

**Course Title:** Microprocessors and Assembly Language Lab (CSE-4504)

Department of Computer Science and Engineering (CSE)  
**Islamic University of Technology (IUT), Gazipur**

**Lab # 02**

*8086 I/O Instructions and Condition Control Instructions using Assembly Language.*

**Objective:**

To understand the 8086 I/O instructions using Assembly Language Program in EMU8086.

**Theory:**

• **The INT Instruction**

To invoke a DOS or BIOS routine, the **INT** (interrupt) instruction is used. It has the format  
INT interrupt\_number

Where, **interrupt\_number** is a number that specifies a routine. In the following, we use a particular DOS routine, INT 21h.

**INT 21h**

INT 21h may be used to invoke a large number of DOS functions; a particular function is requested by placing a function number in the AH register and invoking INT 21h. Here we are interested in the following functions:

Function Number	Routine
1	single-key input
2	single-key output
9	character string output

INT 21h functions expect input values to be in certain registers and return output values in other registers. These are listed as we describe each function.

**Function 1:**

**Single-key Input**

Input: AH=1

Output: AL = ASCII code if character key is pressed  
= 0 if non-character key is pressed.

To invoke the routine, execute these instructions:

```
MOV AH, 1 ; input key function
INT 21h ; ASCII code in AL
```

**Function 2:**

**Single-key Output**

Input: AH=2

DL = ASCII code of the display character or control character

Output: AL = ASCII code of the display character or control character

To display a character with this function, we put its ASCII code in DL. For example, the following instructions cause a question mark to appear on the screen:

```
MOV AH, 2 ; display character function
MOV DL, '?' ; character is '?'
INT 21h ; display character
```

After the character is displayed, the cursor advances to the next position on the line. Function 2 may also be used to perform control functions. If DL contains the ASCII codes of a control character, INT 21h causes the control function to be performed. The principle control characters are as follows:

ASCII code	Symbol	Fucntion
A	LF	line feed (new line)
D	CR	carriage return (start of a line)

- **Conditional Control Transfer Instruction**

Conditional jumps transfer control to another address depending on the values of the flags in the flag register. The jump condition often provided by the CMP instruction:

***CMP destination, source***

Condition	Instruction	Condition	Instruction
Jump if zero flag ZF=1	<b>JZ <i>zero</i></b>	Jump if zero flag ZF=0	<b>JNZ <i>notzero</i></b>
Jump if greater	<b>JG <i>greater</i></b>	Jump if greater than or equal	<b>JGE <i>notless</i></b>
Jump if less	<b>JL <i>less</i></b>	Jump if less than or equal	<b>JLE <i>notgreater</i></b>
Jump if Below	<b>JB <i>smaller</i></b>	Jump if carry flag CF=1	<b>JC <i>carry</i></b>

**Assembly Language Program Example:**

```

ORG 0100h
MAIN PROC
; display prompt
MOV AH, 2
MOV DL, '?'
INT 21h
; input a character
MOV AH, 1
INT 21h
MOV BL, AL
; go to a new line with carriage return
MOV AH, 2
MOV DL, 0DH
INT 21h
MOV DL, 0AH
INT 21h
; display character
MOV DL, BL
INT 21h
; return to DOS
MOV AH, 4CH
INT 21H
MAIN ENDP
END MAIN
RET

```

```

org 100h
START:      mov cl, 03h

LABEL_JNZ:  dec cl
            jnz LABEL_JNZ
            mov bl, 04h
            mov al, 04h

LABEL_JZ:   dec al
            dec bl
            xor bl, al
            jz LABEL_JZ
            mov bl, 02h
            mov al, 06h

LABEL_JG:   dec al
            cmp al, bl
            jg LABEL_JG
            mov bl, 06h
            mov al, 00h

LABEL_JL:   inc al
            cmp al, bl
            jl LABEL_JL

ret

```

### **Tasks to do:**

1. Write an assembly language program that inputs a single letter and shows the same letter in it's opposite case in a new line. (Lower-case to Upper-case or vice-versa).

**Sample Input / Output:**

Input:           a  
Output:          A

Input:           Z  
Output:          z

2. Write an assembly language program that inputs a single letter and shows the next 5 (five) letters in opposite case of input (Lower-case to Upper-case or vice-versa) in a row of a new line and also shows the previous 5 (five) letters in the next line in opposite case of input (Lower-case to Upper-case or vice-versa).

**Sample Input / Output:**

Input:           a  
Output:          BCDEF  
                  ZYXWV

Input:           Z  
Output:          abcde  
                  yxwvu