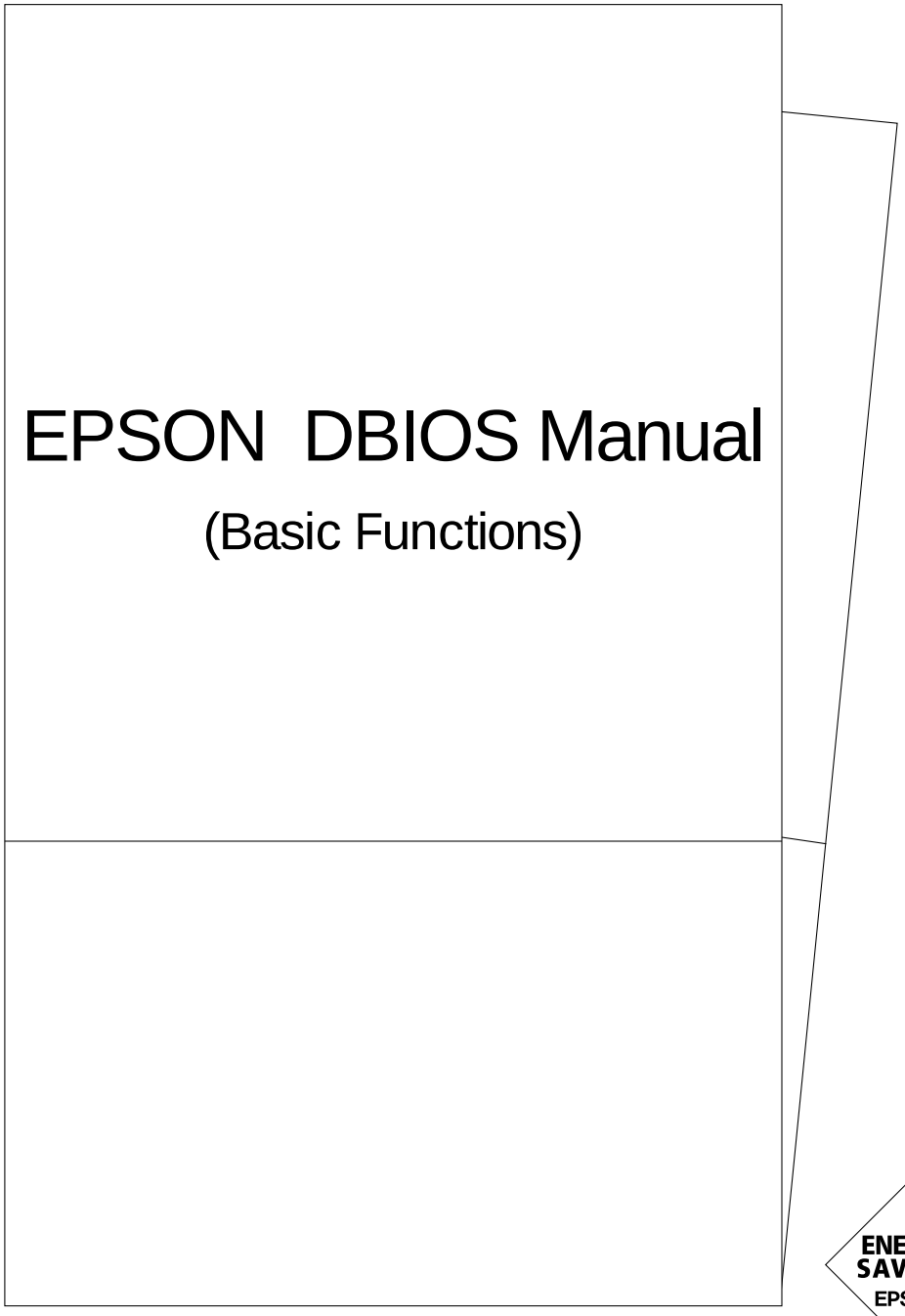
MF1059-02

EPSON

EPSON DBIOS Manual

(Basic Functions)



ENERGY SAVING EPSON

NOTICE

No part of this material may be reproduced or duplicated in any form or by any means without the written permission of Seiko Epson. Seiko Epson reserves the right to make changes to this material without notice. Seiko Epson does not assume any liability of any kind arising out of any inaccuracies contained in this material or due to its application or use in any product or circuit and, further, there is no representation that this material is applicable to products requiring high level reliability, such as, medical products. Moreover, no license to any intellectual property rights is granted by implication or otherwise, and there is no representation or warranty that anything made in accordance with this material will be free from any patent or copyright infringement of a third party. This material or portions thereof may contain technology or the subject relating to strategic products under the control of the Foreign Exchange and Foreign Trade Control Law of Japan and may require an export license from the Ministry of International Trade and Industry or other approval from another government agency.

Table of Contents

1

Basic Functions of EPSON DBIOS

Table of Contents

SPECIAL ROM BIOS LOCATIONS

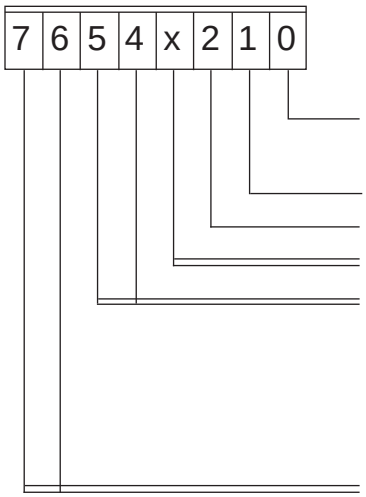
Address	Bytes	Function
F000:F900	60h	The copyright notice is stored here. This string is terminated by 00h.
F000:FFFE	1	Machine type: 0FCh (386/286). This byte is set to maintain compatibility between different manufacturers.
F000:FFFF	1	Checksum of ROM BIOS

BIOS WORK AREA

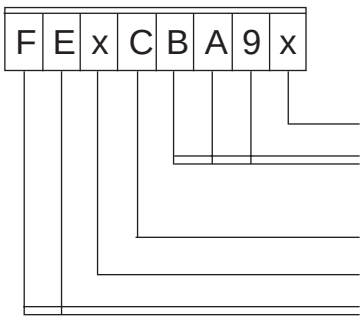
General data area

Address	Bytes	Function
0040:0000	8	Base address of the RS232 ports. Two bytes each are used as the base addresses of RS232 ports 0 through 3. When the value stored here is 0000h, no RS232 devices are connected.
0040:0008	6	Base address of the printer ports. Two bytes each are used as the base addresses of printer ports 0 through 2. When the value stored here is 0000h, no printers are connected.
0040:000E	2	Segment address of extended BIOS data area.

0040:0010 2 Installed hardware code



1 = IPL diskette installed
(there are diskette drives in the system.)
1 = Numeric coprocessor installed
1 = Pointing device installed
(Reserved)
Initial video mode
00 = EGA/VGA
01 = 40x25 using color card
10 = 80x25 using color card
11 = 80x25 B/W using mono card
Number of diskette drives
00 = 1, 01 = 2
only if IPL diskettes installed. (bit 0=1)



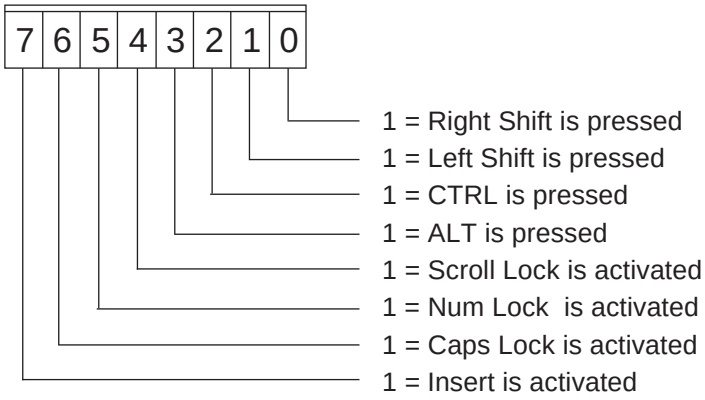
(Reserved)
Number of RS-232 ports installed
00 = 0, 01 = 1
1 = game I/O attached
(not used)
Number of printer ports installed
00 = 0, 01 = 1...

0040:0013 2 RAM memory size (in KB).

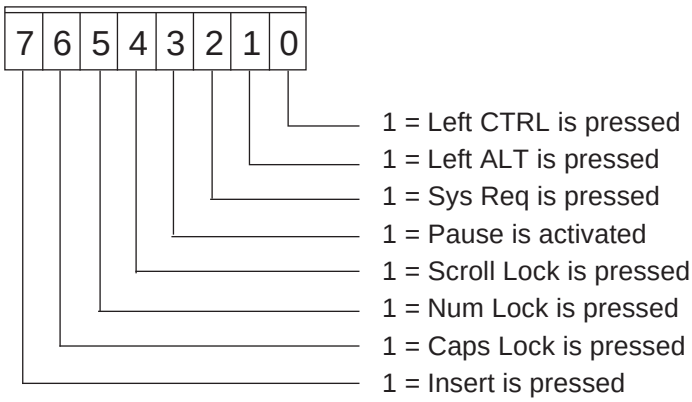
0040:0015 1 Timer 0 interrupt counter
The timer 0 count is retained.

Keyboard data area 1

Address	Bytes	Function
0040:0017	1	Keyboard control flag



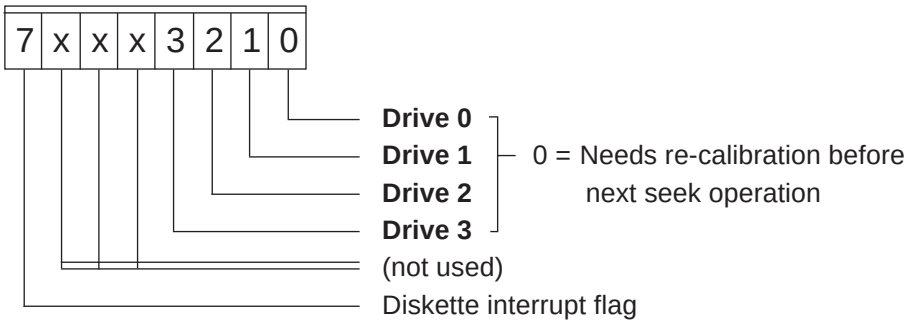
0040:0018	1	Keyboard control flag



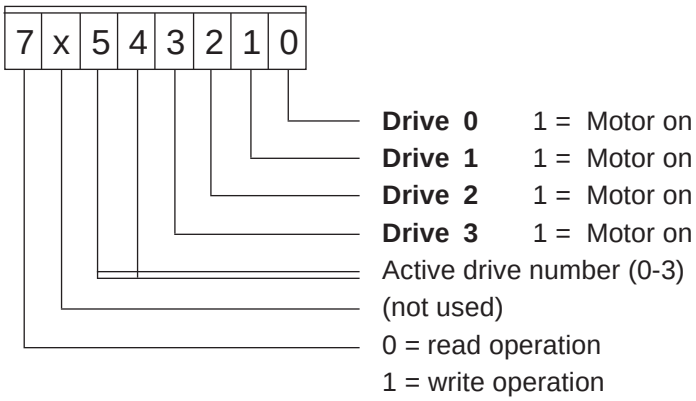
0040:0019	1	Accumulator for alternate keypad entry: The accumulator to add up numbers in ALT shift state.
0040:001A	2	Pointer to top of keyboard buffer: This points to the top character in the keyboard buffer.
0040:001C	2	Pointer to tail of keyboard buffer: This points to the last character in the keyboard buffer. If this variable is equal to 0040:001A, the keyboard buffer is empty.
0040:001E	32	Used for ring buffer (room for 16 characters)

Diskette data area 1

Address	Bytes	Function
0040:003E	1	Drive re-calibration status



0040:003F	1	Motor Status

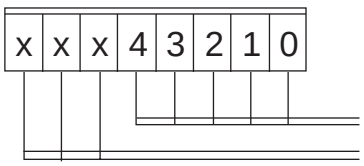


0040:0040	1	Timeout counter for drive motor turn off. This is decremented by the timer ??day?? interrupt routine. When this count reaches 0, the motor is turned off.
0040:0041	1	Last Diskette Status 080h = Timeout 040h = Seek operation failed 020h = Controller has failed 010h = Bad CRC on diskette read 00Ch = Media type not supported 009h = Data exceeded 64K boundary 008h = DMAC has failed 006h = Diskette media was changed

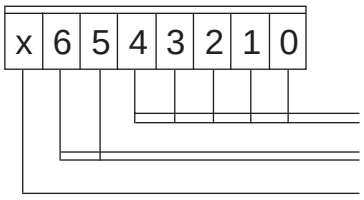
		004h = Requested sector not found 003h = Write protected 002h = Address mark not found 001h = Bad command passed to controller 000h = Operation completed successfully
0040:0042	7	DISKETTE handler use. Data from the diskette controller at the last Result Status Phase.

Video data area

Address	Bytes	Function
0040:0049	1	Current video mode.
0040:004A	2	Number of columns of current mode.
0040:004C	2	The size of the regen buffer for one page in the current video mode, in bytes.
0040:004E	2	The top address of regen buffer for current page. This value is Current page x Screen buffer length.
0040:0050	16	Cursor position for each page (maximum 8). The even bytes are column numbers and the odd bytes are row numbers.
0040:0050	2	Current cursor type (size and blink)



Cursor End Raster
(not used)

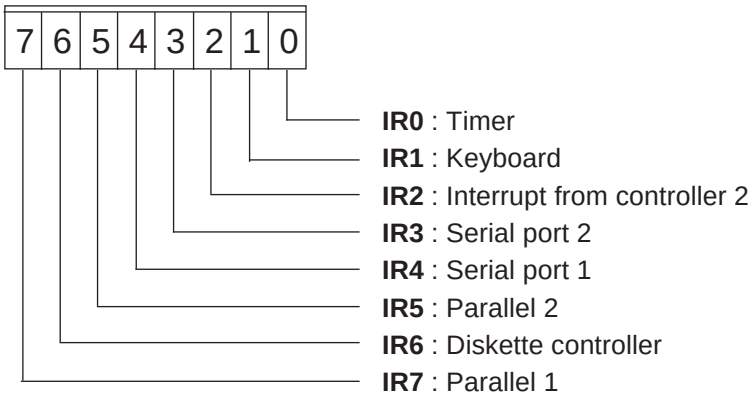


Cursor Start Raster
Blink mode (see 6845 manual)
(not used)

0040:0062	1	Current page number Determined by bits 6 and 5 in address 0040:0010.
0040:0063	2	Base address of active display card = 6845 chip I/O address. This is decided by bits 6-5 at address 0040:0010. 3B4h = monochrome 3D4h = color
0040:0065	1	Current setting of the 3x8 register = Value in I/O 3B8h or 3D8h for the current mode
0040:0066	1	Current setting of the 3x9 register = Value in I/O 3B9h or 3D9h for the current mode

POD data area

Address	Bytes	Function
0040:0067	2	Pointer to optional I/O ROM initialization routine. In INT 15h, this word used for SP saving area.
0040:0069	2	Pointer to optional I/O ROM initialization routine. In INT 15h, this word used for SS saving area.
0040:006B	1	Flag to indicate an interrupt occurred. POD saves interrupt controller #1 ISR data (INT level).



Timer data area

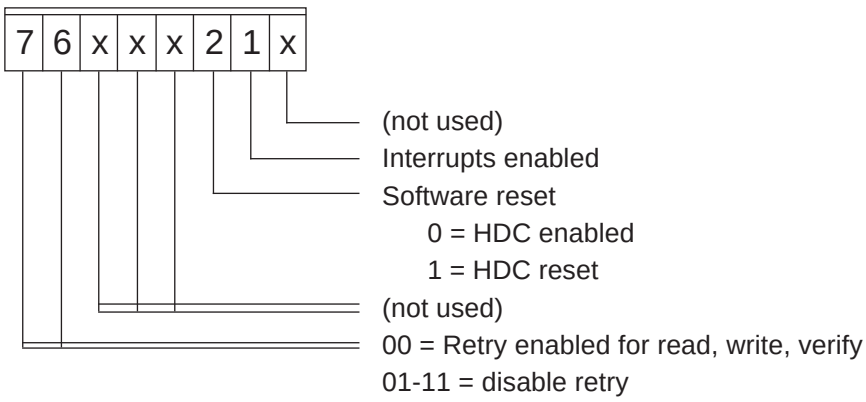
Address	Bytes	Function
0040:006C	2	Low word of timer count
0040:006E	2	High word of timer count
0040:0070	1	Flag indicates that date was updated at system bootup

System data area

Address	Bytes	Function
0040:0071	1	Break key status 00h = Break not pressed 80h = Break has been pressed
0040:0072	2	Cold/warm boot status: Data 1234h means software reset.

Hard disk data area 1

Address	Bytes	Function
0040:0074	1	Hard disk last operation status 0E0h = Status error/error reg=0 0CCh = Write fault 0BBh = Undefined error 0AAh = Drive not ready 080h = Timeout 040h = Seek operation failed 020h = Controller has failed 011h = ECC correction was applied to data 010h = Bad ECC on disk read 00Ah = Bad sector flag detected 009h = Data exceeded 64K boundary 007h = Drive parameter activity failed 005h = Reset failed 004h = Requested sector not found 002h = Address mark not found 001h = Bad command passed to controller 000h = Operation completed successfully
0040:0075	1	Maximum number of drives 00 = no drives 01 = 1, 02 = 2, ...
0040:0076	1	HDD control byte



0040:0077	1	(reserved)

Printer and RS232 time-out variables

Address	Bytes	Function
0040:0078	4	Printer time-out variables: May be changed to create different time-out wait periods. Default = 20.
0040:0078	1	Printer #1 time-out variable
0040:0079	1	Printer #2 time-out variable
0040:007A	1	Printer #3 time-out variable
0040:007B	1	Reserved
0040:007C	4	RS232C time-out variables: May be changed to obtain different time-out waits. Default = 1.
0040:007C	1	RS232C #1 time-out variable
0040:007D	1	RS232C #2 time-out variable
0040:007E	1	RS232C #3 time-out variable
0040:007F	1	RS232C #4 time-out variable

Keyboard buffer

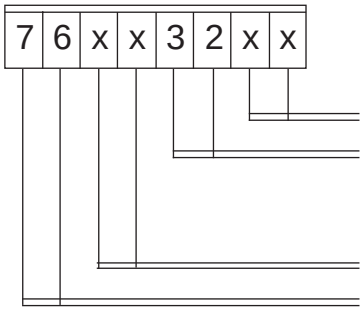
Address	Bytes	Function
0040:0080	2	The start address of keyboard buffer in segment 0040h.
0040:0082	2	The end address of keyboard buffer.

Video data area 2

Address	Bytes	Function
0040:0084	1	Number of rows for current video mode
0040:0085	2	Character height
0040:0087	1	Video Control Status
0040:0088	1	Video Control Status
0040:0089	1	Reserved
0040:008A	1	Reserved

Diskette data area 2

Address	Bytes	Function
0040:008B	1	Current and previous data rate. This value is used to retrieve the data (transfer) rate that was set most recently.



(Reserved)

Previous transfer rate

If transfer rate in 40:90 - 40:91 reaches this, retry is ended and regarded as an actual error.

(Reserved)

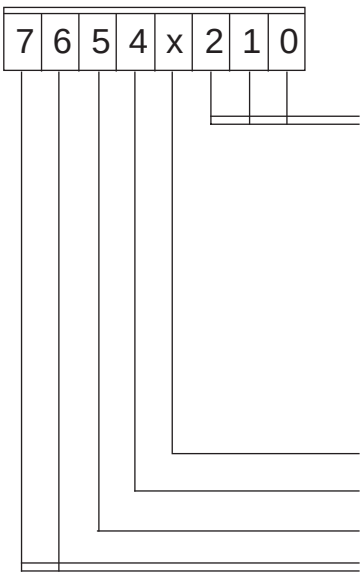
Current transfer rate.

Hard disk data area 2

Address	Bytes	Function
0040:008C	1	Current hard disk function status register: value in status register of disk controller
0040:008D	1	Current hard disk function error register: value in error register of disk controller. See error register details for bit definitions.
0040:008E	1	Hard disk interrupt flag. This flag is set by INT 76h when the hard disk controller has generated an interrupt.
0040:008F	1	Reserved

Diskette data area 3

Address	Bytes	Function
0040:0090	1	Drive 0 media state.
0040:0091	1	Drive 1 media state. Drive 0 and 1 are same format.

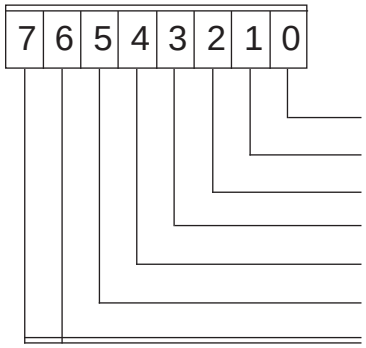


Present state
000 = 360K diskette in 360K drive / unestablished (250)
001 = 360K diskette in 1.2M drive / unestablished (300)
010 = 1.2M diskette in 1.2M drive / unestablished (500)
011 = 360K diskette in 360K drive / established (250)
100 = 360K diskette in 1.2M drive / established (300)
101 = 1.2M diskette in 1.2M drive / established (500)
110 = (reserved)
111 = Other case
(Reserved)
Drive/media status
Double stepping required
Data transfer rate for this drive
00 = 500 K pbs
01 = 300 K pbs
10 = 250 K pbs
11 = reserved

0040:0092	1	Reserved
0040:0093	1	Reserced
0040:0094	1	Drive 0 current cylinder
0040:0095	1	Drive 1 current cylinder

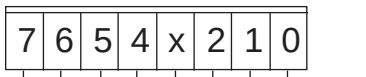
Keyboard data area 3

Address	Bytes	Function
0040:0096	1	Enhanced keyboard shift status



1 = Previous code is E1h
1 = Previous code is E0h
1 = Right CTRL is pressed
1 = Right ALT is pressed
1 = 101/102-key keyboard is attached
1 = Power on default setting is NUM lock
Used by the POD when determining whether the keyboard type is "84-key" or "101/102-key."

0040:0097	1	Keyboard LED flags



1 = Scroll LED is on
1 = Num LED is on
1 = Caps LED is on
(Reserved)
1 = ACK was received
1 = RESEND was received
1 = LED update proceeding
1 = Transmit error occurred

RTC data area

Address	Bytes	Function
0040:0098	2	Offset address of user wait flag
0040:009A	2	Segment address of user wait flag
0040:009C	2	Low word of user wait flag
0040:009E	2	High word of user wait flag
0040:00A0	1	Wait active flag

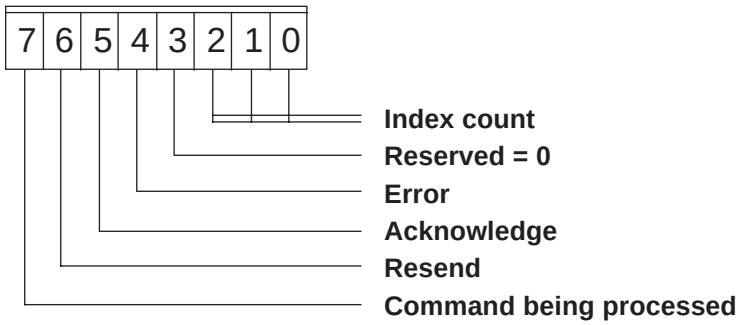
Other

Address	Bytes	Function
0040:00A1	7	Reserved for NETWORK
0040:00A8	4	Pointer to Video Parameters and Overrides
0040:00AC	84	Reserved
0050:0000	1	Print screen status: 00h = Ready 01h = Busy FFh = Error

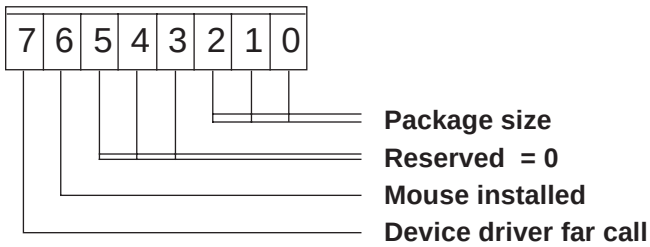
EXTENDED BIOS DATA AREA

The Power-on Diagnostics (POD) allocated the last 1K in the 640K base memory as the extended BIOS data area. 40:0E points to the extended BIOS data area segment.

Offset Address	Bytes	Function
0000	2	Size of extended BIOS data area (K units)
0002	32	Reserved
Pointing Device Area		
0022	2	Device driver far call offset
0024	2	Device driver far call segment
0026	1	Pointing device flag 1



0027	1	Pointing device flag 2



0028	7	Receive Data Buffer
002F	7	Reserved
0036	4	Reserved
003A	2	Reserved
Extend Hard Disk Parameters Area		
003D	16	Extended hard disk parameters for drive 0
004D	16	Extended hard disk parameters for drive 1
005D	16	Extended hard disk parameters for drive 2
006D	16	Extended hard disk parameters for drive 3
007D - 03FF		Reserved

CPU EXCEPTIONS

The following is a list of CPU interrupts which the ROM BIOS does not handle, or which it uses for other purposes. Some of the interrupt vectors are replaced by DOS.

INT 00h - divide error exception
INT 01h - single step (debugging exception) interrupt
INT 02h - NMI interrupt
INT 03h - break point interrupt
INT 04h - INTO detected overflow exception
INT 05h - BOUND range exceeded exception
INT 06h - invalid opcode exception
INT 07h - processor extension not available exception
INT 08h - double exception detected
INT 09h - coprocessor extension segment overrun interrupt
INT 0Ah - invalid task state segment
INT 0Bh - segment not present
INT 0Ch - stack segment overrun or not present
INT 0Dh - general protection exception

HARDWARE INTERRUPTS USED BY THE BIOS

The BIOS interrupt vector table is stored at address 0000:0000h. Each vector is a double-word pointer to an interrupt handler or a table. Some vectors and tables are user replaceable. This section lists the default routines provided by the BIOS.

INT 08h: IRQ 0 - system timer
INT 09h: IRQ 1 - keyboard interrupt handler*
INT 0Ah: IRQ 2 - cascaded interrupts from interrupt controller 2
INT 0Bh: IRQ 3 - secondary serial communication interrupt
INT 0Ch: IRQ 4 - primary serial communication interrupt
INT 0Dh: IRQ 5 - secondary parallel port
INT 0Eh: IRQ 6 - floppy-disk interrupt
INT 0Fh: IRQ 7 - primary parallel interrupt
INT 70h: IRQ 8 - real time clock (RTC) interrupt
INT 71h: IRQ 9 - redirected to 0Ah (IRQ2 handler)
INT 72h: IRQ 10
INT 73h: IRQ 11
INT 74h: IRQ 12 - Pointing device interrupt
INT 75h: IRQ 13 - numeric coprocessor error interrupt
INT 76h: IRQ 14 - hard-disk interrupt
INT 77h: IRQ 15 -

* INT09h is used to handle keyboard hardware interrupts from IRQ1. When the BIOS reads a scan code from the 8042, the scan code is set in the AL register, the carry flag is set, and then INT 15h (function 4Fh) is called. Application programs can trap INT 15h in order to examine, and in some cases modify, the original scan code generated by the keyboard. When control returns to the keyboard BIOS, the scan code is discarded if the carry flag has been reset; otherwise, the scan code is passed to the AL register and stored in the keyboard buffer as usual.

SOFTWARE INTERRUPTS USED BY THE BIOS OR SYSTEM

The following interrupt vectors are used by the BIOS as pointers to service routines and handlers.

INT 1Ch - used by timer (periodic interrupt handler) INT 8

IINT 40h - original floppy-disk handler

Points to INT 13h (original floppy-disk service routine) if a hard disk controller is installed.

INT 4Ah - alarm service

POINTERS TO TABLES

The following interrupt vectors are used by the BIOS as pointers to tables.

INT 1Dh - video parameters

This is a table of the parameters set in the CRTC. They are provided in order to ensure compatibility.

INT 1Eh - diskette base parameters

The Disk Base Table of floppy drive parameters contains the following default data: These are the settings for 3.5-inch 1.44MB media.

Offset	Value	Description
0	10101111	Step Rate (0Ah); Head-Unload Time (0Fh)
1	00000010	Head-Load Time (1); DMA Mode (0)
2	00100101	Wait time until Motor Off (37h)
3	00000010	Sector Length Code (2)
4	00010010	End Of Track - Last Sector Number (18)
5	00011011	Gap Length for Read/Write Operations (1Bh)
6	11111111	Data Transfer Length (0FFh)
7	01101100	Gap Length for Format Operation (108)
8	11110110	Format "Filler" Byte (0F6h)
9	00001111	Head Settle Time, in ms (15)
0Ah	00001000	Motor Start Time, in 1/8 s (8)
0Bh	01001111	Maximum Track Number (79)
0Ch	00000000	Transfer Rate, in K bps (500)

Notes:

1. The first byte is called the First Specify Byte, and consists of two 4-bit parameters: the Step Rate and the Head-Unload Time. The Step Rate default value is 0Ah. The Head-Unload Time default value is 0Fh.
2. The second byte is called the Second Specify Byte, and consists of the 7-bit parameter, the Head Load Time value, and a 2-bit parameter, the DMA Mode value. The default Head-Load Time is 1. The DMA Mode value is set to 0 to indicate that the DMA will be utilized in data transfer.
3. The Wait Time Until Motor Off parameter is the number of clock "ticks" (18.2/second) that the motor continues to run after a drive access has been completed. The value "37" results in a wait time of approximately 2 seconds.
4. The fourth byte contains the Sector Length Code, which is the same code used in the Format Function (see Function 05h) to indicate the sector length. The codes and values they represent are as follows:

Code	Sector Length
0	128 bytes
1	256 bytes
2	512 bytes
3	1024 bytes

5. The seventh byte is the Data Transfer Length, in bytes. This parameter is only used when the sector length is not specified.
6. The Format Filler Byte is the value that the Format function writes to the disk in lieu of data. The default value is 0F6h.

INT 1Fh - graphics mode character font

This is a pointer to the default table for the 8x8 dot graphic characters used in graphics modes. The vector points to the base of the data for character code 80h: the default address is F000: FA6E. Graphics characters consist of 8 bytes each; each byte represents a horizontal line of 8 dots.

INT 41h - hard disk parameters for drive 0

The Hard Disk Parameter Table for Hard Disk 0 contains the following 16 bytes of data:

Offset	Description
00-01	Maximum Number of Cylinders
02	Maximum Number of Heads
03-04	(not used)
05-06	Write Precompensation Cylinder
07	(not Used)
08	Controller Command Byte Bit 7 or Bit 6 = 1: Disable Retries Bit 3 = 1: Drive has more than 8 heads
09-0B	(not used)
0C-0D	Landing Zone Cylinder Number
0E	Sectors per Cylinder
0F	(reserved)

INT 46h - hard disk parameters for drive 1

INT 41h + 32 hard disk parameters for drive 2

INT 41h + 48 hard disk parameters for drive 3

The hard disk parameter tables for hard disks 1 through 3 use the same format as the table for drive 0.

BIOS FUNCTIONS

This section describes the various functions provided by BIOS. These functions can be called by the user. In addition, user routines can be used by changing the pointers in the interrupt vector table.

INT 05h - PRINT SCREEN

This interrupt is called by INT 05h, and uses no flags or registers.

INT 05h

INPUT
(None)

OUTPUT
(None)

Notes:

1. This function is used to send the ASCII codes of the characters currently displayed on the screen to printer number 00h. This interrupt uses INT 10h to read the screen and INT 17h to send the data to the printer. The cursor position is saved by this function prior to execution, and is restored upon completion. Interrupts are enabled during the execution of this function, but a subsequent INT 05h called prior to the completion of this function will be ignored.

2. The number of screen lines to be printed can be controlled by writing the last-line number into address 0040:0084h. Printing always begins with line 0. Thus, placing a value of 04h in 0040:0084h would result in the first 5 lines of the screen being printed.

3. Low memory byte 0050:0000h contains the status of this function as follows:

Value	Meaning
00h	Most Recent Print Screen was successfully completed
01h	Print Screen currently in progress
FFh	Error occsurred during execution

4. It is sometimes advisable for the user to supply their own handler if special capabilities are required, such as using special printers or printer functions, printing screen data, or redirecting the screen data to other devices. In this case, create INT 05h for special functions in an application. If the handler needs to determine whether INT05h was called by the application, it can do so by verifying that the word prior to the instruction that was called on the stack is equal to CDh, 05h. Otherwise, the interrupt was called by the keyboard [PrtSc] routine.

INT 10H - VIDEO SERVICES

Note: The INT 10H functions are valid in systems equipped with a VGA controller.

FUNCTION 00h - SET VIDEO MODE

This function is called by INT 10h, (AH)=00h. The (AL) register must contain the appropriate video mode value. Below is a list of the video modes set with the (AL) register.

INT 10h

(AH)=00h

INPUT	
(AL) =	Set video mode

Notes:

1. Use function (AH) = 12h, (BL) = 30h to set the scan lines in Alpha-Numeric modes (400, 350 or 200).

2. The cursor is not displayed in Graphics modes.

3. The screen memory buffer is immediately cleared when the video mode is set unless bit 7 of the (AL) register is set to 1.

FUNCTION 01h - SET CURSOR SIZE

This function is called by INT 10h, (AH)=01h, and the (CX) register value determines the size of the cursor as follows:

INT 10h

(AH)=01h

INPUT	
(CH) =	Beginning scan line
(CL) =	Ending scan line

OUTPUT
(None)

Note:

1. A value of 0 represents the top line of the character matrix.

2. Only the low five bits of the (CH) and (CL) registers are valid. Bits 7 - 5 should normally be set to 0 to avoid erratic effects of this function.

3. In graphics mode, the cursor is not displayed.

4. Although the BIOS maintains a separate cursor position for each page, only one cursor type is supported for all pages.

FUNCTION 02h - SET CURSOR POSITION

This function is called by INT 10h, (AH)=02h and uses the (BH) and (DX) registers to hold the Page, Row and Column number parameters as follows:

INT 10h

(AH)=02h

INPUT	
(BH) =	Page Number
(DH) =	Row Number
(DL) =	Column Number

OUTPUT
(None)

Notes:

1. The origin (Row and Column values of 00, 00 respectively) is the upper-left corner of the display screen.

2. Although there is no cursor display in the Graphics modes, the logical cursor position is maintained for use in character I/O and is therefore kept current.

3. The value in (BH), page number, is not value checked. Therefore, if the value is larger than the allowed limit the behavior of this function may be erratic.

FUNCTION 03h - READ CURSOR POSITION AND SIZE

This function is called by INT 10, (AH)=03h and uses the (BH), (CX), and (DX) registers to return cursor parameters as follows:

INT 10h

(AH)=03h

INPUT	
(BH) =	Page Number

OUTPUT	
(CH) =	Beginning Scan Line of Cursor
(CL) =	Ending Scan Line of Cursor
(DH) =	Current Row Number of Cursor
(DL) =	Current Column Number of Cursor

Notes:

1. The Page Number (BH) is input by the user. This function then reports the cursor position and size information for that display page.

2. This function is used to read the settings for functions 01h and 02h.

FUNCTION 04h - READ LIGHT-PEN POSITION

This function is not supported. On return, the (AH) register will contain the value 00h, and the (BX), (CX) and (DX) registers will not be preserved.

FUNCTION 05h - SET/GET ACTIVE DISPLAY PAGE

This function is called by INT 10h, (AH)=05h and uses the (AL) register to store Display Page number data as follows:

INT 10h

(AH)=05h

INPUT	
(AL) =	Display Page Number

OUTPUT
(None)

Notes:

1. Display Page numbering begins with 00 for the first page. For example, for mode 2, which allows up to 8 pages, the Display Page Register Values range from 0-7.

2. This function has no effect in Graphics modes.

FUNCTION 06h - SCROLL WINDOW UP

This function is called by INT 10h, (AH)=06h and uses the (AL), (BH), (CX), and (DX) registers to store window location and scrolling parameters as follows:

INT 10h

(AH)=06h

INPUT	
(AL) =	Number of Lines to scroll (blank lines inserted)
(BH) =	Display Attribute for the blank lines inserted
(CH) =	Row Number of Upper Left corner of Window
(CL) =	Column Number of Upper Left corner of Window
(DH) =	Row Number of the Lower Right corner of Window
(DL) =	Column Number of the Lower Right corner of Window

OUTPUT
(None)

Notes:

1. The Text mode operation of this function is straightforward. In Graphics mode, the window coordinates must be given in whole-character increments, not in pixels. Check the video mode documentation to confirm the horizontal and vertical character resolution for the graphics mode to be used.

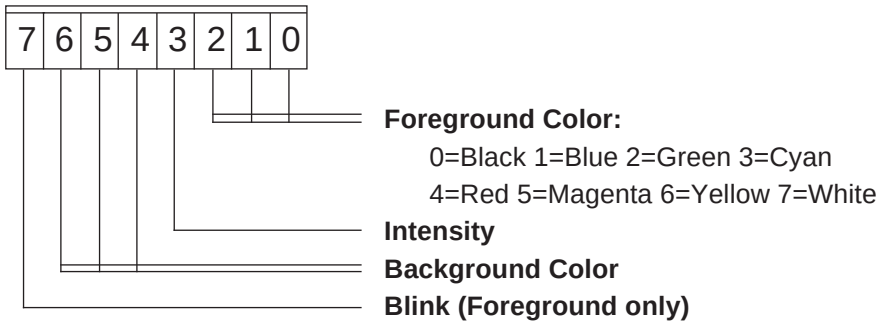
2. The window to be scrolled is defined by the Upper Left corner set by the (CX) register and the Lower Right corner set by the (DX) register.

3. Scrolling is accomplished by the insertion of (AL) blank lines (character-equivalent in Graphics modes) at the bottom of the window. Lines are inserted in the bottom of the window in this function, and in the top of the window in function 07h.

4. A value of 00 in (AL) will blank the entire window.

5. In order to avoid the display of "snow," which occurs when data is written to the VRAM during a display period, the BIOS inhibits the screen display during scrolling.

6. In Text modes, blank lines that are inserted are set with the display attributes that are set in the (BH) register. (In Graphics modes, the Display Attribute parameter is ignored.) The (BH) register value is defined as follows:



FUNCTION 07h - SCROLL WINDOW DOWN

This function is called by INT 10h, (AH)=07h and uses the (AL), (BH), (CX), and (DX) registers to store window location and scrolling parameters as follows:

INT 10h

(AH)=07h

INPUT	
(AL) =	Number of Lines to scroll
(BH) =	Display Attribute for the blank lines inserted
(CH) =	Row Number of Upper Left corner of Window
(CL) =	Column Number of Upper Left corner of Window
(DH) =	Row Number of the Lower Right corner of Window
(DL) =	Column Number of the Lower Right corner of Window

OUTPUT
(None)

Notes:

1. The Text mode operation of this function is straightforward. In Graphics mode, the window coordinates must be given in whole-character increments, not in pixels.

2. The window to be scrolled is defined by the Upper Left corner set by the (CX) register and the Lower Right corner set by the (DX) register.

3. Scrolling is accomplished by the insertion of (AL) blank lines (character-equivalent in Graphics modes) at the top of the window. Lines are inserted in the top of the window in this function, and in the bottom of the window in function 6.

4. A value of 00 in (AL) will blank the entire window.

5. In Text modes, blank lines that are inserted are set with the display attributes that are set in the (BH) register. In Graphics modes, the Display Attribute parameter is ignored.

FUNCTION 08h - READ CHARACTER AND ATTRIBUTE

This function is called by INT 10h, (AH)=08h and uses the (AX) and (BH) registers to store the page, character and attribute data as follows:

INT 10h

(AH) =08h

INPUT	
(BH) =	Display Page

OUTPUT	
(AH) =	Attribute of character at cursor position
(AL) =	ASCII code of character at cursor position

Notes:

1. The user supplies the Display Page parameter.

2. In Graphics modes the Attribute is not returned.

3. If, in Graphics modes, there is no standard ASCII character at the cursor position at the time of this interrupt the (AL) register will contain 00h.

4. For Read/Write operations in Graphics modes involving the upper-ASCII (128-255) Characters the user must initialize the pointer at interrupt 1Fh to point to the table containing the code points for ASCII Characters 128-255.

FUNCTION 09h - WRITE CHARACTER AND ATTRIBUTE

This function is called by INT 10h, (AH)=09h and uses the (AL), (BX) and (CX) registers to store the Character, Display Page, Attribute, and Iteration Count data as follows:

INT 10h

(AH)=09h

INPUT	
(AL) =	ASCII Code of Character to write
(BH) =	Display Page
(BL) =	Attribute/Foreground Color of Character
(CX) =	Number of Times to write Character

OUTPUT
(None)

Notes:

1. The Display Page is supplied by the user.

2. The Attribute parameter (BL) designates the Character Attribute in Text modes, and the Foreground Color in Graphics modes.

3. In the 256-color mode, the (BH) register value determines the background color.

4. In Graphics modes other than the 256-color mode, the Foreground Color specified in (BL) can be XOR'd on the screen by setting bit 7 in the (BL) register to 1. This is done by placing a value of 80h (128 dec.) in the (BL) register.

5. In Text modes, characters written to the screen multiple times may "wrap-around" from the far right side of the screen to the far left side of the screen (column 0), one line down.

6. There is no "wrap-around" in Graphics modes. Consequently, writing characters beyond the end of the line may produce incorrect results.

7. A value of 0 in the (CX) register will cause FFFF+1 (65,536) Characters to be written to the screen.

8. The cursor position is not changed by this interrupt. It does not advance to the end of the Characters written. (See INT 10, function E)

9. For Read/Write operations in Graphics modes (except modes 11h and 13h) involving the upper-ASCII (128-255) characters, the user must initialize the pointer at interrupt 1Fh to point to the table containing the code points for characters 128-255. For modes 11h and 13h, the character code-points for characters 128-255 are located in BIOS.

FUNCTION 0AH - WRITE CHARACTER

This function is called by INT 10h, (AH)=0Ah and uses the (AL), (BX) and (CX) registers to store Character, Display Page, Attribute, and Iteration Count data as follows:

INT 10h

(AH)=0Ah

INPUT

(AL) =	ASCII Code of Character to Write
(BL) =	Foreground Color
(BH) =	Display Page
(CX) =	Number of Times to Write Character

OUTPUT

(None)

Notes:

1. The Foreground Color parameter (BL) is only valid for Graphics modes. In Text modes, this function does not alter the current Attribute of the screen.

2. In Graphics modes other than the 256-color mode, the Foreground Color specified in (BL) can be XOR'd on the screen by setting bit 7 in the (BL) register to 1. This is done by placing a value of 80h (128 dec.) in the (BL) register.

3. The display page parameter (BH) must be set to a larger value than the maximum page number.

4. In Text modes, characters written to the screen multiple times will "wrap-around" from the far right side of the screen to the far left side of the screen (column 0), one line down.

5. There is no "wrap-around" in Graphics modes.

6. A value of 0 in the (CX) register will cause FFFFh+1 (65,536) Characters to be written to the screen.

7. In Graphics modes, writing the character multiple times may only work correctly on one line. Writing characters beyond the end of the screen line may have incorrect results.

8. The cursor position is not changed by this interrupt. It does not advance to the end of the Characters written. (See INT 10, function E)

9. For Read/Write operations in Graphics modes (except modes 11h and 13h) involving the upper-ASCII (128-255) characters, the user must initialize the pointer at interrupt 1Fh to point to the table containing the code points for characters 128-255. For modes 11h and 13h only, the character code-points for characters 128-255 are located in BIOS.

FUNCTION 0BH - SET COLOR PALETTE

This function is called by INT 10h, (AH)=0Bh and uses the (BX) register to store the Palette parameters as follows:

INT 10h

(AH)=0Bh

INPUT	
(BH) =	Color ID
(BL) =	Color/Palette To Be Used

OUTPUT
(None)

Notes:

1. This function is used to select the color palette in Graphics modes, or the border color in Text modes.

2. The (BH) register value, Color ID, determines the meaning of the (BL) register value. If the (BH) register value is 0, then the (BL) register value determines the Background color in Graphics modes 4, 5, the Foreground color in Mode 6, or the Border color in Text modes 0, 1, 2, 3. If the (BH) register value is 1, then the (BL) register value selects the Palette in Graphics modes 4 and 5. This function has no meaning in Text modes 0, 1, 2, 3 when the (BH) register has a value of 1.

Register		**Modes**		
(BH)	(BL)	0, 1, 2, 3	4, 5	6
0	xxx!!!!!	Border	Background	Foreground
1	xxxxxxx!	No effect	Palette	No effect

* The bit indicated by x is ignored

! Indicates a valid bit

3. When the (BH) register is set to 0, the (BL) register values for Text mode border colors can be 0-1Fh. (BL) register values of 10h-1Fh indicate high-intensity border colors.

   In graphics modes, when the (BH) register is set to 0, the (BL) register values for background colors can be 0-Fh. When the (BH) register is set to 1, bit 0 of the (BL) register value determines the palette in Graphics mode 4 as follows:

(BL)	Palette Colors*		
0	1=Green,	2=Red,	3=Yellow
1	4=Cyan,	5=Magenta,	6=White

* Color 0 always equals the current background color.

