc.
1.6	2/9/95	Updates in section 4.
1.7	3/2/95	Modified EDX values and CPUID instruction results.
1.8	5/23/95	Add PCR0 Register
1.9	7/6/95	Changes to Table 4-5, Section 6.
1.10	7/24/95	Added recommended Config Settings for production. Added Table 4-9
1.11	8/2/95	Changed M1sc to 5x86 CPU, etc.
1.12	0/22/95	Table 4-9 was changed: LSSER is set to 1 for PCI- and VL- based systems

Table of Contents

1. Introduction

This document is intended for 5x86™ CPU system BIOS writers. It is not a stand-alone document but supplements other 5x86 CPU and Cyrix® documentation, including the Cx486DX/DX2 Data Book, "Bus Interface Differences, 5x86 CPU versus Cx486DX/DX2 and Intel DX4," and the Cyrix SMM Programmer's Guide. This document includes information on 5x86 CPU detection, configuration-register definition, and recommendations for configuration-register programming. This document discusses only the differences between the Cx486DX/DX2 and the 5x86 CPU that may affect BIOS programming.

2. Configuration Register Index Assignments

The 5x86 processor provides configuration registers used to control the on-chip cache, system management mode (SMM), and other unique 5x86 features. Access to the configuration registers is achieved by writing the index of the register to I/O port 22h. I/O port 23h is then used for data transfer. Each port-23h data transfer must be preceded by a port-22h register index selection, otherwise the second and later port-23h operations are directed off-chip and produce external I/O cycles. Reads of I/O port 22h are always directed off-chip. Table 2-1 lists the 5x86 CPU configuration register index assignments.

After reset, configuration registers with indexes within the C0-CFh and FE-FFh ranges are accessible. In order to prevent potential conflicts with other devices that may use ports 22h and 23h to access their registers, the remaining registers (indexes 00-BFh and D0-FDh) are accessible only if the MAPEN(3-0) bits in CCR3 are set to 1h. See Section 4.2.3 for more information on the MAPEN(3-0) bit locations. With MAPEN(3-0)=1h, an access to a register in the 00-FFh range will not create an external I/O bus cycle. Registers with indexes C0-CFh, FE-FFh are accessible regardless of the state of MAPEN(3-0). If the register index number is outside the C0-CFh or FE-FFh ranges and if MAPEN(3-0) is set to 0h, external I/O bus cycles occur. Table 2-1 lists the MAPEN(3-0) values required to access each configuration register. The configuration registers are described in more detail later in this document. Appendix A contains example code for accessing the 5x86 configuration registers.

Table 2-1. Configuration Register Index Assignments

REGISTER INDEX	REGISTER NAME	ACRONYM	WIDTH (BITS)	MAPEN(3-0) FOR ACCESS
00h-19h	*Reserved*	--	--	--
20h	Performance Control Reg. 0	PCR0	8	1h
21h-C0h	*Reserved*	--	--	--
C1h	Configuration Control 1	CCR1	8	x
C2h	Configuration Control 2	CCR2	8	x
C3h	Configuration Control 3	CCR3	8	x
C4h-CCh	*Reserved*	--	--	--
CDh-CFh	SMM Address Region	SMAR	24	x
D0h-E7	*Reserved*	--	--	--
E8h	Configuration Control 4	CCR4	8	1h
E9h-EFh	*Reserved*	--	--	--
F0h	Power Management	PMR	8	1h
F1h-FDh	*Reserved*	--	--	--
FEh	Device Identification 0	DIR0	8	x
FFh	Device Identification 1	DIR1	8	x

x = Don't Care

3. Detecting a 5x86 CPU

There are two methods for detecting the presence of a 5x86 microprocessor. The first, and recommended, method is an extension of the standard method used to detect Cyrix CPUs, which uses the CPU Device Identification Registers (DIRs). The second detection method utilizes the CPUID instruction. The CPUID instruction is not available until Stepping 1 of the 5x86 CPU. When available, the CPUID instruction is disabled at reset and will not execute until it is enabled through the configuration registers. The following paragraphs describe each method.

3.1 Standard CPU Identification Method

System BIOS can determine if a 5x86 microprocessor exists by first determining if a Cyrix CPU exists. If a Cyrix CPU exists, the DIRs can be read to identify the type of Cyrix CPU. Previous versions of Cyrix CPUs (such as early versions of the Cx486SLC and Cx486DLC) did not contain the DIRs. However, all CPUs currently in production contain DIRs.

