Computer Organization and Architecture (EET2211)

LAB III: Evaluate Different Logical operations on two 16 bit Data

Siksha 'O' Anusandhan Deemed to be University, Bhubaneswar

Branch: Computer Science and Engineering			**Section:** 'D'
S. No.	**Name**	**Registration No.**	**Signature**
52	Saswat Mohanty	1941012407	Saswat Mohanty

Marks: ______/10

Remarks:

Teacher's Signature

I. OBJECTIVE:

1. AND two 16 bit numbers using direct addressing mode.
2. OR two 16 bit numbers using direct addressing mode.
3. NOT of a 16 bit number using direct addressing mode.
4. XOR of two 16 bit numbers using direct addressing mode.

II. PRE-LAB

For Obj. 1:

a) **Explain direct addressing mode briefly.**

It is the addressing mode in which the effective address of the memory location is written directly in the instruction.

b) **Examine & analyze the output obtained from AND of two 16 bit numbers.**

mov ax,[1000h]

mov bx,[1002h]

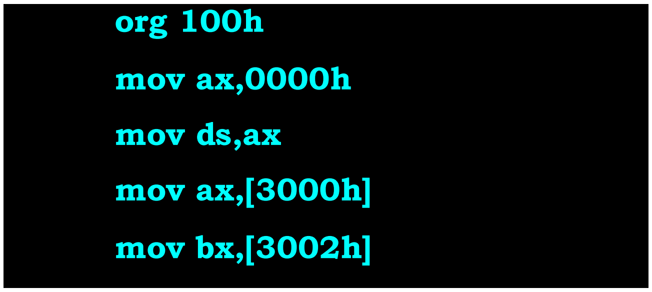
and ax,bx

[1000h] = 1234h

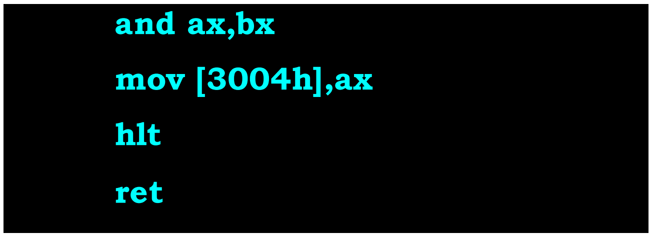
[1002h] = 4321h

Output: 220h

c) **Write the assembly code.**



```
org 100h
mov ax,0000h
mov ds,ax
mov ax,[3000h]
mov bx,[3002h]
```



```
and ax,bx
mov [3004h],ax
hlt
ret
```

For Obj. 2:

a) **Examine & analyze the output obtained from OR of two 16 bit numbers.**

mov ax,[1000h]

mov bx,[1002h]

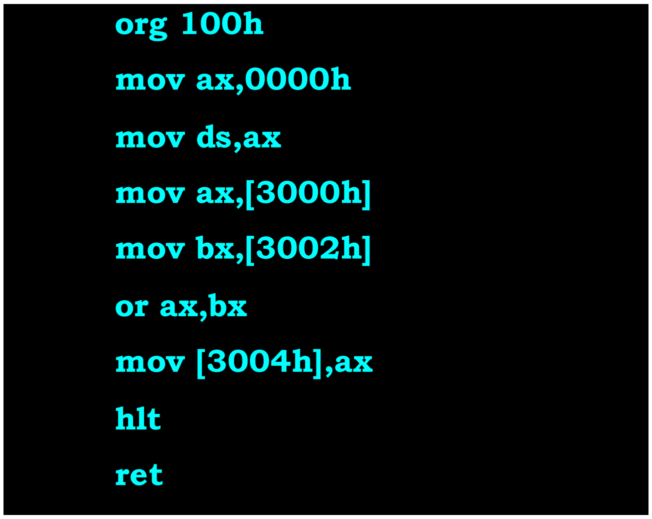
or ax,bx

[1000h] = 1234h

[1002h] = 4321h

Output: 5335h

b) **Write the assembly code.**



```
org 100h
mov ax,0000h
mov ds,ax
mov ax,[3000h]
mov bx,[3002h]
or ax,bx
mov [3004h],ax
hlt
ret
```

For Obj. 3:

a) **Examine & analyze the output obtained from NOT of a 16 bit number.**

```
mov ax,1234h
not ax
```

Output: EDCBh

b) Write the assembly code.

```
org 100h
mov ax,0000h
mov ds,ax
mov ax,[3000h]
not ax
mov [3002h],ax
hlt
ret
```

For Obj. 4:

a) Examine & analyze the output obtained from XOR of two 16 bit numbers.

```
mov ax,[1000h]
mov bx,[1002h]
xor ax,bx
```

[1000h] = 1234h

[1002h] = 4321h

Output: 5115h

b) Write the assembly code.

```
org 100h
mov ax,0000h
mov ds,ax

mov ax,[3000h]

mov bx,[3002h]

xor ax,bx

mov [3004h],ax

hlt

ret
```

