QUEST FOR EXCELLENCE	**MAHARASHTRA INSTITUTE OF TECHNOLOGY AURANGABAD**	**LABORATORY MANUAL**
	PRACTICAL EXPERIMENT INSTRUCTION SHEET	
	DEPARTMENT: COMPUTER SCIENCE AND ENGINEERING	
LABORATORY :		A.Y.: 2020-21
Class: SY CSE	PART: I	SUBJECT: DEMP

MASTER LIST OF EXPERIMENT

1. Implementation of Boolean expression using AND/OR/NOT & NAND/NOR logic.
2. Realization of Half & Full Adder using logic gates.
3. Realization of Half & Full Subtractor using logic gates.
4. Design & Implement 8:1 Multiplexer and 1:8 De-multiplexer using logic gates.
5. Demonstrate the working of flip-flop.
6. Write an Assembly language program to print the string in 8086.
7. Write an Assembly language program for 8-bit & 16-bit addition in 8086.
8. Write an Assembly language program for 8-bit &16-bit subtraction in 8086.
9. Write an Assembly Language Program for 8-bit multiplication & 16-bit division in 8086.
10. Write an Assembly Language Program for finding smallest number from an array in 8086.

PREPARED BY : Dr. Geeta Tripathi	APPROVED BY : HCSED

EXPERIMENT NO.2

AIM: Realization of Half adder and Full adder using logic gates.

APPRATUS:

A. Components required for half adder.

Component	Quantity
1.Trainer kit	1
2.IC7408 2-input AND GATE	1
3.IC7486 2-input EX-OR GATE	1
4.2mm patch cords.	

B. Component required for full adder.

Component	Quantity
1.Omega type LTB 860	1
2.IC7408 2-input AND GATE	1
3.IC7486 2-input EX-OR GATE	1
4.IC7432 2-input OR GATE	1
5.2mm patch cords.	

THEORY :

A. HALF ADDER: Half adder is a combinational unit with 2-input and 2-output.it is a basic building block for addition of two 'single' bit numbers. It has two inputs A and B and output 'sum', 'Carry' namely. This circuit is not suitable for multi bit building addition where carry is propagated to next bit.

B.FULL ADDER: Full adder 3 single bit adder circuit. It can add two one-bit number A,B

and carry Cin. The full adder in 3 inputs and 2 output combinational circuite.it can be used for multi-bit addition.

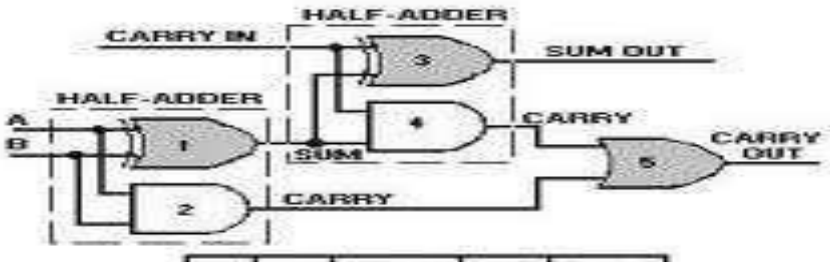
A full adder can be implemented in many different ways such as with a custom transistor level circuit or composed of other gates. One example implementation is with

$S=(A+B)+Cin$ and $Cout=(A.B)+(Cin(A \quad B))$.

$\text{Input} : \{A,B,Cin\} \longrightarrow \text{Output} : \{S,Out\}$

In this implementation the final OR GATE before the carry out output may be replaced by an XOR GATE without output may be replaced by an XOR GATE without altering the resulting logic.

A full adder can be constructed from two half by connecting A and B to the input of one half adders. Connecting the sum from to an input to the second adder. Connecting Cin to the other input and or the two carry outputs Equivalently. S could be made the three bit XOR of A,B,Ci and Co could be made the three bit majority function A,B and Ci.



A	B	CARRY IN	SUM OUT	CARRY OUT
0	0	0	0	0
0	1	0	1	0
1	0	0	1	0
1	1	0	0	1
0	0	1	1	0
0	1	1	0	1
1	0	1	0	1
1	1	1	1	1

	MAHARASHTRA INSTITUTE OF TECHNOLOGY AURANGABAD	LABORATORY MANUAL
QUEST FOR EXCELLENCE	PRACTICAL EXPERIMENT INSTRUCTION SHEET	
	DEPARTMENT: COMPUTER SCIENCE AND ENGINEERING	

LABORATORY :		A.Y.: 2020-21
Class: SY CSE	PART: I	SUBJECT: DEMP

PROCEDURE :

A. HALF ADDER :

1. Make connection as shown in fig.
2. Apply +5 volt to pin 14 and GND to pin 7.
3. Apply the data to input A, B.
4. Switch on the instrument.
5. Observe the output on S, C using LED display.
6. Repeat the steps for other combinations of inputs and verify the truth table.

B. FULL ADDER :

1. Make connection as shown in fig.
2. Apply +5 volt to pin 14 and GND to pin 7.
3. Apply the data to input A, B and Cin.
4. Switch on the instrument.
5. Observe the output on S, C using LED display.
6. Repeat the steps for other combinations of inputs and verify the truth table.

CONCLUSION: In this way I have verified truth table for half and full adder.

Rubrics for Practical Assessment:

Cognitive (3)	Affective (3)	Psychomotor (3)	Total (9)

Sign of Course Teacher with Date

PREPARED BY : Dr. Geeta Tripathi	APPROVED BY : HCSED

	MAHARASHTRA INSTITUTE OF TECHNOLOGY AURANGABAD	LABORATORY MANUAL
QUEST FOR EXCELLENCE	PRACTICAL EXPERIMENT INSTRUCTION SHEET	
	DEPARTMENT: COMPUTER SCIENCE AND ENGINEERING	
LABORATORY :		A.Y.: 2020-21
Class: SY CSE	PART: I	SUBJECT: DEMP

EXPERIMENT NO.3

AIM: Realization of Half subtractor and Full subtractor using gates.

APPRATUS:

A. Component required for half subtractor.

Component	Quantity
1.ST2611 Digital Lab	1
2.IC7408 2-input AND GATE	1
3.IC7486 2-input EX-OR GATE	1
4.IC 7404 Hex inverter	1
5.2mm patch cords.	

B. Component required for full subtractor

Component	Quantity
1.ST2611 Digital Lab	1
2.IC7408 2-input AND GATE	1
3.IC7486 2-input EX-OR GATE	1
4.IC7432 2-input OR GATE	1
5.IC 7404 Hex inverter	1
6.2mm patch cords	

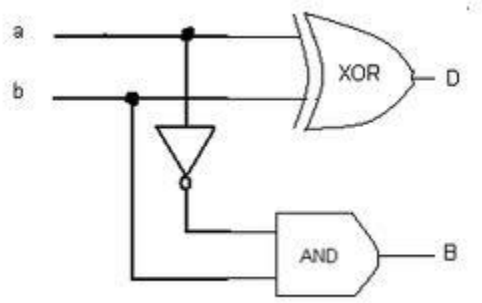
	MAHARASHTRA INSTITUTE OF TECHNOLOGY AURANGABAD	LABORATORY MANUAL
QUEST FOR EXCELLENCE	PRACTICAL EXPERIMENT INSTRUCTION SHEET	
	DEPARTMENT: COMPUTER SCIENCE AND ENGINEERING	

LABORATORY :		A.Y.: 2020-21
Class: SY CSE	PART: I	SUBJECT: DEMP

THEORY :

A.HALF SUBTRACTOR:

Half subtrctor is a combinational unit with 2-input X, Y and 2-output 'difference (0)','borrow (B)'namely. It can perform the subtractions of two binary bits, but while performing the subtraction it does not take into account the borrow of loIr significant stage.