3.1.1 Detecting a Cyrix CPU

Detection of a Cyrix CPU is accomplished by checking the state of the undefined flags following execution of the divide instruction that divides 5 by 2 (5÷2). The undefined flags in a Cyrix microprocessor remain unchanged following the divide operation. An Intel device will modify some of the undefined flags. Appendix B contains example code for detecting a Cyrix CPU.

3.1.2 Identifying a Cyrix CPU

After determining that a Cyrix microprocessor exists, its DIRs can be read to identify its type. The DIRs are a subset of the 5x86 configuration registers. The 5x86 DIRs exist at register indexes FEh and FFh as shown in Table 2-1 and contain device identification, stepping, and revision information as shown in Table 3-1.

Table 3-1. 5x86 CPU Device Identification Register Contents

REGISTER	DESCRIPTION	BIT POSITION	CONTENTS	CORE/BUS CLOCK RATIO
DIR0	Device Identification	7-0	DEVID(7-0) = 28h or 2Ah DEVID(7-0) = 29h or 2Bh DEVID(7-0) = 2Ch or 2Eh DEVID(7-0) = 2Dh or 2Fh	1/1 2/1 4/1 3/1
DIR1	Device Stepping Device Revision	7-4 3-0	SID(3-0) = xxh RID(3-0) = xxh	-- --

x = TBD

3.2 CPU Identification Using CPUID Instruction

The CPUID instruction will not be available until Stepping 1 of the device.
Stepping 1 is indicated by DIR1=0001xxxx.

This method utilizes the 5x86 CPU Identification (CPUID) instruction. The ability to write and then read bit 21 of the EFLAG register indicates that the CPU supports the CPUID instruction. The CPUID instruction and the ability to write to bit 21 of the EFLAG register are disabled at reset. See section 4.2.4 for details on enabling the CPUID instruction. The CPUID instruction, opcode 0FA2h, provides information to identify Cyrix as the vendor and to indicate the revision and stepping of the CPU being tested.

For the CPUID instruction, the EAX register is loaded with a value to indicate the information that should be returned by the instruction. Following execution of the CPUID instruction with a zero value in EAX, the EAX, EBX, ECX, and EDX registers contain information resulting from the pseudocode shown in Figure 3-1. The EAX register contains the highest input value understood by the CPUID instruction, which is "1" for the 5x86 CPU. The EBX, ECX, and EDX registers contain the vendor identification string "CyrixInstead".

Following execution of the CPUID instruction with an input value of one loaded into the EAX register, EAX(7-0) will contain the same value as DIR0, and EAX(15-8) will contain 04h.

Bit 0 in the EDX register indicates if there is an FPU on-chip. For the 5x86 CPU, EDX(0) = 1 indicates that the FPU is on-chip, EDX(0)=0 indicates that the FPU is not present.

```
switch (EAX)
      {
      case (0):
            EAX := 1
            EBX := 69  72  79  43   /* 'i' 'r' 'y' 'C' */
            EDX := 73  6e  49  78   /* 's' 'n' 'I' 'x' */
            ECX := 64  61  65  74   /* 'd' 'a' 'e' 't' */
            break

      case (1):
            EAX[7-0]  := DIR0
            EAX[15-8] := 04h
            EDX[0]    := 1    /* 1=FPU Built In,0=No FPU */
            break

      default:
            EAX, EBX, ECX, EDX : Undefined
      }
```

Figure 3-1. Pseudocode Information Returned by CPUID Instruction

3.3 CPU EDX Value After RESET

Some CPU detection algorithms may use the value of the CPU's EDX register following RESET. The 5x86 processor's EDX register pseudocode contents following reset will be as shown below.

EDX following reset: EDX[31-16] = undefined
Stepping 0, Revision 1, and earlier:
EDX[15-8] = DIR1
EDX[7-0] = DIR0
Stepping 0, Revision 2, and later:
EDX[15-8] = 04h
EDX[7-0] = 90h

4. Configuration Register Bit Definitions

On-chip configuration registers are used to control the on-chip cache, system management mode, and other unique features of the 5x86 CPU. All bits in the configuration registers are initialized to zero following reset unless specified otherwise. The appropriate register settings will vary depending on system design. Therefore, the BIOS creation utilities or setup screen must have the capability to easily define and modify the contents of these registers (see paragraph 4.2). This will allow OEMs and integrators to easily configure these register settings with the values appropriate for the system design. The following paragraphs describe the categories of configuration registers, and the purpose and associated bit assignments for each register.