FUNCTION 0CH - WRITE PIXEL DOT

This function is called by INT 10h, (AH)=0Ch and uses the (AL), (BH), (DX) and (CX) registers to store the Color, Display Page, and Pixel location data as follows:

INT 10h

(AH)=0Ch

INPUT	
(AL) =	Color Code
(BH) =	Display Page
(CX) =	Column Number of Pixel
(DX) =	Row Number of Pixel

OUTPUT
(None)

Notes:

1. The origin (Column 0, Row 0) is the upper-left corner of the screen.

2. Except in the 256-color mode, the Color specified in (AL) can be XOR'd with the current color of the pixel by setting bit 7 in the (AL) register to 1. This is done by placing a value of 80h (128 dec.) in the (AL) register.

FUNCTION 0DH - READ PIXEL DOT

This function is called by INT 10h, (AH)=0Dh and uses the (AL), (BH), (DX) and (CX) registers to store Pixel Color and Location data as follows:

INT 10h

(AH)=0Dh

INPUT

(BH) =	Display Page
(CX) =	Column Number of Pixel
(DX) =	Row Number of Pixel

OUTPUT

(AL) =	Color Code

Note:

The origin (Column 0, Row 0) is the upper-left corner of the screen.

FUNCTION 0EH - WRITE CHARACTER - TELETYPE MODE

This function is called by INT 10h, (AH)=0Eh and uses the (AL) and (BL) registers to store Character, Page, and Foreground Color as follows:

INT 10h

(AH)=0Eh

INPUT	
(AL) =	ASCII Code of Character to Write
(BL) =	Foreground Color

OUTPUT
(None)

Notes:

1. This function writes characters to the active display page only.

2. In Text modes the current screen Attribute is not affected by this function.

3. The Foreground Color (BL) must be input for each Character written in Graphics modes.

4. The following ASCII Character Codes are considered Commands, not Characters. Therefore, do not supply an attribute code for them.

ASCII Code	Command
7	Beep
8	Backspace
0Ah	Line Feed
0Dh	Carriage Return

5. The Cursor is advanced one position after the Character has been written to the screen. "Wrapping" and scroll-up are also performed automatically by this function.

FUNCTION 0Fh - GET CURRENT VIDEO STATE

This function is called by INT 10h, (AH)=0Fh and uses the (AX) and (BH) registers to return the Character Columns, Mode, and Active Display Page data as follows:

INT 10h

(AH)=0Fh

INPUT
(None)

OUTPUT	
(AH) =	Number of Character Columns on-screen
(AL) =	Current Video Mode
(BH) =	Active Display Page

Note:

See Function 00 for more information on Video Modes, including the valid Display Page parameters for each mode.

FUNCTION 10h - COLOR PALETTE REGISTERS INTERFACE

This function is called by INT 10h, (AH) = 10h, and provides seven subfunctions determined by the (AL) register value, as described below.

(AL)=00h - Set Palette Register

INPUT	
(BH) =	Palette Register to Set
(BL) =	Color Value

OUTPUT
(None)

(AL)=01h - Set Overscan Register

INPUT	
(BH) =	Value

OUTPUT
(None)

(AL)=02h - Set Overscan and All Pallette Register

INPUT	
(ES) =	Segment Address of Table
(DX) =	Offset Address of Table

OUTPUT
(None)

Notes:

1. These palette registers are the 16 colors available on-screen in the EGA compatible modes.

2. The table consists of 16 bytes of data representing the color values for the 16 palette registers. Following this is a one-byte value for the overscan register.

(AL)=03h - Intensity/Blinking Toggle

INPUT	
(BL) =	00h: Intensity Bit Enabled 01h: Blinking Enabled

OUTPUT
(None)

(AL)=07h - Get Palette Register Value

INPUT	
(BH) =	Palette Register to Read

OUTPUT	
(BL) =	Color Value

(AL)=08h - Read Overscan Register

INPUT
(None)

OUTPUT	
(BH) =	Value

(AL)=09h - Read Overscan and All Palette Register Values

INPUT	
(ES) =	Segment Address of Buffer
(DX) =	Offset Address of Buffer

OUTPUT
(ES:DX) are preserved

Note:

The table consists of 16 bytes of data representing the color values for the 16 palette registers. Following this is a one-byte value for the overscan register.

(AL)=10h - Set Color Palette Register

INPUT	
(BX) =	Color Palette Register Number
(DH) =	Red Color Level Value
(CH) =	Green Color Level Value
(CX) =	Blue Color Level Value

Note:

This function is used to set a color into one of the 256 color palette registers. The (DH), (CH), and (CL) registers should be loaded with the red, green and blue (respectively) color levels. Only the first 6 bits of these registers is significant.

(AL)=12h - Set Block of Color Palette Registers

INPUT	
(ES:DX) =	Pointer to Color Values Table
(BX) =	Starting Color Palette Register
(CX) =	Number of Color Palette Registers to Set

OUTPUT
(None)

Note:

This function is used to load the color values for a number of color palette registers at the same time. The user defines a table of 18-bit color values, where the first 6 bits define the red value, the second 6 bits the green value, and the final 6 bits the blue value. The pointer to this table is loaded in the (ES:DX) register. The table must contain one 18-bit color value for every register indicated by the (CX) register value. The color palette registers set through this function must be a contiguous block of registers, but the block may be of any length (1 - 256) and may start anywhere in the set of 256 color palette registers.

(AL)=13h - Select Active Palette Register Page

(BH)=00h - Set Paging Mode

INPUT

(BL) =	00h - 4 pages of 64 palette registers
	01h - 16 pages of 16 palette registers

OUTPUT

(BH) =	None

(BH)=01h - Page Select

INPUT

(BL) =	Active Palette Register Page

OUTPUT

(BH) =	None

Notes:

1. The display controller contains 256 color palette registers, each of which consists of a user-definable 18-bit color value. For those display modes that display less than 256 colors on screen simultaneously, this function provides a way choose which "page" of the color palette register set is active. Pages can be defined as containing either 16 or 64 color palette registers.

2. In the Page Select subfunction, (BH)=01h, the (BL) register value parameters depend on the paging mode selected, i.e. 00h - 04h, or 00h - 0Fh.

3. This function is invalid with the 256-color mode, which is capable of displaying up to 256 colors on screen simultaneously.

(AL)=15h - Read Color Palette Register

INPUT

(BX) =	Color Palette Register to be Read

OUTPUT

(DH) =	Red Level Value
(CH) =	Green Level Value
(CL) =	Blue Level Value

(AL)=17h - Read Block of Color Palette Registers

INPUT	
(ES:DX) =	Pointer to Destination Address of Values
(BX) =	Starting Color Palette Register
(CX) =	Number of Color Palette Registers to Read

OUTPUT
(ES:DX) is preserved

Note:

This function is the read-equivalent of the set-block function. The block of (CX) color palette registers starting with color palette register (BX) is read, and the resulting values stored in a table at the address given in (ES:DX).

(AL)=1Ah - Get Active Palette Register Page

INPUT
(None)

OUTPUT	
(BH) =	Active Palette Register Page
(BL) =	Paging Mode

Note:

This function is the read equivalent to function (BH)=13h.

(AL)=1Bh - Convert Block of Color Values to Gray Scales

INPUT	
(BX) =	Starting Color Palette Register
(CX) =	Number of Palette Registers To Convert

OUTPUT
(None)

Note:

This function converts the red, green and blue color values into gray scale values using a weighted-sum method. The value in each register in the affected block is recalculated by assigning a weight to each color: red 30%, green 59%, and blue 11%. The original values in the color palette registers are not preserved by this function.

FUNCTION 11H - LOAD CHARACTER GENERATOR

This function is called with INT 10h, (AH)=11h, and is used to load character font tables into the VRAM buffer area and the character generator. This function is comprised of subfunctions to load characters stored in ROM or user-defined characters for use in graphics and alphanumeric modes. Subfunctions 00h - 14h are for use in alphanumeric modes; subfunctions 20h - 24h are for use in graphics modes. In addition, subfunctions 00h - 04h are used when switching between fonts of like size (e.g. a user-defined 8 x 14 font and the 8 x 14 font table defined in ROM). Subfunctions 10h - 14h loads the character sets and also initializes the video controller to match the new font, and should be used when changing between character sets of different size. Each of the subfunctions is described below.

(AL)=00h -Load User-Defined Alphanumeric Font

INPUT

(ES:BP) =	Pointer to User's Font Table
(BH) =	Bytes Per Character
(BL) =	Destination Block in VRAM Buffer
(CX) =	Number of Characters to Load
(DX) =	First Character's Offset Into User's Table

OUTPUT

(None)

Notes:

1. The (BL) value, destination block, will be 00 through 07.

2. This function will load a font table into the VRAM buffer, but the font does not become active until loaded into the character generator using subfunction (AL)=03h. Only two font tables (a maximum of 512 characters) can be loaded into the character generator.

(AL)=Load 8 x 14 Font from ROM

INPUT

(BL) =	Destination Block in Video Buffer

OUTPUT

(None)

(AL)=02h - Load 8 x 8 Double-Dot Font from ROM

INPUT

(BL) = Destination Block in Video Buffer

OUTPUT

(None)

(AL)=03h - Load Block Into Character Generator

INPUT

(BL) = Video Buffer Block(s) to Load

OUTPUT

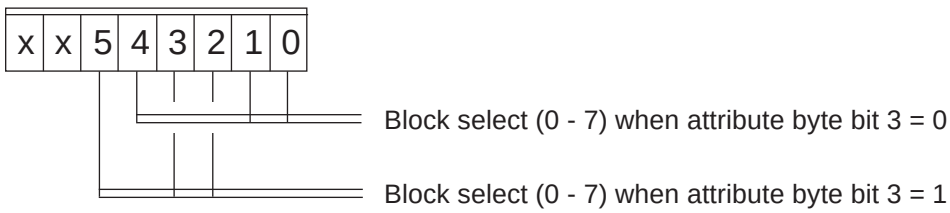
(None)

Note:

The (BL) register value determines which buffer blocks become active. The binary value of bits 4, 1 and 0 determine which block (0 - 7) is active when the character attribute byte bit 3 (intensity) is 0. When the attribute byte bit 3 is equal to 1, bits 5, 3 and 2 are used to select the active block (0 - 7). A 512-character set requires two blocks. This function allows the attribute byte bit 3 to be used to switch between the two blocks. The attribute byte bit 3 does not function as an intensity toggle in this case.

If the two groups of bits in the (BL) register (4, 1, 0 and 5, 3, 2) are the same, then only one block is selected and the attribute byte bit 3 is used to toggle intensity on and off as usual.

(BL) register



(AL)=04h - Load 8 x 16 Font from ROM

INPUT	
(BL) =	Destination Block in Video Buffer

OUTPUT
(None)

Note:

This subfunction loads the 8 x 16 font from ROM into the VRAM buffer block designated by the (BL) value.

Subfunctions (AL)=10h through (AL)=14h load a character set into the VRAM buffer and initialize the video controller to match the font. These subfunctions initialize the maximum scan line, cursor start, cursor finish and underline location registers in the CRTC. In addition, the active page is page 0, and the bytes/character, regen buffer length and row length are recalculated.

The following subfunctions should be called only after a Mode Set (INT 10, (AH)=00h) call.

(AL)=10h - Load User-Defined Alphanumeric Font

INPUT	
(ES:BP) =	Pointer to User's Font Table
(BH) =	Bytes Per Character
(BL) =	Destination Block in VRAM Buffer
(CX) =	Number of Characters to Load
(DX) =	First Character's Offset Into User's Table

OUTPUT
(None)

Notes:

1. The (BL) value, destination block, will be 00 through 07.

2. This function will load a font table into the VRAM buffer, but the font does not become active until loaded into the character generator using subfunction (AL)=03h. Only two font tables (a maximum of 512 characters) can be loaded into the character generator.

(AL)=11h - Load 8 x 14 Font from ROM

INPUT

(BL) =	Destination Block in Video Buffer

OUTPUT

(None)

(AL)=12h - Load 8 x 8 Double-Dot Font from ROM

INPUT

(BL) =	Destination Block in Video Buffer

OUTPUT

(None)

(AL)=14h - Load 8 x 16 Font from ROM

INPUT

(BL) =	Destination Block in Video Buffer

OUTPUT

(None)

Note:

This subfunction loads the 8 x 16 font from ROM into the VRAM buffer block designated by the (BL) value.

The following subfunctions are valid for graphics modes only, and should be called only after a Mode Set (INT 10, (AH)=00h) call.

(AL)=20h - Load User-Defined 8 x 8 Graphics Font

INPUT

(ES:BP) =	Pointer to User's Font Table

OUTPUT

(None)

Notes:

1. This subfunction is used in CGA-compatible graphics modes 4, 5, and 6 only.

2. In graphics modes, font tables are not stored in VRAM, nor is the character generator used. Instead, the character-definition points are used directly from VRAM. This subfunction sets the INT 1Fh pointer to the address in (ES:BP). The INT 1Fh pointer points to the upper 128 8 x 8 characters for modes 4, 5, and 6. Therefore, the pointer must point to character 80h (128). Note that to change pointers to the entire 256 character set, this subfunction and the following subfunction must both be used.

(AL)=21h - Load User-Defined Character Font

INPUT

(ES:BP) =	Pointer to User's Font Table
(BL) =	Number of Rows for Screen: 00 = use value in (DL) 01 = 14 02 = 25 03 = 43
(CX) =	Bytes per Character
(DL) =	Screen Rows, if (BL) = 00

OUTPUT

(None)

Notes:

1. This subfunction sets the INT 43h pointer to point at a user-defined font table. In modes 4, 5, and 6, this subfunction must be used to change the font definition for the first 128 characters, i.e., beginning with character 00h.

2. In all other graphics modes, this subfunction changes the pointer to the entire table of 256 characters.

(AL)=22h - Set 43h Pointer to 8 x 14 ROM Font Table

INPUT
(BL) = Number of Rows for Screen

OUTPUT
(None)

Note:

This subfunction calls subfunction 21h, with the (CX) value set to 14, and the (ES:BP) pointer set to the 8 x 14 font table in ROM.

(AL)=23h - Set 43h Pointer to 8 x 8 ROM Font Table

INPUT
(BL) = Number of Rows for Screen

OUTPUT
(None)

Note:

This subfunction calls subfunction 21h, with the (CX) value set to 8, and the (ES:BP) pointer set to the 8 x 8 font table in ROM.

(AL)=24h - Set 43h Pointer to 8 x 16 ROM Font Table

INPUT
(BL) =Number of Rows for Screen

OUTPUT
(None)

Note:

This subfunction calls subfunction 21h, with the (CX) value set to 16, and the (ES:BP) pointer set to the 8 x 16 font table in ROM. (This function does not work with modes 4, 5, and 6.)

(AL)=30 - Get Font Information

INPUT	
(BH) =	Font Pointer to Return: 00h = Current INT 1Fh pointer 01h = Current INT 43h pointer 02h = 8 x 14 ROM Font Pointer 03h = 8 x 8 ROM Font Pointer (default INT 43h pointer) 04h = 8 x 8 ROM Font Pointer (default INT 1Fh pointer) 05h = 9 x 14 ROM Font Pointer 06h = 8 x 16 ROM Font Pointer 07h = 9 x 16 ROM Font Pointer

OUTPUT	
(ES:BP) =	Pointer to Font Table
(CX) =	Bytes per Character
(DL) =	Current Number of Rows for Screen

FUNCTION 12H - EXTENDED FUNCTIONS

This function is used to call several subfunctions that provide various support services. Note that subfunctions for function 12h are called by (BL) register values, and not (AL) register values as in other functions. When the routine returns, the value 12h is placed in the (AL) register if the function is supported.

(BL)=10h - Get EGA Information

INPUT

(None)

OUTPUT

(BH) = 00h - Current Mode is Color
01h - Current Mode is Monochrome

(BL) = Amount of VRAM:
00h = 64K bytes
01h = 128K bytes
02h = 192K bytes
03h = 256K bytes

(CH) = Adapter Bits

(CL) = Switch Setting

(BL)=20h - Use Alternate Print Screen Function

INPUT

(None)

OUTPUT

(None)

(BL)=30h - Set Scan Lines For Text Modes

INPUT

(AL) = Scan Lines:
00h = 200
01h = 350
02h = 400

OUTPUT

(AL) = 12h (if supported)

(BL)=31h - Load Default Palette On Mode Change

INPUT	
(AL) =	0: = Load Default Palette on Mode Change
	1: = No Change To Color Palette Registers on Mode Change

OUTPUT	
(AL) =	12h (if supported)

(BL)=32h - Enable/Disable Video

INPUT	
(AL) =	0: = Video Enabled
	1: = Video Disabled

OUTPUT		
(AL)	=	12h (if supported)

Note:

This function will enable/disable the video buffer and I/O port address decoding for the currently active display.

(BL)=33h - Automatic Gray Scale Conversion

INPUT	
(AL) =	0: Enable Automatic Gray Scale Conversion
	1: Disable Automatic Gray Scale Conversion

OUTPUT	
(AL) =	12h (if supported)

Note:

This subfunction, when enabled, will automatically convert the color values in the color palette registers to gray scales (using the method described in function 10h, subfunction 1Bh) upon any mode change or color palette access (i.e. function 00h or 10h).

(BL)=34h - Enable/Disable Cursor Emulation

INPUT

(AL) =	0: Enable Cursor Emulation
	1: Disable Cursor Emulation

OUTPUT

(AL) =	12h (if supported)

Note:

This subfunction, when enabled, will modify the values in the Set Cursor Size function (01h) to match the currently loaded character font.

(BL)=35h - Active Display Switch

(AL)=00h - Switch I/O Channel Adapter Video Off

INPUT

(ES:DX) =	Pointer to SAVESTATE Table Destination

(AL)=01h - 01h - Switch System Video Adapter On

(AL)=02h - Switch Active Display Off

INPUT

(ES:DX) =	Pointer to SAVESTATE Table Destination

(AL)=03h - 03h - Switch Inactive Display On

INPUT

(ES:DX) =	Pointer to Display's SAVESTATE Table

Note:

This function provides a means to switch between two video controllers, the built-in video controller and an adapter on the I/O channel, when their use of the BIOS RAM data area and/or hardware resources conflict.

The SAVESTATE table is 128 bytes of information that contains the current state of the video adapter to be switched off. When an adapter is to be reactivated, the pointer in (ES:BX) should point to that adapter's table so that the adapter will resume functioning as it had prior to being disabled.

(BL)=36h - Switch Video Screen On/Off

INPUT

(AL) =	00h - Switch Screen Off
	01h - Switch Screen Off

OUTPUT

(AL) =	12h (if supported)

FUNCTION 13h - WRITE STRING

This function is called by INT 10h, (AH)=13h and uses the (AL), (BX), (CX), (DX), (ES), and (BP) registers to store the Subfunction, Display Page, Attribute, Cursor Position, String Length, and String Address data as follows:

INT 10h

(AH)=13h

INPUT	
(AL) =	Subfunction (see below)
(BH) =	Display Page Number
(BL) =	Attribute
(CX) =	Number of Characters (see below)
(DH) =	Row Number of Starting Cursor Position
(DL) =	Column Number of Starting Cursor Position
(ES) =	Segment Address of Beginning of String
(BP) =	Offset Address of Beginning of String

OUTPUT
(None)

Notes:

1. This function writes a string of characters onto the display in a “teletype-like” manner. There are four subfunctions for this service which determine how the String will be interpreted and whether or not the cursor position will change when this service is called. The Subfunctions are designated by the (AL) register value as follows:

   (AL)=00h
   Cursor is NOT moved. String is interpreted as CHAR, CHAR, CHAR, CHAR, etc. (BL) is the Attribute value used.

   (AL)=01h
   Cursor IS moved to the position following the last character written by this function. String is interpreted as CHAR, CHAR, CHAR, CHAR, etc. (BL) is the Attribute value used.

   (AL)=02h
   Cursor is NOT moved. String is interpreted as CHAR, ATTR, CHAR, ATTR, etc. (BL) is ignored.

   (AL)=03h
   Cursor IS moved to the position following the last character written by this function. String is interpreted as CHAR, ATTR, CHAR, ATTR, etc. (BL) is ignored.

2. The following ASCII Character Codes are considered Commands, not Characters. Therefore, do not supply an attribute code for them.

ASCII Code	Command
7	Beep
8	Backspace
Ah	Line Feed
Dh	Carriage Return

FUNCTION 1Ah - GET/SET DISPLAY AND ADAPTER COMBINATION CODE

This function is called by INT 10h, (AH)=1Ah, and uses the (AL), (BL), and (BH) registers to return or set the CRT and display adapter type code.

INT 10h

(AH)=1Ah

(AL)=00h

INPUT	
(AL) =	Get Display and Adapter Combination Code

OUTPUT	
(AL) =	1Ah: Function Supported
(BL) =	Currently Active Display's Code
(BH) =	Secondary Display's Code

(AL)=01h

INPUT	
(AL) =	01h - Set Display and Adapter Combination Code
(BL) =	Currently Active Display's Code
(BH) =	Secondary Display's Code

OUTPUT	
(AL) =	1Ah - Function Supported

Note:

The display and adapter combination codes are defined as follows:

Code	Meaning
00h	No display attached
01h	Monochrome monitor and monochrome display adapter
02h	Color monitor and color (CGA) adapter
03h	(Reserved)
04h	Color monitor and EGA-compatible adapter
05h	Monochrome monitor and EGA-compatible adapter
06h	PGA-compatible monitor and PGA-compatible adapter
07h	Analog monochrome monitor and VGA-compatible adapter
08h	Analog color monitor and VGA-compatible adapter
09h-0Ah	(Reserved)
0Bh	Analog monochrome monitor and MCGA-compatible adapter
0Ch	Analog color monitor and MCGA-compatible adapter

FUNCTION 1Bh - GET VIDEO PERFORMANCE AND STATUS

This function is called by INT 10, (AH)=1Bh, and uses the (BX), (ES:DI) and (AL) register to store the destination buffer data as follows:

INT 10h

(AH)=1Bh

INPUT	
(ES:DI) =	Pointer to Destination Buffer (size = 40h bytes)
(BX) =	00

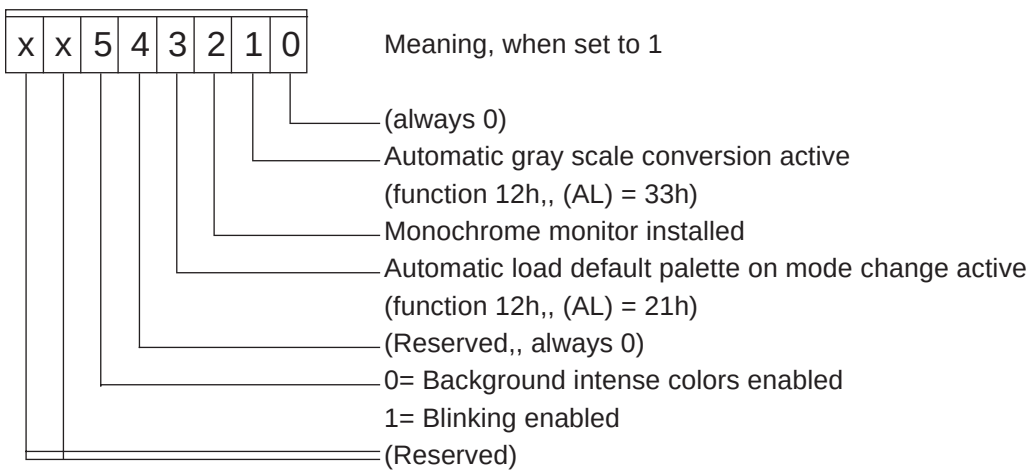
OUTPUT	
(AL) =	1Bh - Function Supported

Notes:

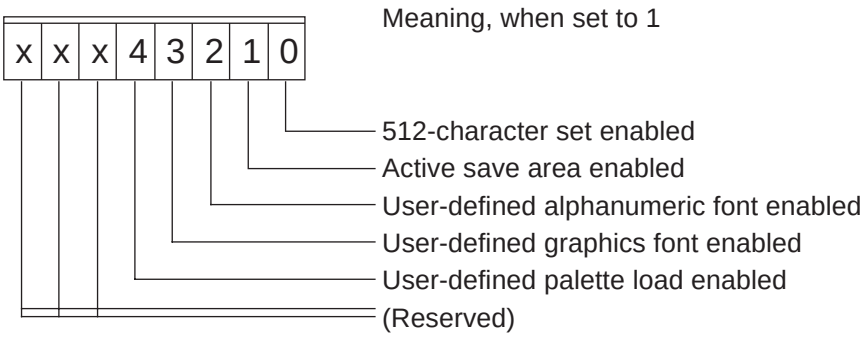
1. This function returns 40h bytes of information concerning the current status of the video system. The first 4 bytes of this table are the offset and segment addresses (respectively) for the performance characteristics of the system, for example, video modes that are supported.

2. The format of the video status table is as follows:

Offset	Bytes	Meaning
00h	2	Offset address of performance table
02h	2	Segment address of performance table
04h	1	Current video mode
05h	2	Current columns
07h	2	Size (in bytes) of the screen buffer
09h	2	Starting address of the screen buffer
0Bh	16	Position of cursor for all 8 display pages (2 bytes per page)
1Bh	2	Cursor size (1st/2nd byte = beginning/ending scan line)
1Dh	1	Currently active display page
1Eh	2	Beginning address of CRTC I/0 ports
20h	1	Current value in CRTC mode control register (03B8h for monochrome adapter; 03D8h for color adapter)
21h	1	Current value for border control register (03B9h for monochrome adapter; 03D9h for color adapter)
22h	1	Current number of rows on screen
23h	2	Number of scan lines per character
25h	1	Display and adapter combination code, currently active display (See function 1Ah)
26h	1	Display and adapter combination code, secondary display (See function 1Ah)
27h	2	Number of colors supported in current video mode
29h	1	Number of display pages supported in current video mode
2Ah	1	Total number of scan lines in current video mode 0 = 200 lines 1 = 350 lines 2 = 400 lines 3 = 480 lines Other = (Reserved)
2B-2Ch	2	(Reserved)
2Dh	1	Current state information



2E - 30h	3	(Reserved)
30h	1	VRAM available: 0 = 64K bytes 1 = 128K bytes 2 = 192K bytes 3 = 256K bytes (All other values reserved)
32h	1	Alternate video parameter tables active

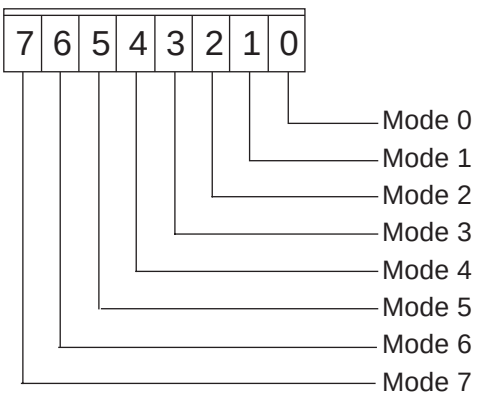


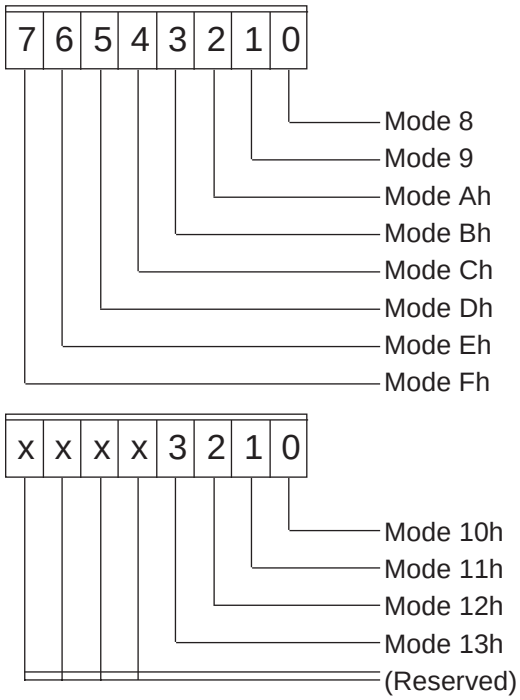
33-3Fh	13	(Reserved)

Figure 1-1. Video Status Table Format

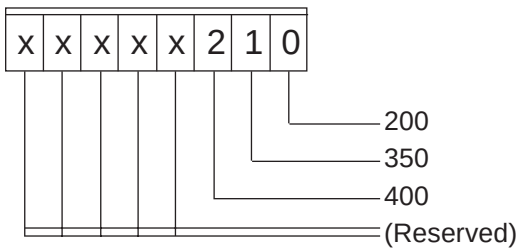
3. The format of the video performance table is as follows:

Offset	Bytes	Meaning
00h	3	Video modes supported, when set to 1





03-06h	4	(Reserved)
07h	1	Number of scan lines available in alphanumeric modes



08h	1	Number of font blocks available (in video buffer) in alphanumeric modes
09h	1	Maximum number of active font blocks (in character generator) in alphanumeric modes
0A-0Bh	2	Miscellaneous performance functions supported, when set to 1

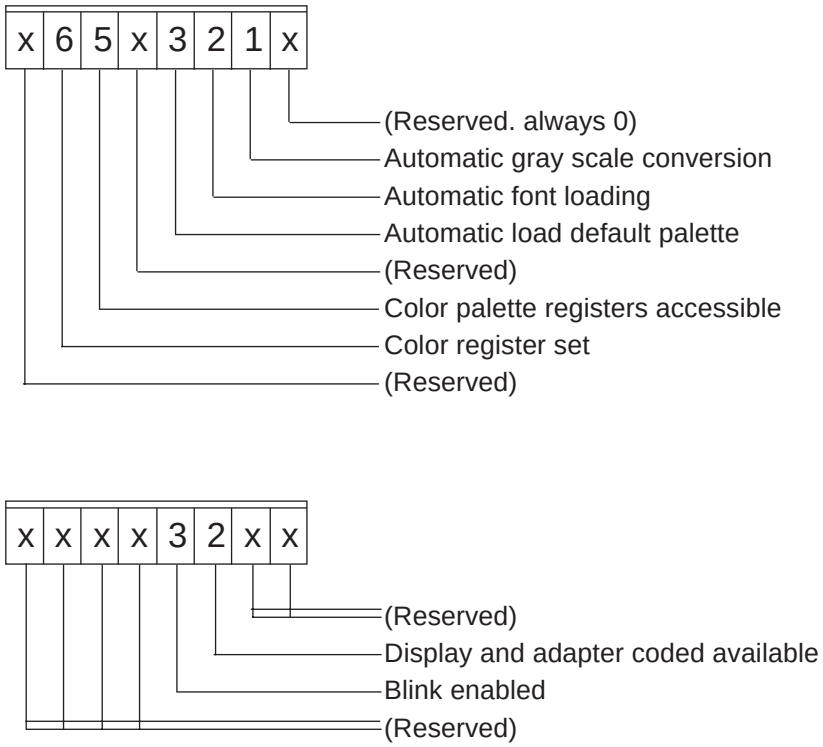


Figure 1-2. Video Performance Table Format

FUNCTION 1Ch - SAVE/RESTORE VIDEO STATE

This function is called by INT 10, (AH)=1Ch, and uses the (AL) register to call subfunctions as listed below.

INT 10h

(AH)=1Bh

(AL)=00h - Get Buffer Size

INPUT	
(CX) =	States to Save/Restore (see bit map below)

OUTPUT	
(AL) =	1Ch - Function Supported
(BX) =	Buffer Size (Number of 64K byte blocks)

(AL)=01h - Save Video State

INPUT	
(CX) =	States to Save/Restore (see bit map below)
(ES:BX) =	Pointer to Buffer

OUTPUT	
(AL) =	1Ch - Function Supported

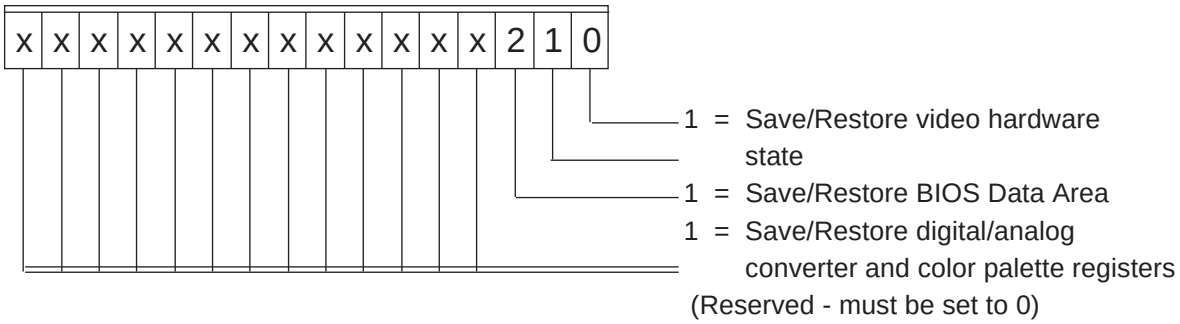
(AL)=02h - Restore Video State

INPUT	
(CX) =	States to Save/Restore (see bit map below)
(ES:BX) =	Pointer to Buffer

OUTPUT	
(AL) =	1Ch - Function Supported

Notes:

1. The (CX) register value is defined as follows:



2. The current video state is changed during a save operation. Therefore, in order to save and maintain the current video state, the save subfunction must be followed by the restore subfunction.

INT 11h - EQUIPMENT DETERMINATION

This interrupt is called by INT 11h, and returns a list of the currently-installed equipment in the (AX) register as follows:

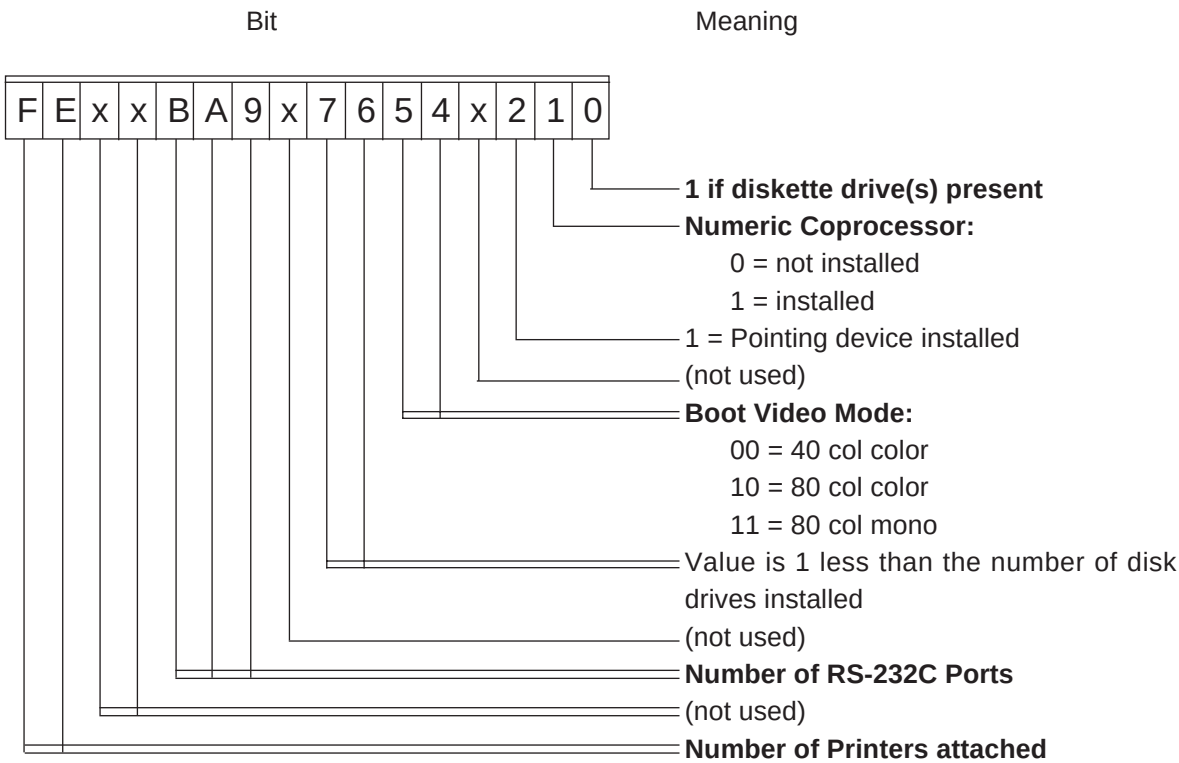
INT 11h

INPUT
(None)

OUTPUT	
(AX) =	Currently Installed Equipment List Code

Notes:

1. The (AX) register bits are set to 1 or more to indicate the presence of equipment as follows:



2. This information is determined at boot time and is stored at memory location 0040:0010h.

INT 12h - MEMORY SIZE DETERMINATION

This interrupt returns the amount of RAM installed in the system in the (AX) register.

INT 12h

INPUT	
(None)	

OUTPUT	
(AX) =	Kilobytes of base RAM installed in system

Notes:

1. This data is determined by the POD and denotes the amount of system memory available (Base RAM) after the results of the RAM test have been taken into account.

2. This data is exactly the same as that in the word at memory location 0040:0013h. This is the total base memory, after subtracting the size of the extended BIOS work area. Note, therefore, that the value returned by this function does not include any of the extended memory.
   For details on how to get the extended memory size, refer to the explanation of BIOS INT 15h function 88h.

INT 13h - DISKETTE/DISK SERVICES

This interrupt provides the BIOS diskette and disk handling services.
Functions 00h-08h and 15h-18h provide services for floppy disk drives. Functions 00h-0Dh and 10h-15h provide hard disk services. Below is a summary of the functions which are supported by each:

Function (AH)	Meaning	Floppy Drive	Hard Drive
00h	Reset Diskette/Disk System	Y	Y
01h	Get Last Disk Operation Status	Y	Y
02h	Read Sector(s)	Y	Y
03h	Write Sector(s)	Y	Y
04h	Verify Sector(s)	Y	Y
05h	Format Track/Cylinder(s)	Y	Y
06h	(not used)	–	–
07h	(not used)	–	–
08h	Return Current Drive Parameters	Y	Y
09h	Initialize HDD Parameters	N	Y
0Ah	Read ECC Sector(s)	N	Y
0Bh	Write ECC Sector(s)	N	Y
0Ch	Seek Cylinder	N	Y
0Dh	Reset Disk System (Alternate)	N	Y
0Eh	(not used)	–	–
0Fh	(not used)	–	–
10h	Test Hard Disk Drive Ready	N	Y
11h	Recalibrate Hard Disk Drive	N	Y
12h	(not used)	–	–
13h	(not used)	–	–
14h	Controller Diagnostics	N	Y
15h	Get Drive Type	Y	Y
16h	Test Diskette Change	Y	N
17h	Set Diskette Type for FORMAT	Y	N
18h	Set Media Type	Y	N

NOTE: For the above functions, returned parameters may differ depending on whether a hard disk drive or floppy disk drive is specified in the DL register. A value of 80h in this register indicates hard disk drive 0 and a value of 81h indicates hard disk drive 1. Values 00h and 01h indicate a floppy disk drive.

Status codes returned in the (AH) register

The (AH) register bits are significant to indicate the status of the latest operation as follows:

(AH)	Meaning
E0h	HDC unable to determine cause of error
CCh	Write Fault error on the selected drive
BBh	(Undefinederror)
AAh	Drive Not Ready error
80h	Time Out - disk did not respond within time limit
40h	Bad Seek - R/W Head failed to go to correct track
20h	Disk/DisketteControllermalfunction
11h	Data Error - ECC correction was applied to data
10h	Disk: Bad ECC detected during a Read Diskette: CRC error - bad data in sector
0Ah	Sector Previously Marked as a "Bad Sector"
09h	DMA Error - Data buffer straddles 64KB boundary of actual memory
08h	DMAControllermalfunction
07h	Initialization Error - Drive Parameter Table error
06h	Diskette change was detected
05h	Reset Failure Error
04h	Bad Sector - sector not found
03h	Write protect error
02h	Sector ID not found or not valid
01h	Bad command sent to Disk/Diskette Controller
00h	Normal end

NOTE: If CF is zero at completion of function, process has ended normally and AH is always zero. If CF is 1, an error has occurred and the status code is set in AH.

Definitions:

ECC Error Check and Correction
CRC Cyclic Redundancy Check
DMA Direct Memory Access

Register convention

The following register/parameter conventions are used in the diskette/disk functions.

Register	Meaning
(AH)	sector number to be read/written
(CL)	(lower 6 bits) start sector location
(CL)	(upper 2 bits) start cylinder location
and (CH)	start cylinder location (upper 2 bits are in CL register)
(DH)	head number
(DL)	00h ... first floppy drive 01h ... second floppy drive 80h ... first hard disk drive 81h ... second hard disk drive 82h ... third hard disk drive 83h ... fourth hard disk drive
(ES:BX)	buffer address location

Notes:

1. On entry, the (AL) register value is not verified. The Number of Sectors is media- and drive-type dependent, as shown by the following table:

Media Type	Drive Type	Sectors
320K	360K/1.2M	1 - 8
360K	360K/1.2K	1 - 9
1.2M	1.2M	1 - 15
720K	720K/1.44M	1 - 9
1.44M	1.44M	1 - 18
Hard Disk	Variable	Variable

   IMPORTANT: Sector numbers begin with 1, not 0. Do not request to read/write 0 Sectors.

2. The (CH) register value, cylinder number, is not value checked. It will be 0 - 39 for 320K and 360K diskettes, or 0 - 79 for 1.2M (5.25") drive and 720K/1.44M (3.5") diskettes. For hard disk drives, the value will be 0 - 1023. Because 10 bits are required to store the value 1023, the high two bits of the cylinder number are stored in the high two bits of the (CL) register.

3. The (CL) register value follows the same parameter guidelines as in Note 1 above.

4. The (DH) register value, head number, is not value checked. It should be 0 - 1 for diskette drives, or 0 - 255 for hard disk drives.

5. The (DL) register value, drive number, is value checked. It must be either 00h or 01h for diskette drives, or 80h, 81h, 82h, or 83h for hard disk drives.

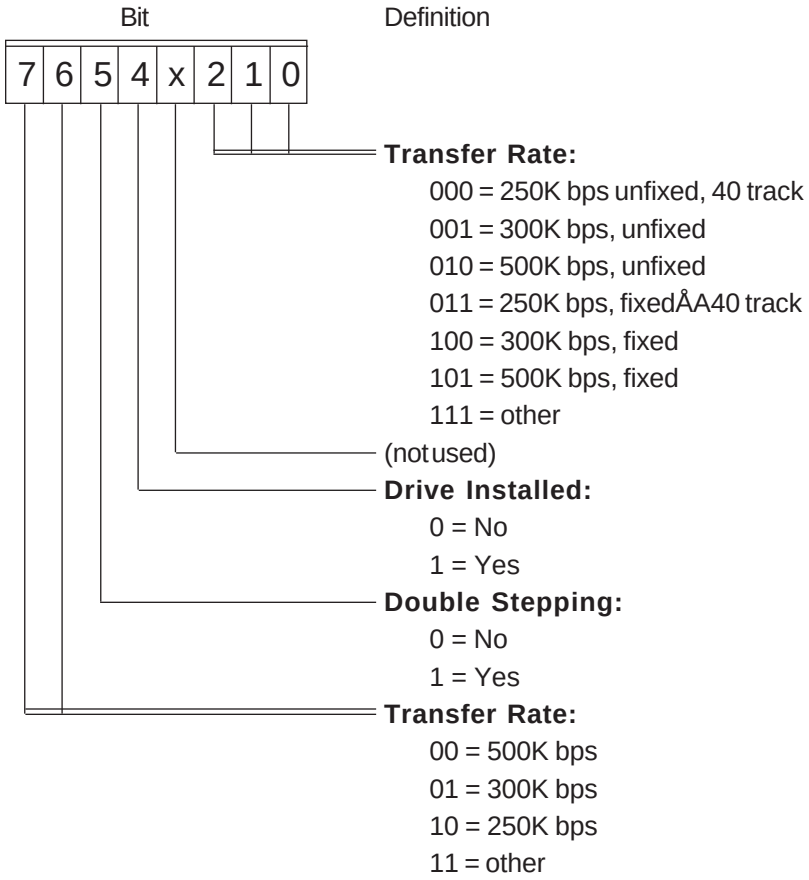
6. The input or output buffer address is entered in (ES:BX).

7. For hard disks, the highest sector number that can be read is 128 (80h) when starting at the paragraph boundary, which means that a maximum of 127 sectors can be read. Because a read request at sector 0 implies 256 sectors, a read request at sector 0 should not be issued.

Diskette drive operation parameters - The Disk Base Table

Operation of the floppy disk drives is determined by the information in an 11 byte table of parameters called the Disk Base Table. It is pointed to by the vector stored in INT 1Eh. Almost all operating systems (such as DOS and Unix) provide a dedicated disk base table, with the vector pointing to the table set at INT 1Eh.

Old DISKETTE_TYPE:



Hard Disk Drive Parameters

This BIOS can handle four IDE HDDs as listed below:

(1) HDD 0 : Primary channel master drive
(2) HDD 1 : Primary channel slave drive
(3) HDD 2 : Secondary channel master drive
(4) HDD 3 : Secondary channel slave drive

When handling an HDD through the INT 13h disk service, 80h through 83h are used as the four disk drive numbers. The combinations of HDDs that are supported and their drive numbers are shown below.

