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

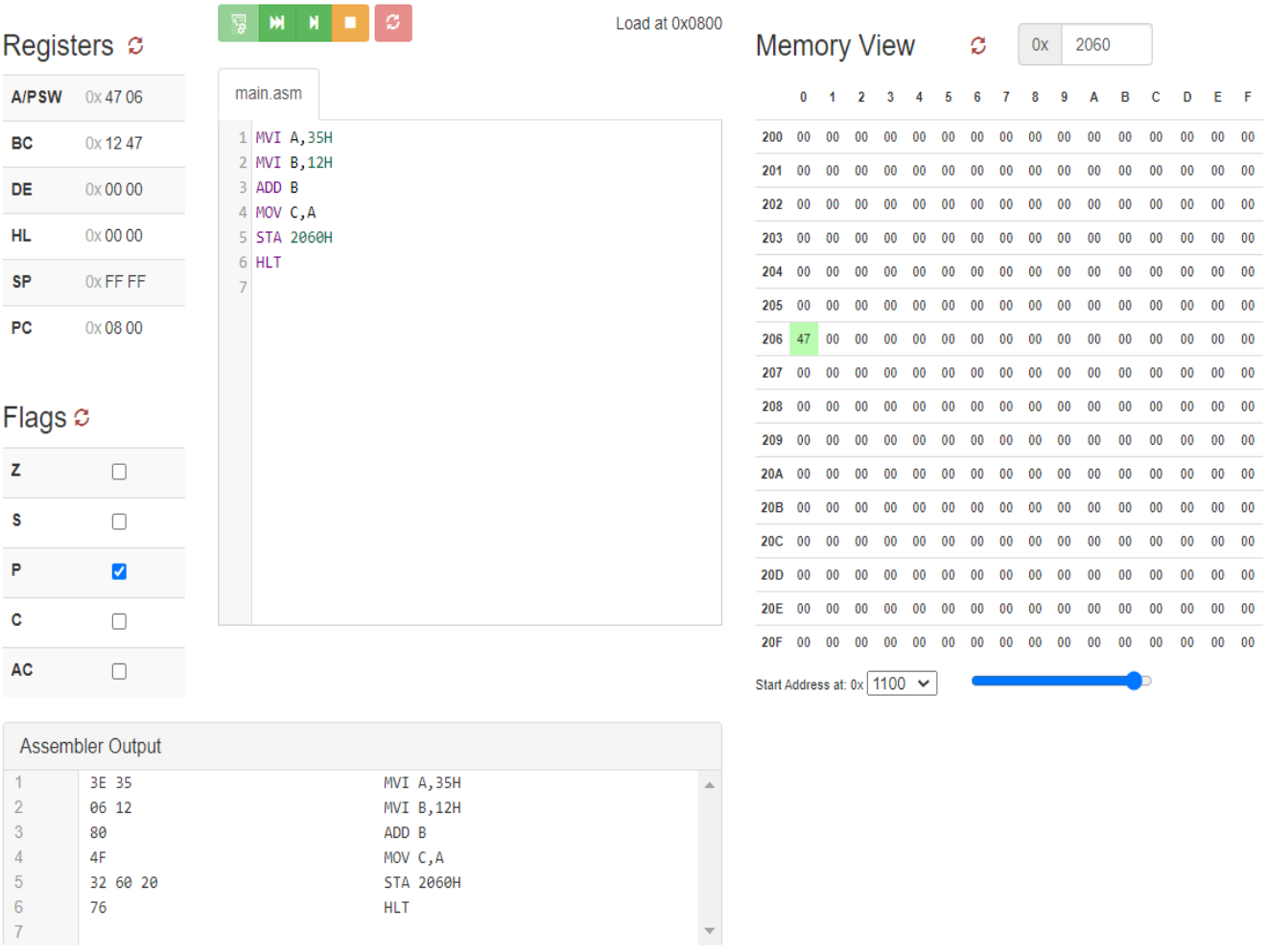
PRACTICAL-1

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:



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.

