BIOS and DOS Interrupts

INTERRUPT 10h FUNCTIONS (VIDEO)
INTERRUPT 16h FUNCTIONS (KEYBOARD)
INTERRUPT 21h FUNCTIONS (DOS SERVICES)

Interrupt Number	Description
0	***Divide by Zero.*** Internal: activated when attempting to divide by zero.
1	***Single Step.*** Internal: activated when a memory error is executed.
2	***Nonmaskable.*** Internal: activated when a memory error occurs.
3	***Breakpoint.*** Internal: activated when the 0CCh (INT 3) instruction is executed.
4	***Overflow.*** Activated when the INTO instruction is executed and the Overflow flag is set.
5	***Print Screen.*** Activated either by the INT 5 instruction or tying SHIFT-PRTSC.
6	***(Reserved)***
7	***(Reserved)***
8	***Timer Interrupt.*** Updates the BIOS clock 18.2 times per second. For your own programming, see INT 1Ch.
9	***Keyboard Hardware Interrupt.*** Activated when a key is pressed. Reads the key from the keyboard port and stores it in the keyboard typeahead buffer.
0A-0D	***(Reserved)***
0E	***Diskette Interrupt.*** Activated when a disk seek is in progress.
0F	***(Reserved)***
10	***Video Services.*** Routines for manipulating the video display (see the complete list in a subsequent table).
11	***Equipment Check.*** Return a word showing all the peripherals attached to the system.
12	***Memory Size.*** Return the amount of memory (in 1,024-byte blocks) in AX.
13	***Disk Services.*** Reset the disk controller, get the status of the most recent disk access, read and write physical sectors, and format a disk.
14	***Asynchronous (Serial) Port Services.*** Initialize and read or write the asynchronous communications port, and return the port's status.
16	***Keyboard Services.*** Read and inspect keyboard input (see the complete list in a subsequent table).
17	***Printer Services.*** Initialize, print, and return the status of the printer.
18	***ROM BASIC.*** Execute cassette BASIC in ROM.
19	***Boot Strap.*** Reboot DOS.
1A	***Time of Day.*** Get the number of timer ticks since the machine was turned on, or set the counter to a new value. Ticks occur 18.2 times per second.
1B	***Keyboard Break.*** This interrupt handler is executed by INT 9h when CTRL-BREAK is pressed.
1C	***User Timer Interrupt.*** Empty routine, executed 18.2 times per second. May be used by your own program.
1D	***Video Parameters.*** Point to a table containing initialization and information for the Video Controller chip.
1E	***Diskette Parameters.*** Point to a table containing initialization information for the diskette controller.
1F	***Graphics Table.*** Table kept in memory of all extended graphics with ASCII codes greater than 127.
20	***Terminal Program.*** Terminate the current program (INT 21h, function 4Ch should be used instead.
21	***DOS Services*** (see the complete list in the next table).
22	***DOS Terminate Address.*** Point to the address of the parent program or process. When the current program ends, this will be the return address.
23	***DOS Break Address.*** DOS jumps here when CTRL-BREAK is pressed.
24	***DOS Error Address.*** DOS jumps to this address when there is a critical error in the current program, such as a disk media error.
25	***DOS Disk Read.*** Used for reading logical sectors from the disk.
26	***DOS Disk Write.*** Used for writing logical sectors on the disk.
27	***Terminate and Stay Resident.*** Exit to DOS or the calling program, but leave the current program in memory.
28-3F	(Reserved)
40,41	***Fixed Disk Services.*** Fixed disk controller.
4B-7F	(Available for application programs)
80-F0	(***Reserved:*** used by BASIC)
F1-FF	(Available for application programs)

INTERRUPT 21h FUNCTIONS (DOS SERVICES)

Function number is loaded in AH for each function

Function	Description
0	***Program Terminate.*** Terminate the current program (use function 4Ch instead).
1	***Keyboard Input.*** Wait for a character from the standard input device: echo it on the console. **Output:** AL = character.
2	***Display Output.*** Display the character in DL on the standard output device (console).
3	***Auxiliary Input.*** Wait for a character from the standard auxiliary device (serial). **Output:** AL = character.
4	***Auxiliary Output.*** Send the character in DL to the standard auxiliary device (serial).
5	***Pointer Output.*** Send the character in DL to the standard printer device.
6	***Direct Console Input/Output.*** If DL = FFh, read a waiting character from the console. **Output:** AL = character. If DL = 00h-FEh, output a character instead.
7	***Direct Console Input Without Echo.*** Wait for a character from the standard input device. The character is returned in AL but is not echoed.
8	***Console Input Without Echo.*** Wait for a character from the standard input device. The character is returned in AL but is not echoed. May be terminated by using CTRL-BREAK.
9	***Print String.*** Output an ASCIIZ string of characters to the standard output device. **Input:** DS:DX = address of the string.
0A	***Buffered Keyboard Input.*** Read a string of characters from the standard input device. **Input:** DS:DX = address of the buffer.
0B	***Check Standard Input Status.*** Check to see if an input character is waiting. **Output:** AL = 0FFh if the character is ready; otherwise, AL = 0.
0C	***Clear Keyboard Buffer, Invoke Input Function.*** Clear the console input buffer, and then execute an input function. **Input:** AL = desired function (1, 6, 7, 8, or 0Ah).
0D	***Disk Reset.*** Flush all file buffers.
0E	***Select Disk.*** Set the default drive. **Input:** DL = drive number (0 = A, 1 = B, etc.)
0F	***Open File.*** Open a file described by an FCB (file control block). **Input:** DS:DX = address of the unopened FCB.
10	***Close File.*** Close the file whose FCB is pointed to by DS:DX.
11	***Search for First Entry.*** Search for the first matching filename in the current directory. **Input:** DS:DX = address of an unopened FCB.
12	***Search for Next Entry.*** Search for the next matching filename in the current directory. **Input:** DS:DX = address of an unopened FCB. **Output:** AL = 0 if filename found, FFh if not found.
13	***Delete File.*** Delete all matching directory entries that match the filename. **Input:** DX:DX = address of an unopened FCB with the filename. **Output:** AL = 0 if the file was deleted. AL = FFh if the file was not found.
14	***Sequential Read.*** Read the next record of the file pointed to by an opened FCB into memory, and increment the record number. **Input:** DS:DX points to the FCB. **Output:** AL = 0 if successful.
15	***Sequential Write.*** Write the record identified by the FCB from the current disk transfer address (DTA). **Input:** DS:DX points to the FCB. **Output:** AL = 0 if successful.
16	***Create File.*** Create a new file, or open one if it already exist. **Input:** DS:DX points to an unopened FCB. **Output:** AL = 0 successful.
17	***Rename File.*** Rename a file. Both the old and new names are in a modified, unopened FCB. **Input:** DS:DX points to the FCB. **Output:** AL = 0 if successful; AL = FFh if the file was not found or the new name currently exists.
18	***(Reserved)***

