

BIOS Information

The basic input/output system (BIOS) program controls all internal and external I/O devices. BIOS routines enable the assembly language programmer to do block (disk) I/O operations or character-level I/O operations without concern for device address or characteristics. Services such as time, date, and memory size determination are provided by the BIOS.

The BIOS is accessed through program interrupts of the microprocessor. Each BIOS entry point is available through its own interrupt. For example, to determine the amount of base RAM available in the computer, INT 12h invokes the BIOS routine for determining memory size and returning the value to the caller.

All parameters passed to and from the BIOS routines go through the CPU registers. The prolog of each BIOS function described in this chapter indicates the registers used on the call and return. For the memory size example, no parameters are passed in. The memory size, in 1 KB increments, is returned in the AX register.

In general, the BIOS routines save all registers except for AX and the flags. Other registers are modified on return only if they are returning a value to the caller.

4.1 Non-maskable Interrupt (INT 02h)

The system board and I/O channel parity error signals are connected to the NMI pin of the CPU. If a memory parity error occurs, the hardware invokes this routine. The routine ensures that the reason for the interrupt is a memory parity error by examining a register. If no memory parity error occurred, the routine returns to the interrupt operation. Other devices, such as the EGA, in CGA emulation mode, may use the NMI. The parity checking aspect is never impaired.

Input: None

Output: Error message and system is halted

4.2 Print Screen (INT 05h)

This routine is invoked to print the screen to printer 1. The cursor position at the time this routine is invoked is saved and restored upon completion. If the print screen key is pressed when this routine is executing, it is ignored.

Input: None

Output: Location 50:00 = 0 Normal termination or print screen not called
 = 1 Print screen is in progress; ignore request
 = FF Error during print screen

4.3 System Timer Hardware Interrupt (INT 08h)

This routine handles interrupts from the interval timer and maintains a count of interrupts since power-on for use by the system clock. It also decrements motor control count and turns off the motor when it expires. At every interrupt, it invokes a user routine through interrupt 1Ch.

Input: None

Output: None

4.4 Keyboard Hardware Interrupt (INT 09h)

This routine supports an interface between system and keyboard (I/O device). Every time you strike or release a key, an interrupt occurs and the routine is invoked to place ASCII code and scan code in keyboard buffer or set keyboard flag. It also checks for Ctrl-Alt-+ key combination to toggle speed between the 16-MHz modes (depends on the default setting) and other modes.

Input: None

Output: None

4.5 Diskette Hardware Interrupt (INT 0Eh)

This interrupt diskette drive hardware interrupt IRQ 6. When a hardware action of the disk is successful, it generates an IRQ 6. Service routine of IRQ 6 sets bit 7 of 40:3E.

Input: None

Output: None

4.6 Video I/O (INT 10h)

This routine handles interrupts from the video interface. Detailed descriptions are included with each function. All video BIOS routines are accessed by executing the software interrupt 10h. The heart of the routine is its call to the ROM BIOS to configure the video hardware for a particular video mode. The video BIOS is more than an interface to program the VGA register hardware. It also maintains a complete set of data areas that contain important video-state and functionality information. The following table summarizes the video functions of INT 10h. Table 4-1 lists the video functions.

Table 4-1 INT 10h - Video Functions

Address	Function
(AH) = 00h	Set mode
(AH) = 01h	Set cursor type
(AH) = 02h	Set cursor position
(AH) = 03h	Read cursor position
(AH) = 04h	Read light pen position
(AH) = 05h	Select active display page
(AH) = 05h	Scroll active page up
(AH) = 07h	Scroll active page down
(AH) = 08h	Read attribute/character at current cursor position
(AH) = 09h	Write attribute/character at current cursor position
(AH) = 0Ah	Write character at current cursor position
(AH) = 0Bh	Set color palette
(AH) = 0Ch	Write dot
(AH) = 0Dh	Read dot
(AH) = 0Eh	Write teletype to active page
(AH) = 0Fh	Read current video state
(AH) = 10h	Set palette registers
(AH) = 11h	Character generator
(AH) = 12h	Alternate select
(AH) = 13h	Write string
(AH) = 14h~19h	Reserved
(AH) = 1Ah	Read/write display combination code
(AH) = 1Bh	Return functionality/state information
(AH) = 1Ch	Save/restore video state
(AH) = 1Dh~FFh	Reserved

4.6.1 VGA BIOS Test Functions

Mode Support

The VGA BIOS supports video modes 0~7 and 0Dh~13h. The mode-support test uses BIOS function 0 (select video mode) to set each of the 15 video modes in turn. Then each mode is tested by using BIOS function 0Fh (get video status) to check the number of text columns expected in that mode and to see that the mode selected is in fact the one reported by the BIOS. Text is displayed in each mode using BIOS function 13h (display character string) to help confirm the correct operation of each mode.

Cursor Operation

The video BIOS has several functions for cursor operations - both moving the text cursor and specifying its size. The test for this operation positions the cursor sizes using BIOS function 2 (set cursor location) and selects various cursor sizes using BIOS function 1 (set cursor size). The results of these tests are confirmed by using BIOS function 3 (get cursor state) to check that the reported state is the same as that which was set.

Lack of Light Pen Support

Unlike the EGA, the VGA interface does not have light pen support. BIOS function 4 (return light pen position) is used to check that the VGA correctly reports that no light pen is installed.

Multiple-Page Support

The BIOS can write text and graphics to multiple video memory pages when the adapter has sufficient memory for multiple-page support. To test this capability, BIOS function 5 (select video page) displays each of many pages in several different video modes; BIOS function 13h (display character string) places text on each of the nondisplayed video pages before those pages are brought to the screen.

Screen Scrolling

BIOS function 6 (scroll up) and 7 (scroll down) allow rectangular regions of text to be moved on the display and to be erased completely. The screen-scrolling test uses BIOS function 13h (display character string) and function 8 (read character/attribute) to put text on the screen and read it back after it has been scrolled.

Text I/O

This test covers BIOS functions 8 (read character/attribute), 9 (write character/attribute), 0Ah (write character/attribute), 0Ah (write character at cursor), 0Eh (write TTY), and 13h (display character string) to examine all BIOS text I/O. The text I/O test is performed in several video modes using function 0 (select video mode) to choose among modes.

Overscan Control

The overscan color for 16-color, 4-color, and 2-color text and graphics modes can be selected through the overscan control function (0Bh). This function also selects between the two 4-color palettes available in 4-color graphics modes. This feature is exercised in the palette-loading test.

Graphics I/O

To check the graphics I/O for graphics modes supported by the VGA, the user must evaluate BIOS functions 0Ch (write pixel) and 0Dh (read pixel). Every pixel on the screen is drawn and every possible pixel value available in the current video mode is used; then each pixel is read back to ensure that the correct value was written.

Mode Inquiry

The mode inquiry function 0Fh (get video status) gives the current video mode, active video page, and number of text columns on the display. The test for this function validates the values returned by the function, including that of the active video page when multiple pages of display memory are in use.

Palette Register Control

BIOS function 10h (set palette registers) contains 16 subfunctions to support a wide variety of palette-register control operations. Individual palette registers, digital-analog converter (DAC) registers, video-DAC-mask registers, and the display overscan color can be programmed or read either individually or in blocks.

Character Generator Support

BIOS function 11h (character-generator support) also incorporates a large set of subfunctions - 22 of them. These operations control the VGA ROM and RAM-loadable character sets stored in the VGA ROM (8 x 8, 8 x 14, and 8 x 16 pixels) can be selected as the active character font for the display, or a user-defined character set can be loaded from system memory.

In addition to font selection, function 11h's subfunctions have a set of inquiry operations that provide information about the VGA ROM character-set address and auxiliary interrupt vectors used.

Video Configuration

BIOS function 12h (alternate select) supports nine subfunctions that return various bits of information about the current video-system configuration. A number of the video system operations parameters can be selected through this function, including display switching, video-refresh control, CPU access to display memory, and gray-scale summing (used for mapping color to monochrome screens). The video-configuration test exercises BIOS functions 1Ah and 1Bh. Function 1Ah provides inquiry and selection between a primary and a secondary video display; function 1Bh fills out a 64-byte buffer with a complete set of functionality and configuration information about display hardware and current mode settings.

Video-State Save/Restore

This BIOS function (1Ch), added to help support multitasking operating systems, allows the complete control and hardware state of the VGA to be either read or programmed in a single operation. By combining this function with a read and restore of video-display memory, a complete screen state switch can be accomplished.

4.6.2 VGA BIOS Data Areas

Standard Data Area

This block of information is stored at a fixed location in the video system low memory. In segment 40h, offsets 49h through 0ABh hold a set of system parameters and pointers to additional data areas. The test of the standard data area exercises several VGA BIOS functions and checks that the data area is updated properly after each test is completed.

Primary-State Save Area

This auxiliary data buffer is used by both the EGA and the VGA. A double-word pointer in the standard data area points to it, and it contains a table of address pointers to parameter- and character-set definition tables. The test of the primary-state save area checks the contents of this buffer across several BIOS operations.

Secondary-State Save Area

Unique to the VGA, this area is pointed to by a double-word pointer in the primary-state save area. This buffer holds pointers to additional information new in the VGA, including user-palette tables and display combination tables.

Video Parameter Table

This table holds the hardware CRT controller (CRTC) and other register values used in programming each of the video modes supported on the VGA. The video parameter table test examines each of the parameter tables to ensure that the video modes supported by the system are exactly the same as those supported by IBM VGA. Because video monitors are highly dependent on the synchronization signals sent by the CRTC, any variation in those signals could cause a monitor, designed to be compatible for IBM VGA, to fail to work properly.

Parameter-Save Buffer

Another auxiliary BIOS data table, this buffer holds the current set of register values programmed into the VGA graphics controller palette and overscan registers.

Alternate-Text Character Set

The VGA BIOS supports an alternate-text character-set definition area, which allows the definition of alternate character sets on a mode-by-mode basis. When a text video mode is selected that has an alternate character set defined for it, this table is used to load that character set automatically.

Alternate-Graphics Character Set

This area is a buffer that functions just as the text-character definition table. It is used for graphics rather than text modes.

Display Combination Table

The VGA allows BIOS support for more than one video adapter in the system by using a display-adapter combination table. The table describes the list of legal display-adapter combinations available in the BIOS.

User-Palette Table

This data area has a complete set of user overrides for combinations of attribute-controller and video-DAC registers. When this table is filled out and a new BIOS video mode is selected, the alternate user definitions for these registers are used instead of the ROM BIOS definitions.

4.6.3 Set Display Mode

This function supports variable CRT modes for the system programmer to set. It programs the CRT controller with some parameters to reach the desired condition. It also loads character font to character generator as text mode set, or set up interrupt vector as graphics set.

Input: AH = 0h
 AL Contains mode value

Text Mode:

AL = 00h	40 x 25 B&W
AL = 01h	40 x 25 Color
AL = 02h	80 x 25 B&W
AL = 03h	80 x 25 Color (power on default)
AL = 07h	80 x 25 B&W Card

Graphics Mode:

AL = 04h	320 x 200 color
AL = 05h	320 x 200 B&W
AL = 06h	640 x 200 B&W
AL = 11h	640 x 480 color
AL = 12h	640 x 480 color
AL = 13h	320 x 200 color

Enhanced Features:

Text Mode:

AL=18h 132x25 color
AL=19h 132x50 color

Graphics Mode:

AL=23h 800x600 16/256k colors

Output: None

4.6.4 Set Cursor Type

This function programs the CRTC "cursor start/end" register to set the desired cursor type. You can suppress the cursor or set different cursor sizes on screen.

Input:	AH = 01h	Set cursor type
	CH (bits 4~0)	Start line for cursor
	CL (bits 4~0)	End line for cursor

Output: None

4.6.5 Set Cursor Position

This function programs the CRTC cursor position register to display the cursor at the desired location. When the specified page number differs from the active display page, there is no visible reaction.

Input:	AH = 02h	
	DH,DL	Row, Column (0,0 is upper left side)
	BH	Page number (must be 0 for graphics modes)

Output: None

4.6.6 Read Cursor Position

This function indicates the current cursor position on the identified page.

