__/__/____

NATIONAL INSTITUTE OF TECHNOLOGY SILCHAR

CACHAR, ASSAM

LABORATORY EXCERCISE BOOK

B.TECH. IIIRD SEM.

NAME: SUBHOJIT GHIMIRE
SCH. ID.: 1912160
BRANCH: C.S.E. - B
SUBJECT: MICROPROCESSOR LAB
CODE : EE224

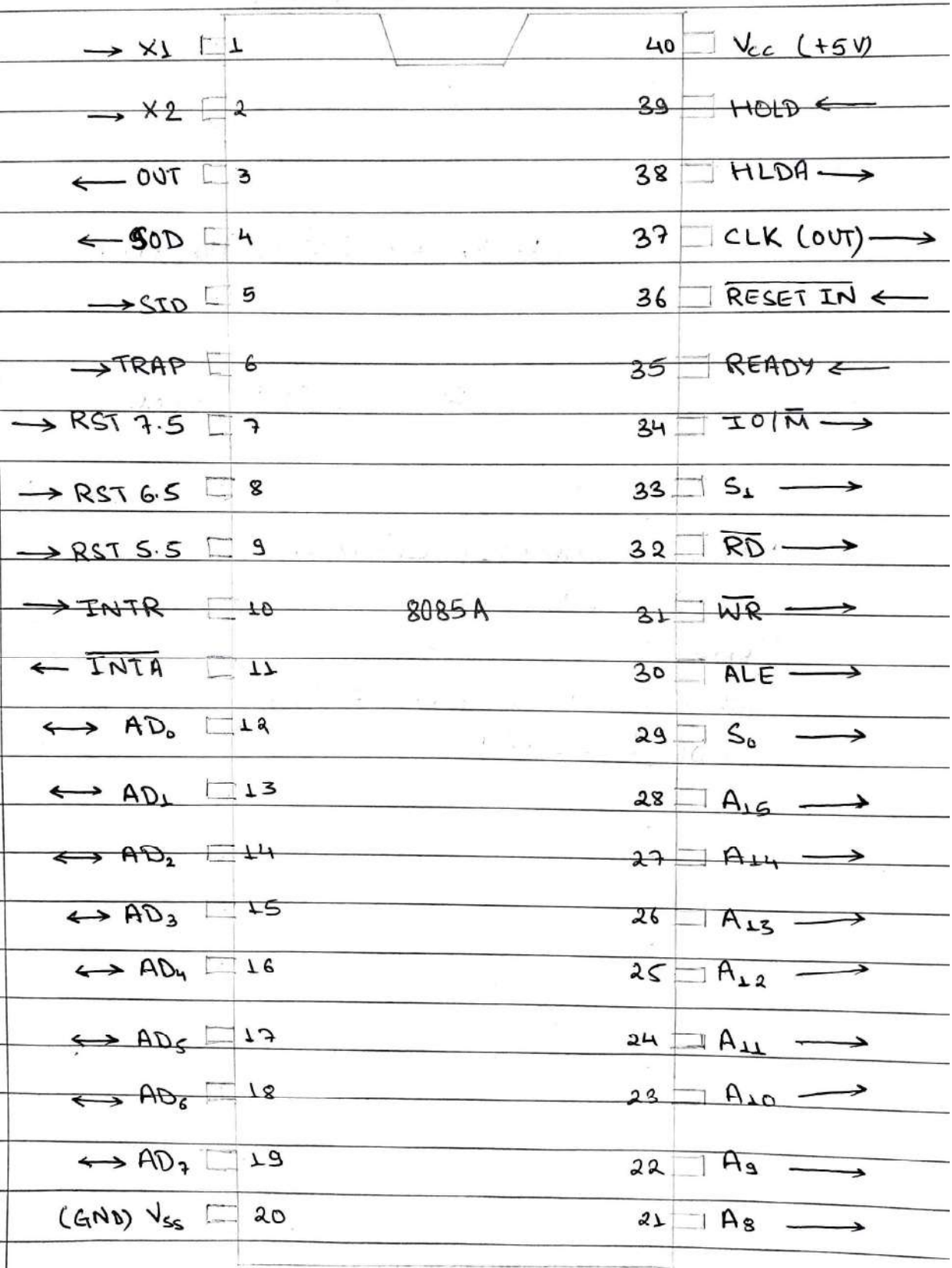


figure: Pin Diagram of 8085 Microprocessor

__/__/____

EXPLANATION OF 8085 MP PINS:

1. A8-A15 (Output): These are address bus and used for the most significant bits of the memory address.

2. $AD_0 - AD_7$ (Input/Output): These are time multiplexed address data bus and are used for the least significant 8 bits of the memory address during the first clock cycle and then for data during the second and third clock cycles.

3. ALE (Address Latch Enable): It goes high during the 1st clock cycle of a machine. It enables the lower 8 bits of the address to be latched either in the memory or external latch.

4. IO/M: It is a status signal; when it goes high, the address on the address bus is for I/O device, otherwise for memory.

5. S_1: These are status signals to distinguish various types of operation.

6. RD (Output): It is used to control read signal.

7. WR (Output): It is used to control write operation

8. HOLD (Input): It is used to indicate that another device is requesting the use of address and data bus.

9. HLDA (Output): It is an acknowledgement signal used to indicate HOLD request has been received.

10. INTR (Input): When it goes high, the microprocessor suspends its normal sequence of operations.

11. INTA (Output): It is an ~~input~~ interrupt acknowledgement signal sent by MP after INTR is received.

12. RST 5.5, 6.5, 7.5 and TRAP: These are various interrupt signals. Here, TRAP has the highest priority.

13. RESET IN (Input): It resets the PC to zero.

14. RESET OUT (Output): It indicates CPU being reset.

15. X_1, X_2 (Input): This circuitry is required to produce a suitable check for operation of the MP.

16. Clk (Output): It is clock output for the user. Its frequency is same at which processor operates.

17. SID (Input): It is used for data lines for serial input.

18. SOD (Output): It is used for data line for serial output.

19. Vcc: It is the +5 Volts supply.

20. Vss: It is a ground reference

15/09/2020
Experiment - 01

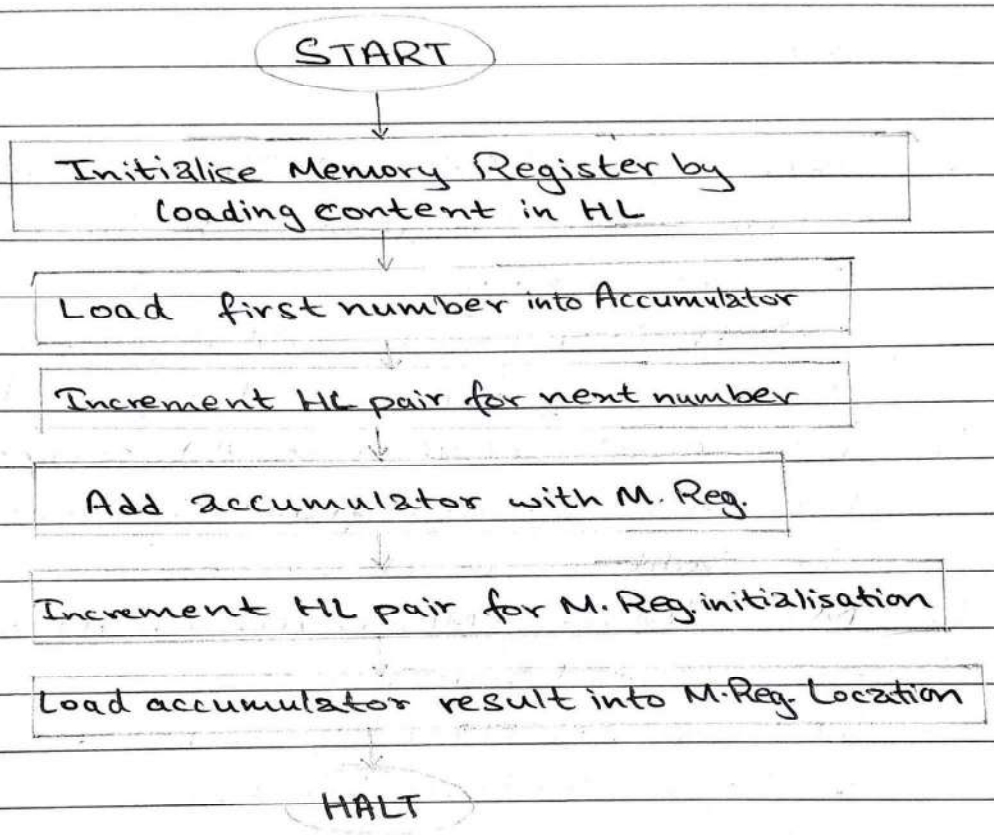
AIM: ADDITION OF TWO 8-BIT NUMBERS.