III. LAB:

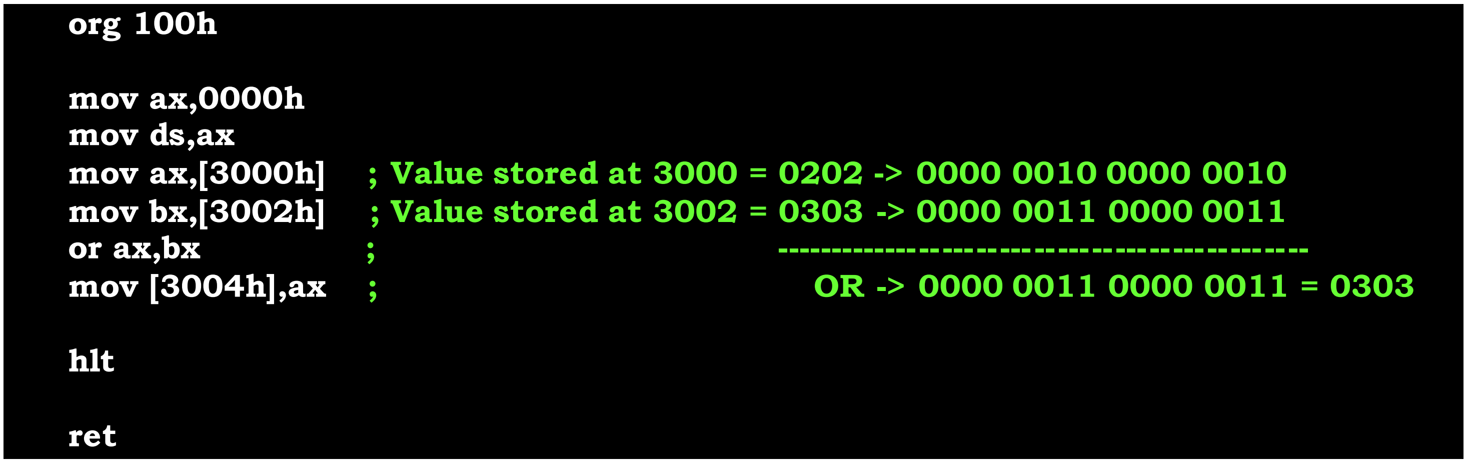
Assembly Program:

For Obj. 1

```
; SASWAT MOHANTY
; 1941012407

; AND two 16 bit numbers using direct addressing mode

org 100h

mov ax,0000h
mov ds,ax
mov ax,[3000h]    ; Value stored at 3000 = 0202 -> 0000 0010 0000 0010
mov bx,[3002h]    ; Value stored at 3002 = 0202 -> 0000 0010 0000 0010
and ax,bx         ;                         ------------------------------------------------
mov [3004h],ax    ;                              AND ->  0000 0010 0000 0010 = 0202

hlt

ret
```

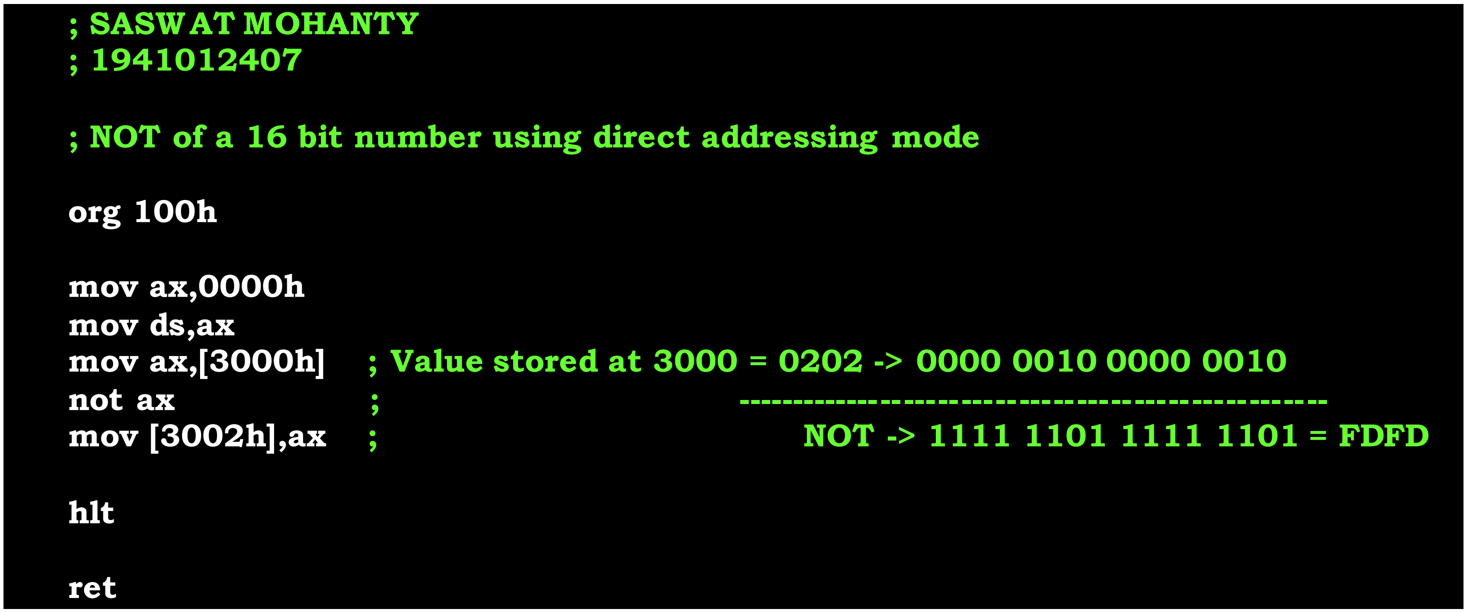
For Obj. 2

```
; SASWAT MOHANTY
; 1941012407

; OR two 16 bit numbers using direct addressing mode
```