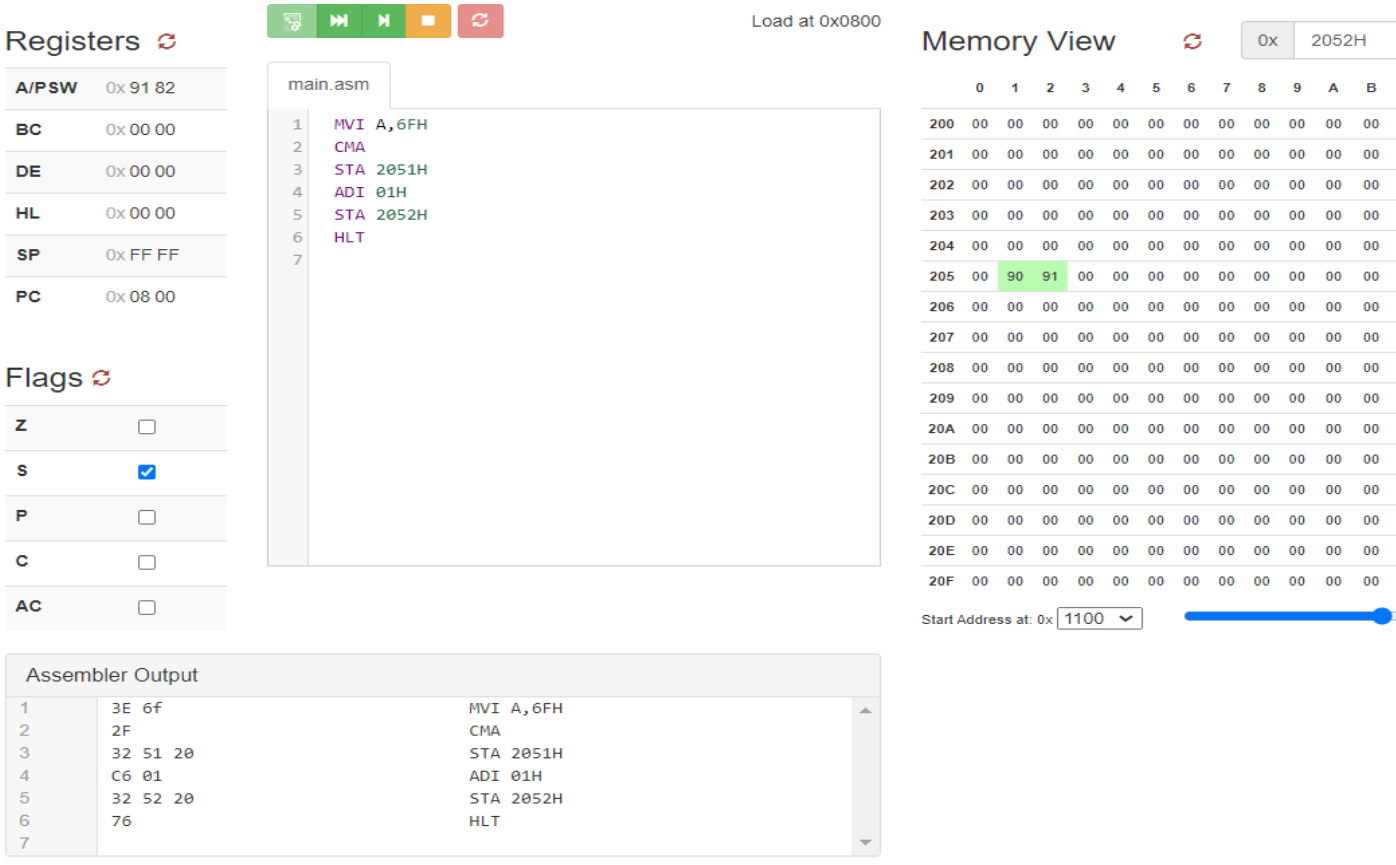
PRACTICAL-2

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:



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.

PRACTICAL-3

AIM: Write a program to perform 16-bit addition of two numbers.

PROGRAM:

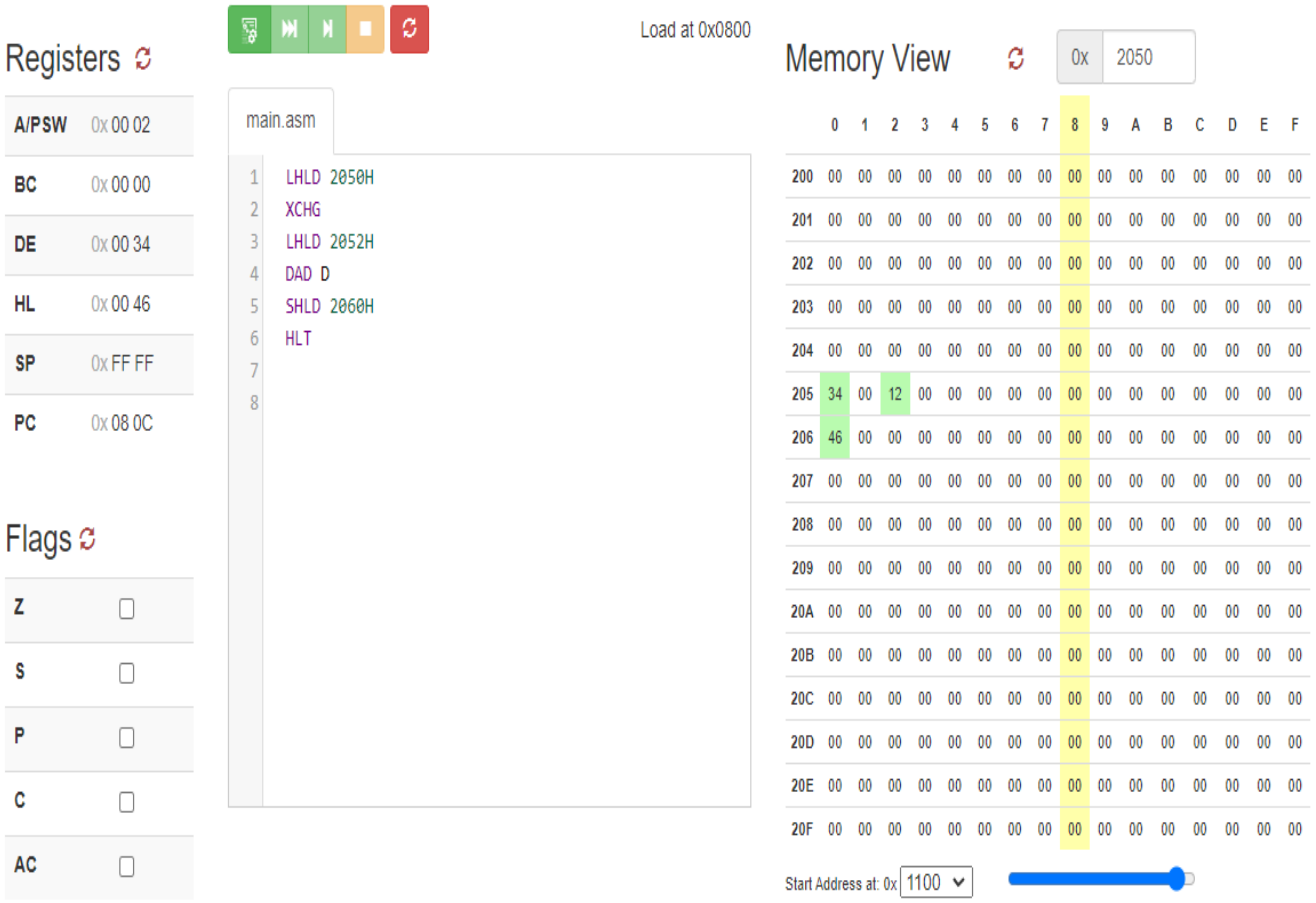
(3A)