THEORY:

1. ORG Address (ORIGIN) Directive reserves the starting address for Program Code or data in specified memory array.
2. LXI H (LOAD ADDRESS PAIR IMMEDIATELY) loads 16-bit data in register pair designated by operand.
3. MOV A, M (MOVE M TO A) copies the data ~~byte~~ into accumulator from the memory specified by the address in H-L pair.
4. MVI (MOVE IMMEDIATE DATA) moves immediate value to specified register.
5. INR R (INCREMENT REGISTER) increment the specified register content by 1.
6. INX H (INCREMENT REGISTER PAIR) increments the contents of the register pair by one.
7. ADD M (ADDITION) adds the contents of memory to accumulator.
8. RST 1 (RESET) finishes the execution of the current instruction and stops any further execution.
9. DB (DEFINE BYTE) Directives defined to store values in specified memory array.

__/__/____

FLOW CHART:



PROGRAM:

#ORG 7000H

Memory Address	Mnemonics	Comments	Hexcod
7000	LXI H,	Get address of 1st no. in HL pair	21
7001			01
7002			75
7003	MOV A,M	Move number into accumulator	7E
7004	INX H	HL points address	23
7005	ADD M	Add the 2nd number	86
7006	INX H	HL points	23
7007	MOV M,A	Store result in	77
7008	RST 1	Terminate	CF
#ORG 7501H		Store input at the address	
#DB 12H, 13H		Get two 8 bit no. in successive location	

__/__/____

RESULT :

INPUT : 7501-13H, 7502-12H

OUTPUT : A-25H, 7503-25H

8085 Simulator

File Edit Tools Settings Simulation Subroutine View Load Sample Program Help

Editor Assembler

8085 Assembly Language Editor

Assembler Disassembler

```
# ORG 7000H
        LXI H, 7501
        MOV A,M
        INX H
        ADD M
        INX H
        MOV M,A
        RST 1
# ORG 7501H
# DB 12H, 13H
```

Autocorrect Assemble

Registers Memory Devices

Registers :

Register	Value	7	6	5	4	3	2	1	0
Accumulator	25	0	0	1	0	0	1	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	00	0	0	0	0	0	0	0	0
Register E	00	0	0	0	0	0	0	0	0
Register H	75	0	1	1	1	0	1	0	1
Register L	03	0	0	0	0	0	0	1	1
Memory(M)	25	0	0	1	0	0	1	0	1

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

Type	Value
Stack Pointer(SP)	FFFE
Memory Pointer (HL)	7503
Program Status Word(PSW)	2500
Program Counter(PC)	0008
Clock Cycle Counter	55
Instruction Counter	7

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

8085 Simulator

File Edit Tools Settings Simulation Subroutine View Load Sample Program Help

Editor Assembler

Registers Memory Devices

Registers :

Register	Value	7	6	5	4	3	2	1	0
Accumulator	25	0	0	1	0	0	1	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	00	0	0	0	0	0	0	0	0
Register E	00	0	0	0	0	0	0	0	0
Register H	75	0	1	1	1	0	1	0	1
Register L	03	0	0	0	0	0	0	1	1
Memory(M)	25	0	0	1	0	0	1	0	1

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

Type	Value
Stack Pointer(SP)	FFFE
Memory Pointer (HL)	7503
Program Status Word(PSW)	2500
Program Counter(PC)	0008
Clock Cycle Counter	55
Instruction Counter	7

