



Parul
University



Parul University

FACULTY OF ENGINEERING AND TECHNOLOGY
BACHELOR OF TECHNOLOGY

COMA-LAB (303105210)

4th Semester

Computer Science Department

Laboratory Manual



CERTIFICATE

This is to certify that

*Mr./Ms Bharat sinh Rathod with enrolment no.
2303051051232 has successfully completed his laboratory
experiments in the from the
COMA-LAB (303105210) department of Computer
Engineering & Science during the academic year
.....**2024-25**.....*



Date of Submission:

Staff In charge:.....

Head of Department.....

INDEX

Sr. No.	Date	Experiment Name	Page No.	Sign
1	25/11/24	Write a program to add two 8-bit numbers.	2-3	
2	02/12/24	Write a program to find one's complement and two's complement of 8-bit numbers.	4-5	
3	16/12/24	Write a program to perform 16-bit addition of two numbers.	6-7	
4	23/12/24	Write a program to multiply two 8-bit numbers using addition.	8-9	
5	06/01/25	Write an ALP to transfer a block of data from memory location 2010H to 2080H.	10-12	
6	20/01/25	Write a program to perform addition of 6 bytes of data stored at memory location starting from 2050H. Use register B to save carry generated while performing addition. Display sum and carry at consecutive locations 2070H and 2071H.	13-14	
7	10/02/25	[A] Write a program to find the largest number of given two 8-bit numbers at 2050H & 2051H memory location. Store the result at 2060H. [B] Write a program to find the largest number in a set of 8 readings stored at 2050H. Display the number at 2060H.	15-20	
8	10/02/25	Write a program to arrange the numbers in ascending order. The numbers are stored at 2050H onwards. [5 numbers]	21-23	
9	17/02/25	Write a program to convert a number from BCD to Binary.	24-26	
10	17/02/25	Write a program to convert from binary to ASCII	27-29	
11	03/03/25	Introduction to 8086 Microprocessor.	30-32	

CSE Semester - IV

PRACTICAL-1

AIM: Write a program to add two 8-bit numbers and store result at memory location / using LDA .

PROGRAM (1A):

```
MVI A,35H
```

```
MVI B,12H
```

```
ADD B
```

```
MOV C,A
```

```
STA 2060H
```

```
HLT
```

OUTPUT:

The screenshot displays an 8085 assembly simulator interface. On the left, the **Registers** panel shows the status of A/PSW, BC, DE, HL, SP, and PC. The **Flags** panel shows Z, S, P, C, and AC flags. The central pane shows the **main.asm** code being loaded at 0x0800. On the right, the **Memory View** shows a hex dump of memory starting at 0x1100. At the bottom, the **Assembler Output** panel shows the assembly process for each instruction.

Register	Value
A/PSW	0x4706
BC	0x1247
DE	0x0000
HL	0x0000
SP	0xFF FF
PC	0x0800

Flag	Status
Z	<input type="checkbox"/>
S	<input type="checkbox"/>
P	<input checked="" type="checkbox"/>
C	<input type="checkbox"/>
AC	<input type="checkbox"/>


```

main.asm
1 MVI A,35H
2 MVI B,12H
3 ADD B
4 MOV C,A
5 STA 2060H
6 HLT
7
  
```


Address	Hex	ASCII
200	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
201	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
202	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
203	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
204	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
205	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
206	47 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
207	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
208	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
209	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
20A	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
20B	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
20C	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
20D	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
20E	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
20F	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	

Line	Address	Hex	Instruction
1	3E 35		MVI A,35H
2	06 12		MVI B,12H
3	80		ADD B
4	4F		MOV C,A
5	32 60 20		STA 2060H
6	76		HLT
7			

CSE Semester - IV

PROGRAM (1-B)

```
LDA 2051H
MOV B,A
LDA 2052H
ADD B
MOV C,A
STA 2060H
HLT
```

Conclusion:-

This practical demonstrated how to add two 8-bit numbers using registers and store the result in memory. It highlighted the use of instructions like MVI, ADD, and STA to perform arithmetic operations and memory management.

