

INT 10h / AH = 0 - 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.
13h - graphical mode. 40x25. 256 colors. 320x200 pixels. 1 page.

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 bit 5 of CH is set to 0, the cursor is visible. when bit 5 is 1,
the cursor is not visible.

```

; hide blinking text cursor:
    mov ch, 32
    mov ah, 1
    int 10h

; show standard blinking text cursor:
    mov ch, 6
    mov cl, 7
    mov ah, 1
    int 10h

; show box-shaped blinking text cursor:
    mov ch, 0
    mov cl, 7
    mov ah, 1
    int 10h
; note: some bioses required CL to be >=7,
; otherwise wrong cursor shapes are displayed.

```

INT 10h / AH = 2 - set cursor position.
input:
DH = row.
DL = column.
BH = page number (0..7).
example:
 mov dh, 10
 mov dl, 20
 mov bh, 0
 mov ah, 2
 int 10h

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.

NT 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 = 0Ch - change color for a single pixel.
input:
AL = pixel color
CX = column.
DX = row.

INT 10h / AH = 0Dh - get color of a single pixel.
input:
CX = column.
DX = row.
output:
AL = pixel color

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.
example:
 mov al, 'a'
 mov ah, 0eh
 int 10h
; note: on specific systems this
; function may not be supported in graphics mode.

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 the emulator and windows
command prompt).
BH = 0 (to avoid problems on some adapters).
example:
 mov ax, 1003h
 mov bx, 0
 int 10h

bit color table:
character attribute is 8 bit value, low 4 bits set fore color, high 4
bits set background color.
note: the emulator and windows command line prompt do not support
background blinking, however to make colors
look the same in dos and in full screen mode it is required to turn off
the background blinking.

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

example:
 mov ah, 1
 int 21h

INT 21h / AH=2 - write character to standard output.
entry: DL = character to write, after execution AL = DL.
example:

```

    mov ah, 2
    mov dl, 'a'
    int 21h

```

INT 21h / AH=5 - output character to printer.
entry: DL = character to print, after execution AL = DL.

example:

```
mov ah, 5
mov dl, 'a'
int 21h
```

INT 21h / AH=7 - character input without echo to AL.
if there is no character in the keyboard buffer, the function waits until any key is pressed.

example:

```
mov ah, 7
int 21h
```

INT 21h / AH=9 - output of a string at DS:DX. String must be terminated by '\$'.

INT 21h / AH=0Ah - input of a string to DS:DX, first byte is buffer size, second byte is number of chars actually read. this function does not add '\$' in the end of string. to print using INT 21h / AH=9 you must set dollar character at the end of it and start printing from address DS:DX + 2.
INT 21h / AH=2Ah - get system date;
return: CX = year (1980-2099). DH = month. DL = day. AL = day of week (00h=Sunday)

INT 21h / AH=2Ch - get system time;
return: CH = hour. CL = minute. DH = second. DL = 1/100 seconds.

INT 21h / AH=4Ch - return control to the operating system (stop program).

INT 33h / AX=0000 - mouse initialization. any previous mouse pointer is hidden.
returns:
if successful: AX=0FFFFh and BX=number of mouse buttons.
if failed: AX=0

example:
mov ax, 0
int 33h

INT 33h / AX=0001 - show mouse pointer.

example:
mov ax, 1
int 33h

INT 33h / AX=0002 - hide visible mouse pointer.

example:
mov ax, 2
int 33h

INT 33h / AX=0003 - get mouse position and status of its buttons.

returns:
if left button is down: BX=1
if right button is down: BX=2
if both buttons are down: BX=3

CX = x
DX = y
example:
mov ax, 3
int 33h