Laboratory Manual

Microprocessor & Assembly Language (CS-330)

5th Semester

Name: Munib-ul-hassan

Roll No:

Section: 2019-CS-037 "A"

Batch: 2019

Department of Computer Science & Information Technology
Sir Syed University of Engineering & Technology
University Road, Karachi
https://www.ssuet.edu.pk

List of Laboratory Experiments

Lab No.	Date	TITLES	Page No.	Signature
1.	23/2	To familiarize the students with the 8086/88 Assembler, the Central Processing Uit (CPU), and refresh the hexadecimal number systems.	1-5	
2.	23/2	To introduce System commands using DEBUG programming utility (at command prompt using PC).	6-11	
3.	9/3	To introduce Program control commands using DEBUG programming utility (at command prompt using PC).	12-15	
4.	9/3	To learn how to create and assemble an executable Assembly Language Programming using Assembler and Linker utilities.	16-17	
5.		Develop understanding to use different BIOS and DOS Interrupt Services using MASM/TASM utility.	18-19	
6.		Implementation of the Program Flow Control using "JMP" and "LOOP" instructions in Assembly language.	20-24	
7.		Implementation of the basic arithmetic operations using Assembly instructions, such as Half-Adder, Full Adder, Half-Subtractor, Full Subtractor, Multiplier and Divisor.	25-27	
8.		Implementation of Logical instructions using Assembly language.	28-30	
9.		Implementation of ASCII and BCD arithmetic instructions with the help of Logical operation using 8086/88 Assembly language.	31-33	
10.		Implementation of procedures using CALL instruction in Assembly language.	34-35	
11.		To perform computation by using Assembly Language Programming. • To generate Fibonacci series • To generate Factorial of a number	36-37	
12.		To implement the string manipulation instructions provided in the 8086/88 Assembly language.	38-39	

c) **Conversion from decimal to hexadecimal**

- Divide the given number by sixteen, keeping track of the remainder
- First remainder is bit 0 with the weight 1 (LSB, least-Significant Bit)
- Second remainder is bit 1
- Do the same till the last digit
-

d) **Conversion from binary to decimal**

- Multiply each bit by 2^n, the "weight" of the bit
- n is the sequence number of bits from LSB which is the right most bit
- Add the results

e) **Conversion from binary to octal**

- Group bits in threes, from right to left
- Convert to octal digits

f) **Conversion from binary to hexadecimal**

- Group bits in four, from right to left
- Convert to hexadecimal digits

g) **Conversion from octal hexadecimal**

- Use binary as an intermediary

h) **Conversion from hexadecimal to octal**

- Use binary as an intermediary

A Quick Example

Following is a quick example of such conversions.

$$25_{10} = \overset{1}{\underset{3}{1}}\overset{9}{\underset{1}{1001}}_2 = 31_8 = 19_{16}$$

V. Activity:

Convert and fill the table give below. Use your roll number to fill the last (blank) row:

Decimal	Binary	Octal	Hexadecimal
33	100001	41	21
117	1110101	165	75
451 ~~411000011~~	111000011	703	1C3
431	110101111	657	1AF

09/08/21

LAB 2

- Display the memory contents starting from memory location 0100h

```
Z:\>C:

C:\>debug
-d 0100
073F:0100  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  ................
073F:0110  00 00 8F E9 00 F0 87 74-B2 00 8C 00 2E 07 2E 07  .......t........
073F:0120  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  ................
073F:0130  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  ................
073F:0140  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  ................
073F:0150  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  ................
073F:0160  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  ................
073F:0170  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  ................
-
```

Display the memory contents of first 10 bytes starting from memory location 0100h

```
-d 0100 0110
073F:0100  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  ................
073F:0110  00                                               .
-
```

