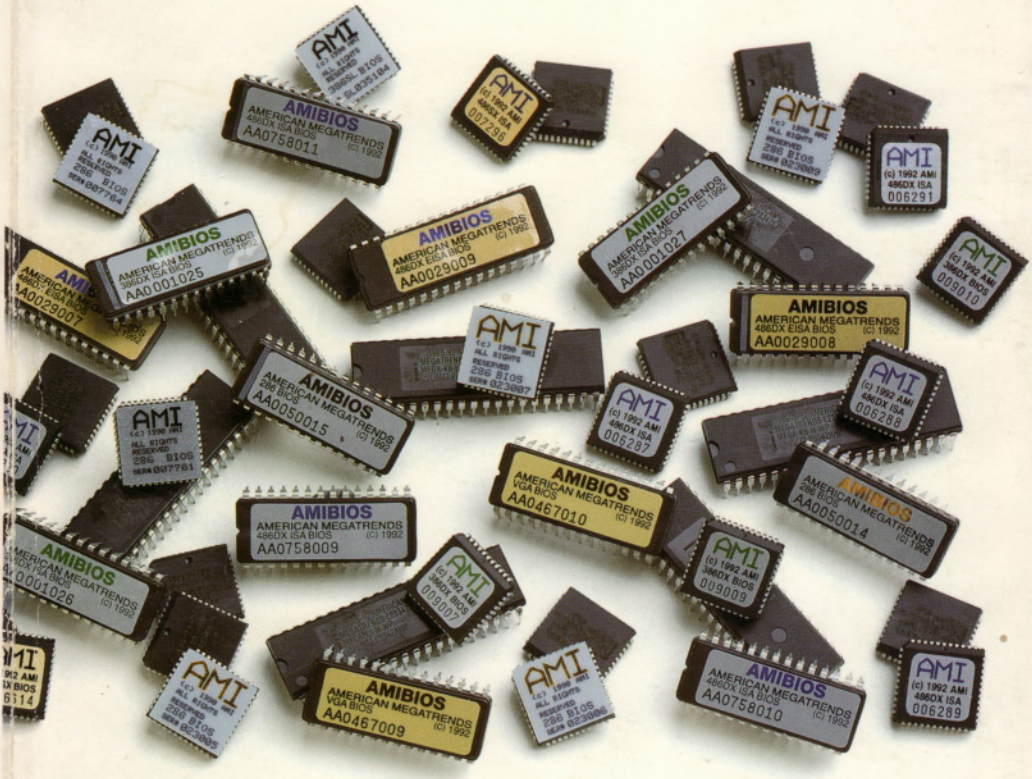
American Megatrends

ISA and EISA

Hi-Flex AMIBIOS Technical Reference



Hi-Flex AMIBIOS Technical Reference

American Megatrends

American Megatrends, Inc.

Hi-Flex ISA and EISA AMIBIOS

Technical Reference

MAN-PR-BIOSTR
8/1/93

American Megatrends, Inc.
6145F North Belt Parkway
Norcross, GA 30071

Limited Warranty

No warranties are made, either express or implied, with regard to the contents of this work, its merchantability, or fitness for a particular use.

American Megatrends assumes no responsibility for errors and omissions or for the uses made of the material contained herein or reader decisions based on such use.

Limitations of Liability

In no event shall American Megatrends be held liable for any loss, expenses, or damages of any kind whatsoever, whether direct, special, indirect, incidental, or consequential, arising from or arising out of the use or inability to use the contents of this manual.

Trademarks

American Megatrends acknowledges the following trademarks:

Intel and Pentium are registered trademarks of Intel Corporation.
AMD is a registered trademark of Advanced Micro Devices, Inc.
Cyrix is a registered trademark of Cyrix Corporation.
DesqView is a registered trademark of Quarterdeck Office Systems, Inc.
MS-DOS, Microsoft Windows, Windows NT, Xenix, and Microsoft are registered trademarks of Microsoft Corporation.
Unix is a registered trademark of American Telephone and Telegraph Company Bell Laboratories.
Austek is a trademark of Austek Corporation.
Headland Technologies is a registered trademark of Headland Technologies, Inc.
Weitek is a registered trademark of Weitek Corporation.
IBM, AT, VGA, EGA, XGA, PS/2, MCA, OS/2, and MicroChannel are registered trademarks of International Business Machines Corporation. PC-DOS, XT, and CGA are trademarks of International Business Machines Corporation.
SMC is a registered trademark of SMC Corporation.
Avocet is a registered trademark of Avocet Systems, Inc.

Revision History

4/92	Initial release.
10/92	Revised for 6/6/92 core AMIBIOS.
12/92	Corrected minor errors.
5/24/93	Added APM, PCI, and Socket Service BIOS function documentation.
8/1/93	Updated to PCMCIA Version 2.00 Socket Service specifications.

Table of Contents

Table of Contents, Continued

Table of Contents, Continued

Table of Contents, Continued

Table of Contents, Continued

Table of Contents, Continued

Table of Contents, Continued

Preface

To the Reader

This manual provides technical details about the operation of the AMIBIOS for ISA and EISA systems. We assume that you are familiar with the Intel x86 architecture, x86 assembler language, and both ISA and EISA system architecture. Common acronyms and abbreviations are listed in the Glossary, which begins on page 421.

Technical Support

Call American Megatrends Technical Support at 404-246-8600 if you have a problem with any AMIBIOS. The Technical Support Department also operates a Bulletin Board Service. The BBS supplies technical information about all American Megatrends products, including Product Tech Tips, the latest information about chipset-specific BIOS products, and technical documentation.

BBS

The American Megatrends BBS permits OEMs, VARs, system integrators, and all customers to access technical information about motherboard, software, and BIOS products. Product Engineering Change Notices, Tech Tips, Technical Notes, and technical manuals are available on the BBS.

Data Transmission Rates

The BBS automatically handles modems with data transmission rates from 1,200 to 14,400 bps.

If using an HST modem, call 404-246-8780.

If using a non-HST modem, call 404-246-8782.

BBS Phone Numbers

The following table lists the characteristics of the BBS phone numbers. The BBS requires no parity, 8 data bits, and 1 stop bit.

Phone Number	Characteristics
404-246-8780	Supports HST and v.42bis.
404-246-8781	Supports HST and v.42bis.
404-246-8782	Dual standard. Can handle 2400 or 9600 bps. Supports v.32bis and v.42bis. Can handle up to 14,400 baud.
404-246-8783	Supports v.32bis and v.42bis.

Acknowledgments

American Megatrends Senior BIOS engineers Shankar Mandal, Sandip Datta Roy, Sanjeev Kumar, and Debkumar De developed AMIBIOS. Engineers who contributed include Bob Gordon, Jeff Kidd, Charles Hanes, Chip Aaron, John Pennington, Lee Davis, Dick Holmberg, Eddy Quicksall, Will Gysin, Thomas Gilreath, Terry Lauer, Mohan Nair, Sanjay Sehgal, Nandkumar Phadte, and Vivek Saxena. This manual was written by Anindya Mukherjee and Paul Narushoff.

Chapter 1

Introduction

The architecture of ISA and EISA systems is layered. The lowest layer is the computer — the hardware itself. The highest layer is the applications software that the user interfaces with. Systems software lies between applications software and hardware.

Systems software can consist of several elements: the operating system kernel, the operating system shell, and additional device drivers. Operating environments (Microsoft® Windows®, for example) exist in a layer between the operating system and applications software, as do multitasking supervisors or DOS extenders like Desqview®.

The BIOS (Basic Input/Output System) is a collection of routines between the hardware and the systems software. The BIOS ROM contains hard disk utilities, device drivers, interrupt service routines, and other code and data between the system hardware and the systems software.

Types of Systems Discussed

This manual discusses the AMIBIOS for ISA and EISA systems. These systems contain Intel 80286, 80386SX, 80386DX, 80486SX, or 80486DX microprocessors.

It does not discuss IBM® PC, XT™, PS/2®, or compatible systems.

Overview

In This Chapter

The following topics are discussed in this chapter:

- BIOS as Interface,
- Parts of the ROM BIOS, and
- Types of BIOS.

The BIOS as Interface

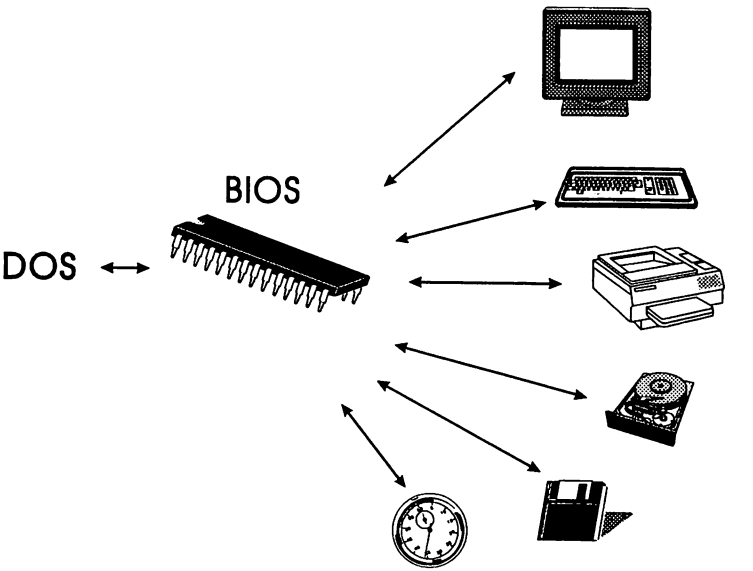
The BIOS is the software layer between the systems software and the hardware in ISA and EISA systems. The BIOS works in two directions.

One part of the BIOS receives and processes requests from programs to perform the standard BIOS I/O services. The mechanism for these requests is called an interrupt, discussed in detail beginning on page 117. Interrupts are invoked by software programs. In an assembler program, the INT mnemonic is followed by an interrupt number that specifies the type of service and a function number that specifies the exact service to be performed. For example:

```
MOV     AH,00h  ;specifies function 00h get ;character from
                keyboard
INT     16h     ;requests INT 16h Keyboard Service
```

BIOS and the Hardware

The other side of the BIOS communicates with the hardware (video display, disk drives, keyboard, serial and parallel ports, and so on) in the language and codes used by each device. This side of the BIOS also handles any hardware device-generated interrupts. For example, when a key is pressed on the keyboard, a hardware interrupt (IRQ1) is generated. The BIOS INT 09h interrupt service routine is called to handle this interrupt. The following figure illustrates the role that the system BIOS plays.

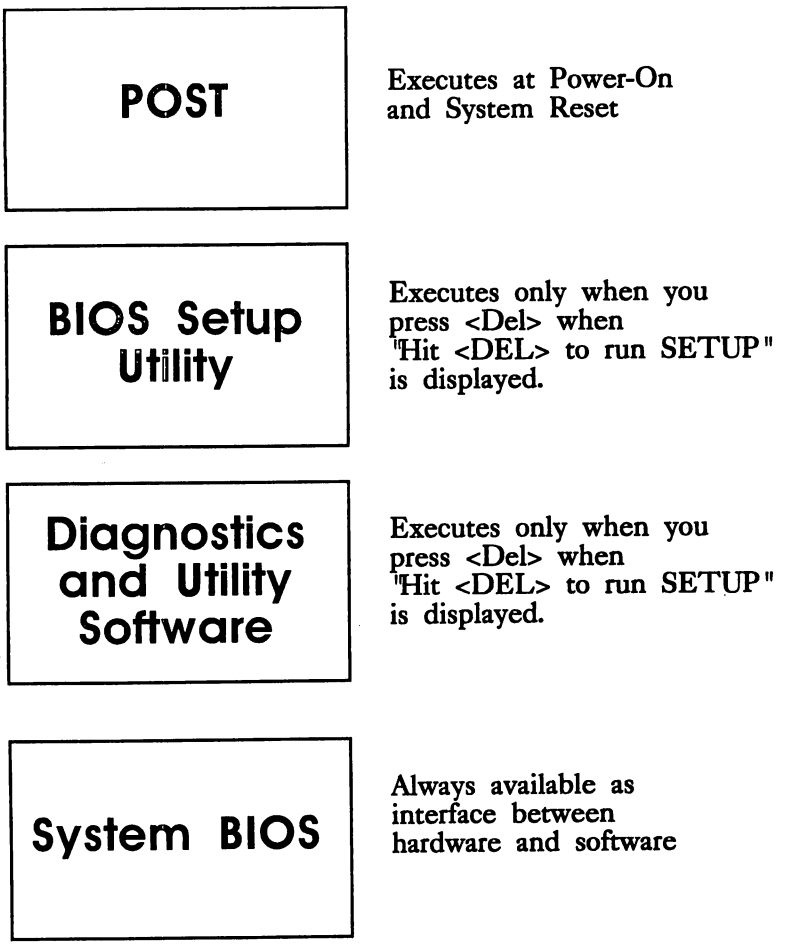


Parts of the System ROM BIOS

The four system ROM elements in a computer with an AMIBIOS are:

- POST,
- the ROM BIOS itself,
- the hard disk utility programs, and
- the BIOS Setup utility.

The following graphic depicts these elements:



POST

ROMs includes BIOS Power-On Self Test diagnostic and booting code that tests the system components, initializes certain data structures, and boots DOS. The Power-On Self Test (POST) in the system BIOS performs several functions. POST:

- executes a diagnostic and reliability test of the system, the ROM programs, and system RAM,
- initializes the chips and the standard parts of the computer system and places a record of the system configuration in CMOS RAM and in low system memory,
- sets up the interrupt vector table,
- detects optional equipment in the system, and
- boots the operating system.

BIOS POST is discussed in greater detail beginning on page 90.

cont'd

System BIOS

The BIOS is a part of the code stored in ROM that is in active use the entire time that a computer is on. The ROM BIOS provides the fundamental services needed for the proper operation of the system.

Hard Disk Utility

The hard disk utilities in the AMIBIOS ROMs include formatting, auto interleave and media analysis.

AMIBIOS Setup Utility

AMIBIOS Setup stores system configuration values in CMOS RAM. The hard disk drive type, type of floppy drives and monitor, and the day, date, and time can be set through Standard CMOS Setup.

Newer versions of the AMIBIOS have Advanced CMOS Setup and Advanced CHIPSET Setup to configure advanced system characteristics, such as RAM and ROM wait states, DMA Clock origination, and memory relocation.

Types of BIOS

In ISA and EISA computers, the types of BIOS (Basic Input Output System) include:

- the system BIOS,
- the video BIOS,
- optional adaptor ROM BIOSes, and
- the keyboard controller BIOS.

This manual describes the system and keyboard controller BIOS features. The video BIOS is best discussed in the context of EGA®, VGA®, and XGA® video standards, which all require a separate BIOS.

System BIOS

The BIOS tests the system components, loads (bootstraps) the operating system, and remains active for requests by the operating system to activate device drivers that service the hardware components. The BIOS takes the instructions from the operating system and translates these commands to the exact instructions that the hardware itself understands. The BIOS maintains data about various system components. When a component is unable to perform, the BIOS reports it to the operating system.

The system ROM BIOS code is 64 KB long and resides at F0000h – FFFFFh in ISA systems. In EISA systems, it is 128 KB long and resides at E0000h – FFFFFh. This area is addressed to ROM but can be shadowed to RAM. ROM operates at about 120 – 180 ns; RAM usually operates at 60 – 100 ns. Shadowing is discussed in detail on page 11.

cont'd

Video BIOS

All ISA and EISA systems that use EGA, VGA, or XGA have video BIOSes. The system BIOS video service only handles the most basic video functions. Advanced video modes must be translated via a video BIOS, usually installed on the video adapter card.

Keyboard Controller BIOS

Every ISA and EISA system must also have a keyboard controller BIOS to translate the signals from the keyboard into codes that the BIOS and the system can understand. The keyboard controller BIOS is discussed in detail beginning on page 361.

Adaptor ROM BIOS

Many adapter cards have code in ROM. For example, ESDI hard disk drive controllers have a ROM that assists in translating this interface to code that the computer can understand and vice versa.

Adaptor ROM resides between C8000h and EFFFFh. This area also can be shadowed in AMIBIOSes to speed the operation of the devices that have adaptor ROMs, provided that the motherboard or chipset used in the system supports adaptor ROM shadowing.

Chapter 2

AMIBIOS Features

The three types of BIOS features in the AMIBIOS are:

- standard features,
- Setup-dependent features, and
- features dependent on AMIBCP options.

Standard BIOS Features

- automatically detects processor type,
- automatically detects memory size,
- configures non-standard systems,
- supports extended BIOS services,
- supports hardware-specific features,
- peripheral controller support,
- cache controller support,
- parallel and serial port support,
- shadowing,
- diagnostics,
- hard disk utilities, and
- autodetect hard disk.

AMIBIOS Setup features are accessed by pressing Del when

`Hit <DEL> if you want to run SETUP` *(or DIAGS in older BIOS)*

Automatically Detects Processor Type

The BIOS knows whether an 80386SX, 80386DX, 80486SX, 80486DX, 80486DX2, Pentium®, or other Intel or compatible processor (AMD® 386DXL, Cyrix® Cx486SLC, Cx486DLC, Cx486S, Cx486S2, or IBM 486SLC or 486SLC2) is present and executes accordingly.

cont'd

Automatically Detects Memory Size

AMIBIOS checks system and cache memory and reports the amount on the initial BIOS screen and the BIOS System Configuration Screen. In systems with more than 1 MB, the BIOS reports 384K less RAM than it finds, to account for the address space between 640K and 1024K unavailable to DOS.

Configures Non-Standard Systems

You can configure systems that are missing a keyboard, monitor, or disk drive through AMIBIOS SETUP. Select *Not Installed* as the setting for the missing device in Standard CMOS Setup. All missing device error messages are suppressed, permitting normal boot.

Supports Extended BIOS Services

AMIBIOS includes INT 14h Function 04h Extended Initialize and Function 05h Extended Serial Port Control, a PS/2®-compatible BIOS feature.

INT 15h Functions are: C1h and C2h for PS/2-type mouse support, INT 15h Function C3h Fail-Safe Timer Enable, which makes sure a program does not turn off interrupts for too long, INT 15H APM functions, and the EISA AMIBIOS supports INT 15h Function D8h, EISA Configuration (see page 220).

INT 16h support includes several useful functions that no other BIOS has: Function F0h Set CPU Speed, Function F1h Read CPU Speed, Function F4h, Subfunction 00h Read Cache Controller, Subfunction 01h Enable Cache Controller, and Subfunction 02h Disable Cache Controller.

INT 1Ah support includes Card Services, Socket Services, and PCI BIOS functions. Both the ISA and EISA AMIBIOS support the use of hardware interrupt IRQ12 for the mouse.

Supports Hardware-Specific Features

Many ISA chipsets have paged memory, memory interleaving, EMS support, and power management features. An AMIBIOS for a specific chip must be customized to support some of these features. The BIOS in your computer may have special Setup options to configure these features. You may not be able to access these advanced options, since the manufacturer can customize BIOS Setup to automatically configure them.

Peripheral Controller Support

AMIBIOS has built-in support for National Semiconductor® PC87310, PC87311, and PC87312, C&T® 82C710, 82C711, 82C712, 82C721, and 82C725, VLSI® 82C106 and 82C107, Intel® 82341, and other peripheral controllers.

Cache Controller Support

AMIBIOS has built-in support for Intel C6, Headland Technologies® HT44, Austek, and other standalone cache memory controllers. Many chipsets have integrated cache controllers.

Parallel and Serial Port Support

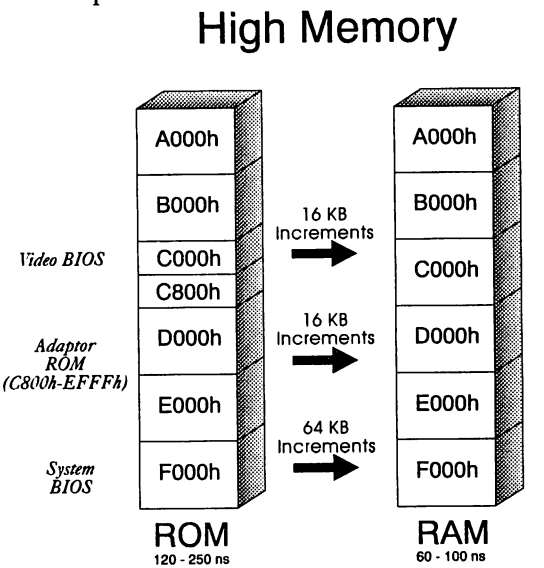
AMIBIOS supports up to four serial ports and four parallel ports. The fourth parallel port is not supported if PS/2-type mouse support is enabled.

Shadowing

Shadowing copies the BIOS from ROM to RAM to improve system performance. In a system with no shadow option, the ROM BIOS is executed from relatively slow ROM (150 – 250 ns). The BIOS executes much faster when the ROM BIOS is copied to RAM (60 – 100 ns) and the system is instructed to access the BIOS from RAM. In most cases, system BIOS shadowing should never be turned off.

System BIOS ROM Shadowing

The system BIOS resides in the 64 KB address space between F0000h and FFFFFh in ISA systems (and in the 128 KB space between E0000h and FFFFFh in EISA systems). The system BIOS shadow feature is often automatically enabled by the AMIBIOS. If not, it is an option on Standard CMOS Setup or Advanced CMOS Setup screens that should always be enabled to enhance BIOS performance.



In shadowing, the BIOS code is copied from slow ROM to faster RAM.

Adaptor ROM Shadow

The area between C8000h and EFFFFh in ISA systems (and C8000h – DFFFFh in EISA systems) is available for use by other ROM devices. Often, the hard disk drive controller ROM is stored here. Many network cards also use this space. AMIBIOS allows this area to be shadowed from ROM to RAM to speed access to the code in this area.

EGA and VGA RAM Shadow

Video ROM shadowing can often speed execution in two ways: running the video BIOS from 16-bit instead of 8-bit memory, and running the video BIOS from fast RAM instead of relatively slow ROM.

The memory space from C0000h – C7FFFh is reserved for video ROM. Often, only the EGA BIOS (C0000h - C3FFFh), accessed through an 8-bit bus, is located in this area. The relatively slow execution of this device driver from ROM makes the video I/O slow. AMIBIOS allows you to map this space to system RAM, where it executes about twice as fast.

Video ROM shadowing copies the video ROM from C0000h–C7FFFh to RAM. Memory from C0000h – DFFFFh can be accessed on the 16-bit expansion slot. Any 8-bit I/O memory in that space is automatically disabled.

Diagnostics

Older AMIBIOS included hardware diagnostic routines. All recent AMIBIOS include only hard disk drive utilities.

Hard Disk Utilities

The Hi-Flex AMIBIOS, available since early 1991, provides several hard disk utilities, including format, auto interleave, and media analysis.

Autodetect Hard Disk

AMIBIOS Setup in newer AMIBIOS products reads and reports configuration information for SCSI, IDE, and other non-ST512 hard disk drives.

AMIBIOS Setup Features

The following AMIBIOS Setup utility features are only available in the Hi-Flex AMIBIOS (available since early 1991) with Advanced CMOS Setup.

Type of Feature	Setup Options
Memory Features	▪ **Above 1 MB Memory Test** Executes the POST memory routines on the RAM above 1 MB (if present on the system). If disabled, the BIOS only checks the first 1 MB of RAM. ▪ **Memory Test Tick Sound** Turns the ticking sound on or off. ▪ **Memory Parity Error Check** Enables or disables parity error checking for all system RAM. ▪ **Hard Disk Type 47 RAM Area** Specifies the type 47 data storage area – 0:300h in lower system RAM or in the top 1 KB of memory, starting at address 639K or 511K (depending on the amount of base memory). Type 47 data is stored in shadow RAM if shadowing is enabled. ▪ **Fast Gate A20 Option** Fast Gate A20 is a hardware circuit that enables Gate A20 faster. The BIOS controls access to this circuitry. Address Gate A20 in the Intel x86 architecture controls access to memory addresses above 1 MB by enabling or disabling access to processor address line 20. Some programs both enter protected mode and use the CMOS RAM Shutdown byte to return to real mode through the BIOS. For these programs, Gate A20 must be constantly enabled and disabled by the keyboard controller, which is a slow process.

Type of Feature	Setup Options
Cache Memory Control	■ **Internal Cache Memory** appears only on 80486-based systems. It enables or disables access to the 8 KB internal cache in the microprocessor. ■ **External Cache Memory** appears only on systems that have a caching scheme external to the CPU. This option enables or disables the testing and autosizing of cache memory in POST.
Keyboard and Mouse Control Features	■ **Typematic Rate Programming** Typematic Rate Programming enables or disables the Typematic Rate Delay and Typematic Rate options. ■ **Typematic Rate Delay (milliseconds) and** ■ **Typematic Rate (Characters per Second)** Typematic Rate Delay and Typematic Rate control the speed at which a keystroke is repeated. The character associated with the keystroke is repeatedly displayed when a key is pressed and held down. After the Typematic Rate Delay, the character repeats at a rate set by the Typematic Rate. ■ **System Boot Up Num Lock** You can turn off the **Num Lock** function when the system is powered on. You can use both sets of arrow keys on the keyboard when the **Num Lock** function is turned off. ■ **Mouse Support Option** Enables support for a PS/2-type mouse or pointing device. If this option is disabled, the BIOS does not reserve the top 1 KB of the DOS applications area memory (639K or 511K depending on the amount of base memory) for the extended BIOS Data Area.
Message Display Control Features	■ **Hit <DEL> Message Display** Disabling this option prevents *Hit <DEL> to run Setup* , from appearing when the system boots. ■ **Wait for <F1> if Any Error** Disabling this option eliminates the need for user responses to Press **F1** to continue.

AMIBIOS Setup Features, Continued

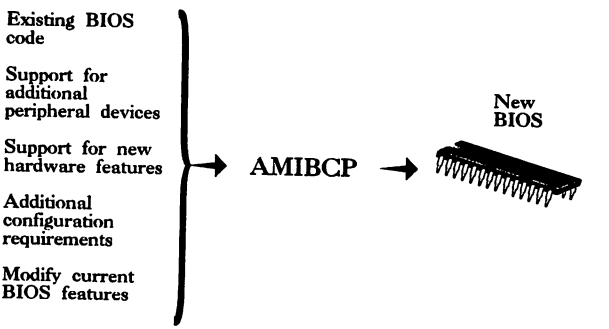
Type of Feature	Setup Options
Coprocessor Enable Features	■ **Numeric Processor** Enable BIOS testing for a math coprocessor. ■ **Weitek® Processor** Enable BIOS testing for a Weitek math coprocessor.
Boot Up Options	■ **Floppy Drive Seek at Boot** This option performs a Seek on drive A: at system boot. The default is Disabled for a faster boot. ■ **System Boot Up Sequence** The system can boot first from drive A: or drive C:. ■ **System Boot Up Speed** Sets the speed at which the system boots.
Speed Control	■ **Turbo Switch Function** Enables the system turbo (processor speed switching) switch, if this switch is supported in hardware.
Security Features	■ **Password Check Option** The password option prevents unauthorized system boot or AMIBIOS Setup use. ■ **Boot Sector Virus Protection** Newer AMIBIOS products automatically report when any program attempts to format or write to the boot sector on a hard disk drive.

AMIBCP-Dependent BIOS Features

The Hi-Flex AMIBIOS has many other features that are actually options set by the system manufacturer via AMIBCP (BIOS Configuration Program). This BIOS has been available since early 1991.

Before AMIBCP, most AMIBIOS features had to be individually customized by American Megatrends. With AMIBCP, the system manufacturer can customize the BIOS, resulting in much faster system development and a much greater degree of freedom in adding or changing system features.

AMIBCP controls which of the five Setup screens appear on the Setup menu, the options that appear in the Setup menus, and the settings that each option can be configured to. AMIBCP configures the implementation of processor speed switching, cache memory control, the turbo switch option, and cache memory controller implementation, as well as many other BIOS features. The following graphic illustrates how AMIBCP is used.



cont'd

AMIBCP Options that affect BIOS Setup Options

The following BIOS features can be configured or disabled by the OEM or system manufacturer through AMIBCP:

- Halt on Error During POST,
- Initialize CMOS RAM at Every Boot,
- Keyboard Controller Output Pin 23, 24 Blocked,
- Mouse Support in BIOS and Keyboard Controller,
- Wait for <F1> In Case of POST Error,
- Display Floppy Error During POST,
- Display Video Error During POST, and
- Display Keyboard Error During POST.

The following BIOS features appear on the AMIBIOS Setup menu screen. All options may not appear in the AMIBIOS Setup utility screen in your computer because they can be turned on and off by the OEM before the system is sold. Some of the above options are only implemented in certain types of systems. For example, the *Power Management BIOS Setup Option* only appears in battery-backed portable, notebook, handheld, and other computers that have power management features.

- CMOS Setup Option,
- Advanced CMOS Setup Option,
- Advanced CHIPSET Setup Option,
- Power Management BIOS Setup Option,
- Peripheral Setup Option,
- BIOS Setup Defaults Auto Configuration Option,
- Power-On Defaults Auto Configuration Options,
- Change Password Options, and
- Hard Disk Utility Option.

Speed Selector and Cache Enable Keychord

You can define any unused keystroke combination (keychord) using AMIBCP to configure keyboard speed switching and enable or disable cache memory via a keychord. The following table shows the default settings.

Feature	Default Keychord
High speed	<Ctrl><Alt><gray +>
Low speed	<Ctrl><Alt><gray ->.
Enable cache memory	<Ctrl><Alt><Shift> <gray +>
Disable cache memory	<Ctrl><Alt><Shift> <gray ->

Software I/O Delay

If the software delay is 0 (the default value), it specifies a fixed delay of 15 µseconds. It can be set anywhere from 0 to 255 units. Each nonzero unit adds a fixed delay value to the processor speed. This option may be set as follows:

If the system speed is...	select...
16	10
20	12
25	14
33	18

Power-On Delay

This option specifies a short delay when power is turned on to allow the power supply output to stabilize. This parameter can be set to 0 – 255 seconds. The default is 0 seconds in an AMIBIOS based on the 6/6/92 core BIOS and 2 seconds in an AMIBIOS based on the 12/12/91 core BIOS.

cont'd

Refresh Value

This is the value used to program the refresh timer. This parameter sets the refresh value for system RAM. The default is 18, which translates to 15 µseconds. The range of values is 0 – 255.

Serial and Parallel Ports 1 – 4

Any of the I/O ports can be set to a beginning I/O port address defined by the OEM. The range of valid values is from 00000h – FFFFFh. The default starting I/O ports are:

Port	Default Starting I/O Port
Serial Port 1	3F8h
Serial Port 2	2F8h
Serial Port 3	3E8h
Serial Port 4	2E8h
Parallel Port 1	3BCh
Parallel Port 2	378h
Parallel Port 3	278h
Parallel Port 4	None

Parallel Port 4 is not supported if PS/2-type mouse support is enabled.

Speed Switching, Turbo Switch, Cache Control

Four BIOS features are implemented via 8042 pins, chipset-specific internal registers, or I/O ports. The following table lists these BIOS features and the possible methods used to implement the feature via AMIBCP. A discussion of the specific means of implementation follows this table.

Feature	8042 pin	Internal registers	I/O Ports
Clock Switching	Yes	Yes	Yes
Cache Control	Yes	Yes	Yes
Turbo Switch Input Pin	Yes	No	No
Memory Controller Reset	Yes	No	No

Implementation Methods Are Not Mutually Exclusive

If a keyboard pin implements processor speed switching, another keyboard pin can be used to implement cache control, or turbo switch, or memory controller reset.

If special I/O ports are used to implement speed switching, other I/O ports can also be used to implement cache control.

cont'd

Using Internal Registers

If processor speed switching or cache control is implemented via internal registers, the implementation method must be specific to the system architecture and to the particular chipset used in the system.

Processor Speed Switching

Systems using the Hi-Flex AMIBIOS allow the end user to change the processor speed using the keychord <Ctrl> <Alt> <Gray +> (low to high) and <Ctrl> <Alt> <Gray -> (high to low). These default keychords and the transitions they cause can be changed using AMIBCP.

A *speed switching method* must be specified in AMIBCP for the BIOS to recognize that a keychord causes a speed change. AMIBCP supports three speed switching methods. Any deviations from these methods may require BIOS customization. These methods are:

- using an 8042 pin,
- using internal registers, or
- using I/O ports.

Any of these three methods can be used to implement processor speed switching in a system with an AMIBIOS.

Processor Speed Switching Via an 8042 Pin

If the system has a Keyboard Controller AMIBIOS (version F or later), speed switching can be implemented using an 8042 pin to toggle the clock speed. The options:

- No Speed switching through Keyboard Controller
- Pin 23, 24 High means High Speed
- Pin 23, 24 Low means High Speed
- Pin 23 High means High Speed
- Pin 23 Low means High Speed
- Pin 24 High means High Speed
- Pin 24 Low means High Speed
- Pin 27 High means High Speed
- Pin 27 Low means High Speed
- Pin 28 High means High Speed
- Pin 28 Low means High Speed
- Pin 29 High means High Speed
- Pin 29 Low means High Speed
- Pin 30 High means High Speed
- Pin 30 Low means High Speed

8042 Pin — Method of Operation

If *No Speed switching through Keyboard Controller* is chosen, the BIOS does not recognize any keychord to toggle system speed through the keyboard controller. If one of the fourteen pin options is chosen, the BIOS toggles the voltage level of the chosen pin. The voltage level change switches the system speed if the appropriate hardware is implemented on the motherboard.

For example, if *Pin 23 High means High Speed* is chosen, the BIOS toggles the voltage level of 8042 pin 23 when the processor speed switching keychord is pressed. If the pin level is High, the speed is set High. If the pin level is Low, the speed is set to Low. 8042 Pins 23, 24, 27, or 28 cannot be used if the BIOS has PS/2 mouse support.

cont'd

Processor Speed Switching Via Internal Registers

Some chipsets allow speed switching to be implemented via chipset registers. The routine to toggle the speed is activated within the BIOS. The speed switching keychord toggles the system speed. This method can be enabled or disabled through AMIBCP.

Processor Speed Switching Via I/O Ports

Speed switching can also be accomplished using specially designated port address(es) via AMIBCP. To use this method, the port addresses, port data values, and port mask values must be specified for both the high and low speed.

The port value is read and ANDed with the mask value. The new value is ORed with the data value. The final value is written back to the port.

Processor Speed Switching Via I/O Ports, cont'd

Example

Assume the port address for high speed is 1234h. Bit 7 of the value read should be 0 to toggle the speed Low. When the speed switching keychord is pressed, the BIOS reads the port value as FFh.

The mask value is ANDed with the port value (7Fh). The new value is 7Fh.

Bit 7 is now 0. The BIOS ORs the I/O port contents with the data value (00h), producing a final value (7Fh) that is written back to port 1234h. The speed toggling hardware knows that bit 7 set to 0 means Low speed.

Assumed Speed Switching Conditions

It is assumed that the decoding circuitry and the functional toggling circuitry are provided on the motherboard. The BIOS reads and writes to the specified ports when the speed switching keychord is pressed. The actual speed change circuitry must be implemented in the motherboard design.

Both High speed and Low speed I/O ports should allow read and write operations and are limited to a four-digit hex address.

cont'd

Cache Control Options

Systems using the Hi-Flex AMIBIOS allow the end user to enable or disable the system cache memory using the keychord <Ctrl> <Alt> <Shift> <Gray +> (disabled to enabled) and <Ctrl> <Alt> <Shift> <Gray -> (enabled to disabled). These default keychords and the transitions they cause can be changed using AMIBCP.

For the BIOS to recognize that a keychord causes a cache enable or disable change, a *cache control method* must be specified, and the OEM can choose a method via AMIBCP. Three methods are supported in AMIBCP. Any deviations from these methods may require BIOS customizations. These methods are:

- a pin on the 8042 keyboard controller,
- chipset internal registers, or
- specially assigned I/O ports.

Cache Control Options, cont'd

Using a Pin on the 8042 Keyboard Controller

If the system has a Keyboard Controller AMIBIOS (version F or later), cache control can be implemented via an 8042 pin for the cache memory status. The options are:

- No Cache Control through Keyboard Controller
- Pin 23, 24 High means Cache ON
- Pin 23, 24 Low means Cache ON
- Pin 23 High means Cache ON
- Pin 23 Low means Cache ON
- Pin 24 High means Cache ON
- Pin 24 Low means Cache ON
- Pin 27 High means Cache ON
- Pin 27 Low means Cache ON
- Pin 28 High means Cache ON
- Pin 28 Low means Cache ON
- Pin 29 High means Cache ON
- Pin 29 Low means Cache ON
- Pin 30 High means Cache ON
- Pin 30 Low means Cache ON

If *No Cache Control through Keyboard Controller* is chosen, the BIOS does not recognize any keychord to toggle the cache memory status. If an 8042 pin option is chosen, the BIOS toggles the voltage level of the chosen pin. The different pin voltage levels change cache memory status, provided the appropriate hardware is implemented on the motherboard.

For example, if *Pin 23 High means Cache ON* is chosen, the BIOS toggles the voltage level of 8042 pin 23 when the cache control keychord is pressed. If the pin level is High, cache is enabled. If the pin level is Low, cache is disabled. Pins 23, 24, 27, or 28 cannot be used if PS/2-type Mouse Support is present in the BIOS.

cont'd

Cache Control Options — Using Internal Registers

Some chipsets permit cache control to be implemented via registers internal to the chipset. If this method is chosen, the routine to toggle the cache is activated inside the BIOS. The appropriate keychord toggles the cache status. This method is enabled via AMIBCP.

Using I/O Ports

Cache control can also be implemented via designated I/O port addresses. To use this method, the port addresses, port data values, and port mask values must be specified for both cache enable and cache disable. The port value is read and ANDed with the mask value. The new value is ORed with the data value. The final value is written back to the port.

Example

The port address for cache enabled is 1234h. To disable cache, bit 7 of the value read should be 0.

When the cache disable keychord (usually Ctrl Alt Shift -) is pressed, the BIOS reads the port value as FFh. The mask value is ANDed with the port value (7Fh). The new value is 7Fh. This makes bit 7 equal to 0. Now the BIOS ORs the contents of the I/O port and the data value (00h). The final value (7Fh) is written to port 1234h. The cache status hardware knows that bit 7 set to 0 means cache disabled.

Cache Memory Enable Hardware Must be Present

The cache memory enable decoding circuitry and functional toggling circuitry must be present on the motherboard. The BIOS reads and writes to the specified ports when the cache control keychord is pressed. The cache memory status must be implemented in the motherboard design. Both cache enable and disable I/O ports should allow read and write operations and are limited to a four-digit hex address.

Turbo Switch 8042 Pin Options

Systems using the Hi-Flex AMIBIOS support a turbo switch that allows you to toggle the system speed. If the system has an American Megatrends Keyboard Controller BIOS (version F or later), the system speed can be toggled via an 8042 pin. If the CPU is in protected mode, (most probably because QEMM, Microsoft Windows, DesqView, or EMM386 is running), the turbo switch operation is automatically disabled to prevent conflicts.

The Turbo Switch 8042 Pin options are:

- No Turbo Switch Input Pin through Keyboard Controller
- Pin 27 High means High Speed
- Pin 27 Low means High Speed
- Pin 28 High means High Speed
- Pin 28 Low means High Speed
- Pin 29 High means High Speed
- Pin 29 Low means High Speed
- Pin 30 High means High Speed
- Pin 30 Low means High Speed
- Pin 31 High means High Speed
- Pin 31 Low means High Speed
- Pin 33 High means High Speed
- Pin 33 Low means High Speed

If *No Turbo Switch Input Pin through Keyboard Controller* is chosen, the BIOS notes which keyboard controller pin is chosen as the turbo pin and continually monitors this pin. When the pin status changes, the BIOS switches the processor speed.

For example, if *Pin 28 High means High Speed* is chosen, the BIOS monitors the voltage level of 8042 pin 28 constantly. When the turbo switch is pressed, it changes the processor speed. If the pin level is High, the speed is set High. If the pin level is Low, the speed is set to Low.

Note: Pins 27 and 28 cannot be used if PS/2-type Mouse Support is enabled. Of course, the same 8042 pin cannot be used for speed switching or cache control.

cont'd

Memory Controller Reset Options

The Intel 82335SX chipset requires that the memory controller be reset every time the system is rebooted. Memory controller reset can be implemented via the AMIBIOS using an 8042 pin.

This option can be used if the system has an American Megatrends Keyboard Controller BIOS (version F or later). The three options are:

- No Reset Available through Keyboard Controller
- Reset through Pin 23
- Reset through Pin 24

If *No Reset Available through Keyboard Controller* is chosen, the only way to reset the memory controller chip is to physically power down the system.

If an 8042 pin option is chosen, the BIOS toggles the voltage level of the 8042 pin and the memory controller is reset, permitting continued operation.

For example, if *Reset through Pin 23* is chosen, the BIOS toggles the voltage level of pin 23 of the 8042.

Note: An 8042 pin cannot be used for memory controller reset if PS/2-type mouse support is present in the BIOS. Also, the same 8042 pin cannot be used for memory controller reset and speed switching or cache control.

Chapter 3

System Memory Map

The following table shows the use of the first megabyte of memory.

Address Range	Length	Description
000000 – 0002FFh	768 bytes	BIOS Interrupt Vector Table
000300 – 0003FFh	256 bytes	BIOS Stack Area
000400 – 0004FFh	256 bytes	BIOS Data Area
000500 – 09FFFFh	640 KB	Applications Memory, used by the operating system, device drivers, TSRs, and all DOS applications.
0A0000 – 0BFFFFh	128 KB	Video Buffer (EGA and VGA).
0B0000 – 0B7FFFh	32 KB	Video Buffer (for Monochrome, CGA color, and VGA monochrome).
0B8000 – 0BFFFFh	32 KB	Video Buffer (for CGA, EGA color, and VGA color).
0C0000 – 0C7FFFh	32 KB	Video ROM (EGA and VGA)
0C8000 – 0CFFFFh	32 KB	Unused. Reserved for Adaptor ROMs (other devices requiring ROMs).
0D0000 – 0DFFFFh	64 KB	Used by Adaptor ROMs, such as Network Controllers, Hard Disk Controllers, and SCSI Host Adapters.
0E0000 – 0EFFFFh	64 KB	Used by System ROM, which can include Network Controllers with boot-up capabilities, and other devices. If the system BIOS is 128 KB in length (an EISA BIOS), the first 64 KB of ROM is here.
0F0000 – 0FFFFFh	64 KB	System AMIBIOS, which includes the BIOS Setup utility and hard disk utilities.

Chapter 4

BIOS Data Area

When an ISA or EISA system is powered-on, the BIOS Data Area is created at location 000400h. It is 256 bytes long (000400 – 0004FFh) and contains information about the system environment. This information can be accessed and changed by any program. Much of the operation of ISA and EISA systems is controlled by this data. This data is loaded by POST (the BIOS Power-On Self Test) during startup. The following table lists the contents of all BIOS data area locations. All addresses are offsets from 000400h.

Offset	BIOS Service	Description
00h	INT 14h	Serial Port (COM) 1 — least significant byte.
01h	INT 14h	Serial Port (COM) 1 — most significant byte.
02h	INT 14h	Serial Port (COM) 2 — least significant byte.
03h	INT 14h	Serial Port (COM) 2 — most significant byte.
04h	INT 14h	Serial Port (COM) 3 — least significant byte.
05h	INT 14h	Serial Port (COM) 3 — most significant byte.
06h	INT 14h	Serial Port (COM) 4 — least significant byte.
07h	INT 14h	Serial Port (COM) 4 — most significant byte.
08h	INT 17h	Parallel Port (LPT) 1 — least significant byte
09h	INT 17h	Parallel Port (LPT) 1 — most significant byte
0Ah	INT 17h	Parallel Port (LPT) 2 — least significant byte
0Bh	INT 17h	Parallel Port (LPT) 2 — most significant byte
0Ch	INT 17h	Parallel Port (LPT) 3 — least significant byte
0Dh	INT 17h	Parallel Port (LPT) 3 — most significant byte
0Eh	POST	Extended BIOS Data Area Segment — least significant byte
0Fh	POST	Extended BIOS Data Area Segment — most significant byte

Offset	BIOS Service	Description
10h–11h	INT 11h	Equipment List Bits 15–14 Number of parallel adapters 00b None installed 01b One installed 10b Two installed 11b Three installed Bits 13–12 Reserved Bits 11–9 Number of serial adapters installed 000b None installed 001b One installed 010b Two installed 011b Three installed 100b Four installed Bit 8 Reserved Bits 7–6 Number of floppy disk drives. 00b One drive 01b Two drives Bits 5–4 Initial video mode 00b EGA or PGA 01b 40 x 25 color 10b 80 x 25 color 11b 80x25 Monochrome Bit 3 Reserved Bit 2 PS/2-type pointing device present if set. Bit 1 Math coprocessor present if set. Bit 0 Floppy disk drive A: present if set.
12h	POST	Interrupt Flag used in POST.
13h	INT 12h	Memory size in KB — least significant byte.
14h	INT 12h	Memory size in KB — most significant byte.
15h–16h		Reserved
17h	INT 16h	Keyboard Status Byte Bit 7 System in Insert Mode if set. Bit 6 <Caps Lock> Key on if set. Bit 5 <Num Lock> key on if set. Bit 4 <Scroll Lock> Key on if set. Bit 3 <Alt> key pressed if set. Bit 2 <Ctrl> key pressed if set. Bit 1 Left <Shift> Key pressed if set. Bit 0 Right <Shift> Key pressed if set.
18h	INT 16h	Extended Keyboard Status Byte Bit 7 <Ins> key pressed if set. Bit 6 <Caps Lock> key pressed if set. Bit 5 <Num Lock> key pressed if set. Bit 4 <Scroll Lock> pressed if set. Bit 3 <Ctrl><Num Lock> state active Bit 2 <SysReq> key pressed if set. Bit 1 Left <Alt> key pressed if set. Bit 0 Left <Ctrl> key pressed if set.
19h		Reserved
1Ah–1Bh	INT 16h	Pointer to the address of the next character in the keyboard buffer.
1Ch–1Dh	INT 16h	Pointer to the address of the last character in the keyboard buffer.
1Eh–3Dh	INT 16h	Keyboard buffer (32 bytes). If the address in 1Ah is the same as the address in 1Ch, the buffer is empty. If the address in 1Ch is two bytes from the address in 1Ah, the buffer is full.

Offset	BIOS Service	Description
3Eh	INT 13h	Floppy Disk Drive Calibration Status Bits 7–4 Reserved. Should be 00h. Bits 3–2 Reserved Bit 1 Floppy Drive B: needs recalibration if 0. Bit 0 Floppy Drive A: needs recalibration if 0.
3Fh	INT 13h	Floppy Disk Drive Motor Status Bit 7 0 Current operation is Write or Format. 1 Current operation is Read or Verify. Bit 6 Reserved Bits 5–4 Drive select 00b Drive A: select 01b Drive B: select Bits 3–2 Reserved Bit 1 Drive A: motor is on if set. Bit 0 Drive B: motor is on if set.
40h	INT 13h	Floppy Disk Drive Motor Timeout This value is decremented by one 18.2 times per second (via the INT 08h timer interrupt). When the value becomes zero, the drive motor is powered off. The value refers to the last disk drive accessed.
41h	INT 13h	Floppy Disk Drive Status. These values are valid for the last floppy disk drive accessed. Bit 7 Drive not ready if set. Bit 6 Seek error detected if set. Bit 5 Floppy disk controller failed if set. Bits 4-0 Error Codes 00h No error occurred 01h Illegal function requested 02h Address mark not found 03h Write protect error 04h Sector not found 06h Drive door was opened 08h DMA overrun error 09h DMA boundary error 0Ch Unknown media type 10h CRC failed on floppy read 20h Controller failure 40h Seek failed 80h Timeout
42h–48h	INT 13h	Floppy disk controller status bytes and command bytes for the hard disk controller.
49h	INT 10h	Current Video Display Mode setting.
4Ah–4Bh	INT 10h	Number of text columns per line of current video mode.
4Ch–4Dh	INT 10h	Current page size in bytes.
4Eh–4Fh	INT 10h	Offset address of current display page, relative to the start of video RAM. Video RAM starts at B800h in CGA and B000h in MDA.
50h–5Fh	INT 10h	Current cursor position for each video page. Up to eight display pages are possible. Two bytes per page store the current cursor position for each page. The MSB specifies the row (line) value and the LSB specifies the column value of the cursor. Change the cursor position using INT 10h functions. *Do not change the values in this location.*
60h	INT 10h	Starting line of the cursor.
61h	INT 10h	Ending line of the cursor.
62h	INT 10h	Current video display page number.

BIOS Data Area, Continued

Offset	BIOS Service	Description
63h–64h	INT 10h	I/O port address of the video display adapter (the CRT Controller address register). It is 3B4h if a monochrome adapter is used and 3D4h if color is used.
65h	INT 10h	Video display adapter mode register. The mode register is I/O port 3B8h if a monochrome adapter is used, 3D8h if a CGA adapter is used, or 3D9h if EGA or VGA is used.
66h	INT 10h	Current palette color.
67h–6Bh		Adaptor ROM address.
6Ch–6Fh	INT 1Ah	Counter for INT 1Ah. Incremented by one every time the INT 08h timer interrupt occurs (18.2 times per second). When 24 hours has passed, this counter is reset to 0.
70h	INT 1Ah	Timer 24-hour flag. 0 if the timer is between 0 and 24 hours. Set to 1 when the time crosses 24 hours. Must be manually reset.
71h	INT 16h	Break Key pressed flag. Bit 7 Set if <Ctrl><Break> or <Ctrl><C> is pressed.
72h–73h	POST	Soft reset flag. If 1234h, memory test is skipped on reboot.
74h–77h	INT 13h	Status of last hard disk drive operation. 00h No error 01h Invalid function request 02h Address mark not found 04h Sector not found 05h Reset failed 07h Drive parameter activity failed 08h DMA overrun on operation 09h Data boundary error 0Ah Bad sector flag selected 0Bh Bad track detected 0Dh Invalid number of sectors on format 0Eh Control data address mark detected 0Fh DMA arbitration level out of range 10h Uncorrectable ECC or CRC error 11h ECC corrected data error 20h General controller failure 40h Seek operation failed 80h Timeout AAh Drive not ready BBh Undefined error occurred CCh Write fault on selected drive E0h Status error, or error register is 0 FFh Sense operation failed
75h	13h	Number of hard disk drives
76h–77h	13h	Hard disk drive work area
78h	INT 17h	Parallel port 1 timeout counter
79h	INT 17h	Parallel port 2 timeout counter
7Ah	INT 17h	Parallel port 3 timeout counter
7Bh		Reserved
7Ch	INT 14h	Serial port 1 timeout counter
7Dh	INT 14h	Serial port 2 timeout counter
7Eh	INT 14h	Serial port 3 timeout counter
7Fh	INT 14h	Serial port 4 timeout counter
80h–81h	INT 16h	Starting address of the keyboard buffer (usually 01Eh).
82h–83h	INT 16h	Ending address of the keyboard buffer (usually 03Eh).

Offset	BIOS Service	Description
84h	INT 10h	Number of displayed character rows minus one.
85h-86h	INT 10h	Height of character matrix.
87h	INT 10h	Bit 7 Equal to bit 7 of the video mode number passed to INT 10h by the programmer. Bits 6–4 Video RAM 000b 64K 001b 128K 010b 192K 011b 256K 100b 512K 110b 1024K Bit 3 0 Video subsystem active 1 Video subsystem inactive Bit 2 Reserved Bit 1 0 Color monitor 1 Monochrome monitor Bit 0 0 Alphanumeric cursor emulation disabled 1 Alphanumeric cursor emulation enabled
88h	INT 13h	Data transmission speed of the hard disk drive.
89h	INT 10h	VGA Video Flags Bits 7 and 4 0 0 350-line mode 0 1 400-line mode 1 0 200-line mode 1 1 Reserved Bit 6 0 Display switch disabled 1 Display switch enabled Bit 5 Reserved Bit 3 1 Default palette loading enabled 0 Default palette loading disabled Bit 2 0 Color monitor 1 Monochrome monitor Bit 1 0 Gray scale summing disabled 1 Gray scale summing enabled Bit 0 0 VGA inactive 1 VGA active
8Ah–8Bh		Reserved
8Ch–95h	INT 13h	Hard disk and floppy disk drive variables.
96h	INT 16h	Extended Keyboard Status Bit 7 Read ID in progress if set. Bit 6 Last code was first ID if set. Bit 5 Forced Num Lock if set. Bit 4 101 and 102-key keyboard used if set Bit 3 Right <Alt> key active Bit 2 Right <Ctrl> key active. Bit 1 Last code was E0h. Bit 0 Last code was E1h.
97h	INT 16h	Extended Keyboard Status Bit 7 Keyboard error occurred if set. Bit 6 LED is being updated if set. Bit 5 Resend code received if set. Bit 4 Acknowledge code received if set. Bit 3 Reserved Bit 2 Caps Lock LED is on if set. Bit 1 Num Lock LED is on if set. Bit 0 Scroll Lock LED is on if set.

Offset	BIOS Service	Description
98h–99h		Segment part of user wait flag address
9Ah–9Bh		Offset part of user wait flag address
9Ch–9Fh		Wait count
A0h	INT 1Ah	Wait active flag Bit 7 Wait time has elapsed if set. Bits 6–1 Reserved Bit 0 INT 15h AH = 86h occurred if set.
A1h–A7h		Reserved
A8h–ABh	INT 10h	INT 10h pointer to EGA and VGA parameter control block
ACh–EFh		Reserved
F0h–FFh		Intra-Applications Communication Area. Stores data that can be used by different applications programs.

Chapter 5

BIOS Data

The system BIOS includes parameters for various peripheral devices that help it to initialize the system. This information is stored in arrays and tables in the BIOS. A complete map of the locations of these tables and of all device service routines is included in the ROM BIOS and is listed below.

In This Chapter

The following topics are discussed in this chapter:

- ROM Compatibility Table,
- Floppy Disk Drive Parameter Table,
- Hard Disk Drive Parameter Table,
- Hard Drive Types,
- Hard Drive Data Transfer Rates,
- Video Parameter Table,
- System Configuration Data Table, and
- Data Transmission Rate Initialization Table.

ROM Compatibility Table

Both the ISA and EISA system BIOS assure compatibility with older PC and XT standards by maintaining a list of vectors to the IBM-compatible vector.

Address	BIOS Service Table
FE05Bh	POST entry point
FE2C3h	NMI Handler entry point
FE3FEh	INT 13h Hard Disk Drive Service entry point
FE401h	Hard Disk Drive Parameter Table
FE6F2h	INT 19h Boot Load Service entry point
FE6F5h	Configuration Data Table
FE729h	Data Transmission Rate Generator Table
FE739h	INT 14h Serial Communications Service entry point
FE82Eh	INT 16h Keyboard Service entry point
FE987h	INT 09h Keyboard Service entry point
FEC59h	INT 13h Floppy Disk Service entry point
FEF57h	INT 0Eh Floppy Disk Hardware Interrupt Service Routine entry point
FEFC7h	Floppy Disk Controller Parameter Table
FEFD2h	INT 17h Parallel Printer Service entry point
FF045h	INT 10h Video Service Functions 00h through 0Fh entry point
FF065h	INT 10h Video Service entry point
FF0A4h	MDA and CGA Video Parameter Table (INT 1Dh)
FF841h	INT 12h Memory Size Service entry point
FF84Dh	INT 11h Equipment List Service entry point
FF859h	INT 15h Systems Services entry point
FFA6Eh	Low-order 128 characters of the 320 x 200 and 640 x 200 graphics fonts
FFE6Eh	INT 1Ah Time-of-Day Service entry point
FFEA5h	INT 08h System Timer Interrupt Service Routine entry point
FFEF3h	Initial Interrupt Vector offsets loaded by POST
FFF53h	IRET Instruction for Dummy Interrupt Handler
FFF54h	INT 05h Print Screen Service entry point
FFFF0h	Power-On entry point
FFFF5h	ROM Date (in ASCII). Eight characters in mm/dd/yy format.
FFFFEh	System Model ID (always FCh).

Floppy Disk Drive Parameters

The floppy diskette parameter table is pointed to by the INT 1Eh vector. The table is eleven bytes long.

Offset	Description		
00h	Bits 7–4	Head Unload Time in milliseconds. The amount of time needed to allow the drive head to settle after it is lifted from the drive surface.	
		0h 32 ms	
		1h 64 ms	
		2h 96 ms	
		3h 128 ms	The default for 2.88 MB drives is 120
		4h 160 ms	
		5h 192 ms	
		6h 240 ms	The default value for 1.2 MB floppy drives.
		7h 256 ms	
		8h 288 ms	
		9h 320 ms	
		Ah 352 ms	
		Bh 384 ms	
		Ch 399 ms	The default value for 360 KB floppies in a 1.2 MB floppy drive.
		Dh 448 ms	
		Eh 480 ms	The default value for 360 KB and 720 KB floppy drives.
		0Fh 512 ms	
	Bits 3–0	Step Rate in milliseconds. The amount of time needed for a drive head to move from one track to another.	
		00h 2	The default for 1.2 MB and 2.88 drives is 3.0 ms.
		01h 4	The default for 360 KB floppy in 1.2 MB drive is 4.8 ms.
		02h 6	The default value for 360 KB, 720 KB, and 1.44 MB floppy drives.
		03h 8	
		04h 10	
		05h 12	
		06h 14	
		07h 16	
		08h 18	
		09h 20	
		0Ah 22	
		0Bh 24	
		0Ch 26	
		0Dh 28	
		0Eh 30	
		0Fh 32	

Floppy Parameters, Continued

Offset	Description
01h	Head Load Time. The amount of time in milliseconds needed to allow the drive head to settle after it is lowered onto the drive surface. The value ranges from 00h – 7Fh in increments of 4 milliseconds. See the following table for default values. Bits 7–0 00h 4 ms 01h 8 ms 02h 12 ms 03h 16 ms 04h 20 ms 05h 24 ms 06h 28 ms 07h 32 ms 08h 36 ms 09h 40 ms ... ... 7Fh 512 ms Bit 0 Non-DMA Mode Flag (Always 0 to indicate that DMA is used).
02h	Motor Wait Timer. The amount of Time that a floppy drive can be inactive before the drive motor is shut off. This value ranges from 0 to 255 in increments of 1. The timer ticks approximately 18.2 per second. The Motor Wait Time value can be calculated as follows: TIME = Selected timer tick value divided by 18.2 Bits 7–0 00h 0 timer ticks 01h 1 timer tick 02h 2 timer ticks ... ... 37h 37 timer ticks (Default for all floppy drives — approximately 2.03 seconds) ... ... FFh 255 timer ticks
03h	Number of Bytes per Sector Bits 7–0 00h 128 bytes per sector 01h 256 bytes per sector 02h 512 " " " (Default for all floppy drives). 03h 1024 bytes per sector
04h	Number of Sectors Per Track Bits 7–0 08h 8 sectors per track (320 KB 5¼" drives) 09h 9 sectors per track (360 KB and 720 KB 5¼" drives) 0Fh 15 sectors per track (1.2 MB 5¼" drives) 12h 18 sectors per track (720K and 1.44 MB 3½" drives) 24h 36 Sectors per track (2. 88 MB 3½" drives)

Offset	Description
05h	Gap Length. The length of the gap between sectors. Bits 7–0 00h 0 01h 1 ... ... 1Bh 27 The default value for all 3½" drives and 1.2 MB 5¼" drives. ... ... 2Ah 42 The default for 360 KB and 720 KB floppy drives. ... ... FFh 255
06h	Data Length — always set to FFh.
07h	Gap length for format. This value is used for the same purpose as the gap length, but it is used in formatting only. Bits 7–0 00h 0 01h 1 ... ... 50h 80 The default value for 360 KB, 720 KB, and 2.88 MB floppy drives. 51h 84 The default value for 1.2 MB floppy drives. ... ... 57h 108 The default value for 1.44 MB floppy drives. ... ... FFh 255
08h	Fill Byte for Formatting — always set to F6h
09h	Head Settle Time. The amount of time in milliseconds that must elapse to allow the heads to settle after a Seek operation. Bits 7–0 00h 0 ms 01h 1 ms 02h 2 ms ... ... 0Fh 15 ms (the default value for all floppy drives) ... ... FFh 255 ms
0Ah	Motor Start Time. The amount of time it takes the drive motor to reach optimal speed. The values are in eighths of a second. Bits 7–0 00h 0 01h 1 ⅛th of a second 02h 2 ¼ second ... ... 08h 8 one second (the default value for all floppy drives) ... ... FFh 255 31⅞ of a second

Summary of Default Settings for Floppy Drives

The following table summarizes the default settings for all floppy disk parameter table values in the AMIBCP:

Parameter	360 KB Floppy in 360KB Drive	360 KB Floppy in 1.2 MB Drive	1.2 MB Floppy in 1.2 MB Drive	720 KB 3½"	1.44 MB 3½"	2.88 MB 3½"
Step Rate (ms)	6.0	4.8	3.0	6.0	6.0	3.0
Head Unload Time (ms)	480	399	240	480	240	120
Head Load Time (ms)	4.0	3.3	2.0	4.0	2.0	1.0
Motor Wait Time (in timer ticks)	37	37	37	37	37	37
Gap Length	42	42	27	42	27	27
Gap Length for Format	80	80	84	80	108	80
Head Settle Time (ms)	15	15	15	15	15	15
Motor Start Time (in ⅛ths of a second)	8	8	8	8	8	8
Number of Bytes per Sector	512	512	512	512	512	512
Cluster Size	1024	1024	512	1024	512	512
Tracks	40	40	80	80	80	80
Sectors per Track	9	9	15	9	18	36

Hard Disk Parameter Table

The hard disk drive parameter table (drive type table) is located at F000:E401h. The vector table entries for INT 41h contains the entry points for the hard disk drive types selected via BIOS Setup for hard disk drive C:. INT 46h contains the vector for hard disk drive D:. Each drive type entry consists of 16 bytes, in the following format:

Offset	Description
00h–01h	Number of Cylinders. Byte 01h is the most significant byte.
02h	Number of heads.
03h–04h	Reserved
05h–06h	Starting write precompensation cylinder. Byte 06h is the most significant byte. The size of a sector gets progressively smaller as the track diameter diminishes. Yet each sector must still hold 512 bytes. Write precompensation circuitry on the hard disk compensates for the physical difference in sector size by boosting the write current for sectors on inner tracks. This parameter is the track number where write precompensation begins.
07h	Reserved
08h	Control Byte Bits 7–6 Enable or Disable Retries 00h Enable retries. All other values disable retries. Bit 5 Set if defect map is located at last cylinder plus one. Bit 4 Reserved. Always set to 0. Bit 3 Set if more than 8 heads. Bits 2–0 Reserved. Always set to 0.
09h–0Bh	Reserved
0Ch–0Dh	Landing Zone. This number is the cylinder location where the heads normally park when the system is shut down.
0Eh	Number of Sectors per Track. Hard disk drives that use MFM have 17 sectors per track. RLL drives have 26 sectors per track. RLL and ESDI drives have 34 sectors per track. IDE and SCSI drives may have even more sectors per track.
0Fh	Reserved

cont'd

Hard Disk Parameter Table, Continued

The AMIBIOS uses a standard hard disk drive type table that has 45 entries for drive types 0 – 14 and 16 – 46. See page 48 for a complete list of the hard disk drive parameters.

These drive types can be used to configure DOS drives C: and D:.

This table can also be modified using AMIBCP. See the *AMIBCP User's Guide* for more information.

User-Definable Drives

The Hi-Flex AMIBIOS also supports two user-definable drive types (Type 47 C: and Type 47 D:) to be used with hard disk drives not defined in the standard drive table. You can configure these drive types via AMIBIOS Setup.

Location of Hard Drive Parameters

The hard disk drive parameters are stored in CMOS RAM registers 1Bh – 23h (drive C:) and 24h – 2Ch (drive D:). The format is shown on page 45. The BIOS rewrites these parameters at system boot to a different location to permit quicker access.

Hard Disk Drive Type Selection

The BIOS first makes sure that shadow RAM is enabled. If so, the BIOS copies these parameters to the locations in the drive table specified by the INT 41h (Drive C:) and INT 46h (Drive D:) vectors.

If shadow RAM is disabled or the system does not support shadow RAM, the parameters are copied to either of two secondary locations:

- the BIOS Stack Area (000300h – 000301h), or
- the upper 1 KB of DOS memory (09FFFEh – 09FFFFh).

The OEM selects the secondary location that is to be used through AMIBCP, or can allow you to choose the secondary location via AMIBIOS Setup.

Hard Disk Drive Capacity

The capacity of a hard disk drive can be determined using the following formula:

(Number of heads) X (Number of cylinders) X (Number of sectors per track) X (512 - Number of bytes per sector)

Hard Disk Drive Types

The following values are the default hard drive parameter values. The OEM can modify these values using AMIBCP.

The Control Byte, the only hard disk drive parameter not shown in the following table, is described on page 45. The Control Byte is almost always 00h in the AMIBIOS Hard Disk Parameter Table.

The only exceptions are hard disk drive types 9, 25, 27, 29, 32, 35, 44, 45, and 46, where it is 80h.

Type	Number of Cylinders	Number of Heads	Write Precompensation	Landing Zone	Number of Sectors	Size
1	306	4	128	305	17	10 MB
2	615	4	300	615	17	20 MB
3	615	6	300	615	17	31 MB
4	940	8	512	940	17	62 MB
5	940	6	512	940	17	47 MB
6	615	4	65535	615	17	20 MB
7	462	8	256	511	17	31 MB
8	733	5	65535	733	17	30 MB
9 *	900	15	65535	901	17	112 MB
10	820	3	65535	820	17	20 MB
11	855	5	65535	855	17	35 MB
12	855	7	65535	855	17	50 MB
13	306	8	128	319	17	20 MB
14	733	7	65535	733	17	43 MB
16	612	4	0	663	17	20 MB
17	977	5	300	977	17	41 MB
18	977	7	65535	977	17	57 MB
19	1024	7	512	1023	17	60 MB
20	733	5	300	732	17	30 MB
21	733	7	300	732	17	43 MB
22	733	5	300	733	17	30 MB
23	306	4	0	336	17	10 MB
24	925	7	0	925	17	54 MB
25 *	925	9	65535	925	17	69 MB
26	754	7	754	754	17	44 MB
27 *	754	11	65535	754	17	69 MB
28	699	7	256	699	17	41 MB
29 *	823	10	65535	823	17	68 MB
30	918	7	918	918	17	53 MB
31	1024	11	65535	1024	17	94 MB
32 *	1024	15	65535	1024	17	128 MB
33	1024	5	1024	1024	17	43 MB

Hard Disk Drive Types, Continued

Type	Number of Cylinders	Number of Heads	Write Precompensation	Landing Zone	Number of Sectors	Size
34	612	2	128	612	17	10 MB
35 *	1024	9	65535	1024	17	77 MB
36	1024	8	512	1024	17	68 MB
37	615	8	128	615	17	41 MB
38	987	3	987	987	17	25 MB
39	987	7	987	987	17	57 MB
40	820	6	820	820	17	41 MB
41	977	5	977	977	17	41 MB
42	981	5	981	981	17	41 MB
43	830	7	512	830	17	48 MB
44 *	830	10	65535	830	17	69 MB
45 *	917	15	65535	918	17	114 MB
46 *	1224	15	65535	1223	17	152 MB
47	Enter hard disk drive parameters supplied by disk drive manufacturer.					

* Control Byte is 80h

Hard Disk Drive Data Transfer Rates

The following table lists the data transfer rates for some common types of hard disk drives.

