Green University of Bangladesh
Department of Computer Science and Engineering (CSE)

Faculty of Sciences and Engineering

Semester: (Fall, Year:2025), B.Sc. in CSE (Day)

Lab Report NO: 02

Course Title: Microprocessors, Microcontrollers, and Embedded System

Course Code: CSE 304 **Section:** 232-D1

Lab Experiment Name: Implementation of conditional statement using assembly language

Student Details

	Name	ID
1.	Rukonuzzaman Topu	232002280

Submission Date : 29/10/2025

Course Teacher's Name : Jarin Tasnim Tonvi

[For Teachers use only: Don't Write Anything inside this box]

Lab Report Status

Marks: ………………………………… Signature:....................

Comments:... Date:..............................

1. INTRODUCTION

This lab emphasizes applying conditional branching in assembly language to explore how a microprocessor makes decisions using flag registers. In contrast to high-level languages that rely on straightforward if–else statements, assembly programming demands explicit manipulation of status flags and jump instructions (like JE, JNE, JG, etc.) to guide the program's flow. Through these hands-on exercises, students gain a clear understanding of logical comparisons, decision-making, and control transfer at the hardware-instruction level.

2. OBJECTIVES

• To learn the principles of conditional branching and the functioning of flag registers in assembly programming.
• To apply conditional and unconditional jump instructions for creating decision-making structures such as if, if–else, and switch/case.
• To develop optimized and efficient 16-bit assembly programs that demonstrate effective use of branching and control flow.

3. PROCEDURE

Problem-1: Find the Largest Number Between Two Inputs

1. The program begins by displaying a message prompting the user to enter the first number (0–9).
2. It reads the character input, converts it from ASCII to numeric form by subtracting 30H, and stores the result in register BL.
3. Next, it asks for the second number, performs the same conversion, and stores the value in register BH.
4. The two values, BL and BH, are compared using the CMP instruction.
    a. If the values are equal, the program displays "Both numbers are equal."
    b. If the first number is greater, it displays "The largest number is:" followed by the first number.
    c. Otherwise, it shows the second number as the largest.
5. The program outputs text and numbers using DOS interrupts:
    a. INT 21H with AH = 9 for string output,
    b. INT 21H with AH = 2 for single character display.
6. Finally, the program terminates gracefully with INT 21H using AH = 4CH.

Problem-2: Check Whether a Number Is Divisible by 5

Input Phase:

1. The program prompts the user to enter a single-digit number (0–9).
2. The input is read using INT 21H with AH = 1, which captures a single ASCII character.
3. The ASCII value is converted to its numeric equivalent by subtracting 30H and then stored in register AL.

Computation Phase:

1. The constant 5 is loaded into register BL.
2. The program divides AL by BL using the DIV BL instruction.
3. The quotient is stored in AL, while the remainder automatically goes into AH.
4. If AH = 0, it means the number is perfectly divisible by 5; otherwise, it is not.

Decision & Output Phase:

1. The CMP instruction is used to compare AH with 0.
2. If they are equal, the program displays "Divisible by 5."
3. If not equal, it displays "Not divisible by 5."
4. The message is printed to the screen using INT 21H with AH = 9, where DX holds the address of the string.

