Because BIOS interrupt calls use CPU register-based parameter passing, the calls are oriented to being made from assembly language and cannot be directly made from most high-level languages (HLLs). However, a high level language may provide a library of wrapper routines which translate parameters from the form (usually stack-based) used by the high-level language to the register-based form required by BIOS, then back to the HLL calling convention after the BIOS returns. In some variants of C, BIOS calls can be made using inline assembly language within a C module. (Support for inline assembly language is not part of the ANSI C standard but is a language extension; therefore, C modules that use inline assembly language are less portable than pure ANSI standard C modules.)

Invoking an interrupt

Invoking an interrupt can be done using the INT x86 assembly language instruction. For example, to print a character to the screen using BIOS interrupt 0x10, the following x86 assembly language instructions could be executed:

```
mov ah, 0x0e    ; function number = 0Eh : Display Character
mov al, '!'     ; AL = code of character to display
int 0x10        ; call INT 10h, BIOS video service
```

Interrupt table

A list of common BIOS interrupt classes can be found below. Note that some BIOSes (particularly old ones) do not implement all of these interrupt classes.

The BIOS also uses some interrupts to relay hardware event interrupts to programs which choose to receive them or to route messages for its own use. The table below includes only those BIOS interrupts which are intended to be called by programs (using the "INT" assembly-language software interrupt instruction) to request services or information.

Interrupt vector	Description
05h	Executed when Shift-Print screen is pressed, as well as when the BOUND instruction detects a bound failure.
	Video Services

<table>
<tr><td>10h</td><td>
<table>
<tr><th>AH</th><th>Description</th></tr>
<tr><td>00h</td><td>Set Video Mode</td></tr>
<tr><td>01h</td><td>Set Cursor Shape</td></tr>
<tr><td>02h</td><td>Set Cursor Position</td></tr>
<tr><td>03h</td><td>Get Cursor Position And Shape</td></tr>
<tr><td>04h</td><td>Get Light Pen Position</td></tr>
<tr><td>05h</td><td>Set Display Page</td></tr>
<tr><td>06h</td><td>Clear/Scroll Screen Up</td></tr>
<tr><td>07h</td><td>Clear/Scroll Screen Down</td></tr>
<tr><td>08h</td><td>Read Character and Attribute at Cursor</td></tr>
<tr><td>09h</td><td>Write Character and Attribute at Cursor</td></tr>
<tr><td>0Ah</td><td>Write Character at Cursor</td></tr>
<tr><td>0Bh</td><td>Set Border Color</td></tr>
<tr><td>0Ch</td><td>Write Graphics Pixel</td></tr>
<tr><td>0Dh</td><td>Read Graphics Pixel</td></tr>
<tr><td>0Eh</td><td>Write Character in TTY Mode</td></tr>
<tr><td>0Fh</td><td>Get Video Mode</td></tr>
<tr><td>10h</td><td>Set Palette Registers (EGA, VGA, SVGA)</td></tr>
<tr><td>11h</td><td>Character Generator (EGA, VGA, SVGA)</td></tr>
<tr><td>12h</td><td>Alternate Select Functions (EGA, VGA, SVGA)</td></tr>
<tr><td>13h</td><td>Write String</td></tr>
<tr><td>1Ah</td><td>Get or Set Display Combination Code (VGA, SVGA)</td></tr>
<tr><td>1Bh</td><td>Get Functionality Information (VGA, SVGA)</td></tr>
<tr><td>1Ch</td><td>Save or Restore Video State (VGA, SVGA)</td></tr>
<tr><td>4Fh</td><td>VESA BIOS Extension Functions (SVGA)</td></tr>
</table>
</td></tr>
<tr><td>11h</td><td>Returns equipment list</td></tr>
<tr><td>12h</td><td>Return conventional memory size</td></tr>
<tr><td></td><td>Low Level Disk Services</td></tr>
</table>