CSE Semester - IV

PRACTICAL-2

AIM: Write a program to find one's complement and two's complement of 8-bit numbers.

PROGRAM:

```

MVI A,6FH
CMA
STA 2051H
ADI 01H
STA 2052H
HLT

```

OUTPUT:

The screenshot displays an 8086 assembly simulator interface. On the left, the **Registers** window shows the state of various registers: A/PSW (0x9182), BC (0x0000), DE (0x0000), HL (0x0000), SP (0xFFFF), and PC (0x0800). Below this, the **Flags** window shows the status of Z, S, P, C, and AC flags, with the Sign flag (S) being set. The central pane shows the **main.asm** source code with line numbers 1 through 7. On the right, the **Memory View** window shows a memory dump starting at address 200, with the value 90 91 at address 205, indicating the one's complement of 6FH. At the bottom, the **Assembler Output** window shows the machine code generated for each instruction: 3E 6F for MVI A,6FH; 2F for CMA; 32 51 20 for STA 2051H; C6 01 for ADI 01H; 32 52 20 for STA 2052H; and 76 for HLT.

Conclusion:-

The program successfully computed the one's complement (using CMA) and two's complement (by adding 1 with ADI) of an 8-bit number. It emphasized understanding binary arithmetic operations and their applications in digital systems.

CSE Semester - IV

PRACTICAL-3**AIM:** Write a program to perform 16-bit addition of two numbers.**PROGRAM:****(3A)**

```

LHLD 2050H
XCHG
LHLD 2052H
DAD D
SHLD 2060H
HLT

```

Input:2050H:

1234H

2052H:

1312H

Output:

The screenshot displays an 8085 assembly simulator interface. On the left, the **Registers** panel shows the following values: A/PSW (0x0002), BC (0x0000), DE (0x0034), HL (0x0046), SP (0xFFFF), and PC (0x080C). Below this, the **Flags** panel shows Z, S, P, C, and AC flags, all of which are currently cleared (indicated by empty checkboxes). The central panel shows the assembly code for `main.asm`:

```

1 LHLD 2050H
2 XCHG
3 LHLD 2052H
4 DAD D
5 SHLD 2060H
6 HLT
7
8

```

On the right, the **Memory View** panel shows a memory dump starting at address 200. The address 2050 is highlighted in the address field. The memory dump shows the following values at the relevant addresses:

Address	Value
2050	34
2052	12
2060	46

At the bottom right, the **Start Address** is set to 0x1100.

CSE Semester - IV

PROGRAM (3B)

```

LXI H,1234H
LXI B,1312H
DAD B
SHLD 2060H
HLT

```

The screenshot displays an 8085 assembly simulator interface. On the left, the **Registers** panel shows the state of various registers: A/PSW (0x0002), BC (0x1312), DE (0x0000), HL (0x2546), SP (0xFFFF), and PC (0x0800). Below this, the **Flags** panel shows Z, S, P, C, and AC flags, all of which are currently cleared (indicated by empty checkboxes). The central area shows the assembly code for `main.asm` with five instructions: `LXI H,1234H`, `LXI B,1312H`, `DAD B`, `SHLD 2060H`, and `HLT`. On the right, the **Memory View** panel shows a memory dump starting at address 200. Address 206 is highlighted, showing the value 46 25. At the bottom, the **Assembler Output** panel shows the machine code generated for each instruction: `21 34 12 LXI H,1234H`, `01 12 13 LXI B,1312H`, `09 DAD B`, `22 60 20 SHLD 2060H`, and `76 HLT`.

Conclusion:-

This practical illustrated 16-bit addition using register pairs (e.g., LHLD, DAD) to handle larger numbers. It showcased the 8085's ability to manipulate 16-bit data through combined register operations.