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

Input:2050H:
1234H

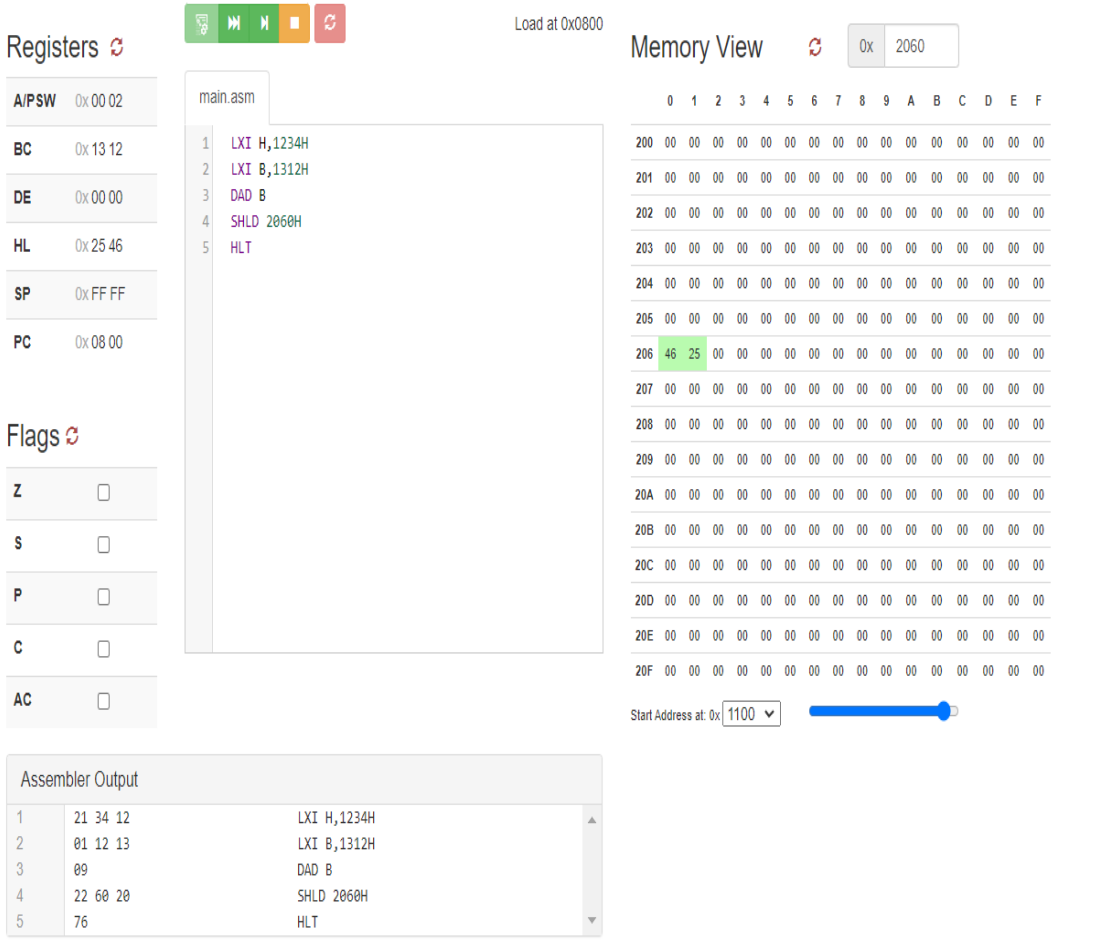
2052H:
1312H

Output:



PROGRAM (3B)

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

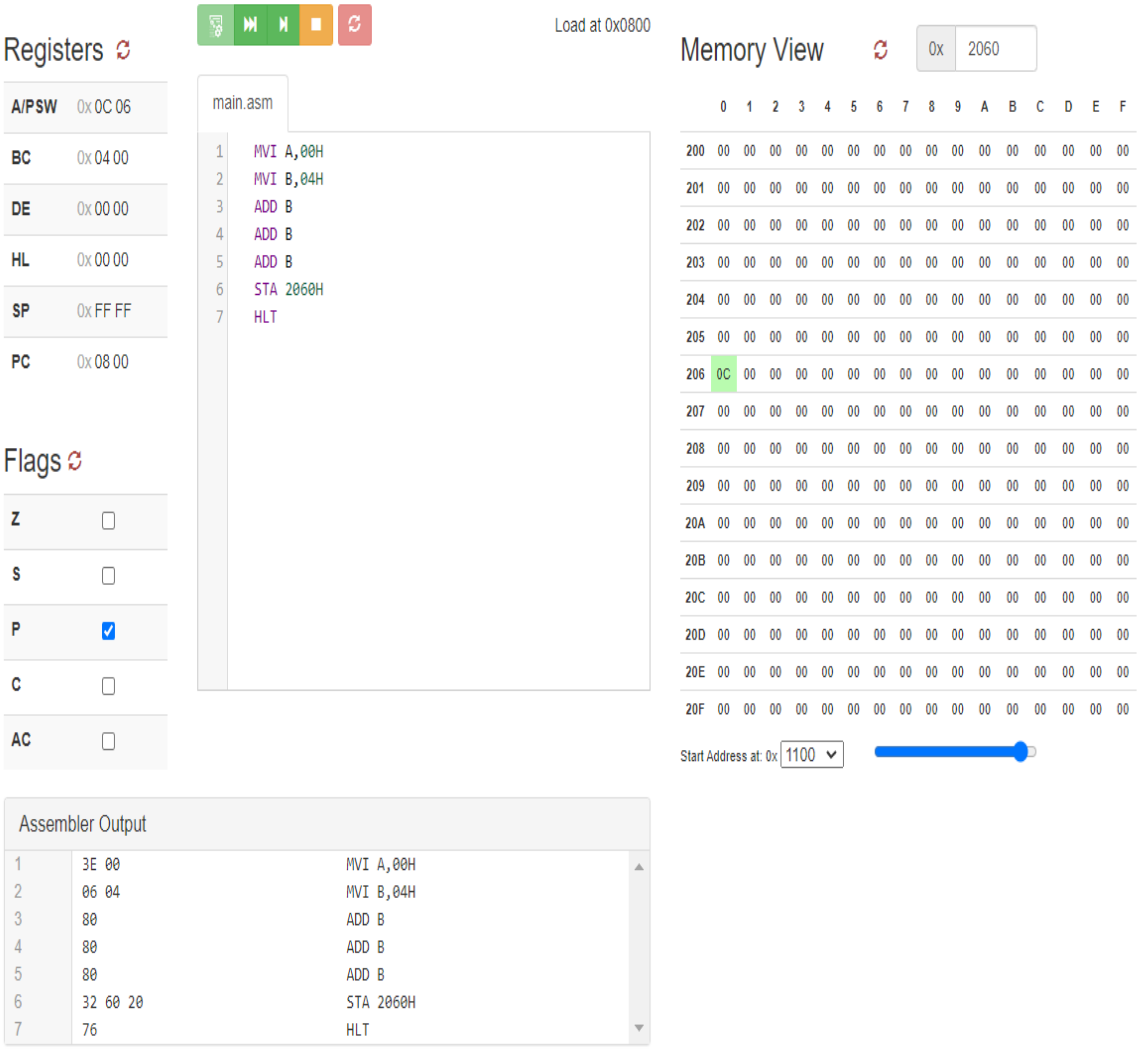
AIM: Write a program to multiply two 8-bit numbers using addition.