Drive Interface Type	Bit Rate	Byte Rate
ST506 and ST412	5 Mbs	0.625 MBs
RLL	7.5 Mbs	0.9375 MBs
IDE	7.5 Mbs	0.9375 MBs
ESDI	10 Mbs	1.25 MBs
SCSI-2	80 – 320 Mbs	10 – 40 MBs

Video Parameter Table

This table always contains one or more entries for each available video mode, including video modes specified in the MDA, CGA, EGA, PGA, XGA, or VGA standards. If VGA is used in the system, this table contains at least 29 entries in the following format:

Offset	Description
00h	Number of displayed character columns (the same value as in 40:49h).
01h	Number of displayed character rows - 1 (the same value as in 40:84h).
02h	Height of character matrix (the same value as in 40:85h).
03h	Size of video buffer in bytes (the same value as in 40:4Ch).
05h	The value for Sequencer Registers 1 through 4.
09h	The value for the Miscellaneous Output Register.
0Ah	Values for CRTC Registers 00h through 18h.
23h	The values for Attribute Control Registers 00h through 13h.
37h	The values for Graphics Controller Registers 0 through 8.

System Configuration Data

The System Configuration Table is located at F000:E6F5h. This table can be moved to system memory by invoking INT 15h Function C0h Return System Configuration Parameters.

Offset	Description
00h	Number of bytes in this table. It must be at least eight bytes.
02h	Model Byte (always FCh)
03h	Submodel Byte (always 01h)
04h	BIOS Revision Level. Should be zeros if the first release of the BIOS.
05h	Feature Information Byte 1 Bit 7 If set, the hard disk drive BIOS is using DMA Channel 3. Bit 6 If set, a second interrupt controller chip is present in the system. Bit 5 If set, a Real Time Clock is present. Bit 4 If set, a Keyboard Intercept (INT 15h Function 4Fh) has been called by the keyboard interrupt service (INT 09h). Bits 3–0 Reserved, should be zeros.
06h	Reserved, should be zeros.
07h	Reserved, should be zeros.
08h	Reserved, should be zeros.
09h	Reserved, should be zeros.

Data Transmission Rate Initialization Table

The data transmission rate initialization table is located at F000:E729h in the ROM BIOS.

Data Transmission Rate	Divisor
110	0417h
150	0300h
300	0180h
600	00C0h
1200	0060h
2400	0030h
4800	0018h
9600	000Ch
19200	0006h

Data Transmission Rate Divisors

The input frequency to the device is 1.8432 MHz. The values in the table are calculated as follows:

```
1,843,200 divided by 16 = 115,200 divided by data
transmission rate = Divisor
```

For example, a data transmission rate of 2400 has a divisor of 115,200 divided by 2,400, which equals 30h.

Chapter 6

CMOS RAM

Systems that adhere to ISA standards have at least 64 bytes of CMOS RAM to store system initialization and configuration parameters.

How CMOS RAM is Configured

Most of these parameters are set by the system manufacturer and the user via the BIOS Setup utility. The AMIBIOS provides the AMIBIOS Setup utility in the BIOS ROM that can be accessed during startup.

Accessing CMOS RAM Directly

You can access CMOS RAM via an assembly language program. To read CMOS RAM, use the following Intel x86 assembler instructions:

```
OUT    70h,Register Number
IN     71h
```

To write to CMOS RAM, use the following instructions:

```
OUT    70h,Register Number
OUT    71h,New_Value
```

If the most significant bit of the *Register Number* is set when reading or writing CMOS RAM, the Nonmaskable Interrupt (NMI) is disabled during the operation.

How CMOS RAM is Organized

CMOS RAM is divided into several parts:

Location	Length	Description
00h – 0Fh	16 bytes	Real Time Clock data.
10h – 2Fh	32 bytes	ISA configuration data.
30h – 3Fh	16 bytes	AMIBIOS-specific configuration data.
40h – 7Fh	64 bytes	Extended CMOS RAM. Available in many systems. Many chipsets incorporate this additional CMOS RAM to store advanced configuration information.

EISA CMOS RAM

EISA Extended CMOS RAM stores EISA-specific information and is configured by the EISA Configuration Utility (ECU). See page 345 for additional information about EISA.

EISA Extended CMOS RAM consists of between 4,096 and 8,192 bytes of CMOS memory and is accessed via INT 15h Function D8h.

EISA Extended CMOS RAM can also be accessed via a series of I/O ports (see page 83).

CMOS RAM Map

A map of CMOS RAM as configured by the AMIBIOS is shown in the following table. This section assumes that a Motorola MC146818 or compatible Real Time Clock is used. For some registers, only the definitions used during initialization are shown.

Offset	Description
00h	Real Time Clock – Seconds field Contains the seconds value of the current time.
01h	Real Time Clock – Seconds Alarm Contains the seconds value for the RTC alarm.
02h	Real Time Clock – Minutes field Contains the minutes value of the current time.
03h	Real Time Clock – Minutes Alarm Contains the minutes value for the RTC alarm.
04h	Real Time Clock – Hours Contains the hours value of the current time.
05h	Real Time Clock – Hours Alarm Contains the hour value for the RTC alarm.
06h	Real Time Clock – Day of Week Contains the current day of the week.
07h	Real Time Clock – Date Contains the day field (0 – 31) of the current date.
08h	Real Time Clock – Month Contains the month field of the current date.
09h	Real Time Clock – Year Contains the year field of the current date.
0Ah	Status Register A Bit 7 Update in progress if set. 0 Can read date or time. 1 Can't read date or time because an update is in progress. Bits 6–4 Divider that identifies the time-based frequency to use. The BIOS initializes this field to 010b, a 32.768 KHz time base. Bits 3–0 Rate selection bits that define output frequency and periodic interrupt rate. The BIOS initializes these bits to 0110b, which sets a 1.024 KHz square wave clock pulse and a 976.562 second interrupt rate.

Offset	Description
0Bh	Status Register B Bit 7 Halt Cycle to Set Clock 0 Update counter once per second. 1 Halt the counter to set the clock. Bit 6 Periodic Interrupt 0 Disable 1 Enable Bit 5 Alarm Interrupt 0 Disable 1 Enable Bit 4 Update-Ended Interrupt 0 Disable 1 Enable Bit 3 Square Wave 0 Disable square wave. 1 Use the square wave rate set in Status Register A. Bit 2 Date and Time Mode 0 Use BCD format. 1 Use binary format. Bit 1 24 or 12 Hour Mode 0 Set 12 hour mode. 1 Set 24 hour mode. Bit 0 Daylight Savings Time 0 Disable 1 Enable
0Ch	Status Register C Bit 7 IRQ Flag (read-only) Bit 6 Periodic Interrupt Flag (read-only) Bit 5 Alarm Interrupt Flag (read-only) Bit 4 Update Interrupt Flag (read only) Bits 3–0 Reserved (should be set to 0).
0Dh	Status Register D Bit 7 Valid CMOS RAM 0 CMOS battery low, CMOS RAM invalid. 1 CMOS RAM battery good, CMOS RAM valid. Bits 6–0 Reserved (should be set to 0).

Offset	Description
0Eh	Diagnostic Status Bit 7 RTC Chip Power 0 Power valid. 1 Power invalid. Bit 6 CMOS RAM Checksum error 0 CMOS RAM checksum valid. 1 CMOS RAM checksum invalid. Bit 5 CMOS RAM Configuration Mismatch 0 CMOS RAM configuration matches the newly determined configuration. 1 CMOS RAM configuration does not match newly determined configuration. Bit 4 CMOS RAM Memory Size Mismatch 0 CMOS RAM memory size matches newly determined memory size. 1 CMOS RAM memory size does not match newly determined memory size. Bit 3 Hard disk drive C: initialization 0 Initialization passed, attempting to boot. 1 Failed initialization. No boot attempt. Bit 2 Time status indicator 0 Time is valid. 1 Time is not valid. Bits 1–0 Reserved. Should be 0.

Offset	Description
0Fh	Shutdown Status. The contents of the CPU registers are saved to memory and the CPU is reset when the CPU is switched from protected to real mode. If a program requests a shutdown (using a DWORD JMP instruction), the segment address of the program is stored in 40:67h and the offset address at 40:69h. Just before reset, a shutdown code is written to CMOS RAM 0Fh so that after reset, the BIOS knows the reason for the shutdown. **Code** **Explanation** 00h Normal POST execution. 01h Chipset initialization for real mode reentry. 02h–03h Used internally by the BIOS. 04h Jump to bootstrap code. 05h User-defined shutdown. Issue an EOI, flush the keyboard buffer, and jump to the doubleword pointer at 40:67h. The interrupt controller and math coprocessor are initialized. 06h Jump to the doubleword pointer at 40:67h with no EOI. 07h Return to INT 15h Function 87h. 08h Return to POST memory test. 09h INT 15h Function 87h Block Move shutdown request. 0Ah User-defined shutdown requested. The interrupt controller and math coprocessor are not initialized. The BIOS returns via a jump to the DWORD pointer at 40:67h with no EOI.
10h	Floppy Drive Type Bits 7–4 Drive A: Type 0h No Drive 1h 360 KB Drive 2h 1.2 MB Drive 3h 720 KB Drive 4h 1.44 MB Drive 5h 2.88 MB Drive Bits 3–0 Drive B: Type 0h No Drive 1h 360 KB Drive 2h 1.2 MB Drive 3h 720 KB Drive 4h 1.44 MB Drive 5h 2.88 MB Drive

Offset	Description
11h	Bit 7 Mouse Support Option 0 Disable 1 Enable Bit 6 Above 1 MB Memory Test 0 Disable 1 Enable Bit 5 Memory Test Tick Sound 0 Disable 1 Enable Bit 4 Memory Parity Error Check 0 Disable 1 Enable Bit 3 Hit <DEL> Message Display 0 Disable 1 Enable Bit 2 Hard Disk Type 47 RAM Area 0 Store at 0:300h 1 Store in Upper 1K of DOS area Bit 1 Wait for <F1> if Any Error 0 Disable 1 Enable Bit 0 System Boot Up Num Lock 0 Off 1 On
12h	Hard Disk Data Bits 7–4 Hard Disk Drive C: Type 0000b No drive installed 0001b Hard drive Type 1 0010b Hard drive type 2 ... . 1110b Drive Type 14 1111b Hard Disk Type 16 – 46 (actual Hard Drive Type is in CMOS RAM 19h) Bits 3–0 Hard Disk Drive D: Type 0000b No drive installed 0001b Hard drive Type 1 0010b Hard drive type 2 ... ... 1110b Drive Type 14 1111b Hard Disk Type 16 – 46 (actual Hard Drive Type is in CMOS RAM 1Ah)

Offset	Description
13h	Bit 7 Typematic Rate Programming 0 Disabled 1 Enabled Bits 6–5 Typematic Rate Delay (in milliseconds) 00b 250 ms 01b 500 ms 10b 750 ms 11b 1000 ms Bits 4–2 Typematic Rate (in characters per second) 000b 6 cps 100b 15 cps 001b 8 cps 101b 20 cps 010b 10 cps 110b 24 cps 011b 12 cps 111b 30 cps
14h	Equipment Byte Bits 7–6 Number of Floppy Drives 00b No Drive 01b One Drive 10b Two drives Bits 5–4 Monitor Type 00b Not CGA or MDA 01b 40x25 CGA 10b 80x25 CGA 11b MDA (Monochrome) Bit 3 Display Enabled 0 Not installed 1 Installed Bit 2 Keyboard Enabled 0 Not installed 1 Installed Bit 1 Math coprocessor Installed 0 Absent 1 Present Bit 0 Floppy Drive Installed. Always set to 1.
15h	Base Memory (in 1K increments), least significant byte.
16h	Base Memory (in 1 K increments), most significant byte.
17h	Extended Memory (in 1K increments), least significant byte.
18h	Extended Memory (in 1 K increments), most significant byte.
19h	Hard Disk C: Drive Type if Bits 7–4 of 12h are 1111b. 00h–0Fh Reserved 10h–2Eh Hard Drive Type 16 – 46
1Ah	Hard Disk D: Drive Type if Bits 7–4 of 12h are 1111b. 00h–0Fh Reserved 10h–2Eh Hard Drive Type 16 – 46
1Bh	User-Defined Drive C: Number of Cylinders, least significant byte.
1Ch	User-Defined Drive C: Number of Cylinders, most significant byte.
1Dh	User-Defined Drive C: Number of Heads.

Offset	Description
1Eh	User-Defined Drive C: Write Precompensation Cylinder, least significant byte.
1Fh	User-Defined Drive C: Write Precompensation Cylinder, most significant byte.
20h	User-Defined Drive C: Control Byte (80h if the number of heads is equal or greater than eight).
21h	User-Defined Drive C: Landing Zone, least significant byte
22h	User-Defined Drive C: Landing Zone, most significant byte
23h	User-Defined Drive C: Number of Sectors
24h	User-Defined Drive D: Number of Cylinders, least significant byte
25h	User-Defined Drive D: Number of Cylinders, most significant byte
26h	User-Defined Drive D: Number of Heads
27h	User-Defined Drive D: Write Precompensation Cylinder, least significant byte
28h	User-Defined Drive D: Write Precompensation Cylinder, most significant byte
29h	User-Defined Drive D: Control Byte (80h if the number of heads is equal or greater than 8)
2Ah	User-Defined Drive D: Landing Zone, least significant byte
2Bh	User-Defined Drive D: Landing Zone, most significant byte
2Ch	User-Defined Drive D: Number of Sectors

Offset	Description
2Dh	Bit 7 Weitek Processor 0 Absent 1 Present Bit 6 Floppy Drive Seek At Boot 0 Disable 1 Enable Bit 5 System Boot Up Sequence 0 C:, A: 1 A:, C: Bit 4 System Boot Up CPU Speed 0 High 1 Low Bit 3 External Cache Memory. Set to 0 if no external cache available in the system. 0 Disable 1 Enable Bit 2 Internal Cache Memory. Set to 0 if no internal cache memory in system. 0 Disable 1 Enable Bit 1 Fast Gate A20 Option. Set to 0 if system does not have Fast Gate A20. 0 Disable 1 Enable Bit 0 Turbo Switch Function 0 Disable 1 Enable
2Eh	Standard CMOS checksum, most significant byte.
2Fh	Standard CMOS checksum, least significant byte.
30h	Extended memory found by BIOS, least significant byte.
31h	Extended memory found by BIOS, most significant byte.
32h	Century byte. The BCD value for the century of the current date.
33h	Information Flag Bit 7 BIOS Length 0 64 KB 1 128 KB Bits 6–1 Reserved. Should be 0. Used as scratchpad for chipset-specific functions during POST. Bit 0 POST Cache test 0 Cache bad 1 Cache good

Offset	Description	
34h	Bit 7	Boot Sector Virus Protection (Only if 06/06/92 core AMIBIOS used)
		.0 Disabled
		1 Enabled
	Bit 6	Password Checking Option (Bit 7 also used if core BIOS is earlier than 12/12/91)
		0 Disabled
		0 Password checking only when entering BIOS Setup.
		11b Password checking at every boot.
	Bit 5	Adaptor ROM Shadow C800h,16K
		0 Disabled
		1 Enabled
	Bit 4	Adaptor ROM Shadow CC00,16K
		0 Disabled
		1 Enabled
	Bit 3	Adaptor ROM Shadow D000,16K
		0 Disabled
		1 Enabled
	Bit 2	Adaptor ROM Shadow D400,16K
		0 Disabled
		1 Enabled
	Bit 1	Adaptor ROM Shadow D800,16K
		0 Disabled
		1 Enabled
	Bit 0	Adaptor ROM Shadow DC00,16K
		0 Disabled
		1 Enabled

Offset	Description
35h	Bit 7 Adaptor ROM Shadow E000,16K 0 Disabled 1 Enabled Bit 6 Adaptor ROM Shadow E400,16K 0 Disabled 1 Enabled Bit 5 Adaptor ROM Shadow E800,16K 0 Disabled 1 Enabled Bit 4 Adaptor ROM Shadow EC00,16K 0 Disabled 1 Enabled Bit 3 System ROM Shadow F000,64K 0 Disabled 1 Enabled Bit 2 Video ROM Shadow C000,16K 0 Disabled 1 Enabled Bit 1 Video ROM Shadow C400,16K 0 Disabled 1 Enabled Bit 0 Numeric Processor Test 0 Disabled 1 Enabled
36h	Chipset-specific information.
37h	Password Seed and Color Option Bits 7–4 Password Seed used in the password encryption algorithm. *Do not change.* Bits 3–0 AMIBIOS Setup screen color. See page 65 for the monochrome table. 07h White (light gray) on black. 70h Black on white (light gray). 17h White (light gray) on Blue. 20h Black on green. 30h Black on turquoise. 47h White (light gray) on red. 57h White (light gray) on magenta. 60h Black on brown (dark yellow). 70h Black on white (light gray).
38h–3Dh	Encrypted Password. These six bytes store the AMIBIOS Password. The Password is uses a proprietary encryption format. The encryption scheme will not be published.
3Eh	MSB of Extended CMOS Checksum (includes 34h – 3Dh).
3Fh	LSB of Extended CMOS Checksum (includes 34h – 3Dh).

AMIBIOS Setup Monochrome Table

The following table lists the monochrome values for AMIBIOS Setup screens.

Color Number	Third Window	First Window	Second Window	Main Window
0	07h	70h	70h	07h
1	70h	70h	70h	70h
2	07h	07h	07h	07h
3	70h	07h	07h	70h
4	70h	70h	07h	07h
5	07h	70h	70h	70h
6	07h	07h	07h	70h
7	70h	07h	07h	07h
8	70h	70h	70h	07h
9	07h	70h	07h	07h
A	07h	07h	70h	70h
B	70h	07h	70h	07h
C	07h	70h	07h	70h
D	07h	07h	70h	07h
E	70h	07h	70h	70h
F	70h	70h	07h	70h

AMIBIOS Setup Color Table

The following table lists the monochrome values for AMIBIOS Setup screens.

Color Number	Third Window	First Window	Second Window	Main Window
0	57h	60h	17h	20h
1	20h	47h	30h	57h
2	70h	30h	57h	60h
3	60h	17h	20h	70h
4	57h	30h	47h	20h
5	20h	17h	60h	57h
6	17h	60h	57h	30h
7	30h	47h	20h	17h
8	70h	20h	17h	60h
9	60h	57h	30h	70h
A	70h	70h	70h	70h
B	07h	07h	07h	07h
C	70h	07h	70h	07h
D	07h	70h	07h	70h
E	17h	20h	47h	30h
F	30h	57h	60h	17h

Chapter 7

I/O Port Addresses

The microprocessor communicates with and controls many parts of the system via the I/O ports. The I/O ports are doorways through which information passes as it travels from an I/O device (such as a keyboard or serial port) to the microprocessor and vice versa.

The ISA architecture includes a 64 KB I/O memory area that is used to access external devices. Intel 386, 486, and Pentium® architectures allow for 8-, 16-, or 32-bit I/O ports. The I/O ports from 0000h–00FFh address devices on the motherboard. Ports 0100h–02FFh address devices attached to the system via expansion slots. I/O Port addresses 01F0h – 01F8h are reserved for a hard disk controller.

Accessing I/O Ports

Components that Use the I/O Ports

Most of the support chips in an AT (Intel 8259 Programmable Interrupt Controller, Intel 8254 Programmable Interval Timer, Intel 8237 Programmable DMA Controller and equivalents) use the I/O port to communicate with other parts of the system.

How Ports Are Identified

Each port is identified by a 16-bit port number (from 0 – 65,535, or 00000h – FFFFFh).

cont'd

Ports Accessed by Hex Port Number

The microprocessor sends data or control information to a specific I/O port by specifying the port number. The I/O port responds by passing data or status information through the port to the microprocessor.

Just as it does when accessing memory, the microprocessor uses the data and address buses as paths for communication with the I/O ports.

To access a port, the microprocessor first sends a signal on the control bus. This signal notifies all I/O devices that the information on the bus is an I/O port address. Then it sends the I/O port address. The device that is assigned to that specific I/O port responds.

The I/O port number addresses a memory location that is part of the I/O device, but is not part of system memory. Special assembler I/O instructions are used to signal a port access and send information to and from I/O devices.

Programmer Access

A programmer can use the Intel assembly language instructions IN and OUT to write to and read from I/O port addresses. For example:

```
MOV    DX,03CCh     ;read video register address
IN     AL,DX        ;read byte from I/O port
OR     AL,10h       ;set bit 4 for RAMDAC control
MOV    DX,03C2h     ;write register address
OUT    DX,AL        ;write value to I/O port
```

ISA and EISA I/O Port Assignments

All ISA and EISA computers use standard I/O ports. The standard I/O port addresses for all AMIBIOS products are listed in the following tables. Some systems also use customized I/O port assignments.

Hardware I/O Port Addresses

The I/O port addresses from 0000h–00FFh are used by components mounted on the motherboard.

I/O Port	Read/ Write	Description
0000h – 001Fh are used by the 8237 DMA Controller 1		
0000h	R/W	DMA channel 0 address byte 0 (low byte), followed by byte 1.
0001h	R/W	DMA channel 0 word count byte 0 (low byte), followed by byte 1.
0002h	R/W	DMA channel 1 address byte 0 (low byte), followed by byte 1.
0003h	R/W	DMA channel 1 word count byte 0 (low byte), followed by byte 1.
0004h	R/W	DMA channel 2 address byte 0 (low byte), followed by byte 1.
0005h	R/W	DMA channel 2 word count byte 0 (low byte), followed by byte 1.
0006h	R/W	DMA channel 3 address byte 0 (low byte), followed by byte 1.
0007h	R/W	DMA channel 3 word count byte 0 (low byte), followed by byte 1.
0008h	R	DMA channels 0–3 status register Bit 7 1 Channel 3 request Bit 6 1 Channel 2 request Bit 5 1 Channel 1 request Bit 4 1 Channel 0 request Bit 3 1 Terminal count on channel 3 Bit 2 1 Terminal count on channel 2 Bit 1 1 Terminal count on channel 1 Bit 0 1 Terminal count on channel 0
0008h	W	DMA channels 0–3 command register Bit 7 0 DACK sense active low 1 DACK sense active high Bit 6 0 DREQ sense active low 1 DREQ sense active high Bit 5 0 Late write selection 1 Extended write selection Bit 4 0 Fixed priority 1 Rotating priority Bit 3 0 Normal timing 1 Compressed timing Bit 2 0 Enable controller 1 Disable controller Bit 1 0 Disable memory-to-memory transfer 1 Enable memory-to-memory transfer Bit 0 Reserved
0009h	W	DMA write request register

I/O Port	Read/ Write	Description
000Ah	R/W	DMA channel 0–3 mask register Bits 7–3 Reserved Bit 2 0 Clear mask bit 1 Set mask bit Bits 1–0 Channel select 00 Channel 0 01 Channel 1 10 Channel 2 11 Channel 3
000Bh	W	DMA channel 0–3 mode register Bits 7–6 Mode select 00 Demand mode 01 Single mode 10 Block mode 11 Cascade mode Bit 5 0 Address increment select 1 Address decrement select Bit 4 0 Disable autoinitialization 1 Enable autoinitialization Bits 3–2 Select type of operation 00 Verify operation 01 Write to memory 10 Read from memory 11 Reserved Bits 1–0 Channel select 00 Channel 0 01 Channel 1 10 Channel 2 11 Channel 3
000Ch	W	DMA clear byte pointer flip/flop
000Dh	R	DMA read temporary register
000Eh	W	DMA clear mask register
000Fh	W	DMA write mask register
I/O ports 0020h – 0021h are used by the programmable interrupt controller.		
0020h	W	If bit 4 is set, this is the programmable interrupt controller Initialization Command Word 1 (ICW1). Bits 7–5 000 Only used in 8080 or 8085 mode Bit 4 1 Using ICW1 Bit 3 0 Edge-triggered mode 1 Level-triggered mode Bit 2 0 Successive interrupt vectors separated by eight bytes 1 Successive interrupt vectors separated by four bytes Bit 1 0 Cascade mode 1 Single mode. ICW3 is not necessary Bit 0 0 ICW4 is not necessary 1 ICW4 is necessary

I/O Port	Read/ Write	Description
0021h	W	This port can represent ICW2, ICW3, and ICW4 in sequence after ICW1 is written to I/O port 0020h. **If ICW2** Bits 7–3 Address lines A0–A3 of the base vector address for the interrupt controller. Bits 2–0 Reserved (should be zeroes) **If ICW3 for the slave controller (00A1h)** Bits 7–3 Reserved Bits 2–0 Slave ID **If ICW4** Bits 7–5 Reserved (should be zeroes) Bit 4 0 No special fully-nested mode 1 Special fully-nested mode Bits 3–2 Mode 00 Non-buffered mode 01 Non-buffered mode 10 Buffered mode/slave 11 Buffered mode/master Bit 1 0 Normal EOI 1 Auto EOI Bit 0 0 8085 mode 1 8086 and 8088 mode
0021h	R/W	Programmable interrupt controller master interrupt mask register — Operation Command Word 3 (OCW1) Bit 7 0 Enable parallel printer interrupt Bit 6 0 Enable floppy disk drive interrupt Bit 5 0 Enable hard disk drive interrupt Bit 4 0 Enable serial port 1 interrupt Bit 3 0 Enable serial port 2 interrupt Bit 2 0 Enable video interrupt Bit 1 0 Enable keyboard/mouse/RTC interrupt Bit 0 0 Enable timer interrupt
0021h	W	If Bits 4 and 3 are 0, Programmable interrupt controller OCW2 Bits 7–5 000 Rotate in automatic EOI mode (clear) 001 Non-specific EOI 010 No op 011 Specific EOI 100 Rotate in automatic EOI mode (set) 101 Rotate on nonspecific EOI command 110 Set priority command 111 Rotate on specific EOI command Bits 4–3 Reserved (should be zeroes) Bits 2–0 The interrupt request to which the command applies

I/O Port	Read/ Write	Description
0020h	R	Programmable interrupt controller Interrupt request and In-Service registers programmed by OCW3 **Interrupt request register** Bits 7–0 0 No active request for the corresponding interrupt line 1 Active request for the corresponding interrupt line **Interrupt in-service register** Bits 7–0 0 The corresponding interrupt line is not being serviced now 1 The corresponding interrupt line is being serviced now
0020h	W	If Bit 4 is 0 and Bit 2 is 1, Programmable interrupt controller OCW3 Bit 7 Reserved (should be zero) Bits 6–5 00 No op 01 No op 10 Reset special mask 11 Set special mask Bit 4 Reserved (should be zero) Bit 3 Reserved (One) Bit 2 0 No poll command 1 Poll command Bits 1–0 00 No op 01 No op 10 Read interrupt request register on next read of port 0020h 11 Read interrupt in-service register on next read of port 0020h
I/O ports 0040 – 005Fh can be used by the Programmable Interrupt Timer.		
0040h	R/W	Programmable interrupt timer R/W counter 0, keyboard controller channel 0
0041h	R/W	Programmable interrupt time channel 1
0042h	R/W	Programmable interrupt timer miscellaneous register channel 2
0043h	W	Programmable interrupt timer mode port. Control word register for counters 0 and 2 Bits 7–6 00 Counter 0 select 01 Counter 1 select 10 Counter 2 select Bits 5–4 00 Counter latch command 01 R/W counter bits 0–7 only 10 R/W counter bits 8–15 only 11 R/W counter bits 0–7 first, then bits 8–15 Bits 3–1 Select mode 000 Mode 0 programmable one-shot x10 Mode 1 rate generator x11 Mode 3 square wave generator 100 Mode 4 software-triggered strobe 101 Mode 5 hardware-triggered strobe Bit 0 0 Binary counter is 16 bits 1 Binary code decimal (BCD) counter
0044h	W	Programmable interrupt controller miscellaneous register (EISA)y

I/O Port	Read/ Write	Description
0047h	W	Programmable interrupt timer Control word register four counter 0 (EISA) Bits 7–6 00 Counter 0 All other values reserved Bits 5–4 00 Counter latch command select counter 0 01 R/W counter bits 0–7 only All other values reserved
0048h	R/W	Programmable interrupt Timer
0060h	R	Keyboard controller data port or keyboard input buffer. If Keyboard input buffer (can also be 64h): Bit 7 0 Keyboard inhibited Bit 6 0 Primary display is VGA 1 Primary display is MDA. Bit 5 0 System BIOS performs diagnostics on the motherboard in an infinite loop. 1 Any other diagnostic function Bit 4 Motherboard RAM 0 256 KB 1 512 KB or greater Bits 3–1 Reserved Bit 0 0 The motherboard passed the diagnostics tests when diagnostic mode was enabled. The LED blinks in manufacturing diagnostic mode.
0060h	W	Keyboard output port (can also be port 64h) Bit 7 0 Keyboard data is being transferred Bit 6 0 The keyboard clock signal is being used in data transfer Bit 5 0 PC-type mouse being used 1 PS/2-type mouse being used Bit 4 0 Output buffer full, IRQ1 generated 1 Output buffer not full Bits 3–2 Reserved Bit 1 0 The system processor address 20 line is inhibited on the system bus. 1 Address line 20 is not inhibited Bit 0 0 Reset system processor 1 This bit should always be kept at 1.
0061h	R	Port B control register (EISA systems) Bit 7 1 Parity check Bit 6 1 Channel check Bit 5 1 Timer 2 output Bit 4 1 Toggles with each refresh request Bit 3 1 Channel check enable Bit 2 1 Parity check enable Bit 1 1 Speaker data enable Bit 0 1 Timer 2 gate to speaker enable
0061h	W	Port B Control register (EISA systems) Bits 7–4 Reserved Bit 3 1 Channel check enable Bit 2 1 Parity check enable Bit 1 1 Speaker data enable Bit 0 1 Timer 2 gate to speaker enable

I/O Port	Read/ Write	Description
0064h	R	Keyboard controller read status Bit 7 0 No parity error 1 Parity error on last byte of transmission from keyboard Bit 6 0 No timeout 1 Received a timeout on last transmission Bit 5 0 No timeout 1 Transmission from keyboard controller to keyboard timed out Bit 4 0 Keyboard inhibited 1 Keyboard not inhibited Bit 3 0 Data. System writes to input buffer via I/O port 60h 1 Command. System writes to input buffer via I/O port 64h Bit 2 System Flag status. Set to 0 after a power on reset. The keyboard controller sets this bit according to the command from the system. Bit 1 0 Input buffer (60h or 64h) is empty 1 Input buffer full Bit 0 0 Output buffer has no data 1 Output buffer full
0070h	R	Real Time Clock (CMOS RAM) register and NMI mask Bit 7 1 NMI disabled Bits 6–0 0 CMOS RAM index
0071h	R/W	CMOS RAM data register port
0080h	R	Manufacturing test port (POST checkpoints can be accessed via this port).
0080h	R/W	Temporary storage for additional DMA page register
0081h	R/W	DMA channel 2 address byte 2
0082h	R/W	DMA channel 2 address byte 3
0083h	R/W	DMA channel 1 address byte 2
0084h	R/W	Additional DMA page register
0085h	R/W	Additional DMA page register
0086h	R/W	Additional DMA page register
0087h	R/W	DMA channel 0 address byte 2
0088h	R/W	Additional DMA page register
0089h	R/W	DMA channel 6 address byte 2
008Ah	R/W	DMA channel 7 address byte 2
008Bh	R/W	DMA channel 5 address byte 2
008Ch	R/W	Additional DMA page register
008Dh	R/W	Additional DMA page register
008Eh	R/W	Additional DMA page register
008Fh	R/W	DMA refresh page register
00A0h – 00A1h are used for the slave programmable interrupt controller. Except for the differences noted below, the bit definitions are the same as those for addresses 0020h – 0021h.		
00A0h	R/W	Programmable interrupt controller 2

I/O Port	Read/ Write	Description
00A1h	R/W	Programmable interrupt controller 2 mask (OCW1) Bit 7 0 Reserved Bit 6 0 Enable hard disk drive interrupt Bit 5 0 Enable coprocessor exception interrupt Bit 4 0 Enable mouse interrupt Bits 3–2 Reserved (should be zeroes) Bit 1 0 Enable redirect cascade Bit 0 0 Enable real time clock interrupt
00C0h	R/W	DMA channel 4 memory address bytes 1 and 0 (low)
00C2h	R/W	DMA channel 4 transfer count bytes 1 and 0 (low)
00C4h	R/W	DMA channel 5 memory address bytes 1 and 0 (low)
00C6h	R/W	DMA channel 5 transfer count bytes 1 and 0 (low byte)
00C8h	R/W	DMA channel 6 memory address bytes 1 and 0 (low byte)
00CAh	R/W	DMA channel 6 transfer count bytes 1 and 0 (low byte)
00CCh	R/W	DMA channel 7 memory address bytes 1 and 0 (low byte)
00CEh	R/W	DMA channel 7 transfer count bytes 1 and 0 (low byte)
00D0h	R	DMA channels 4–7 status register Bit 7 1 Channel 7 request Bit 6 1 Channel 6 request Bit 5 1 Channel 5 request Bit 4 1 Channel 4 request Bit 3 1 Terminal count on channel 7 Bit 2 1 Terminal count on channel 6 Bit 1 1 Terminal count on channel 5 Bit 0 1 Terminal count on channel 4
00D0h	W	DMA channel 4–7 command register Bit 7 0 DACK sense active low 1 DACK sense active high Bit 6 0 DREQ sense active high 1 DREQ sense active low Bit 5 0 Late write selection 1 Extended write selection Bit 4 0 Fixed priority 1 Rotating priority Bit 3 0 Normal timing 1 Compressed timing Bit 2 0 Enable controller 1 Disable controller Bit 1 0 Disable memory-to-memory transfer 1 Enable memory-to-memory transfer Bit 0 Reserved
00D2h	W	DMA channel 4–7 write request register

I/O Port Addresses, Continued

I/O Port	Read/ Write	Description
00D4h	W	DMA channel 4–7 write single mask register bit Bits 7–3 Reserved (should be zeroes) Bit 2 0 Clear mask bit 1 Set mask bit Bits 1–0 00 Channel 4 select 01 Channel 5 select 10 Channel 6 select 11 Channel 7 select
00D6h	W	DMA channels 4–7 mode register Bits 7–6 00 Demand mode 01 Single mode 10 Block mode 11 Cascade mode Bit 5 0 Address increment select 1 Address decrement select Bit 4 0 Disable Autoinitialization 1 Enable autoinitialization Bits 3–2 00 Verify operation 01 Write to memory 10 Read from memory 11 Reserved Bits 1–0 00 Channel 4 select 01 Channel 5 select 10 Channel 6 select 11 Channel 7 select
00D8h	W	DMA channel 4–7 clear byte pointer flip/flop
00DAh	R	DMA channel 4–7 read temporary register
00DAh	W	DMA channel 4–7 master clear
00DCh	W	DMA channel 4–7 clear mask register
00DEh	W	DMA channel 4–7 write mask register
00F0h		Math coprocessor clear busy latch
00F1h		Math coprocessor reset
00F28–00FFh	R/W	Math coprocessor
I/O ports 0170h–0177h are used as a secondary hard disk area. See the definition of I/O ports 01F0–01F7h for the bit definitions.		
0170h	R/W	Hard disk 1 data register
0171h	R	Hard disk 1 error register
0171h	W	Hard disk 1 write precompensation register
0172h	R/W	Hard disk 1 sector count
0173h	R/W	Hard disk 1 sector number
0174h	R/W	Hard disk 1 number of cylinders, low byte
0175h	R/W	Hard disk 1 number of cylinders, high byte
0176h	R/W	Hard disk 1 drive/head register
0177h	R	Hard disk 1 status register
0177h	W	Hard disk 1 command register

I/O Port	Read/ Write	Description
01F0h	R/W	Hard disk 0 data register base port
01F1h	R	Hard disk 0 error register **Diagnostic mode** Bits 7–3 Reserved Bits 2–0 Diagnostic mode errors 001 No errors 010 Controller error 011 Sector buffer error 100 ECC device error 101 Control processor error **Operation mode** Bit 7 0 Block is not bad 1 Bad block detected Bit 6 0 No error 1 Uncorrectable ECC error Bit 5 Reserved Bit 4 0 ID not found 1 ID found Bit 3 Reserved Bit 2 0 Command aborted 1 Command completed Bit 1 0 Track 000 found 1 Track 000 not found Bit 0 0 DAM found (CP-3002 is always 0) 1 DAM not found
01F1h	W	Hard disk 0 write precompensation register
01F2h	R/W	Hard disk 0 sector count
01F3h	R/W	Hard disk 0 sector number
01F4h	R/W	Hard disk 0 number of cylinders, low byte
01F5h	R/W	Hard disk 0 number of cylinders, high byte
01F6h	R/W	Hard disk 0 drive/head register Bit 7 1 Bit 6 0 Bit 5 1 Bit 4 Drive select 0 first hard disk drive 1 Second hard disk drive Bits 3–0 Head select bits
01F7h	R	Hard disk 0 status register Bit 7 1 Controller is executing a command Bit 6 1 Drive is ready Bit 5 1 Write fault Bit 4 1 Seek complete Bit 3 1 Sector buffer requires servicing Bit 2 1 Disk data read corrected Bit 1 An index. Set to 1 at each disk revolution Bit 0 1 Previous command ended with an error
01F7h	R	Hard disk drive 0 command register

I/O Port Addresses, Continued

I/O Port	Read/ Write	Description
0200 – 020Fh	R/W	Game controller ports
0201h	R/W	Game port I/O data
020C – 020Dh		Reserved for special use by AMIBIOS.
021Fh		Reserved for special use by AMIBIOS.
0278 – 027Fh		Parallel port 2. See the descriptions of I/O ports 0378h–037Ah for the parallel port bit definitions.
02E8h – 02EFh		Serial port 4. See the descriptions of I/O ports 03F8h–03FFh for the serial port bit definitions.
02F8 – 02FFh		Serial port 2. See the descriptions of I/O ports 03F8h–03FFh for the serial port bit definitions.
0300 – 031Fh		Prototype card
0364 – 0367h		Reserved for special use by AMIBIOS.
036C – 036Fh		Reserved for special use by AMIBIOS.
I/O ports 0372h–0377h are used for the secondary floppy disk controller. See the definitions of I/O ports 03F2h–03F7h for the bit definitions.		
0372h	W	Floppy disk controller 2 digital output register
0374h	R	Floppy disk controller 2 status register
0375h	R/W	Floppy disk controller 2 data register
0376h	R/W	Floppy disk controller 2 control register
0377h	R	Floppy disk controller 2 digital input register
0377h	W	Select register for floppy disk data transfer rate
0378h	R/W	Parallel port 1 data port
0379h	R/W	Parallel port 1 status port Bit 7 0 Busy Bit 6 0 Acknowledge Bit 5 1 Out of paper Bit 4 1 Printer is selected Bit 3 0 Error Bit 2 0 IRQ occurred Bits 1–0 Reserved
037Ah	R/W	Parallel port 1 control port Bits 7–5 Reserved Bit 4 1 Enable IRQ Bit 3 1 Select printer Bit 2 0 Initialize printer Bit 1 1 Automatic line feed Bit 0 1 Strobe

I/O Port	Read/ Write	Description
037Bh	R/W	Hercules configuration switch registers Bits 7–2 Not used Bit 1 0 Disable upper 32 KB of graphics mode buffer 1 Enable upper 32 KB of graphics mode buffer at B800:0000h Bit 0 0 Disable graphics mode
03B0–03B3h	R/W	Video registers. See the Video I/O Port tables on page 88 for more information about video I/O ports.
03B4h	R/W	MDA CRTC index register
03B5h	R/W	MDA CRTC data registers
03B8h	R/W	MDA mode control register
03BCh – 03BFh		Parallel port 3. See the descriptions of I/O ports 0378h–037Ah for the parallel port bit definitions.
03C0 – 03CFh		EGA and VGA video subsystem
03C2h	R	CGA input status register
03C3h	R/W	Video subsystem enable
03C4h	R/W	CGA sequencer index register
03C5h	R/W	Other CGA sequencer registers
03CAh	R	CGA feature control register
03D4h	W	Video CRTC index register
03D5h	W	Other CRTC registers
03D8h	R/W	CGA mode control register
03D9h	R/W	CGA palette register
03E8 – 03EFh		Serial port 3. See the descriptions of I/O ports 03F8h–03FFh for the serial port bit definitions.
03F2h	W	Floppy disk controller digital output register Bits 7–6 Reserved. Should be zero. Bit 5 1 Enable motor on floppy drive 1 Bit 4 1 Enable motor on floppy drive 0 Bit 3 1 Enable DMA for floppy drives Bit 2 0 Controller reset Bit 1 Reserved. Should be zero. Bit 0 0 Select floppy drive 0 1 Select floppy drive 1
03F4h	R	Floppy disk controller status register Bit 7 1 data register is ready Bit 6 0 Transfer from system to controller 1 Transfer from controller to system Bit 5 1 Non-DMA mode Bit 4 1 Floppy disk controller busy Bits 3–2 Reserved Bit 1 1 Drive 1 is busy Bit 0 1 Drive 0 is busy
03F5h	R/W	Floppy disk controller data register

I/O Port	Read/ Write	Description
03F6h	R	Floppy disk controller control port Bits 7–4 Reserved Bit 3 0 Reduce write current 1 Head select enable Bit 2 0 Disable floppy disk reset 1 Enable floppy disk reset Bit 1 0 Enable floppy disk initialization 1 Disable floppy disk initialization Bit 0 Reserved
03F7h	R	Floppy disk controller input register. Bits 7–1 apply to the floppy drive that is currently selected. Bit 7 1 Floppy disk change line Bit 6 1 Write gate Bit 5 Head select 3/Reduced write current Bit 4 Head select 2 Bit 3 Head select 1 Bit 2 Head select 0 Bit 1 Select drive 1 Bit 0 Select drive 0
03F7h	W	Floppy disk controller select register for data transfer rate Bits 7–2 Reserved Bits 1–0 00 500 Kbs mode 01 300 Kbs mode 00 250 kbs mode
03F8h	W	Transmitter Holding Register (contains the character to be sent). Bit 0, the least significant bit, is sent first. Bits 7–0 Contains data bits 7–0 when the Divisor Latch Access Bit (DLAB) is 0.
03F8h	R	Receiver Buffer Register (contains the received character). Bit 0, the least significant bit, is received first. Bits 7–0 Contains data bits 7–0 when the Divisor Latch Access Bit (DLAB) is 0.
03F8h	R/W	Divisor Latch, low byte Both divisor latch registers store the data transmission rate divisor. Bits 7–0 Bits 7–0 of divisor when DLAB is 1.
03F9h	R/W	Divisor Latch, high byte. Bits 7–0 Bits 15–8 of data transmission rate divisor when DLAB is 1.
03F9h	R/W	Interrupt Enable Register. Permits the serial port controller interrupts to enable the chip interrupt output signal. Bits 7–4 Reserved Bit 3 Modem status interrupt enable if set. Bit 2 Receiver line status interrupt enable if set. Bit 1 Transmitter Holding register empty interrupt enable if set. Bit 0 Received data available interrupt enable when DLAB is 0 if set.

I/O Port	Read/ Write	Description
03FAh	R	Interrupt ID Register. Information about a pending interrupt is stored here. When the ID register is addressed, the highest priority interrupt is held and no other interrupts are acknowledged until the microprocessor services that interrupt. Bits 7–3 Reserved Bits 2–1 The pending interrupt that has the highest priority. 11 Receiver Line Status Interrupt, priority is the highest. 10 Received Data Available, second in priority. 01 Transmitter Holding Register Empty, third in priority. 00 Modem Status Interrupt, fourth in priority. Bit 0 Interrupt pending if set to logical 0. If logical 1, no interrupt is pending.
03FBh	R/W	Line Control Register Bit 7 Divisor Latch Access Bit (DLAB) 0 Access receiver buffer, transmitter holding register, and interrupt enable register. 1 Access Divisor Latch of baud rate generator. Bit 6 Set Break Control. Serial output is forced to spacing state and remains there if set. Bit 5 Stick Parity. Bit 4 Even Parity Select. Bit 3 Parity Enable. Bit 2 Number of Stop Bits per Character. 0 One stop bit. 1 1½ stop bits if 5-bit word length is selected. 2 stop bits if 6, 7, or 8-bit word length is selected. Bits 1–0 Number of Lines per character 00 5-Bit word length. 01 6-Bit word length. 10 7-Bit word length. 11 8-Bit word length.
03FCh	R/W	Modem Control Register Bits 7–5 Reserved Bit 4 Loopback mode for diagnostic testing of serial port if set. The output from the transmitter shift register is looped back to the receiver shift register input. Transmitted data is immediately received so the microprocessor can verify the transmit and receive data serial port paths. Bit 3 Force OUT2 interrupt if set. Bit 2 Force OUT1 active if set. Bit 1 Force Request To Send active if set. Bit 0 Force Data Terminal Ready active if set.

I/O Port	Read/ Write	Description
03FDh	R	Line Status Register Bit 7 Reserved Bit 6 Transmitter shift and holding registers empty if set. Bit 5 Transmitter holding register empty if set. The controller is ready to accept a new character to send. Bit 4 Break interrupt if set. The received data input is held in the zero bit state longer than the transmission time of the start bit + data bits + parity bits + stop bits. Bit 3 Framing error if set. The stop bit that follows the last parity or data bit is zero. Bit 2 Parity error if set. The character has incorrect parity. Bit 1 Overrun error if set. A character was sent to the receiver buffer before the previous character in the buffer could be read, which destroys the previous character. Bit 0 Data Ready if set. A complete incoming character has been received and sent to the receiver buffer register.
03FEh	R	Modem Status Register Bit 7 Data Carrier Detect if set. Bit 6 Ring Indicator if set. Bit 5 Data Set Ready if set. Bit 4 Clear To Send if set. Bit 3 Delta Data Carrier Detect if set. Bit 2 Trailing Edge Ring Indicator if set. Bit 1 Delta Data Set Ready if set. Bit 0 Delta Clear To Send if set.
03FFh	R/W	Serial port 1 scratch register
I/O ports 0401h–04D6h are only used by EISA systems.		
0401h	R/W	DMA channel 0 word count byte 2, high byte
0403h	R/W	DMA channel 1 word count byte 2, high byte
0405h	R/W	DMA channel 2 word count byte 2, high byte
0407h	R/W	DMA channel 3 word count byte 2, high byte
040Ah	W	Extended DMA chaining mode register, channels 0–3 Bits 7–5 Reserved Bit 4 0 Generate IRQ13 1 Generate terminal count Bit 3 0 Do not start chaining 1 Programming complete Bit 2 0 Disable buffer chaining mode (default value) 1 Enable buffer chaining mode Bits 1–0 DMA channel select 00 Channel 0 01 Channel 1 10 Channel 2 11 Channel 3
040Ah	R	Channel interrupt (IRQ13) status register Bits 7–5 Interrupt on channels 7–5 Bit 4 Reserved Bits 3–0 Interrupt on channels 3–0

I/O Port	Read/ Write	Description
040Bh	W	DMA extended mode register for channels 0–3 Bit 7 0 Enable stop register Bit 6 0 Terminal count is an output for this channel (default) Bits 5–4 DMA cycle timing 00 ISA-compatible (default) 01 Type A timing mode 10 Type B timing mode 11 DMA burst mode Bits 3–2 Address mode 00 8-bit I/O, count by bytes (default) 01 16-bit I/O, count by words, address-shifted 10 32-bit I/O, count by bytes 11 16-bit I/O, count by bytes Bits 1–0 DMA channel select
0461h	R/W	Extended NMI status and control register Bit 7 1 NI pending from fail-safe timer (read only) Bit 6 1 NMI pending from bus timeout NMI status (read only) Bit 5 1 NMI pending (read only) Bit 4 Reserved Bit 3 1 Bus timeout NMI enable (R/W) Bit 2 1 Fail-safe NMI enable (R/W) Bit 1 1 NMI I/O port enable (R/W) Bit 0 RSTDRV. Bus reset (R/W) 0 Normal bus reset 1 Reset bus asserted
0462h	W	Software NMI register. Writing to this register causes an NMI if NMIs are enabled.
0464h	R	Bus master status latch enable register (slots 1–8). Identifies the last bus master to control the bus. Bit 7 0 Slot 8 Bit 6 0 Slot 7 Bit 5 0 Slot 6 Bit 4 0 Slot 5 Bit 3 0 Slot 4 Bit 2 0 Slot 3 Bit 1 0 Slot 2 Bit 0 0 Slot 1
0465h	R	Bus master status latch enable register (slots 9–16). Identifies the last bus master to control the bus. Bit 7 0 Slot 16 Bit 6 0 Slot 15 Bit 5 0 Slot 14 Bit 4 0 Slot 13 Bit 3 0 Slot 12 Bit 2 0 Slot 11 Bit 1 0 Slot 10 Bit 0 0 Slot 9
0481h	R/W	DMA channel 3 address byte 3, high byte
0483h	R/W	DMA channel 2 address byte 3, high byte
0485h	R/W	DMA channel 1 address byte 3, high byte

I/O Port Addresses, Continued

I/O Port	Read/ Write	Description
0487h	R/W	DMA channel 7 address byte 3, high byte
0489h	R/W	DMA channel 6 address byte 3, high byte
048Bh	R/W	DMA channel 7 address byte 3, high byte
048Dh	R/W	DMA channel 5 address byte 3, high byte
04C6h	R/W	DMA channel 5 word count byte 2, high byte
04CAh	R/W	DMA channel 6 word count byte 2, high byte
04CEh	R/W	DMA channel 7 word count byte 2, high byte
04D0h	W	IRQ0–IRQ7 interrupt edge/level registers Bit 7 1 IRQ7 is level-sensitive Bit 6 1 IRQ6 is level-sensitive Bit 5 1 IRQ5 is level-sensitive Bit 4 1 IRQ4 is level-sensitive Bit 3 1 IRQ3 is level-sensitive Bits 2–0 Reserved
04D1h	W	IRQ8–IRQ15 interrupt edge/level registers Bit 7 1 IRQ15 is level-sensitive Bit 6 1 IRQ14 is level-sensitive Bit 5 Reserved Bit 4 1 IRQ12 is level-sensitive Bit 3 1 IRQ11 is level-sensitive Bit 2 1 IRQ10 is level-sensitive Bit 1 1 IRQ9 is level-sensitive Bit 0 Reserved
04D4h	R	Chaining mode status register Bits 7–5 1 Enable Channels 7–5 Bit 4 Reserved Bits 3–0 1 Enable Channels 3–0
04D4h	W	Extended DMA chaining mode register, channels 4–7 Bits 7–5 Reserved Bit 4 0 Generate IRQ13 1 Generate terminal count Bit 3 0 Do not start chaining 1 Programming complete Bit 2 0 Disable buffer chaining mode (default value) Bits 1–0 Select DMA channel 00 Channel 4 01 Channel 5 10 Channel 6 11 Channel 7
04D6h	W	DMA extended mode register for channels 4–7. See I/O port 040Bh for the bit settings.
0500h-07FFh		A copy of all I/O port assignments from 0110h-03FFh is placed here in EISA systems.
0800h–08FFh	R/W	I/O port access registers for EISA CMOS RAM
0900h-0BFFh		A copy of all I/O port assignments from 0110h-03FFh is placed here in EISA systems.
0C00h	R/W	Page register to write to SRAM or I/O

I/O Port	Read/ Write	Description
0C80h	R/W	EISA motherboard ID Bit 7 Reserved. Should be zero. Bits 6–2 First letter of manufacturer code. Bits 1–0 First two bits of second letter of manufacturer code
0C81h	R/W	EISA motherboard ID Bits 7–5 Remaining 3 bits of second letter of manufacturer code. Bits 4–0 Third letter of manufacturer code.
0C82h	R/W	EISA motherboard ID Bits 7–0 Reserved for use by manufacturer. Often used for manufacturer's product number. American Megatrends EISA motherboards have the serial number in BCD.
0C83h	R/W	EISA motherboard ID Bits 7–3 Product Revision Number assigned by EISA motherboard manufacturer. Bits 2–0 EISA Bus Version (initial version is 001) 001 is currently the only standard value defined for this field, but, in practice, EISA motherboard and adapter card manufacturers have been using this field. In American Megatrends EISA motherboards: Bits 7–4 EISA Configuration file revision number Bit 3 Reserved Bits 2–0 EISA bus version
0C84h	R/W	American Megatrends EISA Motherboard ID data Bits 7–3 Schematic release (for internal use only) Bits 2–0 PCB release (for internal use only)
0C85h	R/W	American Megatrends EISA Motherboard ID data CPU speed in MHz (in BCD)
0D00h-0FFFh		A copy of all I/O port assignments from 0110h-03FFh is placed here in EISA systems.
EISA Adapter Card Ports In the following rows, *n* can be 1h – Fh and represents EISA expansion slots 1 – 15.		
*n*000-*n*0FFh		EISA expansion slot *n*.
*n*100-*n*3FFh		A copy of all I/O port assignments from 0100h – 03FFh.
*n*4000-*n*4FFh		EISA expansion slot *n*.
*n*500-*n*7FFh		A copy of all I/O port assignments from 0100h – 03FFh.
*n*800-*n*8FFh		EISA expansion slot *n*.
*n*900-*n*BFFh		A copy of all I/O port assignments from 0100h – 03FFh.
*n*C00-*n*CFFh		EISA expansion slot *n*.
*n*D00-*n*FFFh		A copy of all I/O port assignments from 0100h – 03FFh.

I/O Port	Read/ Write	Description
*n*000- *n*0FFh		EISA expansion slot *n*.
*n*100- *n*3FFh		A copy of all I/O port assignments from 0100h – 03FFh.
*n*4000- *n*4FFh		EISA expansion slot *n*.
*n*500- *n*7FFh		A copy of all I/O port assignments from 0100h – 03FFh.
*n*800- *n*8FFh		EISA expansion slot *n*.
*n*900- *n*BFFh		A copy of all I/O port assignments from 0100h – 03FFh.
*n*C00- *n*CFFh		EISA expansion slot *n*.
*n*D00- *n*FFFh		A copy of all I/O port assignments from 0100h – 03FFh.

EISA Adapter Card Compressed ID

The following I/O ports are used by EISA adapter cards in the slot number corresponding to *n (1h – Fh)*.

I/O Port	Description of Contents
0*n*C80h	Bit 7 Reserved. Should be zero. Bits 6–2 First letter of manufacturer code. Bits 1–0 First two bits of second letter of manufacturer code.
0*n*C81h	Bits 7–5 Remaining 3 bits of second letter of manufacturer code. Bits 4–0 Third letter of manufacturer code.
0*n*C82h	Bits 7–4 First hex digit of product number. Bits 3–0 Second hex digit of product number.
0*n*C83h	Bits 7–4 First hex digit of revision number. Bits 3–0 Second hex digit of revision number.
0*n*C84h	EISA Adapter Card Control Register Bits 7–3 Reserved. Should be zero. Bit 2 IOCHKRST (write/only) 0 Normal Operation. 1 Reset the adapter card after sending an Active High Pulse. Bit 1 IOCHKERR (read/only) 0 No I/O error pending. 1 I/O error detected by adapter card. Bit 0 ENABLE (Read/Write) 0 Adapter Card Disable. 1 Adapter Card Enable.

Some I/O devices (such as the video controllers) also use system memory addresses as well as their assigned I/O port addresses. This technique, memory-mapped I/O, makes the microprocessor think that the devices are part of system memory. Memory mapped devices are easier to program because they permit more flexible memory instructions.

The following table defines the video ports used in the MDA and CGA video standard. The EGA, VGA, Super VGA, and XGA video standards use I/O ports much more extensively, but since the system BIOS does not perform EGA, VGA, or XGA video, they are not discussed here.

MDA I/O Ports

The 6845 CRTC index register is mapped to I/O port 03B4h. The index value in port 03B4h controls the register that appears at I/O port 03B5h. The 68345 mode control register is accessed directly through I/O port 03B8h.

I/O Port	Read/ Write	Description
03B4h	W	CRTC index register
03B5h	W	Index 00h Horizontal total Index 01h Horizontal displayed Index 02h Horizontal sync position Index 03h Horizontal sync pulse width Index 04h Vertical total Index 05h Vertical displayed Index 06h Vertical sync position Index 07h Vertical sync pulse width Index 08h Interlace mode Index 09h Maximum scan lines Index 0Ah Cursor start Index 0Bh Cursor end Index 0Ch Start address, high byte Index 0Dh Start address, low byte Index 0Eh Cursor location, high byte Index 0Fh Cursor location, low byte Index 10h Light pen, high byte Index 11h Light pen, low byte
03B8h	W	Mode control register
03BAh	R	CRT status register

cont'd

CGA I/O Ports

The 6845 CRTC index register is mapped to I/O port 03D4h. The value written to 03D4h controls the register that appears in port 03D5h.

I/O Port	Read/ Write	Description
03D4h	W	CRTC index register
03D5h	W	Index 00h Horizontal total Index 01h Horizontal displayed Index 02h Horizontal sync position Index 03h Horizontal sync pulse width Index 04h Vertical total Index 05h Vertical displayed Index 06h Vertical sync position Index 07h Vertical sync pulse width Index 08h Interleaved mode Index 09h Maximum scan lines Index 0Ah Cursor start Index 0Bh Cursor end Index 0Ch Start address, high byte Index 0Dh Start address, low byte Index 0Eh Cursor location, high byte Index 0Fh Cursor location, low byte Index 10h Light pen, high byte Index 11h Light pen, low byte
03D8h	W	Mode control register
03D9h	W	Palette register
03DAh	R	CRT status register
03DBh	W	Clear light pen latch
03DCh	W	Preset light pen latch

Chapter 8

Power On Self Test

The first routine that executes in the system BIOS is called the Power On Self Test (POST). POST must be executed before any ISA or EISA system can be used.

POST performs diagnostic tests on system memory and key system components. It also initializes BIOS configuration tables. It then boots the operating system.

Starting POST

You can start POST in several different ways:

Starting POST	What the BIOS Does
Turn the system on.	Jumps to the address pointed to by the processor reset vector (FFFF0h). All POST tests and initializations are then executed. If successful, POST calls INT 19h Bootstrap Loader.
Press the reset button.	Jumps to the address pointed to by the processor reset vector (FFFF0h). All POST tests and initializations are then executed. If successful, POST calls INT 19h Bootstrap Loader.
Press <Ctrl> <Alt> <Del>.	The INT 09h keyboard hardware interrupt service transfers control to POST. POST does not test memory above 64 KB, but all other tests and initializations are performed. POST then calls INT 19h Bootstrap Loader.

POST Phases

Before POST Enables NMIs

NMI and I/O checks are disabled. Before NMIs are enabled, the BIOS POST:

1. Writes data in all motherboard and adapter card memory locations to establish that parity is good.

2. Enables the onboard and 32-bit slot memory parity checks by writing to I/O port 61h with data bit 2 set to zero.

3. Enables the I/O channel check signal by writing to I/O Port 61h with data bit 3 set to zero.

POST Functions

POST usually performs the following tests in the following order. In some AMIBIOSes, some tests are performed in a slightly different sequence. The errors that can be generated by the test are listed below.

If the contents of 0Fh at CMOS RAM location 0Fh (Shutdown Byte) is 00h, POST performs all tests and initializations.

Processor Register Test

The following values are loaded consecutively into all registers: 0555h, 0AAAh, 0CCCh, and 0F0Fh. If any register does not retain any of these values, Beep Code 5 is issued.

ROM BIOS Checksum Test

A checksum is performed on the system BIOS. If it is incorrect, Beep Code 9 is issued.

Keyboard Controller Test

The BIOS issues a keyboard controller BAT command. If the response is not 55h, Beep Code 6 is issued.

CMOS Shutdown Register Test

The BIOS writes the values 55h, then AAh to 0Fh in CMOS RAM. If there is an error, *CMOS not operational* is displayed and the system halts and must be rebooted. Replace the battery on error.

System Timer Test

The *CH-1 timer error* or *CH-2 timer error* is displayed if this test fails on channel 1 or 2 of the timer. Beep Code 4 is issued if timer channel 1 does not work.

Memory Refresh Test

Timer channels 0 and 1 are tested. Beep Code 1 is issued if either channel does not work.

Base 64 KB Test

An address test, sequential read and write, and random read and write test are performed on the first 64 KB of RAM. Beep Code 3 is issued if any errors, including parity errors, occur.

Cache Memory Test

Memory reads are performed with secondary cache memory enabled. Then memory tests are performed with cache disabled. If cache memory is not performing as expected, *Cache Memory bad, do not enable cache* is displayed.

cont'd

CMOS RAM Battery Test

POST reads the CMOS RAM status register (40:8Dh) to see if the battery is on. It then reads CMOS diagnostic data and calculates CMOS RAM checksums. The following error messages may be generated: *CMOS battery state low, CMOS system option not set,* or *CMOS checksum error.*

Display Verification

POST does a vertical and horizontal retrace and a sequential read and write of 4 KB in different display modes. Beep Code 8 is issued if there is any error. Other messages that may be issued are: *Display switch setting not proper* or *CMOS display type mismatch.*

Enter Protected Mode

This test issues INT 15h Function 89h. *8042 Gate-A20 error, system halted* is displayed if it cannot get to protected mode.

Address Line Test

A test pattern is written to both conventional and extended memory. Beep Code 3 is issued if any error occurs.

Conventional and Extended Memory Test

Zeroes are written to extended and conventional memory unless <Esc> is pressed. Sequential and random read and write tests are performed. The amount of memory tested is displayed. Beep Code 3 or 7 is issued if there are errors.

DMA Controller Test

Several patterns are written to DMA page registers 80h through 8Fh and then DMA registers 0 through 7. The following errors may be generated: *DMA error, system halted, DMA #1 error, system halted,* or *DMA #2 error, system halted.*

Keyboard Test

The keyboard self-test command is issued. A stuck key check is performed. The keyboard interface test is then performed. Possible errors are: *Keyboard error* or *KB/Interface error.*

System Configuration Verification

The floppy and hard disk areas are initialized and a Seek is performed on drive A: or drive C:. Possible errors are: *FDD controller failure, HDD controller error, C: drive failure,* or *D: drive failure.*

The memory size is verified. *CMOS memory size mismatch* is generated if there is an error.

Adaptor ROMs are checked and the timer data area is initialized. Possible errors: *CMOS time & date not set.*

The parallel and serial ports are configured. *Keyboard is locked* is displayed if the keyboard is locked.

POST Error Handling

One of the primary POST functions is to find and indicate any conditions in the system that prevent proper operation. The POST diagnostic routines look for system errors and report them in the following manner.

If...	Then...
the error occurs before the display device is initialized,	a series of beeps sound. Beep codes indicate that a fatal error has occurred. The AMIBIOS Beep Codes are described on page 393.
the error occurs after the display device is initialized,	the error message is displayed. Displayed BIOS messages are described on page 394. A prompt to press <F1> also appears.

cont'd

Beep Codes

Beep codes occur when the BIOS cannot successfully configure the display. They indicate a serious problem. Errors that cause beep codes occur during POST. POST is performed every time the system is powered on.

All errors except Beep Code 8 are fatal. Fatal errors do not allow the system to continue. Beep codes are described on page 393.

Displayed Errors

If POST is able to configure the system display, it can display errors on the screen. In general, these errors are not as serious as the beep codes. Displayed POST messages are described on page 394.

POST Diagnostic Codes

POST also produces a series of diagnostic codes that indicate specific milestones in the POST code. These codes are described beginning on page 99. AMIBIOS POST codes are accessible via the Manufacturing Test Port (I/O Port 80h).

POST Memory Test

Normally, the only visible POST routine is the memory test. A sample of this screen follows.

```
AMIBIOS (C) 1992 American Megatrends Inc.

xxxxx KB OK

Hit <DEL> if you want to run SETUP



(C) American Megatrends Inc.
XX-XXXX-XXXXXX-XXXXXXXX-XXXXXX-XXXX-X
```

BIOS Identification String

An Identification String is displayed at the left bottom corner of the screen, below the copyright message. In some AMIBIOSes, you can press <Ins> during system boot to display two additional BIOS Identification Strings. These strings display the options in the AMIBIOS. Appendix D on page 417 lists the contents of the Identification Strings.

Freezing the Screen

If your system has a Hi-Flex AMIBIOS, make sure that *Wait for <F1> If any Error* in Advanced CMOS Setup is *Enabled* before using the following procedure.

Freeze the screen by powering on the system and holding a key down on the keyboard. Copy the BIOS Identification Strings on a piece of paper and refer to Appendix D on page 417. Press <F1> to continue the boot process.

POST Memory Test

When POST Completes

The following message is displayed after POST is completed:

```
Hit <DEL> if you want to run SETUP
```

Press <Del> to access the AMIBIOS Setup utility.

BIOS Configuration Summary Screen

The AMIBIOS displays the BIOS Configuration Summary screen (see the sample screen below) when the POST routines complete successfully. This screen may be slightly different in your computer. This screen does not appear in AMIBIOSes manufactured before 12/15/1988.

```
System Configuration (C) Copyright 1985-1991 American Megatrends Inc.

Main Processor          : 80486          Base Memory Size      : 640 KB
Numeric Coprocessor     : Present        Ext. Memory Size      : 7808 KB
Floppy Drive A:         : 1.2 MB 5¼      Hard Disk C: Type     : 44
Floppy Drive B:         : 1.44 MB 3½     Hard Disk D: Type     : None
Display Type:           : VGA or EGA     Serial Port(s)        : 3F8
ROM-BIOS Date:          : 07/07/91       Parallel Port(s)      : 378

Memory Found                             Memory Configured

Bank 1=1 MB Bank 2=1 Meg                 Bank 1=1 MB Bank 2=1 Meg

Shadow RAM      F000=Enable                    Cache Memory=64K

C000=Enable     C400=Enable     C800=Enable     CC00=Enable
D000=Disable    D400=Disable    D800=Disable    DD00=Disable
E000=Disable    E400=Disable    E800=Disable    EC00=Disable
```

ROM Extensions

An adaptor ROM on an adapter card is an optional extension to the system BIOS. Extension ROMs can either replace existing ROM BIOS device service routines or add new service routines. Examples of ROM extensions include an ESDI hard disk drive BIOS or a SCSI BIOS.

POST detects ROM extensions and allows them to initialize themselves and test and initialize the devices that they control. The ROM extensions then return control to POST.

Handling ROM Extensions

By convention, ROM extensions can appear on any 2 KB boundary between C0000h and FFFFFh. POST searches RAM from C0000h through FFFFFh in 2 KB increments for ROM extensions. A ROM extension at E0000h must be 64 KB long.

Identifying a ROM Extension

ROM Extensions must have a standard header. The information in the header indirectly identifies the type of device and its use. The following table lists the most important parts of a ROM extension header.

Offset	Contents	Description
0	55AAh	ROM extension header identifier.
1		Length code. The length in 512-byte (½K) units. A 64 KB ROM extension has a length code of 128.
2		Three-byte instruction. Normally, this field has a one-byte FAR RETURN instruction or a three-byte JMP instruction.
5	Varies	The header contains a number of other fields.
Last	00h	Usually 00h.

System Boot

POST's last act is to boot the operating system by invoking INT 19h, as follows:

If...	and...	then...
The boot sequence is A:, C:,	a bootable floppy disk is in drive A:,	INT 19h reads the boot sector on the floppy disk and places its contents at 7C00h.
The boot sequence is A:, C:,	drives A: and C: have no bootable disk,	INT 19h invokes INT 18h.
The boot sequence is A:, C:,	the floppy disk in drive A: is not bootable, but drive C: is bootable,	INT 19h reads the boot sector on drive A: and places its contents at 7C00h.
The Boot sequence is C:, A:,	a boot sector is found on drive C:,	INT 19h reads the boot sector on drive C: and places its contents at 7C00h.
The boot sequence is C:, A:,	drive C: has no boot sector (the hard disk is not formatted for boot) but drive A: does,	INT 19h reads the boot sector on drive A: and places its contents at 7C00h.
The boot sequence is C:, A:,	neither drive C: or A: has a boot sector,	INT 19h invokes INT 18h.

cont'd

INT 18h

INT 18h is called if INT 19h does not find a boot sector. INT 18h can also be vectored to a routine that takes over the boot process — to permit booting over a network, for example. INT 18h usually points to a routine that displays

```
No Boot Device Available
```

AMIBIOS POST Checkpoint Codes

POST is performed by the BIOS when the system is reset or rebooted. POST performs diagnostics tests on system parts and initializes key system components. When a POST routine completes, a code is written to I/O port address 80h. Display this code by attaching diagnostic equipment to port 80h.