4.1 Performance Control Register 0 (PCR0)

The Performance Control Register allows advanced fifth-generation performance enhancements to be enabled or disabled. Figure 4-1 and Table 4-1 list the PCR0 bits and briefly describe their application.

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
LSSER	*Reserved*	*Reserved*	*Reserved*	*Reserved*	LOOP_EN	BTB_EN	RSTK_EN

Figure 4-1. Performance Control Register (PCR0)

Table 4-1. PCR0 Bit Definitions

BIT NAME	BIT NO.	DESCRIPTION
LSSER	7	If not set, (default state) memory reads and writes are allowed to reorder for optimum performance (load store serializing disabled). Memory accesses between 640K and 1M will always be issued in execution order. If set, all memory reads and writes will occur in execution order (load store serializing enabled). LSSER should be set to ensure that memory-mapped I/O devices operating outside of the address range 640K to 1M will operate correctly. See Paragraph 4.5.
LOOP_EN	2	If set, the 5x86 CPU will not flush the prefetch buffer if the destination of a jump is already present in the prefetch buffer. This eliminates the need for a read from the cache and thus improves performance. See Paragraph 4.5.
BTB_EN	1	If set, the Branch Target Buffer is enabled and branch prediction occurs. If clear, no branch prediction will occur. See Paragraph 4.5.
RSTK_EN	0	If set, the Return Stack is enabled so RET instructions will speculatively execute the code following the associated CALL to improve performance. If clear, the Return Stack is not enabled and optimum performance will not be achieved. See Paragraph 4.5.

4.2 Configuration Control Registers

There are four configuration control registers in the 5x86 CPU (CCR1 through CCR4) that control the cache, power management, and other unique features. Figures 4-2 through 4-6 and Tables 4-2 through 4-8 describe the CCRs and, briefly, their application. It is recommended that the BIOS utilities allow OEMs to create a setup screen that includes the ability to program at least the following CCR bits.

a) WT1 (CCR2 bit 4)
b) LINBRST (CCR3 bit 2)
c) IORT(2-0) See Paragraph 4.5.

The following paragraphs provide diagrams of the registers and detailed descriptions of the configuration control register bits.

4.2.1 Configuration Control Register 1 (CCR1)

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
Reserved	*Reserved*	*Reserved*	*Reserved*	MMAC	SMAC	USE_SMI	*Reserved*

Figure 4-2. Configuration Control Register 1 (CCR1)

Table 4-2. CCR1 Bit Definitions

BIT NAME	BIT NO.	DESCRIPTION
MMAC	3	If set, data accesses to addresses within the SMM address space are issued to main memory instead of system management memory. This is only used within an SMM service routine to access normal memory which overlaps SMM memory.
SMAC	2	If set, any access to addresses within the SMM address space will access system management memory instead of main memory. SMI# input is ignored while SMAC is set. Setting SMAC to 1 allows access to SMM memory without entering SMM. This is useful for initializing or testing SMM memory.
USE_SMI	1	If set, SMI# and SMADS# pins are enabled. If clear, SMI# pin is ignored and SMADS# pin floats.

4.2.2 Configuration Control Register 2 (CCR2)

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
USE_SUSP	BWRT	*Reserved*	WT1	SUSP_HLT	LOCK_NW	USE_WBAK	*Reserved*

Figure 4-3. Configuration Control Register 2 (CCR2)

Table 4-3. CCR2 Bit Definitions

BIT NAME	BIT NO.	DESCRIPTION
USE_SUSP	7	If set, SUSP# and SUSPA# pins are enabled. If clear, SUSP# pin is ignored and SUSPA# pin floats. These pins should only be enabled if the external system logic (chipset) supports them.
BWRT	6	If set, enables burst write cycles. This should only be set if the system logic supports burst writes. If set, the CPU will perform a 4 dword burst write on line replacements and write-back cycles as a result of a snoop hit. See Paragraph 4.5.
WT1	4	If set, designates that any cacheable accesses in the address region 640 KBytes to 1 MByte are defined write-through. With WT1=1, any write to a cached value in this range will also be issued to the external bus.
SUSP_HLT	3	If set, execution of the HLT instruction causes the CPU to enter low power suspend mode. This bit should be used cautiously since the CPU must recognize and service an INTR, NMI, SMI or RESET to exit the "HLT initiated" suspend mode.
LOCK_NW	2	If set, the NW bit in CR0 becomes read only and the CPU ignores any writes to this bit. This should be set to 1 after setting the CR0 NW to prevent inadvertent modification of the NW bit.
USE_WBAK	1	If set, enables the CPU write-back cache interface pins which include CACHE#, INVAL, WM_RST, and HITM#. If WBAK is set to 0, INVAL and WM_RST are ignored and the HITM# and CACHE# outputs float.

4.2.3 Configuration Control Register 3 (CCR3)

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
MAPEN(3-0)				SMM_MODE	LINBRST	NMI_EN	SMI_LOCK

Figure 4-4. Configuration Control Register 3 (CCR3)

Table 4-4. CCR3 Bit Definitions

BIT NAME	BIT NO.	DESCRIPTION
MAPEN(3-0)	7-4	If set to 1h, all configuration registers are accessible. If clear, only configuration registers with indexes C0-CFh, FEh, and FFh are accessible.
SMM_MODE	3	If set, the CPU's SMM hardware interface pins are redefined to function like the SMM hardware interface pins on the Intel SL Enhanced 486.
LINBRST	2	If set, the CPU will use a linear address sequence when performing burst cycles. If clear, the CPU will use a “1+4” address sequence when performing burst cycles. The “1+4” address sequence is compatible with the Intel 486 burst address sequence.
NMI_EN	1	If set, NMI interrupt is recognized while in SMM. This bit should only be set while in SMM after the appropriate NMI interrupt service routine has been set up.
SMI_LOCK	0	If set, the CPU prevents modification of the following SMM configuration bits except when operating in SMM. CCR1: USE_SMI, SMAC, MMAC CCR3: NMI_EN, SMM mode SMAR: Starting address and block size. Once set, the SMI_LOCK bit can only be cleared by RESET.

4.2.4 Configuration Control Register 4 (CCR4)

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
CPUIDEN	*Reserved*	*Reserved*	DTE_EN	MEM_BYP	IORT(2-0)		

Figure 4-5. Configuration Control Register 4 (CCR4)

Table 4-5. CCR4 Bit Definitions

BIT NAME	BIT NO.	DESCRIPTION
CPUIDEN	7	If set, this bit enables writing to bit 21 of the EFLAG register (indicating that the CPU supports the CPUID instruction) and the CPUID instruction will execute normally. See Paragraph 4.5. If clear, bit 21 of the EFLAG register is not writable and the CPUID instruction is an invalid opcode. *This bit is only available on Stepping 1 and later versions of the 5x86 CPU. The CPUID instruction is disabled on all prior devices.*
DTE_EN	4	If set, the Directory Table Entry cache is enabled.
MEM_BYP	3	If set, Memory Bypassing is enabled. Data can be read from the write buffers prior to being written to external memory.
IORT(2-0)	2-0	Specifies the minimum number of bus clocks between I/O accesses (I/O recovery time). The delay time is the minimum time from the beginning of one I/O cycle to the beginning of the next (that is, ADS# to ADS# time). The default I/O recovery time is set to 32 bus clocks (0.97us at 33 MHz, 0.60 us at 50 MHz). If the chipset already has I/O recovery time implemented, the programmed delay should be set to no delay (0h) to optimize system performance. See Paragraph 4.5. 0h = no delay 4h = 16 clocks 1h = 2 clocks 5h = 32 clocks (default after RESET) 2h = 4 clocks 6h = 64 clocks 3h = 8 clocks 7h = 128 clocks

4.3 Power Management Register (PMR)

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
Reserved	*Reserved*	*Reserved*	*Reserved*	*Reserved*	HLF_CLK	CLK(1-0)	

Figure 4-6. Power Management Register (PMR)