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

Assembler

*	Address	Label	Mnemonics	Hexcode	Bytes	M-Cycles	T-States
√	7000		LXI H,7501	21	3	3	10
	7001			01			
	7002			75			
√	7003		MOV A,M	7E	1	2	7
√	7004		INX H	23	1	1	6
√	7005		ADD M	86	1	2	7
√	7006		INX H	23	1	1	6
√	7007		MOV M,A	77	1	2	7
√	7008		RST 1	CF	1	3	12

Simulate

Start From → 7000

Run all At a Time | Step By Step

Created by : Jubin Mitra

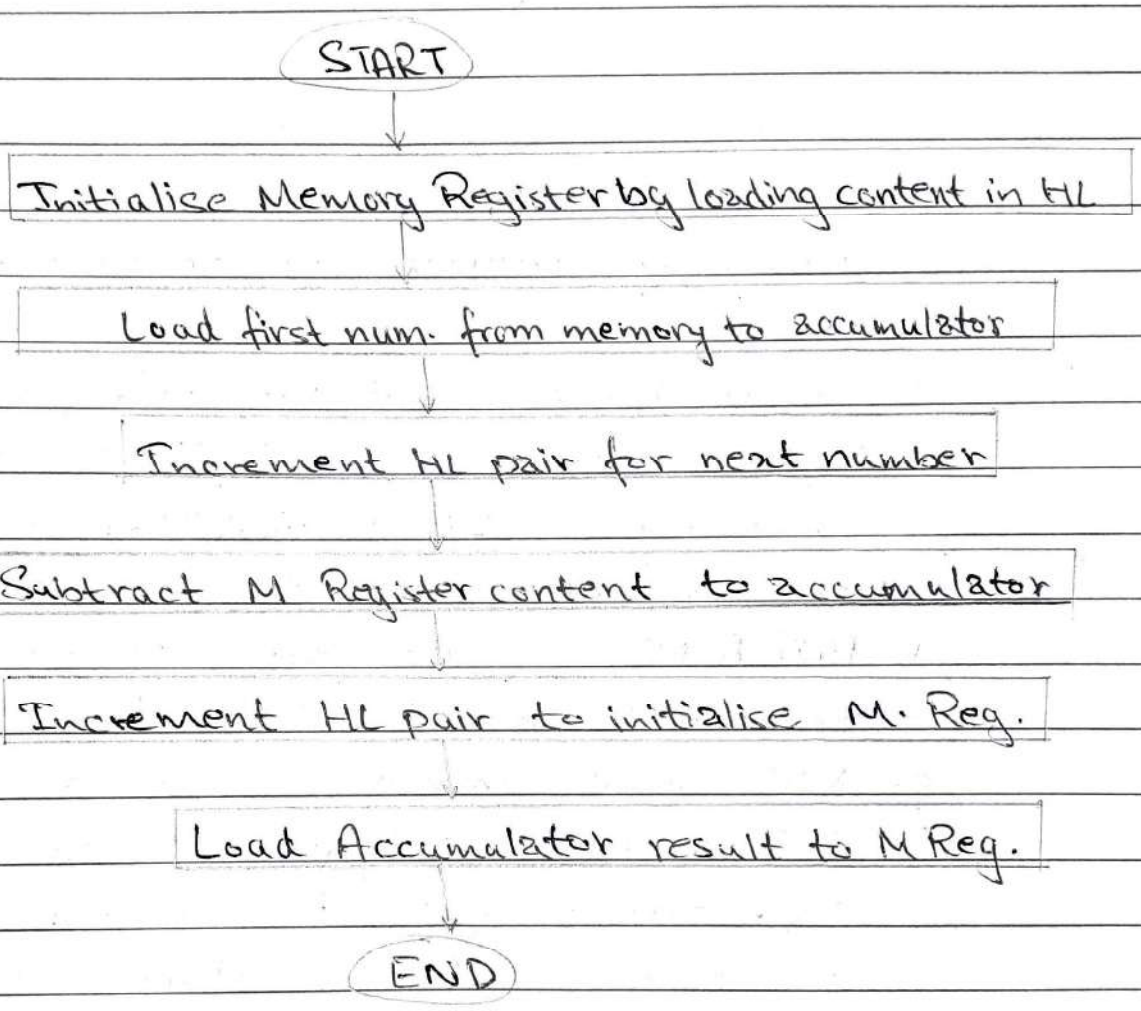
29/09/2020
Experiment-02

AIM: SUBTRACTION OF TWO 8-BIT NUMBERS AND TWO 16-BIT NUMBERS.

THEORY:

1. ORG Address Directive reserves the starting code address for Program Code or data in specified memory array.
2. LXI H loads 16 bit data in register pair designated by operand.
3. LHLD Address (LOAD HL PAIR DIRECT) loads 16 bit data from specified address to designate in register pair.
4. MOV A,M copies data byte into accumulator from the memory specified by the address in HL pair.
5. MVI moves immediate value to specified register.
6. SBB instruction subtracts specified register content and carry flag to Accumulator and stores result in the Accumulator.
7. INC Address instruction jumps the execution to the specified Address if carry flag is reset.
8. INR instruction increments specified register content by 1 value.
9. INX H increments contents of register pair by 1.
10. SUB M subtracts contents of register to accumulator.
11. STA address copies the contents of the accumulator to the memory location specified in the instruction.
12. SHLD Address instruction stores HL pair content to specified address.
13. RST 1 finishes the execution of the current instruction and stops further execution.

FLOWCHART:



PROGRAM (Subtraction of two 8-bit numbers)

Address	Mnemonics	Comment	Hexcode.
	#ORG, 7000H		
7000	LXI H, 7501	// Get address of 1st no. in HL pair	21
7001			01
7002			75
7003	MOV A, M	// Move no. into accumulator	7E
7004	INX H	// HL points to 7502 H	23
7005	SBB M	// Subtract 2nd no from 1st no.	9E
7006	INX H	// HL points to 7503H	23
7007	MOV M, A	// Move contents of acc. to memory	77

__/__/____

7008	RST 1	// Terminate	CF
	#ORG 7501H	// Store no. at address	
	#DB 20, 10	// Get two 8 bit no. at successive locations	

PROGRAM (Subtraction of two 16- bit numbers)

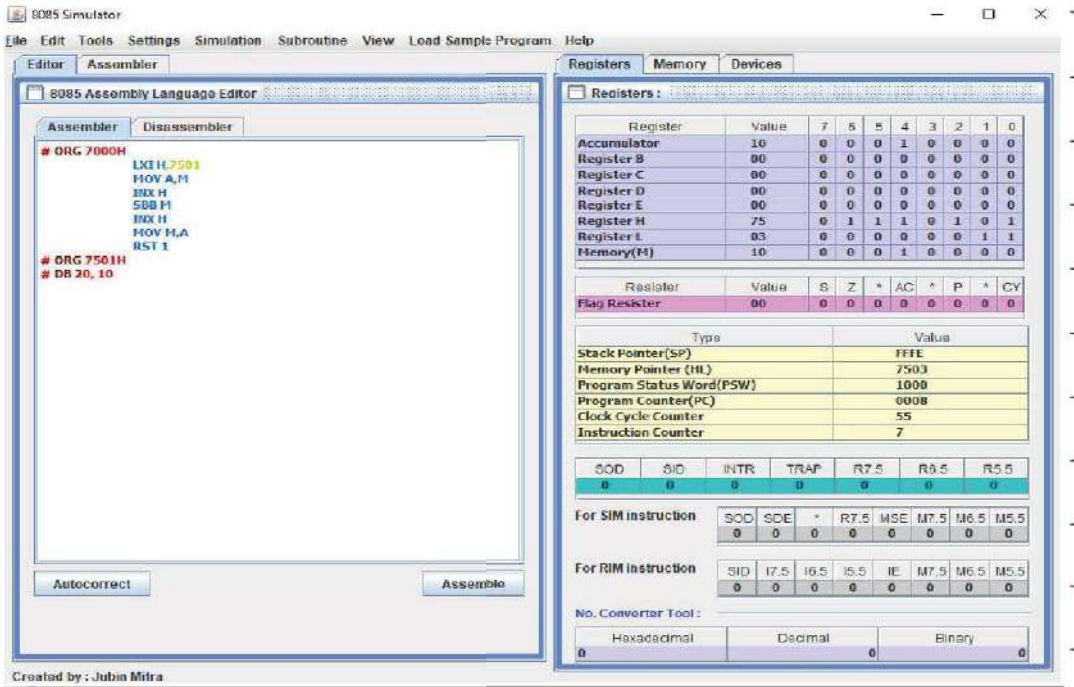
Address	Mnemonics	Comments	Hexcodes
	#ORG 7000H		
7000	LHLD 7501	// Get 1st 16 bit no. in HL Pair	2A
7001			01
7002			75
7003	XCHG	// Exchange HL pair with DE	EB
7004	LHLD 7503	// Get 2nd 16 bit no. in HL pair.	2A
7005			03
7006			75
7007	MOV A,E	// Get lower byte of 1st number	7B
7008	SUB L	// Subtract lower byte of 2nd number.	95
7009	MOV L,A	// Store the result in reg. L.	6F
700A	MOV A,D	// Get higher byte of 1st number.	7A
700B	SBB H	// Subtract higher byte of 2nd no. with borrow	9C
700C	MOV H,A	// Mov from acc. to H	67
700D	SHLD 7505	// Store 16 bit result at 7505H & 7506H	22
700E			05
700F			75
7010	RST 1	// Terminate	CF
	#ORG 7501H	// Stores inputs at the address	
	#DB 30,40,10,20	// Get two 16 bit nos. from successive locations.	