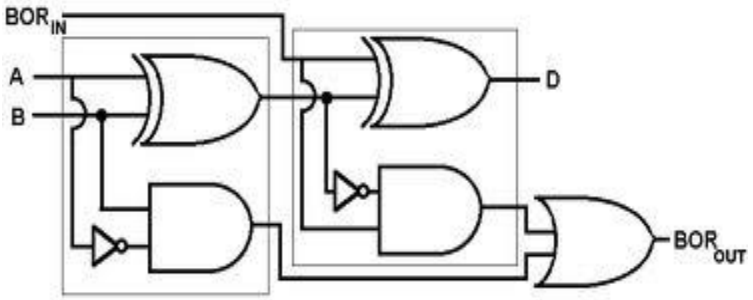


A	B	C	BORROW	DIFFERENCE
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	1	0
1	0	0	0	1
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

B. FULL SUBTRACTOR:

The full subtrctor is combinational with three input X,Yand borrow Bin.X is minuend Y IS SUBSTRCTED AND Bin borrow from the previous stage D is difference output and B is borrow output. It can be used in multi bit subtrctor.

PREPARED BY : Dr. Geeta Tripathi	APPROVED BY : HCSED



Inputs			Outputs	
a	*b*	*Bin*	Diff	Borrow
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

PROCEDURE:

A.HALF SUBTRCTOR:

1. Make connection as shown in fig.
2. Apply +5 volt to pin 14 and GND to pin 7.
3. Apply the data to input X,Y as shown in truth table.
4. Switch on the instrument.
5. Observe the output on D, B using LED display .
6. Repeat the steps for other combinations of inputs and verify the truth table.

B.FULL SUBTRCTOR:

1. Make connection as shown in fig.
2. Apply +5 volt to pin 14 and GND to pin 7.
3. Apply the data to input X, Y and Bor in.
4. Switch on the instrument.
5. Observe the output on D, B using LED display.
6. Repeat the steps for other combinations of inputs and verify the truth table.

CONCLUSION: In this way I have verified truth table for half and full subtractor.

Rubrics for Practical Assessment:

Cognitive (3)	Affective (3)	Psychomotor (3)	Total (9)

Sign of Course Teacher with Date

QUEST FOR EXCELLENCE	**MAHARASHTRA INSTITUTE OF TECHNOLOGY AURANGABAD**	**LABORATORY MANUAL**
	PRACTICAL EXPERIMENT INSTRUCTION SHEET	
	DEPARTMENT: COMPUTER SCIENCE AND ENGINEERING	
LABORATORY :		A.Y.: 2020-21
Class: SY CSE	PART: I	SUBJECT: DEMP

EXPERIMENT NO.4

Aim: Design & Implement 8:1 Multiplexer and 1:8 De-multiplexer using logic gates.

.**Theory:**

Part A: A Multiplexer (MUX) is a digital switch that has multiple inputs (sources) and a single output. The select/control lines determine which input is connected to the output. If the connected input is zero, the output is zero and if the connected input is one, output is one.

- A multiplexer has
  - N control inputs
  - 2^N data inputs
  - 1 output
- A multiplexer routes (or connects) the selected data input to the output.
  - The value of the control inputs determines the data input that is selected.

Following is the block diagram design of multiplexer:

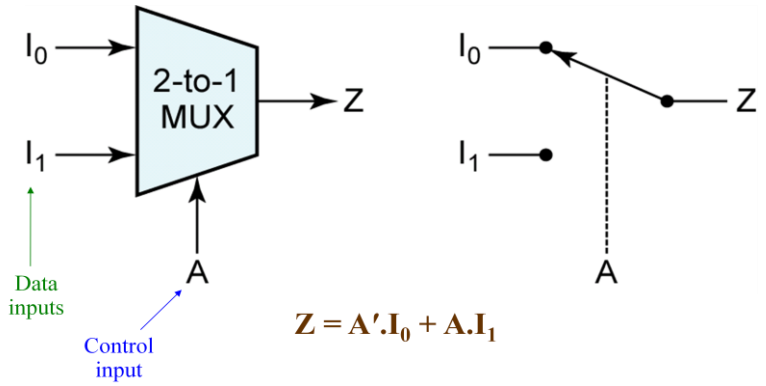
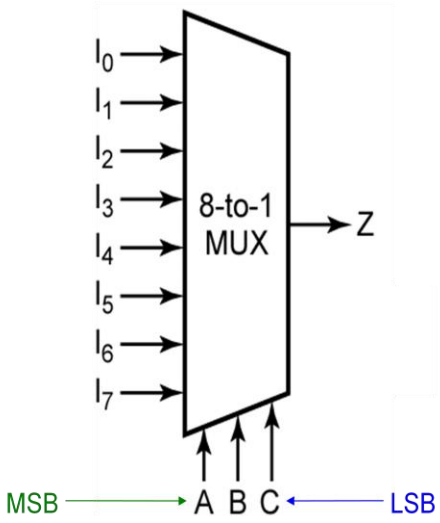