19	***Get Current Disk***. Return the current default driver. **Output:** AL = drive number (0 = A, 1 = B, etc.).
1A	***Set Disk Transfer Address***. Set the DTA to the location pointed to by DS:DX.
1B	***Allocation Table Information***. Get information about the allocation table for the default drive. **Output:** DS:BX points to the media descriptor byte, DX = number of clusters, AL = number of sectors per cluster, and CX = sector size.
1C	***Allocation Table Information for Specific device***. Same as function 1Bh, except that you can specify the drive number in DL when the function is called (0 = default, 1 = drive A, etc.).
1D,1E	***(Reserved)***
1F	***Get Drive Parameter Block (DPB) for current drive***. Gets information about the disk parametersfor the current drive **Output:** AL = 0 if successful. DS:BX = segment:offset of DPB
20	***(Reserved)***
21	***Random Read***. Read the record addressed by the current FCB into memory at the disk transfer address. **Input:** DS:DX points to the FCB. **Output:** AL = 0 if successful.
22	***Random Write***. Write the record addressed by the current FCB into memory from the disk transfer address. **Input:** DS:DX points to the FCB. **Output:** AL = 0 if successful.
23	***File Size***. Return the number of records in a file based on the record size in the file's FCB. **Input:** DS:DX points to the unopened FCB. **Output:** If successful, AL = 0 and the random record field in the FCB contains the number of records in the file.
24	***Set Relative (Random) Record Field***. Set the random record field to the offset as the current block and record fields in the file's FCB. **Input:** DS:DX points to the FCB.
25	***Set Interrupt Vector***. Set an entry in the Interrupt Vector Table to a new address. **Input:** DS:DX points to the interrupt-handling routine that will be inserted in the table; AL = the interrupt number.
26	***Create New Program Segment***. Create a new program segment by copying the current program segment prefix (PSP) to a new segment address. Modify the PSP so it can be sued by a new program. This function has been superseded by function 4Bh (EXEC).
27	***Random Block Read***. Read a specified number of records from the file based on the random record filed in its FCB. **Input:** DS:DX points to the opened FCB, and CX = number of records to read. **Output:** AL = 0 if successful, and CX = number of records read.
28	***Random Block Write***. Write a specified number of records to the file based on the random record field in its FCB. **Input:** DS:DX points to the opened FCB, and CX = number of records to be written. **Output:** AL = 0 if successful, and CX = number of records written.
29	***Parse Filename***. The command line is parsed for filename in the form ***d:filename.ext*** , and an unopened FCB is created.
2A	***Get Date***. Return the system date. **Output:** AL = day of the week (0-6, where Sunday = 0), CX = year, DH = month, and DL = DAY.
2B	***Set Date***. Set the system date. **Input:** CX = year, DH = month, and DL = day. **Output:** AL = 0 if the date was valid.
2C	***Get Time***. Return the system time. **Output:** CH = hour, CL = minutes, DH = seconds, and DL = hundredths of seconds.
2D	***Set time***. Set the system time. **Input:** CH = hour, DH = seconds, CL = minutes, and DL = hundredths of seconds. **Output:** AL = 0 if the time was valid.
2E	***Set/Reset Verify Switch***. Either set or turn off the switch that DOS uses to perform an extra verity during each disk read/write operation. **Input:** AL = 0 to turn verify switch off, or AL = 1 to turn the switch on.
2F	***Get Disk Transfer Address (DTA)***. Return the current DTA in ES:BX.
30	***Get DOS Version Number***. Return the current DOS version number. **Output:** AL = major version number, and AH = minor version number.
31	***Terminate Process and Remain Resident***. Terminate the current program or process, and attempt to set the current memory allocation to the number of paragraphs specified in DX. **Input:** AL = return code, and DX = requested number of paragraphs.
32	***Get DOS Driver Parameter Block (DPB) for Specific Driver*** ***Input:*** *DL = Drive number* ***Output:*** *DS:BX = Segment:offset of DPB*

