INDEX

PRACTICAL-1

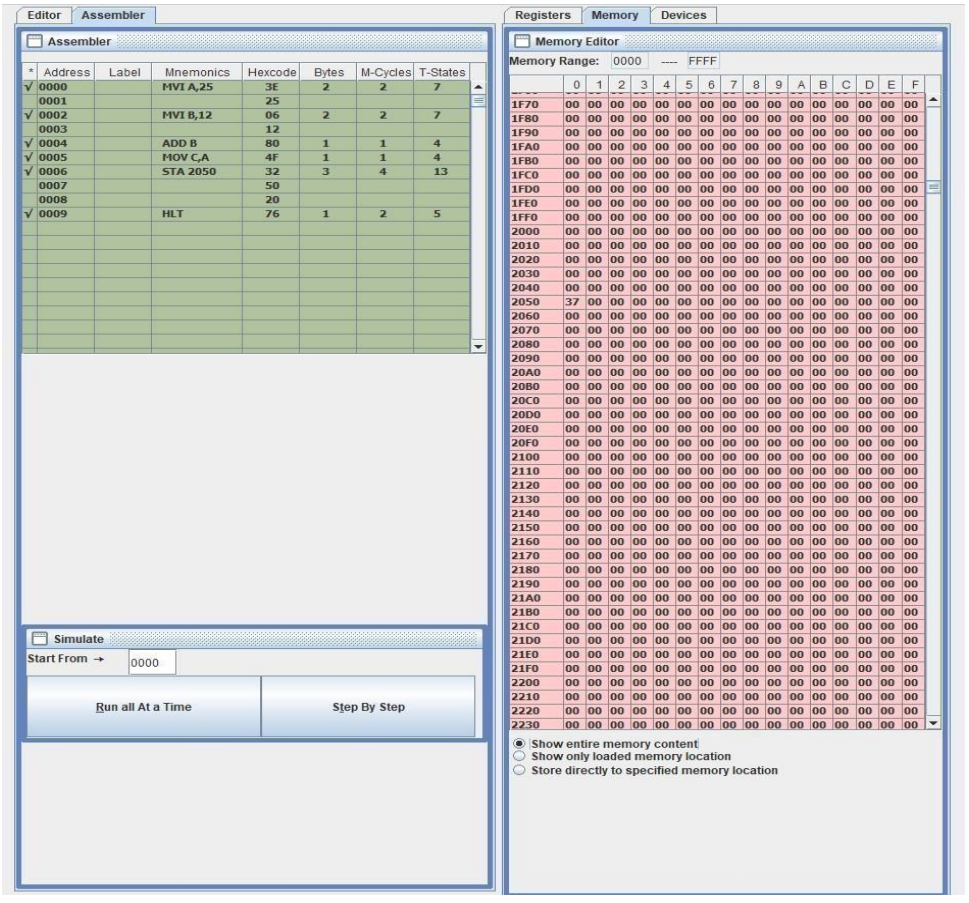
AIM: Write a program to add two 8-bit numbers.

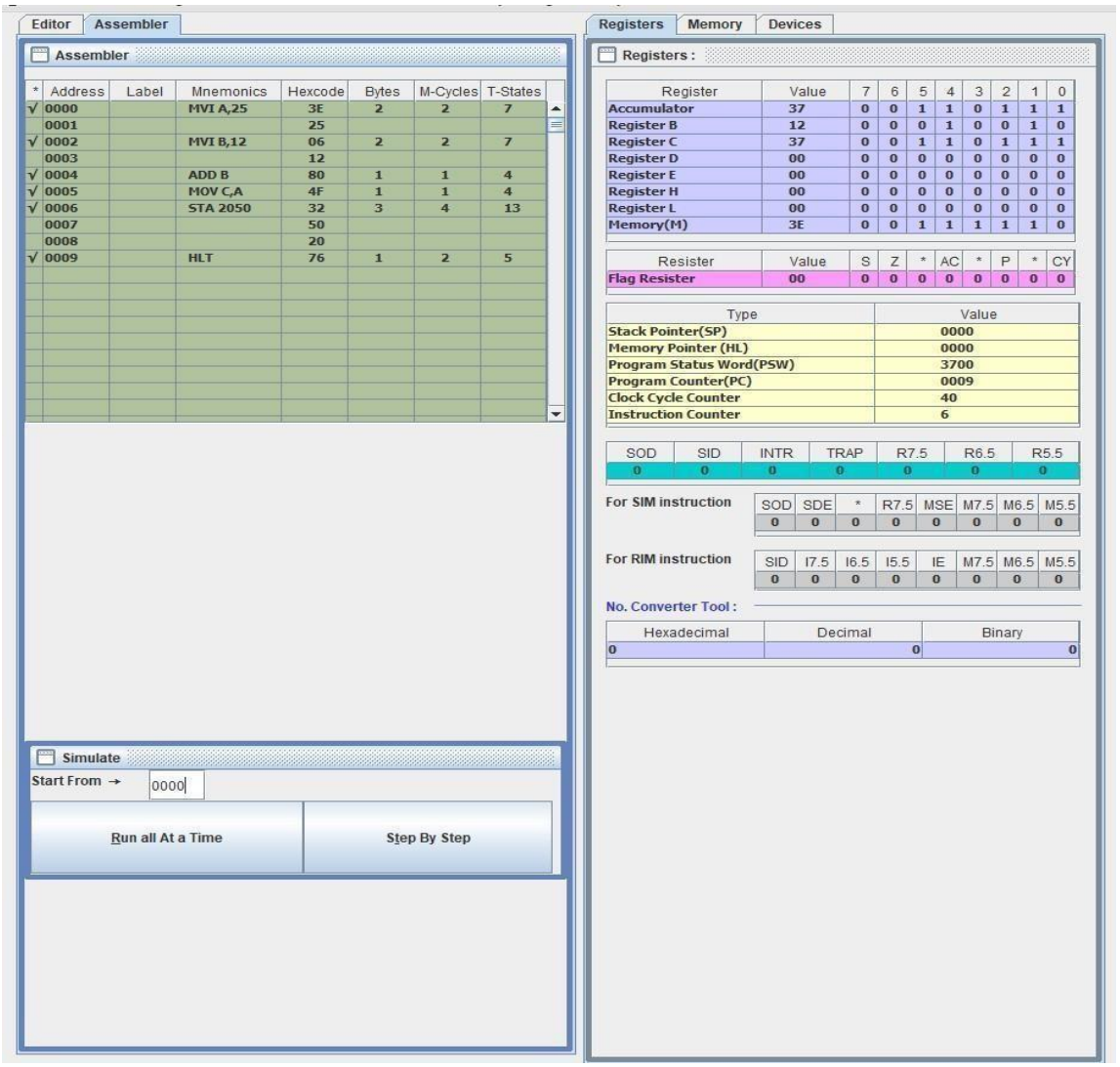
PROGRAM:

```
MVI A,25H
MVI B,12H
ADD B
MOV C, A
STA 2050H
HLT
```

OUTPUT:

2050H: 37H





*	Address	Label	Mnemonics	Hexcode	Bytes	M-Cycles	T-States
√	0000		MVI A,25	3E	2	2	7
	0001			25			
√	0002		MVI B,12	06	2	2	7
	0003			12			
√	0004		ADD B	80	1	1	4
√	0005		MOV C,A	4F	1	1	4
√	0006		STA 2050	32	3	4	13
	0007			50			
	0008			20			
√	0009		HLT	76	1	2	5

Register	Value	7	6	5	4	3	2	1	0
Accumulator	37	0	0	1	1	0	1	1	1
Register B	12	0	0	0	1	0	0	1	0
Register C	37	0	0	1	1	0	1	1	1
Register D	00	0	0	0	0	0	0	0	0
Register E	00	0	0	0	0	0	0	0	0
Register H	00	0	0	0	0	0	0	0	0
Register L	00	0	0	0	0	0	0	0	0
Memory(M)	3E	0	0	1	1	1	1	1	0

Resister	Value	S	Z	*	AC	*	P	*	CY
Flag Resister	00	0	0	0	0	0	0	0	0

Type	Value
Stack Pointer(SP)	0000
Memory Pointer (HL)	0000
Program Status Word(PSW)	3700
Program Counter(PC)	0009
Clock Cycle Counter	40
Instruction Counter	6

SOD	SID	INTR	TRAP	R7.5	R6.5	R5.5
0	0	0	0	0	0	0

For SIM instruction

SOD	SDE	*	R7.5	MSE	M7.5	M6.5	M5.5
0	0	0	0	0	0	0	0

For RIM instruction

SID	I7.5	I6.5	I5.5	IE	M7.5	M6.5	M5.5
0	0	0	0	0	0	0	0

PRACTICAL-2

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

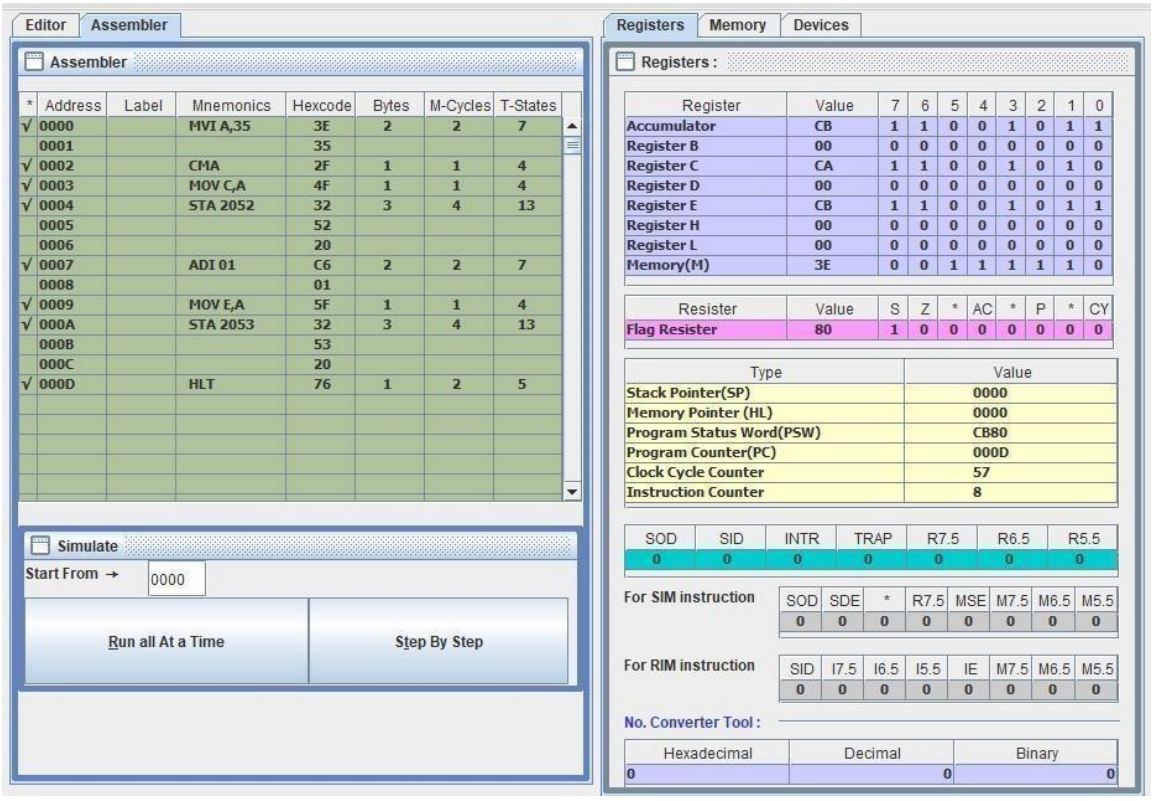
PROGRAM:

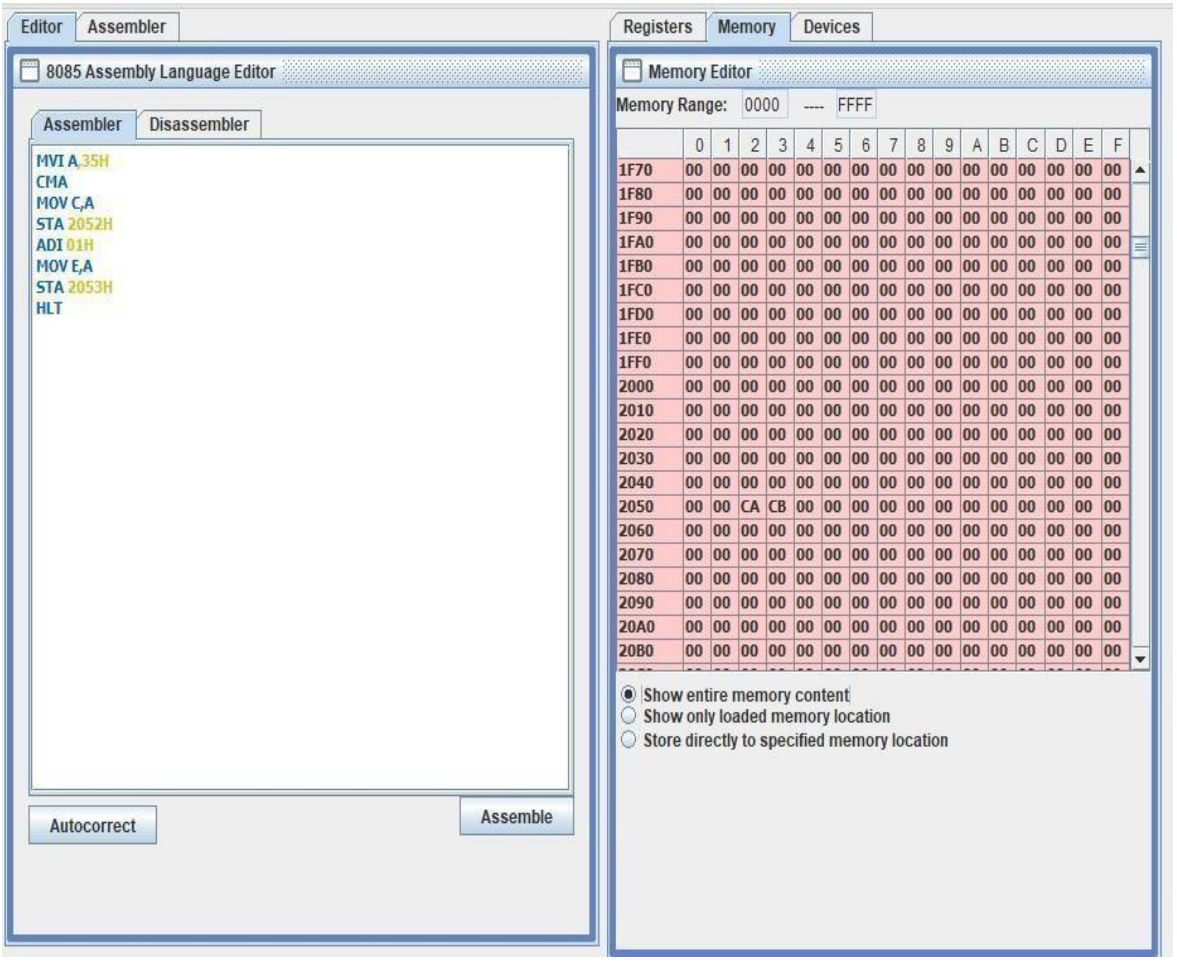
```
MVI A,35H
CMA
MOV C,A
STA 2052H
ADI 01H
MOV E,A
STA 2053H
HLT
```

OUTPUT:

2052H: CAH

2053H: CBH





	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
1F70	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
1F80	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
1F90	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
1FA0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
1FB0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
1FC0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
1FD0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
1FE0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
1FF0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
2000	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
2010	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
2020	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
2030	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
2040	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
2050	00	00	CA	CB	00	00	00	00	00	00	00	00	00	00	00	00
2060	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
2070	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
2080	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
2090	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
20A0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
20B0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00

PRACTICAL-3

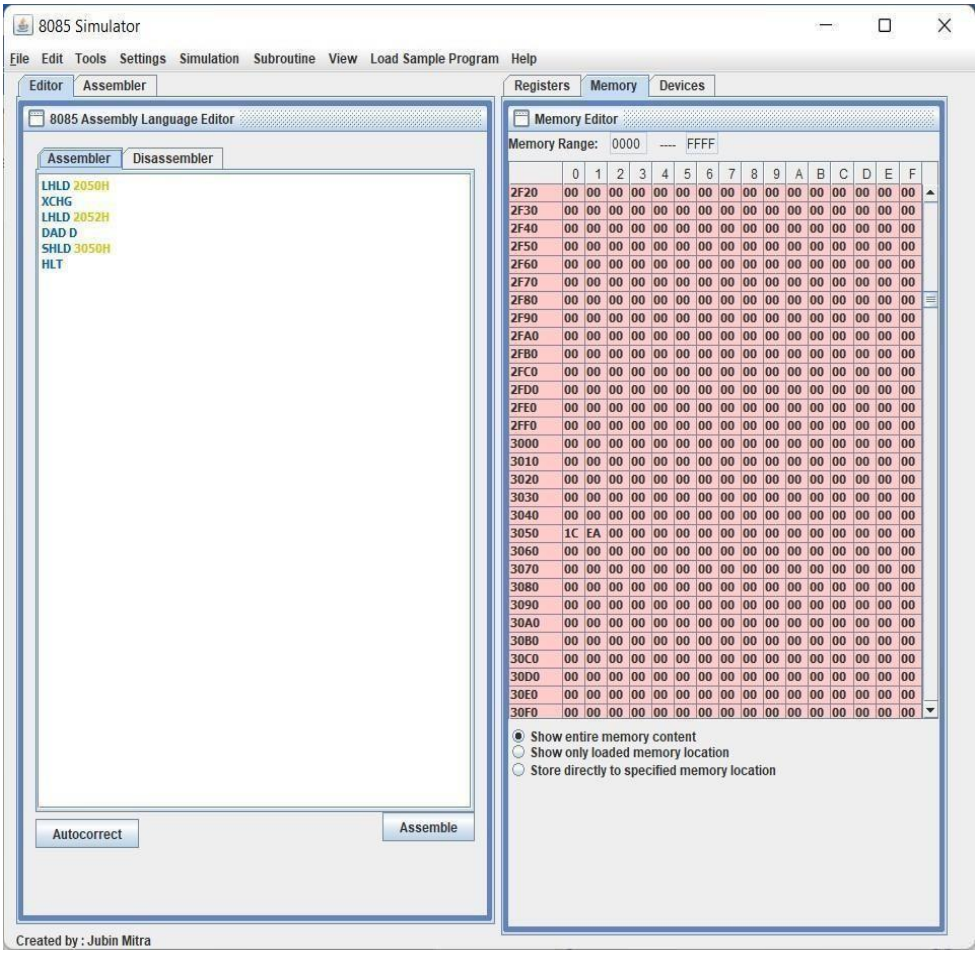
AIM: Write a program to perform 16-bit addition of two numbers.

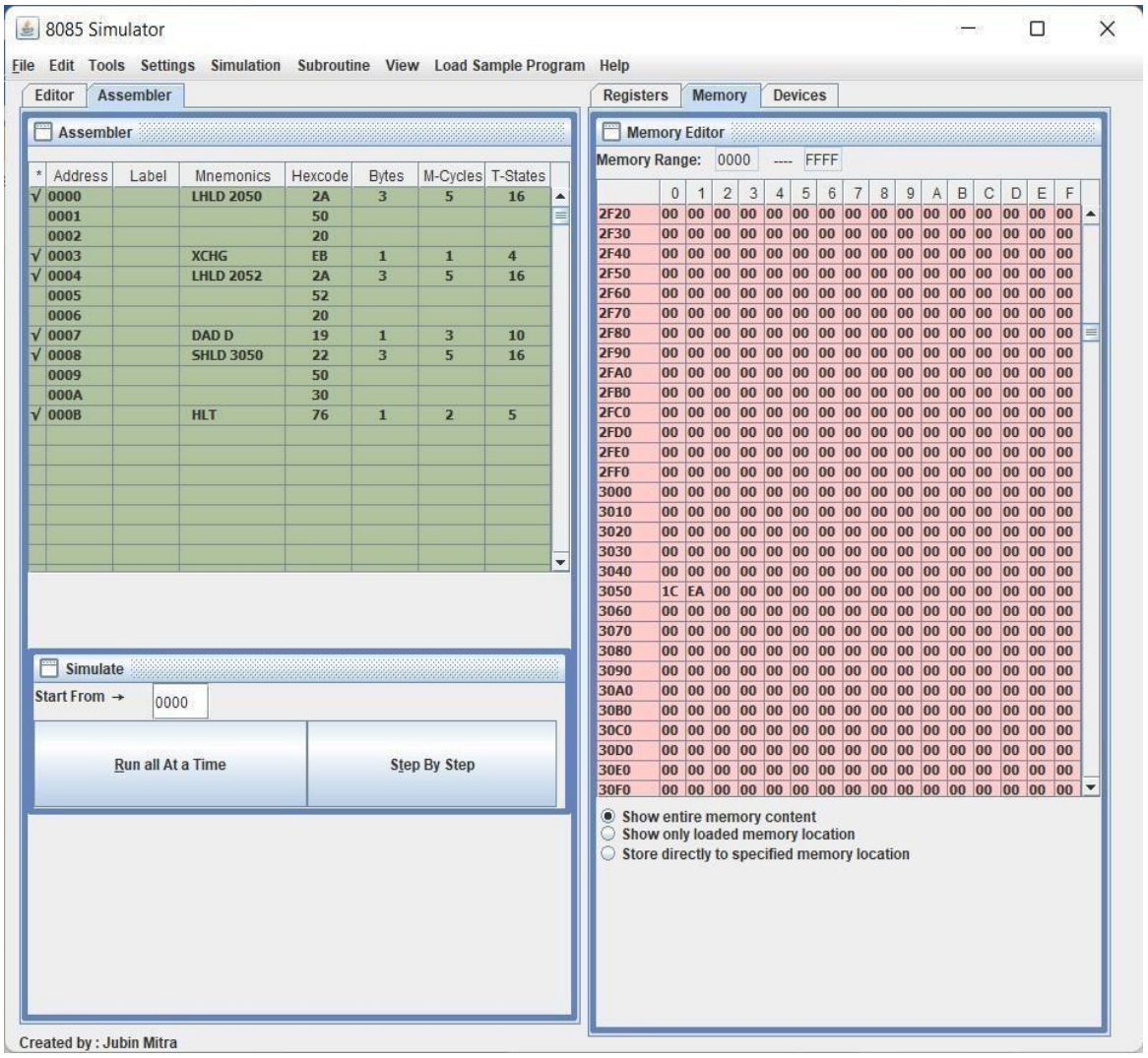
PROGRAM:

```
LHLD 2050H
XCHG
LHLD 2052H
DAD  D
SHLD 3050H
HLT
```

Input: 2050H: 02H
2051H: 20H

Output: 3050H: 1CH
3051H: EAH




8085 Simulator
File Edit Tools Settings Simulation Subroutine View Load Sample Program Help
Editor Assembler
Assembler
* Address Label Mnemonics Hexcode Bytes M-Cycles T-States
√ 0000 LHLD 2050 2A 3 5 16
0001 50
0002 20
√ 0003 XCHG EB 1 1 4
√ 0004 LHLD 2052 2A 3 5 16
0005 52
0006 20
√ 0007 DAD D 19 1 3 10
√ 0008 SHLD 3050 22 3 5 16
0009 50
000A 30
√ 000B HLT 76 1 2 5
Simulate
Start From → 0000
Run all At a Time
Step By Step
Registers Memory Devices
Memory Editor
Memory Range: 0000 ---- FFFF
0 1 2 3 4 5 6 7 8 9 A B C D E F
2F20 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
2F30 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
2F40 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
2F50 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
2F60 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
2F70 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
2F80 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
2F90 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
2FA0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
2FB0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
2FC0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
2FD0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
2FE0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
2FF0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
3000 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
3010 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
3020 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
3030 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
3040 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
3050 1C EA 00 00 00 00 00 00 00 00 00 00 00 00 00 00
3060 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
3070 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
3080 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
3090 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
30A0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
30B0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
30C0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
30D0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
30E0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
30F0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
Show entire memory content
Show only loaded memory location
Store directly to specified memory location
Created by : Jubin Mitra

