INSTITUTE OF COMPUTER TECHNOLOGY

B-TECH COMPUTER SCIENCE ENGINEERING 2025-26

SUBJECT: MICROCONTROLLER & APPLICATION

NAME: Rahul Prajapati

ENRLL. NO: 23162171020

BRANCH: CYBER SECURITY

BATCH: 52

PRACTICAL_4

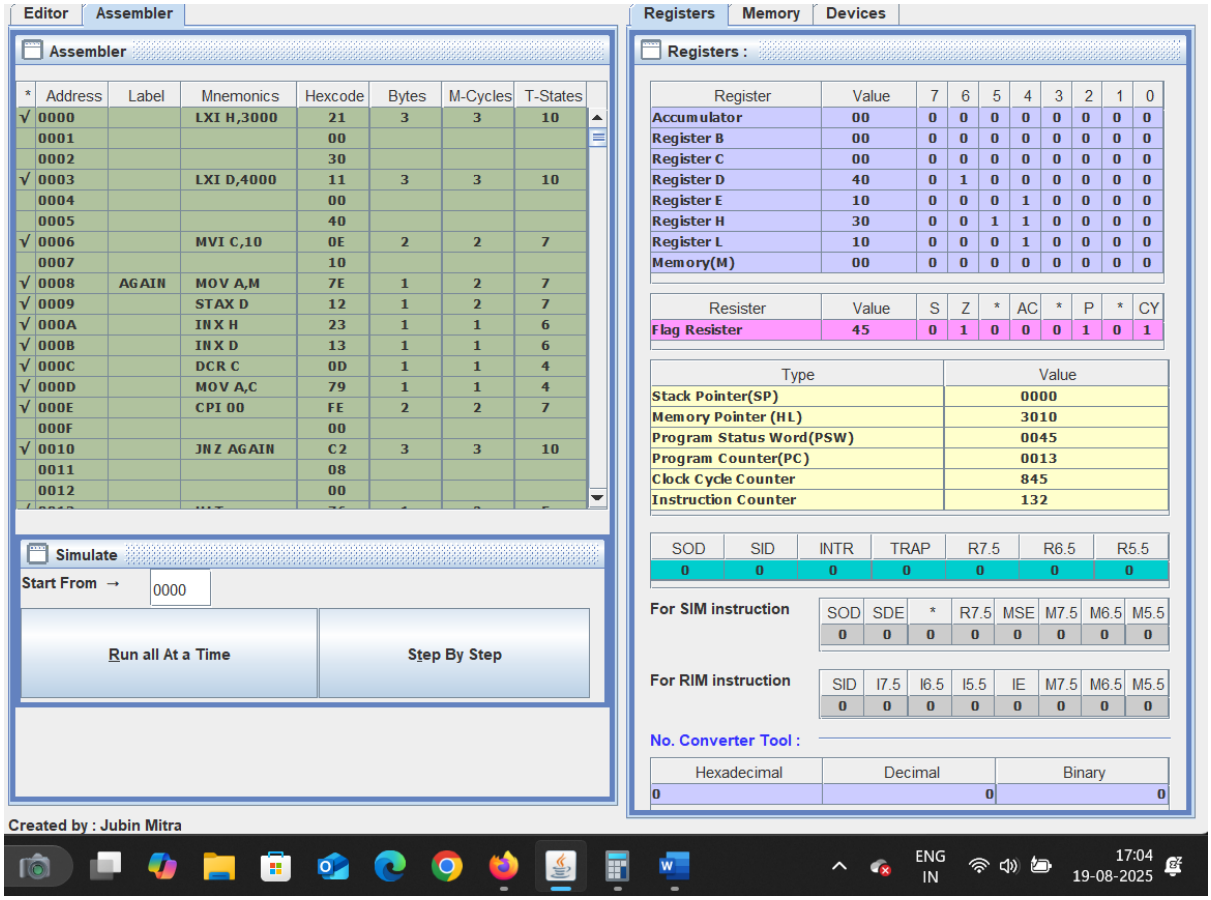
Aim:- Learning Techniques on Block Transfer of Data.

