Computer Organization and Architecture (EET2211)

LAB II: Analyze and Evaluate the Branching operation in the 8086 Microprocessor.

Siksha 'O' Anusandhan (Deemed to be University), Bhubaneswar

Branch: CSE		**Section:** 44	
S. No.	**Name**	**Registration No.**	**Signature**
27	E. Jagadeeswar Patro	2241016309	E. Jagadeeswar Patro

Marks: _____/10

Remarks:

Teacher's Signature

I. OBJECTIVE:

1. Find the sum and average of N 16-bit numbers.
2. Count no. of 0's in an 8-bit number.
3. Move a block of 16-bit data from one location to other.
4. Multiplication of two 16-bit numbers without using MUL instruction in direct addressing mode.

II. PRE-LAB

Note: For each objective in prelab describe the following points:

- Write the pseudocode.
- Write the assembly code with description (ex. Mov ax,3000h – ax<-3000h)
- Examine & analyze the input/output of assembly code.

Objective 1:-

pseudocode:

```
set SI to 2000h
load the byte at address SI into CL
Set CH to 00H
Copy CX into BX
Set AX to 0000H
Loop1: Increment SI
       Increment SI
       Add the data at address SI to AX
       Jump if no carry to Loop2
       Increment CH
Loop2: Decrement CL
       Jump if not zero to Loop1
       Increment SI
       Increment SI
       Store AX at address SI
       Increment SI
       Increment SI
       Store CH at address SI
       Move CH into DL
       Divide DX:AX by BX
       Increment SI
       Increment SI
       Store AX at address SI
```

Increment SI
Increment SI
Store DX at address SI
Halt

Assembly Code :

```
mov ax, 0000h
mov ds, ax
mov SI, 2000h
mov cl, [SI]
mov ch, 00h
mov bx, cx
mov ax, 0000h
loop1: inc SI
       inc SI
       add ax, [SI]
       jnc loop2
       inc ch
loop2: dec cx
       jnz loop1
       inc SI
       inc SI
       mov [SI], ax
       inc SI
       inc SI
       mov [SI], ch
       mov dl, ch
       div bx
       inc SI
       inc SI
       mov [SI], ax
       inc SI
       inc SI
       mov [S2], dx
hlt
```

Objective 2 : Count no of 0's in an 8-bit number

pseudocode :

input 8-bit data at [2000H]
store it at al
cl = 08h
loop 1 : shift AL right by 1 bit
Jump if carry to Loop2
Increment CH
loop 2 : Decrement CL
Jump if not zero to loop 1
store value of ch at [2001h]
hlt

Assembly Code :

```
mov ax, 0000h
mov ds, ax
mov al, [2000h]
mov cl, 08h
loop 1 : shr al, 01
         jc loop2
         inc ch
loop 2 : dec cl
         jnz loop 1
mov [2001h], ch
hlt
```

Objective 3: Move a block of 16-bit data from one location to another.

pseudocode:

```
set SI to 2000h for input
set DI to 2010h for output
set CL to 05h ; count value
Loop: move the data at address SI to BX
      store BX at address DI
      Increment SI 2 times
      Increment DI 2 times
      Decrement CL
      jump if not zero to 'Loop'
Halt
```

Assembly Code:

```
mov ax, 0000h
mov ds, ax
mov SI, 2000h
mov DI, 2010h
mov cl, 05
loop: mov bx,[SI]
      mov [DI], bx
      inc SI
      inc SI
      inc DI
      inc DI
      dec cl
      jnz loop
hlt
```

Objective 4: Multiplication of two 16-bit numbers without using MUL instruction in direct addressing mode.