Figure A

For 8:1 MUX, there are 3 control inputs and eight data inputs and one output, as shown in the Figure B. The data on the output will be as per the truth table given in figure B.



Control lines			Output
A	B	C	Z
0	0	0	I_0
0	0	1	I_1
0	1	0	I_2
0	1	1	I_3
1	0	0	I_4
1	0	1	I_5
1	1	0	I_6
1	1	1	I_7

$$Z = A'.B'.C'.I_0 + A'.B'.C.I_1 + A'.B.C'.I_2 + A'.B.C.I_3 + A.B'.C'.I_0 + A.B'.C.I_1 + A'.B.C'.I_2 + A.B.C.I_3$$

Figure B

As per the Boolean Expression, implementation of the 8:1 MUX using basic gates can be done as shown in Figure C.

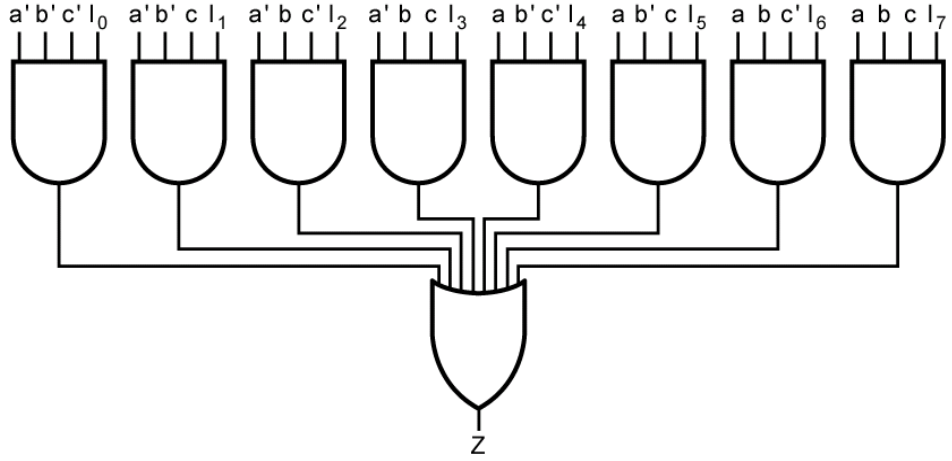


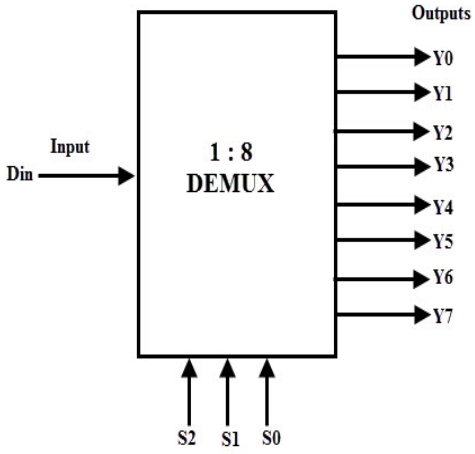
Figure C

Part B: A De-multiplexer (DE-MUX) has one input/data (sources) and multiple output lines.

The select/control lines determine which input is connected to the output. If the connected input is zero, the output is zero and if the connected input is one, output is one.

- A demultiplexer has
  - N control inputs
  - 1 data input
  - 2^N outputs
- A demultiplexer routes (or connects) the data input to the selected output.
  - The value of the control inputs determines the output that is selected.
- A demultiplexer performs the opposite function of a multiplexer.