PRACTICAL-4

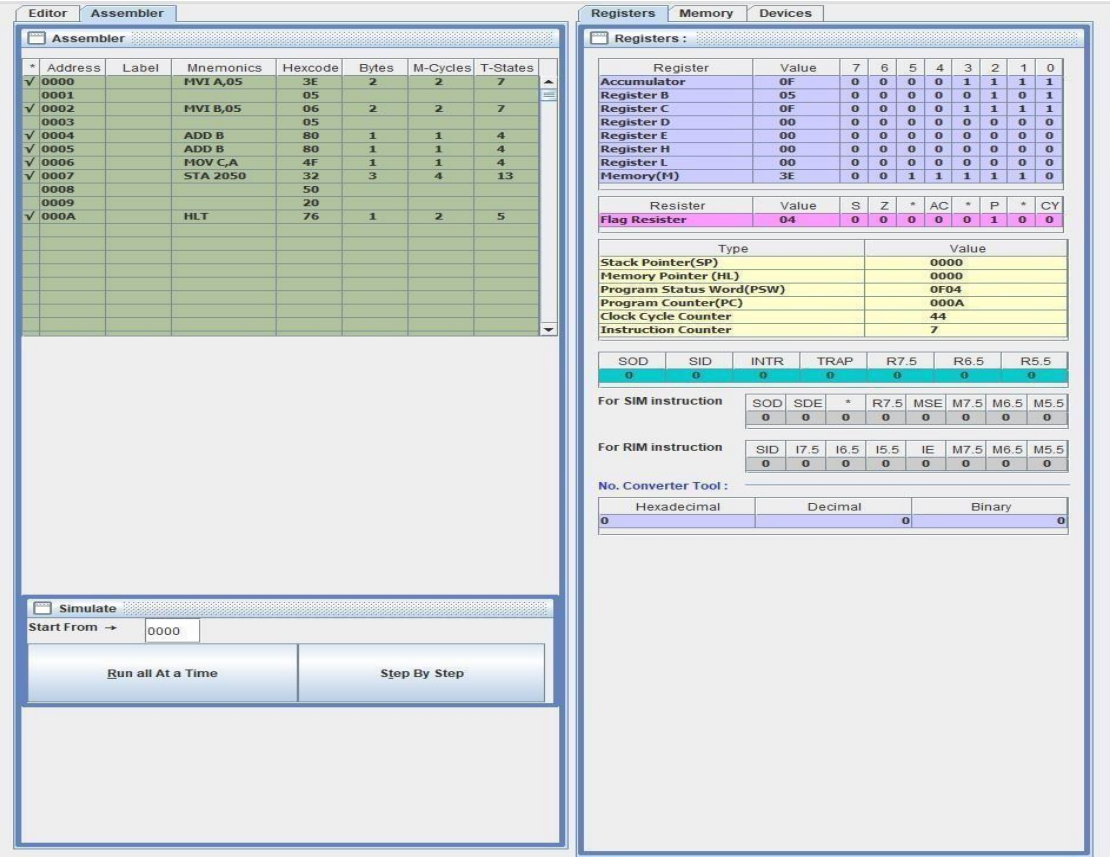
AIM: Write a program to multiply two 8-bit numbers using addition.

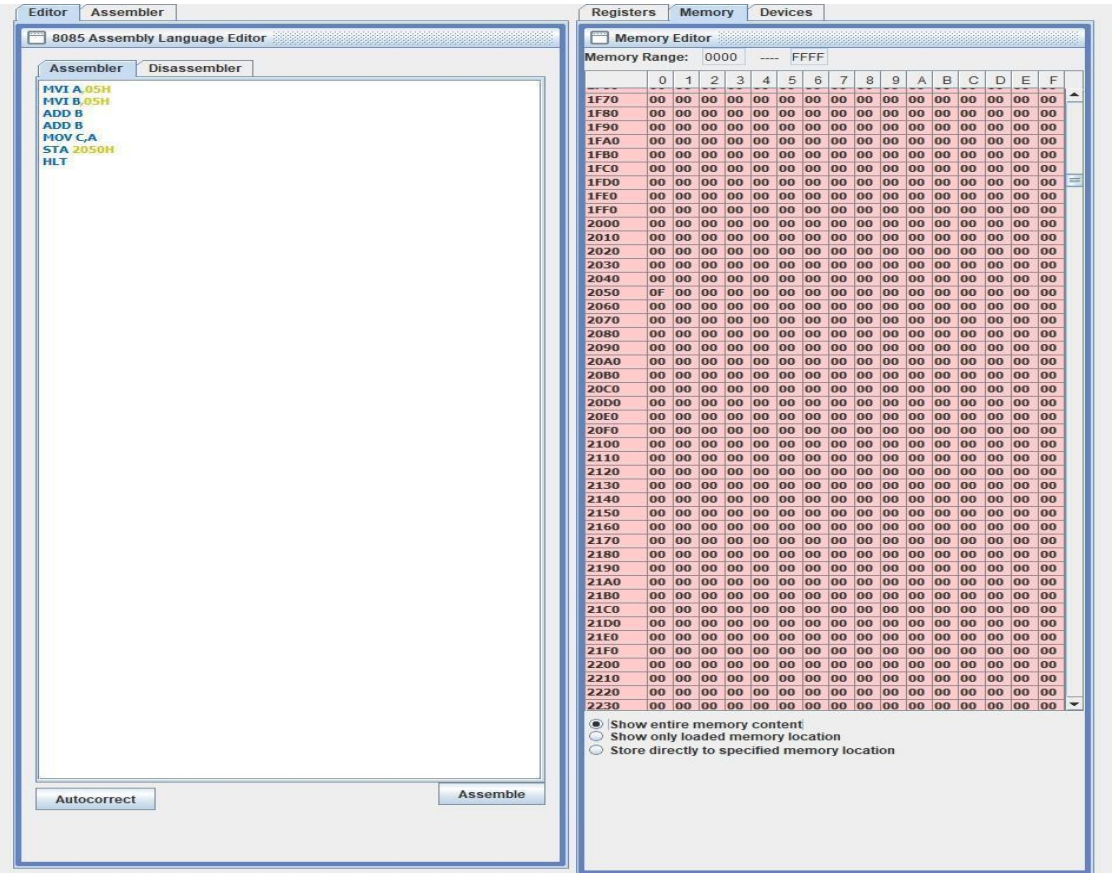
PROGRAM: To multiply two numbers: 5 and 3.

```
MVI A,05H
MVI B,05H
ADD B
ADD B
MOV C,A
STA 2050H
HLT
```

OUTPUT:

2050H: 0FH




Editor
Assembler
8085 Assembly Language Editor
Assembler
Disassembler
MVI A,05H
MVI B,05H
ADD B
ADD B
MOV C,A
STA 2050H
HLT
Autocorrect
Assemble
Registers
Memory
Devices
Memory Editor
Memory Range: 0000 ---- FFFF
Show entire memory content
Show only loaded memory location
Store directly to specified memory location

PRACTICAL-5

AIM: Write an ALP to transfer a block of data from memory location 2010H to 2080H.

PROGRAM:

```
        LXI H,2010H
        LXI D,2080H
        MVI C,10H
VISH1:  MOV A,C STAX
        D
        INX H
        INX D
        DCR C
        JNZ VISH1
        HLT
```

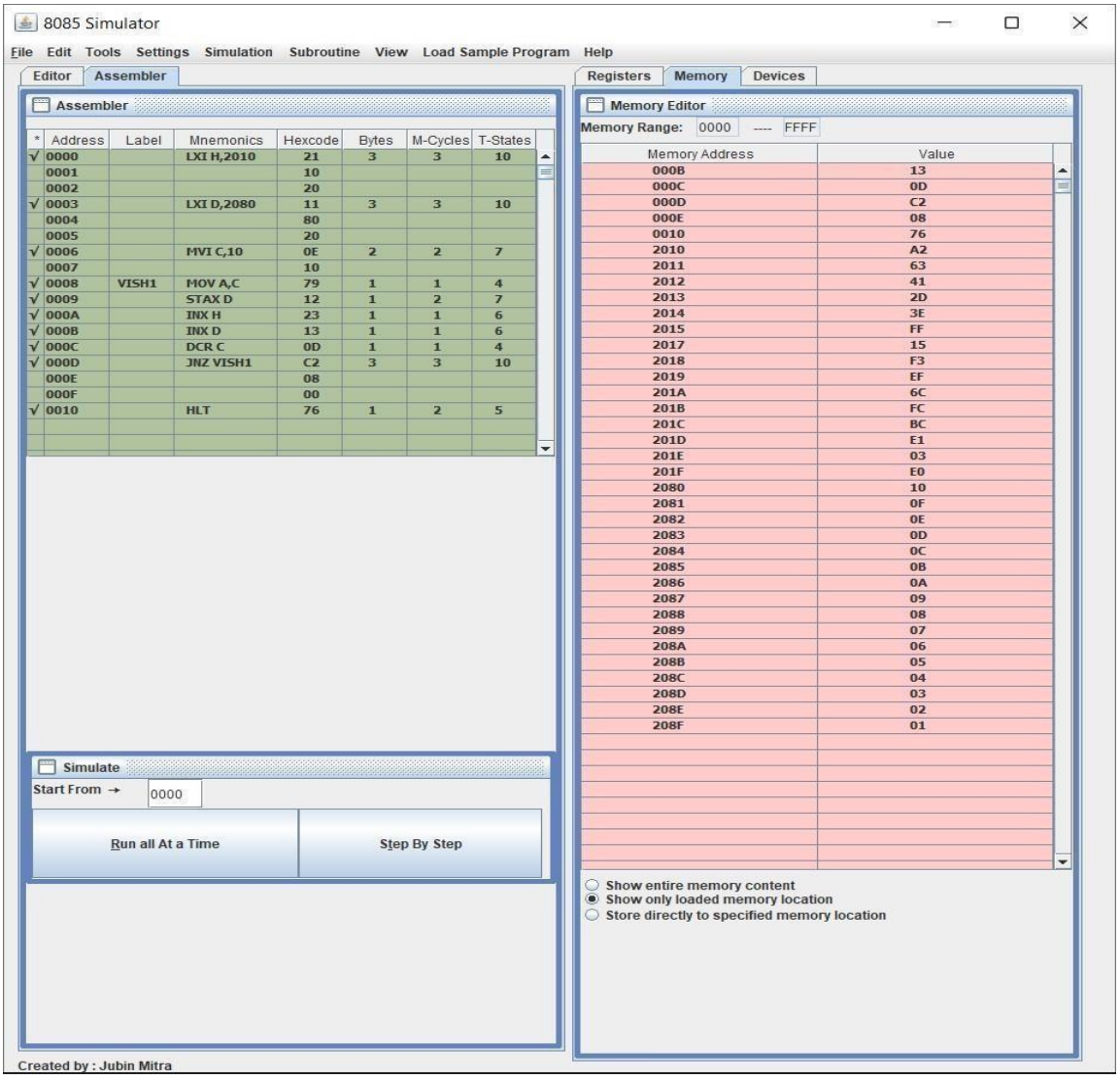
INPUT:

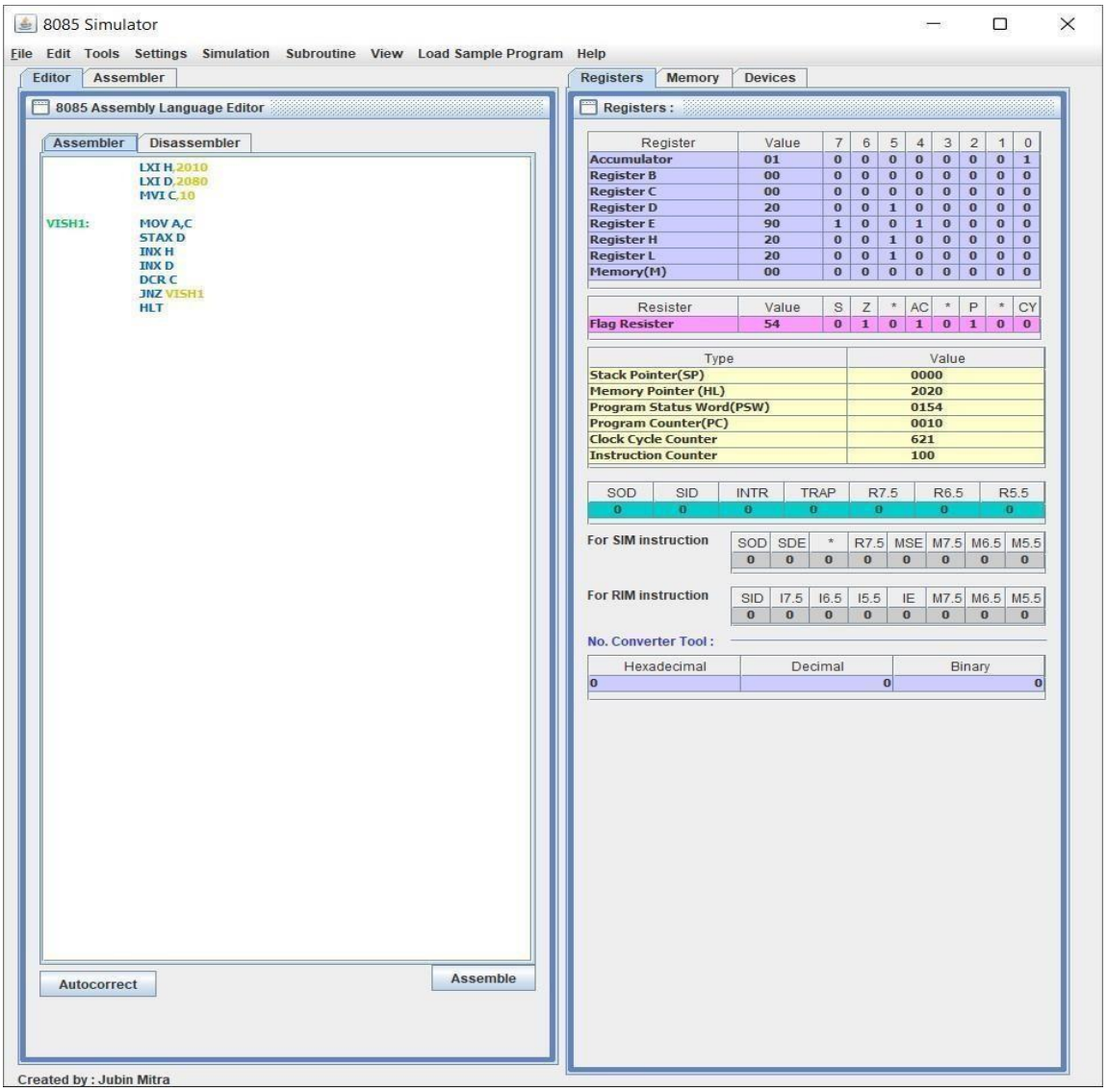
2010H: A2
2011H: 63H
2012H: 41H
2013H: 2DH
2014H: 3EH
2015H: FFH
2017H: 15H
2018H: F3H
2019H: EFH
201AH: 6CH
201BH: FCH
201CH: BCH
201DH: E1H
201EH:03H
201FH: E0H