Primary	channel	Secondary	channel	Drive number			
master	slave	master	slave	80h	81h	82h	83h
×	×	×	×	--	--	--	--
O	×	×	×	PM	--	--	--
×	×	O	×	SM	--	--	--
O	O	×	×	PM	PS	--	--
×	×	O	O	SM	SS	--	--
O	×	O	×	PM	SM	--	--
O	O	O	×	PM	PS	SM	--
O	×	O	O	PM	SM	SS	--
O	O	O	O	PM	PS	SM	SS

O: Connected ×: Not connected

PM : Primary channel master drive
PS : Primary channel slave drive
SM : Secondary channel master drive
SS : Secondary channel slave drive

Hard Disk Parameter Table

A hard disk parameter table is a necessity in order to operate a hard disk drive. Although it depends on the type of HDD, the hard disk parameter table consists of the following 16 bytes of data:

HDD parameter table

Bytes	Description
00-01	Maximum number of cylinders
02	Maximum number of heads
03-04	Unused
05-06	Write compensation ??backup?? cylinder
07	Unused
08	Control command byte
09-11	Unused
12-13	Landing zone cylinder number
14	Number of sectors per cylinder
15	Reserved

An HDD parameter table is required for each HDD that is used.
Each of the HDD parameter tables are stored in the following locations:
(1) First HDD (drive 80h): Address stored in INT 41h (0:104h)
(2) Second HDD (drive 81h): Address stored in INT 46h (0:118h)
(3) First HDD (drive 82h): Address stored in INT 41h (0:104h) + 32 bytes
(4) First HDD (drive 83h): Address stored in INT 41h (0:104h) + 48 bytes

BIOS sets the following parameters beforehand.
There are four methods by which BIOS sets these parameters, according to the CMOS-RAM HDD setting.

(1) When "0" (No drives)
BIOS does not set up an HDD parameter table.
(2) When "1" (auto_detect LBA)
BIOS gets the HDD parameters by using the HDD's "identify drive" command, and creates its HDD parameters based on those values.
The HDD parameters that are produced permit use of that HDD in LBA mode.
(3) When "2" (auto_detect NORMAL)
BIOS gets the HDD parameters by using the HDD's "identify drive" command, and creates its HDD parameters based on those values.
(4) When "14" (user_define LBA)
BIOS creates the HDD parameter table based on the HDD parameters recorded in CMOS.
The HDD parameters that are produced permit use of that HDD in LBA mode.
(5) When "15" (user_define NORMAL)
BIOS creates the HDD parameter table based on the HDD parameters recorded in CMOS

The HDD type settings in the CMOS RAM area and the locations where the HDD parameters are recorded are listed below.

HDD setting	HDD parameters cylinder, sector/track, head
(1) HDD 0 : 12hbit#7-4	20h:21h, 22h, 26h#7-4
(2) HDD 1 : 12hbit#3-0	23h:24h, 25h, 26h#3-0
(3) HDD 2 : 19hbit#7-4	27h:28h, 29h, 2dh#7-4
(4) HDD 3 : 19hbit#3-0	2ah:2bh, 2ch, 2dh#3-0

FUNCTION 00h - Floppy disk/hard disk system reset

This function is called by INT 13h (AH)=00h. It resets the floppy disk controller and all floppy disk drives or the hard disk controller and all hard disk drives. The function cannot be executed while other disk operations are being carried out. Because the function causes a recalibration of the read/write heads, it affects all BIOS disk support services.

INT 13h

(AH)=00h

INPUT	
(DL)	Drivenumber

OUTPUT	
(AH)	Statuscode

Note:

It is recommended that this function be called before a retry when there has been a Read, Write, Verify, or Format error.

FUNCTION 01h - GET PREVIOUS DISK OPERATION STATUS

This function is called by INT 13h, (AH)=01h and uses the (AH) register to fetch the status byte of the previous hard or floppy disk operation as follows:

INT 13h

(AH)=01h

INPUT	
(DL) =	Drive number

OUTPUT	
(AH) =	Status Code (this function)
(AL) =	Previous Disk Operation's Status Code

Note:

The Status Code is defined at the beginning of this section (INT 13h).

FUNCTION 02h - READ DISK SECTOR(S)

This function is called by INT 13h (AH)=02h. As shown below, it uses the AL, BX, CX, DX, and ES registers to specify the drive, head, track, sector, and buffer address for reading data from a floppy disk or hard disk sector.

INT 13h

(AH)=02h

INPUT	
(AL) =	Number of Sectors to Read
(BX) =	Offset Address of Input Buffer
(CH) =	Cylinder Number (Track Number)
(CL) =	Cylinder/Sector Number
(DH) =	Head Number
(DL) =	Drive Number
(ES) =	Segment Address of Input Buffer

OUTPUT	
(AH) =	Status Code
(AL) =	Number of Sectors Read (Diskette only)
(CF) =	Error Flag: 0 = No Error 1 = Error: Code in (AH)

Note:

If an error occurs while reading a floppy disk sector with this function, the number of correctly read sectors is entered in the AL register.

FUNCTION 03h - WRITE DISK SECTOR(S)

This function is called by INT 13h (AH)=03h. As shown below, it uses the AL, BX, CX, DX, and ES registers to specify the drive, head, track, sector, and buffer address for writing data to a floppy disk or hard disk sector.

INT 13h

(AH)=03h

INPUT	
(AL) =	Number of Sectors to Write
(BX) =	Offset Address of Output Buffer
(CH) =	Cylinder Number (Track Number)
(CL) =	Cylinder/Sector Number
(DH) =	Head Number
(DL) =	Drive Number
(ES) =	Segment Address of Output Buffer

OUTPUT	
(AH) =	Status Code
(AL) =	Number of Sectors Written (Diskette only)
(CF) =	Error Flag: 0 = No Error 1 = Error: Code in (AH)

Note:

If an error occurs while writing a floppy disk sector with this function, the number of correctly written sectors is entered in the AL register.

FUNCTION 04h - VERIFY SECTOR(S)

This function is called by INT 13h (AH)=04h. As shown below, it uses the AL, BX, CX, and DX registers to specify the drive, head, track, and sector for verifying data in a floppy disk or hard disk sector.

INT 13h

(AH)=04h

INPUT	
(AL) =	Number of Sectors to Verify
(CH) =	Cylinder Number (Track Number)
(CL) =	Cylinder/Sector Number
(DH) =	Head Number
(DL) =	Drive Number

OUTPUT	
(AH) =	Status Code
(AL) =	Number of Sectors Verified (Diskette only)
(CF) =	Error Flag: 0 = No Error 1 = Error: Code in (AH)

Note:

1. Diskettes: This function will find and read a sector on a diskette and verify that the cyclic redundancy check (CRC) is valid for the requested sector(s). Note that there is no comparison between the data in memory and the data written on the diskette. Only the CRC is checked for validity with its respective sector.

2. Hard disk: This function executes an error checking and correction (ECC) routine and verifies that the resulting data is valid for the requested sector(s). Note that there is no comparison between the data in memory and the data written on the hard disk. Only the ECC is checked for validity with its respective sector.

FUNCTION 05h - FORMAT TRACK/CYLINDER

This function is called by INT 13h (AH)=05h. As shown below, it uses the AL, BX, CX, DX, and ES registers to specify the sector, pointer, cylinder, drive, and head for formatting a floppy disk or hard disk.

INT 13h

(AH)=05h

INPUT	
(AL) =	Number of Sectors Per Track
(BX) =	Offset Address of Pointer
(CH) =	Cylinder Number (Track Number)
(DH) =	Head Number
(DL) =	Drive Number
(ES) =	Segment Address of Pointer

OUTPUT	
(AH) =	Status Code
(CF) =	Error Flag: 0 = No Error 1 = Error: Code in (AH)

Notes:

1. When the (DL) value, Drive Number, is 0 or 1, use Function 17h or 18h of INT 13h to set the diskette type (DASD) before using this function.

2. Diskette formatting:
   The Pointer in ES:BX points to a table containing one 4-byte entry for each sector to be formatted. These sector attribute parameters, C, H, R, N, are used to determine logical addresses during read, write, and verify functions. They specify the following data for each sector:

C - 1st Byte	Track Number
H - 2nd Byte	Head Number
R - rd Byte	Sector Number
N - 4th Byte	Bytes per Sector: 00 = 128 bytes 01 = 256 bytes 02 = 512 bytes 03 = 1024 bytes

When formatting 320K/360K diskettes:

The Gap Length For Formatting parameter (offset 7) in the Diskette Base-Table must be changed to 050h. The Last Sector Number parameter (Offset 4) must also be changed to reflect the desired number of sectors per track. This will be 8 for 320K diskettes and 9 for 360K diskettes. Restore the initial values when the formatting procedure is completed.

The Diskette Base Table is pointed to by the vector at INT 1Eh. Most operating systems (e.g., DOS) use their own table.

3. Hard Disk Formatting:
   The Pointer in ES:BX points to a table, 512 bytes in length, that contains a pair of bytes (F, N) for each sector to be formatted. They specify the following data for each sector:

F	1st Byte - Sector Good/Bad	00h = Good Sector 80h = Bad Sector
N	2nd Byte - Sector Number	

Note that disk interleave is determined using the number of sectors entered in the table as 2-byte entries.

Hard Disk sector size is fixed at 512 bytes.

FUNCTION 08h - RETURN CURRENT DRIVE PARAMETERS

This function is called by INT 13h, (AH)=08h, and uses the (AX), (CX), and (DX) registers (or the (AX), (BX), (CX), (DX), (DI) and (ES) registers) to return the Status, Cylinder, Sector, Head, and Drive parameters as follows:

INT 13h

(AH)=08h

INPUT	
(DL) =	Drive Number

IF (DL)=80h or 81h:

OUTPUT	
(AH) =	Status Code
(CH) =	Cylinder Number Parameter Upper Limit
(CL) =	Cylinder Number/Sectors Parameter Upper Limit
(DH) =	Head Number Parameter Upper Limit
(DL) =	Number of installed drives
(CF) =	Error Flag: 0 = No Error 1 = Error: Code in (AH)

IF (DL)=00h, 01h

OUTPUT	
(AH) =	Status Code
(BL) =	Drive Type
(CH) =	Upper limit of track number parameter
(CL) =	Upper limit of track/sector number parameter
(DH) =	Upper limit of head number parameter
(DL) =	Number of Installed Drives
(DI) =	Offset Address of Parameter Table
(ES) =	Segment Address of Parameter Table
(CF) =	Error Flag: 0 = No Error 1 = Error: Code in (AH)

Notes:

In the case of a hard disk

1. If the hard disk drive whose number is specified in the (DL) register has not been installed, the (CF) flag is set to 1 and 01h is returned in the (AH) register.

In the case of a diskette

2. If the diskette drive whose number is specified in the (DL) register has not been installed, the 0 is returned for all parameters other than the number of drives, for which a correct value is returned in the (DL) register.

3. The (BL) register value indicates the drive type as follows:
   0 = No Drive
   1 = 360K Drive
   2 = 1.2M Drive
   3 = 720K Drive (3.5")
   4 = 1.44M Drive (3.5")

4. The (CH) register value designates the default number of tracks for the diskette drive. 1.2M, 720K and 1.44M drives return a value of 79. 360K drives return a value of 39.

5. The (ES:DI) register pair is a pointer to the Disk Base table as defined at the beginning of the explanation of INT 13h.

FUNCTION 09h - INITIALIZE HARD DISK CONTROLLER

This function is called by INT 13h (AH)=09h. As shown below, it uses the AH and DL registers to initialize the hard disk controller.

INT 13h

(AH)=09h

INPUT	
(DL) =	Hard Disk Drive Number

OUTPUT	
(AH) =	Status Code
(CF) =	Error Flag: 0 = No Error 1 = Error: Code in (AH)

Note:

This function writes the user-defined (or the hard disk's own) hard disk parameter table (HDPT) to the disk controller to initialize the disk drive.

A user-defined HDPT (described at the beginning of this section) can be accessed by this function if the vector pointing to it is positioned at INT 41h for hard disk drive 80h, INT 46h for hard disk drive 81h, INT 41h + 32 bytes for hard disk drive 82h, and INT 41h + 48 bytes for disk drive 83h.

FUNCTION 0Ah - READ ECC SECTOR

This function is called by INT 13h (AH)=0Ah. As shown below, it uses the AX, BX, CX, DX, and ES registers to specify the buffer pointer, cylinder, sector, head, and drive for reading data with ECC from a hard disk drive.

INT 13h

(AH)=0Ah

INPUT	
(AL) =	Number of sector to be read
(ES:BX) =	Segment:Address of Input Buffer
(CH) =	Cylinder Number
(CL) =	Cylinder/Sector Number
(DH) =	Head Number
(DL) =	Drive Number

OUTPUT	
(AH) =	Status Code
(CF) =	Error Flag: 0 = No Error 1 = Error: Code in (AH)

Notes:

1. This function reads ECC data from a hard disk. An ECC data sector contains the standard hard-disk 512K bytes, plus 4 bytes for Error Checking and Correction (ECC).

2. If Error Code 11h is returned in the (AH) register on exit from this function, this indicates that a data error was detected and corrected by the ECC. Although the data is probably good, Error Code 11h is returned to indicate a possible abnormaly. It is advisable to verify the accuracy of the data in question.

FUNCTION 0Bh - WRITE ECC SECTOR

This function is called by INT 13h (AH)=0Bh. As shown below, it uses the AX, BX, CX, DX, and ES registers to specify the buffer pointer, cylinder, sector, head, and drive for writing data with ECC to a hard disk drive.

INT 13h

(AH)=0Bh

INPUT	
(AL) =	Number of Sectors to be written
(ES) =	Segment Address of Output Buffer
(BX) =	Offset Address of Output Buffer
(CH) =	Cylinder Number
(CL) =	Cylinder/Sector Number
(DH) =	Head Number
(DL) =	Drive Number

OUTPUT	
(AH) =	Status Code
(CF) =	Error Flag: 0 = No Error 1 = Error: Code in (AH)

Notes:

1. This function writes ECC data to a hard disk. An ECC data sector contains the standard hard-disk 512K bytes, plus 4 bytes for Error Checking and Correction (ECC).

2. If Error Code 11h is returned in the (AH) register on exit from this function, this indicates that a data error was detected and corrected by the ECC. Although the data is probably good, Error Code 11h is returned to indicate a possible abnormaly. It is advisable to verify the accuracy of the data in question.

FUNCTION 0Ch - SEEK

This function is called by INT 13h, (AH)=0Ch and uses the (AH), (CX), and (DX) registers to perform a seek on a hard disk drive with the specified, sector, head, and drive data as follows:

INT 13h

(AH)=0Ch

INPUT	
(CX) =	Cylinder Number
(DH) =	Head Number
(DL) =	Drive Number

OUTPUT	
(AH) =	Status Code
(CF) =	Error Flag: 0 = No Error 1 = Error: Code in (AH)

FUNCTION 0Dh - RESET HARD DISK DRIVE

This function is called by INT 13h, (AH)=0Dh, and uses the (AH), and (DL) registers to reset a hard disk drive as follows:

INT 13h

(AH)=0Dh

INPUT	
(DL) =	Drive Number

OUTPUT	
(AH) =	Status Code
(CF) =	Error Flag: 0 = No Error 1 = Error: Code in (AH)

Note:

This function resets the hard disk controller and the hard disk drive designated in the (DL) register. All read/write heads on the designated hard disk are re-calibrated. A hardware reset is sent to the hard disk controller, setting all registers to default values and initializing the hard disk controller with respect to the Hard Disk Parameter Tables.

FUNCTION 10h - TEST DRIVE READY

This function is called by INT 13h (AH)=10h. As shown below, it uses the DL and AH registers to test the ready condition of the hard disk drive.

INT 13h

(AH)=10h

INPUT	
(DL) =	Drive Number

OUTPUT	
(AH) =	Status Code
(CF) =	Error Flag: 0 = No Error
	1 = Error: Code in (AH)

Note:

This function simply executes the error-detection routine normally executed with most of the other functions and, as with the other functions, returns the result in the (AH) register. This same error-detection routine is normally executed as the first step in most of the other hard disk functions, whereas this function allows the drive status to be tested prior to the execution of other functions.

FUNCTION 11h - RECALIBRATE HARD DISK DRIVE

This function is called by INT 13h (AH)=11h. As shown below, it uses the DL and AH registers to recalibrate a hard disk drive as follows.

INT 13h

(AH)=11h

INPUT	
(DL) =	Drive Number

OUTPUT	
(AH) =	Status Code
(CF) =	Error Flag: 0 = No Error 1 = Error: Code in (AH)

Notes:

1. This function is similar to the Reset functions (0Dh and 00) except that only the hard disk drive is affected. The hard disk controller is not reset by this function.

2. When this function is called, the drive heads are recalibrated immediately. This processing returns the head to track 0.

FUNCTION 14h - CONTROLLER SELF-DIAGNOSTICS

This function is called by INT 13h (AH)=14h. As shown below, it uses the DL and AH registers to activate the self-diagnosis function of the hard disk controller and to return the error register value and status code.

INT 13h

(AH)=14h

INPUT	
(DL) =	Drive Number

OUTPUT	
(AH) =	Status Code
(CF) =	Error Flag: 0 = No Error
	1 = Error: Error Code in (AH)

Note:

This function sends an internal diagnostics command to the hard disk controller. A value of 0 returned in the (CF) flag indicates the hard disk controller diagnostics detected no errors. If there is an error during the execution of this function, the (CF) flag will have a value of 1 on exit, and the error code can be found in the (AH) register.

FUNCTION 15h - GET DRIVE TYPE

This function is called by INT 13h, (AH)=15h, and uses the (AH), (CX), and (DX) registers to fetch the drive type, and number of sectors as follows:

INT 13h

(AH)=15h

INPUT	
(DL) =	Drive Number

IF (DL)=80h or 81h:

OUTPUT	
(AH) =	Drive Type
(CX) =	Upper word of the Number of Sectors
(CF) =	Lower word of the Number of Sectors

IF (DL)=00h or 01h:

OUTPUT	
(AH) =	Drive Type

Notes:

1. This function returns the drive type of the drive designated on entry in the (DL) register. Drive types returned by this function are set in the (AH) register as follows:

(AH) Value	Drive Type
0 =	No Drive
1 =	Floppy drive without disk-change detection
2 =	Floppy drive with disk-change detection
3 =	Hard disk drive

   Note that some Floppy disk drives, usually 1.2M, 720K, and 1.44M drives, are able to detect a diskette change. It is common to test for this feature in order to determine whether the drive is a 320/360K drive or a 1.2M drive. However, this method is not always accurate.

2. The Number of Sectors returned in the (CX) and (DX) registers applies only to hard disks. The value returned in these registers is the capacity of the drive and is given as the number of 512 byte sectors that can be stored on the disk.

FUNCTION 16h - TEST DISKETTE CHANGE

This function is called by INT 13h, (AH)=16h, and uses the (AH) and (DL) registers to specify the drive number and check the diskette change status as follows:

INT 13h

(AH)=16h

INPUT	
(DL) =	Diskette Drive Number

OUTPUT	
(AH) =	Diskette Change Status

Notes:

1. The (AH) register value, Diskette Change Status, will have a value of 00 if no diskette change was detected. If a diskette change was detected, the (AH) register will have a value of 06.

   If (CF) = 0, (AH) = 00h: DISKETTE CHANGE signal inactive (door has not been opened)
   If (CF) = 1, (AH) = 01h: Bad command (invalid drive)
   If (CF) = 1, (AH) = 06h: DISKETTE CHANGE signal active (door has been opened)

2. The (DL) register value, Diskette Drive Number, is value checked and must be either 00h or 01h.

FUNCTION 17h - SET DISKETTE TYPE

This function is called by INT 13h, (AH)=17h, and uses the (AL) and (DL) registers to specify the drive number and set the diskette type as follows:

INT 13h

(AH)=17h

INPUT	
(AL) =	Diskette Type
(DL) =	Drive Number

OUTPUT	
(AH) =	Status code

Note:

This function is used to set the diskette type for floppy disk drive functions such as Function 5 - Format. The (AL) register value depends on the diskette/drive combination as follows:

(AL) Value	Diskette/Drive Types
00 =	No Drive
01 =	320/360K diskette in a 320/360K drive
02 =	320/360K diskette in a 1.2M drive
03 =	1.2M diskette in a 1.2M drive
04 =	720K diskette in a 720K drive
05 =	720K diskette in a 1.44M drive
06 =	1.44M diskette in a 1.44M drive

FUNCTION 18h - SET MEDIA TYPE

This function uses the (AH), (CX), and (DL) registers to specify the Cylinder, Sector and Drive number and set the diskette media type as follows:

INT 13h

(AH)=18h

INPUT	
(CH) =	Number of Cylinders
(CL) =	Number of Sector
(DL) =	Drive Number

OUTPUT	
(AH) =	Status Code
(CF) =	Error Flag: 0 = No Error 1 = Error: Error Code in (AH)

Notes:

1. This function will reset the current drive specifications (DISKETTE_TYPE) to the indicated cylinders and sectors/track if these specifications pass two tests. The BIOS first attempts to match the parameters with the existing BIOS parameter table. This test serves to determine whether the media parameters requested by the BIOS are supported.

   The second test is an attempt to match the drive-type, inferred by the user-supplied parameters, with the CMOS equipment byte to determine if the drive type is actually installed in the machine. If both these tests are passed, the DISKETTE_TYPE byte is re-initialized with parameters taken from the parameter table for the drive/media combination requested.

2. See the beginning of this section on INT 13h for a description of the DISKETTE_TYPE byte.

INT 14h - SERIAL COMMUNICATION SERVICES

Port number definition

The number of serial ports is checked by the POD. In the following BIOS functions, (DX) is the "logical" port number and this is not necessarily equal to the serial port number. If there is only one serial port, it is identified as logical serial port number 0000h.
The number of serial ports is checked during the POD. In the following BIOS functions, (DX) is the "logical" port number and this is not necessarily equal to the serial port number. If there is only one serial port, it is identified as logical serial port number 0000h.

Register convention

The following registers are used in all serial communication processing in order to keep a consistent user interface when each function is to be invoked:

On Entry:

(AH)	Function code
(AL)	(See each function)
(DX)	Logical serial number

On Exit:

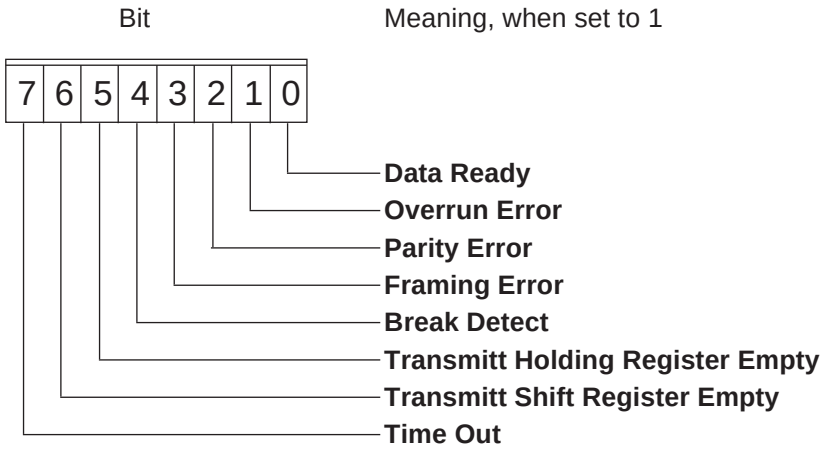
(AH)	Line status
(AL)	Modem status
(DX)	Logical serial number (Unchanged from entry)

Notes:

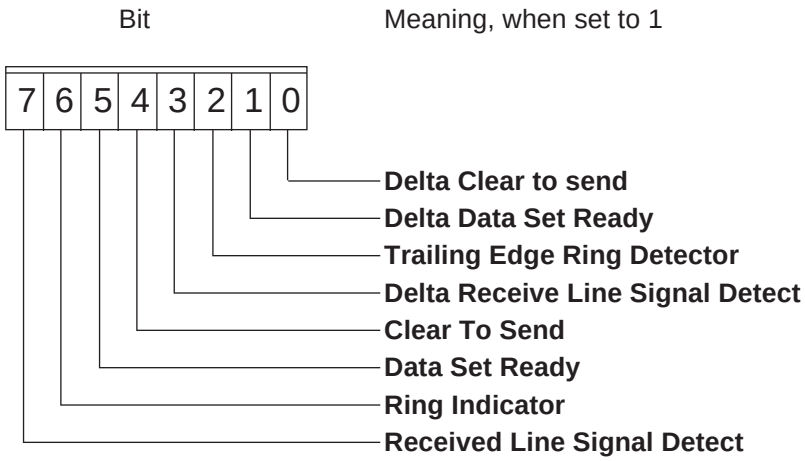
1. On entry, the (AH) register is checked. If the value is outside 00h-04h, the INT routine is terminated without flags changing and without returning the line or modem status bytes.

2. On entry, the (DX) register, (logical serial number, not serial port number) is value checked. It will be in the range 0000h-0003h. If the value is outside this range, the INT routine is terminated immediately. (The BIOS reserves areas for 4 serial ports in he RAM work area.)

3. On exit, the (AH) register contains the Line Status Code. The bits in the (AH) register are set to 1 to indicate the following:



4. On exit, the (AL) register contains the Modem Status Code. The bits of the (AL) register are set to indicate the following:



FUNCTION 00h - INITIALIZE SERIAL PORT

This function is called by INT 14h, (AH)=00h, and uses the (AX) and (DX) registers to initialize a serial port according to the serial I/O parameters as follows:

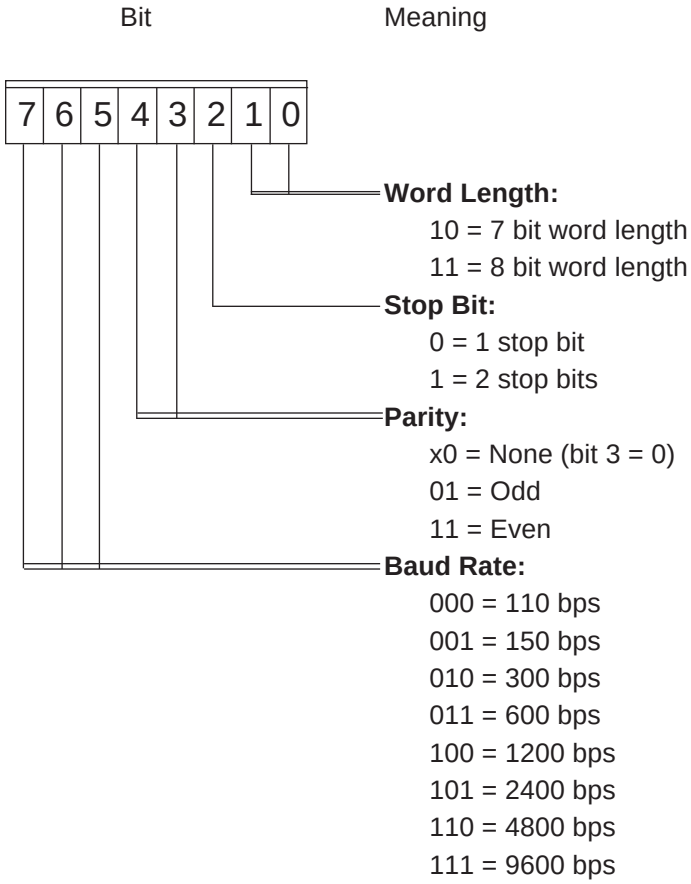
INT 14h

(AH)=00h

INPUT	
(AL) =	Serial Port I/O Parameters
(DX) =	Serial Number

OUTPUT	
(AH) =	Line Status Code
(AL) =	Modem Status Code

The (AL) register value sets the I/O parameters as follows:



FUNCTION 01h - SEND CHARACTER

This function is called by INT 14h, (AH)=01h, and uses the (AX) and (DX) registers to send character data as follows:

INT 14h

(AH)=1

INPUT	
(AL) =	ASCII Code of Character Sent
(DX) =	Serial Number (0, 1, 2 or 3)

OUTPUT	
(AH) =	Line Status Code

Notes:

1. This function sends the ASCII code designated in the (AL) register out to the logical serial port designated in the (DX) register.

2. On exit, the (AL) register is preserved.

3. On exit, the (AH) register contains the Line Status Code (as described in Note 3 of Function 0 of this interrupt). When bit 7 is 1, it indicates that the function was unable to send the character code over the line.

FUNCTION 02h - RECEIVE CHARACTER

This function is called by INT 14h, (AH)=02h, and uses the (AX) and (DX) registers to receive the character data as follows:

INT 14h

(AH)=02h

INPUT	
(DL) =	Serial Number (0, 1, 2, 3)

OUTPUT	
(AH) =	Line Status Code, Errors Only
(AL) =	ASCII Code of Character Received

FUNCTION 03h - GET SERIAL PORT STATUS

This function is called by INT 14h, (AH)=03h, and uses the (AX) and (DX) registers to fetch the line status and modem status as follows:

INT 14h

(AH)=03h

INPUT	
(DX) =	Serial Number (0, 1, 2 or 3)

OUTPUT	
(AH) =	Line Status Code
(AL) =	Modem Status Code

FUNCTION 04h - EXTENDED INITIALIZE

INT 14h

(AH)=04h

INPUT

(AL) =	Break:	
		00 = No break
		01 = Break
(BH) =	Parity:	
		00 = None
		01 = Odd
		02 = Even
		03 = Stick parity, odd
		04 = Stick parity, even
(BL) =	Stop bit:	
		00 = one
		01 = two (1.5 for 5bit word)
(CH) =	Word length, bits:	
		00 = 5
		01 = 6
		02 = 7
		03 = 8
(CL) =	Baud rate:	
		00 = 110 bps
		01 = 150
		02 = 300
		03 = 600
		04 = 1200
		05 = 2400
		06 = 4800
		07 = 9600
		08 = 19200
(DX) =	Serial Number (0, 1, 2, 3)	

OUTPUT

(AH) =	Line status code
(AL) =	Modem status code

FUNCTION 05h - EXTENDED COMMUNICATION PORT CONTROL

INT 14h

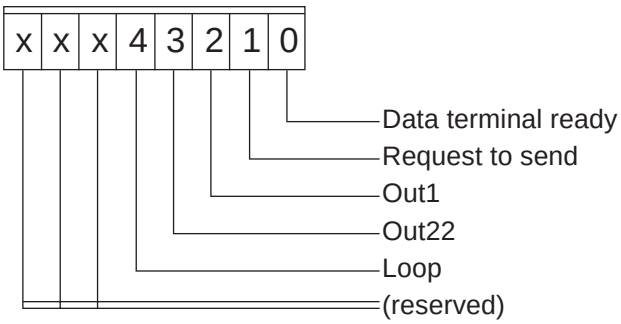
(AH)=05h

INPUT	
(AL) =	00: Read modem control register
	01: Write modem control register
(BL) =	Modem control register
(DX) =	Serial Number (0, 1, 2, 3)

OUTPUT	
(AH) =	Line Status Code
(AL) =	Modem Status Code

Notes:

1. The bit definition of the (BL) register is as follows:



INT 15h - EXTENDED BIOS SERVICES

FUNCTION 4Fh - KEYBOARD SCAN CODE INTERCEPT

This function is invoked by INT 09h handler when a keyboard scan code is read from the 8042 (keyboard controller). The INT 09h handler invokes this function after setting the scan code in register (AL) and setting the Carry flag. This allows applications to trap INT 15h to check and/or modify the scan codes generated by the keyboard. If upon return to the INT 09h handler the Carry flag has been reset, then the INT 09h handler discards the scan code. Otherwise, the scan code in register (AL) is stored in the keyboard buffer as usual.

INT 15h

(AH)=4Fh

INPUT	
(AL) =	Scan code received from keyboard
(CF) =	1

OUTPUT	
(AL) =	Scan Code
(CF) =	0 = BIOS discards scan code (AL)
	1 = BIOS uses scan code (AL)

FUNCTION 80h - OPEN DEVICEE

This function is called by INT 15h, (AH)=80h, and uses the (BX) and (CX) registers to specify the device and process identification data as follows:

INT 15h

(AH)=80h

INPUT	
(BX) =	Device ID
(CX) =	Process ID

OUTPUT	
(AH) =	00h
(CF) =	0
(IF) =	1

Note:

Reserved for future installation.

FUNCTION 81h - CLOSE DEVICE

This function is called by INT 15h, (AH)=81h, and uses the (BX) and (CX) registers to specify the device and process identification data as follows:

INT 15h

(AH)=81h

INPUT	
(BX) =	Device ID
(CX) =	Process ID

OUTPUT	
(AH) =	00h
(CF) =	0
(IF) =	1

Note:

Reserved for future installation.

FUNCTION 82h - TERMINATE PROGRAM

This function is called by INT 15h, (AH)=82h, and uses the (BX) register to specify the device identification data as follows:

INT 15h

(AH)=82h

INPUT	
(BX) =	Device ID

OUTPUT	
(AH) =	00h
(CF) =	0
(IF) =	1

Note:

Reserved for future installation.

FUNCTION 83h - EVENT WAIT

This function is called by INT 15h, (AH)=83h, and uses the (AL), (BX), (CX), (DX), and (ES) registers to specify the Set/Cancel, User's Flag Byte address, and Time Interval and await an event as follows:

INT 15h

(AH)=83h

INPUT	
(AL) =	00h -Set
	01h - Cancel
(BX) =	Offset Address of User's Flag Byte
(ES) =	Segment Address of User's Flag Byte
(CX) =	High-Order Word of Time Interval
(DX) =	Low-Order Word of Time Interval

OUTPUT	
(CF) =	1 if Event Wait Previously Set

Notes:

1. This function immediately returns to the caller. After the period of time specified in the (CX:DX) register, this function will turn on the high-order bit of the Caller's Flag Byte located at the address given in the (ES:BX) registers.

2. The time interval data is set in the (CX) and (DX) registers as the number of microseconds to wait. The resolution is 976 microseconds.

3. The (CF) flag is set to 1 if an Event Wait was already set at the time this function was attempted. The attempt will fail, the previously set Event Wait will remain in effect, and all registers will be preserved.

FUNCTION 84h - READ JOYSTICK

This function is called by INT 15h, (AH)=84h, and uses the (AX), (BX), (CX), and (DX) registers to read the switch and position data as follows:

INT 15h

(AH)=84h

IF (DX)=0000h

INPUT	
(DX) =	0000h - Read Joystick Switches

OUTPUT	
(AL) =	Switch Settings

IF (DX)=0001h

INPUT	
(DX) =	0001h - Read Joystick Position Inputs

OUTPUT	
(AX) =	X-Axis Value, Joystick 1
(BX) =	Y-Axis Value, Joystick 1
(CX) =	X-Axis Value, Joystick 2
(DX) =	Y-Axis Value, Joystick 2

Notes:

1. As indicated above, there are two subfunctions to this function. When the (DX) register contains 0000h on entry, the (AL) register is returned with the 4 high-order bits set to match the joystick(s) switch settings. Bits 7 and 6 indicate the current settings of switch 1 and 2 of Joystick 1. Likewise, bits 5 and 4 indicate the switch settings of Joystick 2.

   When the (DX) register is set to 0001h, the resistive inputs of the joysticks are returned in the (AX), (BX), (CX), and (DX) registers as indicated.

2. Only the low-order byte of the (DX) register is checked on entry.

3. The Joystick I/O port is 0201h.

4. If no Joystick port is attached this function is a dummy return. The (AH) register is returned with a value of 86h, and the (CF) flag is set to 1. All other registers are preserved.

5. When a joystick is not connected and this function is used to read the resistive inputs, the (DX) register returns a value of 0000h. If a joystick is connected the value in the (DX) register will always be greater than 0000h.

FUNCTION 85h - SYSTEM REQUEST KEY SUPPORT

This function is called by INT 15h, (AH)=85h, and uses the (AL) register to pass the Sys Req key status as follows:

INT 15h

(AH)=85h

INPUT	
(AL) =	00h - Sys Req was Pressed
	01h - Sys Req Released

OUTPUT	
(AH) =	00h
(CF) =	0
(IF) =	1

Note:

This function is called when Sys Req has been either pressed or released. The (AL) register contains a value of 00h if Sys Req was pressed. When Sys Req is released, a function call is made with (AL)=01h. If this function is not supported by the user's code, this function returns all registers intact and sets the (CF) flag to 0.

FUNCTION 86h - EXECUTION WAIT

This function is called by INT 15h, (AH)=86h, and uses the (CX) and (DX) registers to specify the time interval data and suspend execution as follows:

INT 15h

(AH)=86h

INPUT	
(CX) =	High-Order Word of Time Interval
(DX) =	Low-Order Word of Time Interval

OUTPUT	
(CF) =	1 if Event Wait previously set

Notes:

1. This function temporarily suspends execution for the length of time input in the (CX) and (DX) register pair. Once the time interval has expired, this function returns to the caller.

2. The Time Interval is specified in the (CX) and (DX) registers as the number of microseconds to wait until this function returns to the caller. The resolution is 976 microseconds.

3. This function will abort and the (CF) flag will be set to 1 if the Event Wait function, (AH)=83h, is already active.

FUNCTION 87h - MOVE MEMORY BLOCK

This function is called by INT 15h, (AH)=87h, and uses the (CX), (ES), and (SI) registers to set the block size and descriptor table pointer and move a memory block as follows:

INT 15h

(AH)=87h

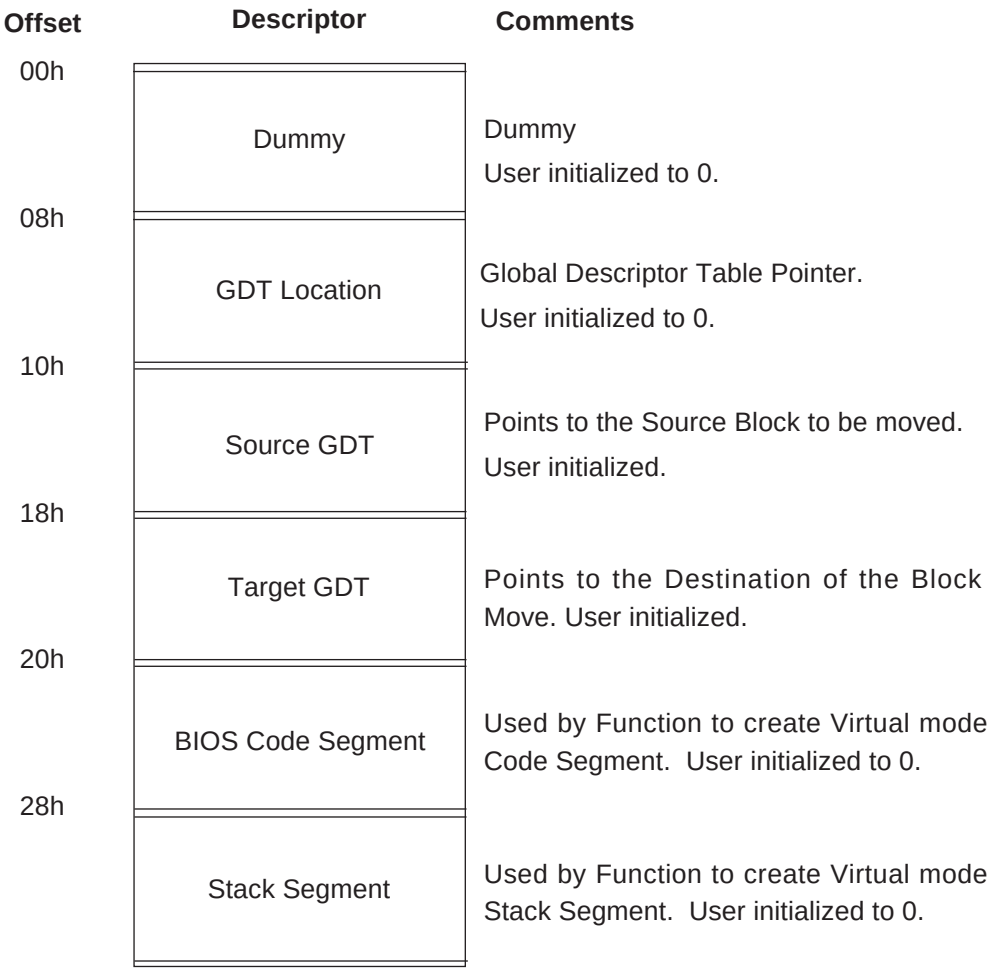
INPUT	
(CX) =	Number of Words To Move
(ES:SI) =	Segment and Offset Address of Descriptor Table

OUTPUT	
(AH) =	Status Code
(CF) =	1 = Error: Status Code in (AH) (ZF)= 0, (IF)= 0 0 = Memory Block Moved Successfully (ZF)= 1, (IF)= 1

Notes:

1. This function can be is used in Protected mode to move a block of memory into extended memory, that is, memory beyond 1M bytes. The number of words to move is entered in the (CX) register. The maximum number of words that can be moved with this function is 8000h.

2. The (ES:SI) register pair points to the start of the Descriptor Table constructed by the user. The Descriptor Table must have the following construction:



Target and Source GDTs are constructed as follows:

Offset	Definition
00-01	Segment Limit: 1-FFFFh bytes, but must be at least (2 x (CX))-1
02-04	24 Physical Address: A23 - A0
05	Access Rights Byte: 93h Recommended.
06	Reserved.
07	Physical Address: A31 - A24

3. On exit, the (AH) register contains the Status Code of the function as follows:

(AH)	Meaning
0	Block Move was Successful
1	RAM Parity Error (Cleared on Exit)
2	Exception Interrupt Error
3	Gate A20 Control Error

FUNCTION 88h - GET EXTENDED MEMORY SIZE

This function is called by INT 15h, (AH)=88h, and uses the (AX) register to return the amount of extended memory installed as follows:

INT 15h

(AH)=88h

INPUT
(None)

OUTPUT	
(AX) =	Number of 1K Blocks of extended memory

Notes:

1. Extended memory is memory beyond 1024K bytes. The amount of extended memory installed is determined by the POD. The amount of extended memory is then stored in CMOS RAM at address 30h and 31h. This function reads the contents of this CMOS RAM.

2. See INT 12h for information on Base Memory determination.

FUNCTION 89h - SWITCH TO VIRTUAL MODE

This function is called by INT 15h, (AH)=89h, and uses the (BX), (ES), and (SI) registers to pass the interrupt and descriptor table pointers information and switch to the virtual mode as follows:

INT 15h

(AH)=89h

INPUT	
(BH) =	Interrupt Descriptor Table Number to First 8 Hardware Interrupts
(BL) =	Interrupt Descriptor Table Number to Second 8 Hardware Interrupts
(ES:SI) =	Segment and Offset Address of GDT

OUTPUT	
(AH) =	00h - Successful Switch to Virtual Mode FFh - Failure to Switch to Virtual Mode
(IF) =	1 - Interrupts Enabled 0 - Failure to Switch to Virtual Mode

Notes:

1. This function will switch the CPU into Protected mode. Control is then passed to the next instruction of "INT 15h" to invoke this function.

2. The (BH) and (BL) registers point to the offset address, within the Interrupt Descriptor Table segment, where the first 8 and second 8, respectively, hardware interrupts are located. The Interrupt Descriptor Table is initialized by the GDT as explained below.

3. The Intel 8259-compatible Interrupt Controllers are reinitialized with the Hardware Interrupt Pointers given in the (BX) register. Interrupt Controller #1 is initialized with the pointer to the first 8 hardware interrupts as designated in the (BH) register. Interrupt Controller #2 is likewise initialized with the pointer in the (BL) register.

4. The Protected mode Interrupt Descriptor Table must not be allowed to overwrite any portion of the Real mode BIOS Interrupt Descriptor Table.

5. The user must hereafter handle all I/O commands, because the BIOS will no longer be available for use by the user.

6. The user must initialize the Exception Interrupt Table and Handler.

7. Interrupts are disabled on exit, i.e. when control passes to the user.

8. The (ES:SI) register pair points to a GDT previously built by the caller. The GDT must have the following construction:

Offset	Descriptor	Comments
00h	Dummy	Dummy User initialized to 0.
08h	GDT	Global Descriptor Table Segment Pointer. User initialized.
10h	IDT	Interrupt Descriptor Table Segment Pointer. User initialized.
18h	Data Segment	User's Data Segment Pointer. User initialized.
20h	Extra Segment	User's Extra Segment Pointer. User initialized.
28h	Stack Segment	User's Stack Segment Pointer. User initialized.
30h	Code Segment	User's Code Segment where this function is to return to. User initialized.
38h	Temporary BIOS Code Segment0	Temporary Code Segment for this BIOS function. User can not use until function exits User initialized to 0.

9. The individual descriptors in the GDT are constructed as follows:

Offset	Definition
00-01	Segment Length: 1-0FFFFh bytes
02-04	Physical Address: A23 - A0
05	Access Rights Byte
06	Reserved
07	Physical Address: A31 - A24

FUNCTION 90h - DEVICE BUSY

This function is called by INT 15h, (AH)=90h, and uses no registers except those used to call it.

INT 15h

(AH)=90h

INPUT	
(AL) =	Device Type Code

OUTPUT	
(AH) =	00h
(CF) =	0
(IF) =	1

Notes:

1. This function is called by the BIOS or by the user's BIOS to signal that a potential Wait condition exists on the CPU because a Device Busy condition exists on the hardware. This interrupt provides a "hook" for multitasking code to save the state of the current task and switch processing to another task.

2. The BIOS uses the following guidelines for Device Type Code ranges:

   00h through 7Fh: Serially reusable devices. These are devices that can only be used by one task at a time.