PRACTICAL-4**AIM:** Write a program to multiply two 8-bit numbers using addition.**PROGRAM:** To multiply two numbers: 4 and 3.

```

MVI A,00H
MVI B,04H
ADD B
ADD B
ADD B
STA 2060H
HLT

```

OUTPUT:

2060H: 0CH

Registers

Register	Value
AX/PSW	0x0C06
BC	0x0400
DE	0x0000
HL	0x0000
SP	0xFFFF
PC	0x0800

Flags

Z	<input type="checkbox"/>
S	<input type="checkbox"/>
P	<input checked="" type="checkbox"/>
C	<input type="checkbox"/>
AC	<input type="checkbox"/>

main.asm

```

1  MVI A,00H
2  MVI B,04H
3  ADD B
4  ADD B
5  ADD B
6  STA 2060H
7  HLT

```

Memory View

Address	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
200	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
201	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
202	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
203	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
204	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
205	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
206	0C	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
207	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
208	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
209	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
20A	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
20B	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
20C	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
20D	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
20E	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
20F	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00

Start Address at: 0x 1100

Assembler Output

Line	Address	Instruction
1	3E 00	MVI A,00H
2	06 04	MVI B,04H
3	80	ADD B
4	80	ADD B
5	80	ADD B
6	32 60 20	STA 2060H
7	76	HLT

CSE Semester - IV

USING LOOPING :-

```
MVI A,00H
MVI B,04H
MVI C,03H
NXT:ADD B
DCR C
JNZ NXT
STA 2060H
HLT
```

Conclusion:-

Two methods were explored: repeated addition and looping to multiply 8-bit numbers. The practical demonstrated basic multiplication logic and emphasized efficiency through loop-based approaches.

PRACTICAL-5

AIM: Write an ALP to transfer a block of data from memory location 2050H to 2060H.

PROGRAM:

```
LXI H, 2050H
LXI D,2060H
MVI C,08H
NXT:MOV A,M
STAX D
INX H
INX D
DCR C
JNZ NXT
HLT
```

INPUT: 2050H:05H
 2051H:
 09H
 2052H:
 07H
 2053H:
 02H
 2054H:
 04H
 2055H:
 05H
 2056H:
 03H
 2057H:
 01H

CSE Semester - IV

OUTPUT:

2060H:05H
 2061H:
 09H
 2062H:
 07H
 2063H:
 02H
 2064H:
 04H
 2065H:
 05H
 2066H:
 03H
 2067H:
 01H

Registers

A/PSW	0x 01 56
BC	0x 00 00
DE	0x 20 68
HL	0x 20 58
SP	0x FF FF
PC	0x 08 11

Flags

Z	<input checked="" type="checkbox"/>
S	<input type="checkbox"/>
P	<input checked="" type="checkbox"/>
C	<input type="checkbox"/>
AC	<input checked="" type="checkbox"/>

main.asm

```

1 LXI H,2050H
2 LXI D,2060H
3 MVI C,08H
4 NXT:MOV A,M
5 STAX D
6 INX H
7 INX D
8 DCR C
9 JNZ NXT
10 HLT

```

Memory View

0x 2050

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
200	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
201	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
202	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
203	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
204	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
205	05	09	07	02	04	05	03	01	00	00	00	00	00	00	00	00
206	05	09	07	02	04	05	03	01	00	00	00	00	00	00	00	00
207	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
208	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
209	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
20A	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
20B	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
20C	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
20D	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
20E	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
20F	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00

Start Address at: 0x 1100

Conclusion:-

A block of data was transferred between memory locations using pointer registers (LXI H, LXI D) and loops. This highlighted efficient memory management and bulk data operations in assembly.

PRACTICAL-6

AIM: Write a program to perform addition of 6 bytes of data stored at memory location starting from 2050H. Use register B to save carry generated while performing addition. Display sum and carry at consecutive locations 2060H and 2061H.