- Fill the memory location starting from 0200h with your name and 3-digit Roll #

```
-F 0200 "Munib ul hassan 037"
-d 0200
073F:0200  4D 75 6E 69 62 20 75 6C-20 68 61 73 73 61 6E 20  Munib ul hassan
073F:0210  30 33 37 4D 75 6E 69 62-20 75 6C 20 68 61 73 73  037Munib ul hass
073F:0220  61 6E 20 30 33 37 4D 75-6E 69 62 20 75 6C 20 68  an 037Munib ul h
073F:0230  61 73 73 61 6E 20 30 33-37 4D 75 6E 69 62 20 75  assan 037Munib u
073F:0240  6C 20 68 61 73 73 61 6E-20 30 33 37 4D 75 6E 69  l hassan 037Muni
073F:0250  62 20 75 6C 20 68 61 73-73 61 6E 20 30 33 37 4D  b ul hassan 037M
073F:0260  75 6E 69 62 20 75 6C 20-68 61 73 73 61 6E 20 30  unib ul hassan 0
073F:0270  33 37 4D 75 6E 69 62 20-75 6C 20 68 61 73 73 61  37Munib ul hassa
```

- Note the last offset (memory) value of the last digit of your roll #

- Move the block of memory where you filled your name and Roll # to the memory location starting from 0400h

```
-M 0200 0230 0400
-d 0400
073F:0400  4D 75 6E 69 62 20 75 6C-20 68 61 73 73 61 6E 20  Munib ul hassan
073F:0410  30 33 37 4D 75 6E 69 62-20 75 6C 20 68 61 73 73  037Munib ul hass
073F:0420  61 6E 20 30 33 37 4D 75-6E 69 62 20 75 6C 20 68  an 037Munib ul h
073F:0430  61 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  a...............
073F:0440  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  ................
073F:0450  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  ................
073F:0460  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  ................
073F:0470  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  ................
```

Compare the two memory

blocks (one starting from 0100h & the other from 0400h)

```
-c 0100 0120 0400
073F:011C  34  00  073F:011C
073F:011E  2E  00  073F:011E
073F:011F  07  00  073F:011F
-
```

- Edit 2018-CS- to the memory block starting from 0400h before your Roll #

```
-E 0411 "CS-19-"
-D 0400
073F:0400  4D 75 6E 69 62 20 75 6C-20 68 61 73 73 61 6E 20   Munib ul hassan
073F:0410  30 43 53 2D 31 39 2D 62-20 75 6C 20 68 61 73 73   0CS-19-b ul hass
073F:0420  61 6E 20 30 33 37 4D 75-6E 69 62 20 75 6C 20 68   an 037Munib ul h
073F:0430  61 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   a...............
073F:0440  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
073F:0450  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
073F:0460  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
073F:0470  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
-
```

- Compare the two memory blocks (one starting from 0100h & the other from 0400h)

```
-C 200 230 400
073F:0211  33  43  073F:0411
073F:0212  37  53  073F:0412
073F:0213  4D  2D  073F:0413
073F:0214  75  31  073F:0414
073F:0215  6E  39  073F:0415
073F:0216  69  2D  073F:0416
```

- Calculate the sum and difference of 75 and 34 using H command

```
-H 75 34
00A9  0041
```

Activity- 2:

Assemble and Unassemble the following code and fill the following table .

Logical Address	Opcode	Assembly code	Comments
0735;0100	B80002	MOV AX, 0200H	; Store "0200" in AX
0735:0103	BB0004	MOV BX, 0400H	; Store "0400" in bx register
0735:0106	01DB	ADD AX, BX	; store the sum value of AX and BX in AX register

Activity- 3:

Give answers to the following questions after unassemble the code:

*Why the values **0200H** and **0400H** are written as **0002** and **0004**, in the opcode, respectively?*

Write the number of bytes and the value of IP register taken by each instruction.

Ans: The value of 0200 and 0400 are written in 0002 and 0004 because of the byte system (Little Endian) in which the least significant byte (the "little end") of the data is placed at the byte with the lowest address.