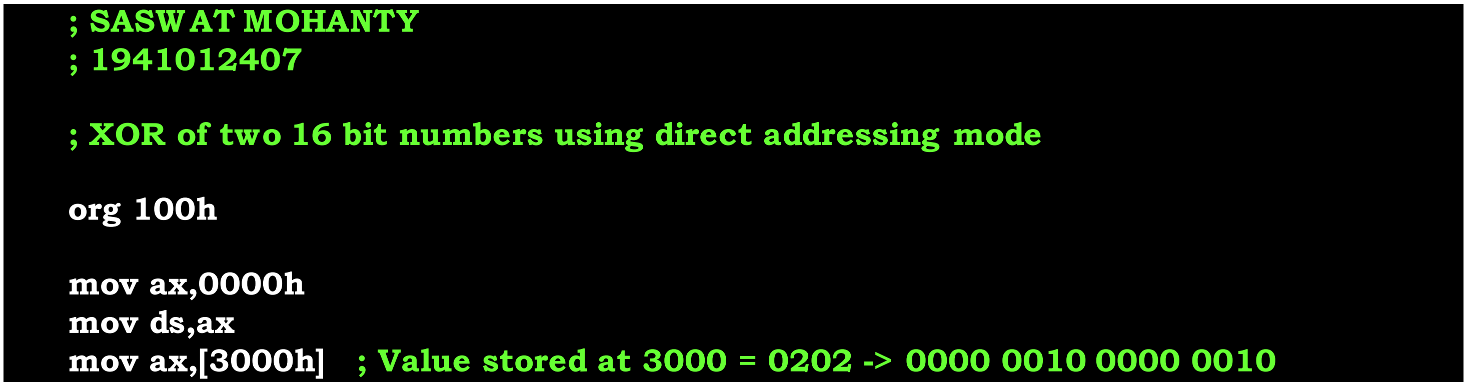
```
org 100h

mov ax,0000h
mov ds,ax
mov ax,[3000h]    ; Value stored at 3000 = 0202 -> 0000 0010 0000 0010
mov bx,[3002h]    ; Value stored at 3002 = 0303 -> 0000 0011 0000 0011
or ax,bx          ;                       ------------------------------------------
mov [3004h],ax    ;                          OR -> 0000 0011 0000 0011 = 0303

hlt

ret
```

For Obj. 3



```
; SASWAT MOHANTY
; 1941012407

; NOT of a 16 bit number using direct addressing mode

org 100h

mov ax,0000h
mov ds,ax
mov ax,[3000h]    ; Value stored at 3000 = 0202 -> 0000 0010 0000 0010
not ax            ;                    ------------------------------------------------
mov [3002h],ax    ;                         NOT -> 1111 1101 1111 1101 = FDFD

hlt

ret
```

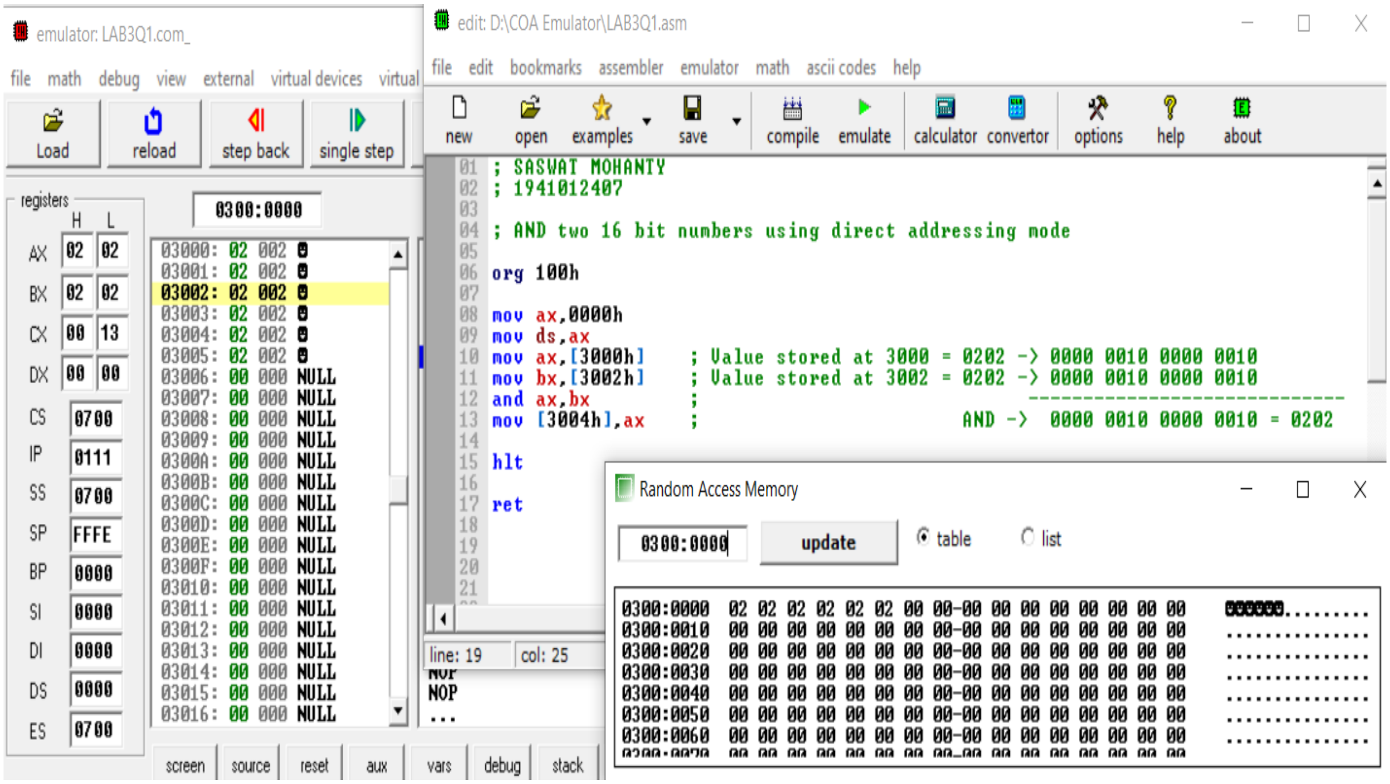
For Obj. 4



```
; SASWAT MOHANTY
; 1941012407

; XOR of two 16 bit numbers using direct addressing mode

org 100h

mov ax,0000h
mov ds,ax
mov ax,[3000h]   ; Value stored at 3000 = 0202 -> 0000 0010 0000 0010
mov bx,[3002h]   ; Value stored at 3002 = 0303 -> 0000 0011 0000 0011
xor ax,bx        ;                        -----------------------------------------------
mov [3004h],ax   ;                           XOR -> 0000 0001 0000 0001 = 0101

hlt

ret
```

Observations (with screen shots):

For Obj. 1



registers

	H	L
AX	02	02
BX	02	02
CX	00	13
DX	00	00
CS	0700	
IP	0111	
SS	0700	
SP	FFFE	
BP	0000	
SI	0000	
DI	0000	
DS	0000	
ES	0700	

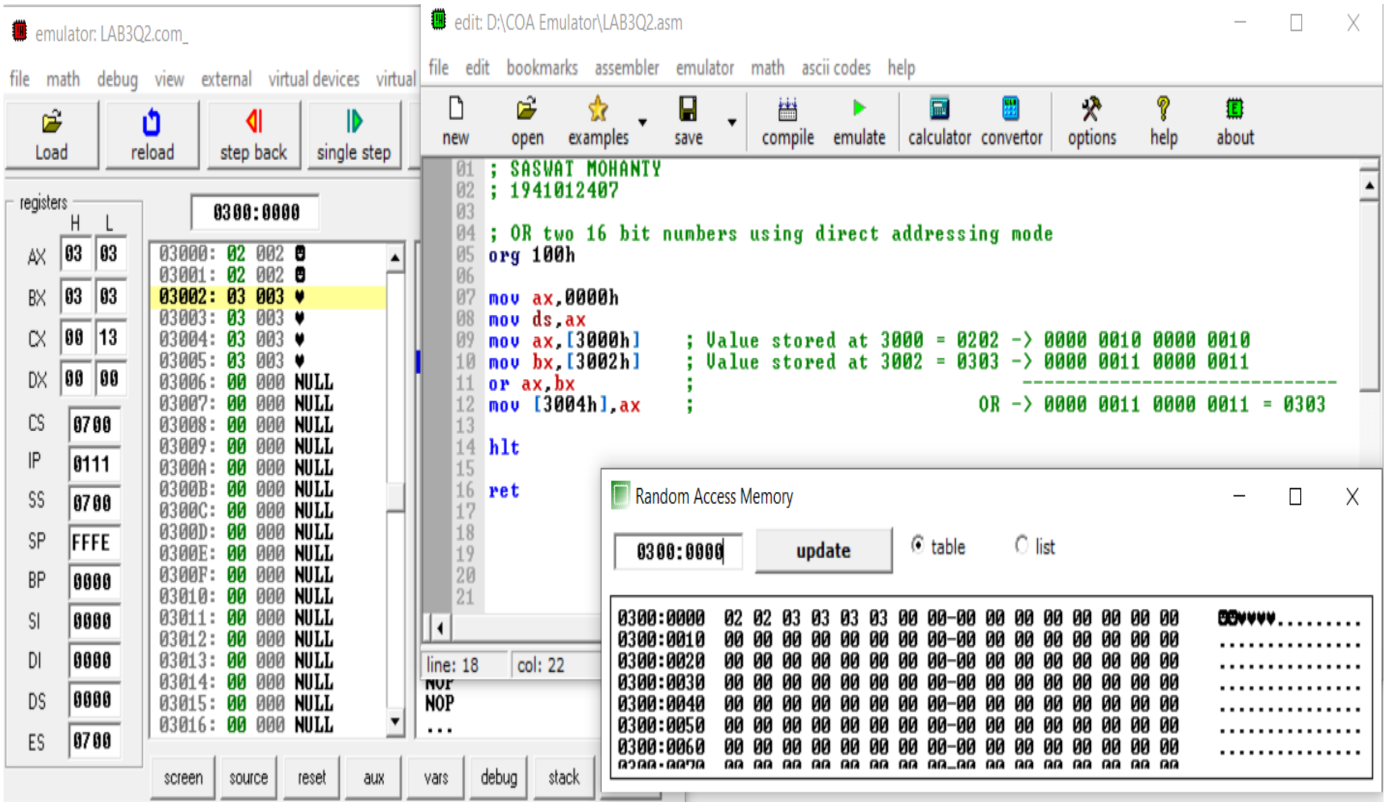
```
0300:0000

03000: 02 002 ☻
03001: 02 002 ☻
03002: 02 002 ☻
03003: 02 002 ☻
03004: 02 002 ☻
03005: 02 002 ☻
03006: 00 000 NULL
03007: 00 000 NULL
03008: 00 000 NULL
03009: 00 000 NULL
0300A: 00 000 NULL
0300B: 00 000 NULL
0300C: 00 000 NULL
0300D: 00 000 NULL
0300E: 00 000 NULL
0300F: 00 000 NULL
03010: 00 000 NULL
03011: 00 000 NULL
03012: 00 000 NULL
03013: 00 000 NULL
03014: 00 000 NULL
03015: 00 000 NULL
03016: 00 000 NULL
```

```
NOP
NOP
...
```

```
; SASWAT MOHANTY
; 1941012407

; AND two 16 bit numbers using direct addressing mode

org 100h

mov ax,0000h
mov ds,ax
mov ax,[3000h]      ; Value stored at 3000 = 0202 -> 0000 0010 0000 0010
mov bx,[3002h]      ; Value stored at 3002 = 0202 -> 0000 0010 0000 0010
and ax,bx           ;                              ------------------------------
mov [3004h],ax      ;                         AND ->  0000 0010 0000 0010 = 0202

hlt

ret
```

```
0300:0000

0300:0000  02 02 02 02 02 02 00 00-00 00 00 00 00 00 00 00    ☻☻☻☻☻☻.........
0300:0010  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00    ...............
0300:0020  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00    ...............
0300:0030  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00    ...............
0300:0040  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00    ...............
0300:0050  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00    ...............
0300:0060  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00    ...............
0300:0070  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
```

For Obj. 2


emulator: LAB3Q2.com_
file math debug view external virtual devices virtual
Load reload step back single step
screen source reset aux vars debug stack
edit: D:\COA Emulator\LAB3Q2.asm
file edit bookmarks assembler emulator math ascii codes help
new open examples save compile emulate calculator convertor options help about
Random Access Memory
update ● table ○ list

```
registers
      H   L
AX   03  03
BX   03  03
CX   00  13
DX   00  00
CS   0700
IP   0111
SS   0700
SP   FFFE
BP   0000
SI   0000
DI   0000
DS   0000
ES   0700
```

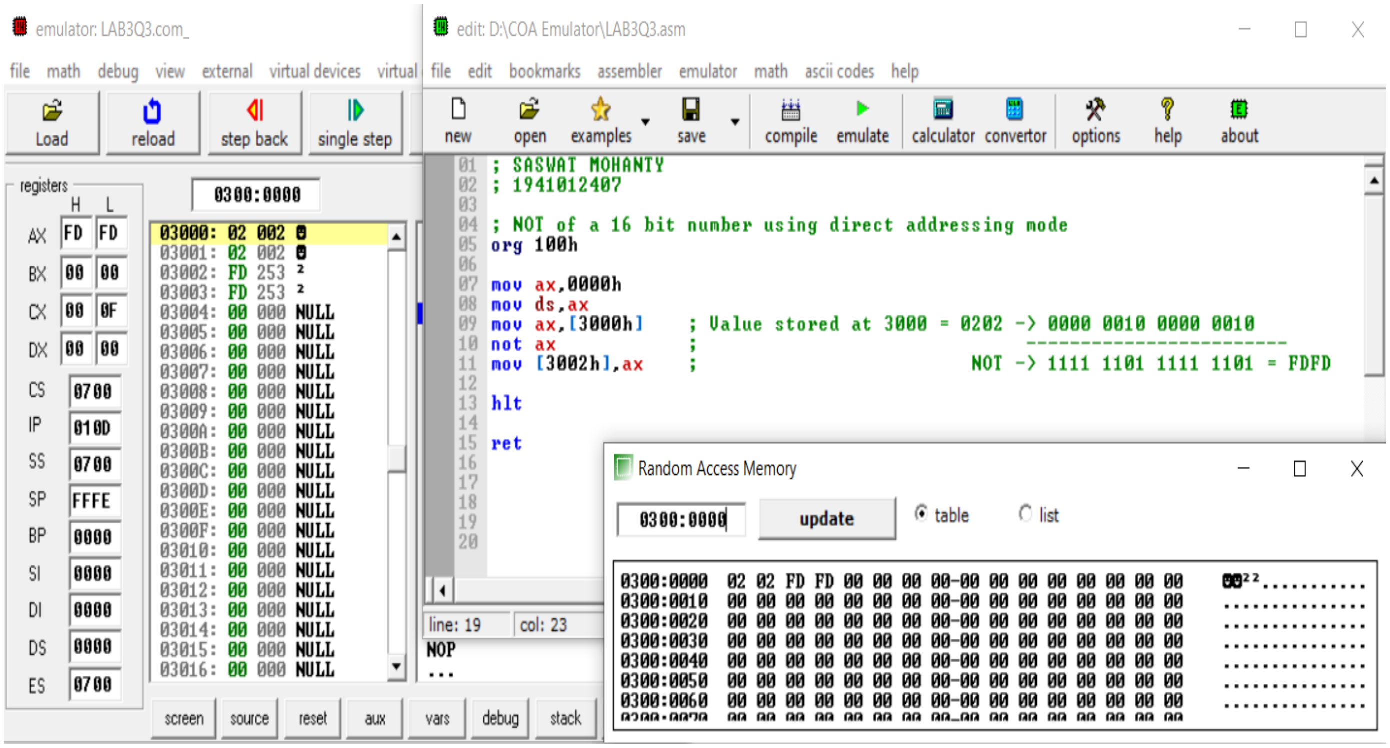
```
0300:0000
03000: 02 002 ☻
03001: 02 002 ☻
03002: 03 003 ♥
03003: 03 003 ♥
03004: 03 003 ♥
03005: 03 003 ♥
03006: 00 000 NULL
03007: 00 000 NULL
03008: 00 000 NULL
03009: 00 000 NULL
0300A: 00 000 NULL
0300B: 00 000 NULL
0300C: 00 000 NULL
0300D: 00 000 NULL
0300E: 00 000 NULL
0300F: 00 000 NULL
03010: 00 000 NULL
03011: 00 000 NULL
03012: 00 000 NULL
03013: 00 000 NULL
03014: 00 000 NULL
03015: 00 000 NULL
03016: 00 000 NULL
```

```
; SASWAT MOHANTY
; 1941012407

; OR two 16 bit numbers using direct addressing mode
org 100h

mov ax,0000h
mov ds,ax
mov ax,[3000h]     ; Value stored at 3000 = 0202 -> 0000 0010 0000 0010
mov bx,[3002h]     ; Value stored at 3002 = 0303 -> 0000 0011 0000 0011
or ax,bx           ;                           -------------------------
mov [3004h],ax     ;                        OR -> 0000 0011 0000 0011 = 0303

hlt

ret
```

```
line: 18    col: 22
NOP
...
```

```
0300:0000
0300:0000  02 02 03 03 03 03 00 00-00 00 00 00 00 00 00 00   ☻☻♥♥♥♥..........
0300:0010  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
0300:0020  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
0300:0030  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
0300:0040  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
0300:0050  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
0300:0060  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
```

For Obj. 3


emulator: LAB3Q3.com_
file math debug view external virtual devices virtual
Load reload step back single step
screen source reset aux vars debug stack
edit: D:\COA Emulator\LAB3Q3.asm
file edit bookmarks assembler emulator math ascii codes help
new open examples save compile emulate calculator convertor options help about
Random Access Memory
update ● table ○ list

```
registers
      H   L
AX   FD  FD
BX   00  00
CX   00  0F
DX   00  00
CS   0700
IP   010D
SS   0700
SP   FFFE
BP   0000
SI   0000
DI   0000
DS   0000
ES   0700
```