PROGRAM:

```
LXI H,2050H
MVI C,06H
MVI A,00H
MOV B,A
NXT1: ADD M
      JNC NXT
      INR B
NXT: INX H
      DCR C
      JNZ NXT1
      STA 2060H
      MOV A,B
      STA 2061H
      HLT
```

CSE Semester - IV

INPUT: 2050H: 01H
 2051H: 02H
 2052H: F1
 2053H: 04H
 2054H: 05H
 2055H: F1

OUTPUT: 2060H: EE ,
 2061H: 01

The screenshot displays an 8085 assembly simulator interface. On the left, the **Registers** panel shows A/PSW at 0x0157, BC at 0x0100, DE at 0x0000, HL at 0x2056, SP at 0xFFFF, and PC at 0x081A. Below it, the **Flags** panel shows Z, S, P, C, and AC flags, with Z, P, C, and AC checked. The central **main.asm** window contains the following assembly code:

```

1 LXI H,2050H
2 MVI C,06H
3 MVI A,00H
4 MOV B,A
5 NXT1:ADD M
6 JNC NXT
7 INR B
8 NXT:INX H
9 DCR C
10 JNZ NXT1
11 STA 2060H
12 MOV A,B
13 STA 2061H
14 HLT
  
```

On the right, the **Memory View** panel shows a hex dump of memory starting at address 200. The input data is visible at addresses 2050-2055: 01, 02, F1, 04, 05, F1. The output data is visible at addresses 2060-2061: EE, 01. The Start Address is set to 0x1100.

Conclusion:-

The program added six bytes of data while tracking carry using register B. It emphasized multi-byte addition, carry handling, and storing results in consecutive memory locations.

PRACTICAL-7**(A)**

AIM: Write a program to find the largest number of given two 8-bit numbers at 2050H & 2051H memory location. Store the result at 2060H.

PROGRAM:

```
LXI H,2050H
MOV A,M
INX H
CMP M
JNC NXT
MOV A,M
NXT:STA 2060H
HLT
```

INPUT: 2050H: 6AH
 2051H: F4H

OUTPUT: 2060H: F4H

CSE Semester - IV

Registers ↻

A/PSW	0x F4 13
BC	0x 00 00
DE	0x 00 00
HL	0x 20 51
SP	0x FF FF
PC	0x 08 0E

Flags ↻

Z	<input type="checkbox"/>
S	<input type="checkbox"/>
P	<input type="checkbox"/>
C	<input checked="" type="checkbox"/>
AC	<input checked="" type="checkbox"/>

main.asm

```

1 LXI H,2050H
2 MOV A,M
3 INX H
4 CMP M
5 JNC NXT
6 MOV A,M
7 NXT:STA 2060H
8 HLT
9

```

Memory View ↻

0x 2050

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
200	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
201	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
202	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
203	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
204	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
205	6A	F4	00	00	00	00	00	00	00	00	00	00	00	00	00	00
206	F4	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
207	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
208	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
209	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
20A	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
20B	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
20C	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
20D	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
20E	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
20F	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00

Start Address at: 0x 1100

Conclusion:-

This practical compared two 8-bit numbers using CMP and stored the larger value. It demonstrated conditional branching (JNC) for decision-making in algorithms.

CSE Semester - IV

PRACTICAL-7
(B)

AIM: Write a program to find the largest number in a set of 8 readings stored at 2050H.
Display the number at 2060H.

PROGRAM:

```
LXI H,2050H
MVI C,08H
MOV A,M
NXT1:INX H
CMP M
JNC NXT
MOV A,M
NXT:DCR C
JNZ NXT1
HLT
STA 2060H
```

OBSERVATIONS:**Input:**

```
2050H: 22H
2051H: 2FH
2052H: 6DH
2053H: 13H
2054H: 2AH
2055H: 9EH
2056H: EAH
2057H: 08H
```

Output:

```
2060H: EAH
```

CSE Semester - IV

Registers

A/PSW	0x EA 56
BC	0x 00 00
DE	0x 00 00
HL	0x 20 58
SP	0x FF FF
PC	0x 08 14

Flags

Z	<input checked="" type="checkbox"/>
S	<input type="checkbox"/>
P	<input checked="" type="checkbox"/>
C	<input type="checkbox"/>
AC	<input checked="" type="checkbox"/>

main.asm

```

1 LXI H,2050H
2 MVI C,08H
3 MOV A,M
4 NXT1:INX H
5 CMP M
6 JNC NXT
7 MOV A,M
8 NXT:DCR C
9 JNZ NXT1
10 STA 2060H
11 HLT

