



# Capital University of Science and Technology

Department of Computer Science

## CS2523 – Computer Organization and Assembly Language

### ASSIGNMENT NO. 5: Effective Address Calculation, Jumps/Branching, Relative Addressing, Bit Manipulation

**CLO: 1.** Define concepts in the design of microprocessor as state machine and designing its data path and its controller. [C1- Remembering]

**CLO: 3.** Implement assembly programs of intermediate complexity using the intel 8088 architecture. The student should also be able to convert intermediate complexity program in high level language into assembly code. [C3- Applying]

**Semester:** Fall 22

**Max Marks:** 10

**Instructor:** Ms. Tayyaba Zaheer

**Assigned Date:** December 28, 2022

**Due Date:** January 04, 2023

**Name:**

**Reg. No.**

#### Guidelines:

You are required to submit the **screenshots of code and output of the program (where required) and concepts in your own words i.e. must be hand written** in the assignment file (word or pdf – pictures attached must be readable and in portrait mode) as **courseCode\_studentReg#\_studentName** via Microsoft Teams.

#### Important Note:

- 1) Must not copy from other students, so do it all yourself.
- 2) **Assignment** should be **hand written**.

#### Description:

Emu8086 is an 8086-microprocessor emulator and disassembler. Emu8086 permits to assemble, emulate and debug 8086 programs (16bit/DOS).

**Tasks: [Hint: you can take help from lectures]**

**Task#1:** Effective Address Calculation:

**(03 marks)**

**Question:** What is the effective address generated by each of the following instruction?

Initially BX=0x0100, label=0x0234, [label]=0x0010, and SI=0x00E1

(Offset in part a is in decimal)

- a) `mov ax, [bx+40]`
- b) `mov ax, [bx+label]`

c) mov ax, [bx+si]

**Task#2:** Conditional, Unconditional Jumps and Relative Addressing:

**(04 marks)**

**Question:** Analyze the given Relative Address/Jump Address and explain the reason behind the value 0110 i.e. JNZ 0110 [You can consult the helping material provided with the assignment i.e. “conditionalJumpsCodes” and “RelativeAddressing”]

```
DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program: AFD
AX 0000 SI 0000 CS 19F5 IP 011C Stack +0 0000 Flags 7204
BX 001A DI 0000 DS 19F5 +2 20CD
CX 0003 BP 0000 ES 19F5 HS 19F5 +4 9FFF OF DF IF SF ZF AF PF CF
DX 0002 SP FFFE SS 19F5 FS 19F5 +6 EA00 0 0 1 0 0 0 1 0

CMD >

011A FEC9 DEC CL
011C 75F2 JNZ
011E B8004C MOV AX,4C00
0121 CD21 INT 21
0123 0000 ADD [BX+SI],AL
0125 0000 ADD [BX+SI],AL
0127 0000 ADD [BX+SI],AL
0129 0000 ADD [BX+SI],AL
012B 0000 ADD [BX+SI],AL

1 0 1 2 3 4 5 6 7
DS:0000 CD 20 FF 9F 00 EA F0 FE
DS:0008 AD DE 1B 05 C5 06 00 00
DS:0010 18 01 10 01 18 01 92 01
DS:0018 01 01 01 00 02 FF FF FF
DS:0020 FF FF FF FF FF FF FF FF
DS:0028 FF FF FF FF EB 19 C0 11
DS:0030 A2 01 14 00 18 00 F5 19
DS:0038 FF FF FF FF 00 00 00 00
DS:0040 05 00 00 00 00 00 00 00
DS:0048 00 00 00 00 00 00 00 00

2 0 1 2 3 4 5 6 7 8 9 A B C D E F
DS:0000 CD 20 FF 9F 00 EA F0 FE AD DE 1B 05 C5 06 00 00 = f.Ω≡ i |.+. ...
DS:0010 18 01 10 01 18 01 92 01 01 01 01 00 02 FF FF FF .....f. ....
DS:0020 FF FF FF FF FF FF FF FF FF FF FF FF EB 19 C0 11 δ. L.
DS:0030 A2 01 14 00 18 00 F5 19 FF FF FF FF 00 00 00 00 6.....J. ....
DS:0040 05 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....

1 Step 2ProcStep 3Retrieve 4Help ON 5BRK Menu 6 7 up 8 dn 9 le 10 ri
```

DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program: AFD

AX 0000 SI 0000 CS 19F5 IP 011C Stack +0 0000 Flags 7204  
 BX 001A DI 0000 DS 19F5 +2 20CD  
 CX 0003 BP 0000 ES 19F5 HS 19F5 +4 9FFF OF DF IF SF ZF AF PF CF  
 DX 0002 SP FFFE SS 19F5 FS 19F5 +6 EA00 0 0 1 0 0 0 1 0

CMD >

011A	FEC9	DEC	CL
011C	75F2	JNZ	0110
011E	B8004C	MOV	AX,4C00
0121	CD21	INT	21
0123	0000	ADD	[BX+SI],AL
0125	0000	ADD	[BX+SI],AL
0127	0000	ADD	[BX+SI],AL
0129	0000	ADD	[BX+SI],AL
012B	0000	ADD	[BX+SI],AL

DS:0000 CD 20 FF 9F 00 EA F0 FE  
 DS:0008 AD DE 1B 05 C5 06 00 00  
 DS:0010 18 01 10 01 18 01 92 01  
 DS:0018 01 01 01 00 02 FF FF FF  
 DS:0020 FF FF FF FF FF FF FF FF  
 DS:0028 FF FF FF FF EB 19 C0 11  
 DS:0030 A2 01 14 00 18 00 F5 19  
 DS:0038 FF FF FF FF 00 00 00 00  
 DS:0040 05 00 00 00 00 00 00 00  
 DS:0048 00 00 00 00 00 00 00 00

2 0 1 2 3 4 5 6 7 8 9 A B C D E F  
 DS:0000 CD 20 FF 9F 00 EA F0 FE AD DE 1B 05 C5 06 00 00  
 DS:0010 18 01 10 01 18 01 92 01 01 01 01 00 02 FF FF FF  
 DS:0020 FF FF FF FF FF FF FF FF FF FF FF FF EB 19 C0 11  
 DS:0030 A2 01 14 00 18 00 F5 19 FF FF FF FF 00 00 00 00  
 DS:0040 05 00 00 00 00 00 00 00 00 00 00 00 00 00 00

1 Step 2ProcStep 3Retrieve 4Help ON 5BRK Menu 6 7 up 8 dn 9 le 10 ri

**Task#3: Bit Manipulation: (03 marks)**

**Question 1:** Suppose AL contains 10011011b and CF= 0. Give the new contents of AL after each of the following instructions is executed. Assume the preceding initial condition for each part of this Question is AL contains 10011011b and CF= 0.

- a) SHL AL,1
- b) SHR AL, CL if CL contains 3
- c) ROL AL ,1
- d) SAR AL, CL if CL contains 3
- e) RCR AL,CL if CL contains 2

**Question 2:** Need to turn on Bit 4 and Bit 7 of a byte (remember that the bit on the right-hand side is Bit 0). Define the mask and logical operator accordingly and show your result.

**Solution:**

7	6	5	4	3	2	1	0	Bit Position
0	0	0	0	0	0	0	0	Data
								Mask
1	0	0	1	0	0	0	0	Logical Operator – ???