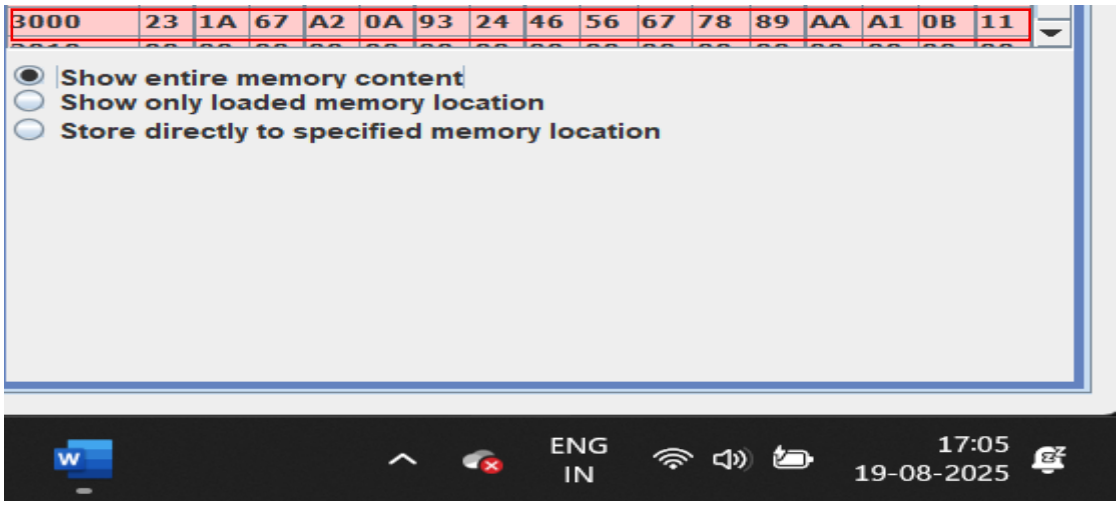
1. Write a program to transfer a block of data from 3000H to 4000H. The size of block is 16 bytes.

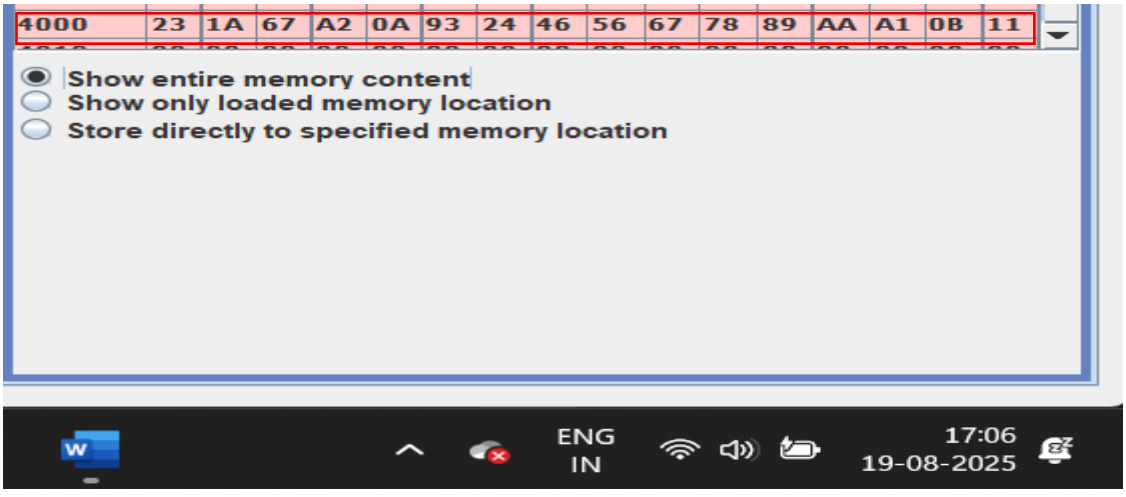
Assembler | Disassembler

```
# ORG 3000H
# DB
23H,1AH,67H,A2H,0AH,93H,24H,46H,56H,67H,78H,89H,AAH,A1H,0BH,11H
# ORG 0000H
            LXI H,3000//LOAD 16-BIT ADDRESS INTO HL PAIR
            LXI D,4000//LOAD 16-BIT ADDRESS INTO DE PAIR
            MVI C,10//STORE 10H IN REGISTER C FOR COUNTER

AGAIN:      MOV A,M//MOVE DATA OF MEMORY ADDRESS TO A
            STAX D//STORE DATA OF ACCUMULATOR TO DL PAIR'S
MEMORY ADDRESS
            INX H//INCREMENT HL PAIR ADDRESS
            INX D//INCREMENT DE PAIR ADDRESS
            DCR C//DECREMENT COUNTER
            MOV A,C//MOVE COUNTER TO ACCUMULATOR
            CPI 00//COMPARE WITH ACCUMULATOR
            JNZ AGAIN//JUMP IF ZERO FLAG IS RESET
            HLT//STOP
```