Code	Description
01h	Processor register test about to start. NMIs are disabled next.
02h	NMIs are Disabled. The Power-On delay is starting.
03h	Power-On delay has been completed. Initializations required before the keyboard BAT is done are now in progress.
04h	The initializations required before the keyboard BAT are completed. Reading the keyboard SYS bit to check for soft reset or power-on next.
05h	The soft reset or power-on setting has been determined. Next, enabling the ROM and disabling shadow RAM and Cache Memory, if any.
06h	ROM is enabled. Calculating the ROM BIOS checksum and waiting for the keyboard controller input buffer to be free.
07h	The ROM BIOS checksum test passed and the keyboard controller input buffer is free. Issuing a BAT command to the keyboard controller next.
08h	A BAT command has been issued to the keyboard controller. Verifying the BAT command next.
09h	The keyboard controller BAT result was verified. A keyboard controller command byte is to be written next.
0Ah	A keyboard controller command byte code has been issued. Writing the command byte data next.
0Bh	The keyboard controller command byte has been written. Issuing the Pin 23, 24 blocking and unblocking command next.
0Ch	Pin 23, 24 of the keyboard controller has been blocked and unblocked. The keyboard controller NOP command is issued next.
0Dh	The keyboard controller NOP command processing is done. The CMOS RAM shutdown register test is performed next.
0Eh	The CMOS RAM shutdown register Read/Write test passed. Calculating the CMOS RAM checksum and updating the DIAG byte next.

Code	Description
0Fh	The CMOS RAM checksum calculation is done and the DIAG byte is written. CMOS RAM initialization begins next if CMOS RAM is to be initialized during every boot.
10h	CMOS RAM initialization (if any) is done. Next, the CMOS RAM status register is initialized for Date and Time.
11h	The CMOS RAM status register has been initialized. Disabling the DMA and interrupt controllers next.
12h	DMA controllers 1 and 2 and interrupt controllers 1 and 2 are disabled. Disabling the video display and initializing port B next.
13h	The video display is disabled and port B is initialized. Chipset initialization and auto memory detection are about to begin.
14h	Chipset initialization and auto memory detection are done. The 8254 Channel 2 timer test is about to start.
15h	The 8254 Channel 2 timer test is half-completed. The entire 8254 Channel 2 timer test is completed next.
16h	The entire 8254 Channel 2 timer test is done. The 8254 Channel 1 timer test is done next.
17h	The 8254 Channel 1 timer test is done. The 8254 Channel 0 timer test is completed next.
18h	The 8254 Channel 0 timer test is done. About to start memory refresh.
19h	Memory refresh has been started. The memory refresh test is performed next.
1Ah	The memory refresh line is toggling. Checking the 15 μsecond ON/OFF time next.
1Bh	The memory refresh test has been completed. The base 64 KB memory test is about to start.
20h	The base 64 KB memory test has been started. The address line test is to be done next.
21h	The address line test passed. Toggling parity next.
22h	The parity toggle has been completed. Performing a sequential data read/write test next.
23h	The base 64 KB sequential data read/write test passed. Performing any necessary system initialization before interrupt vector initialization.
24h	The system configuration required before vector initialization has been completed. Interrupt vector initialization is about to begin.
25h	Interrupt vector initialization is done. Reading the input port of the 8042 for the turbo switch setting (if any).
26h	The input port of the 8042 has been read. Initializing global data for the turbo switch.
27h	Global data initialization is done. Initialization after the interrupt vector initialization will be done next.
28h	Initialization after interrupt vector initialization is completed. Setting monochrome mode next.
29h	Monochrome mode is set. Setting color mode next.
2Ah	Color mode is set. Toggling parity before the optional Video ROM test next.
2Bh	Parity toggle completed. About to do any system initialization required before the video ROM check.

POST Checkpoint Codes, Continued

Code	Description
2Ch	Initialization before video ROM control is done. Looking for video ROM next. Control passed to video ROM next.
2Dh	The video ROM check is done. Next, do processing after the video ROM returns control.
2Eh	Finished processing after the video ROM had control. If an EGA or VGA adapter is not found, the display memory read/write test is next.
2Fh	No EGA or VGA adapter has been found. The display memory read/write test is about to begin.
30h	The display memory read/write test passed. About to look for retrace check.
31h	The display memory read/write test or retrace check failed. About to perform the alternate display memory read/write test.
32h	The alternate display memory read/write test passed. About to look for alternate display retrace checking.
33h	The video display check is completed. Verification of the display type with switch setting and the actual adapter card is next.
34h	Verification of the display adapter is done. The display mode is set next.
35h	The display mode has been set. The BIOS ROM data area is about to be checked.
36h	The BIOS ROM data area check is completed. Setting the cursor for the Power-On message next.
37h	Cursor setting for the Power-On message is done. Displaying the Power-On message next.
38h	The Power-On message has been displayed. Reading the new cursor position next.
39h	The new cursor position has been read and saved. Displaying the BIOS Identification String next.
3Ah	The BIOS Identification String has been displayed. Displaying the "Hit <DEL>..." message next.
3Bh	The "Hit <DEL>..." message has been displayed. The virtual mode memory test is about to start.
40h	Preparing the virtual mode test. Verifying from display memory next.
41h	Returned to POST after verifying from display memory. Preparing the descriptor tables next.
42h	The descriptor tables have been prepared. Entering virtual mode for the memory test next.
43h	Entered virtual mode. Enabling interrupts for diagnostics mode next.
44h	Interrupts are enabled if the diagnostics switch is on. Initializing data to check the memory wraparound at 0:0h next.
45h	Data has been initialized for the memory wraparound check. Checking for memory wraparound at 0:0h and finding the total system memory size next.
46h	The memory wraparound test has been done. The memory size calculation has been done. About to write memory test patterns.
47h	The memory test patterns were written to extended memory. Writing patterns in conventional memory (first 640 KB) next.
48h	The patterns to be tested were written to conventional memory. Finding the amount of memory below 1 MB next.

Code	Description
49h	The amount of memory below 1 MB was found and verified. Finding the amount of memory above 1 MB next.
4Ah	The amount of memory above 1 MB was found and verified. Performing the BIOS ROM data area check next.
4Bh	The BIOS ROM data area check is done. Checking the Del key status and clearing the memory below 1 MB for a soft reset next.
4Ch	The memory below 1 MB has been cleared via a soft reset. Clearing the memory above 1 MB next.
4Dh	The memory above 1 MB has been cleared via a soft reset. Saving the memory size next.
4Eh	The memory test has started. No soft reset was performed. About to display the first 64 KB memory test.
4Fh	The memory size display has started. This display is updated during the memory test. Running the sequential and random memory test next.
50h	The test of memory below 1 MB completed. Adjusting the memory size for relocation and shadowing next.
51h	The memory size has been adjusted for memory relocation above 1 MB and shadowing options. The test of memory above 1 MB is next.
52h	The test of memory above 1 MB has completed. Preparing for real mode next.
53h	The CPU registers have been saved, including the memory size. Entering real mode next.
54h	Shutdown was successful and the CPU is in real mode. Restoring the registers saved during preparation for shutdown next.
55h	The registers have been restored. Disabling the Gate A20 address line next.
56h	The Gate A20 address line was disabled successfully. Checking the BIOS ROM data area next.
57h	The BIOS ROM data area check is partially completed. Completing the BIOS ROM data area check next.
58h	The BIOS ROM data area check has completed. Clearing the "Hit Del" message next.
59h	The "Hit Del" message has been cleared. About to start the DMA and interrupt controller tests.
60h	The DMA page register test passed. About to verify from display memory.
61h	The display memory verification test is done. About to perform the DMA Controller 1 base register test.
62h	The DMA Controller 1 base register test passed. Performing the DMA Controller 2 base register test next.
63h	The DMA Controller 2 base register test passed. Performing the BIOS ROM data area check next.
64h	The BIOS ROM data area check is partially done. The BIOS ROM data area check is completed next.
65h	The BIOS ROM data area check is done. Programming DMA Controllers 1 and 2 next.
66h	DMA Controller 1 and 2 programming was completed. Initializing the 8259 interrupt controller next.
67h	The 8259 initialization is done. Starting the keyboard test next.

POST Checkpoint Codes, Continued

Code	Description
80h	The keyboard test has started. Issuing the keyboard reset command next and clearing the output buffer.
81h	The keyboard reset command completed successfully. Next, checking for stuck keys and issuing the interface test command if there was an error.
82h	The keyboard controller interface test is done. About to write a command byte and initialize the circular buffer.
83h	The command byte has been written and the global data initialization is done. About to check for locked keys.
84h	Lock key checking is done. About to check for a memory size mismatch with CMOS RAM data.
85h	The memory size check has been completed. About to display a soft error and check for password or bypass Setup.
86h	The password has been checked. About to do programming before Setup.
87h	The programming before Setup has been completed. Calling the BIOS Setup program next.
88h	Returned from the BIOS Setup program and cleared the screen. Programming after Setup.
89h	The programming after Setup is completed. Displaying the Power-On screen message next.
8Ah	The first screen message has been displayed. About to display the "Wait..." message.
8Bh	The "Wait..." message has been displayed. About to perform system and video BIOS shadowing.
8Ch	System and video BIOS shadowing was successful. About to perform Setup options programming after Standard CMOS Setup.
8Dh	The Setup options are programmed. The mouse check and initialization is done next.
8Eh	The mouse check and initialization is done. Checking the floppy disk next.
8Fh	The floppy disk check indicated that the floppy drive needs to be initialized. Floppy drive configuration is next.
90h	Floppy drive configuration has completed. The test for the presence of a hard disk drive is next.
91h	The hard disk presence test has completed. Hard disk configuration is next.
92h	Hard disk configuration has completed. Checking the BIOS ROM data area next.
93h	The BIOS ROM data area check was partially completed. The entire BIOS ROM data area check is completed next.
94h	The BIOS ROM data area check has fully completed. Setting the base and extended memory sizes next.
95h	The memory size has been adjusted because of mouse support and hard disk type 47. Verifying from display memory next.
96h	Returned after verifying from display memory. Initializing before C8000h adaptor ROM control next.
97h	Any initialization that had to be done before control is passed to the adaptor ROM at C8000h option has been completed. The adaptor ROM check and control test is next. Relinquishing control to adaptor ROM at C8000h.

Code	Description
98h	The adaptor ROM control test has been done. About to do required processing after the adaptor ROM returns control.
99h	Any initialization for the option ROM test was done. Configuring the timer data area and the parallel printer base address next.
9Ah	Set the timer data area and the parallel printer base address. Setting the RS-232 base address next.
9Bh	Set the RS-232 base address. Initializing before the coprocessor test next.
9Ch	The required initialization before the coprocessor test has been done. Initializing the coprocesssor next.
9Dh	The coprocessor has been initialized. Performing any initialization after the coprocessor test next.
9Eh	Initialization after the coprocessor test is completed. Checking the Extended Keyboard, Keyboard ID and Num Lock keyboard settings next.
9Fh	The Extended Keyboard flags have been checked, the Keyboard ID flag set, and Num Lock is set On or Off as specified. The Keyboard ID command is issued next.
A0h	The Keyboard ID command has been issued. The Keyboard ID flag reset is next.
A1h	The Keyboard ID flag reset has been done. The cache memory tests follow.
A2h	The cache memory test has been done. Displaying any soft errors next.
A3h	Soft error display is complete. Setting the keyboard typematic rate next.
A4h	The keyboard typematic rate is set. Programming the memory wait states next.
A5h	Memory wait states programming is done. The screen is cleared next.
A6h	The screen has been cleared. Enabling parity and NMIs next.
A7h	NMIs and parity have been enabled. Performing any initialization required before passing control to the adaptor ROM at E0000h next.
A8h	Initialization before E0000h adaptor ROM control has been done. The E0000h adaptor ROM gets control next.
A9h	Returned from E0000h adaptor ROM control. Performing any initialization required after E0000h adaptor ROM control next.
A0h	Initialization after E0000h adaptor ROM control has been done. Displaying the system configuration next.
00h	The system configuration has been displayed. Passing control to INT 19h Bootstrap Loader next.

EISA POST Checkpoint Codes

The EISA AMIBIOS can generate additional POST codes:

Code	Description
27h	This is an ISA BIOS checkpoint, but in an EISA BIOS, the following EISA checkpoints execute after it, not checkpoint 28h, executed next in an ISA BIOS.
F0h	Initialization after the interrupt vector is completed. Initializing the EISA slots next.
F1h	EISA slot initialization is completed. Setting up the extended NMI test next.
F2h	Configuration for the extended NMI test has been done. Testing the extended NMI next. After this checkpoint, EISA BIOS POST returns to ISA BIOS POST checkpoint 28h.
28h	This is a standard ISA BIOS checkpoint. The extended NMI test has completed. Setting monochrome mode next.
38h	This is an ISA BIOS checkpoint that is modified in the EISA BIOS. The Power-On message display has completed. Reading the new cursor position next. In an ISA BIOS, the next checkpoint is 39h, but in the EISA BIOS, it is F3h.
F3h	The new cursor position has been read and saved. Displaying any errors that occurred during slot initialization next. The EISA BIOS goes to 39h next, an ISA checkpoint.
67h	This is an ISA BIOS checkpoint. The ISA BIOS goes to 80h next, but the EISA BIOS goes to F4h. 8259 initialization has completed. Programming the 8259 mask registers next.
F4h	The 8259 mask register programming has completed. About to enable the extended NMI. EISA BIOS POST returns to a standard ISA BIOS checkpoint, 80h.
80h	This is a standard ISA BIOS checkpoint that EISA BIOS POST returns to. The process of enabling extended source is done. About to start the keyboard test. Clearing the output buffer. Checking for stuck keys. About to issue the keyboard reset command.
8Dh	This is a standard ISA BIOS checkpoint. The ISA BIOS goes to 8Eh next but EISA BIOS POST goes to F5h.
F5h	The Wait message is displayed. About to program the interrupt controller for edge or level sensitivity for a PS/2-type mouse. About to start the mouse check and mouse initialization. EISA BIOS POST now returns to the standard ISA BIOS checkpoint, 8Bh.
8Eh	The mouse check and initialization has completed. About to perform system and video BIOS shadowing.

Chapter 9

Using Interrupts

Overview

The interrupt is the method used in ISA and EISA systems to access BIOS services. Both software programs and peripheral devices use interrupts:

- hardware peripheral devices use interrupts to report an event or request that an action be performed.
- software programs use the INT mnemonic to request certain action from a peripheral device.

What an Interrupt Does

An interrupt essentially stops other microprocessor operations. The number specified with software interrupts instructs the BIOS to perform an operation using a specific peripheral device.

Requesting a Software Interrupt

Invoke a software interrupt from any assembler language program. Place the interrupt number after the assembler mnemonic INT. The microprocessor executes the instructions identified by the interrupt number when it finds an INT mnemonic. These instructions make up an interrupt service routine (or device service routine).

cont'd

Microprocessor Interrupt Handling

When the microprocessor receives an interrupt signal (if it is a software interrupt, it also contains an interrupt number), it usually stops all other activity and activates a subroutine stored in system memory. These subroutines are either interrupt service routines (ISRs) or device service routines (DSRs). The ISR or DSR is keyed to the interrupt number (either software or hardware). The ISR or DSR contains the code that executes the task or routine requested by the INT mnemonic and interrupt number.

Using Registers to Further Define the Interrupt

Before a software interrupt is invoked, special prespecified codes and parameters may have to be placed in some of the microprocessor's registers (AX, BX, CX, DX) to further specify the operation that the interrupt routine is to perform. The interrupt routine output is usually returned in the processor registers or flags.

Types of Interrupts

Interrupt Type	Description	Range
Processor	Generated or processed by the microprocessor.	00h – 04h
Hardware	Generated by hardware devices. Eight of these are hardwired to either the microprocessor or the motherboard. IRQs 2, 8, 9, and 11 – 15 are reserved.	08h – 0Fh 70h – 77h
Software	Handled by the BIOS. 05h, 10h through 1Ah, and 40h, 41h, 42h, 43h, 46h, and 4Ah are reserved. The rest are available for revectoring to programmer-written routines.	05h – 07h 10h – 1Fh 40h – 5Fh
DOS	Only available when DOS is active. INTs 20h through 3Fh are reserved for DOS.	20h – 3Fh
BASIC	Reserved	80h – BFh
User	INT 67h is used for EMS. All others available for revectoring to user-written routines.	60h – 6Fh

Interrupts Not Discussed

In this book, we are concerned with only the following types of interrupts. And only certain software interrupts are discussed in detail.

Interrupt Type	Description
Processor (Logical)	Generated by the microprocessor as a result of an unusual program result, such as an attempt to divide by zero or a nonmaskable interrupt.
Hardware	Generated by the computer circuits and managed by the Programmable Interrupt Controller (the Intel 8259).
Software	BIOS interrupts invoked by a systems software or applications software program or used internally by the BIOS.

Processor Interrupts

These interrupts are invoked by the processor as a result of an unusual program result. Interrupts 00h – 04h are reserved for the processor. For example, INT 00h means that a program tried to divide a value by 0. The processor generates this interrupt and halts processing.

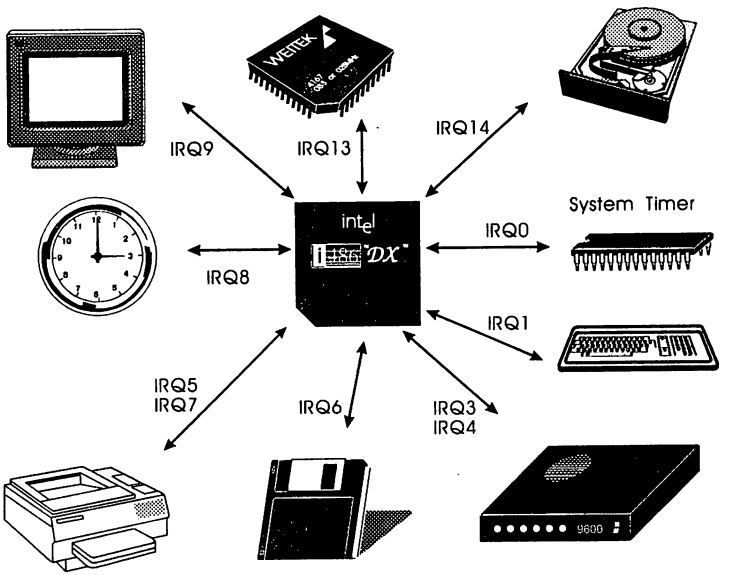
Hardware Interrupts

When a hardware device or program needs the processor, it sends a signal or instruction to the processor, requesting a certain service or task. After the interrupt handler performs its task, system activities continue at the same point that the interrupt occurred. Hardware interrupts are invoked by peripheral devices by setting an assigned Interrupt Request (IRQ) line. In EISA systems, IRQs can be assigned via the EISA Configuration Utility (ECU).

For example, every time a key is pressed on the keyboard, the keyboard hardware generates a hardware interrupt (IRQ). Hardware interrupts are vectored to Interrupt Service Routines (ISRs) that generally reside in the BIOS. INTs 08h – 0Fh and 70h – 77h are reserved by the BIOS for hardware interrupts.

cont'd

Hardware Interrupts, cont'd



Hardware Interrupt Priorities

The hardware interrupt priority is: NMI, IRQ0, IRQ1, IRQ2, IRQ8, IRQ9, IRQ10, IRQ11, IRQ12, IRQ13, IRQ14, IRQ15, IRQ3, IRQ4, IRQ5, IRQ6, and IRQ7.

Hardware Interrupts, cont'd

The hardware interrupts that actually interface with the BIOS are IRQs 0, 1, 3, 4, 5, 6, 7, 13 and 14.

System Interrupt	Interrupt Request Line	Description
08h	IRQ0	System Timer (Timer Channel 0 output)
09h	IRQ1	Keyboard Controller Output Buffer Full Interrupt
0Ah	IRQ2	Cascade from Second Programmable Interrupt Controller
0Bh	IRQ3	COM2 or COM 4
0Ch	IRQ4	COM1 or COM 3
0Dh	IRQ5	LPT2 Parallel Port
0Eh	IRQ6	Floppy disk drives
0Fh	IRQ7	LPT1, Parallel Port
70h	IRQ8	Real Time Clock Interrupt
71h	IRQ9	Vertical Retrace Interrupt from Video Display Adapter
72h	IRQ10	Reserved
73h	IRQ11	Reserved
74h	IRQ12	Reserved in ISA. Mouse controller in EISA.
75h	IRQ13	Math coprocessor interrupt on any error
76h	IRQ14	AT Hard disk drive controller
77h	IRQ15	Usually reserved. Used in a Power Management AMIBIOS for software suspend request.

cont'd

Software Interrupts

Software Interrupts are invoked via the x86 assembly language INT mnemonic. Most software interrupts are vectored to device service routines (DSRs) in the ROM BIOS or operating system.

Exceptions:

- BIOS software interrupts 1Dh, 1Eh, 1Fh, 41h, 42h, 43h, and 46h do not service a specific device but instead return ROM-resident hardware parameter tables.
- INTs 20h – 3Fh are software interrupts that are only to be used by the operating system (by convention).
- INTs 05h, 10h – 1Ah, 1Dh – 1Fh, 40h, 41h, 42h, 43h, 44h, and 46h can only be used by the system BIOS.

Interrupt Numbers

Every interrupt in the Intel x86 architecture is assigned a unique interrupt number, whether it comes from the microprocessor, hardware, or software.

Restricted Interrupt Numbers

To maintain IBM compatibility, certain ranges of interrupts are reserved for special use.

INTs 60 – 67h are used for user software interrupts.

INTs 20h – 3Fh is reserved for the operating system.

Interrupt Numbers and Interrupt Vectors

Each interrupt number is associated with a specific interrupt vector. An interrupt vector is in the doubleword segment:offset format of the routine assigned to an interrupt number. Interrupt vectors are stored in a table beginning at address 0:0000h. The vector for INT 00h is stored at address 0:0h through 0:03h. The vector for INT 02h is stored at 0:08h to 0:0Bh, the vector for INT 03h is stored at 0:0Ch to 0:0Fh, and so on.

POST (see page 90) writes the interrupt vectors to low memory and initializes the vector address of all interrupts used by the BIOS. When the operating system boots, it initializes all operating system-specific interrupt vectors.

Applications programs that add their own interrupt routines are responsible for initializing the interrupt vectors for their own interrupts.

Interrupt Vector Table

The originator of the interrupt does not need to know the memory address of the required interrupt handler. It only needs to know the interrupt number. The interrupt number points to the interrupt vector table, a table in low memory that contains the segmented address of the interrupt handling subroutine.

The interrupt handler's address is called the interrupt vector and the table is the interrupt vector table.

The interrupt vector table is normally supervised by the BIOS and DOS. When new interrupt handling routines are created, the programmer either uses an existing interrupt number and vector, or assigns a new one. See page 116 for more information about interrupt vectors.

Interrupts Return to Next Instruction after Executing

Interrupts automatically save the contents of CS and IP on the stack, so the system can return to where it was after the interrupt is processed. The IRET instruction performs this function. IRET also restores the flags, CS, and EIP.

The interrupt process also saves the flag register on the stack and clears the interrupt flag (IF), temporarily preventing additional interrupts.

It is a convention when writing interrupt routines to turn interrupts back on as soon as possible.

Available Interrupts

Software interrupts 40h through 6Fh are defined, used, and possibly made available through the operating system or a memory manager. A programmer can use INTs 60h – 6Fh (except 67h) to add a new BIOS service.

Nonmaskable Interrupt

The Nonmaskable interrupt (NMI) is a special case. It is generated by hardware devices and is used to demand immediate attention from the microprocessor. It often signals an emergency, such as a low voltage condition or a memory error.

The BIOS generates INT 02h to handle NMIs. See page 120 for additional information about Nonmaskable Interrupts and INT 02h.

Unexpected Interrupt Handler

The AMIBIOS initializes unused interrupt vectors to the BIOS unexpected interrupt handler.

The unexpected interrupt handler routine processes all interrupts that are either:

- user-defined processes (INT 1Ch and INT 4Ah, for example), or
- not meaningful to the BIOS (INT 73h or INT 7Fh, for example).

If an unexpected interrupt occurs, the AMIBIOS either:

- returns to the caller with CF set to 1 and all registers preserved, or
- revectors the interrupt to a caller-supplied interrupt processing routine.

BIOS Register Conventions

Input to BIOS Interrupt Function Calls

The Intel x86 processor registers are used in the following manner to input values to a BIOS device service routine.

Register	Conventional Use
CS and IP	Automatically loaded, reserved, and restored as part of the interrupt process.
DS and ES	Preserved by the interrupt services.
SS	Not changed by the interrupt services.
SP	Preserved because, by coding convention, all BIOS device service routines leave the stack clean before returning.
AX	May be changed by the BIOS service. You cannot be sure that the contents of this register are the same after returning from a BIOS service.
BX	May be changed by the BIOS service. You cannot be sure that the contents of this register are the same after returning from a BIOS service.
CX	May be changed by the BIOS service. You cannot be sure that the contents of this register are the same after returning from a BIOS service.
DX	May be changed by the BIOS service. You cannot be sure that the contents of this register are the same after returning from a BIOS service.
SI and DI	May be changed by the BIOS service.

Output from BIOS Interrupt Function Calls

The Intel x86 processor registers are used in the following manner for output values from a BIOS device service routine.

Register	Conventional Use
AH	Used to return error information.
AL	Sometimes used to return error information.
CF	The Carry Flag (CF) is set if an error occurred when the Interrupt request was processed.
FLAG Bits	All flag bits might be changed by the BIOS service. You cannot depend on any bit to be the same.

Chapter 10

BIOS Interrupts

The system interrupts supported in AMIBIOS are routines used to access I/O devices without directly accessing the hardware. The system interrupts supported by the AMIBIOS are described in this chapter.

How Interrupts are Used

Interrupts are mainly associated with specific peripheral devices. Most interrupts have functions selected by placing a value in AH. The functions specify the activity to be performed by the interrupt service. Functions are described under each interrupt heading. The functions that the interrupt performs are described and the required input and output values are listed.

Types of Interrupts

The BIOS interrupts are all software interrupts. See page 107 for a complete description of the types of interrupts.

Interrupt Vector Table

Each time the BIOS initializes the system, it creates a table of interrupt vectors (pointers to the location (address) of the interrupt service routine entry point) at 0:0000h as follows:

Byte	Description
First	Least significant byte of offset (OLSB).
Second	Most significant byte of offset (OMSB).
Third	Least significant byte of segment (SLSB).
Fourth	Most significant byte of segment (SMSB).

cont'd

Interrupt Vector Table, Continued

Vector Example

If the four-byte entry for an interrupt is stored in the interrupt vector table as 7D EA 00 F0, the interrupt entry point address is F000:EA7Dh.

Using the Interrupt Vector Table

By replacing the existing vector in an Interrupt Vector table entry with a pointer to your own BIOS routine, you can easily add a new BIOS service or replace an existing service. An entry for each BIOS interrupt number from 00h through BFh appears in this table. Counting in hex by fours, you can easily determine the address of the interrupt vector table to be added or replaced.

For example, the interrupt vector table entry for INT 10h is at 0:0040h (4 times 10h). The interrupt vector table entry for INT 47h (an interrupt that is available for use by end users), is 0:011Ch.

BIOS Interrupt Summary

INT Code	Type	Function	Turn to
00h	Processor	Divide by Zero	Page 119
01h	Processor	Single Step	Page 120
02h	Processor	Nonmaskable interrupt (NMI)	Page 120
03h	Processor	Breakpoint	Page 121
04h	Processor	Arithmetic Overflow	Page 122
05h	Software	Print Screen	Page 122
06h	Processor	Invalid Op Code	Page 122
07h	Processor	Coprocessor Not Available	Page 122
08h	Hardware	Timer Tick	Page 124
09h	Hardware	Keyboard Device	Page 125
0Ah	Hardware	IRQ2 Cascade from Second Interrupt Controller	Page 127
0Bh	Hardware	Serial Port (COM2)	Page 127
0Ch	Hardware	Serial Port (COM1)	Page 127
0Dh	Hardware	Parallel Printer (LPT2)	Page 127
0Eh	Hardware	IRQ6 Floppy Disk	Page 127
0Fh	Hardware	IRQ7 Parallel Printer (LPT1)	Page 127

Interrupt Summary, Continued

INT Code	Type	Function	Turn to
10h	Software	Video Service	Page 128
11h	Software	Equipment List Service	Page 139
12h	Software	Memory Size Service	Page 139
13h	Software	Hard Disk Service	Page 140
13h	Software	Floppy Disk Service	Page 154
14h	Software	Serial Communications Service	Page 165
15h	Software	Systems Services	Page 178
16h	Software	Keyboard Service	Page 240
17h	Software	Parallel Printer Service	Page 249
18h	Software	ROM Basic	Page 251
19h	Software	Bootstrap Loader	Page 252
1Ah	Software	Time of Day Service	Page 260
1Bh	Software	<Ctrl><Break>	Page 332
1Ch	Software	User Timer Tick Service	Page 332
1Dh	Software	Video Control Parameter Table	Page 333
1Eh	Software	Floppy Disk Base Table	Page 333
1Fh	Software	Video Graphics Table	Page 333
20h – 3Fh	Software	DOS interrupts	*
40h	Software	Floppy Disk Revector	*
41h	Software	Hard Disk C: Parameter Table	*
42h	Software	EGA Default Video Driver	*
43h	Software	Video Graphics Characters	*
44h	Software	Novell NetWare API	*
46h	Software	Hard Disk D: Parameter Table	*
45h, 47h–49h	Software	Available	*
4Ah	Software	User Alarm	*
4Bh – 63h			
4Bh–63h	Software	Available	*
64h	Software	IPX (Novell NetWare)	
65h – 66h	Software	Available	
67h	Software	EMS	*
68h – 6Fh	Software	Available	
70h	Hardware	Real Time Clock	Page 334
71h	Hardware	Redirect Interrupt Cascade	*
72h – 74h	Hardware	Reserved. Do not use.	*
75h	Hardware	Math Coprocessor Exception	Page 335
76h	Hardware	Hard Disk	Page 335
77h	Hardware	Suspend Request	Page 336
78h – 79h	Software	Available	*
7Ah	Software	Novell NetWare API	*
78h – BFh	Software	Available	*

BIOS Stack Area

The AMIBIOS uses the location at 30:0000h – 30:00FFh as a stack area. This area (the BIOS Stack Area) is used primarily for BIOS calculations and temporary storage.

Following the interrupt vector entry format shown above, the addresses for INTs C0h through FFh should occur in this space. For this reason, INTs C0h through FFh are not supported in the AMIBIOS.

INT 00h through INT 07h

The first eight interrupts (00h through 07h) are called by the processor directly, but they can also be called via any software program using the INT instruction. See the INT 05h example below. All processors in the Intel x86 family handle the INT mnemonic.

INT 00h Divide by Zero

Input: None

Output: None

Description:

INT 00h is a logical or processor interrupt. INT 00h is generated by the microprocessor to handle any division operation that has a denominator value of zero. The exact behavior is dependent on the operating system or application program in use when the interrupt occurs. Most programs display an error message, such as "Divide By Zero" and then terminate.

INT 01h Single Stepping

Input: Trap bit = 1

Output: None

Description:

INT 01h is a logical or processor interrupt. INT 01h traces the execution of each instruction in a software program. Most debugging utility programs use this interrupt.

INT 02h Nonmaskable Interrupt (NMI)

Input: None

Output: None

Description:

An NMI is a hardware interrupt. The BIOS generates INT 02h, an interrupt service routine that handles NMIs in response to a hardware NMI. The hardware NMI is used primarily to halt the system when memory errors occur. The status bits (I/O port 61h) indicate whether the NMI was caused by an memory parity check or I/O check. You can prevent the execution of all software interrupts by invoking CLI, with the exception of INT 02h, which handles NMIs. The NMI cannot be masked by CLI, but NMIs can be turned off, as described on the next page.

When the operating system boots the system, it resets the interrupt vector that corresponds to the NMI to its own routine. When an error that causes an NMI occurs, the operating system NMI routine calls the BIOS INT 02h NMI handling routine. The BIOS NMI handling routine displays an error message that describes the type of hardware error that caused the NMI.

INT 02h Nonmaskable Interrupt (NMI)

Disabling NMIs

The NMI is disabled by writing to I/O port 70h with data bit 7 set and enabled by writing to port 70h with data bit 7 reset.

How NMIs Occur

- an onboard dynamic RAM parity failure,
- a 32-bit adapter card memory parity failure,
- an error reported by the I/O channel adapter card through the I/O channel check (-IOCHCK) signal,
- a bus timeout on an EISA slot,
- when a program sets bit 7 of I/O Port 462h (EISA only),
- a fail-safe timer NMI (EISA only), or
- when an EISA card is enabled or disabled.

INT 03h Breakpoint

Input: None

Output: None

Description:

INT 03h is a logical or processor interrupt. It provides a single-byte instruction (CCh) that halts the execution of a program so the programmer can evaluate the microprocessor registers and other areas of memory. INT 03h is useful in debugging. It is called by many commercial debugging programs.

INT 04h Overflow Error

Input: Overflow bit of FLAGS register = 1

Output: None

Description:

INT 04h is a logical or processor interrupt. When a numeric overflow occurs following a mathematical operation, the Overflow bit in the FLAGS register is set. The INTO instruction calls INT 04h when executed afterwards. If INT 04h is invoked, the Overflow bit is not read. INT 04h is not used often, so most operating systems set it to an IRET.

INT 05h Print Screen

Input: None

Output: None

Description:

INT 05h is a software interrupt. The system dumps the contents of the screen to a printer attached to the system when <Print Screen> is pressed. By using the INT instruction, programmers can also accomplish the same task. INT 05h only works in text modes. It does not dump graphics screens.

INT 06h Invalid Op Code

Input: None

Output: None

Description: INT 06h is a logical or processor interrupt that is called after an Invalid Op Code Exception Error is generated by the processor. The AMIBIOS allows you to replace the interrupt vector table entry for INT 06h with a routine.

INT 07h Coprocessor Not Available

Input: None

Output: None

Description:

INT 07h is a logical or processor interrupt. INT 07h is called by the processor when the emulation bit (EM) in the CPU control register is set and an ESC instruction is encountered.

Programs that use coprocessor emulation can trap this interrupt and provide another routine to be executed when this interrupt occurs.

Interrupts 08h Through 0Fh

INTs 08h – 0Fh are generated by the interrupt controller and correspond to IRQs 0 – 7. They are vectors to handle IRQs 08h – 0Fh. Since these interrupts are not generated by the CPU directly, the interrupt controller sets the corresponding IRQ line to request that the CPU generate the appropriate interrupt (08h – 0Fh) when invoked.

IRQs have a fixed priority scheme: NMI, 0, 1, 2, 8, 9, 10, 11, 12, 13, 14, 15, 3, 4, 5, 6, and 7. The interrupt controller determines which interrupt request (IRQ) has the highest priority and then forwards that request to the CPU. All ISA and EISA systems have two interrupt controllers and sixteen IRQ lines. The IRQ 2 line cascades the second interrupt controller to the first interrupt controller. IRQ 2 is not used by any other devices and gives IRQs 8 – 15 a higher priority than IRQs 3 – 7.

INT 08h Timer Interrupt (IRQ0)

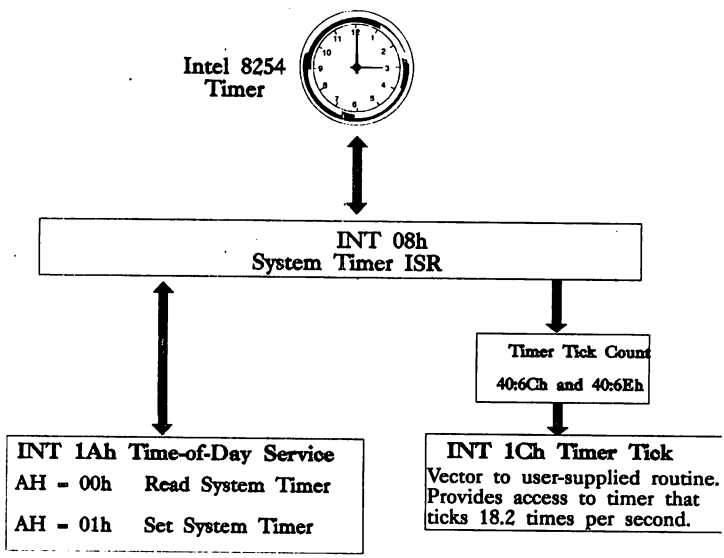
Input: None

Output: None

Description:

INT 08h can be used to measure time increments independent of the system clock frequency. INT 08h is called approximately 18.2 times per second. INT 08h increments the system time count at location 40:6Ch through 40:6Eh every time it is called. If the system time count (40:6Ch) exceeds 24 hours, the Timer Overflow Flag (40:70h) is set, the date is incremented by the BIOS, and the system time count is reset to 0. INT 08h also decrements the floppy disk motor count at 40:40h. When the count reaches 0, INT 08h turns the floppy drive motor off.

INT 08h also issues an INT 1Ch Timer Tick interrupt every time it is called. Programmers can revector INT 1Ch to their own routines and use the clock feature for timed events. The following graphic illustrates how the system timer is used with the BIOS.

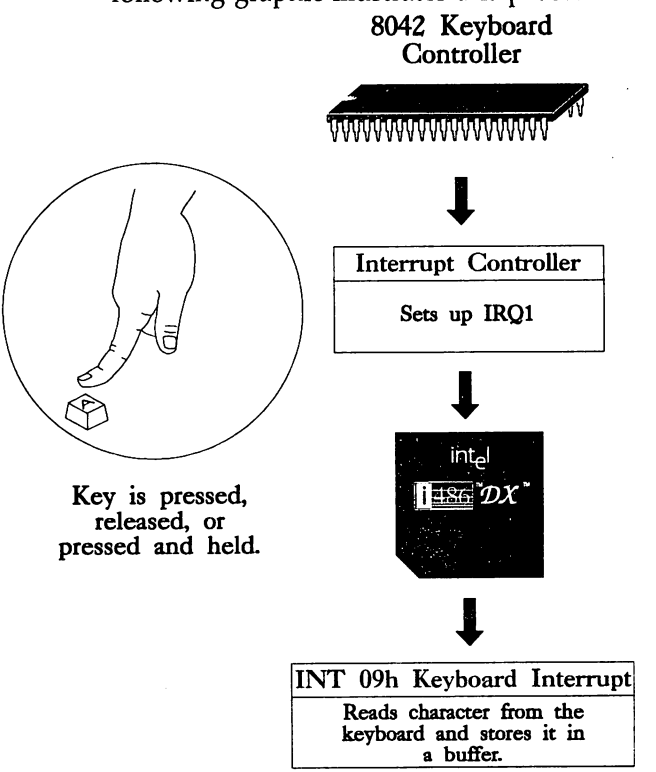


INT 09h Keyboard Interrupt (IRQ1)

Input: None

Output: None

Description: ISA systems generally use 8042 processors to control the keyboard and keyboard registers. If a key is pressed, released, or pressed and held, the 8042 issues an interrupt signal to the interrupt controller. The interrupt controller sets IRQ line 1 so the microprocessor can issue an interrupt. The BIOS INT 09h routine is then called. INT 09h reads the character from the keyboard and stores it in a buffer. The following graphic illustrates this process.



Keyboard Key Processing

When...	the BIOS...
The BIOS receives the ASCII Scan code for any key.	The ASCII scan code is read from port 60h and is placed in the 32-byte keyboard buffer (40:1Eh).
<Ctrl>, <Shift>, or <Alt> is pressed.	The Keyboard Control flags (40:17h and 40:18h) and the Keyboard Mode State and Type flag (40:96h) are updated.
the <Ctrl><Alt><Del> keychord is pressed.	The reset flag (40:72h) is set to 1234h and the routine jumps to the POST tests, followed by a system reboot. This allows POST to skip the memory test it usually performs.
<Pause> is pressed.	The system enters a wait loop until a valid ASCII character key is pressed.
<Print Screen> is pressed.	The BIOS issues an INT 05h call.
The <Ctrl><Break> keychord is pressed.	The BIOS issues an INT 1Bh call.
<SysReq> is pressed.	INT 15h Function 85h is called. This routine is initialized by the BIOS to do nothing. Other software programs can trap this interrupt and provide an interrupt handler for <SysReq>.

Testing for any Keystroke

After reading the scan code from I/O port 60h, an INT 15h Function 4Fh is issued. This function is initialized by the BIOS to do nothing. Other software programs can trap this interrupt and provide an interrupt handler.

INTs 0Ah Through 0Fh Miscellaneous Interrupts

Input: None

Output: None

Description:

These interrupts are defined by other external peripheral devices attached to the system.

In ISA systems, INTs 0Ah – 0Fh are often attached to the IRQ lines shown in the following table. When these interrupts are invoked, the corresponding IRQ line is enabled, alerting the microprocessor that the attached device must be serviced.

Interrupt	Common Hardware Interface
0Ah	IRQ2 Cascade to second interrupt controller.
0Bh	IRQ3 Serial port 2 or 4.
0Ch	IRQ4 Serial port 1 or 3.
0Dh	IRQ5 Parallel port 2.
0Eh	IRQ6 Floppy disk drive.
0Fh	IRQ7 Parallel port 1.

Some of the above IRQ assignments are not fixed (3, 4, 5, 7) they can vary from system to system. AMIDiag Version 4.0 or later lists the hardware interrupt assignments for your system.

EISA systems are even more flexible. The EISA Configuration Utility (ECU) allows you to assign IRQs in any order to any EISA adapter card, with few restrictions.

The above table is probably not accurate for most EISA systems.

INT 10h Video Service

INT 10h, the video interrupt routine, has seventeen functions supported by the system BIOS. The system BIOS only supports two video display adapters: monochrome display adapter (MDA) and color graphics adapter (CGA). The BIOS support for EGA, VGA, and XGA display adapters is provided on the video adapter. If EGA is used, INT 42h points to the BIOS Video Service Routine. Both the EGA and VGA video BIOS reside at C000h.

INT 10h Functions

Function	Title
00h	Set Video Mode
01h	Set Cursor Type
02h	Set Cursor Position
03h	Return Cursor Position
04h	Return Light Pen Position
05h	Set Current Video Page
06h	Scroll Text Upward
07h	Scroll Text Downward
08h	Return Character or Attribute
09h	Write Character or Attribute
0Ah	Write Character
0Bh	Set Color Palette Subfunction BH = 00h Set Palette Subfunction BH = 01h Set Color Palette
0Ch	Write Graphic Pixel
0Dh	Read Graphic Pixel
0Eh	Write Character
0Fh	Return Video Display Mode
13h	Write Character String

Note that the IBM BIOS destroys the contents of AX, BX, SI, DI, and BP after all INT 10h function calls, but the AMIBIOS does not.

cont'd

Function 00h Set Video Mode

Input: AH = 00h
AL = Video Mode
00h 40 x 25 text mode, monochrome with CGA card
01h 40 x 25 text mode, color with CGA card
02h 80 x 25 text mode, monochrome with CGA card
03h 80 x 25 text mode, color with CGA card
04h 320 x 200 four-color graphics, with CGA card
05h 320 x 200 monochrome, with CGA card
06h 640 x 200 monochrome, with CGA card
07h 80 x 25 monochrome, with monochrome card

Output: No registers set.

Description:

Function 00h sets the video mode. Only the video modes supported in the MDA and CGA video standards are supported by the system BIOS. This function programs the CRTC, selects a default color palette, and clears the video buffer if the proper flag is set in the save area.

Video Modes

Mode	Adapter	Resolution	Type	Colors	Lines and Rows	Array	Max. Pages	Buffer
0, 1	CGA	320x200	Text	16/256K	40x25	8x8	8	B800h
2, 3	CGA	642x200	Text	16/256K	80x25	8x8	4	B800h
4, 5	CGA	320x200	Graphics	4/256K	40x25	8x8	1	B800h
6	CGA	640x200	Graphics	2/256K	80x25	8x8	1	B800h
7	MDA	720x350	Text	None	80x25	9x14	1	B000h

Function 01h Set Cursor Type

Input: AH = 01h
CH = Starting cursor line (bits 4–0). If 20h, the cursor is disabled.
CL = Ending Cursor Line (bits 4 – 0)

Output: No registers set. 40:60h is updated.

Description:

Function 01h sets the cursor type. If using MDA, valid values are 0–13. Using CGA, valid values are 0–7. If CH is 20h, the cursor is disabled. This function programs the CRTC to display the text cursor type. Only one cursor type is maintained for each video page. The BIOS default values are:

Monochrome (MDA)	CH	=	Starting Cursor Line	11
	CL	=	Ending Cursor Line	12
Color (CGA)	CH	=	Starting Cursor Line	6
	CL	=	Ending Cursor Line	7

Function 02h Set Cursor Position

Input: AH = 02h
BH = Video Page Number
DH = Line on Screen
DL = Column on Screen

Output: No registers set. 40:50h is updated.

Description:

Function 02h positions the cursor in a video page. Valid values for DH are 0 – 24. Valid values for DL are 0 – 39 in 40-column mode and 0 – 79 in 80-column mode. If the current video page number is in BH, the BIOS programs the CRTC to update the current cursor position on the specified page.

cont'd

Function 03h Return Cursor Position

Input: AH = 03h
BH = Video Page Number

Output: CH = Beginning Line of the Blinking Cursor
CL = Ending Line of the Blinking Cursor
DH = Line on Screen
DL = Column on Screen

Description:

Function 03h reads the current cursor position on the specified video page. This function is used only in text mode.

Function 04h Return Light Pen Position

Input: AH = 04h

Output: AH = 00h Position is Unreadable
01h Position is Readable
04h Light pen disabled or no valid light pen address.
BX = Column on Graphic Screen (Pixel)
CH = Line on Graphic Screen (Raster Line)
CL = Raster line if resolution of mode is less than 200 lines.
DH = Line on Text Screen
DL = Column on Text Screen

Description:

This function can be used to determine the position of the light pen. This routine is not accurate in graphics mode and is ineffective when used on monochrome monitors with long image-retention phosphors. The raster line value is always a multiple of two, and depending on graphic screen size, the pixel value is a multiple of four (in 320 x 200 mode) or a multiple of eight (in a 640 x 200 mode).

Function 05h Set Current Video Page

Input: AH = 05h
AL = Video Page Number

Output: None

Description:

Function 05h sets a new video page or selects the portion of the video buffer to be displayed. This function is ignored if CGA is used because CGA uses the entire 16K video buffer. The BIOS programs the CRTC Start Address Registers in video modes 0 – 3. The BIOS maintains the current cursor location in up to eight video pages at 40:50h. When a new video page is selected, the BIOS moves the cursor to the position the cursor was at the last time the requested video page was displayed.

Function 06h Scroll Text Upward

Input:
AH = 06h
AL = Number of scrolling lines
BH = Color or attribute for scrolling lines
CH = Line Number of upper left window corner
CL = Column number of upper left window corner
DH = Line number of lower right window corner
DL = Column number of lower right window corner

Output: None

Description:

Function 06h creates a window defined by values specified in CH, CL, DH, and DL. It scrolls the number of window lines upward through the window. The number of lines is defined in AL, and the color or attribute of the new lines is in BH. If AL is set to 00h, the window is cleared.

cont'd

Function 07h Scroll Text Downward

Input:
- AH = 07h
- AL = Number of Scrolling Lines
- BH = Color or Attribute of Scrolling Lines
- CH = Line Number of Upper Left Corner of Window
- CL = Column Number of Upper Left Corner of Window
- DH = Line Number of Lower Right Corner of Window
- DL = Column Number of Lower Right Corner of Window

Output: None

Description: Function 07h creates a window (defined by values in CH, CL, DH, and DL) and scrolls a number of window lines downward through the window. The number of lines to be scrolled is in AL, and the color or attribute of the new lines is in BH. If AL is set to 00h, the window is cleared.

Function 08h Return Character or Attribute

Input:
- AH = 08h
- BH = Video Page Number

Output:
- AH = Color or Attribute of Character
- AL = ASCII Code of Character

Description: Function 08h gets the ASCII code of the character at the current cursor location on the video page specified in BH. The function returns the character attribute or color in AH.

Function 09h Write Character or Attribute

Input:

AH	=	09h
AL	=	ASCII Code of Character to be Written
BH	=	Video Page Number (or background pixel value if in 320x200x256 color mode)
BL	=	Attribute or Color of Character (or background pixel value in graphics mode)
CX	=	Number of Repetitions

Output: None

Description:

INT 10h Function 09h writes a character(s) to the current cursor position on the video page specified in BH. You can also specify the character attribute or color and the number of times the character is to be written. The new cursor position is not changed.

Function 0Ah Write Character

Input:

AH	=	0Ah
AL	=	ASCII Code of Character to be Written
BH	=	Video Page Number (background pixel value if in 320x200x256 color mode)
BL	=	Foreground pixel value (in graphics mode only)
CX	=	Number of repetitions

Output: None

Description:

INT 10h Function 0Ah writes a character(s) to the current cursor position on the video page specified in BH. You can also specify the number of times the character is to be written. The new cursor position is not changed.

---

cont'd

Function 0Bh Subfunction 00h Set Palette

Input: AH = 0Bh
BH = 00h
BL = Screen Border and Background Color

Output: No registers set. 40:66h is updated.

Description:

INT 10h Function 0Bh subfunction 00h sets the screen background and border color. If the system is running in text mode, only the screen border color is defined. If the system is running in graphics mode, both the background and the screen border colors are defined. Use INT 10h Function 10h instead of this function if the system is using EGA or VGA.

Function 0Bh Subfunction 01h Set Color Palette

Input: AH = 0Bh
BH = 01h
BL = Number of Color Palette

Output: No registers set. 40:66h is updated.

Description:

Function 0Bh subfunction 01h is valid only in 320x200 graphics mode. It also sets the screen color palette. The two palettes in 320x200 mode are:

Palette	Colors
Palette 0	Green, Red, and Yellow
Palette 1	Cyan, Magenta, and White

Function 0Ch Write Graphic Pixel

Input:
AH = 0Ch
AL = Pixel Color Number
BH = Video Page Number (can only use in video modes that permit multiple pages)
CX = Screen Column Number
DX = Screen Line Number

Output: None

Description:

Function 0Ch draws a color graphic pixel at the specified coordinates in CX and DX. Specify the video page in BH and the pixel color number in AL. The BH value is ignored in 320x200x256 color mode. If VGA or EGA is used, the BH value is ignored in 320x200x4 color mode.

Function 0Dh Read Graphic Pixel

Input:
AH = 0Dh
BH = Video Page Number (can only use with video modes that permit multiple pages)
CX = Screen Column Number
DX = Screen Line Number

Output: AL = Pixel Color Number

Description:

Function 0Dh reads the color of the pixel specified in CX and DX. The current video page is specified in BH.

cont'd

Function 0Eh Write Character

Input: AH = 0Eh
AL = ASCII Code of the Character
BH = Active page
BL = Foreground color of character (graphics modes)

Output: No registers set. 40:50h is updated.

Description: Function 0Eh writes a character to the current video page at the current cursor position. The cursor column position is incremented after writing the character. If the end of a line is reached, the cursor row position is also incremented and the column position is set to 0. Certain ASCII codes are interpreted as control characters: 07h = beep, 08h = backspace, 0Ah = line feed, and 0Dh = carriage return.

Function 0Fh Return Video Display Mode

Input: AH = 0Fh

Output: AH = Number of Display Columns
AL = Video mode
00h 40x25 text mode, monochrome with CGA
01h 40x25 text mode, color with CGA
02h 80x25 text mode, monochrome with CGA
03h 80x25 text mode, color with CGA
04h 320x200 four-color graphics, with CGA
05h 320x200 monochrome, with CGA
06h 640x200 monochrome, with CGA
07h 80 x 25 monochrome, with monochrome
BH = Current Video Page

Description: Returns the current video mode in AL, the current page number in BH, and the number of columns allowed in this video mode in AH.

Function 13h Write Character String

Input:

AH	=	13h
AL	=	Output Mode: 00h Attribute in BL, do not update cursor position. 01h Attribute in BL, update cursor position. 02h Attribute in string buffer, do not update cursor position. 03h Attribute in string buffer, update cursor position.
BH	=	Video Page Number
BL	=	Attribute of All Characters in Character String
CX	=	Number of characters in buffer
DH	=	Screen Line Number
DL	=	Screen Column Number
ES:BP	=	Segment:Offset Address of String Buffer

Output: No registers set. 40:50h is updated.

Description:

Function 13h writes character strings to the screen and wraps the string to the next line if it is too long for the current text line. Specify the video page number in BH, the screen line number in DH, and the screen column number in DL where the string is to be displayed. The string should be stored in a buffer in RAM. The buffer address segment is in ES and the offset in BP. The number of characters to be displayed from the buffer should be in CX. If output modes 0 or 2 are used, this function does not change the cursor position. If output modes 1 or 3 are used, this function sets the final cursor position to the next position past the last character displayed. If the output mode is 0 or 1, the attribute for all characters in the string is determined by the value in BL. In modes 2 and 3, the string consists of sets of two bytes. The first byte is the ASCII value of the character and the second byte is the attribute of the character.

INT 11h Return System Configuration

Input: None

Output: AX = Configuration Code:

Bits 15–14	Number of parallel ports installed
Bits 13–12	Reserved
Bits 11–9	Number of serial ports installed
Bit 8	Reserved
Bits 7–6	Number of floppy drives is bit 0 is 1
	00 One floppy disk drive
	01 Two floppy disk drives
Bits 5–4	00b VGA or EGA
	01b Video mode is 40x25 CGA
	10b Video mode is 80x25 CGA
	11b Video mode is 80x25 MDA
Bit 3	Reserved
Bit 2	PS/2 mouse present if set
Bit 1	Math coprocessor installed if set
Bit 0	One or more floppy disk drives if set

Description: INT 11h reads the system configuration code. The video mode reported by INT 11h is the mode used when the system was initially booted. Use INT 10h Function 0Fh to find the current video mode.

INT 12h Return Total Memory Size

Input: None

Output: AX = Memory size in kilobytes

Description: INT 12h returns the amount of real mode memory available in the system. Real mode memory is memory below the first megabyte address. Use INT 15h Function 88h to find the amount of memory beyond the first megabyte.

Functions

The INT 13h functions discussed in this chapter are:

Function	Title
00h	Reset Hard Disk Drive
01h	Return Hard Disk Drive Status
02h	Read Disk Sectors
03h	Write Disk Sectors
04h	Verify Disk Sectors
05h	Format Disk Cylinder
06h	Format Disk Track and Mark Lead Sectors
07h	Format Entire Disk Starting at Specified Cylinder
08h	Return Disk Parameters
09h	Initialize Hard Disk Controller
0Ah	Read Hard Disk Sectors and Error Correction Codes
0Bh	Write Hard Disk Sectors and Error Correction Codes
0Ch	Seek Hard Disk Cylinder
0Dh	Reset Hard Disk Controller
10h	Test Unit Ready
11h	Recalibrate Hard Disk
14h	Perform Internal Controller Diagnostic
15h	Return Drive Type

cont'd

Error Codes

For most hard disk drive functions, the following error codes are returned through register AH. All error codes appear in AH.

Code	Description	Code	Description
00h	No error	0Dh	Invalid number of sectors for format on hard disk drive
01h	Function invalid	0Eh	Control data address mark found on hard disk drive
02h	Address mark not found	0Fh	DMA arbitration level out of range
03h	Write attempted on write protected floppy disk	10h	Read error (uncorrectable CRC or ECC)
04h	Sector not found	11h	ECC data error corrected on hard disk drive
05h	Hard disk drive reset failed	20h	Error in floppy disk controller
06h	Floppy disk replaced	40h	Track not found on seek
07h	Hard disk drive parameter is corrupt	80h	Timeout, drive not responding
08h	DMA overflow occurred	AAh	Hard disk drive not ready
09h	DMA crossed 64 KB segment boundary	BBh	Unknown error on hard disk drive
0Ah	Hard disk drive bad sector flag	CCh	Hard disk drive write error occurred
0Bh	Hard disk drive bad track flag	E0h	Hard disk drive status register error
0Ch	Floppy disk media type not found	FFh	Hard disk drive sense operation failed

INT 13h Coding Conventions

For most INT 13h functions, the sector number is placed in CL and the cylinder number in CH.

On a hard disk drive, the cylinder number consists of 10 bits. The lower 8 bits are placed in CH (cylinder number), and the upper 2 bits are placed in CL. The lower 6 bits of CL contain the beginning sector number.

INT 40h Revector for Floppy Functions

INT 13h handles both floppy disk and hard disk drive BIOS functions. If the system has a hard disk drive, the floppy disk device service routine actually resides at INT 40h. All BIOS floppy functions are actually revectored to INT 40h and then executed.

Function 00h Reset Disk Drive

Input: AH = 00h
DL = 80h Hard Disk Drive C:
= 81h – FFh are valid. 81h = D:, 82h = E:, etc.

Output: AH = 00h No error
= Other values are error codes (see page 141).
CF = 0 No error
= 1 Error

Description:

INT 13h Function 00h should be used when an error follows a disk operation. Function 00h resets the disk controller and recalibrates the hard drives attached to the controller. If Function 00h is called for a hard disk drive, the floppy controller is reset and then the hard disk drive controller is reset.

cont'd

Function 01h Return Disk Drive Status

Input: AH = 01h
DL = 80h Hard Disk Drive C:
= 81h – FFh are valid. 81h = D:, 82h = E:, etc.

Output: AH = 00h No error
= Other values are error codes (see page 141).
CF = 0 No error
= 1 Error

Description:

INT 13h Function 01h can be used to read the status of the last operation.

Function 02h Read Disk Sectors

Input: AH = 02h
AL = Number of Sectors to Read
CH = Cylinder Number (low 8 bits)
CL = High two bits of cylinder number in bits 7–6
DH = Head Number
DL = 80h Hard Disk Drive C:
= 81h – FFh are valid. 81h = D:, 82h = E:, etc.
ES:BX = Buffer Segment:Offset Address

Output: AH = 00h No error
= Other values are error codes (see page 141).
CF = 0 No error
= 1 Error

Description:

Function 02h reads the specified number of sectors from a specified track on one side of a disk. The sector(s) are read from the disk and then stored in a buffer at address ES:BX.

Function 03h Write Disk Sectors

Input:
AH = 03h
AL = Number of Sectors to Write
CH = Cylinder Number (low 8 bits)
CL = High two bits of cylinder number in bits 7–6
DH = Head Number
DL = 80h Hard Disk Drive C:
= 81h – FFh are valid. 81h = D:, 82h = E:, etc.
ES:BX = Buffer Offset:Segment Address

Output:
AH = 00h No error
= Other values are error codes (see page 141).
CF = 0 No error
= 1 Error

Description: Function 03h writes the number of sectors in AL to the cylinder number in CH using the disk drive head specified in DH. The beginning sector number is in CL. The data written to the sectors comes from the buffer at address ES:BX.

Function 04h Verify Disk Sectors

Input:
AH = 04h
AL = Number of Sectors to Verify
CH = Cylinder Number (low 8 bits)
CL = High two bits of cylinder number in bits 7–6
DH = Head Number
DL = 80h Hard Disk Drive C:
= 81h – FFh are valid. 81h = D:, 82h = E:, etc.

Output:
AH = 00h No error
= Other values are error codes (see page 141).
CF = 0 No error
= 1 Error

Description: Function 04h verifies that the ECC code after each sector is correct for the data in that sector.

cont'd

Function 05h Format Disk Track

Input:

AH	=	05h
AL	=	Interleave Factor
CH	=	Cylinder Number (low 8 bits)
CL	=	High two bits of cylinder number in bits 7–6
DL	=	80h Hard Disk Drive C:
	=	81h – FFh are valid. 81h = D:, 82h = E:, etc.
ES:BX	=	Buffer Segment:Offset Address

Output:

AH	=	00h No error
	=	Other values are error codes (see page 141).
CF	=	0 No error
	=	1 Error

Description:

Function 05h formats an entire track or cylinder on a disk. A buffer with sector data is in ES:BX. The buffer contains a two-byte record:

Byte 0	00h — good sector (80h — bad sector)
Byte 1	Sector number

Function 06h Format Track and Mark Lead Sectors

Input:
AH = 06h
AL = Interleave Factor
CH = Cylinder Number (low 8 bits)
CL = High two bits of cylinder number in bits 7–6
DH = Head Number
DL = 80h Hard Disk Drive C:
= 81h – FFh are valid. 81h = D:, 82h = E:, etc.

Output:
AH = 00h No error
= Other values are error codes (see page 141).
CF = 0 No error
= 1 Error

Description: Function 06h formats an entire track or cylinder of a hard disk and marks the bad sectors that it finds so they cannot be used. See INT 13h Function 05h above for more about formatting.

Function 07h Format Entire Disk Starting at Specified Cylinder

Input:
AH = 07h
AL = Interleave Factor
CH = Cylinder Number (low 8 bits)
CL = High two bits of cylinder number in bits 7–6
DL = 80h Hard Disk Drive C:
= 81h – FFh are valid. 81h = D:, 82h = E:, etc.

Output:
AH = 00h No error
= Other values are error codes (see page 141).
CF = 0 No error
= 1 Error

Description: Formats an entire hard disk, starting at the cylinder number specified in CH and CL. Function 06h also marks bad sectors so these sectors cannot be used. See Function 05h for more on formatting.

cont'd

Function 08h Return Disk Parameters

Input: AH = 08h
DL = 80h Hard Disk Drive C:
= 81h – FFh are valid. 81h = D:, 82h = E:, etc.

Output: AH = 00h Error Code
= Other values are error codes (see page 141).
AL = 00h
CF = 0 No error
= 1 Error
CH = Lower 8 bits of last cylinder number
CL = High two bits of last cylinder number and six bits for last sector number
DH = Last head number
DL = Number of disk drives
ES:DI = Address of disk parameter table from BIOS

Description:

INT 13h Function 08h retrieves the parameters for a hard disk drive from the ROM BIOS.

Function 09h Initialize Hard Disk Controller

Input: AH = 09h
DL = 80h Hard Disk Drive C:
= 81h – FFh are valid. 81h = D:, 82h = E:, etc.

Output: AH = 00h No error
= Other values are error codes (see page 141).
CF = 0 No error
= 1 Error

Description:

INT 13h Function 09h initializes the hard disk controller with the values in the BIOS hard disk parameter table. The vector address for INT 41h points to the drive C: disk parameters and the vector for INT 46h points to the drive D: parameters. On an ISA system, the blocks are 16 bytes, in the following format:

Offset	Description
00h – 01h	Number of cylinders. Byte 01h is the most significant byte.
02h	Number of heads.
03h – 04h	Reserved
05h – 06h	Starting write precompensation cylinder. Byte 06h is the MSB.
07h	ECC burst length
08h	Control Byte Bits 7–6 Enable or Disable Retries 00h Enable retries. All other values disable retries. Bit 5 Set if defect map is located at last cylinder plus one. Bit 4 Reserved. Always set to 0. Bit 3 Set if more than 8 heads. Bits 2–0 Reserved. Always set to 0.
09h – 0Bh	Reserved
0Ch – 0Dh	Landing Zone
0Eh	Number of Sectors per Track
0Fh	Reserved

cont'd

Function 0Ah Read Hard Disk Sectors and Error Correction Codes

Input:

AH	=	0Ah
AL	=	Number of Sectors to Read
CH	=	Lower eight bits of last cylinder number
CL	=	Highest two bits of last cylinder number and six bits for beginning sector number
DH	=	Head Number
DL	=	80h Hard Disk Drive C:
	=	81h – FFh are valid. 81h = D:, 82h = E:, etc.
ES:BX	=	Buffer Segment:Offset Address

Output:

AH	=	00h No error
	=	Other values are error codes (see page 141).
CF	=	0 No error
	=	1 Error

Description:

Function 0Ah reads the number of sectors in AL from the hard disk specified in DL and the location specified in CH and CL using the head number specified in DH and stores it to memory. It also reads a four-byte ECC code for each sector.

INT 13h Function 02h also reads sectors from the hard disk, but terminates the operation when a read error occurs.

Function 0Ah does not terminate on error.

Function 0Bh Write Hard Disk Sectors and Error Correction Codes

Input:

AH	=	0Bh
AL	=	Number of Sectors to Write
CH	=	Lower eight bits of last cylinder number
CL	=	Highest two bits of last cylinder number and six bits for beginning sector number
DH	=	Head Number
DL	=	80h Hard Disk Drive C:
	=	81h – FFh are valid. 81h = D:, 82h = E:, etc.
ES:BX	=	Buffer Segment:Offset Address

Output:

AH	=	00h No error
	=	Other values are error codes (see page 141).
CF	=	0 No error
	=	1 Error

Description:

Function 0Bh writes the number of sectors specified in AL to the hard disk specified in DL using the head number specified in DH. It also writes a four-byte Error Correction Code (ECC) for each sector. The four-byte ECC must follow the data to be written to each sector.

The data to be written to the drive is stored at the location pointed to in ES:BP. The buffer must contain 512 bytes of data followed by a four-byte ECC, then another 512 bytes of data and another four-byte ECC, and so on.

cont'd

Function 0Ch Seek Hard Disk Cylinder

Input: AH = 0Ch
CH = Cylinder Number (lower eight bits)
CL = Cylinder Number (upper two bits)
DH = Head Number
DL = 80h Hard Disk Drive C:
= 81h – FFh are valid. 81h = D:, 82h = E:, etc.

Output: AH = 00h No error
= Other values are error codes (see page 141).
CF = 0 No error
= 1 Error

Description:

Function 0Ch moves the hard disk heads to the specified cylinder but does not transfer data. You do not have to call this function before calling Functions 0Ah Read or 0Bh Write, since functions 0Ah and 0Bh perform a Seek command.

Function 0Dh Reset Hard Disk Controller

Input: AH = 0Dh
DL = 80h Hard Disk Drive C:
= 81h – FFh are valid. 81h = D:, 82h = E:, etc.

Output: AH = 00h No error
= Other values are error codes (see page 141).
CF = 0 No error
CF = 1 Error

Description:

Function 0Dh resets the specified hard disk drive. Unlike Function 00h, this function does not reset the floppy controller.

Function 10h Test Unit Ready

Input: AH = 10h
DL = 80h Hard Disk Drive C:
= 81h – FFh are valid. 81h = D:, 82h = E:, etc.

Output: AH = 00h No error
= Other values are error codes (see page 141).
CF = 0 No error. Hard Disk Drive Ready
= 1 Error. Hard Disk Drive Not Ready

Description:

Function 10h determines if the hard disk drive specified in DL is ready.

Function 11h Recalibrate Hard Disk

Input: AH = 11h
DL = 80h Hard Disk Drive C:
= 81h – FFh are valid. 81h = D:, 82h = E:, etc.

Output: AH = 00h No error
= Other values are error codes (see page 141).
CF = 0 No error
= 1 Error

Description:

Function 11h recalibrates the specified hard disk drive, places the read/write head at cylinder 0, and returns the drive status in AH.

---

cont'd

Function 14h Perform Internal Controller Diagnostic

Input: AH = 14h
DL = 80h Hard Disk Drive C:
= 81h – FFh are valid. 81h = D:, 82h = E:, etc.

Output: AH = 00h No error
= Other values are error codes (see page 141).
CF = 0 No error
= 1 Error

Description:

Function 14h executes a diagnostic self-test routine built into ISA hard disk controllers. This diagnostic routine returns the status and results in AH.

Function 15h Return Drive Type

Input: AH = 15h
DL = 80h Hard Disk Drive C:
= 81h – FFh are valid. 81h = D:, 82h = E:, etc.

Output: AH = 00h No drive present
= 03h Drive is a hard disk
CF = 00h No error
= 1 Error
CX:DX = Number of 512 byte sectors

Description:

If AH is 03h, the drive is a hard disk and CX:DX contains the number of 512-byte sectors.

Functions

The INT 13h Floppy Disk functions discussed in this chapter are:

Function	Title
00h	Reset Floppy Disk Drive
01h	Return Disk Drive Status
02h	Read Floppy Disk Sectors
03h	Write Disk Sectors
04h	Verify Disk Sectors
05h	Format Disk Track
08h	Return Disk Parameters
15h	Return Drive Type
16h	Disk Media Change Status
17h	Set Floppy Disk Type
18h	Set Floppy Disk Type before Format

cont'd

Error Codes

For most floppy and hard disk drive functions, the following error codes are returned through register AH. All error codes appear in AH.