```

Memory View

0x 2050

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
200	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
201	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
202	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
203	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
204	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
205	22	2F	6D	30	2A	9E	EA	08	00	00	00	00	00	00	00	00
206	EA	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
207	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
208	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
209	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
20A	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
20B	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
20C	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
20D	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
20E	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
20F	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00

Start Address at: 0x 1100

Conclusion:-

A loop-based approach was used to find the largest number in a dataset. It showcased iterative comparison and the use of counters (DCR C) for processing arrays.

PRACTICAL-8

AIM: Write a program to arrange the numbers in ascending order. The numbers are stored at 2050H onwards. [9 numbers]

PROGRAM:

```
        MVI B, 08H
START: LXI H, 2050H
        MVI C, 08H
BACK: MOV A, M
        INX H
        CMP M
        JC SKIP
        JZ SKIP
        MOV D, M
        MOV M, A
        DCX H
        MOV M, D
        INX H
SKIP: DCR C
        JNC BACK
        DCR B
        JNZ START
        HLT
```

INPUT: 2050H: 05H
2051H: 04H
2052H: 03H
2053H: 01H
2054H: 02H
2055H: 07H
2056H: 0DH
2057H: 0AH

OUTPUT: 2050H: 01H
2051H: 02H
2052H: 03H
2053H: 04H
2054H: 05H
2055H: 07H
2056H: 0DH
2057H: 0AH

CSE Semester - IV

Registers

A/PSW	0x0157
BC	0x0007
DE	0x0200
HL	0x2051
SP	0xFFFF
PC	0x081E

Flags

Z	<input checked="" type="checkbox"/>
S	<input type="checkbox"/>
P	<input checked="" type="checkbox"/>
C	<input checked="" type="checkbox"/>
AC	<input checked="" type="checkbox"/>

Load at 0x0800

main.asm

```

1 MVI B, 08H
2 START: LXI H, 2050H
3 MVI C, 08H
4 BACK: MOV A, M
5 INX H
6 CMP M
7 JC SKIP
8 JZ SKIP
9 MOV D, M
10 MOV M, A
11 DCX H
12 MOV M, D
13 INX H
14 SKIP: DCR C
15 JNC BACK
16 DCR B
17 JNZ START
18 HLT
19
20

```

Memory View

0x

2050

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
200	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
201	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
202	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
203	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
204	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
205	01	02	03	04	05	07	0A	0D	00	00	00	00	00	00	00	00
206	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
207	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
208	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
209	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
20A	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
20B	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
20C	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
20D	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
20E	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
20F	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00

Start Address at: 0x 1100

Conclusion:-

The program implemented a bubble sort algorithm to arrange numbers in ascending order. It emphasized nested loops and swapping logic for sorting operations in assembly.

PRACTICAL-9

AIM: Write a program to convert a number from BCD to Binary.

PROGRAM:

```
LDA 2010H
MOV B,A
ANI 0FH
MOV C,A
MOV A,B
ANI 0FH
JZ SKIPMUL
RRC
RRC
RRC
RRC
MOV D,A
XRA A
MVI E,0AH
SUM:ADD D
DCR E
JNZ SUM
SKIPMUL:ADD C
STA 2020H
HLT
```

INPUT: 2010H: 72H

OUTPUT: 2020H: 48H

CSE Semester - IV

The screenshot displays an 8085 assembly simulator interface. On the left, the **Registers** section shows: A/PSW at 0x4816, BC at 0x7202, DE at 0x0700, HL at 0x0000, SP at 0xFFFF, and PC at 0x081F. Below, the **Flags** section shows Z, S, and C as 0, while P and AC are 1. The central **main.asm** window contains 20 lines of assembly code, with line 14 (MVI E, 0AH) highlighted. On the right, the **Memory View** shows a hex dump starting at address 200, with address 202 highlighted containing the value 48. At the bottom right, the **Start Address** is set to 0x1100.

