Computer Organization and Architecture (EET2211)

LAB I: Analyze the Arithmetic and Logical operations using different Addressing Modes of the 8086 Microprocessor.

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

Branch:		**Section:**	
S. No.	**Name**	**Registration No.**	**Signature**
50	Dhiraj Kumar Pasayat	2241016218	Dhiraj ku Pasayat

Marks: ______/10

Remarks:

Teacher's Signature

I. OBJECTIVE:

1. Perform Addition, Subtraction, Multiplication, and Division of two 16-bit numbers using immediate addressing mode and store the results using direct addressing mode.
2. Perform the following operations on two 8-bit data (**data1, data2**) given in memory locations and store the result in another memory location using indirect addressing mode.
   - i. Swapping of nibble of **data1**
   - ii. `Y= (data1 and data2) or (data1 xor data2)`
3. Find the Gray code of an 8-bit binary number.
4. Find the 2's complement of an 8-bit number.

II. PRE-LAB

- Explain the addressing modes involved in instructions.

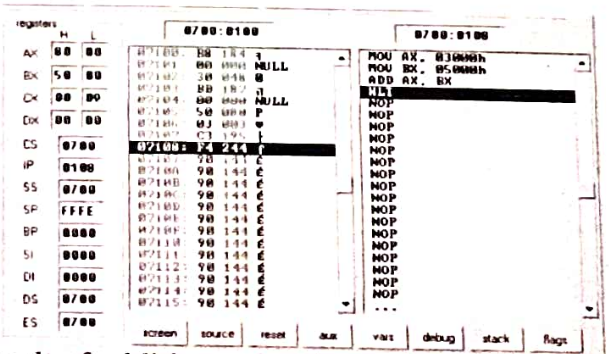
For each objective in prelab describe the following points:

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

III. LAB

Note: For each objective do the following job and assessment:

- Screenshots of the Assembly language program (ALP)
- Observations (with screenshots)



registers

	H	L
AX	00	00
BX	50	00
CX	00	09
DX	00	00

CS	0700
IP	0108
SS	0700
SP	FFFE
BP	0000
SI	0000
DI	0000
DS	0700
ES	0700

```
0700:0100
07100:  B8  184  ╕
07101:  00  000  NULL
07102:  30  048  0
07103:  BB  187  ╗
07104:  00  000  NULL
07105:  50  080  P
07106:  03  003  ♥
07107:  C3  195  ├
07108:  F4  244  ⌠
07109:  90  144  É
0710A:  90  144  É
0710B:  90  144  É
0710C:  90  144  É
0710D:  90  144  É
0710E:  90  144  É
0710F:  90  144  É
07110:  90  144  É
07111:  90  144  É
07112:  90  144  É
07113:  90  144  É
07114:  90  144  É
07115:  90  144  É
```

```
0700:0100
MOV AX, 03000h
MOV BX, 05000h
ADD AX, BX
HLT
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
...
```

Fig. 1. Execution result of addition using immediate and direct addressing mode of 8086 emulator.

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

Input:

Sl. No.	**Memory Location**	**Operand (Data)**
1		
2		
...		

Output:

Sl. No.	**Memory Location**	**Operand (Data)**
1		
2		
...		

IV. CONCLUSION

V. POST LAB

1. Discuss different general-purpose registers used in 8086 microprocessors.
2. Explain the concept of segmented memory. What are its advantages?
3. Explain the physical address formation in 8086.
4. Write an assembly program to multiply 05H and 04H without using arithmetic instruction.
5. Write the function of the following logical instructions.
   a) SHL/SAL b) SHR c) SAR d) ROR e) ROL

II. PRE-LAB

- Explain the addressing modes involved in instructions.

Immediate addressing mode:- In this type of addressing, immediate data is a part of instruction and appears in the form of successive bite or bites

Eg:- MOV AX, 1250H

Direct addressing mode:- In this addressing mode, a 16-bit memory address or offset is directly specified in instruction as a part of it

Eg:- MOV AX, [5000H]

Indirect addressing mode:- In this addressing mode, the data is stored in an register or a memory location and is accessed through the register or memory location

Eg:- MOV AX, [BX]

It is further classified into Register Indirect and Memory Indirect

- For each objective in prelab describe the foll^{n} points:
  - Write assembly code with description
  - Examine and analyze the input / output of assembly code.