Code	Description	Code	Description
00h	No error	0Dh	Invalid number of sectors for format on hard disk drive
01h	Function invalid	0Eh	Control data address mark found on hard disk drive
02h	Address mark not found	0Fh	DMA arbitration level out of range
03h	Write attempted on write protected floppy disk	10h	Read error (uncorrectable CRC or ECC)
04h	Sector not found	11h	ECC data error corrected on hard disk drive
05h	Hard disk drive reset failed	20h	Error in floppy disk controller
06h	Floppy disk replaced	40h	Track not found on seek
07h	Hard disk drive parameter is corrupt	80h	Timeout, drive not responding
08h	DMA overflow occurred	AAh	Hard disk drive not ready
09h	DMA crossed 64 KB segment boundary	BBh	Unknown error on hard disk drive
0Ah	Hard disk drive bad sector flag	CCh	Hard disk drive write error occurred
0Bh	Hard disk drive bad track flag	E0h	Hard disk drive status register error
0Ch	Floppy disk media type not found	FFh	Hard disk drive sense operation failed

INT 13h Coding Conventions

For most INT 13h functions, the sector number is placed in CL and the cylinder number in CH.

INT 40h Revector for Floppy Functions

INT 13h handles both floppy disk and hard disk drive BIOS functions. If the system has a hard disk drive, the floppy disk service routine actually resides at INT 40h. All BIOS floppy functions are actually revectored to INT 40h and then executed.

Function 00h Reset Disk Drive

Input: AH = 00h
DL = 00h Floppy Drive A:
01h Floppy Drive B:

Output: AH = 00h No error
= Other values are error codes (see page 155).
CF = 0 No error
= 1 Error

Description:

INT 13h Function 00h should be used when an error follows a disk operation. Function 00h resets the disk controller and recalibrates the floppy drives attached to the floppy controller. If Function 00h is issued for a hard disk drive, the floppy controller is reset, then the hard disk drive controller is reset.

cont'd

Function 01h Return Disk Drive Status

Input: AH = 01h
DL = 00h Floppy Drive A:
01h Floppy Drive B:

Output: AH = 00h No error
= Other values are error codes (see page 155).
CF = 0 No error
= 1 Error

Description:

INT 13h Function 01h can be used to read the status of the last disk operation.

Function 02h Read Disk Sectors

Input: AH = 02h
AL = Number of Sectors to Read
CH = Track Number
CL = Beginning Sector Number
DH = Side 0 or 1
DL = 00h Floppy Drive A:
01h Floppy Drive B:
ES:BX = Buffer Segment:Offset Address

Output: AH = 00h No error
= Other values are error codes (see page 155).
AL = Number of sectors actually read
CF = 0 No error
= 1 Error

Description:

Function 02h reads the specified number of sectors from a specified track on one side of a disk. The sector(s) are read from the disk and then stored in a buffer at address ES:BX.

Function 03h Write Disk Sectors

Input:

AH	=	03h
AL	=	Number of Sectors to Write
CH	=	Track Number
CL	=	Beginning Sector Number
DH	=	Floppy Side 0 or 1
DL	=	00h Floppy Drive A: 01h Floppy Drive B:
ES:BX	=	Buffer Offset:Segment Address

Output:

AH	=	00h No error
	=	Other values are error codes (see page 155).
AL	=	Number of sectors actually written
CF	=	0 No error
	=	1 Error

Description:

Function 03h writes the number of sectors in AL to the track in CH on one side (specified in DH) of a floppy disk. The beginning sector number is in CL. The data written to the sectors comes from the buffer at address ES:BX.

cont'd

Function 04h Verify Disk Sectors

Input:
AH = 04h
AL = Number of Sectors to Verify
CH = Track Number
CL = Sector Number
DH = Floppy Side 0 or 1
DL = 00h Floppy Drive A:
01h Floppy Drive B:

Output:
AH = 00h No error
= Other values are error codes (see page 155).
AL = Number of sectors actually read and verified.
CF = 0 No error
= 1 Error

Description:

Function 04h verifies that the ECC code at the end of each sector is correct for the data contained in that sector.

Function 05h Format Disk Cylinder

Input:
AH = 05h
AL = Number of Sectors to Format
CH = Track Number
DH = Floppy Side 0 or 1
DL = 00h Floppy Drive A:
01h Floppy Drive B:
ES:BX = Buffer Segment:Offset Address

Output:
AH = 00h No error
= Other values are error codes (see page 155).
CF = 0 No error
= 1 Error

Description:

Function 05h formats an entire track or cylinder on a disk. A buffer containing sector information is passed through ES:BX.

The buffer contains a four-byte record for each sector in the track, in the following format:

Byte 0 Track number
Byte 1 Head number
Byte 2 Logical sector number
Byte 3 Number of bytes per sector:
0 128 bytes per sector
1 256 bytes per sector
2 512 bytes per sector (ISA and EISA Standard)
3 1024 bytes per sector

Call INT 13h function 17h or 18h to set the floppy disk media type before invoking this function.

cont'd

Function 08h Return Disk Parameters

Input:

AH	=	08h
DL	=	00h Floppy Drive A: 01h Floppy Drive B:

Output:

AH	=	00h Error Code
	=	Other values are error codes (see page 155).
BL	=	Drive type (for floppy drives): 01h for 360 KB, 40 track 5¼" 02h for 1.2 MB, 80 track 5¼" 03h for 720 KB, 80 track 3½" 04h for 1.44 MB, 80 track 3½"
CF	=	0 No error
	=	1 Error
CH	=	Lower 8 bits of last cylinder number
CL	=	High two bits of last cylinder number and six bits for last sector number
DH	=	Last head number
DL	=	Number of disk drives
ES:DI	=	Address of disk parameter table from BIOS

Function 08h Return Disk Parameters, cont'd

Description:

INT 13h Function 08h retrieves the parameters for a floppy disk drive from the ROM BIOS.

00h is returned in BL when:

- the drive type is known but CMOS RAM data is invalid or not present,
- the CMOS RAM battery is low, or
- the CMOS RAM checksum value is corrupt.

If the specified drive is not installed, all returned values are 00h.

The value for AX, ES, BX, CX, DH, DI is 0, and DL is the number of drives present if any of the following is true:

- the specified drive number is invalid,
- the specified drive type is unknown and CMOS RAM is not present,
- the CMOS RAM battery is low or the CMOS RAM checksum is invalid, or
- the drive type is unknown and the drive type stored in CMOS RAM is invalid.

cont'd

Function 15h Return Drive Type

Input: AH = 15h
DL = 00h Floppy Drive A:
01h Floppy Drive B:

Output: AH = 00h No drive present
= 01h Drive does not have change line support
= 02h Drive does have change line support
CF = 0 No error
= 1 Error

Description:

Function 15h determines if floppy disk change line information ia available.

Function 16h Disk Media Change Status

Input: AH = 16h
DL = 0 Floppy Drive A:
= 1 Floppy Drive B:

Output: AH = 00h No floppy disk (media) change
= 01h Invalid floppy disk parameter
= 06h Floppy disk was changed since last access
= 80h Floppy disk drive not ready

Description:

Function 16h determines if a media change was made since the last floppy disk access.

Function 17h Set Floppy Disk Type

Input: AH = 17h
AL = Floppy Disk Format
= 01h 320 or 360 KB floppy in 320 or 360 KB drive
= 02h 360 KB floppy in 1.2 MB floppy drive
= 03h 1.2 MB floppy in 1.2 MB floppy drive
= 04h 720 KB floppy in 720 KB floppy drive
DL = 00h Floppy Drive A:
01h Floppy Drive B:

Output: AH = 00h No error
= Other values are error codes (see page 155).
CF = 0 No error
= 1 Error

Description: Function 17h sets the format of a disk in a floppy drive and sets the data rate and media type if the drive supports the disk change line.

Function 18h Set Floppy Disk Type before Format

Input: AH = 18h
CH = Maximum number of tracks
CL = Sectors per track
DL = 00h Floppy Drive A:
01h Floppy Drive B:

Output: AH = 00h Specified track and sector data is supported
= Other values are error codes (see page 155).
CF = 0 No error
= 1 Error
ES:DI = Pointer to drive parameter table if AH is 00h.

Description: Sets the media type before formatting a floppy disk. Call Function 18h before Function 05h.

INT 14h Serial Communications Service

INT 14h accesses and controls the serial ports. The AMIBIOS permits up to four serial ports to be configured. These serial ports are initialized to the following starting I/O port addresses: 3F8h, 2F8h, 3E8h, and 2E8h.

The default values for the serial I/O port addresses used in a Hi-Flex AMIBIOS can be modified via AMIBCP.

INT 14h Functions

Functions 00h through 03h are defined in ISA standards. Functions 04h and 05h are defined in PS/2 standards and are only available in a Hi-Flex AMIBIOS dated 080891 (August 8, 1991) or later.

Function	Title
00h	Initialize Serial Port
01h	Send Character to Serial Port
02h	Receive Character from Serial Port
03h	Read Serial Port Status
04h	Extended Initialize Serial Port
05h	Extended Serial Port Control

Serial I/O Ports

The Serial port I/O consists of eight contiguous I/O ports, in the following format:

I/O Port	Read and Write Status	Description
Base	Write	Transmitter Holding Register (contains the character to be sent). Bit 0, the least significant bit, is sent first. Bits 7–0 Contains data bits 7–0 when the Divisor Latch Access Bit (DLAB) is 0.
Base	Read	Receiver Buffer Register (contains the received character). Bit 0, the least significant bit, is received first. Bits 7–0 Contains data bits 7–0 when the Divisor Latch Access Bit (DLAB) is 0.
Base	Read and Write	Divisor Latch, low byte Both divisor latch registers store the data transmission rate divisor. Bits 7–0 Bits 7–0 of divisor when DLAB is 1.
Base + 1	Read and Write	Divisor Latch, high byte. Bits 7–0 Bits 15–8 of data transmission rate divisor when DLAB is 1.
Base + 1	Read and Write	Interrupt Enable Register. Permits the serial port controller interrupts to enable the chip interrupt output signal. Bits 7–4 Reserved Bit 3 Modem status interrupt enable if set. Bit 2 Receiver line status interrupt enable if set. Bit 1 Transmitter Holding register empty interrupt enable if set. Bit 0 Received data available interrupt enable when DLAB is 0 if set.

I/O Port	Read and Write Status	Description
Base + 2	Read	Interrupt ID Register. Information about a pending interrupt is stored here. When the ID register is addressed, the highest priority interrupt is held and no other interrupts are acknowledged until the microprocessor services the interrupt with the highest priority. Bits 7–3 Reserved Bits 2–1 The pending interrupt that has the highest priority. 11b Receiver Line Status Interrupt, priority is the highest. 10b Received Data Available, second in priority. 01b Transmitter Holding Register Empty, third in priority. 00b Modem Status Interrupt, fourth in priority. Bit 0 Interrupt pending if set to logical 0. If logical 1, no interrupt is pending.
Base + 3	Read and Write	Line Control Register Bit 7 Divisor Latch Access Bit (DLAB) 0 Access receiver buffer, transmitter holding register, and interrupt enable register. 1 Access Divisor Latch of baud rate generator. Bit 6 Set Break Control. Serial output is forced to spacing state and remains there if set. Bit 5 Stick Parity. Bit 4 Even Parity Select. Bit 3 Parity Enable. Bit 2 Number of Stop Bits per Character. 0 One stop bit. 1 1½ stop bits if 5-bit word length is selected. 2 stop bits if 6, 7, or 8-bit word length is selected. Bits 1–0 Number of Lines per character 00b 5-Bit word length. 01b 6-Bit word length. 10b 7-Bit word length. 11b 8-Bit word length.

I/O Port	Read and Write Status	Description
Base + 4	Read and Write	Modem Control Register Bits 7–5 Reserved Bit 4 Loopback mode for diagnostic testing of serial port if set. The output from the transmitter shift register is looped back to the receiver shift register input. Transmitted data is immediately received so the microprocessor can verify the transmit and receive data serial port paths. Bit 3 Force OUT2 interrupt if set. Bit 2 Force OUT1 active if set. Bit 1 Force Request To Send active if set. Bit 0 Force Data Terminal Ready active if set.
Base + 5	Read Only	Line Status Register Bit 7 Reserved Bit 6 Transmitter shift and holding registers empty if set. Bit 5 Transmitter holding register empty if set. The controller is ready to accept a new character to send. Bit Break interrupt if set. The received data input is held in the zero bit state longer than the transmission time of the start bit + data bits + parity bits + stop bits. Bit 3 Framing error if set. The stop bit that follows the last parity or data bit is zero. Bit 2 Parity error if set. The character has incorrect parity. Bit 1 Overrun error if set. A character was sent to the receiver buffer before the previous character in the buffer could be read, which destroys the previous character. Bit 0 Data Ready if set. A complete incoming character has been received and sent to the receiver buffer register.
Base + 6	Read Only	Modem Status Register Bit 7 Data Carrier Detect if set. Bit 6 Ring Indicator if set. Bit 5 Data Set Ready if set. Bit 4 Clear To Send if set. Bit 3 Delta Data Carrier Detect if set. Bit 2 Trailing Edge Ring Indicator if set. Bit 1 Delta Data Set Ready if set. Bit 0 Delta Clear To Send if set.
Base + 7	Read and Write	Reserved

cont'd

Function 00h Initialize Serial Port

Input: AH = 00h
AL = Parameter byte
Bits 7–5 Data transmission rate
000b 110
001b 150
010b 300
011b 600
100b 1200
101b 2400
110b 4800
111b 9600
Bits 4–3 Parity
00b No parity
01b Odd parity
10b No parity
11b Even parity
Bit 2 Number of stop bits
0 One bit
1 Two bits
Bits 1–0 Data length
10b Seven bits
11b Eight bits
DX = Serial Port Number. Index to serial port base table at 40:00h.
00h COM 1
01h COM 2
02h COM 3
03h COM 4

Function 00h Initialize Serial Port, cont'd

Output: AH = Line Status
- Bit 7 Timeout if set.
- Bit 6 Transmit Shift Register is empty if set.
- Bit 5 Transmit Holding Register is empty if set.
- Bit 4 Break signal detected if set.
- Bit 3 Framing error detected if set.
- Bit 2 Parity error detected if set.
- Bit 1 Data overrun error detected if set.
- Bit 0 Receive data ready if set.

AL = Modem Status
- Bit 7 Receive line signal detected if set.
- Bit 6 Ring indicator if set.
- Bit 5 Data set ready if set.
- Bit 4 Clear to send if set.
- Bit 3 Delta receive line signal detect if set.
- Bit 2 Trailing edge ring indicator if set.
- Bit 1 Delta data set ready if set.
- Bit 0 Delta clear to send if set.

Description:

Function 00h initializes the specified serial port with the parameters in the parameter byte (AL). It returns the line status and the modem status.

cont'd

Function 01h Send Character to Serial Port

Input: AH = 01h
AL = Character to be sent
DX = Serial port number. Index to serial port base table at 40:00h.
00h COM 1 01h COM 2
02h COM 3 03h COM 4

Output: AH = Line Status
Bit 7 Timeout error if set.
Bit 6 Transmit Shift and Holding Register empty if set.
Bit 5 Transmit Holding Register empty if set.
Bit 4 Break Interrupt if set.
Bit 3 Framing Error if set.
Bit 2 Parity Error if set.
Bit 1 Data overrun error detected if set.
Bit 0 Receive data ready if set.
AL = Character Sent

Description:

Function 01h sends a character to the serial port. It returns the line status.

Function 02h Receive Character from Serial Port

Input: AH = 02h
DX = Serial Port Number. Index to serial port base table at 40:00h.
00h COM 1 01h COM 2
02h COM 3 03h COM 4

Output: AH = Line Status
Bit 7 Timeout error if set.
Bit 6 Transmit Shift and Holding Register empty if set.
Bit 5 Transmit Holding Register empty if set.
Bit 4 Break Interrupt if set.
Bit 3 Framing Error if set.
Bit 2 Parity Error if set.
Bit 1 Data overrun error detected if set.
Bit 0 Receive data ready if set.
AL = Character Received

Description:

Function 02h receives a character in AL from the serial port. Function 02h also returns the port status in AH.

cont'd

Function 03h Return Serial Port Status

Input: AH = 03h
DX = Serial Port Number. Index to serial port base table at 40:00h.
00h COM 1 01h COM 2
02h COM 3 03h COM 4

Output: AH = Line Status
Bit 7 Timeout error if set.
Bit 6 Transmit Shift and Holding Register empty if set.
Bit 5 Transmit Holding Register empty if set.
Bit 4 Break Interrupt if set.
Bit 3 Framing Error if set.
Bit 2 Parity Error if set.
Bit 1 Data overrun error detected if set.
Bit 0 Receive data ready if set.
AL = Modem Status
Bit 7 Receive line signal detected if set.
Bit 6 Ring indicator if set.
Bit 5 Data set ready if set.
Bit 4 Clear to send if set.
Bit 3 Delta receive line signal detect if set.
Bit 2 Trailing edge ring indicator if set.
Bit 1 Delta data set ready if set.
Bit 0 Delta clear to send if set.

Description:

Function 03h returns the status of the specified serial port. Function 03h differs from function 00h. Function 03h has no initialization process, but Function 00h does.

Function 04h Extended Initialize Serial Port

Input:

AH = 04h

AL = 00h No break signal
01h Break signal

BH = 00h No parity
01h Odd parity
02h Even parity
03h Stick parity odd
04h Stick parity even

BL = 00h 1 Stop bit
01h 2 Stop bits if data length is 6, 7, or 8 bits
10h 1½ Stop bits if data length is 5 bits

CH = 00h Data length is 5 bits
01h Data length is 6 bits
02h Data length is 7 bits
03h Data length is 8 bits

CL = 00h 110 bps
01h 150 bps
02h 300 bps
03h 600 bps
04h 1200 bps
05h 2400 bps
06h 4800 bps
07h 9600 bps
08h 19200 bps

DX = Serial port number. Index to serial port base table at 40:00h.
00h COM 1 01h COM 2
02h COM 3 03h COM 4

cont'd

Function 04h Extended Initialize Serial Port, cont'd

Output: AH = Line Status
- Bit 7 Timeout if set
- Bit 6 Transmit Shift Register is empty if set.
- Bit 5 Transmit Holding Register is empty if set.
- Bit 4 Break signal detected if set.
- Bit 3 Framing error detected if set.
- Bit 2 Parity error detected if set.
- Bit 1 Data overrun error detected if set.
- Bit 0 Receive data ready if set.

AL = Modem Status
- Bit 7 Receive line signal detected if set.
- Bit 6 Ring indicator if set.
- Bit 5 Data set ready if set.
- Bit 4 Clear to send if set.
- Bit 3 Delta receive line signal detect if set.
- Bit 2 Trailing edge ring indicator if set.
- Bit 1 Delta data set ready if set.
- Bit 0 Delta clear to send if set.

Description:

Function 04h initializes the specified serial port with the parameters in the parameter byte (AL). Function 04h returns the line and modem status (if a modem is attached).

Function 04h differs from Function 00h because the input parameters are different.

Function 05h Extended Serial Port Control Subfunction AL = 00h Read from Modem Control Register

Input: AH = 05h
AL = 00h Read from Modem Control Register
DX = Serial Port Number. Index to serial port base table at 40:00h.
00h COM 1 01h COM 2
02h COM 3 03h COM 4

Output: AH = Line Status
Bit 7 Timeout if set.
Bit 6 Transmit Shift Register is empty if set.
Bit 5 Transmit Holding Register is empty if set.
Bit 4 Break signal detected if set.
Bit 3 Framing error detected if set.
Bit 2 Parity error detected if set.
Bit 1 Data overrun error detected if set.
Bit 0 Receive data ready if set.

AL = Modem Status
Bit 7 Receive line signal detected if set.
Bit 6 Ring indicator if set.
Bit 5 Data set ready if set.
Bit 4 Clear to send if set.
Bit 3 Delta receive line signal detect if set.
Bit 2 Trailing edge ring indicator if set.
Bit 1 Delta data set ready if set.
Bit 0 Delta clear to send if set.

BL = Modem Control Register
Bits 7–5 Reserved
Bit 4 Loop for testing if set.
Bit 3 OUT2 if set.
Bit 2 OUT1 if set.
Bit 1 Request to send if set.
Bit 0 Data terminal ready if set.

Description: Function 05h reads or sets the modem control register for the specified serial port.

cont'd

Function 05h Extended Serial Port Control Subfunction AL = 01h Write to Modem Control Register

Input: AH = 05h
AL = 01h Write to Modem Control Register
DX = Serial Port Number. Index to serial port base table at 40:00h.
00h COM 1 01h COM 2
02h COM 3 03h COM 4

Output: AH = Line Status
Bit 7 Timeout if set.
Bit 6 Transmit Shift Register empty if set.
Bit 5 Transmit Holding Register empty if set.
Bit 4 Break signal detected if set.
Bit 3 Framing error detected if set.
Bit 2 Parity error detected if set.
Bit 1 Data overrun error detected if set.
Bit 0 Receive data ready if set.

AL = Modem Status
Bit 7 Receive line signal detected if set.
Bit 6 Ring indicator if set.
Bit 5 Data set ready if set.
Bit 4 Clear to send if set.
Bit 3 Change in receive line signal detect if set.
Bit 2 Trailing edge ring indicator if set.
Bit 1 Change in data set ready if set.
Bit 0 Change in clear to send if set.

BL = Modem Control Register
Bits 7–5 Reserved
Bit 4 Loop for testing if set.
Bit 3 OUT2 if set.
Bit 2 OUT1 if set.
Bit 1 Request to send if set.
Bit 0 Data terminal ready if set.

Description: Function 05h reads or sets the modem control register for the specified serial port.

INT 15h Systems Services

INT 15h provides a variety of services:

Category	Description and INT 15h Functions
EISA Support	INT 15h Function D8h, subfunctions 00h through 04h, are defined only in the EISA specifications and are supported in the EISA BIOS.
Multitasking Services	The BIOS provides six hooks that can be used by programmers: INT 15h Functions 80h, 81h, 82h, 85h, 90h, and 91h are defined in the ISA standard and are available in the BIOS but do not perform any service. Software developers can trap or redirect the vectors of these interrupt functions to point to programmer-supplied service routines. No routines for these functions are provided in the BIOS.
Protected Mode Services	Function 87h Move Block provides a way to move large blocks of information from conventional to extended memory. Function 89h switches to protected mode.
Wait Routines	Functions 83h and 86h provide wait control. Function 86h does not return control to the calling program until a specified interval completes. Function 83h returns control to the caller immediately but sets a bit when a predetermined wait period is finished.
System Information	Function C1h returns the extended BIOS data area address. Function C0h returns system configuration data. Function 88h returns the extended memory size.
Advanced Power Management	Function 53h provides power management functions that conform to the Microsoft/Intel APM specification for systems that have power management functions.
PS/2 Support	Functions 4Fh, C1h, and C2h are defined in the PS/2 specification. AMIBIOS supports some PS/2-defined operations, including all PS/2 mouse operations. The programmer can invoke these mouse functions if the system includes the necessary hardware as well as the appropriate American Megatrends Keyboard Controller BIOS (version KF or later). Function C2h PS/2 Mouse Support is supported in all AMIBIOSes dated August 8, 1991 (080891) or later.
Tape Cassette Services	The only INT 15h function on the original PC was cassette tape I/O. In AMIBIOS, these functions (00h, 01h, 02h, and 03h) are not supported. If called, the BIOS sets the Carry Flag in the FLAGS register and returns AH = 86h (no cassette present). You can trap Functions 00h – 03h and substitute your own code.
Joystick support	Function 84h provides joystick support for up to two joysticks.

cont'd

INT 15h Functions

Function	Title
4Fh	Keyboard Intercept
53h	Advanced Power Management AL = 00h APM Installation Check AL = 01h APM Real Mode Interface Connect AL = 02h APM 16-Bit Protected Mode Interface Connect AL = 03h APM 32-Bit Protected Mode Interface Connect AL = 04h APM Interface Disconnect AL = 05h CPU Idle AL = 06h CPU Busy AL = 07h Set Power State AL = 08h Enable Power Management AL = 09h Restore BIOS Power-On Defaults AL = 0Ah Get Power Status AL = 0Bh Get PM Event AL = 0Ch Get Power State AL = 0Dh Enable Device Power Management AL = 80h OEM-Defined APM Functions BH = 7Fh APM Installation Check BH = 00h-7Eh; 80h-FFh OEM-Defined Function
80h	Device Open (replaced by BIOS user routine)
81h	Device Close (replaced by BIOS user routine)
82h	Program Termination (replaced by BIOS user routine)
83h	Set Event Wait Interval
84h	Joystick Support DX = 001h Read Current Switch Settings DX = 01h Read Resistive Inputs
85h	System Request Key (replaced by BIOS user routine)
86h	Wait
87h	Move Block
88h	Return Extended Memory Size
89h	Switch to Protected Mode
90h	Device Busy Loop (replaced by BIOS user routine)
91h	Interrupt Complete (replaced by BIOS user routine)
C0h	Return System Configuration Parameters
C1h	Return Address of Extended BIOS Data Area
C2h	PS/2 Mouse Support
C3h	Fail-Safe Timer
D8h	EISA Support

Function 4Fh PS/2 Keyboard Intercept

Input: AH = 4Fh
AL = Scan Code

Output: AL = Scan Code
CF = 0 Scan Code processed but should not go to keyboard buffer.
= 1 Scan Code processed or modified and should go to keyboard buffer.

Description:

INT 09h calls this function each time a key is pressed. Function 4Fh can be used to search the data from a keyboard. If the specified scan code is found, the routine provided by the programmer is executed. This routine can modify the scan code.

cont'd

Function 53h Subfunction AL = 00h APM Installation Check

Mode: Real Mode

Input:
AH = 53h
AL = 00h
BX = Power Device ID
= 0000h BIOS

Output:
AH = 1 APM major version number (in BCD)
AL = 1 APM minor version number (in BCD)
BH = P (in ASCII)
BL = M (in ASCII)
CF = 0 APM is supported by the BIOS.
= 1 APM is not supported by the BIOS.
CX = APM Flags
Bit 3 1 BIOS Power Management is disabled.
Bit 2 0 A *CPU Idle* call does not slow the processor clock speed or stop the clock.
Bit 1 1 The 32-bit protected mode interface is supported.
Bit 0 1 The 16-bit protected mode interface is supported.

Description:

This subfunction allows the APM driver (the calling program) to ascertain the APM specification version that is supported. It also specifies if the system BIOS supports APM.

Function 53h Subfunction AL = 01h APM Real Mode Interface Connect

Mode: Real Mode

Input:
AH = 53h
AL = 01h
BX = Power Device ID
= 0000h BIOS

Output:
AH = Error code if unsuccessful
= 02h A real mode interface connection is already established.
= 05h A 16-Bit protected mode interface connection is already established.
= 07h A 32-Bit protected mode interface connection is already established.
= 09h Device ID unrecognized.
CF = 0 Successful
= 1 Unsuccessful
CX = APM 16-bit data segment (real mode segment base address)

Description:

This subfunction initializes the interface between the APM Driver (the calling program) and the BIOS. Before the interface is established, the BIOS provides OEM-defined power management. Once the interface is defined, the APM driver and the BIOS coordinate power management activities.

cont'd

Function 53h Subfunction AL = 02h APM 16-Bit Protected Mode Interface Connect

Mode: Real Mode

Input:
AH = 53h
AL = 02h
BX = Power Device ID
= 0000h BIOS

Output:
AH = 00h Successful
= Error code if unsuccessful
= 02h A real mode interface connection is already established.
= 05h A 16-Bit protected mode interface connection is already established.
= 06h The 16-bit protected mode interface is not supported.
= 07h A 32-Bit protected mode interface connection is already established.
= 09h Device ID unrecognized
AX = APM 16-bit code segment or the real mode segment base address
BX = Offset of the entry point into the BIOS
CF = 0 Successful
= 1 Unsuccessful
CX = APM 16-bit data segment (real mode segment base address)
DI = BIOS code segment length
SI = BIOS data segment length

Description:

This subfunction initializes the 16-bit protected mode interface between the APM Driver (the calling program) and the BIOS. This function must be invoked from real mode. This interface allows a routine making a call in protected mode to invoke BIOS functions without switching into real or Virtual 8086 mode.

Function 53h Subfunction AL = 02h APM 16-Bit Protected Mode Interface Connect, cont'd

Initializing Descriptors

The APM 16-bit protected mode interface uses two consecutive segment/selector descriptors as a 16-bit code and data segment.

The calling program must initialize these descriptors with the segment base and length information returned by this call. The selectors can be in the GDT or LDT and must be valid when the BIOS is called in protected mode.

The code segment descriptor must specify protection level 0. The BIOS function must be invoked with CPL = 0 so the BIOS can execute privileged instructions.

The calling program invokes the BIOS using the 16-bit interface by making a FAR call to the code segment selector that the calling program initialized and the offset returned in BX from this call.

The calling program must supply a stack that can handle both the BIOS and potential interrupt handlers.

The calling program's stack becomes active when interrupts are enabled in the BIOS functions. The BIOS does not switch stacks when interrupts are enabled, including the NMI.

The BIOS 16-bit protected mode interface must be called with a 16-bit stack.

When a BIOS function is called in protected mode, the current I/O permission bitmap must permit access to the I/O ports that the BIOS uses.

cont'd

Function 53h Subfunction AL = 03h APM 32-Bit Protected Mode Interface Connect

Mode: Real Mode

Input: AH = 53h
AL = 03h
BX = Power Device ID
= 0000h BIOS

Output: AH = 00h Successful
= Error code if unsuccessful
= 02h A real mode interface connection is already established.
= 05h A 16-Bit protected mode interface connection is already established.
= 07h A 32-Bit protected mode interface connection is already established.
= 08h The 32-bit protected mode interface is not supported.
= 09h Device ID unrecognized
AX = APM 32-bit code segment or the real mode segment base address
EBX = Offset of the entry point into the BIOS
CF = 0 Successful
= 1 Unsuccessful
CX = APM 16-bit data segment (real mode segment base address)
DI = BIOS code segment length
DX = APM data segment (real mode segment base address)
SI = BIOS data segment length

Description: This subfunction, which must be invoked from real mode, initializes the 32-bit protected mode interface between the APM Driver (the calling program) and the BIOS. This interface allows a routine making a call in protected mode to invoke BIOS functions without switching to real or Virtual 8086 mode.

Function 53h Subfunction AL = 03 APM 32-Bit Protected Mode Interface Connect, cont'd

Initializing Descriptors

The APM 32-bit protected mode interface uses three consecutive segment/selector descriptors as 32-bit code, 16-bit code, and data segment. Both the 32-bit and 16-bit code segment descriptors are needed because the BIOS 32-bit interface can call other BIOS routines.

The calling program must initialize these descriptors with the segment base and length information returned by this call. The selectors can be in the GDT or LDT and must be valid when the BIOS is called in protected mode.

The code segment descriptor must specify protection level 0. The BIOS function must be invoked with CPL = 0 so the BIOS can execute privileged instructions.

The calling program invokes the BIOS using the 32-bit interface by making a FAR call to the 32-bit code segment selector that the calling program initialized and the offset returned in EBX from this call.

The calling program must supply a stack that can handle both the BIOS and potential interrupt handlers.

The calling program's stack becomes active when interrupts are enabled in the BIOS functions. The BIOS does not switch stacks when interrupts are enabled, including the NMI.

The BIOS 32-bit protected mode interface must be called with a 32-bit stack.

When a BIOS function is called in protected mode, the current I/O permission bitmap must permit access to the I/O ports that the BIOS uses.

cont'd

Function 53h Subfunction AL = 04h APM Interface Disconnect

Mode: Real Mode, 16-Bit Protected Mode, 32-Bit Protected Mode

Input:
AH = 53h
AL = 04h
BX = Power Device ID
= 0000h BIOS

Output:
AH = Error code if unsuccessful
= 03h Interface disconnected
= 09h Device ID unrecognized
CF = 0 Successful
= 1 Unsuccessful

Description:

This subfunction:

- disconnects the BIOS and the APM driver,
- restores the BIOS default functions, and
- returns control of power management to the BIOS.

All power management parameters in effect when APM is disconnected will remain in effect.

Function 53h Subfunction AL = 05h CPU Idle

Mode: Real Mode, 16-bit Protected Mode, 32-bit Protected Mode

Input:
AH = 53h
AL = 05h
BX = Power Device ID
= 0000h BIOS

Output:
AH = Error code if unsuccessful
= 03h Interface disconnected
CF = 0 Successful
= 1 Unsuccessful

Description:

Call this function to inform the BIOS that the system is idle. The BIOS will suspend the system until the next system event, which is usually an interrupt. This function permits the BIOS to implement power-saving actions, such as a CPU HLT instruction or slowing the CPU clock.

cont'd

Function 53h Subfunction AL = 06h CPU Busy

Mode: Real Mode, 16-bit Protected Mode, 32-bit Protected Mode

Input:
AH = 53h
AL = 06h
BX = Power Device ID
= 0000h BIOS

Output:
AH = Error code if unsuccessful
= 03h Interface disconnected
CF = 0 Successful
= 1 Unsuccessful

Description:

You only need to invoke this subfunction if *INT 15h AH = 53h Subfunction AL = 05h CPU Idle* was previously invoked. Check bit 2 in CX after invoking *Function 53h Subfunction AL = 00h APM Installation Check* to determine if the BIOS will slow the clock during an *INT 15h AH = 53h Subfunction AL = 05h CPU Idle* call.

This subfunction tells the BIOS that the system is busy. The BIOS restores the CPU clock speed to full speed.

Do not call this function when the CPU is already operating at full speed. While it is not illegal to do so, it adds system overhead.

Function 53h Subfunction AL = 07h Set Power State

Mode: Real Mode, 16-bit Protected Mode, 32-bit Protected Mode

Input:
AH = 53h
AL = 07h
BX = Power Device ID
= 0001h All devices under APM
= 01*xx*h Display (*xx* = unit number). Use *xx* = FF to specify all devices in a class.
= 02*xx*h Secondary storage
= 03*xx*h Parallel ports
= 04*xx*h Serial ports
= E000h – EFFFh OEM-defined device IDs
CX = Power state
0000h APM enabled (not supported for Device ID 0001h)
0001h Standby
0002h Suspend
0003h Off
0004h – 001Fh Reserved system states
0020h – 003Fh OEM-defined system states
0040h – 007Fh OEM-defined device states
0080h – FFFFh Reserved device states

Output: AH = Error code if unsuccessful
= 01h Power management disabled
= 03h Interface disconnected
= 09h Device ID unrecognized
= 0Ah Parameter value out of range
= 60h Unable to enter requested state
CF = 0 Successful
= 1 Unsuccessful

Description:

This subfunction sets the specified power state for the specified device.

cont'd

Function 53h Subfunction AL = 07h Set Power State, cont'd

Examples - System Standby

The following example places the system in Standby State. The calling program invokes this function in response to a *System Standby Request Notification* from the BIOS. The calling program can also invoke this function at any time if it determines that the system is idle and should go to Standby. Standby State is exited when any interrupt occurs.

Input: AH = 53h
AL = 07h
BX = 0001h All devices under APM
CX = 0001h System standby

Function 53h Subfunction AL = 08h Enable Power Management

Mode: Real Mode, 16-bit Protected Mode, 32-bit Protected Mode

Input:
AH = 53h
AL = 08h
BX = Power Device ID
= 0001h All devices under APM
= FFFFh All devices under APM (as specified in the APM 1.0 specification)
CX = Function code
= 0000h Disable power management
= 0001h Enable power management

Output:
AH = Error code if unsuccessful
= 01h Power management disabled
= 03h Interface disconnected
= 09h Device ID unrecognized
= 0Ah Parameter value out of range
CF = 0 Successful
= 1 Unsuccessful

Description:

This subfunction enables (or disables) automatic power down. When disabled, the BIOS does not automatically power devices down, enter Suspend State, enter the Standby State, or perform any power-saving steps in response to Function 53h Subfunction AL = 05h CPU Idle calls.

cont'd

Function 53h Subfunction AL = 09h Restore BIOS Power-On Defaults

Mode: Real Mode, 16-bit Protected Mode, 32-bit Protected Mode

Input:
AH = 53h
AL = 09h
BX = Power Device ID
= 0001h All devices under APM
= FFFFh All devices under APM (as specified in the APM 1.0 documents)

Output:
AH = Error code if unsuccessful
= 03h Interface disconnected
= 09h Device ID unrecognized
CF = 0 Successful
= 1 Unsuccessful

Description:

This subfunction reinitializes the BIOS power-on default values.

Function 53h Subfunction AL = 0Ah Get Power Status

Mode: Real Mode, 16-bit Protected Mode, 32-bit Protected Mode

Input:
AH = 53h
AL = 0Ah
BX = Power Device ID
= 0001h BIOS

Output:
AH = Error code if unsuccessful
= 09h Device ID unrecognized
BH = Line status
= 00h Offline
= 01h Online
= 02h On backup power
= FFh Unknown
BL = Battery status
= 00h High
= 01h Low
= 02h Critical
= 03h Charging
= FFh Unknown
CF = 0 Successful
= 1 Unsuccessful
CL = Remaining battery life (percentage of charge)
= 0 – 100 Percentage of full charge
= 255 Unknown
DX = Remaining battery life (time units)
= Bit 15 0 Time unit is seconds
1 Time unit is minutes
= Bits 14-0 Number of seconds or minutes of battery life left
0000h – 7FFFh Valid number
FFFFh Unknown

Description:

This subfunction returns the current system power status.

cont'd

Function 53h Subfunction AL = 0Bh Get PM Event

Mode: Real Mode, 16-bit Protected Mode, 32-bit Protected Mode

Input: AH = 53h
AL = 0Bh

Output: AH = Error code if unsuccessful
= 03h Interface disconnected
= 80h No power management events pending
CF = 0 Successful
= 1 Unsuccessful

Description:

This subfunction returns the next power management event or indicates that no power management events are pending. Power management events can apply to a device or to the APM system.

This subfunction should be invoked until no power management events are pending or an error occurs.

Function 53h Subfunction AL = 0Ch Get Power State

Mode: Real Mode, 16-bit Protected Mode, 32-bit Protected Mode

Input:
AH = 53h
AL = 0Ch
BX = Power Device ID
= 0001h All devices under APM
= 01*xx*h Display (*xx* is the unit number). Specify *xx* = FF to include all devices in a class.
= 02*xx*h Secondary storage (*xx* is the unit number).
= 03*xx*h Parallel ports (*xx* is the unit number).
= 04*xx*h Serial ports (*xx* is the unit number).
= E00h – EFFFh OEM-defined power device IDs.

Output:
AH = Error Code if unsuccessful
= 01h Power management disabled
09h Device ID unrecognized
CF = 0 Successful
= 1 Unsuccessful
CX = 0000h APM enabled
= 0001h Standby
= 0001h Suspend
= 0003h Off
= 0004h – 001Fh Reserved system states
= 0020h – 003Fh OEM-defined system states
= 0040h – 007Fh OEM-defined device states
= 0080h – FFFFh Reserved device states

Description:

This subfunction returns the device power state for a specific Device ID. *0001h All devices under APM* or *all devices in a class* (xFFxh) is returned for the specified Power Device ID when that device has been used in an *AL = 07h Set Power State* call. When the power device ID has not been used in an *AL = 07h Set Power State* call, this function is unsuccessful and returns AH = 09h Device ID unrecognized. Use this subfunction to find out if BIOS power management is enabled for a device. This subfunction returns AH = 01h if BIOS power management is disabled for a device.

cont'd

Function 53h Subfunction AL = 0Dh Enable Device Power Management

Mode: Real Mode, 16-bit Protected Mode, 32-bit Protected Mode

Input:
AH = 53h
AL = 0Dh
BX = Power Device ID
= 0001h All devices under APM
= 01*xx*h Display (*xx* is the unit number). Specify *xx* = FF to include all devices in a class.
= 02*xx*h Secondary storage (*xx* is the unit number).
= 03*xx*h Parallel ports (*xx* is the unit number).
= 04*xx*h Serial ports (*xx* is the unit number).
= E00h – EFFFh OEM-defined power device IDs.
CX = Function code
= 0000h Disable power management
= 0001h Enable power management

Output:
AH = Error code if unsuccessful
= 01h Power management disabled
= 03h Interface disconnected
= 09h Device ID unrecognized
= 0Ah Parameter value out of range
CF = 0 Successful
= 1 Unsuccessful

Description:

This subfunction enables (or disables) automatic power down for the specified device. When disabled, the BIOS does not automatically power the device down.

Function 53h Subfunction AL = 80h BH = 7Fh APM Installation Check (OEM-Defined APM Functions)

Mode: Real Mode, 16-bit Protected Mode, 32-bit Protected Mode

Input:
AH = 53h
AL = 80h
BH = 7Fh OEM APM installation check

Output:
AH = Error code if unsuccessful
= 03h Interface disconnected
BX = OEM ID
CF = 0 Successful
= 1 Unsuccessful
CX = Optional OEM-Specific information
DX = Optional OEM-Specific information

Description: Call this subfunction to find out if the BIOS supports OEM hardware-dependent functions.

cont'd

Function 53h Subfunction AL = 80h BH = OEM-Defined Function Code

Mode: Real Mode, 16-bit Protected Mode, 32-bit Protected Mode

Input:

AH	=	53h
AL	=	80h
BH	=	00h – 7Eh OEM-Defined function code
	=	80h – FFh OEM-Defined function code

Output:

AH	=	Error code if unsuccessful
	=	03h Interface disconnected
CF	=	0 Successful
	=	1 Unsuccessful
CX	=	Optional OEM-Specific information
DX	=	Optional OEM-Specific information

Description:

Call this subfunction to access OEM product-specific APM functions.

INT 15h Power Management Error Codes

These error codes appear in AH after a function call.

AH	Description	Generated by
01h	Power management disabled	AL = 07h Set Power State AL = 08h Enable Power Management AL = 0Ah Get Power Status AL = 0Dh Enable Device Power Management
02h	Real mode interface connection already established	AL = 01h APM Real Mode Interface Connect AL = 02h APM 16-Bit Protected Mode Interface Connect AL = 03h APM 32-Bit Protected Mode Interface Connect
03h	Interface disconnected	AL = 04h APM Interface Disconnect AL = 05h CPU Idle AL = 06h CPU Busy AL = 07h Set Power State AL = 08h Enable Power Management AL = 09h Restore BIOS Power-On Defaults AL = 0Bh Get PM Event AL = 0Dh Enable Device Power Management AL = 80h OEM APM Function
05h	16-bit protected mode interface already established	AL = 01h APM Real Mode Interface Connect AL = 02h APM 16-Bit Protected Mode Interface Connect AL = 03h APM 32-Bit Protected Mode Interface Connect
06h	16-bit protected mode interface not supported	AL = 02h APM 16-Bit Protected Mode Interface Connect
07h	32-bit protected mode interface already established	AL = 01h APM Real Mode Interface Connect AL = 02h APM 16-Bit Protected Mode Interface Connect AL = 03h APM 32-Bit Protected Mode Interface Connect
08h	32-bit protected mode interface not supported	AL = 03h APM 32-Bit Protected Mode Interface Connect

AH	Description	Generated by
09h	Device ID Unrecognized	AL = 01h APM Real Mode Interface Connect AL = 02h APM 16-Bit Protected Mode Interface Connect AL = 03h APM 32-Bit Protected Mode Interface Connect AL = 04h APM Interface Disconnect AL = 07h Set Power State AL = 08h Enable Power Management AL = 09h Restore BIOS Power-On Defaults AL = 0Ah Get Power Status AL = 0Ch Get Power State AL = 0Dh Enable Device Power Management
0Ah	Parameter values out of range	AL = 07h Set Power State AL = 08h Enable Power Management AL = 0Dh Enable Device Power Management
60h	Unable to enter requested state	AL = 07h Set Power State
80h	No power management events pending	AL = 0Bh Get PM Event
86h	Reserved. No APM present.	

Function 80h Device Open

Input: AH = 80h
BX = Device ID
CX = Process ID

Output: Programmer-defined

Description: Functions 80h, 81h, and 82h can be used to handle multiprocessing systems. The system program manager can trap these interrupt functions and provide individual service routines for these operations. The routine provided by the programmer for Function 81h should detach a logical device from a specified process.

Function 81h Device Close

Input: AH = 81h
BX = Device ID
CX = Process ID

Output: Programmer-defined

Description:

Functions 80h, 81h, and 82h can be used to handle multiprocessing systems. The system program manager can trap these interrupt functions and provide individual service routines for these operations. The routine provided by the programmer for Function 81h should detach a logical device from a specified process.

Function 82h Process Termination

Input: AH = 82h
BX = Process ID

Output: Programmer-defined

Description:

Functions 80h, 81h and 82h can be used to handle multiprocessing systems. The system program manager can trap these interrupt functions and provide individual service routines for these operations. The routine provided by the programmer for Function 82h should terminate a process.

cont'd

Function 83h Event Wait

Input: AH = 83h
AL = 00h Request Wait or 01h Cancel Wait
CX:DX = Number of Microseconds
ES:BX = Pointer to a Flag

Output: AH = 00h
AL = Value written to CMOS RAM Register B if successful.
= 00h Function is busy
CF = 0 No error
= 1 Function is busy

Description: Function 83h sets a flag after a specified number of µseconds has elapsed. Bit 7 of the first byte at ES:BX is set after the wait has expired. The µseconds to delay must be a multiple of 976.

Function 84h Joystick Support

Input: AH = 84h
DX = 00h Read Current Switch Settings
01h Read Resistive Inputs

Output: If DX was set to 0:
AL = Bits 7–4 Switch Settings
Bits 3–0 Reserved
If DX was set to 1:
AX = Joystick A x coordinate
BX = Joystick A y coordinate
CX = Joystick B x coordinate
DX = Joystick B y coordinate
CF = 0 No error
= 1 Value in DX is incorrect

Description: Function 84h reads the switches and inputs of a joystick attached via a game adapter. If a game adapter is not installed, 00h is returned.

Function 85h SysReq Key Handler

Input: AH = 85h
AL = 00h Key Make (Depressed)
01h Key Break (Released)

Output: Programmer-defined

Description:

A multitasking operating system can use Function 85h to see when the SysReq key is pressed or released. The programmer can trap this function and provide another service routine. The BIOS returns AH = 00h and the Carry Flag is set to 0.

Function 86h Wait Function

Input: AH = 86h
CX:DX = Number of Microseconds to Wait

Output: CF = 0 No error
= 1 Error

Description:

Function 86h delays the system for a specified number of microseconds.

cont'd

Function 87h Move Extended Memory Block

Input: AH = 87h
CX = Number of Words to Move
ES:SI = Address of Descriptor Table

Output: AH = 00h No error
= 01h RAM Parity Error (Parity Error Cleared)
= 02h Exception INT Error
= 03h Gate Address 20 (GA20) Failed
CF = 0 No error
= 1 Error

Description:

Function 88h moves data between conventional (DOS) memory and extended memory. It uses a Global Descriptor Table (GDT) in the following format (all offsets are with respect to ES:SI):

Offset	Entry Description
00h – 07h	Dummy entry, should be all zeroes.
08h – 0Fh	GDT entry (ES:SI)
10h – 17h	Source GDT entry
18h – 1Fh	Destination GDT entry
20h – 27h	Temporary BIOS CS entry
28h – 2Fh	Temporary SS area

Initialize the source GDT and destination GDT entries. All other entries should be initialized to zero. Interrupts are disabled while this function is performed.

Function 88h Return Size of Extended Memory

Input: AH = 88h

Output: AX = Number of contiguous 1 KB Blocks of Extended Memory

Description: Function 88h returns the size of extended memory (memory above 1 MB) installed on the system. The number of 1 KB blocks is specified in AX.

Function 89h Switch to Protected Mode

Input: AH = 89h
BH = Offset to Interrupt Descriptor Table that points to the beginning of the first eight hardware interrupts.
BL = Offset to Interrupt Descriptor Table that points to the beginning of the next eight hardware interrupts.
ES:SI = Address of Descriptor Table.

Output: AH = 00h No error
= FFh Error
CF = 0 No error
= 1 Error

Description:

Function 89h switches the microprocessor to protected mode from real mode. In the *IBM PC/AT Technical Reference Manual,* protected mode was called virtual mode.

cont'd

Function 89h Switch to Protected Mode, cont'd

GLobal Descriptor Table

Initialize a Global Descriptor Table (GDT) as follows. All offsets are with respect to ES:SI.

Offset	Table Entry
00h – 07h	Dummy entry, should be all zeroes.
08h – 0Fh	Pointer to GDT.
10h – 17h	Interrupt Descriptor Table (IDT) entry.
18h – 1Fh	Programmer-defined DS entry.
20h – 27h	Programmer-defined ES entry.
28h – 2Fh	Programmer-defined SS entry.
30h – 37h	Programmer-defined CS entry.
38h – 3Fh	Temporary BIOS CS entry.

The programmer should initialize the GDT, IDT, DS, ES, SS, and CS entries. The temporary BIOS CS entry should be zero. The dummy entry should be all zeroes.

The entry at offset 08h is actually a pointer to the GDT table. Its value consists of the physical address derived from ES:SI (pointer to GDT = ((ES * 10) + SI)) and the segment limit (length of the GDT). For additional information on Global Descriptor Tables, see the *Intel i486 Programmers Reference Manual*.

Function 90h Device Busy Loop

Input: AH = 90h
AL = Device Type Code
00h Hard disk drive
01h Floppy disk drive
02h Keyboard
03h PS/2-type mouse
80h Network
FCh Hard disk reset
FDh Floppy disk drive motor
FEh Printer
ES:BX = Pointer to a request block if AL = 80h–FFh (a reentrant device).

Output: Programmer-defined

Description:

Function 90h is provided for system-level device drivers to perform a wait for I/O completion. The service routine is provided by the drivers. Serially reusable devices must be given device types from 00h – 7Fh. Reentrant devices must have a type between 80h and BFh. Wait-only calls that have no corresponding INT 15h Function 91h Interrupt Complete call must have device types C0h – FFh.

cont'd

Function 91h Interrupt Complete

Input: AH = 91h
AL = Device Type Code
00h Hard disk drive
01h Floppy disk drive
02h Keyboard
03h PS/2-type mouse
80h Network
FCh Hard disk reset
FDh Floppy disk drive motor
FEh Printer
ES:BX = Points to a request block if AL = 80h – FFh (a reentrant device).

Output: Programmer-defined

Description:

Function 91h is provided for system-level device drivers to signal that I/O has been completed. The service routine is provided by the drivers.

Function C0h Return System Configuration Parameter

Input: AH = C0h

Output: AH = 00h No error
AH = 86h
CF = 0 No error
ES:BX = Address of Configuration Parameter Table

Function C0h Return Configuration Parameter, cont'd

Description: Function C0h returns a pointer to the System Configuration Table. The format of this table is:

Offset	Initial Value	Description
00h – 01h		Number of Bytes in this table (must be at least 8)
02h	FCh	Model Byte (always FCh).
03h	01h	Submodel Byte (always 01h).
04h		BIOS Revision Level
05h		Feature Information Byte Bit 7 DMA channel 3 used if set Bit 6 Interrupt controllers cascaded if set Bit 5 Real time clock available if set Bit 4 Keyboard intercept (INT 15h Function 4Fh) available if set Bits 3–0 Reserved, should be zeroes.
06h – 09h		Reserved

Since this book deals only with AMIBIOS for ISA and EISA systems, the value for byte 02h is FCh and for 03h is 01h. These values are the same for all ISA and EISA computers.

Function C1h Return Address of Extended BIOS Data Area

Input: AH = C1h

Output: CF = 0 No error
= 1 Error
ES = Segment of Extended BIOS Data Area

Description: Function C1h returns the segment of the extended BIOS data area.

cont'd

Function C2h PS/2 Mouse Support

Function C2h, originally defined in the PS/2 specification, controls a PS/2-type mouse or pointing device. Support for a PS/2-type mouse is provided by the AMIBIOS if the system has the proper hardware and an American Megatrends Keyboard Controller BIOS version F (KF) or later. See page 361 for more information about the Keyboard Controller BIOS.

Function C2h Subfunction 00h Enable or Disable Mouse

Input: AH = C2h
AL = 00h
BH = 00h Disable
= 01h Enable

Output: AH = 00h No error
= 01h Invalid subfunction number
= 02h Invalid input values
= 03h Mouse interface error
= 04h Resend required
= 05h Far call is not installed
CF = 0 No error
= 1 Error

Description:

INT 15h Function C2h Subfunction 00h enables or disables the mouse.

Function C2h Subfunction 01h Reset Mouse

Input: AH = C2h
AL = 01h

Output: AH = 00h No error
= 01h Invalid subfunction number
= 02h Invalid input values
= 03h Mouse interface error
= 04h Resend required
= 05h Far call is not installed
CF = 0 No error
= 1 Error

Description:

INT 15h Function C2h Subfunction 01h resets the mouse and sets the sample rate, resolution, and other attributes to the default values. The mouse is also disabled by default.

The default settings are:

Parameter	Disabled State
Mouse	Disabled
Sample Rate	100 samples per second
Resolution	4 counts per millimeter
Data package size	unchanged
Scaling	1:1

cont'd

Function C2h Subfunction 02h Set Sample Rate

Input: AH = C2h
AL = 02h
BH = 00h 10 samples per second
01h 20 samples per second
02h 40 samples per second
03h 60 samples per second
04h 80 samples per second
05h 100 samples per second (default)
06h 200 samples per second

Output: AH = 00h No error
= 01h Invalid subfunction number
= 02h Invalid input values
= 03h Mouse interface error
= 04h Resend required
= 05h Far call is not installed
CF = 0 No error
= 1 Error

Description:

INT 15h Function C2h Subfunction 02h sets the mouse sample rate. The default sample rate is 100 samples per second.

Function C2h Subfunction 03h Set Resolution

Input: AH = C2h
AL = 03h
BH = 00h 1 count per millimeter
01h 2 counts per millimeter
02h 4 counts per millimeter (default)
03h 8 counts per millimeter

Output: AH = 00h No error
= 01h Invalid subfunction number
= 02h Invalid input values
= 03h Mouse interface error
= 04h Resend required
= 05h Far call is not installed
CF = 0 No error
= 1 Error

Description: INT 15h Function C2h Subfunction 03h sets the mouse resolution rate. The default is 4 counts per millimeter.

Function C2h Subfunction 04h Return Mouse Type

Input: AH = C2h
AL = 04h

Output: AH = 00h No error
= 01h Invalid subfunction number
= 02h Invalid input values
= 03h Mouse interface error
= 04h Resend required
= 05h Far call is not installed
BH = Device ID
CF = 0 No error
= 1 Error

Description: INT 15h Function C2h Subfunction 04h returns the mouse device ID number.

cont'd

Function C2h Subfunction 05h Initialize Mouse Interface

Input: AH = C2h
AL = 05h
BH = Data Packet Size (1 to 8, representing 1 – 8 bytes)

Output: AH = 00h No error
= 01h Invalid subfunction number
= 02h Invalid input values
= 03h Mouse interface error
= 04h Resend required
= 05h Far call is not installed
CF = 0 No error
= 1 Error

Description:

INT 15h Function C2h Subfunction 05h performs the same operations as Subfunction 01h, but it also sets the data packet size of the mouse interface. The same default values specified in subfunction 01h are used and the packet size must be in BH.

The default settings are:

Parameter	Disabled State
Mouse	Disabled
Sample Rate	100 samples per second
Resolution	4 counts per millimeter
Data package size	unchanged
Scaling	1:1

Function C2h Subfunction 06h Mouse Status or Set Scaling Factor

Input: AH = C2h
AL = 06h
BH = 00h Return mouse status
= 01h Set 1:1 scaling factor
= 02h Set 2:1 scaling factor

Output: AH = 00h No error
= 01h Invalid subfunction number
= 02h Invalid input values
= 03h Mouse interface error
= 04h Resend required
= 05h Far call is not installed
BL = Status Byte (If BH was 00h, BL is the status byte)
Bit 7 Reserved
Bit 6 0 Stream mode is used
1 Remote mode is used
Bit 5 0 Mouse disabled
1 Mouse enabled
Bit 4 0 1:1 scaling is used
1 2:1 scaling is used
Bit 3 Reserved
Bit 2 1 Left button pressed
Bit 1 Reserved
Bit 0 1 Right button pressed
CF = 0 No error
= 1 Error
CL = Resolution rate
00h 1 count per millimeter
01h 2 counts per millimeter
02h 4 counts per millimeter
03h 8 counts per millimeter

cont'd

Function C2h Subfunction 06h Mouse Status or Set Scaling Factor, cont'd

Output:, cont'd

DL = Sample rate

0Ah	10 samples per second
14h	20 samples per second
28h	40 samples per second
3Ch	60 samples per second
50h	80 samples per second
64h	100 samples per second
C8h	200 samples per second

Description:

INT 15h Function C2h Subfunction 06h can be used to ascertain the mouse status or to set the mouse scaling factor.

Function C2h Subfunction 07h Set Mouse Handler Address

Input: AH = C2h
AL = 07h
ES:BX = Address of Programmer Routine

Output: AH = No error
= 01h Invalid subfunction number
= 02h Invalid input values
= 03h Mouse interface error
= 04h Resend required
= 05h Far call is not installed
CF = 0 No error
= 1 Error

Description:

This subfunction attaches a programmer-supplied mouse routine to the BIOS mouse service routine. When the BIOS routine receives data from the mouse, the programmer-supplied routine is called by the BIOS. Place the following four parameters on the stack before calling this function:

Address	Description
SS:SP + 0Ah	Status word Bits 15–8 Reserved Bit 7 y coordinate has overflowed if set to 1 Bit 6 x coordinate has overflowed if set to 1 Bit 5 y coordinate is negative if set to 1 Bit 4 x coordinate is negative if set to 1 Bits 3–2 Reserved. Bit 3 should be 1 and Bit 2 should be 0. Bit 1 Right button pressed if set to 1 Bit 0 Left button pressed if set to 1
SS:SP + 08h	x coordinate
SS:SP + 06h	y coordinate
SS:SP + 04h	z coordinate (should be 00h)

The programmer-supplied routine should exit via a far return and must not remove the parameters from the stack.

cont'd

Function C3h Fail-Safe Timer Control

Input: AH = C3h
AL = 00h Disable fail-safe timer
= 01h Enable fail-safe timer
BX = Fail-safe timer value

Output: CF = 0 No error
= 1 Error

Description:

INT 15h Function C3h enables or disables the EISA fail-safe timer. When enabled, the value in BX becomes the timer count value. The fail-safe timer is placed in mode 0 operation, the fail-safe timer NMI is enabled, and the value in BX is copied to the BIOS extended data area. CF is set if there is an invalid input.

When disabled, the fail-safe timer value in the BIOS extended data area is cleared.

Function D8h EISA Support

Function D8h configures EISA controllers and stores values in EISA Extended CMOS RAM. This function is the only way in which EISA Extended CMOS RAM should be accessed.

This function has four subfunctions that are primarily used by the EISA Configuration Utility (ECU) with the Configuration (CFG) files supplied by EISA product manufacturers with EISA adapter cards and motherboards.

All EISA subfunctions (00h/80h through 04h/84h) are described in this section. Functions 00 – 04h are used for 16-bit cards. Functions 80h – 84h are used for 32-bit cards. Improper use of these subfunctions could cause an EISA system to operate erratically.

EISA Extended CMOS RAM

EISA-specific configuration data is stored in I/O-mapped EISA Extended CMOS RAM. There must be at least 4 KB of EISA Extended CMOS RAM, in addition to the required 64 bytes of ISA CMOS RAM.

EISA Devices

Any controller in an EISA system can be called an EISA device. There can be up to 64 devices in an EISA system: 16 physical devices and 48 virtual (logical) devices.

EISA Devices and Slots

EISA controllers and EISA devices are essentially the same. EISA slots are used as addresses in EISA systems and are the actual physical expansion slots on the EISA motherboard. EISA devices are addressed by their physical or logical slot number. The EISA motherboard is always Slot 0. The physical slots are 1 – 15.

cont'd

EISA Device Number

A physical device resides in an actual expansion slot on the EISA motherboard and is numbered 1 – 15. This number is the EISA device number.

Embedded Devices

The motherboard can have one or more devices on it that are called embedded devices, which are also EISA devices. Embedded device numbers begin after the last physical device number. If the last physical device is 7, then the first embedded device is 8.

Virtual Devices

A virtual device is often a software device driver that uses system resources but does not physically exist. ISA devices on the motherboard can be virtual devices. Virtual devices are numbered sequentially after the last physical or embedded device. If the last physical or embedded device is 6, then the first virtual device is 7.

Device Functions

A device can have more than one function. Some standard functions are: memory, serial port, parallel port, floppy disk, and hard disk.

Function D8h Subfunction 00h (80h) Read Slot Configuration Information

Input: AH = D8h
AL = 00h (for 16-bit addressing)
= 80h (for 32-bit addressing)
CL = Slot Number (virtual and embedded devices included)
00h Motherboard
01h Slot 1
... ...
0Fh Slot 15

Function D8h Subfunction 00h (80h) Read Slot Configuration Information, cont'd

Output: AH = 00h No error
= 80h Invalid slot number
= 81h Invalid function number
= 82h EISA Extended CMOS RAM is corrupt
= 83h Slot is empty
= 86h Invalid BIOS call
= 87h Invalid system configuration

AL = CFG and Slot Status
Bit 7 0 Duplicate CFG ID not found.
1 Duplicate CFG ID found.
Bit 6 0 Product ID was readable.
1 Product ID was not readable.
Bits 5–4 00b Slot is an expansion slot.
01b Slot is an embedded device.
10b Slot is a virtual device.
11b Reserved
Bits 3–0 0000b No duplicate CFG ID found.
0001b First duplicate CFG ID used.
0010b Second duplicate CFG ID used.
... ...
1111b Fifteenth duplicate CFG ID used.

BH = Major Revision Level of ECU
BL = Minor Revision Level of ECU
CF = 0 No error
= 1 Error
CH = MSB of CFG Checksum
CL = LSB of CFG Checksum
DH = Number of Device Function

cont'd

Function D8h Subfunction 00h (80h) Read Slot Configuration Information, cont'd

Output: DL = Combined Function Information Byte
- Bits 7–6 Reserved
- Bit 5 Slot has one or more port initialization entries if this bit is set.
- Bit 4 Slot has one or more port range entries if this bit is set.
- Bit 3 Slot has one or more DMA entries if this bit is set.
- Bit 2 Slot has one or more IRQ entries if this bit is set.
- Bit 1 Slot has one or more memory entries if this bit is set.
- Bit 0 Slot has one or more function type entries if this bit is set.

DI (LSB) = Byte 0 of compressed ID
DI (MSB) = Byte 1 of compressed ID
SI (LSB) = Byte 2 of compressed ID
SI (MSB) = Byte 3 of compressed ID

Description:

Function D8h Subfunction 00h returns EISA configuration information for a specified slot by reading information directly from EISA Extended CMOS RAM. The slots can be the motherboard, an adapter card, an embedded device, or a virtual device. Each slot has a corresponding CFG file that is used by the ECU to configure the slot properly.

Duplicate CFG Files

If the system finds that more than one CFG file exists for the specified slot, a duplicate ID condition occurs and bit 8 of AL is set. Bits 3 to 0 of AL indicate the duplicate ID that was used.

Function D8h Subfunction 00h (80h) Read Slot Configuration Information, cont'd

Device ID Number

DI and SI contain a four-byte compressed ID number pertaining to the device installed in the specified slot. This number identifies the manufacturer of the device, the device product number, and the product revision number.

Product ID	Description
DI (LSB)	Bit 7 Reserved, should be zero. Bits 6–2 First character of the manufacturer code. Bits 1–0 First two bits of second character of the manufacturer code.
DI (MSB)	Bits 7–5 Remaining three bits of second character of the manufacturer code. Bits 4–0 Third character of the manufacturer code.
SI (LSB)	Adapter card: Bits 7–4 First hex digit of the manufacturer's product number. Bits 3–0 Second hex digit of the manufacturer's product number. Motherboard: Bits 7–0 Reserved for manufacturer.
SI (MSB)	Adapter card: Bits 7–4 Third hex digit of the manufacturer's product number. Bits 3–0 Product revision number Motherboard: Bits 7–3 Reserved for manufacturer's use. Bits 2–0 EISA bus version number (001 in initial version). 001 is currently the only standard value defined for this field, but, in practice, EISA motherboard and adapter card manufacturers have been using this field for their own purposes.

cont'd

Function D8h Subfunction 01h (81h) Read Function Configuration Information

Input:

AH	=	D8h
AL	=	01h (for 16-bit addressing)
	=	81h (for 32-bit addressing)
CH	=	Function Number (from 0 through *m* - 1, where *m* = the contents of DH from Subfunction 00h)
CL	=	Slot Number (virtual and embedded devices included)
		00h EISA Motherboard
		01h Slot 1
		02h Slot 2
		... ...
		... ...
		0Fh Slot 15
DS:SI	=	Address of Data Buffer (16-bit addressing)
DS:ESI	=	Address of Data Buffer (32-bit addressing)

Output:

AH	=	00h No error
	=	80h Invalid slot number
	=	81h Invalid function number
	=	82h EISA Extended CMOS RAM is corrupt
	=	83h Slot is empty
	=	86h Invalid BIOS call
	=	87h Invalid system configuration
CF	=	0 No error
	=	1 Error
DS:SI	=	Return data buffer address (if 16-bit call)
DS:ESI	=	Return data buffer address (if 32-bit call)

Function D8h Subfunction 01h (81h) Read Function Configuration Information, cont'd

Description:

Function D8h Subfunction 01h reads the specified function information directly from CMOS RAM. The calling software can find the number of functions for a particular device using subfunction 00h (80h).

With subfunction 01h (81h), the caller receives information about each specific device function. This subfunction reads a 320-byte table and then writes this table to the memory buffer address specified in DS:SI. Each block of a variable-length data field describes an individual EISA adapter card. The table format is:

Offset	Description
00h	First Byte of Compressed ID Bit 7 Reserved, should be zero. Bits 6–2 First character of the manufacturer code. Bits 1–0 First two bits of second character of the manufacturer code.
01h	Second Byte of Compressed ID Bits 7–5 Remaining three bits of second character of the manufacturer code. Bits 4–0 Third character of the manufacturer code.
02h	Third Byte of Compressed ID Adapter card: Bits 7–4 First hex digit of the manufacturer's product number. Bits 3–0 Second hex digit of the manufacturer's product number. Motherboard: Bits 7–0 Reserved for manufacturer's use.
03h	Fourth Byte of Compressed ID Adapter card: Bits 7–4 Third hex digit of the manufacturer's product number. Bits 3–0 Product revision number Motherboard: Bits 7–3 Reserved for manufacturer's use. Bits 2–0 EISA bus version number (001 is initial version).

Offset	Description
04h – 05h	ID and Slot Information Byte 0 Bit 7 0 No duplicate ID is present. 1 Duplicate ID found. Bit 6 0 ID is readable. 1 ID is unreadable. Bits 5–4 Device Type 00b Expansion device 01b Embedded device 10b Virtual device Bits 3–0 Number of Duplicate CFG filenames 0000b No duplicate CFG 0001b First duplicate CFG ... ... 1110b Fourteenth duplicate CFG 1111b Fifteenth duplicate CFG Byte 1 Bit 7 0 Configuration is successful. 1 Configuration is unsuccessful. Bits 6–2 Reserved, should be zeros. Bit 1 0 EISA IOCHKERR not supported. 1 EISA IOCHKERR supported. Bit 0 0 EISA ENABLE not supported (adapter card cannot be enabled or disabled). 1 EISA ENABLE supported (adapter card can be enabled or disabled). The EISA specification allows EISA adapter cards to be enabled or disabled via software. If bit 0 of byte 1 above is set, external software can disable the adapter card. Similarly, the availability of IOCHKERR allows external software to check expansion slots for pending errors.
06h – 07h	Revision levels of the CFG overlay files used for a specified slot. Both bytes are 0 if no overlay file exists. Byte 0 Minor revision level of the CFG overlay file. Byte 1 Major revision level of the CFG overlay file.
08h – 21h	Selections made by the system ECU. The possible choices for the specified slot function are counted here. The actual names of the choices are specified in the CFG file. Byte 0 Selection 1 Byte 1 Selection 2 ... ... Byte 24 Selection 25 Byte 25 Selection 26
22h	Slot function information Bit 7 0 Slot function is enabled. 1 Slot function is disabled. Bit 6 CFG is using free form data if set. Bit 5 Port initialization entry(s) follows if set. Bit 4 Port range entry(s) follows if set. Bit 3 DMA entry(s) follows if set. Bit 2 IRQ entry(s) follows if set. Bit 1 Memory entry(s) follows if set. Bit 0 Type and Subtype string follows if set.