Conclusion:-

BCD-to-binary conversion was achieved using masking (ANI), shifting (RRC), and multiplication via repeated addition. The practical highlighted numeral system conversion techniques.

PRACTICAL-10

AIM: Write a program to convert from binary to ASCII.

PROGRAM:

```
LXI H,8000H
MOV A,M
MOV B,A
STC
CMC
SUI 0AH
JC NUM
ADI 41H
JMP STORE
NUM: MOV A,B
ADI 30H
STORE: INX H
MOV M,A
HLT
```

INPUT: 8000H: 0FH

OUTPUT: 8001H: 46H

CSE Semester - IV

The screenshot displays a 68000 assembly simulator interface. On the left, the **Registers** panel shows the state of various registers: A/PSW (0x4612), BC (0x0F00), DE (0x0000), HL (0x8001), SP (0xFFFF), and PC (0x0817). Below this, the **Flags** panel shows Z, S, P, C, and AC (checked). The central panel displays the assembly code for `main.asm`, which includes instructions like `LXI H,8000H`, `MOV A,M`, `MOV B,A`, `STC`, `CMC`, `SUI 0AH`, `JC NUM`, `ADI 41H`, `JMP STORE`, `NUM: MOV A,B`, `ADI 30H`, `STORE: INX H`, `MOV M,A`, and `HLT`. On the right, the **Memory View** panel shows a hexadecimal dump of memory starting at address 800. The first byte at address 800 is 0F, and the second byte is 46, which corresponds to the value in the A/PSW register. The rest of the memory is filled with zeros. A slider at the bottom indicates the start address is 7700.

Conclusion:-

Binary-to-ASCII conversion was performed by checking if the value was a digit (0-9) or letter (A-F). This demonstrated conditional logic and ASCII encoding fundamentals.

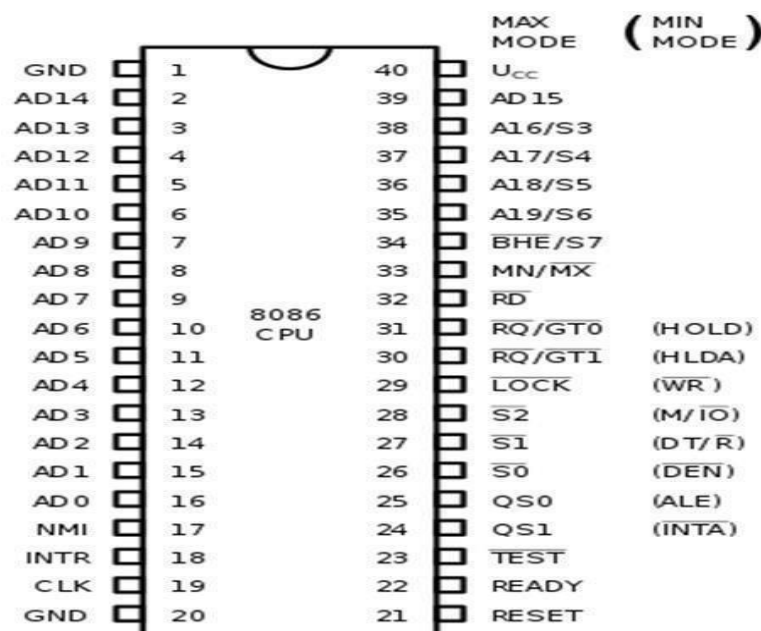
PRACTICAL-11

AIM: Introduction to 8086 Microprocessor.