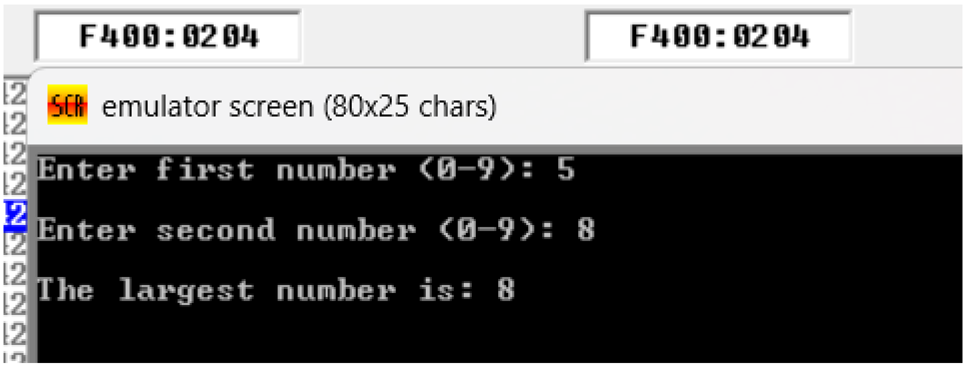
4. IMPLEMENTATION

Source Code: Find the Largest Number Between Two Inputs

```
.MODEL SMALL
.STACK 100H

.DATA
   STR1 DB 'Enter first number (0-9): $'
   STR2 DB 0DH,0AH, 'Enter second number (0-9): $'
   STR3 DB 0DH,0AH, 'The largest number is: $'
   STR4 DB 0DH,0AH, 'Both numbers are equal: $'
   NL   DB 0DH,0AH,'$'

.CODE
MAIN PROC
```

```
    MOV AX, @DATA
    MOV DS, AX
    MOV DX, OFFSET STR1
    MOV AH, 9
    INT 21H

READ_FIRST:
    MOV AH, 1
    INT 21H
    CMP AL, '0'
    JB  READ_FIRST
    CMP AL, '9'
    JA  READ_FIRST
    SUB AL, '0'
    MOV BL, AL


    MOV DX, OFFSET NL
    MOV AH, 9
    INT 21H


    MOV DX, OFFSET STR2
    MOV AH, 9
    INT 21H

READ_SECOND:
    MOV AH, 1
    INT 21H
    CMP AL, '0'
    JB  READ_SECOND
    CMP AL, '9'
    JA  READ_SECOND
    SUB AL, '0'
    MOV BH, AL


    MOV DX, OFFSET NL
    MOV AH, 9
    INT 21H

    CMP BL, BH
    JE  EQUAL_CASE
    JG  FIRST_IS_LARGER

    MOV DX, OFFSET STR3
    MOV AH, 9
    INT 21H
```

```
   MOV DL, BH
   ADD DL, '0'
   MOV AH, 2
   INT 21H
   JMP DONE

FIRST_IS_LARGER:
   MOV DX, OFFSET STR3
   MOV AH, 9
   INT 21H

   MOV DL, BL
   ADD DL, '0'
   MOV AH, 2
   INT 21H
   JMP DONE

EQUAL_CASE:
   MOV DX, OFFSET STR4
   MOV AH, 9
   INT 21H

   MOV DL, BL
   ADD DL, '0'
   MOV AH, 2
   INT 21H

DONE:
   ; final newline
  MOV DX, OFFSET NL
   MOV AH, 9
   INT 21H
   MOV AH, 4CH
   INT 21H
MAIN ENDP
END MAIN
```
Output:



```
Enter first number (0-9): 5
Enter second number (0-9): 8
The largest number is: 8
```

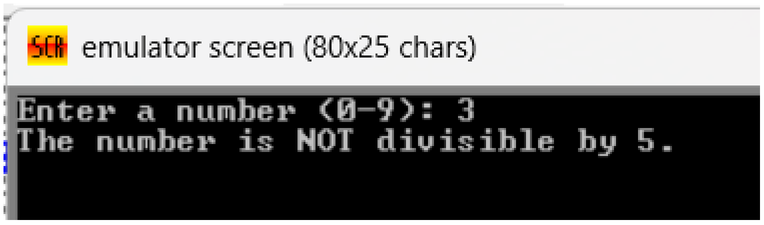
Source Code: Check Whether a Number Is Divisible by 5

```
.MODEL SMALL
.STACK 100H

.DATA
   STR1 DB 'Enter a number (0-9): $'
   STR2 DB 0DH,0AH,'The number is divisible by 5.$'
   STR3 DB 0DH,0AH,'The number is NOT divisible by 5.$'

.CODE
MAIN PROC
   MOV AX, @DATA
   MOV DS, AX

   MOV DX, OFFSET STR1
   MOV AH, 9
   INT 21H

   MOV AH, 1
   INT 21H
   SUB AL, 30H
   MOV AH, 0
   MOV BL, 5
   DIV BL

   CMP AH, 0
   JE DIVISIBLE
   JNE NOT_DIVISIBLE

DIVISIBLE:
   MOV DX, OFFSET STR2
   MOV AH, 9
   INT 21H
   JMP EXIT

NOT_DIVISIBLE:
   MOV DX, OFFSET STR3
   MOV AH, 9
```

```
    INT 21H

EXIT:
    MOV AH, 4CH
    INT 21H

MAIN ENDP
END MAIN
```

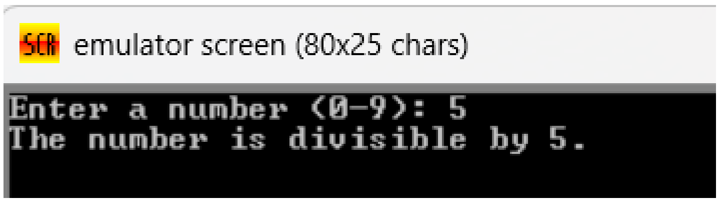
Output:

1.



```
Enter a number (0-9): 3
The number is NOT divisible by 5.
```

2.



```
Enter a number (0-9): 5
The number is divisible by 5.
```

DISCUSSION

This lab covers two fundamental conditional tasks in assembly language: determining the greater of two numbers and verifying divisibility by 5. In both programs, decision-making relies on the status flags and conditional jump instructions of the processor. In the first task, the two inputs are compared using the CMP instruction, followed by conditional jumps such as JAE or JB to identify and display the larger number. In the second task, the DIV instruction divides the input by 5, and the remainder in the AH register is examined—if it is zero, the number is divisible by 5.

Through these exercises, students developed a clearer understanding of low-level decision processes in microprocessors. Manually implementing comparisons and branching helped illustrate how high-level if–else logic is realized through flag-based control. Overall, the lab improved proficiency in assembly coding, arithmetic operations, and logical flow management within the CPU environment.