Editor
Assembler
Assembler
* Address Label Mnemonics Hexcode Bytes M-Cycles T-States
√ 0000 LXI H,3000 21 3 3 10
0001 00
0002 30
√ 0003 LXI D,4000 11 3 3 10
0004 00
0005 40
√ 0006 MVI C,10 0E 2 2 7
0007 10
√ 0008 AGAIN MOV A,M 7E 1 2 7
√ 0009 STAX D 12 1 2 7
√ 000A INX H 23 1 1 6
√ 000B INX D 13 1 1 6
√ 000C DCR C 0D 1 1 4
√ 000D MOV A,C 79 1 1 4
√ 000E CPI 00 FE 2 2 7
000F 00
√ 0010 JNZ AGAIN C2 3 3 10
0011 08
0012 00
Simulate
Start From →
0000
Run all At a Time
Step By Step
Registers
Memory
Devices
Registers :
Register Value 7 6 5 4 3 2 1 0
Accumulator 00 0 0 0 0 0 0 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 40 0 1 0 0 0 0 0 0
Register E 10 0 0 0 1 0 0 0 0
Register H 30 0 0 1 1 0 0 0 0
Register L 10 0 0 0 1 0 0 0 0
Memory(M) 00 0 0 0 0 0 0 0 0
Resister Value S Z * AC * P * CY
Flag Resister 45 0 1 0 0 0 1 0 1
Type Value
Stack Pointer(SP) 0000
Memory Pointer (HL) 3010
Program Status Word(PSW) 0045
Program Counter(PC) 0013
Clock Cycle Counter 845
Instruction Counter 132
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
ENG
IN
17:04
19-08-2025


3000 23 1A 67 A2 0A 93 24 46 56 67 78 89 AA A1 0B 11
Show entire memory content
Show only loaded memory location
Store directly to specified memory location
ENG
IN
17:05
19-08-2025


4000 23 1A 67 A2 0A 93 24 46 56 67 78 89 AA A1 0B 11
Show entire memory content
Show only loaded memory location
Store directly to specified memory location
ENG
IN
17:06
19-08-2025

2. Write a program to reverse a string stored in memory starting from FF00H. The length of the string is given in memory location 3FFF H. Store the reversed string at location 3050H.

Assembler | Disassembler

```
# ORG FF00H
# DB 21H,22H,23H,24H,25H,26H,27H,28H,29H,30H
# ORG 0000H
            MVI A,0A//STORE 0AH IN ACCUMULATOR
            STA 3FFF//STORE ACCUMULATOR'S DATA INTO 3FFFH
            LDA 3FFF//LOAD THE CONTENT OF 3FFFH INTO
ACCUMULATOR
            MOV C,A//MOVE ACCUMULATOR DATA INTO REGISTER C
FOR COUNTER
            LXI H,FF09//LOAD ADDRESS INTO HL PAIR
            LXI D,3050//LOAD ADDRESS INTO DL PAIR

AGAIN :     MOV A,M//MOVE MEMORY'S DATA INTO ACCUMULATOR
            STAX D//STORE ACCUMULATOR'S DATA INTO ADDRESS
WHICH IS IN DL PAIR
            DCX H//DECREMENT HL PAIR
            INX D//INCREMENT DE PAIR
            DCR C//DECREMENT COUNTER
            MOV A,C//MOVE COUNTER INTO ACCUMULATOR
            CPI 00//COMPARE 00H WITH ACCUMULATOR
            JNZ AGAIN//JUMP IF ZERO FLAG IS SET TO 0
            HLT//STOP
```