Obj1:-

MOV AX, 3234H - AX ← 3234H

MOV CX, AX - CX ← AX

ADD AX, 1111H - AX ← AX(3234H) + 1111H

MOV [5000H], AX - [5000H] ← 4345H (AX)

MOV AX, CX - AX ← CX

SUB AX, 1111H - AX ← AX(3234H) - 1111H

MOV [5002H], AX - [5002H] ← 2123H (AX)

MOV AX, CX - AX ← CX

MOV BX, 0002H - BX ← 0002H

MUL BX - AX X BX, Higher 16bit ~~Quotient~~ → DX

Lower 16bit → AX

MOV [5004H], DX - [5004H] ← 0000H

MOV [5006H], AX - [5006H] ← 64B8 ~~9808~~ H

```
MOV  AX, CX          -  AX ← CX
DIV  BX              -  AX ÷ BX, Quotient → DX, Remainder → AX
MOV  [7008H], DX     -  [7008H] ← 9F08H
MOV  [700AH], AX     -  [700AH] ← 0
HLT
```

Input: Output :-

AX → 32 34H   Addition :- 4345H

Substraction :- 2123H

Multiplication :- 646FH

Division :- 9F08H

Obj 2:

i)

```
MOV  SI, ~~3000H~~ 3000H    -  SI ← 3000H
MOV  AL, [SI]               -  AL ← [SI]
ROR  AL, 04                 -  23H in Rotated Right without carry
                               4 times
MOV  [SI+1], AL             -  [SI+1] ← 23H
HLT
```

Input :- 32H   Output :- 23H

ii)

```
MOV  SI, 2000H      -  SI ← 2000H
MOV  AL, [SI]       -  AL ← [2000H]
MOV  BL, [SI+1]     -  BL ← [2001H]
MOV  DL, AL         -  DL ← AL
AND  AL, BL         -  AL ← AL and BL
XOR  DL, BL         -  DL ← DL ~~and~~ xor BL
OR   AL, DL         -  AL ← AL or DL
MOV  [SI+2], AL     -  [SI+2] ← AL
HLT
```

Input :- 12H   Output
34H

Obj 3

```
MOV SI, 2000H      -   SI ← 2000H
MOV AL, [SI]       -   AL ← [2000H]
MOV BL, AL         -   BL ← AL
SHR AL, 01         -   AL is Bitwise shifted to right
XOR AL, BL         -   AL ← AL XOR BL
MOV [SI+1], AL     -   [2001H] ← AL
HLT
```

Input:	Output
B5	EF

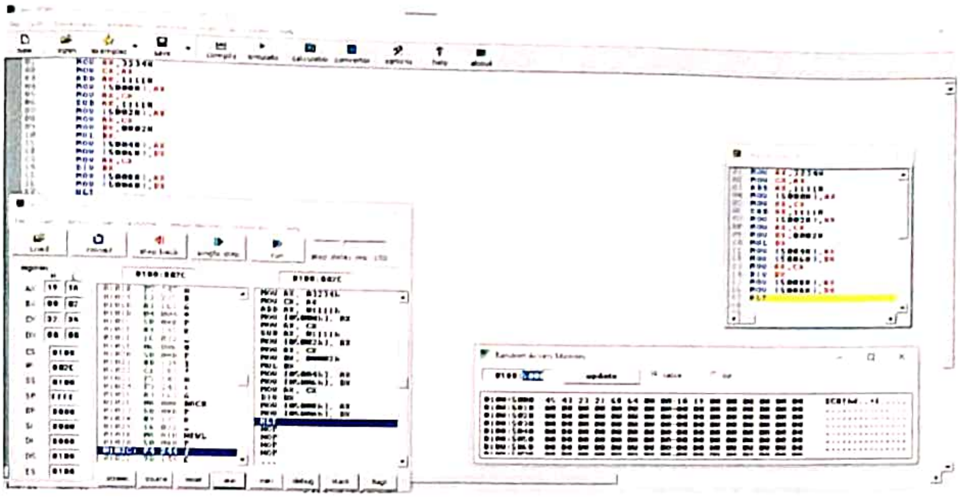
Obj 4

```
MOV SI, 3000H      -   SI ← 3000H
MOV AL, [SI]       -   AL ← [3000H]
NOT AL             -   AL ← AL compliment
ADD AL, 01         -   AL ← AL + 1
MOV [SI+1], AL     -   [3001H] ← AL
HLT
```

Input	Output
31	DF

III. LAB.

OBJ 1:



```
MOV AX,3234H
MOV CX,AX
ADD AX,1111H
MOV [5000H],AX
MOV AX,CX
SUB AX,1111H
MOV [5002H],AX
MOV AX,CX
MOV BX,0002H
MUL BX
MOV [5004H],AX
MOV [5006H],DX
MOV AX,CX
DIV BX
MOV [5008H],AX
MOV [500AH],DX
HLT
```