pseudocode -

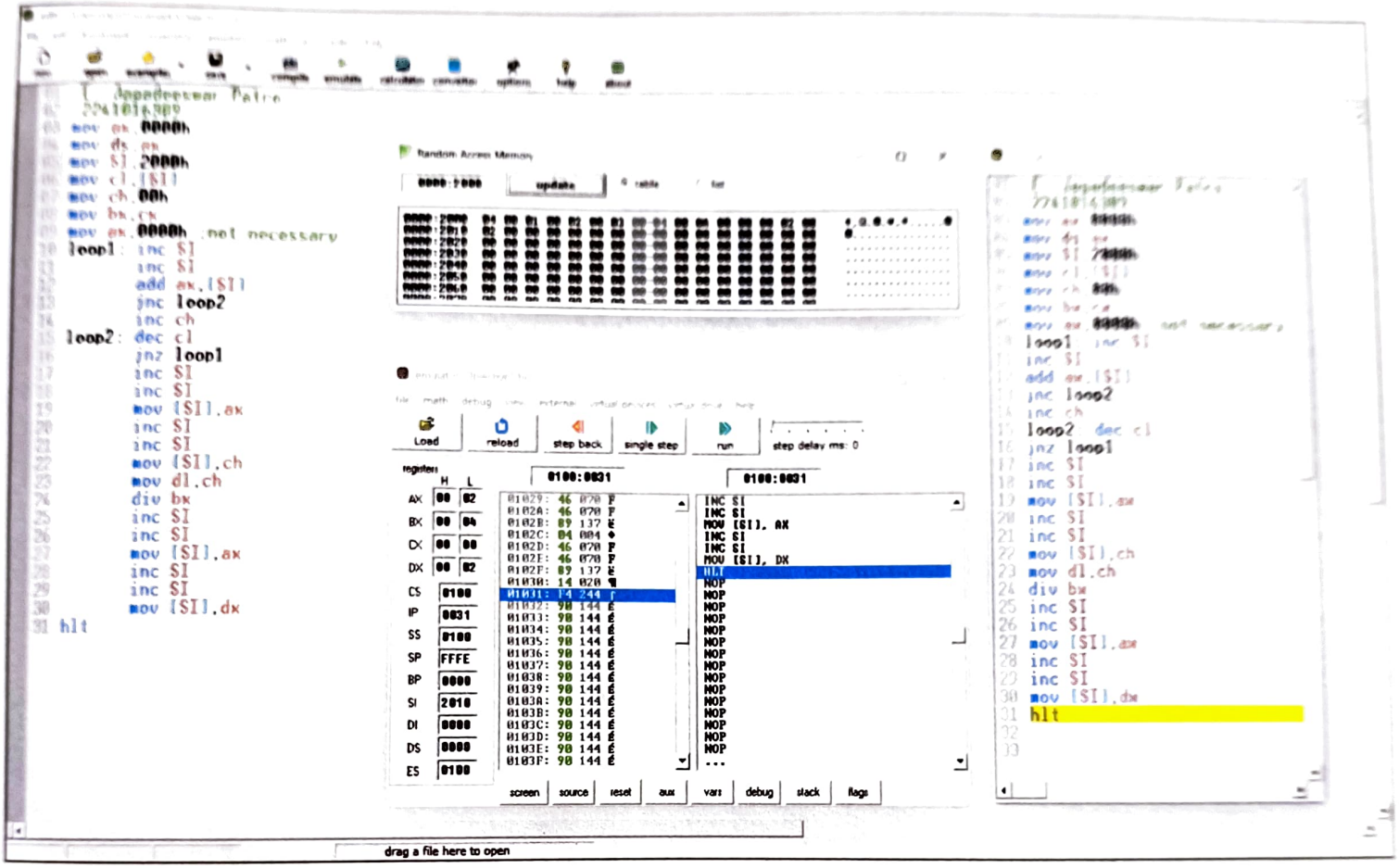
move data at address 2000H to BX
move data at address 2002H to CX
Set DX to 0000H ; count is initialized to zero
Loop 1:
    AX + BX stored at AX
    Jump if no carry to Loop 2
    Increment dx
Loop 2:
    Decrement CX
    Jump if not zero to Loop 1
    store AX at address 2010h
    store DX at address 2012h
Halt

Assembly Code :-

```
mov ax, 0000h
mov ds, ax
mov bx, [2000h]
mov cx, [2002h]
mov dx, 0000h
loop1: add ax, bx
       jnc loop2
       inc dx
loop2: dec cx
       jnz loop1
       mov [2010h], ax
       mov [2012h], dx
hlt
```

III. LAB

Objective 1:



From this result, I have observed.....