OUTPUT:

2080H: 10H
2081H: 0FH
2082H: 0EH
2083H: 0DH
2084H: 0CH
2085H: 0BH
2086H: 0AH
2087H: 09H

2088H: 08H
2089H: 07H
208AH: 06H
208BH: 05H
208CH: 04H
208DH: 03H
208EH: 02H
208FH: 01H




8085 Simulator
File Edit Tools Settings Simulation Subroutine View Load Sample Program Help
Editor Assembler
8085 Assembly Language Editor
Assembler Disassembler
LXI H,2010
LXI D,2080
MVI C,10
VISH1: MOV A,C
STAX D
INX H
INX D
DCR C
JNZ VISH1
HLT
Autocorrect
Assemble
Created by : Jubin Mitra
Registers Memory Devices
Registers :
Register Value 7 6 5 4 3 2 1 0
Accumulator 01 0 0 0 0 0 0 0 1
Register B 00 0 0 0 0 0 0 0 0
Register C 00 0 0 0 0 0 0 0 0
Register D 20 0 0 1 0 0 0 0 0
Register E 90 1 0 0 1 0 0 0 0
Register H 20 0 0 1 0 0 0 0 0
Register L 20 0 0 1 0 0 0 0 0
Memory(M) 00 0 0 0 0 0 0 0 0
Resister Value S Z * AC * P * CY
Flag Resister 54 0 1 0 1 0 1 0 0
Type Value
Stack Pointer(SP) 0000
Memory Pointer (HL) 2020
Program Status Word(PSW) 0154
Program Counter(PC) 0010
Clock Cycle Counter 621
Instruction Counter 100
SOD SID INTR TRAP R7.5 R6.5 R5.5
0 0 0 0 0 0 0
For SIM instruction
SOD SDE * R7.5 MSE M7.5 M6.5 M5.5
0 0 0 0 0 0 0 0
For RIM instruction
SID I7.5 I6.5 I5.5 IE M7.5 M6.5 M5.5
0 0 0 0 0 0 0 0
No. Converter Tool :
Hexadecimal Decimal Binary
0 0 0

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 2070H and 2071H.

PROGRAM:

```
 MVI A,00H
MOV B,A
LXI H,2050H
MVI C,06H
NXTBYTE: ADD M
JNC  NXTMEM
INR B
NXTMEM: INX H
DCR C
JNZ NXTBYTE
LXI H,2070H
MOV M,A
INX H
MOV M,B
HLT
```

INPUT: 2050H: 08H

2051H: 10H

2052H: 3AH

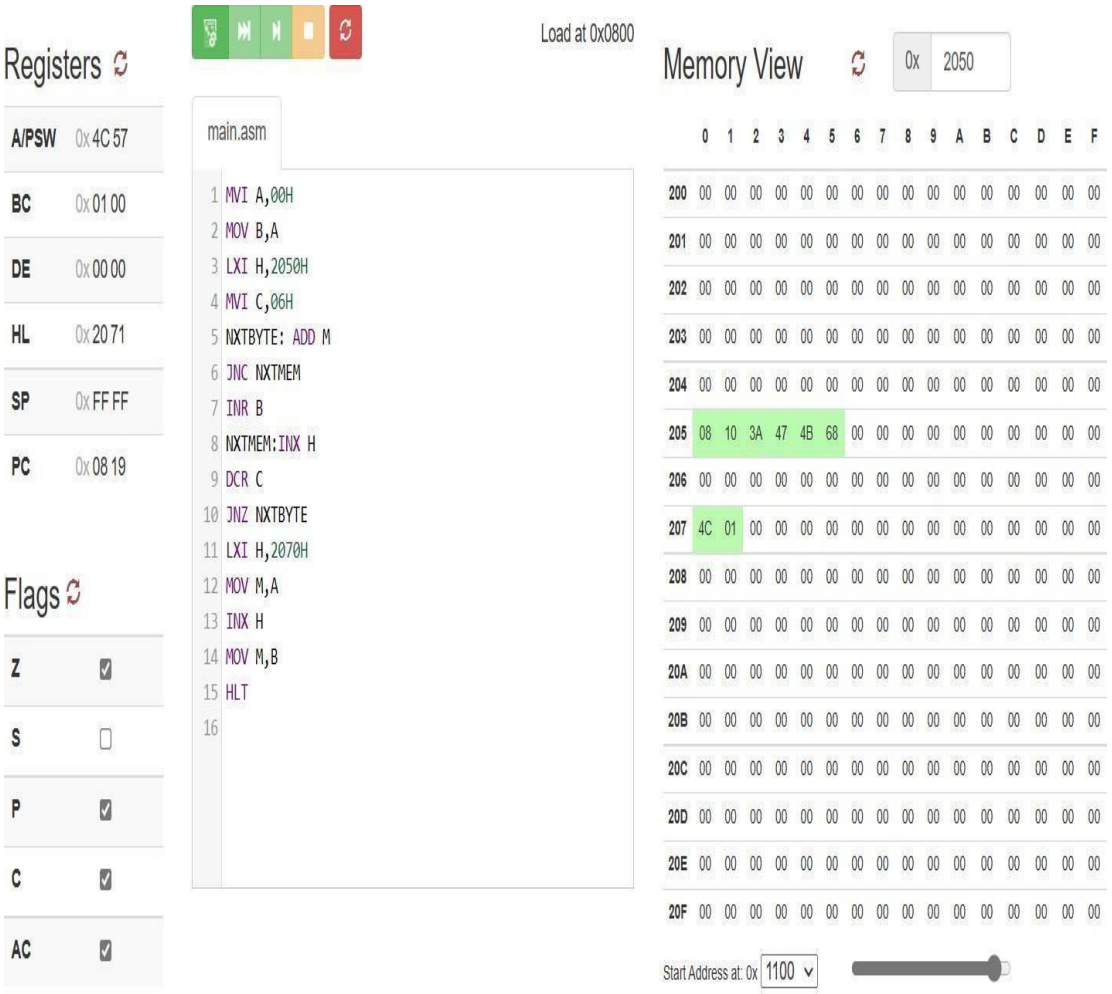
2053H: 47H

2054H: 4BH

2055H: 68H

OUTPUT: 2070H: 4CH

2071H: 01H



PRACTICAL-7

(A)

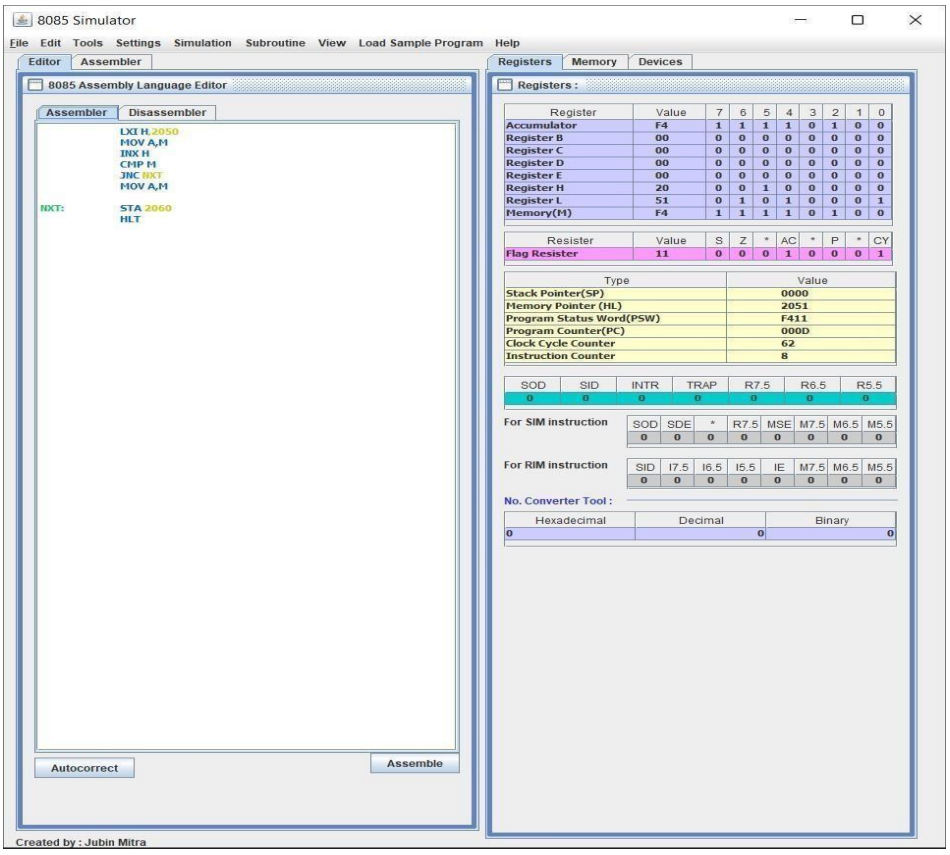
AIM: Write a program to find the largest number of given two 8-bit numbers at 2050H & 2051H memory location. Store the result at 2060H.