From this result, I have observed:

Input:

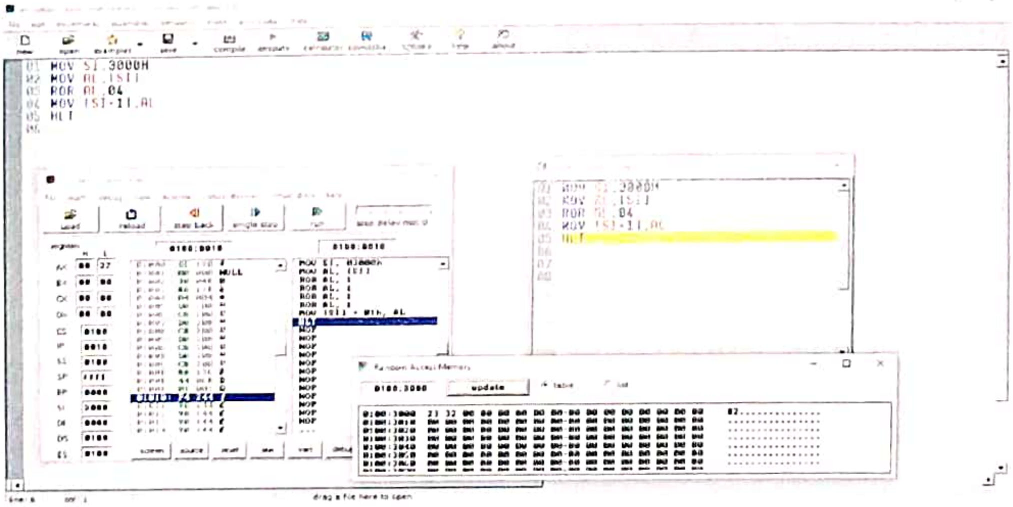
Sl.no	Operation	Data
1.	ADD	3234H ,1111H
2.	SUB	3234H, 1111H
3.	MUL	3234H, 0002H
4.	DIV	3234H, 0002H

Output:

Sl.no	Memory Location	Operand (Data)
1.	5000H,5001H	ADD (45 43)
2.	5002H,5003H	SUB (23 21)
3.	5004H,5005H	MUL (68 64)
4.	5006H,5007H	MUL (00 00)
5.	5008H,5009H	DIV (00 1A)
6.	5009H, 500AH	DIV (19 00)

OBJ 2:

a)



```
MOV SI,3000H
MOV AL,[SI]
ROR AL,04
MOV [SI+1],AL
HLT
```

From this result I have observed

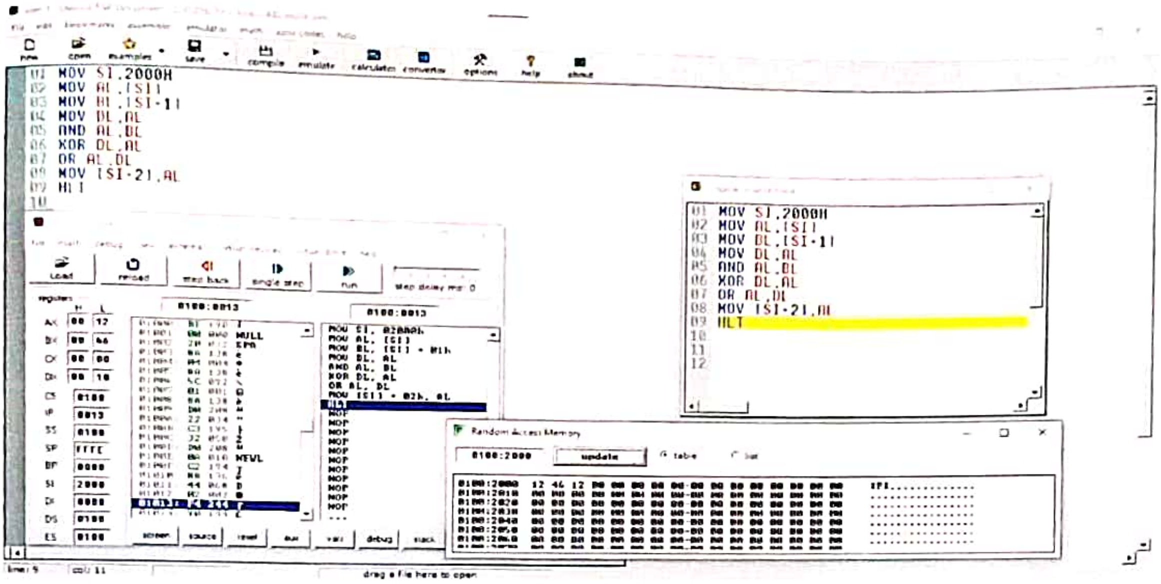
Input:

Sl.No.	Memory Location	Data
1	3000H	23H

Output:

Sl.No.	Memory Location	Data
1	3001H	32H

b)



```
MOV SI,2000H
MOV AL,[SI]
MOV BL,[SI+1]
MOV DL,AL
AND AL,BL
XOR DL,AL
OR AL,DL
MOV [SI+2],AL
HLT
```

From this result I have observed

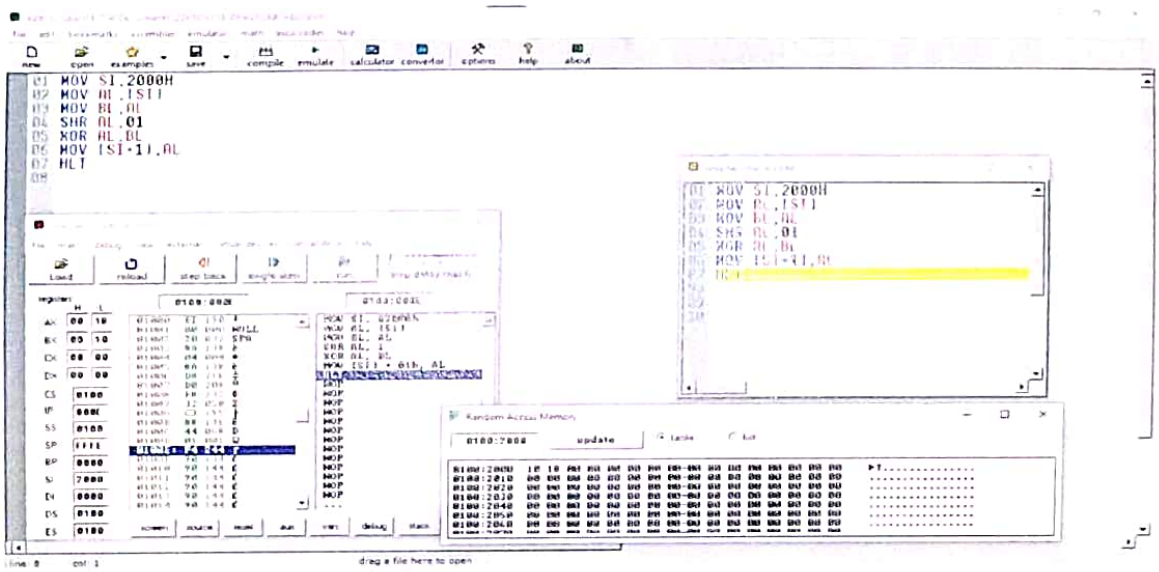
Input:

Sl.No.	Memory Location	Data
1	2000H	12H
2.	2001H	46H

Output:

Sl.No.	Memory Location	Data
1	2002H	12H

OBJ 3:



```
MOV SI,2000H
MOV AL,[SI]
MOV BL,AL
SHR AL,01
XOR AL,BL
MOV [SI+1],AL
HLT
```

From this result I have observed

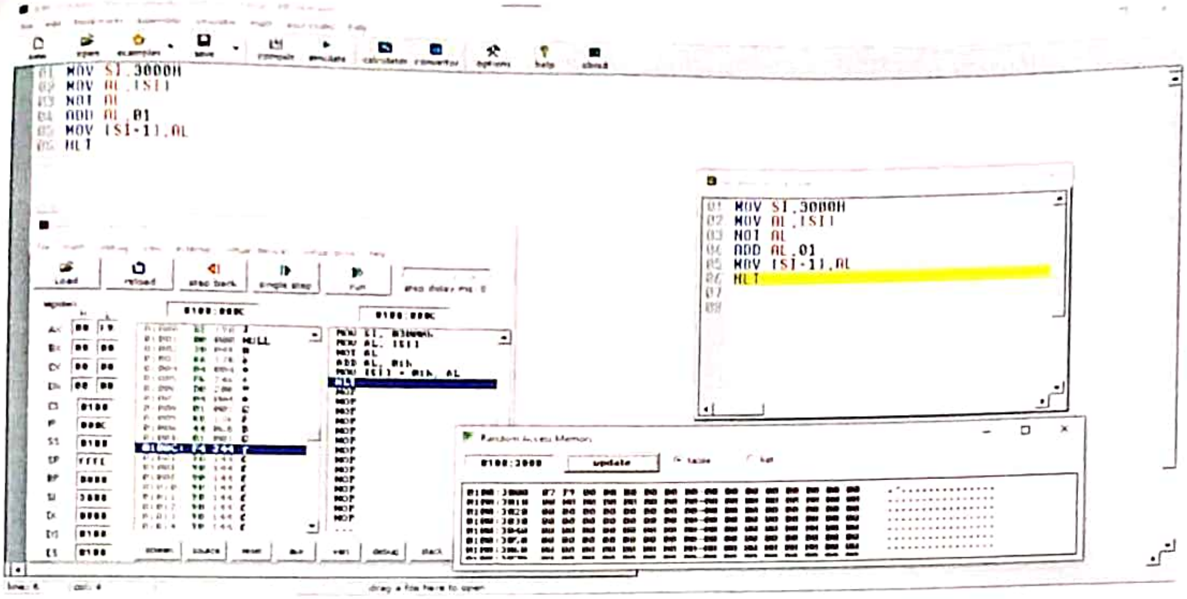
Input:

Sl.No.	Memory Location	Data
1	2000H	10H

Output:

Sl.No.	Memory Location	Data
1	2001H	18H

OBJ 4:



```
MOV SI,3000H
MOV AL,[SI]
NOT AL
ADD AL,01
MOV [SI+1],AL
HLT
```

```
MOV SI,3000H
MOV AL,[SI]
NOT AL
ADD AL,01
MOV [SI+1],AL
HLT
```

From this result I have observed

Input:

Sl.No.	Memory Location	Data
1	3000H	07H

Output:

Sl.No.	Memory Location	Data
1	3001H	F9H

IV. CONCLUSION

In this experiment we were able to analyze different arithmetic and Logical operation using Direct, Indirect and Immediate addressing modes:

V. POST LAB

1) Discuss different general-purpose registers used in 8086 microprocessors

Solⁿ There are four 16-bit general purpose registers in 8086 microprocessor:-

i) AX → It is used as a 16 bit accumulator. Lower 8 bits of AX is designated as AL and higher 8 bits as AH

ii) Bx → It is used to provide offset storage for creating the physical address in some addressing mode Reffered as Base Register. Lower 8 bits designated as BL, higher 8 bits as BH

iii) CX → It is used as a default counter in case of String and Loop instructions. Lower 8 bits designated as CL, higher 8 bits as CH

iv) DX → It is a general purpose register which may be used as an operand or a destination register. Lower 8 bits designated as DL, higher 8 bits as DH

2) Explain the concept of segmented memory. What are its advantages

Solⁿ Segmentation is a memory management technique used in Intel 8086 microprocessor. It logically divides main memory into different segments, each with its own base address

The four segment registers are:

P.T.O

- Code Segment Register (CS) :- used to address a memory location in the core segment of the memory
- Data segment Register (DS) :- Points to the Data Segment where the data is stored
- Extra segment Register (ES) :- Also refers to a segment which is another data segment in memory
- Stack segment Register (SS) :- Addresses the Stack segment used for storing stack data.

The advantages of Segmented memory are:-

- Efficient memory organization and management
- Isolation of Data and Code
- Processes can share data across segments, facilitating communication between different parts of a program
- Supports Memory protection mechanisms, preventing ~~[illegible]~~ unauthorized access to specific segments.

3) Explain the physical address formation in 8086.

Soln The 8086 microprocessor generates a physical address by combining segment and offset addresses for efficient memory access and management. It is given by

$$P.A. = \text{Segment address} \times 10H + \text{offset address}$$

4) Write an assembly code program to multiply 05H and 04H without using arithmetic instruction.

Soln

```
MOV  AX, 05H       -  AX ← 05H
SHL  AX, 02        -  AX ← 05H shifted left two times
MOV  [5000H], AX   -  [5000H] ← AX
```

Write the function of the following logical instructions.
a) SHL / SAL b) SHR c) SAR d) ROR e) ROL

Solⁿ a) SHL / SAL :- Shift Logical / Arithmetic Left:
It shift the operand word or byte bit by bit to the left. and insert zeros in the ~~to~~ LSB.

b) SHR :- Shift Logical Right. It shifts the operand word or byte bitwise towards right and inserts zeros in the shifted position

c) SAR :- Shift Arithmetic Right. It performs right shift on the word or byte and inserts the Most significant bit of the operand in the newly inserted position

d) ROR :- Rotate right without carry. It rotates the word or byte bitwise by a specific number towards right

e) ROL :- Rotate Left without carry. ~~It~~ It rotates the the word or byte bitwise by a specific number towards left