Input:	AH = 03h BH	Page number (must be 0 for graphics modes)
Output:	DH, DL CH, CL	Row, Column of current cursor position Cursor mode currently set

4.6.7 Read Light Pen Position

This function returns the current light pen position.

Input:	AH = 04h	
Output:	AH = 00h	Light pen not supported

4.6.8 Select Active Display Page (Valid only for Alpha modes)

This function selects the active display page.

Input:	AH = 05h AL = New page number (zero-based)
Output:	None

4.6.9 Scroll Window Up

This function scrolls the given area up only in the active page. This is done by moving the display buffer of the map address onto the screen.

Input:	AH = 06h	
	AL	Number of lines. Input lines blanked at bottom of window
	AL = 0	Means blank entire window
	CH,CL	Row, Column of upper left corner of scroll
	DH,DL	Row, Column of lower right corner of scroll
	BH	Attribute to be used on blank line
Output:	None	

4.6.10 Scroll Window Down

This function scrolls the given area down only in the active page. This is done by moving the display buffer of the map address onto the screen.

Input:	AH = 07h	
	AL	Number of lines. Input lines blanked at top of window
	AL = 0	Means to blank the entire window
	CH,CL	Row, Column of upper left corner of scroll
	DH,DL	Row, Column of lower right corner of scroll
	BH	Attribute to be used on blank line
Output:	None	

4.6.11 Read Attribute/Character at Current Cursor Position

This function calculates the map address of the cursor position, then gets the attribute and character from the buffer. In graphics mode, it ignores the display page.

Input:	AH = 08h	
	DH	Display page (valid for alpha modes only)
Output:	AL	Character read
	AH	Attribute of character read (alpha modes only)

4.6.12 Write Attribute/Character at Current Cursor Position

This function calculates the map address of the cursor position, then writes the attribute and character to the buffer. For read/write character interface in graphics mode, the characters are formed from a character image which is maintained in the system ROM.

Input:	AH = 09h	
	AL	Character to write
	BH	Display page (valid for alpha modes only)
	BL	Attribute of character (alpha)
		Color of character (graphics)
	CX	Count of characters to be written
Output:	None	

4.6.13 Write Character Only at Current Cursor Position

In text mode, this function finds the buffer address of the cursor position, then puts character to the buffer. For read/write character interface in graphics mode, the characters are formed from a character image which is maintained in the system ROM.

Input:	AH = 0Ah	
	AL	Character to be written
	BH	Display page (valid for alpha modes only)
	CX	Count of characters to be written
Output:	None	

4.6.14 Set Color Palette

This function programs the CRTIC border control register to set the desired color.

Input:	AH = 0Bh	
	BH	Palette color to be set (0~127)
	BL	Color value to be used
Output:	None	

4.6.15 Write Dot

This function writes a dot at the specified location.

Input:	AH = 0Ch	
	AL	Color value
	CX	Column number
	DX	Row number

Output: None

4.6.16 Read Dot

This function reads a dot at the desired location of the screen.

Input:	AH = 0Dh	
	CX	Column number
	DX	Row number

Output: AL = dot

4.6.17 Write Character as Teletype to Active Page

This function writes a character at the cursor position of the active page, then move the cursor to the next position. The attribute remains the same in text mode. The cursor moves to the next line and scrolls the screen, if applicable.

Input:	AH = 0Eh	
	AL	Character to write
	BL	Foreground color in graphics mode

Output: No registers

4.6.18 Current Video State

This function returns the current video mode by reading the BIOS data area.

Input:	AH = 0Fh	
Output:	AH	Number of character columns on screen
	AL	Mode currently set
	BH	Current active display page

Color Palette Interface

This function supports different ways to program the color palette registers.

Input:	AH = 10h	
	AL = 00h	Set individual palette registers
	BH	New color value
	BL	Register to set (no value check)
	AL = 01h	Set overscan register
	BH	Color Value
	AL = 02h	Set all palette registers and overscan register
	ES:DX	Pointer to 17-byte table
		Byte 0 ~ 15 : palette values
		Byte 16 : overscan values
	AL = 03h	Toggle intensity/blink bit
	BL = 00h	Enable intensity
	= 01h	Enable blinking (value check)
	AL = 07h	Read individual palette registers
	BL	Register to set (no value check)
Output:	BH	Value read
	AL = 08h	Read overscan register
	AL = 09h	Read all palette registers and overscan register
	ES:DX	Pointer to 17-byte buffer, return the following:
		Byte 0 ~ 15 : palette values
		Byte 16 : overscan values
	AL = 10h	Set individual color palette
	BX	Set color register (only BL is used)
	DH	Set red value
	CH	Set green value
	CL	Set blue value
	AL = 12h	Set block of color registers (R. G. B. of DAC)
	BX	Set first color register (only BL is used)
	CX	Number of color registers to set (no value check)
	ES:DX	Pointer to the color value table
		(Format: R, G, B, R, G, B...)

	AL = 13h	Select color page
	BL = 00h	Select paging mode
	BH	Paging mode
	= 00h	Select 4 register block of 64 registers
	<> 00h	Select 16 register block of 16 registers
	BL <> 00h	Select page
	BH	Page mode (0-base)
		For 64-register block mode:
	00h	Select 1st block of 64 color registers
	01h	Select 2nd block of 64 color registers
	02h	Select 3rd block of 64 color registers
	03h	Select 4th block of 64 color registers
		For 16-register block mode:
	00h	Select 1st block of 64 color registers
	01h	Select 2nd block of 64 color registers
	02h	Select 3rd block of 64 color registers
	:	
	:	
	0Fh	Select 16th block of 64 color registers
	AL = 15h	Read individual color register
	BX	Color register to be read (only BL is used)
Output:	DH	Red value
	CH	Green value
	CL	Blue value
	AL = 17h	Read block of color registers
	BX	Read first color register
	CX	Number of color register to be read
	ES:DX	Pointer to a data buffer (Format: R, G, B, R, G, B...)
	AL = 18h	Set PEL mask register
	BL	Set value
	AL = 19h	Read PEL mask register
	BL	Read value
Output:	AL = 1Ah	Read color page state
	BL	Current paging mode
	0	4 x 64 mode
	1	16 x 16 mode
	BH	Current page
	AL = 1Bh	Sum color values (R, G, B) to gray shades
	BX	Sum first color register (only BL is used)
	CX	Number of color registers to sum

Character Generator Load

This function loads character font to character generator and displays the pattern selected.

Input:

AH = 11h	
AL = 00h	User-load alpha mode
BH	Number of bytes per character (no value check)
BL	Block to load (0-7) (no value check)
CX	Count of characters
DX	Character offset (starting ASCII code)
ES:BP	Pointer to user font table

AL = 01h	ROM 8 x 14 font
BL	Block to load (0-7) (no value check)

AL = 02h	ROM 8 x 8 font
BL	Block to load (0-7) (no value check)

AL = 03h	Set block specifier
BL	Character generator block selected
AL = 04h	ROM 8 x 16 font
BL	Block to load (0-7) (no value check)

AL = 10h, 11h, 12, 14h are similar to AL = 00h, 01h, 02h 04h respectively, with the following exceptions:

1. Points (byte/char = 40:85) are reassigned according to the font selected.
2. Rows (40:84) = $\text{INT}[(\text{scan line})/\text{points}] - 1$
3. Length of display buffer (40:4C):
(No. of rows) x columns x 2
4. CRTCs are programmed as follows:

R09 = points - 1	maximum scan line
R0A = points - 2	cursor start
R0B = points - 1	cursor end
R12 = vertical displacement end	
350 and 400 scan line modes:	
$[(\text{No. of rows on screen}) \times \text{points}] - 1$	
200 scan line mode:	
$\{[(\text{No. of rows on screen}) \times \text{points}] \times 2\} - 1$	
R14 = points - 1 (for mode 7 only)	underline location

AL = 20h	User-load graphics characters (INT 1Fh -- font 8x8)
ES:BP	pointer to user font table

AL = 21h	User-load graphics characters (INT 43h)
BL	Row specifier
CX	Number of bytes per character
ES:BP	Pointer to user font table

AL = 22h	ROM 8 x 14 font (INT 43h)
BL	Row specifier

	AL = 23h	ROM 8 x 8 font (INT 43h)
	BL	Row specifier
	AL = 24h	ROM 8 x 16 font (INT 43h)
	BL	Row specifier
	AL = 30h	Information
	BH = 00h	Return current INT 1Fh pointer
	BH = 01h	Return current INT 43h pointer
	BH = 02h	Return ROM 8 x 14 font pointer
	BH = 03h	Return ROM 8 x 8 font pointer
	BH = 04h	Return ROM 8 x 8 font pointer
	BH = 05h	Return ROM 9 x 14 font alternate
	BH = 06h	Return ROM 8 x 16 font alternate
	BH = 07h	Return ROM 9 x 16 font alternate
Output:	CX	Number of bytes per character
	DX	Rows
	ES:BP	Pointer to the desired table

Alternate Select

This function supports the selection of the desired environment of different effects.

Input:	AH = 12h	
	BL = 10h	Return VGA information
Output:	BH = 00h	Color mode (address 3DX available)
	= 00h	Monochrome mode (address 3BX available)
	BL	Memory value:
		00h 64 KB
		01h 128 KB
		02h 192 KB
		03h 256 KB
	CH	Adaptor bits
	CL	Switch settings
	BL = 20h	Select alternate print screen routine
	BL = 30h	Select scan lines for alpha modes
	AL = 0	200 scan lines
	= 1	350 scan lines
	= 2	400 scan lines
	> 2	Function not supported
Output:	AL = 12h	Function supported
	= 00h	Function not supported
	Note: This function performs on the next mode setting before the mode is set.	
	BL = 31h	Default palette loading
	AL = 00h	Enable default palette loading
	= 01h	Disable default palette loading (value check)
Output:	AL = 12h	Function supported

	= 00h	Function not supported
	BL = 32h	Video I/O port
	AL = 00h	Enable the address decode and display buffer
	= 01h	Disable the address decode and display buffer
Output:	AL = 12h	Function supported
	= 00h	Function not supported
	BL = 33h	Gray shades summing (video summing enable)
	AL = 00h	Enable summing
	= 01h	Disable summing (value check)
Output:	AL = 12h	Function supported
	= 00h	Function not supported
	BL = 34h	Cursor emulation (40:87 bit 0)
	AL = 00h	Enable cursor emulation
	= 01h	Disable cursor emulation (value check)
Output:	AL = 12h	Function supported
	= 00h	Function not supported
	BL = 35h	Adaptor/system display switch
	AL = 00h	Disable the initial display
	ES:DX	Pointer to switch state buffer (128 bytes reserved)
	AL = 01h	Enable the initial display
	AL = 02h	Disable active display switch (save to buffer)
	ES:DX	Pointer to switch state buffer
	AL = 03h	Enable inactive display switch (load from buffer)
	ES:DX	Pointer to switch state buffer saved
	AL = 80h	Set bit 6 (40:89) on
	BL = 36h	Enable video refresh control
	AL = 00h	Enable refresh
	AL = 01h	Disable refresh

Read/Write Display Combination Code

This function tells if the onboard display or the add-on display is active.

Input: AH = 1Ah
 AL = 00h Read display combination codes

Output: BL Active display code
 BH Alternate display code
 AL = 1Ah Function supported

Input: AL = 01h Write display combination codes
 BL Active display code
 BH Alternate display code

Output: AL = 1Ah Function supported

Valid display codes:

00h	No display
01h	Monochrome with 5151 (monochrome)
02h	CGA with 5153/4 (color)
03h	Reserved
04h	EGA with 5153/4 (color)
05h	EGA with 5151 (monochrome)
06h	Professional graphics system with 5175 (color)
07h	VGA adapter with analog monochrome
08h	VGA adapter with analog color
09h	Reserved
0Ah	8086-system video with 5153 or 5154 color
0Bh	8086-system video with analog b/w
0Ch	8086-system video with analog color
0Dh~FEh	Reserved
FFh	Unknown

Return Functionality and Video State Information

This function tells the static information of the system and the user-changeable dynamic information, such as the current video state.