Editor | Assembler

Assembler

*	Address	Label	Mnemonics	Hexcode	Bytes	M-Cycles	T-States
√	0008		MOV C,A	4F	1	1	4
√	0009		LXI H,FF09	21	3	3	10
	000A			09			
	000B			FF			
√	000C		LXI D,3050	11	3	3	10
	000D			50			
	000E			30			
√	000F	AGAIN	MOV A,M	7E	1	2	7
√	0010		STAX D	12	1	2	7
√	0011		DCX H	2B	1	1	6
√	0012		INX D	13	1	1	6
√	0013		DCR C	0D	1	1	4
√	0014		MOV A,C	79	1	1	4
√	0015		CPI 00	FE	2	2	7
	0016			00			
√	0017		JNZ AGAIN	C2	3	3	10
	0018			0F			
	0019			00			
√	001A		HLT	76	1	2	5

Simulate

Start From → 0000

Run all At a Time | Step By Step

Registers | Memory | Devices

Registers :

Register	Value	7	6	5	4	3	2	1	0
Accumulator	00	0	0	0	0	0	0	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	30	0	0	1	1	0	0	0	0
Register E	5A	0	1	0	1	1	0	1	0
Register H	FE	1	1	1	1	1	1	1	0
Register L	FF	1	1	1	1	1	1	1	1
Memory(M)	00	0	0	0	0	0	0	0	0

Resister	Value	S	Z	*	AC	*	P	*	CY
Flag Resister	45	0	1	0	0	0	1	0	1

Type	Value
Stack Pointer(SP)	0000
Memory Pointer (HL)	FEFF
Program Status Word(PSW)	0045
Program Counter(PC)	001A
Clock Cycle Counter	569
Instruction Counter	87