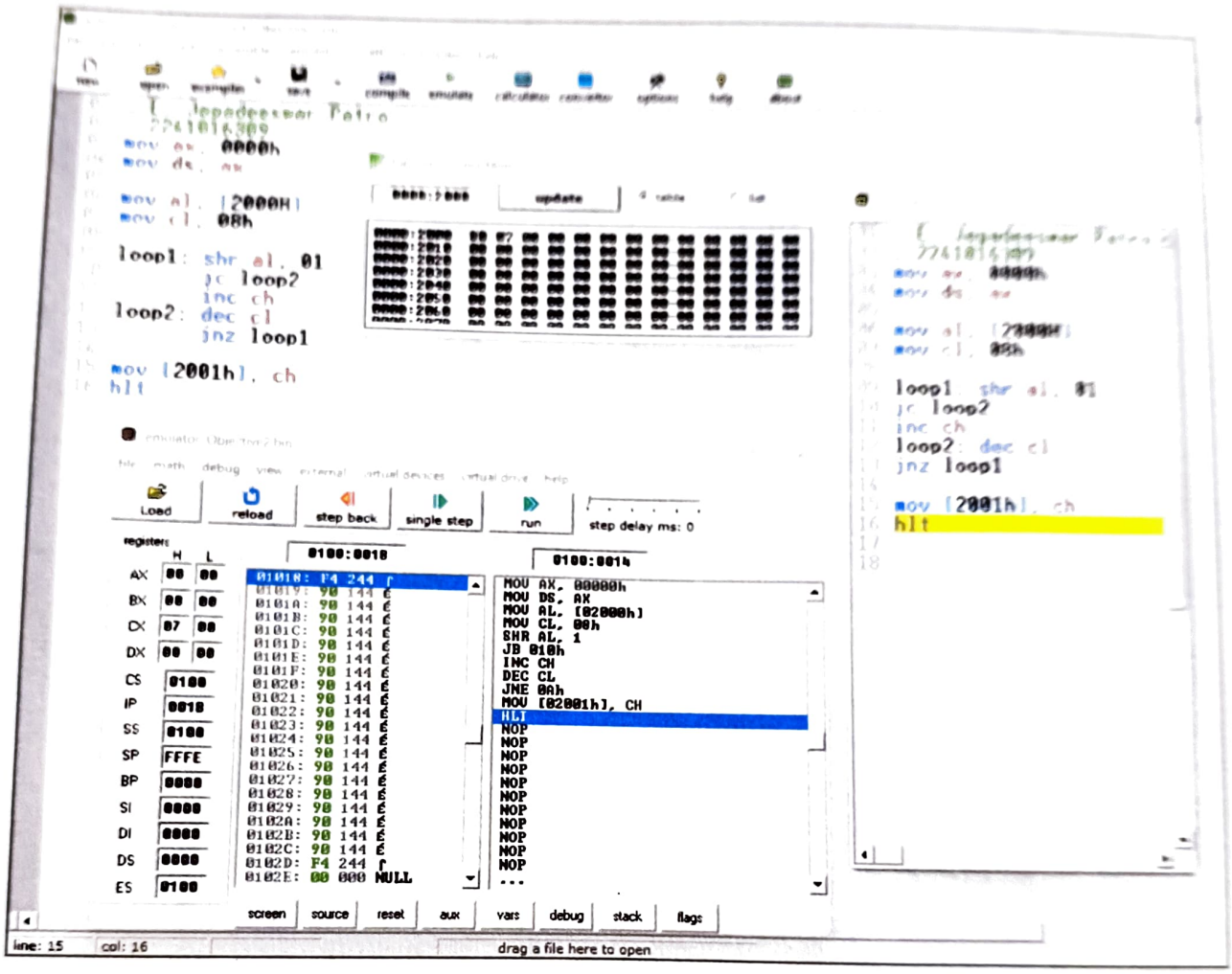
Input:

Sl. No.	Memory Location	Operand (Data)
1	0000 : 2000	04
2	0000 : 2002	01
3	0000 : 2004	02
4	0000 : 2006	03
5	0000 : 2008	04

Output:

Sl. No.	Memory Location	Operand (Data)
1	0000 : 200A	0A
2	0000 : 200E	02
3	0000 : 2010	02

Objective 2:



From this result, I have observed.....