Input: AH = 1Bh
 BX Implementation type (only BX = 00h supported)
 ES:DI User buffer pointer

Output: AL = 1Bh Function supported

Save/Restore Video State

Input: AH = 1Ch
 AL = 00h Return save/restore state buffer size
 CX States requested
 (If CX = 0, then null returns with AL = 0.)

Output: AL = 1Ch Function supported
 BX Save/restore buffer size block count

Input: AL = 01h Save state

	CX	States requested (If CX = 0, then null returns with AL = 0.)
	ES:BX	Buffer pointer to save state
Output:	AL = 1Ch	Function supported
Input:	AL = 02h	Restore state
	CX	States requested (If CX = 0, then null returns with AL = 0.)
	ES:BX	Buffer pointer to save state
Output:	AL = 1Ch	Function supported
	AL > 02h	Function not supported (Null returns with AL = 0.)

4.6.19 Write String

This function displays a string of characters on the screen.

Input:	AH = 13h	Pointer to string to be written
	ES:BP	Length of character string to be written
	CX	Cursor position for string to be written
	DX	Page number
	BH	Write character string
	AL = 00h	Cursor is not moved
	BL -- Attribute, string is {char, ..., char}	Write character string and move cursor
	AL = 01h	Cursor is moved
	BL -- Attribute, string is {char, ..., char}	Write character & attribute string (Valid for Alpha Modes only)
	AL = 02h	String is {char, attr, ..., char, attr}, Cursor is not moved
	AL = 03h	Write character & attribute string and move cursor; (Valid for Alpha Modes only) String is {char, attr, ..., char, attr}, Cursor is moved
Output:	None	
Reserved		

These are reserved functions.

Input:	AH = 14h~19h, 1Dh~FFh
Output:	None

4.7 Equipment Determination (INT 11h)

This routine indicates the hardware environment is installed on the system.

Input: None

Output:

AX is set	
Bits 15~14	Number of printers attached
Bit 13	Serial printer attached
Bit 12	Not used
Bits 11~9	Number of RS232 ports attached
Bit 8	Not used
Bits 7~6	Number of diskette drives (00 = 1; 01 = 2 only if bit 0 = 1)
Bits 5~4	Initial video mode 00 = Unused 01 = 40 x 25 color 10 = 80 x 25 color 11 = 80 x 25 monochrome
Bit 3	Not used
Bit 2	Not used
Bit 1	Math coprocessor (0 = not installed; 1 = installed)
Bit 0	Diskette drive (0 = not installed; 1 = installed)

4.8 Memory Size Determination (INT 12h)

This routine returns the RAM size below address 10000h (the maximum size is 639K, 1K is used as expanded BIOS area).

Input: None

Output: AX Number of contiguous 1KB blocks of memory

4.9 Diskette I/O (INT 13h, Part 1)

This interface provides access to 360 KB, 720 KB, 1.2 MB, 1.44 and 2.88MB diskette drives supported by the system.

Functions:	AH = 00h	Reset diskette drive
	AH = 01h	Read status
	AH = 02h	Read diskette
	AH = 03h	Write diskette
	AH = 04h	Verify diskette sectors
	AH = 05h	Format diskette track
	AH = 06h~07h	Reserved
	AH = 08h	Read drive parameters
	AH = 09h~14h	Reserved
	AH = 15h	Read DASD type
	AH = 16h	Disk change line status
	AH = 17h	Set DASD type for format
	AH = 18h	Set media type for format

	AH = 19h~FFh	Reserved
Input:	AH	Function number
	AL	Number of sectors (value check)
	CH	Track number (non-value check)
	CL	Sector number (non-value check)
	DH	Head number (non-value check)
	DL	Drive number (value-check)
	ES:BX	Transfer address (boundary check)
Output:	CY = 1	If operation error
	CY = 0	If operation correct
	AH	Status of operation
	AH = 00h	No error
	AH = 01h	Invalid function request
	AH = 02h	Address mark not found
	AH = 03h	Write protect error
	AH = 04h	Requested sector not found
	AH = 06h	Media has been changed
	AH = 08h	Reserved
	AH = 09h	Attempt to DMA across 64KB boundary
	AH = 0Ch	Media type not found
	AH = 10h	CRC error on diskette read
	AH = 20h	General controller failure
	AH = 40h	Seek operation failed
	AH = 80h	Timeout

4.9.1 Reset Diskette Drive

Input:	AH = 00h	
	DL = 0~1	Drive number
Output:	AH	Status of operation
	CY = 1	If error
	CY = 0	If no error

4.9.2 Read Status

Input:	AH = 01h	
	DL = 0~1	Drive number
Output:	AH	Status of the drive system
	CY = 1	If error
	CY = 0	If no error

4.9.3 Diskette Read

Input:	AH = 02h	
	AL	Number of sectors
	CH	Track number
	CL	Sector number
	DL	Drive number
	DH	Head number
	ES:BX	Address of buffer
Output:	AH	Status of operation
	AL	Number of transfers
	CY = 1	If error
	CY = 0	If no error

4.9.4 Diskette Write

Input:	AH = 03h	
	AL	Number of sectors
	CH	Track number
	CL	Sector number
	DL	Drive number (only 0~1 legal)
	DH	Head number (only 0~1 legal)
	ES:BX	Address of buffer
Output:	AH	Status of operation
	AL	Number of transfers
	CY = 1	If error is detected
	CY = 0	If no error

4.9.5 Diskette Verify

Input:	AH = 04h	
	AL	Number of sectors
	CH	Track number
	CL	Sector number
	DH	Head number (only 0~1 legal)
	DL	Drive number (only 0~1 legal)
Output:	AH	Status of operation
	AL	Number of transfers
	CY = 1	If error
	CY = 0	If no error

4.9.6 Format Diskette Track

Input:	AH = 05h	
	AL	Number of sectors
	CH	Track number
	DH	Head number (only 0~1 legal)
	DL	Drive number (only 0~1 legal)
	ES:BX	Address of buffer
Output:	AH	Status of operation
	CY = 1	If error is detected
	CY = 0	If no error

4.9.7 Diskette Drive Parameters

Input:	AH = 08h	
	DL	Drive number (only 0~1 legal)
Output:	ES:DI	Pointer to drive parameter table
	CH	Maximum number of tracks (Low order 8 bits)
	CL (bits 7~6)	Maximum number of tracks, (High order 2 bits)
	(bits 5~0)	Maximum sectors per track
	DH	Maximum head number
	DL	Number of diskette drives installed
	BH	0
	BL (bits 7~4)	0
	(bits 3~0)	Valid drive type value in CMOS
		01 360 KB drive
		02 1.2 MB drive
		03 720 KB drive
		04 1.44 MB drive
	AX	0

4.9.8 Read DASD Type

Input:	AH = 15h	
	DL	Drive number
Output:	AH = 00h	Drive not present
	AH = 01h	Diskette, no change line available
	AH = 02h	Diskette, change line available
	AH = 03h	Fixed disk
	CY = 0	No error
	CY = 1	Invalid drive number

4.9.9 Disk Change Line Status

Input:	AH = 16h	
	DL	Drive (0~1)
Output:	AH = 00h	
	CY = 0	Disk change line not active
	AH = 06h	
	CY = 1	Disk change line active and carry bit on
	AH = 01h	
	CY = 1	Invalid drive number
	AH = 80h	
	CY = 1	Diskette drive not ready

4.9.10 Set DASD Type for Format

Input:	AH = 17h	
	AL = 00h	Not used
	AL = 01h	360 KB diskette in 360 KB drive
	AL = 02h	360 KB diskette in 1.2 MB drive
	AL = 03h	1.2 MB diskette in 1.2 MB drive
	AL = 04h	720 KB diskette in 720 KB drive
	AL = 05h	720 KB diskette in 1.44 MB drive
	DL	Drive number (0~1)
Output:	AH	Status of operation
	CY = 0	Successful operation
		(AH = 0 on return, except for read DASD type AH = 15)
	CY = 1	Failed operation (AH has error for read/write/verify)

4.9.11 Set Media Type for Format

Input:	AH = 18h	
	CH	Maximum number of tracks
		(Low order 8 of 10 bits)
		CL (bits 7~6) Maximum number of tracks
Output:		(High order two bits)
	CL (bits 5~0)	Maximum sectors per track
	DL	Drive number (0~1 allowed, value checked)
	ES:DI	Pointer to drive parameter table for this media type;
		unchanged if AH is non-zero
	CY = 0	Track and sectors/track combination
	AH = 00h	Successful, track and sector combination supported
	CY = 1	Sectors/track not supported
	AH = 01h	Function is not available
	AH = 0Ch	Track and sector combination not supported

4.10 Fixed Disk I/O (INT 13h, Part 2)

This interface provides access to 5.25-inch fixed disk drives through the fixed disk controller.

Functions:	AH = 00h	Disk reset
	AH = 01h	Read status
	AH = 02h	Read disk
	AH = 03h	Write disk
	AH = 04h	Verify disk sectors
	AH = 05h	Format disk track
	AH = 06h~07h	Reserved
	AH = 08h	Disk drive parameters
	AH = 09h	Disk Init. parameters
	AH = 0Ah	Disk read long
	AH = 0Bh	Disk write long
	AH = 0Ch	Disk seek
	AH = 0Dh	Disk Alt_Reset
	AH = 0Eh~0Fh	Reserved
	AH = 10h	Disk Test_Ready
	AH = 11h	Disk recalibrate
	AH = 12h~13h	Reserved
	AH = 14h	Disk diagnostics
	AH = 15h	Disk DASD type
	AH = 16h~FFh	Reserved
	AH = 19h	Park heads
Input:	AH	Function number
	AL	Number of sectors
	CH	Cylinder number (0~1023)
	CL	Sector number (1~17)
		High 2 bits of cylinder number are placed in the high 2 bits of the CL register
	DH	Head number (0~15 are allowed)
	DL	Drive number (80h~81h for disk)
Output:	ES:BX	Transfer address
		See individual functions

4.10.1 Disk Reset

Input:	AH = 00h DL = 80h~81h	For fixed disk
Output:	AH	Status of operation
	= 00h	No error
	= 01h	Invalid function request
	= 02h	Address mark not found
	= 03h	Write protect error
	= 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 detected
	= 0Bh	Bad cylinder detected
	= 0Dh	Invalid number of sectors format
	= 0Eh	Control data address mark detected
	= 0Fh	DMA arbitration level out of range
	= 10h	Uncorrectable error checking and correction (ECC) or cyclic redundancy check CRC error
	= 11h	ECC corrected data error
	= 20h	General controller failure
	= 40h	Seek operation failed
	= 80h	Time-out
	= AAh	Not ready
	= BBh	Undefined error counted
	= CCh	Write fault on selected drive
	= E0h	Status error/error register = 0
	= FFh	Sense operation failed
	CY = 1	If error
	CY = 0	If no error

4.10.2 Read Status

Input:	AH = 01h DL = 80h~81h	Drive number
Output:	AH	Status of the system
	CY = 1	If error is detected
	CY = 0	If no error

4.10.3 Read Disk

Input:	AH = 02h	
	AL	Number of sectors
	CH	Cylinder number bits 0~7
	CL	Sector number - bits 0~5
		Bits 6~7 are placed in bits 8~9 of the cylinder number
	DL	Drive number
	DH	Head number
	ES:BX	Address of buffer
Output:	AH	Status of operation
	CY = 1	If error is detected
	CY = 0	If no error

4.10.4 Write Disk

Input:	AH = 03h	
	AL	Number of sectors
	CH	Cylinder number bits 0~7
	CL	Sector number - bits 0~5
		Bits 6~7 are placed in bits 8~9 of the cylinder number
	DH	Head number
	DL	Drive number
	ES:BX	Address of buffer
Output:	AH	Status of operation
	CY = 1	If error is detected
	CY = 0	If no error

4.10.5 Verify Disk Sectors

Input:	AH = 04h	
	AL	Number of sectors
	CH	Cylinder number bits 0~7
	CL	Sector number - bits 0~5
		Bits 6~7 are placed in bits 8~9 of the cylinder number
	DL	Drive number
	DH	Head number
Output:	None	