Table 4-6. PMR Bit Definitions

BIT NAME	BIT NO.	DESCRIPTION
HLF_CLK	2	If set, the core clock frequency will run at half the external bus frequency whenever the bus interface is idle. When an external bus transfer occurs, the core clock frequency will automatically increase in frequency for the duration of the transfer. When the transfer is complete, the core will revert to half the frequency of the bus. See Paragraph 4.5.
CLK(1-0)	1-0	The CLK(1-0) bits can be used to change the internal core frequency relative to the bus frequency at any time. Any attempt to modify the core frequency higher than specified for the device will be ignored. These bits are initialized, during reset, to the clock multiplier specified by the CLKMUL pin (2X, 3X, etc.). The following core/bus frequency ratios are available: CLK(1-0) = 00: 1/1 CLK(1-0) = 01: 2/1 CLK(1-0) = 11: 3/1 CLK(1-0) = 10: 4/1 (not available on production parts) After reset, these bits reflect the current operating core/bus frequency. If the current core/bus frequency is 2/1 or 3/1, the mode can only be changed to 1/1 and then back to 2/1 or 3/1. If the mode after reset is 4/1, the mode can be changed among 4/1, 2/1 and 1/1. See Paragraph 4.5.

4.4 SMM Address Region Register (SMAR)

The SMAR is used to define the SMM memory region. The SMAR has three eight-bit registers associated with it that define the region starting address and block size. Table 4-6 below shows the format for the SMAR and lists the index assignments for the SMAR starting address and block size. The region starting address is defined by the upper 12 bits of the physical address. The region size is defined by the BSIZE(3-0) bits as shown in Tables 4-6 and 4-7. The BIOS and/or its utilities should allow definition of the SMAR. There is one restriction when defining the SMM address region: the region starting address must be on a block-size boundary. For example, a 128-KByte block is allowed to have a starting address at 0 KBytes, 128 KBytes, 256 KBytes, and so on.

Table 4-7. SMAR Index Assignments

STARTING ADDRESS			REGION BLOCK SIZE
A31-A24	A23-A16	A15-A12	BSIZE(3-0)
BITS (7-0)	BITS (7-0)	BITS (7-4)	BITS (3-0)
CDh	CEh	CFh	

Table 4-8. BSIZE(3-0) Bit Definitions

BSIZE(3-0)	REGION SIZE	BSIZE(3-0)	REGION SIZE
0h	Disabled	8h	512 KBytes
1h	4 KBytes	9h	1 MByte
2h	8 KBytes	Ah	2 MBytes
3h	16 KBytes	Bh	4 MBytes
4h	32 KBytes	Ch	8 MBytes
5h	64 KBytes	Dh	16 MBytes
6h	128 KBytes	Eh	32 MBytes
7h	256 KBytes	Fh	4 KBytes

4.5 Required Configuration Register Settings

It is recommended that certain Configuration Register bits be set to a specified value to assure full compatibility. Failure to set the registers to the values in Table 4-9 can lead to unpredictable behavior. Recommendations will change with later steppings of the CPU

Table 4-9. Required Bit Settings

Field Name	Register Location	Required Settings for Compatibility
LSSER	PCR0(7)	1 for PCI or VL systems
LOOP_EN	PCR0(2)	0
BTB_EN	PCR0(1)	0
RSTK_EN	PCR0(0)	0
CPUIDEN	CCR4(7)	0
IORT	CCR4(2-0)	000
HLF_CLK	PMR(2)	0
CLK(1-0)	PMR(1-0)	Keep at default setting: 11=3x, 01=2x
BWRT	CCR2(6)	0

5. Programming Model Differences versus i486

5.1 INVD and WBINVD Instructions

The INVD and WBINVD instructions are used to invalidate the contents of the internal and external caches. The WBINVD instruction first writes back any modified lines in the cache and then invalidates the contents. It ensures that cache coherency with system memory will be maintained regardless of the cache operating mode, write-through, or write-back. Following invalidation of the internal cache, the CPU generates a special bus cycles to indicate that external caches should also write back modified data and invalidate their contents.

On the 5x86 CPU, the INVD functions identically to the WBINVD. The 5x86 CPU always writes all modified internal cache data to external memory before invalidating the internal cache contents. In contrast, the Pentium invalidates the contents of its internal caches without writing back the “dirty” data to system memory. The Pentium's behavior could result in a data incoherency between the CPU's internal cache and system memory.