   Specific Device IDs of this type used by the BIOS:
   - 00 - Hard Disk Drive
   - 01 - Diskette Drive
   - 02 - Keyboard
   - 03 - Pointing device

   80h through 0BFh: Simultaneously reusable devices. These devices support multiple simultaneous I/O calls and can be accessed by more than one task at a time.

   Specific Device IDs of this type used by the BIOS:
   - 80h - Network

0C0h through 0FFh: Non-Interrupt devices. These are devices that do not send an interrupt to indicate a POST condition. The dispatching routine must determine when a busy condition no longer exists.

Specific Device IDs of this type used by the BIOS:

0FDh - Diskette Motor Start

0FEh - Parallel Printer

FUNCTION 91h - DEVICE READY

This function is called by INT 15h, (AH)=91h, and uses no registers except those used to call it. This function may be used for multitasking purposes.

INT 15h

(AH)=91h

INPUT

(AL) =	Device Type Code

OUTPUT

(AH) =	00h

Notes:

1. This function is called by the BIOS or the user's BIOS to signal that an "Interrupt Complete," - device ready - condition currently exists with a device that had previously entered a "device-busy" loop. This interrupt provides a multitasking dispatcher with an opportunity to set a "task ready" status flag so that processing of that task can resume when

2. The BIOS uses the following guidelines for Device Type Code ranges:

   00h through 7Fh: Serially reusable devices. These are devices that can only be used by one task at a time.

   Specific Device IDs of this type used by the BIOS:
   - 00 - Hard Disk Drive
   - 01 - Diskette Drive
   - 02 - Keyboard
   - 03 - Pointing device

   80h through 0BFh: Simultaneously reusable devices. These devices support multiple simultaneous I/O calls and can be accessed by more than one task at a time.

   Specific Device IDs of this type used by the BIOS:
   - 80h - Network

0C0h through 0FFh: Non-Interrupt devices. These are devices that do not send an interrupt to indicate a POST condition. The dispatching routine must determine when a busy condition no longer exists.

Specific Device IDs of this type used by the BIOS:
0FDh - Diskette Motor Start
0FEh - Parallel Printer

FUNCTION C0h - GET PARAMETER TABLE POINTER

This function is called by INT 15h, (AH)=C0h, and uses the (AH), (BX), and (ES) registers and the (CF) to return the parameter table pointer as follows:

INT 15h

(AH)=C0h

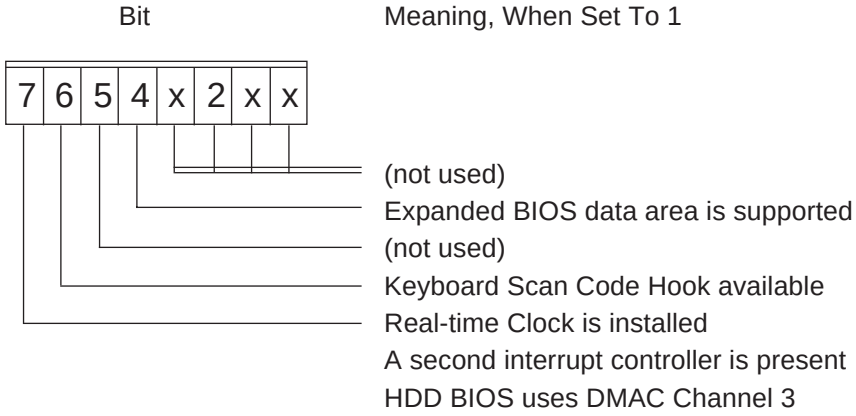
INPUT	
(None)	

OUTPUT	
(AH) =	00h
(BX) =	Offset Address of Pointer
(ES) =	Segment Address of Pointer
(CF) =	0

Note:

This function returns the pointer to the Configuration Status table. It has the following format:

Offset	Definition
00-01	8 = Number of bytes in the remainder of this table
02	Computer Model - 0FCh
03	Computer Type - 01h
04	BIOS Revision Number - 00h
05	System Configuration Byte:



06	(Reserved)
07	(Reserved)
08	(Reserved)
09	(Reserved)

FUNCTION C1h - RETURN EXTENDED BIOS DATA SEGMENT

INT 15h

(AH)=0C1h

INPUT
(None)

OUTPUT	
(ES) =	Segment address of Extended BIOS data area
(CF) =	0 : No error
	1 : Function not supported

FUNCTION C2h - POINTING DEVICE

This function is called by INT 15h, (AH)=C20h, and contains 8 subfunctions called by the (AL) register. Each of the subfunctions is described below.

INT 15h

(AH)=C2h

(AL)=00h 00h Enable Pointing Device

INPUT		
(BH) =	0 = Disable	
	1 = Enable	

OUTPUT		
(AH) =	Status code	
(BX) =	Error Flag:	0 = No error
		1 = Error: Code in (AH)

Note:

The status code in the (AH) register has the following format:

(AH) Value	Meaning
00h	no error
01h	function call invalid
02h	input invalid interface
03h	interface error
04h	resend
05h	pointing device FAR CALL not installed
06-FFh	(reserved)

All subfunctions return the above codes.

(AL)=01h Pointing Device Reset

(AL)=02h Sample Rate Set

INPUT

(BH) =	Sample Rate Code

OUTPUT

(AH) =	Status Code	
(CF) =	Error Flag:	0 = No Error
		1 = Error: Code in (AH)

Note:

The sample rate code is defined as follows:

(BH) Value	Sample Rate
00h	10 reports/sec
01h	20 reports/sec
02h	40 reports/sec
03h	60 reports/sec
04h	80 reports/sec
05h	100 reports/sec
06h	200 reports/sec

(AL)=03h Pointing Device Resolution Set

INPUT

(BH) =	Resolution Value:
00h =	1 count/mm
01h =	2 count/mm
02h =	4 count/mm
03h =	8 count/mm

OUTPUT

(AH) =	Status Code	
(CF) =	Error Flag:	0 = No Error
		1 = Error: Code in (AH)

(AL)=04h Pointing Device Type Read

INPUT

(None)

OUTPUT

(AH) =	Status Code	
(BH) =	Device ID	
(CF) =	Error Flag:	0 = No Error
		1 = Error: Code in (AH)

(AL)=05h Pointing Device Data Package Initialize

INPUT

(BH) =	Data Package Size (in bytes)

OUTPUT

(AH) =	Status Code	
(CF) =	Error Flag:	0 = No Error
		1 = Error: Code in (AH)

Note:

Valid values for the (BH) register are 01h-08h.

(AL)=06h Extended Commands

This subfunction provides 3 services, designated by the (BH) register value, as described below:

(BH)=00h Get Status

INPUT

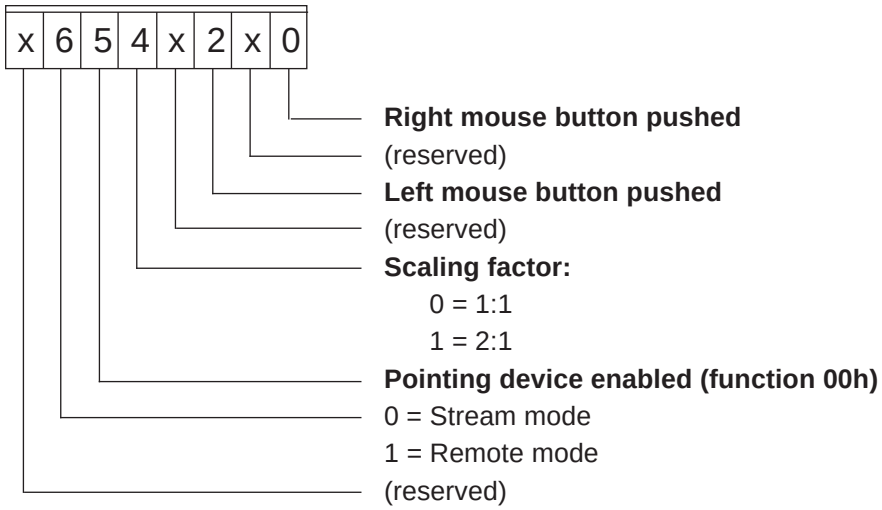
(None)

OUTPUT

(BL) =	Status Byte 1
(CL) =	Status Byte 2
(DL) =	Status Byte 3

Note:

The 3 status bytes have the following format
Status byte 1:



Status byte 2: current resolution
00h 1 count/mm
01h 2 count/mm
02h 4 count/mm
03h 8 count/mm

Status byte 3: current pointing device sampling rate

(BH)=01h Set Scaling to 1:1

(BH)=02h Set Scaling to 2:1

(AL)=07h Device Driver FAR CALL initialization

INPUT	
(ES:BX)=	Segment and Offset of Pointing Device Driver

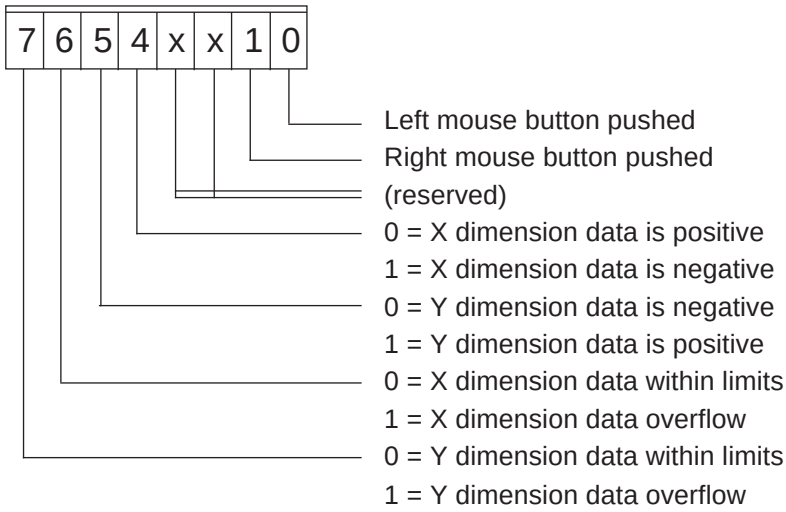
OUTPUT	
(AH) =	Status Code
(CF) =	Error Flag: 0 = No Error 1 = Error: Code in (AH)

Note:

1. This function calls the device driver for the pointing device. The (ES:BX) pointer can be reset by setting all bits to 0.

2. When the device driver is called, the BIOS automatically puts the following 8 bytes on the stack:

Byte	Definition
0	00h
1	Status:



2-3	X dimension data:	Byte 2 = 00h Byte 3 = X data
4-5	Y dimension data:	Byte 4 = 00h Byte 5 = Y data
6-7	Z dimension data:	Byte 6 = 00h Byte 7 = Z data

FUNCTION F0h - LOADING RESET

This function, called by INT 15h, (AH)=F0h, resets the system.
Unlike the normal reset operation, which jumps to the reset vector (FFFF:FFF0h), the loading reset restarts the system from the boot block, with the result that the entire system, including shadow RAM, is initialized.

INT 15h

(AH)=F0h

INPUT
(None)

OUTPUT
(Reset)

FUNCTION F1h - SHADOW RAM WRITE ENABLE

This function, called by INT 15h, (AH)=F1h, enables/disables writes to shadow RAM. The default setting is that shadow RAM writes are disabled; this function can be used to enable shadow RAM writes.

INT 15h

(AH)=F1h

INPUT	
(AL) =	0 : Set shadow RAM to read only
(AL) =	1 : Set shadow RAM to read/write

OUTPUT	
(AH) =	0
(CF) =	0 : Normal end

INT 16h - KEYBOARD SERVICES

Register convention

For each function, the following registers are used to maintain a consistent user interface:

On Entry:

AH	Function code
AL, flags, BX, CX	See each function

On Exit:

AH, AL, flags	See each function
BX	Preserved
CX	Preserved

FUNCTION 00h - GET KEYBOARD CHARACTER

This function is called by INT 16h, (AH)=00h, and uses the (AX) register to fetch the scan code and ASCII code as follows:

INT 16h

(AH)=00h

INPUT	
(None)	

OUTPUT	
(AH) =	Scan Code
(AL) =	ASCII Code

Notes:

1. This function will read the next character from the keyboard buffer. The character is moved from the keyboard buffer into the (AX) register.

   If there are no characters in the keyboard buffer, this function will wait indefinitely until a character is entered from the keyboard. Note that this will halt program execution until a character is entered. Use INT 16h, Function 01h to test for the presence of a character in the buffer before calling this function.

2. The (AH) register will contain either the Scan code for the keyboard character, or the character code for a special key, such as a Function key.

3. The (AL) register will contain either the ASCII code for the keyboard character, or 00h to indicate a special key has been pressed.

4. Use function 10h to distinguish between duplicate keys or to detect keys not found on a standard AT keyboard, and to evaluate keyboard input from an “enhanced” keyboard.

FUNCTION 01h - TEST FOR CHARACTER READY

This function is called by INT 16h, (AH)=01h, and uses the (AX) register and the (ZF) flag to return the scan code, ASCII code, and character presence information as follows:

INT 16h

(AH)=01h

INPUT	
(None)	

OUTPUT		
(AH) =	Scan Code	
(AL) =	ASCII Code	
(ZF) =	Character Ready:	0 = Character in Buffer
		1 = No Character in Buffer

Notes:

1. This function tests whether there is a character in the keyboard buffer. If there is, the (ZF) flag will have a value of 0.

2. This function also returns the Scan code and ASCII code of the character waiting in the buffer, but does not remove the character from the buffer.

3. The (AH) register will contain either the Scan code for the keyboard character, or the character code for a special key, such as a Function key.

4. The (AL) register will contain either the ASCII code for the keyboard character, or 00h to indicate a special key has been pressed.

5. Use function 11h to distinguish between duplicate keys or to detect keys not found on a standard AT keyboard, and to evaluate keyboard input from an “enhanced” keyboard.

FUNCTION 02h - GET SHIFT STATUS

This function is called by INT 16h, (AH)=02h, and uses the (AL) register to fetch the shift status information as follows:

INT 16h

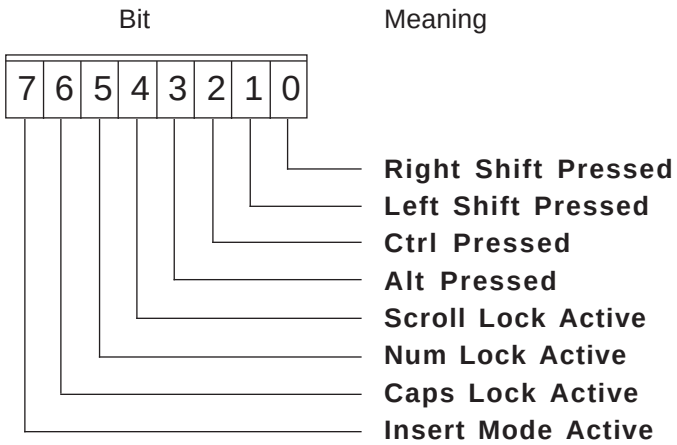
(AH)=02h

INPUT
(None)

OUTPUT	
(AL) =	Shift Status Code

Notes:

1. This function returns the Shift Status of the keyboard as a one-byte code to the (AL) register. The shift status is indicated as follows:



2. Use function 12h to distinguish between duplicate keys or to detect keys not found on a standard AT keyboard, and to evaluate keyboard input from an “enhanced” keyboard.

FUNCTION 03h - SET TYPEMATIC RATE AND DELAY

This function is called by INT 16h, (AH)=03h, (AL)=05h, and uses the (BX) register to store the Typematic Rate parameters as follows:

INT 16h

(AH)=03h

(AL)=05h

INPUT	
(BH) =	Typematic Delay
(BL) =	Typematic Rate

OUTPUT
(None)

Notes:

1. The (BH) register value, Typematic Delay, should not exceed 3.

   The formula for the Typematic Delay value is 250 ms x (1 + binary value of bits 1 and 0 of (BH)). This value has a variance of 20%.

   If all variables are out of range, this function return immediately without setting the delay or the rate.

2. The (BL) register value, Typematic Rate, should not exceed 63d. Only the lowest (least significant) 5 bits of this byte should be used. The remaining bits should be set to zero (0). The Typematic Rate, i.e. characters-per-second, is determined by the value of the (BL) register as shown below.

Bit Rate	Bit Rate	Bit Rate
00000 = 30.0	01011 =10.9	10110 = 4.3
00001 = 26.7	01100 =10.0	10111 = 4.0
00010 = 24.0	01101 = 9.2	11000 = 3.7
00011 = 21.8	01110 = 8.6	11001 = 3.3
00100 = 20.0	01111 = 8.0	11010 = 3.0
00101 = 18.5	10000 = 7.5	11011 = 2.7
00110 = 17.1	10001 = 6.7	11100 = 2.5
00111 = 16.0	10010 = 6.0	11101 = 2.3
01000 = 15.0	10011 = 5.5	11110 = 2.1
01001 = 13.3	10100 = 5.0	11111 = 2.0
01010 =12.0	10101 = 4.6	

FUNCTION 05h - PUT CHARACTER IN KEYBOARD BUFFER

This function is called by INT 16h, (AH)=05h, and uses the (CX) and (AL) registers to put the scan code and ASCII code data as follows:

INT 16h

(AH)=05h

INPUT	
(CH) =	Scan Code
(CL) =	ASCII Code

OUTPUT	
(AL) =	Error Code: 00h = No Error
	01h = Buffer Full

Note:

This function provides software emulation of keyboard entries. If there is no room in the buffer, the character is not saved and the (AL) register will have a value of 01h on return. There is room for 15 characters in the keyboard buffer.

FUNCTION 10h - GET EXTENDED KEYBOARD CHARACTER

This function is called by INT 16h, (AH)=10h, and uses the (AX) register to fetch the scan code and ASCII code as follows:

INT 16h

(AH)=10h

INPUT	
(None)	

OUTPUT	
(AH) =	Scan Code
(AL) =	ASCII Code

Notes:

1. This function is similar to function 00h, except that the scan code is not converted. Hence, the unique scan and ASCII code of the key is returned by this function. Applications which do not need to distinguish between duplicate keys or do not need to detect F11 or F12 (or other new keys found on the "Enhanced" keyboard) may use function 00h.

2. This function will read the next character from the keyboard buffer. The character is moved from the keyboard buffer into the (AX) register.

   If there are no characters in the keyboard buffer, this function will wait indefinitely until a character is entered from the keyboard. Note that this will halt program execution until a character is entered. Use INT 16h, Function 01h to test for the presence of a character in the buffer before calling this function.

3. The (AH) register will contain either the Scan code for the keyboard character, or the character code for the special keys, such as a Function key.

4. The AL register contains the ASCII code generated by pressing an alphanumeric key on the keyboard, or 00h if a special key was pressed, or E0h when an extended key was pressed.

FUNCTION 11h - TEST FOR EXTENDED CHARACTER READY

This function is called by INT 16h, (AH)=11h, and uses the (AX) register and the (ZF) flag to return the scan code, ASCII code, and character presence information as follows:

INT 16h

(AH)=11h

INPUT	
(None)	

OUTPUT		
(AH) =	Scan Code	
(AL) =	ASCII Code	
(ZF) =	Character Ready:	0 = Character in Buffer
		1 = No Character in Buffer

Notes:

1. This function is similar to function 01h, except that the scan code is not converted. Hence, the unique scan and ASCII code combination of the key is returned by this function. Applications which do not need to distinguish between duplicate keys or detect the new keys on the "Enhanced" keyboard (e.g. F11 and F12) may use function 01h.

2. This function tests whether there is a character ready in the keyboard buffer. If there is, the (ZF) flag will have a value of 0.

3. This function also returns the Scan code and ASCII code of the character waiting in the buffer, but does not remove the character from the buffer.

4. The (AH) register will contain either the Scan code for the keyboard character, or the character code for the special keys, such as a Function key.

5. The AL register contains the ASCII code generated by pressing an alphanumeric key on the keyboard, or 00h if a special key was pressed, or E0h when an extended key was pressed.

FUNCTION 12h - GET EXTENDED SHIFT STATUS

This function is called by INT 16h, (AH)=12h, and uses the (AX) register to fetch extended shift status information as follows:

INT 16h

(AH)=12h

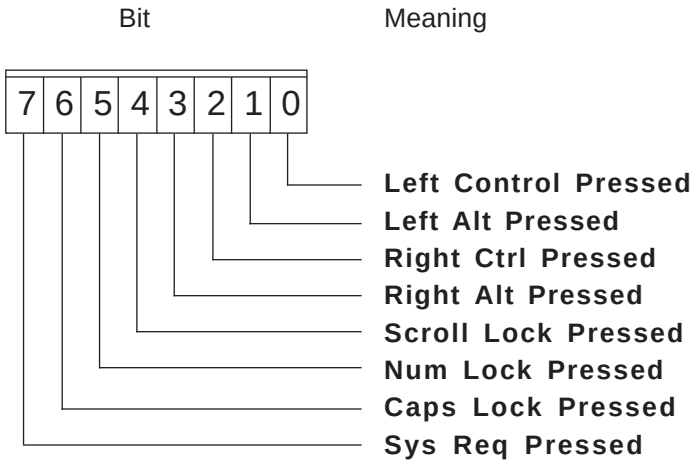
INPUT
(None)

OUTPUT	
(AH) =	Extended Shift Status Code
(AL)=	Shift Status Code

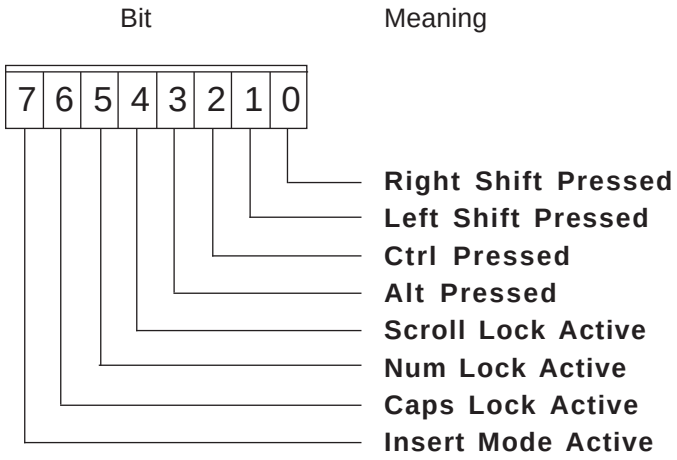
Notes:

This function returns a detailed Shift Status of the keyboard, similar to function 02h, except that by using this function it is possible to distinguish between the individual shift keys.

(AH)=Extended Shift Status Code



(AL)=Shift Status Code



INT 17h - PARALLEL INTERFACE SERVICES

Register convention

For each function, the following registers are used to keep a consistent user interface when each function is invoked:

On Entry:

AH	Function code
AL	(See each function)
DX	Logical printer number

On Exit:

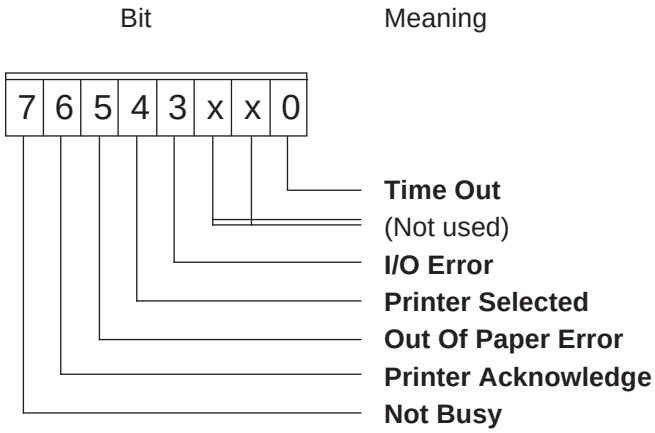
AH	Status
AL	(Preserved)
DX	(Preserved)

Notes:

1. On entry, if (AH) is outside the range 00h - 02h then the function is terminated immediately with no other register changes and no status returned in the (AH) register.

2. The value of DX indicates the printer number, which stands for the parallel printer used for text printout. The printer number can be either 0000, 0001, or 0002. For details on the printer base table, refer to the section "BIOS Work Area." The printer number 0000 is used by the printer screen service INT 05h. If the port address 378h, 278h, or 3BCh is detected, the port address is set in the printer number 0000, 0001 or 0002 and the printer port is initialized by the POD.

3. The (AH) register returns the Printer Status code as below. Note that bit 0 set to 1 means that the character could not be printed.



FUNCTION 00h - PRINT CHARACTER

This function is called by INT 17h, (AH)=00h, and uses the (AX) and (DX) registers to print the specified character code as follows:

INT 17h

(AH)=00h

INPUT	
(AL) =	Character Code
(DX) =	Printer Number (0, 1, or 2)

OUTPUT	
(AH) =	Printer Status Code

Note:

The (AL) register is preserved on exit.

FUNCTION 01h - INITIALIZE PRINTER

This function is called by INT 17h with (AH)=01, and uses the (AH) and (DX) registers to initialize the printer as follows:

INT 17h

(AH)=01h

INPUT

(DX) =	Printer Number (0, 1, or 2)

OUTPUT

(AH) =	Printer Status Code

FUNCTION 02h - GET PRINTER STATUS

This function is called by INT 17h, (AH)=02, and uses the (AH) and (DX) registers to get the printer status as follows:

INT 17h

(AH)=02h

INPUT	
(DX) =	Printer Number (0, 1, 2)

OUTPUT	
(AH) =	Printer Status Code

INT 18h - BOOT FAIL

This handler is invoked when a system boot is attempted from a non-system diskette and/or the hard disk fails.

If invoked, this handler displays the following message:

Non-System disk or disk error
Replace and strike any key when ready

and waits for any key input. When a key is pressed, this interrupt calls INT19h.

INT 19h - BOOT STRAP LOADER

This interrupt terminates the current procedure (DOS or stand-alone application) and attempts to boot or re-boot an operating system or stand-alone application.

This handler will first try to boot up from diskette drive 0, and then from hard disk 0. If the boot sector (cylinder 0, head 0, sector 1), is successfully read, the boot sector data is written to RAM, starting at address 0000:7C00h and control is transferred to that address. If this fails, INT 18h is called.

INT 1Ah - REAL-TIME CLOCK SERVICES

Register convention

Throughout each of the following RTC functions, the following registers are used to maintain a consistent user interface:

On Entry:

AH	Function code
AL, CX, DX, flags	(See each function)

On Exit:

AH	Return status
AL, CX, DX, flags	(See each function)

1. On entry, if (AH) is outside the range (00h - 07h) this function returns immediately with (AH) preserved and (CF)=1. If the function is successfully executed, then (AH) is reset to 00h.

2. On return the Interrupt Enable Flag (IF) is always set to 1.

FUNCTION 00h - READ CURRENT CLOCK COUNT

This function is called by INT 1Ah, (AH)=00h, and uses the (AL), (CX), and (DX) registers to read the clock count data as follows:

INT 1Ah

(AH)=00h

INPUT
(None)

OUTPUT	
(AL) =	24 Hour Status
(CX) =	High Order Word of Clock Count
(DX) =	Low Order Word of Clock Count

Notes:

1. The (AL) register value, 24 Hour Status, will be 0 if less than 24 hours have passed since the clock count was last read or set. If more than 24 hours have elapsed, the value in the (AL) register is the number of days that have elapsed.

   Note that if more than 24 hours have been counted (in other words, several days have elapsed), the value in the (AL) register will be the number of days, but calling this function will reset the 24-Hour Status byte to 00h.

2. The clock count is set in the (CX) and (DX) registers. The (CX) register holds the upper portion of the count, and the (DX) register the lower portion. The clock count (double word) is kept in memory at address 0040:006Ch. When the clock count reaches 1800B0h it is reset to 00000000h, and the 24-Hour Status byte is incremented.

FUNCTION 01h - SET CLOCK COUNT

This function is called by INT 1Ah, (AH)=01h, and uses the (CX) and (DX) registers to set the clock count data as follows:

INT 1Ah

(AH)=1h

INPUT	
(CX) =	High Order Byte of Clock Count
(DX) =	Low Order Byte of Clock Count

OUTPUT
(None)

Note:

This function is used to set the clock count, as entered in the (CX) and (DX) registers into system memory. The clock "ticks" approximately 18.2 times a second. The maximum value for the clock count is 1800B0h. When this value is reached, the next clock count is set to 00000000h, and the 24-Hour Status byte is incremented.

FUNCTION 02h - READ TIME FROM REAL-TIME CLOCK

This function is called by INT 1Ah, (AH)=02h, and uses the (CX) and (DX) registers to read the real-time clock data as follows:

INT 1Ah

(AH)=02h

INPUT
(None)

OUTPUT	
(CH) =	Hours (in BCD)
(CL) =	Minutes (in BCD)
(DH) =	Seconds (in BCD)
(DL) =	Daylight savings time code
(CF) =	1 on Error

Notes:

1. This function returns time information from the real-time clock. The Hours, Minutes, and Seconds are returned in the (CH), (CL), and (DH) registers respectively. The values are returned in BCD format, with each byte registers representing two decimal digits.

2. The time data is stored in CMOS RAM as follows:

Hours	Byte 04 (in BCD)
Minutes	Byte 02 (in BCD)
Seconds	Byte 00 (in BCD)

3. The (CF) flag will be set to 1 if the real-time clock is not operating, or if some other error occurred during the execution of this function.

FUNCTION 03h - SET REAL-TIME CLOCK

This function is called by INT 1Ah, (AH)=03h, and uses the (CX) and (DX) registers to set the hour, minute, second, and daylight savings time code as follows:

INT 1Ah

(AH)=03h

INPUT	
(CH) =	Hours (in BCD)
(CL) =	Minutes (in BCD)
(DH) =	Seconds (in BCD)
(DL) =	Daylight Savings Time Code

OUTPUT
(None)

Notes:

1. This function is used to set the time in the real-time clock. The values set in the (CH), (CL), and (DH) registers must be in BCD format, with each byte representing two decimal digits.

2. The (DL) register value, Daylight Savings Time Code, can be set to 1 to indicate daylight savings time. Otherwise it should have a value of 0.

3. The time data is stored in CMOS RAM as follows:

Hours	Byte 04 (in BCD)
Minutes	Byte 02 (in BCD)
Seconds	Byte 00 (in BCD)
Daylight Savings Time Code Enable	Byte 0Bh (Status Register B), Bit 0

FUNCTION 04h - READ DATE FROM REAL-TIME CLOCK

This function is called by INT 1Ah, (AH)=04h, and uses the (CX) and (DX) registers to read the century, year, month, and day data as follows:

INT 1Ah

(AH)=04h

INPUT
(None)

OUTPUT	
(CH) =	Century (in BCD)
(CL) =	Year (in BCD)
(DH) =	Month (in BCD)
(DL) =	Day (in BCD)
(CF) =	1 on Error

Notes:

1. The (CH) register value is returned in BCD format. It will be either 19d or 20d.

2. The Year, Month, and Day are returned in the (CL), (DH), and (DL) registers, respectively, in BCD format. BCD format uses one byte to represent two decimal digits.

3. The date is stored in CMOS RAM as follows:

Century	Byte 32h (in BCD)
Year	Byte 09h (in BCD)
Month	Byte 08h (in BCD)
Day	Byte 07h (in BCD)

4. The (CF) flag will be set to 1 if the real-time clock is not operating, or if some other error occurred during the execution of this function.

FUNCTION 05h - SET DATE IN REAL-TIME CLOCK

This function is called by INT 1Ah, (AH)=05h, and uses the (CX) and (DX) registers to set the century, year, month, and day data as follows:

INT 1Ah

(AH)=05h

INPUT	
(CH) =	Century (in BCD)
(CL) =	Year (in BCD)
(DH) =	Month (in BCD)
(DL) =	Day (in BCD)

OUTPUT
(None)

Notes:

1. The (CH) register value must be entered in BCD format. It will be either 19d or 20d.

2. The Year, Month, and Day must be entered in the (CL), (DH), and (DL) registers, respectively, in BCD format. BCD format uses one byte to represent two decimal digits.

3. The date is stored in CMOS RAM as follows:

Century	Byte 32h (in BCD)
Year	Byte 09h (in BCD)
Month	Byte 08h (in BCD)
Day	Byte 07h (in BCD)

FUNCTION 06h - SET ALARM TIME

This function is called by INT 1Ah, (AH)=06h, and uses the (CX) and (DH) registers to set the hour, minute, and second alarm data as follows:

INT 1Ah

(AH)=06h

INPUT	
(CH) =	Hours (in BCD)
(CL) =	Minutes (in BCD)
(DH) =	Seconds (in BCD)

OUTPUT	
(CF) =	1 if Alarm Previously Enabled

Notes:

1. This function is used to automatically invoke a user-written routine at the set alarm time. The alarm hour, minute, and second are specified in the (CH), (CL), and (DH) registers, respectively. The alarm time can be set to a maximum of 23 hours, 59 minutes and 59 seconds. The alarm will continue to be active until the Reset Alarm function, (AH)=07h, is invoked. No more than one alarm may be enabled at any time.

2. The alarm data is stored in CMOS RAM as follows:

Hours	Byte 05 (in BCD)
Minutes	Byte 03 (in BCD)
Seconds	Byte 01 (in BCD)
Alarm Enable	0Bh (Status Register B), Bit 5

3. The vector to the user-written routine must be placed in the vector table for INT 4Ah. When the alarm time is reached, INT 4Ah will be invoked and the code it points to will be executed.

4. The (CF) flag will be set to 1 if the alarm has been previously set, the real-time clock is not operating, or if some other error occurred during the execution of this function.

FUNCTION 07h - RESET ALARM

This function is called by INT 1Ah, (AH)=07h, and uses no registers other than those used to call it.

INT 1Ah

(AH)=07h

INPUT
(None)

OUTPUT
(None)

Note:

This function is used to cancel the alarm. When this function is invoked the Alarm Enable bit in CMOS RAM (0Bh, bit 5) is set to 0, and the alarm is disabled.

WATCHDOG TIMER SUPPORT FUNCTIONS

Watchdog Timer on the EASI Bus

The watchdog timer's "count-out" output signal is output on the WDTIM# pin on the EASI bus. If the watchdog timer is to be used, connect this pin to either the IOCHCK# pin or the IRQn pin, whichever is preferable.

The WDTIM# pin is normally high. If the watchdog timer counts out, the WDTIM# pin goes low for about 30[micro]s, and then returns to the high level. Because the watchdog timer begins to count again after counting out, the WDTIM# pin goes low and an interrupt is generated on a regular cycle, unless the BIOS counter reset function is called.

The operation of the watchdog timer is controlled through BIOS calls.

BIOS Function

The watchdog timer's software interface is defined as an extension of the Int 1Ah software interrupt (timer service).

Function number	Description
08h	Get watchdog timer information
09h	Get protect mode interface address
0Ah	Counter reset

FUNCTION 08h - GET WATCHDOG TIMER INFORMATION

INT 1Ah

(AH) = 08h

INPUT

(None)

OUTPUT

(AL) =	0	The watchdog timer is stopped
	1	The watchdog timer is counting down
(CY) =	0	No error
	1	Error

This function returns the status of the watchdog timer.

When the output carry flag is "0", the watchdog timer is available. The current status of the watchdog timer is returned in the AL register.

When the output carry flag is "1", the watchdog timer is unavailable, because it is not supported, or there is a problem with the counter, or for other reasons. The value that is returned in the AL register is meaningless in this case.

FUNCTION 09h - GET PROTECT MODE INTERFACE ROUTINE ADDRESS

INT 1Ah

(AH) = 09h

INPUT

(None)

OUTPUT

(DX) =	Segment address of the protect mode interface routine	
(AX) =	Offset for the protect mode interface routine in the above segment	
(CY) =	0	No error
	1	Error

This routine returns the entry address for the interface routine that is needed in order to use the watchdog timer in protect mode. If the output carry flag is “1”, the protect mode interface is not supported.

Using the protect mode interface

The protect mode interface is used as follows:

1. While in real mode, use Int 1Ah function 09h to get the protect mode interface routine address.

2. ??With GDT in LDT,?? create a code segment descriptor for the protect mode interface routine address. The segment base is the product of the value returned in DX by this function multiplied by 16.

3. Set values in each register in the same manner as for an Int 1Ah call, and execute a far call of the protect mode interface routine.

The protect mode interface routine is written in accordance with the following conditions.

- It is a 16-bit code segment. (The data can be read.)
- It is executed at authorization level 0.
- The segment base address is the product of the value returned in DX by this function multiplied by 16.
- The entry address offset is the value returned in AX by this function.

FUNCTION 0Ah - COUNTER RESET

INT 1Ah

(AH) - 0Ah

INPUT

(AL) =	0	Stop watchdog timer
	1	Start count
(CX) =	Initial value of counter (Specify in units of 100[micro]s.)	

OUTPUT

(None)

This function resets the watchdog timer.
When AL is “1”, the initial value is written to the watchdog timer counter, and counting begins anew.
When AL is “0”, the watchdog timer stops operating. The value in CX has no meaning. After it calls this function once with A = “1”, an application program must call the function again within the time specified by the initial value. If this function is not called again within the specified time, WDTIM# goes low and an interrupt is generated.

2

EPSON DBIOS Features

Table of Contents

USER SETTINGS

The EPSON DBIOS is equipped with a number of additional features that increase its usefulness to the user. This section describes the additional feature.

The settings for all of these features are written in the CMOS-RAM built into the system. For systems which do not provide a CMOS-RAM back-up power supply, it will be necessary to use the default CMOS-RAM setting parameter table.

Test level control

While booting up the system, the POD checks the memory installed in the system, initializes the circuits in the system, and initializes the peripheral devices that are connected to the system.

In most cases where a failure occurs during this initialization sequence, or the POD detects a problem while testing one of the devices, the POD displays an error message on the display device.

If a major problem that prevents system startup is detected, the POD halts the system boot process at that point. In this event, the POD step ID of the position where the POD stopped is output to I/O port 80h; this information can be used to help pinpoint the problem.

(For details, refer to the appendices "POD Step ID Numbers" and "POD Error Messages" at the end of this manual.)

In order to reduce the system boot time, it is possible to set one of three test levels (Full Test, Easy Test, or No Test) for most initialization steps.

"Full Test" confirms every check item that can be checked in each POD step.

"Easy Test" performs only those general checks that are essential to system operation.

"No Test" executes only the initialization sequence, only performing the bare minimum of checks that are necessary for system startup.

If "No Test" is set, it is possible that the system could start up with an undetected problem. Note that in a worst case scenario, such a situation could result in damage to the system.

The default test level in the EPSON DBIOS is "Easy Test." The test level can be changed by using the RAK (ROM Adaptation Kit) to change the BIOS parameters.

Keyboard typematic rate setting

The typematic rate and the typematic delay can be set individually for the initial keyboard typematic rate settings when the system starts up.

Keyboard numeric lock on/off

This function controls the ON/OFF of the keyboard numeric lock when the system starts up.

User-definable hard disk parameters

The EPSON DBIOS allows the user to select one of the following two methods for setting the hard disk parameters.

1) Automatic recognition by BIOS
2) Manual setting by the user

In either case, all of the following information must be set as appropriate in the hard disk parameters:

1) Type of hard disk parameters (LBA or CHS)
2) Number of heads (1 to 65,535)
3) Number of cylinders (1 to 16)
4) Number of sectors (1 to 255)

The values set for items 2, 3, and 4 are the exact values indicated for the number of heads, cylinders, and sectors in the hard disk documentation. Therefore, these values are the same, whether item 1 is "LBA" or "CHS."

For most typical situations (in which the system is running MS-DOS, Windows, etc.), select "LBA" for item 1.

Selecting "LBA" for some operating systems (such as Netware, SCO Unix, and others), however, will create hard disk access problems. For details, refer to the operating system documentation.

The hard disk parameters can be set on the setup screen. The parameters that are set in the setup screen are stored in CMOS-RAM.

Password function

When the password function is enabled, at start-up the system requests a password to be entered from the keyboard, and refuses to start if the correct password is not entered. If an incorrect password is entered for three consecutive times, the system refuses further password input, and does not start up.

Network server mode

The network server mode provides an extra level of protection from unauthorized operation or unintentional mis-operation when the system is operating as a network server.

In network server mode, keyboard input is not accepted until a system administrator's password is input correctly.

When the system is in network server mode with the password function enabled, the system does not request a password at startup, and bootup proceeds as though no password were set.

Password input is monitored by the "keyboard controller", and until the correct password is input, keyboard input is not transferred to the system.

Serial port control

This function serves to specify the I/O addresses of the internal serial ports of the system. Available settings are "Disabled", "COM1 (3F8h)", "COM2 (2F8h), "COM3 (3E8h), and "COM4 (2E8h)".

Serial port I/O addresses		
Port	I/O address	Interrupt
COM1	3F8h	IRQ4
COM2	2F8h	IRQ3
COM3	3E8h	IRQ11
COM4	2E8h	IRQ10

The interrupt used for each serial port can be selected using the system setup parameters described below. Available interrupt settings are IRQ4, IRQ3, IRQ11, and IR10. COM B can also be used as infrared port.

Parallel port control

This function specifies the I/O addresses of the system's built-in parallel ports. These ports can either be set as "disabled," "primary," or "secondary."

Parallel port I/O addresses		
Port	I/O address	Interrupt
Primary	378h	IRQ7
Secondary	278h	IRQ5

The operation mode of the parallel ports can be set to either "unidirectional," "bi-directional," or "EPP (Enhanced Parallel Port)."
The EPSON DBIOS supports EPP functions according to EPP BIOS version 1.0.

Diskette boot control

This enables or disables booting from a floppy disk.
This function is normally valid when the user is running application software from the hard disk. It restricts hard disk management and software updating to authorized users only.

ROM setup

You can setup the system with the setup program installed on the BIOS ROM.
To start the setup program please reset the system and wait until the following message appears.

Press <F10> to run setup.

then press F10 key.

PCIC initialization

This function initializes the PCIC.

This function makes it possible to use an ATA card that is installed in a PCMCIA socket as a hard disk drive.

This function can initialize the following sockets:

1) PCMCIA socket #0
2) PCMCIA socket #1

These sockets can be allocated as the following HDDs:

1) Primary IDE master drive
2) Secondary IDE master drive

AUTOMATIC SETTINGS

This section covers settings made automatically by the POD.

FDC setting

When an external floppy disk controller is connected to the system, the internal floppy disk controller is automatically disabled.

HDD type setting

When a hard disk drive is connected to the IDE interface of the system, the drive parameters are set automatically.

This function can be used only if the HDD in the CMOS-RAM is set to Type 1 or Type 2. The function can be used when the hard disk contains Conner compatible parameter information.

When a hard disk is connected, the BIOS sends a "Read ID" command to the HDD and uses the parameters returned by the HDD.

To enable this function, the following three conditions must be met.

1. The user must have specified Type 1 or Type 2 as HDD type in the CMOS-RAM.
2. The drive must support the Conner "Read ID" command.
3. The drive parameters must be within the following range.
   - Default logical cylinders: ≤ 65535
   - Default logical heads: ≤ 16
   - Default logical sectors/track: ≤ 255
   - Disk capacity ≤ 8.4GB

Large-Capacity Hard Disks

Conventional IDE interface and BIOS specifications limited HDD capacity to a maximum of 528MB. However, the BIOS of this system supports also large-capacity hard disks exceeding 528MB. The upper limit is 8.4GB for each drive, resulting in a total maximum capacity of 16.8GB when two drives are connected. The BIOS fulfills the so-called Enhanced IDE specifications, but high-speed data transfer as defined in the E-IDE specifications is dependent on the hardware specifications.

When "LBA" has been specified for the HDD type, the BIOS uses translation mode for accessing the HDD. The cylinder number, head number, and sector number passed to the BIOS by the operating system or the application software are converted to logical block addresses and passed to the HDD along with data indicating that access is carried out in translation mode. When automatic recognition is specified, the enhanced drive parameter table is created while POD is being executed, the enhanced drive parameter table is created.
When "NORMAL" has been specified for the HDD type, the BIOS uses automatic mode for accessing the HDD. The cylinder number, head number, and sector number passed to the BIOS by the operating system are passed as is to the HDD.
The operating system determines which type should be used. Type 1 will be appropriate for most cases, but some operating systems (such as Netware and Unix) require Type 2.

Parallel port setting

If the I/O address specified in CMOS-RAM for the internal parallel port conflicts with an externally connected parallel port, the internal port is automatically disabled.

COM A serial port setting

If the I/O address specified in CMOS-RAM for the internal COM A serial port conflicts with an externally connected serial port, the COM A serial port is automatically disabled.

COM B serial port setting

If the I/O address specified in CMOS-RAM for the internal COM B serial port conflicts with COM A serial port or an externally connected serial port, the COM B serial port is automatically disabled.

3

Keyboard Interface

Table of Contents

The keyboard routine contained in the EPSON DBIOS converts keyboard scan codes into a format referred to as extended ASCII.
Extended ASCII is a set of 1-byte cahracter codes and scan codes with values that range from 0 to 255. It includes codes for certain special keyboard functions, and for functions handled through the keyboard routine or through interrupts.

CHARACTER CODES

1-byte character codes are passed through the BIOS keyboard routine to the system or application program via the (AL) register.

Extended function codes

Some functions cannot be represented by standard ASCII codes, and are identified by an extended code. When an identifying code of 00, E0h, or F0h is returned in the (AL) register, the system or application program examines the scan code in (AH). This scan code can be used to identify the specific function.

Shift states

There are a number of keys that cause shifted states of operation on the keyboard. They are slightly different for the main keyboard and the numeric keypad.

Typewriter area, function keys and cursor control

There are four main shift states for most of the keyboard. Note that the [Caps Lock] key only affects alphabetic keys. For the typewriter area only, there is also a graphics mode which only applies when an international keyboard layout is selected with a KEYB command.

Some special functions are also invoked by pressing [Ctrl] - [Alt] with other special non-alphanumeric keys.

The five shift states are:

Shift State	Produced by
Unshifted	No [Caps Lock], or [Caps Lock] and [Shift]
Shift	[Caps Lock] or [Shift]
Ctrl	[Ctrl]
Alt	[Alt]
Graphics	[Alt Gr] or [Ctrl]-[Alt]; only with international keyboard layouts

Numeric keypad

There are only four shift states for the numeric keypad, including some special functions. The codes are the same for all keyboard layouts.

The four shft states are:

Shift State	Produced By
Normal	No [Num Lock] or [Num Lock] and [Shift]
Num Lock	[Num Lock] or [Shift]
Ctrl	[Ctrl]; invoke editing functions
Alt	[Alt]

There are no pre-defined [Alt] key functions for the numeric keypad, but holding down [Alt] and typing a decimal ASCII code on the number keys passes the corresponding single-byte code directly to the system via the keyboard routine.

Shift key priorities

When the [Alt], [Ctrl], and [Shift] keys are pressed in combination with alphanumeric keys, a valid code is generated. The order of precedence is as follows:

1 [Alt]
2 [Ctrl]
3 [Shift]

The only valid combination of these keys is [Ctrl]-[Alt] which is used with the [DEL] key to perform a system reset.

Special shifted functions

The following functions are performed by using shift keys in combination with others, and do not pass codes to the system:

The [Scroll Lock] key simply changes the shift status of that key without sending a code. The shift status can be used by applications if needed.

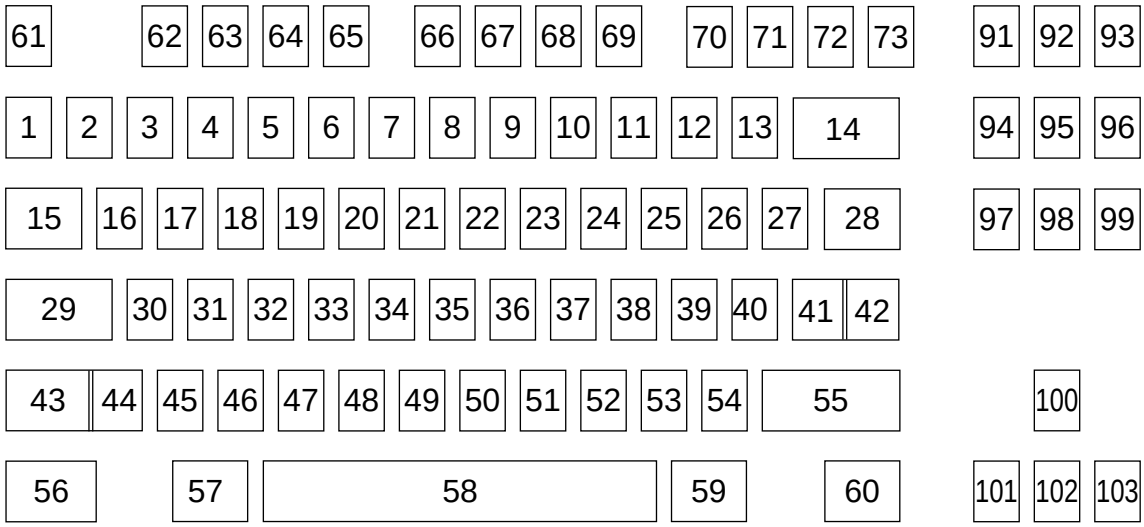
Two other keys send a code to the system and also call BIOS interrupt routines. The combination of [Ctrl] and [Pause] is the [Break] function, and calls the dummy interrupt INT 1Bh. The programmer can modify the action of [Break] accordingly, though DOS treats this combination in the same way as [Ctrl]-C.

Function	Produced by
Pause	[Pause]
PrintScreen	[PrintScreen] or [Shift]-[PrintScreen]
ScrollLock	[ScrollLock]
SystemReset	[Ctrl]-[Alt]-[Del]

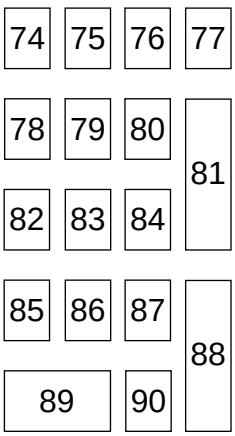
CHARACTER AND SCAN CODE TABLES

The following tables contain the specific scan codes used by the computer. The complete USA layout table is given; the codes for the typewriter area are given first, then the codes for the special function keys, the numeric keypad, and the cursor control keys. (These are followed by tables of differences for the main keyboard in each international layout supported by DOS.)

The scan codes and character codes are all given in hexadecimal;
an - indicates that the key combination is suppressed in the keyboard routine. The decimal key numbers used in the first column of the table are shown in this diagram:



Numeric keypad keys



Note: Key number 28 is only present on the US ASCII keyboard, and key numbers 41 and 44 are only present on the other keyboard layouts.

US ASCII

Typewriter area

Key Number	Key Marking	Scan Code	Key Number	Key Marking	Scan Code
1	'	0E	30	A	1C
2	1	16	31	S	1B
3	2	1E	32	D	23
4	3	26	33	F	2B
5	4	25	34	G	34
6	5	2E	35	H	33
7	6	36	36	J	3B
8	7	3D	37	K	42
9	8	3E	38	L	4B
10	9	46	39	;	4C
11	0	45	40	'	52
12	-	4E	42	Enter	5A
13	=	55	43	Shift	12
14	Backspace	66	45	Z	1A
15	Tab	0D	46	X	22
16	Q	15	47	C	21
17	W	1D	48	V	2A
18	E	24	49	B	32
19	R	2D	50	N	31
20	T	2C	51	M	3A
21	Y	35	52	,	41
22	U	3C	53	.	49
23	I	43	54	/	4A
24	O	44	55	Shift	59
25	P	4D	56	Ctrl	14
26	[	54	57	Alt	11
27	]	5B	58	(Space)	29
28	\	5D	59	Alt	E0, 11
29	Caps Lock	58	60	Ctrl	E0, 14

Special function keys

Key Number	Key Marking	Scan Code	Key Number	Key Marking	Scan Code
61	Esc	76	71	F10	09
62	F1	05	72	F11	78
63	F2	06	73	F12	07
64	F3	04	91	Prnt Scrn	E0, 7C
65	F4	0C	92	Scrll Lck	7E
66	F5	03	93	Pause	E1, 14,
67	F6	0B			77, E1,
68	F7	83			F0, 14
69	F8	0A			F0, 77
70	F9	01			

Numeric keypad

Key Number	Key Marking	Scan Code
74	Num Lock	77
75	/	E0, 4A
76	*	7C
77	-	7B
78	7	6C
79	8	75
80	9	7D
81	+	79
82	4	6B
83	5	73
84	6	74
85	1	69
86	2	72
87	3	7A
88	Enter	E0, 5A
89	0	70
90	.	71

Cursor control

Key Number	Key Marking	Scan Code
94	Insert	E0, 70
95	Home	E0, 6C
96	Page Up	E0, 7D
97	Delete	E0, 71
98	End	E0, 69
99	Page Down	E0, 7A
100		E0, 75
101		E0, 6B
102		E0, 72
103		E0, 74

United Kingdom

Key Number	Key Marking	Scan Code
1	'	0E
3	2	1E
4	3	26
40	'	52
41	#	1C
44	\	66

In addition there is one key that produces a different character in the graphics shift mode:

		Graphics Mode	
Key Number	Scan Code	Code	Character
1	0E	DD	▮

France

Key Number	Key Marking	Scan Code
1	2	66
2	&	16
3	é	1E
4	"	26
5	'	25
6	(	2E
7	-	36
8	è	3D
9	_	3E
10	ç	46
11	à	45
12	)	4E
16	A	15

Key Number	Key Marking	Scan Code
17	Z	1D
26	^	54
27	$	5B
30	Q	1C
39	M	4C
40	ù	52
41	*	5A
44	<	61
45	W	1A
51	,	3A
52	;	41
53	:	49
54	!	4A

In addition there are 12 keys that produce a different character in the graphics shift mode:

		Graphics Mode	
Key Number	Scan Code	Code	Character
3	1E	DD	-
4	26	23	#
5	25	7B	{
6	2E	5B	[
7	36	7C	|
8	3D	60	'

		Graphics Mode	
Key Number	Scan Code	Code	Character
9	3E	5C	\
10	46	5E	^
11	45	40	@
12	4E	5D	]
13	55	7D	}
27	5B	0F	

Germany

Key Number	Key Marking	Scan Code
1	^	0E
3	2	1E
4	3	26
7	6	36
8	7	3D
9	8	3E
10	9	46
11	O	45
12	ß	4E
13	'	55
21	Z	35

Key Number	Key Marking	Scan Code
26	Ü	54
27	+	5B
39	Ö	4C
40	Ä	52
41	#	5A
44	<	61
45	Y	1A
52	,	41
53	.	49
54	-	4A

In addition there are 11 keys that produce a different character in the graphics shift mode:

Key Number	Scan Code	Graphics Mode	
		Code	Character
3	1E	FD	2
4	26	FC	n
8	3D	7B	{
9	3E	5B	[
10	46	5D	]
11	45	7D	}

Key Number	Scan Code	Graphics Mode	
		Code	Character
12	4E	5C	\
16	15	40	@
27	5B	7E	~
44	61	7C	|
51	3A	E6	µ

Italy

Key Number	Key Marking	Scan Code
1	\	0E
3	2	1E
4	3	26
7	6	36
8	7	3D
9	8	3E
10	9	46
11	O	45
12	´	4E
13	Ì	55

Key Number	Key Marking	Scan Code
26	è	54
27	+	5B
39	ò	4C
40	à	52
41	ù	5A
44	<	61
52	,	41
53	.	49
54	-	4A

In addition there are four keys that produce a different character in the graphics shift mode:

		Graphics Mode	
Key Number	Scan Code	Code	Character
26	54	5B	[
27	5B	5D	]
39	4C	40	@
40	52	23	#

Spain

Key Number	Key Marking	Scan Code
1	º	0E
3	2	1E
4	3	26
7	6	36
8	7	3D
9	8	3E
10	9	46
11	0	45
12	´	4E
13	¡	55

Key Number	Key Marking	Scan Code
26	`	54
27	+	5B
39	Ñ	4C
40	´	52
41	Ç	5A
44	<	61
52	,	41
53	.	49
54	-	4A

In addition there are nine keys that produce a different character in the graphics shift mode:

Key Number	Scan Code	Graphics Mode	
		Code	Character
1	0E	5C	\
2	16	7C	|
3	1E	40	@
4	26	23	#
7	36	AA	¬

Key Number	Scan Code	Graphics Mode	
		Code	Character
26	54	5B	[
27	5B	5D	]
40	52	7B	{
41	5A	7D	}

KEYBOARD CONTROLLER

The keyboard controller processes serial data from the keyboard as follows:

1. The keyboard controller receives serial data from the keyboard in the form of a scan code within an 11-bit frame:

Bit	Definition
0	Start Bit
1-8	Data
9	Parity Bit (Odd)
10	Stop Bit

   If the transfer is not completed within 2 ms, the controller places a value of 0FFh in its output buffer, and turns on the transfer time-out bit (6) of its status register (read I/O port 64h).

2. The keyboard controller checks the parity of the data. If a parity error is detected, the controller automatically sends a RESEND command to the keyboard. If the controller continues to detect a parity error, it places a value of 0FFh in its output buffer and turns on the parity bit (7) of its status register (read I/O port 64h).

3. The controller translates the keyboard scan code into a system scan code.

4. The system scan code is placed in the controller's output buffer, then the controller sends a signal to the system indicating that data is present and generates an interrupt.

Data may also be sent to the keyboard by writing to the controller's input buffer (write to I/O port 60h) using the same 11-bit format described above. The controller automatically inserts an odd parity bit. A transfer time-out error will occur if the keyboard does not begin to receive the data within 15 ms, or if the reception process takes longer than 2 ms. If a transfer time-out error occurs, the controller will place a value of 0FEh in its output buffer, and turn on the transfer time-out bit (6) of its status register.

The keyboard must respond to all transfers. Additional data should not be sent to the keyboard until it has acknowledged receipt of the previous data. If the keyboard does not acknowledge receipt of data within 25 ms, the controller places 0FEh in its output buffer and turns on the transfer time-out bit (6) of its status register. If the response from the keyboard contains a parity error, the controller places 0FEh in its output buffer, and turn on the transfer time-out and parity bits (6 and 7) of its status register.

Keyboard controller registers

Access to the keyboard controller is through four registers: an output buffer, a data input buffer, a command input buffer, and a status register. Figure 4-1 shows the I/O port addresses of the keyboard controller registers.

60h	Read	Output buffer
	Write	Input buffer, data
64h	Read	Status register
	Write	Input buffer, command

Figure 3-1. I/O Port Addresses of Keyboard Controller Registers

Input buffers

The keyboard controller has two input buffer addresses. A byte sent to port 60h will be interpreted as a byte of data for the keyboard, and bit 3 of the status register will be set to 0. Writing a byte to port 64h will cause bit 3 of the status register to be set to 1, and the byte will be interpreted as a command to the keyboard controller. Neither address can be written to unless the Input Buffer Full flag (bit 1 of the status register) is 0.

Keyboard controller commands

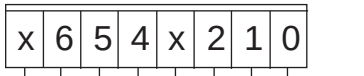
The keyboard controller commands are written to I/O port 64h. In the case of some commands, the command may be followed by a parameter that is written to address 60h when the command is issued to the controller. A list of hexadecimal keyboard controller commands is shown below.

20h Read Command Byte

This command is used to read the current keyboard controller command byte from port 60h. The command byte controls the operation of the keyboard controller. The byte format is explained in the command description below.

60h Write Command Byte

This command can be used to change the keyboard controller command byte that will next be written to port 60h. The format is explained below.



Output-Buffer-Full Interrupt Enable:
When set to 1, the keyboard controller will assert IRQ1 when it has placed data in its output buffer. When set to 0, the interrupt is disabled.

Auxiliary Device Interrupt Enable:
When set to 1, IRQ12 is activated when data from an auxiliary device (such as a mouse) are placed in the output. When set to 0, the interrupt is disabled.

System Flag:
The value of this bit will be copied to the System Flag (bit 2) of the status register.

(Reseved)

Keyboard Disable:
When set to 1, the clock signal of the keyboard is forced low, disabling the keyboard and thereby preventing the receipt or transmission of data.

Auxiliary Device Disable:
When set to 1, the clock signal of the auxiliary device (mouse) is forced LOW, disabling the auxiliary device and thereby preventing the receipt or transmission of data.

PC-compatible mode:
When set to 1, the controller converts the scan code from the keyboard to a PC compatible scan code.

(Reserved, always 0)

Figure 3-2. Keyboard Controller Command Byte

A4h Test Password Installed

This command causes the keyboard controller to verify whether or not a password has been installed. The controller returns the code FAh in its output buffer if the password has been installed, and F1h if the password has not been installed

A5h Load Security

This command causes the keyboard controller to load the password bytes internally. Following this command, the password (up to 45 characters in length) is written to port 60h. The code "00h" is written at the end of the password.

A6h Enable Security

This command enables the password security feature. Once this command is sent to the keyboard controller, no data that is entered through the keyboard is passed to the system until the correct password is entered.

A7h Disable Auxiliary Device

This command forces the clock line for the auxiliary device LOW, disabling the auxiliary device.

This command has the same effect as issuing a command byte with bit 5 set to 1.

A8h Enable Auxiliary Device

This command forces the clock line for the auxiliary device back to the HIGH state, enabling the auxiliary device.

This command has the same effect as issuing a command byte with bit 5 set to 0.

A9h Auxiliary Device Interface Test

This command causes the keyboard controller to test the auxiliary device interface. The controller returns the test result as a status code in the output buffer. The status code is explained below.

00 No error
01 Auxiliary device clock signal stack LOW
02 Auxiliary device clock signal stack HIGH
03 Auxiliary device data signal stack LOW
04 Auxiliary device data signal stack HIGH

AAh Self Test

This command will cause the controller to execute a diagnostics test. If no errors are found, the keyboard controller will place 55h in its output buffer.

ABh Interface Test

This command will cause the controller to execute a diagnostics test of the keyboard interface signals: the clock and data lines. The controller will place a resulting status code in its output buffer, as follows:

00 No error
01 Keyboard clock signal stuck LOW
02 Keyboard clock signal stuck HIGH
03 Keyboard data signal stuck LOW
04 Keyboard data signal stuck HIGH

AD Disable Keyboard

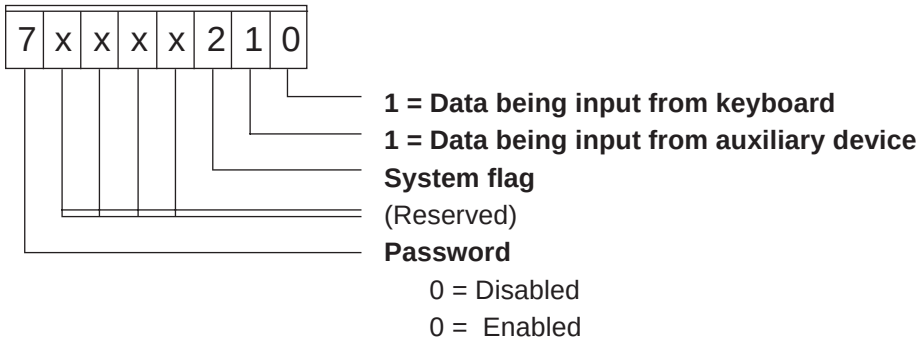
This command forces the clock line for the keyboard back to the high state, enabling the keyboard. This command has the same effect as if the command byte were issued with bit 4 set to "0".

AE Enable Keyboard

This command forces the clock line for the keyboard back to the high state, enabling the keyboard. This command has the same effect as if the command byte were issued with bit 4 set to "0".

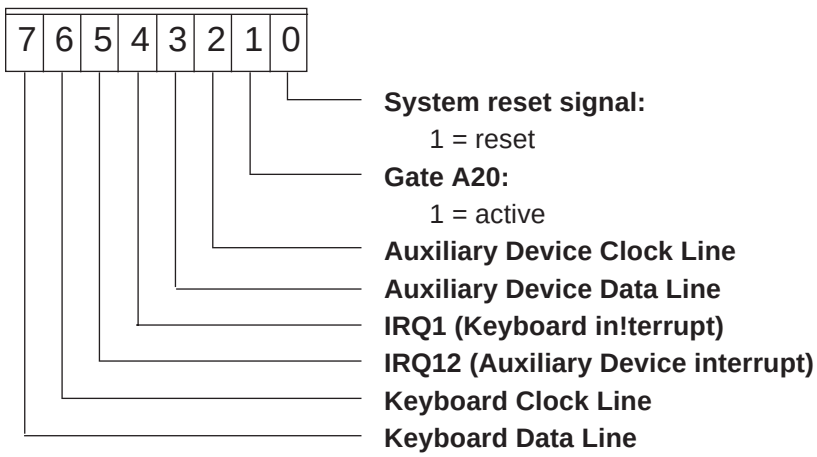
C0 Read Input Port

This command causes the controller to read its input port and place the value in its output buffer. Bit 0 of the status register, Output Buffer Full, should be 0 before this command is issued.



D0 Read Output Port

This command causes the controller to read the contents of its output port and place the resulting value in the output buffer. Bit 0 of the status register, Output Buffer Full, should be 0 before this command is issued. The value of the output port reflects the system status, and follows the format described below, where 0 = Low and 1 = High.



D1 Write Output Port

This command causes the next data written to 60h to be placed in the controller's output oort. This command can be used only to change the status of the Gate A20 signal (bit 1). The other bits are not changed

D2 Echo Keyboard Data

This command causes the next data written to 60h to be placed in the controller's output buffer, and generates keyboard interrupt IRQ1, if enabled.

D3 Echo Auxiliary Device Data

This command causes the next data next written to 60h to be placed in the controller's output buffer, and generates auxiliary device interrupt IRQ12, if enabled.

D4 Write to Auxiliary Device

This command causes the next data written to 60h to be sent to the auxiliary device.

E0 Read Test Inputs

This command causes the controller to read the current status of the two test inputs TEST0 and TEST1, and return the value in its output buffer. TEST0 indicates the status of the auxiliary device clock line, and TEST1 indicates the status of the keyboard clock line.

F0-FF Pulse Output Port LOW

This command causes the controller to force LOW the designated bit(s) for approximately 6É s

Note: Bit 0 of the keyboard controller output port is connected to the system reset signal. When set to 0, a system reset is triggered.

Status register

The keyboard controller status register is defined below in Figure 4-3.

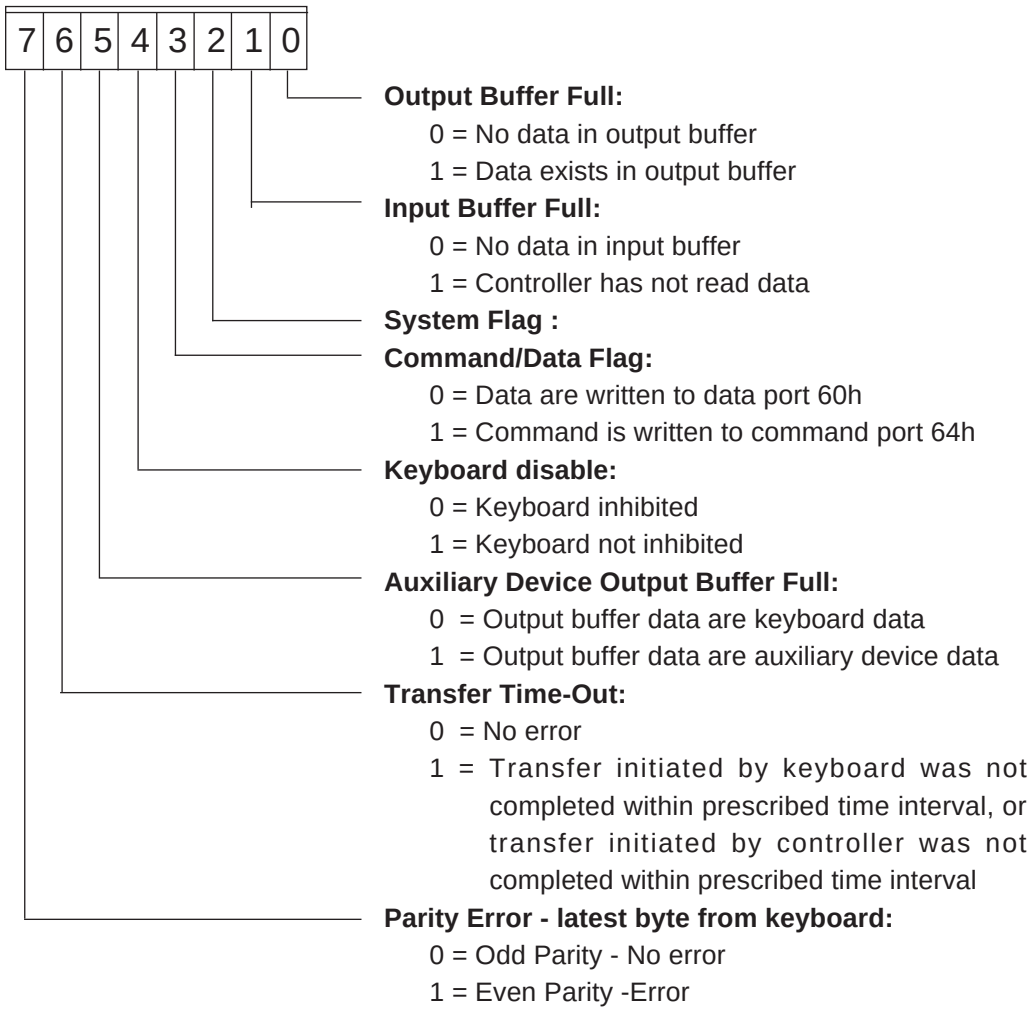


Figure 3-3. Keyboard Controller Status Register

4

Minimizer (Power Management)

Table of Contents

OVERVIEW

"Minimizer" is the name of the power management BIOS provided in the system.

This firmware minimizes power consumption while the system is running application programs. Minimizer can be used with a variety of system applications, and is designed to reduce power consumption without harming the operability of the system.

Minimizer controls the three operating states listed below, both for the system as a whole and for individual devices. For details, refer to the chapter entitled "Functions".

Full operation
Standby
Suspended

In order to actually use Minimizer, enable the Minimizer's power management control bit in CMOS-RAM, and then start up the system. Minimizer is loaded into SM-RAM by the boot block and is initialized in the final step of BIOS startup. The power management function begins to operate afterwards. At this point, the power management control parameters stored in the Minimizer area in CMOS-RAM are read and set. For details on the flow of the Minimizer initialization process and on the setting of the power management control parameters, refer to the chapter entitled "Installation".

Minimizer resides between the operating system and device drivers and the hardware, implementing power management through joint control of the hardware with the OS and the drivers. Information is exchanged between Minimizer and the operating system or device drivers via a BIOS interface called "APM" (Advanced Power Management). In addition, Minimizer also exchanges information with external devices through power management signals. These details are described in the chapter entitled "External Interface".

FUNCTIONS

The Minimizer functions are summarized below.

CPU Standby

Minimizer detects the CPU has been idle for a certain period of time, and then automatically stop the clock of CPU and reduce power consumption. It is also possible to switch the CPU into standby state by using the APM function.
Recovery from CPU standby state is accomplished by the detection of any system activity signal from device or external source (IRQ or NMI, etc.).

Local Standby

Minimizer detects a device has been idle for a certain period of time, and then automatically shut off the supply of power to that device or switch that device into low power consumption state.
This function makes it possible to reduce power consumption when devices are not being used by an application program, compared to power consumption during normal operation.
Minimizer automatically returns such a device back to normal operation when the device is to be used again. This operation is performed in the following manner:
When Minimizer detects an access to a device, it automatically restores the supply of power to that device or switches the device back to normal operation. Although this process may require a certain amount of time for some devices, the process itself is transparent from the standpoint of the application, so there is no effect on the operation of the application programs.
In other words, the process of switching the device from the low power consumption state back to normal operation is not apparent from the application program. After switching the device to normal operation, Minimizer then returns control to the application program..

Global Standby

Minimizer switches the whole system into global standby state if the entire system has been idle for a certain period of time. It is also possible to switch the entire system into global standby state by using the APM function. At the global standby state, all devices are set into standby state and CPU clock is stopped. This function makes further reductions of power consumption possible.
Recovery from global standby state is accomplished by the detection of any system activity signal from device or external source (IRQ or NMI, etc.).

Suspend

Minimizer switches the whole system into suspended state if the entire system has been idle for a certain period of time. It is also possible to switch the entire system into suspended state by either using the APM function or by inputting the suspend button signal. This function makes it possible to further reduce power consumption than global standby. However, devices cannot be replaced or the device configuration cannot be changed while the system is in suspended state.

The system returns to the normal operating state from suspended state in response to either the suspend button signal, the modem ring indicator signal, or the real-time clock alarm. The processing performed by Minimizer is transparent from the standpoint of application programs in these cases as well.

The functions of Minimizer can be enabled or disabled by setting flags in CMOS-RAM (nonvolatile memory). In addition, although Minimizer is intended primarily for use in systems being powered by battery, it can also be used when the system is connected to AC power. In this case, AC power consumption can also be reduced.
Although Minimizer is located in the same flash ROM as BIOS, before operation it is loaded into a part of system memory called SM-RAM, where Minimizer functions independently from BIOS. In addition, in order to avoid having any effect on DOS or BIOS, Minimizer does not invoke any BIOS functions nor DOS functions under power management operation. In order to accomplish independent control of the power supply for individual devices, Minimizer is designed to control the hardware directly.

Minimizer is designed to handle power management for a system's internal memory and devices only; it cannot perform power management for external memory and devices that are added to a system.

CPU Standby

State Control in CPU Standby

Device Name	Normal operation		Standby state
CPU			
CPU	EFI	→	0MHz

STPCLK# = "Low"

State Transitions in CPU Standby

Device Name	Normal operation		Standby state
CPU			
CPU	EFI	→	0MHz

Cause: The system activity defined for [System Event Registers] did not occur within the time set for the CPU standby timer.

EFI	←	0MHz

Cause: The stop break defined for <Stop Break Registers> occurred.ÅB

Definitions for [System Event Registers]

IRQ1, IRQ3-IRQ7,IRQ9-IRQ12, IRQ14, IRQ15
Modem Ring
NMI
I/O Channel Check

The above causes can be enabled/disabled through the settings in CMOS-RAM by Setup program.

Definitions for the <Stop Break Registers>

IRQ1, IRQ3-IRQ7,IRQ9-IRQ12, IRQ14, IRQ15
NMI
I/O Channel Check

The above causes can be enabled/disabled through the settings in CMOS-RAM by Setup program.

Local Standby

State Control in Local Standby

ON: Power is supplied.
OFF: Power is turned off under the control of the SMOUT signal or by VGA controller.

Device Name	Normal operation		Standby state
Internal Video			
VGA controller	Normal operation	→	Standby state
VRAM	Normal refresh	→	Normal refresh
LCD backlight	ON	→	OFF (controlled by the VGA controller)
	FPVEE = "High"		FPVEE = "Low"
COMA			
COMA	ON	→	OFF
	SMOUT0 = "High"		SMOUT0 = "Low"
COMB			
COMB	ON	→	OFF
	SMOUT1 = "High"		SMOUT1 = "Low"
HDD			
Internal HDD	ON	→	OFF
	SMOUT2 = "High"		SMOUT2 = "Low"

Power is not supplied to this device while in local standby state.

Local standby can be enabled/disabled for each individual device by changing the settings in CMOS-RAM through the Setup program.

State Transitions in Local Standby

Device Name	Normal operation		Standby state
Video			
VGA controller	Normal operation	→	Standby state
VRAM	Normal refresh	→	Normal refresh
LCD backlight	ON	→	OFF

Cause: The system activity defined for [I/O Access] did not occur within the time set for the local standby timer.

VGA controller	Normal operation	→	Standby state
VRAM	Normal refresh	→	Normal refresh
LCD backlight	ON	→	OFF

Cause: The stop break event defined for <I/O Trap> occurred.

Power is not supplied to this device while in local standby state.

[I/O Access]

60h

<I/O Trap>

60h

[Remarks]

Video local standby activity and stop break are detected by monitoring the keyboard and PS/2 mouse port. Accessing the video I/O port or VRAM does not have any effect on this activity or stop break.
Video local standby is only available for the internal video only.

Device Name	Normal operation	Standby state
COMA		
COMA	ON	→ OFF
Cause:	The system activity defined for [I/O Access] did not occur within the time set for the local standby timer.	
	ON	→ OFF
Cause:	The stop break event defined for <I/O Trap> occurred.	

Power is not supplied to this device while in local standby state.

[I/O Access]

Determined according to COMA port address setting.
COM1: 3F8h-3FFh, COM2:2F8h-2FFh, COM3:3E8h-3EFh, COM4:2E8h-2EFh

<I/O Trap>

COM1: 3F8h-3FFh, COM2:2F8h-2FFh, COM3:3E8h-3EFh, COM4:2E8h-2EFh

[Remarks]

Because the I/O controller power is on in standby state, the contents of the registers and the initialization states of the ports are retained.
Serial local standby is only available for the internal serial port only.
Do not activate serial local standby when a serial mouse is connected or any communication is connected by this port.

Device Name	Normal operation	Standby state
COMB		
COMB	ON →	OFF

Cause: The system activity defined for [I/O Access] did not occur within the time set for the local standby timer.

	ON →	OFF

Cause: The stop break event defined for <I/O Trap> occurred.

Power is not supplied to this device while in local standby state.

[I/O Access]

Determined according to COMB port address setting.
COM1: 3F8h-3FFh, COM2:2F8h-2FFh, COM3:3E8h-3EFh, COM4:2E8h-2EFh

<I/O Trap>

COM1: 3F8h-3FFh, COM2:2F8h-2FFh, COM3:3E8h-3EFh, COM4:2E8h-2EFh

[Remarks]

Because the I/O controller power is on the standby state, the contents of the registers and the initialization states of the ports are retained.
Serial local standby is only available for the internal serial port only.
Do not activate serial local standby when a serial mouse is connected or any communication is connected by this port.

Device Name	Normal operation	Standby state
HDD		
HDD	ON →	OFF

Cause: The system activity defined for [I/O Access] did not occur within the time set for the local standby timer.

	ON →	OFF

Cause: The stop break event defined for <I/O Trap> occurred.

Power is not supplied to this device while in local standby state.

[I/O Access]

1F0-1F7h, 3F6h

<I/O Trap>

1F0-1F7h, 3F6h

HDD local standby is available for the internal HDD only.

Local standby for an external HDD is based on the external HDD's own standby timer. The Minimizer sends the "standby" command to the HDD during initialization. Subsequently, the Minimizer has no effect regarding local standby for the external HDD.

Global Standby

State Control in Global Standby
ON: Power is supplied.
OFF: Power is turned off under the control of the SMOUT signal or by

Device Name	Normal operation		Standby state
CPU			
CPU	EFI	→	0 MHz STPCLK#="Low"
I/O			
BIOS ROM	ON	→	ON
Internal I/O	ON	→	ON
Memory			
On-board memory	Normal refresh	→	Normal refresh
Internal Video			
VGA controller	Normal operation	→	Standby state
VRAM	Normal refresh	→	Normal refresh
LCD back light	ON	→	OFF (controlled by the VGA controller)
	FPVEE = "High"		FPVEE = "Low"

Keyboard Controller

Keyboard Controller	Normal operation	→	Normal operation

Keyboard

Keyboard	ON	→	ON

Mouse

PS/2 Mouse	ON	→	ON

COMA

COMA	ON	→	(*1)

COMB

COMB	ON	→	(*1)

Parallel port

Parallel port	ON	→	(*1)

FDD

FDC	Normal operation	→	Standby state
FDD	ON	→	(*1)

HDD

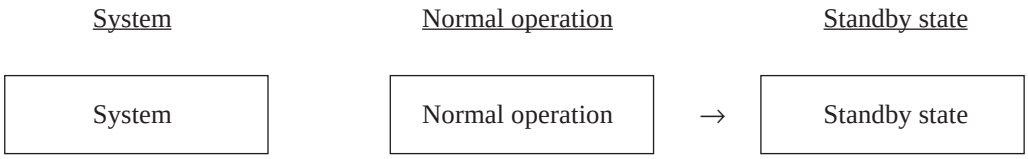
Internal HDD	ON	→	(*1)

Power is not supplied to this device while in global standby state.

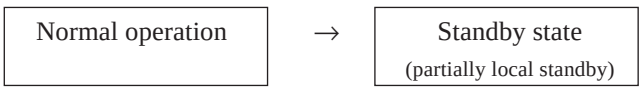
Global standby can be enabled/disabled through the settings in CMOS-RAM, which can be changed by Setup program.
An external HDD is placed in the standby state by means of the "standby immediate" command.

(*1): Indicates that the state depends on the configuration and specifications of the connected hardware.

State Transitions in Global Standby



Cause: The system activity defined for [System Event Registers] did not occur within the time set for the global standby timer.



Cause: The stop break defined for <Stop Break Registers> occurred.

Definitions for [System Event Registers]

IRQ1, IRQ3-IRQ7, IRQ9-IRQ12, IRQ14, IRQ15
Modem Ring
NMI
I/O Channel Check

The above causes can be enabled/disabled through the settings in CMOS-RAM by Setup program.

Definitions for the <Stop Break Registers>

IRQ1, IRQ12
Modem Ring
NMI
I/O Channel Check

The stop breaks of IRQs except IRQ1 and IRQ12 are ineffective during Global standby state, because all devices except keyboard device and PS/2 mouse device are set to standby state during Global standby state.
The above causes can be enabled/disabled through the settings in CMOS-RAM by Setup program.

Suspended State

State Control in Suspended State

ON: Power is supplied.

OFF: Power is turned off under the control of the SMOUT signal and the SUSSTAT# signal or by VGA controller.

Device Name	Normal operation		Standby state
CPU			
CPU	EFI	→	Suspended state
I/O			
BIOS ROM	ON	→	ON
Internal I/O	ON	→	Suspended state
Memory			
On-board memory	Normal refresh	→	Refresh operation only
Internal Video			
VGA controller	Normal operation	→	Suspended state
VRAM	Normal refresh	→	Refresh operation only
LCD back light	ON	→	OFF (controlled by the VGA controller)
	FPVEE = "High"		FPVEE = "Low"

Keyboard Controller

Keyboard Controller	Normal operation	→	Standby state

Keyboard

Keyboard	ON	→	(*1)

Mouse

PS/2 Mouse	ON	→	(*1)

COMA

COMA	ON	→	(*1)

COMB

COMB	ON	→	(*1)

Parallel port

Parallel port	ON	→	(*1)

FDD

FDC	Normal operation	→	Standby state
FDD	ON	→	(*1)

HDD

Internal HDD	ON	→	(*1)

Power is not supplied to this device while in suspended state.

Auto suspend can be enabled/disabled through the settings in CMOS-RAM, which can be changed by Setup program.
(*1): Indicates that the state depends on the configuration and specifications of the connected hardware.
During suspend processing, Minimizer switches the peripherals which are under the control of the system into suspended state or the power off state; the state that the devices are actually switched to depend on the configuration and specifications of the hardware connected to the system. This is because the state depends on how the connected hardware processes the SMOUT signal output and the suspend signal (SUSSTAT#) from the system.
The design method for turning the power off in suspended state is explained here for each individual device. Because the SMOUT signal is used for devices that exercise local standby control, it is recommended that the suspend signal be used for devices that do not exercise to local standby control.

1) Keyboard, mouse
   Local standby processing is not performed for the PS/2 mouse or keyboard. Therefore, the power for these devices can be turned off by adding to the hardware a circuit that turns off the power to the devices when SUSSTAT# signal is detected. During resume processing, it is necessary for the hardware to detect the high level of the SUSSTAT# signal, and then turn the power to the devices back on.

2) COMA, COMB
   Local standby processing can be performed for COMA and COMB. Therefore, the power can be turned off by adding to the hardware a circuit that turns off the power to the devices by using the SMOUT signal. During resume processing, the power for the devices is turned back on when the system sets SMOUT back to on.

3) Parallel port, FDD
   Local standby processing is not performed for the parallel port or FDD. Therefore, the power for these devices can be turned off by adding to the hardware a circuit that turns off the power to the devices when SUSSTAT# signal is detected. During resume processing, it is necessary to the hardware to detect the high level of SUSSTAT# signal, and then turn the power to the devices back on.

4) HDD
   Local standby processing can be performed for the internal HDD. Therefore, the power can be turned off by adding to the hardware a circuit that turns off the power to the internal HDD by using the SMOUT signal. During resume processing, the power for the device is turned back on when the system sets SMOUT back to on.
   An external HDD is placed in the standby state by means of the "standby immediate" command.

Note:

In suspend processing, devices which can be set to local standby are set to local standby state and then Minimizer switches the corresponding SMOUT signal to the low level. After resume processing, the corresponding SMOUT signals are left at the low level and the local standby state is maintained.
This is done in order to reduce system power consumption as much as possible until my access occurs to the local standby devices. Note that the internal video is restored to normal operation state so that the screen display operates.

State Transitions in Suspended State

System	Normal operation		Standby state
System	Normal operation	→	Suspended state

Cause: The system activity defined for [System Event Registers] did not occur within the time set for the Auto Suspend Timer.
Software Suspend request by APM was generated.
Suspend/Resume Button SMI was generated.
Battery Low SMI was generated.

Normal operation	→	Suspended state (partially local standby)

Cause: Suspend/Resume Button was pressed.
Ring Indicator was generated.
Real Time Clock Alarm was generated.

Definitions for [System Event Registers]

IRQ1, IRQ3-IRQ7, IRQ9-IRQ12, IRQ14, IRQ15
Modem Ring
NMI
I/O Channel CheckÉN

The above causes can be changed through the settings in CMOS-RAM by Setup program.
The following causes of resume processing can be enabled/disabled through the settings in CMOS-RAM by Setup program.

Ring Indicator
Real Time Clock Alarm

Note:

If a suspend request is generated while a device is being accessed, suspend processing waits until the access to the device is completed.

Device Information Saved in Suspend State

CPU

CPU

[Registers, states that are saved/restored]
CPU Registers

[Registers, states that cannot be saved/restored]
None

BIOS ROM

[Registers, states that are saved/restored]
None

[Registers, states that cannot be saved/restored]
None

Internal I/O

[Registers, states that are saved/restored]

20-21h	Interrupt Controller 1
70h	CMOS-RAM address port and NMI mask
71h	CMOS-RAM data port
A0-A1h	Interrupt Controller 2

[Registers, states that cannot be saved/restored]
None

Memory

On-board memory

[Registers, states that are saved/restored]
None

[Registers, states that cannot be saved/restored]
None

Keyboard Controller

Keyboard Controller

[Registers, states that are saved/restored]
None

[Registers, states that cannot be saved/restored]
None

Keyboard

Keyboard

[Registers, states that are saved/restored]
- Typematic rate/delay
- LED setting

[Registers, states that cannot be saved/restored]
None

Mouse

PS/2 Mouse

[Registers, states that are saved/restored]
- Scaling
- Resolution
- Sampling Rate

[Registers, states that cannot be saved/restored]
None

FUNCTIONS

COMA

[Registers, states that are saved/restored]
None

[Registers, states that cannot be saved/restored]
None

COMB

[Registers, states that are saved/restored]
None

[Registers, states that cannot be saved/restored]
None

Parallel port

[Registers, states that are saved/restored]
None

[Registers, states that cannot be saved/restored]
None

FDD

[Registers, states that are saved/restored]
None

[Registers, states that cannot be saved/restored]
None

FDD

FDD

[Registers, states that are saved/restored]
None

[Registers, states that cannot be saved/restored]
None

HDD

HDD

[Registers, states that are saved/restored]

1F2h	Sector Count Register
1F3h	Sector Number Register
1F4h	Cylinder Low Register
1F5h	Cylinder High Register
1F6h	SDH Register

HDD initialization states
-Port check
-Self-diagnose
-Set parameter

[Registers, states that cannot be saved/restored]

1F0h	Data Register
1F1h	Error Register
1F7h	Status Register

Power is not supplied to this device while in suspended state.

Notes on Suspended State

Device Changes

Because Minimizer does not check for device changes during resume processing, changing the hardware configuration (video, HDD, keyboard, PS/2 mouse, etc.) while the system is in the suspended state may result in problems with operations after the resume processing is executed.

Suspend Warning Timer

During device initializing process, for example the mouse driver initialization, the system should not enter the suspend state, because Minimizer cannot save the initializing state of a device for suspend/resume processing.

To avoid this, you should set the Suspend Warning Timer with enough margin. This means the Suspend Warning Timer should be longer than the internal time of I/O access during the device initialization.

The Suspend Warning Timer is only valid for the suspend request issued by pressing the suspend/resume button. The Suspend Warning Timer is set to two seconds.

APM connection

When APM was used to set the local device to the standby or suspend condition, the device must be returned to the normal operating condition before being next accessed. If the device is accessed or a BIOS function using that device is called (INT xxh) without first returning the device to the normal condition, the system may become unstable.

INSTALLATION

Parameters in CMOS-RAM

The following power management control parameters are stored in the extended CMOS-RAM area used for Minimizer.

Item	Range of Settings	Memory Used
Powermanagement	Enable/Disable	1 bit
Time and Date correction	Enable/Disable	1 bit
CPU standby	Disable, 4, 8, 12, . . . 1020 seconds	1 byte
Globalstandby	Disable, 4, 8, 12, . . . 262140 seconds	2 byte
Autosuspend	Disable, 4, 8, 12, . . . 131068 seconds	2 byte

Item	Range of Settings	Memory Used
Localstandby		
-Video	Disable, 4, 8, 12, . . . 1020 seconds	1 byte
-COMA	Disable, 4, 8, 12, . . . 1020 seconds	1 byte
-COMB	Disable, 4, 8, 12, . . . 1020 seconds	1 byte
-HDD	Disable, 4, 8, 12, . . . 1020 seconds	1 byte
Batterylowcontrol		
-Suspend on Low battery	Enable/Disable	1 bit
-Warning beep	Enable/Disable	1 bit
Resumecontrol		
-Modem ring	Enable/Disable	1 bit
-Alarm	Enable/Disable[hh:mm:ss] hh=0-23, mm=0-59, ss=0-59	1 bit +3 bytes
System event setting	IRQ1, IRQ3-IRQ7, IRQ9-IRQ12, IRQ14, IRQ15, Modem ring, NMI, I/O Channel Check	3 bytes
Stop break setting	IRQ1, IRQ3-IRQ7, IRQ9-IRQ12, IRQ14, IRQ15, Modem ring, NMI, I/O Channel Check	3 bytes

All control parameters can be set/changed by the Setup program.

For proper Minimizer operation, set appropriate values into CMOS-RAM with the Setup program and re-boot the system. When the Power management bit is enabled, the power management function is activated.

CPU Standby

The following procedure is needed in order to use the CPU standby function.

1) Set the Power management bit in CMOS-RAM to "enabled" by the Setup program. The proper checksum for the Minimizer CMOS-RAM is also set at the same time.
   If the system has no RTC battery backup, use the Parameter Modification Tool of the ROM Adaptation Kit to set the default CMOS settings in the BIOS code in the same way.