4.10.6 Format Disk Track

Input: AH = 05h
 AL Number of sectors
 CH Cylinder number bits 0~7
 CL Sector number -- bits 0~5
 Bits 6~7 are placed in bits 8~9 of the cylinder number
 DL Drive number
 DH Head number
 ES:BX Address of buffer for reads and writes
 This points to a 512-byte buffer; the first 2 * (No. of
 sectors/track) -- bytes contain F, N for each sector
 where: F = 00h for a good sector,
 80h for a bad sector
 N = sector number

Output: None

4.10.7 Disk Drive Parameters

Input: AH = 08h
 DL Drive number

Output: AX = 0
 BH = 0
 DL Number of diskette drives installed
 DH Maximum usable head value
 CH Maximum usable cylinder value (lower 8 bits)
 CL Maximum usable cylinder value (bits 0~5)
 and the high (bits 6~7) cylinder value bits

4.10.8 Initialize Disk Parameters

Input: AH = 09h
 DL Drive number

Output: AH Status if fixed drive present
 CY = 1 If error is detected
 CY = 0 If no error

4.10.9 Disk Read Long

Input:	AH = 0Ah	
	AL	Number of sectors
	CH	Cylinder number bits 0~7
	CL	Sector number - bits 0~5
		Bits 6~7 are placed in bits 8~9 of the cylinder number
	DH	Head number
	DL	Drive number
Output:	ES:BX	Address of buffer, size must accommodate 4 bytes of ECC per sector (i.e. 516 bytes/sector)
	AH	Status of operation
	CY = 1	If error is detected
	CY = 0	If no error

4.10.10 Disk Write Long

Input:	AH = 0Bh	
	AL	Number of sectors
	CH	Cylinder number bits 0~7
	CL	Sector number -- bits 0~5
		Bits 6~7 are placed in bits 8~9 of the cylinder number
	DH	Head number
	DL	Drive number
Output:	ES:BX	Address of buffer, each sector of data must be followed by the 4 bytes ECC
	AH	Status of operation
	CY = 1	If error is detected
	CY = 0	If no error

4.10.11 Disk Seek

Input:	AH = 0Ch	
	CH	Cylinder number bits 0~7
	CL	Sector number -- bits 0~5
		Bits 6~7 are placed in bits 8~9 of the cylinder number
	DL	Drive number
Output:	DH	Head number
	AH	Status of operation
	CY = 1	If error is detected
	CY = 0	If no error

4.10.12 Disk Alternate Reset

Input:	AH = 0Dh	
	DL	Drive number
Output:	AH	Status (if fixed drive present)
	CY = 1	If error is detected
	CY = 0	If no error

4.10.13 Disk Ready Test

Input:	AH = 10h	
	DL	Drive number
Output:	AH	Status (if fixed drive present)
	CY = 1	If error is detected
	CY = 0	If no error

4.10.14 Disk Recalibrate

Input:	AH = 11h	
	DL	Drive number
Output:	AH	Status (If fixed drive present)
	CY = 1	If error is detected
	CY = 0	If no error

4.10.15 Disk Diagnostics

Input:	AH = 14h	
	DL	Drive number
Output:	AH	Status if fixed drive present
	CY = 1	If error is detected
	CY = 0	If no error

4.10.16 Read DASD Type

Input:	AH = 15h	
	DL	Drive number
Output:	AH = 00h	Drive not present
	AH = 01h	Diskette, no change line available
	AH = 02h	Diskette, change line available
	AH = 03h	Fixed disk
	CX:DX	Contains total number of sectors on disk
	CY = 1	If error is detected
	CY = 0	If no error

4.11 RS232 I/O (INT 14h)

This routine allows access to the asynchronous communication I/O port. No checking for communications port number is provided. If port address is 0, then null is returned.

Functions:	AH = 00h	Initialize the communications port
	AH = 01h	Send character
	AH = 02h	Receive a character
	AH = 03h	Return the communications port status
	AH = 04h	Extended initialization
	AH = 05h	Extended communications port control
	AH = 06h~FFh	Reserved

Input:	AH	Function number
	DX	Communications port number 0~3

Output: See individual functions

4.11.1 Initialize the Communication Port

Input:	AH = 00h	
	DX	Device number (0, 1, 2, 3)
	AL	Parameters for initialization

7	6	5	4	3	2	1	0
- Baud Rate -			- Parity -		- Stop Bit -		- Word Length -
000 - 110			00 - none		0 - 1		10 - 7 bits
001 - 150			01 - odd		1 - 2		11 - 8 bits
010 - 300			11 - even				
- Baud Rate -							
011 - 600							
100 - 1200							
101 - 2400							
110 - 4800							
111 - 9600							

Output:	AH	Line control states are as follows:
		Bit 7 Time out
		Bit 6 Transmitter shift register empty
		Bit 5 Transmitter holding register empty
		Bit 4 Break detected
		Bit 3 Framing error
		Bit 2 Parity error
		Bit 1 Overrun error
		Bit 0 Data ready
	AL	Modem states are as follows:
		Bit 7 Received line signal detect
		Bit 6 Ring indicator
		Bit 5 Data set ready
		Bit 4 Clear to send
		Bit 3 Delta receive line signal detect
		Bit 2 Trailing edge ring detector
		Bit 1 Delta data set ready
		Bit 0 Delta clear to send

4.11.2 Send a Character

Input: AH = 01h
 AL Character
 DX Device number (0, 1, 2, 3)

Output: AL Character
 AH Status of operation
 Bit 7 = 0 Character was sent
 Bit 7 = 1 Error encountered
 If Bit 7 of AH is 0, then:
 Bit 6 Transmitter shift register empty
 Bit 5 Transmitter holding register empty
 Bit 4 Break detected
 Bit 3 Framing error
 Bit 2 Parity error
 Bit 1 Overrun error
 Bit 0 Data ready

4.11.3 Receive a Character

Input: AH = 02h
 DX Device number (0, 1, 2, 3)

Output: AH = 00h No error
 AL Character received
 AH Status of operation
 Bit 7 = 0 Character was sent
 Bit 7 = 1 Error encountered
 If Bit 7 of AH is 0, then:
 Bit 6 Transmitter shift register empty
 Bit 5 Transmitter holding register empty
 Bit 4 Break detected
 Bit 3 Framing error
 Bit 2 Parity error
 Bit 1 Overrun error
 Bit 0 Data ready

4.11.4 Return the Communication Port Status

Input:	AH = 03h	
	DX	Device number (0, 1, 2, 3)
Output:	DX	RS232 card selector (0, 1)
	AH	Communication line states are as follows:
		Bit 7 Time out
		Bit 6 Transmitter shift register empty
		Bit 5 Transmitter holding register empty
		Bit 4 Break detected
		Bit 3 Framing error
		Bit 2 Parity error
		Bit 1 Overrun error
		Bit 0 Data ready
	AL	Modem states are as follows:
		Bit 7 Received line signal detect
		Bit 6 Ring indicator
		Bit 5 Data set ready
		Bit 4 Clear to send
		Bit 3 Delta receive line signal detect
		Bit 2 Trailing edge ring detector
		Bit 1 Delta data set ready
		Bit 0 Delta clear to send

4.11.5 Extended Initialization

Input:	AH = 04h	
	AL	Break
	BH	Parity
	BL	Stop bit
	CH	Word length
	CL	Baud rate
Output:	AL	Modem status
	AH	Line control status

4.11.6 Extended Communications Port Control

Input:	AH = 05h	
	AL = 0	Read modem control register
	DX	Device number (0, 1, 2, 3)

Output:	BL	Modem control register
		Bit 0 =1 Data terminal ready
		Bit 1 =1 Request to send
		Bit 2 =1 Out1
		Bit 3 =1 Out2
		Bit 4 =1 Loop
		Bit 5 ~ 7 Reserved
Input:	AL = 1	Write modem control register
	DX	Device number (0, 1, 2, 3)
	BL	Data to write
Output:	AL	Modem status
	AH	Line control status

4.12 System Service Routines (INT 15h)

Functions:	AH = 00~03h	Cassette I/O functions
	AH = 04~4Eh	Unused functions
	AH = 4Fh	Keyboard intercept (null)
	AH = 50~7Fh	Unused functions
	AH = 80h	Device open (null)
	AH = 81h	Device close (null)
	AH = 82h	Program termination (null)
	AH = 83h	Event wait (null)
	AH = 84h	Joystick support
	AH = 85h	System request key pressed (null)
	AH = 86h	Wait
	AH = 87h	Move block
	AH = 88h	Extended memory size determination
	AH = 89h	Processor to virtual mode
	AH = 90h	Device busy (null)
	AH = 91h	Interrupt complete (null)
	AH = C0h	Return system configuration parameters
Input:	AH	Function number
Output:	See individual functions	

4.12.1 Cassette I/O Functions (null)

These routines support an interface between the cassette port and the system. Returns for these functions are always AH = 86h, CY = 1 if cassette port is not present.

Input:	AH = 00~03h	
Output:	AH = 86h	Cassette port not present
	CY = 1	

4.12.2 Keyboard Intercept (null)

The keyboard intercept is called asynchronously by the hardware keyboard interrupt 09h routine. This allows for a keystroke to be changed or absorbed. Normally the system returns with the scan code unchanged, but the operating system can redirect an interrupt 15h to its own routine and do the following:

- Replace (AL) with a different scan code and return with the carry flag set, effectively changing the keystroke.
- Process the keystroke and return with the carry flag cleared causing interrupt 09 routine to ignore the keystroke.

Input: AH = 4Fh

Output: AL Scan code
CY = 1

4.12.3 Device Open (null)

This function is reserved for the operating system.

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

Output: AH = 0
CY = 0

4.12.4 Device Close (null)

This function is reserved for the operating system.

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

Output: AH = 0
CY = 0

4.12.5 Program Termination (null)

This function is reserved for the operating system.

Input: AH = 82h
BX Device ID

Output: AH = 0
CY = 0

4.12.6 Event Wait

This function sets a timer which counts off CX:DX microseconds. When the timer expires, the high bit of the byte pointed to by ES:BX is set. It is the user's responsibility to insure that the bit is clear initially, and to monitor the bit after invoking this function.

Input:	AH = 83h	
	AL = 0	Set interval
	AL = 1	Cancel
	ES:BX	Pointer to a byte
	CX:DX	Number of microseconds to wait
Output:	CY = 0	Not busy (AL not equal to 0)
	CY = 1	Busy (AL = 0)
	AH = 0	

4.12.7 Joystick Support

This routine supports an interface between the joystick and the system.

Input:	AH = 84h	
	DX = 0	Read current switch settings
	DX = 1	Read resistive inputs
Output:	CY = 1	Invalid call
	If DX = 0:	
	AL = Switch setting (bits 7~4)	
	If DX = 1:	
	AX = a(x) value	
	BX = a(y) value	
	CX = b(x) value	
	DX = b(y) value	

4.12.8 System Request Key Pressed (null)

When the SysReq key is pressed, the BIOS keyboard hardware interrupt routine loads AX with 8500h and invokes INT 15. Normally this function is null, i.e., it just returns, but the user may revert it. When the key is released, INT 15 is again invoked, this time with 8501h in AX.

Input:	AH = 85h
Output:	AH = 0
	CY = 0

4.12.9 Wait

This function waits for the specified number of microseconds before returning to the user.

Input:	AH = 86h CX:DX	Number of microseconds to elapse before return to caller
Output:	CY = 0 CY = 1	Function is successful Wait function already in progress

4.12.10 Move Block

This function transfers up to 32768 words to or from the extended memory.

Input:	AH = 87h CX ES:SI	Number of words moved, maximum count = 8000h 32K words Pointer to Global Descriptor Table established by the user
Output:	AH = 0 AH = 1 AH = 2 AH = 3 CY = 0 CY = 1	Successful RAM parity (parity error cleared on return) Exception interrupt occurred Gate address line 20 failed Successful Error

4.12.11 Extended Memory Size Determination

This routine returns the number of consecutive 1K blocks above 1M. This value is obtained from CMOS RAM locations 30h and 31h, which were set when the system was started.

Input:	AH = 88h	
Output:	AX	Number of consecutive 1K blocks starting at 1M

4.12.12 Switch Processor to Protected Mode

This routine sets the processor to the protected mode.