5.2 Control Register 0 (CRO) CD and NW Bits

The CPU's CR0 register contains, among other things, CD and NW bits which are used to control the on-chip cache. CR0, like the other system-level registers, is accessible only to programs that run at privilege 0, the highest privilege level. Table 5-1 lists the cache operating modes for the possible states of the CD and NW bits.

The CD and NW bits are set to 1 (cache disabled) after reset. For highest performance, the cache should be enabled in write-back mode by setting CD to 0 and NW to 1. Sample code for enabling the cache is in Appendix C. To completely disable the cache, it is recommended that CD and NW be set to 1 as shown in Tabel 5-1. The 5x86 cache will always accept invalidation cycles, even when the cache is disabled.

Table 5-1. Cache Operating Modes

CD	NW	OPERATING MODES
1	1	Cache disabled (recommended setting and default after reset)
1	0	Cache disabled
0	1	Cache enabled in write-back mode
0	0	Cache enabled in write-through mode

6. Initialization Sequence

The 5x86 CPU features and capabilities described above should be enabled/initialized during BIOS POST. A recommended sequence for initializing the 5x86 CPU is shown below.

1) Program PCR0, DTE_EN, and MEM_BYP for optimum performance and compatibility.
2) Program I/O recovery time using IORT(0-2) in CCR4. Set to zero unless needed by system.
3) Program WT1.
4) Define SMAR.
5) Program USE_SUSP.
6) Program SUSP_HLT.
7) Enable cache (see Appendix C for example code).
8) Program LOCK_NW.
9) Initialize/enable SMM (see Cx486DX/DX2 SMM Programmer's Guide).

Appendix A - Programming 5x86 CPU Configuration Registers

Sample code for setting USE_SMI=1 in CCR1.

```
mov   al,  0c1h   ; set index for CCR1
out   22h, al     ; select CCR1 register
in    al,  23h    ; read current CCR1 value
or    al,  02h    ; set USE_SMI bit
mov   ah,  al
mov   al,  0c1h   ; set index for CCR1
out   22h, al     ; select CCR1 register
mov   al,  ah
out   23h, al     ; write new value to CCR1
```

Appendix B - Detecting a Cyrix CPU

```
assume  cs:_TEXT

public   _iscyrix

_TEXT segment byte public 'CODE'

;*****************************************************************
;       Function: int iscyrix ()
;
;       Purpose:        Determine if Cyrix CPU is present
;       Technique:      Cyrix CPUs do not change flags where flags
;                       change in an undefined manner on other CPUs
;       Inputs:         none
;       Output:         ax == 1 Cyrix present, 0 if not
;*****************************************************************
_iscyrix    proc   near
            .486
            xor   ax, ax         ; clear ax
            sahf                 ; clear flags, bit 1 always=1 in flags
            mov   ax, 5
            mov   bx, 2
            div   bl             ; operation that doesn't change flags
            lahf                 ; get flags
            cmp   ah, 2          ; check for change in flags
            jne   not_cyrix      ; flags changed, therefore NOT CYRIX
            mov   ax, 1          ; TRUE Cyrix CPU
            jmp   done

not_cyrix:
            mov  ax, 0           ; FALSE NON-Cyrix CPU
done:
            ret
_iscyrix    endp

_TEXT ends
end
```

Appendix C - Enabling the 5x86 CPU Cache

Sample code for enabling the 5x86 CPU write-back cache and setting LOCK_NW=1.

```
;**** disable cache
 mov  eax, cr0
 or   eax, 040000000h   ;  CD bit in CR0 = 1
 mov  cr0, eax
; **** write back and invalidate the cache
wbinvd
; **** enable write back mode
 mov  eax, cr0
 and  eax, 0ffffffffh   ; NW bit in CR0 = 1
 mov  cr0, eax
; **** set LOCK_NW=1 to prevent modification of NW
 mov  al, 0c2h          ; set index for CCR2
 out  22h, al           ; select CCR2 register
 in   al, 23h           ; read current CCR2 value
 or   al, 04h           ; set LOCK_NW=1
 mov  ah, al
 mov  al, 0c2h          ; set index for CCR2
 out  22h, al           ; select CCR2 register
 mov  al, ah
 out  23h, al           ; write new value to CCR2
; ****enable cache
 mov  eax, cr0
 and  eax, 0bfffffffh   ; CD bit in CR0 = 0
 mov  cr0, eax
```