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



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:

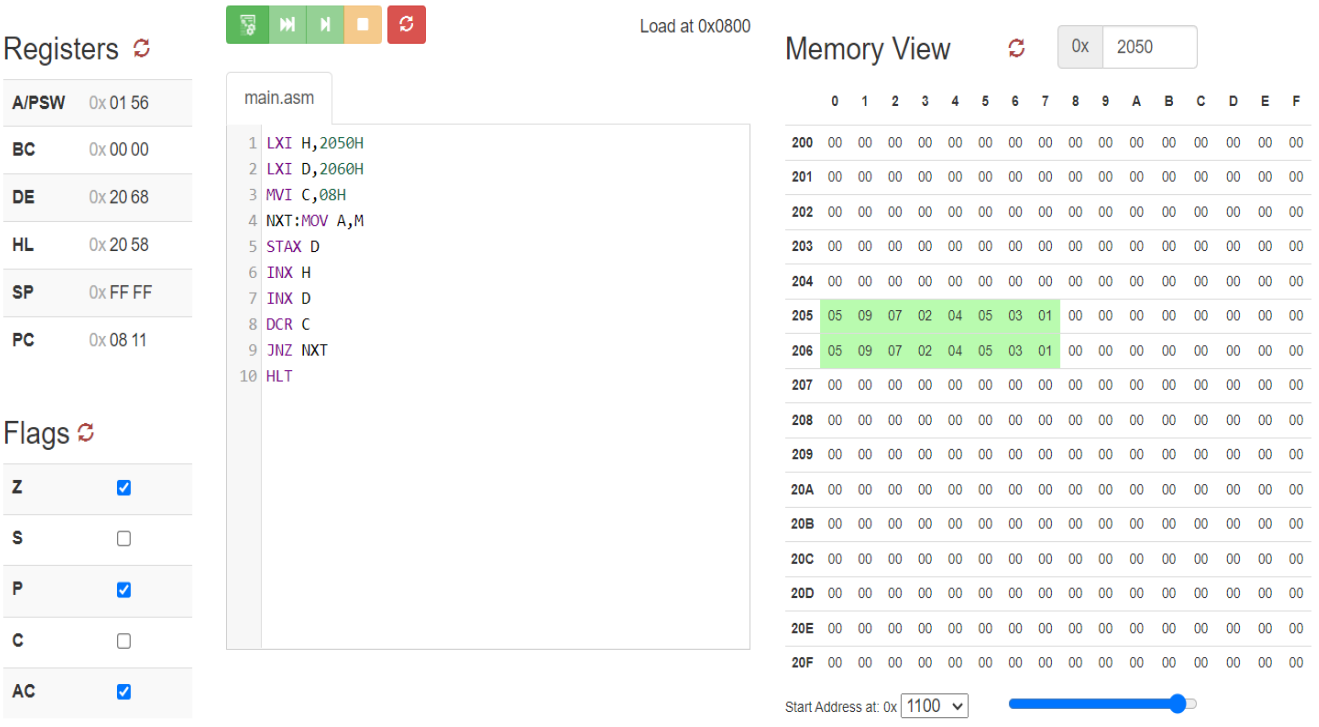
```
LXI H, 2050H
LXI D,2060H
MVI C,08H
NXT:MOV  A,M
STAX  D
INX H
INX D
DCR C
JNZ NXT
HLT
```

INPUT:

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

OUTPUT:

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



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
```

INPUT: 2050H: 01H

2051H: 02H

2052H: F1

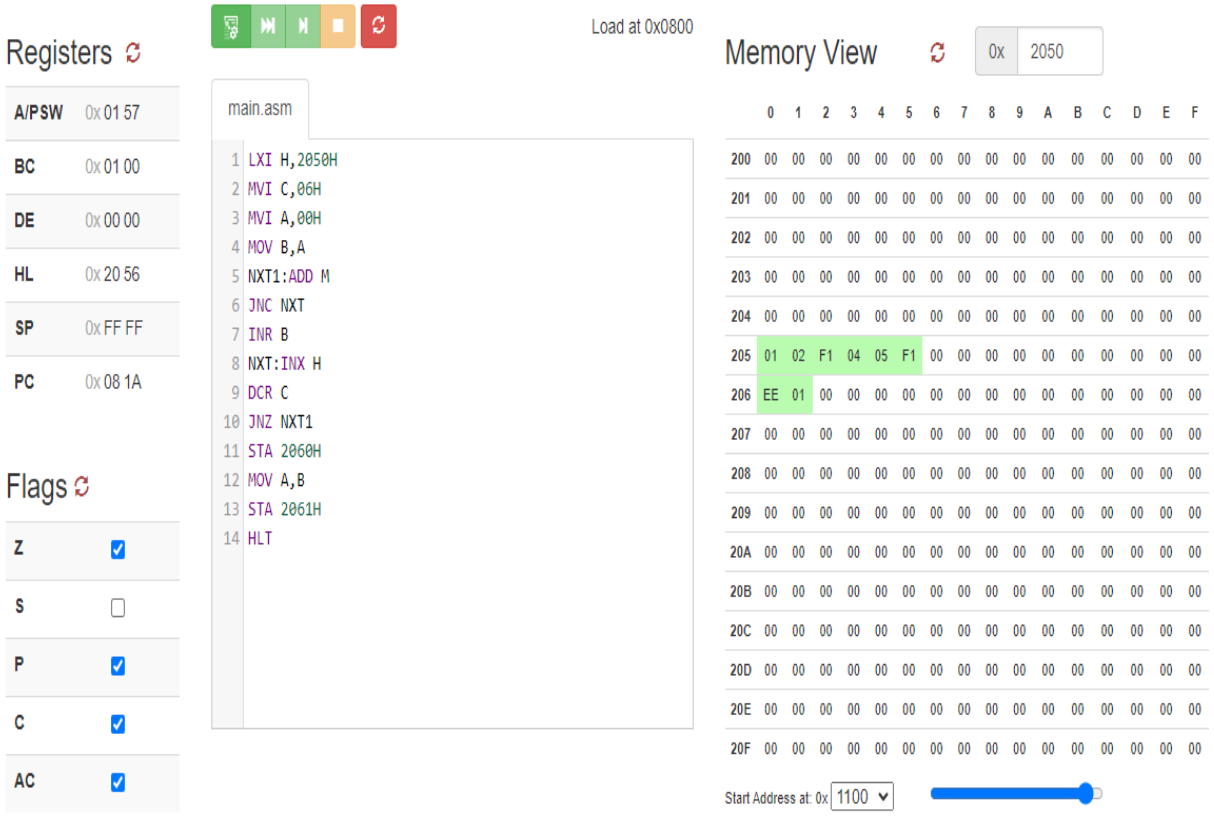
2053H: 04H

2054H: 05H

2055H: F1

OUTPUT: 2060H: EE ,

2061H:01



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)

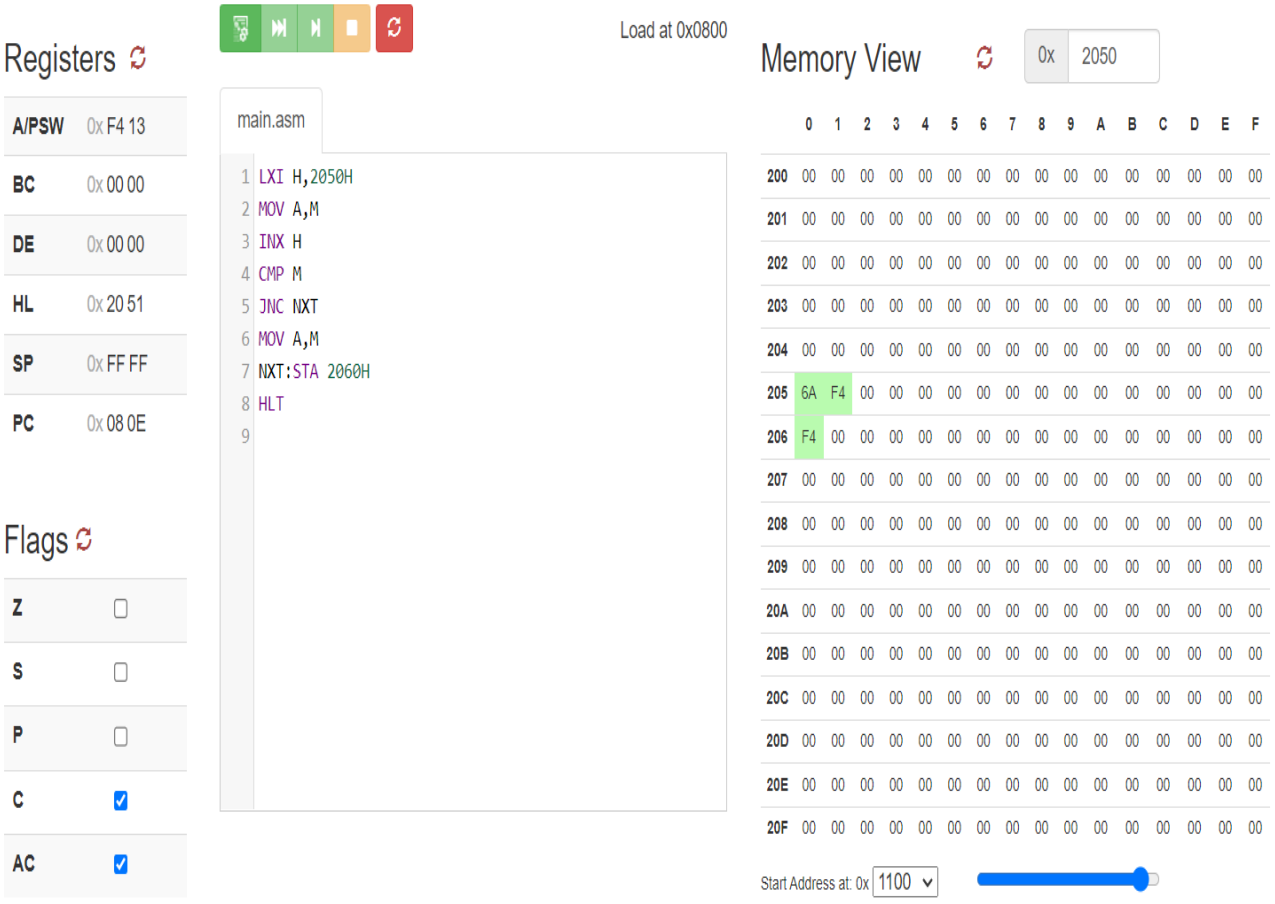
AIM: Write a program to find the largest number of given two 8-bit numbers at 2050H & 2051H memory location. Store the result at 2060H.

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



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.

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
```