Instructions	no of bytes	IP address
		0100
MOV AX 0200H	3	0103
MOVE BX 400H	3	0106
~~MOVE~~ ~~AD~~ AX,BX ADD	3	

LAB 03

TASK- 1:

Assemble a program using DEBUG programming utility to move the decimal values in registers as given below AX = 5431_{10} BX = 9321_{10} CX = 45032_{10} DX = 23102_{10}

```
DOSBox 0.74-3, Cpu speed:   3000 cycles, Frameskip 0, Program:   DEBUG

 HAVE FUN!
 The DOSBox Team http://www.dosbox.com

Z:\>SET BLASTER=A220 I7 D1 H5 T6

Z:\>MOUNT C D:/masm615
Drive C is mounted as local directory D:/masm615\

Z:\>C:

C:\>debug
-a100
073F:0100 MOV AX, 34E5
073F:0103 MOV BX,1256
073F:0106 MOV CX, A154
073F:0109 MOV DX, 2316
073F:010C INT 4
073F:010E
-G=100 109

AX=34E5  BX=1256  CX=A154  DX=0000  SP=00FD  BP=0000  SI=0000  DI=0000
DS=073F  ES=073F  SS=073F  CS=073F  IP=0109   NV UP EI PL NZ NA PO NC
073F:0109 BA1623        MOV     DX,2316
-
```

TASK- 2:

Apply the program control command **T** to execute the code. Capture the screen. Write down and analyze the values of each registers including IP and Flag registers.

```
073F:0109 BA1623          MOV     DX,2316
-T 4

AX=34E5  BX=1256  CX=A154  DX=2316  SP=00FD  BP=0000  SI=0000  DI=0000
DS=073F  ES=073F  SS=073F  CS=073F  IP=010C   NV UP EI PL NZ NA PO NC
073F:010C CD04            INT     04

AX=34E5  BX=1256  CX=A154  DX=2316  SP=00F7  BP=0000  SI=0000  DI=0000
DS=073F  ES=073F  SS=073F  CS=0070  IP=0008   NV UP DI PL NZ NA PO NC
0070:0008 FE38            ???     [BX+SI]                          DS:1256=00

AX=34E5  BX=1256  CX=A154  DX=2316  SP=00F7  BP=0000  SI=0000  DI=0000
DS=073F  ES=073F  SS=073F  CS=0070  IP=000C   NV UP DI PL NZ NA PO NC
0070:000C CF              IRET

AX=34E5  BX=1256  CX=A154  DX=2316  SP=00FD  BP=0000  SI=0000  DI=0000
DS=073F  ES=073F  SS=073F  CS=073F  IP=010E   NV UP EI PL NZ NA PO NC
073F:010E AE              SCASB
-
```

TASK- 3:

Apply the program control command **G** to execute the code. Capture the screen. Write down and analyze the values of each registers including IP and Flag registers.

```
-G 100 100

AX=0000  BX=0000  CX=0000  DX=0000  SP=00FD  BP=0000  SI=0000  DI=0000
DS=073F  ES=073F  SS=073F  CS=073F  IP=0100   NV UP EI PL NZ NA PO NC
073F:0100 CC              INT     3
-G 103 103

AX=0000  BX=0000  CX=0000  DX=0000  SP=00FD  BP=0000  SI=0000  DI=0000
DS=073F  ES=073F  SS=073F  CS=073F  IP=0100   NV UP EI PL NZ NA PO NC
073F:0100 CC              INT     3
-G106 106

AX=0000  BX=0000  CX=0000  DX=0000  SP=00FD  BP=0000  SI=0000  DI=0000
DS=073F  ES=073F  SS=073F  CS=073F  IP=0100   NV UP EI PL NZ NA PO NC
073F:0100 CC              INT     3
-G 109 109

AX=0000  BX=0000  CX=0000  DX=0000  SP=00FD  BP=0000  SI=0000  DI=0000
DS=073F  ES=073F  SS=073F  CS=073F  IP=0100   NV UP EI PL NZ NA PO NC
073F:0100 CC              INT     3
-
```