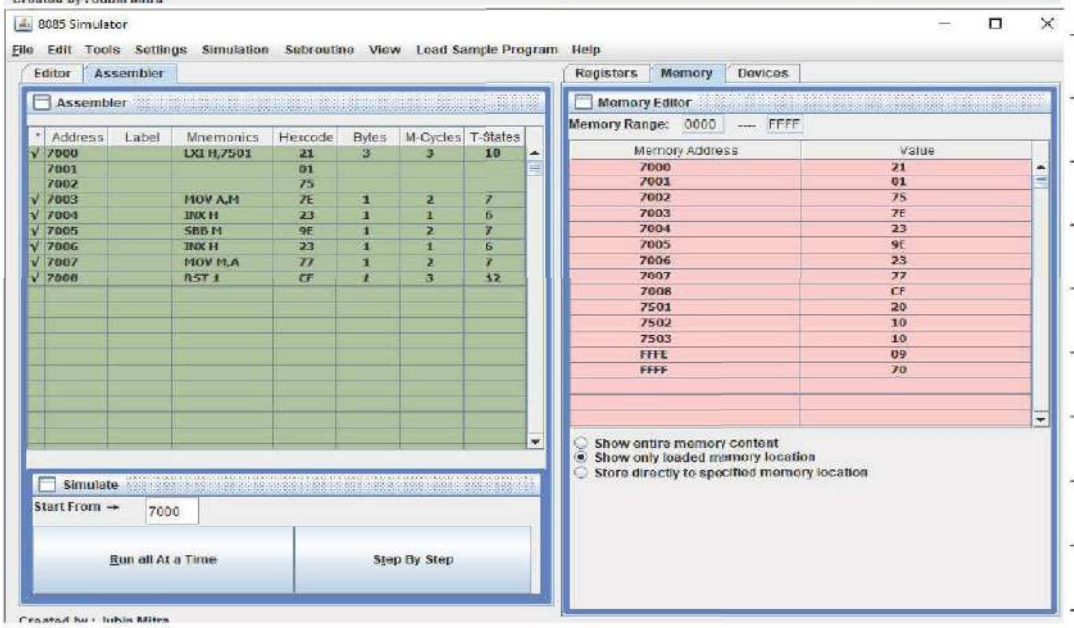
RESULT:

for subtraction of two 8-bit numbers,

INPUT - 7501 - 20H ; 7502 - 10H

OUTPUT - 7503 - 10H





for subtraction of two - 16-bit numbers.

INPUT - 7501 - 30H ; 7502 - 40H

7503 - 10H ; 7504 - 20H

OUTPUT - 7505 - 20H

7506 - 20H

for subtraction of two - 16-bit numbers.