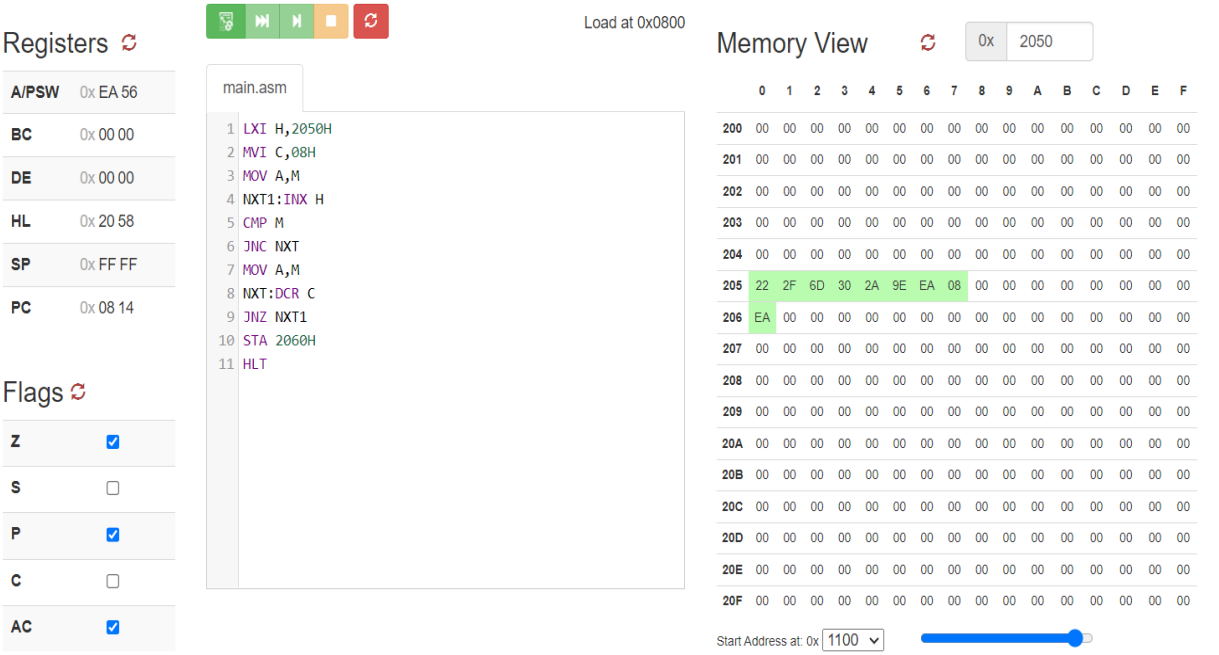
OBSERVATIONS:

Input:

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

Output:

2060H: EAH



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:

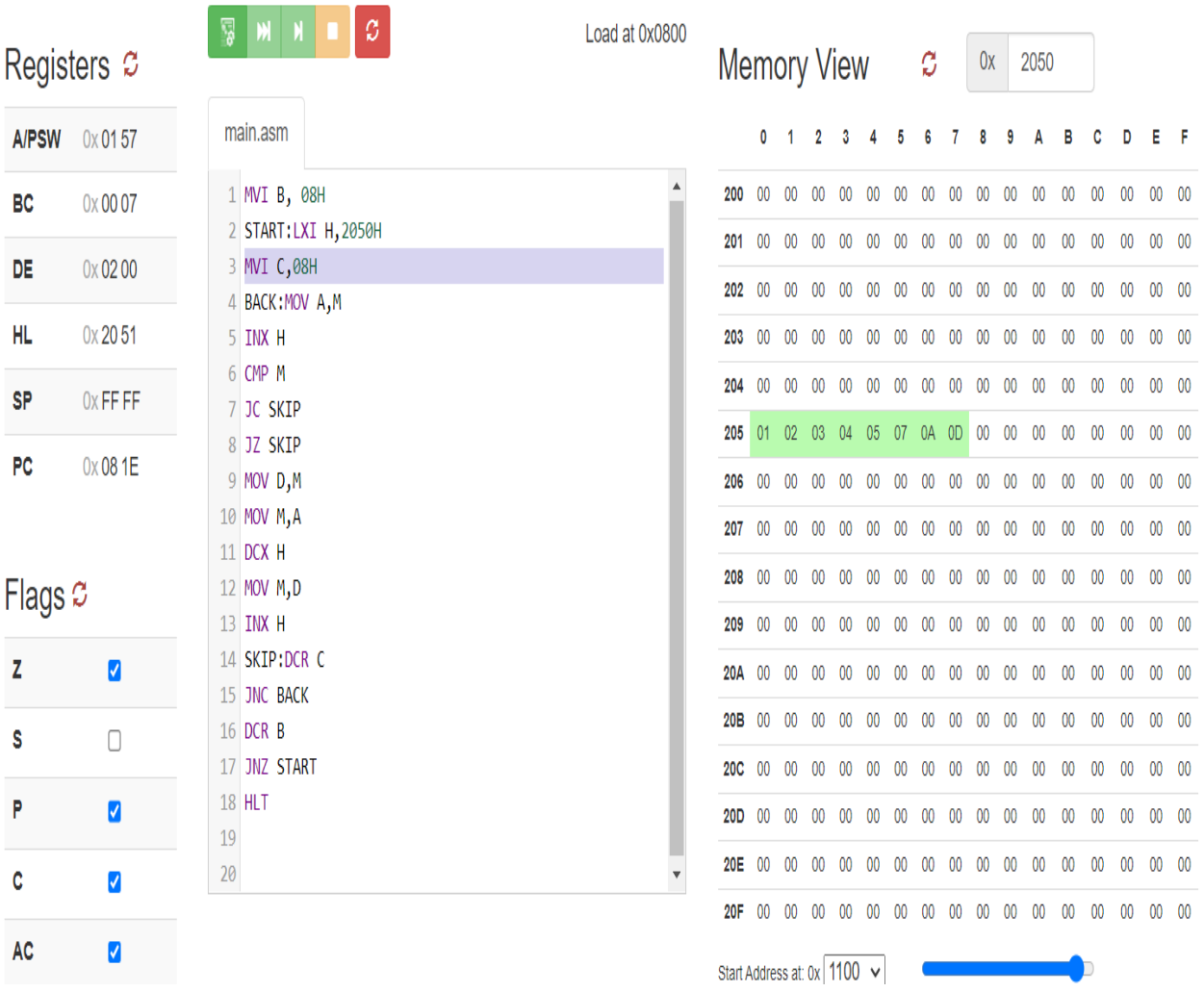
```
        MVI B, 08H
START:  LXI H,2050H
        MVI C,08H
BACK:   MOV A,M
        INX H
        CMP M
        JC SKIP
        JZ SKIP
        MOV D,M
        MOV M,A
        DCX H
        MOV M,D
        INX H
SKIP:   DCR C
        JNC BACK
        DCR B
        JNZ START
        HLT
```