Following is the block diagram design and truth table of demultiplexer as in Figure A.



Data Input	Select Inputs			Outputs							
D	S_2	S_1	S_0	Y_7	Y_6	Y_5	Y_4	Y_3	Y_2	Y_1	Y_0
D	0	0	0	0	0	0	0	0	0	0	D
D	0	0	1	0	0	0	0	0	0	D	0
D	0	1	0	0	0	0	0	0	D	0	0
D	0	1	1	0	0	0	0	D	0	0	0
D	1	0	0	0	0	0	D	0	0	0	0
D	1	0	1	0	0	D	0	0	0	0	0
D	1	1	0	0	D	0	0	0	0	0	0
D	1	1	1	D	0	0	0	0	0	0	0

Figure A

The circuit can be implemented using Basic logic gates as in figure B.

	MAHARASHTRA INSTITUTE OF TECHNOLOGY AURANGABAD	LABORATORY MANUAL
QUEST FOR EXCELLENCE	PRACTICAL EXPERIMENT INSTRUCTION SHEET	
	DEPARTMENT: COMPUTER SCIENCE AND ENGINEERING	

LABORATORY :		A.Y.: 2020-21
Class: SY CSE	PART: I	SUBJECT: DEMP

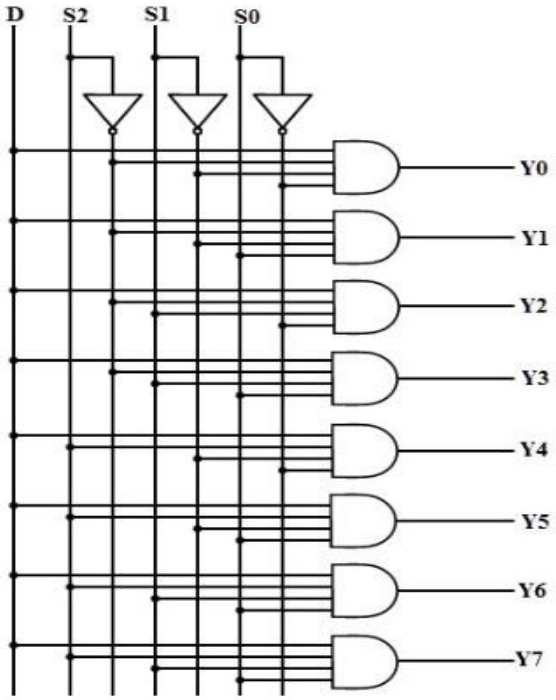


Figure B

Conclusion: The design and implementation of 8:1 MUX and 1:8DeMUX studied.

Rubrics for Practical Assessment:

Cognitive (3)	Affective (3)	Psychomotor (3)	Total (9)

Sign of Course Teacher with Date

PREPARED BY : Dr. Geeta Tripathi	APPROVED BY : HCSED

EXPERIMENT NO. 5

Aim: Demonstrate the working of flip-flop.

Theory:

A flip flop is an electronic circuit with two stable states that can be used to store binary data. The stored data can be changed by applying varying inputs. Flip-flops and latches are fundamental building blocks of digital electronics systems used in computers, communications, and many other types of systems.

1) R-S flip flop
2) D flip flop
3) J-K flip flop
4) T flip flop

1) R-S flip flop

The basic NAND gate RS flip flop circuit is used to store the data and thus provides feedback from both of its outputs again back to its inputs. The RS flip flop actually has three inputs, SET, RESET and its current output Q relating to its current state as shown in figure below.

