Lab Manual for Computer Organization and Assembly Language

Lab-3

Arithmetic Operations – Part 1

Table of Contents

1. Introduction

Each personal computer has a microprocessor that manages the computer's arithmetical, logical, and control activities.

2. Objective

- To get basic understanding of Assembly Language arithmetic operations.
- To be able to write assembly code to add or subtract values
- To be able to take input from user and compute the values

3. Concept Map

This section provides you the overview of the concepts that will be discussed and implemented in this lab.

3.1 Advantages of Assembly Language

- How programs interface with OS, processor, and BIOS;
- How data is represented in memory and other external devices;
- How the processor accesses and executes instruction;
- How instructions access and process data;
- How a program accesses external device.
- It requires less memory and execution time;
- It allows hardware-specific complex jobs in an easier way;
- It is suitable for time-critical jobs;
- It is most suitable for writing interrupt service routines and other memory resident programs

3.2 Basic Arithmetic Operations

There are four basic arithmetic operations in assembly language. These are: Addition, Subtraction, Division and Multiplication. However, in this lab we will focus on addition and subtraction.

In assembly language we need to know the way to handle on digit number and more than one-digit number. Initially we will practice with one-digit numbers addition and subtraction. For example: following are one-digit numbers al=1, bl=3, al+bl=3.

Below instructions are used to add values of registers:

(add destination, source) or (sub destination, source): it is equivalent to destination = destination + source.

Knowledge of ASCII code is required to do addition and subtraction in assembly language. Like, if we want to add two numbers and their outcome will be showed on console then we must add ASCII code (48, which is the code of digit 0) to display the actual number on screen. Similarly,

when user enters any value in register at runtime, that value is stored but in ASCII code. So, to receive value in code we first need to convert that ASCII code to equivalent decimal value. Therefore, 48 is subtracted from the ASCII number.

The processor supports the following data sizes −

- Word: a 2-byte data item
- Double word: a 4-byte (32 bit) data item
- Quad word: an 8-byte (64 bit) data item
- Paragraph: a 16-byte (128 bit) area
- Kilobyte: 1024 bytes
- Megabyte: 1,048,576 bytes

4. Walk through task

Below code is used to take two single digit numbers from user on runtime and output of their sum is shown on the console. Variables to store user value are declared in data segment and string variables are used to show relevant messages for input and output with new line code.

4.1 Code (add two one-digit numbers)

```
.model small

.data

str1 db "enter first number $"
str2 db 0ah,0dh,"enter second number $"
str3 db 0ah,0dh,"sum of two numbers is $"

.code

;data segment enabling code
mov ax, @data
mov ds, ax


;code to display string 1
lea dx, str1
mov ah, 09
int 21h

;code to take first input value
mov ah, 01
int 21h

;input is stored in al register so copy it
mov bl, al

;code to display string 2
lea dx, str2
mov ah, 09
int 21h
```

```
32
33 ;code to take second input value
34 mov ah, 01
35 int 21h
36
37 ;second input is stored in al register so copy it
38 mov cl, al
39
40 ;add two values
41 add bl, cl
42
43 ;ASCII values are added so subtract 48
44 sub bl, 48
45
46 ;code to display string 3
47 lea dx, str3
48 mov ah, 09
49 int 21h
50
51 ;output must be in dl register
52 mov dl, bl
53
54 ;code to display output value in dl
55 mov ah, 02
56 int 21h
```

4.2 Output of Code

5. Procedure & Tools

In this section, you will study how to setup and emu 8086.

5.1 Tools

- Open emu8086.
- Make new project.
- Enter your code.
- Click on Emulate.
- Click on Run.

6. Practice Tasks

This section will provide more practice exercises which you need to finish during the lab. You need to finish the tasks in the required time. When you finish them, put these tasks in the following folder:
\\fs\assignments$\COAL\Lab4

6.1 Practice Task 1: [Expected time = 15mins]

Add two numbers and show their ASCII number or representation?

(Hint: Do not convert ASCII to decimal by adding 48.)

6.2 Practice Task 2: [Expected time = 15mins]

Add two numbers by initializing their values in data segment. Show the result of addition on the console screen. Don't use any variable to store variable. Do the addition operation on default register al and bl?

6.3 Practice Task 3: [Expected time = 15mins]

Take two numbers from user and subtract their ASCII number. Show the outcome of their subtraction in decimal number. Hint: Do not convert ASCII to decimal by adding 48.

6.4 Practice Task 4: [Expected time = 15mins]

Make three variables and take two values from user at runtime and store them in variable number one and two. Add first two values in variables. Take third value from user and store it in variable number 3. Third value should be of one digit and smaller than the sum of two previous values. Subtract the value in third variable from the addition result of first two variables. Display the output on the screen?

6.5 Out comes

After completing this lab, student will be able to do basic arithmetic addition and subtraction in assembly language programs.

7. Evaluation Task (Unseen): [Expected time = 30mins for tasks]

The lab instructor will give you unseen task depending upon the progress of the class.

8. Evaluation criteria

The evaluation criteria for this lab will be based on the completion of the following tasks. Each task is assigned the marks percentage which will be evaluated by the instructor in the lab whether the student has finished the complete/partial task(s).

Table 3: Evaluation of the Lab

Sr. No.	Task No	Description	Marks
1	1	Problem Modeling	20

2	2	Procedures and Tools	10
3	3	Practice tasks and Testing	35
4	4	Evaluation Tasks (Unseen)	20
5		Comments	5
6		Good Programming Practices	10

9. Further Reading

This section provides the references to further polish your skills.

9.1 Slides

The slides and reading material can be accessed from the folder of the class instructor available at \\fs\lectures$