Input:	AH = 89h ES:SI BH BL	Pointer to Global Descriptor Table Offset into Interrupt Descriptor Table where the first eight 8259 interrupts are to occur Offset into IDT where second eight 8259 interrupts are to occur
Output:	AH = 0	If successful; segment registers set for protected mode operation, AX and BP are destroyed

4.12.13 Device Busy (null)

This function is reserved for the operating system.

Input: AH = 90h
 AL Type code

Output: CY = 0 If no error
 CY = 1 If error

Type	Description	Timeout
00	fixed disk	yes
01	diskette	yes
02	keyboard	no
80	network	no
FC	fixed disk reset	yes
FD	drive motor start	yes
FE	printer	yes

4.12.14 Interrupt Complete (null)

This function is reserved for the operating system.

Input: AH = 91h
 AL Type code

Output: None, AX is destroyed

4.12.15 Return System Configuration Parameters

This routine gives information about the model of the machine, the BIOS revision level, and some hardware features.

Input: AH = C0h

Output: CY = 0
 AH = 0
 ES:BX Pointer to Descriptor table in ROM

Table 4-2 System Description

Size	Description	Value
Word	Descriptor length (bytes)	008h
Byte	Model	0FCh
Byte	Submodel	01h
Byte	BIOS revision level	000h
Byte	Feature information	04h
	Bit 7 = 1	BIOS uses DMA channel 3
	Bit 6 = 1	Cascaded interrupt level 2
	Bit 5 = 1	Real-time clock present
	Bit 4 = 1	System hook in keyboard interrupt routine
	Bit 3 = 0	Reserved
	Bit 2 = 0	Reserved
	Bit 1 = 0	Reserved
	Bit 0 = 0	Reserved

Reserved (null)

These are reserved functions.

Input: AH = 04h~4Eh, 50h~7Fh, 92h~BFh, C1h~FFh

Output: CY = 1
AH = 86h Invalid function

4.12.16 Pointing Device

This function supports programming for the pointing device, such as a mouse.

Enable/Disable

Input: AH = C2h
AL = 0
BH = 0 Disable the pointing device
= 1 Enable the pointing device
> 1 Invalid function call

Output: CY = 0 Operation completed
CY = 1 Unsuccessful operation
AH Status
00 No error
01 Invalid function call
02 Invalid function input
03 Error
04 Reserved
05 No far call installed
06 Reserved

Reset

Input: AH = C2h
 AL = 1

Output: Same
 BX Device ID

Set Sample Rate

Input: AH = C2h
 AL = 2
 BH Rate value

Output: Same

Set Resolution

Input: AH = C2h
 AL = 3
 BH Resolution value

Output: Same

Read Device Type

Input: AH = C2h
 AL = 4

Output: Same
 BH Device ID

Initialization

Input: AH = C2h
 AL = 5
 BH Data package size

Output: Same

Extended Command

Input: AH = C2h
 AL = 6
 BH = 0 Return status
 BH = 1 Set scaling to 1:1
 BH = 2 Set scaling to 2:1
 BH > 2 Illegal function call

Output: Same
 BL Status byte 1
 CL Status byte 2
 DL Status byte 3

Device Driver Far Call

Input: AH = C2h
 AL = 7
 ES Segment pointer
 BX Offset pointer

Output: None

4.12.17 Acer Extended Function (Int 15h)

Input: AH = 0C0h
 CX = 4341H ("AC")
 DX = 5245H ("ER")
 AL = See following table
 BX = See following table

Output: CY = 1 Error occurred
 CY = 0 Function call is OK. See following table.

Table 4-3 Acer Extended Functions

Input		Output	Description
BX	AX		
0		es:bx	Pointer of BIOS message table
1			Enter BIOS setup
2	0	AL	Get system speed AL = 1, High speed AL = 2, Low speed
	1		Set High speed (not supported)
	2		Set Low speed (not supported)
	3		Toggle speed (not supported)
3	0	AL	Get RAM BIOS status AL = 1, RAM BIOS enable AL = 2, RAM BIOS disable
	1		Enable RAM BIOS (not supported)
	2		Disable RAM BIOS (not supported)
5		es:bx	Return Acer model ID es:bx --- pointer of Acer model ID byte 0-2 = OEM ID:ACR byte 3-4 = Mode; ID:89
6	0	AL	Get write protect function status Bit 0 -- WDD write protect (1) Bit 1 -- FDD write protect (1)
	1		Set write protect for WDD
	2		Reset write protect for WDD
	3		Toggle write protect for WDD

Table 4-3 Acer Extended Functions (continued)

Input		Output	Description
BX	AX		
	4		Set write protect for FDD
	5		Reset write protect for FDD
	6		Toggle write protect for FDD
7	0	AL	Return cache controller status AL = 1, cache On status AL = 2, cache Off status
	1		Set cache On (not supported)
	2		Set cache Off (not supported)
	3		Get external cache size AL = 0 External cache does not exist AL = 1 16K external cache AL = 2 32K external cache AL = 3 64K external cache AL = 4 128K external cache AL = 5 256K external cache AL = 6 512K external cache
8	0	AL	Get RAM video status A1 = 0, RAM video disable A1 = 1, RAM vide enable
	1		Enable RAM BIOS (not supported)
	2		Disable RAM BIOS (not supported)
9	0	AL 00	Get parallel direction (not supported)
	1		Set parallel input (not supported)
	2		Set parallel output (not supported)

4.13 Keyboard I/O (INT 16h)

The keyboard generates interrupts on IRQ1, hardware interrupt 9. The code for this routine provides an interface between the hardware and the system software, reading the keys from the keyboard and putting them in the keyboard buffer. INT 16h provides the user with an interface to access keys placed into the keyboard buffer.

Functions:	AH = 00h	Read next character
	AH = 01h	Read buffer status
	AH = 02h	Return current shift status
	AH = 03h	Set typematic rate and delay
	AH = 05h	Place ASCII character/scan code in keyboard buffer
	AH = 06h~0Fh	Reserved
	AH = 10h	Read for enhanced keyboard
	AH = 11h	Read buffer status for enhanced keyboard
	AH = 12h	Read the extended shift status for enhanced keyboard
	AH = 13h~FFh	Reserved

4.13.1 Read Next Character

This function returns the scan code and ASCII code of the next character in the keyboard buffer and updates the buffer pointer. If the keyboard buffer is empty, the function waits for a key.

Input: AH = 00h

Output: AH Scan code
AL ASCII character (0 for special keys, functions keys, etc.)

4.13.2 Read Buffer Status

This function returns the keyboard buffer status to the user, that is, it tells the user whether a keystroke is available or not. The buffer pointer is not updated, even though the key is returned. Use function 00h to update the buffer pointer.

Input: AH = 01h

Output: ZF = 1 No keystroke queued
ZF = 0 Keystroke is available in queue, key in AX

4.13.3 Return Shift Status

When a "shift" key is pressed, no scan code/ASCII code is put into the keyboard buffer. Instead, a shift status (one bit) is set. This function indicates which key is pressed.

Input: AH = 02h

Output: AL Shift status
Bit 0 = 1 Right-Shift depressed
Bit 1 = 1 Left-Shift depressed
Bit 2 = 1 Ctrl-Shift depressed
Bit 3 = 1 Alt-Shift depressed
Bit 4 = 1 Scroll Lock state
Bit 5 = 1 Num Lock state
Bit 6 = 1 Caps Lock state
Bit 7 = 1 Insert state
All other registers are restored

4.13.4 Set Typematic Rate and Delay

When a key (except a shift key) is depressed, the keyboard handler repeats the key until the key is released. This function allows the user to set the typematic rate and delay time. The typematic rate is the rate at which the handler repeats the key. The delay time is the time between the first keystroke (normal) and the second keystroke (repeated).

Input:	AH = 03h		
	BL	Typematic rate (bits 4~0)	
	BL = 00h	30.0	BL = 10h 7.5
	BL = 01h	26.7	BL = 11h 6.7
	BL = 02h	24.0	BL = 12h 6.0
	BL = 03h	21.8	BL = 13h 5.5
	BL = 04h	20.0	BL = 14h 5.0
	BL = 05h	18.5	BL = 15h 4.6
	BL = 06h	17.1	BL = 16h 4.3
	BL = 07h	16.0	BL = 17h 4.0
	BL = 08h	15.0	BL = 18h 3.7
	BL = 09h	13.3	BL = 19h 3.3
	BL = 0Ah	12.0	BL = 1Ah 3.0
	BL = 0Bh	10.9	BL = 1Bh 2.7
	BL = 0Ch	10.0	BL = 1Ch 2.5
	BL = 0Dh	9.2	BL = 1Dh 2.3
	BL = 0Eh	8.5	BL = 1Eh 2.1
	BL = 0Fh	8.0	BL = 1Fh 2.0
	BH	Delay value (bits 1~0)	
	BH = 00h	250 milliseconds	
	BH = 01h	500 milliseconds	
	BH = 02h	750 milliseconds	
	BH = 03h	1000 milliseconds	

Output: None

4.13.5 Place ASCII/Scan Code in Keyboard Buffer

This function places an ASCII character/scan code combination into the keyboard buffer as if a key had been pressed.

Input:	AH = 05h	
	CL	ASCII code
	CH	Scan code
Output:	AL = 00h	Successful operation
	AL = 01h	Keyboard buffer full

4.13.6 Extended Read Interface for the Enhanced Keyboard

This function corresponds to AH = 0, but supports the enhanced keyboard.

Input: AH = 10h

Output: AH Scan code
AL ASCII code

4.13.7 Extended Buffer Status for the Enhanced Keyboard

This function corresponds to AH = 1, but supports the enhanced keyboard.

Input: AH = 11h

Output: ZF = 1 No keystroke queued
ZF = 0 Keystroke is available in queue, key in AX

4.13.8 Return Extended Shift Status for the Enhanced Keyboard

This function corresponds to AH = 1, but supports the enhanced keyboard.

Input: AH = 12h

Output: AL
Bit 0 = 1 Right-Shift key depressed
Bit 1 = 1 Left-Shift key depressed
Bit 2 = 1 Control key depressed
Bit 3 = 1 Alt key depressed
Bit 4 = 1 Scroll Lock active
Bit 5 = 1 Num Lock active
Bit 6 = 1 Caps key active
Bit 7 = 1 Ins key active
Output: AH Extended shift status
Bit 0 = 1 Left-Ctrl-Shift
Bit 1 = 1 Left-Alt-Shift
Bit 2 = 1 Right-Ctrl-Shift
Bit 3 = 1 Right-Alt-Shift
Bit 4 = 1 Scroll-Lock-Shift
Bit 5 = 1 Num-Lock-Shift
Bit 6 = 1 Caps-Lock-Shift
Bit 7 = 1 SysReq-Shift

4.14 Printer I/O (INT 17h)

This routine handles interrupts from the printer interface.

Functions:	AH = 00h	Print character
	AH = 01h	Initialize printer port
	AH = 02h	Read printer status
	AH = 03h~FFh	Reserved

Input:	AH	Function number
	DX	Device number (0~2)

Output: See individual functions

4.14.1 Print Character

Input:	AH = 00h	
	AL	Character
	DX	Device number
Output:	AH = 01h	Print failed (timeout)
	AH = 0Fh	Invalid port
	AH	Printer status
		Bit 0 = 1 Timeout
		Bit 1~2 Reserved
		Bit 3 = 1 I/O error
		Bit 4 = 1 Printer selected
		Bit 5 = 1 Out of paper
		Bit 6 = 1 Acknowledge
		Bit 7 = 1 Not busy

4.14.2 Initialize Printer Port

Input:	AH = 01h	
	DX	Device number (0~2)
Output:	AH	Printer status
		Bit 0 = 1 Timeout
		Bit 1~2 Reserved
		Bit 3 = 1 I/O error
		Bit 4 = 1 Printer selected
		Bit 5 = 1 Out of paper
		Bit 6 = 1 Acknowledge
		Bit 7 = 1 Not busy

4.14.3 Read Printer Status

Input:	AH = 02h	
	DX	Device number
Output:	AH	Printer status
		Bit 0 = 1 Timeout
		Bit 1~2 Reserved
		Bit 3 = 1 I/O error
		Bit 4 = 1 Printer selected
		Bit 5 = 1 Out of paper
		Bit 6 = 1 Acknowledge
		Bit 7 = 1 Not busy

4.15 System Boot (INT 19h)

After the BIOS self-test and initialization is over, control is passed to DOS by using this function. This routine reads the boot-sector from the diskette drive into the main memory and executes it.