33	***Set of subfunctions***. **Input:** AL = **Value** **Subfunction** 00H Get Ctrl-Break Flag 01H Set Ctrl-Break Flag 02H Get and Set Extended Control-Break Checking State 05H Determine Boot Drive Number 06H Get Actual Version Number
34	***Get Address of InDOS Flag***. Reads the address of Critical Section (InDOS) flag. This indicates that INT 21H function is working. **Output:**ES:BX = segment:offset of InDOS flag
35	***Get Interrupt Vector***. Get the segment-offset value of an interrupt vector. **Input:** AL = interrupt number. **Output:** ES:BX = address of the interrupt handler.
36	***Get Disk Free Space***. Return the amount of disk free space. **Input:** DL = drive number (0 = default, 1 = A, etc.). **Output:** AX = sector per cluster, or FFFFh if the drive number is invalid; BX = number of available clusters, CX = bytes per sector, and DX = clusters per drive.
37	***(Reserved)***
38	***Return Country Dependent Information***. Return a pointer to a list of country-specific data, such as the currency symbol, thousands separator, and decimal separator. for details.
39	***Create Subdirectory***. Create a new subdirectory based on a given path name. **Input:** DS:DX points to an ASCIIZ string with the path and directory name. **Output:** AX = error code if the Carry flag is set.
3A	***Remove Subdirectory***. Remove a subdirectory. **Input:** DS:DX points to an ASCIIZ string with the path and directory name. **Output:** AX = error code if the Carry flag is set.
3B	***Change the Current Directory***. Change to a different directory. **Input:** DS:DX points to an ASCIIZ string with the new directory path. **Output:** AX = error code if the Carry flag is set.
3C	***Create a File***. Create a new file or truncate an old file to 0 bytes. Open the file for output. **Input:** DS:DX points to an ASCIIZ string with the filename, and CX = file attribute. **Output:** AX = error code if the Carry flag is set; otherwise AX = the new file handle.
3D	***Open a File***. Open a file for input, output, or input-output. **Input:** DS:DX points to an ASCIIZ string with the filename, and AL = the access code (0 = read, 1 = write, 2 = read/write). **Output:** AX = error code if the Carry flag is set, otherwise AX = the new file handle.
3E	***Close a File Handle***. Close the file or device specified by a file handle. **Input:** BX = file handle from a previous open or create. **Output:** AX = error code if the Carry flag is set.
3F	***Read from a file or Device***. Read a specified number of bytes from a file or device. **Input:** BX = file handle, DS:DX points to an input buffer, and CX = number of bytes to read. **Output**: AX = error code if the carry flag is set.
40	***Write to a File or Device***. Write a specified number of bytes to a file or device. **Input:** BX = file handle, DS:DX points to the output buffer, and CX = number of bytes to write. **Output:** AX = error code if the Carry flag is set.
41	***Delete a File***. Remove a file from a specified directory. **Input:** DS:DX points to an ASCIIZ string with the filename. **Output:** AX = error code if the Carry flag is set.
42	***Move File Pointer***. Move the file read/write pointer according to a specified method. **Input:** CX:DX = distance (bytes) to move the file pointer. AL = method code, and BX = file handle. The method codes are as follows: 0 = move from beginning of the file, 1 = move to the current location plus an offset, and 2 = move to the end of file plus an offset. **Output:** AX = error code if the Carry flag is set.
43	***Get/Set file Attribute***. Get or set the attribute of a file. **Input:** DS:DX = pointer to a ASCIIZ path and filename, CX = attribute, and AL = function code (1 = set attribute, 0 = get attribute). **Output:** AX = error code if the Carry flag is set.