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

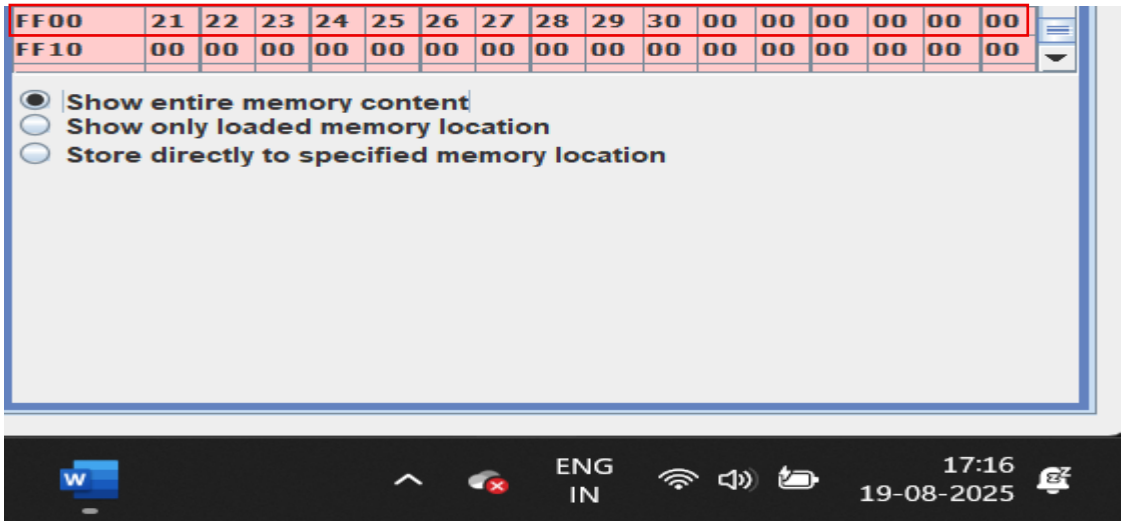
For RIM instruction

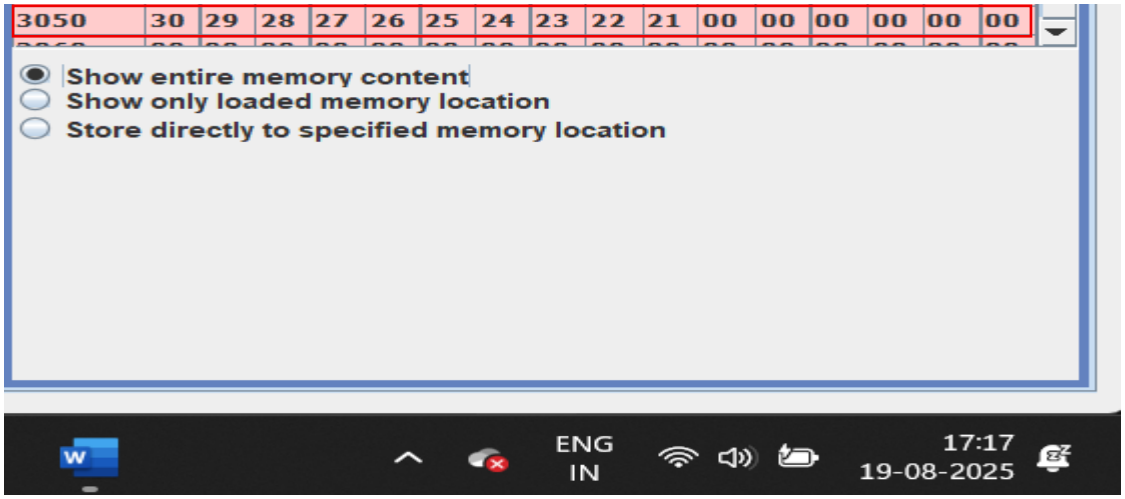
SID	I7.5	I6.5	I5.5	IE	M7.5	M6.5	M5.5
0	0	0	0	0	0	0	0

No. Converter Tool :