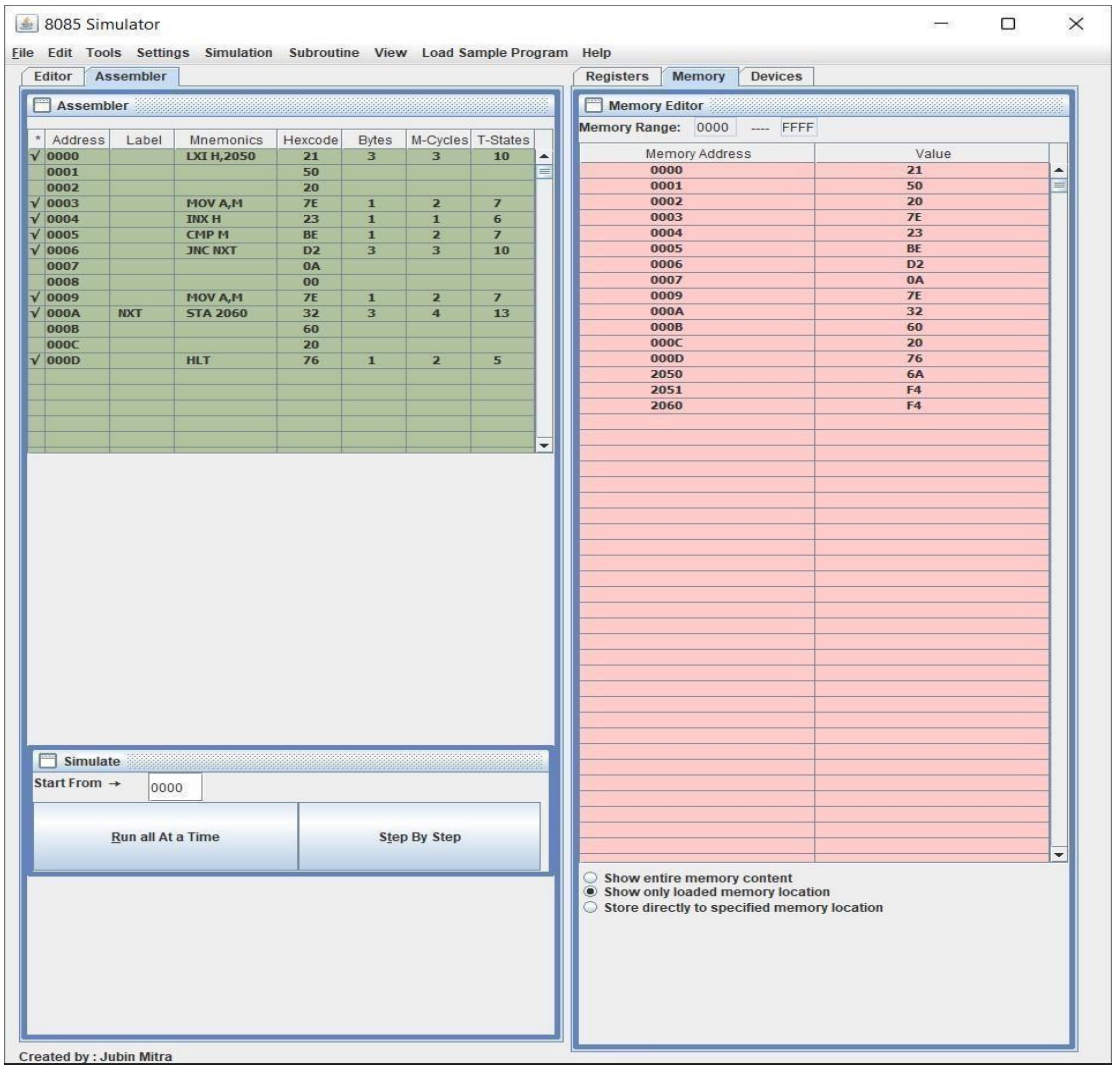
PROGRAM:

```
        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


8085 Simulator
File Edit Tools Settings Simulation Subroutine View Load Sample Program Help
Editor Assembler
8085 Assembly Language Editor
Assembler Disassembler
LXI H,2050
MOV A,M
INX H
CMP M
JNC NXT
MOV A,M
NXT: STA 2060
HLT
Autocorrect
Assemble
Created by : Jubin Mitra
Registers Memory Devices
Registers :
Register Value 7 6 5 4 3 2 1 0
Accumulator F4 1 1 1 1 0 1 0 0
Register B 00 0 0 0 0 0 0 0 0
Register C 00 0 0 0 0 0 0 0 0
Register D 00 0 0 0 0 0 0 0 0
Register E 00 0 0 0 0 0 0 0 0
Register H 20 0 0 1 0 0 0 0 0
Register L 51 0 1 0 1 0 0 0 1
Memory(M) F4 1 1 1 1 0 1 0 0
Resister Value S Z * AC * P * CY
Flag Resister 11 0 0 0 1 0 0 0 1
Type Value
Stack Pointer(SP) 0000
Memory Pointer (HL) 2051
Program Status Word(PSW) F411
Program Counter(PC) 000D
Clock Cycle Counter 62
Instruction Counter 8
SOD SID INTR TRAP R7.5 R6.5 R5.5
0 0 0 0 0 0 0
For SIM instruction
SOD SDE * R7.5 MSE M7.5 M6.5 M5.5
0 0 0 0 0 0 0 0
For RIM instruction
SID I7.5 I6.5 I5.5 IE M7.5 M6.5 M5.5
0 0 0 0 0 0 0 0
No. Converter Tool :
Hexadecimal Decimal Binary
0 0 0


8085 Simulator
File Edit Tools Settings Simulation Subroutine View Load Sample Program Help
Editor Assembler
Assembler
* Address Label Mnemonics Hexcode Bytes M-Cycles T-States
√ 0000 LXI H,2050 21 3 3 10
0001 50
0002 20
√ 0003 MOV A,M 7E 1 2 7
√ 0004 INX H 23 1 1 6
√ 0005 CMP M BE 1 2 7
√ 0006 JNC NXT D2 3 3 10
0007 0A
0008 00
√ 0009 MOV A,M 7E 1 2 7
√ 000A NXT STA 2060 32 3 4 13
000B 60
000C 20
√ 000D HLT 76 1 2 5
Simulate
Start From → 0000
Run all At a Time
Step By Step
Registers Memory Devices
Memory Editor
Memory Range: 0000 ---- FFFF
Memory Address Value
0000 21
0001 50
0002 20
0003 7E
0004 23
0005 BE
0006 D2
0007 0A
0009 7E
000A 32
000B 60
000C 20
000D 76
2050 6A
2051 F4
2060 F4
Show entire memory content
Show only loaded memory location
Store directly to specified memory location
Created by : Jubin Mitra

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
        MVI B,00H
NXT1:   MOV A,M
        CMP B
        JNC NEXT
        MOV B,A
NEXT:   INX H
        DCR C
        JNZ NXT1
        MOV A,B
        STA 2060H
        HLT
```

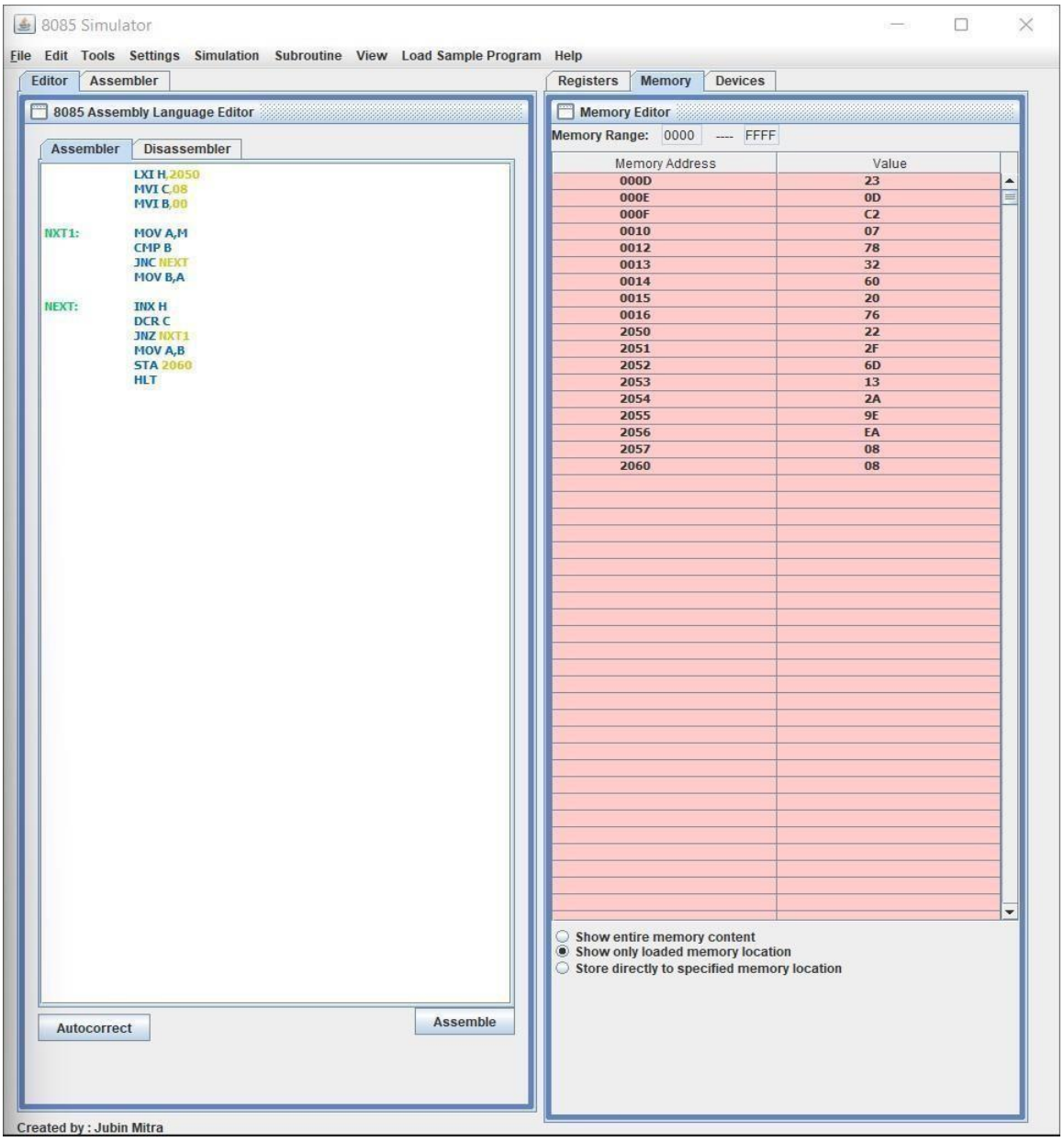
OBSERVATIONS:

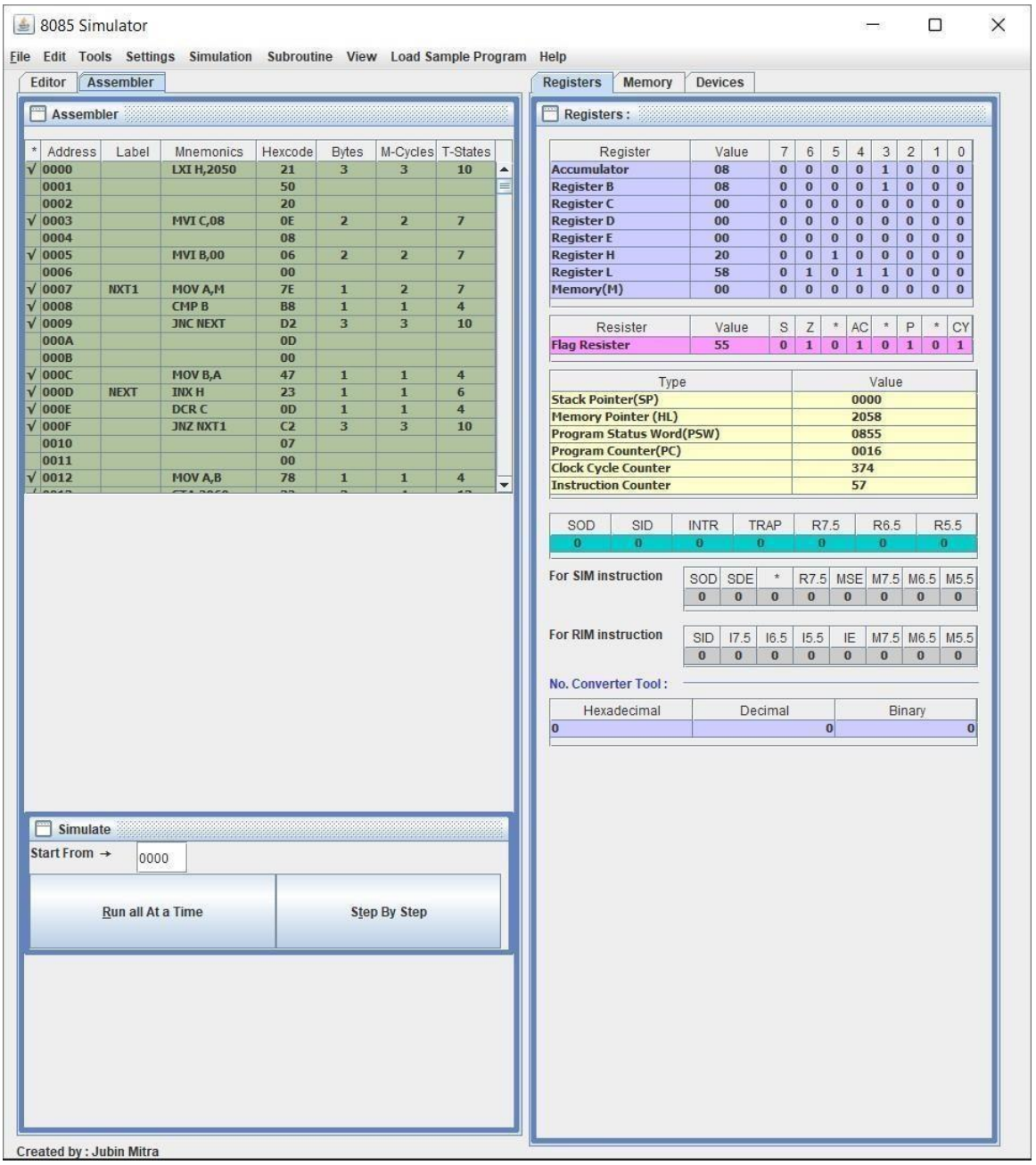
Input:

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

Output:

2060H: EAH


8085 Simulator
File Edit Tools Settings Simulation Subroutine View Load Sample Program Help
Editor Assembler
8085 Assembly Language Editor
Assembler Disassembler
LXI H,2050
MVI C,08
MVI B,00
NXT1: MOV A,M
CMP B
JNC NEXT
MOV B,A
NEXT: INX H
DCR C
JNZ NXT1
MOV A,B
STA 2060
HLT
Autocorrect
Assemble
Created by : Jubin Mitra
Registers Memory Devices
Memory Editor
Memory Range: 0000 ---- FFFF
Memory Address Value
000D 23
000E 0D
000F C2
0010 07
0012 78
0013 32
0014 60
0015 20
0016 76
2050 22
2051 2F
2052 6D
2053 13
2054 2A
2055 9E
2056 EA
2057 08
2060 08
Show entire memory content
Show only loaded memory location
Store directly to specified memory location



*	Address	Label	Mnemonics	Hexcode	Bytes	M-Cycles	T-States
√	0000		LXI H,2050	21	3	3	10
	0001			50			
	0002			20			
√	0003		MVI C,08	0E	2	2	7
	0004			08			
√	0005		MVI B,00	06	2	2	7
	0006			00			
√	0007	NXT1	MOV A,M	7E	1	2	7
√	0008		CMP B	B8	1	1	4
√	0009		JNC NEXT	D2	3	3	10
	000A			0D			
	000B			00			
√	000C		MOV B,A	47	1	1	4
√	000D	NEXT	INX H	23	1	1	6
√	000E		DCR C	0D	1	1	4
√	000F		JNZ NXT1	C2	3	3	10
	0010			07			
	0011			00			
√	0012		MOV A,B	78	1	1	4

Register	Value	7	6	5	4	3	2	1	0
Accumulator	08	0	0	0	0	1	0	0	0
Register B	08	0	0	0	0	1	0	0	0
Register C	00	0	0	0	0	0	0	0	0
Register D	00	0	0	0	0	0	0	0	0
Register E	00	0	0	0	0	0	0	0	0
Register H	20	0	0	1	0	0	0	0	0
Register L	58	0	1	0	1	1	0	0	0
Memory(M)	00	0	0	0	0	0	0	0	0

Resister	Value	S	Z	*	AC	*	P	*	CY
Flag Resister	55	0	1	0	1	0	1	0	1

Type	Value
Stack Pointer(SP)	0000
Memory Pointer (HL)	2058
Program Status Word(PSW)	0855
Program Counter(PC)	0016
Clock Cycle Counter	374
Instruction Counter	57

SOD	SID	INTR	TRAP	R7.5	R6.5	R5.5
0	0	0	0	0	0	0

For SIM instruction

SOD	SDE	*	R7.5	MSE	M7.5	M6.5	M5.5
0	0	0	0	0	0	0	0

For RIM instruction

SID	I7.5	I6.5	I5.5	IE	M7.5	M6.5	M5.5
0	0	0	0	0	0	0	0

No. Converter Tool :

Hexadecimal	Decimal	Binary
0	0	0

PRACTICAL-8

AIM: Write a program to arrange the numbers in ascending order. The numbers are stored at 2050H onwards. [5 numbers]

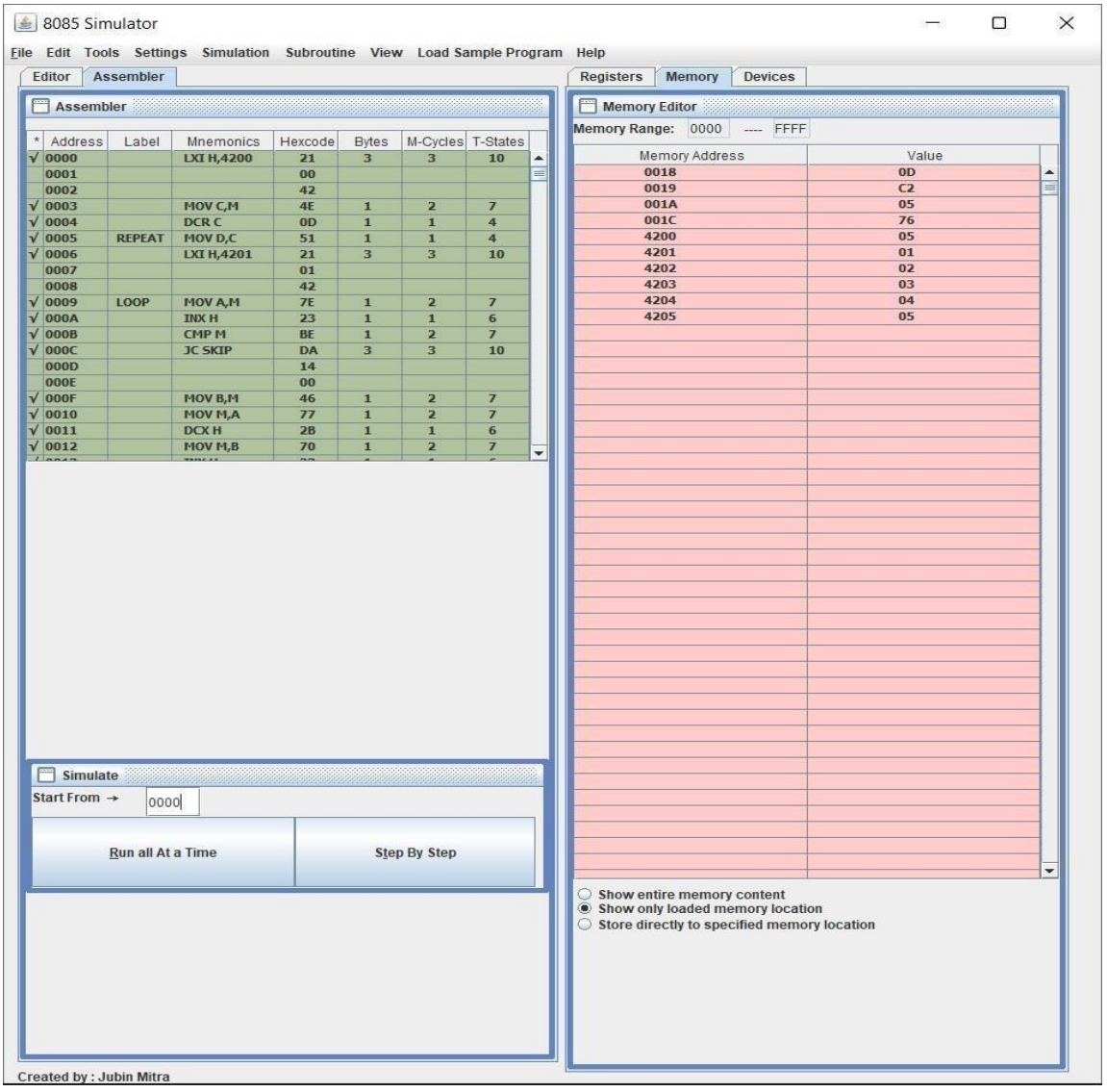
PROGRAM:

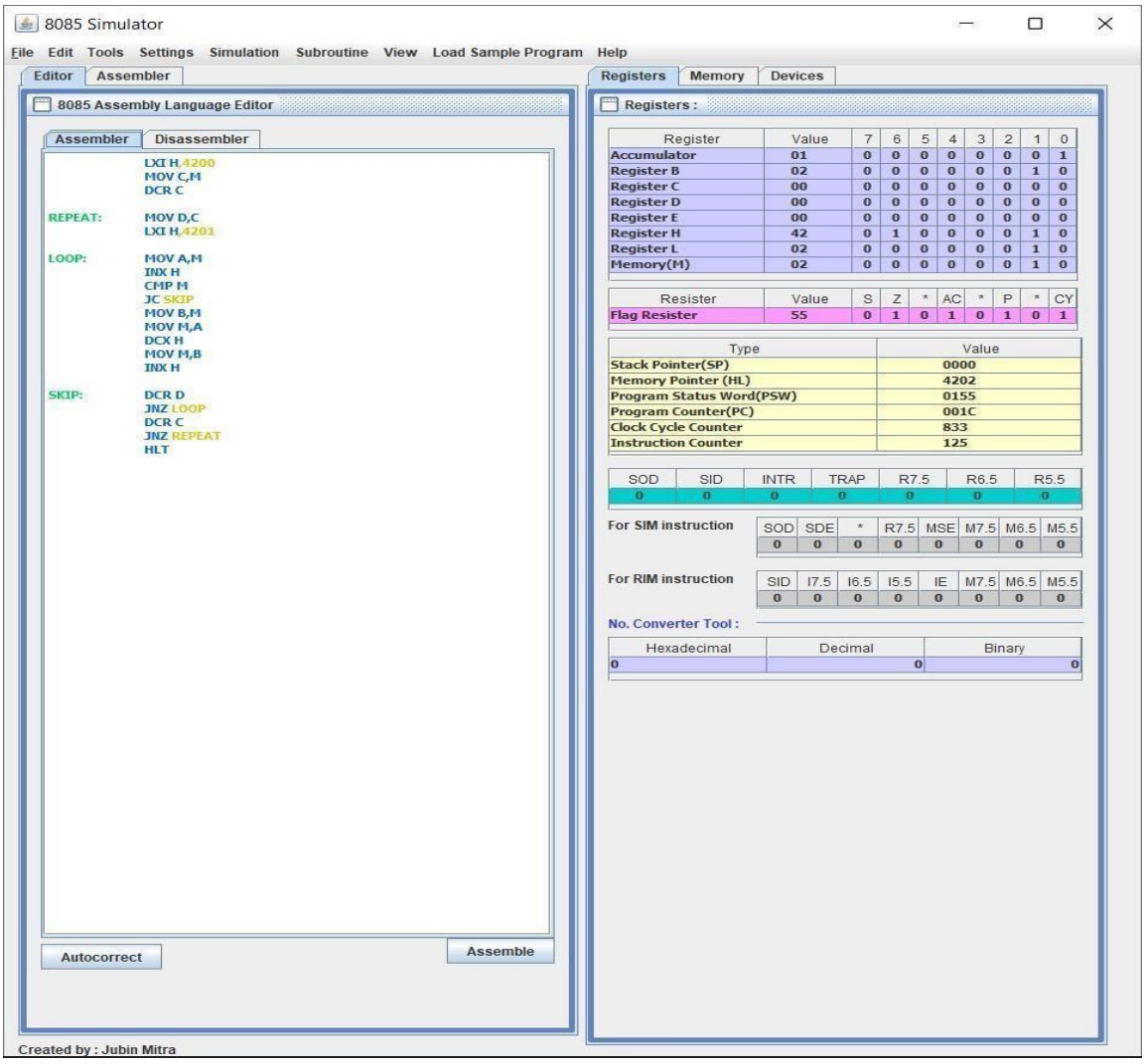
```
        LXI H,4200
        MOV C,M
        DCR C

REPEAT: MOV D,C
        LXI H,4201

LOOP:   MOV A,M
        INX H
        CMP  M
        JC SKIP
        MOV B,M
        MOV M,A
        DCX H
        MOV M,B
        INX H

SKIP:   DCR D
        JNZ LOOP
        DCR C
        JNZ REPEAT
        HLT
```

INPUT: 4200H: 05(Array size)
4201H: 05H
4202H: 04H
4203H: 03H
4204H: 01H
4205H: 02H

OUTPUT: 4200H: 05(Array Size)

4201H: 01H
4202H: 02H
4203H: 03H
4204H: 04H
4205H: 05H




8085 Simulator
File Edit Tools Settings Simulation Subroutine View Load Sample Program Help
Editor Assembler
8085 Assembly Language Editor
Assembler Disassembler
LXI H,4200
MOV C,M
DCR C
REPEAT: MOV D,C
LXI H,4201
LOOP: MOV A,M
INX H
CMP M
JC SKIP
MOV B,M
MOV M,A
DCX H
MOV M,B
INX H
SKIP: DCR D
JNZ LOOP
DCR C
JNZ REPEAT
HLT
Autocorrect
Assemble
Registers Memory Devices
Registers :
Register Value 7 6 5 4 3 2 1 0
Accumulator 01 0 0 0 0 0 0 0 1
Register B 02 0 0 0 0 0 0 1 0
Register C 00 0 0 0 0 0 0 0 0
Register D 00 0 0 0 0 0 0 0 0
Register E 00 0 0 0 0 0 0 0 0
Register H 42 0 1 0 0 0 0 1 0
Register L 02 0 0 0 0 0 0 1 0
Memory(M) 02 0 0 0 0 0 0 1 0
Resister Value S Z * AC * P * CY
Flag Resister 55 0 1 0 1 0 1 0 1
Type Value
Stack Pointer(SP) 0000
Memory Pointer (HL) 4202
Program Status Word(PSW) 0155
Program Counter(PC) 001C
Clock Cycle Counter 833
Instruction Counter 125
SOD SID INTR TRAP R7.5 R6.5 R5.5
0 0 0 0 0 0 0
For SIM instruction
SOD SDE * R7.5 MSE M7.5 M6.5 M5.5
0 0 0 0 0 0 0 0
For RIM instruction
SID I7.5 I6.5 I5.5 IE M7.5 M6.5 M5.5
0 0 0 0 0 0 0 0
No. Converter Tool :
Hexadecimal Decimal Binary
0 0 0
Created by : Jubin Mitra