If diskette time-out is encountered, the routine attempts to boot from the fixed disk drive. If this fails then INT 18h is invoked.

Input:	None
Output:	None

4.16 Clock Services (INT 1Ah)

This routine handles interrupts from the real-time clock.

Functions:	AH = 00h	Read the system timer count
	AH = 01h	Set the system timer count
	AH = 02h	Read the real-time clock time
	AH = 03h	Set the real-time clock time
	AH = 04h	Read the real-time clock date
	AH = 05h	Set the real-time clock date
	AH = 06h	Set the real-time clock alarm
	AH = 07h	Reset the real-time clock alarm
	AH = 08h~B0h	Reserved
	AH = B1h	PCI_FUNCTION_ID
	and AL = 01h	PCI_BIOS_PRESENT
	and AL = 02h	FIND_PCI_DEVICE
	and AL = 03h	FIND_PCI_CLASS_CODE
	and AL = 06h	GENERATE_SPECIAL_CYCLE
	and AL = 08h	READ_CONFIG_BYTE
	and AL = 09h	READ_CONFIG_WORD
	and AL = 0Ah	READ_CONFIG_DWORD
	and AL = 0Bh	WRITE_CONFIG_BYTE
	and AL = 0Ch	WRITE_CONFIG_WORD
	and AL = 0Dh	WRITE_CONFIG_DWORD
	AH = B2h~FFh	Reserved

Input:	Function number
--------	-----------------

Output: See individual functions

4.16.1 Read the System Timer Count

This function reads the system timer from the BIOS data area. This value is incremented at each system timer interrupt (INT 08h), about 18.2 times per second.

Input: AH = 00h

Output: CX High count word
DX Low count word
AL = 0 Timer has not passed 24 hours since last read
AL <> 0 Otherwise
AH = 0

4.16.2 Set the System Timer Count

This function sets the system timer in the BIOS data area.

Input: AH = 01h
CX High count word
DX Low count word

Output: AH = 0

4.16.3 Read the Real-Time Clock Time

This function reads hours, minutes, and seconds from the real-time clock.

Input: AH = 02h

Output: CH Hours in BCD
CL Minutes in BCD
DL = 00h Daylight Savings Time disabled
DL = 01h Daylight Savings Time enabled
DH Seconds in BCD
CY = 0 Successful
CY = 1 Clock is busy

4.16.4 Set the Real-Time Clock Time

This function sets hours, minutes, and seconds to the real-time clock.

Input:	AH = 03h	
	CH	Hours in BCD format
	CL	Minutes in BCD format
	DH	Seconds in BCD format
	DL = 00h	Daylight Savings Time disabled
	DL = 01h	Daylight Savings Time enabled
Output:	AH = 00h	
	CY = 0	Successful
	CY = 1	Clock is busy

4.16.5 Read the Real-Time Clock Date

This function reads century, year, month, and day from the real-time clock.

Input:	AH = 04h	
Output:	CH	Century in BCD (19 or 20)
	CL	Year in BCD
	DH	Month in BCD
	DL	Day in BCD
	AH = 00h	
	CY = 0	Successful
	CY = 1	Clock is busy

4.16.6 Set the Real-Time Clock Date

This function sets century, year, month, and day from the real-time clock.

Input:	AH = 05h	
	CH	Century in BCD (19 or 20)
	CL	Year in BCD
	DH	Month in BCD
	DL	Day in BCD
Output:	AH = 00h	
	CY = 0	Successful
	CY = 1	Clock is busy

4.16.7 Set the Real-Time Clock Alarm

This function sets the alarm (hours, minutes, and seconds) to the real-time clock. When the time is up, INT 4Ah is invoked.

Input: AH = 06h
 CH Hours in BCD
 CL Minutes in BCD
 DH Seconds in BCD

Output: AH = 00h
 CY = 0 Successful
 CY = 1 Clock is busy

4.16.8 Reset the Real-Time Clock Alarm

This function resets the alarm.

Input: AH = 07h

Output: AH = 00h
 CY = 0 Successful
 CY = 1 Clock is busy

Reserved

These functions are reserved.

Input: AH = 08h~FFh

Output: CY = 1 Invalid function

4.16.9 PCI BIOS Present

This function allows the caller to determine whether the PCI BIOS interface function set is present, and what the current interface version level is. It also provides information about what hardware mechanism for accessing configuration space is supported, and whether or not the hardware supports generation of PCI Special Cycles.

Input: AH = B1h
 AL = 01h

Output: EDX "PCI", "P" in DL, "C" in DH, etc. There is a 'space' character in the upper byte.
 AH Present status:
 00h = BIOS Present IFF EDX set properly.
 AL Hardware mechanism
 BH Interface Level Major Version
 BL Interface Level Minor Version
 CL Number of last PCI bus in the system
 CF Present status:
 set = No BIOS present
 reset = BIOS present IFF EDX set properly.

NC, AH = 0
EDX = ICP

4.16.10 Find PCI Device

This function returns the location of PCI devices that have a specific Device ID and Vendor ID. Given a Vendor ID, Device ID, and an Index (N), the function returns the bus number, device number, and function number of the nth device/function whose Vendor ID and Device ID match the input parameters.

Input:	AH	B1h
	AL	02h
	CX	Device ID (0..65535)
	DX	Vendor ID (0..65534)
	SI	Index (0..N)
Output:	BH	Bus number (0..255)
	BL	Device number in upper 5 bits
		Function number in bottom 3 bits
	AH	Return code:
		00h = SUCCESSFUL
		81h = DEVICE_NOT_FOUND
		83h = BAD_VENDOR_ID
CF	Completion status:	
	set = error	
	cleared = success	

4.16.11 Find PCI Class Code

This function returns the location of PCI devices that have a specific Class Code. Given a Class Code and an Index (N), the function returns the bus number, device number, and function number of the nth device/function whose Class code matches the input parameters.

Input:	AH	B1h
	AL	03h
	ECX	Class Code (in lower 3 bytes)
	SI	Index (0..N)
Output:	BH	Bus Number (0..255)
	BL	Device Number in upper 5 bits
		Function Number in bottom 3 bits
	AH	Return Code:
		00h = SUCCESSFUL
		86h = DEVICE_NOT_FOUND
	CF	Completion Status:
	set = error	
	cleared = success	

4.16.12 Generate Special Cycle

This function allows for generation of PCI special cycles. The generated special cycle is then broadcasted on a specific PCI bus in the system.

Input:	AH	= B1h	
	AL	= 06h	
	BH		Bus Number (0..255)
	EDX		Special Cycle Data
Output:	AH		Return Code: 00h = SUCCESSFUL 81h = FUNC_NOT_SUPPORTED
	CF		Completion Status: set = error reset = success

4.16.13 Read Configuration Byte

This function allows reading individual bytes from the configuration space of a specific device.

Input:	AH	= B1h	
	AL	= 08h	
	BH		Bus Number (0..255)
	BL		Device Number in upper 5 bits Function Number in bottom 3 bits
	DI		Register Number (0..255)
Output:	CL		Byte Read
	AH		Return Code: 00h = SUCCESSFUL 87h = BAD_REGISTER_NUMBER
	CF		Completion Status: set = error reset = success

4.16.14 Read Configuration Word

This function allows reading individual words from the configuration space of a specific device. The Register Number parameter must be a multiple of two (i.e., bit 0 must be set to 0).

Input:	AH	= B1h	
	AL	= 09h	
	BH		Bus Number (0..255)
	BL		Device Number in upper 5 bits Function Number in bottom 3 bits
	DI		Register Number (0,2,4,...254)
Output:	CX		Word Read
	AH		Return Code: 00h = SUCCESSFUL 87h = BAD_REGISTER_NUMBER
	CF		Completion Status: set = error reset = success

4.16.15 Read Configuration Dword

This function allows reading individual words from the configuration space of a specific device. The Register Number parameter must be a multiple of four (i.e., bits 0 and 1 must be set to 0).

Input:	AH	= B1h
	AL	= 0Ah
	BH	Bus Number (0..255)
	BL	Device Number in upper 5 bits
	DI	Function Number in bottom 3 bits
Output:	ECX	Dword Read
	AH	Return Code:
		00h = SUCCESSFUL
		87h = BAD_REGISTER_NUMBER
	CF	Completion Status:
		set = error
		reset = success

4.16.16 Write Configuration Byte

This function allows writing individual bytes to the configuration space of a specific device.

Input:	AH	= B1h
	AL	= 0Bh
	BH	Bus Number (0..255)
	BL	Device Number in upper 5 bits
	DI	Function Number in bottom 3 bits
Output:	DI	Register Number (0..255)
	CL	Byte Value to Write
	AH	Return Code:
		00h = SUCCESSFUL
		87h = BAD_REGISTER_NUMBER
	CF	Completion Status:
		set = error
		reset = success

4.16.17 Write Configuration Word

This function allows writing individual words from the configuration space of a specific device. The Register Number parameter must be a multiple of two (i.e., bit 0 must be set to 0).

Input:	AH	B1h
	AL	0Ch
	BH	Bus Number (0..255)
	BL	Device Number in upper 5 bits
	DI	Function Number in bottom 3 bits
	CX	Register Number (0,2,4,...254)
Output:		Word Value to Write
	AH	Return Code:
		00h = SUCCESSFUL
		87h = BAD_REGISTER_NUMBER
	CF	Completion Status:
		set = error reset = success

4.16.18 Write Configuration Dword

This function allows writing individual dwords from the configuration space of a specific device. The Register Number parameter must be a multiple of four (i.e., bits 0 and 1 must be set to 0).

Input:	AH	PCI Function ID
	AL	ODH
	BH	Write configuration dword
	BL	Bus number (0..255)
		Device number in upper 5 bits
		Function number in lower 3 bits
Output:	DI	Register number (0, 4, 8, ...252)
	ECX	Dword value to write
	AH	Return code:
		0 = SUCCESSFUL
		87h = BAD REGISTER NUMBER
	CF	Completion Status set = error, reset = success

4.16.19 Reserved

This function is reserved.

Input:	AH	08h ~ B0h or B2h~ FFh
Output:	CY	1 Invalid function

4.17 Configuration Data Block Structure

A 320-byte data block pointed to by DS:SI contains information on the expansion board functions. The field sizes of the data block are fixed sizes. This section details the structures of the function configuration data block.

