LAB5 –BIOS INTERRUPTS – DISPLAY

5.1 Introduction to BIOS Interrupts for Display

In addition to the DOS function call INT 21h, we also have video BIOS function calls at INT 10h.The BIOS function calls require less time to execute than the DOS function calls. BIOS function calls allow more control over the video display than do the DOS function calls. For instance, BIOS function calls allow the cursor to be placed at a particular location on the screen while the DOS function calls do not.

There are four main aspects to Video Display

- Setting an appropriate video mode (or resolution, as you know it)
- Reading/Setting the cursor position
- Reading/Writing an ASCII character at a given cursor position
- Working at the pixel level on the screen (for e.g., drawing a line, square on the screen

Blocking Function

To hold the display and its characteristics program must not be allowed to exit. So before .exit statement we have to specify a blocking function

e.g. of a blocking function

```
        mov     ah,07h
x1:     int     21h
        cmp     al,'%'
        jnz     x1
```

System will then retain programmed display mode until % key is pressed.

5.2 Display Interrupts

Purpose: Get Display mode

Input

AH=0Fh

Output

AL=current video mode

AH=number of character columns

BH=page number

Purpose: Set Display mode

Input:

AH = 0

AL = desired video mode

These video modes are supported:

00h - text mode. 40x25. 16 colours. 8 pages.

03h - text mode. 80x25. 16 colours. 8 pages.

12h - graphical mode. 80x25. 256 colours. 720x400 pixels. 1 page.

Note though 8 pages we always use only the first page

Output

Mode updated

Display cleared

Example: Program Segment to set video mode

```
.CODE
        MOV     AH, 00H
        MOV     AL, 3
        INT     10H
```

Purpose: Get cursor position and size

Input

AH = 3

BH = page number (usually 0)

Output

DH = row.

DL = column.

CH = cursor start line.

CL = cursor bottom line.

Purpose: set cursor position.

Input:

AH = 02_H

DH = row.

DL = column.

BH = page number (0...7). Usually 0

Example: Program Segment to set cursor position

```
.CODE
        MOV     AH, 02H
        MOV     DL, 40
        MOV     DH, 12
        MOV     BH, 0
        INT     10H
```

Purpose: Set cursor size

Input:

AH = 01h

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

CL = bottom cursor line (bits 0-4).

Output:

Cursor size changed.

Purpose: Read character at Cursor position

Input:

AH = 08h

Bh = 0 (page no.)

Output:

Error if CF = 1, AX = error code (6)

AH = attribute.

AL = character.

Attribute

The attribute byte is used to specify the foreground and background of the character displayed on the screen.

Bits 2-1-0 represent the foreground colour

Bit 3 represents the intensity of foreground colour (0-low , 1- high intensity)

Bits 6-5-4 represent the background colour

Bit 7 is used for blinking text if set to 1

The 3 bit colour code (with their high intensity counterparts (if bit3 is 1) is

000 -black (gray)

001 -blue (bright blue)

010 -green (bright green)

011 -cyan (bright cyan)

100 -red (bright red)

101 -magenta (bright magenta)

110 -brown (yellow)

111 -white (bright white)

Purpose: Write character at cursor position

Input:

AH = 09h

AL = character to display.

BH = page number.

BL = attribute.

CX = number of times to write character.

Output:

Character displayed at current cursor position CX number of times

Purpose: Fill a pixel

Input:

AH = 0Ch

AL = pixel color

CX = column.

DX = row

Example: Program Segment to cursor position

```
Mov     al, 12h
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.
```

Tasks

1. Display 'DOLL' ***blinking*** in the center of the screen with White letters on Black background with screen resolution at 720X400 pixels in text VGA mode (80colsX25rows) with 16 colours.(cursor needs to be advanced with each character. Use video mode 3
2. Implement a 'dual-window 'editor. The screen should be divided into two equal divisions. You may choose to split the screen horizontally or vertically. The first half must have a background colour of blue and foreground of yellow. The second half must have a background of white and a foreground of bright green. Use video mode 3. As the user types in characters on the keyboard, they must be simultaneously displayed on both the halves. If the user types '$#', then the program must terminate and it must return to the previous video mode.
3. Draw a rectangle (lines red in colour) of area 100x80 pixels with top left corner of the rectangle at pixel position (80, 70). Use video mode 12h
4. Display the following pattern on the top left of the screen. The area of the outermost rectangle is 150 x 250