44	***I/O Control for Devices***. **Input:**AL = **Value** **Subfunction** 00H Get Device Information 01H Set Device Information 02H IOCTL: Read Control Data from Character Device Driver 03H IOCTL: Write Control Data from Character Device Driver 04H IOCTL: Read Control Data from Block Device Driver 05H IOCTL: Write Control Data from Block Device Driver 06H IOCTL: Check Input Status 07H IOCTL: Check Output Status 08H IOCTL: Check if Block Device is Removable 09H IOCTL: Check if Block Device is Remote 0AH IOCTL: Check if Handle is Remote 0BH IOCTL: Change Sharing Retry Acount 0CH IOCTL: Generic I/O Control for Character Device 0DH IOCTL: Generic I/O Control for Block Device 0EH IOCTL: Get Logical Drive Map 0FH IOCTL: Set Logical Drive Map 10H IOCTL: Query Generic IOCTL Capability (Handle) 11H IOCTL: Query Generic IOCTL Capability (Drive)
45	***Duplicate a File Handle***. Return a new file handle for a file that is currently open. **Input:** BX = file handle. **Output:** AX = error code if the Carry flag is set.
46	***Force a Duplicate of a Handle***. Force the handle in CX to refer to the same file at the same position as the handle in BX. **Input:** BX = existing file handle, and CX = second file handle. **Output:** AX = error code if the Carry flag is set.
47	***Get Current Directory***. Get the full path name of the current directory. **Input:** DS:SI points to a 64-byte area to hold the directory path, and DL = drive number. **Output:** A buffer at DS:SI is filled with the path, and AX = error code if the Carry flag is set.
48	***Allocate Memory***. Allocate a requested number of paragraphs of memory, measured in 16-byte blocks. **Input:** BX - number of paragraphs requested. **Output:** AX = segment of the allocated block, BX = size of the largest block available (in paragraphs), and AX = error code if the Carry flag is set.
49	***Free Allocated Memory***. Free memory that was previously allocated by function call 48h. **Input:** ES = segment of the block to be freed. **Output:** AX = error code if the Carry flag is set.
4A	***Modify Memory Blocks***. Modify allocated memory blocks to contain a new block size. The block will shrink or grow. **Input:** EX = segment of the block, and BX = requested number of paragraphs. **Output:** AX = error code if the Carry flag is set, and BX = maximum number of available blocks.
4B	***Load or Execute a Program***. Create a PSP for another program, load it into memory, and execute it. **Input:** DS:DX points to an ASCIIZ string with the drive, path, and filename of the program; ES:BX points to a parameter block, and AL = function value. Function values in AL: 0 load and execute the program; 1 load but do not execute (overlay program). 3 load overlay 5 set execution state **Output:** AX = error code if the Carry flag is set.
4C	***Terminate a Process***. Usual way to terminate a program and return to either DOS or a calling program. **Input:** AL = 8-bit return code, which can be queried by DOS function 4Dh or by the ERRORLEVEL command in a batch file.
4D	***Get Return Code of a Process***. Get the return code of a process or program, generated by either function call 31h or function call 4Ch. **Output:** AL = 8-bit code returned by the program, AH = type of exit generated: 0 = normal termination, 1 = terminated by CTRL-BREAK, 2 = terminated by a critical device error, and 3 = terminated by a call to function call 31h.
4E	***Find First Matching File***. Find the first filename that matches a given file specification. **Input:** DS:DX points to an ASCIIZ drive, path, and file specification; CX = file attribute to be used when searching. **Output:** AX = error code if the Carry flag is set; otherwise, the current DTA is filled with the filename, attribute, time, date, and size. DOS function call 1Ah (Set DTA) is usually called before this function.
4F	***Find Next Matching File***. Find the next filename that matches a given file specification. This is always called after DOS function 4Eh. **Output:** AX = error code if the Carry flag is set; otherwise, the current DTA is filled with the file's information.
50	***Set Current Process ID (Set PSP address)***. Sets a new value of PID into "current process" variable **Input:** BX = segment address of new PSP
51	***Get Current Process ID (Get PSP Address)***. Reads the segment address of the current PSP. **Output:** BX = segment address of current PSP
52	***(Reserved)***
53	***Translate BIOS Parameter Block to DOS Drive Parameter Block***. Transforms drive Parameter information from the BIOS Parameter Block which is stored on the disk to the format used by DOS. **Input:**DS:SI = segment:offset of BIOS Parameter Block (BPB) ES:DI = segment:offset for Driver Parameter Block (DPB) **Output:** ES:BP address of buffer wih information
54	***Get Verify Setting***. Get the current value of the DOS verify flag (see DOS function 2Eh). **Output:** AL = current verify flag (1 = on, 0 = off).
55	***(Reserved)***
56	***Rename a File***. Rename a file or move a file to another directory. **Input:** DS:DX points to a ASCIIZ string that specifies the current drive, path, and filename; ES:DI points to the new path and filename. **Output:** AX = error code if the Carry flag is set.
57	***Get/Set a File's Date and Time***. Get or set the date and time stamp for a file. **Input:** AL = to get the date/time, or AL = 1 to set the date/time; BX = file handle, CX = new file time, and DX = new file date. **Output:** AX = error code if the Carry flag is set; otherwise, CS = current file time, and DX = current file date.
58	***Memory Functions***. Memory allocation: **Input:** AL = **Value** **Subfunction** 00H Get Allocation Strategy 01H Set Allocation Strategy 02H Get Upper-Memory Link 03H Set Upper-Memory Link
59	***Get extended Error Information***. Return additional information about a DOS error, including the error class, locus, and recommended action. **Input:** BX = DOS version number (0 for version 3.xx). **Output:** Ax = extended error code, BH = error class, BL = suggested action, and CH = locus.
5A	***Create Unique File***. Generate a unique (temporary) filename in a specified directory. **Input:** DS:DX points to a ASCIIZ pathname, ending with a backslash (\); CX = desired file attribute. **Output:** AX = error code if the Carry flag is set; otherwise, DS:DX points to the path with the new filename appended.
5B	***Create New File***. Try to create a new file, but fail if the filename already exists. This prevents you from overwriting an existing file. **Input:** DS:DX points to an ASCIIZ string with the path and filename. **Output:** AX = error code if the Carry flag is set.
5C	***Lock/Unlock File Access***. Lock or unlock a range of bytes in an opened file (networking).
5D	***(Reserved)***
5E	***Functions designed for Networks***. **Input:** AL = Subfunction **Value** 00 Get Local Machine Name 01 Set Redirection Mode 02 Set Printer Setup String 03 Get Printer Setup String
5F	***I/O Redirection***. **Input:** AL = Subfunction **Value** 00 Get Redirection Mode 01 Set Redirection Mode 02 Get Redirection List Entry 03 Redirect Device