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



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

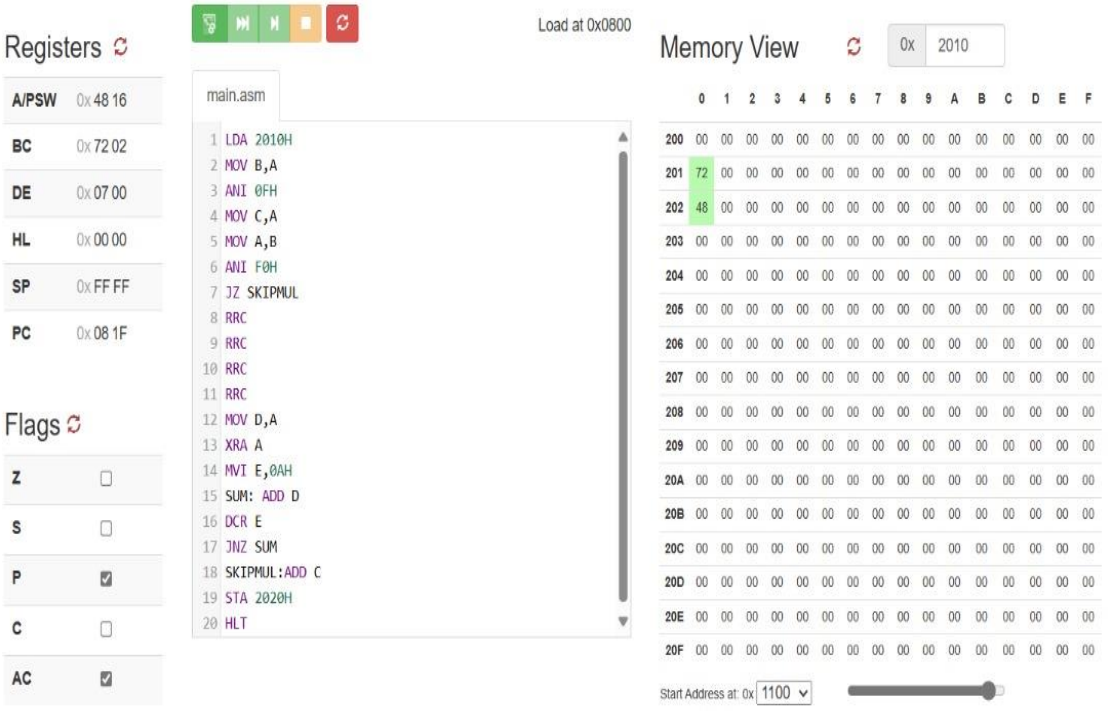
AIM: Write a program to convert a number from BCD to Binary.