2) Set the CPU Standby Time in CMOS-RAM.
   Use the Setup program to set a suitable value for the CPU Standby Time in CMOS-RAM. The possible settings are "disable" or from 4 to 1020 seconds in four-second increments.
   If the system has no RTC battery backup, use the Parameter Modification Tool of the ROM Adaptation Kit to set the default settings for the BIOS code in the same way.

3) Set the System Event information in CMOS-RAM.
   Use the Setup program to set a suitable value for the System Event information in CMOS-RAM. The System Event information is used to inform to the Minimizer that the system is operating; if the specified event does not occur within the period of time set for CPU standby, the CPU enters to standby state.

   The system events listed below can be set. (More than one event can be set.)
   - IRQ1, IRQ3-IRQ7, IRQ9-IRQ12,IRQ14,IRQ15
   - Modem Ring
   - NMI
   - I/O Channel Check

System event

Proceed with care when selecting a system event, and allow the IRQ generated by the device to be recognized as a system event. Do not select a non-connected IRQ not generated by a device, because such an interrupt may be stacked HIGH.
If the system has no RTC battery backup, use the Parameter Modification Tool of the ROM Adaptation Kit to set the default CMOS settings in the BIOS code in the same way.

4) Set the Stop Break information in CMOS-RAM.
   Use the Setup program to set a suitable value for the Stop Break information in CMOS-RAM. The Stop Break information is a trigger for switching the CPU from standby state to normal operation state; if the specified event occurs, the CPU is released from standby state and returns to normal operation state.

   The stop breaks listed below can be set. (More than one event can be set.)
   - IRQ1, IRQ3-IRQ7, IRQ9-IRQ12,IRQ14,IRQ15
   - NMI
   - I/O Channel Check

Proceed with care when selecting a stop break event, and allow the IRQ generated by the device to be recognized as a stop break event. Do not select a non-connected IRQ not generated by a device, because such an interrupt may be stacked HIGH.
If the system has no RTC battery backup, use the parameter modification table of the ROM Adaptation Kit to set the BIOS code default CMOS-RAM according to the same method.

5) Write the default CMOS-RAM settings to the FLASH ROM.
   If the system has no RTC battery backup, use a dedicated ROM writer to write the BIOS code in the flash ROM, using the default CMOS-RAM values that were set.

6) Perform a system reset.
   Reset the system. The Minimizer shall be initialized with the specified settings, and the CPU standby function is enabled.

Local Standby

The following procedure is needed in order to use the local standby function.

1) Set the Power management bit in CMOS-RAM to "enabled" by the Setup program. The proper checksum for the Minimizer CMOS-RAM is also set at the same time.
   If the system has no RTC battery backup, use the Parameter Modification Tool of the ROM Adaptation Kit to set the default CMOS settings in the BIOS code in the same way.

2) Set the Local Standby Time in CMOS-RAM.
   Use the Setup program to set a suitable value for the Local Standby Time in CMOS-RAM. The possible settings are "disable" or from 4 to 1020 seconds in four-second increments.
   If the system has no RTC battery backup, use the Parameter Modification Tool of the ROM Adaptation Kit to set the default CMOS settings in the BIOS code in the same way.

3) Write the Minimizer setting information and the default CMOS settings to the flash ROM.
   Write the BIOS code with the altered Minimizer parameter table and default CMOS setting values to the flash ROM using the CARD Type PC ROM Writer.
   If the system has RTC battery backup, this step is unnecessary.

4) Reset the system.
   Reset the system; when the system starts up again, the Minimizer is initialized using the specified information and local standby begins functioning for devices.

Global Standby

The following procedure is needed in order to use the global standby function.

1) Set the Power management bit in CMOS-RAM to "enabled" by the Setup program. The proper checksum for the Minimizer CMOS-RAM is also set at the same time.
   If the system has no RTC battery backup, use the Parameter Modification Tool of the ROM Adaptation Kit to set the default CMOS settings in the BIOS code in the same way.

2) Set the Global Standby Time in CMOS-RAM.
   Use the Setup program to set a suitable value for the Global Standby Time in CMOS-RAM. The possible settings are "disable" or from 4 to 262140 seconds in four-second increments.
   If the system has no RTC battery backup, use the Parameter Modification Tool of the ROM Adaptation Kit to set the default CMOS settings in the BIOS code in the same way.

3) Set the System Event information in CMOS-RAM.
   Use the Setup program to set a suitable value for the System Event information in CMOS-RAM. The System Event information is used to inform to the Minimizer that the system is operating; if the specified event does not occur within the period of time set for global standby, the systems switches to global standby state. In global standby state, local devices are also set to standby state, and all the SMOUT signal assigned to local devices are set to Off.
   The system events listed below can be set. (More than one event can be set.)
   - IRQ1, IRQ3-IRQ7, IRQ9-IRQ12,IRQ14,IRQ15
   - Modem Ring
   - NMI
   - I/O Channel Check

   The system events should be carefully selected so that the IRQs which devices generate can be recognized as the system event. Note that the non-connected IRQs which devices never generate should not be selected, because non-connected IRQs sometimes get stuck to HIGH.
   If the system has no RTC battery backup, use the Parameter Modification Tool of the ROM Adaptation Kit to set the default CMOS settings in the BIOS code in the same way.

4) Set the Stop Break information in CMOS-RAM.
Use the Setup program to set a suitable value for the Stop Break information in CMOS-RAM. The Stop Break information is a trigger for switching the system from standby state to normal operation state; if the specified event occurs, the system is released from standby state and returns to normal operation state. Upon recovery from the global standby state, local devices remain in the local standby state.

The stop breaks listed below can be set. (More than one event can be set.)
IRQ1, IRQ12
Modem Ring
NMI
I/O Channel Check

The stop breaks should be carefully selected so that the IRQs which devices generate can be recognized as the stop break. Note that the non-connected IRQs which devices never generate should not be selected, because non connected IRQs sometimes get stuck to HIGH. If the system has no RTC battery backup, use the Parameter Modification Tool of the ROM Adaptation Kit to set the default CMOS settings in the BIOS code in the same way.

5) Write the Minimizer setting information and the default CMOS settings to the flash ROM.
Write the BIOS code with the altered Minimizer parameter table and default CMOS setting values to the flash ROM using the CARD Type PC ROM Writer.
If the system has RTC battery backup, this step is unnecessary.

6) Reset the system.
Reset the system; when the system starts up again, the Minimizer is initialized using the specified information and global standby begins functioning.

Suspend

The following procedure is needed in order to use the suspend function.

1) Set the Power management bit in CMOS-RAM to "enabled" by the Setup program. The proper checksum for the Minimizer CMOS-RAM is also set at the same time.
If the system has no RTC battery backup, use the Parameter Modification Tool of the ROM Adaptation Kit to set the default CMOS settings in the BIOS code in the same way.

2) Set the Auto Suspend Time in CMOS-RAM.
To switch the system to suspended state by using just the APM function or the SRBTN# signal, set the Auto Suspend Time to "disable".
Use the Setup program to set value for the Auto Suspend Time in CMOS-RAM. The possible settings are "disable" or from 4 to 262140 seconds in four-second increments.
If the system has no RTC battery backup, use the Parameter Modification Tool of the ROM Adaptation Kit to set the default CMOS settings in the BIOS code in the same way.

3) Set the System Event information in CMOS-RAM.
Use the Setup program to set a suitable value for the System Event information in CMOS-RAM. The System Event information is used to inform to the CPU that the system is operating; if the specified event does not occur within the period of time set for suspend, the systems switches to suspended state. In suspended state, local devices are also set to suspended state, and all the SMOUT signal assigned to local devices are set to Off.

The system events listed below can be set. (More than one event can be set.)
IRQ1, IRQ3-IRQ7, IRQ9-IRQ12,IRQ14,IRQ15
Modem Ring
NMI
I/O Channel Check

The system events should be carefully selected so that the IRQs which devices generate can be recognized as the system event. Note that the non-connected IRQs which devices never generate should not be selected, because non-connected IRQs sometimes get stuck to HIGH.
If the system has no RTC battery backup, use the Parameter Modification Tool of the ROM Adaptation Kit to set the default CMOS settings in the BIOS code in the same way.

4) Set the Battery Low Control in CMOS-RAM.
Use the Setup program to set selections for the Low Battery Suspend Control and Warning Beep Control in CMOS-RAM. The possible settings are "disable" or "enable". If the Low Battery Suspend Control is enabled, the system enters the suspended state immediately on the battery low condition. If the Warning Beep Control is enabled, the Minimizer beeps when in the battery warning condition or battery low condition, and before entering the suspended state.
If the system has no RTC battery backup, use the Parameter Modification Tool of the ROM Adaptation Kit to set the default CMOS settings in the BIOS code in the same way.

5) Set the Resume Control in CMOS-RAM.
   Use the Setup program to set selections for the Alarm Resume Control and Modem Ring Resume Control in RTC. The possible settings are "disable" or "enable". If the Alarm Resume Control is enabled, the system recovers from suspended state by RTC alarm. If the Modem Ring Resume Control is enabled, the system returns from the suspended state in response to a modem ring on COMA or COMB.
   If the system has no RTC battery backup, use the Parameter Modification Tool of the ROM Adaptation Kit to set the default CMOS settings in the BIOS code in the same way.

6) Set the Resume Alarm Time in CMOS-RAM.
   Perform this processing when the Alarm Resume Control is enabled.
   Use the Setup program to set Resume Alarm Hour, Resume Alarm Minute, Resume Alarm Second in CMOS-RAM.
   If the system has no RTC battery backup, use the Parameter Modification Tool of the ROM Adaptation Kit to set the default CMOS settings in the BIOS code in the same way.

7) Write the Minimizer setting information and the default CMOS settings to the flash ROM.
   Write the BIOS code with the altered Minimizer parameter table and default CMOS setting values to the flash ROM using the CARD Type PC ROM Writer.
   If the system has RTC battery backup, this step is unnecessary.

8) Reset the system.
   Reset the system; when the system starts up again, the Suspend/Resume function operates in accordance with the settings.

EXTERNAL INTERFACE

The Minimizer resides between the operating system and device drivers and the hardware, and performs power management in coordination with the software. ...
For this purpose, information is exchanged between the Minimizer and the operating system or device driver via the APM (Advanced Power Management) BIOS interface.

The data concerning power management sent from the system to the external hardware is:

SMOUT (System Management OUTput,n=0,1,2,3)	4 signals
SUSSTAT# (SUSpend STATus)	1 signal

The data concerning power management sent from the external hardware to the system is:

EXTSMI# (External System Management Interrupt)	1 signal
SRBTN# (Suspend Resume BuTtoN)	1 signal
BATWRN# (BATtery WaRNing)	1 signal
BATLOW# (BATtery LOW)	1 signal

Regarding EXTSMI#, because the corresponding processing is not defined in Minimizer, SMI interrupts are disabled. (Reserved)

APM Interface

The APM interface provided is based on the specifications for "Advanced Power Management Version 1. 0" and "Advanced Power Management Revision 1. 1".
An entry from application programs to APM is provided through INT 15h.

```
mov   AH,53h
mov   AL, function number
mov   BX, device ID
int   15h
```

The APM functions are listed below. All functions are supported by Minimizer.

Function number	Processing
00h	Installationcheck
01h	InterfaceConnect
02h	Protect Mode Connect 16 bit
03h	Protect Mode Connect 32 bit
04h	InterfaceDisconnect
05h	CPU Idle
06h	CPU Busy
07h	Set Power State - System Stand-by - System Suspend
08h	Enable/Disable Power Management Function
09h	Restore Power on defaults
0Ah	Get Power Status
0Bh	Get PM Event
*0Ch	Get Power State
*0Dh	Enable/Disable Device Power Management
*0Eh	APM Driver Version
*0Fh	Engage/Disengage Power Management
80h	OEM APM Function

*Function number marked with asterisk are APM Revision 1.1 functions.

If the APM interface is connected, then when a power management request (SMI) is generated, the Minimizer maintains SMI request information and waits for instructions from the APM interface. The application program (OS) can get requests from the Minimizer and then perform power management processing for the Minimizer.

For details concerning the APM interface, refer to the specifications for "Advanced Power Management Version 1. 0" and "Advanced Power Management Revision 1. 1".

APM OEM Function

APM parameters can be changed while the Minimizer is running by calling APM OEM functions.

Parameter index

This section explains the indexes that are used to access Minimizer parameters, and explains the parameters..

Index	Size	Parameter
00h	Byte	Bit 7 : Power Management Control = 1 : Enable = 0 : Disable All other bits are ignored.
01h	Byte	CPU Standby Time 00h : Disable 01h-FFh : 4-1020 seconds (in units of 4 seconds)
02h	Word	Global Standby Time 0000h : Disable 0001h - FFFFh : 4 - 262140 seconds (in units of 4 seconds)
04h	Word	Auto suspend time 0000h : Disable 0001h - FFFFh : 4 - 262140 seconds (in units of 4 seconds)
06h	Byte	Video Local Standby Time 0000h : Disable 01h-FFh : 4-1020 seconds (in units of 4 seconds)
07h	Byte	COMA Local Standby Time 00h : Disable 01h-FFh : 4-1020 seconds (in units of 4 seconds)
08h	Byte	COMB Local Standby Time 00h : Disable 01h-FFh : 4-1020 seconds (in units of 4 seconds)
09h	Byte	HDD Local Standby Time 00h : Disable 01h-FFh : 4-1020 seconds (in units of 4 seconds)
0Ch	Byte	Bit 5 : Low Battery Suspend Control = 0 : Do not enter suspend state = 1 : Enter suspend state on low battery condition. All other bits are ignored.

Index	Size	Parameter
0DH	Byte	Bit 6 : Modem Ring Resume Control 0 = Disable 1 = Enable Bit 5 : Alarm Resume Control 0 = Disable 1 = Enable All other bits are ignored.
0Eh	Byte	Resume alarm hours (BCD)
0Fh	Byte	Resume alarm minutes (BCD)
10h	Byte	Resume alarm seconds (BCD)

Read function

This function reads the current setting of a Minimizer parameter.

INPUT		
(AX)=	5380h	
(BH)=	00h	
(BL)=	Parameter Index	

OUTPUT		
(CX)=	Parameter value If the parameter is a byte data, only (CL) has valid parameter value.	
(CF)=	Error flag	0 = no error 1 = error

Set function

This function writes a new setting for a Minimizer parameter.

INPUT		
(AX)=	5380h	
(BH)=	01h	
(BL)=	Parameter Index	
(CX)=	Parameter value If the parameter is a byte data, only (CL) is used.	

OUTPUT		
(CF)=	Error flag	0 = no error 1 = error

Note:

Parameters set by this function takes place when the activate function (described later) is called.
Two or more parameters can be continuously set before calling the activate function.

Activate function

This function activates new settings.

INPUT	
(AX)=	5380h
(BH)=	02h

OUTPUT	
(CF)=	0

Note:

Executing this function resets the CPU standby timer, the global standby timer, the automatic suspend timer, and the local standby timers to their initial values, and starts them counting down again.

Cancel function

This function returns all Minimizer parameters to the values that are saved in CMOS-RAM.

INPUT	
(AX)=	5380h
(BH)=	03h

OUTPUT	
(CF)=	0

Note:

Executing this function resets the CPU standby timer, the global standby timer, the automatic suspend timer, and the local standby timers to their initial values, and starts them counting down again.

SMOUT Pin Assignments

The SMOUT pins are used to turn on/off the supply of power to the devices that are subject to local standby control by the Minimizer.
There are four SMOUT signals in the system interface; these four signal can be used for local standby control for a maximum of four types of devices. (Local standby control for the internal VGA controller is accomplished without using any SMOUT signals.)
The followings are default SMOUT signal assignment:

SMOUT0 COMA
1 - On
0 - Off

SMOUT1 COMB
1 - On
0 - Off

SMOUT2 HDD
1 - On
0 - Off

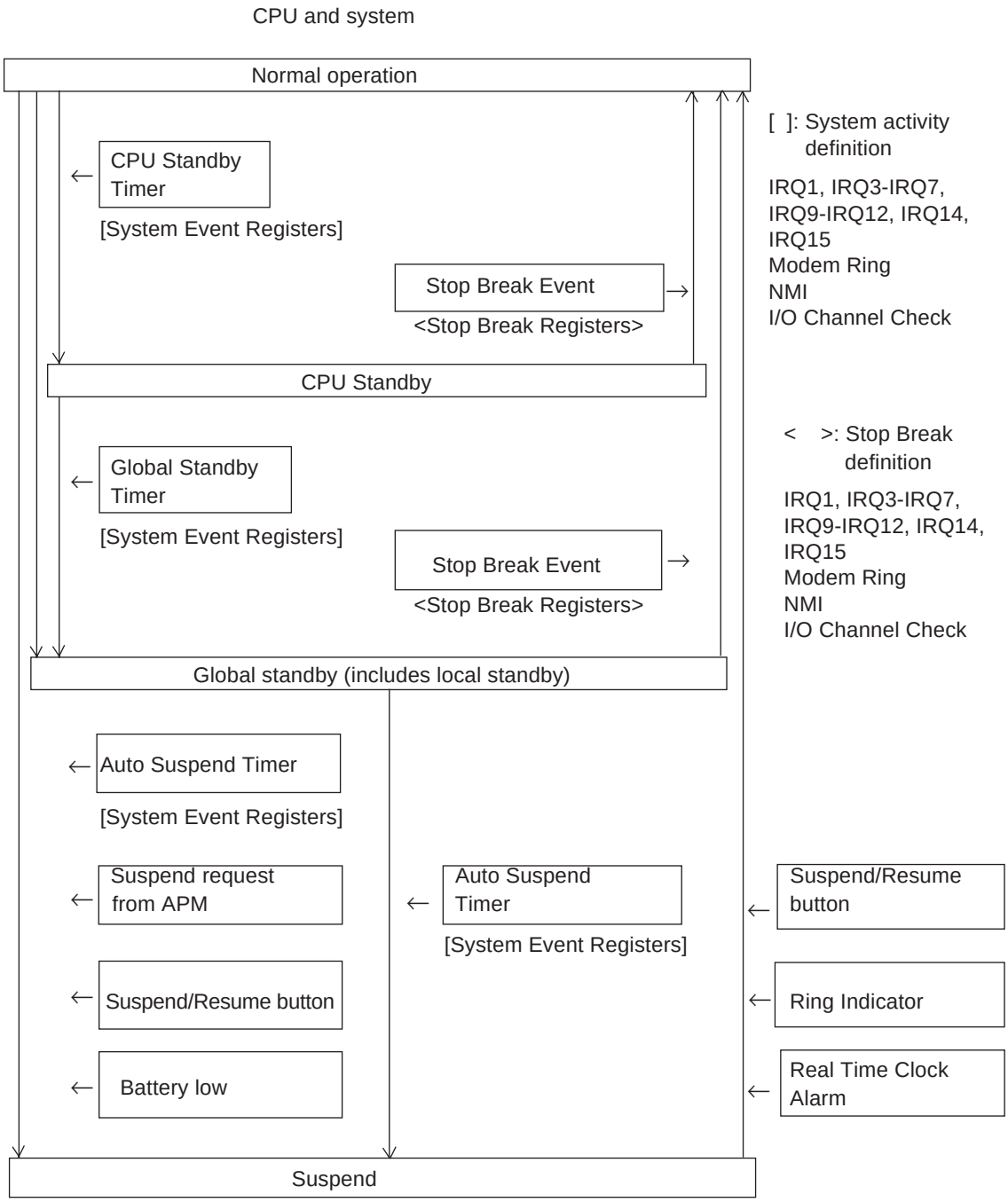
SMOUT3 -VPPON (flash ROM Vpp -12V)
1 - On
0 - Off
Flash ROM erasure/writing

Low Battery Detection

The Minimizer's low battery processing is executed by setting the BATWRN# or BATLOW# signals low from the system interface.
Because the system does not have an internal means of checking the battery power status or the remaining battery capacity, whenever the system is being powered by battery and the remaining battery capacity becomes low, the external hardware should first input the BATWRN# signal through the system interface.
Minimizer sounds a beep during BATWRN# signal is low if the Warning Beep Control in CMOS-RAM is enabled.
When the remaining battery capacity becomes even lower, the external hardware should input the BATLOW# signal through the system interface.

Minimizer sounds a beep if the Warning Beep Control in CMOS-RAM is enabled, and switches the system to suspended state if the Low Battery Suspend Control is enabled. This reduces the system's power consumption, extending the time until the battery becomes dead.

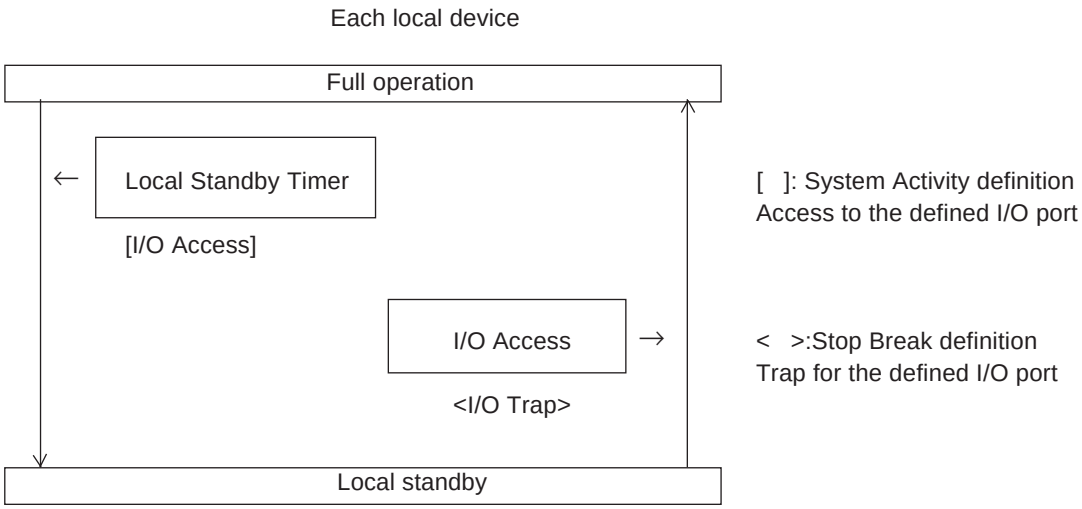
SYSTEM STATE TRANSITION DIAGRAM



Note:

When recovering from suspended state, devices other than the internal video returned to local standby state if local standby functionality are enabled for the device. The internal video returns to the normal operating state for displaying the screen.

SYSTEM STATE TRANSITION DIAGRAM



Note:

The four devices listed below can use the local standby function.

Internal video
HDD
COMA
COMB

5

EPSON DBIOS Setup

Table of Contents

1 INTRODUCTION

1.1 What Is "BIOS Setup?"

All IBM PC/AT-compatible systems store information concerning peripheral devices, etc., in CMOS-RAM. "BIOS setup" (or simply "setup") refers to the process of setting the contents of this CMOS-RAM in accordance with the system BIOS. "SETUP" is the name of the BIOS setup program for Epson's IBM PC/AT-compatible BIOS ("EPSON DBIOS"). Two features of SETUP are its menu-driven operation and a function that automatically corrects the settings.

Note:
Because SETUP displays values in its menus after they have already been corrected automatically, it is possible that the information that is displayed does not accurately reflect the values stored in CMOS-RAM. As a result, SETUP is not suitable for debugging work. Note that SETUP will not operate properly if it is used with any BIOS other than EPSON DBIOS.

1.2 Why Is BIOS Setup Necessary?

When the system is started up (i.e., the power is turned on), the BIOS initializes the system and executes the system diagnostics. CMOS-RAM (a form of auxiliary storage) is referenced during these operations for the following reasons:

- In order to speed up the initialization of devices for which automatic detection would be slow (for example, FDDs and main memory
- In order to initialize devices that support multiple operating modes and optional functions (for example, parallel ports and video cards)
- In order to permit use of the BIOS extended functions as best suited for the system (for example, power management)
- In order to explicitly disable specific devices and BIOS functions (as might be necessary for special applications)

Therefore, it is essential to execute the BIOS setup processing and set CMOS-RAM as appropriate.
In addition, it is also necessary to execute BIOS setup again if the device configuration is changed (by adding a hard disk drive, for example) or if the system diagnostics that are executed by the BIOS discover an error.

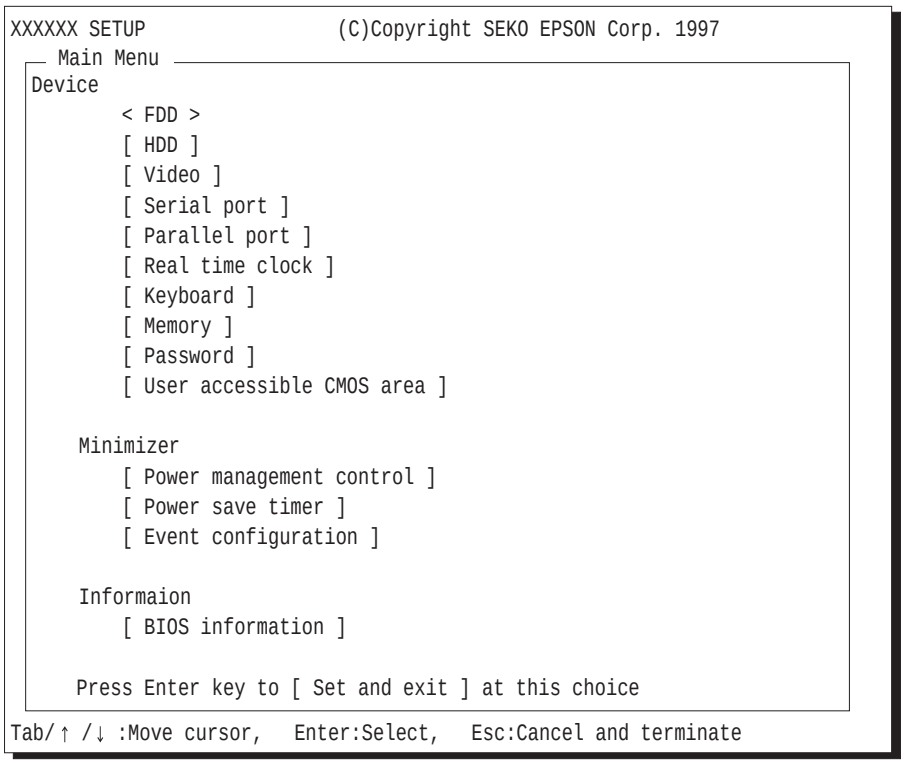
Note:
EPSON DBIOS includes a function that copies an image of CMOS-RAM that is stored in ROM to CMOS-RAM when the power is turned on, eliminating the need for executing the setup processing. The ROM Adaptation Kit is required in order to use this function. Furthermore, other restrictions on the use of this function make it unsuitable for general applications.

2 SETUP OPERATION METHOD

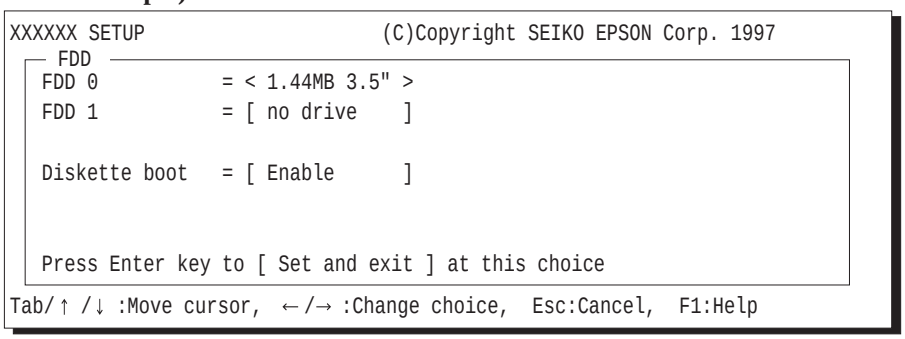
2.1 Overview of SETUP

SETUP is a menu-driven CMOS-RAM setup program. EPSON DBIOS includes approximately 60 items to be set in CMOS-RAM; because SETUP has these items grouped into sub menus by device or function, the necessary settings can be made efficiently.
SETUP is operated through the keyboard. The Tab key, the arrow keys, the Enter key and the F1 key can all be used. Examples of SETUP's main menu and a sub menu are shown below.

(Main Menu Example)



(Sub Menu Example)



If the meaning of a menu item or term is unclear, a help message can be displayed by moving the cursor to that item and then pressing the F1 key.

2.2 Operating Procedure

The basic EPSON DBIOS setup procedure is as follows:

(1) Prepare for SETUP.
(2) Start up SETUP.
(3) Confirm results of CMOS-RAM check.
(4) Perform menu operations (changing settings).
(5) Exit SETUP (overwrite CMOS-RAM and reset the system)

2.2.1 Prepare for SETUP

A keyboard and a display (video system) are required in order to be able to run SETUP. The actual SETUP program itself is embedded in the EPSON DBIOS.

2.2.2 Start up SETUP

To start up SETUP, turn the system power on and then press the SETUP key (the F10 key) before the operating system starts up. The following message appears during the system startup processing:

```
"Press <F10> to run SETUP"
```

2.2.3 Confirm Results of CMOS-RAM Check

SETUP first checks whether the current CMOS-RAM settings are correct or not. The program displays the following message, indicating that the program is currently checking the settings in CMOS-RAM:

```
"...checking CMOS contents. Please wait."
```

If no errors are found, the program automatically switches to the menu screen. If an error is found, SETUP displays an error message and then automatically corrects the item in question. Once all items have been checked and automatically corrected where necessary, the program displays the following confirmation before proceeding to the menu screen:

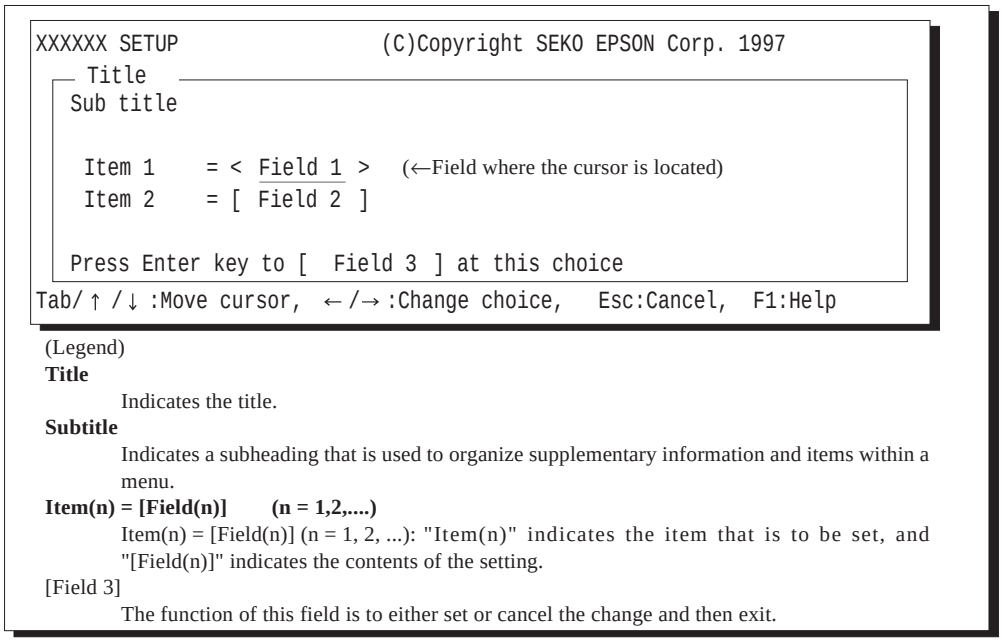
```
"...corrected. Press any key to continue"
```

The error messages are explained in another section of this manual. Because the contents that are automatically corrected vary for different items, these items are explained individually in other sections that explain individual sub menus (items).

2.2.4 Perform Menu Operations

Select a sub menu from the main menu, and then make the appropriate settings in the sub menu. Exit the sub menu, and return to the main menu. Repeat this procedure. Guidance messages concerning the expected keyboard input, etc., are always displayed at the bottom of the screen.

The operation method is explained below using the following example.



(Legend)

Title

Indicates the title.

Subtitle

Indicates a subheading that is used to organize supplementary information and items within a menu.

Item(n) = [Field(n)] (n = 1,2,....)

Item(n) = [Field(n)] (n = 1, 2, ...): "Item(n)" indicates the item that is to be set, and "[Field(n)]" indicates the contents of the setting.

[Field 3]

The function of this field is to either set or cancel the change and then exit.

Move cursor

Press [Tab], [Shift] + [Tab], or the up and down arrow keys in order to move the cursor between fields.

Change choice

Press the left and right arrow keys to change the setting in the field where the cursor is located. Depending on the field, some accept numeric input, while others accept text input.

Set and exit

There is always a field ("[Field 3]" in the above example) contained in the message **"Press Enter key to [ ] at this choice"** at the bottom of the menu screen. Moving the cursor here and selecting [Set and exit] saves the changes and exits the menu.

Cancel and exit

Selecting [Cancel and exit] for **" Press Enter...[ ]" ("[Field 3]"** in the above example) cancels the changes and exits the menu. The [Esc] key also performs the same function.

Help

Pressing the [F1] key displays the help message for that field.

2.2.5 Exit SETUP (Overwrite CMOS-RAM and Reset the System)

Selecting [Set and exit] in the message **"Press Enter key to [ ] at this choice"** at the bottom of the main menu saves the settings for all menu items in CMOS-RAM and then resets the system and exits SETUP. To cancel the changes and exit SETUP, either select [Cancel and exit] in the main menu, or press the [Esc] key.

2.3 Sub Menus

2.3.1 Overview of Sub Menus

The organization of the SETUP sub menus and the main items to be set are listed below.

Device: These are setting items that are grouped by device.

[FDD] (Floppy Disk Drive)
FDD type, prohibiting bootup from floppy disk

[HDD] (Hard Disk Drive)
HDD type, user-defined parameters, ATA boot

[Video] (Internal VGA Controller)
Display selection, LCD (panel display) option functions

[Serial port]
Serial port number (I/O address), infrared communications (IrDA)

[Parallel port]
Parallel port number (I/O address), bi-directional parallel, EPP

[Real time clock] (RTC)
Time, date

[Keyboard]
NumLock status, Typematic (key repeat speed)

[Memory]
Size of extended memory on the ISA bus

[Password]
Password setting

[User accessible CMOS area] (User CMOS area)
Set any value in the CMOS-RAM index from 70h to 7Fh

Minimizer: These are items related to the power management BIOS (Minimizer).

[Power management control]
Power management enabled/disabled, warnings/countermeasures when battery is low, resume in response to modem or alarm

[Power save timer]
System power saver (CPU standby, Global standby, Auto suspend) and device (Video, HDD, COMA, and COMB) auto standby timer

[Event configuration] (System power saver event settings)
Idle detection event setting (System event), standby recovery event setting (Stop break event)

Information: This is a supplementary information item

[BIOS information]
Version information, etc., display

2.3.2 Explanation of Individual Sub Menus

This section explains how to set each item in the SETUP sub menus. For the sake of convenience, the setting items will be expressed in the format "Item = [Field]".

2.3.2.1 [FDD] (Floppy Disk Drive)

FDD (n) = [(drive type)] (n=0, 1)

This item sets the drive type for FDD 0 and FDD 1.

[(drive type)]

[No drive] :
No drive, or the drive is disabled.

[360KB 5.25"], [1.2MB 5.25"], [720KB 3.5"], [1.44MB 3.5"] :
Use the selected type of floppy disk drive.

Effect of the automatic correction feature:

If no drive is detected and this item is set to anything other than [No drive], the setting is automatically corrected to [No drive].

Diskette boot = [Disable/Enable]

This item enables/disables bootup from floppy disk.

[Disable/Enable]

[Disable] : Bootup from floppy disk is prohibited.

[Enable] : If the system includes both an FDD and an HDD, bootup from floppy disk takes priority.

2.3.2.2 [HDD] Hard Disk Drive

Primary/Secondary IDE = [(IDE interface setting)]

Master/Slave = [(drive type)] [cyl.] [head] [sec.]

EPSON DBIOS supports two IDE interfaces, one primary (1F0h) and one secondary (170h). Because a master drive and a slave drive can be connected to each IDE interface, up to four HDDs can be used. This item also is used to set "ATA boot" (a function that allocates a PCMCIA socket to an IDE interface, allowing an ATA PC card to be used directly as an HDD.

[(IDE interface setting)]

[Disable] :
Disables the IDE interface. (No HDDs are used.)

[Enable] :
Uses the interface as a normal IDE interface. (HDDs are used.)

[PCMCIA Socket#0] :
Assigns the PC card in PCMCIA socket #0 (card slot 0) as an HDD.

[PCMCIA Socket#1] :
Assigns the PC card in PCMCIA socket #1 (card slot 1) as an HDD.

[(drive type)]

[No drive] :
No drive, or the drive is disabled.

[Auto detection LBA] :
Automatically detects HDD parameters. Uses LBA mode for an HDD with a capacity of 528MB or more. (Recommended setting)

[Auto detection non LBA] :
Automatically detects HDD parameters. Does not use LBA mode, even for an HDD with a capacity of 528MB or more. (This is a special setting that is used when using CHS mode.)

[User defined LBA] :
Uses the user-defined drive parameters. Uses LBA mode for an HDD with a capacity of 528MB or more.

[User defined non LBA] :
Uses the user-defined drive parameters. Uses CHS mode for an HDD with a capacity of 528MB or more.

[Non-HDD IDE device] :
Use this setting when a device that is not a hard disk drive, such as a CD-ROM drive, is connected. (BIS enables the interface for such drives, but does not initialize the interface. A separate driver is required.)

[cyl.] [head][sec.] (User-defined HDD parameters)

When the drive type is specified as either [User defined LBA] or [User defined non LBA], the cursor can be moved to the fields that are used for setting the user-defined HDD parameters. The [cyl.] field is used to set the number of cylinders, the [head] field is used to set the number of heads, and the [sec.] field is used to set the number of sectors. Numeric values can be input directly into these fields.

Secondary boot = [(bootup from secondary IDE drive)]

This item is used in order to boot from an HDD that is connected to the secondary IDE (170h) interface. This item is not displayed if the secondary IDE (170h) interface is not set to [Enable] or [PCMCIA socket#0/1].
Using the secondary boot requires special caution, and is intended only for special applications.

[(bootup from secondary IDE drive)]

[Disable]: (recommended setting)

Does not boot up from a secondary IDE drive. In other words, the system tries to boot up from hard drives in the following sequence according to the normal HDD drive assignment rule:
Primary Master→Primary Slave→Secondary Master→Secondary Slave
(Drives that do not exist are skipped.

[Enable] :

Boots up from a secondary IDE drive. This is accomplished by changing the HDD drive assignment sequence rule as follows:
Secondary Master→Secondary Slave→Primary Maste→Primary Slave
This setting is invalid if no secondary master drive exists.

Note:
Do not use the secondary boot (booting up from a secondary IDE drive) feature except in special applications. The reason for this is that some operating systems give the primary IDE drive priority in their drive assignment rules. For example, enabling secondary boot under MS-DOS does not create any problems. In Windows, however, because the drive assigned for DOS does not match the assignment of resources (in which the primary IDE is given priority) assumed by Windows, 32-bit disk access becomes impossible and performance suffers.

Effect of the automatic correction feature:

If no drive is detected and the drive type is set to anything other than [No drive], the setting is automatically corrected to [No drive]. When a new drive is detected, the setting is automatically corrected to [Auto detection LBA].

EPSON DBIOS supports "ATA boot" only when the PCMCIA controller is expanded on the ISA bus.

2.3.2.3 [Video] Internal VGA Controller

Display select = [(output destination display device)]

This item sets the output destination display device for the internal VGA controller upon startup (after a system reset or after turning the power on).

[(output destination display device)]

[CRT] :
Directs display data to a CRT only. However, if the BIOS did not detect a CRT connected to the system while it was running its diagnostics, the system automatically starts in [CRT & LCD] mode.

[LCD] :
Directs display to an LCD only. Note that in the SETUP program, "LCD" is used as a generic label for all panel display devices.

[CRT & LCD] :
Directs display to a CRT and to an LCD simultaneously. Note that high-resolution video mode cannot be used with simultaneous display.

LCD font expansion = [Enable/Disable] (font expansion)

When [LCD] or [CRT & LCD] is specified, in some video modes part of the LCD in the vertical direction becomes a non-display area, and characters appear smaller. The font expansion function stretches the characters in the vertical direction so that they are easier to read.

[Enable/Disable]

[Enable] : Font expansion enabled. (Recommended setting)

[Disable] : Font expansion disabled.

LCD vertical alignment = [Top/Center] (positioning in vertical direction)

When font expansion is disabled, this setting positions, in the vertical direction, the screen that is displayed on the LCD. Either [Top] or [Center] can be selected.

LCD text mode reverse = [Normal/Reverse]

LCD graphics mode reverse = [Normal/Reverse]

These items set the halftone display for a monochrome LCD as either [Normal] or [Reverse]. This setting can be made independently for text modes (modes 0, 1, 2, 3, or 7) and graphics mode.

Effect of the automatic correction feature:

The video settings are stored in a dedicated area in CMOS-RAM for video settings. A checksum is used to assess the reliability of the settings. If a checksum error is detected, all settings are automatically corrected to their default settings.

The "internal VGA controller" is a Seiko-Epson VGA/LCD controller.

2.3.2.4 [Serial port] Internal Serial Port

COMA/COMB serial port = [(serial port number)]

EPSON DBIOS supports four serial port channels, COM1 through COM4. The two internal serial port channels called "COMA" and "COMB" can be used by assigning a port number (I/O address).

COMB can also be used as an infrared communications port.

[(serial port number)]

[Disable] :
Does not use the internal serial port.

[COM1 (F8h)], [COM2 (2F8h)], [COM3 (3E8h)], [COM4 (E8h)] :
Uses the corresponding internal serial port as one of COM1 through COM4. A list of extended external serial ports added on the bus is displayed at the top of the menu screen; select a port number that does not conflict with an external serial port. It is not possible to assign COMA and COMB to the same port number.

COMB port mode = [(infrared communications option)]

COMB IR polarity = [(infrared communications signal polarity)]

COMB IR method = [(infrared communications method)]

This group of settings is used when using COMB as an infrared communications port. For details on infrared communications, refer to the manual for the hardware.

[(infrared communications option)] (COMB mode)

[Serial port]: Uses COMB as a serial port. (default)
[IR port]: Uses COMB as an infrared communications port.

[(infrared communications signal polarity)]

[Active high]: IRTXD uses positive logic when sending. (default)
[Active low]: IRTXD uses negative logic when sending.

[(infrared communications method)]

[IrDA]: IrDA-SIR-1.0 standard (default)
[ASK (Sharp)]: Digital ASK (compatible with Sharp PDAs)

Effect of the automatic correction feature:

If the port number assigned to COMA or COMB conflicts with an external serial port, COMA/COMB is re-assigned to an unused port number.

"Internal serial port" refers to the serial controller that is included in Seiko-Epson's PC/AT chip set.

2.3.2.5 [Parallel port] Internal Parallel Port

Internal parallel port = [(parallel port number)]
Internal parallel mode = [(extended mode option)]

EPSON DBIOS supports two parallel port channels, Primary (378h) and Secondary (278h). The internal parallel port channel can be used by assigning a port number (I/O address). The internal parallel port has two modes, bi-directional parallel and Enhanced Parallel Port (EPP). It is necessary to specify which mode is to be used.

[(parallel port number)]

[Disable] :
Does not use the internal parallel port.

[Primary(378h)], [Secondary(278h)] :
Uses the internal parallel port as one of the port numbers. A list of extended external parallel ports added on the bus is displayed at the top of the menu screen; select a port number that does not conflict with an external parallel port.

[(extended mode option)]

[Normal] :	Normal parallel port (default)
[Bi-directional] :	Bi-directional parallel port
[EPP] :	Enhanced parallel port

Effect of the automatic correction feature:
If the internal parallel port conflicts with an external parallel port, the internal parallel port is re-assigned to an unused port number.

"Internal parallel port" refers to the parallel port controller that is included in Seiko-Epson's PC/AT chip set.

2.3.2.6 [Real time clock] Internal RTC

New time = [(HH:MM:SS)]
New date = [(MM/DD/YY)]

This menu is used to set the date and time. Once the date and time have been set, the settings are written to the internal RTC as soon as the menu is exited by selecting [Set and exit].
"Internal RTC" refers to the RTC that is incorporated into Seiko-Epson's PC/AT chip set. A backup for the internal RTC's power supply is needed.

[(HH:MM:SS)]

[(MM/DD/YY)]
The left and right arrow keys can be used to increase/decrease the values; the values can also be input directly. Use the [Backspace] key to clear a value that has been input.

2.3.2.7 [Keyboard]

NumLock = [On/Off]

This item sets the NumLock function for the keyboard at startup.

[On/Off]
[On]: The NumLock function is enabled. (The NumLock LED is on.)
[Off]: The NumLock function is disabled. (The NumLock LED is off.)

Typematic rate = [(key repeats/second)]

Typematic delay = [(delay until start of key repeat)]

These items set the keyboard repeat speed. These items set the delay (unit: milliseconds) until the "key repeat" function starts and the number of "key repeats" per second.