Offset	Description
23h – 62h	80-character ASCII string describing the slot device. The string has types and subtypes. The manufacturer determines the type and subtype format, but the following conventions are often used: **Type String** COM Communications device KEY Keyboard MEM Memory card MFC Multifunction card MSD Mass storage device NET Network card NPX Math coprocessor OSE Operating system or environment OTH Other PAR Parallel port PTR Pointing device SYS Motherboard VID Video adapter card , Delimiter for Type string fragments ; End of Type string and beginning of Subtype string 0 End of Subtype strings The unused part of the 80-character string should be zero (not including the Subtype delimiter).
73h – B1h	Memory Configuration Section. Nine seven-byte entries: Byte 0 Memory Configuration Byte Bit 5 0 Memory is not shared 1 Memory is shared Bits 4–3 00b SYS (base/extended memory) 01b EXP (expanded memory) 10b VIR (virtual memory) 11b OTH (other memory) Bit 1 0 Memory is not cached 1 Memory is cached Bit 0 0 Memory is ROM (read only) 1 Memory is RAM (read and write) Byte 1 Memory Data Size Bits 3–2 Decode Size 00b 20 address lines 01b 24 address lines 10b 32 address lines Bits 1–0 Data Access Size 00b Byte 01b Word (16 bits) 10b Doubleword (32 bits) Bytes 2–4 Starting Memory Address divided by 100h Bytes 5–6 Memory Size divided by 400. If 0000h, memory size is 64 MB. Size is specified in 1024 byte increments.

Offset	Description
B2h – BFh	Hardware Interrupt Configuration Section. Seven two-byte entries: Byte 0 Bit 6 0 Interrupt is not shared 1 Interrupt is shared Bit 5 0 Interrupt is edge-triggered 1 Interrupt is level-triggered Bits 3–0 Interrupt number 0000b IRQ0 0001b IRQ1 ... ... 1110b IRQ14 1111b IRQ15 Byte 1 Reserved, should be zero.
C0h – C7h	DMA Channel Description Section. Four two-byte entries: Byte 0 Bit 6 0 DMA channel is not shared 1 DMA channel is shared Bits 5–3 Reserved, should be zeros. Bits 2–0 DMA Channel Number 000b Channel 0 001b Channel 1 ... ... 110b Channel 6 111b Channel 7 Byte 1 Bits 7–6 Reserved, should be zeros. Bits 5–4 DMA Timing 00b ISA-compatible timing 01b Type A timing 10b Type B timing 11b Type C (Burst) timing Bits 3–2 DMA Transfer Size 00b Byte transfers 01b Word transfers (16 bits) 10b Doubleword transfers (32 bits) Bits 1–0 Reserved, should be zeros.
C8h – 103h	I/O Port Information consists of 20 three-byte entries: Byte 0 Bit 6 0 Port is not shared 1 Port is shared Bit 5 Reserved, should be zero. Bits 4–0 Number of Ports (starting at 0) 00000b One port 00001b Two sequential ports 00010b Three sequential ports ... ... 11110b Thirty-one sequential ports 11111b Thirty-two sequential ports Byte 1 LSB of I/O Port Address Byte 2 MSB of I/O Port Address

Offset	Description
104h – 13Fh	I/O Port Initialization Data Section. Entries vary in length. Byte 0 Initialization Type Bits 6–3 Reserved, should be zeros. Bit 2 0 Write value to port 1 Use both mask and value Bits 1–0 Data Access Size *If Byte 0, bit 2 is 0, the following format is used:* 00b Byte 3 is the initialization value. 01b Byte 3 is the LSB of the initialization value. Byte 4 is the MSB of the initialization value. 10b Byte 3 is the LSB of the initialization value. Byte 4 is the second byte of the initialization value. Byte 5 is the third byte of the initialization value. Byte 6 is the MSB of the initialization value. *If Byte 0, bit 2 is 1, the following format is used:* 00b Byte 3 is the initialization value. Byte 4 is mask value. 01b Byte 3 is the LSB of the initialization value. Byte 4 is the MSB of the initialization value. Byte 5 is the LSB of the mask value. Byte 6 is the MSB of the mask value. 10b Byte 3 is the LSB of the initialization value. Byte 4 is the second byte of the initialization value. Byte 5 is the third byte of the initialization value. Byte 6 is the MSB of the initialization value. Byte 7 is the LSB of the mask value. Byte 8 is the second byte of the mask value. Byte 9 is the third byte of the mask value. Byte 10 is the MSB of the mask value. Byte 1 LSB of Port Address Byte 2 MSB of Port Address

Note: If bit 6 of the Function Information Section (22h) is set, the table is not in the table format described above, but uses free-form data. Entries through Type and Subtype (23h) are the same, but starting at 73h, the data in the table is in the board manufacturer's proprietary format.

cont'd

Function D8h Subfunction 02h (82h) Clear EISA CMOS RAM

Input: AH = D8h
AL = 02h (for 16-bit addressing)
= 82h (for 32-bit addressing)
BH = Major Revision Number of ECU
BL = Minor Revision Number of ECU

Output: AH = 00h
= 84h Error while clearing CMOS RAM
= 86h Invalid BIOS call
= 88h ECU is not supported
AL = Major Revision Number of ECU supported by BIOS (if AH = 88h).
CF = 0 Error
= 1 No error

Description:

Function D8h Subfunction 02h clears EISA Extended CMOS RAM. This routine does not clear the ISA CMOS RAM, which contains the date, time, hard disk drive type, and basic system configuration.

Function D8h Subfunction 03h (83h) Write to EISA CMOS RAM

Input: AH = D8h
AL = 03h (if CS specifies 16-bit addressing)
= 83h (if CS specifies 32-bit addressing)
CX = Length of table (if 0, then slot is empty)
DS:SI = Address of data buffer (16-bit addressing)
DS:ESI = Address of data buffer (32-bit addressing)

Output: AH = 00h No error
= 84h Error writing to EISA Extended CMOS RAM
= 85h CMOS RAM is full
= 86h Invalid BIOS call
= 87h EISA configuration is locked
CF = 0 No error
= 1 Error

Description:

Function D8h Subfunction 03h writes the configuration data specified in the data buffer pointed to by DS:SI to EISA Extended CMOS RAM. This function does not write to ISA CMOS RAM, which contains the basic system parameters. The data to be written to EISA Extended CMOS RAM should begin at address DS:SI (DS:ESI if using 32-bit addressing) for the length specified in CX. The last two bytes in the table are reserved for the checksum of the CFG file to be used.

EISA Configuration Data Table

The format for the EISA configuration data at DS:SI (DS:ESI) is:

Offset	Description
00h	First Byte of Compressed ID Bit 7 Reserved, should be zero. Bits 6–2 First character of the manufacturer code. Bits 1–0 First two bits of second character of the manufacturer code.
01h	Second Byte of Compressed ID Bits 7–5 Remaining three bits of second character of the manufacturer code. Bits 4–0 Third character of the manufacturer code.

Offset	Description
02h	Third Byte of Compressed ID Adapter card: Bits 7–4 First hex digit of the manufacturer's product number. Bits 3–0 Second hex digit of the manufacturer's product number. Motherboard: Bits 7–0 Reserved for manufacturer's use.
03h	Fourth Byte of Compressed ID Adapter card: Bits 7–4 Third hex digit of the manufacturer's product number. Bits 3–0 Product revision number Motherboard: Bits 7–3 Reserved for manufacturer's use. Bits 2–0 EISA bus version number (001 is initial version).
04h – 05h	ID and Slot Information Byte 0 Bit 7 0 No duplicate ID is present. 1 Duplicate ID found. Bit 6 0 ID is readable. 1 ID is unreadable. Bits 5–4 Device Type 00b Expansion device 01b Embedded device 10b Virtual device Bits 3–0 Number of Duplicate CFG filenames 0000b No duplicate CFG 0001b First duplicate CFG ... ... 1110b Fourteenth duplicate CFG 1111b Fifteenth duplicate CFG Byte 1 Bit 7 0 Configuration is successful. 1 Configuration is unsuccessful. Bits 6–2 Reserved, should be zeros. Bit 1 0 EISA IOCHKERR not supported. 1 EISA IOCHKERR supported. Bit 0 0 EISA ENABLE not supported (adapter card cannot be enabled or disabled). 1 EISA ENABLE supported (adapter card can be enabled or disabled). The EISA specification allows EISA adapter cards to be enabled or disabled via software. If bit 0 of byte 1 above is set, external software can disable the adapter card. Similarly, the availability of IOCHKERR allows external software to check expansion slots for pending errors.

Offset	Description
06h – 07h	Revision levels of the CFG overlay files used for a specified slot. Both bytes are 0 if no overlay file exists. Byte 0 Minor revision level of the CFG overlay file. Byte 1 Major revision level of the CFG overlay file.
The rest of this table is repeated once for every EISA function in the system. There can be *1 through n* EISA functions. Most EISA Adapter Cards have more than one function. The last function is empty and has a length of 0. All functions must fit in 340 bytes.	
2 bytes, but they do not count as part of the function length.	Function Length. The length does not include these two bytes or the checksum at the end of EISA CMOS RAM. The last function must be set to length 0. Byte 0 LSB of the length of the following function entry. Byte 1 MSB of the length of the following function entry.
2 to 27 bytes for each function.	Selections made by the system ECU. The possible choices for the specified slot function are counted here. The actual names of the choices are specified in the CFG file. Byte 0 Selection 1 Byte 1 Selection 2 ... ... Byte 24 Selection 25 Byte 25 Selection 26
1 byte for each function.	Slot function information Bit 7 0 Slot function is enabled. 1 Slot function is disabled. Bit 6 CFG is using free form data if set. Bit 5 Port initialization entry(s) follows if set. Bit 4 Port range entry(s) follows if set. If not set, the port range section is length 0. Bit 3 DMA entry(s) follows if set. If not set, the DMA entry section is length 0. Bit 2 IRQ entry(s) follows if set. If not set, the IRQ entry section is length 0. Bit 1 Memory entry(s) follows if set. If not set, the Memory section is length 0. Bit 0 Type and Subtype string follows if set.

Offset	Description
2 – 81 bytes for each function.	Byte 0 Length of the following field Bytes 1-80 A 1 - 80-character ASCII string describing the slot device. The string has types and subtypes. For example, TYPE=COM, AMI; COM1 would be: 0ChCOM,AMI;COM1 The manufacturer determines the type and subtype format, but the conventions are: **Type String** COM Communications device KEY Keyboard MEM Memory card MFC Multifunction card MSD Mass storage device NET Network card NPX Math coprocessor OSE Operating system or environment OTH Other PAR Parallel port PTR Pointing device SYS Motherboard VID Video adapter card , Delimiter for Type string fragments ; End of Type string and beginning of Subtype string 0 End of Subtype strings The unused part of the 80-character string should be zero (not including the Subtype delimiter).

Offset	Description
7 to 63 bytes for each function.	Memory Configuration Section. 0 to Nine seven-byte entries: Byte 0 Memory Configuration Byte Bit 7 0 Last entry 1 More entries follow Bit 6 Reserved, should be zero. Bit 5 0 Memory is not shared 1 Memory is shared Bits 4–3 00b SYS (base/extended memory) 01b EXP (expanded memory) 10b VIR (virtual memory) 11b OTH (other memory) Bit 1 0 Memory is not cached 1 Memory is cached Bit 0 0 Memory is ROM (read only) 1 Memory is RAM (read and write) Byte 1 Memory Data Size Bits 7-4 Reserved, should be zeros. Bits 3–2 Decode Size 00b 20 address lines 01b 24 address lines 10b 32 address lines Bits 1–0 Data Access Size 00b Byte 01b Word (16 bits) 10b Doubleword (32 bits) Bytes 2–4 Starting Memory Address divided by 100h Bytes 5–6 Memory Size divided by 400. If 0000h, memory size is 64 MB. Size is specified in 1024 byte increments.
2 - 14 bytes for each function.	IRQ Configuration Section. 1 to 7 two-byte entries. Byte 0 Bit 7 0 Last entry 1 More entries follow Bit 6 0 Interrupt is not shared 1 Interrupt is shared Bit 5 0 Interrupt is edge-triggered 1 Interrupt is level-triggered Bit 4 Reserved (should be 0) Bits 3–0 Interrupt number 0000b IRQ0 0001b IRQ1 ... ... 1110b IRQ14 1111b IRQ15 Byte 1 Reserved, should be zero.

Offset	Description
0 - 4 entries for each function. 2 - 8 bytes for each entry.	DMA Channel Description Section. 0 – 4 two-byte entries. Byte 0 Bit 7 0 Last entry 1 More entries follow Bit 6 0 DMA channel is not shared 1 DMA channel is shared Bits 5–3 Reserved, should be zeros. Bits 2–0 DMA Channel Number 000b Channel 0 001b Channel 1 ... ... 110b Channel 6 111b Channel 7 Byte 1 Bits 7–6 Reserved, should be zeros. Bits 5–4 DMA Timing 00b ISA-compatible timing 01b Type A timing 10b Type B timing 11b Type C (Burst) timing Bits 3–2 DMA Transfer Size 00b Byte transfers 01b Word transfers (16 bits) 10b Doubleword transfers (32 bits) Bits 1–0 Reserved, should be zeros.
1 to 20 entries for each function. 3 to 60 bytes for each entry.	I/O Port Information consists of 0 to 20 three-byte entries: Byte 0 Bit 7 0 Last entry 1 More entries follow Bit 6 0 Port is not shared 1 Port is shared Bit 5 Reserved, should be zero. Bits 4–0 Number of Ports (starting at 0) 00000b One port 00001b Two sequential ports 00010b Three sequential ports ... ... 11110b Thirty-one sequential ports 11111b Thirty-two sequential ports Byte 1 LSB of I/O Port Address Byte 2 MSB of I/O Port Address

Offset	Description
0 - 60 bytes for each function. 0 - 20 entries for each function.	I/O Port Initialization Data Section. Entries vary in length. Byte 0 Initialization Type Bit 7 0 Last entry 1 More entries follow Bits 6–3 Reserved, should be zeros. Bit 2 0 Write value to port 1 Use both mask and value Bits 1–0 Data Access Size *If Byte 0, bit 2 is 0, the following format is used:* 00b Byte 3 is the initialization value. 01b Byte 3 is the LSB of the initialization value. Byte 4 is the MSB of the initialization value. 10b Byte 3 is the LSB of the initialization value. Byte 4 is the second byte of the initialization value. Byte 5 is the third byte of the initialization value. Byte 6 is the MSB of the initialization value. *If Byte 0, bit 2 is 1, the following format is used:* 00b Byte 3 is the initialization value. Byte 4 is mask value. 01b Byte 3 is the LSB of the initialization value. Byte 4 is the MSB of the initialization value. Byte 5 is the LSB of the mask value. Byte 6 is the MSB of the mask value. 10b Byte 3 is the LSB of the initialization value. Byte 4 is the second byte of the initialization value. Byte 5 is the third byte of the initialization value. Byte 6 is the MSB of the initialization value. Byte 7 is the LSB of the mask value. Byte 8 is the second byte of the mask value. Byte 9 is the third byte of the mask value. Byte 10 is the MSB of the mask value. Byte 1 LSB of Port Address Byte 2 MSB of Port Address
The following field is not included in the entries for each function. It only occurs once at the very end of this table.	
2 bytes	Checksum of the CFG file that configured this table Byte 0 LSB of the EISA configuration file checksum. Byte 1 MSB of the EISA configuration file checksum.

Note: If bit 6 of the Function Information Section (22h) is set, the table is not in the table format described above, but uses free-form data. Entries through the Type and Subtype field are the same, but starting with the Memory Configuration field, the motherboard manufacturer's proprietary format is used.

cont'd

Function D8h Subfunction 04h (84h) Read Slot Device Compressed ID

Input: AH = D8h
AL = 04h (for 16-bit addressing)
= 84h (for 32-bit addressing)
CL = Slot Number (virtual and embedded devices included)
00h Motherboard
01h Slot 1
02h Slot 2
... ...
... ...
0Fh Slot 15

Output: AH = 00h No error
= 80h Invalid slot number
= 83h Slot is empty
= 86h Invalid BIOS call
= 87h Invalid system configuration
CF = 0 No error
= 1 Error
DI (LSB) = Byte 0 of Compressed ID
DI (MSB) = Byte 1 of Compressed ID
SI (LSB) = Byte 2 of Compressed ID
SI (MSB) = Byte 3 of Compressed ID

Description:

Function D8h Subfunction 04h (84h) returns the compressed ID from the device installed in the specified slot. The slot can be the motherboard, an adapter card, an embedded device, or a virtual device.

DI and SI contain a four-byte compressed ID number of the device installed in the specified slot. The compressed ID format is described on page 224.

INT 16h Keyboard Service

INT 16h controls the system keyboard. Functions 00h through 02h are used with XT-compatible keyboards (83 and 84-key) only. Functions 10h through 12h are used with AT enhanced keyboards (101 and 102-key) only. Functions 03h and 05h can be used with either type of keyboard.

INT 16h Functions

Function	Description
00h	Read Character
01h	Return Keyboard Status
02h	Return Keyboard Flags
03h	Set Keyboard Typematic Rate Parameters
05h	Push Character and Scan Code to Buffer
10h	Enhanced Keyboard Read Character
11h	Enhanced Keyboard Write Character
12h	Enhanced Keyboard Return Keyboard Flags
F0h	Set CPU Speed
F1h	Read CPU Speed
F4h	Cache Controller AL = 00h Read Cache Controller Status AL = 01h Enable Cache Controller AL = 02h Disable Cache Controller

Function 00h Read Character

Input: AH = 00h

Output: AH = Scan code or character ID if special character.
AL = ASCII code

Description:

INT 16h Function 00h reads a character from the keyboard and returns the scan and ASCII codes for that character.

Function 01h Return Keyboard Status

Input: AH = 01h

Output: AH = Scan code of character ID if special character (only if ZF is 0).
AL = ASCII code or character translation
ZF = 0 Character waiting
= 1 No character waiting

Description:

INT 16h Function 01h determines if a character is waiting for input. If so, it returns the character and its scan code. Function 01h does not remove the character from the keyboard buffer. The character must be read using Function 00h to be removed from the buffer.

Function 02h Return Keyboard Flags

Input: AH = 02h

Output: AL = Keyboard Flags
Bit 7 Insert mode on
Bit 6 Caps Lock key on
Bit 5 Num Lock key on
Bit 4 Scroll Lock key on
Bit 3 Alt key pressed
Bit 2 Ctrl key pressed
Bit 1 Left Shift key pressed
Bit 0 Right Shift key pressed

Description:

INT 16h Function 02h returns the Keyboard Flags Byte (40:17h in the BIOS Data Area). The Keyboard Flags Byte describes the state of certain keys.

Function 03h Set Typematic Rate Parameters

Input: AH = 03h
AL = 05h
BH = Typematic delay
00h 250 ms
01h 500 ms
02h 750 ms
03h 1000 ms
BL = Typematic rate

BL	Rate	BL	Rate
00h	30.0 characters per second	10h	7.5 characters per second
01h	26.7 characters per second	11h	6.7 characters per second
02h	24.0 characters per second	12h	6.0 characters per second
03h	21.8 characters per second	13h	5.5 characters per second
04h	20.0 characters per second	14h	5.0 characters per second
05h	18.5 characters per second	15h	4.6 characters per second
06h	17.1 characters per second	16h	4.3 characters per second
07h	16.0 characters per second	17h	4.0 characters per second
08h	15.0 characters per second	18h	3.7 characters per second
09h	13.3 characters per second	19h	3.3 characters per second
0Ah	12.0 characters per second	1Ah	3.0 characters per second
0Bh	10.9 characters per second	1Bh	2.7 characters per second
0Ch	10.0 characters per second	1Ch	2.5 characters per second
0Dh	9.2 characters per second	1Dh	2.3 characters per second
0Eh	8.6 characters per second	1Eh	2.1 characters per second
0Fh	8.0 characters per second	1Fh	2.0 characters per second

Output: None

Description:

Function 03h sets the keyboard typematic rate parameters. The typematic rate delay is the length of the delay between the first key character printed on the screen and first repeated character. The typematic rate is the number of characters to be repeated per second.

Function 05h Push Character and Scan Code to Buffer

Input: AH = 05h
CH = Scan Code to be pushed
CL = Character to be pushed

Output: AL = 00h No error
= 01h Keyboard buffer full
CF = 0 No error
= 1 Keyboard buffer is full

Description:

INT 16h Function 05h places the specified character and scan code in the keyboard buffer.

Function 10h Enhanced Keyboard Read Character

Input: AH = 10h

Output: AH = 00h Scan code or character ID if special character.
AL = ASCII code

Description:

Function 10h reads a character from the keyboard buffer and returns its ASCII code and scan code.

Function 10h should be used with enhanced keyboards only.

Function 11h Enhanced Keyboard Return Status

Input: AH = 11h

Output: AH = Scan Code or character ID if special character.
AL = ASCII code of character
ZF = 0 Character waiting
= 1 No character waiting

Description:

Function 11h determines if a character is waiting for input. If so, it returns the character and its scan code. Function 11h does not remove the character from the keyboard buffer. The character must be read via Function 10h to be removed from the buffer.

Function 11h should be used only with enhanced keyboards.

Function 12h Return Enhanced Keyboard Flags

Input: AH = 12h

Output: AX = Keyboard Flags

00h	Right Shift key pressed
01h	Left Shift key pressed
02h	Ctrl key pressed
03h	Alt key pressed
04h	Scroll Lock is on
05h	Num Lock is on
06h	Caps Lock is on
07h	Insert mode is on
08h	Left Ctrl key is pressed
09h	Left Alt key is pressed
0Ah	Right Ctrl key is pressed
0Bh	Right Alt key is pressed
0Ch	Scroll Lock key is pressed
0Dh	Num Lock key is pressed
0Eh	Caps Lock key is pressed
0Fh	SysReq key is pressed

Description:

INT 16h Function 12h returns the Keyboard Flags at 40:17h and 40:18h and the Extended Keyboard Flags Byte (40:97h). These flags describe the state of various keys on the keyboard.

Function 12h should be used only with enhanced keyboards.

Function F0h Set CPU Speed

Input: AH = F0h
AL = 00h or 01h Low Speed
02h High Speed

Output: None

Description:

Function F0h sets the CPU speed to Low or High. This function returns no values and does not destroy the contents of any registers. This function is only available if the BIOS date is 6/6/92 or later.

Function F1h Read CPU Speed

Input: AH = F1h

Output: AL = 00h or 01h Low Speed
= 02h High speed

Description:

Function F1h reads the current CPU speed. This function destroys the contents of AL, but no other registers. This function is only available if the BIOS date is 6/6/92 or later.

Function F4h Subfunction 00h Read Cache Controller Status

Input: AH = F4h
AL = 00h

Output: AH = None if cache controller cannot be enabled.
= E2h Successful
AL = Cache Controller Status
00h Cache controller not present
= 01h Cache memory enabled
= 02h Cache memory disabled
CX = Cache Memory Size
Bit 15 0 Cache size information is valid
1 Cache size information is invalid
Bits 14–0 Cache memory size in KB
DH = Cache Write Technology
Bit 7 0 Cache write information is valid
1 Cache write information is not valid
Bits 6–1 Reserved (Set to 0)
Bit 0 0 Write-through caching algorithm used
1 Write-back caching algorithm
DL = Cache Type
Bit 7 0 Cache type information is valid
1 Cache type information is not valid
Bits 6–1 Reserved (Set to 0)
Bit 0 0 Cache type is direct-mapped
1 Cache type is two-way set-associative

Description: Function F4h Subfunction AL = 00h returns cache controller status information. If unsuccessful, no register values are changed. The values in AX, CX, and DX are destroyed if successful. This function is only available if the BIOS date is 6/6/92 or later.

Function F4h Subfunction 01h Enable Cache Controller

Input: AH = F0h
AL = 01h

Output: AH = None if cache controller cannot be enabled.
= E2h if cache controller can be enabled.

Description:

Function F4h Subfunction AL = 01h enables the cache controller. The register content is not changed if the cache controller cannot be enabled. AH is destroyed if successful. This function is only available if the BIOS date is 6/6/92 or later.

Function F4h Subfunction 02h Disable Cache Controller

Input: AH = F1h
AL = 02h

Output: AL = None if cache controller cannot be disabled
= E2h If successful

Description:

Function F4h Subfunction AL = 02h disables the cache controller. The register content is not changed if the cache controller cannot be enabled. AH is destroyed if successful. This function is only available if the BIOS date is 6/6/92 or later.

INT 17h Parallel Port Service

INT 17h controls the parallel ports. The AMIBIOS uses three parallel ports, initialized to the following beginning I/O port addresses: 03BCh, 0378h, and 0278h, if present. The default values for the parallel ports in the Hi-Flex AMIBIOS can be modified via AMIBCP.

INT 17h Parallel Printer Functions

The INT 17h parallel printer functions are:

Function	Description
00h	Write Character
01h	Initialize Parallel Port
02h	Return Parallel Port Status

Function 00h Write Character

Input: AH = 00h
AL = Character
DX = Parallel Port Number. Index to parallel port lead address table at 40:08h.
00h LPT 1 01h LPT 2
02h LPT 3

Output: AH = Port Status
Bit 7 Printer not busy if set to 1.
Bit 6 Printer acknowledge if set to 1.
Bit 5 Out of paper if set to 1.
Bit 4 Printer selected if set to 1.
Bit 3 I/O error if set to 1.
Bits 2–1 Reserved
Bit 0 Printer timed-out is et to 1.

Description: Function 00h writes a character to the specified parallel port. The status is returned in AH.

Function 01h Initialize Parallel Port

Input: AH = 01h
DX = Parallel Port Number. Index to parallel port lead address table at 40:08h.
00h LPT 1 01h LPT 2 02h LPT 3

Output: AH = Parallel Port Status
Bit 7 Printer not busy if set to 1.
Bit 6 Printer acknowledge if set to 1.
Bit 5 Out of paper if set to 1.
Bit 4 Printer selected if set to 1.
Bit 3 I/O error if set to 1.
Bits 2–1 Reserved
Bit 0 Printer timed-out if set to 1.

Description: INT 17h Function 01h initializes the specified parallel port. The Parallel Port Status is returned in AH.

Function 02h Read Parallel Port Status

Input: AH = 02h
DX = Parallel Port Number. Index to parallel port lead address table at 40:08h.
00h LPT 1 01h LPT 2 02h LPT 3

Output: AH = Port Status
Bit 7 Printer not busy if set to 1.
Bit 6 Printer acknowledge if set to 1.
Bit 5 Out of paper if set to 1.
Bit 4 Printer selected if set to 1.
Bit 3 I/O error if set to 1.
Bits 2–1 Reserved
Bit 0 Printer timed-out if set to 1.

Description: INT 17h Function 02h returns the specified parallel port status in AH.

INT 18h ROM BASIC

Input: None

Output: None

Description:

On the original IBM PC, INT 18h transferred control to ROM BASIC. ROM BASIC is not supported by IBM anymore. If INT 18h is invoked, the BIOS halts the system and displays:

```
NO BOOT DEVICE AVAILABLE
```

The only way to regain control of the system is to reboot.

Other Uses of INT 18h

Some network cards contain boot ROMs so that a system attached to a network can boot without using a hard disk or floppy disk. These ROMs trap INT 18h to gain access to the system. For this reason, INT 18h has been included in AMIBIOS.

INT 19h System Boot Control

Input: None

Output: None

Description:

INT 19h transfers control to the operating system.

The BIOS reads the boot sector (sector 1, track 0) from the primary boot device (floppy drive A: or hard disk C:) and writes that data to 0000:7C00h. The BIOS gives control to the data at that address, which in turn loads (or boots) the operating system.

If the BIOS does not find a boot sector on the primary boot device, it looks for a boot sector on the secondary boot device. The primary and secondary boot devices are floppy drive A:, then hard disk drive C:.

If no boot sector is found on either drive A: or C:, INT 18h is invoked. See the INT 18h description on page 251.

System Boot Up Sequence Option

The *System Boot Up Sequence* option in the AMIBIOS Advanced CMOS Setup permits you to set the boot sequence to either *C:, then A:* or *A:, then C:*.

An option to only boot from the C: drive has been added to newer AMIBIOSes, to prevent the inadvertent entry of viruses to the system via the A: drive.

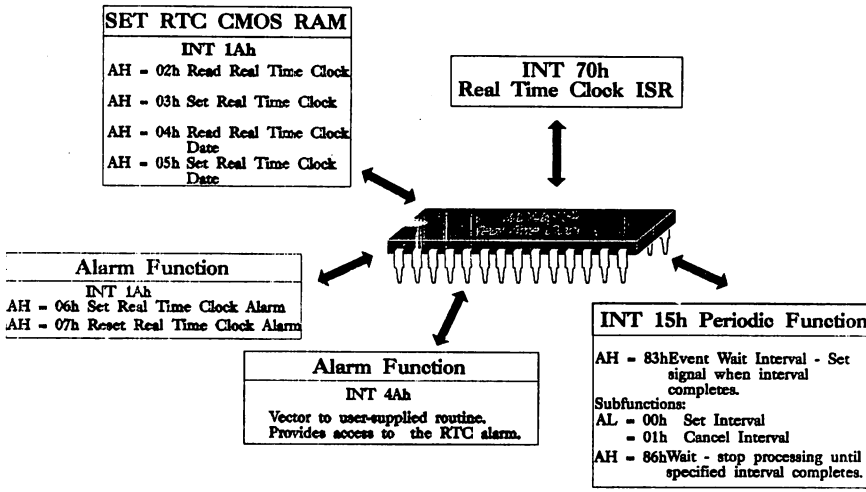
Using AMIBCP to Change Boot Sequence

The OEM can also use AMIBCP to change the system boot sequence, setting drive C: as the primary boot device, and drive A: as the secondary boot device.

If...	and...	then...
The Advanced CMOS Setup option *System Boot Up Sequence* is set to A:, C:,	a bootable floppy disk is in drive A:,	INT 19h reads the boot sector on the floppy disk and places its contents at 7C00h.
The Advanced CMOS Setup option *System Boot Up Sequence* is set to A:, C:,	Drive A: has no bootable disk: or the floppy disk in drive A: is not bootable,	INT 19h invokes INT 18h. INT 18h displays: NO BOOT DEVICE AVAILABLE
The Advanced CMOS Setup option *System Boot Up Sequence* is set to C:, A:,	the boot sector is found on drive C:	INT 19h reads the boot sector on the floppy disk and places its contents at 7C00h.
The Advanced CMOS Setup option *System Boot Up Sequence* is set to C:, A:,	Hard Disk Drive C: has no boot sector (most likely the drive type is not properly configured)	INT 19h invokes INT 18h. INT 18h displays: NO BOOT DEVICE AVAILABLE

INT 1Ah Real Time Clock Service

INT 1Ah functions set or read the system Real Time Clock and perform PCMCIA Socket Service functions. The Real Time Clock ISR is INT 70h. See the INT 08h discussion on page 124 for a discussion of timers. The following graphic illustrates how the real time clock is used with the BIOS.



INT 1Ah Socket Services

Socket Services is an extension to system BIOS software interrupt 1Ah Real Time Clock Service. All Socket Services are function calls to INT 1Ah.

Socket Services provides the software interface to the hardware controlling PCMCIA-compatible cards (memory and I/O) in sockets. Socket Services provides the lowest level access to PCMCIA cards but does not interpret the content of the cards.

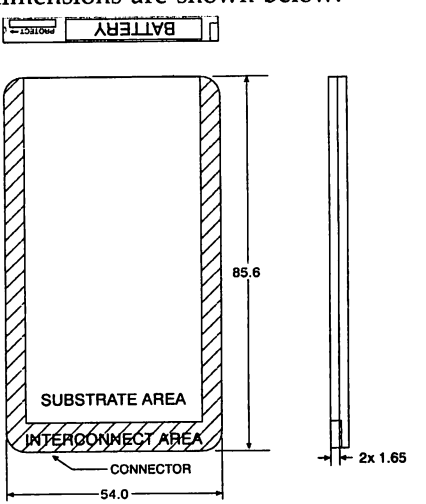
PCMCIA

The PC Card specification has been defined by PCMCIA. The Personal Computer Memory Card International Association (PCMCIA) consists of over 300 international manufacturers of computer hardware, software, semiconductors, connectors, peripherals and system BIOS manufacturers (including American Megatrends, Inc.).

PCMCIA develops methods or systems of data interchange suitable for the needs of portable computing. PCMCIA has developed a standard for PC Cards.

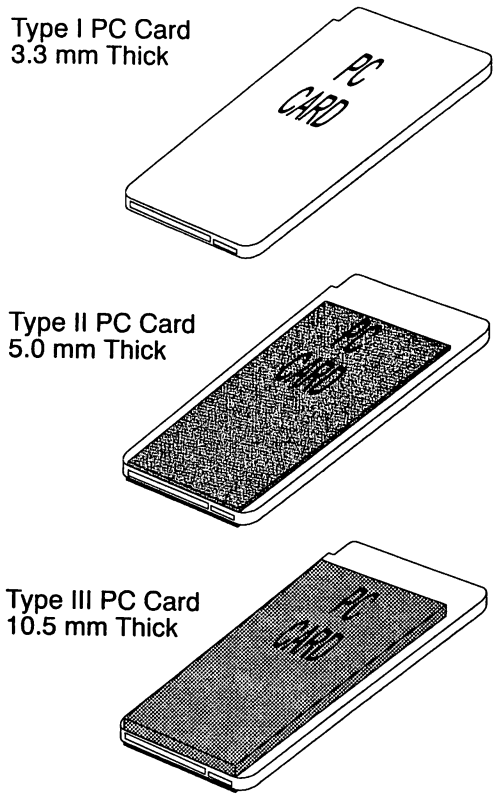
PC Card Size

A PC Card is a small form factor electronic device a little thicker than a credit card. PC Cards provide functions such as added memory for data interchange between computers. Additionally, these cards are used to expand the I/O capabilities of a computer by adding such functions as serial or parallel ports, SCSI ports, Network Ports, and Fax/Modems. The PC Card dimensions are shown below.



PCMCIA Card Types

The PCMCIA specifications describe the physical, electrical, and software requirements for three card types: Types I, II, and III. All types use the same 68-pin edge connector to connect to the computer, but differ in width. The differences in the PC Card types is shown below.



PCMCIA Card Types, cont'd

Type I Cards

Type I Cards are used primarily for various types of memory upgrades such as RAM, FLASH, One Time Programmable (OTP), or electrically erasable/programmable read only memory (EEPROM).

Type II Cards

Type II cards can be used for memory enhancements as described in Type I above or for I/O functions such as FAX/Modems, LAN connections, or other host communications.

Type III Cards

Type III PC Cards are twice the thickness of Type II cards and can be used for memory enhancements and/or I/O functions requiring additional head room on the card such as rotating media devices and radio communication devices.

Form Follows Function

Since all three cards adhere to the same electrical interface, the type of card chosen by the card designer depends totally on the function being implemented. The functionality of the card depends on the components located inside the card and the software residing inside the computer.

Where Can PC Cards be Used?

PC Cards can be used in Laptop Computers, Palmtop Computers, Pen Computers, Desktop Computers, or any other type of computing device that adheres to the specifications. PC Cards make communication between portable computers and desktop computers or peripherals easy and affordable.

PCMCIA Advantages

Unlike ISA adapter cards, which require actual physical configuration jumpers and switches, PC Cards are configured through software. PCMCIA hardware and software provide an easy-to-install, self-configuring Plug & Play solution for portable and desktop computers. Once PCMCIA software has been installed and initialized, PC Cards can be inserted and removed with no concern for system performance.

PCMCIA Software

The primary architectural software building blocks required by PCMCIA PC Cards are system Socket Services and Card Services.

Socket Services Function

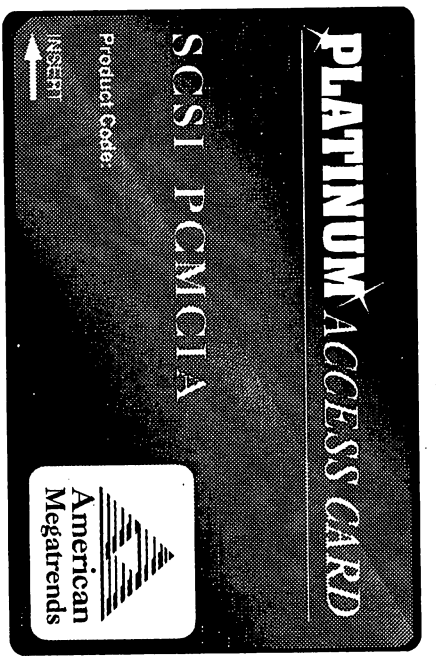
Socket Services are BIOS-resident and provide the system a means of access to the functions of the sockets themselves. Socket Services are used by the computer system to identify how many sockets are in the computer system and whether a card has been removed (hot extraction) or inserted into (hot insertion) the system while the system is powered on. Socket Services are functions called via the system BIOS INT 1Ah Real Time Clock Service.

Card Services

Card Services is a software/system management layer that allocates and manages the resources of the system to the cards such as interrupt levels, DMA channels, and memory windows. These activities can occur only after Socket Services has determined that there is a PC Card in one of the system sockets. Card Services also releases system resources for use by other system software if Socket Services determines that a particular PC Card has been removed from one of the system sockets. Card Services is the system software level interface for the operating systems for PC Card and Socket Services.

PCMCIA Hardware Standards

The following illustration is approximately the actual size of a PCMCIA PC Card.



Summary of INT 1Ah Real Time Clock Functions

INT 1Ah also provides functions for Socket Services, described on pages 263 through 317.

Function	Description
00h	Return Clock Tick Count
01h	Set Clock Tick Count
02h	Return Current Time
03h	Set Current Time
04h	Return Current Date
05h	Return Current Date
06h	Set Alarm
07h	Reset Alarm
8Ah – 9Ch	Socket Services Functions (see page 263)
9Dh	Card Services Functions
B1h	PCI BIOS Functions

Function 00h Return Clock Tick Count

Input: AH = 00h

Output: AL = 00h Midnight has not passed since last call.
CX:DX = Clock Tick Count (CX is the MSB)

Description:

Function 00h returns the value of the timer tick counter from 40:6Ch through 40:6Fh. The value is the number of ticks counted since midnight. Approximately 18.2 timer ticks occur every second.

The contents of 40:70h Timer Overflow are returned in AL. This value is zero if the timer has not overflowed past 24 hours since the last call.

Function 01h Set Clock Tick Count

Input: AH = 01h
CX:DX = Clock Tick Count (CX is MSB)

Output: None

Description:

Function 01h sets the clock tick counter in 40:6Ch – 6Fh to the value specified in CX and DX. Approximately 18.2 ticks occur a second. The Timer Overflow flag at 40:70h is reset to 0 by this function.

Function 02h Return Current Time

Input: AH = 02h

Output: CF = 0 Successful
= 1 Clock has stopped running.
CH = Number of Hours in Binary Coded Decimal (BCD)
CL = Number of Minutes (in BCD)
DH = Number of Seconds (in BCD)
DL = 00h Standard time
= 01h Daylight savings time

Description:

Function 02h reads the current time from Real Time Clock CMOS RAM.

Function 03h Set Current Time

Input: AH = 03h
CH = Number of Hours (in BCD)
CL = Number of Minutes (in BCD)
DH = Number of Seconds (in BCD)
DL = 00h Standard time
= 01h Daylight savings time

Output: AL = Value written to CMOS RAM Register B

Description:

Function 03h writes a specified time to Real Time Clock CMOS RAM.

Function 04h Return Current Date

Input: AH = 04h

Output: CF = 0 Successful
= 1 Clock has stopped running.
CH = Century (in BCD)
CL = Year (in BCD)
DH = Month (in BCD)
DL = Day (in BCD)

Description:

Function 04h reads the current date from Real Time Clock CMOS RAM.

Function 05h Set Current Date

Input: AH = 05h
CH = Century (in BCD)
CL = Year (in BCD)
DH = Month (in BCD)
DL = Day (in BCD)

Output: AL = Value written to Register B of RTC CMOS RAM

Description: Writes the specified date to RTC CMOS RAM.

Function 06h Set Alarm

Input: AH = 06h
CH = Hours (in BCD)
CL = Minutes (in BCD)
DH = Seconds (in BCD)

Output: CF = 0 No error
= 1 The alarm is already set.

Description: Function 06h sets an alarm for the specified time in RTC CMOS RAM and enables the clock interrupt request line (IRQ8). Trap the INT 4Ah vector (0:128h) and replace it with the address of your own alarm service routine.

Function 07h Reset Alarm

Input: AH = 07h

Output: AL = Value written to Register B in RTC CMOS RAM

Description: Resets all alarms in Real Time Clock CMOS RAM. It does not disable the clock interrupt request line (IRQ8).

Socket Services Function Summary

Function	Name	Turn to
80h	Get Adapter Count	Page 265
83h	Get Socket Services Version Number	Page 266
84h	Inquire Adapter	Page 267
85h	Get Adapter	Page 270
86h	Set Adapter	Page 271
87h	Inquire Window	Page 273
88h	Get Window	Page 280
89h	Set Window	Page 282
8Ah	Get Page	Page 284
8Bh	Set Page	Page 286
8Ch	Inquire Socket	Page 288
8Dh	Get Socket	Page 291
8Eh	Set Socket	Page 294
8Fh	Get Status	Page 297
90h	Reset Card	Page 300
95h	Inquire EDC (Error Detection Code)	Page 301
96h	Get EDC	Page 303
97h	Set EDC	Page 304
98h	Start EDC	Page 305
99h	Pause EDC	Page 306
9Ah	Resume EDC	Page 306
9Bh	Stop EDC	Page 307
9Ch	Read EDC	Page 307
9Dh	Get Vendor Info	Page 308
9Eh	Acknowledge Interrupt	Page 309
9Fh	Prior Handler	Page 310
A0h	Get SS Addr	Page 312
A1h	Access Offsets	Page 315
AEh	Vendor-Specific Information	Page 317

Socket Services Calling Conventions

Socket Services functions are invoked through software interrupt 1Ah. The general convention for invoking the socket services functions is:

Input: AH = Function number
AL = Adapter number
BH = Window number
BL = Socket number or Page number

Other input parameters may be added, depending on the specific function.

Output: CF = 0 Successful
= 1 Error
AH = Error code

Function 80h Get Adapter Count

This function returns the number of adapters supported by Socket Services and can be used to determine the presence of the Socket Services handler.

Input: AH = 80h

Output: AL = Number of adapters (one-based)
CF = 0 Successful. Socket Services handler present.
= 1 Error. Socket Services handler not present.
CX = the string *SS*

Even if the Socket Services handler is present, there may not be any adapter installed. In this case, this function should return with CF set, *SS* in CX, and 00h in AL. The caller of this function must handle this situation properly.

Function 83h Get SS Info

This function returns the version of both Implementor and PCMCIA Socket Services compliance levels. Version numbers are returned as binary coded decimals (BCD) values.

Input: AH = 83h
AL = Adapter number (zero-based)

Output: AH = Error code
= 00h Successful
= 01h Bad adapter
AL = PCMCIA Socket Services Version Number
00h Insures compatibility with Release 1.01.
BX = Socket Services Interface Specification Compliance Level (0200h for PCMCIA V2.00)
CF = 0 Successful
= 1 Error
CH = Number of adapters supported by this handler.
CL = First adapter supported by this handler.

If more than one type of adapter is present in the system, there may be more than one Socket Services handlers present. This function determines the support level of Socket Services for the specified adapter.

Function 84h Inquire Adapter

This function returns information about the specified adapter.

Input:

AH	=	84h
AL	=	Adapter number (zero-based)
ES:EDI	=	Pointer to a buffer supplied by the calling program that will be filled with information about the adapter by Socket Services.

Output:

AH	=	Error code
	=	00h Successful
	=	01h Bad adapter
BH	=	Number of windows (one-based)
BL	=	Number of sockets (one-based)
CF	=	0 Successful
	=	1 Error
CX	=	Number of EDCs (Error Detection Code) (can be 0 – the total number of sockets)
ES:EDI	=	Pointer to buffer containing adapter characteristics and power management tables.

The buffer pointed to by the contents of ES:EDI is supplied by the calling program and must have the following format:

```
typedef struct tagAISTRUCT {
    WORD wBufferLength;
    WORD wDataLength;
    ACHARTBL CharTable
    WORD wNumPwrEntries = NUM_ENTRIES;
    PWRENTRY PwrEntry[NUM_ENTRIES];
} AISTRUCT;
```

The CharTbl structure is defined below. wBufferLength must be set by the calling program to the size of AISTRUCT minus four bytes. wDataLength is set by Socket Services to the size of the information block returned. If the wDataLength value is greater than the wBufferLength value, the information is truncated.

PWRENTRY

PWRENTRY is a two-member structure. The first member is a binary value representing a DC voltage level in tenths of a volt with a maximum of 25.5 VDC. The second member specifies the power signals that may be set to the specified voltage level (either Vcc, Vpp1, or Vpp2). All sockets on an adapter should use the same power levels. Make one PWRENTRY for each supported voltage. PWRENTRY only indicates that it is possible to set power pins to a certain power level. It is up to the calling program to determine if the specified combination of power levels is valid for the PC Card in the socket. The PWRENTRY structure is shown below:

```
typedef struct tagPWRENTRY {
   BYTE PowerLevel;
   BYTE ValidSignals;
} PWRENTRY
```

where:

PowerLevel — the DC voltage level in tenths of a volt. Power levels from 0 (N/C) through 25.5 VDC are valid.

ValidSignals — flags that indicate if voltage is valid for specific signals. A combination of the following can be used:

Vcc	Voltage level valid for the Vcc signal
Vpp1	Voltage level valid for the Vpp1 signal
Vpp2	Voltage level valid for the Vpp2 signal

Sample AISTRUCT

```
AISTRUCT AdapterInfo = {
    24,              //Size of calling program-supplied buffer is 24 //bytes
    24,              //Size of data returned is 24 bytes
    {0,              //Indicators, power, and data bus width are controlled
                     //at the socket
    0xDEB8           //Status changes may be routed to IRQ levels
                     //3, 4, 5, 7, 9, 10, 11, 12, 14, and 15
                     //as an active high signal
    0},              //Status changes are not available on
                     //any level as an active low signal
    3,               //Number of PWRENTRY elements
    ((VCC | VPP1 | VPP2) << 8) | 0
                     //Vcc, Vpp1, and Vpp2 - No Connect
    ((VCC | VPP1 | VPP2) << 8) | 50
                     //Vcc, Vpp1, and Vpp2 - 5.0 VDC
    ((VPP | VPP2 | << 8) | 120
                     //Vpp1 and Vpp2 - 12.0 VDC
```

ACHATbl Structure

```
typedef struct tagACHATBL {    //Same format as Socket
                               //characteristics except
   WORD AdpCaps;               //CHARTBL has different values
   DWORD ActiveHigh;
   DWORD ActiveLow;
} ACHATBL;
```

AdpCaps

AdpCaps (Adapter capabilities) is structured as follows:

Indicators

0 There are individual indicators for each socket.
1 Indicators for write protect, card lock, battery status, busy status, and XIP status are shared by all sockets on the adapter.

Data bus width

0 Data bus width set individually for each window.
1 All windows on the adapter must use the same data bus width.

Power Level

0 Power levels can be individually set for each socket.
1 The adapter requires all sockets to be set to the same power level controls.

ActiveHigh

A doubleword bitmap of the status change interrupt levels that can be routed active high.

ActiveLow

A doubleword bitmap of the status change interrupt levels that can be routed active low.

Function 85h Get Adapter

This function returns the current configuration of the specified adapter.

Input: AH = 85h

Output: AH = Error code
AL = Adapter number (zero-based).
CF = 0 Successful
= 1 Error
DH = Adapter attributes
Bit 1 Preserve state information in power-down
1 True
Bit 0 Reduce power consumption
1 True
DI = Status change interrupt routing
Bit 7 IRQ enabled
1 Status change is enabled.
Bit 6 IRQ high
1 Status change interrupt is active high.
Bits 4-0 IRQ level

Bit 0 of DH (Reduce power consumption) indicates if the adapter hardware is attempting to conserve power. Before using the adapter, full power must be restored via INT 1Ah AH = 86h Set Adapter.

If Bit 1 of DH (Preserve State Information) is set to 1, all adapter and socket status are retained in reduced-power mode. If this bit is set to 0, the software that placed the adapter in reduced-power mode must save all adapter and socket status.

The ability to reduce power consumption is not available in all adapters. Reduced power settings may not result in any power savings. The Inquire Adapter function (AH = 84h) indicates if it is possible to share the status change interrupt. This function returns the form of interrupt sharing (if any) currently being performed.

Function 86h Set Adapter

This function sets the configuration of the specified adapter. The card status change interrupt is enabled or disabled through this function.

Input: AH = 86h
AL = Adapter number (zero-based)
DH = Adapter attributes
Bit 1 State information in power down
1 Preserve status information
Bit 0 Power consumption
1 Reduce
DI = Status change interrupt routing
Bit 7 IRQ enabled
1 Status change is enabled.
Bit 6 IRQ high
1 Status change interrupt is active high. If the adapter status change level is not programmable, this setting must match the actual hardware signal level.
Bits 4-0 IRQ level

Output: AH = Error code
= 00h Successful
= 01h Bad adapter
= 06h Bad IRQ
CF = 0 Successful
= 1 Error

Bit 0 of DH (Reduce power consumption) indicates the adapter hardware is attempting to conserve power. Reduced power settings may not actually reduce power consumption because power management features are vendor-specific.

Before using the adapter, full power must be restored using this function.

If Bit 1 of DH (Preserve state information) is set to 1, all adapter and socket status are retained in reduced-power mode.

If this bit is set to 0, the software that placed the adapter in reduced-power mode must save all adapter and socket status.

Function 87h Inquire Window

This function returns information about the specified window on the specified adapter.

Input:
- AH = 87h
- AL = Adapter number (zero-based)
- BH = Window number (zero-based)
- ES:EDI = Pointer to a buffer provided by the calling program that holds window information.

Output:
- AH = Error code
  - = 00h Successful
  - = 01h Bad adapter
  - = 11h Bad window
- BL = Capabilities
  - Bit 7 Use PC Card -WAIT signal
    - 1 Windows use the -WAIT signal from a PC Card to generate additional wait states.
  - Bits 6-3 Reserved (set to 0)
  - Bit 2 I/O space
    - 1 The window can be used to map I/O ports on a PC Card to the host system I/O space.
  - Bit 1 Attribute memory
    - 1 The window can be used to map PC Card attribute memory to the host computer system memory.
  - Bit 0 Common memory
    - 1 The window can be used to map PC Card common memory to host computer system memory.
- CF = 0 Successful
  - = 1 Error
- CX = Bitmap of assignable sockets
- ES:EDI = Pointer to either the memory window characteristics table (see pages 274 through 277) or the I/O window characteristics table (see pages 277 through 279.

Memory Window Characteristics Table

```
typedef struct tagMEMWINTBL {
   FLAGS16 MemWndCaps;
   WORD  FirstByte;
   WORD  LastByte;
   WORD  MinSize;
   WORD  MaxSize;
   WORD  ReqGran;
   WORD  ReqBase;
   WORD  ReqOffset;
   WORD  Slowest;
   WORD  Fastest;
} MEMWINTBL;
```

where:

MemWndCaps is a set of memory window characteristic flags, as follows:

Memory Windows Characteristics Table Flags

Base — If set, the base address of the window is programmable within the range specified by FirstByte and LastByte. If set to 0, the window base address is fixed in system memory at the location specified in FirstByte and LastByte is undefined.

Size — If set, the window size is programmable within the range specified by MinSize and MaxSize.

Enable — If set, the windows can be disabled without reprogramming its characteristics. If 0, the calling program must preserve window state information before disabling the window.

8bit — If set, the window can be programmed for an 8-bit data bus width.

16bit — If set, the window can be programmed for a 16-bit data bus width.

Memory Window Characteristics Table Flags, cont'd

Balign — If set, the window base address must be a multiple of the windows size. If 0, the base address can be any valid address.

Pow2 — If set, a fixed-length window must be equal to a power of two of the Reqgran value. If 0, window size could be any value on a 4 KB boundary between 4 KB and 64 KB.

Calgn — If set, PC Card offsets must be in increments equal to the size of the window.

Pavail — If set, the windows can be divided into multiple pages via hardware. If 0, the window can only be addressed as a single page. If 0, the calling program must preserve page state information before disabling the page.

Pshare — If set, the window paging hardware is sharable with another window. A request to use the paging hardware may fail if another window is using it. This value is only valid if Pavail is set.

*The calling program should check **Pshare** when using window paging. If set, the calling program must make sure that a subsequent INT 1Ah AH = 89h Set Window request is successful before using the window. To determine if the page is available, assign it to a window by invoking INT 1Ah AH = 89h Set Window and make sure AH = 00h upon return from Socket Services.*

Penbl — If set, the page can be disabled without reprogramming its characteristics.

Wp — If set, the PC Card memory window mapped to the host computer system can be write-protected.

Memory Window Characteristics Table, resumed

FirstByte The first byte this window can use in the host memory system. If the window base address is not programmable, this is the same as the window base address.

LastByte The last byte this window can use in the host memory system. The last byte of the window cannot exceed this value. This value is not used if the window base address is not programmable.

MinSize The minimum window size. The window must meet all granularity and base requirements and must be within the *MinSize* and *MaxSize* values.

MaxSize The maximum window size. The window must meet all granularity and base requirements and must be within the *MinSize* and *MaxSize* values. If *MaxSize* is 0, the window size is the largest value that may be represented by the SIZE data type plus one.

ReqGran The units required for defining the windows size because of hardware constraints. If the window is a fixed size, this value is the same as Min Size and MaxSize.

ReqBase If Balign is 0, this value specifies the boundary alignment for setting the window base address via INT 1Ah AH = 89h Set Window.

ReqOfst If *Calign* is 0, this value specifies the boundary alignment for setting the window base address via INT 1Ah AH = 8Bh Set Page. This field is undefined if *Calign* is set.

Slowest This value is the slowest access speed supported by this window.

Fastest This value is the fastest access speed supported by this window.

Memory Window Characteristics Table, resumed

Slowest and *Fastest* are in the format specified by the PCMCIA Device Speed Code and Extended Device Speed Codes.

Bit 7 of *Slowest* and *Fastest* is reserved and is always set to 0.

I/O Window Characteristics Table

```
typedef struct tagIOWINTBL {
   WORD  IOWndCaps;
   WORD  FirstByte;
   WORD  LastByte;
   WORD  MinSize;
   WORD  MaxSize;
   WORD  ReqGran;
   WORD  AddrLines;
   WORD  EISASlot;
} IOWINTBL;
```

where:

IOWndCaps is a set of I/O window characteristic flags, as follows:

I/O Window Characteristics Table Flags

Base	If set, the base address of the window is programmable within the range specified by FirstByte and LastByte. If set to 0, the window base address is fixed in system I/O space at the location specified in FirstByte and LastByte is undefined.
Size	If set, the window size is programmable within the range specified by MinSize and MaxSize.
Wenable	If set, the windows can be disabled without reprogramming its characteristics. If 0, the calling program must preserve window state information before disabling the window.

I/O Window Characteristics Table Flags, cont'd

8bit — If set, the window can be programmed for an 8-bit data bus width.

16bit — If set, the window can be programmed for a 16-bit data bus width.

Balign — If set, the window base address must be a multiple of the windows size. If 0, the base address can be any valid address.

Pow2 — If set, a fixed-length window must be equal to a power of two of the Reqgran value. If 0, window size could be any value between the *MinSize* and *MaxSize* values.

Inpck — If set, the window supports the -INPACK signal from a PC Card. -INPACK allows windows to overlap in I/O space.

EISA — If set, the window supports EISA-type I/O mapping as would an EISA system. EISASlot specifies the slot-specific address decodes for this window.

Cenable — If set, EISA-like common addresses can be ignored. If 0 and the window is programmed for EISA-like I/0 mapping, the PC Card receives a Card Enable signal when an access is made to an EISA common address. This value is only valid if *EISA* is set.

I/O Window Characteristics Table, resumed

FirstByte — The first byte this window can use in the host I/O space. If the window base address is not programmable, this is the same as the window base address.

I/O Window Characteristics Table, cont'd

LastByte The last byte this window can use in the host I/O space. The last byte of the window cannot exceed this value. This value is not used if the window base address is not programmable.

MinSize The minimum window size. The window must meet all granularity and base requirements and must be within the *MinSize* and *MaxSize* values.

MaxSize The maximum window size. The window must meet all granularity and base requirements and must be within the *MinSize* and *MaxSize* values. If *MaxSize* is 0, the window size is the largest value that may be represented by the SIZE data type plus one.

ReqGran The units required for defining the windows size because of hardware constraints. If the window is a fixed size, this value is the same as Min Size and MaxSize.

AddrLins The number of address lines decoded by the window. Usually either 10 or 16. If a window only decodes 10 address lines, accesses to address above 1 KB will drive Card Accesses to a PC Card when the ten least significant address lines fall within the range defined by the base address and the window size.

EISASlot The upper byte for window-specific EISA I/O decoding. This value specifies the upper four address lines used for EISA slot-specific address that drive Card Enables. This filed is not used if *EISA* is 0.

Function 88h Get Window

This function returns the current configuration of the specified window on the specified adapter.

Input: AH = 88h
AL = Adapter number (zero-based)
BH = Window number (zero-based)

Output: AH = Error code
= 00h Successful
= 01h Bad adapter
= 11h Bad window
BL = Socket number (zero-based)
CF = 0. Successful
= 1 Error
CX = Size of window (In bytes for I/O windows. In 4 KB units for memory windows).
DH = Window state (bit-mapped)

The meaning of Bits 3 and 4 varies, depending on whether this function is reporting about an I/O window or a memory window.

Bit 4 I/O mapping (If I/O window)
0 ISA I/O mapping
1 EISA I/O mapping

Bit 4 Memory type (If memory window)
0 Common memory
1 Attribute memory

Bit 3 (If I/O window)
0 Nonspecific slot access disabled
1 Nonspecific slot access enabled

Bit 3 Memory page (if memory window)
0 Single page window
1 Window is divided into multiple 16 KB pages with PC Card offset addresses that can be set individually via Function AH = 8Bh Set Page.

Output: cont'd

DH = Window state (bit-mapped), cont'd

- Bit 2 16/8-bit data path
  - 0 The window can use an 8-bit data bus width.
  - 1 The window can use a 16-bit data bus width.
- Bit 1 Window enabling
  - 0 The window is disabled.
  - 1 The window is enabled and can map a PC Card to the host system memory or I/O space.
- Bit 0 I/O Mapping
  - 0 Common or attribute memory is mapped to the host memory space.
  - 1 PC Card registers are mapped to the host I/O space.

DL = Access speed. Select only one. Not used for I/O windows. See the PCMCIA specification for the speed codes.

DI = Windows base address (In bytes if an I/O window. In 4 KB units if a memory window).

Function 89h Set Window

This function sets the configuration of the specified window on the specified adapter. The area of the PC Card memory array mapped to the host memory is managed by the INT 1Ah AH = 8Ah Get Page and INT A1h AH = 8Bh Set Page functions for memory-mapped windows.

Input:

AH = 89h
AL = Adapter number (zero-based)
BH = Window number (zero-based)
BL = Socket number (one-based) Zero value if socket number is undefined.
CX = Window size (in 4 KB units for memory windows and in bytes for I/O windows)
DH = Window attributes (bit-mapped)

- Bits 7-6 Reserved (set to 0)
- Bit 5 Type of I/O
  - 0 ISA I/O mapping
  - 1 EISA I/O mapping.
- Bit 4 Divided into pages (memory only)
  - 0 No
  - 1 Yes
- Bit 3 Data path width
  - 0 8-bit
  - 1 16-bit
- Bit 2 Window enable or disable information
  - 0 Disable
  - 1 Enable
- Bit 1 If I/O window (Bit 0) is 1
  - 0 I/O
  - 1 EISA-mapped
- Bit 1 If memory window (bit 0) is 0
  - 0 Common memory
  - 1 Attribute memory
- Bit 0 Type of window
  - 0 Memory window
  - 1 I/O window

cont'd

Output: *Continued*

DL = Access speed
Bit 7 Reserved (set to 0)
Bit 6 600 ns
Bit 5 300 ns
Bit 4 250 ns
Bit 3 200 ns
Bit 2 150 ns
Bit 1 100 ns
Bit 0 Wait line monitoring

DI = Card offset address (memory only, 4 KB units)

SI = Window base address (bytes for I/O, 4 KB units for memory)

Output: AH = Error code
= 00h Successful
= 01h Bad adapter
= 02h Bad attribute
= 03h Bad base
= 0Ah Bad size
= 0Bh Bad socket
= 17h Bad speed
= 0Ch Bad type
= 11h Bad window

CF = 0 Successful
= 1 Error

Function 8Ah Get Page

This function returns the current configuration for the specified page in the specified window on the specified adapter.

Input:
- AH = 8Ah
- AL = Adapter number (zero-based)
- BH = Window number (zero-based)
- BL = Page number (zero-based)

Output:
- AH = Error code
  - = 00h Successful
  - = 01h Bad adapter
  - = 08h Bad page number
  - = 11h Bad window
- CF = 0 Successful
  - = 1 Error
- DI = Memory card offset (in 4 KB units)
- DL = Page attributes
  - Bits 7-3 Reserved (set to 0)
  - Bit 2 Write-protection
    - 1 Page is write-protected by page mapping hardware in the socket.
  - Bit 1 I/O mapping enable
    - 1 PC Card attribute memory is mapped to system memory or I/O space (if page is also enabled).
  - Bit 0 Memory mapping enable
    - 0 PC Card common memory is mapped to system memory (if page is also enabled).
    - 1 PC Card attribute memory is mapped to system memory (if page is also enabled).

This function is valid for memory windows, not valid for I/O windows.

The maximum page number is the window size in bytes divided by 16 KB - 1.

The associated socket number is implied by the prior INT 1Ah AH = 89h Set Window function call.

Page attributes indicate if the page is currently enabled.

If the hardware does not allow individual pages to be disabled or enabled (the entire window must be disabled or enabled), this attribute should return always with Bit 1 of DL set unless the entire window is disabled.

The memory card offset is the absolute memory card address (in 4 KB units) mapped to host system memory space for that page.

Function 8Bh Set Page

This function sets the configuration for the specified page in the specified window on the specified adapter.

Input: AH = 8Bh
AL = Adapter number (zero-based)
BH = Window number (zero-based)
BL = Page number (zero-based)
DI = Memory card offset (4 KB unit)
DL = Page attributes
Bits 7-3 Reserved (set to 0)
Bit 2 Write-protection
1 Page is write-protected by page mapping hardware in the socket.
Bit 1 I/O mapping enable
1 PC Card attribute memory is mapped to system memory or I/O space (if page is also enabled).
Bit 0 Memory mapping enable
0 PC Card common memory is mapped to system memory (if page is also enabled).
1 PC Card attribute memory is mapped to system memory (if page is also enabled).

Output: AH = Error code
= 00h Successful
= 01h Bad adapter
= 02h Bad attribute
= 07h Bad offset
= 08h Bad page
= 11h Bad window
CF = 0 Successful
= 1 Error

This function is valid for memory windows but is not valid for I/O windows.

The maximum page number is equal to the window size in bytes divided by 16 KB - 1.

The associated socket number is implied by the prior Set Window function call.

If the hardware does not allow individual pages to be disabled or enabled (the entire window can be disabled or enabled), this function should return an error on an attempt to disable a page.

The memory card offset is the absolute memory card address (in 4 KB units) mapped to host system memory space for that page.

Function 8Ch Inquire Socket

This function returns information about the specified socket on the specified adapter.

Input: AH = 8Ch
AL = Adapter number (zero-based)
BL = Socket number (zero-based)
ES:EDI = Pointer to buffer supplied by the calling program to hold the information about the socket.

Output: AH = Error code
= 00h Successful
= 01h Bad adapter
= 0Bh Bad socket

BH = Status change interrupt flags. Before an event can trigger a status change interrupt on a socket, the corresponding value in the Status Change Interrupt Mask parameter in INT 1Ah AH = 8Dh Set Socket must be set and status change interrupts must be enabled.

Bit 7 PC Card Card Detect signal
1 Enabled

Bit 6 PC Card RDY/BSY signal
1 Enabled

Bit 5 PC Card BVD2 (battery weak) signal
1 Enabled

Bit 4 PC Card BVD1 (dead battery) signal
1 Enabled

Bit 3 Externally-generated signal to insert a PC Card in the socket
1 Enabled

Bit 2 Externally-generated signal to eject a PC Card from the socket
1 Enabled

Bit 1 Externally-generated signal from a mechanical or electric card lock
1 Enabled

Bit 0 PC Card Write-Protect signal
1 Enabled

Output: cont'd

CF = 0 Successful
= 1 Error

DH = Status change events that the socket can report on. If an event is not reportable by INT 1Ah AH = 8Fh Get Status, it is set to 0. The bit settings are exactly the same as for BH on the previous page.

DL = Hardware indicators

Bit 7 XIP status
1 Enabled

Bit 6 Card busy status
1 Enabled

Bit 5 Battery status
1 Enabled

Bit 4 Card lock status
1 Enabled

Bit 3 Externally-generated signal to insert a PC Card in the socket
1 Enabled

Bit 2 Externally-generated signal to eject a PC Card from the socket
1 Enabled

Bit 1 Externally-generated signal from a mechanical or electric card lock
1 Enabled

Bit 0 PC Card Write-Protect signal
1 Enabled

ES:EDI = Pointer to buffer supplied by the calling program to hold the information about the socket. The required table structure is shown below.

Socket Information Table Structure

```
SISTRUCT SocketInfo = {
   10,            //Size of buffer provided by calling
                  //program is 10 bytes
   10,            //Size of data returned is 10 bytes
   {0,            //Indicators, power, and data bus width
                  //controlled at the socket.
   0xDEB8,        //PC Card IRQ signal can be routed to IRQs
                  // 3, 4, 5, 7, 9, 10, 11, 12, 14, and 15
                  //as an active high signal.
   0},            //PC Card IREQ routing not available on
                  //any level as an active low signal.
};
```

Socket Characteristics Structure

```
typedef struct tagSCHARTBL {  //same as adapter
   WORD SktCaps;              //except for this member
   DWORD ActiveHigh;
   DWORD ActiveLow;
} SCHARTBL;
```

where:

SktCaps are flags that specify socket characteristics.

IF_MEMRY The socket supports memory-only interfaces as per Release 2.01.

IF_IO The socket supports I/O port and memory interfaces as per Release 2.01.

ActvHgh A bitmap of the IRQ levels available for routing an inverted PC Card IREQ signal when an unmasked event occurs.

ActvLw A bitmap of the IRQ levels available for routing the normal PC Card IREQ signal when an unmasked event occurs. Normal PC Card IREQ signals can be shared in a host system.

Function 8Dh Get Socket

This function returns the current configuration of the specified socket on the specified adapter.