4.17.1 Read Function Configuration Data Block

The subfunction 01h under the Read/Write Slot Configuration from CMOS reads the contents of the configuration data block structure. This data block has the following structure:

Four-Byte Compressed ID			Total bytes = 4, Offset = 00h
Byte 0	bit 7	Reserved = 0	
	bits 6-2	Character 1	
	bits 1-0	Character 2	
Byte 1	bits 7-5	Character 2	
	bits 4-0	Character 3	
Byte 2	bits 7-4	First hex digit of product number	
	bits 3-0	Second hex digit of product number	
Byte 3	bits 7-4	Third hex digit of product number	
	bits 3-0	One-digit product revision number	
ID and Slot Information			Total bytes = 2, Offset = 04h
Byte 0	bit 7	0 = No duplicate ID is present 1 = Duplicate is present	
	bit 6	0 = ID is readable 1 = ID is not readable	
	bits 5,4	Slot type 00 = Expansion slot 01 = Embedded device 10 = Virtual device 11 = Reserved	
	bits 3-0	Numeric identifier for duplicate CFG filenames (IDs) 0000 If no duplicate CFG filenames 0001 First duplicate CFG file 1111 15th duplicate CFG file	
Byte 1	bit 7	0 = Configuration is complete 1 = Configuration is not complete	
	bits 6-2	Reserved	
	bit 1	1 = EISA IOCHKERR supported 0 = EISA not IOCHKERR supported	
	bit 0	1 = EISA ENABLE supported (expansion board can be disabled) 0 = EISA ENABLE not supported (board can't be disabled)	

CFG File Extension Revision Level		Total bytes = 2, Offset = 06h
Byte 0	= Minor revision level (0 if no CFG file extension exists)	
Byte 1	= Major revision level (0 if no CFG file extension exists)	
Selections		Total bytes = 26, Offset = 08h
Byte 0	= First selection	
Byte 1	= Second selection	
Byte 25	= 26th selection	
Function Information		Total bytes = 1, Offset = 022h
Byte 0	bit 7 0 = Function is not disabled 1 = Function is disabled bit 6 CFG extension as free-form data bit 5 Port initialization entry follows bit 4 Port range entry follows bit 3 DMA entry follows bit 2 Interrupt (IRQ) entry follows bit 1 Memory entry follows bit 0 Type/subtype ASCII string entry follows	
Type and Subtype ASCII String		Total bytes = 80, Offset = 23h
Byte 0	= First character of ASCII string	
Byte 1	= Second character of ASCII string	
Byte 79	= 80th character of ASCII string	
Memory Configuration		Total bytes = 63, Offset = 73h
Byte 0	= Memory configuration byte	
bit 7	0 = Last entry 1 = More entries follow	
bit 6	Reserved	
bit 5	0 = Not shared memory 1 = Shared memory	
bits 4,3	Memory type 00 = Base or extended 01 = Expanded 10 = Virtual 11 = Other	
bit 2	Reserved	
bit 1	0 = Not cached 1 = Cached	
bit 0	0 = Read only (ROM) 1 = Read/write (RAM)	

Memory Configuration		Total bytes = 63, Offset = 73h
Byte 1	= Memory data size	
	bits 7-4	Reserved
	bits 3,2	Decode size
		00 = 20
		01 = 24
		10 = 32
		11 = Reserved
	bits 1,0	Data size (access size)
		00 = BYTE
		01 = WORD
		10 = DWORD
		11 = Reserved
Byte 2	= LSByte memory start address (divided by 100h)	
Byte 3	= Middle byte memory start address	
Byte 4	= MSByte memory start address	
Byte 5	= LSByte memory size (divided by 400h)	
Byte 6	= MSByte memory size (0 in this context means 64 MB)	
Interrupt Configuration		Total bytes = 14, Offset = B2h
Byte 0		
	bit 7	0 = Last entry 1 = More entries follow
	bit 6	0 = Not shared 1 = Shared
	bit 5	0 = Edge-triggered 1 = Level-triggered
	bit 4	Reserved
	bits 3-0	Interrupts (0-F)
Byte 1	= Reserved	

DMA Channel Description			Total bytes = 8, Offset = C0h
Byte 0	bit 7	0 = Last entry 1 = More entries follow	
	bit 6	0 = Not shared 1 = Shared	
	bits 5-3	Reserved (0)	
	bits 2-0	DMA channel number (0-7)	
Byte 1	bits 7,6	Reserved (0)	
	bits 5,4	DMA timing 00 - Default timing (ISA compatible) 01 - Type A timing 10 - Type B timing 11 - Type C timing (burst mode)	
	bits 3,2	Transfer size 00 - 8-bit (byte) transfer 01 - 16-bit (word) transfer 10 - 32-bit (dword) transfer 11 - Reserved	
	bits 1,0	Reserved (0)	
Port Information			Total bytes = 60, Offset = C8h
Byte 0	bit 7	0 = Last entry 1 = More entries follow	
	bit 6	0 = Not shared 1 = Shared	
	bit 5	Reserved	
	bits 4-0	Number of ports (minus 1) 00000 = 1 port 00001 = 2 sequential ports ... 11111 = 32 sequential ports	
Byte 1	= LSByte I/O port address		
Byte 2	= MSByte I/O port address		

Initialization Data		Total bytes = 60, Offset = 104h
Byte 0	= Initialization type	
bit 7	0 = Last entry 1 = More entries follow	
bits 6-3	Reserved (0)	
bit 2	Port value or mask value 0 = Write value to port 1 = Use mask and value	
bits 1,0	Type of access 00 = Byte address 01 = Word address 10 = Dword address 11 = Reserved	
If byte 0, bit 2 = 0 (no mask):		
bits 1,0	Port width to write 00 - byte 3 = port value 01 - byte 3 = LSB of port value - byte 4 = MSB of port value 10 - byte 3 = LSB of port value - byte 4 = Second byte port value - byte 5 = Third byte of port value - byte 6 = MSB of port value 11 - Reserved	
If byte 0, bit 2 = 1 (use mask):		
bits 1,0	Number of bytes/port value/mask 00 - byte 3 = port value - byte 4 = port mask (byte) 01 - byte 3 = LSB of port value - byte 4 = MSB of port value - byte 5 = LSB mask (word) - byte 6 = MSB mask (word) 10 - byte 3 = LSB of port value - byte 4 = Second byte of port value - byte 5 = Third byte of port value - byte 6 = MSB of port value - byte 7 = LSB mask (dword) - byte 8 = Second byte of port mask (dword) - byte 9 = Third byte of port mask (dword) - byte 10 = MSB mask (dword) 11 - Reserved	
Byte 1	= LSByte of port I/O address	
Byte 2	= MSByte of port I/O address	

When the freeform data bit is set in the function information byte (bit 6), the 320 byte data structure has the following specific format:

Four-Byte Compressed ID			Total bytes = 4, Offset = 00h
Byte 0	bit 7	Reserved = 0	
	bits 6-2	Character 1	
	bits 1-0	Character 2	
Byte 1	bits 7-5	Character 2	
	bits 4-0	Character 3	
Byte 2	bits 7-4	First hex digit of product number	
	bits 3-0	Second hex digit of product number	
Byte 3	bits 7-4	Third hex digit of product number	
	bits 3-0	One-digit product revision number	
ID and Slot Information			Total bytes = 2, Offset = 04h
Byte 0	bit 7	0 = No duplicate ID is present 1 = Duplicate is present	
	bit 6	0 = ID is readable 1 = ID is not readable	
	bits 5,4	Slot type 00 = Expansion slot 01 = Embedded device 10 = Virtual device 11 = Reserved	
	bits 3-0	Numeric identifier for duplicate CFG filenames (IDs) 0000 If no duplicate CFG filenames 0001 First duplicate CFG file ... 1111 15th duplicate CFG file	
Byte 1	bit 7	0 = Configuration is complete 1 = Configuration is not complete	
	bits 6-2	Reserved	
	bit 1	1 = EISA IOCHKERR supported 0 = EISA not IOCHKERR supported	
	bit 0	1 = EISA ENABLE supported (expansion board can be disabled) 0 = EISA ENABLE not supported (board can't be disabled)	
CFG File Extension Revision Level			Total bytes = 2, Offset = 06h
Byte 0	= Minor revision level (0 if no CFG file extension exists)		
Byte 1	= Major revision level (0 if no CFG file extension exists)		
Selections			Total bytes = 26, Offset = 08h
Byte 0	= First selection		
Byte 1	= Second selection		
...			
Byte 25	= 26th selection		

Function Information		Total bytes = 1, Offset = 022h
Byte 0	bit 7 0 = Function is not disabled 1 = Function is disabled bit 6 CFG extension as free-form data bit 5 Port initialization entry follows bit 4 Port range entry follows bit 3 DMA entry follows bit 2 Interrupt (IRQ) entry follows bit 1 Memory entry follows bit 0 Type/subtype ASCII string entry follows	
Type and Subtype ASCII String		Total bytes = 80, Offset = 23h
Byte 0	= First character of ASCII string	
Byte 1	= Second character of ASCII string	
...		
Byte 79	= 80th character of ASCII string	
Freeform Data		Total bytes = 2 to 205, Offset = 73h
Byte 0	= Length of following data block	
Byte 1	= First byte of freeform data	
...		
Byte 204	= 204th byte of freeform data	

4.17.2 Write Function Configuration Data Block

In the Write CMOS subfunction, a slot with a single function has the following configuration data block structure (note that you do not include the offset value, and another block, Function Length, is added):

Four-Byte Compressed ID			Total bytes = 4
Byte 0	bit 7	Reserved = 0	
	bits 6-2	Character 1	
	bits 1-0	Character 2	
Byte 1	bits 7-5	Character 2	
	bits 4-0	Character 3	
Byte 2	bits 7-4	First hex digit of product number	
	bits 3-0	Second hex digit of product number	
Byte 3	bits 7-4	Third hex digit of product number	
	bits 3-0	One-digit product revision number	
ID and Slot Information			Total bytes = 2
Byte 0	bit 7	0 = No duplicate ID is present 1 = Duplicate is present	
	bit 6	0 = ID is readable 1 = ID is not readable	
	bits 5,4	Slot type 00 = Expansion slot 01 = Embedded device 10 = Virtual device 11 = Reserved	
	bits 3-0	Numeric identifier for duplicate CFG filenames (IDs) 0000 If no duplicate CFG filenames 0001 First duplicate CFG file ... 1111 15th duplicate CFG file	
Byte 1	bit 7	0 = Configuration is complete 1 = Configuration is not complete	
	bits 6-2	Reserved	
	bit 1	1 = EISA IOCHKERR supported 0 = EISA not IOCHKERR supported	
	bit 0	1 = EISA ENABLE supported (expansion board can be disabled) 0 = EISA ENABLE not supported (board can't be disabled)	
CFG File Extension Revision Level			Total bytes = 2
Byte 0	= Minor revision level (0 if no CFG file extension exists)		
Byte 1	= Major revision level (0 if no CFG file extension exists)		

Function Length		Total bytes = 2
Byte 0, 1	= Length of the following function entry Length does not include these two bytes or the checksum at the end of CMOS.	
Selections		Total bytes = 2 to 27
Byte 0	Length of the following selections filed	
Byte 1	= First selection	
Byte 2	= Second selection	
...		
Byte 26	= 26th selection	
Function Information		Total bytes = 1
Byte 0	bit 7 0 = Function is not disabled 1 = Function is disabled bit 6 CFG extension as free-form data bit 5 Port initialization entry follows bit 4 Port range entry follows bit 3 DMA entry follows bit 2 Interrupt (IRQ) entry follows bit 1 Memory entry follows bit 0 Type/subtype ASCII string entry follows	
Type and Subtype ASCII String		Total bytes = 2 to 81
Byte 0	= Length of the following ASCII string filed	
Byte 1	= First character of ASCII string	
Byte 2	= Second character of ASCII string	
...		
Byte 80	= 80th character of ASCII string	

Memory Configuration		Total bytes = 63
Byte 0	= Memory configuration byte	
bit 7	0 = Last entry 1 = More entries follow	
bit 6	Reserved	
bit 5	0 = Not shared memory 1 = Shared memory	
bits 4,3	Memory type 00 = Base or extended 01 = Expanded 10 = Virtual 11 = Other	
bit 2	Reserved	
bit 1	0 = Not cached 1 = Cached	
bit 0	0 = Read only (ROM) 1 = Read/write (RAM)	
Byte 1	= Memory data size	
bits 7-4	Reserved	
bits 3,2	Decode size 00 = 20 01 = 24 10 = 32 11 = Reserved	
bits 1,0	Data size (access size) 00 = BYTE 01 = WORD 10 = DWORD 11 = Reserved	
Byte 2	= LSByte memory start address (divided by 100h)	
Byte 3	= Middle byte memory start address	
Byte 4	= MSByte memory start address	
Byte 5	= LSByte memory size (divided by 400h)	
Byte 6	= MSByte memory size (0 in this context means 64 MB)	
Interrupt Configuration		Total bytes = 2 to 14
Byte 0		
bit 7	0 = Last entry 1 = More entries follow	
bit 6	0 = Not shared 1 = Shared	
bit 5	0 = Edge-triggered 1 = Level-triggered	
bit 4	Reserved	
bits 3-0	Interrupts (0-F)	
Byte 1	= Reserved	

