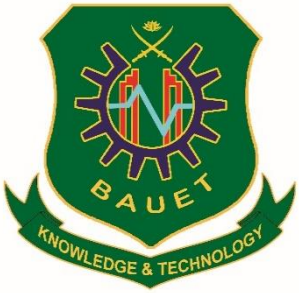
Bangladesh Army University of Engineering & Technology (BAUET)
Qadirabad, Natore-6431




# Department of Computer Science and Engineering (CSE)

# Lab Report: 02

**Course Code:** CSE-3222
**Course Title:** Programming with Assembly language Sessional
**Experiment Name:** Introduction to arithmetic operations assembly language
**Experiment Date:** 10.08.25
**Submission Date:** 24.08.25

**SUBMITTED BY**

Name: Tanvir Akter Sonia

ID: 0812220105101068

Batch: 16th

Year: 3rd

Semester: 2nd

Section: B

**SUBMITTED TO**

MD. Fatin Nbbrash Nakib
Lecturer, Dept. of CSE, BAUET

Tanvir Anjom Siddique
Lecturer, Dept. of CSE, BAUET

**Experiment No:02**

**Experiment Name :** Introduction to arithmetic operations assembly language

**Objectives:**

1. To learn how numbers are calculated in assembly language.
2. To see how the CPU changes when doing arithmetic.
3. To use basic math steps to make bigger tasks.

## Theory:

**Arithmetic Operations**

Arithmetic operations are fundamental in assembly programming, allowing the CPU to perform mathematical calculations. These operations manipulate numerical values stored in registers or memory.

**Addition and Subtraction Instructions**

**Addition (ADD)**

The ADD instruction adds a value (either a constant or the contents of another register) to a destination register or memory location. The result is stored in the destination operand.

Example-

ADD destination, source

**Subtraction (SUB)**

The SUB instruction subtracts a value from the destination and stores the result in the destination operand.

Example-

SUB destination, source

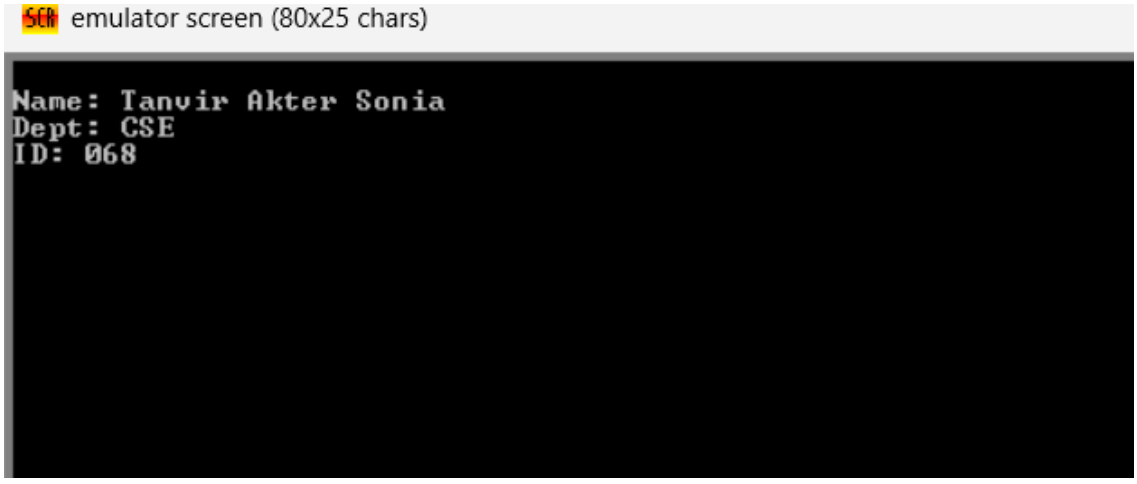
Problem No:1.1

Problem Name :write a assembly language to print Name ,Dept and ID

**Source Code:**

```
.MODEL SMALL
.STACK 100H
.DATA
    MSG_NAME DB 0DH,0AH,"Name: Tanvir Akter Sonia$"
    MSG_DEPT DB 0DH,0AH,"Dept: CSE$"
    MSG_ID   DB 0DH,0AH,"ID: 068$"

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

    ; Print Name
    MOV AH, 09H
    LEA DX, MSG_NAME
    INT 21H

    ; Print Dept
    MOV AH, 09H
    LEA DX, MSG_DEPT
    INT 21H

    ; Print ID
    MOV AH, 09H
    LEA DX, MSG_ID
    INT 21H

    ; Exit program
    MOV AH, 4CH
    INT 21H
MAIN ENDP
END MAIN
```

**Output:**



**Problem No:2.1**

**Problem Name: Write a Assembly Program to Calculate 5 – (A + A) + 3**

**Source Code:**

```
.MODEL SMALL
.STACK 100H
.DATA

    MSG1    DB "ENTER A NUMBER FOR A (0-9):$"
    RESMSG  DB 0DH,0AH,"RESULT = $"

    A       DB ?
    RESULT  DB ?

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

    ; Prompt for A
    MOV AH, 09H
    LEA DX, MSG1
    INT 21H

    ; Read A from user
    MOV AH, 01H
    INT 21H
    SUB AL, 30H         ; convert ASCII to number
    MOV A, AL

    ; Perform: 5 - (A + A) + 3
    MOV AL, A           ; AL = A
    ADD AL, A           ; AL = A + A
    MOV BL, AL          ; BL = (A + A)

    MOV AL, 5           ; AL = 5
    SUB AL, BL          ; AL = 5 - (A + A)
    ADD AL, 3           ; AL = 5 - (A + A) + 3
    MOV RESULT, AL

    ; Show result message
    MOV AH, 09H
    LEA DX, RESMSG
    INT 21H

    ; Show result value
    MOV AL, RESULT
    ADD AL, 30H
    MOV DL, AL
    MOV AH, 02H
    INT 21H

    ; Exit program
    MOV AH, 4CH
    INT 21H
MAIN ENDP
END MAIN
```

**Output:**

```
ENTER A NUMBER FOR A (0-9):2
RESULT = 4
```

**Discussion:**

In this experiment, we learned how arithmetic operations are performed in assembly language and why they are important. Arithmetic operations such as addition, subtraction, multiplication, and division are the basic building blocks of any program. In assembly language, these operations are done using specific CPU instructions that work directly on registers and memory locations. By performing arithmetic step by step, we can clearly see how data moves between registers, how results are stored, and how the processor handles calculations.

Understanding arithmetic operations in assembly language helps us develop logical thinking and gain a deeper knowledge of how computers process numbers at the hardware level. It also improves our ability to write efficient programs for tasks that require high speed and low memory usage. Even though high-level languages can perform these operations more easily, knowing how to do them in assembly provides valuable insight into the inner workings of a computer.