Input: AH = 8Dh
AL = Adapter number (zero-based)
BL = Socket number (zero-based)

Output: AH = Error code
= 00h Successful
= 01 Bad adapter
= 0Bh Bad socket
BH = Status change interrupt enable mask
Bit 7 Card detect change
1 Enabled
Bit 6 Ready change
1 Enabled
Bit 5 Battery warning change
1 Enabled
Bit 4 Battery dead change
1 Enabled
Bit 3 Insertion request
1 Enabled
Bit 2 Ejection request
1 Enabled
Bit 1 Card lock
1 Enabled
Bit 0 Write protect
1 Enabled
CF = 0 Successful
= 1 Error
CH = Bits 3-0 Vcc level
CL = Bits 7-4 Vpp1 level
Bits 3-0 Vpp2 level

Function 8Dh Get Socket, Continued

Output: cont'd

DH = Bit-mapped socket state

- Bit 7 Card detect change
  - 1 Enabled
- Bit 6 Ready change
  - 1 Enabled
- Bit 5 Battery warning change
  - 1 Enabled
- Bit 4 Battery dead change
  - 1 Enabled
- Bit 3 Insertion request
  - 1 Enabled
- Bit 2 Ejection request
  - 1 Enabled
- Bit 1 Card lock
  - 1 Enabled
- Bit 0 Write protect
  - 1 Enabled

DL = Indicators

- Bit 7 XIP status
  - 1 Enabled
- Bit 6 Card busy status
  - 1 Enabled
- Bit 5 Battery status
  - 1 Enabled
- Bit 4 Card lock status
  - 1 Enabled
- Bit 3 Externally-generated signal to insert a PC Card in the socket
  - 1 Enabled
- Bit 2 Externally-generated signal to eject a PC Card from the socket
  - 1 Enabled
- Bit 1 Externally-generated signal from a mechanical or electric card lock
  - 1 Enabled
- Bit 0 PC Card Write-Protect signal
  - 1 Enabled

Output: cont'd

DI = IRQ level steering (valid I/O cards only)

Bit 9	I/O and memory interface	
	1 Enabled	
Bit 8	Memory interface	
	1 Enabled	
Bit 7	IRQ enabled	
	1 Enabled	
Bit 5	IRQ high	
	1 Enabled	
Bits 4-0	IRQ level	
	00h-0Fh	IRQ 00h-0Fh
	10h	NMI
	11h	I/O check
	12h	Bus error
	13h	Vendor-unique

The voltage levels Vcc, Vpp1, Vpp2 are the indexes into power management table.

Function 8Eh Set Socket

This function sets the current configuration of the specified socket on the specified adapter.

Input:

AH	=	8Eh	
AL	=	Adapter number (zero-based)	
BL	=	Socket number (zero-based)	
BH	=	Status change interrupt enable mask	
		Bit 7	Card detect change
			1 Enabled
		Bit 6	Ready change
			1 Enabled
		Bit 5	Battery warning change
			1 Enabled
		Bit 4	Battery dead change
			1 Enabled
		Bit 3	Insertion request
			1 Enabled
		Bit 2	Ejection request
			1 Enabled
		Bit 1	Card lock
			1 Enabled
		Bit 0	Write protect
			1 Enabled
CH	=	Bits 3-0	Vcc level
CL	=	Bits 7-4	Vpp1 level
		Bits 3-0	Vpp2 level

Input: cont'd

DH = Bit-mapped socket attributes

- Bit 7 Card detect change
  - 1 Enabled
- Bit 6 Ready change
  - 1 Enabled
- Bit 5 Battery warning change
  - 1 Enabled
- Bit 4 Battery dead change
  - 1 Enabled
- Bit 3 Insertion request
  - 1 Enabled
- Bit 2 Ejection request
  - 1 Enabled
- Bit 1 Card lock
  - 1 Enabled
- Bit 0 Write protect
  - 1 Enabled

DL = Indicators

- Bit 7 XIP status
  - 1 Enabled
- Bit 6 Card busy status
  - 1 Enabled
- Bit 5 Battery status
  - 1 Enabled
- Bit 4 Card lock status
  - 1 Enabled
- Bit 3 Externally-generated signal to insert a PC Card in the socket
  - 1 Enabled
- Bit 2 Externally-generated signal to eject a PC Card from the socket
  - 1 Enabled
- Bit 1 Externally-generated signal from a mechanical or electric card lock
  - 1 Enabled
- Bit 0 PC Card Write-Protect signal
  - 1 Enabled

Input: *Continued*

DI = IRQ level steering (valid for I/O cards only)

Bit 9	I/O and memory interface	
	1 Enabled	
Bit 8	Memory interface	
	1 Enabled	
Bit 7	IRQ enabled	
	1 Enabled	
Bit 5	IRQ high	
	1 Enabled	
Bits 4-0	IRQ level	
	00h-0Fh	IRQ 00h-0Fh
	10h	NMI
	11h	I/O check
	12h	Bus error
	13h	Vendor-unique

Output: AH = Error code
= 00h Successful
= 01h Bad adapter
= 02h Bad attribute
= 0Bh Bad socket

CF = 0 Successful
= 1 Error

This function waits until the requested Vpp power level becomes valid.

Function 8Fh Get Status

This function returns the status of a PC Card in the specified socket on the specified adapter. This function must not be invoked during hardware interrupt processing. It should not be invoked by the calling program's status change hardware interrupt handler.

Input: AH = 8Fh
AL = Adapter number (zero-based)
BL = Socket number (zero-based)

Output: AH = Error code
= 00h Successful
= 01h Bad adapter
= 0Bh Bad socket
BH = Card state
Bit 7 Card changed
1 Enabled
Bit 6 Reserved (set to 0)
Bit 5 Card insertion complete
1 Enabled
Bit 4 Card ejection complete
1 Enabled
Bit 3 Card insertion request pending
1 Enabled
Bit 2 Card ejection request pending
1 Enabled
Bit 1 Card lock
1 Enabled
Bit 0 Write protect
1 Enabled
CF = 0 Successful
= 1 Error

Output: cont'd

DH = Socket state
- Bit 7 Card changed
  - 1 Enabled
- Bit 6 Reserved (set to 0)
- Bit 5 Card insertion complete
  - 1 Enabled
- Bit 4 Card ejection complete
  - 1 Enabled
- Bit 3 Card insertion request pending
  - 1 Enabled
- Bit 2 Card ejection request pending
  - 1 Enabled
- Bit 1 Card lock
  - 1 Enabled
- Bit 0 Write protect
  - 1 Enabled

DL = Card attributes (bit-mapped)
- Bit 7 XIP status
  - 1 Enabled
- Bit 6 Card busy status
  - 1 Enabled
- Bit 5 Battery status
  - 1 Enabled
- Bit 4 Card lock status
  - 1 Enabled
- Bit 3 Externally-generated signal to insert a PC Card in the socket
  - 1 Enabled
- Bit 2 Externally-generated signal to eject a PC Card from the socket
  - 1 Enabled
- Bit 1 Externally-generated signal from a mechanical or electric card lock
  - 1 Enabled
- Bit 0 PC Card Write-Protect signal
  - 1 Enabled

Output: cont'd

DI = IRQ level steering (valid I/O cards only)

Bit 9	I/O and memory interface	
	1 Enabled	
Bit 8	Memory interface	
	1 Enabled	
Bit 7	IRQ enabled	
	1 Enabled	
Bit 5	IRQ high	
	1 Enabled	
Bits 4-0	IRQ level	
	00h-0Fh	IRQ 00h-0Fh
	10h	NMI
	11h	I/O check
	12h	Bus error
	13h	Vendor-unique

Function 90h Reset Socket

This function resets the specified socket on the specified adapter and returns the socket hardware to the power-on default state: Vcc, Vpp1, and Vpp2 are set to 5VDC, IRQ routing is disabled, memory-type mapping is set, and all windows, pages, and EDC generators are disabled. The calling program must make sure that a PC Card is not accessed before ready after this function returns.

Input: AH = 90h
AL = Adapter number (zero-based)
BL = Socket number (zero-based)

Output: AH = Error code
= 00h Successful
= 01h Bad adapter
= 0Bh Bad socket
= 14h No PC Card in socket
CF = 0 Successful
= 1 Error

This function sets the RESET pin on the card to the reset state and then resets the RESET pin to non-reset state, ensuring that the minimum reset pulse width is met. The caller must ensure that the card is not accessed before it is ready after returning.

Function 95h Inquire EDC

This function returns the capabilities of the specified EDC (Error Detection Code) generator.

Socket Services supports two types of EDC generation: 8-bit checksums and 16-bit CRC SDLC.

EDC generation can be produced by read or write accesses. Code that uses many sequential reads and writes must use EDC generation carefully. Bidirectional EDC generation may not work with flash EPROM programming routines because these routines typically require many reads and writes.

EDC generation may not be available with memory-mapped implementations. EDC generators must be configured via INT 1Ah AH = 97h Set EDC.

Input:	AH	=	95h
	AL	=	Adapter number (zero-based)
	BH	=	EDC generator number (zero-based)
Output:	AH	=	Error code
		=	00h Successful
		=	01h Bad adapter
		=	04h Bad EDC
	CF	=	0 Successful
		=	1 Error
	CX	=	Assignable sockets (Bit 0 is socket 0, bit 1 is socket 1, etc)

Output: cont'd

DH = EDC capabilities (Bit-mapped)

- Bits 7-5 Reserved (set to 0)
- Bit 4 Pausable EDC
  - 1 EDC generation can be paused.
- Bit 3 Memory-mapped support
  - 1 EDC generation is supported during window access.
- Bit 2 Register-based support
  - 1 EDC generation is supported through register-based access.
- Bit 1 Bidirectional code generation
  - 1 The EDC generator supports bidirectional code generation.
- Bit 0 Unidirectional code generation
  - 1 The EDC generator supports unidirectional code generation.

DL = Supported EDC types

- Bits 7-2 Reserved (set to 0)
- Bit 1 16-Bit CRC-SDLC
  - 1 The EDC generator supports 8-bit checksum code generation.
- Bit 0 8-Bit checksum
  - 1 The EDC generator supports 8-bit checksum code generation.

Not every hardware implementation provides EDC code generation.

The output of this function describes the EDC functions of the specified EDC generator.

EDC generators can be shared between sockets.

Card Services or higher-level software arbitrates the use of EDC generators.

Function 96h Get EDC

This function returns the current configuration of the specified EDC generator. A generator is not assigned if the socket number returned is zero.

Input:
- AH = 96h
- AL = Adapter number (zero-based)
- BH = EDC generator number (zero-based)

Output:
- AH = Error code
  - = 00h Successful
  - = 01h Bad adapter
  - = 04h Bad EDC
- BL = Socket number of the physical socket that the EDC generator is assigned to (zero-based).
- CF = 0 Successful
  - = 1 Error
- DH = EDC attributes (Bit-mapped)
  - Bits 7-2 Reserved (set to 0)
  - Bit 1 If unidirectional only (Bit 0) is 1
    - 0 EDC computing only on read accesses.
    - 1 EDC computing only on write accesses.
  - Bit 0 Unidirectional only
    - 0 EDC computing on both read and write accesses.
    - 1 EDC computing in only one direction.
- DL = EDC type (mutually exclusive bitmap)
  - Bits 7-2 Reserved (set to 0)
  - Bit 1 16-Bit CRC-SDLC EDC checksum generated by EDC.
  - Bit 0 8-Bit checksum generated by EDC.

Function 97h Set EDC

This function sets the error detection and correction configuration of the specified EDC generator.

Input:

AH = 97h
AL = Adapter number (zero-based)
BH = EDC generator number (zero-based)
BL = Socket number (zero-based)
DH = EDC attributes (Bit-mapped)
- Bits 7-2 Reserved (set to 0)
- Bit 1 EDC computes on reads or writes
  - 0 Reads
  - 1 Writes
- Bit 0 Unidirectional
  - 1 EDC generator compute in only one direction.

DL = EDC type (mutually exclusive bitmap)
- Bits 7-2 Reserved (set to 0)
- Bit 1 16-Bit CRC-SDLC
  - 1 16-bit EDC checksum generated.
- Bit 0 8-Bit CRC-SDLC
  - 1 8-bit EDC checksum generated.

Output:

AH = Error code
= 00h Successful
= 01h Bad adapter
= 02h Bad attribute
= 04h Bad EDC
= 0Bh Bad socket
CF = 0 Successful
= 1 Error

Function 98h Start EDC

This function starts the specified previously configured EDC generator. This function load initialization values into the EDC generator.

Input:	AH	=	98h
	AL	=	Adapter number (zero-based)
	BH	=	EDC generator number (zero-based)
Output:	AH	=	Error code
		=	00h Successful
		=	01h Bad adapter
		=	04h Bad EDC
	CF	=	0 Successful
		=	1 Error

Function 99h Pause EDC

This function pauses EDC generation on the specified configured and computing EDC generator. This function is only supported if Bit 4 of DH is set when INT 1Ah AH= 95h Inquire EDC is invoked.

Input: AH = 99h
AL = Adapter number (zero-based)
BH = EDC generator number (zero-based)

Output: AH = Error code
= 00h Successful
= 01h Bad adapter
= 04h Bad EDC
CF = 0 Successful
= 1 Error

Function 9Ah Resume EDC

This function resumes the EDC generation on the specified configured and paused EDC generator. This function can only be used if bit 4 of DH as returned by the INT 1Ah AH = 95h Inquire EDC function is set.

Input: AH = 9Ah
AL = Adapter number (zero-based)
BH = EDC generator number (zero-based)

Output: AH = Error code
= 00h Successful
= 01h Bad adapter
= 04h Bad EDC
CF = 0 Successful
= 1 Error

Function 9Bh Stop EDC

This function stops the EDC generation on the specified configured and computing EDC generator.

Input: AH = 9Bh
AL = Adapter number (zero-based)
BH = EDC generator number (zero-based)

Output: AH = Error code
= 00h Successful
= 01h Bad adapter
= 04h Bad EDC
CF = 0 Successful
= 1 Error

Function 9Ch Read EDC

This function reads the calculated EDC value computed by the specified EDC generator. The computed value may be incorrect if the EDC generator has been used incorrectly.

Input: AH = 9Ch
AL = Adapter number (zero-based)
BH = EDC generator number (zero-based)

Output: AH = Error code
= 00h Successful
= 01h Bad adapter
= 04h Bad EDC
CF = 0 Successful
= 1 Error
DX = Computed checksum or CRC. This can be an 8-bit or 16-bit value depending on the value of Bits 0 and 1 in DL as returned by INT 1Ah AH = 95h Inquire EDC.

Function 9Dh Get Vendor Info

This function returns information about the vendor implementing Socket services for the specified adapter.

Input:	AH	=	9Dh
	AL	=	Adapter number (zero-based)
	BH	=	EDC generator number (zero-based)
	ES:EDI	=	Address of buffer where vendor information is stored
Output:	AH	=	Error code
		=	00h Successful
		=	01h Bad adapter
		=	15h Bad function
	CF	=	0 Successful
		=	1 Error
	ES:EDI	=	Address of buffer where vendor information is stored
	DX	=	Vendor release number in BCD

The buffer pointed to by the value in ES:EDI must have the following format:

```
typedef struct tagVISTRUCT {
   WORD wBufferlength = (BUF_SIZE - 4);
   WORD wDataLength;     Set by Socket Services
   char szImplementor[BUF_SIZE - 4];
} VISTRUCT;
```

If the wData Length value is greater than the wBufferLength value, the information is truncated.

Function 9Eh Acknowledge Interrupt

This function returns status change information for sockets on the specified adapter. Socket Services does not enable interrupts while this function is being performed.

The calling program should enable status change interrupts from adapter hardware via INT 1Ah AH = 86h Set Adapter.

The calling program must install an interrupt handler on the appropriate vector.

Specific events can be masked or unmasked for each socket via INT 1Ah AH = 8Eh Set Socket.

When a status change occurs, the calling program's status change handler receives control and invokes INT 1Ah AH = 9Eh Acknowledge Interrupt. This function permits Socket Services to prepare the adapter hardware to generate another interrupt if another status change occurs.

Socket Services preserves status change information if it is not preserved by the adapter hardware.

If this function is called and no status change has occurred on the specified adapter, Socket Services returns with AH and CX = 00h.

Input:	AH	=	9Eh
	AL	=	Adapter number (zero-based)
Output:	AH	=	Error code
		=	00h Successful
		=	01h Bad adapter
	CF	=	0 Successful
		=	1 Error
	CX	=	A bitmap that represents the sockets that have changed status.

Function 9Fh Get and Set Prior Handler

This function replaces or acquires the entry point of a prior handler for the specified adapter.

If this Socket Services handler is the first installed in the INT 1Ah chain, the values returned when this function is issued with BL = 0 should be the entry point to the Time of Day handler.

This function might fail if the Socket Services it addresses are in the system BIOS ROM as the first extension to the Time of Day handler. To circumvent this problem, register the value returned by this function to this Socket Services with a replacement Socket Services implementation.

Warning

This function should only be used with the first adapter serviced by a Socket Services handler as returned by Function 80h Get SS Info. If a handler services more than one adapter, subsequent requests to the handler for adapters other than the first adapter will return the same information and set the same internal variables.

Warning

A calling program should not add Socket Services that increase the number of adapters or sockets supported.

To provide support for additional adapters and sockets, new Socket Services handlers should be added to the end of the handler chain. Adjusting internal prior handlers should be used only to replace an old Socket Services implementation with an updated version.

Function 9Fh Get and Set Prior Handler, Continued

Input:

AH	=	9Fh
AL	=	Adapter number (zero-based)
BL	=	Mode 00h Get prior handler 01h Set prior handler
CX:DX	=	If BL = 1, contains a pointer to a new prior handler. It now returns the entry point of the old prior handler.

Output:

AH	=	Error code 00h Successful 01h Bad adapter 15h Bad function
CF	=	0 Successful
	=	1 Error
CX:DX	=	Contains a pointer to a new prior handler and returns the entry point of the old prior handler.

Function A0h Get and Set SS Addr

Warning

This function should only be used with the first adapter serviced by a Socket Services handler as returned by Function 80h Get SS Info. If a handler services more than one adapter, subsequent requests to the handler for adapters other than the first adapter will return the same information and set the same internal variables.

This function returns code and data area descriptions and provides a method for passing address mode-specific data area descriptors to a Socket Services handler.

If Socket Services must access other memory regions, the value in CX is the number of unique memory regions that Socket Services must address as well as the main data segment.

Card Services uses the entry point returned by this function to establish the appropriate address mode-specific pointers to the code and main data areas before calling the entry point.

The entry points returned by this function must receive control from a CALL instruction. The real mode, 16:16, and 16:32 entry points require a FAR CALL. The 00:32 entry point requires a NEAR CALL. When using an entry point that has been returned by this function in all address modes except real mode, the calling program must establish a pointer to the main data area in DS:ESI.

Input:

AH	=	A0h	
AL	=	Adapter number (zero-based)	
BH	=	Mode	
		00h	Real mode
		01h	16:16 Protected mode
		02h	16:32 Protected mode
		03h	00:32 Protected mode

Input: cont'd

BL = Subfunction

- BL = 00h Socket Services returns the number of additional data areas in this parameter.
- BL = 01h Socket Services returns a description of any additional data areas in the buffer supplied by the calling program at ES:EDI.
- BL = 02h Socket Services accepts the number of mode-specific pointers to additional data areas in the buffer pointed to in ES:EDI specified in CX.

ES:EDI = Contains a pointer to a buffer supplied by the calling program. The buffer must be the appropriate length.

Output: AH = Error code

- 00h Successful
- 01h Bad adapter
- 02h Bad attribute
- 15h Bad function
- 16h Bad mode

CF = 0 Successful

= 1 Error

CX = Number of additional data areas.

- If BL = 00h Socket Services returns the number of additional data areas in this parameter.
- If BL = 01h Socket Services returns a description of any additional data areas in the buffer supplied by the calling program at ES:EDI.
- If BL = 02h Socket Services accepts the number of mode-specific pointers to additional data areas in the buffer pointed to in ES:EDI specified in CX.

ES:EDI = Contains a pointer to a buffer supplied by the calling program. The buffer must be the appropriate length.

> ***Warning***
>
> Any CS selector should be readable and executable so Socket Services can reference constant data that may reside in ROM. The calling program must also make sure that Socket Services has the appropriate privileges to permit access to I/O ports.

Buffer Table Entry if BL = 00h

Offset	Description
00h	32-bit linear base address of the code segment in system memory.
04h	Limit of the code segment. This value must be less than 64 KB in real mode and 16:16 in protected mode.
08h	Entry point offset. This value must be less than 64 KB in real mode and 16:16 in protected mode.
0Ch	32-bit linear base address of the main data segment in system memory. This field is ignored if 00:32 (flat) protected mode addressing is used.
10h	The limit of the data segment. This value must be less than 64 KB in real mode and 16:16 in protected mode.
14h	The data area offset. This field is only used if 32-bit protected mode addressing is used.

Buffer Table Entry if BL = 01h

Offset	Description
00h	32-bit linear base address of the additional data segment in system memory. This field is ignored if 00:32 (flat) protected mode addressing is used.
04h	Limit of the code segment. This value must be less than 64 KB in real mode and 16:16 in protected mode.
08h	Data area offset. This field is only used if 00:32 (flat) protected mode addressing is used.

Buffer Table Entry if BL = 02h

Offset	Description
00h	32-bit offset. This field is ignored if 16:16 protected mode addressing is used. 16:16 protected mode addressing assumes 0 in this field.
04h	Selector. This field is only used if 00:32 (flat) protected mode addressing is used.
08h	Reserved

Function A1h Get Access Offsets

This function fills the buffer pointed to by ES:EDI with an array of offsets for low-level, adapter-specific, optimized PC Card access routines for adapters that use registers or I/O ports to access PC Card memory. Adapters that access PC Card memory through windows mapped to host system memory do not support this function.

It is assumed that all requested offsets are in the Socket Service code segment. All sockets on an adapter must use the same entry point for a certain address mode. These offsets can be different for different address modes. A calling program can use the values returned by this function to create an internal table, permitting the routines at these offsets to be called in a manner appropriate to the address mode they will be used in.

16-bit offsets are returned in all modes. The offset must be combined with information returned by Function A0h Get and Set SS Addr that describes the location of the code segment. Offsets returned by this function are relative to the code segment.

For real, 16:16, and 16:32 address modes, the routines at these offsets use FAR RET instructions to return to the calling program, so this function must be invoked with a FAR CALL instruction. In 00:32 (flat) protected address mode, the routines at the returned offsets use NEAR RET instructions and must be invoked with a NEAR CALL instruction.

Input:

AH	=	A1h
AL	=	Adapter number (zero-based)
BH	=	Mode
		00h Real mode
		01h 16:16 Protected mode
		02h 16:32 Protected mode
		03h 00:32 Protected mode
CX	=	Number of access offsets
ES:EDI	=	Pointer to a buffer supplied by the calling program for the array of access offsets. The value in CX specifies the number of entries in the buffer.

Output: AH = Error code
= 00h Successful
= 01h Bad adapter
= 15h Bad function
= 16h Bad Mode
CF = 0 Successful
= 1 Error
DX = Number of access offsets supported by this Socket Services handler for the specified adapter.
ES:EDI = Pointer to a buffer supplied by the calling program for the array of access offsets. The value in CX specifies the number of entries in the buffer.

Offset Order

Offsets are returned in the following order:

1 Set Address,
2 Set Auto Increment,
3 Read Byte,
4 Read Word,
5 Read Byte with Auto Increment,
6 Read Word with Auto Increment,
7 Read Words,
8 Read Words with Auto Increment,
9 Write Byte,
10 Write Word,
11 Write Byte with Auto Increment,
12 Write Word with Auto Increment,
13 Write Words,
14 Write Words with Auto Increment,
15 Compare Byte,
16 Compare Byte with Auto Increment,
17 Compare Words, and
18 Compare Word with Auto Increment.

Function AEh Vendor-Specific

This function handles vendor-specific information. The vendor can add proprietary extensions to Socket Services via this interface.

See the vendor technical documentation for additional information about INT 1Ah AH = AEh.

Input: AH = AEh
AL = Adapter number (zero-based)
all other registers are vendor-specific

Output: AH = Error code
CF = 0 Successful
= 1 Error

Socket Services Error Codes

Code	Explanation
00h	Successful
01h	Invalid adapter
02h	Invalid attribute
03h	Invalid base system memory address
04h	Invalid EDC generator
06h	Invalid IRQ level
07h	Invalid card offset
08h	Invalid Page
09h	Incomplete read request
0Ah	Invalid window size
0Bh	Invalid socket
0Ch	Invalid window type
0Dh	Invalid Vcc level
0Eh	Invalid Vpp1 and Vpp2 level
11h	Invalid window
12h	Incomplete write request
14h	No card present
15h	Function not supported
16h	Invalid mode
17h	Invalid speed
18h	Busy

INT 1Ah Function 9Dh Intel ExCA Card Service Functions

INT 1Ah Function 9Dh supports the following Card Service functions, as specified in the Intel ExCA specifications.

Type	AL Value	Function
Client Services	00h	Get Number of Sockets
	02h	Register Client
	03h	Deregister Client
	05h	Register SCB
	06h	Deregister SCB
	0Ah	Get Status
	0Bh	Reset Card
	1Ch	Modify Window
	1Eh	Map Mem Page
Resource Management	19h	Request I/O
	1Ah	Release I/O
	1Bh	Request Memory
	1Dh	Release Memory
	22h	Request IRQ
	23h	Release IRQ
Bulk Memory Services	14h	Open Region
	15h	Read Memory
	16h	Write Memory
	17h	Copy Memory
	18h	Erase Memory
	24h	Close Region
Client Utilities	0Ch	Get First Tuple
	0Dh	Get Next Tuple
	0Eh	Determine First Region
	0Fh	Determine Next Region
	10h	Get First Region
	11h	Get Next Region
	12h	Get First Partition
	13h	Get Next Partition
Advanced Client Services	1Fh	Return SS Entry
	20h	Map Log To Phy
	21h	Map Log Phy To Log
	01h	Initialize
	04h	Enumerate Clients
	07h	Register MTD
	08h	Deregister MTD
	09h	Enumerate MTDs

Peripheral Component Interconnect (PCI) BIOS Calls

PCI is a way to physically interconnect highly integrated peripheral components and processor/memory systems. PCI BIOS functions provide a software interface to the PCI hardware.

PCI is an Intel specification for a 486 CPU Local Bus standard. The PCI specification also provides the electrical specifications for peripheral chip makers and the logic requirements for a PCI Controller. PCI establishes a local bus standard that permits a large variety of I/O components to be directly connected to the CPU bus using no glue logic. The PCI architecture is essentially a CPU-to-local bus bridge with FIFO buffers. PCI signals are multiplexed.

Unlike other local bus specifications, PCI has a standalone PCI Controller to manage the data transfer between PCI peripherals and the memory/CPU.

PCI Features

Up to ten PCI peripherals can be used in the same system on the PCI bus, including the PCI Controller and an optional expansion bus controller for EISA, ISA, or MCA. PCI uncouples the CPU from the expansion bus while still maintaining a 33 MHz 32-bit path to peripheral devices. The PCI bus works at 33 MHz and can use either a 32-bit or 64-bit data connection path to the CPU.

The PCI Controller queues reads and writes between the memory/CPU and PCI peripheral devices.

Concurrent Operation

The CPU in a PCI system runs concurrently with PCI bus mastering peripherals. Although bus mastering peripheral devices are specified, impressive data transfer rates can be achieved without splitting resource utilization between the CPU and a bus mastering device. PCI peripheral devices can operate at data transfer rates up to 33 MBs in an ISA environment.

PCI Bus Mastering

Up to ten bus mastering devices can operate simultaneously on the PCI bus. PCI devices can be bus masters, slaves, or a combination of bus master and slave. The PCI specification also provides for full burst mode for both reads and writes. The 486 CPU only permits burst mode on reads.

Multiplexing

PCI is a multiplexed version of the Intel 80486 bus. Multiplexing allows more than one signal to be sent on the same electrical path. The control mechanisms are extended to optimize I/O support.

PCI Device Drivers

The system BIOS in a PCI system provides information about where the device is in memory or I/O space and which interrupt vector the device will generate. This information comes directly from the configuration registers of the peripheral component, not from CMOS RAM or an internal BIOS table. PCI BIOS functions can access these configuration registers and provide this information.

Expansion ROM Code

All expansion ROM in a PCI system must be fully relocatable. It must be able to call a PCI system BIOS function to see where its device was placed in memory or I/O space.

PCI BIOS Interface

All software in a system with the PCI bus should use system BIOS functions to access PCI features. The system BIOS in a PCI system supports multiple operating and addressing modes. Some of the functions of the system BIOS in a PCI system are:

- allows the calling program to find a PCI Controller,
- provides access to special PCI functions,
- allows the calling program to determine the interrupt level, and
- allows the calling program to access configuration space (either memory or I/O ports).

PCI BIOS Calls

PCI-specific BIOS function calls can be used in real mode, 16-bit protected mode, or 32-bit protected mode. Real mode function calls are made via INT 1Ah AH = B1h. Protected mode access is provided by calling the BIOS through a protected mode entry point, specified by calling INT 1Ah Function B1h AL = 01h/81h PCI BIOS Present.

INT 1Ah Function B1h Calling Conventions

Every PCI function can be invoked with two codes: one for 32-bit mode and the other for all other modes. The EAX, EBX, ECX, and EDX registers and all flags may be modified by every function call. All other registers will be preserved. CF indicates the completion status of the function call.

Protected Mode PCI BIOS Function Calls

Access the protected mode interface by calling through a protected mode entry point provided by the INT 1Ah Function B1h AL = 01h/81h PCI BIOS Present function. The code segment descriptor must specify protection level 0. All INT 1Ah Function B1h PCI BIOS functions must be invoked with CPL = 0. The code segment descriptor must permit access to the 64 KB of code that starts at the 16-byte boundary immediately below the protected mode entry point.

Function B1h Subfunction AL = 01/81 PCI BIOS Present

This subfunction indicates if the PCI BIOS interface is present. The current PCI BIOS interface version level is also returned. Information about hardware mechanisms for accessing PCI configuration space and PCI Special Cycles support is also provided.

Input:
- AH = B1h
- AL = 01h real mode operation
- = 81h protected mode operation
- BH = EDC generator number (zero-based)

Output:
- AH = 00h PCI BIOS interface present
- = Any other value is an error code.
- AL = Hardware mechanism
  - 5 1 Special cycle supported via Config mechanism 1
  - 4 1 Special cycle supported via Config mechanism 2
  - 1 1 Config. Mechanism #2 supported
  - 0 1 Config. Mechanism #1 supported
- BH = Interface Level Major Version (in BCD)
- BL = Interface Level Minor Version (in BCD)
- CF = 0 PCI BIOS interface present
- = 1 No PCI BIOS interface present
- CL = Number of PCI buses in system (zero-based)
- EDI = Physical address of entry point to PCI BIOS functions for protected mode access
- EDX = " PCI"

Function B1h Subfunction AL = 02/82 Find PCI Device

This subfunction returns the location of PCI devices. Specify the Device ID in CX, Vendor ID in DX, and a Device Index in SI. This function returns the PCI bus number in BL and the Device Number of the specified *(nth)* device in BH.

You can find all PCI devices with the same Vendor ID and Device ID by making consecutive calls to this function and incrementing the Device Index by one each time until code 86h is returned in AH.

Input:	AH	=	B1h
	AL	=	02h real mode operation
		=	81h protected mode operation
	CX	=	Device ID (0 through 65535)
	DX	=	Vendor ID (1 through 65534)
	SI	=	Device Index (0 through *n*)
Output:	AH	=	00h Successful
		=	82h Incorrect Device ID
		=	83h Incorrect Vendor ID
		=	86h Device not found
	BH	=	Bits 7-3 Device Number
	BL	=	Bus Number (0 through 255)
	CF	=	0 No error
		=	1 Error

Function B1h Subfunction AL = 03/83 Find PCI Class Code

This subfunction returns the location of PCI devices with the specified Class Code. Specify the Class Code in ECX and a Device Index in SI. The function returns the Bus Number in BL, the Device Number in BH, and the Function Number of the *nth* device in the bottom three bits of BH.

You can find all PCI devices with the same Class Code by making consecutive calls to this function and incrementing the Device Index by one each time until code 86h is returned in AH.

Input:	AH	=	B1h
	AL	=	03h real mode operation
		=	83h protected mode operation
	ECX	=	Class Code in low three bytes
	SI	=	Device Index (0 through *n*)
Output:	AH	=	00h Successful
		=	86h Device not found
	BH	=	Bits 7-3 Device Number
			Bits 1-0 Function Number
	BL	=	Bus number (0 through 255)
	CF	=	0 No error
		=	1 Error

Function B1h Subfunction AL = 06/86 Generate Special Cycle

This subfunction generates PCI Special Cycles that are broadcast on a specific PCI bus.

Input:	AH	=	B1h
	AL	=	06h real mode operation
		=	86h protected mode operation
	EDX	=	Special Cycle Data
Output:	AH	=	00h Successful
		=	81h Function not supported
	CF	=	0 No error
		=	1 Error

Function B1h Subfunction AL = 08/88 Read Configuration Byte

This subfunction reads individual bytes from the configuration space of the specified PCI device.

Input:	AH	=	B1h
	AL	=	08h real mode operation
		=	88h protected mode operation
	BH	=	Bits 7-3 Device Number
			Bits 2-0 Function Number
	BL	=	Bus Number (0 through 255)
	DI	=	Register Number (0 through 255)
Output:	AH	=	00h Successful
		=	84h Incorrect Bus Number
	CF	=	0 No error
		=	1 Error
	CL	=	Byte read

Function B1h Subfunction AL = 09/89 Read Configuration Word

This subfunction reads individual words from the configuration space of the specified PCI device. The Register Number must be a multiple of 2.

Input:	AH	=	B1h
	AL	=	09h real mode operation
		=	89h protected mode operation
	BH	=	Bits 7-3 Device Number
			Bits 2-0 Function Number
	BL	=	Bus Number (0 through 255)
	DI	=	Register Number (0 through 255)
Output:	AH	=	00h Successful
		=	84h Incorrect Bus Number
		=	87h Incorrect Register Number
	CF	=	0 No error
		=	1 Error
	CX	=	Word read

Function B1h Subfunction AL = 0A/8A Read Configuration Dword

This subfunction reads individual doublewords from the configuration space of the specified PCI device. The Register Number must be a multiple of 4.

Input:	AH	=	B1h
	AL	=	0Ah real mode operation
		=	8Ah protected mode operation
	BH	=	Bits 7-3 Device Number
			Bits 2-0 Function Number
	BL	=	Bus Number (0 through 255)
	DI	=	Register Number (0 through 255)
Output:	AH	=	00h Successful
		=	84h Incorrect Bus Number
		=	87h Incorrect Register Number
	CF	=	0 No error
		=	1 Error
	ECX	=	Doubleword read

Function B1h Subfunction AL = 0B/8B Write Configuration Byte

This subfunction writes individual bytes to the configuration space of the specified PCI device.

Input:	AH	=	B1h
	AL	=	0Bh real mode operation
		=	8Bh protected mode operation
	BH	=	Bits 7-3 Device Number
			Bits 2-0 Function Number
	BL	=	Bus Number (0 through 255)
	CL	=	Byte value to write
	DI	=	Register Number (0 through 255)
Output:	AH	=	00h Successful
		=	84h Incorrect Bus Number
	CF	=	0 No error
		=	1 Error

Function B1h Subfunction AL = 0C/8C Write Configuration Word

This subfunction writes individual words to the configuration space of the specified PCI device. The Register Number must be a multiple of 2.

Input:	AH	=	B1h
	AL	=	0Ch real mode operation
		=	8Ch protected mode operation
	BH	=	Bits 7-3 Device Number
			Bits 2-0 Function Number
	BL	=	Bus Number (0 through 255)
	CX	=	Word value to write
	DI	=	Register Number (0 through 255)
Output:	AH	=	00h Successful
		=	84h Incorrect Bus Number
		=	87h Incorrect Register Number
	CF	=	0 No error
		=	1 Error

Function B1h Subfunction AL = 0D/8D Write Configuration Dword

This subfunction writes individual doublewords to the configuration space of the specified PCI device. The Register Number must be a multiple of 4.

Input:

AH	=	B1h
AL	=	0Dh real mode operation
	=	8Dh protected mode operation
BH	=	Bits 7-3 Device Number
		Bits 2-0 Function Number
BL	=	Bus Number (0 through 255)
ECX	=	Doubleword value to write
DI	=	Register Number (0 through 255)

Output:

AH	=	00h Successful
	=	84h Incorrect Bus Number
	=	87h Incorrect Register Number
CF	=	0 No error
	=	1 Error

INT 1Ah Function B1h Error Codes

The following error codes may appear in AH after any INT 1Ah Function B1h function call.

AH Value	Description
00h	Successful
81h	Function Not Supported
82h	Incorrect Device ID
83h	Incorrect Vendor ID
84h	Incorrect Bus Number
86h	Device Not Found
87h	Incorrect Register Number
EEh	Internal Error

INT 1Bh <Ctrl> <Break>

Input: None

Output: None

Description:

INT 1Bh is called by the operating system to terminate the current application when you press <Ctrl> <Break>. The BIOS sets this routine to an IRET instruction. The next time the operating system boots, it resets the routine to point to its own interrupt service routine.

INT 1Ch Periodic Timer Interrupt

Input: None

Output: None

Description:

The system timer calls INT 08h 18.2 times per second. After each call to INT 08h, INT 1Ch is called to permit access to the system timer by any applications program. The BIOS sets this routine to an IRET instruction. The next time the operating system boots, it resets the routine to point to its own interrupt service routine.

INT 1Dh Video Parameter Table

Input: None

Output: None

Description:

The vector for INT 1Dh points to a table of video parameters.

INT 1Eh Floppy Disk Parameter Table

Input: None

Output: None

Description:

The vector for INT 1Eh points to a table of floppy disk parameters.

INT 1Fh Video Graphics Characters

Input: None

Output: None

Description:

The vector for INT 1Fh points to a table of video graphics characters.

INT 4Ah Alarm ISR

When the Real Time Clock alarm is activated, the Real Time Clock generates an interrupt request at the time specified in INT 1Ah Function 06h. INT 4Ah can be invoked when the alarm occurs. The program that issued INT 1Ah Function 06h must redirect the INT 4Ah vector (0:128h) to a routine that processes the alarm.

INTs 70h through 77h

An ISA system has two interrupt controllers. The second controller calls INTs 70h to 77h. Only INTs 70h, 74h, 75h, and 76h are described in this book. The programmer cannot revector any of the interrupts from INT 70h – 77h to his own routine.

INT 70h Real Time Clock Interrupt (IRQ8)

Input: None

Output: None

Description:

The AMIBIOS services INT 70h by determining the reason the interrupt was called and correcting the situation that caused INT 70h. INT 70h ticks about 1024 times per second.

INT 71h IRQ9

Input: None

Output: None

Description:

When IRQ9 occurs, the interrupt is routed through the IRQ2 transfer vector (INT 0Ah) by the BIOS and the slave interrupt controller's interrupt is cleared so the interrupt appears to be an IRQ2.

INT 74h PS/2 Mouse Interrupt (IRQ12)

Input: None

Output: None

Description:

INT 74h is the interrupt service routine for BIOS PS/2-type mouse support. The PS/2 mouse sends data to the keyboard controller. The keyboard controller generates IRQ12. Mouse data is transmitted in packets. The BIOS INT 74h collects these packets and stores them in the extended BIOS data area. INT 74h also sets the appropriate flags.

INT 75h Math Coprocessor Interrupt (IRQ13)

Input: None

Output: None

Description:

INT 75h is called when the math coprocessor attached to the system generates an exception and the exception interrupt has been enabled. This interrupt is passed on to the BIOS INT 02h NMI processing routine.

INT 76h AT Hard Disk Drive Interrupt (IRQ14)

Input: None

Output: None

Description:

The hard disk drive controller calls INT 76h when a hard disk drive access is completed.

INT 77h Software Suspend Request (IRQ15)

Input: None

Output: None

Description:

Some American Megatrends Power Management BIOSes process an INT 77h as a suspend request. The BIOS powers down the system when it receives an INT 77h.

Any applications software program can issue an INT 77h to power down the system if the system has one of these Power Management BIOSes.

Chapter 11

Power Management AMIBIOS

Power management is the coordination and manipulation of power-consuming computer system component devices to minimize the system power consumption and maximize battery life. Power management techniques include turning power off to a specific device and slowing or stopping the device's clock.

Power management features are useful in small portable computers, such as laptops, notebook, and handheld models. Since many of these computers run on battery power, a primary design goal is to conserve power use so the system can run as long as possible without being recharged.

System BIOS is a Logical Place to Start

Because it directly controls system hardware, the system BIOS is the logical place to implement power management.

The AMIPMBIOS (American Megatrends Power Management BIOS) does just that. AMIPMBIOS adheres to the APM specification.

APM

AMIPMBIOS adheres to the Advanced Power Management (APM) Specifications developed jointly by Intel and Microsoft. APM is a layered approach to Power Management that defines a cooperative environment where the AMIPMBIOS, operating system, and applications programs work together to reduce power consumption and conserve battery power.

APM takes a system-wide view of power management. The AMIPMBIOS, operating system, and software applications programs all play a role. The operating system can provide precise power management information to the AMIPMBIOS, permitting the BIOS to intelligently conserve power use.

APM Features

- APM can be implemented in any operating system. Microsoft offers APM support in MS-DOS® 5.0 and above and Microsoft Windows® 3.1 and above. APM is compatible with applications that are not aware of APM.

- APM is an open platform-independent specification that can be implemented on any Intel x86-based microprocessor. Additional APM support is provided in the Intel 386SL. Intel and Microsoft made APM an open specification for all AMIPMBIOSes.

- APM is simple for PC users. Microsoft has shipped APM drivers for MS-DOS 5.0 and Windows 3.1. Once configured, the end user does not have to configure or adjust any parameters.

APM Power States

The APM specification defines four power states:

Power State	affects...
Ready	applies to both individual system components and to the system as a whole.
Standby	applies to both individual system components and to the system as a whole.
Suspend	a low power condition that applies to the system as a whole but not to individual components.
Off	applies to both individual system components and to the system as a whole.

Ready State

In Ready, the system or device is fully powered up and ready for use. The system can be active or idle.

Standby State

This is an intermediate system-dependent state that tries to conserve power. The Standby state is entered when the CPU is idle and no device activity occurs for a prespecified length of time. The system does not return to the Ready State until:

- a device raises a hardware interrupt, or
- any controlled device is accessed.

All data and operational parameters are preserved when the system is in Standby.

cont'd

Suspend State

The Suspend state is the lowest level of power consumption available that still preserves operational data and parameters. This state can be initiated either by AMIPMBIOS or software one layer above the BIOS.

AMIPMBIOS can place the system in Suspended state with no notification if it detects a situation that requires an immediate response, for example, when the battery power becomes critically low.

When the system is in Suspend state, no computation is performed until normal activity is resumed and the system leaves this state. Activity cannot resume unless signaled by an external event, such as a key press, Real Time Clock alarm, and so on.

Off State

The system is powered down and inactive in Off state. Data and operational parameters may or may not be preserved in this state.

State Changes

The system and devices in the system can change from one power state to another by explicit command or automatically, based on APM parameter settings and system activity.

Power capabilities differ from device to device. Some devices may not be able to enter all states. Some devices may have built-in automatic power management features invisible to the system. These devices are outside the scope of this manual.

Power Management

Power management features control the power consumption of many system components. Almost every facet of power consumption is monitored. When the system is idle for an end user-specified period of time, the system automatically enters Power Down Mode. The end user can also power down the system by pressing an externally-mounted *Power Down* switch.

IDLE Mode

In IDLE Mode, the CPU receives a very low clock frequency and all other clocks except the DRAM refresh are stopped. The clock can also be stopped for a static CPU. Pressing the externally-mounted *IDLE* switch brings the system out of IDLE Mode.

Power Management Interrupt

INT 77h is the Software Power Management Interrupt (SPMI) in the AMIPMBIOS. The request to change the state of the machine to Power Down Mode comes to the BIOS via OEM-specified sequence microcode.

Microsoft Windows® Support

AMIPMBIOS automatically supports True Suspend/Resume power management features under Windows 3.0 and 3.1 in Real and Protected Mode.

OS/2, Unix, and Xenix Support

All AMIPMBIOS power management features work under OS/2, Unix, Xenix, DOS, and any other operating environment.

cont'd

Modes

The power management scheme is implemented in different levels. Each level saves more power than the previous level and each level can be accessed directly or incrementally. These levels are:

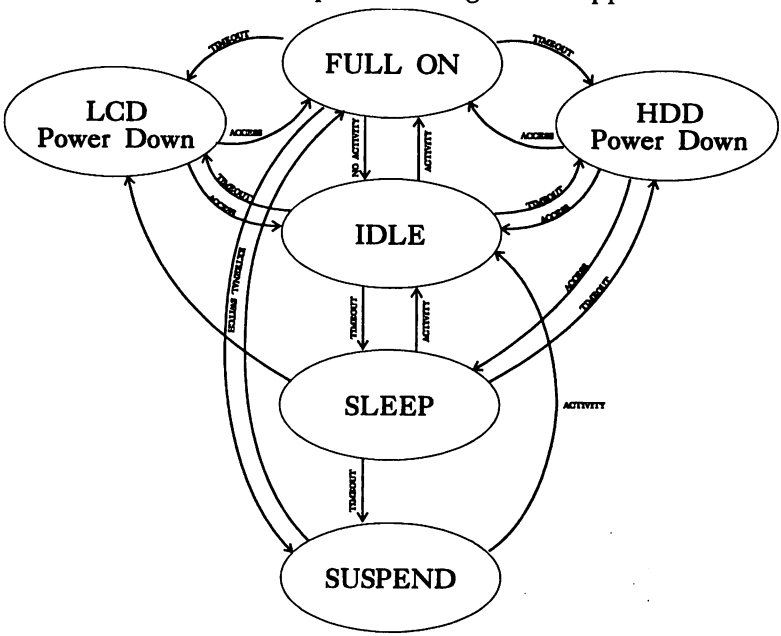
Full On Mode This is full power mode. A system built on a power management chipset initially powers on in this mode. The LCD and hard disk drive are powered off in inactive for a set length of time. The timeout values are set via AMIBIOS Setup and AMIBCP. When AMIPMBIOS determines that the system does not need maximum power, it enters IDLE Mode.

IDLE Mode is entered when the CPU has been idle for a specified length of time. AMIPMBIOS automatically enters this mode. AMIPMBIOS returns to FULL ON Mode when additional power is required.

SLEEP Mode AMIPMBIOS determines if the system performance has dropped to a level such that the system can function efficiently in SLEEP mode. SLEEP Mode can only be entered from IDLE Mode. The length of time that the BIOS waits before entering SLEEP mode is set in AMIBIOS Setup and AMIBCP.

SUSPEND Mode uses the least amount of power necessary for the system to function. SUSPEND mode is entered from SLEEP Mode. The system can go from IDLE mode directly to SUSPEND Mode via timers configured by AMIBIOS Setup and AMIBCP. If an external switch is pressed, the system can go to SUSPEND Mode from any other mode. Pressing the switch again returns the system to FULL ON Mode.

The following graphic illustrates the power management modes and AMIPMBIOS power management support.



AMIBIOS Power Down State Diagram

Chapter 12

EISA Overview

This chapter describes the EISA bus and the interaction between the Hi-Flex EISA AMIBIOS, American Megatrends EISA Configuration Utility (ECU), and EISA systems.

EISA

EISA is an acronym for Extended Industry Standard Architecture. EISA is basically a superset of the Industry Standard Architecture (ISA), based on the original IBM AT specifications. The EISA specifications allow 32-bit memory addressing to be used by the microprocessor, DMA devices, and bus mastering devices. EISA devices can also perform either 16-bit or 32-bit data transfers.

EISA Features

The EISA specification introduced the following features:

- bus mastering, with an arbitration scheme to prioritize bus access and use,
- 32-bit burst mode DMA and three additional DMA transfer modes,
- 16 additional data lines, allowing 32-bit data transfers,
- 8 additional address lines, allowing up to 4 GB of address space,
- complete compatibility with XT and ISA standards, and
- both level-triggered and edge-triggered interrupts.

cont'd

EISA and ISA Differences

One of the most important difference between ISA and EISA is that EISA system configuration is done through software, rather than the hardware switches used by an ISA system. In other words, I/O ports, ROM addresses, IRQ lines, and DMA lines for EISA motherboards and expansion cards are configured using an EISA Configuration Utility (ECU), where an ISA card uses DIP switches and jumpers.

EISA Bus Specifications

EISA computers have 32-bit expansion slots that are fully compatible with 8-bit and 16-bit ISA expansion slots.

EISA expansion slots have 188 pins. The upper 98 pins are exactly the same as the standard ISA pinouts. The lower 90 pins are used for EISA bus signals.

Data can flow on the EISA bus much faster than on the ISA Bus. Not only does EISA provide a wider 32-bit bus, it also provides a maximum 33 MB per second bus transfer rate. An ISA bus can transfer data at a maximum rate of only 8 MB per second.

EISA achieves this higher throughput via high-speed burst transfers, which use only one clock cycle. Normal EISA (ISA-compatible) transfers use two clock cycles.

Attribute	EISA Bus	ISA Bus
Number of Data Lines	32	16
Number of Address Lines	32	24
Bus Clock Rate	about 8 MHz	about 8 MHz
Bus Modes	8-, 16-, and 32-bit	8- and 16-bit
Burst Transfer Rate	33 MB per second	8 MB per second at 0 wait states
DMA characteristics	Supports 8, 16, and 32-bit DMA	Supports 8 and 16-bit DMA
Normal DMA Transfer Rate	5.3 Mbs	1.2 – 1.6 Mbs
Maximum DMA Transfer Rate	33 Mbs	4 Mbs
DMA Cycle Time	0.12 – 1.0 μseconds	1.25 – 1.67 μseconds
Adapter Card Pin Count	188 pins	98 pins
Bus Master	Multiple intelligent 8-, 16-, and 32-bit bus masters	Limited bus mastering
Configuring Adapter Cards	Autoconfiguration through ECU. DIP switch and Jumper setting still available.	Only DIP switch and jumper setting available.

32-bit Memory Addressing

32 memory address lines are available in EISA systems. EISA systems use 80386DX or 80486 microprocessors that allow 32-bit memory addressing. Up to 4 GB of physical RAM can be configured in an EISA system.

An ISA card used in an EISA system can address only up to 16 MB of RAM (because it uses 24-bit ISA memory addressing) but EISA adapter cards can use all available RAM.

Bus Masters

A bus master is a device that takes control of the bus during data transfers supervised by the bus master. The EISA specification permits up to fifteen intelligent bus mastering devices. Although it is possible to add bus mastering to an ISA system, it can only be done on a limited scale and the bus master cannot be intelligent.

With an EISA bus master, the microprocessor does not have to monitor all data transfers. An intelligent EISA bus mastering device uses a dedicated I/O processor and local memory to facilitate and manage data transfers on the EISA bus.

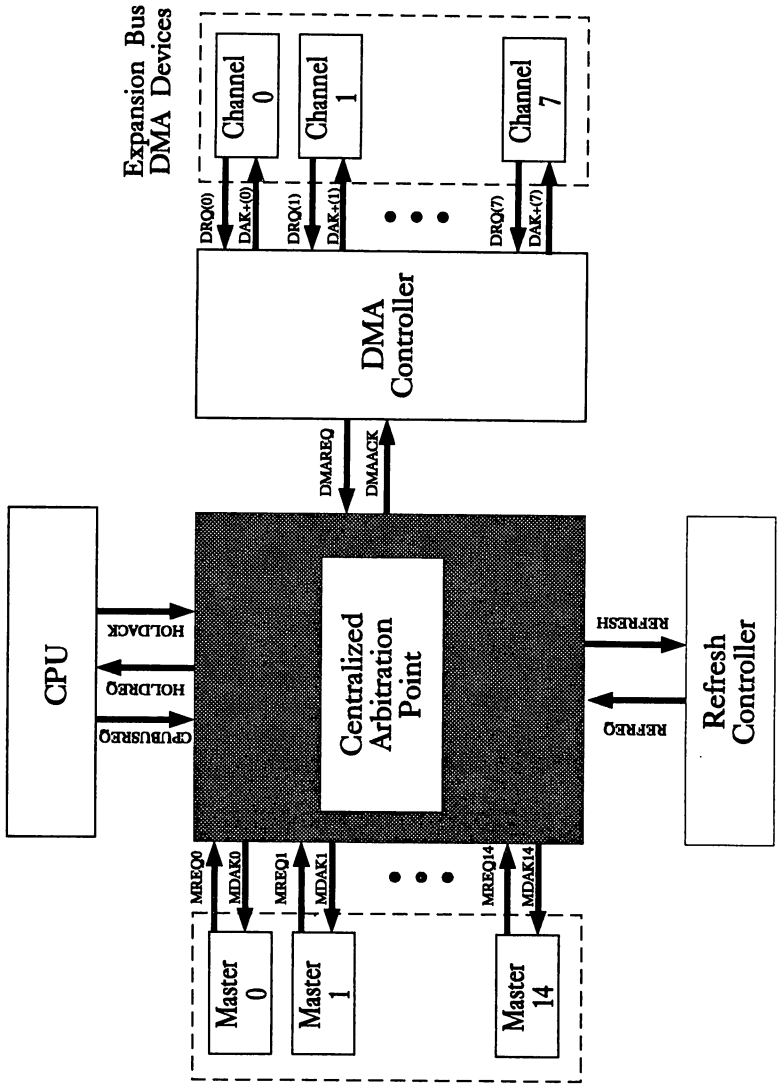
Bus Master Arbitration

Memory refresh, DMA, and each EISA slot has a preassigned priority level. Each priority level has its own line to the central arbitration point. EISA bus arbitration allows the latency for each device on the bus to be determined. The EISA bus master then knows how much system response time to allocate for all devices on the bus. Several I/O processors can run concurrently on an EISA bus.

Arbitration Priority

Memory refresh and DMA have the highest priority. The assignment of arbitration levels 1 – 15 to bus master expansion slots is defined by the EISA motherboard manufacturer.

The following block diagram shows the EISA components involved in arbitration. It identifies 15 bus masters, which is the limit in the EISA Specification. Current EISA chipsets support up to 8 bus masters.



Common Clock Signal

EISA devices can synchronize data transfers to a common clock signal generated and optimized by the EISA motherboard.

Type of Data Transfer	Rate
Standard transfers	two clock cycles
EISA burst transfers	one clock cycle
Bus masters slave devices	1.5 clock cycles

Bus Master Components

An EISA bus master includes a dedicated I/O processor and local memory. The I/O processor is a specialized processor that drives the address, data, and control signals for intelligent peripherals, which become slave devices, during a bus cycle. Bus masters improve system performance by taking on simple tasks that would otherwise be the responsibility of the host processor.

Bus Master Uses

The EISA bus master is designed for I/O peripherals that need optimum performance or advanced memory access functions (such as non-ordered scatter/gather data operations). EISA can use 32-bit devices that contain dedicated I/O processors and require fast bus data transfer rates using a fast burst transfer mode.

EISA DMA

EISA DMA devices have seven channels, just like ISA, but DMA transfer is much faster and supports 8-, 16-, and 32-bit data transfers.

EISA Data Transfer Cycles

Four cycle control sequences for transferring data between the DMA device and memory are available. These cycles are:

EISA Cycle Type	Execution Rate
ISA-compatible	Executes one transfer in eight bus cycles. Two additional bus cycles are added for each wait state. ISA DMA devices can use this cycle to transfer data to or from 8-, 16-, or 32-bit memory.
Type A	Executes one transfer cycle in six bus cycles (longer if the transferred data requires data size translation). Supports 8-, 16-, and 32-bit DMA devices. Data size translation is performed automatically for transfers to mismatched memory.
Type B	Executes one transfer in four bus cycles (longer if the transferred data requires data size translation). Supports 8-, 16-, and 32-bit DMA devices and perform automatic data-size translation for transfers to mismatched memory. Transfer time can be cut in half in some ISA devices by using this type of transfer.
Type C (Burst DMA)	Executes one transfer cycle in one bus cycle. Adds one cycle for each simultaneous transfer and each additional wait state. Supports 8-, 16-, and 32-bit DMA devices and perform automatic data-size translation for transfers to mismatched memory.

Using Type A and B Faster than ISA-Compatible

Most ISA-compatible DMA devices can transfer data about 120% faster by programming the EISA DMA controller to use Type A and B transfers instead of ISA-compatible timing.

DMA Modes

DMA Mode	Description
ISA-compatible	DMA request and acknowledge cycles are performed during each DMA transfer cycle.
Block Transfer	The peripheral device that requires service makes a DMA request. The DMA controller performs a DMA acknowledge cycle and executes DMA transfer cycles continuously until the DMA request is removed or the terminal count is reached. Devices that use ISA-compatible timing should not use this mode.
Demand Transfer	The peripheral device that requires service makes a DMA request. The DMA controller performs a DMA acknowledge cycle. Bus transfers continue until the terminal count register value is reached. Devices that use ISA-compatible timing should not use this mode.
Cascade	A bus mastering device that wants bus ownership asserts a DMA request on the channel. The DMA controller performs a DMA acknowledge cycle. Bus ownership is transferred to the ISA bus-mastering requester. DMA channel 4 uses this mode to cascade DMA channels 0–3 Controller Block to the DMA Channels 4–7 Controller Block. A DMA channel can be programmed in cascade mode for use with external 16-bit bus masters.

Benefits of Arbitration

Arbitration provides increased efficiency and performance. Arbitration manages the time between the DMA device request and the grant events.

Arbitration does not decrease ISA compatibility. Existing hardware and software can take advantage of arbitration without modification, so it may actually improve compatibility with older systems.

EISA DMA Cycle Types

DMA Cycle Type	Size of Transfer	Maximum Transfer Rate (MBs)	Compatibility
ISA-Compatible	8-bit	1.0	All ISA
ISA-Compatible	16-bit	2.0	All ISA
Type A	8-bit	1.3	Mostly ISA
Type A	16-bit	2.6	Mostly ISA
Type A	32-bit	5.3	EISA Only
Type B	8-bit	2.0	Some ISA
Type B	16-bit	4.0	Some ISA
Type B	32-bit	8.0	EISA Only
Burst DMA (Type C)	8-bit	8.2	EISA only
Burst DMA (Type C)	16-bit	16.5	EISA Only
Burst DMA (Type C)	32-bit	33.0	EISA Only

Interrupt Handling Under EISA

The original PC and ISA buses use edge-triggered interrupts. Edge-triggered interrupts are easy to implement but are also susceptible to false triggering and cannot be shared with other interrupts. Edge-triggered interrupts are signalled by the rising edge of the interrupt signal wave form. Other than the line that the signal came from, there is no way for the system to identify an edge-triggered interrupt. Edge-triggered interrupts cannot be shared.

EISA supports edge-triggered interrupts to maintain ISA compatibility, but also provides level-triggered interrupts. Level-triggered interrupts are less susceptible to noise and allow multiple peripherals to share the same interrupt level. Level-triggered interrupts are signaled by a continuous logic-level voltage, permitting interrupt sharing.

EISA System Configuration

EISA permits automatic configuration of system resources and adapter cards and lowers dependence on switches and jumpers.

EISA specifies a product identification mechanism for motherboards and adapter cards. The computer automatically interrogates each adapter card during POST for the product identifier, compares it with the ID stored in EISA Extended CMOS RAM, and configures the adapter card accordingly.

EISA Configuration Files

EISA adapter cards come with a configuration file (CFG file). EISA motherboards come with both a CFG file and an ECU. The ECU configures adapter cards that have EISA .CFG files and store configuration information in EISA CMOS RAM. Using an ECU guarantees that conflicts or contention issues between adapter cards are minimized. The ECU controls the assignment of all necessary system resources.

Configuration Characteristics

EISA configuration consists of the following elements:

- an ECU for motherboard and adapter card configuration,
- CFG files for the motherboard and adapter cards,
- EISA CMOS RAM to store configuration parameters,
- a mechanism to save and restore a backup copy of the configuration parameters,
- BIOS routines to read from and write to CMOS RAM, and
- automatic detection and initialization of adapter cards by the BIOS during BIOS Power-On Self Test (POST).

Adapter Cards and EISA Slot Numbers

Each adapter card (EISA or ISA) is installed in an expansion slot on the EISA motherboard. The slots are numbered from 1 to 15. The EISA motherboard is always slot 0.

Each EISA expansion slot has a unique I/O address space of 1024 bytes (1 KB). The BIOS uses these registers and the information written to EISA CMOS RAM to initialize the EISA adapter cards during POST.

If the slot is occupied by an ISA adapter card, the I/O space is limited to 00100h – 003FFh. The ECU can display the proper switch and jumper settings for the ISA adapter card or device if a CFG file is provided with the ISA adapter card or device.

American Megatrends EISA Configuration Utility

The function of the ECU is to read and write the system configuration and adapter card parameters such that a conflict-free environment is established.

CFG Files

The ECU reads the CFG files that are provided by the manufacturer of every EISA adapter card. The CFG files contain the product ID, the product's system resource requirements, and the product's initialization information.

cont'd

Configuration Data Stored in EISA CMOS RAM

Initialization information is read by the ECU and stored in EISA CMOS RAM. A backup copy of the CMOS RAM configuration data is also stored on disk. The BIOS reads CMOS RAM and executes the initialization instructions during EISA BIOS POST when the system is rebooted. Initialization usually consists of reading and writing to I/O ports assigned to the device.

EISA System Resources

EISA system resources include:

- DMA channels,
- memory,
- interrupt request lines (IRQs), and
- I/O ports.

The ECU verifies that the resources requested by the slot device are not already assigned to another device and then allocates them. The allocation information is stored in EISA CMOS RAM and is accessed by the BIOS during POST.

EISA Configuration Overlay Files

Manufacturers may not be able to perform all initializations in the framework of a CFG file. Features and resources may be specific to the adapter card and may not be configurable by the ECU. For these situations, the EISA specification permits CFG file extensions, or overlay files.

EISA Product ID

I/O port addresses 0zC80 – 0zC83h (z = the slot number) store the EISA four-byte compressed product ID number. The I/O port information differs for motherboards and adapter cards.

For an EISA motherboard

I/O Port	Description
0C80h	Bit 7 Reserved. Should be 0. Bits 6–2 First letter of the manufacturer code. Bits 1–0 First two bits of the second letter of the manufacturer code.
0C81h	Bits 7–5 Remaining bits of the second letter of the manufacturer code. Bits 4–0 Third letter of the manufacturer code.
0C82h	Bits 7–0 Manufacturer's product number.
0C83h	Bits 7–0 Product revision number.

For an EISA Adapter Card

I/O Port	Description
zC80h	Bit 7 Reserved. Should be 0. Bits 6–2 First letter of the manufacturer code. Bits 1–0 First two bits of the second letter of the manufacturer code.
zC81h	Bits 7–5 Remaining bits of the second letter of the manufacturer code. Bits 4–0 Third letter of the manufacturer code.
zC82h	Bits 7–4 Second hex digit of the product number. Bits 3–0 First hex digit of the product number.
zC83h	Bits 7–4 Product revision level. Bits 3–0 Third hex digit of the product number.

CFG Filenames

CFG filenames must consist of an exclamation point, the product ID, and the DOS file extension .CFG.

File Naming Example

!AMI16B4.CFG is the sample CFG file. The following table identifies the components of this CFG file name.

Code	Description
!	Identifies a CFG file.
AMI	Manufacturer ID.
16	Product Number.
B4	Product Revision Level.

Duplicate File Names

The ECU renames CFG files when it finds duplicate CFG filenames. The ECU changes the exclamation point in the filename to the number of the duplicate.

For example, if the ECU finds multiple CFG files for AMI16B4, the first CFG file is named !AMI16B4.CFG, the next 1AMI16B4.CFG, the next 2AMI16B4.CFG, and so on.

EISA System AMIBIOS

The EISA System BIOS works with the ECU to:

- EISA BIOS POST routines use the information stored in EISA CMOS RAM to initialize the system.
- The EISA BIOS provides software routines to read and write this information to and from CMOS RAM. These routines can be called using a software INT instruction.

EISA BIOS Interrupts

There are two BIOS INT 15h Function D8h routines used by the ECU to manipulate the information in CMOS RAM, briefly described below. Detailed explanations of these routines begin on page 220.

Clear EISA CMOS RAM

Call INT 15h with AH = D8h and AL = 02h (16-bit addressing) or AL = 82h (32-bit addressing).

Write EISA CMOS RAM

Call INT 15h with AH = D8h and AL = 03h (16-bit addressing) or AL = 83h (32-bit addressing).

Reading EISA CMOS RAM

The BIOS POST routines and other software drivers use:

- INT 15h Function D8h subfunction 00h/80h *Read Slot Configuration Information,* and
- INT 15h Function D8h subfunction 01h/81h *Read Function Configuration Information*

to read CMOS RAM data. Detailed explanations of these routines begin on page 220.

Chapter 13

Keyboard Controller BIOS

American Megatrends manufactures keyboard controller BIOSes for both the standard 8042 and the Intel 80C51SL keyboard controllers. This chapter documents both the 8042 and 80C51SL Keyboard Controller BIOS. *End users cannot program the keyboard controller without special equipment.*

American Megatrends 8042 Keyboard Controller BIOS

The 8042 Keyboard Controller BIOS provides ISA-compatible and extended keyboard commands and supports AT and PS/2-type keyboards, mouses, and pointing devices. The keyboard controller in most ISA and EISA systems is an Intel 8042. The 8042 is a single-chip microcomputer that supports an Intel 80286, 80386, or 80486 PC keyboard interface.

You need an 8042 assembler to program the keyboard controller. Avocet Systems, Inc. makes an 8042/8742 Assembler. *You cannot reprogram the keyboard controller already in your computer.*

Keyboard Controller BIOS Features

The Keyboard Controller BIOS automatically detects the keyboard type. It operates at 6 – 12 MHz and supports:

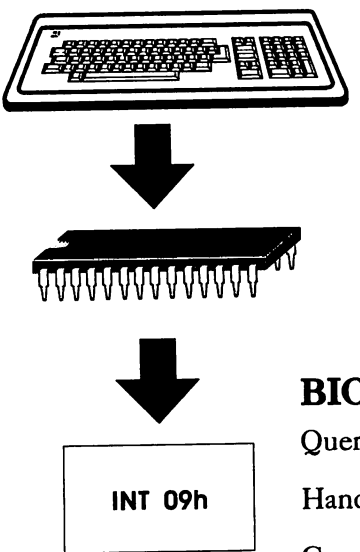
- PS/2-type mouse devices, if hardware support is present,
- 83 or 84-key keyboards,
- 101 or 102-key enhanced keyboards, and
- AT enhanced keyboards and PS/2-type keyboards.

8042 and 8742

The 8042 is an EPROM and the 8742 is an EEPROM. We use the 8042 keyboard controller to refer to both the 8042 and the 8742, since the functionality is the same.

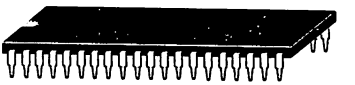
Keyboard Controller Functions

The following figure graphically illustrates the function of the keyboard controller in an ISA or EISA system.



The following figure describes the functions performed by the keyboard controller.

Keyboard Controller



Receives and Translates Serial Data

1. Receives serial data from keyboard.
2. Checks parity.
3. Translates the data to a system scan code, if neccessary.
4. Places received and processed data in the data buffer.
5. Notifies the interrupt processor.

Executes System Commands

1. Executes commands via the controller command buffer.
2. Places the result in the data buffer.
3. Interrupts the system microprocessor.

Reports Errors

Reports errors to the system via status registers.

Transmits/Receives System Data

1. Places it in system buffer.
2. Inserts a parity bit.
3. Sends it to the keyboard in serial format.

It also receives keyboard responses and reports to the system microprocessor.

cont'd

Keyboard Controller Receives Data from the Keyboard

The keyboard sends data in an 11-bit serial format to the keyboard controller. The process is described in the following table.

Step	Performed by	Action
1	the keyboard	The data begins with a start bit (low level), followed by 8 data bits (least significant data bit first), an odd parity bit, and a stop bit (high level).
2	the keyboard	Data sent is synchronized with the keyboard clock.
3	the keyboard controller	On receiving a byte of data from the keyboard, the keyboard controller places the data in its one-byte receive-data buffer and disables the keyboard interface until that data is picked up by the system microprocessor. This avoids data overrun.
4	the microprocessor	Reads the data from the keyboard controller receive-data buffer.
5	the keyboard controller	On parity error, the controller requests that the keyboard resend the data. If the error is repeated, the controller sets the parity error bit in its status register. The keyboard controller sets the time-out bit in the status register if all eleven bits are not received within two milliseconds from the start of the transmission. If either error occurs, FFh is placed in the receive-data buffer.