DMA Channel Description			Total bytes = 2 to 8
Byte 0	bit 7	0 = Last entry 1 = More entries follow	
	bit 6	0 = Not shared 1 = Shared	
	bits 5-3	Reserved (0)	
	bits 2-0	DMA channel number (0-7)	
Byte 1	bits 7,6	Reserved (0)	
	bits 5,4	DMA timing 00 - Default timing (ISA compatible) 01 - Type A timing 10 - Type B timing 11 - Type C timing (burst mode)	
	bits 3,2	Transfer size 00 - 8-bit (byte) transfer 01 - 16-bit (word) transfer 10 - 32-bit (dword) transfer 11 - Reserved	
	bits 1,0	Reserved (0)	
Port Information			Total bytes = 3 to 60
Byte 0	bit 7	0 = Last entry 1 = More entries follow	
	bit 6	0 = Not shared 1 = Shared	
	bit 5	Reserved	
	bits 4-0	Number of ports (minus 1) 00000 = 1 port 00001 = 2 sequential ports ... 11111 = 32 sequential ports	
Byte 1	= LSByte I/O port address		
Byte 2	= MSByte I/O port address		

Initialization Data		Total bytes = 4 to 60
Byte 0	= Initialization type	
bit 7	0 = Last entry 1 = More entries follow	
bits 6-3	Reserved (0)	
bit 2	Port value or mask value 0 = Write value to port 1 = Use mask and value	
bits 1,0	Type of access 00 = Byte address 01 = Word address 10 = Dword address 11 = Reserved	
If byte 0, bit 2 = 0 (no mask):		
bits 1,0	Port width to write	
	00 - byte 3 = port value	
	01 - byte 3 = LSB of port value - byte 4 = MSB of port value	
	10 - byte 3 = LSB of port value - byte 4 = Second byte port value - byte 5 = Third byte of port value - byte 6 = MSB of port value	
	11 - Reserved	
If byte 0, bit 2 = 1 (use mask):		
bits 1,0	Number of bytes/port value/mask	
	00 - byte 3 = port value - byte 4 = port mask (byte)	
	01 - byte 3 = LSB of port value - byte 4 = MSB of port value - byte 5 = LSB mask (word) - byte 6 = MSB mask (word)	
	10 - byte 3 = LSB of port value - byte 4 = Second byte of port value - byte 5 = Third byte of port value - byte 6 = MSB of port value - byte 7 = LSB mask (dword) - byte 8 = Second byte of port mask (dword) - byte 9 = Third byte of port mask (dword) - byte 10 = MSB mask (dword)	
	11 Reserved	
Byte 1	= LSByte of port I/O address	
Byte 2	= MSByte of port I/O address	

For expansion boards with more than one function, you would expand the data block with something like this:

Configuration Data for 2nd function	Function length
.	.
.	.
.	.
Configuration Data for 3rd function	Function length
.	.
.	.
.	.
Configuration Data for <i>n</i> th function	Function length
.	.
.	.
.	.
Configuration File Checksum	Total bytes = 2
Byte 1 = MSByte of configuration file checksum	
Byte 0 = LSByte of configuration file checksum	

When the freeform data bit is set in the function information byte (bit 6), the 320 byte data structure has the following specific format:

Four-Byte Compressed ID			Total bytes = 4
Byte 0	bit 7	Reserved = 0	
	bits 6-2	Character 1	
	bits 1-0	Character 2	
Byte 1	bits 7-5	Character 2	
	bits 4-0	Character 3	
Byte 2	bits 7-4	First hex digit of product number	
	bits 3-0	Second hex digit of product number	
Byte 3	bits 7-4	Third hex digit of product number	
	bits 3-0	One-digit product revision number	

ID and Slot Information		Total bytes = 2
Byte 0	bit 7 0 = No duplicate ID is present 1 = Duplicate is present bit 6 0 = ID is readable 1 = ID is not readable bits 5,4 Slot type 00 = Expansion slot 01 = Embedded device 10 = Virtual device 11 = Reserved bits 3-0 Numeric identifier for duplicate CFG filenames (IDs) 0000 If no duplicate CFG filenames 0001 First duplicate CFG file ... 1111 15th duplicate CFG file	
Byte 1	bit 7 0 = Configuration is complete 1 = Configuration is not complete bits 6-2 Reserved bit 1 1 = EISA IOCHKERR supported 0 = EISA not IOCHKERR supported bit 0 1 = EISA ENABLE supported (expansion board can be disabled) 0 = EISA ENABLE not supported (board can't be disabled)	
CFG File Extension Revision Level		Total bytes = 2
Byte 0	= Minor revision level (0 if no CFG file extension exists)	
Byte 1	= Major revision level (0 if no CFG file extension exists)	
Selections		Total bytes = 2 to 27
Byte 0	Length of the following selections filed	
Byte 1	= First selection	
Byte 2	= Second selection	
...		
Byte 26	= 26th selection	
Function Information		Total bytes = 1
Byte 0	bit 7 0 = Function is not disabled 1 = Function is disabled bit 6 CFG extension as free-form data bit 5 Port initialization entry follows bit 4 Port range entry follows bit 3 DMA entry follows bit 2 Interrupt (IRQ) entry follows bit 1 Memory entry follows bit 0 Type/subtype ASCII string entry follows	

Type and Subtype ASCII String		Total bytes = 2 to 81
Byte 0	= Length of the following ASCII string filed	
Byte 1	= First character of ASCII string	
Byte 2	= Second character of ASCII string	
...		
Byte 80	= 80th character of ASCII string	
Freeform Data		Total bytes = 2 to 205
Byte 0	= Length of following data block	
Byte 1	= First byte of freeform data	
...		
Byte 204	= 204th byte of freeform data	

The 8K EISA CMOS data structure is as follows:

Byte 0 ~ 0Dh	Reserved
Byte 0Eh	Counter for writing routine with each slot
Byte 0Fh	Checksum byte
...	
Byte 10h, 11h	Data length of slot 0
Bytes 12h, 13h	Data pointer of slot 0
Byte 14h, 15h	Data length slot 1
Bytes 16h, 17h	Data pointer of slot 1
...	
Byte 10Ch, 10Dh	Data length slot 63
Bytes 10Eh, 10Fh	Data pointer of slot 63
Bytes 110h ~ 1FFFh	Part of saving data

NOTES: *Before you write into CMOS, your program must call subfunction 02h to clear all of CMOS. Call a writing routine for each slot. Since there is no slot number parameter in the subfunction Write CMOS, include an internal counter in the writing routine. When you exit the writing routine for a slot, increment byte 0 for the next slot.*

Byte 0Fh is a checksum byte for all 8K CMOS. You must do checksum once for every write routine.

To avoid conflicts, we suggest that you make your coding as independent as possible.

4.1	Non-maskable Interrupt (INT 02h)	1
4.2	Print Screen (INT 05h)	1
4.3	System Timer Hardware Interrupt (INT 08h)	2
4.4	Keyboard Hardware Interrupt (INT 09h)	2
4.5	Diskette Hardware Interrupt (INT 0Eh)	2
4.6	Video I/O (INT 10h).....	2
4.6.1	VGA BIOS Test Functions	3
4.6.2	VGA BIOS Data Areas	5
4.6.3	Set Display Mode	7
4.6.4	Set Cursor Type	7
4.6.5	Set Cursor Position	8
4.6.6	Read Cursor Position	8
4.6.7	Read Light Pen Position	8
4.6.8	Select Active Display Page (Valid only for Alpha modes)	8
4.6.9	Scroll Window Up	8
4.6.10	Scroll Window Down	9
4.6.11	Read Attribute/Character at Current Cursor Position	9
4.6.12	Write Attribute/Character at Current Cursor Position	9
4.6.13	Write Character Only at Current Cursor Position.....	9
4.6.14	Set Color Palette	10
4.6.15	Write Dot.....	10
4.6.16	Read Dot.....	10
4.6.17	Write Character as Teletype to Active Page.....	10
4.6.18	Current Video State	11
4.6.19	Write String	17
4.7	Equipment Determination (INT 11h).....	18
4.8	Memory Size Determination (INT 12h)	18
4.9	Diskette I/O (INT 13h, Part 1)	18
4.9.1	Reset Diskette Drive	19
4.9.2	Read Status	19
4.9.3	Diskette Read	20
4.9.4	Diskette Write	20
4.9.5	Diskette Verify.....	20
4.9.6	Format Diskette Track	21
4.9.7	Diskette Drive Parameters.....	21
4.9.8	Read DASD Type	21
4.9.9	Disk Change Line Status	22
4.9.10	Set DASD Type for Format.....	22
4.9.11	Set Media Type for Format	22
4.10	Fixed Disk I/O (INT 13h, Part 2)	23
4.10.1	Disk Reset.....	24
4.10.2	Read Status	24
4.10.3	Read Disk	25
4.10.4	Write Disk	25
4.10.5	Verify Disk Sectors.....	25
4.10.6	Format Disk Track	26

4.10.7	Disk Drive Parameters	26
4.10.8	Initialize Disk Parameters	26
4.10.9	Disk Read Long	27
4.10.10	Disk Write Long	27
4.10.11	Disk Seek.....	27
4.10.12	Disk Alternate Reset	28
4.10.13	Disk Ready Test.....	28
4.10.14	Disk Recalibrate.....	28
4.10.15	Disk Diagnostics	28
4.10.16	Read DASD Type	28
4.11	RS232 I/O (INT 14h).....	29
4.11.1	Initialize the Communication Port	29
4.11.2	Send a Character.....	30
4.11.3	Receive a Character	30
4.11.4	Return the Communication Port Status	31
4.11.5	Extended Initialization	31
4.11.6	Extended Communications Port Control.....	31
4.12	System Service Routines (INT 15h).....	32
4.12.1	Cassette I/O Functions (null)	32
4.12.2	Keyboard Intercept (null).....	33
4.12.3	Device Open (null)	33
4.12.4	Device Close (null).....	33
4.12.5	Program Termination (null)	33
4.12.6	Event Wait	34
4.12.7	Joystick Support.....	34
4.12.8	System Request Key Pressed (null)	34
4.12.9	Wait.....	35
4.12.10	Move Block	35
4.12.11	Extended Memory Size Determination	35
4.12.12	Switch Processor to Protected Mode.....	35
4.12.13	Device Busy (null)	36
4.12.14	Interrupt Complete (null)	36
4.12.15	Return System Configuration Parameters	36
4.12.16	Pointing Device	37
4.12.17	Acer Extended Function (Int 15h)	39
4.13	Keyboard I/O (INT 16h).....	40
4.13.1	Read Next Character	41
4.13.2	Read Buffer Status.....	41
4.13.3	Return Shift Status.....	41
4.13.4	Set Typematic Rate and Delay	42
4.13.5	Place ASCII/Scan Code in Keyboard Buffer	42
4.13.6	Extended Read Interface for the Enhanced Keyboard	43
4.13.7	Extended Buffer Status for the Enhanced Keyboard	43
4.13.8	Return Extended Shift Status for the Enhanced Keyboard.....	43
4.14	Printer I/O (INT 17h)	44
4.14.1	Print Character.....	44
4.14.2	Initialize Printer Port.....	44

4.14.3	Read Printer Status.....	45
4.15	System Boot (INT 19h)	45
4.16	Clock Services (INT 1Ah)	45
4.16.1	Read the System Timer Count	46
4.16.2	Set the System Timer Count.....	46
4.16.3	Read the Real-Time Clock Time	46
4.16.4	Set the Real-Time Clock Time.....	47
4.16.5	Read the Real-Time Clock Date	47
4.16.6	Set the Real-Time Clock Date	47
4.16.7	Set the Real-Time Clock Alarm	48
4.16.8	Reset the Real-Time Clock Alarm	48
4.16.9	PCI BIOS Present	48
4.16.10	Find PCI Device	49
4.16.11	Find PCI Class Code	49
4.16.12	Generate Special Cycle	50
4.16.13	Read Configuration Byte	50
4.16.14	Read Configuration Word	50
4.16.15	Read Configuration Dword.....	51
4.16.16	Write Configuration Byte	51
4.16.17	Write Configuration Word	52
4.16.18	Write Configuration Dword.....	52
4.16.19	Reserved.....	52
4.17	Configuration Data Block Structure.....	53
4.17.1	Read Function Configuration Data Block.....	53
4.17.2	Write Function Configuration Data Block.....	60