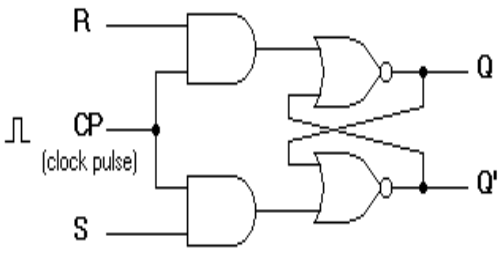


Figure-1:R-S flip flop circuit diagram

INPUTS			OUTPUT	STATE
CLK	S	R	Q	
X	0	0	No Change	Previous
↑	0	1	0	Reset
↑	1	0	1	Set
↑	1	1	-	Forbidden

Figure-2:Characteristics table of R-S flip flop

2) D flip flop

A D flip flop has a single data input. This type of flip flop is obtained from the SR flip flop by connecting the R input through an inverter, and the S input is connected directly to data input. The modified clocked SR flip-flop is known as D-flip-flop and is shown below. From the truth table of SR flip-flop we see that the output of the SR flip-flop is in unpredictable state when the inputs are same and high. In many practical applications, these input conditions are not required. These input conditions can be avoided by making them complement of each other.

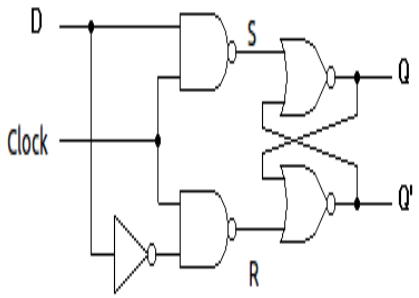


Figure-3:Circuit diagram of D flip flop

Input			Output	
D	reset	clock	Q	Q'
0	0	0	0	1
0	0	1	0	1
0	1	0	0	1
0	1	1	0	1
1	0	0	0	1
1	0	1	1	0
1	1	0	0	1
1	1	1	0	1

Figure-4:Characteristics table of D flip flop

3) J-K flip flop

In a RS flip-flop the input R=S=1 leads to an indeterminate output. The RS flip-flop circuit may be re-joined if both inputs are 1 than also the outputs are complement of each other as shown in characteristics table below.

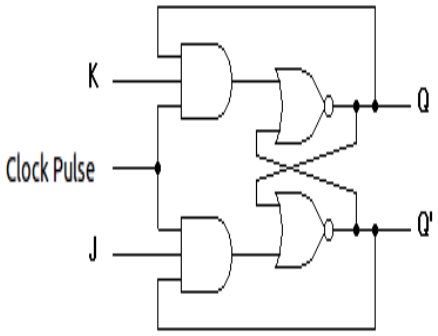


Figure-5:Circuit diagram of J-K flip flop

Trigger	Inputs		Output				Inference
			Present State		Next State		
CLK	J	K	Q	Q'	Q	Q'	
⊠	x	x	-		-		Latched
⊓	0	0	0	1	0	1	No Change
⊓			1	0	1	0	
⊓	0	1	0	1	0	1	Reset
⊓			1	0	0	1	
⊓	1	0	0	1	1	0	Set
⊓			1	0	1	0	
⊓	1	1	0	1	1	0	Toggles
⊓			1	0	0	1	

Figure-6:Characteristics table of J-K flip flop

4) T flip flop

T flip-flop is known as toggle flip-flop. The T flip-flop is modification of the J-K flip-flop. Both the JK inputs of the JK flip – flop are held at logic 1 and the clock signal continuous to change as shown in table below.

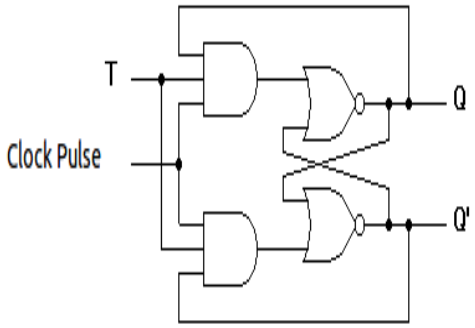


Figure-7:Circuit diagram of T flip flop

T flip-flop

T	Clock	Q	Q'
0	↑	Q	Q'
1	↑	Q'	Q
X	↓	Q	Q'

Figure-8:Characteristics table of T flip flop

Conclusion: The demonstration for the four flip flops done.

EXPERIMENT NO.6

Aim: Write Assembly language program to print the string in 8086.

Requirement:

1. TASM or emu8086 Software

2. Pentium – 4 PC

3. Data transfer and copy instruction

Theory:

Initialization of variables:

Variables are declared and initialized in the data segment part of segment register. For string ie for data string it is initialized as data byte and string specified such as 0dh,0ah,'$' are also declared.