<table>
<tr><td>13h</td><td>
<table>
<tr><th>AH</th><th>Description</th></tr>
<tr><td>00h</td><td>Reset Disk Drives</td></tr>
<tr><td>01h</td><td>Check Drive Status</td></tr>
<tr><td>02h</td><td>Read Sectors</td></tr>
<tr><td>03h</td><td>Write Sectors</td></tr>
<tr><td>04h</td><td>Verify Sectors</td></tr>
<tr><td>05h</td><td>Format Track</td></tr>
<tr><td>08h</td><td>Get Drive Parameters</td></tr>
<tr><td>09h</td><td>Init Fixed Drive Parameters</td></tr>
<tr><td>0Ch</td><td>Seek To Specified Track</td></tr>
<tr><td>0Dh</td><td>Reset Fixed Disk Controller</td></tr>
<tr><td>15h</td><td>Get Drive Type</td></tr>
<tr><td>16h</td><td>Get Floppy Drive Media Change Status</td></tr>
<tr><td>17h</td><td>Set Disk Type</td></tr>
<tr><td>18h</td><td>Set Floppy Drive Media Type</td></tr>
<tr><td>41h</td><td>Extended Disk Drive (EDD) Installation Check</td></tr>
<tr><td>42h</td><td>Extended Read Sectors</td></tr>
<tr><td>43h</td><td>Extended Write Sectors</td></tr>
<tr><td>44h</td><td>Extended Verify Sectors</td></tr>
<tr><td>45h</td><td>Lock/Unlock Drive</td></tr>
<tr><td>46h</td><td>Eject Media</td></tr>
<tr><td>47h</td><td>Extended Seek</td></tr>
<tr><td>48h</td><td>Extended Get Drive Parameters</td></tr>
<tr><td>49h</td><td>Extended Get Media Change Status</td></tr>
<tr><td>4Eh</td><td>Extended Set Hardware Configuration</td></tr>
</table>
</td></tr>
<tr><td>14h</td><td>Serial port services
<table>
<tr><th>AH</th><th>Description</th></tr>
<tr><td>00h</td><td>Serial Port Initialization</td></tr>
<tr><td>01h</td><td>Transmit Character</td></tr>
<tr><td>02h</td><td>Receive Character</td></tr>
<tr><td>03h</td><td>Status</td></tr>
</table>
</td></tr>
<tr><td></td><td>Miscellaneous system services</td></tr>
</table>

15h

AH	AL	Description
00h		Turn on cassette drive motor (IBM PC/PCjr only)
01h		Turn off cassette drive motor (IBM PC/PCjr only)
02h		Read data blocks from cassette (IBM PC/PCjr only)
03h		Write data blocks to cassette (IBM PC/PCjr only)
4Fh		Keyboard Intercept
83h		Event Wait
84h		Read Joystick (BIOSes from 1986 onward)
85h		Sysreq Key Callout
86h		Wait
87h		Move Block
88h		Get Extended Memory Size
89h		Switch to Protected Mode
C0h		Get System Parameters
C1h		Get Extended BIOS Data Area Segment
C2h		Pointing Device Functions
C3h		Watchdog Timer Functions - PS/2 systems only
C4h		Programmable Option Select - MCA bus PS/2 systems only
D8h		EISA System Functions - EISA bus systems only
E8h	01h	Get Extended Memory Size (Newer function, since 1994). Gives results for memory size above 64 Mb.
E8h	20h	Query System Address Map. The information returned from E820 supersedes what is returned from the older `AX=E801h` and `AH=88h` interfaces.

