Interrupts currently supported by emulator

Quick reference:

A list of supported interrupts with descriptions:

INT 10h / **AH = 00h** - set video mode.

input:
AL = desired video mode.

These video modes are supported:

00h - Text mode 40x25, 16 colors, 8 pages.

03h - Text mode 80x25, 16 colors, 8 pages.

INT 10h / **AH = 01h** - set text-mode cursor shape.

input:
CH = cursor start line (bits 0-4) and options (bits 5-7).
CL = bottom cursor line (bits 0-4).

When bits 6-5 of CH are set to **00**, the cursor is visible, to hide a cursor set these bits to **01** (this CH value will hide a cursor: 28h - 00101000b). Bit 7 should always be zero.

INT 10h / **AH = 02h** - set cursor position.

input:
DH = row.
DL = column.
BH = page number (0..7).

INT 10h / **AH = 03h** - get cursor position and size.

input:
BH = page number.
return:
DH = row.
DL = column.
CH = cursor start line.
CL = cursor bottom line.

INT 10h / **AH = 05h** - select active video page.

input:
AL = new page number (0..7).
the activated page is displayed.

INT 10h / **AH = 06h** - scroll up window.
INT 10h / **AH = 07h** - scroll down window.

input:
AL = number of lines by which to scroll (00h = clear entire window).
BH = attribute used to write blank lines at bottom of window.
CH, CL = row, column of window's upper left corner.
DH, DL = row, column of window's lower right corner.

INT 10h / **AH = 08h** - read character and attribute at cursor position.

input:
BH = page number.
return:
AH = attribute.
AL = character.

INT 10h / **AH = 09h** - write character and attribute at cursor position.

input:
AL = character to display.
BH = page number.
BL = attribute.
CX = number of times to write character.

INT 10h / **AH = 0Ah** - write character only at cursor position.

input:
AL = character to display.
BH = page number.
CX = number of times to write character.

INT 10h / **AH = 0Eh** - teletype output.

input:
AL = character to write.

This functions displays a character on the screen, advancing the cursor and scrolling the screen as necessary. The printing is always done to current active page.

INT 10h / **AH = 13h** - write string.

input:
AL = write mode:
bit 0: update cursor after writing;
bit 1: string contains attributes.
BH = page number.
BL = attribute if string contains only characters (bit 1 of AL is zero).
CX = number of characters in string (attributes are not counted).
DL,DH = column, row at which to start writing.
ES:BP points to string to be printed.

INT 10h / **AX = 1003h** - toggle intensity/blinking.

input:
BL = write mode:
0: enable intensive colors.
1: enable blinking (not supported by emulator!).
BH = 0 (to avoid problems on some adapters).

Bit color table:
Character attribute is 8 bit value, low 4 bits set foreground color, high 4 bits set background color. Background blinking not supported.

HEX BIN COLOR

0	0000	black
1	0001	blue
2	0010	green
3	0011	cyan
4	0100	red
5	0101	magenta
6	0110	brown
7	0111	light gray
8	1000	dark gray
9	1001	light blue
A	1010	light green
B	1011	light cyan
C	1100	light red
D	1101	light magenta
E	1110	yellow
F	1111	white

INT 11h - get BIOS equipment list.

return:
AX = BIOS equipment list word, actually this call returns the contents of the word at 0040h:0010h.

Currently this function can be used to determine the number of installed number of floppy disk drives.

Bit fields for BIOS-detected installed hardware:

Bit(s)	Description
15-14	number of parallel devices.
13	not supported.
12	game port installed.
11-9	number of serial devices.
8	reserved.
7-6	number of floppy disk drives (minus 1): 00 single floppy disk; 01 two floppy disks; 10 three floppy disks; 11 four floppy disks.
5-4	initial video mode: 00 EGA,VGA,PGA, or other with on-board video BIOS; 01 40x25 CGA color; 10 80x25 CGA color (emulator default); 11 80x25 mono text.
3	not supported.
2	not supported.
1	math coprocessor installed.
0	set when booted from floppy (always set by emulator).

INT 12h - get memory size.

return:
AX = kilobytes of contiguous memory starting at absolute address 00000h, this call returns the contents of the word at 0040h:0013h.

Floppy drives are emulated using *FLOPPY_0(..3)* **files.**

INT 13h / **AH = 00h** - reset disk system, (currently this call doesn't do anything).

INT 13h / **AH = 02h** - read disk sectors into memory.
INT 13h / **AH = 03h** - write disk sectors.

input:

AL = number of sectors to read/write (must be nonzero)
CH = cylinder number (0..79).
CL = sector number (1..18).
DH = head number (0..1).
DL = drive number (0..3 , depends on quantity of FLOPPY_? files).
ES:BX points to data buffer.

return:

CF set on error.
CF clear if successful.
AH = status (0 - if successful).
AL = number of sectors transferred.

Note: each sector has **512** bytes.

INT 15h / **AH = 86h** - BIOS wait function.

input:

CX:DX = interval in microseconds

return:

CF clear if successful (wait interval elapsed),
CF set on error or when wait function is already in progress.

Note:

the resolution of the wait period is 977 microseconds on many systems, Emu8086 uses 1000 microseconds period.

INT 16h / **AH = 00h** - get keystroke from keyboard (no echo).

return:

AH = BIOS scan code.
AL = ASCII character.
(if a keystroke is present, it is removed from the keyboard buffer).

INT 16h / **AH = 01h** - check for keystroke in keyboard buffer.

return:

ZF = 1 if keystroke is not available.
ZF = 0 if keystroke available.
AH = BIOS scan code.
AL = ASCII character.
(if a keystroke is present, it is not removed from the keyboard buffer).

INT 19h - system reboot.

Usually, the BIOS will try to read sector 1, head 0, track 0 from drive A: to 0000h:7C00h. Emulator just stops the execution, to boot from floppy drive select from the menu: **'Virtual Drive' -> 'Boot from Floppy'**

INT 1Ah / **AH = 00h** - get system time.

return:

CX:DX = number of clock ticks since midnight.
AL = midnight counter, advanced each time midnight passes.

Notes:
There are approximately **18.20648** clock ticks per second, and **1800B0h** per 24 hours.

AL is not set by emulator yet!

MS-DOS can not be loaded completely in emulator yet, so I made an emulation for some basic DOS interrupts also:

INT 20h - exit to operating system.

INT 21h / **AH=09h** - output of a string at DS:DX.

INT 21h / **AH=0Ah** - input of a string to DS:DX, fist byte is buffer size, second byte is number of chars actually read.

INT 21h / **AH=4Ch** - exit to operating system.

INT 21h / **AH=01h** - read character from standard input, with echo, result is stored in AL.

INT 21h / **AH=02h** - write character to standard output, DL = character to write, after execution AL = DL.

Development process never stops,
so check my **homepage** from time to time for an update.