Possible Errors When Sending Data to the Keyboard

Data is sent to the keyboard in the same serial format as data received from the keyboard.

If...	then...
the time between request to send and start of transmission is greater than 15 milliseconds,	the transmit time-out error bit is set in the status register.
the duration of transmission is greater than 2 milliseconds,	the transmit time-out error bit is set in the status register.
the keyboard must acknowledge every transmission from the controller. If the acknowledgement has a parity error,	the keyboard controller sets both the parity and transmit time-out error status bits.
the acknowledgement does not arrive within 25 milliseconds,	both the receive and transmit time-out error bits are set.

Other Error Results

- FEh is placed in the data buffer if any of these errors occur.
- There are no retries on errors that occur during transmissions to the keyboard.

cont'd

Keyboard Inhibit — 8042

The keyboard can sometimes be inhibited by shorting a jumper. See the manual for the motherboard.

All transmissions from the system to the keyboard still occur when the keyboard is inhibited. The keyboard controller tests all data received from the keyboard. If it is a response to a command sent to the keyboard, it is placed in the data buffer. It is ignored otherwise.

Keyboard Controller and System Interface

The system communicates with the keyboard controller through an input buffer, an output buffer and a status register.

System/Keyboard Communication Method	Reads	Writes
Keyboard status register	64h	N/A
Keyboard output buffer	60h	N/A
Keyboard input buffer	N/A	When the input buffer is written through I/O port 64h, the controller interprets it as a command. If the input buffer is written through I/O port 60h, the data is interpreted either as: ■ a parameter attached to a keyboard command to the controller, or ■ a piece of data to be transmitted to the keyboard.

Keyboard Controller Status Register

Bit	Description
7	Parity Error 0 No parity error. 1 The last byte received from the keyboard had a parity error. The keyboard sends data with odd parity.
6	Receive Time-Out 0 No timeout. 1 A data transmission from the keyboard to the keyboard controller was not properly completed within the predefined time limit.
5	Transmit Time-Out 0 No timeout. 1 A data transmission from the keyboard controller to the keyboard was not properly completed within the predefined time limit.
4	Inhibit Switch This bit reflects the state of the keyboard inhibit switch. This bit is updated whenever the controller writes to the output buffer. 0 Keyboard inhibited 1 Keyboard not inhibited
3	Command/Data Used by the keyboard controller to determine whether the input buffer contains the command or data. 0 The system writes to the input buffer through I/O port 60h. 1 The system writes to the input buffer through I/O port 64h.
2	System Flag The keyboard controller can set this bit to 0 or 1 depending on the command from the system. It is set to 0 after power on reset.
1	Input Buffer Full 0 The keyboard controller input buffer (60h or 64h) is empty. 1 The system has written to the input buffer. It is reset to 0 when the controller reads the input buffer.
0	Output Buffer Full 0 The keyboard controller output buffer has no data. 1 The keyboard controller has written to the output buffer. The keyboard controller returns to 0 when the system reads the output buffer (60h).

Keyboard Controller I/O Ports

The 8042 keyboard controller has two 8-bit I/O ports. One port is an input port and the other an output port. The following table lists the bit definitions for the I/O ports.

Keyboard Input Port Definitions

Bit	Description
7	Keyboard inhibit switch 0 Keyboard inhibited. 1 Keyboard not inhibited.
6	Display type switch 0 Primary display is color graphics adapter. 1 Primary display is monochrome display adapter.
5	Manufacturing diagnostics 0 The system BIOS performs diagnostics on the motherboard in an infinite loop. 1 Any other function.
4	RAM on motherboard 0 Total 256 KB of onboard RAM. 1 512 KB or greater onboard RAM.
3–1	Reserved
0	Diagnostic LED *(for American Megatrends motherboards only)* 0 The motherboard passed the diagnostic tests when diagnostic mode is enabled. The LED blinks in manufacturing diagnostic mode.

Keyboard Output Port Definitions

Bit	Description
7	Keyboard data (output). The data being transferred.
6	Keyboard clock (output). The clock signal used for data transfer.
5	Mouse type 0 PC-type mouse 1 PS/2-type mouse. Generates IRQ12.
4	Output buffer full interrupt to the system from the keyboard 0 Generates an IRQ1. 1 Not full, no IRQ1.
3–2	Reserved
1	Gate address 20 of system processor 0 The system processor address 20 is inhibited on the system bus. Address 20 remains zero for any system processor bus cycle. 1 The system processor address 20 is allowed on the system bus.
0	Reset system microprocessor The software should set this bit to 1 and keep it set for the system microprocessor to work.

Commands to Keyboard Controller

System Commands to the 8042 Keyboard Controller are sent via I/O port address 60h or 64h.

Command	ISA/EISA	MCA (PS/2)
00h-1Fh	Read 8042 RAM to I/O port 60h. The address is specified in bits D5-D0 of the command. These commands to the controller are used exclusively with the AMIBIOS system BIOS.	Read 8042 RAM to I/O Port 60h. The address is specified in bits D5-D0 of the command. These commands to the controller are used exclusively with the AMIBIOS system BIOS.
20h	Read Controller Command Byte. The 8042 places the command byte in the output buffer, making it available in I/O port 60h.	Read Controller Command Byte. The 8042 places the command byte in the output buffer, making it available in I/O port 60h.
20h-3Fh	Read 8042 RAM to I/O port 60h. The address is specified in bits D5-D0 of the command.	Read 8042 RAM to I/O port 60h. The address is specified in bits D5-D0 of the command.
40h-5Fh	Write Controller RAM. The next byte from the system in I/O port 60h is stored in 8042 RAM locations 20h-3Fh. These commands simulate commands 60h-7Fh. These commands to the controller are used exclusively with the AMIBIOS system BIOS.	Write Controller RAM. The next byte from the system in I/O port 60h is stored in 8042 RAM locations 20h-3Fh. These commands simulate commands 60h-7Fh. These commands to the controller are used exclusively with the AMIBIOS system BIOS.
60h	Write Controller Command Byte. The next byte of data from the system in I/O port 60h is used as the Controller Command Byte (CCB).	Write Controller Command Byte. The next byte of data from the system in I/O port 60h is used as the Controller Command Byte (CCB).

Keyboard Controller Commands, Continued

Command	ISA/EISA	MCA (PS/2)
60h	The Controller Command Byte format is: Bit 7 Reserved Bit 6 IBM PC compatibility mode 1 The controller converts the scan code received to the PC-compatible scan code. It also converts the two-byte break sequence from the AT-compatible keyboard to the one-byte PC break code format. Bit 5 Reserved in ISA Systems. Should be 0 for proper operation. in PS/2 Systems: Disable Mouse 1 Disable the mouse interface by driving the clock line low. Data is not received. Bit 4 Disable keyboard. 1 Disables the keyboard interface by driving the *clock* line low. Data is not received. Bit 3 Inhibit override 1 Disables the keyboard inhibit function through the keyboard lock. Bit 2 System Flag The keyboard controller writes the value written in this bit to bit 2 of the status register. Bit 1 Not used in ISA systems. Should be zero. In PS/2 systems: 1 The controller generates an interrupt to the system when it places mouse data in the output buffer. Bit 0 Used in both ISA and PS/2 systems. 1 The controller generates an interrupt to the system when it places keyboard data in the output buffer.	
60h-7Fh	Write Controller RAM. The next byte from the system in I/O port 60h is stored in 8042 RAM locations 20h-3Fh.	Write Controller RAM. The next byte from the system in I/O port 60h is stored in 8042 RAM locations 20h-3Fh.
A0h	Output Copyright Message. A string of standard ASCII characters that ends with a null (0) is placed in I/O port 60h.	Output Copyright Message. A string of standard ASCII characters that ends with a null (0) is placed in I/O port 60h.
A1h	Output Controller Version Number. A single byte of the controller version number in standard ASCII format is placed in I/O port 60h. This command to the controller is used exclusively with the AMIBIOS system BIOS.	Output Controller Version Number. A single byte of the controller version number in standard ASCII format is placed in I/O port 60h. This command to the controller is used exclusively with the AMIBIOS system BIOS.

Keyboard Controller Commands, Continued

Command	ISA/EISA	MCA (PS/2)
A2h	Resets keyboard controller lines P22 and P23 low. These lines can be used for speed switching via the keyboard controller. This command to the controller is used exclusively with the AMIBIOS system BIOS. After executing this command, the keyboard controller sends one garbage byte to the system, indicating completion. The system must clear the garbage byte.	Not valid.
A3h	Sets keyboard controller lines P22 and P23 high. These lines can be used for speed switching via the keyboard controller. This command to the controller is used exclusively with the AMIBIOS system BIOS. After executing this command, the keyboard controller sends one garbage byte to the system, indicating completion. The system must clear the garbage byte.	Not valid.
A4h	Write Clock = Low. Resets an internal flag that indicates that the system clock is Low. 0 indicates that the clock is Low. This command to the controller is used exclusively with the AMIBIOS system BIOS.	Test password installed. Checks to see if a new password has been installed. The result is in I/O port 60h: FAh Password installed. F1h Password not installed.
A5h	Write Clock = High. Sets an internal flag to indicate that the clock is High. 1 indicates that the clock is High. This command to the controller is used exclusively with the AMIBIOS system BIOS.	Load Security. Initiates the password load procedure. The data stream following the command is the new password and is terminated by a null (0). The password is stored in scan code format.
A6h	Read Clock. Returns an internal flag to indicate that the clock is Low or High. 1 indicates that the clock is High. This command to the controller is used exclusively with the AMIBIOS system BIOS.	Enable security. Enables keyboard controller security features. Valid only when a new password is installed.

Keyboard Controller Commands, Continued

Command	ISA/EISA	MCA (PS/2)
A7h	Write Cache Bad. Resets an internal flag to indicate that the cache is bad. 0 indicates that the cache is bad. This command to the controller is used exclusively with the AMIBIOS system BIOS.	Disable Auxiliary Device. Disables the clock line of the auxiliary device and sets bit 5 of the CCB. All data transmissions to or from the auxiliary device are blocked by this command.
A8h	Write Cache Good. Sets an internal flag to indicate that the cache is good. 1 indicates that the cache is good. This command to the controller is used exclusively with the AMIBIOS system BIOS.	Enable Auxiliary Device. Enables the clock line to the auxiliary device and clears bit 5 of the CCB (see the Command 60h description).
A9h	Read Cache Bad or Good. Returns an internal flag to indicate that the cache is bad or good. 0 indicates that the cache is bad. This command to the controller is used exclusively with the AMIBIOS system BIOS.	Auxiliary device Interface Test. Checks the clock and data lines of the auxiliary device. The test result is placed in I/O port 60h: 00h Successful 01h Clock line stuck on Low. 02h Clock line stuck on High. 03h Data line stuck on Low. 04h Data line stuck on High.
AAh	Self Test. The keyboard controller runs an internal diagnostics test. 55h is placed in the output buffer if the test is successful. FCh appears in the output buffer if the test is not successful.	Self Test. The keyboard controller runs an internal diagnostics test. 55h is placed in the output buffer if the test is successful. FCh appears in the output buffer if the test is not successful.
ABh	Interface Test. Instructs the controller to test keyboard clock and data lines. The test result placed in the output buffer is: **Result** **Description** 00 No error detected. 01 Keyboard clock line stuck low. 02 Keyboard clock line stuck high. 03 Keyboard data line stuck low. 04 Keyboard data line stuck high.	
ADh	Disable Keyboard. Disables the keyboard clock line and sets Bit 4 in the CCB to 1. Any keyboard command enables the keyboard.	Disable Keyboard. Disables the keyboard clock line and sets Bit 4 in the CCB. Any keyboard command enables the keyboard.
AEh	Enable Keyboard. Enables the keyboard clock line and clears Bit 4 in the CCB (see Command 60h).	Enable Keyboard. Enables the keyboard clock line and clears Bit 4 in the CCB (see Command 60h).

Keyboard Controller Commands, Continued

Command	ISA/EISA	MCA (PS/2)
B0h	Resets the keyboard controller P10 line low. This command to the controller is used exclusively with the AMIBIOS system BIOS. After executing this command, the keyboard controller sends one garbage byte to the system, indicating completion. The system must clear the garbage byte.	Not valid
B1h	Resets the keyboard controller P11 line low. This command to the controller is used exclusively with the AMIBIOS system BIOS. After executing this command, the keyboard controller sends one garbage byte to the system, indicating completion. The system must clear the garbage byte.	Not valid
B2h	Resets the keyboard controller P12 line low. This command to the controller is used exclusively with the AMIBIOS system BIOS. After executing this command, the keyboard controller sends one garbage byte to the system, indicating completion. The system must clear the garbage byte.	Resets keyboard controller P12 line low. This command to the controller is used exclusively with the AMIBIOS system BIOS. After executing this command, the keyboard controller sends one garbage byte to the system, indicating completion. The system must clear the garbage byte.
B3h	Resets the keyboard controller P13 line low. This command to the controller is used exclusively with the AMIBIOS system BIOS. After executing this command, the keyboard controller sends one garbage byte to the system, indicating completion. The system must clear the garbage byte.	Resets the keyboard controller P13 line low. This command to the controller is used exclusively with the AMIBIOS system BIOS. After executing this command, the keyboard controller sends one garbage byte to the system, indicating completion. The system must clear the garbage byte.

Keyboard Controller Commands, Continued

Command	ISA/EISA	MCA (PS/2)
B4h	Resets the keyboard controller P22 line low. This command to the controller is used exclusively with the AMIBIOS system BIOS. After executing this command, the keyboard controller sends one garbage byte to the system, indicating completion. The system must clear the garbage byte.	Not valid
B5h	Resets the keyboard controller P23 line low. This command to the controller is used exclusively with the AMIBIOS system BIOS. After executing this command, the keyboard controller sends one garbage byte to the system, indicating completion. The system must clear the garbage byte.	Not valid
B8h	Sets the keyboard controller P10 line high. This command to the controller is used exclusively with the AMIBIOS system BIOS. After executing this command, the keyboard controller sends one garbage byte to the system, indicating completion. The system must clear the garbage byte.	Not valid
B9h	Sets the keyboard controller P11 line high. This command to the controller is used exclusively with the AMIBIOS system BIOS. After executing this command, the keyboard controller sends one garbage byte to the system, indicating completion. The system must clear the garbage byte.	Not valid

Keyboard Controller Commands, Continued

Command	ISA/EISA	MCA (PS/2)
BAh	Sets the keyboard controller P12 line high. This command to the controller is used exclusively with the AMIBIOS system BIOS. After executing this command, the keyboard controller sends one garbage byte to the system, indicating completion. The system must clear the garbage byte.	Sets the keyboard controller P12 line high. This command to the controller is used exclusively with the AMIBIOS system BIOS. After executing this command, the keyboard controller sends one garbage byte to the system, indicating completion. The system must clear the garbage byte.
BBh	Sets the keyboard controller P13 line high. This command to the controller is used exclusively with the AMIBIOS system BIOS. After executing this command, the keyboard controller sends one garbage byte to the system, indicating completion. The system must clear the garbage byte.	Sets the keyboard controller P13 line high. This command to the controller is used exclusively with the AMIBIOS system BIOS. After executing this command, the keyboard controller sends one garbage byte to the system, indicating completion. The system must clear the garbage byte.
BCh	Sets the keyboard controller P22 line high. This command to the controller is used exclusively with the AMIBIOS system BIOS. After executing this command, the keyboard controller sends one garbage byte to the system, indicating completion. The system must clear the garbage byte.	Not valid
BDh	Sets the keyboard controller P23 line high. This command to the controller is used exclusively with the AMIBIOS system BIOS. After executing this command, the keyboard controller sends one garbage byte to the system, indicating completion. The system must clear the garbage byte.	Not valid
C0h	Read Input Port. The keyboard controller reads the input port and places the data in the output buffer.	Read Input Port. The keyboard controller reads the input port and places the data in the output buffer.

Keyboard Controller Commands, Continued

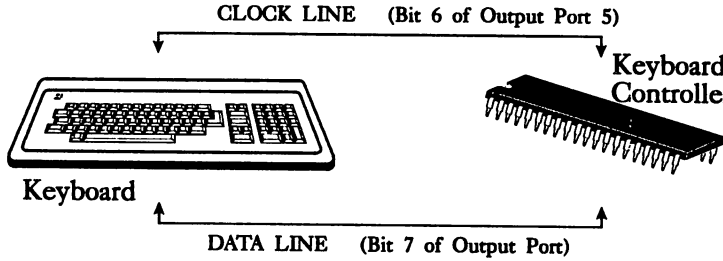
Command	ISA/EISA	MCA (PS/2)
C2h	Not valid	Poll Input Port High. Places bits 7-4 of the Input Port are placed in bits 7-4 of I/O Port 64h.
C3h	Not valid	Poll Input Port Low. Places bits 3-0 of the Input Port are placed in bits 7-4 of I/O Port 64h.
C8h	Unblock keyboard controller lines P22 and P23. The system can make lines P22 and P23 active low or active high via D1h after this command executes. Issue this command before issuing D1h. This command to the controller is used exclusively with the AMIBIOS system BIOS.	Not valid
C9h	Block keyboard controller lines P22 and P23. The system *cannot* make lines P22 and P23 active low or active high via D1h after this command executes. Issue this command before issuing D1h. This command to the controller is used exclusively with the AMIBIOS system BIOS.	Not valid
CAh	Read Mode. This command outputs information about the keyboard controller mode (ISA or PS/2) to I/O Port 60h Bit 0. 0 ISA (AT) interface 1 PS/2 (MCA) interface This command to the controller is used exclusively with the AMIBIOS system BIOS.	Read Mode. This command outputs information about the keyboard controller mode (ISA or PS/2) to I/O Port 60h Bit 0. 0 ISA (AT) interface 1 PS/2 (MCA) interface This command to the controller is used exclusively with the AMIBIOS system BIOS.
CBh	Write Mode. Sets or resets the keyboard controller mode between AT (ISA) and PS/2. Before writing this mode, read the mode byte using command CAh. Then modify only bit 0, leaving the other bits unchanged, and write the mode byte back.	Write Mode. Sets or resets the keyboard controller mode between AT (ISA) and PS/2. Before writing this mode, read the mode byte using command CAh. Then modify only bit 0, leaving the other bits unchanged, and write the mode byte back.
D0h	Read Output Port. Outputs the status of the keyboard controller output port (P2) to I/O Port 60h. The keyboard controller reads the output port and places the data in the output buffer.	Read Output Port. Outputs the status of the keyboard controller output port (P2) to I/O Port 60h. The keyboard controller reads the output port and places the data in the output buffer.

Keyboard Controller Commands, Continued

Command	ISA/EISA	MCA (PS/2)
D1h	Write Output Port. The data byte that follows this command is written to the keyboard controller output port (P2). Writes data via I/O port 60h to the output port. Make sure that output port bit 0 is not written as 0, because a 0 in bit 0 resets the system processor.	Write Output Port. The data byte that follows this command is written to the keyboard controller output port (P2). Writes data via I/O port 60h to the output port. Make sure that output port bit 0 is not written as 0, because a 0 in bit 0 resets the system processor.
D2h	Not valid	Write Keyboard Output Buffer. This command sends the data byte that follows the command in I/O Port 60h straight to the system as it is initiated by the device.
D3h	Not valid	Write Auxiliary Device Output Buffer. The next data byte in I/O Port 60h from the system is made available immediately to the system as if it is initiated by an auxiliary device.
D4h	Not valid	Write Auxiliary Device. The next data byte to I/O Port 60h is transmitted to an Auxiliary device.
E0h	Read Test Inputs. This command makes the status of the Test inputs T0 and T1 available to the system at I/O Port 60h. The T0 status is in Bit 0 (0 is enabled). The T1 status is in Bit 1 (0 is enabled).	Read Test Inputs. This command makes the status of the Test inputs T0 and T1 available to the system at I/O Port 60h. The T0 status is in Bit 0 (0 is enabled). The T1 status is in Bit 1 (0 is enabled).
F0h-FFh	Pulse Output Port. Bits 3-0 of the output port of the keyboard controller may be pulsed low for approximately 6 µseconds. Bits 3-0 of this command specify the output port bits to be pulsed. The corresponding bits in the command indicate the bits to be pulsed. 0 Bit should be pulsed. 1 Bit should not be modified. Note that bit 0 of the output port is connected to the reset of the system processor, so the processor can be reset by pulsing this bit.	Pulse Output Port. Bits 1-0 of the output port of the keyboard controller may be pulsed low for approximately 6 µseconds. Bits 1-0 of this command specify the output port bits to be pulsed. The corresponding bits in the command indicate the bits to be pulsed. 0 Bit should be pulsed. 1 Bit should not be modified. Note that bit 0 of the output port is connected to the reset of the system processor, so the processor can be reset by pulsing this bit.

Keyboard Controller/Keyboard Interface

The keyboard controller communicates with the keyboard over a clock line (bit 6 of output port 5) and a data line (bit 7 of the output port). The following graphic illustrates this process.



The keyboard controller reads the data line through test input T1 and the clock line through test input T0.

The keyboard supplies the clock for all data transmission to and from the keyboard.

Data is made available after the rising edge of the clock and is sampled on the falling edge.

cont'd

Keyboard/Keyboard Controller Communication Protocol

The hardware protocol for communication between the keyboard and the keyboard controller is as follows:

Step	Performed by	Action
1	the keyboard	When the keyboard wants to send data, it first checks the clock line for a high level. *The keyboard controller can prevent the keyboard from sending data by driving the clock line low through bit 6 of the output port.*
2	the keyboard	Sends the data if the clock and data lines are high (enabled). Otherwise it stores data in its own buffer.
3	the keyboard	Checks the state of the clock line at an interval of 60 microseconds, to sense whether the keyboard controller intends to send data.
4	the keyboard controller	When the keyboard controller wants to send data, it forces the clock line low for more than 60 microseconds and then releases the clock line with the data line low.
5	the keyboard	The low data line is accepted by the keyboard as a start bit (request to send) and the keyboard starts clocking the data in. After the tenth bit, the keyboard forces the data line low for one clock period (the stop bit). This action informs the keyboard controller that the keyboard has received the data.

80C51SL Keyboard Controller BIOS

The American Megatrends 80C51SL Keyboard Controller BIOS is an advanced keyboard controller BIOS with many features not found in the 8042 BIOS.

This section describes the Intel 80C51SL and the 80C51SL keyboard controller BIOS and provides a sample application. The information in this chapter should provide sufficient technical data for an OEM to design a system with the 80C51SL keyboard controller BIOS.

80C51SL Features

The Intel 80C51SL Keyboard Controller includes:

- standard 8042 ISA-Host interface functionality,
- keyboard scanning,
- direct external keyboard support,
- direct PS/2-type mouse support, and
- power management (both 80C51SL-specific and system-wide).

Flash Memory

The Intel 80C51SL Keyboard Controller supports external flash memory, which can be used for 80C51SL program memory and all standard AT keyboard commands.

It supports many keyboards and can reference multiple key scan matrix definitions in the keyboard firmware in 28F256A flash memory.

cont'd

Automatically Detects AT and PS/2 Keyboards

AT and PS/2 keyboards are automatically detected by the Intel 80C51SL.

Supports Many Keyboards

The Intel 80C51SL supports 83 or 84-key and 101 or 102-key keyboards, and both internal and external keyboards.

Mouse Support

The 80C51SL Keyboard Controller supports PS/2-type mouse pointing devices through the external PS/2 mouse connector. It also supports the Keytronic Keymouse, a part of the Keytronic U046001 keyboard.

80C51SL Keyboard Controller Operating Speeds

The 80C51SL operates at speeds between 6 and 12 MHz.

80C51SL BIOS Features

Suspend/Resume Support

The 80C51SL Keyboard Controller BIOS supports a user-definable keychord (defined via AMIBIOS Setup) to change to suspend mode and then resume mode.

Power Management Features

The 80C51SL Keyboard Controller BIOS returns to Idle Mode between keystrokes and enters Power Down Mode during Suspend when running on battery power.

This BIOS supports a software-configurable Power Down Mode during Suspend when running on battery power. The Power Down Mode switches to the lowest available power saving state. The crystal is stopped and only RAM data is saved.

Battery Monitoring Support

The 80C51SL Keyboard Controller BIOS supports battery temperature and voltage monitoring. It also monitors battery charging to prevent overcharge.

External Memory Support

The 80C51SL Keyboard Controller BIOS supports external 32 KB 28F256A flash memory.

Supported Keys — Internal and External Keyboards

The 80C51SL BIOS supports any set of keys for any keyboard without modification.

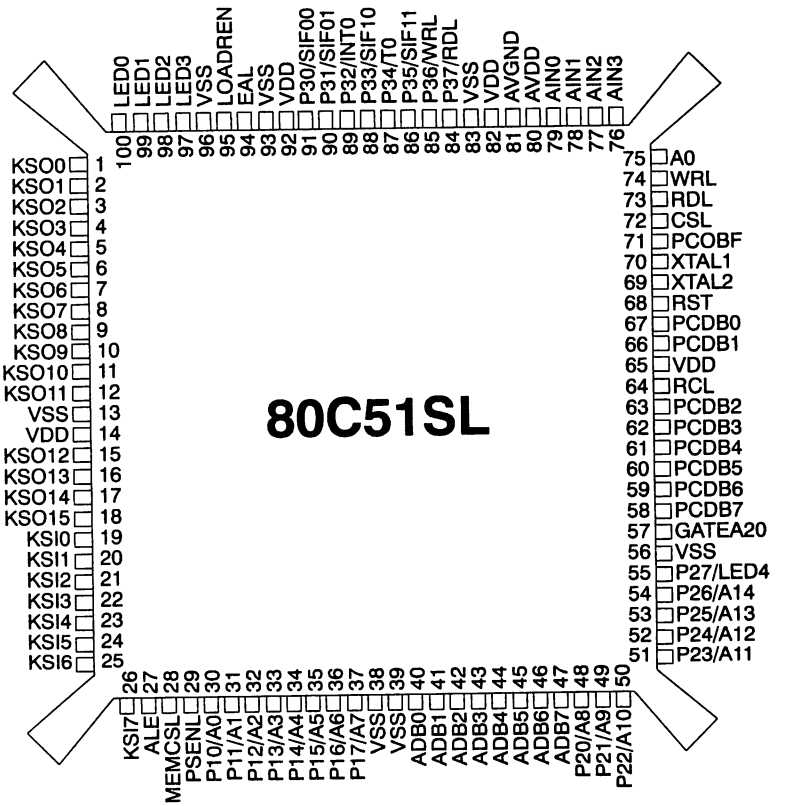
The 80C51SL BIOS supports any key scan matrix that can be downloaded to the 80C51SL RAM, with multiple key scan matrix definitions referencable in the keyboard firmware resident in 28F256A flash memory.

80C51SL Keyboard Controller Description

The Intel 80C51SL keyboard controller is a 100-pin PQFP (plastic quad flat pack) superset of the 80C51 architecture.

The 80C51SL Keyboard Controller BIOS is in the masked ROM in a standard production version of the 80C51SL.

The following specifications are intended for use with the -BG silicon version of the 80C51SL. The basic layout and pin configuration of the 80C51SL keyboard controller is shown below:



80C51SL Pin Configuration

Interfaces

American Megatrends has addressed three interfaces to be used with the keyboard controller:

- keyboard,
- host, and
- OEM-defined interfaces.

The external memory interface is not available at this time.

80C51SL Functions

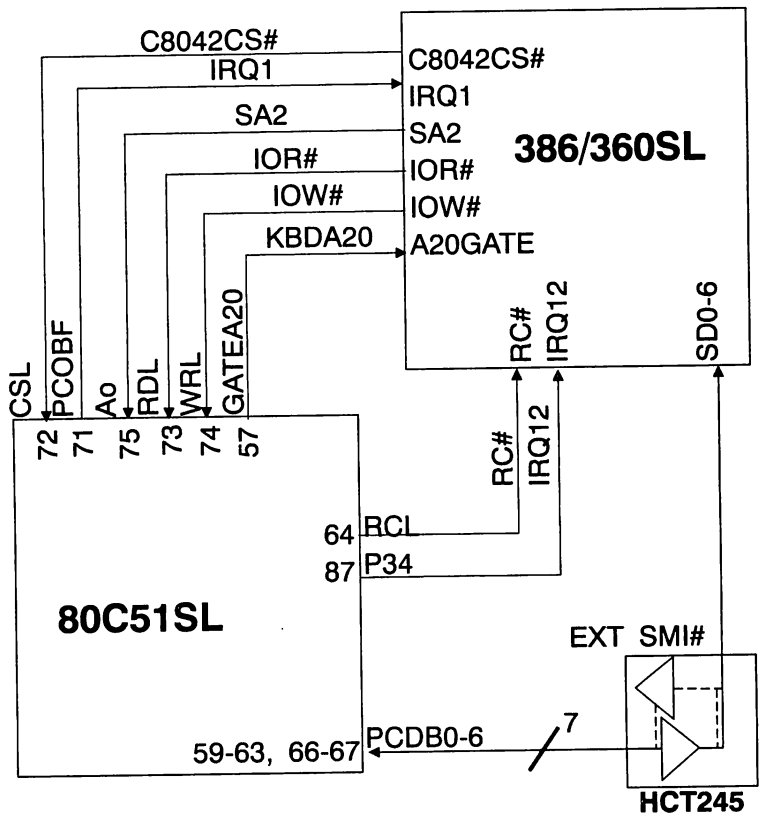
Through these interfaces, the 80C51SL Keyboard Controller BIOS supports the following functions described in the sample application on the following page.

- Scanned keyboard support,
- External keyboard support,
- PS/2-style mouse support,
- LED control support,
- Fast Gate A20 logic,
- Fast CPU Reset logic, and
- Power management logic.

cont'd

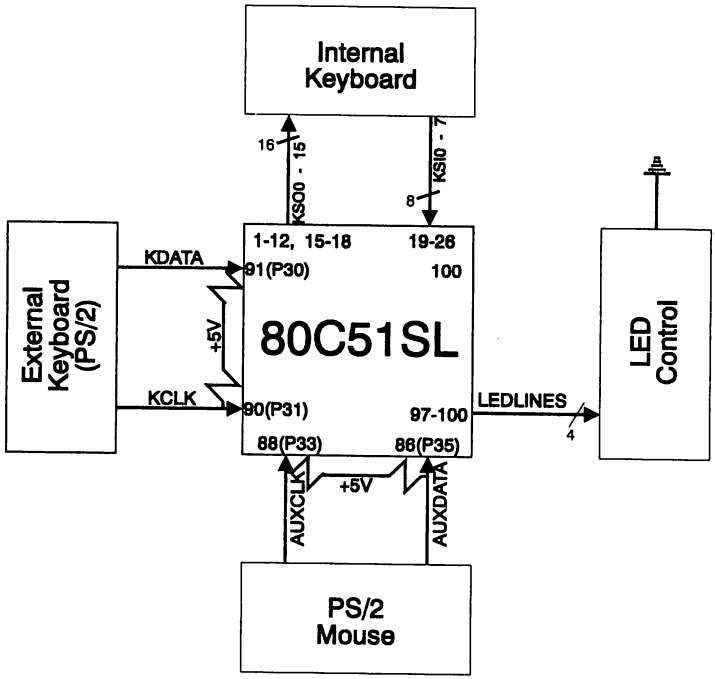
Sample 80C51SL Application

The following block diagram is for a design example of an American Megatrends application of the 80C51SL.



80C51SL Keyboard Interface

The following diagram shows the pins used for interfacing keyboards and input devices to the 80C51SL:



cont'd

80C51SL Internal Keyboard

The keyboard lines used for the keyboard scan logic are:

- scan-out,
- slew rate-controlled,
- open-drain lines (KSO0 through KSO15), and
- Schmitt trigger sense lines (KSI0 through KSI7) with internal pull-up resistors of 5 – 20 KOhms

The scanout lines are controlled by the four low-order bits of port 0 (also the direct connection to the trigger sense lines).

Supported Keyboard Scan Matrixes

The 80C51SL directly supports a 16x8 scan matrix. However, multiple scan lines can be connected together for a larger matrix. KSO1 scans bits 1 and 12 and KSO14 scans bits 15 and 18. Bits 9 and 12 of the input matrix drive KSI1.

80C51SL External Keyboard and PS/2 Mouse

An external keyboard and mouse can be used with the 80C51SL via two industry-standard two-wire, bidirectional TTL interfaces. The four signal pins to these interfaces are high-drive, open-drain output, bidirectional port pins:

- P30 (SIF00) keyboard clock,
- P31 (SIF01) keyboard data,
- P33 (SIF10) mouse clock, and
- P35 (SIF11) mouse data.

In addition, the same external connector can support both mouse and keyboard, which frees two I/O port pins.

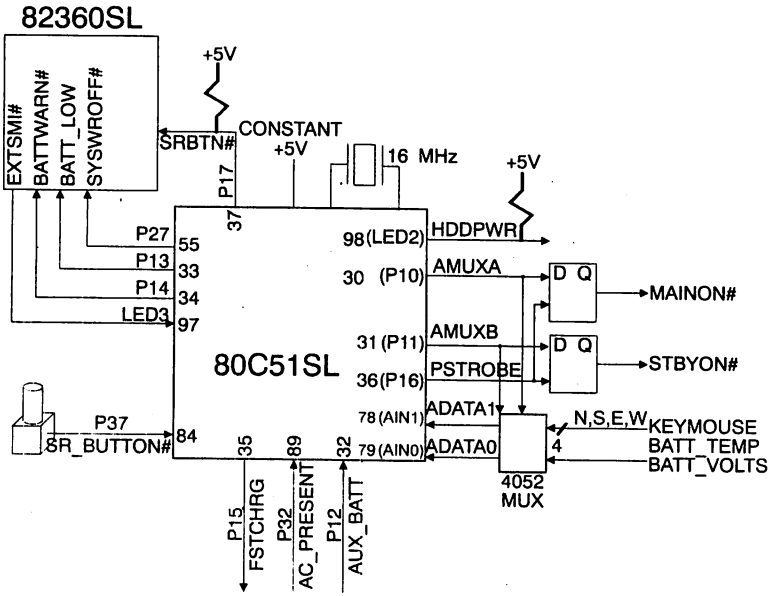
80C51SL LED Control

Only LED1 and LED0 transmit serial LED control instead of directly driving the LEDs. This permits direct interfacing to a predefined video card LED interface. The remaining LED drivers are used in the OEM-defined interface.

80C51SL Host Interface

Host interfacing is performed in the same manner as is done in the 8042 Keyboard Controller BIOS.

The 80C51SL offers an 8042-style interface for compatibility. A diagram of this interfacing scheme is shown below.



cont'd

American Megatrends Interface for 80C51SL Keyboard Controller BIOS

The diagram on the previous page is an American Megatrends implementation of a design in the OEM-defined interface.

Through this interface,:

- suspend/resume,
- power,
- battery monitoring, and
- battery charging

are controlled.

The power requirement is +5 Volts. The clock must be driven with a 16 MHz crystal, which can be done through a small auxiliary power supply or linear regulator, driven by the main system battery.

Interface Design Features

The following design features are discussed in this section:

- Suspend/Resume,
- Power Management, and
- Battery Monitoring and Charging.

Suspend/Resume

Direct suspend/resume can be controlled by an external button via P37. Keystroke control is achieved through P17.

80C51SL BIOS Interface Design Features, cont'd

Power Management

P10 (AMUXA), P11 (AMUXB), and P16 (PSTROBE) control system power.

The two power status signals are MAINON# (normal operation) and STBYON# (Suspend Mode). If both signals are inactive, the system automatically goes to Off mode.

This status, called SYSPWROFF#, is checked through P27 before MAINON# or STBYON# are switched.

LED3 can be defined as a hot key to display the power management menu.

Power Savings Modes

The 80C51SL power saving modes are Idle Down and Power Down. The following table describes these modes.

Idle Down Mode	Power Down Mode
Invoked between keystrokes.	Occurs during system suspend mode.
In idle-down mode, the internal clock signal to the 80C51SL is disabled while still maintaining register data.	All internal clocks are disabled in power-down mode. Only configurable registers, host registers, RAM, GATEA20, and RCL are maintained.
Can be configured to exit during a host write, key press, or an active external interrupt detection. MEMCSL is inactive in either mode unless it is configured as general purpose I/O.	Can be configured to exit during a host write, key press, or an active external interrupt detection. MEMCSL is inactive in either mode unless it is configured as general purpose I/O.

cont'd

80C51SL BIOS Interface Design Features, cont'd

Hard Disk Power Management

Hard disk power management is achieved through LED2. Since control is via the 80C51SL, power management is used for many hard disk drive types without modifying power management routines. A write to bit 6 of internal Port 00h from an Intel 386SL controls the pin for LED2.

Battery Monitoring and Charging

The Analog-to-Digital (A/D) channel AIN0 monitors the battery. AIN1 monitors the mouse. AIN0 or ADATA0 monitors battery temperature and voltage using a 4052 analog MUX. AIN1 (ADATA1) monitors the north, south, east, and west signals from an external mouse.

The two other A/D channels (AIN2 and AIN3) are unused.

The 80C51SL controls both battery monitoring and charging. When monitoring battery usage under normal conditions, the 80C51SL can monitor battery voltage and control the BATTWARN# and BATT_LOW# signal from an 8260SL.

BATTDEAD# is not needed since the 80C51SL does not resume if the battery is unusable. An additional backup battery can be detected through P12. An AC source is detected through P32. When charging from an AC source, the 80C51SL monitors BATT_VOLTS and optionally BATT_TEMP to control fast or trickle charging, depending on system parameters.

Appendix A

AMIBIOS Error Messages and Beep Codes

Errors can occur during POST (Power On Self Test), which is performed every time the system is powered on. Fatal errors (see below) are communicated through a series of audible beeps. All errors except Beep Code 8 are fatal errors. Fatal errors do not allow the system to continue the boot process.

Most displayed errors (see page 394) allow the system to continue the boot process.

Beeps	Error message	Description
1	Refresh Failure	The memory refresh circuitry on the motherboard is faulty.
2	Parity Error	Parity error in the first 64 KB of memory.
3	Base 64 KB Memory Failure	Memory failure in first 64 KB.
4	Timer Not Operational	Memory failure in the first 64 KB of memory, or Timer 1 on the motherboard is not functioning.
5	Processor error	The CPU (Central Processing Unit) on the motherboard generated an error.
6	8042 – Gate A20 Failure	The keyboard controller (8042) may be bad. The BIOS cannot switch to protected mode.
7	Processor Exception Interrupt Error	The CPU generated an exception interrupt.
8	Display Memory Read/Write Error	The system video adapter is either missing or its memory is faulty. This is not a fatal error.
9	ROM Checksum Error	The ROM checksum value does not match the value encoded in the BIOS.
10	CMOS Shutdown Register Read/Write Error	The shutdown register for CMOS RAM failed.
11	Cache Error/ External Cache Bad	The external cache is faulty.

Displayed Fatal Errors

The system halts after the following screen messages and cannot usually be rebooted until a physical change is made in the system.

- CMOS inoperational
- 8042-Gate A20 error
- DMA error
- DMA #1 error
- DMA #2 error
- FDD Controller failure
- HDD Controller failure
- INTR #1 error
- INTR #2 error
- On Board parity error
- On Board Parity error

Format

```
ERROR Message Line 1
ERROR Message Line 2
Press <F1> to RESUME
```

The <F1> prompt message is not displayed if *Wait for <F1> If Any Error* in Advanced CMOS Setup has been Disabled. For most displayed error messages, there is only one message. If a second message appears, it is

```
RUN SETUP
```

If this message occurs, press <F1> to run AMIBIOS Setup.

Error Message	Explanation
8042 Gate – A20 Error	Gate A20 on the keyboard controller (8042) is not working. Replace the 8042.
Address Line Short!	Error in the address decoding circuitry on the motherboard.
C: Drive Error	Hard disk drive C: does not respond. Run the Hard Disk Utility to correct this problem. Also, check the C: hard disk type in Standard CMOS Setup to make sure that the hard disk drive type is correct.
C: Drive Failure	Hard disk drive C: does not respond. Replace the hard disk drive.
Cache Memory Bad, Do Not Enable Cache!	Cache memory is defective. Replace it.

Error Message	Explanation
CH-2 Timer Error	Most AT systems include two timers. There is an error in timer 2.
CMOS Battery State Low	CMOS RAM is powered by a battery. The battery power is low. Replace the battery.
CMOS Checksum Failure	After CMOS RAM values are saved, a checksum value is generated for error checking. The previous value is different from the current value. Run AMIBIOS Setup.
CMOS System Options Not Set	The values stored in CMOS RAM are either corrupt or nonexistent. Run AMIBIOS Setup.
CMOS Display Type Mismatch	The video type in CMOS RAM does not match the type detected by the BIOS. Run AMIBIOS Setup.
CMOS Memory Size Mismatch	The amount of memory on the motherboard is different than the amount in CMOS RAM. Run AMIBIOS Setup.
CMOS Time and Date Not Set	Run Standard CMOS Setup to set the date and time in CMOS RAM.
D: Drive Error	Hard disk drive D: does not respond. Run the Hard Disk Utility. Also check the D: hard disk type in Standard CMOS Setup to make sure that the hard disk drive type is correct.
D: drive failure	Hard disk drive D: does not respond. Replace the hard disk.
Diskette Boot Failure	The boot disk in floppy drive A: is corrupt. It cannot be used to boot the system. Use another boot disk and follow the screen instructions.
Display Switch Not Proper	Some systems require a video switch on the motherboard be set to either color or monochrome. Turn the system off, set the switch, then power on.
DMA Error	Error in the DMA controller.
DMA #1 Error	Error in the first DMA channel.
DMA #2 Error	Error in the second DMA channel.
FDD Controller Failure	The BIOS cannot communicate with the floppy disk drive controller. Check all appropriate connections after the system is powered down.
HDD Controller Failure	The BIOS cannot communicate with the hard disk drive controller. Check all appropriate connections after the system is powered down.
INTR #1 Error	Interrupt channel 1 failed POST.
INTR #2 Error	Interrupt channel 2 failed POST.
Invalid Boot Diskette	The BIOS can read the disk in floppy drive A:, but cannot boot the system. Use another boot disk.
Keyboard Is Locked...Unlock It	The keyboard lock on the system is engaged. The system must be unlocked to continue.
Keyboard Error	There is a timing problem with the keyboard. Set the *Keyboard* option in Standard CMOS Setup to *Not Installed* to skip the keyboard POST routines.
KB/Interface Error	There is an error in the keyboard connector.

Error Message	Explanation
No ROM BASIC	Cannot find a bootable sector on either disk drive A: or hard disk drive C:. The BIOS calls INT 18h which generates this message. Use a bootable disk.
Off Board Parity Error	Parity error in memory installed in an expansion slot. The format is: `OFF BOARD PARITY ERROR ADDR (HEX) = (XXXX)` *XXXX* is the hex address where the error occurred. Run AMIDiag to find and correct memory problems.
On Board Parity Error	Parity error in motherboard memory. The format is: `ON BOARD PARITY ERROR ADDR (HEX) = (XXXX)` *XXXX* is the hex address where the error occurred. Run AMIDiag to find and correct memory problems.
Parity Error ????	Parity error in system memory at an unknown address. Run AMIDiag to find and correct memory problems.

EISA Error Messages

Error Message	Explanation
EISA CMOS Checksum Failure	The Checksum for EISA CMOS is incorrect. The battery for EISA CMOS RAM may need to be replaced.
EISA CMOS inoperational	A Read or Write error occurred in extended CMOS RAM. The battery may need to be replaced.
Expansion Board not ready at Slot *X, Y, Z*	The BIOS cannot find the adapter card in Slot *X*, *Y*, or *Z*. Make sure the card is in the correct slot and is seated.
Fail-Safe Timer NMI Inoperational	Devices that depend on the fail-safe NMI timer cannot operate correctly.
ID information mismatch for Slot *X, Y, Z*	The ID of the EISA adapter card in Slot *X*, *Y*, or *Z* does not match the ID in EISA CMOS RAM. Run the ECU.
Invalid Configuration Information for Slot *X, Y, Z*	Configuration data for EISA adapter cards *X*, *Y*, or *Z* is not correct. The card cannot be configured. Run the ECU.
Software Port NMI Inoperational	The software port NMI is not working. Operation can continue, but when NMIs occur, the system may go down.

ISA NMI Messages

ISA NMI Message	Explanation
Memory Parity Error at *xxxxx*	Memory failed. If the memory location can be determined, it is displayed as *xxxxx*. If not, the message is *Memory Parity Error ????*.
I/O Card Parity Error at *xxxxx*	An expansion card failed. If the address can be determined, it is displayed as *xxxxx*. If not, the message is *I/O Card Parity Error ????*.
DMA Bus Time-out	A device has driven the bus signal for more than 7.8 microseconds.

EISA NMI Error Messages

The EISA AMIBIOS can generate additional EISA-specific NMI messages. They are:

EISA NMI Message	Explanation
BUS Timeout NMI at Slot *n*	There was a Bus Timeout NMI at Slot *n*.
(E)nable (D)isable Expansion Board?	Type E to enable the adapter card that had an NMI or D to disable it.
Expansion Board Disabled at Slot *n*	The EISA adapter card in Slot *n* has been disabled.
Expansion Board NMI at Slot *n*	An adapter card NMI was generated from Slot *n*.
Fail-Safe Timer NMI	A fail-safe timer NMI has been generated.
Software Port NMI	A software port NMI has been generated.

Appendix B

Upgrading the BIOS

Although American Megatrends does not upgrade BIOSes, some of our distributors do. Call American Megatrends Sales at 404-263-8181 for information about upgrade BIOSes.

Ordering an Upgrade System BIOS

Please have the following information available when ordering an upgrade BIOS:

- manufacturer and model number of your computer
- processor type,
- processor speed,
- date of BIOS,
- BIOS Identification string, and
- RAM configuration and speed.

How To Find the Processor Type and Speed

To find the processor type and speed:

- the AMIBIOS System Configuration screen displays the processor type, or
- AMIDiag Version 4.0 or later displays this information.

cont'd

How to Find the BIOS Date

Again, AMIDiag Version 4.0 or later displays this information. But the quickest method is to read the AMIBIOS System Configuration Screen that appears at system boot.

The BIOS date is 8 bytes located at F:FFF5h in memory.

How to Find the BIOS Identification String

The string is displayed at the bottom of the first AMIBIOS screen. Press <Ins> during system power-on to display any additional reference strings.

Why You Should Update the System BIOS

It's quite possible that a system BIOS upgrade can offer additional support that might be useful to you. A new BIOS might add support for 1.44 MB 3½ inch floppy drives, PS/2-type mouses and keyboards, additional hard disk drives, or other features.

Only system BIOS upgrades are discussed in this book.

See Appendix C for a list of AMIBIOS features for each release of the AMIBIOS since 1986.

The Cost of Upgrading the System BIOS

Discuss the price of a BIOS upgrade with one of the American Megatrends distributors.

The Benefits of Upgrading to a New BIOS

Each revision of the core AMIBIOS adds support for additional peripheral devices, new features, improves the speed of the BIOS code, and fixes bugs.

See the table in Appendix C for a list of the major features of every version of the AMIBIOS to determine more exactly the features you can add to your system if you upgrade to a newer AMIBIOS.

Some of the potential benefits include the ability to:

- configure new higher-density 3.5 inch drives,
- configure new or unusual hard disk drives,
- use serial devices that operate at higher data transfer rates,
- performance improvement,
- use a PS/2-type mouse or keyboard,
- use advanced Setup options. The newer AMIBIOSes provide many more Setup options, allowing you to control more system functions, and
- use the AMIBIOS hard disk utilities that come with the BIOS.

cont'd

Potential Problems of Upgrading the BIOS

Do not try to perform a BIOS upgrade yourself without consulting an American Megatrends dealer or distributor. If you choose the wrong BIOS, it either may not work or may give unpredictable results and may corrupt CMOS RAM to the point that the system can be unusable.

This is not a complete disaster. When CMOS RAM is corrupted, simply remove the battery and leave the system alone for about 30 minutes. Make sure you know the hard disk drive type before you do this.

Then replace the battery and turn the system on. CMOS RAM should be blank, since it has been without power. You are now free to configure the system from scratch.

Enter the date, time, disk configuration information, monitor type and other basic system configuration, store it in CMOS RAM, and reboot.

Installing System BIOS ROM

The ROM BIOS chips are installed in 28-pin DIP sockets. One to four BIOS chips may be present. The following table identifies the BIOS ROMs.

If you have an...	the ROM size is...	the chip type is...	and the number of BIOS chips is...
EISA System	128 KB	27010 or	1 ROM chip
		27512 or	2 ROM chips
		27256	4 ROM chips
ISA (AT-compatible) System	64 KB	27512 or	1 ROM chip
		27256 or	2 ROM chips
		27128	4 ROM chips

Removing the Old BIOS ROMs

Use an IC remover or a screwdriver to gently remove the old BIOS ROM chips. Do not use excessive force to remove the chips. Applying excessive force can damage the motherboard.

Unpacking the New BIOS ROM Chips

Unpack the BIOS ROM chips. See the following table to identify BIOS chip labels. Inspect chips for bent pins. Using pliers, gently straighten any bent pins so that all pins are parallel and straight.

Install each ROM chip so that pin 1 of a ROM is inserted in pin 1 of the corresponding socket, and all other pins on the ROM chip fit in the pin sockets by number. Press each ROM firmly but gently in the socket.

Steps to Upgrade the System BIOS

Step	Action
1	Turn the computer off and remove the cover.
2	Wear an antistatic wristband to properly ground yourself.
3	Unpack the ROM BIOS chips on a grounded antistatic mat.
4	Check the size and number of the existing ROM BIOS chips. AMIBIOS is available in the following sets: One 27512 ROM chip (64 KB) Two 27256 ROM chips (32 KB each) Four 27128 ROM chips (16 KB each). Read the system documentation to verify that the set of AMIBIOS you received is the proper size and number for the motherboard.
5	Remove the old chips. Note which chips are ODD and EVEN (for a two-chip set) or which chips are numbered 0, 1, 2, and 3 (for a four-chip set). The new chips should directly replace the old chips. The EVEN chip from the new set should replace the EVEN chip.
6	Install the chips. Each ROM chip is notched and there is a corresponding notch on the receiving ROM socket. When installing the chips, make sure that they are facing so that the notch on the chip matches the notch on the socket.
7	Replace the cover.

Appendix C

AMIBIOS History

This appendix includes a history of AMIBIOS releases. The major upgrades, features, and newly-supported devices are listed for each BIOS release. American Megatrends, Inc. has been providing AT-compatible system BIOSes since 1986. The AMIBIOS has always incorporated all standard IBM AT BIOS features, and has always been noted for its performance. This appendix lists many of the additional features added by American Megatrends since 1986.

BIOS Identification String

The BIOS Identification string, which appears in the lower left corner when the system boots, provides information about AMIBIOS. See Appendix D for additional information about the AMIBIOS Identification String.

If your Motherboard has Caching

In general, older AMIBIOSes (AMI BIOS and AMI BIOS Plus) for non-American Megatrends motherboards with cache memory are customized. You must contact the motherboard manufacturer to update the BIOS. American Megatrends motherboards usually have a BIOS ID that begins with DAMI, DAMX, or EDAMI.

AMI 286 and 386 BIOS - January 1987

- 1.44 MB 3½" floppy disk drive support.
- Support for IDE hard disk drives.
- User-defined hard disk drive type 47.
- INT 15h Function 4Fh Keyboard Intercept Support.
- INT 15h Function C0h PS/2 Mouse Support

Reference Number - The BIOS reference number appears on the bottom line of the screen during startup, as follows:

Ref. TTTT-XXXX-042088-Kn

TTTT	BIOS type
XXXX	customer number
042088	the BIOS release date
K*n*	the keyboard BIOS version number. If *n* is 0, it is not an American Megatrends keyboard Controller BIOS.

INT 13h Park Heads - A new INT 13h function, AH = 19h Park Heads, has been added. DL must contain the drive number (80h or 81h).

INT 13h Function AH = 08h Return Parameter - This function returns the actual number of cylinders in the hard disk drive, regardless of the number. Previously, the highest number returned was 1024.

Using Default Values for C&T NEAT- and 386 CHIPSet-based systems - The C&T NEAT or C&T 386 chipset registers can be programmed incorrectly. When this happens, the system can be difficult to boot. By pressing <Ins> at power-on or after a hard reset, the end users can program the chipset registers with default values stored in CMOS RAM, thereby booting the system. The end user then must run Setup to optimize configuration values.

Enhanced IRMA 3270 Emulation support.

Improved 1.44 MB 3½" drive compatibility.

Supports Toshiba ND-04DT-A 360 KB floppy.

INT 10h - Video I/O functions execute approximately 25% faster.

Extended Setup for the AMI BIOS for the NEAT 286 and C&T 386 Chipsets - EASY SETUP is now available for the AMI BIOSes for the C&T NEAT and 386 chipsets. It configures DRAM wait states, clock speed, and shadowing options.

Translates 80286 LOADALL - This is necessary only in the AMI 386 BIOS. OS/2, RAMDRIVE, and certain other programs use the 80286 LOADALL instruction, which does not exist in the 80386 instruction set. The BIOS translates LOADALL to an 80386 format, but needs an extra 100 bytes to do so. The BIOS can either use the BIOS Stack Area from 0:300h–0:400h, or use the top 1 KB of user memory (at 639K).

OS/2 and RAMDISK Support Option (1/2) - Typing 1 specifies that the BIOS should use the BIOS Stack Area at 0:300h. Typing 2 specifies that the BIOS should use the top 1 KB of the DOS memory area.

OS/2 can be booted from a 1.44 MB 3½" floppy.

ESDI, SCSI, and RLL hard disk drives - If using ESDI, SCSI, or RLL hard disk drive, the AMIBIOS in your system should be 092588 or later.

Configure Workstations - The end user can configure file servers or diskless workstations by bypassing keyboard, video, and floppy error messages.

No F1 after Error Message - The end user does not have to press <F1> key after error message display.

Serial Ports - A maximum of four serial ports is now supported. COM1 is 3F8h, COM2 is 2F8h, COM 3 is 3E8h, and COM4 is 2E8h. The BIOS only supports data transfer and programmability for COM1 and COM2.

System Configuration - A new System Configuration Screen displays number and type of drives installed, total RAM, math coprocessor presence, amount of cache memory, and other system configuration data.

1024 Cylinders - The BIOS recognizes a maximum of 1024 cylinders on a hard disk drive to resolve DOS and Speed Store problems when using a drive with more than 1024 cylinders.

Problems with a Western Digital WD1003V-MM2 hard drive controller and a Miniscribe hard disk drive have been resolved.

82C302C Support - The 82C302C uses a 4 KB page size instead of the 2 KB page size used by the 82C302. The new AMIBIOS automatically detects which chip is in the system and supports either chip.

C&T 386 Soft Reset Bypass - BIOS default values or user-configured Extended CMOS Setup values were programmed into the chipset registers at cold boot and at soft reset. Now they are only set at cold boot.

C&T 386 Memory - The BIOS will now accept 256 KB RAM chips in banks 0 and 1 and 1 MB RAM chips in banks 2 and 3. The BIOS automatically assigns physical banks 2 and 3 as logical banks 0 and 1 and uses all available memory.

AMI 286 and 386 BIOS - 12/15/1988, Continued

C&T 386 and NEAT Clock Switching - Clock switching via a keyboard controller pin, programming the processor clock speed, or programming the processor clock and bus speed are now supported. A keyboard controller pin can also be used to switch the Turbo LED pin on and off.

Additional NEAT Feature - The AMIBIOS for the NEAT chipset now supports the 80386SX processor and 80387SX math coprocessor.

82C212B Support for NEAT BIOS - The AMIBIOS for the NEAT chipset automatically detects the 82C212 or 82C212B and programs the chipset registers accordingly. Extended CMOS Setup supports both chips.

AMI 286 and 386 BIOS - 2/25/89

64 MB - Support for up to 64 MB has been added, if the system hardware supports memory above 16 MB.

IDE - Support for Conner IDE interface drives has been added.

OS/2 - Support for OS/2 in the scratch RAM area has been added. You must use Type 47 for hard disk drives for OS/2.

SCSI - Support for the Western Digital 8-Bit SCSI Controller has been added.

AMI 286 and 386 BIOS - 3/25/1989 and 3/30/1989

New CMOS Setup Utility Features:

- full-screen editing
- user-defined drive types for drives C:/D:
- bypass keyboard, floppy and video error reporting

Diagnostics for 1.44 MB Floppy Drives - The BIOS Diagnostics utility will now run on 1.44 MB floppy drives.

Diagnostics for User-Defined Hard Drives - The BIOS Diagnostics utility will now support any user-defined hard disk drive.

NEAT and C&T 386 Chipset BIOS - Previous AMI BIOS products forced the system to 1 DRAM wait state if only 1 bank of memory was used. The BIOS now allows 0 wait states if selected by the end user in Extended CMOS Setup.

AMI 386 Mark II AT/XT BIOS - 4/25/1989

Automatic CMOS Memory Size Adjustment - In previous versions of this BIOS, the end user had to execute Setup twice when setting the *Shadow RAM* and *256KB Relocation* options.

No System Configuration Screen - The BIOS System Configuration screen is no longer displayed.

AMI 286 and 386 BIOS - 4/30/89

User-defined Drive Type 47 implemented in CMOS Setup.

AMI 286 and 386 BIOS - 9/15/89

INT 15h - Function 87h now returns error codes via I/O port 80h.

POST - The keyboard timeout period in POST has been enlarged to accommodate some keyboards with a slow response time.

Seek - The timeout values for hard disk Seek have been enlarged to accommodate some slow hard disk drives.

Floppy Test - Previous AMIBIOS products did not test the floppy disk drives if no floppy drives were configured in CMOS RAM. If floppy drives are part of the system, they are now tested.

AMI 286 and 386 BIOS - 12/15/89

Western Digital 8-bit SCSI Controller support added.

Also corrected problems with Western Digital 8-bit hard drive controllers that conflict in use of BIOS Data Area.

AMIBIOS - 4/9/90

IDE hard drive support fixed. To use an IDE drive in your system, the AMIBIOS Date should be 040990 or later.

PS/2-compatible mouse support added.

INT 19h modified to support IBM Token Ring Network Card.

Floppy controller reset added during bootup to work with DR DOS.

Up to 4 GB of RAM recognized with some chipsets.

AMIBIOS - 10/15/90

Hard drive type 47 will now work under Novell.

The POST memory test performs faster.

INT 15h Block Memory Move now performs faster.

AMIBIOS for HT12 Chipset - 11/15/1990

Shadowing - Older versions of the AMIBIOS for the HT12 chipset may have had problems configuring shadowing under these conditions:

- the motherboard has 1 MB,
- the user selects remapping and no shadowing, and
- then disables remapping and enables shadowing.

This release fixes the above problem.

Error Message - This release also eliminates the CMOS Memory Size Mismatch error message when the end user toggles the shadowing and remapping options.

AMIBIOS for HT12 Chipset - 10/15/90

Shadowing - The previous version of this BIOS had trouble with shadowing when there are 2.5 MB of RAM in the system, with BANK1 populated by 256 KB RAM and BANK2 with 1 MB RAM. ROM Diagnostics may not work with this configuration either. This is a hardware problem, but American Megatrends implemented a BIOS solution.

AMIBIOS - 02/02/91

Can now run OS/2 in CGA mode.

New BIOS Setup screens - Advanced CMOS Setup and Advanced CHIPSET Setup.

INT 15h Function C2h PS/2 Mouse Support Added

Hi-Flex AMIBIOS Release - 3/15/1991

Turbo Switch 8042 Pins - The following 8042 keyboard controller pins can be used for Turbo Switch Input Pins: 27, 28, 29, 30, 31, and 33. Pins 23 and 24 cannot be used as Turbo Switch Input Pins. Pins 23, 24, 27, 28, 29, and 30 can be used for clock switching.

Turbo Switch Option has been added to Advanced CHIPSET Setup. It can be Enabled or Disabled. If the Turbo Switch option is Present and Enabled and the Turbo Switch is Low at Power-on, the CPU speed is set low. Otherwise, the speed is set in Standard CMOS Setup.

BIOS ID Strings - The last two (of three) BIOS Identification Strings do not appear on the screen. Press <Ins> during POST to display these strings.

Timer Channel 1 - The Timer Channel 1 test for refresh has been removed, which corrects a problem on some 33 MHz 80486DX-based systems.

The AMIBIOS now recognizes up to 4 GB of RAM without customization.

AMIBIOS 4/91

No major features added.

AMIBIOS 07/07/91 and 09/09/91

No major features added.

AMIBIOS 12/12/91

Additional BIOS Setup Screens - Peripheral Setup and Power Management Setup.

BIOS Size - The run-time BIOS size is only 32 KB.

Serial Ports - Baud rates up to and including 19,200 bps are now supported.

Setup Option - The Daylight Saving option has been removed from Standard CMOS Setup.

INT 15h - Function AH = C1h Get Extended Data has been added.

Processor - The BIOS automatically detects the following processor types: Intel 80486DX, 80486SX, 80386DX, and 80386SX.

8042 Pin - Pin 32 of the keyboard controller can be used to remove the password checking facility. If pin 32 of the keyboard controller is connected to GND, the password is set as uninstalled. Please note that a null password is not a valid password.

Password - The password option in Advanced CMOS Setup has only two settings: Setup, or Always. Please note that Bit 7 of CMOS register 34h is available for use.

New Features

- 2.88 MB 3½" floppy drive support has been added in the BIOS.

- INT 16h has these new functions:
  - Function F0h Set CPU Speed
  - Function F1h Read CPU Speed
  - Function F4h Subfunction 00h Read Cache Controller Status
  - Function F4h Subfunction 01h Enable Cache Controller
  - Function F4h Subfunction 02h Disable Cache Controller

- *Boot Sector Write Protection* — This Advanced CMOS Setup option warns the end user any time a program attempts to format or write to the boot sector on the hard disk drive.

- *Auto Detect Hard Disk Drive* — this AMIBIOS Setup main menu option detects hard drive parameters for IDE, SCSI, and other non-MFM drives.

- AMIBIOS now automatically detects AMD386DXL, Cyrix Cx486SLC and Cx486DLC, IBM 486SLC,and Intel 80486DX, 80486DX2, 80486DX3, 80486SX, 80386DX, 80386SX, and Overdrive™ processors.

Appendix D

Identification Strings

There are three BIOS Identification strings in the Hi-Flex AMIBIOS. Only Identification String 1 appears at the bottom of the screen during boot-up. The end user must press <Ins> to display Identification Strings 2 and 3.

Identification String Line 1

The BIOS identification string appears on the bottom of the screen during BIOS POST. The bytes of Identification String 1 are numbered as follows:

```
xx-xxxx-xxxxxx-xxxxxxxx-xxxxxx-xxxxxxxx-x

12 4–7  9–14    16–23    25–30  32–39    41
```

Byte	Description
1	Processor Type 0 8086 or 8088 2 80286 3 80386 4 80486
2	Size of BIOS 0 64K BIOS 1 128K BIOS
4–5	Major Version Number
6–7	Minor Version Number
9–15	Reference Number
16	Halt on Post Error. Set to 1 if On.
17	Initialize CMOS in every boot. Set to 1 if On.
18	Block pins 22 and 23 of the keyboard controller. Set to 1 if On.
19	Mouse support in BIOS/keyboard controller. Set to 1 if On.
20	Wait for <F1> if error found. Set to 1 if On.
21	Display Floppy error during POST. Set to 1 if On.
22	Display Video error during POST. Set to 1 if On.

Byte	Description
23	Display Keyboard error during POST. Set to 1 if On.
25–26	BIOS Date. Month (1-12).
27–28	BIOS Date. Date (1-31).
29–30	BIOS Date. Year (0-99).
32–39	Chipset Identification. BIOS Name.
41	Keyboard controller version number.

```
xx-xxxx-xxxxxxxxxxxxxxx-xxxxxx-xxxxxxxx-x

12 4–7  9–15   16–23    25–30  32–39    41
```

Byte	Description
1	Processor Type 0 8086 or 8088 2 80286 3 80386 4 80486
2	Size of BIOS 0 64 KB BIOS 1 128 KB BIOS
4–5	Major Version Number
6–7	Minor Version Number
9–15	Reference Number
16	Halt on Post Error. Set to 1 if On.
17	Initialize CMOS in every boot. Set to 1 if On.
18	Block pins 22 and 23 of the keyboard controller. Set to 1 if On.
19	Mouse support in BIOS and keyboard controller. Set to 1 if On.
20	Wait for F1 if error found. Set to 1 if On.
21	Display Floppy error during POST. Set to 1 if On.
22	Display Video error during POST. Set to 1 if On.
23	Display Keyboard error during POST. Set to 1 if On.
25–26	BIOS Date. Month (1-12).
27–28	BIOS Date. Date (1-31).
29–30	BIOS Date. Year (0-99).
32–39	Chipset Identification. BIOS Name.
41	Keyboard controller version number.

Identification String Line 2

```
xxx-x-xxxx-xx-xx-xxxx-xx-xx-xxx

123-5-7    -12-15-18   -23-26-29
```

Byte	Description
1–2	Pin number for clock switching through keyboard controller.
3	Indicates High signal on pin switches clock to High(H) or Low (L).
5	Clock switching through chipset registers 0 No clock switching through chipset registers. 1 Clock switching through chipset registers.
7–10	Port address to switch clock high through special port.
12–13	Data value to switch clock high through special port.
15–16	Mask value to switch clock high through special port.
18–21	Port Address to switch clock low through special port.
23–24	Data value to switch clock low through special port.
26–27	Mask value to switch clock low through special port.
29–31	Turbo Switch Input Pin information (Pin number for Turbo Switch Input Pin).

AMI BIOS and AMI BIOS Plus Identification Strings

AMI BIOS and AMI BIOS Plus were sold from 1986 through 1990.

The general format of the BIOS Reference string in this type of AMI BIOS is:

```
Ref. TTTT-XXXX-042088-Kn
```

TTTT	BIOS type
XXXX	customer number
042088	the BIOS release date
K*n*	the keyboard BIOS version number. If *n* is 0, it is not an American Megatrends keyboard controller BIOS.

Glossary

512K – 640K Onboard

This is a BIOS Setup option used in older AMIBIOSes. You can specify if the memory between 512 KB and 640 KB is on the motherboard.

640K – 1 MB Relocation

This AMIBIOS Setup option allows you to enable the relocation of the 640K – 1 MB memory area. This area is usually reserved for ROM, but if shadow RAM is not enabled and the RAM in this area is available, it can be remapped to a location above 1 MB so the RAM can be used as extended memory. This option is also known as memory rollover.

Above 1 MB Memory Test

This an AMIBIOS Setup option. POST tests all system memory if enabled. Only the first 1 MB of memory is tested if disabled.

Access Time

The required time to read or write data to RAM or other storage device. Since the operating environment and varying conditions affect access time, this is usually given as an average.

ACK

A control code (06h) sent to a sending station or computer by the receiving unit to acknowledge either that the receiver is ready to accept transmissions or that transmitted data arrived without error. The ability to receive and send acknowledgement signals is built into the hardware and software. For example, the keyboard controller BIOS and serial ports send and receive ACK commands. See also: NAK.

Adapter Card

A flat rectangular fiberglass board with electronic circuitry. Inserted in an expansion slot on the computer's main bus, it provides additional system functions, such as device controllers or video adapters.

Adaptor ROM

The read-only memory on the adapter, which contains code to control the adapter device. An adapter is a peripheral card that extends the operational capabilities of the system. Many hard disk drive controllers have adaptor ROMs.

Address

Every memory location is numbered consecutively. This number is the address of the memory location. An address can be a label, number, or name that identifies a register, memory location, or a location on a disk drive or external device accessed via an I/O port.

Address Bus

One or more lines (conductors) that carry address codes from the microprocessor to other parts of the system.