PRACTICAL-9

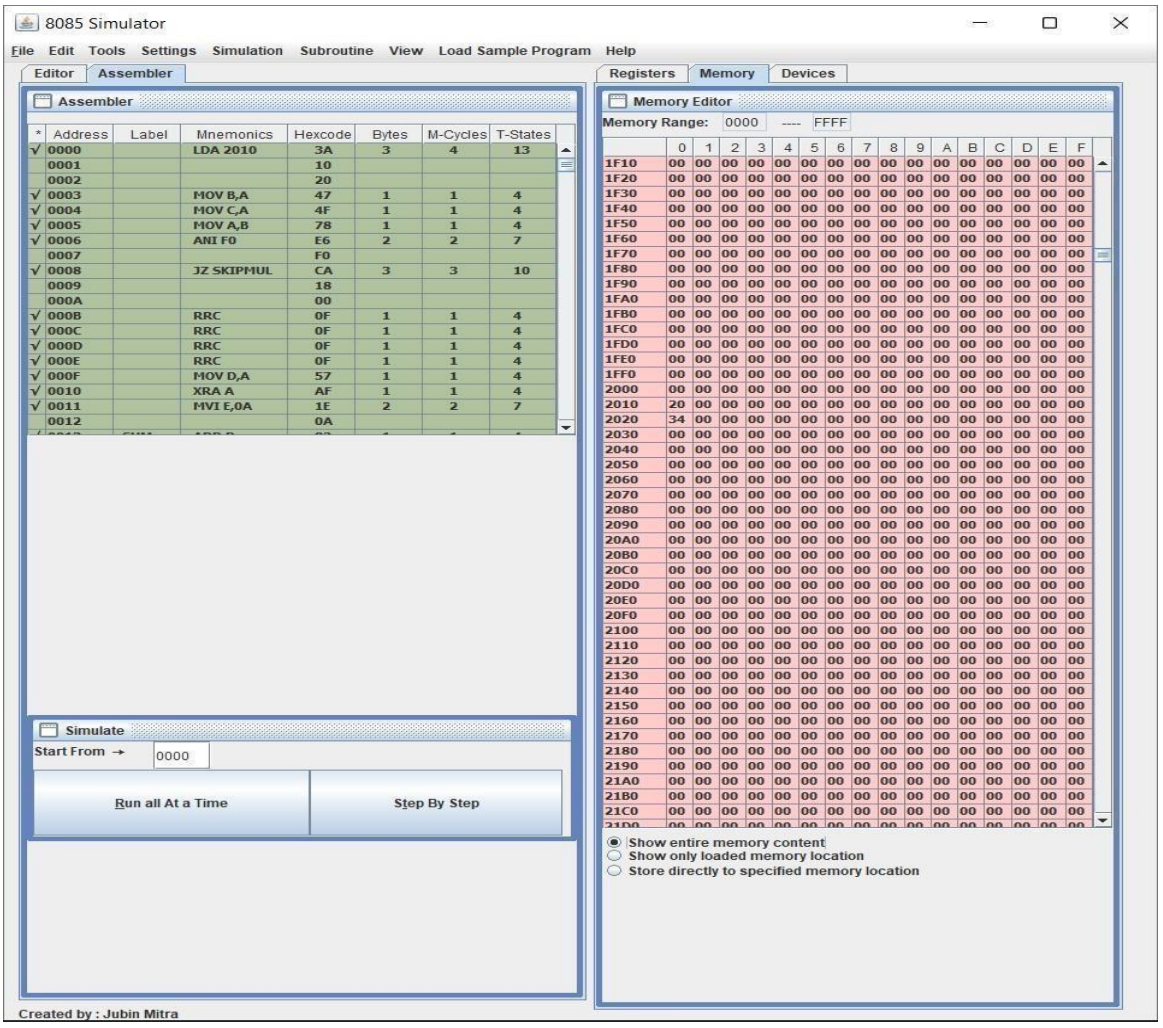
AIM: Write a program to convert a number from BCD to Binary.

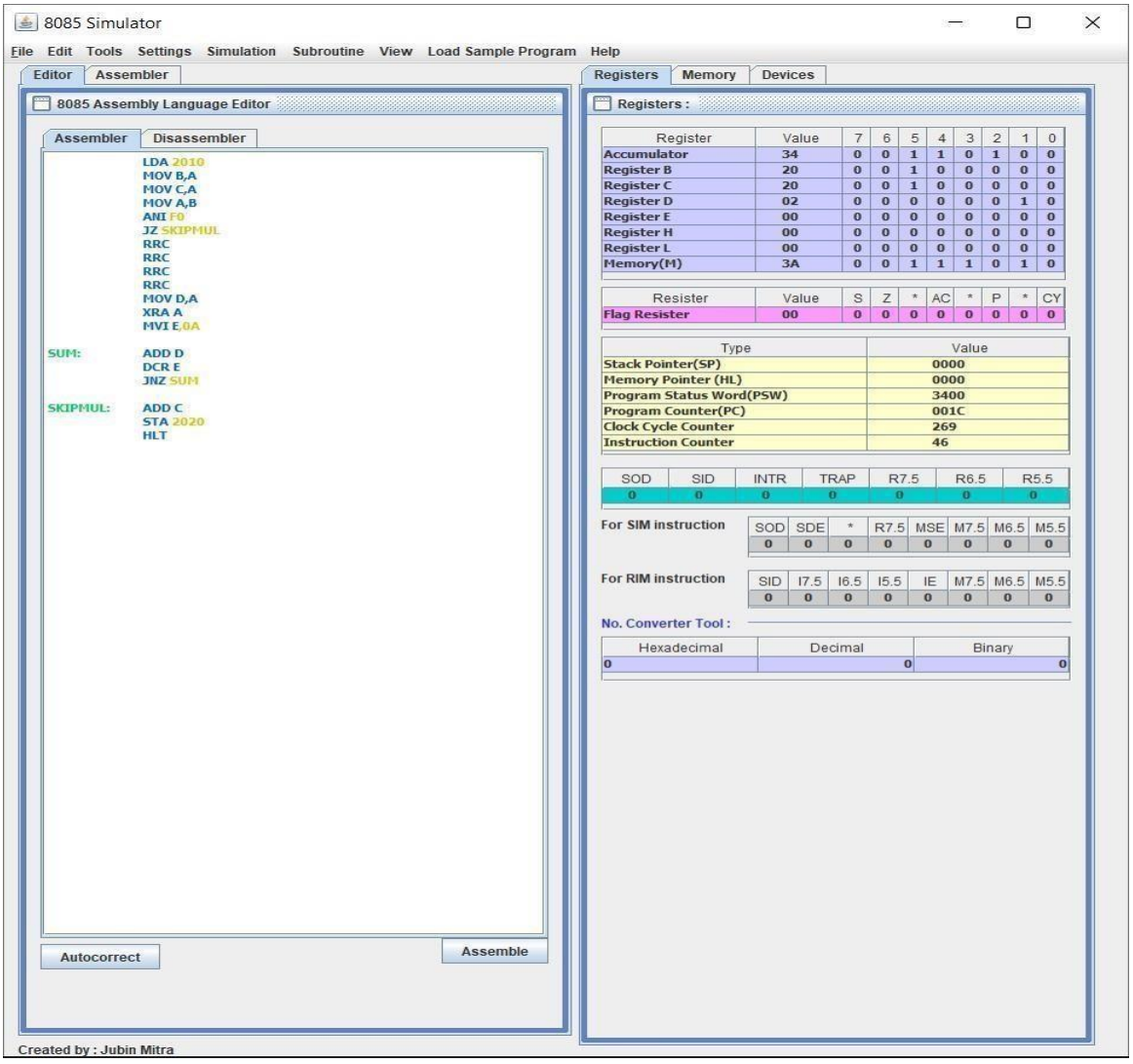
PROGRAM:

```
        LDA 2010H
        MOV B,A
        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: 20H

OUTPUT: 2020H: 34H


8085 Simulator
File Edit Tools Settings Simulation Subroutine View Load Sample Program Help
Editor Assembler
Assembler
Simulate
Start From → 0000
Run all At a Time
Step By Step
Created by : Jubin Mitra
Registers Memory Devices
Memory Editor
Memory Range: 0000 ---- FFFF
Show entire memory content
Show only loaded memory location
Store directly to specified memory location


8085 Simulator
File Edit Tools Settings Simulation Subroutine View Load Sample Program Help
Editor Assembler
8085 Assembly Language Editor
Assembler Disassembler
LDA 2010
MOV B,A
MOV C,A
MOV A,B
ANI F0
JZ SKIPMUL
RRC
RRC
RRC
RRC
MOV D,A
XRA A
MVI E,0A
SUM: ADD D
DCR E
JNZ SUM
SKIPMUL: ADD C
STA 2020
HLT
Autocorrect
Assemble
Registers Memory Devices
Registers :
Register Value 7 6 5 4 3 2 1 0
Accumulator 34 0 0 1 1 0 1 0 0
Register B 20 0 0 1 0 0 0 0 0
Register C 20 0 0 1 0 0 0 0 0
Register D 02 0 0 0 0 0 0 1 0
Register E 00 0 0 0 0 0 0 0 0
Register H 00 0 0 0 0 0 0 0 0
Register L 00 0 0 0 0 0 0 0 0
Memory(M) 3A 0 0 1 1 1 0 1 0
Resister Value S Z * AC * P * CY
Flag Resister 00 0 0 0 0 0 0 0 0
Type Value
Stack Pointer(SP) 0000
Memory Pointer (HL) 0000
Program Status Word(PSW) 3400
Program Counter(PC) 001C
Clock Cycle Counter 269
Instruction Counter 46
SOD SID INTR TRAP R7.5 R6.5 R5.5
0 0 0 0 0 0 0
For SIM instruction
SOD SDE * R7.5 MSE M7.5 M6.5 M5.5
0 0 0 0 0 0 0 0
For RIM instruction
SID I7.5 I6.5 I5.5 IE M7.5 M6.5 M5.5
0 0 0 0 0 0 0 0
No. Converter Tool :
Hexadecimal Decimal Binary
0 0 0
Created by : Jubin Mitra

PRACTICAL-10

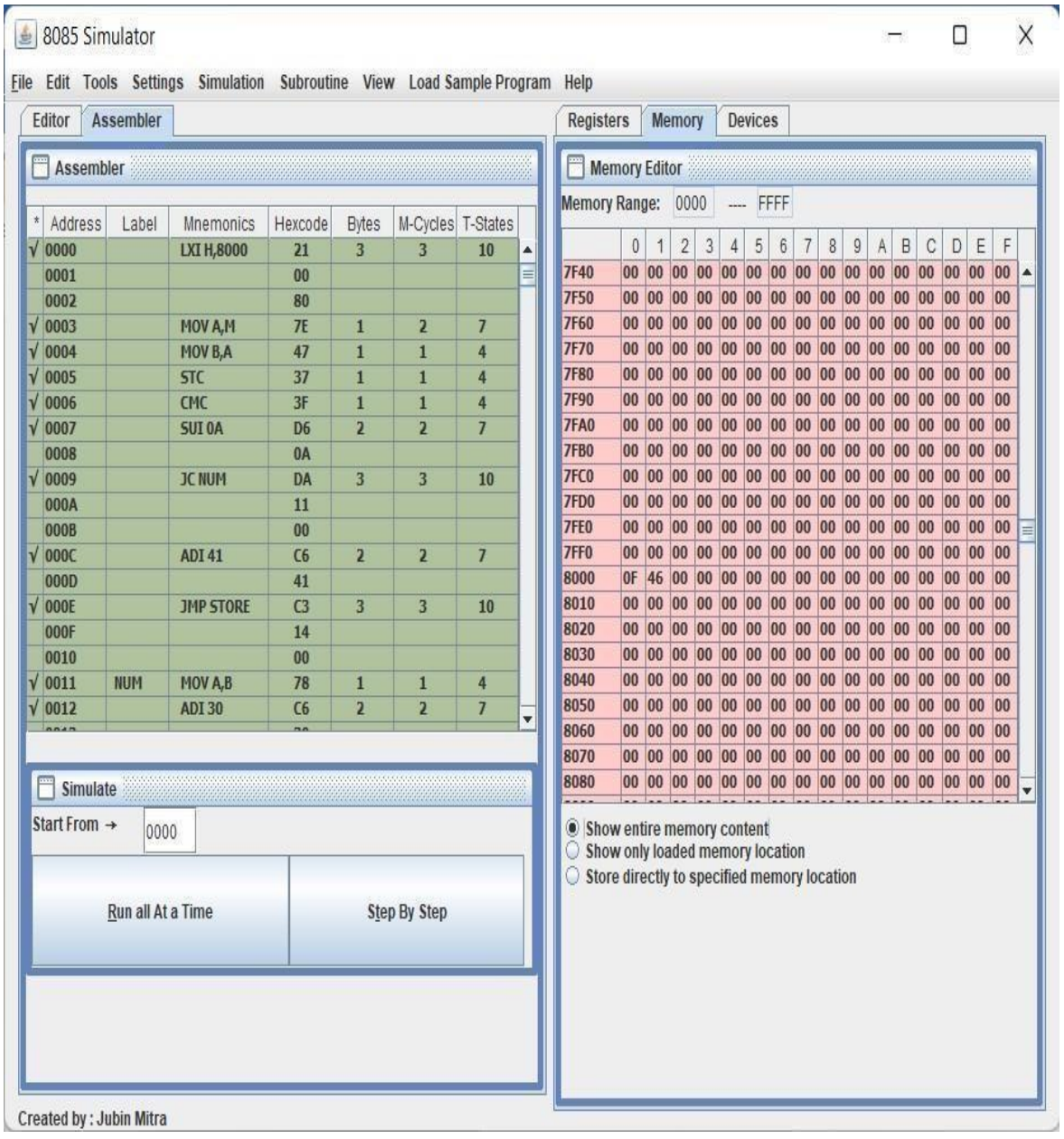
AIM: Write a program to convert from binary to ASCII.

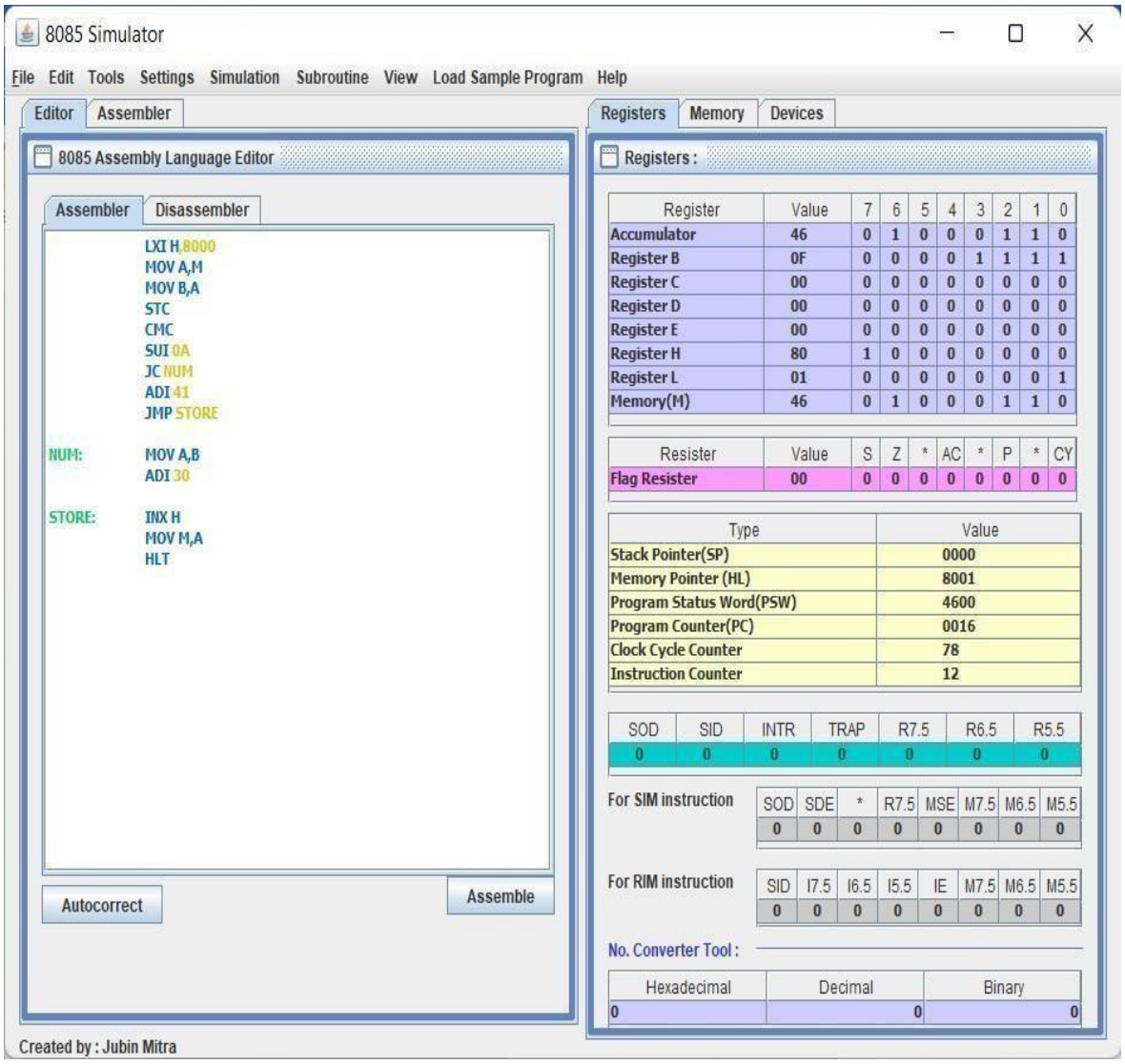
PROGRAM:

```
        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


8085 Simulator
File Edit Tools Settings Simulation Subroutine View Load Sample Program Help
Editor Assembler
Registers Memory Devices
Assembler
* Address Label Mnemonics Hexcode Bytes M-Cycles T-States
√ 0000 LXI H,8000 21 3 3 10
0001 00
0002 80
√ 0003 MOV A,M 7E 1 2 7
√ 0004 MOV B,A 47 1 1 4
√ 0005 STC 37 1 1 4
√ 0006 CMC 3F 1 1 4
√ 0007 SUI 0A D6 2 2 7
0008 0A
√ 0009 JC NUM DA 3 3 10
000A 11
000B 00
√ 000C ADI 41 C6 2 2 7
000D 41
√ 000E JMP STORE C3 3 3 10
000F 14
0010 00
√ 0011 NUM MOV A,B 78 1 1 4
√ 0012 ADI 30 C6 2 2 7
Simulate
Start From → 0000
Run all At a Time
Step By Step
Memory Editor
Memory Range: 0000 ---- FFFF
0 1 2 3 4 5 6 7 8 9 A B C D E F
7F40 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
7F50 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
7F60 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
7F70 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
7F80 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
7F90 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
7FA0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
7FB0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
7FC0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
7FD0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
7FE0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
7FF0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8000 0F 46 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8010 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8020 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8030 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8040 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8050 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8060 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8070 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8080 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
Show entire memory content
Show only loaded memory location
Store directly to specified memory location
Created by : Jubin Mitra


8085 Simulator
File Edit Tools Settings Simulation Subroutine View Load Sample Program Help
Editor Assembler
8085 Assembly Language Editor
Assembler Disassembler
LXI H,8000
MOV A,M
MOV B,A
STC
CMC
SUI 0A
JC NUM
ADI 41
JMP STORE
NUM: MOV A,B
ADI 30
STORE: INX H
MOV M,A
HLT
Autocorrect
Assemble
Created by : Jubin Mitra
Registers Memory Devices
Registers :
Register Value 7 6 5 4 3 2 1 0
Accumulator 46 0 1 0 0 0 1 1 0
Register B 0F 0 0 0 0 1 1 1 1
Register C 00 0 0 0 0 0 0 0 0
Register D 00 0 0 0 0 0 0 0 0
Register E 00 0 0 0 0 0 0 0 0
Register H 80 1 0 0 0 0 0 0 0
Register L 01 0 0 0 0 0 0 0 1
Memory(M) 46 0 1 0 0 0 1 1 0
Resister Value S Z * AC * P * CY
Flag Resister 00 0 0 0 0 0 0 0 0
Type Value
Stack Pointer(SP) 0000
Memory Pointer (HL) 8001
Program Status Word(PSW) 4600
Program Counter(PC) 0016
Clock Cycle Counter 78
Instruction Counter 12
SOD SID INTR TRAP R7.5 R6.5 R5.5
0 0 0 0 0 0 0
For SIM instruction
SOD SDE * R7.5 MSE M7.5 M6.5 M5.5
0 0 0 0 0 0 0 0
For RIM instruction
SID I7.5 I6.5 I5.5 IE M7.5 M6.5 M5.5
0 0 0 0 0 0 0 0
No. Converter Tool :
Hexadecimal Decimal Binary
0 0 0

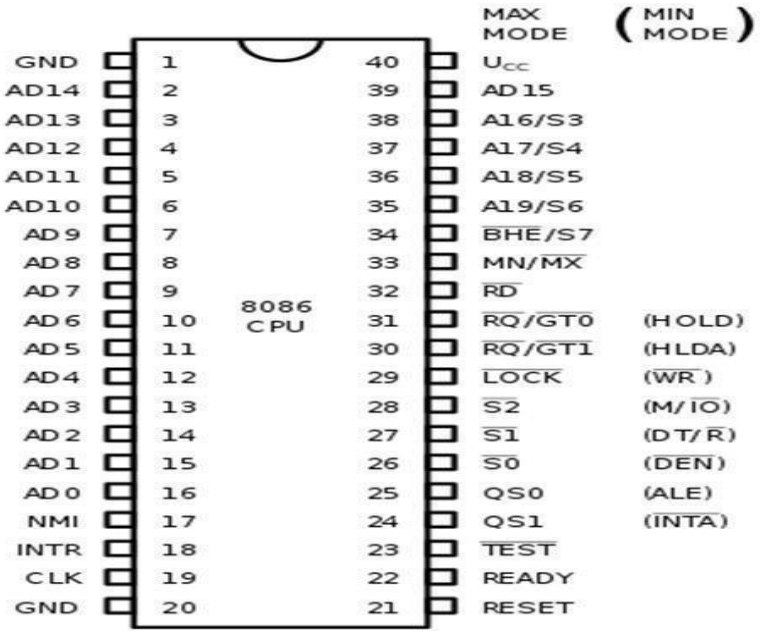
PRACTICAL-11

AIM: Introduction to 8086 Microprocessor.

Features of 8086

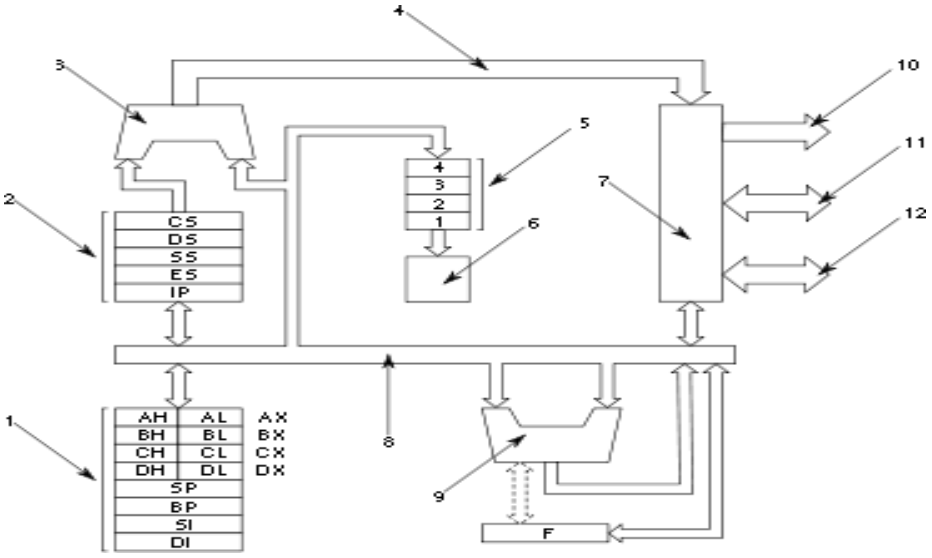
- 8086 is a 16bit processor. It's ALU, internal registers works with 16bit binary word.
- 8086 has a 16bit data bus. It can read or write data to a memory/port either 16bits or 8 bit at a time
- 8086 has a 20bit address bus which means, it can address upto 1MBmemory location
- Frequency range of 8086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

We have studied the basic structure of the 8086 micrprocessor.