16h

Keyboard services

AH	Description
00h	Read Character
01h	Read Input Status
02h	Read Keyboard Shift Status
05h	Store Keystroke in Keyboard Buffer
10h	Read Character Extended
11h	Read Input Status Extended
12h	Read Keyboard Shift Status Extended

Printer services

<table>
<tr><td>17h</td><td><table><tr><th>AH</th><th>Description</th></tr><tr><td>00h</td><td>Print Character to Printer</td></tr><tr><td>01h</td><td>Initialize Printer</td></tr><tr><td>02h</td><td>Check Printer Status</td></tr></table></td></tr>
<tr><td>18h</td><td>Execute Cassette BASIC: On IBM machines up to the early PS/2 line, this interrupt would start the ROM Cassette BASIC. Clones did not have this feature and different machines/BIOSes would perform a variety of different actions if INT 18h was executed, most commonly an error message stating that no bootable disk was present. Modern machines would attempt to boot from a network through this interrupt.</td></tr>
<tr><td>19h</td><td>After POST this interrupt is used by the BIOS to load the operating system. A program can call this interrupt to reboot the computer (but must ensure that hardware interrupts or DMA operations will not cause the system to hang or crash during either the reinitialization of the system by BIOS or the boot process).</td></tr>
<tr><td>1Ah</td><td>Real Time Clock Services<table><tr><th>AH</th><th>Description</th></tr><tr><td>00h</td><td>Read RTC</td></tr><tr><td>01h</td><td>Set RTC</td></tr><tr><td>02h</td><td>Read RTC Time</td></tr><tr><td>03h</td><td>Set RTC Time</td></tr><tr><td>04h</td><td>Read RTC Date</td></tr><tr><td>05h</td><td>Set RTC Date</td></tr><tr><td>06h</td><td>Set RTC Alarm</td></tr><tr><td>07h</td><td>Reset RTC Alarm</td></tr></table></td></tr>
<tr><td></td><td>PCI Services - implemented by BIOSes supporting PCI 2.0 or later</td></tr>
</table>

<table>
<tr><td>1Ah</td><td><table>
<tr><th>AX</th><th>Description</th></tr>
<tr><td>B101h</td><td>PCI Installation Check</td></tr>
<tr><td>B102h</td><td>Find PCI Device</td></tr>
<tr><td>B103h</td><td>Find PCI Class Code</td></tr>
<tr><td>B106h</td><td>PCI Bus-Specific Operations</td></tr>
<tr><td>B108h</td><td>Read Configuration Byte</td></tr>
<tr><td>B109h</td><td>Read Configuration Word</td></tr>
<tr><td>B10Ah</td><td>Read Configuration Dword</td></tr>
<tr><td>B10Bh</td><td>Write Configuration Byte</td></tr>
<tr><td>B10Ch</td><td>Write Configuration Word</td></tr>
<tr><td>B10Dh</td><td>Write Configuration Dword</td></tr>
<tr><td>B10Eh</td><td>Get IRQ Routine Information</td></tr>
<tr><td>B10Fh</td><td>Set PCI IRQ</td></tr>
</table></td></tr>
<tr><td>1Bh</td><td>Ctrl-Break handler - called by INT 09 when Ctrl-Break has been pressed</td></tr>
<tr><td>1Ch</td><td>Timer tick handler - called by INT 08</td></tr>
<tr><td>1Dh</td><td>Not to be called; simply a pointer to the VPT (Video Parameter Table), which contains data on video modes</td></tr>
<tr><td>1Eh</td><td>Not to be called; simply a pointer to the DPT (Diskette Parameter Table), containing a variety of information concerning the diskette drives</td></tr>
<tr><td>1Fh</td><td>Not to be called; simply a pointer to the VGCT (Video Graphics Character Table), which contains the data for ASCII characters 80h to FFh</td></tr>
<tr><td>41h</td><td>Address pointer: FDPT = Fixed Disk Parameter Table (1st hard drive)</td></tr>
<tr><td>46h</td><td>Address pointer: FDPT = Fixed Disk Parameter Table (2nd hard drive)</td></tr>
<tr><td>4Ah</td><td>Called by RTC for alarm</td></tr>
</table>

BIOS hooks

DOS

On DOS systems, IO.SYS or IBMBIO.COM hooks INT 13 for floppy disk change detection, tracking formatting calls, correcting DMA boundary errors, and working around problems in IBM's ROM BIOS "01/10/84" with model code 0xFC before the first call.

Bypassing BIOS