Code Segment:

1) Assume CS as code segment and DS as data segment.

2) Data segment is initialized using the following instructions-

```
Mov ax , data

Mov ds , ax
```

3) Then to print message we have use dos prompt function that is display routine that print string from DS:DX

```
Mov dx, offset msg

Mov ah,09h

Int 21h
```

4) Then for returning back from dos prompt we have to call clear screen routine

Mov ah , 4ch

Int 21h

Conclusion: Thus, we have studied & executed assembly language program using TASM to print the string

Rubrics for Practical Assessment:

Cognitive (3)	Affective (3)	Psychomotor (3)	Total (9)

Sign of Course Teacher with Date

EXPERIMENT NO.7

Aim: Write Assembly language program for 8 bit addition and 16 bit addition in 8086.

Objective:- a) For 8-bit addition-

a1=01h

a2=02h

store the result in a3(ASCII)=a1+a2

b) For 16-bit addition-
b1=0001h

b2=0002h

store the result in b3(ASCII)=b1+b2

Requirements:-1) TASM or emu8086 software

2) Pentium-4 PC

3) Data and arithmetic instructions

Theory:-

Initialization of variables: Variables are initialized in the data segment part of segment register. For 8-bit data, it is initialized as data byte (DB) and for 16-bit data it is initialized as data word (DW).

Code Segment:-

1) Assume cs as code segment and ds as data segment.
2) Data segment is initialized using the following instructions-
   Mov ax , data
   Mov ds , ax
3) Then we move the first variable into al(for 8-bit dat) or in ax(for 16-bit data)
   i.e. mov al , a1

   or mov ax , b1

4) Add the second variable to the first.
   i.e. add al , a2

   or add ax , b2

5) Pass the result to the third variable.
   i.e. mov a3 , al

   or mov b3 , ax

6) Display Routine-
   Mov dx , a3

   Mov ah , 02h

   Int 21h

7) Clear Screen Routine-
   Mov ah , 4ch

   Int 21h

Conclusion: Thus we have studied and executed the ALP for addition of two 8 bit and two 16 bit numbers in 8086 using TASM/emu8086.

Rubrics for Practical Assessment:

Cognitive (3)	Affective (3)	Psychomotor (3)	Total (9)

Sign of Course Teacher with Date

	MAHARASHTRA INSTITUTE OF TECHNOLOGY AURANGABAD	LABORATORY MANUAL
QUEST FOR EXCELLENCE	PRACTICAL EXPERIMENT INSTRUCTION SHEET	
	DEPARTMENT: COMPUTER SCIENCE AND ENGINEERING	
LABORATORY :		A.Y.: 2020-21
Class: SY CSE	PART: I	SUBJECT: DEMP

EXPERIMENT NO.8

Aim : Write Assembly language program for 8 bit subtraction and 16 bit subtraction in 8086.

Objective:-

a) For 8-bit Subtraction-
   a1=01h

   a2=02h

   store the result in a3(ASCII)=a1-a2

b) For 16-bit Subtraction-
   b1=0002h

   b2=0001h

   store the result in b3(ASCII)=b1-b2

Requirements:-

1) TASM or emu8086 software

2) Pentium-4 PC

3) Data and arithmetic instructions

Theory:-

Initialization of variables-

Variables are initialized in the data segment part of segment register. For 8-bit data, it is initialized as data byte (DB) and for 16-bit data it is initialized as data word (DW).

Code Segment:-

1) Assume cs as code segment and ds as data segment.

2) Data segment is initialized using the following instructions-

   Mov ax , data

   Mov ds , ax

3) Then we move the first variable into al(for 8-bit data) or in ax(for 16-bit data)

   mov al, a1

   or mov ax , b1

4) Subtract the second variable from the first.
   i.e. sub al , a2

   or sub ax , b2

   Mov dx , result

   Mov dx , result

5) Pass the result to the third variable.
   i.e. mov result , al

   or mov result , ax

6) Display Routine-

Mov ah , 02h

Int 21h

7) Clear Screen Routine-

Mov ah , 4ch

Int 21h

Finally the code ends

i.e. code ends

end

Conclusion: Thus we have Studied and executed the assembly language program for subtraction of two 8 bit and 16 bit numbers using TASM or emu8086.