	04 Cancel Device Redirection
60	(Reserved)
61	(Reserved)
62	***Get Program Segment Prefix (PSP) Address.*** **Output:** BX = PSP address of the current program.
63	(Reserved)
64	(Reserved)
65	***Subfunctions deal with Extended Country information.*** **Input: AL =** **value** **subfunction** 01H Get general international information 02H Get pointer to uppercase table 04H Get pointer to filename uppercase table 05H Get pointer to filename terminator table 06H Get pointer to collating table 07H Get pointer to double-byte chatacter set vector 20H Capitalize character 21H Capitalize string 22H Capitalize ASCIIZ string **Output:**
66	***Get or Set Code Page.*** DOS reads or sets the current code page. **Input:** BX = code page if AL = 02H. **Output:** If successful, BX = active code page, DX = default code page.
67	***Set Handle Count.*** Sets the maximum number of handles available to the calling programs. (Effectively, this only reduces the number available.) **Input:** BX = number of desired handles.
68	***Commit File Handle.*** Forces all data in DOS buffers associated with a specific handle to be written immediately to the specified device. If the handle points to a file which has been updated, the file's directory is also updated.
69	***Get/Set Disk Serial Number.*** Reads or writes the volume label and serial number of a specified disk. **Input:**BL = drive, DS:DX = segment:offset disk serial number info
6C	***Extended Open/Create function.*** Combines functions "create", "open", and "commit" into a common function. **Input:** File attributes and filename **Output:**CS = 1 if file opened, 2 if file created, 3 if file replaced.

INTERRUPT 10h FUNCTIONS (VIDEO)

