# EXPERIMENT NO- 8

**AIM:** WAP to draw rectangle using int10H

**Resource Required**: P-IV and above RAM 128MB, Dot Matrix Printer, Emu 8086, MASM 611/ TASM, Turbo C/C++, Printer, Printout Stationary.

**THEORY:**

**Instructions used in this program are:**

**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.

example:

```
mov al, 13h
mov ah, 0
int 10h
```

---

**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.

---

**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 = 0Ch** - change color for a single pixel.

*input:*

**AL** = pixel color

**CX** = column.

**DX** = row.

example:

```
mov al, 13h
mov ah, 0
int 10h     ; set graphics video mode.
mov al, 1100b
```

```
mov cx, 10
mov dx, 20
mov ah, 0ch
int 10h     ; set pixel.
```

---

**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.

example:

```
mov al, 1
mov bh, 0
mov bl, 0011_1011b
mov cx, msg1end - offset msg1 ; calculate message size.
mov dl, 10
mov dh, 7
push cs
pop es
mov bp, offset msg1
mov ah, 13h
int 10h
jmp msg1end
msg1 db " hello, world! "
msg1end:
```

---

**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
```

**CONCLUSION: We have successfully drawn rectangle using INT 10H instruction in assembly language using EMU8086.**

## Code and Output: