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:			**Section:**
S. No.	**Name**	**Registration No.**	**Signature**
30	**Aishwary Singh Gaharwar**	**1941017093**	Aishwary Singh

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.

Ans: 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
hlt
```

Output:

[1000h] ← 0064

[1002h] ← 00C8

0040

c. Write the assembly code.

```
mov ax, [1000h]
mov bx, [1002h]
and ax,bx
hlt
```

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
hlt
```

Output:

[1000h] ← 0064

[1002h] ← 00C8

00EC

b. Write the assembly code.

```
mov ax, [1000h]
mov bx, [1002h]
or ax, bx
hlt
```

For Obj. 3:

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

```
mov ax, [1000h]
not ax
hlt
```

Output:

[1000h] ← 0007

FFF8

b. Write the assembly code.

```
mov ax, [1000h]
not ax
hlt
```

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

hlt

Output:

[1000h] ← 0064

[1002h] ← 00C8

00AC

b. Write the assembly code.

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

III. LAB:

Assembly Program:

Obj 1:

```
mov ax, [1000h]
mov bx, [1002h]
and ax,bx
hlt
```

Obj2:

```
mov ax, [1000h]
mov bx, [1002h]
or ax, bx
hlt
```

Obj3:

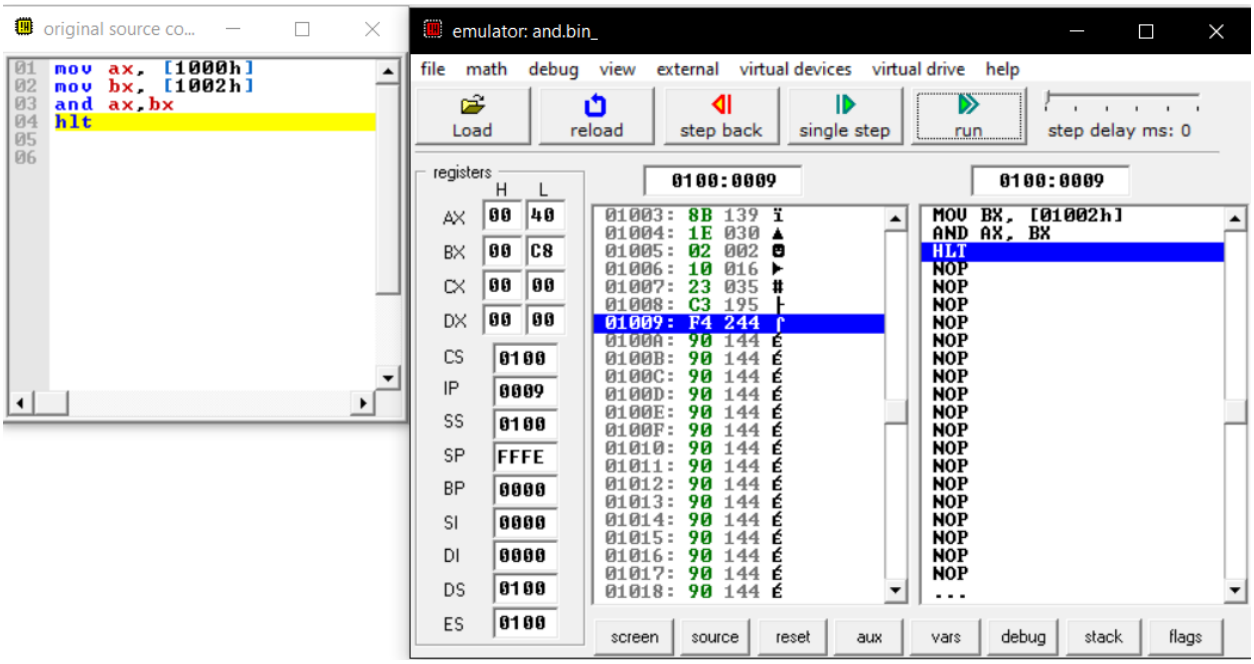
```
mov ax, [1000h]
not ax
hlt
```

Obj4:

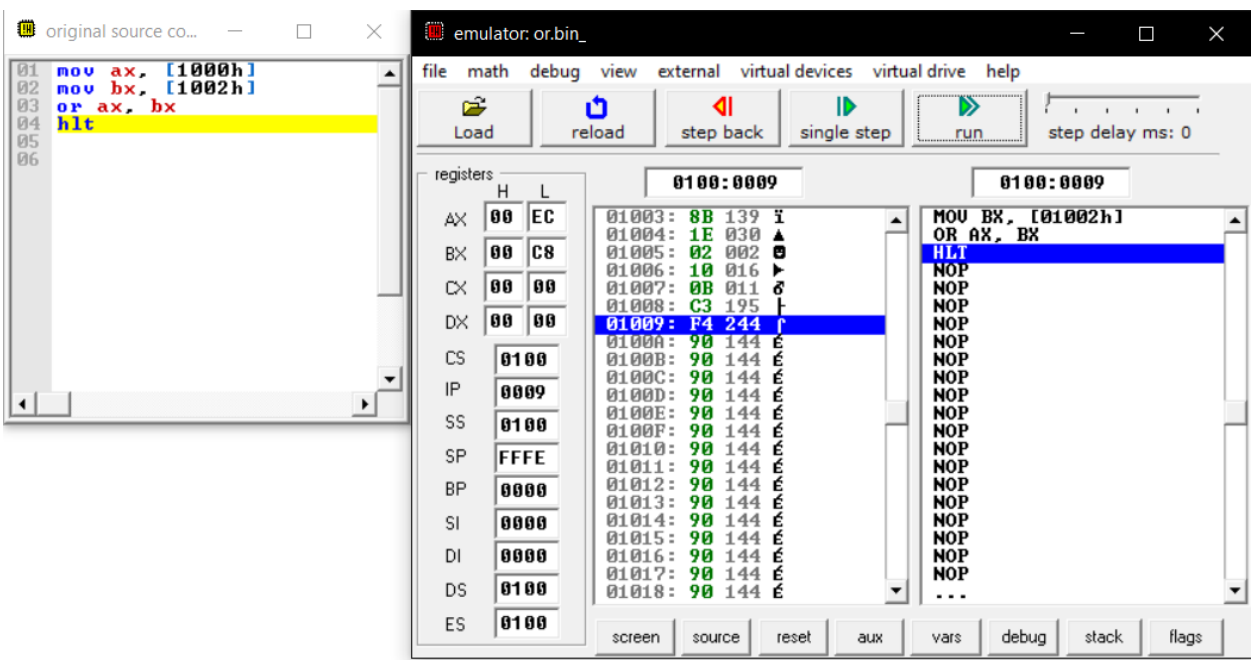
```
mov ax, [1000h]
mov bx, [1002h]
xor ax, bx
hlt
```

Observations (with screen shots):

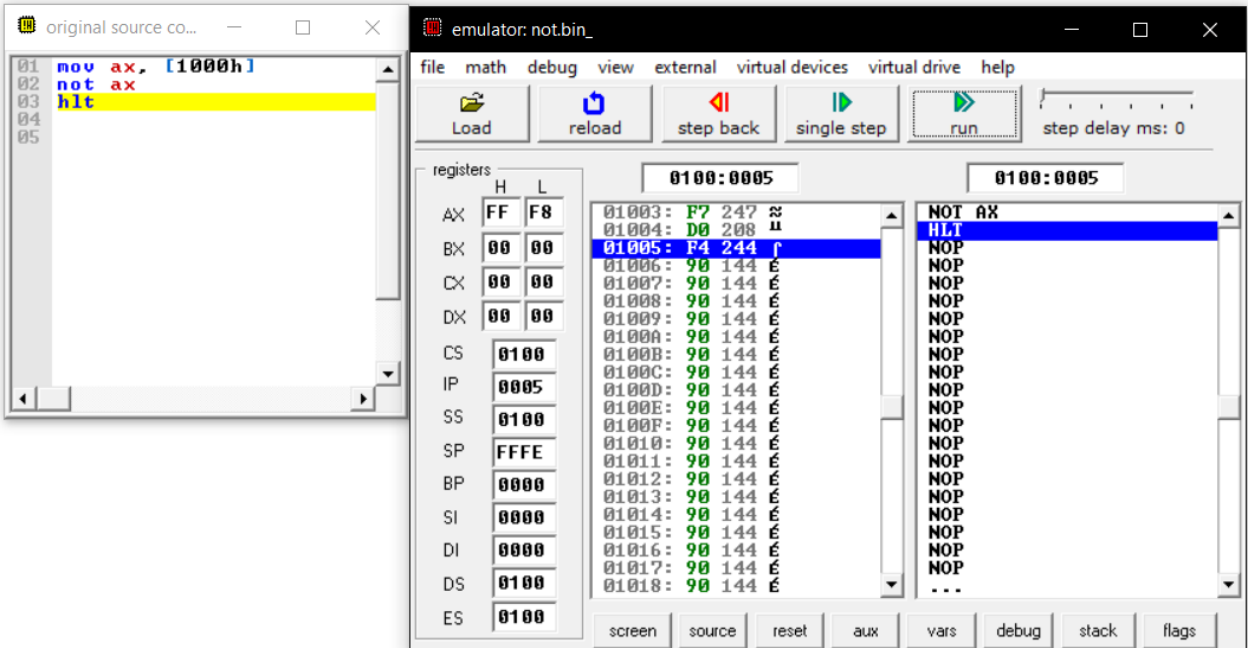
Obj 1:


original source co...
01 mov ax, [1000h]
02 mov bx, [1002h]
03 and ax,bx
04 hlt
emulator: and.bin_
file math debug view external virtual devices virtual drive help
Load
reload
step back
single step
run
step delay ms: 0
registers
H L
AX 00 40
BX 00 C8
CX 00 00
DX 00 00
CS 0100
IP 0009
SS 0100
SP FFFE
BP 0000
SI 0000
DI 0000
DS 0100
ES 0100
0100:0009
01003: 8B 139
01004: 1E 030
01005: 02 002
01006: 10 016
01007: 23 035
01008: C3 195
01009: F4 244
0100A: 90 144
0100B: 90 144
0100C: 90 144
0100D: 90 144
0100E: 90 144
0100F: 90 144
01010: 90 144
01011: 90 144
01012: 90 144
01013: 90 144
01014: 90 144
01015: 90 144
01016: 90 144
01017: 90 144
01018: 90 144
0100:0009
MOV BX, [01002h]
AND AX, BX
HLT
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
...
screen
source
reset
aux
vars
debug
stack
flags

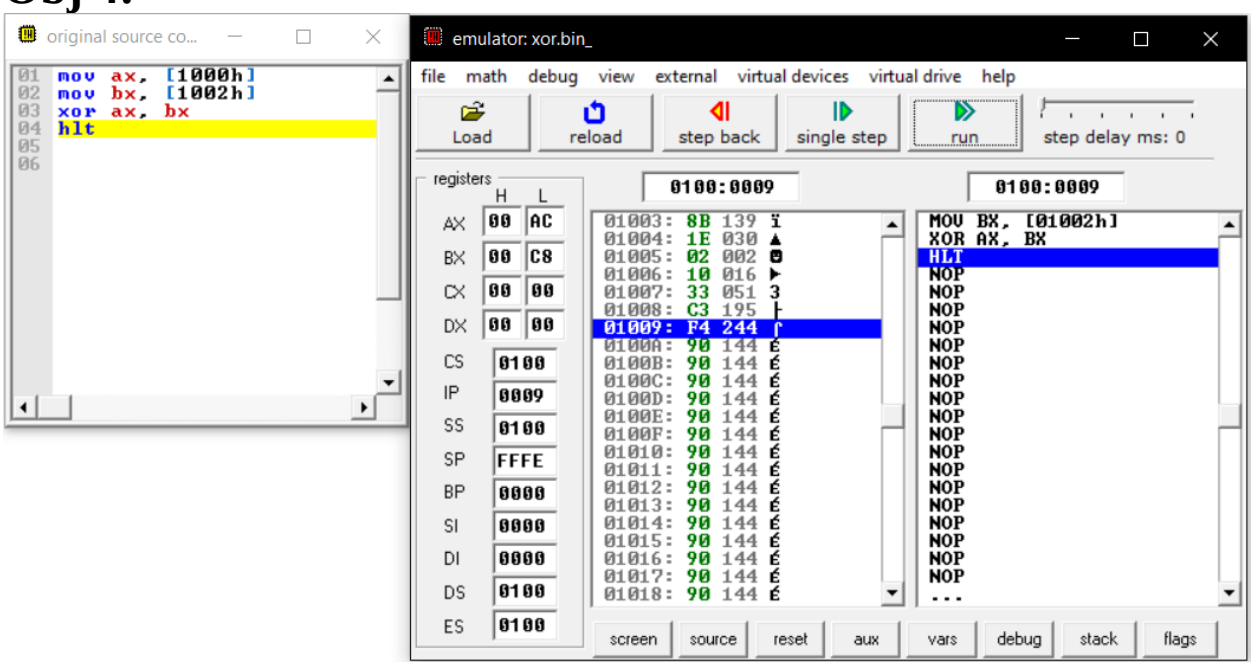
Obj 2:


original source co...
01 mov ax, [1000h]
02 mov bx, [1002h]
03 or ax, bx
04 hlt
emulator: or.bin_
file math debug view external virtual devices virtual drive help
Load
reload
step back
single step
run
step delay ms: 0
registers
H L
AX 00 EC
BX 00 C8
CX 00 00
DX 00 00
CS 0100
IP 0009
SS 0100
SP FFFE
BP 0000
SI 0000
DI 0000
DS 0100
ES 0100
0100:0009
01003: 8B 139
01004: 1E 030
01005: 02 002
01006: 10 016
01007: 0B 011
01008: C3 195
01009: F4 244
0100A: 90 144
0100B: 90 144
0100C: 90 144
0100D: 90 144
0100E: 90 144
0100F: 90 144
01010: 90 144
01011: 90 144
01012: 90 144
01013: 90 144
01014: 90 144
01015: 90 144
01016: 90 144
01017: 90 144
01018: 90 144
0100:0009
MOV BX, [01002h]
OR AX, BX
HLT
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
...
screen
source
reset
aux
vars
debug
stack
flags

Obj 3:



Obj 4:



Conclusion:

Obj 1:

It can be concluded the calculation of offset code for AND operation of two 16-bit numbers using direct addressing mode works correctly.

Obj2:

It can be concluded the calculation of offset code for OR operation of two 16-bit numbers using direct addressing mode works correctly.

Obj 3:

It can be concluded the calculation of offset code for NOT operation of two 16-bit numbers using direct addressing mode works correctly.

Obj 4:

It can be concluded the calculation of offset code for XOR operation of two 16-bit numbers using direct addressing mode works correctly.

IV. POST LAB:

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

Ans: Advantages of Assembly Language over Machine Language:

- Development Time
- Reliability and Security

2. Write the function of the following arithmetic instructions.

a) ADC- Used to add with carry.
b) INC- Used to increment the provided byte/word by 1.
c) DEC- Used to adjust the decimal after the addition/subtraction operation.
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.
   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].