Hexadecimal	Decimal	Binary
0	0	0

Created by : Jubin Mitra



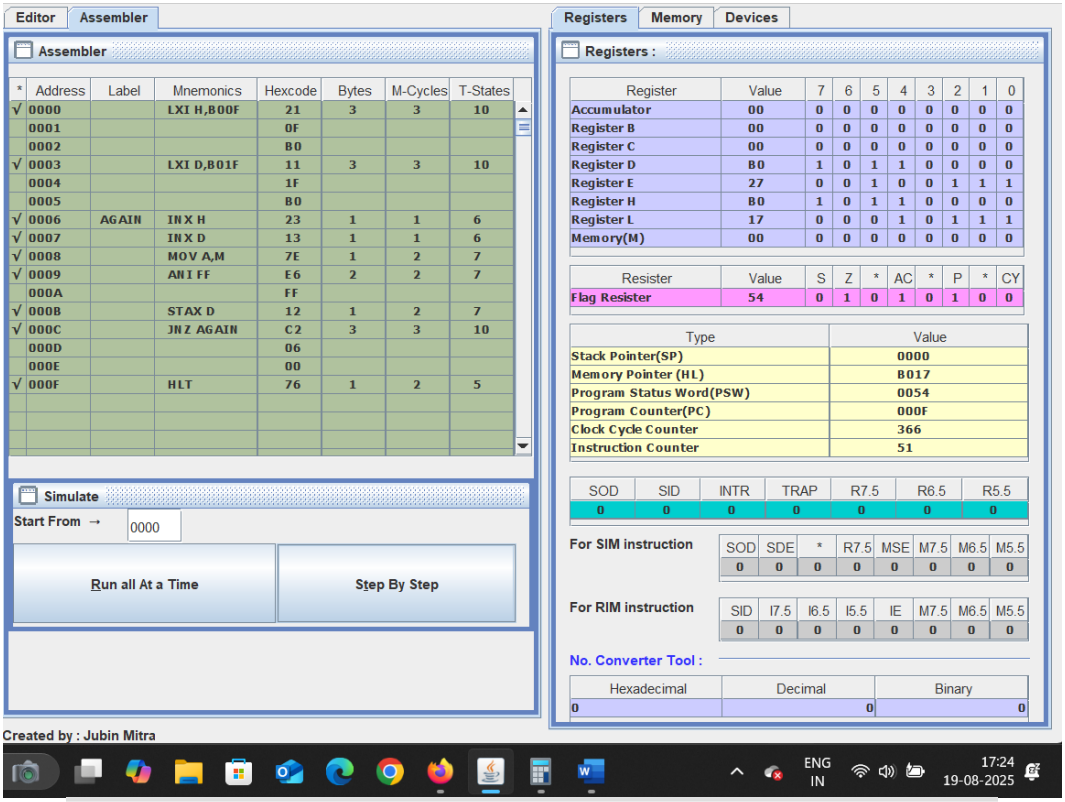


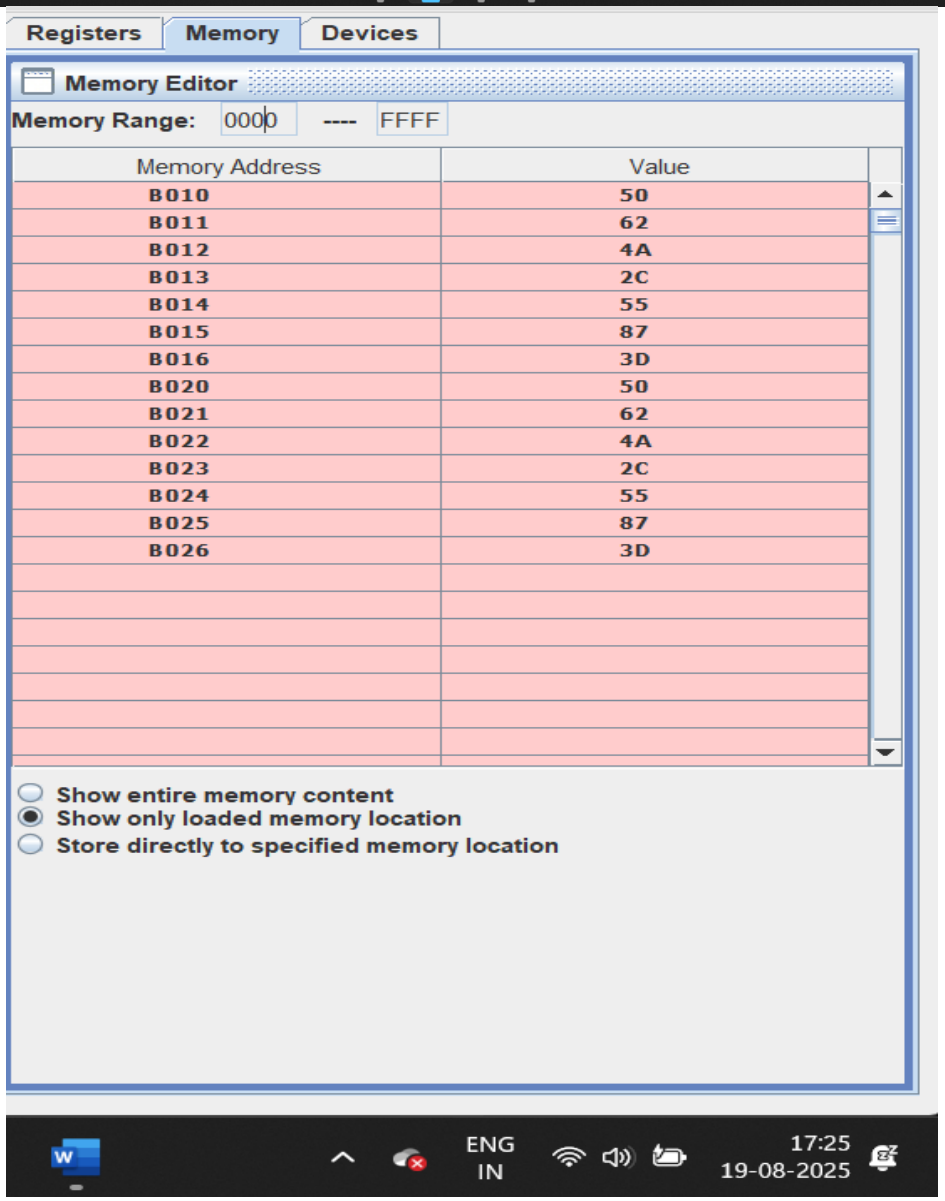
3. Write a program to transfer a block of data from B010H to B020H. The end of block is indicated by 00H.

Assembler | Disassembler

```
// Write a program to transfer a block of data from B010H to B020H The
//end of block is indicated by 00H
# ORG B010H
# DB 50H,62H,4AH,2CH,55H,87H,3DH
# ORG 0000H
            LXI H,B00F          // STORE B00F ADDRESS INTO HL
PAIR
            LXI D,B01F          // STORE B01F ADDRESS INTO DE
PAIR

AGAIN:      INX H       // INCREMENT HL PAIR
            INX D       // INCREMENT DE PAIR
            MOV A,M     // MOVE MEMORY'S DATA INTO ACCUMULATOR
            ANI FF      // PERFORM AND OPERATION WITH
ACCUMULATOR'S DATA AND UPDATE FLAGS
            STAX D      // STORE ACCUMULATOR'S DATA INTO
MEMORY ADDRESS WHICH IS IN DE PAIR
            JNZ AGAIN           // JUMP IF ZERO FLAG IS NOT SET
TO 1
            HLT         // STOP
```