13/04/21

LAB 04

TASK- 1: Write, run and analyze a program that adds 5 bytes of data (given below). Use 5 different byte variables to store the data. Save the result in a byte variable named RESULT. **25h, 12h, 15h, 1Fh, 2Bh**

```
Title lab 4 activity 1
.model small
.stack 100h
.data

A1 DB 25H
A2 DB 12H
A3 DB 15H
A4 DB 1FH
A5 DB 2BH

RESULT DB ?

.code

MAIN PROC

; initialize DS
MOV AX,@DATA
MOV DS,AX
;add the numbers

mov al,A1
mov al,A2
mov al,A3
mov al,A4
mov al,A5

mov RESULT,al

mov ah,4Ch
int 21h

MAIN ENDP

END MAIN
```

Munib

CS19-037

```
Z:\>mount C d:/masm615
Drive C is mounted as local directory d:/masm615\

Z:\>C:

C:\>degug
Illegal command: degug.

C:\>debugger
Illegal command: debugger.

C:\>Debug
-q

C:\>cd BIN

C:\BIN>ls
Illegal command: ls.

C:\BIN>debug lab4a1.eze
```

```
C:\Windows\System32\cmd.exe
(c) 2020 Microsoft Corporation. All rights reserved.

D:\>cd masm615

D:\masm615>cd Bin

D:\masm615\BIN>masm Lab4a1.asm
Microsoft (R) MASM Compatibility Driver
Copyright (C) Microsoft Corp 1993.  All rights reserved.

 Invoking: ML.EXE /I. /Zm /c /Ta Lab4a1.asm

Microsoft (R) Macro Assembler Version 6.14.8444
Copyright (C) Microsoft Corp 1981-1997.  All rights reserved.

 Assembling: Lab4a1.asm
MASM : fatal error A1000: cannot open file : Lab4a1.asm

D:\masm615\BIN>link lab4a1.obg

Microsoft (R) Segmented Executable Linker  Version 5.60.339 Dec  5 1994
Copyright (C) Microsoft Corp 1984-1993.  All rights reserved.

Run File [lab4a1.exe]:
List File [nul.map]:
Libraries [.lib]:
Definitions File [nul.def]:
```

TASK- 2: Write, run and analyze a program that adds 5 bytes of data (given below). Use an array of 6 bytes to store the given 5 bytes of data and save the result in the last byte of array. **25h, 12h, 15h, 1Fh, 2Bh**

LAB4a2 - Notepad

File Edit Format View Help

```
Title lab 4 activity 2
.model small
.stack 100h
.data

Array1 DB 25H,12H,15H,1FH,2BH, ?

.CODE
MAIN PROC

MOV AX,@DATA
MOV DS,AX

MOV SI, offset Array1
MOV AL, [SI]
ADD AL, [SI+1]
ADD AL, [SI+2]
ADD AL, [SI+3]
ADD AL, [SI+4]
MOV [SI+5], AL

MOV AX,4C00H
INT 21H

MAIN ENDP
END MAIN
```

```
C:\>cd masm615

C:\masm615>cd bin

C:\masm615\BIN>masm LAB4a2.asm
Microsoft (R) MASM Compatibility Driver
Copyright (C) Microsoft Corp 1993.  All rights

 Invoking: ML.EXE /I. /Zm /c /Ta LAB4a2.asm

Microsoft (R) Macro Assembler Version 6.14.8444
Copyright (C) Microsoft Corp 1981-1997.  All ri

 Assembling: LAB4a2.asm

C:\masm615\BIN>link LAB4a2.obj

Microsoft (R) Segmented Executable Linker  Versi
Copyright (C) Microsoft Corp 1984-1993.  All rig

Run File [LAB4a2.exe]:
List File [nul.map]:
Libraries [.lib]:
Definitions File [nul.def]:

C:\masm615\BIN>
```

```
HAVE FUN!
The DOSBox Team http://www.dosl

Z:\>SET BLASTER=A220 I7 D1 H5 T6

Z:\>mount c c:\masm615
Drive C is mounted as local direc

Z:\>c:

C:\>cd bin

C:\BIN>LAB4a2

C:\BIN>debug LAB4a2.exe
```

Munib

CS19-037

13/04/21