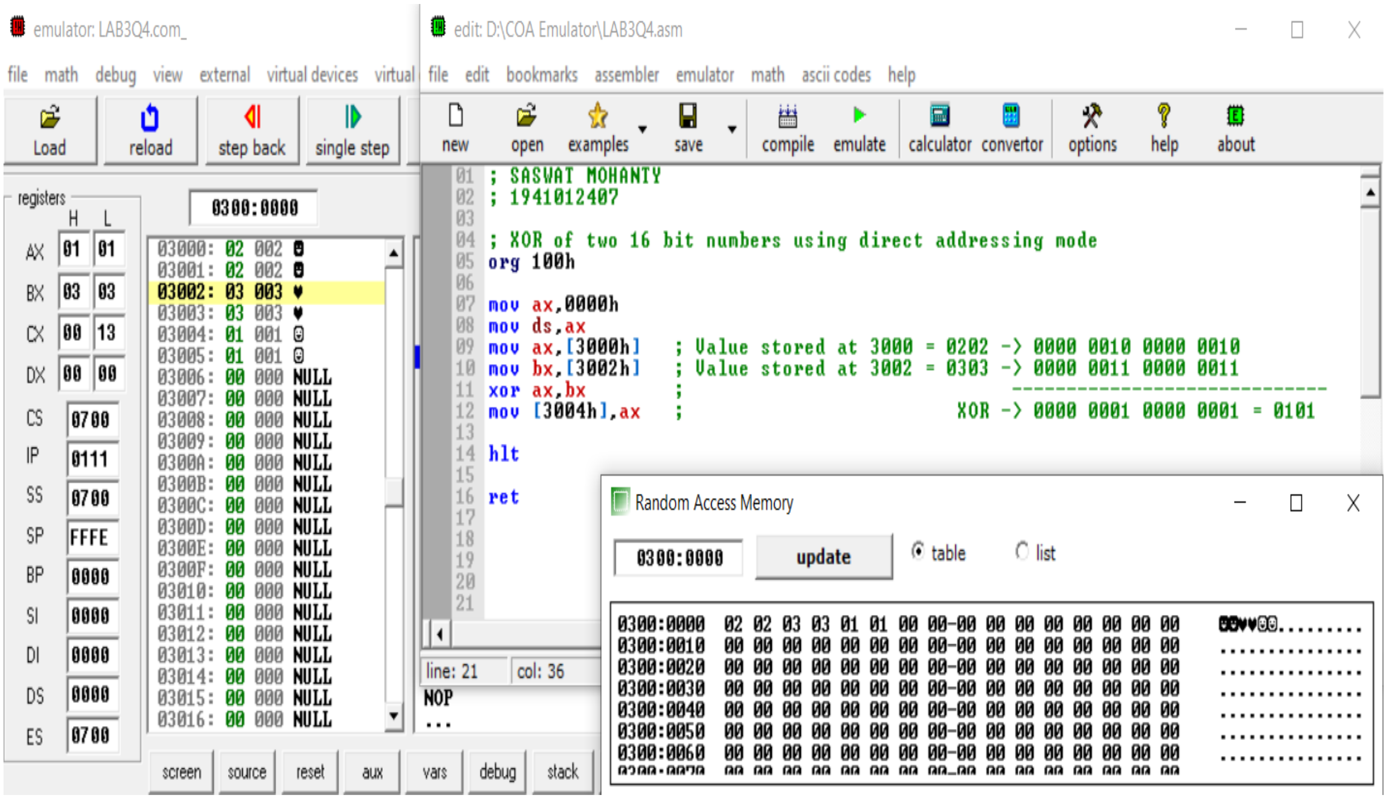
```
0300:0000
03000: 02 002 ☻
03001: 02 002 ☻
03002: FD 253 ²
03003: FD 253 ²
03004: 00 000 NULL
03005: 00 000 NULL
03006: 00 000 NULL
03007: 00 000 NULL
03008: 00 000 NULL
03009: 00 000 NULL
0300A: 00 000 NULL
0300B: 00 000 NULL
0300C: 00 000 NULL
0300D: 00 000 NULL
0300E: 00 000 NULL
0300F: 00 000 NULL
03010: 00 000 NULL
03011: 00 000 NULL
03012: 00 000 NULL
03013: 00 000 NULL
03014: 00 000 NULL
03015: 00 000 NULL
03016: 00 000 NULL
```

```
; SASWAT MOHANTY
; 1941012407

; NOT of a 16 bit number using direct addressing mode
org 100h

mov ax,0000h
mov ds,ax
mov ax,[3000h]     ; Value stored at 3000 = 0202 -> 0000 0010 0000 0010
not ax             ;                           ------------------------
mov [3002h],ax     ;                       NOT -> 1111 1101 1111 1101 = FDFD

hlt

ret
```

```
line: 19    col: 23
NOP
...
```

```
0300:0000
0300:0000  02 02 FD FD 00 00 00 00-00 00 00 00 00 00 00 00   ☻☻²²............
0300:0010  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
0300:0020  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
0300:0030  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
0300:0040  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
0300:0050  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
0300:0060  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
```