Editor
Assembler
Assembler
* | Address | Label | Mnemonics | Hexcode | Bytes | M-Cycles | T-States
√ | 0000 | | LXI H,B00F | 21 | 3 | 3 | 10
| 0001 | | | 0F
| 0002 | | | B0
√ | 0003 | | LXI D,B01F | 11 | 3 | 3 | 10
| 0004 | | | 1F
| 0005 | | | B0
√ | 0006 | AGAIN | INX H | 23 | 1 | 1 | 6
√ | 0007 | | INX D | 13 | 1 | 1 | 6
√ | 0008 | | MOV A,M | 7E | 1 | 2 | 7
√ | 0009 | | ANI FF | E6 | 2 | 2 | 7
| 000A | | | FF
√ | 000B | | STAX D | 12 | 1 | 2 | 7
√ | 000C | | JNZ AGAIN | C2 | 3 | 3 | 10
| 000D | | | 06
| 000E | | | 00
√ | 000F | | HLT | 76 | 1 | 2 | 5
Simulate
Start From →
0000
Run all At a Time
Step By Step
Registers
Memory
Devices
Registers :
Register | Value | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0
Accumulator | 00 | 0 | 0 | 0 | 0 | 0 | 0 | 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 | B0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0
Register E | 27 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 1
Register H | B0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0
Register L | 17 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1
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) | B017
Program Status Word(PSW) | 0054
Program Counter(PC) | 000F
Clock Cycle Counter | 366
Instruction Counter | 51
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
ENG
IN
17:24
19-08-2025


Registers
Memory
Devices
Memory Editor
Memory Range: 0000 ---- FFFF
Memory Address | Value
B010 | 50
B011 | 62
B012 | 4A
B013 | 2C
B014 | 55
B015 | 87
B016 | 3D
B020 | 50
B021 | 62
B022 | 4A
B023 | 2C
B024 | 55
B025 | 87
B026 | 3D
Show entire memory content
Show only loaded memory location
Store directly to specified memory location
ENG
IN
17:25
19-08-2025

❖ Exercise : (Answer the following questions)

Rahul Prajapati 2316217020 19-Aug-2025

Practical - 4

☆ Exercise

1] The register always pointing to the location of next instruction to be fetched is...

⇒ Program Counter (PC)

2] If starting address of 8kb RAM is 3000H. then ending address is...

⇒ 8KB = 8 x 1024 B

$2^3 \times 2^{10} = 3+10 = 13$ Address lines.

then.

0000 0000 0000 0000 ⇒ 0000H

0001 1111 1111 1111 ⇒ 1FFFH

⇒ Size of RAM + starting Address

⇒ 1FFFH + 3000H ⇒ 4FFFH.

→ Ending Address is 4FFFH

3] Define Bus and give its type.

⇒ A Bus is a communication pathway used to transfer data, address, and control signal between components of a computer (CPU, memory, io devices.

Date
Page

19-Aug-2025

* Types of Bus: (i) Data Bus
(ii) Address Bus
(iii) Control Bus.

4. A memory chip in a microcontroller system has eight address lines. How many bytes can be stored in it?

Suppose we have data bits range is following:

D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7	
0	0	0	0	0	0	0	0	00H
1	1	1	1	1	1	1	1	FFH
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0	

$2^7+2^6+2^5+2^4+2^3+2^2+2^1+2^0 = 256$ byte.

∴ therefor 256 bytes can be stored in it.

5. If the starting address of chip is 9000H, what will be the ending address?

⇒ Let Assume we have chip sizes are like 4kb, 8kb 2kb. then.

if chip size 2kb ⇒ Ending = 9000H + 0FFFH = 9FFFH
if chip size 4kb ⇒ Ending = 9000H + 1FFFH = AFFFH
if chip size 8kb ⇒ Ending = 9000H + 3FFFH = CFFFH

Date 14-Aug
Page

6. How many address lines are necessary to access 16 Kbytes EPROM & 8Kb RAM?

=>

for 16 kb EPROM the Address lines required is

$2^4 \times 2^{10} = 2^{14}$ therefor 14 Address line necessary.

for 8 kb RAM the Address lines neccessary is

$2^3 \times 2^{10} = 2^{13}$ therefor 13 Address line neccessa