Function	Description
0	***Set Video Mode.*** Set the video display to monochrome, text, graphics, or color mode. **Input:** AL = display mode.
1	***Set Cursor Lines.*** Identify the starting and ending scan lines for the cursor. **Input:** CH = cursor starting line, and CL = cursor ending line.
2	***Set Cursor Position.*** Position the cursor on the screen. **Input:** BH = video page. DH = row, and DL = column.
3	***Get Cursor Position.*** Get the cursor's screen position and its size. Input; BH = video page. **Output:** CH = cursor starting line. CL = cursor ending line, DH = cursor row, DL = cursor column, AH = status.
4	***Read Light Pen.*** Read the position and status of the light pen. **Output:** CH = pixel row, BX = pixel column, DH = character row, and DL = character column, AH = status.
5	***Set Display Page.*** Select the video page to be displayed. **Input:** AL = desired page number.
6	***Scroll Window Up.*** Scroll a window on the current video page upward, replacing scrolled lines with blanks. **Input:** AL = number of lines to scroll, BH = attribute for scrolled lines, CX = upper left corner row and column, and DX = lower right row and column.
7	***Scroll Window Down.*** Scroll a window on the current video page downward, replacing scrolled lines with blanks. **Input:** AL = number of lines to scroll, BH = attribute for scrolled lines, CX = upper left corner row and column, and DX = lower right row and column.
8	***Read Character and Attribute.*** Read the character and its attribute at the current cursor position. **Input:** BH = display page, AH = attribute byte, and AL = ASCII character code.
9	***Write Character and Attribute.*** Write the character and its attribute at the current cursor position. **Input:** AL = ASCII character, BH = video page, BL = attribute or color, and CX = repetition factor.
0A	***Write Character.*** Write a character only (no attribute) at the current cursor position. **Input:** AL = ASCII character, BH = video page, BL = attribute, and CX = replication factor.
0B	***Set Color Palette.*** Select a group of available colors for the EGA or VGA adapter. **Input:** BL = color valve, and BH = color palette ID.
0C	***Write Graphics Pixel.*** Write a graphics pixel when in color graphics mode. **Input:** AL = pixel value, CX = ***x*** coordinate, and DX = ***y*** coordinate.
0D	***Read Graphics Pixel.*** Read the color of a single graphics pixel at a given location. **Input:** CX = ***x*** coordinate, and DX = ***y*** coordinate. **Output:** AL = pixel color.
0E	***Write Character.*** Write a character to the screen and advance the cursor. **Input:** AL = ASCII character code, BH = video page, BL = attribute or color
0F	***Get current Video Mode.*** Get the current video mode. **Output:** AL = video mode, and BH = active video page.
10	***Set Video Palette.*** (EGA and VGA) Set the video palette register, border color, or blink/intensity bit. **Input:** AL = function code (00 = set palette register, 01 = set border color, 02 = set palette and border color, 03 = set/reset blink/intensity bit), BH = color, BL = palette register to sets. If AL = 02, ES:DX points to a color list.
11	***Character Generator.*** Select the character size for the EGA or VGA display. For example, an 8 by 8 font is used for the 43-line display, and an 8 by 14 font is used for the 25-line display.
12	***Alternate Select Function.*** Return technical information about the EGA or VGA display.
13	***Write String.*** (AT BIOS and later) Write a string of text to the video display. **Input:** AL = mode, BH = page, BL = attribute, CX = length of string, DH = row, DL = column, and ES:BP points to the string.

INTERRUPT 16h FUNCTIONS (KEYBOARD)

<table>
<tr><th>Function</th><th>Description</th></tr>
<tr><td>00</td><td><b><i>Read Character.</i></b> Wait for an input character and keyboard scan code.
<b>Output:</b> AH = scan code, and AL = ASCII character.</td></tr>
<tr><td>01</td><td><b><i>Get Keyboard Status.</i></b> Find out if a character is waiting in the keyboard typeahead buffer. If one is, return the scan code in AH and the ASCII code in AL, and clear the Zero flag. If no key is waiting, set the Zero flag. (Note: the same character will remain in the keyboard buffer.)</td></tr>
<tr><td>02</td><td><b><i>Get Keyboard Flags.</i></b> Return the keyboard flag byte stored in low RAM in AL. The meaning of each bit is:
<table>
<tr><td>Insert on</td><td>Caps Lock on</td><td>Num Lock on</td><td>Scroll Lock on</td><td>Alt key down</td><td>Ctrl key down</td><td>Left shift down</td><td>Right shift down</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td></tr>
</table></td></tr>
<tr><td>10</td><td>Extended keyboard input (F11 + F12)</td></tr>
</table>