[(key repeats/second)]
[2.0] - [30.0] : (larger values are faster)

[(delay until start of key repeat)]
[250], [500], [750], [1000] : (in milliseconds; smaller values are faster)

2.3.2.8 [Memory]

External extended memory = [(size of external memory extended on the bus)]

When using "external memory" that has been extended on the ISA bus as (part of) main memory, it is necessary to set the size of the external extended memory.

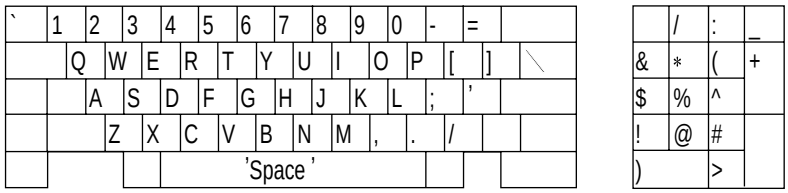
[(size of external memory extended on the bus)]
[0KB]: No external extended memory. (default)
[1KB] to [????KB]: Using ????KB of external memory extended on the bus

2.3.2.9 [Password]

Enter password here **[(input from keyboard)]**
Enter password here again **[(input from keyboard)]**

This item is used for inputting a new password. Input the same password in both fields.

The keys shown below for valid for inputting a password.



If a password is already enabled, this field cannot be used to change the password. (Disable the password first, and then input the new password.)

[(input from keyboard)]
[XXXXXXX]: Up to seven characters can be input. For secrecy, each character that is input is represented by an "X" on the display.

Password function = [(enable the password)]

This item is used to enable the password. When the password is enabled, the following prompt is displayed in the upper left corner of the screen at startup:

3 ○┬ (Input the password after this prompt.)

[(enable the password)]
[Enable]: Enables the password function.
[Disable]: The password is not currently set.

If the password is already enabled, it cannot be disabled through this item. (In other words, the user cannot select "[Disable]".)

To disable the password, input the password at system startup, and then add a single slash ("/") after the password.

Server mode = [Enable/Disable]

Server mode is one of the password functions. When this function is enabled, the system starts up normally without requesting a password, but input from the keyboard is locked out until the password is input. This function is disabled when the password function is disabled.

[Enable/Disable]
[Enable] : Enables server mode.
[Disable] : Disables server mode.

2.3.2.10 [User accessible CMOS area] User CMOS area

Index xx data = [(any value)](xx=70h,71h,...,7Fh)

The CMOS-RAM indexes from 70h to 75h are open to the user, and can be used to store any value.

[(any value)]
[00] - [FF] : Set any byte data (hexadecimal).

(Example of application)
The MS-DOS ROM version reverences the value in CMOS-RAM index 7Eh as the bootup information. The MS-DOS ROM version bootup method can be controlled by using SETUP to change this value.

Effect of the automatic correction feature:
SETUP does not change these values.

2.3.2.11 [Power management control] Power management settings (Minimizer)

Note:
In order to use the power management functions in EPSON DBIOS, your system must support power management. For details on the handling of power management control signals and detection signals, etc., refer to the manual for the hardware (and the applications notes).

Power management = [Disable/Enable]

This item enables the power management BIOS (Minimizer) that is embedded in EPSON DBIOS. This item is a switch that controls all of the power management functions.

[Disable/Enable]
- [Disable]: Does not use the power management functions.
- [Enable]: Enables the power management functions (including APM).

Time correction = [Disable/Enable] (time correction)

This function corrects errors in the time and date in the software after returning from the standby or suspended state.

[Disable/Enable]
- [Disable]: Does not correct the time.
- [Enable]: Corrects the time.

There are instances (such as when running under a processor protect mode, as in Windows) where time correction is not possible, even if this item is enabled. In such a case, either use the operating system's APM support function, or else install a time correction device driver for EPSON BIOS.

Low battery suspend = [Disable/Enable] (Protective function in the event of a low battery)

This item is used to put the system into the suspended state automatically if a drop in the battery voltage is detected.

[Disable/Enable]
- [Disable] : Does not automatically enter suspended state, even if low battery voltage is detected.
- [Enable]: Automatically enters suspended state if low battery voltage is detected.

Warning beep = [Disable / Enable]

This item sets a function that notifies the user with an audible beep (through the system's speaker output signal) that the battery voltage had dropped, or that the suspend button was pressed.

[Disable/Enable]
- [Disable]: Does not notify the user.
- [Enable]: Sounds a beep to notify the user.

Modem ring resume = [Disable/Enable]

This item sets an optional function (resume in response to RING signal) that permits the resume operation to be initiated in response to an external line while the system is in the suspended state.

[Disable/Enable]

[Disable] : Disables resume in response to the RING signal.

[Enable] : Executes the resume operation if a RING signal is received on COMA or COMB.

Alarm resume = [Disable/Enable]

Alarm time = [(HH:MM:SS)]

This item sets an optional function that permits the resume operation to be initiate while the system is in the suspended state at a time that was set beforehand using the alarm function of the internal RTC.

[Disable/Enable]

[Disable] : Disables resume in response to the alarm function.

[Enable] : Executes the resume operation at the set time.

[(HH:MM:SS)]

This sets the time for the resume operation. In addition to using the left and right arrow keys to set the time, it is also possible to input values directly.

Effect of the automatic correction feature:

The power management settings are saved in an area allocated to Minimizer in CMOS-RAM, and the reliability of the contents of that area is verified by checksum. If the checksum does not match, all settings are automatically reset to their default settings.

2.3.2.12 [Power save timer] Power saver timers (Minimizer)

This menu sets the timers that are used to monitor the operating status of the system and devices and automatically put the system into a power saver mode, such as the standby state or the suspended state.

System power saver
System power saver refers to the CPU standby state, the global standby state, and the suspended state (auto suspend). If no events (hardware operations), such as an interrupt signal, occur for a certain period of time, the entire system enters power saver mode. The specific hardware operations that serve as the criteria for determining when the system enters power saver mode can be set individually in the "Event configuration" menu. It is possible for the system to move among multiple power saver modes; for example, the system can move from CPU standby to global standby, which has a greater power saving effect.

Device power saver (local standby)
Device power saver monitors accesses (primarily I/O reads and writes) to individual devices and puts devices that the software is not using into the standby state. This function does not affect the operating status of the system as a whole. (Individual devices are put into the standby state.) The only devices the can be put into the standby state individually are the video device (the internal VGA controller), COMA, COMB, and HDDs.

CPU standby timer = [(00:MM:SS)] (up to 17 minutes)
Gloval standby timer = [(HH:MM:SS)] (up to 36 hours)
Auto suspend timer = [(HH:MM:SS)] (up to 36 hours)

These items are used to set the system power saver timers. In "CPU standby," the CPU is put into the standby state (clock halted). In "Global standby," all devices (except for the keyboard and the PS/2 mouse) are put into the standby state along with the CPU. These standby states are released by hardware operations called "stop break events." (The stop break events are set in the "Event configuration" menu.) Note that the resume operation is required in order to return the system from the suspended state when it was put in that state by the auto suspend feature.

Video/COMA/COMB/HDD local standby timer = [(00:MM:SS)] (up to 17 minutes)

This item is used to set the device power saver timer. If a device that was put into the standby state by this timer is accessed, the Minimizer traps the access and releases the device from the standby state. As an exception concerning the video device, input from the keyboard or the PS/2 mouse is considered to be an access to the video device.

[(00:MM:SS)] (up to 17 minutes)
[(HH:MM:SS)] (up to 36 hours)

[Disable]:	Disables the power saver timer.
[00:00:04] (or higher):	Enables the power saver timer (up to the upper limit, in four-second increments).

Effect of the automatic correction feature:
The effect is the same as for power management control (the power management settings).

2.3.2.13 [Event configuration] Event settings (Minimizer)

This menu sets the events (hardware operations) that serve as the criteria for determining when the system enters power saver mode. Two types of events must be set.

System event (idle detection events)
These events indicate that the system is operating. In other words, if these events are not generated, the system is deemed to be idle (and can therefore be put into a power saver mode).

Stop break event (recovery request events)
These events cause the system to return to the normal operating state from a standby state. Specifically, these operations release CPU standby and global standby.

This menu is used to set individual hardware operations as events.

(hardware operation) = [(stop break event setting: system event setting)]

For each individual hardware operation, one of the four possible combinations of [Enable] and [Disable] for the "stop break event setting" and the "system event setting" can be selected.

[(stop break event setting: system event setting)]
[Disable : Disable]
[Disable : Enable]
[Enable : Disable]
[Enable : Enable]
(Disable: Does not set the hardware operation as that type of event.)
(Enable: Sets the hardware operation as that type of event.)

(Hardware operations)

Modem ring:	COMA/COMB received the RING signal.
NMI:	Nonmaskable interrupt
Active I/O channel check:	IOCHCK# goes low. (One NMI source)
IRQ 1,3,4,5,6,7,9,10,11,12,14,15:	Hardware interrupts

Effect of the automatic correction feature:
The effect is the same as for power management control (the power management settings).

2.3.2.14 [BIOS information] Supplementary information

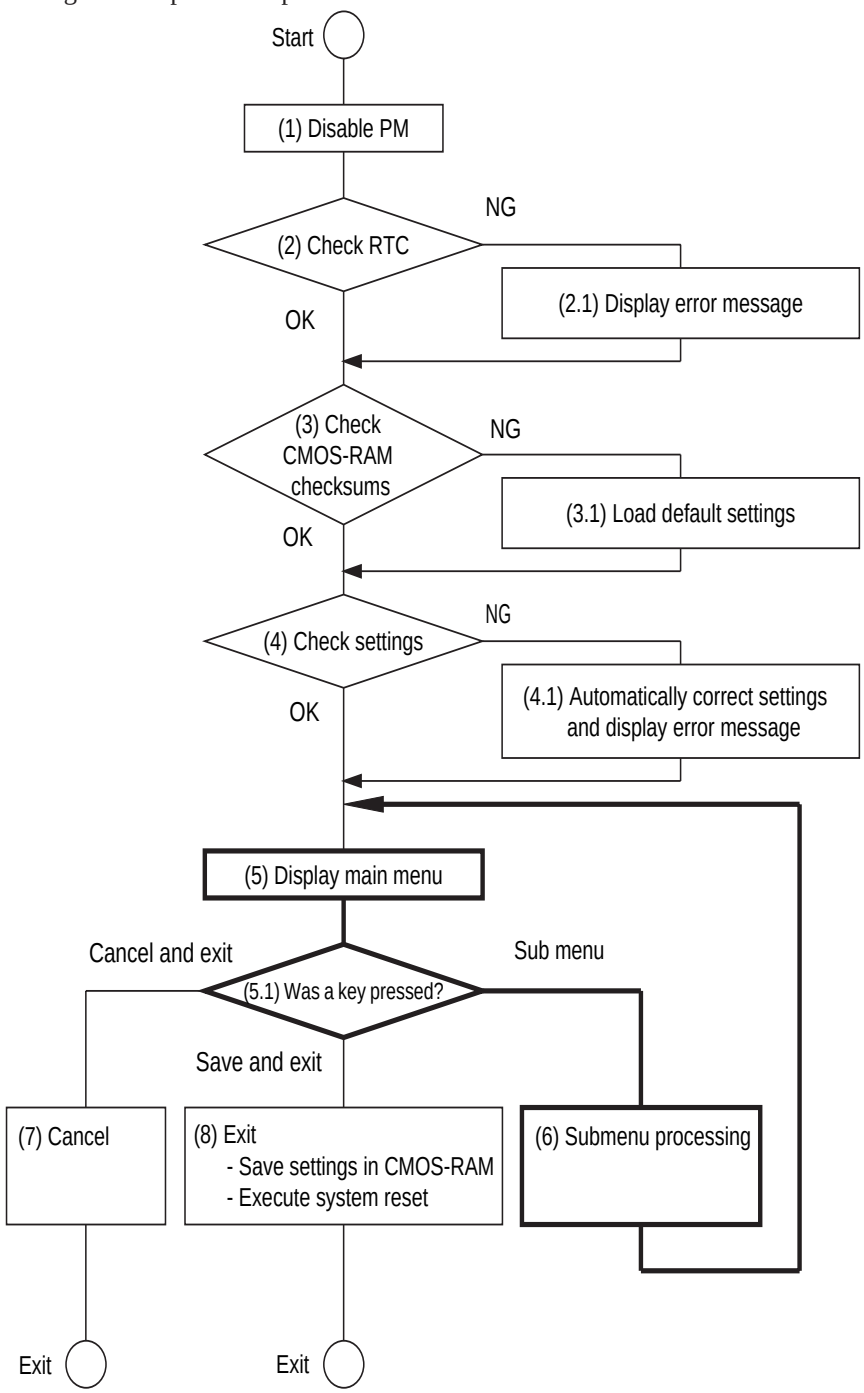
This item displays the BIOS version information, This is not a settable item.

3 OTHER FEATURES OF AND CAUTIONS CONCERNING SETUP

3.1 Details of Operation of SETUP

3.1.1 Flow Chart

The following chart depicts the operation of SETUP in detail.



(1) Disable PM
- SETUP disables power management (PM), using the APM function.

(2) Check RTC
- If there is a problem with the backup power supply for the RTC (CMOS-RAM), SETUP displays an error message (item 2.1 in the flow chart).

(3) Check CMOS-RAM checksums
- SETUP checks the CMOS-RAM checksums; if a checksum does not match, SETUP displays an error message (item 3.1 in the flow chart) and loads the default settings from BIOS-ROM.

(4) Check settings
- SETUP checks for problems in each of the items that are set in CMOS-RAM.
- If a problem is found in one of the settings, SETUP displays an error message (item 4.1 in the flow chart) and automatically corrects the setting.

(5) Display main menu
- SETUP displays the main menu, and waits for the user to issue an instruction.

(6) Sub Menu processing
- SETUP displays a sub menu, and waits for the user to issue an instruction.
- If an error was found in any items as a result of checking the settings I item 3 above, SETUP displays the automatically corrected values. Note that the values stored in CMOS-RAM are still unchanged at this point.
- The values stored in CMOS-RAM remain unchanged even after exiting the sub menu.

(7) Cancel
- This processing is performed when [Cancel and exit] is selected on the main menu, or when the [Esc] key is pressed.
- This processing cancels SETUP.
- When this is the method that is used to exit SETUP, the contents of the new settings that were made in the sub menus are not stored in CMOS-RAM.
- Power management remains disabled after exiting SETUP in this manner.

(8) Exit
- This processing is performed when [Set and exit] is selected on the main menu.
- This processing exits SETUP.
- This method of exiting SETUP is the only processing that actually stores the contents of the new settings that were made in the sub menus in CMOS-RAM, and then resets the system.

3.1.2 Default CMOS-RAM Settings

Before automatically correcting the settings that are stored in CMOS-RAM, SETUP checks the CMOS-RAM checksums. If a checksum does not match, SETUP discards the settings stored in that area in CMOS-RAM and loads the default settings from BIOS-ROM.

The contents of the settings in CMOS-RAM are divided into three areas; each area has its own checksum.

	Area (Index)	Checksum Index
(1) Standard device settings area	00h - 3Fh	2Eh, 2Fh
(2) Video (internal VGA) settings area	40h - 4Fh	48h, 49h
(3) Minimizer settings area	50h - 6Fh	6Ah, 6Bh

SETUP checks the checksum for each individual area. For each area that generates a checksum error, SETUP displays an error message and loads the default settings.

* Processing for the video settings area is performed only if a Seiko-Epson VGA/LCD controller is installed in the system.

3.1.3 Automatic Correction of CMOS-RAM Settings

If an item is found to be in error after checking the settings in CMOS-RAM, the corrected value is displayed in the sub menu. However, at this point the value that is in error remains as is in CMOS-RAM.

CMOS-RAM is updated only by exiting SETUP via [Set and exit] in the main menu. If [Cancel and exit] is used to exit SETUP, the value that is in error remains in CMOS-RAM.

Even if an item is intentionally disabled, SETUP deems that setting to be an error and automatically corrects it. In this type of situation, the user must set that device as desired. For example, if an HDD that is connected to the system is not to be used, the HDD setting must be reset each time that SETUP is started up.

3.1.4 Error Messages

The error messages that are displayed by SETUP are explained below.

"Real-time clock lost power"
The backup power supply for the RTC (CMOS-RAM) is not producing any voltage.

"Standard CMOS area checksum error. Copying default."
A checksum error was detected in the standard device settings area in CMOS-RAM. The default settings are being loaded from BIOS-ROM.

"VGA extended CMOS area checksum error. Copying default."
A checksum error was detected in the video settings area in CMOS-RAM. The default settings are being loaded from BIOS-ROM.

"Minimizer extended CMOS area checksum error. Copying default."
A checksum error was detected in the Minimizer settings area in CMOS-RAM. The default settings are being loaded from BIOS-ROM

"FDD setup error"
The FDD settings do not correspond with the detected hardware.

"HDD setup error"
The HDD settings do not correspond with the detected hardware.

"Parallel port setup error"
There is an error in the parallel port settings. (port conflict)

"Password setup error"
There is an error in the password settings.

"Real time clock setup error"
There is an error in the real-time clock data.

"Real time clock serious error!"
A major error (that prevents operation) was discovered in the real-time clock.

"Serial port setup error"
There is an error in the serial port settings. (port conflict)

"Video setup error"
There is an error in the video settings. (An illegal value has been set.)

"Minimizer setup error"
There is an error in the Minimizer settings. (An illegal value has been set.)

3.2 Cautions

3.2.1 If SETUP Is Cancelled

When SETUP is started up, it disables power management by the APM function. If SETUP is cancelled, the operating system is booted up without resetting the system. Note that in this case, the power management functions, such as the power saver timers, remain in the disabled state.

3.2.2 If an Automatic Reset Is Executed

If the settings are valid an SETUP is exited by using [Set and exit], CMOS-RAM is updated and then the system is reset automatically.

The Int15h F0h function is used for the automatic reset. Although in terms of BIOS operation the effect is the same as if the reset button was pressed, the PWRGOOD signal is not actually toggled and the RESETDRV signal is not actually asserted on the ISA bus, so it is possible that the system will operate abnormally after the automatic reset, depending on the extended device and circuit configurations. If the system does operate abnormally, simply turn the power off, and then turn the power back on.

3.2.3 Addressing the Year 2000 Problem

As a measure for addressing the year 2000 problem, SETUP automatically updates the "century byte" in CMOS-RAM to 20h under the following conditions:

(1) When the date that is set in CMOS-RAM when SETUP is booted up is earlier than 1980
(2) When the year 2000 is reached in processing by the "real-time clock" sub menu in SETUP
(3) When the year 2000 is deemed to already have been reached when SETUP is cancelled or exited

3.3 Regarding the SETUP Libraries

SETUP libraries can be provided consisting of subroutines that detects and automatically corrects errors in the CMOS-RAM settings for individual devices.

However, because only certain development environments (development languages for which libraries can be provided) are supported, and special consultation is necessary concerning the provision of technical support and other materials, please contact Seiko-Epson directly for further details regarding such libraries.

4 GLOSSARY

BIOS, EPSON DBIOS

- "BIOS" (Basic Input/Output System) is the basic I/O program of a PC, and is normally installed in the PC in ROM.
- "EPSON DBIOS" is Seiko-Epson's proprietary BIOS for PC/AT-compatible computers.

CMOS-RAM

- This is memory that is included in the RTC. The RTC's backup power supply is used to retain the contents of CMOS-RAM even when the system power is off.
- If the RTC backup power supply switching circuit is not suitable, the contents of CMOS-RAM may become scrambled, making operation impossible.

Default CMOS-RAM settings

- The default CMOS-RAM settings are stored in BIOS-ROM.
- EPSON DBIOS permits customization of the default CMOS-RAM settings. In addition, the conditions under which the default values can be written can be controlled precisely so that the default values are written: (a) unconditionally, (b) when the backup voltage drops ("lost power"), (c) when a checksum error is generated, etc. However, customization requires the ROM Adaptation Kit (RAK), available separately from Seiko-Epson.

IDE, EIDE, Drive ID (HDD auto detect) , LBA, non LBA, CHS

- IDE is an interface standard for hard disk drives. Generally, IDE uses the following resources:

IDE	I/O port	IRQ
Primarychannel	1F7h - 1FFh§3F6h,3F7h	IRQ14
Secondarychannel	177h - 17Fh§376h,377h	IRQ15

- Seiko-Epson's CARD-PC has a built-in IDE primary channel.
- EIDE ("Enhanced IDE") uses the same interface as IDE, and is extended to support drive IDs (automatically fetched as a parameter), devices other than HDDs, larger capacities, fast data transfer mode, etc. EPSON DBIOS and Seiko-Epson CARD-PCs do not support fast data transfer mode.
- LBA (Logical Block Addressing) is one method that is used to access large-capacity (over 528MB) HDDs. Because of differences in specifications between the BIOS Int13h (disk access function) interface and the conventional IDE standard, earlier BIOS firmware could not access hard disk drives that were larger than 528MB. LBA increases the capacity that can be accessed by increasing the number of heads that an HDD appeared to have. In SETUP, the earlier method for HDD access that was subject to these capacity restrictions is called "non-LBA" (as opposed to "LBA") and "CHS". This earlier method (non-LBA) and LBA are explained below.

(1) Accessing an HDD under the earlier method (non-LBA, or CHS)
The following table lists the upper limits for the HDD parameters (cylinders, heads, sectors) under the BIOS Int13h function and the IDE standards.

	BIOS	IDE	When used within both limitations
Cylinders	1024(10bit)	65536(16bit)	1024(8bit)
Heads	255(8bit)	16(4bit)	16(4bit)
Sectors	63(6bit)	255 (8bit)	63 (6bit)
Totalcapacity	8.4GB	136.9GB	528MB

Therefore, when both specifications are observed, the maximum HDD capacity that BIOS can access is 528MB. Although BIOS provides 8 bits for the head specification, the maximum number of HDD heads permitted under the IDE specification is 16, which only requires 4 bits. The other 4 bits available under BIOS are wasted

(2) Accessing an HDD under LBA mode
If an HDD is put into LBA mode, it now becomes possible to access a specific sector in the HDD by directly specifying the sequential number (the logical sector number) that is assigned to each sector. (Under the earlier method, the cylinder, head and sector all needed to be specified.) Most operating systems access HDDs (using the Int13h function of BIOS) on the basis of the total number of cylinders, heads, and sectors (collectively called the "HDD parameters") reported to the operating system by the BIOS.
However, a BIOS that supports LBA reports virtual HDD parameters to the operating system, with the valid bits for the number of heads extended from 4 bits to 8 bits. When the OS calls the Int13h function, the cylinder, head and sector numbers are converted into a logical sector number, which is then used to access the HDD. As a result, regardless of the operating system it becomes possible to access HDDs that are larger than 528MB. The maximum HDD capacity that BIOS can handle in LBA mode is 8.4GB.

The upper limits for the virtual parameters that BIOS handles in LBA mode are listed in the following table.

HDD capacity (MB)	Sectors	Heads	Cylinders
528 - 1056.9	63	32	Capacity/(63 x 32 x 512)
1056.9 - 2113.9	63	64	Capacity/(63 x 64 x 512)
2113.9 - 4227.8	63	128	Capacity/(63 x 128 x 512)
4227.8 - 8422.6	63	255	Capacity/(63 x 255 x 512)

The formula for converting the cylinder number (cy), head number (hd), and sector number (sc) that are passed to the Int13h function into the logical sector number is shown below.

Logical sector number = (cy x H) + hd) x S + sc -1
H: Number of virtual heads in HDD
S: Number of virtual sectors per cylinder of the HDD

- The following operating systems require the use of non-LBA mode (CHS mode):
  (1) Novel NetWare 3.11,4.0
  (2) SCO UNIX

  The reason is that these operating systems do not use the Int13h function from BIOS; they access the HDD by proprietary methods. (These operating systems are able to access HDDs that are larger than 528MB.)

PCMCIA, PC card, PCIC, Socket #0/#1, PC card slot

- PCMCIA (an abbreviation for "Personal Computer Memory Card Interface Association," a standards organization based in the U.S.) is used as a name for the standards for PC cards.
- The term "PC card" that appears in the section describing the ATA boot operation in this manual refers to an ATA PC card that conforms with PCMCIA 2.1.
- "PCIC" is an abbreviation for "PCMCIA controller." Seiko-Epson's CARD-PC does not have a built-in PCIC.
- Sockets #0 and #1 are the names used by the system (including software) for the interfaces for connecting PC cards to the system. "PC card slot" is the term used for the PC card plug-in connector as viewed from outside of the system.

ATA boot

- This function permits BIOS to directly assign an ATA PC card installed in a PC card slot (in PCIC socket #0 or #1) as an HDD and boot up from that card, without using PCMCIA driver software, such as socket service or card service.
- Note that if ATA boot is used, the PC card cannot be inserted or removed while the system is operating.

Secondary boot

- This is an extended function that is unique to EPSON DBIOS that permits bootup from a hard disk drive that is connected to the secondary IDE (170h). Secondary boot is implemented by changing the drive assignments. In other words, the Master HDD on the secondary IDE becomes drive C.
- The secondary boot function is intended for special applications. Because the secondary boot operation could cause problems with some operating systems, the use of this function and the specifications of the operating system should be given careful consideration before using this function. It is recommended that this function be disabled in normal circumstances.

LCD, flat panel, panel display

- As used in this manual, these terms all refer to the same type of display device.

Font expansion, vertical alignment, LCD reverse

- These are extended LCD (panel display) functions that are unique to EPSON DBIOS.

COM1, 2, 3, 4

- In PC/AT-compatible computers, these are the commonly used names for the serial ports ("COM ports," 1 through 4). Normally, the following resources are assigned to these ports:

	I/O port	IRQ
COM1	3F8h - 3FFh	IRQ4
COM2	2F8h - 2FFh	IRQ3
COM3	3E8h - 3Efh	IRQ11
COM4	2E8h - 2Efh	IRQ10

COMA, COMB, IR-port

- "COMA" and "COMB" are the names for the two serial port channels that are built into Seiko-Epson's PC/AT-compatible chip set that is supported by EPSON DBIOS.
- COMB can also be used as an infrared communications port. There are two major infrared communications systems:
  - Ir-DA (Ir-DA-SR1.0 Version 1.0)
  - ASK (Sharp) (Digital ASK system)

Internal parallel, Normal parallel, Bi-directional, EPP

- "Internal parallel" refer to the single parallel port channel that is built into Seiko-Epson's PC/AT-compatible chip set that is supported by EPSON DBIOS.
- The internal parallel port can be used as either a primary port (378h - 37Fh, IRQ7), or as a secondary port (278h - 27Fh, IRQ5).
- The internal parallel port can also be used as a bi-directional parallel port or as an EPP (Enhanced Parallel port). The term "normal parallel port" is used to distinguish the parallel port when it is used in normal mode from when it is used in one of these extended modes.

Password, Server mode, Network server mode

- The password function is a function that is common to all PC/AT-compatible computers. However, the method that is used to input the password in EPSON DBIOS is different from the methods that are used in BIOS from other companies.
- Server mode (or "Network server mode") is a function that locks out the keyboard until the password is input

User accessible CMOS area

- EPSON DBIOS has a 16-byte area (index: 70h through 7Fh) in CMOS-RAM that is open to the user.
- For example, in a system equipped with the MS-DOS ROM version, the startup device for the MS-DOS ROM version can be selected by setting the appropriate value in CMOS index: 7Eh.

Minimizer

- Minimizer is the power management BIOS that is provided with EPSON DBIOS.
- Minimizer provides reliable and powerful power management by using the SL Enhanced CPU's SMI (System Management Interrupt).
- In addition to automated power management, Minimizer supports the APM functions.

APM, APM1.0, APM1.1

- APM (or APMI) is a software interface for power management that was established by Intel and Microsoft. "APM" is an abbreviation for "Advanced Power Management interface."
- The first version of APM that was established is APM1.0; subsequent revisions (APM1.1, APM1.2) maintain compatibility with the earlier version.
- Minimizer supports APM1.1.

Standby, suspend, resume

- The meanings of these terms as they are used in the power management function are summarized below.

[Standby]

"Standby" is the state in which the system is standing by while the power is on. To the user, the system appears to be operating normally.

Standby requires a means to automatically restore the system to full operation (without the user being aware of this activity).

Standby does not offer as much of a power saving benefit as the suspended state.

[Suspend]

"Suspend" puts the system into a halted state. However, before putting the system into the halted state, the suspend function saves whatever data will be necessary in order to restore the system to its original state (full operation). Although some power must still be supplied to the system in order to maintain the data, the power consumed is very small. (Even compared to the standby state, only a small amount of power is required.) Because the system is halted while it is suspended, it cannot be restored to its original state without the user knowingly performing the necessary procedure (operation).

[Resume]

"Resume" is the operation that restores the system to its original operating state from the suspended state.

CPU standby, Global standby

- These functions put the entire system into standby. The appropriate timer must be set in order for these functions to work.
- The CPU standby function stops the CPU clock.
- The global standby function stops the CPU clock, and also puts all devices other than the keyboard and the PS/2 mouse into the standby state.
- CPU standby or global standby results if a system event does not occur for a specified time.
- If a stop break event occurs while the system is in the CPU standby or global standby state, the system returns to the normal operating state.

Auto suspend

- This is a suspend function. The appropriate timer must be set in order for this function to work.
- This function puts the system into the suspended state automatically if a system event does not occur for a specified time.

Local standby

- This function puts individual devices into the standby state. The appropriate timer must be set in order for this function to work.
- This function is intended for the video device, COMA, COMB, and hard disk drives.
- This function puts the device in question into the standby state if no I/O access is made to the device for a specified time.
- If an I/O access is made to a device that is in the standby state, the device is restored to the normal operating state automatically.
- As an exception, keyboard and PS/2 mouse input is regarded as an access to the video device.

System event

- These are set as hardware operation conditions for triggering the CPU standby, global standby, or auto suspend function. These are primarily set by enabling or disabling individual IRQ numbers.
- If an "enabled" system event occurs, the system is deemed to be operating. In other words, system events prevent the system from being put into the standby or suspended state.

Stop break event

- These are set as hardware operation conditions for releasing the system from the CPU standby or global standby state.
- If an "enabled" system event occurs while the system is in the standby state, the system is restored to its normal operating state.

Appendix

CMOS-RAM Map

Table of Contents

CMOS-RAM MAP

Standard CMOS-RAM Area

Address	Function
*00h-0Dh	Real-Time Clock Information
*0Eh	Self Diagnostic Status byte
*0Fh	Shut Down Status Byte
10h	Diskette Drive Byte
11h	Keyboard Speed Byte
12h	Primary HDD Type Byte
13h	Keyboard Option Byte
14h	Equipment Byte
15h-16h	Base Memory Size in K bytes
17h-18h	Extended Memory Size in K bytes
19h	Secondary HDD Type Byte
1Ah	Boot Select Byte
1Bh	Function Support Byte
1Ch	Parallel Configuration Byte
1Dh	Serial Configuration Byte
1Eh	(Reserved)
1Fh	(Reserved)
20h-22h	User-defined Primary HDD1 Parameter
23h-25h	User-defined Primary HDD2 Parameter
26h	User-defined Primary HDD Parameter
27h-29h	User-defined Secondary HDD1 Parameter
2Ah-2Ch	User-defined Secondary HDD2 Parameter
2Dh	User-defined Secondary HDD Parameter
*2Eh-2Fh	CMOS Checksum
*30h-31h	Extended memory size detected by POD
*32h	Century Byte
*33h	Information Flags
*34h-35h	(Reserved)
*36h-3Dh	Password Buffer
*3Eh-3Fh	Password Checksum

* These bytes are not included in the checksum.

Minimizer extended CMOS-RAM area

Address	Function
50h	Power Management Control Byte
51h	CPU Standby Timer
52h-53h	Global Standby Timer
54h-55h	Auto Suspend Timer
56h	Video Local Standby Timer
57h	COMA Local Standby Timer
58h	COMB Local Standby Timer
59h	HDD Local Standby Timer
5Ah-5Bh	(Reserved)
5Ch	Battery Low Control Byte
5Dh	Resume Control Byte
5Eh	Resume Alarm Hour
5Fh	Resume Alarm Minute
60h	Resume Alarm Second
61h	Stop Break Setting 0
62h	Stop Break Setting 1
63h	Stop Break Setting 2
64h	System Event Setting 0
65h	System Event Setting 1
66h	System Event Setting 2
67h-69h	(Reserved) (=0)
6Ah-6Bh	Minimizer CMOS-RAM Checksum
*6Ch-6Fh	(Reserved)

* These bytes are not included in the checksum.

User accessible CMOS-RAM area

Address	Function
70h-7Fh	Available for use

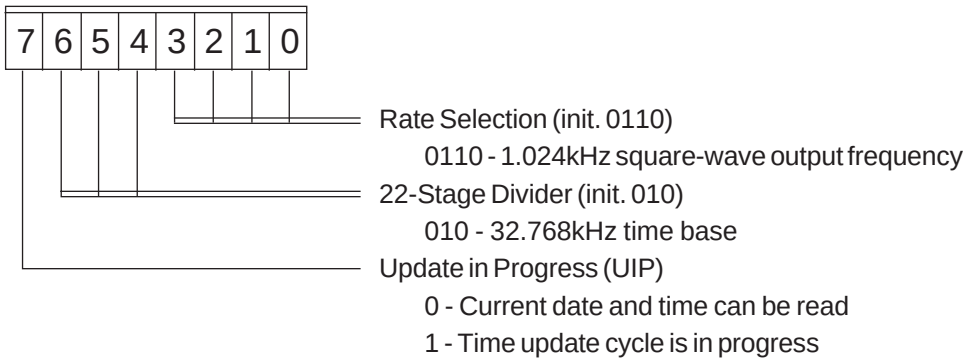
CMOS-RAM BIT ALLOCATIONS

Standard CMOS-RAM area

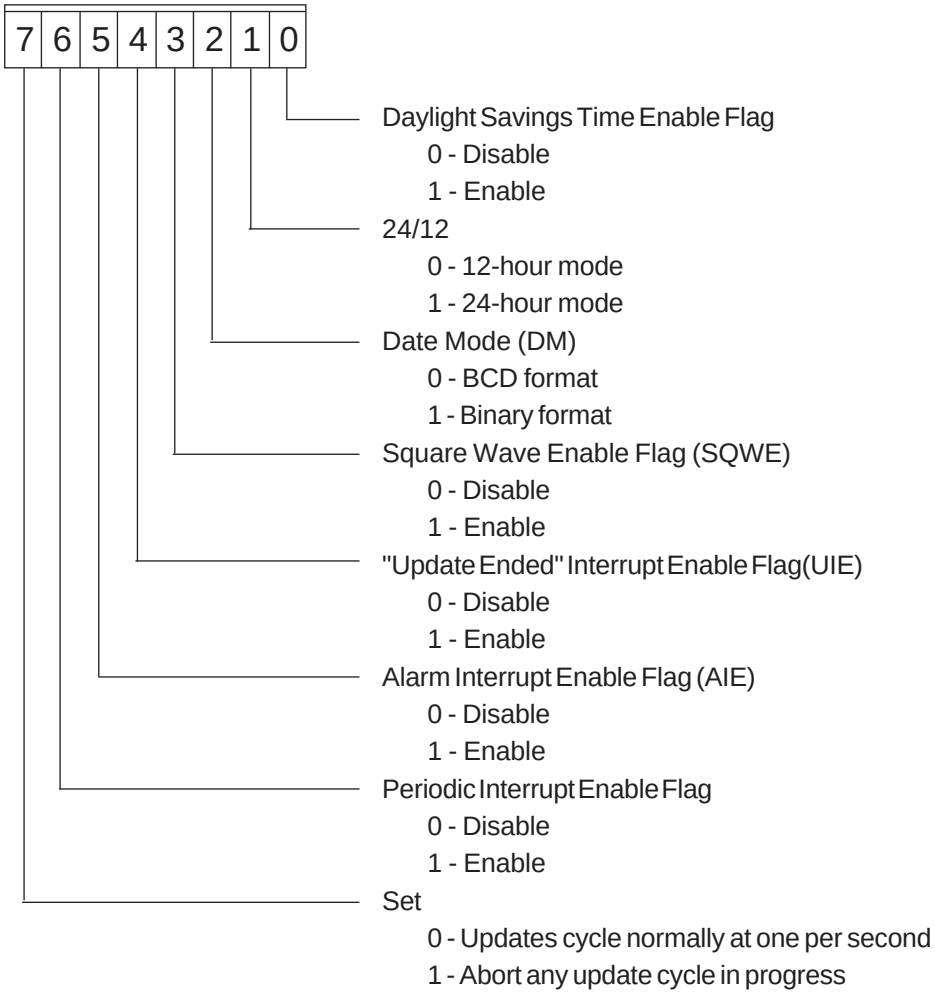
00h - 0Dh - Real Time Clock information

00h - Seconds (BCD)
01h - Second for Alarm (BCD)
02h - Minutes (BCD)
03h - Minute for Alarm (BCD)
04h - Hours (BCD)
05h - Hour for Alarm (BCD)
06h - Day of week
07h - Date (BCD)
08h - Month (BCD)
09h - Year (BCD)

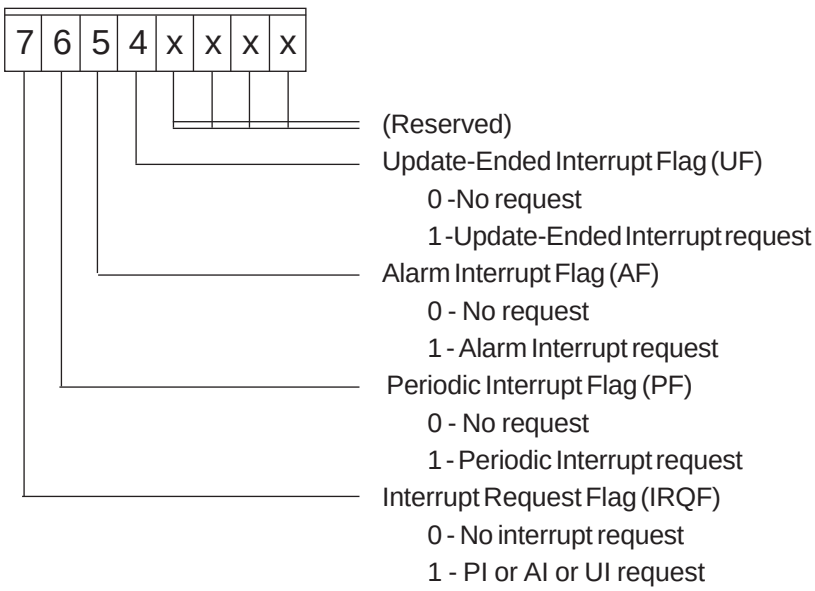
0Ah Status Register A



0Bh Status Register B

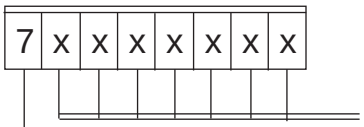


0Ch - Status Register C



CMOS-RAM BIT ALLOCATIONS

0Dh - Status Register D



(Reserved)

Valid RAM Bit (VRB)

- 0 - Real-time clock has lost power
- 1 - Power on the real-time clock

0Eh - Diagnostic Status Byte

(Reserved)

Date/TimeValidity

- 0 - Date and time is valid
- 1 - Date or time is not validd

Hard Disk Drive C Initialization Status

- 0 - Functioning property
- 1 - Initialization failed

Memory Size Miscompare

- 0 - Memory size matches
- 1 - Memory size is different

Incorrectdeviceconfiguration

- 0 - Configuration is correct
- 1 - Device configuration settings in CMOS-RAM do not match actual device configuration

Checksumstatus

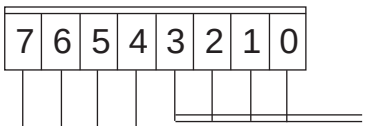
- 0 - Checksum is correct
- 1 - Checksum is incorrect

Real-time clock status

- 0 - Normal
- 1 - Power was lost

0Fh - Shut Down Status Byte

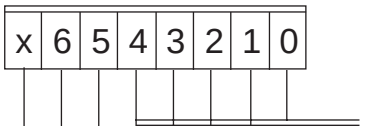
10h - Diskette Drive Byte



Type of second diskette drive
- 0000 - No drive
- 0001 - 360K byte 5.25 inch
- 0010 - 1.2M byte 5.25 inch
- 0011 - 720K byte 3.5 inch
- 0100 - 1.44M byte 3.5 inch

Type of first diskette drive
- 0000 - No drive
- 0001 - 360K byte 5.25 inch
- 0010 - 1.2M byte 5.25 inch
- 0011 - 720K byte 3.5 inch
- 0100 - 1.44M byte 3.5 inch

11h - Keyboard Speed Byte



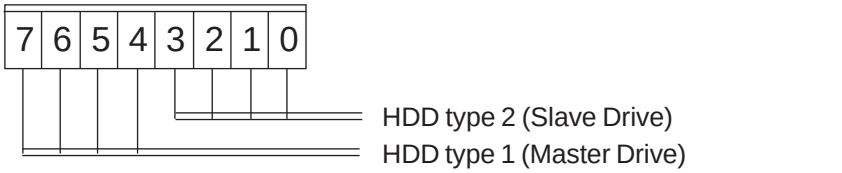
Typematic rate
- Interval = (8 + A) * (2 ** B) * 0.00417 seconds
  - A = binary value of bit 2-0
  - B = binary value of bit 4-3

Typematicdelay
- 00 - 250 ms
- 01 - 500 ms
- 10 - 750 ms
- 11 - 1000 ms

(Reserved)

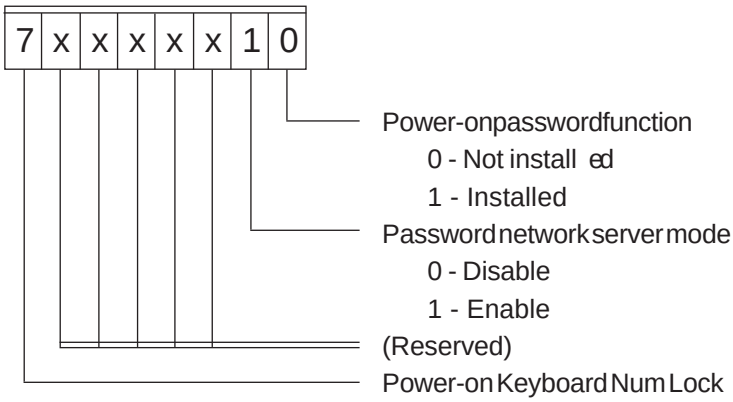
CMOS-RAM BIT ALLOCATIONS

12h - Primary HDD Type Byte



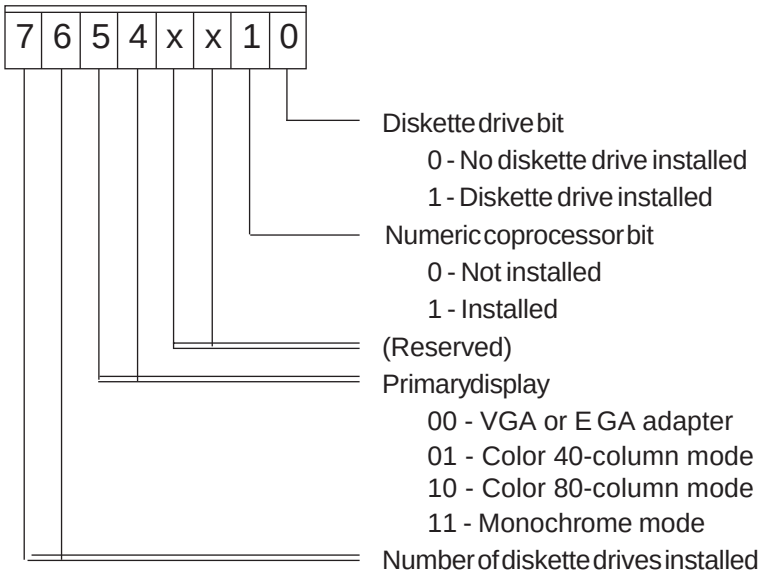
HDD type 2 (Slave Drive)
HDD type 1 (Master Drive)
0000 - No drive
0001 - Auto type detect (LBA)
0010 - Auto type detect (Non LBA)
1110 - User-defined (LBA)
1111 - User-defined (Non LBA)

13h - Keyboard Option Byte



Power-on password function
0 - Not installed
1 - Installed
Password network server mode
0 - Disable
1 - Enable
(Reserved)
Power-on Keyboard Num Lock
0 - Num Lock on
1 - Num Lock off

14h -Equipment Byte



Diskette drive bit
0 - No diskette drive installed
1 - Diskette drive installed
Numeric coprocessor bit
0 - Not installed
1 - Installed
(Reserved)
Primary display
00 - VGA or EGA adapter
01 - Color 40-column mode
10 - Color 80-column mode
11 - Monochrome mode
Number of diskette drives installed
00 - 1 drive
01 - 2 drives
10,, 11 - (Reserved)

15h-16h - Base Memory Size

These bytes are set by SETUP utility.

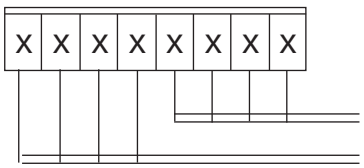
15h - Low byte of Base Memory size in increments of 1K byte.
16h - High byte of Base Memory size in increments of 1K byte.

17h-18h-Extended Memory Size

These bytes are set by SETUP utility.