For Obj. 4


emulator: LAB3Q4.com_
file math debug view external virtual devices virtual
Load reload step back single step
edit: D:\COA Emulator\LAB3Q4.asm
file edit bookmarks assembler emulator math ascii codes help
new open examples save compile emulate calculator convertor options help about
screen source reset aux vars debug stack
Random Access Memory
update
line: 21 col: 36

```
registers
     H   L
AX   01  01
BX   03  03
CX   00  13
DX   00  00
CS   0700
IP   0111
SS   0700
SP   FFFE
BP   0000
SI   0000
DI   0000
DS   0000
ES   0700
```

```
0300:0000

03000: 02 002 ☻
03001: 02 002 ☻
03002: 03 003 ♥
03003: 03 003 ♥
03004: 01 001 ☺
03005: 01 001 ☺
03006: 00 000 NULL
03007: 00 000 NULL
03008: 00 000 NULL
03009: 00 000 NULL
0300A: 00 000 NULL
0300B: 00 000 NULL
0300C: 00 000 NULL
0300D: 00 000 NULL
0300E: 00 000 NULL
0300F: 00 000 NULL
03010: 00 000 NULL
03011: 00 000 NULL
03012: 00 000 NULL
03013: 00 000 NULL
03014: 00 000 NULL
03015: 00 000 NULL
03016: 00 000 NULL
```

```
NOP
...
```

```
; SASWAT MOHANTY
; 1941012407

; XOR of two 16 bit numbers using direct addressing mode
org 100h

mov ax,0000h
mov ds,ax
mov ax,[3000h]   ; Value stored at 3000 = 0202 -> 0000 0010 0000 0010
mov bx,[3002h]   ; Value stored at 3002 = 0303 -> 0000 0011 0000 0011
xor ax,bx        ;                          ---------------------------
mov [3004h],ax   ;                        XOR -> 0000 0001 0000 0001 = 0101

hlt

ret
```

```
0300:0000    ● table    ○ list

0300:0000  02 02 03 03 01 01 00 00-00 00 00 00 00 00 00 00    ☻☻♥♥☺☺..........
0300:0010  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00    ................
0300:0020  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00    ................
0300:0030  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00    ................
0300:0040  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00    ................
0300:0050  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00    ................
0300:0060  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00    ................
0300:0070  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
```

Conclusion:

For Obj. 1:

It can be concluded that the 'and' operation of numbers when dry run and executed in system found to be same. Thus, the program to and two 16-bit numbers was executed.

For Obj. 2:

It can be concluded that the 'or' operation of numbers when dry run and executed in system found to be same. Thus, the program to or two 16-bit numbers was executed.

For Obj. 3:

It can be concluded that the 'not' operation of numbers when dry run and executed in system found to be same. Thus, the program to not a 16-bit number was executed.

For Obj. 4:

It can be concluded that the ‘xor’ operation of numbers when dry run and executed in system found to be same. Thus, the program to xor two 16-bit numbers was executed.

IV. POST LAB:

1. Enlist the advantages of assembly language programming over machine language.

- It allows complex jobs to run in a simpler way.
- It is memory efficient, as it requires less memory.
- It is faster in speed, as its execution time is less.
- It is mainly hardware-oriented.
- It requires less instruction to get the result.
- It is used for critical jobs.
- It is not required to keep track of memory locations.
- It is a low-level embedded system.

2. Write the function of the following arithmetic instructions

a) ADC b) INC c) DEC d) SBB e) DAA

a) **ADC: -** Used to add with carry.

b) **INC: -** Used to increment the provided byte/word by 1.

c) **DEC: -** Used to decrement the provided byte/word by 1.

d) **SBB: -** Used to perform subtraction with borrow.

e) **DAA: -** Used to adjust the decimal after the addition/subtraction operation.

3. Write the function of the following logical instructions

b) SHL/SAL b) SHR c) SAR d) ROR e) ROL

a) **SHL/SAL: -** Used to shift bits of a byte/word towards left and put zero(S) in LSBs.

b) **SHR: -** Used to shift bits of a byte/word towards the right and put zero(S) in MSBs.

c) **SAR: -** Used to shift bits of a byte/word towards the right and copy the old MSB into the new MSB.

d) **ROR: -** Used to rotate bits of byte/word towards the right, i.e., LSB to MSB and to Carry Flag [CF].

e) **ROL: -** Used to rotate bits of byte/word towards the left, i.e. MSB to LSB and to Carry Flag [CF].