Rubrics for Practical Assessment:

Cognitive (3)	Affective (3)	Psychomotor (3)	Total (9)

Sign of Course Teacher with Date

EXPERIMENT NO.9

Aim : Write Assembly language program for 8/16 bit multiplication in 8086 using TASM/emu8086.

Objective:-

a) For 8-bit multiplication-
   a1=01h

   a2=02h

   store the result in a3(ASCII)=a1-a2

b) For 16-bit multiplication-
   b1=0002h

   b2=0001h

   store the result in b3(ASCII)=b1-b2

Requirements:-

1) TASM/emu8086 software

2) Pentium-4 PC

3) Data and arithmetic instructions

Theory:-

Initialization of variables-

Variables are initialized in the data segment part of segment register. For 8-bit data, it is initialized as data byte (DB) and for 16-bit data it is initialized as data word (DW).

Code Segment:-

1) Assume cs as code segment and ds as data segment.
2) Data segment is initialized using the following instructions-
   Mov ax , data

   Mov ds , ax

3) Then we move the first variable into al(for 8-bit data) or in ax(for 16-bit data)
   mov al, num1

   or mov ax , num1

4) Multiply the second variable with the first variable.
   i.e. mul al , num2

   or mul ax, num2

5) Pass the result to the third variable.
   i.e. mov result , al

   or mov result , ax

6) Display Routine-
   Mov dx , result

   Mov ah , 02h

   Int 21h

7) Clear Screen Routine-
   Mov ah , 4ch

   Int 21h

Finally the code ends

i.e. code ends

end

Conclusion: Thus we have Studied and performed the assembly language program for multiplication of two 16 bit numbers using TASM/emu8086.

Rubrics for Practical Assessment:

Cognitive (3)	Affective (3)	Psychomotor (3)	Total (9)

Sign of Course Teacher with Date

	MAHARASHTRA INSTITUTE OF TECHNOLOGY AURANGABAD	LABORATORY MANUAL
QUEST FOR EXCELLENCE	PRACTICAL EXPERIMENT INSTRUCTION SHEET	
	DEPARTMENT: COMPUTER SCIENCE AND ENGINEERING	
LABORATORY :		A.Y.: 2020-21
Class: SY CSE	PART: I	SUBJECT: DEMP

EXPERIMENT NO.10

Aim : Write an Assembly Language Program for finding smallest number from an array in 8086

Objective:-

To find smallest number from a list

List = 56h, 05h, 61h

Store result in variable called as small

Requirements:-

1) TASM/emu8086 software

2) Pentium-4 PC

3) Data and arithmetic instructions

Theory:-

Initialization of variables-

Array is initialized in the data segment part of segment register. All array elements are 8 bit so array initialized with 8 bit and initialize count by number of elements in array – 1.

Code Segment:

1) Assume CS as code segment and DS as data segment.
2) Data segment is initialized using the following instructions-
   Mov ax , data

   Mov ds , ax

3) Then store the offset of the array list into SI means SI will point to the first memory location of the array.

4) Store the content of the location which is pointed by SI that is the first element of the array into AL register.
5) Move count into CL register.
6) Use loop to find out the smallest from the array for that first compare the first element with the next element in an array then while comparing there are two cases:
   i) carry is generated: if carry is generated means first element is smaller than second,so move that second element into first element ie in AL.
   ii) carry is not generated: if carry is not generated then jump on skip routine
7) Then increment SI so that it points to next element and decrement CL that is count by 1.

8) Then check whether CL ie count become zero or not.if it is not zero repeat the loop till it become
zero.

Conclusion: Thus we have studied and executed the ALP for finding the smallest number from given unordered array using TASM/emu8086

Rubrics for Practical Assessment:

Cognitive (3)	Affective (3)	Psychomotor (3)	Total (9)

Sign of Course Teacher with Date