Features of 8086

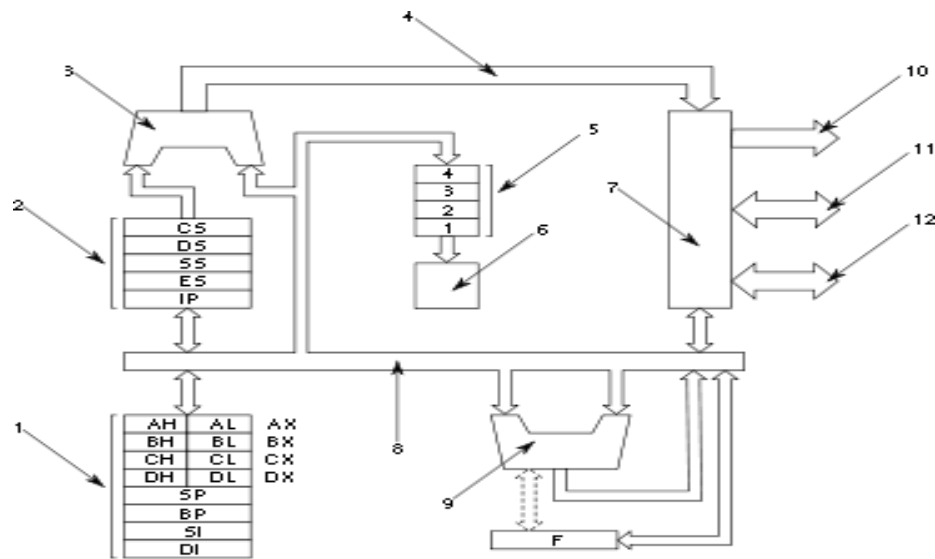
- 8086 is a 16bit processor. It's ALU, internal registers works with 16bit binary word.
- 8086 has a 16bit data bus. It can read or write data to a memory/port either 16bits or 8 bit at a time
- 8086 has a 20bit address bus which means, it can address upto 1MBmemory location
- Frequency range of 8086 is 6-10 MH.

Pin Diagram of 8086Microprocessor



Architecture of 8086 Microprocessor

1=main & index registers; 2=segment registers and IP; 3=address adder; 4=internal address bus; 5=instruction queue; 6=control unit (very simplified!); 7=bus interface; 8=internal databus; 9=ALU; 10/11/12=external address/data/control bus.



Registers

The 8086 has eight more or less general 16-bit registers (including the stack pointer but excluding the instruction pointer, flag register and segment registers). Four of them, AX, BX, CX, DX, can also be accessed as twice as many 8-bit registers while the other four, BP, SI, DI, SP, are 16-bit only.

A 64 KB (one segment) stack growing towards lower addresses is supported in hardware; 16-bit words are pushed onto the stack, and the top of the stack is pointed to by SS:SP. There are 256 interrupts, which can be invoked by both hardware and software. The interrupts can cascade, using the stack to store the return addresses.

The 8086 has 64 K of 8-bit (or alternatively 32 K of 16-bit word) I/O port space.

Flags

8086 has a 16-bit flags register. Nine of these condition code flags are active, and indicate the current state of the processor: Carry flag (CF), Parity flag (PF), Auxiliary carry flag (AF), Zero flag (ZF), Sign flag (SF), Trap flag (TF), Interrupt flag (IF), Direction flag (DF), and Overflow flag (OF).

Segmentation

There are also four 16-bit segment registers that allow the 8086 CPU to access one megabyte of memory in an unusual way. Rather than concatenating the segment register with the address register, as in most processors whose address space exceeded their register size, the 8086 shifts the 16-bit segment only four bits left before adding it to the 16-bit offset ($16 \times \text{segment} + \text{offset}$), therefore producing a 20-bit external (or effective or physical) address from the 32-bit segment: offset pair. As a result, each external address can be referred to by $2^{12} = 4096$ different segment: offset pairs.

Types of Instruction Set

- Data Transfer Instructions
- Arithmetic Instructions
- Logical Instructions
- Shift and Rotate Instructions
- Transfer Instructions
- Subroutine and Interrupt Instructions
- String Instructions
- Processor Control Instructions

Conclusion:-

This theoretical overview covered the 8086 microprocessor's architecture, registers, segmentation, and instruction set. It provided foundational knowledge for programming and system design.