PROGRAM:

```
        LDA 2010H
        MOV B,A
        ANI 0FH
        MOV C,A
        MOV A,B
        ANI F0H
        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



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

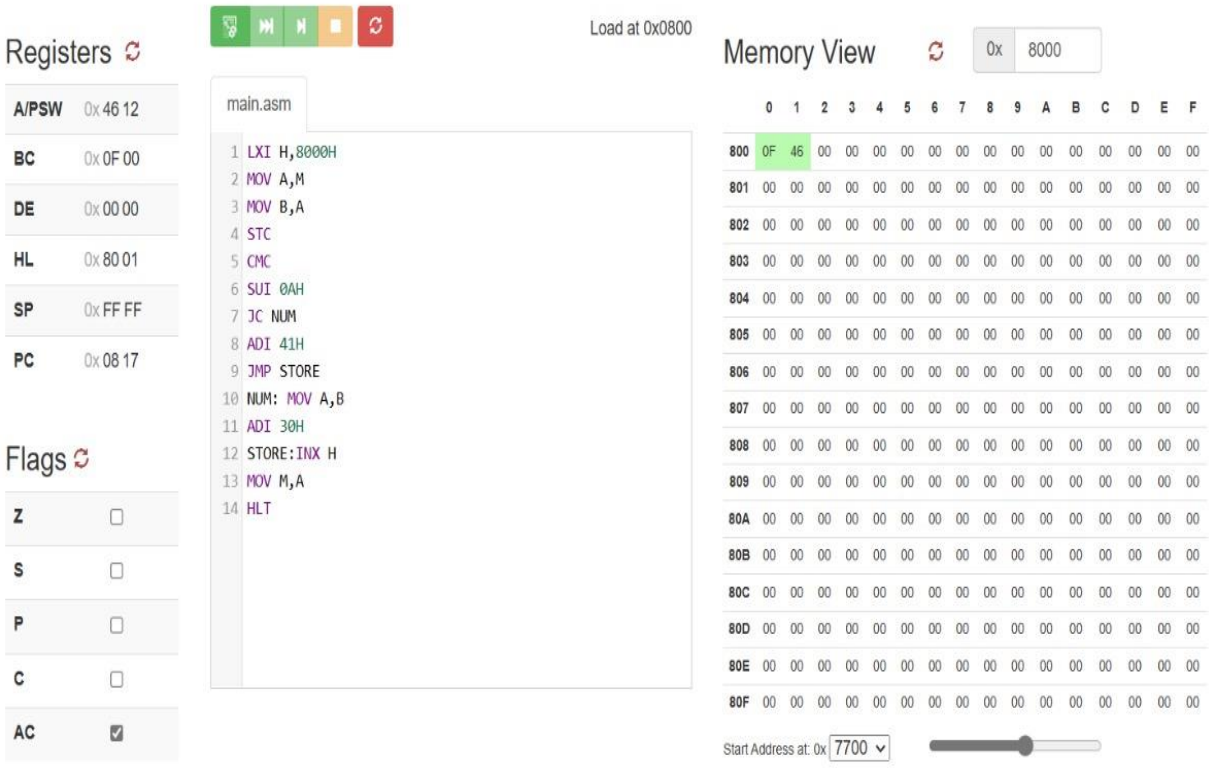
AIM: Write a program to convert from binary to ASCII.

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



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.

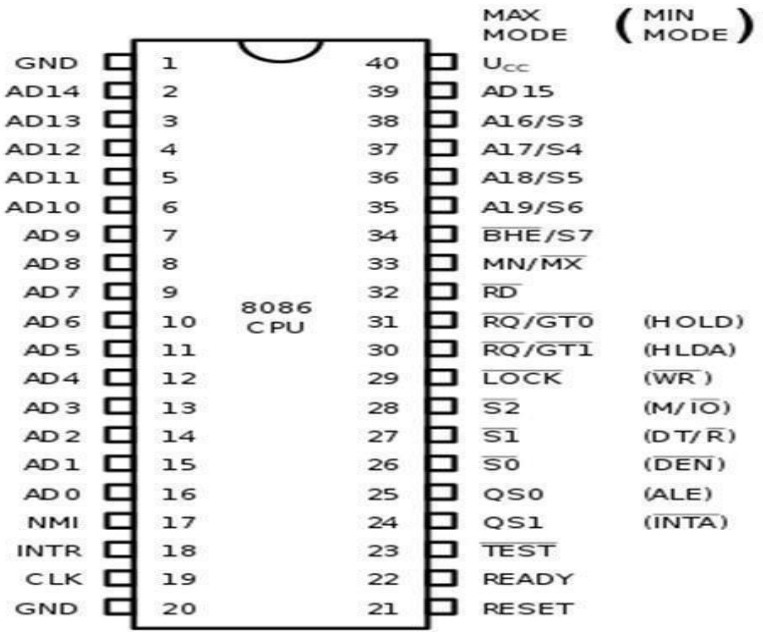
PRACTICAL-11

AIM: Introduction to 8086 Microprocessor.

Features of 8086

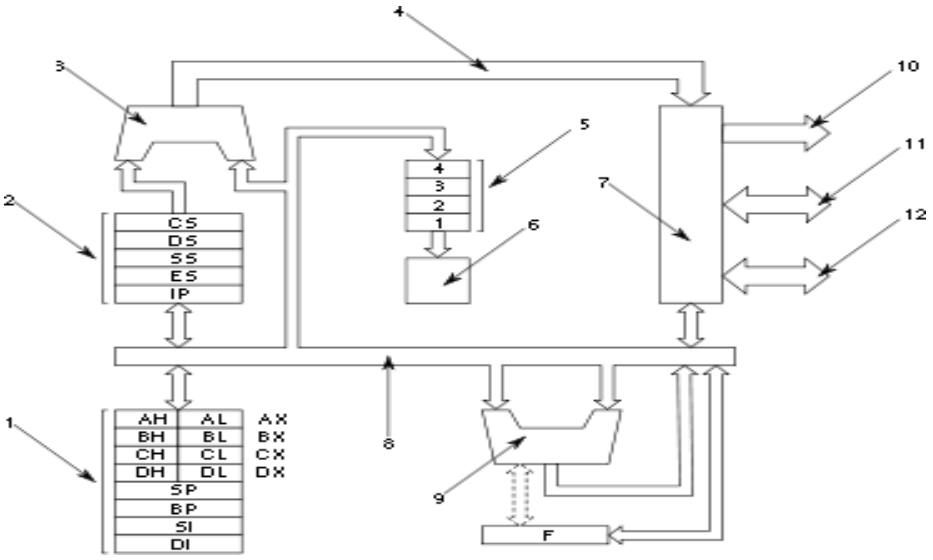
- 8086 is a 16bit processor. It's ALU, internal registers works with 16bit binary word.
- 8086 has a 16bit data bus. It can read or write data to a memory/port either 16bits or 8 bit at a time
- 8086 has a 20bit address bus which means, it can address upto 1MBmemory location
- Frequency range of 8086 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×segment + 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 212 = 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.