Analog

A term used to describe any device that represents values by a continuously varied physical property, such as voltage.

Arbitration

A process where devices compete for possession of the channel on a prioritized basis. Used in EISA computers.

Arbitration Levels

Arbitration levels are the levels of priority assigned to devices that compete for possession of the channel.

Area Cache Enable

This is an AMIBIOS Setup option. Caching can be enabled for specific memory segments. The memory areas that can be cached vary from one chipset to another. The cache values are for the actual RAM space, such as 128 KB or 640 KB, not memory segments.

ASIC

Application-Specific Integrated Circuit. A special type of chip or logic array designed to perform a specific function. The chip starts out as a nonspecific collection of logic arrays. During the manufacturing process, a layer is added to connect the gates for a specific function. By changing the pattern of connectors, the manufacturer can make the chip suitable for many purposes.

ASCII Code

American Standard Code for Information Interchange. An industry standard 7-bit code consisting of control, character, and graphic codes (8 bits if the parity bit is included).

Assembly Language

Assembly language is a way of representing the machine language of a computer. Every computer architecture has its own assembly language, which is a low level language that uses mnemonic instructions instead of binary numbers to represent a machine language instruction.

Asynchronous Event Notification (SCSI)

A process by which a SCSI target can send unsolicited sense information to an initiator using the SCSI SEND command.

Asynchronous Data Transfer (SCSI)

Data transfer (usually at a low rate and independent of any external timing constraints) performed by a SCSI device involving the interlocking of a signal to the initiator (REQ) and a signal to the target (ACK) such that each step of the data transfer protocol must occur before the next step can begin.

AT Bus

The ISA (Industry Standard Architecture) bus. The 16-bit electronic path used to connect the motherboard and peripheral devices.

AT Bus Clock Source

Often used as an option in AMIBIOS Setup. The AT Bus Clock option determines the speed at which I/O bus (AT bus) transfers occur in the system. The standard speed for this clock is approximately 8 MHz. The AT Bus Clock Source option sets the speed at which these types of transfers occur.

AT Bus Command Delay

Often used as an option in AMIBIOS Setup. It is the amount of time (command delay) inserted before a command is executed on the AT bus. Often, separate Setup options appear for 8-bit and 16-bit AT Bus command delay.

AT Bus I/O Command Delay

Often used as an option in AMIBIOS Setup. This is the amount of time (command delay) inserted before an I/O operation is executed on the AT bus. Often, separate AMIBIOS Setup utility options appear for 8-bit and 16-bit I/O commands.

AT Bus Wait States

Often used an option in AMIBIOS Setup. This is the number of wait states inserted before an operation is performed on the AT bus.

AT-Compatible Computer

Any computer that can run software programs written for an IBM AT computer. The AT bus hardware signals and standards are also known as the Industry Standard Architecture (ISA).

Attribute

A characteristic of a data structure that helps define the structure. An element in a data model.

Bank

A group of memory devices arranged for easy access. Most motherboards have two banks of memory devices. Each bank holds four single-inline memory modules (SIMMs).

Bank Switching

A method of expanding the system memory by switching between banks of memory, only one of which is accessible at any one time. Each bank in turn uses the same area of memory, which is set aside before switching begins. The memory bank still retains the memory while inactive. Before a different bank can be used, the operating system or other system software must explicitly switch banks.

Baud

A unit of measurement of the discrete number of signal elements that can be transmitted per second by a device. It is not a measure of the amount of information being transmitted, and it is not the same as Bits Per Second.

BBS

Bulletin Board System. A computer equipped with one or more modems that serves as a message-passing and information center for dial-up users.

Benchmark

A process or program that measures and compares the performance of one or more systems or subsystems.

Binary

The base two numbering system, where the only digits are 0 and 1. It is used by all computers.

BIOS

Basic Input Output System. Systems software that interfaces between the operating system and the hardware.

BIOS Service

A software routine that services a certain type of hardware device and provides an interface between the operating system and the hardware device. These services are single-task, call/return functions.

BIOS Setup Defaults

The AMIBIOS Setup option has two sets of default settings: Power-On and BIOS defaults. The BIOS Setup default values provide optimum performance settings.

Bit

A binary digit that can take either the value 0 or 1. A bit is the smallest unit of information that a computer can process.

Bit-Mapped

A method of storing graphical information in memory in which one memory bit is assigned to each screen pixel. The memory bit is either ON or OFF to indicate that the corresponding screen pixel is either on or off.

Bits per Second

BPS. The number of binary digits that can be transmitted in one second. Generally, modem speeds are given in BPS, not baud rate. Neither baud rate nor BPS take into account the gaps between transmissions, so neither baud rate or BPS truly measure the amount of information being transferred.

BMIC

Bus Master Interface Controller. A chip that arbitrates control of the bus in an EISA system.

Boot

A sequence of system events that prepare the computer for use. Booting begins a routine that flushes memory and loads the operating system. The process of booting a system is also called bootstrap loading.

Buffer

An area of memory or storage that is temporarily reserved for I/O processing.

Burst Mode

A method of data transfer that allows a device to remain inactive for long periods of time and then send large amounts of data in a short time without interruption. Can be used for DMA transfers on the EISA bus.

Bus

A set of wires or other electrical conductors used for communication between two or more devices. Power, data, and control signals are transferred on the bus. Implicit on a bus are specific protocols used to make sure that a communication is correct.

A bus is a shared highway that connects the processor, memory, disk drive controller, and I/O ports and enables these components to transfer information between them. Bus latency is the time delay between a request for control of the bus by a bus master and the granting of control.

Although usually supervised by the microprocessor, the bus can be taken over by a coprocessor on an expansion card in the bus.

Buses are characterized by the number of bits they can carry at a time.

Bus Master

In EISA systems, the owner of the bus could be the EISA motherboard or any 32-bit EISA card that has bus mastering capability. By owning the bus, the bus master can supervise communications between devices on the bus independent of the host CPU. A bus slave device (ISA or EISA) attaches to the bus but does not have the ability to control the bus.

Byte

A unit of data made up of eight contiguous bits. A byte is usually the smallest addressable unit of memory.

Cache

A method of speeding access to information in a slower device by temporarily storing the information in a faster device. For example, data stored in 70 ns DRAM can be stored temporarily in 12–18 ns SRAM cache memory for quicker access. The system that determines which data is stored in SRAM cache memory is called a caching algorithm.

Cache memory uses an associated tag memory for faster searching.

Cache Mapping

Cache mapping is a technique for using cache memory in a system. This option appears on many AMIBIOS Setup screens. It can usually be either direct, two-way, or four-way. Direct mapping is when the cache memory is located in one bank.

N-way Mapping is when the cache memory is divided into *n* banks (two-way, four-way, eight-way, and so on).

CAS

Column Address Strobe. RAM is organized in rows and columns and is accessed via signals (strobes) sent along lines to these rows and columns. The CAS is a signal line which clocks the column address into an internal address latch. The CAS is measured in nanoseconds; the lower the value, the faster the RAM can be accessed. CAS also acts as output enable for a memory cell. When CAS is asserted, the tri-state driver on the data pin is enabled.

CD

Carrier Detect. A signal sent from one modem to another modem to indicate that the modem is online and has received the carrier signal (the high-pitched tone generated by the sending modem). See DCD.

CD-ROM

Compact disk read-only memory. A device that uses laser optics instead of magnetic heads to read data. A CD-ROM is capable of storing large amounts of data.

Centronics Parallel Interface

A de facto standard for data exchange using parallel ports between computers and peripheral devices. It provides eight parallel data lines plus additional lines for control and status information.

CGA

Color Graphics Adapter. The initial color video standard for the IBM XT. CGA monitors operate at a horizontal scanning frequency of 15 – 75 KHz. The BIOS INT 10h Video Service provides functions that control the displays on CGA systems. See also: EGA, CGA, HGC, MDA, and XGA.

Channel

A path constructed specifically for moving data.

Checksum

An error checking mechanism for a range of data. The sum of a data block is compared with the sum of another data block. The carry is ignored. A discrepancy indicates an error.

Chip

An integrated circuit.

Chipset

A small number of integrated circuits that perform the functions of many or all of the discrete logic devices in an ISA or EISA system.

Cluster

A group of sectors on a disk that forms the fundamental storage unit addressed by the operating system. The operating system controls the number of sectors in a cluster. MS-DOS assigns 4 sectors to a cluster.

CISC

Complex Instruction Set Computing. Describes a processor that uses complex assembly language instructions that usually require many clock cycles to execute. The instructions are usually powerful and permit complex and flexible ways to calculate such elements as memory addresses. See also: RISC.

CMOS

Complementary Metal Oxide Semiconductor. A method of producing ICs, or an IC produced by this method. CMOS chips consume very little energy.

CMOS RAM

Nonvolatile, battery-backed, low-power memory in XT, AT, EISA, and PS/2 systems used to store system configuration information. CMOS devices have extremely low power consumption rates and a high tolerance for noise from the power source. See also: DRAM, SRAM.

COM

The DOS logical device name for serial communications. A COM port is a serial communications channel over which data can be transferred between remote devices. This name is part of the four names reserved by DOS for the serial communications ports (COM1 – COM4).

Command Control Block

CCB. A software object prepared by the host software to be used by the SCSI host adapter to provide control information needed by the computer to execute a SCSI command.

Common Command Set

CCS. A de facto standard SCSI command set for communication with hard disk drives. In the SCSI-2 command set, CCS is the basis for all commands to all types of peripheral devices.

Command Descriptor Block

CDB. A block of information passed on the SCSI bus that provides the command, parameter, and address information necessary for the target to execute the specified function. It is prepared by the host software and placed in the CCB to be passed to the target by the SCSI host adapter.

Command Specify Block

A data structure used in SCSI disk I/O operations.

Concurrent Refresh

Often used as an option in AMIBIOS Setup. If memory refresh is performed while the CPU is performing other tasks, concurrent refresh is being used.

Configuration

A collection of system parameters and settings that control communication between the system, major system components, and adapter cards and peripheral devices. Also, the process of establishing the system parameters.

Configuration is performed by the BIOS Setup utility. In the AMIBIOS, there are several parts of the Setup utility that configure various system options and perform diagnostics tests.

Conventional Memory

Memory located between addresses 0 and 640 KB. The only memory that DOS applications programs can directly address. See also: extended memory.

Coprocessor

A processor which assists the main processor, executing in parallel, with floating-point arithmetic calculations. A math coprocessor can process math calculations much faster than the general purpose microprocessors found in personal computers. The 80486 contains the functionality of the Intel 80387 math coprocessor.

CPU

Central Processing Unit. The brain of the computer, where internal storage, processing, and control circuitry is located.

CTS

Clear To Send. A signal from a modem to the computer. A modem signal that indicates that the modem is ready to transmit. CTS uses line 5 in the RS-232C specification.

Cycle Time (memory)

The amount of time it takes for RAM to read from or write to a memory cell. There are two stages to accessing RAM: precharge and access. The capacitor in the memory cell has time to recover from a previous access and stabilize the charge during precharge. During access, a bit is moved between memory and the bus or CPU. The cycle time is the total of the precharge and access times.

Cyclic Redundancy Check

CRC. A bit-oriented type of error-checking that uses checksums. A redundancy check where the check key is produced by a repeating algorithm. See also: ECC, checksum.

Cylinder

A set of tracks on a hard disk drive that can be accessed without moving the read/write heads. A vertical column of tracks on a disk. See also: sector, track.

Cylinder Skew Factor

The cylinder skew factor on a disk drive is the number of sectors between the last sector in the present cylinder and the first sector in the next cylinder plus 1. This factor can vary considerably from one hard drive to another, since selecting a cylinder involves moving the read/write heads. The speed at which the heads move can vary considerably.

DAC

Digital-to-Analog Converter. A device used in VGA hardware to convert commands and data when interfacing between digital computer hardware and an analog monitor.

DCD

Data Carrier Detected. A signal used in serial communications sent by a modem to a computer to indicate that the modem is online and ready for transmission. DCD uses line 8 in the RS-232C specification.

Default

A preset configuration parameter. A value, setting, or option that is preassigned by the program or system.

Density

The number of bits or characters that can be recorded in a specified space.

Descriptor

An 8-byte part of the Intel x86 protected mode virtual address translation table. It contains the starting and ending segment address and access privileges for the segment.

Device Driver

A program that is linked with or attached to an operating system to map the software interface or the operating system to the requirements of attached peripheral devices and host adapters.

Device Service Routine

DSR. A type of BIOS routine that provides a specific set of functions for a particular type of peripheral device. For example, the INT 10h Video Service is a DSR.

Digital-To-Analog Converter

DAC. Used in VGA hardware to convert command and data between digital computer hardware and an analog monitor.

DIP

Dual Inline Package. A standard method of organizing and packaging ICs. The ICs are enclosed in a rectangular plastic housing and are connected to pins that extend from the sides and protrude downward. The pins fit in sockets on the circuit board or can be soldered directly to the board. DIPS often include slide or rocker switches. Other chip packages include: leaderless chip carrier, pin grid array, and surface mount technology.

Direct Memory Access

DMA. A method of communicating between a device and system memory. This method does not use the CPU to manage the data transfer, thus improving system performance. The BIOS uses DMA to transfer data between memory and the disk controllers.

DMA Wait States

This is often an option in AMIBIOS Setup. It is the number of wait states inserted before DMA commands are executed. Often this option appears separately for 8-bit and 16-bit DMA Wait States.

DMA Clock Source

This is often an option in AMIBIOS Setup. This option sets the source for the DMA clock.

DRAM

Dynamic RAM. A form of fairly inexpensive semiconductor memory. DRAM chips store information in integrated circuits that contain capacitors. Because capacitors lose their charge over time, DRAM chips must be continually refreshed, or recharged. While being refreshed, a DRAM chip cannot be read by the microprocessor. Therefore DRAM read and write accesses must often have memory read or write wait states added before they execute. Some American Megatrends motherboards have logic that circumvents this limitation, called refresh bypass.

DRAM Refresh Clock

The dynamic RAM (DRAM) used for data storage must be periodically refreshed so that the values retained within them do not become corrupted (or dissipate) over time.

Typically, refresh is done by sending a pulse to each chip at select intervals of time. The interval of time used between refresh is determined by the DRAM Refresh Clock and can vary significantly.

Drive

A mass storage device that stores information on some form of movable medium, such as rotating disks. See also: Hard Disk Drive, Rewritable Disk Drive, WORM Drive, and Tape Drive.

Driver

A program that extends the capabilities of a computer by enabling the computer to operate peripheral devices, such as WORM drives, CD-ROM drives, or tape drives.

DSR

Data Set Ready. A modem signal that indicates the modem is ready to operate. DSR uses line 6 in the RS-232C specification. DSR also refers to a BIOS device service routine.

DTR

Data Terminal Ready. A modem signal sent by a computer to the modem to indicate that the computer is ready to accept incoming transmissions. DTR also is used to answer a ring (see RI). DTR uses line 30 in the RS-232C specification.

Dual Inline Package

See DIP.

EGA

Enhanced Graphics Adapter. A color video standard that displays resolutions as high as 640x350 with 16 colors from a palette of 64 colors and uses its own video BIOS as well as the video BIOS components of the system BIOS. The EGA standard uses a digital (TTL) video signal that can display up to 64 colors simultaneously on an EGA-compatible monitor. EGA monitors operate at a horizontal scan rate of 21.85 KHz. The EGA video BIOS is usually located on the video adapter card. See also: VGA, CGA, XGA.

EISA

Extended Industry Standard Architecture. A set of standards for a 32-bit bus that is compatible with the AT (ISA) bus. See also: ISA.

EEPROM

Electrically Erasable Programmable Read-Only Memory. An integrated circuit that stores the host adapter firmware and BIOS. It can be erased and reprogrammed by a special device.

A device for storing firmware — information and code that does not change for a long period of time. See also: EPROM.

EMS Hardware Support

There are often several EMS options on AMIBIOS Setup screens. Many chipset designs incorporate EMS (Expanded Memory Specification) hardware support by providing registers for maintaining EMS data.

The EMS driver, usually software-based, must use parts of RAM to provide these data registers. But if the chipset has provided these registers — and the EMS driver knows how to access hardware registers properly — EMS performance can be increased by seven to ten times.

EPROM

Erasable Programmable Read-Only Memory. A chip with a glass window. Useful for storing firmware. EPROMs can be erased by removing the protective cover from the chip package and exposing the semiconductor material to ultraviolet light. They can then be reprogrammed. See also: EEPROM, PROM.

EPROM Wait States

The number of wait states used during read/write operations to and from an EPROM. This is an option that often appears in AMIBIOS Setup screens.

Error Checking and Correction

ECC. An algorithm that detects and corrects all single-bit errors using a coding technique that tests the accuracy of the transmitted data. See also: CRC.

ESDI

Enhanced Small Device Interface. A standard device interface for hard disk drives, floppy drives, and tape drives. ESDI drives can store up to 1 GB of data. ESDI is both a data recording technique that uses a high density storage format to store more data in a given space and a device interface standard that permits much higher data access speeds. ESDI circuitry is included on both the controller card and an embedded controller in the drive. ESDI has a serial command line in addition to standard control lines that accepts 17-bit commands. Data can be transferred at rates from 10 to 20 Mbs using ESDI. Up to 36 sectors are used per track instead of the normal (MFM, ST506 standard) 17 sectors per track.

To use an ESDI hard disk drive, you must also have an ESDI controller. See also: RLL, MFM, SCSI.

Expanded Memory

Up to 32 MB of additional paged memory that resides above conventional (DOS) memory and uses bank-switched memory and a software program called the EMM (Expanded Memory Manager). Application programs can use the Expanded Memory Specification (developed by Lotus, Intel, and Microsoft) to access this type of memory with special hardware or software. See also: extended memory.

Expanded Memory Specification

EMS. A specification and protocol established by a consortium of computer manufacturers. It establishes a set of rules for organizing and accessing extended memory for use as conventional DOS memory. Up to 32 MB of extended memory can be accessed via a 64 KB window in conventional memory divided into four 16 KB memory pages.

Expansion Slot

Almost all ISA and EISA systems have sockets on the motherboard where additional PCBs (adapter cards) can be inserted to expand the system's capabilities by adding additional peripheral devices or memory. Expansion slots must be designed for either 8-, 16-, or 32-bit adapter cards.

Extended I/O Decode

This option often appears in AMIBIOS Setup. In ISA systems, 10 address lines (A9 – A0) are normally used for I/O address decoding. If a design allows for more than 10 address lines to be used for I/O addressing, the Extended I/O Decode option must be enabled.

Extended Memory

The memory above 1 MB accessible only in protected mode. Microsoft Windows, Xenix, and Unix can access this memory, but DOS and ordinary applications programs cannot because they must use real mode. Extended memory is directly addressable when the processor is in protected mode. See also: expanded memory, conventional memory.

External Cache Memory

This is an AMIBIOS Setup option and enables or disables the cache memory that does not reside in the system microprocessor.

Fast Gate A20

This is an AMIBIOS Setup option. Gate A20 in the Intel x86 architecture controls access to memory addresses above 1 MB by enabling or disabling access to processor address line 20. To be able to access conventional memory (from 0 – 1024K), address line A20 must always be low — so Gate A20 must be disabled.

However, some software programs both enter protected mode and shut down through the BIOS. For these programs, Gate A20 must be constantly enabled and disabled via the slow keyboard controller.

Fast Gate A20 is a hardware feature that speeds Gate A20, which in turn speeds programs that constantly switch from conventional memory to extended memory and back. For example, enabling this option allows programs such as Microsoft Windows to execute faster.

FAT

File Allocation Table. A hard disk drive table where DOS tracks both currently used disk space and available disk space for new or extended files.

FCB

File Control Block. DOS maintains information about the status of every file in the file control block.

File

A collection of similar data that is organized in a similar format.

Firmware

Any device or the programs and data on the device that is stored on ROM chips. Used for programs or data that change infrequently.

Flash EPROM

A new type of memory that combines the flexibility of RAM and the permanence of disk storage. It does not need backup power and can be packaged like DRAM but in considerably less space. It is ideal for handheld and laptop systems.

Floppy Disk

A flexible storage device that consists of a thin round piece of mylex coated with a magnetic material enclosed in a flexible paper container (5¼ disks) or rigid plastic container (3½ disks)

Floppy Drive Seek at Boot

This is an AMIBIOS Setup option. If enabled, a Seek command is performed on the floppy drive A: at system boot. This option is usually disabled to speed the boot process.

Gap Length

The gap of unused space between blocks of data on a disk surface. The gap exists so that new data can be written over old data without harming the adjacent data. Disk speed varies slightly, so new data cannot be written precisely in the space occupied by the old data. A gap of unused space is always placed between data.

Gate

An electronic switch that follows a rule of Boolean logic, such as AND, OR, or NOT. A small component of a semiconductor device that produces an electrical output signal logically related to the states of one or more input signals. Logic gates can be combined to allow a device to add, subtract, compare, and perform other operations.

Gate Array

See ASIC.

Gigabyte

1,073,741,824 bytes, as in 1 MB times 1,000, or 2^{30}.

GDT

Global Descriptor Table. Part of the Intel x86 architecture. A table that contains memory segment descriptors that are available to all tasks in the system.

Graphic Mode

A display mode where all pixels can be controlled independently.

Hard Disk Drive

Fixed disk drive. A rotating disk that cannot be removed from its housing. It has a drive mechanism, a read/write mechanism, and hard disks sealed in the drive enclosure. Data on a hard disk can be written over or erased.

Hard Disk Type 47 RAM Area

This is an AMIBIOS Setup option. In the hard disk drive configuration in Standard CMOS Setup, the user-definable hard disk type (type 47) can configure a nonstandard hard disk drive for drive C: or D:. Ordinarily, the data for the type 47 disk type is stored in 0:300h in lower system RAM or in the top 1 KB of applications memory, starting at address 639K.

The end user can specify a user-definable hard disk type for drive C: and drive D:. If specified, the actual drive parameters must be entered by the end user in Standard CMOS Setup.

Hardware

The physical equipment and components in a computer system.

Hardware Interrupt

A type of interrupt (request for service) generated either externally by a hardware device, such as the keyboard, disk drives, and I/O ports, or internally by the microprocessor.

External hardware interrupts are used by devices to request attention from the microprocessor. Internal hardware interrupts are generated by the microprocessor to control events (for example, to report an attempt to divide by zero). Hardware interrupts are assigned levels of priority. Nonmaskable interrupts (NMIs) receive the highest priority because they indicate a serious problem.

Head

The read/write head on a floppy or hard disk drive. This device provides a means of writing to and accessing data on the disk.

Head Load Time

A floppy disk drive parameter. The amount of time needed to allow the drive head to settle after it is lowered onto the drive surface.

Head Settle Time

A floppy disk parameter. The amount of time required for the heads to settle after a Seek operation.

Head Unload Time

A floppy disk drive parameter. The amount of time needed to allow the drive head to settle after it is lifted from the drive surface.

Hercules

An older video display standard that displays monochrome text in a 720 x 348 screen. Also known as HGC — Hercules Graphic Card.

Hexadecimal

The base 16 numbering system that uses the digits 0 through A, B, C, D, E, and F. In this book, hexadecimal numbers are followed by *h*.

Hidden Refresh

Often an option in AMIBIOS Setup. Hidden refresh is used when memory refresh occurs without holding the CPU. Similar to Concurrent Refresh.

High Memory Area

HMA. An extra 64 KB bank of memory that DOS 5.0 can access on systems with an EMS driver. The HMA can be controlled by only one program at a time. Windows, Desqview, and DOS all use the HMA. You can save conventional memory space by placing an application in the HMA. Placing DOS in the HMA saves about 50K.

Hit <DEL> Message Display

This is an AMIBIOS Setup option. Disabling this option prevents the

```
Hit <DEL> if you want to run Setup
```

message from appearing when the system boots.

Host

In SCSI systems, the computer in which a host adapter is installed. The host uses software to request the host adapter services in transferring information to and from the peripheral devices attached to the SCSI bus connector of the host adapter.

Host Adapter

An adapter card installed in the computer expansion bus that provides a SCSI bus connection that allows SCSI devices to be connected to the computer bus. A host adapter is said to be intelligent when it has a high-level software interface to the computer.

IOR/IOW Wait States

Often an option in AMIBIOS Setup, this is the number of added wait states before I/O read and write commands are executed.

Integrated Circuit

Also known as an IC or chip. The packaging of circuit elements such as transistors and resistors on a single chip of silicon or other material. A complete electrical circuit on a small silicon chip that can contain one to over a million components such as transistors, resistors, and capacitors.

IDE

Integrated Device Electronics. A type of computer/hard disk drive interface in which the electronic circuitry for the controller resides on the drive itself. There is no need for a separate controller card. The maximum data transfer rate is approximately 7.5 Mbs. A 40-pin cable is used.

The IDE interface is compatible with the Western Digital ST506 controller originally used by IBM in the AT, but also has features such as lookahead caching that increase overall performance.

IDT

Interrupt Descriptor Table. This table contains the 256 gate descriptors that allow the Intel 386 and 486 processors to locate exception-handling routines. Similar to GDT and LDT.

Initiator

A SCSI device that requests that an operation be performed by another SCSI device (the target). The initiator provides all command information and parameters required to perform the operation, but the details of the operation are actually controlled and sequenced by the target.

Interleaved Memory

A RAM memory system in which interleaving reduces wait states. Memory is often organized in rows of chips totalling 256 KB or 1 MB. After a memory access to one of these rows, the processor must wait an entire memory cycle before it can access another byte of memory in the same row. A two-way interleave places odd-numbered and even-numbered bytes of memory in separate rows, so that if the processor accesses an even-numbered location, it can then access an odd-numbered location without waiting.

Interleave Factor

Information on a hard disk drive is organized in a series of concentric circles called tracks. A single track can hold more than 9 KB of data. But much smaller data elements must often be read or written. So tracks are organized into smaller 512-bye units (sectors). The interleave factor is a method of organizing these sectors to optimize hard disk drive performance.

The interleave factor is the most important element in determining how fast data can be retrieved from a hard disk drive. It is the ratio that states how many physical (actual) sectors must pass under the read head before the next logically numbered sector is read or written to.

Logically-numbered sectors are the order in which sectors are accessed by the hard disk drive controller. Physically-numbered sectors are in the actual physical sequence on the hard disk platter surface. Logical sectors are much more important when considering the way in which data is read, since they are an integral part of the actual reading and writing of data.

The interleave factor is equal to the number of physical sectors between two logical sectors plus 1.

In the following example, a track has 17 sectors. The 17 physical sectors in a track can be numbered in several ways. The following table shows the sector numbering sequences for three interleave factors.

Interleave Factor	Logical Sector numbering sequence
1:1	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17
2:1	1,10,2,11,3,12,4,13,5,14,6,15,7,16,8,17,9
3:1	1,7,13,2,8,14,3,9,15,4,10,16,5,11,17,6,12

A hard disk controller reads data from a track in the sequence in which sectors are logically numbered, not by their physical position on the track. For example, when *logical sectors* 1 and 2 are read on a drive with a 3:1 interleave, physical sectors 7 and 13 rotate beneath the read head before the head reads sector 2 (see above table).

Interleave Factor, cont'd

The hard disk is in constant rotation at a prespecified speed. On a 3:1 interleave factor drive, after Sector 1 is read, the disk read head is positioned over Physical Sector 1.

By the time the hard disk drive controller requests a read for Physical Sector 2, the disk has rotated and the read head is positioned over Physical Sector 4.

If the system has a fast disk controller, it is more efficient to organize data in a 1:1 interleave (Sector 2 immediately follows Sector 1).

If the hard disk controller is not fast enough to request data during the time it takes the disk to rotate from Sector 1 to Sector 2, the controller must wait a full revolution of the disk before the head is again over Sector 2 and it can read from or write to the disk.

If the controller is slow, it is more efficient to organize disk data in a 2:1, 3:1, or other interleave factor (see the table above) so that the controller does not have to wait one or two full revolutions of the drive to access the next logical sector.

Internal Cache Memory

This is an AMIBIOS Setup option. This option only appears on 80486-based systems, since the 80486 is the only processor in the Intel x86 series that has internal cache. This option enables or disables the 8 KB cache inside the 80486.

Interrupt

INT. A signal that stops what the computer system is currently doing, allowing it to perform a higher priority task. The suspension of program execution by a demand for attention from a peripheral device. After the interrupt is serviced, the suspended microprocessor task can be resumed at the point where it stopped.

Interrupts are edge-triggered (by a sudden change in voltage, as in the rising edge of a square wave form), or level-triggered (where a small change in voltage determines the state of the device). ISA devices are edge-triggered. EISA devices can be level-triggered.

Interrupt Channel

A channel designated for interruption instructions from the microprocessor so that input/output or other operations can take place.

Interrupt Handler

A separate set of instructions that performs a specific function required by the task at hand. For example, INT 10h is the Video Service interrupt handler. INT 10h provides many specific functions for managing and manipulating information on a video screen.

Interrupt Service Routine

ISR. A type of BIOS routine that handles a hardware interrupt. The BIOS INT 70h Real Time Clock Interrupt routine is an ISR.

IRQ

Interrupt Request Line. Hardware lines over which devices such as I/O ports, disk drives, and the keyboard can send interrupts (requests for service) to the processor. IRQs are built into the computer's internal hardware and are assigned different priority levels so the processor can determine the relative importance of incoming requests for service. There are 16 IRQs. Adapter cards and system resources must be configured so IRQs do not conflict.

Interrupt Vector

An address or offset from an address that is a pointer to a place in memory where an interrupt routine is stored. An interrupt vector contains the address of the routine and is used when a program needs to call the interrupt routine (ISR or DSR) to perform a service.

Interrupt Vector Table

Also called a jump table. A table of addresses and identifiers for interrupt handlers that are performed in response to requests for certain services (such as video, disk access, serial communications, real time clock, and so on).

ISA

Industry Standard Architecture. Another way of saying IBM AT-compatible. ISA refers to the bus signals and timing used in the original IBM AT. It is an ad-hoc standard. No formal timing specifications have ever been published for the AT bus. See also: EISA.

ISR

Interrupt service routine. A BIOS routine that handles hardware interrupts.

Jumper

A small plug or wire that alters some aspect of hardware operation by shunting the electrical flow, or connecting different points in an electrical circuit. Jumpers are used to configure system features.

KB

Kilobytes. 1,024 bytes.

Kb

Kilobits. 1,024 bits.

Kbs

Kilobits per second. The data transfer speed measured in multiples of 1024 bits per second.

Keyboard Controller

A microprocessor in the keyboard that waits for and reports on keystrokes. A keyboard controller and its associated circuitry make a system more efficient because it handles tasks that would ordinarily be performed by the main system processor.

Keyboard Controller BIOS

A set of software routines that manage the hardware/software interface between the keyboard and the system.

Keychord

When two or more keys are pressed simultaneously. For example, **Ctrl Alt Del** is a keychord. Also a keystroke combination.

Laptop Computer

A small self-contained, lightweight, easily-portable computer powered by rechargeable batteries.

LCD

Liquid Crystal Display. A display type that uses a liquid with a polar molecular structure sandwiched between two transparent electrodes. The molecules align when an electric field is applied, arranging in crystals that polarize the light passing through it. A polarized filter laminated over the electrodes blocks the polarized light, transmitting only the nonpolarized light. In this manner, a grid of electrodes can selectively turn on a cell (pixel) that contains the liquid crystal material, making it turn dark. Sometimes an electroluminescent display is placed behind the LCD screen to illuminate it.

LDT

Local Descriptor Table. This table contains the memory segment descriptors available to tasks that are permitted to use the LDT.

Leaderless Chip Carrier

LCC. A method of packaging ICs which uses contacts instead of legs to connect the chip to the circuit board. The chip has contacts on it base and rests in a socket that also has contacts and the chip is clamped in place.

LED

Light Emitting Diode. A highly-efficient semiconductor device that converts electrical energy to light.

Logical Unit

A physical or virtual device addressed through a target, it is part of a SCSI address that identifies a particular device. Each logical unit has a logical unit number (LUN) by which it is addressed.

Low Level Format

The electronic equivalent of drawing a detailed map on the hard disk drive platters. These electronic marks tell the system when to start and stop an operation.

LPT

DOS device name for parallel printers reserved by DOS. LPT1 – LPT3 can be used by DOS.

Main Memory

The memory between 0 and 1024 KB. Another term for conventional memory.

Math Coprocessor

A special-purpose processor that is optimized for math operations. It executes floating point and large integer operations in hardware up to 100 times faster than the microprocessor can.

MDA

Monochrome Display Adapter. The original video display standard for the IBM PC. Monochrome mode monitors use a resolution of 720 x 350 with each character being 9 x 14.

Media

The material used to store data. For instance, WORM media are magneto-optical disks.

Megabit

Mb. 1,048,576 or 2^{20} bits. Most often used in data flow measurement, as in megabits per second.

Mbs

Megabits per second. The number of millions of *bits* moved in one second. MBs is millions of *bytes* per second.

Megabyte

MB. A measurement of storage equal to 1024 kilobytes, 2^{20}, or 1,048,576 bytes.

Megahertz

The unit of measurement for frequencies of one million cycles per second.

Memory

A device that can store data. Also called RAM (Random Access Memory).

Memory Management Unit

The hardware (often within the microprocessor) that supports the mapping of virtual memory addresses to physical memory addresses.

Memory Test Tick Sound

This is an AMIBIOS Setup option. This option turns the ticking sound during the memory test on or off.

Memory Parity Error Checking

This is an AMIBIOS Setup option. This option enables or disables memory parity error checking for all system RAM. The default setting is enabled.

Microprocessor

The central processing unit of the system.

Modem

Modulator/Demodulator. A device that converts digital data from a computer to analog data for transmission over telephone lines and converts the analog data back to digital form on the receiving end.

Modified Frequency Modulation

MFM. The original method of organizing data on a hard disk drive (a data recording technique), which involves varying the amplitude and frequency of a signal to the hard disk drive.

Motherboard

System board. The board that contains the microprocessor, system ROM BIOS, electronic circuitry, real time clock, and other system components. Adapter cards plug into expansion slots on the motherboard.

Motor Wait Time

A floppy disk drive parameter. The amount of time that a floppy disk drive can be inactive before the drive motor is powered off.

Mouse

A small handheld device that duplicates its movements on a flat surface external to the computer on the computer screen. It essentially replaces the keyboard arrow keys.

Mouse Support Option

This is an AMIBIOS Setup option. This option enables or disables mouse support.

Multifrequency Monitor

A monitor that can display a wide variety of modes from a number of different video standards (EGA, VGA, CGA) because it can scan at different horizontal and vertical frequencies.

Multitasking

Multitasking programs execute multiple applications at the same time.

NAK

Negative Acknowledgement. A control code (15h) transmitted to a sending station or a computer by a receiving unit as a signal that the transmitted information has not arrived or is incorrect. See also: ACK.

Nanosecond

A billionth of a second. Electricity moves at about one foot per nanosecond.

Non-Cacheable Data Area

This is an AMIBIOS Setup option for power management AMIBIOS. The user can enable a predefined non-cacheable data area such that memory in this area is not effected by the system cache. The Non-Cacheable Starting Address and the Non-Cacheable Area Size are usually defined as part of this configuration.

NMI

Nonmaskable Interrupt (INT 02h). A hardware interrupt (request for service) that cannot usually be turned off. An NMI indicates a serious memory or I/O problem. An NMI bypasses and takes precedence over software interrupts and other hardware interrupts. An NMI is only issued under severe circumstances such as an I/O, memory, or power failure.

Nonvolatile memory

Another name for CMOS RAM. A memory (or RAM) device that has very low power consumption and is battery-backed for long-term storage.

Null Modem Cable

Used when connecting two similar RS232-C devices.

Numeric Processor

This is an AMIBIOS Setup option. This option specifies that a math coprocessor is configured.

N-Way Interleave Memory Access

Some motherboard designs allow an interleaved memory architecture. The RAM is divided into *N* banks (where N is a power of 2), and each data is obtained from one bank while the CPU is processing.

For example, if two-way page interleave mode is used, the RAM is divided into two banks, Bank 0 and Bank 1. Initially, data may be fetched from bank 0 and then sent to the CPU for processing. While this data is being processed, the next block of data, assuming it is available at the next sequential address, is fetched from bank 1.

Offset

An addressing method used in Intel x86 processors that defines an address as relative to the start of a memory segment.

OEM

Original Equipment Manufacturer. The company that manufactures a piece of hardware.

Operating System

Systems software that operates as a master control program and controls the execution of applications software. The kernel, or core, of the operating system must always remain in memory. The operating system primarily interfaces with the BIOS.

Paging

The transfer of program segments in and out of memory in virtual memory operations. Pages are 4 KB long in the Intel x86 architecture. Pages are swapped between memory and disk as needed in virtual memory. Paging can only be used in protected mode.

Page Address

An address in memory created by combining the process of segment translation and page translation. The microprocessor's paging feature must be enabled. Logical addresses are converted to physical addresses in two steps:

1. Segment translation, which converts a logical address (a segment selector and segment offset) to a linear address — an address that refers indirectly to a physical address.

2. After the linear address is obtained, paging hardware converts the linear address to a physical address by specifying a page table (an array of 32-bit page specifiers), a page (a 4 KB unit of contiguous addresses in physical memory within the table), and an offset within the page.

Page Frame

A physical address to which a page of virtual memory can be mapped. Page frames are 4 KB long in the Intel x86 architecture.

Paged Memory Management Unit

Accesses and manages memory used by applications or by virtual memory operating systems. A paged memory management unit often translates memory addresses and supports demand-paged virtual memory, which allows an application to think it has more memory than it actually has. This unit is built into high-end Intel x86 processors.

Page Mode Memory Access

When data is accessed from RAM, the CPU activates the RAS (Row Address Strobe) line to specify the row of memory in which the data is to be found. Then it must activate the CAS (Column Address Strobe) line to specify the memory column in which the data is to found. The combination of RAS and CAS specifies a particular RAM location on a specific RAM chip.

When using Page Mode Memory Access, it is assumed that the data is located in the same page as the initial data. This means the CPU need not specify the row (or column) for the next piece of data, and therefore, it need not set the RAS (or CAS) line again. This technique speeds RAM access.

Parallel Port

The channel used to communicate data to the printer. Bits of data can be transmitted over several lines simultaneously.

Parity

An error-checking routine in which the number of 1s (or 0s) must always be the same for each group of bits checked. The parity bit is a bit added to a unit of data that makes the sum total of the unit either odd or even. Computers and peripheral devices tend to use either even or odd parity consistently throughout their architecture.

Partition

A method of dividing a disk during formatting so that the operating system treats each division as a separate disk.

Password Check Option

This is an AMIBIOS Setup option. It can be used to prevent unauthorized system boot or Setup use. If enabled, a prompt appears every time the system is turned on or Setup is executed.

Peripheral Device

An external hardware device used for I/O operations, such as a printer, tape drive, modem, or plotter.

PGA

Pin Grid Array. A method of mounting chips on a circuit board. It is often used for chips that have a large number of pins, such as microprocessors and math coprocessors. PGA chips have pins protruding from the bottom of the chip instead of from the side (as DIPs do).

Physical Address

An address that corresponds to an actual memory location in the hardware.

Picosecond

One-trillionth of a second.

Pinout

A description of the signals from the pins of a chip or connector.

Pixel

Picture Element. The smallest addressable point on a computer screen. A small rectangular element that represents the degree of brightness assigned to that point on the screen.

Plastic Leaderless Chip Carrier

PLCC. An inexpensive type of leaderless chip carrier (LCC) for mounting chips on a circuit board. PLCCs are not physically compatible with LCCs, which are made of ceramic.

POST

Power-On Self Test. Part of the BIOS. A set of routines that test parts of the computer (including the keyboard, memory, and the disk drives) every time the computer is turned on or reset. POST also sets initial values and, if successful, passes control to the bootstrap loader, which boots the operating system. If hardware errors are found, POST either beeps or displays error messages.

Power-On Defaults

In the AMIBIOS Setup, every option has two possible default settings: Power-On and BIOS defaults.

The Power-On default settings consist of the safest set of parameters. Use them if the system is behaving erratically. They should always work but do not provide optimal system performance characteristics.

PQPF

Plastic quad flat pack. A method of packaging integrated circuits.

Printed Circuit Board

PCB. A flat plastic or fiberglass board often coated with copper or another conductor on which chips and other electrical components are mounted in preassigned, predrilled holes. The holes are connected electrically by preprinted conductive metallic pathways on the surface of the board. The metal leads from the electrical components are soldered to the conductive metal pathway to form a connector. PCBs should be held by the edges and must be protected from dirt and static electricity.

Privileged Mode

A mode of execution in protected mode in the Intel x86 architecture in which some software programs can carry out restricted operations that manipulate critical system components (memory and I/O ports).

The kernel of the operating system and device drivers are usually the only type of software that can use the instructions that only operate in privileged mode.

Processor Interrupt

An interrupt generated by the microprocessor, such as INT 00h, Divide by Zero. Also known as a logical interrupt.

PROM

Programmable Read-Only Memory. Data or code can be written to a PROM one time. A PROM cannot be reprogrammed or written to again. See also: EEPROM, EPROM.

Protected Mode

A memory address mode in Intel x86 processors. In protected mode, the processor uses all address lines. In an 80286, up to 16 MB of memory can be directly addressed. In an 80386DX or 80486, up to 4 GB of memory can be directly addressed.

The 80286 internal memory management scheme allows up to 1 GB of virtual memory to be addressed; the 80386DX and 80486 allows up to 64 terabytes of virtual memory to be addressed.

Protected mode addresses have a Selector:Offset format. Protected mode offers multitasking, virtual memory, and data security features. Unix, Xenix, and OS/2 operate in protected mode. See also: *Virtual Memory.*

RAM

Random Access Memory. Any byte of RAM can be accessed directly in a single memory cycle. Information can be read from and written to RAM. RAM is volatile, when power is turned off, it loses its memory.

RAM Wait States

This is a common option in AMIBIOS Setup. It is the number of wait states used during RAM read and write operations.

RAS

Row Address Strobe. RAM is organized in rows and columns and is accessed by signals (strobes). RAS is a signal line that clocks the row address into an internal address latch. RAS is measured in nanoseconds; the lower the value, the faster memory can be accessed.

RC

Receive Clock. Line 17 in the RS-232C specification. Used in synchronous communications to clock received data.

RD

A serial communications signal. See RXD.

Real Mode

An address mode in Intel x86 processors. In real mode, only the first 20 address lines of the processor are used, limiting access to conventional memory (addresses 0 – 1024 KB, or 00000h – FFFFFh). Virtual memory is not supported in real mode. Real mode addresses have a segment:offset format.

Refresh Bypass

A logic scheme used in American Megatrends 80486-based motherboards to circumvent the need to constantly refresh DRAM. Refresh bypass speeds access to system memory, increases system throughput, and reduces the need for memory read and write wait states.

Register

Memory registers are high-speed circuits that are part of the CPU. Registers are system addresses for storing program instructions, data, or addresses. They form the basis for assembly languages.

Resolution

A measure of the quality of the image that can be displayed. It is usually expressed as the number of pixels that can be displayed horizontally and vertically. VGA commonly uses a 640x480 resolution.

Rewritable Disk Drive

Also known as a Magneto-Optical Disk drive. A data storage device that stores large amounts of information on removable disks by a laser-based magnetic process. Up to 650 MB can be stored on an individual drive. Data on the media can be written over or erased.

RI

Ring Indicator. Line 22 in the RS-232C specification. A signal that the telephone is ringing.

RISC

Reduced Instruction Set Computing. A type of microprocessor instruction set design that focuses on rapid and efficient processing of a relatively small and simple set of instructions that can be executed in a minimum number of instruction cycles, usually only one. See also: CISC.

RLL

Run Length Limited. A self-clocking data recording technique used in hard disk drives where 26 sectors can be recorded per track instead of the normal 17 sectors per track. There are three types of advanced RLL (1,7), (2,6), and (3,9).

(3,9) Advanced Run Length Limited permits up to 34 sectors per track. RLL hard disk drives can be used with MFM hard disk drive controllers. See also: ESDI, SCSI, and MFM.

ROM

Read-Only Memory. Devices used to store code and data that cannot be changed. The BIOS is stored in ROM devices. ROM devices are usually significantly slower than RAM devices, so code and data in ROM is often copied to RAM to improve performance.

ROM BIOS

See *BIOS*.

ROM Shadow

This is an option in AMIBIOS Setup. ROM shadow is a technique in which BIOS code is copied from slower ROM to faster RAM (see shadowing). The BIOS is then executed from the RAM.

On some AMIBIOS Setup screens, there are up to 13 shadowing options for each 16K address segment from C0000h through EFFFFh, and an additional shadowing option for the 64K segment at F0000h. If an option is enabled, the code that resides in that 16 KB segment of ROM is copied to RAM.

RS-232C

An industry standard for serial communications connections. It defines the specific lines and signal characteristics used by serial communications controllers to standardize the transmission of serial data between devices. There are two interfaces: DTE (Data Terminal Equipment), used by computers, and DCE (Data Communications Equipment), used by modems.

RTC

Real Time Clock. The Motorola MC146818A or compatible clock in ISA and EISA systems.

RTS

Request To Send. RTS uses line 4 in the RS-232C specification. RTS is sent by the computer to request permission to transfer data.

RXD

Receive Data. A line used to carry received data from one device to another. RXD uses line 3 in the RS-232C specification.

SAA

Systems Application Architecture. An IBM standard for the appearance and operation of applications software that gives software written for all IBM computers (mainframe, mini-, workstation, and PC) a similar look and feel.

SAA defines how an application interfaces with the end user, the operating system, and other computers and devices.

Schottky Diode

Also called a Schottky barrier diode or hot carrier diode. A type of device that passes current in one direction only. It is formed by the contact between a layer of semiconducting material and a layer of metal. Schottky diodes are characterized by extremely fast switching speeds.

SCSI

Small Computer System Interface. An industry standard described by the X3T9.2 ANSI committee that defines a standard high-speed parallel interface specification for connecting computers to peripheral devices (hard disk drives and printers) or to other computers and LANs. A 50-pin connector is used in SCSI devices. Up to seven devices, not including the computer, can be attached to the SCSI bus via a single SCSI connection. Parallel data can be transferred at rates up to 32 Mbs between SCSI devices. See also: ESDI.

SCSI ASC

SCSI additional sense code. Byte 12 of the SCSI extended sense information. Provides a standardized description of the condition described by the sense information.

SCSI Device

A device attached to an SCSI bus cable. The device can be an initiator, target, or both. It can be a peripheral device, a host device, or a device that combines both roles.

Sector

Equally spaced divisions of data within each track on a hard disk or floppy disk drive. A sector is the smallest unit of storage on a disk drive and is set at 512 bytes in DOS.

Segment

Part of the Intel x86 processor address structure. A unit of contiguous, position-independent address space. In real mode, segments are 64 KB blocks of memory, referenced by one byte. In protected mode, programs can allocate segments of any size. The 80286 computes effective addresses by adding a 16-bit offset an a 16-bit segment. The 80386 and 80486 compute effective addresses by adding a 32-bit segment and a 32-bit offset.

Selector

A 16-bit part of an Intel x86 protected mode virtual address. It corresponds to the segment component in the real mode segment:offset address. A selector points to the information which defines a segment, such as a descriptor table and a specific descriptor within that table. A selector is a value contained in a segment register (CS, DS, SS, or ES) when in protected mode. This value determines the segment that is currently being used (with CS, the segment used for executing code).

Serial Communications

The transmission of information between computers or between computers and peripheral devices one bit at a time over a single line. They can be synchronous (controlled by a clock or timing device) or asynchronous (managed by the exchange of signals that control the information flow). Both sender and receiver must use the same data transfer rate, parity, and type of control information.

Shadowing

Copying code from slower ROM to RAM to improve performance. ROM devices usually operate at 150 – 180 ns; RAM usually operates at 70 – 100 ns. The system BIOS and video BIOS are often shadowed to RAM.

Shadow RAM (ROM Shadow)

The area between 640 KB and 1 MB in the memory map (for ISA systems) is reserved for ROM BIOS, including the system BIOS, video BIOS, and adaptor ROM BIOSes. All AMIBIOSes use shadowing, which increases BIOS performance. The addresses that are usually shadowed are:

- ISA system BIOS F0000 – FFFFFh,
- EISA system BIOS E0000 – FFFFFh, and
- Video BIOS C0000 – C7FFFh

These addresses are usually write-protected after shadowing has completed.

SIMM

Single-inline Memory Module. A method of organizing surface mount RAM chips on a small circuit board so that the SIMMs can be mounted vertically or at an angle to save space. See also: DIP.

SIP

Single Inline Package. A method of organizing and type of housing for an electronic component in which all leads (connectors) protrude from one side of the package. See also: DIP.

Single-Tasking

Single-tasking operating systems, such as DOS, can only execute one program module or routine at a time. Information input to a code module or routine must be processed completely before information can be output to another module.

Software Interrupt

A program-generated interrupt (request for service) that halts current processing. A software interrupt requests a service provided by an interrupt handler (a separate set of instructions that performs a specific function required by the task at hand).

SRAM

Static RAM. A type of semiconductor memory based on flip-flop logic circuits. SRAM retains information as long as the chip is powered. An SRAM chip can store only about ¼ the amount of information as a DRAM chip of the same degree of complexity, but SRAM does not require refreshing and is usually much faster (and more expensive) than DRAM. Some SRAM chips operate at 12 ns. SRAM is usually used in cache memory. See also: CMOS RAM and DRAM.

Step Rate

A floppy disk parameter. The amount of time needed for a drive head to move from one track to another.

ST506 Interface

The hardware signal specification developed by Seagate Technologies for the original XT hard disk drive controllers and connectors. The ST506 and ST412 version of this interface is now a de facto standard for transfer of hard disk data. The maximum data transfer rate is 5 Mbs. See also: MFM, RLL, ESDI, and SCSI.

Surface Mount Technology

A method of mounting chips on circuit boards in which the chips are fixed directly to the surface of the board instead of being soldered in predrilled holes.

Synchronous Data Transfer (SCSI)

A method of data transfer on the SCSI bus involving clocking data onto the bus via a fixed-length, fixed-frequency strobe pulse. The Acknowledge signal can be delayed several clock periods from the data request. Synchronous data transfer can be used on the SCSI bus only for data transmission.

System Boot Up CPU Speed

This is an AMIBIOS Setup option. This option sets the speed at which the system boots. The speed can be Low or High. High is the default.

System Boot Up Num Lock

This is an option in AMIBIOS Setup. This option turns the Num Lock key on or off when the system is powered on. If off, both sets of arrow keys can be used.

System Boot Up Sequence

This is an AMIBIOS Setup option. The BIOS normally attempts to boot from floppy drive A: (if present). If unsuccessful, the BIOS attempts to boot from hard disk C: This sequence can be switched. If this option is set to *C:, A:,* the system attempts to boot from the hard drive C:, and then A:. If set to *A:, C:,* the sequence is reversed.

Tape Drive

A data storage device that records data on magnetic media.

Target

A SCSI device that performs an operation requested by an initiator. The target can be a peripheral device performing a service for an initiator or a host adapter performing a device service for an initiator.

Target Identifier

Part of the SCSI address. A SCSI address consists of the Target ID and the LUN. The Target ID represents the device controller. A LUN identifies the specific device attached to the controller. For most SCSI devices, the controller is embedded and the LUN is set at 0.

Task

In the microprocessor, a task is the execution of a single process or set of instructions that perform a particular function. In an operating system, a task is a single job or part of a job executed as a single unit.

TC

Transmit Clock. Line 15 in the RS-232C specification. Used in synchronous communications to synchronize transmitted data.

TD

Transmit data. See TXD.

Terabyte

1,099,511,627,776 bytes, as in 1,000 Gigabytes or 2^{40}.

Termination

The physical location of the first and last device on the SCSI bus. Devices at these ends must have terminators installed; devices located in between must have SCSI bus terminators removed.

Terminators

Special resistor packs used on the SCSI bus to ensure the electrical integrity of the bus signals.

Text Mode

A display mode in which the video display adapter converts ASCII character data directly into display information.

Timeout

When the interval of predetermined time for a certain process or interrupt expires, the process times out, and is usually terminated.

Track

A segment of a floppy or hard disk drive parallel to the edge of the media that is accessed by the drive read/write heads without moving the head. A track is a magnetic ring slightly wider than the read/write head in a disk drive. A disk or disk platter may have 40 to more than 1024 concentric tracks on a surface. Tracks are composed of sectors. Tracks on sequential devices, such as magnetic tape, are parallel to the edge of the medium. See also: sector.

Track Skew Factor

The track skew factor is equivalent to the number of sectors between the last sector in the last track and the first sector in the next track plus 1. If a sequential file is stored over many tracks, the controller reads the first sector in the next track immediately after it reads the last sector in the previous track.

The distance between the last sector in the previous track and the first sector in the next track is the determining factor in how long it takes the read/write head to move to the first sector in the next track. This is the track skew factor.

Most of the time, the track skew factor is less than or equal to the interleave factor. Like the interleave factor, the track skew factor is either 1:1, 2:1, 3:1, and so on, and organizes sectors as follows:

Track Skew Factor	Track Number	Sequence
1:1	Track *n*	1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17
	Track *n+1*	1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17
2:1	Track *n*	1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17
	Track *n+1*	17, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16
3:1	Track *n*	1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17
	Track *n+1*	16, 17, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15

TSR

Terminate-and-stay-resident. A program designed to remain in memory which is activated by a keychord.

TTL

Transistor-To-Transistor Logic. A bipolar logic circuit design that uses transistors connected to each other directly or through resistors. TTL offers high speed and good noise immunity. Many TTL gates can be fabricated in a single integrated circuit.

Turbo Switch Function

This is an option in AMIBIOS Setup. This option enables or disables the turbo (processor speed switching) switch on the computer.

TXD

Transmit Data. A serial communications signal line (pin 2 in RS-232C connectors) that carries transmitted data from one device to another.

Typematic Rate Programming, Rate, and Delay

This is a set of three AMIBIOS Setup options. The *Typematic Rate Programming* option enables or disables the following two options.

The *Typematic Rate Delay* (250, 500, 750, or 1,000 milliseconds) option and the *Typematic Rate* (6, 8, 10, 12, 14, 16 24, or 30 characters per second) option control the speed at which a keystroke is repeated.

UART

Universal Asynchronous Receiver-Transmitter. A single IC that contains both the receiving and transmitting circuitry for asynchronous serial communications. It can also do serial – parallel conversions, and insert and check for synchronizing control bits in serial data. Computers equipped with UARTs can communicate over a simple wire connection.

Unix

A multitasking, multiuser operating system. Several versions of Unix are available for ISA and EISA systems.

Upper Memory

The memory between 640K and 1024K. Also called reserved memory or high DOS memory. Used for the system BIOS, video BIOS, video memory, adaptor ROM, additional device drivers, and TSRs. This area is divided into upper memory blocks (UMBs).

USRT

Universal Synchronous Receiver-Transmitter. A single IC that contains both the receiving and transmitting circuitry for synchronous serial communications. Computers equipped with USRTs can communicate over two communication channels between sender and receiver.

VGA

Video Graphics Array. A video standard that uses it own BIOS and includes many additional video modes and capabilities over and above the modes provided by earlier video standards. VGA also duplicates all modes from earlier video standards. The VGA signal is analog and can display up to 262,144 colors in 64x480 resolution. Super VGA video cards can display resolutions up to 1024x768.

Video Mode

A video mode is a commonly-accepted standard that incorporates the characteristics and parameters for presenting information on a video screen. Each video standard (MDA, CGA, EGA, VGA, XGA, and Super VGA) defines new video modes. All video modes are used either for text or graphics. Characteristics such as the number of colors in a palette, the number of colors that can be displayed at any one time, the screen resolution (640 pixels by 480 pixels), and the number of columns (40, 80, or 120 in text mode) define the different video modes.

Video RAM

VRAM. A special type of DRAM used in high-speed video applications. In normal DRAM, both processor and video circuitry must access RAM by sharing the same control pins on the RAM chips. Video RAM provides separate pins for the microprocessor and the video circuitry. The processor can access video RAM in the same way it accesses system RAM (in parallel). But the video circuitry has a special method of accessing video RAM serially via a separate internal serial-shift register.

Virtual 8086 Mode

In the 80386 microprocessor, a method for emulating an 8088 or 8086 processor. Several 8086 programs can run in protected mode as a single task.

Virtual Address

The address that a program uses to reference memory. The Memory Management Unit translates this address to a physical address before memory is accessed.

Virtual Memory

A technique that allows a program to see the system as providing a large uniform primary memory which in reality is smaller, fragmented, and partially simulated by hard disk drive storage. Paging and segmentation are two common implementations of virtual memory.

Wait for F1 If Any Error

An AMIBIOS Setup option. BIOS error messages are followed by

```
Press F1 to continue.
```

If this option is disabled, the above message does not appear.

Wait State

A phase during program execution when the processor must wait one or more clock cycles until the memory is available to respond to the processor's request. Caused by the fact that the processor operates much faster than the memory. A wait state is a pause that lets the processor synchronize timing with slower memory.

Weitek Processor

This is an AMIBIOS Setup option. This option specifies that a Weitek (WTL3167 or WTL4167) math coprocessor is configured.

WORM Drive

Write Once Read Multiple Times. An optical data storage device that stores information permanently. Unlike other disks, data saved to the disk cannot be written over or erased. If you try to replace data with new data, both remain on the disk, consuming limited disk space. Once the disk is full, you can only read from it.

Write Precompensation

A technique in which the timing of the head current on a floppy or hard disk drive is varied to keep the write signal constant. The size of a sector gets progressively smaller as the track diameter diminishes, but each sector must still hold 512 bytes. Write precompensation circuitry on the hard disk compensates for the physical difference in sector size by boosting the write current for sectors on inner tracks.

Write precompensation is also a hard disk drive parameter that specifies the cylinder number where the write precompensation circuitry begins operating.

Xenix

A multitasking, multiuser operating system that is quite similar to Unix. Several versions of Xenix are available for ISA and EISA computers.

XGA

Extended Graphics Array. A new IBM video standard which permits up to 16-color displays from a palette of 262,000 colors in 1024 x 768 resolution.

XMS

Extended Memory Specification. See *EMS*.

Acronyms and Abbreviations

ACK	Acknowledge (Keyboard command and serial communication signal)
ACR	Hard Disk Adapter Controller Register
ALE	Address Latch Enable — x86 processor signal
ANSI	American National Standards Institute
ARLL	Advanced Run Length Limited (method of encoding hard disk data)
ARB	Access Rights Byte (part of i286, i386, and i486 instruction)
ASCII	American Standard Code for Information Interchange
ASIC	Application-Specific Integrated Circuit
ASR	Hard Disk Adapter Status Register
AT	Advanced Technology
b	Binary
BAT	Basic Assurance Test (keyboard diagnostic)
BCD	Binary-coded decimal
BIOS	Basic Input Output System
bps	Bits per second
CAS	Column Address Strobe (RAM signal)
CCB	Command Control Block (disk drive data structure)
CGA	Color Graphics Adapter
CLK	Clock signal (line) on a microprocessor
CMOS	Complementary Metal Oxide Semiconductor
COM1	Serial Communications Port 1
COM2	Serial Communications Port 2
COM3	Serial Communications Port 3
COM4	Serial Communications Port 4
CRC	Cyclic Redundancy Check
CSB	Command Specify Block (disk drive data structure)
CTS	Clear To Send (serial communication signal)
DAC	Digital to analog converter
DCC	Display Combination Code
DIN	Deutsche Industrie Normal (industry standards organization)
DIP	Dual Inline Package
DMA	Direct Memory Access
DPL	Descriptor Privilege Level — part of x86 instructions
DSR	Data Set Ready (serial communication signal)
DSR	Device Service Routine
DTR	Data Terminal Ready (serial communication signal)
EA	Effective Address
EBCDIC	Extended Binary-Coded Decimal Interchange
ECC	Error Checking and Correction
EGA	Enhanced Graphics Adapter
EIA	Electronic Industries Association
EISA	Extended Industry Standard Architecture
EOI	End of Interrupt
EPL	Extended Privilege Level
EEPROM	Electrically Erasable Programmable Read-Only Memory
EPROM	Erasable Programmable Read-Only Memory

cont'd

ESDI	Enhanced Small Device Interface
ETB	End of Transmission Block
FCB	Format Control Block (data structure for disk information)
GB	Gigabytes
GDT	Global Descriptor Table
GDTR	Global Descriptor Table Register
h	Hexadecimal
ICW	Interrupt Control Word
IDE	Intelligent Device Electronics (method of accessing hard disk drives)
IDT	Interrupt Descriptor Table
INT	Software interrupt
I/O	Input/Output
IRQ	Interrupt Request Line
IRET	Return from an Interrupt
ISA	Industry Standard Architecture (AT-compatible)
ISR	Interrupt Service Return
ISR	Interrupt Status Register
KB	Kilobytes (1,024 bytes)
Kb	Kilobits (1,024 bits)
Kbs	Kilobits per second
LDT	Local Descriptor Table
LDTR	Local Descriptor Table Register
LED	Light-Emitting Diode
LID	Logical ID
LPT1	Parallel Printer Port 1, DOS reserved word
LPT2	Parallel Printer Port 2, DOS reserved word
LPT3	Parallel Printer Port 3, DOS reserved word
LSB	Least Significant Bit (or Byte)
LSI	Large Scale Integration
LUN	Logical Unit Number (SCSI device identifier)
MB	Megabytes (1, 048,576 or 2^{20} bytes)
MBs	Megabytes per second
Mb	Megabits (1, 048,576 or 2^{20} bits)
Mbs	Megabits per second
MCGA	Modified Color Graphics Adapter (video standard used only in low-end PS/2® models)
MDA	Monochrome Display Adapter
MFM	Modified Frequency Modulation (a method of encoding hard disk data)
MGA	An obsolete Graphics Adapter (PCJr only)
MHz	Megahertz
MMU	Memory Management Unit
MSB	Most Significant Byte (or Bit)
MTBF	Mean Time Between Failure
MTTR	Mean Time To Repair
NCB	Network Control Block (data structure for network disk drives)

NMI	Nonmaskable Interrupt
OCW	Operation Control Word (used in Programmable Interrupt Controller)
OEM	Original Equipment Manufacturer
OS/2	Operating System/2
PC	Personal Computer
PCB	Printed Circuit Board
PCLK	Peripheral Clock
PEL	Picture Element (pixel)
PGA	Professional Graphics Array — video standard that nobody uses
PIC	Programmable Interrupt Controller
PIO	Programmed Input/Output
PIT	Programmable Interval Timer
POR	Power-On Reset
POST	Power-On Self Test
RAM	Random Access Memory
RAS	Row Address Strobe (RAM signal)
RGB	Red-Green-Blue
RI	Ring Indicator (Serial communication signal)
RLL	Run Length Limited (method of encoding hard disk data)
ROM	Read-Only Memory
RS232C	Industry standard serial controller interface
RTC	Real Time Clock
RTS	Request To Send (serial communication signal)
SCSI	Small Computer Systems Interface
SDLC	Synchronous Data Link Control
SIMM	Single Inline Memory Module
SIP	Single Inline Package
SMD	Surface Mount Device
SMT	Surface Mount Technology
SSB	Sense Summary Block (data structure for hard disks)
TB	Terabytes
TID	Target ID (SCSI device identifier)
TSS	Task State Segment (part of x86 instructions)
TTL	Transistor-To-Transistor Logic
UART	Universal Asynchronous Receiver/Transmitter
VGA	Video Graphics Array
VLSI	Very Large Scale Integration
XMS	Extended Memory Specification
XT	Extended Technology

Additional Reading

The following books and articles are recommended for additional information about BIOS and related subjects:

Boston, Michael and Narushoff, Paul. *System BIOS for IBM PC/XT/AT Computers and Compatibles*, Phoenix Technical Reference Series. Reading, MA: Addison-Wesley Publishing Company. 1989.

Brown, Ralf, and Kyle, Jim, *PC Interrupts*. Reading, MA: Addison-Wesley Publishing Company. 1991.

Cohen, Howard N. and John Hanel. "A Timing-Independent BIOS." *Byte IBM Special Edition*, Fall 1987, pp. 219–222.

Duncan, Ray. *Advanced MS-DOS Programming*. Redmond, WA: Microsoft Press, 1989.

Duncan, Ray. *IBM ROM BIOS*. Redmond, WA: Microsoft Press, 1988.

Extended Industry Standard Architecture Technical Reference. New York, NY: BCPR Services. 1989.

Glass, L. Brett. "Inside EISA." *Byte*, November 1989, pp 417–425.

Intel Corporation. *i486 Programmer's Reference Manual*. Santa Clara, CA: Intel Corporation. 1989.

Intel Corporation, Microsoft Corporation. *Advanced Power Management (APM) Interface Specification*. 1993

Intel Corporation. *ExCa Card Services Specification*. Hillsboro, OR: 1993

Intel Corporation. *PCI BIOS Specification*. Hillsboro, OR: 1993

Kliewer, Bradley Dyck. *EGA/VGA Programmer's Reference Guide.* New York, NY, McGraw-Hill Book Company, 1988.

Norton, Peter and Wilton, Richard. *The New Peter Norton Programmer's Guide to the IBM PC and PS/2.* Redmond WA: Microsoft Press, 1988.

PCMCIA. *PC Card Standard.* Personal Computer Memory Card International Association, 1992

PCMCIA. *Socket Services Interface Specification.* Personal Computer Memory Card International Association, 1992.

Schulman, Andrew, et al. *Undocumented DOS.* Reading, MA: Addison-Wesley Publishing Company, 1991.

SCSI Specification X3T9.2. ANSI Committee, 1991.

Shiell, Jon. "IBM PC Family BIOS Comparison." *Byte IBM Special Edition,* Fall 1987, pp. 173–180.

Small Computer System Interface (SCSI) Specification. ANSI X3.131-1991.

White, George. "A Bus Tour." *Byte,* September 1989, pp. 296–302.

Wilton, Richard. *Programmer's Guide to PC and PS/2 Video Systems.* Redmond, WA: Microsoft Press, 1987.

Index

American Megatrends, Inc.

American Megatrends, Inc. was founded by Subramonian Shankar and Pat Sarma in 1985. Originally a design and consulting firm for many of the leading PC manufacturers, American Megatrends introduced the AMIBIOS in 1986, which immediately gained industry-wide acceptance.

American Megatrends also developed the first 80386-based motherboard with cache memory in 1986. This design was initially sold as a complete production kit and quickly became an industry standard. American Megatrends began manufacturing motherboards under its own name in 1987 and introduced the first 33 MHz 80386-based motherboard in 1988. American Megatrends has since developed state-of-the-art 80486-based ISA and EISA motherboards.

In 1990, AMIDiag, a hardware diagnostic tool, was awarded Editor's Choice by *PC Magazine*. American Megatrends has been included in *Atlanta Magazine's* Fast Tech 50 list since 1989. American Megatrends ranked second on *Inc Magazine's* list of the 500 fastest-growing privately-owned companies in 1991.

American Megatrends Motherboards

- 80486 Enterprise-IV EISA VLB Motherboard
- 80486 Super Voyager PCI Motherboard
- 80486 Super Voyager VLB-II Motherboard
- Omni Business Partner Motherboard

American Megatrends Peripheral Products

- AudioMaxx Business Audio Card
- MaxxPlus DSP Daughtercard
- Fast Disk EISA Caching SCSI Host Adapter
- Fast Disk NC EISA SCSI Host Adapter
- Fast Disk VLB EISA Caching SCSI Host Adapter
- Fast View VLB Graphics Accelerator Card
- Fast LAN-SCSI VLB Card
- EISA LAN Card
- Platinum Access PCMCIA SCSI Card
- Platinum Access PCMCIA LAN Card

American Megatrends Software Products

- American Megatrends produces a wide variety of ISA, power management, and EISA BIOS products.
- AMIDiag is a system test, hardware diagnostic, and system information tool.

Call American Megatrends Sales at 800-828-9264 for additional product information.

American Megatrends, Inc.
6145-F Northbelt Parkway
Norcross, GA 30071
404-263-8181
800-828-9264

AMIBIOS Technical Reference
MAN-PR-BIOS-TR