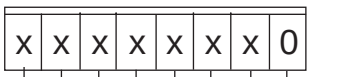
17h - Low byte of Extended Memory size in increments of 1K byte.
18h - High byte of Extended Memory size in increments of 1K byte.

19h - Secondary HDD Type Byte



HDD type 2 (Slave Drive)
HDD type 1 (Master Drive)
- 0000 - No drive
- 0001 - Auto type detect (LBA)
- 0010 - Auto type detect (Non LBA)
- 1110 - User-defined (LBA)
- 1111 - User-defined (Non LBA)

1Ah - Boot Selection Byte

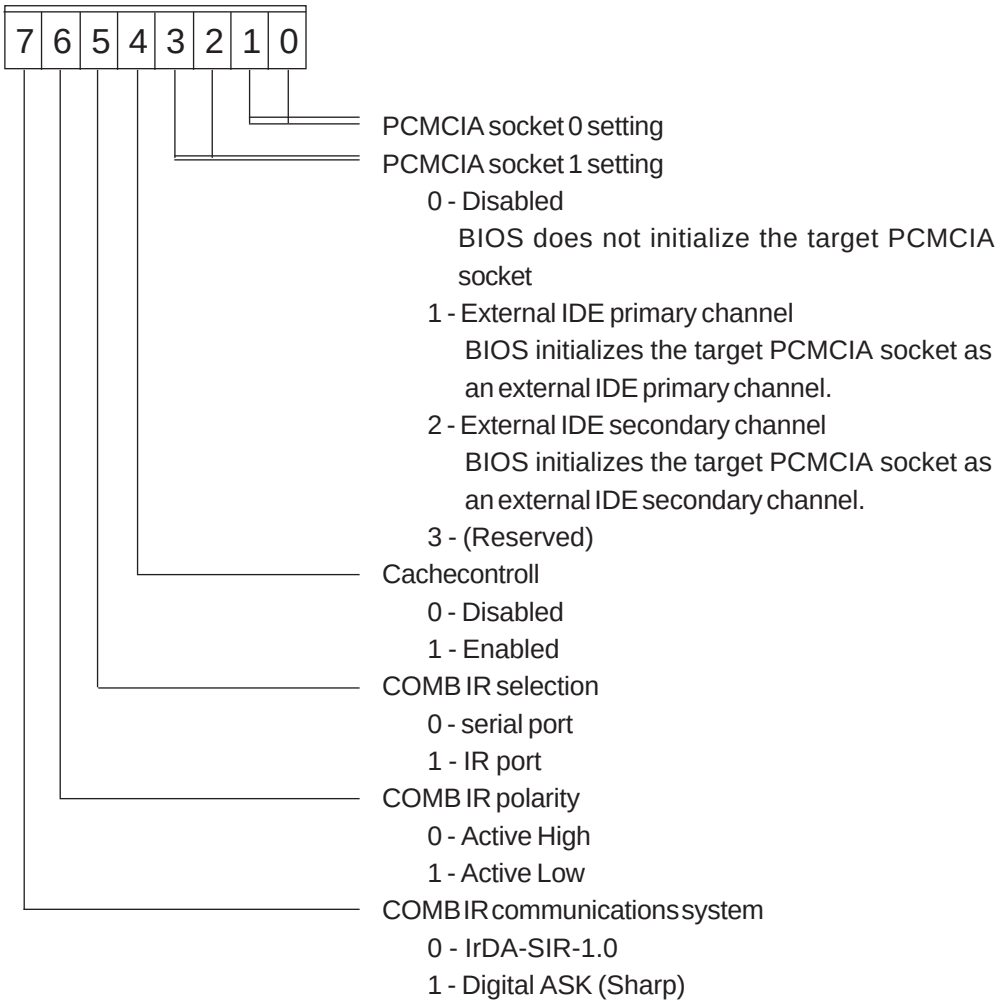


Diskettebootcontrol
- 0 - Enabled
- 1 - Disabled

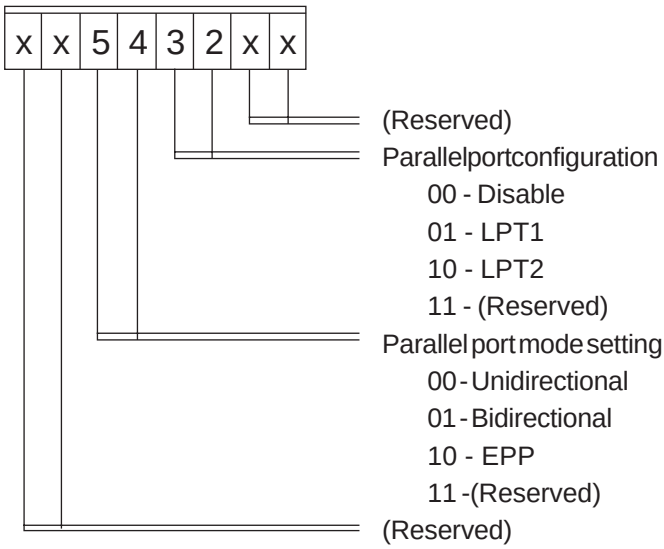
Boot HDD
- 0 - Primary HDD
- 1 - Secondary HDD

(Reserved)

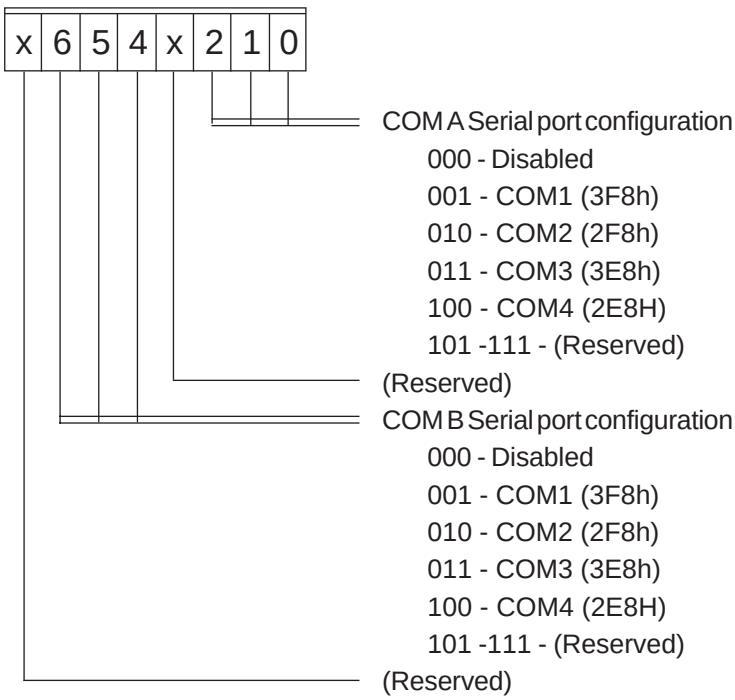
1Bh - Function Support Byte



1Ch - Parallel Configuration byte

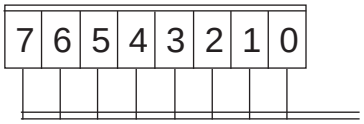


1Dh - Serial Configuration Byte



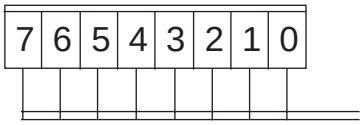
1Eh - 1Fh Reserved

20h - User-defined Primary HDD 1 (Master): Parameter 1



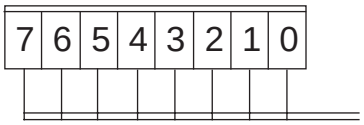
Cylinder lower bits 7 to 0 (-1) of 16 bits

21h - User-defined Primary HDD 1 (Master): Parameter 2



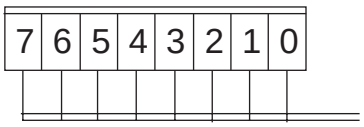
Cylinder upper bits 15 to 8 (-1) of 16 bits

22h - User-defined Primary HDD 1 (Master): Parameter 3



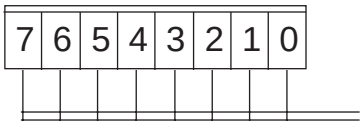
Number of sectors per track

23h - User-defined Primary HDD 2 (Slave): Parameter 1



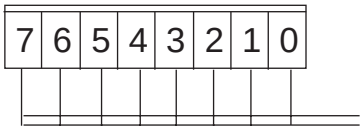
Cylinder lower bits 7 to 0 (-1) of 16 bits

24h - User-defined Primary HDD 2 (Slave): Parameter 2



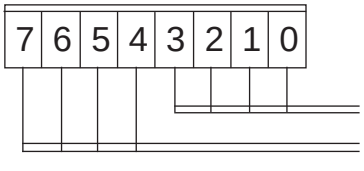
Cylinder upper bits 15 to 8 (-1) of 16 bits

25h - User-defined Primary HDD 2 (Slave): Parameter 3



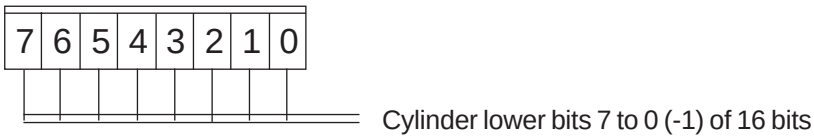
Number of sectors per track

26h - User-defined primary HDD parameters

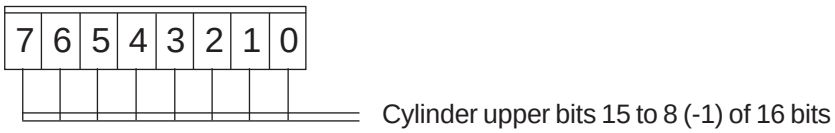


Number of heads for primary HDD2 (Slave) (-1)
Number of heads for primary HDD1 (Master) (-1)

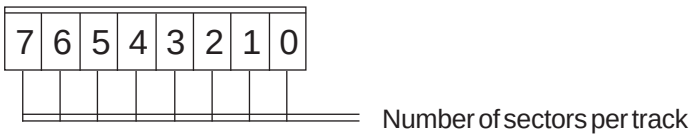
27h - User-defined Secondary HDD 1 (Master): Parameter 1



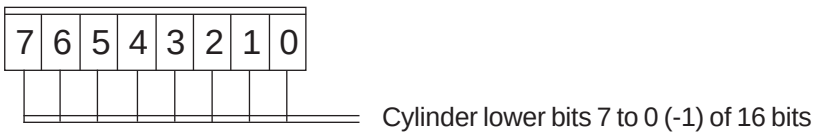
28h - User-defined Secondary HDD 1 (Master): Parameter 2:



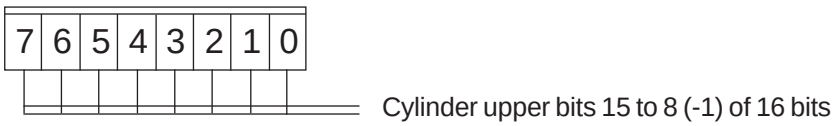
29h - User-defined Secondary HDD 1 (Master): Parameter 3:



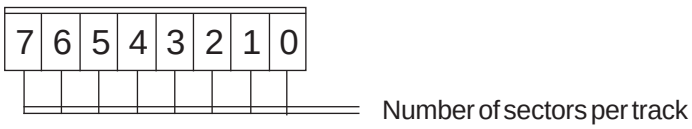
2Ah - User-defined Secondary HDD 2 (Slave): Parameter 1



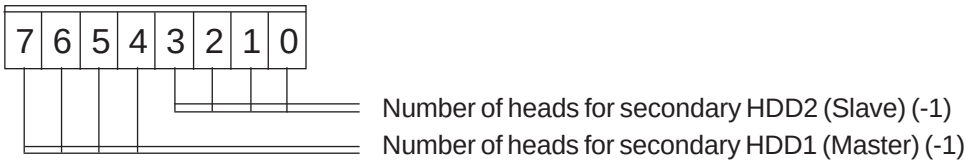
2Bh - User-defined Secondary HDD 2 (Slave): Parameter 2



2Ch - User-defined Secondary HDD 2 (Slave): Parameter 3



2Dh - User-defined secondary HDD parameters



2Eh-2Fh - CMOS-RAM Checksum

CMOS-RAM Checksum for address 10h-2Dh.

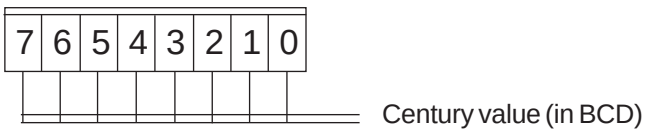
2Eh - High Byte of CMOS-RAM Checksum
2Fh - Low Byte of CMOS-RAM Checksum

30h-31h - Extended Memory Size

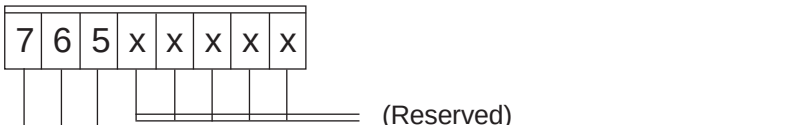
These bytes are calculated by POD.

30h - Low byte of Extended Memory Size in increments of 1K byte.
31h - High byte of Extended Memory Size in increments of 1K byte.

32h - Century Byte



33h - Information Flags



(Reserved)
RTC lost power detection
0 - Power was not lost
1 - Power was lost
FDD setup error
0 - No error
1 - FDD seek error detected
128k (512-640k) memory is installed
0 - Not installed
1 - Installed

34h-35h - Reserved

36h-3Dh - Password Buffer

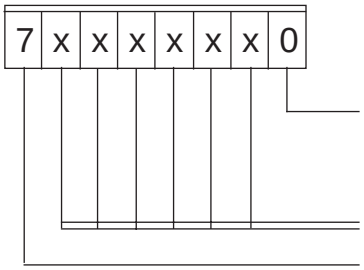
These bytes are set by Setup, or BIOS password routine.

3Eh-3Fh - Password Checksum

3Eh - High Byte of Password Checksum.
3Fh - Low Byte of Password Checksum.

Minimizer Extended CMOS-RAM Area

50h Power Management Control

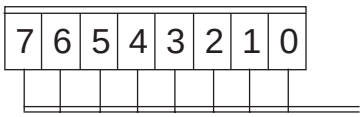


Time and Date Correction Control
- 1 - Correct the time and date
- 0 - Do not correct the time and date

(Reserved)

Power Management Control
- 0 - Disable
- 1 - Enable

51h CPU Standby Timer



CPU Standby Timer

00h	Disable
01h-FFh	4 - 1020 seconds (in units of 4 seconds)

52h - 53h Globbal Standby Timer

52h - Low byte of Global Standby Timer
53h - High byte of Global Standby Timer

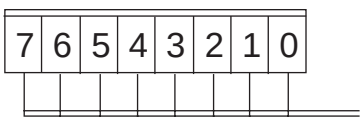
0000h	Disable
0001h - FFFFh	4 - 262140 seconds (in units of 4 seconds)

54h - 55h Auto Suspend Timer

54h - Low byte of Auto Suspend Timer
55h - High byte of Auto Suspend Timer

0000h	Disable
0001h - 7FFFh	4 - 262140 seconds (in units of 4 seconds)

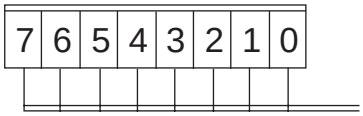
56h Video Standby Timer
(Used in systems with a built-in VGA controller.)



Video Standby Timer

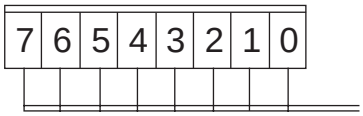
00h	Disable
01h - FFh	4 - 1020 seconds (in units of 4 seconds)

57h COMA Local Standby Timer



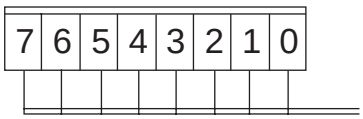
Device 1 Local Standby Timer
- 00h Disable
- 01h - FFh 4 - 1020 seconds (in units of 4 seconds)

58h COMB Local Standby Timer



Device 2 Local Standby Timer
- 00h Disable
- 01h - FFh 4 - 1020 seconds (in units of 4 seconds)

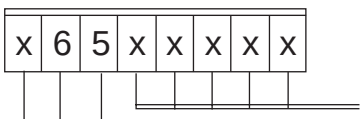
59h HDD Local Standby Timer



Device 3 Local Standby Timer
- 00h Disable
- 01h - FFh 4 - 1020 seconds (in units of 4 seconds)

5Ah - 5Bh Reserved

5Ch Battery Low Control Byte



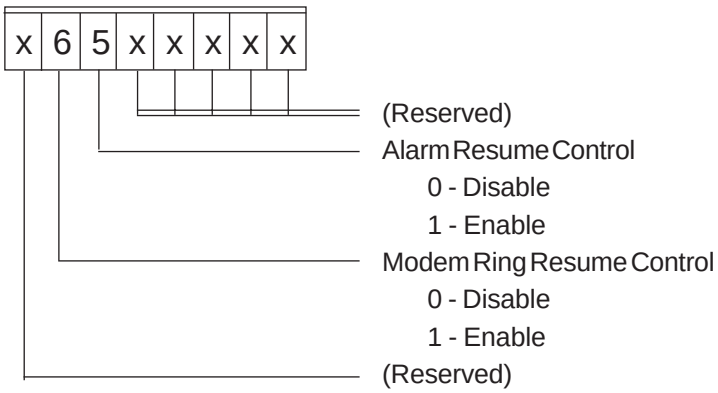
(Reserved)

Low Battery Suspend Control
- 0 - Do not enter suspend state
- 1 - Enter suspend state on low battery condition

Warning Beep Control
- 0 - Disable
- 1 - Enable

(Reserved)

5Dh Resume Control Byte



5Eh Resume Alarm Hour

Hours to resume (BCD)

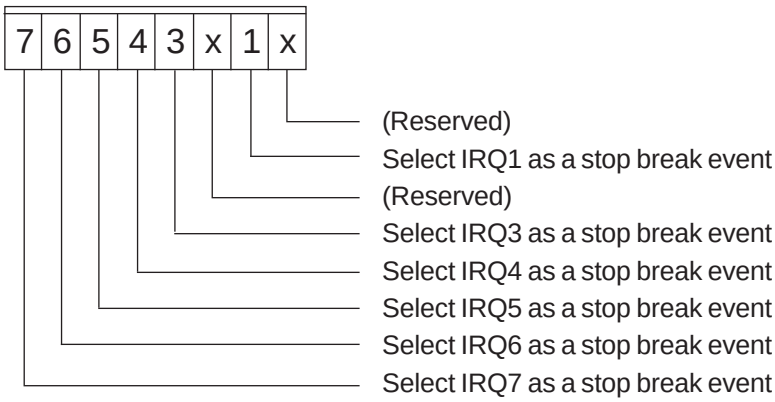
5Fh Resume Alarm Minute

Minutes to resume (BCD).

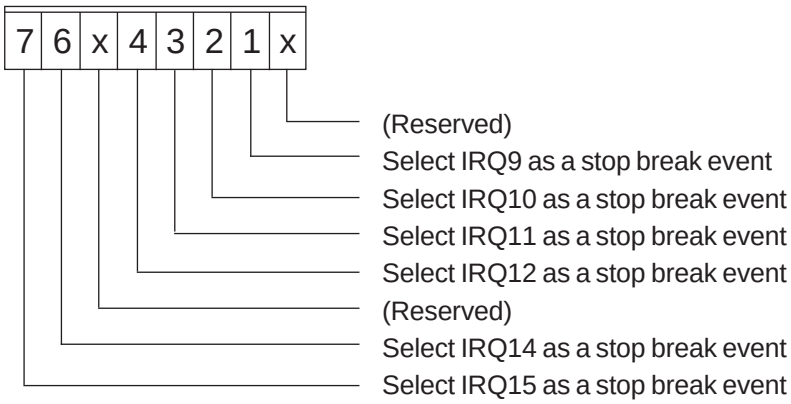
60h Resume Alarm Second

Seconds to resume (BCD).

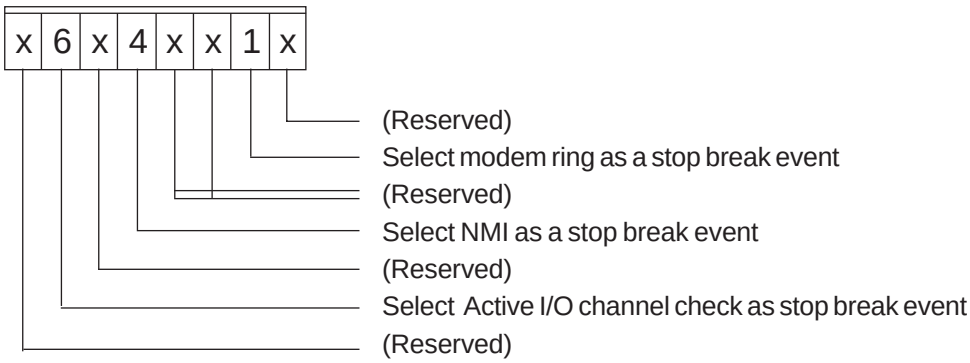
61h Stop Break Setting 0



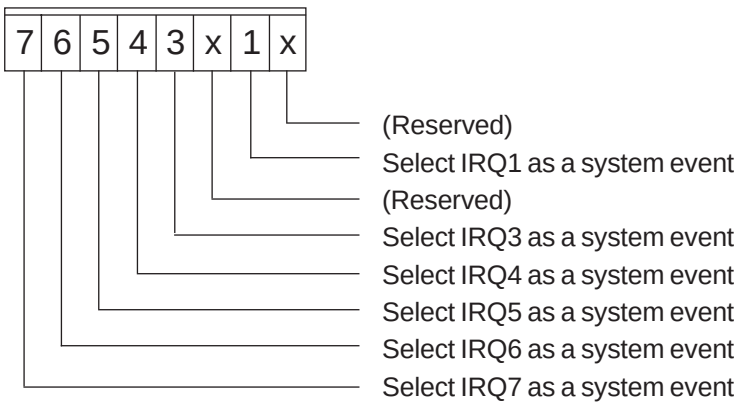
62h Stop Break Setting 1



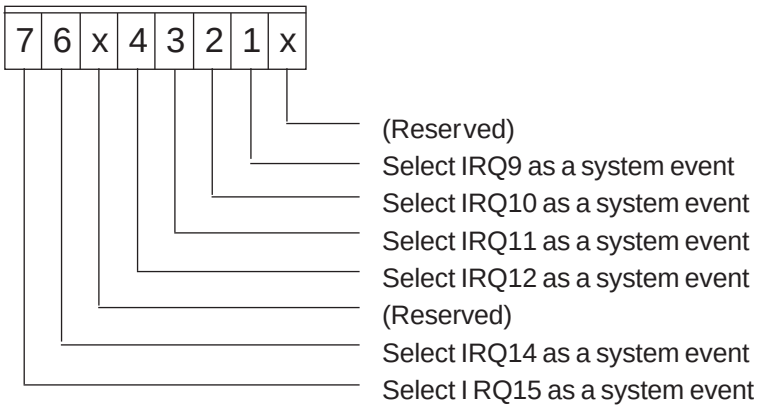
63h Stop Break Setting 2



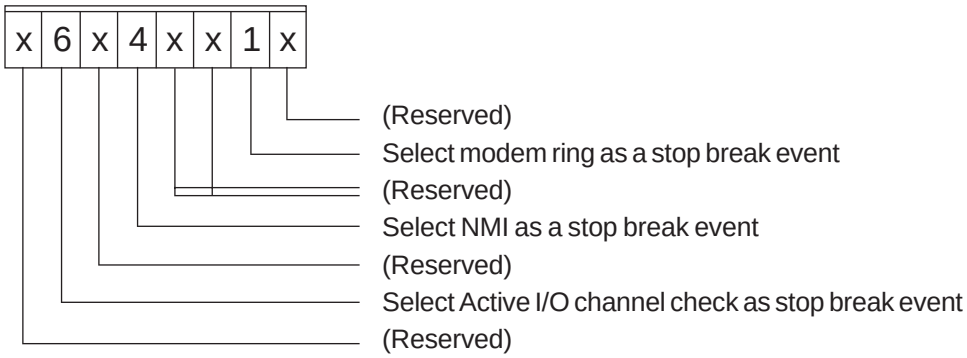
64h System Event Setting 0



65h System Event Setting 1



66h System Event Setting 2



67h-69h (Reserved)

6Ah-6Bh Minimizer CMOS-RAM Checksum

Checksum for address 60h-69h.

6Ah - High byte of Minimizer CMOS-RAM checksum
6Bh - Low byte of Minimizer CMOS-RAM checksum

6Ch-6Fh (Reserved)

POD Step ID Numbers

POD Main Module

StepID	Step name	Description of step
10h	ADiag initialization	POD initializes the connection with the ADiag host.
11h	Chipset initialization	POD initializes the chipset registers that were not set by the boot block (concerning the CPU and I/O).
12h	ADiag initialization	POD re-initializes the connection with the ADiag host. (This step is identical to POD ID 10h.)
13h	Video disable	POD disables external VGA, and detects internal VGA. (If internal VGA exists, POD initializes it and then disables it temporarily.) POD disables any external color/monochrome video card (CGA/MDA).
14h	Base 640K clear	POD clears the base 640K of memory.
15h	RTC CMOS initialization	POD verifies the CMOS restart bit. POD checks the CMOS checksum; if an error is found, the appropriate processing is performed. If power for the real time clock was lost, POD performs the appropriate processing for that. POD clears the CMOS RTC DIAG byte (70.0Eh), and confirms the updating of the UIP bit. POD also checks the validity clock data (year, month, day, hours, minutes, seconds). If an error is detected, POD writes the specified default value to CMOS.
16h	Device information data initialization	POD initializes the device information data byte (40.10h) in the BIOS work area.
17h	Keyboard emulator initialization	POD sets up SMRAM in the main memory space, and detects the Minimizer module. Copies the module to addresses 30000h through 3FFFFh, and then issues an SMI in order to perform keyboard emulator initialization phases 1 and 2. POD then clears addresses 30000h through 3FFFFh and deletes SMRAM from the main memory space.
18h	Timer initialization	POD sets timer 0 to 60[micro]s, and then waits for 75[micro]s in order to confirm that the interrupt is being generated. Next, POD sets timer 0 to 200[micro]s, then waits for 120[micro]s in order to confirm that the interrupt is not being generated, and then waits for 1000[micro]s in order to confirm that the interrupt is being generated. After completing these checks, POD initializes timer 0.

POD Step ID Numbers

StepID	Step name	Description of step
19h	Interrupt vector initialization	POD initializes the software interrupt vectors. After confirming that no NMI is being generated (or issuing a halt if an NMI is being generated), POD initializes the hardware interrupt vectors.
1Ah	Interrupt enable	POD enables interrupts by issuing the STI instruction.
1Bh	System timer initialization	POD reads the time from the RTC and sets the system timer value. In addition, if the century byte (CMOS-RAM 70.32h) is set to 1980 or earlier, POD sets it to 2000. (This addresses the "Year 2000" problem.)
1Ch	Open Gate A20	
1Dh	Video controller initialization	If MDA detection is enabled, POD detects the MDA adapter, sets the MDA information in the BIOS work area device information byte (40.10h), changes the screen mode to 07h, and then checks the MDA card. If CGA detection is enabled, POD detects the CGA adapter, sets the MDA information in the BIOS work area device information byte (40.10h), changes the screen mode to 03h (80-character setting) or 01h (40-character setting), and then checks the CGA card. POID then uses Int 10h Function 0 to clear the screen. In the VGA initial settings, if an internal VGA is present, POD puts the internal VGA into the suspended state temporarily, and checks the option ROM at addresses C0000h through C8000h. POD then enables the external VGA chip through the external VGA enable port (I/O port 46E8h), and initializes the VGA BIOS at addresses C0000h through C8000h.
1Eh	BIOS information display	POD displays the copyright message and the user information.
1Fh	Keyboard initialization	POD checks for the existence of the keyboard controller, issues the self-test command, and then issues the keyboard initialization command. After checking that the keyboard clock is not set low, POD issues the keyboard reset command and confirms that ACK is returned. After waiting for a keyboard interrupt for about one second, POD confirms that the Basic Assurance Test code is returned and that an interrupt is initiated. If an error is detected, error message 301 is displayed. If there is no response from the keyboard, POD checks the interface with the keyboard. If an error is detected, error message 303 is displayed.

StepID	Step name	Description of step
20h	ROM setup prompt display	POD displays the message "Press <F10> to run setup."
21h	Preparation for initialization of PCMCIA HDD	If the PCMCIA socket is enabled, POD confirms that there is no conflict with the CMOS-RAM settings for PCMCIA 0/1 (70.1Bh[3:0]). POD then initializes the PCMCIA 0/1 registers for ATA Boot. After enabling the internal HDD and initializing it, POD disables the internal HDD.
22h	Initialization of all memory	If this is a cold start, then after displaying "Esc to skip", POD tests the main memory control lines. If an error is detected, error message 204 is displayed and execution is halted. After detecting the upper limit of memory, POD performs processing related to the memory hole setting. POD then detects external memory. If external memory and internal memory are located at different addresses, error message 181 is displayed. POD then checks for crosstalk in the memory address lines. If crosstalk is detected, error message 203 is displayed. In addition, the memory is set so that all addresses up to the address where the crosstalk was detected can be used. POD then conducts a memory read/write test. If a location in memory cannot be read or written normally, error message 201 is displayed. In addition, the memory is set so that all addresses up to the address where the error was detected can be used.
23h	Cache enable	If the "enable cache" specification value is set in the CMOS function support byte (70.1Bh[4]), POD enables the cache. (If power for the CMOS is lost, the cache is enabled.)
24h	Extended BIOS work area initialization	POD allocates a 1K extended BIOS work area at the end of base memory.

25h	PS/2 mouse initialization	After enabling the PS/2 mouse interrupt through cascaded control, POD issues the reset command and confirms that ACK is returned. (POD retries this operation up to three times.) If "RESEND" is returned, it is assumed that there is no mouse connected. If ACK is not returned, POD issues the interface test command and waits for one second. If an error or no response is returned, error message 8601 is displayed. When the "Full Test" specification is in effect, POD confirms that IRQ12h is generated and checks the completion code. If the ID is not "00", error message 8603 is displayed. POD sets the "mouse present" flag in the BIOS work area.
26h	Serial port initialization	POD checks the validity of the values set for the addresses of COMA and COMB in CMOS, and then checks for conflicts in the settings for COMA and COMB. (If a conflict is detected, error 1102 is displayed.) If an internal serial port and an external serial port conflict, POD disables the internal serial port. After initializing the serial port setting register, POD checks the serial port if such a check is specified; if an error is detected, error message 1101 is displayed. POD then stores the serial port addresses, the number of serial ports, and the timeout time in the device information data bytes (40.11h),(40.00h) - (40:07h), and (40:7Ch) - (40: 7Fh).
27h	Parallel port initialization	POD checks the validity of the values set for the addresses of the parallel ports in CMOS-RAM. If an internal parallel port and an external parallel port conflict, POD disables the internal parallel port. After initializing the parallel port setting register, POD initializes the extended parallel port settings (standard, bi-directional, EPP). POD then checks the parallel port if such a check is specified; if an error is detected, error message 901 is displayed. POD then stores the parallel port addresses, the number of parallel ports, and the timeout time in the device information data bytes (40.11h),(40.08h) - (40:0Dh), and (40:78h) - (40: 7Ah).

StepID	Step name	Description of step
28h	FDD initialization	If it detects an external FDC, POD disables the internal FDC and checks the FDC main status register. POD initializes the internal FDC and makes the necessary setting for supporting a 3-mode FDD (setting the handling of pin 2 in the FDD interface). After initializing the FDC and the device information data byte (40.10h[7:6]), POD sets the IPL bit in the BIOS work area. If an error is detected, error message 601 is displayed.
29h	HDD initialization	POD initializes the PCMCIA socket in accordance with the value set in 1Bh in CMOS. Before initializing the HDD, POD waits the number of seconds set in the BIOS parameters. If no external HDD controller is connected, POD enables the internal HDD controller. When the "Full Test" specification is in effect, POD detects the internal HDD controller. If it detects an error, POD resets the HDD controller and tries to detect it again. If an error is still detected, POD disables the internal HDD controller. POD initializes the HDC and the secondary HDC.
2Ah	Coprocessor check	POD detects the coprocessor. If the result does not match the coprocessor installation setting in CMOS-RAM, error 701 is displayed.
2Bh	F10 key detection	
2Ch	Close Gate A20	
30h	Option ROM initialization	If the address range starting from E0000h is set as option ROM in the BIOS parameters, POD checks for the existence of option ROM in the area C0000h - EFFFFh, executes a checksum test, and then executes the instructions in option ROM.
31h	Error message display	If any of the following errors were generated before video initialization and stored in the CMOS DIAG byte (70.0Eh), etc., this POD ID is displayed. RTC power loss: error 161 Clock data error: error 163 UIP check error: error 165 CMOS checksum error: error 162 CGA/MDA initialization error: error 401

POD Step ID Numbers

StepID	Step name	Description of step
32h	F1 key input processing	If F1 key input is required (for error processing), the message "Resume = F1 key" is displayed and the system gives two short beeps. The system enters a loop until either the F1 key or the F10 key is pressed. If the keyboard is locked, the message "Unlock system unit keylock" is displayed. If the beep setting (for normal processing) has been made, the system gives one short beep.
33h	F10 key detection	(Identical to POD ID 2Bh.)
34h	Screen clear	POD gets the current screen mode and clears the video screen by issuing a mode change instruction that changes the screen mode to the same screen mode.
35h	POD work area clear	POD clears the IDT definition area and the DMA page register.
36h	E0000h - EFFFFh ROM initialization	If system ROM is located in addresses E0000h through EFFFFh, POD conducts a checksum test and then executes the option ROM (system ROM).
37h	Password processing	In network server mode, POD only sets the 8042 security function. If network server mode is not set, and if power for the CMOS was not lost ,POD calculates the password checksum. If the information in CMOS is valid, POD waits for the password to be input. (Three attempts are allowed.) If "/" is input, password cancellation processing, password change processing, etc., is performed.
38h	Parity and NMI control	POD checks the I/O channels and initializes the ports. In addition, if the internal memory supports parity checking, POD sets the parity check bit for the chip set.
40h	Minimizer startup	POD maps SMRAM to the main memory space, detects the Minimizer module, issues the SMI needed for initialization phase 3, and then selects SMRAM from the main memory space.
41h	ROM Setup startup	If the F1 key was pressed, POD maps the BIOS ROM to FC00000h - FFFFFFh, and detects the Setup module; if the Setup module is found, POD starts up the Setup program.
42h	System timer initialization	(Identical to POD ID 1Bh.)
43h	Remote console initialization	If the remote console is enabled in the BIOS parameters, POD executes handshaking with the serial port in order to set the transfer rate, and receives the signature block. If the remote console connection is normal, POD substitutes the video (int 10h) and keyboard input (int 16h) interrupt handlers.

Boot block

StepID	Step name	Description of step
E0h	CPU initialization	Initializes the CPU's special registers, and the chip set's CPU-related registers. If initialization fails, execution is halted.
E1h	I/O special register initialization	Initializes the chip set's special I/O registers, or initializes the ADiag function. If initialization fails, execution is halted.
E2h	DRAM refresh	Initializes and tests the timer. After initializing the DMA controller, POD starts the DRAM refresh cycle. If initialization fails, execution is halted.
E3h	DRAM type determination	Determines the type of DRAM that is installed. If a memory error is detected, execution is halted.
E4h	Memory check	Checks the memory from 0K to 512K, needed for boot block operation. If a memory error is detected, execution is halted.
E5h	Dispatch loop start	Starts the dispatch loop for the subsequent bot block functions (E6h through EEh).
E6h	Shadow RAM check	Checks shadow RAM. If an error in shadow RAM is detected, execution is halted.
E7h	Memory cache control	If cache memory is installed, enables the memory cache. If a cache error is detected, execution is halted.
E8h	Interrupt table initialization	Allocates the interrupt table to a temporary routine for checking purposes.
E9h	Interrupt controller initialization	Tests and initializes the 8259 interrupt controller. If initialization fails, execution is halted.
EAh	System timer initialization	Initializes and tests the system timer. If initialization fails, execution is halted.
EBh	Load program through serial port	The host looks for the execution file and, if it finds it, sends it through the serial port; the file is then executed.
ECh	Extract BIOS into shadow RAM	Extracts the BIOS image into shadow RAM. If extraction fails, execution is halted.
EDh	POD startup	Initializes the keyboard emulator, and prepares for POD startup (by abandoning the cache and opening Gate A20). If keyboard emulator initialization fails, execution is halted.
EEh	Boot block patch startup	Executes the boot block patch, if any.
EFh	CBIOS startup	Transfers control to CBIOS.

POD ERROR MESSAGES

The POD checks the status of all devices when the system boots up.

If the POD finds a fatal error during the check of the devices, it writes the step ID to port 80h, and then halts the system.
The step ID indicates where the error was found so that it is easier to determine what error caused the halt. If the POD detects a nonfatal error, it displays an error message and waits after F1 is pressed, and all devices that passed the check can be accessed normally.

This section explains the POD error messages.

Message	Port (80h)	Operation after message appears	Hardware source of error
(None)	15h	HALT	RTC

[Explanation] This error message indicates that the RTC CMOS read/write test failed.

Message	Port (80h)	Operation after message appears	Hardware source of error
161 - Real Time Clock has lost power	31h	POD continues	RTC

[Explanation] This error message indicates that the status of the RTC backup battery that is connected to the system is not normal. Replace the backup battery.

Message	Port (80h)	Operation after message appears	Hardware source of error
162 - System options not set	31h	POD continues	RTC

[Explanation] This error message indicates that a checksum error was detected in the CBIOS, VGA, or Minimizer areas in CMOS. This error message may also indicate that the hardware information that is stored in CMOS does not correspond with the actual system hardware. Use SETUP to set the correct information in the CMOS-RAM.

Message	Port (80h)	Operation after message appears	Hardware source of error
163 - Time & Date not set	31h	POD continues	RTC

[Explanation] This error message indicates that the time and date information stored in CMOS is not valid.

Message	Port (80h)	Operation after message appears	Hardware source of error
164 - Memory size error	22h	POD continues	Memory

[Explanation] This error message indicates that the memory size that is set in the CMOS-RAM does not match the size of the memory that is actually installed in the system. Use SETUP to set the correct memory size in CMOS-RAM.

Message	Port (80h)	Operation after message appears	Hardware source of error
165 - Real Time Clock not working	31h	POD continues	RTC

[Explanation] This error message indicates that the "Update in Progress" (UIP) bit in RTC control register A was not updated.

Message	Port (80h)	Operation after message appears	Hardware source of error
181 - System extend memory overlapped	22h	POD continues	Memory

[Explanation] This error message indicates that external memory was detected in the range of addresses where internal memory resides. The system will operate without hindrance, but it is recommended that external memory be positioned so that it does not overlap with internal memory.

Message	Port (80h)	Operation after message appears	Hardware source of error
201 - Memory Error	22h	POD continues	Memory

[Explanation] This error message indicates that an error occurred during the memory read/write test. The memory range is set so that all memory up to the address where the error occurred can be used.

Message	Port (80h)	Operation after message appears	Hardware source of error
203 - Memory address error	22h	POD continues	Memory

[Explanation] This error message indicates that an error was detected during the memory address test. Although memory was divided into 64K blocks and different data was written to each block, the same data was read out of more than one block. The memory range is set so that all memory up to the address where the error occurred can be used.

Message	Port (80h)	Operation after message appears	Hardware source of error
204 - Memory control line error	22h	HALT	Memory

[Explanation] This error message indicates that an error was detected during the memory control line test.

Message	Port (80h)	Operation after message appears	Hardware source of error
301 - Keyboard error	1Fh	POD continues	Keyboard

[Explanation] This error message indicates that an error was discovered during keyboard initialization. Either no keyboard is connected, or one of the keys is stuck. This error does not indicate a problem with the keyboard interface.

Message	Port (80h)	Operation after message appears	Hardware source of error
302 - System unit keylock is locked	1Fh	POD continues	Keyboard

[Explanation] This error message indicates that the keyboard is locked. (This error is not generated by the keyboard emulator. This error is only generated in a system in which a keyboard controller is installed.)

Message	Port (80h)	Operation after message appears	Hardware source of error
303 - Keyboard or system unit error	1Fh	POD continues	Keyboard

[Explanation] This error message indicates that an error was detected during the keyboard interface test.

Message	Port (80h)	Operation after message appears	Hardware source of error
304 - Keyboard or system unit error	1Fh	POD continues	Keyboard

[Explanation] This error message indicates that there is a problem with the keyboard clock line.

Message	Port (80h)	Operation after message appears	Hardware source of error
401 - CRT Error	31h	POD continues	External MDA/ CGA board

[Explanation] This error message indicates that an error occurred during the initialization of an external MDA or CGA board.

Message	Port (80h)	Operation after message appears	Hardware source of error
601 - Diskette error	28h	POD continues	Floppy disk controller

[Explanation] This error message indicates that there is no FDD installed in the system, although the CMOS-RAM settings indicate that an FDD is installed in the system. Correct the setting in CMOS-RAM.

Message	Port (80h)	Operation after message appears	Hardware source of error
701 - Coprocessor install error	2Ah	POD continues	Coprocessor

[Explanation] This error message indicates that there is no coprocessor installed in the system, although the CMOS-RAM settings indicate that a coprocessor is installed in the system. It is possible that the setting in CMOS-RAM is incorrect. However, if the CMOS-RAM settings were made using SETUP, the bit that indicates the existence of a coprocessor is automatically set to the correct value. In this case, it is likely that the coprocessor has failed.

Message	Port (80h)	Operation after message appears	Hardware source of error
901 - Parallel port error	27h	POD continues	Parallel port

[Explanation] This error message indicates that an error occurred during parallel port initialization.

Message	Port (80h)	Operation after message appears	Hardware source of error
1101 - Serial port error	26h	POD continues	Serial port

[Explanation] This error message indicates that an error occurred during serial port initialization.

Message	Port (80h)	Operation after message appears	Hardware source of error
1102 - Serial port conflict	26h	POD continues	Serial port

[Explanation] This error message indicates that the CMOS settings for internal serial ports COMA and COMB are in conflict. Use SETUP to correct the settings for the internal serial ports.

Message	Port (80h)	Operation after message appears	Hardware source of error
1770 - Primary Disk 0 parameter error	29h	POD continues	CMOS-RAM
1771 - Primary Disk 1 parameter error	29h	POD continues	CMOS-RAM
1772 - Secondary Disk 0 parameter error	29h	POD continues	CMOS-RAM
1773 - Secondary Disk 1 parameter error	29h	POD continues	CMOS-RAM

[Explanation] These error messages are displayed when the HDD parameters stored in CMOS-RAM are in error. The drive in question is not initialized.

Message	Port (80h)	Operation after message appears	Hardware source of error
1782 - Disk controller failure	29h	POD continues	HDD

[Explanation] This error message indicates a problem with the IDE HDD controller that is connected to the system.

Message	Port (80h)	Operation after message appears	Hardware source of error
1780 - Primary Disk 0 failure	29h	POD continues	HDD
1781 - Primary Disk 1 failure	29h	POD continues	HDD
1783 - Secondary Disk 0 failure	29h	POD continues	HDD
1784 - Secondary Disk 1 failure	29h	POD continues	HDD

[Explanation] These error messages indicate that HDD initialization failed.

Message	Port (80h)	Operation after message appears	Hardware source of error
1790 - Primary Disk 0 error	29h	POD continues	HDD
1791 - Primary Disk 1 error	29h	POD continues	HDD
1792 - Secondary Disk 0 error	29h	POD continues	HDD
1793 - Secondary Disk 1 error	29h	POD continues	HDD

[Explanation] After initializing an HDD, BIOS tests that HDD. These error messages are displayed when an error occurred during that testing.

Message	Port (80h)	Operation after message appears	Hardware source of error
8601 - Auxiliary device failure	25h	POD continues	Mouse interface

[Explanation] This error message indicates that an error was detected during the PS/2 mouse interface test.

Message	Port (80h)	Operation after message appears	Hardware source of error
8602 - Auxiliary device failure	25h	POD continues	Mouse interface

[Explanation] This error message indicates that an error was detected during PS/2 mouse initialization. This error does not indicate a problem with the PS/2 mouse interface. This message is displayed when the PS/2 mouse does not respond to commands.

Message	Port (80h)	Operation after message appears	Hardware source of error
8603 - Auxiliary device failure	25h	POD continues	Mouse interface

[Explanation] This error message indicates that an error was detected during PS/2 mouse initialization. This error does not indicate a problem with the PS/2 mouse interface. This message is displayed when the PS/2 mouse previously responded to commands, but no longer responds to commands.

Message	Port (80h)	Operation after message appears	Hardware source of error
- Unlock system unit keylock	32h	POD continues	

[Explanation] This error message indicates that the keyboard lock signal is being input to the keyboard controller.

In pursuit of "Saving" Technology, Epson electronic devices. Our lineup of semiconductors, liquid crystal displays and quartz devices assists in creating the products of our customers' dreams.
Epson IS energy savings.

EPSON DBIOS Manual
(Basic Functions)

SEIKO EPSON CORPORATION
ELECTRONIC DEVICES MARKETING DIVISION

Electronic Devices Information on Epson WWW server
http://www.epson.co.jp

First issue March,1998
Printed September, 1998 in Japan Ⓟ B