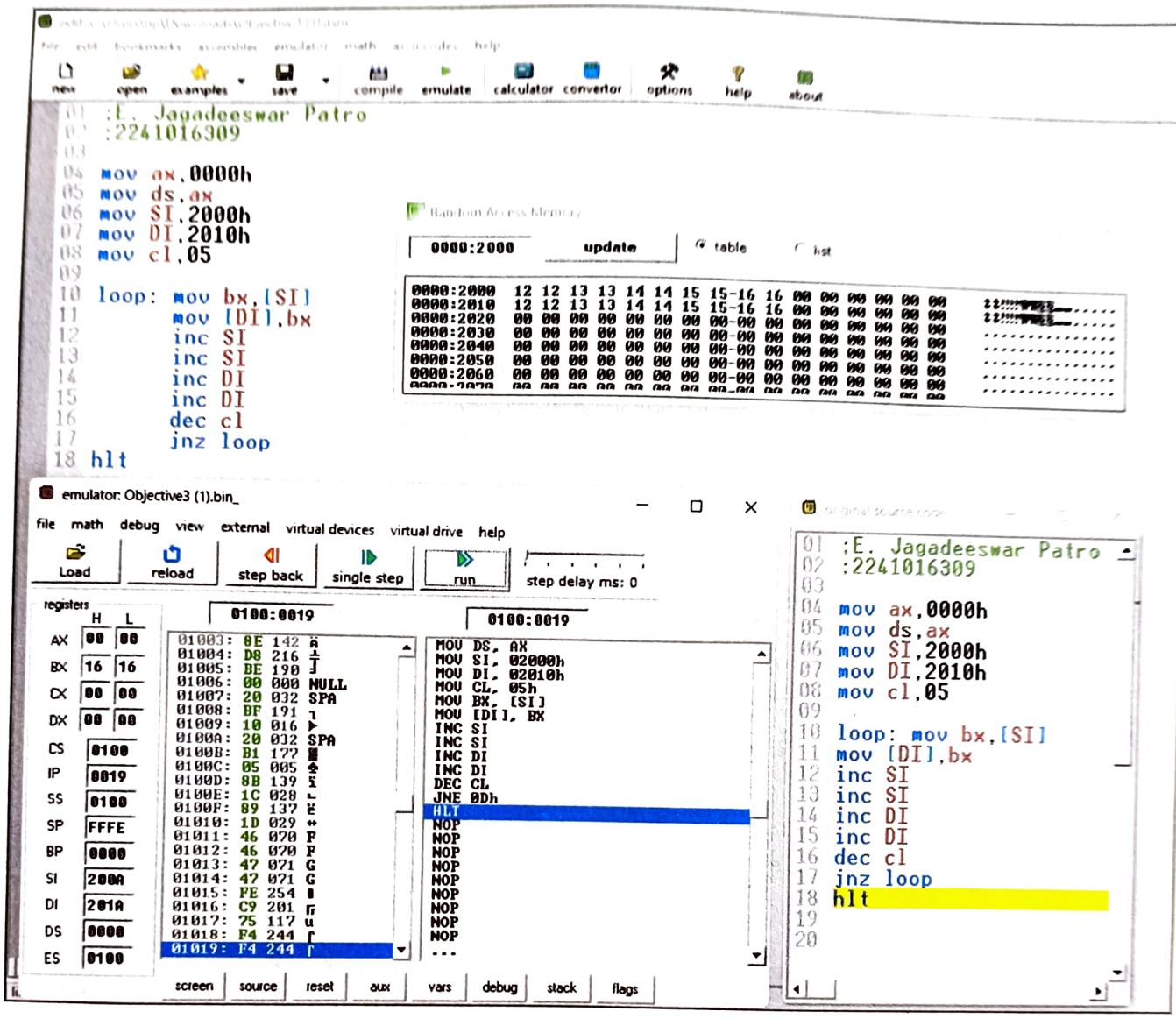
Input:

Sl. No.	Memory Location	Operand (Data)
1	0000 : 2000	80

Output:

Sl. No.	Memory Location	Operand (Data)
1	0000 : 2001	07

Objective 3:



From this result, I have observed.....

Input:

Sl. No.	Memory Location	Operand (Data)
1	0000:2000	1212
2	0000:2002	1313
3	0000:2004	1414
4	0000:2006	1515
5	0000:2008	1616

Output:

Sl. No.	Memory Location	Operand (Data)
1	0000:2010	1212
2	0000:2012	1313
3	0000:2014	1414
4	0000:2016	1515
5	0000:2018	1616

Objective 4:

From this result, I have observed.....

Input:

Sl. No.	Memory Location	Operand (Data)
1	0000 : 2000	08
2	0000 : 2002	03

Output:

Sl. No.	Memory Location	Operand (Data)
1	0000 : 2010	18
2	0000 : 2012	00

IV. CONCLUSION

1) In this experiment we learn how to perform various operations in the x86 assembly language, such as use of loops to perform a set of instructions repeating by checking the flags.

V. POST LAB

1. Analyze the following code and find out the value of registers.

MOV AX, 4246H → AX = 4246H
MOV BX, 123FH → BX = 123FH
AND AX, BX ——→ AX AND BX = 022H
ADD AX, BX ——→ AX + BX = (022H) + (123FH) = (1261H)
ROR AX, 02H —→ AX >> 2 = 4904H
INC BX ——————→ 123F + 1 = 1240H
INC BX ——————→ 1240 + 1 = 1241H
MOV [BX], AX → Store at address of BX
HLT ——————→ Stop program

After Executing; AX = 4904H
BX = 1241H

2. Division of two 16-bit numbers without using DIV instruction in direct addressing mode.

```
MOV AX, [3000H]
MOV BX, [3002H]
XOR CX, CX
L1: CMP AX, BX
    JB L2
    SUB AX, BX
    INC CX
    JMP L1
L2: MOV [2000H], CX
    MOV [2002H], AX

HLT
```