

EXP NO: 1

AIM:

To write an assembly language program to implement 8-bit addition using 8085 processor.

ALGORITHM:

- 1) Start the program by loading the first data into the accumulator.
- 2) Move the data to a register.
- 3) Get the second data and load it into the accumulator.
- 4) Add the two register contents.
- 5) Check for carry.
- 6) Store the value of sum and carry in the memory location.
- 7) Halt.

PROGRAM:

LDA 8500

MOV B, A

LDA 8501

ADD B

STA 8502

RST 1

INPUT:

The screenshot displays an 8085 assembly simulator interface. On the left, the 'Load me at' section shows the following assembly code:

```
1 LDA 8700
2 MOV B, A
3 LDA 8701
4 ADD B
5 STA 8702
6 RST 1
```

On the right, the 'Memory' tab is active, showing a memory dump starting at address 8700. The dump is as follows:

Address (Hex)	Address	Data
21FC	8700	10
21FD	8701	0
21FE	8702	0
21FF	8703	0
2200	8704	0
2201	8705	0
2202	8706	0
2203	8707	0
2204	8708	0
2205	8709	0
2206	8710	0
2207	8711	0

At the bottom, the 'Line No Assembler Message' section shows:

```
0 Program assembled successfully
```

OUTPUT:

Load me at		Data	Stack	KeyPad
1 LDA 8600		Start	8600	
2 MOV B, A		Address (Hex)	Address	Data
3 LDA 8601		2198	8600	10
4 ADD B		2199	8601	0
5 STA 8602		219A	8602	10
6 RST 1		219B	8603	0
		219C	8604	0
		219D	8605	0
		219E	8606	0
		219F	8607	0
		21A0	8608	0
		21A1	8609	0
		21A2	8610	0

RESULT: Thus the program was executed successfully using 8085 processor simulator.

EXP NO: 2

AIM: To write an assembly language program to implement 8-bit subtraction using 8085 processor.

ALGORITHM:

- 1) Start the program by loading the first data into the accumulator.
- 2) Move the data to a register.
- 3) Get the second data and load it into the accumulator.
- 4) Subtract the two register contents.
- 5) Check for borrow.
- 6) Store the difference and borrow in the memory location.
- 7) Halt.

PROGRAM:

```
LDA 8000
MOV B, A
LDA 8001
SUB B
STA 8002
RST 1
```

INPUT:

The screenshot shows the 8085 processor simulator interface. On the left, the assembly code is displayed in a window titled 'Load me at'. The code is as follows:

```
1  
2 LDA 8900  
3 MOV B, A  
4 LDA 8901  
5 SUB B  
6 STA 8902  
7 RST 1  
8
```

On the right, the memory dump window is open, showing a table of memory addresses and their corresponding data values. The 'Start' address is set to 8900.

Address (Hex)	Address	Data
22C4	8900	30
22C5	8901	20
22C6	8902	0
22C7	8903	0
22C8	8904	0
22C9	8905	0
22CA	8906	0
22CB	8907	0
22CC	8908	0
22CD	8909	0
22CE	8910	0
22CF	8911	0

OUTPUT:

The screenshot shows the 8085 processor simulator interface after execution. The assembly code window on the left is the same as in the input section. The memory dump window on the right shows the state of memory after execution. The 'Start' address is still 8900.

Address (Hex)	Address	Data
22C4	8900	20
22C5	8901	30
22C6	8902	10
22C7	8903	0
22C8	8904	0
22C9	8905	0
22CA	8906	0
22CB	8907	0
22CC	8908	0
22CD	8909	0
22CE	8910	0
22CF	8911	0

Below the memory dump, the 'Assembler Message' window shows the following message:

```
Line No Assembler Message  
0 Program assembled successfully
```

RESULT: Thus the program was executed successfully using 8085 processor simulator

EXP NO: 3

AIM: To write an assembly language program to implement 8-bit multiplication using 8085 processor.

ALGORITHM:

- 1) Start the program by loading a register pair with the address of memory location.
- 2) Move the data to a register.
- 3) Get the second data and load it into the accumulator.
- 4) Add the two register contents.
- 5) Increment the value of the carry.
- 6) Check whether the repeated addition is over.
- 7) Store the value of product and the carry in the memory location.
- 8) Halt.

Program:

```
LXI H,100  
MOV B,M  
MVI A,00  
MOV C,A  
INX H  
CONT: ADD M  
JNC SKIP  
INR C  
SKIP: DCR B  
JNZ CONT  
STA 102  
MOV A,C  
STA 103  
HLT
```

INPUT:

The screenshot shows an 8085 assembler interface. On the left, a text editor displays assembly code with line numbers 1 through 15. The code includes instructions like LDA, MOV, MVI, CPI, JZ, XRA, LOOP, DCR, JMP, STA, and RST. On the right, a 'Memory' window is open, showing a table of memory addresses (hex) and their corresponding data. The 'Start' address is set to 8400. The memory dump shows addresses from 20D0 to 20DB, with data values ranging from 10 to 0. Below the memory dump, a 'Line No Assembler Message' section shows a successful assembly message.

Address (Hex)	Address	Data
20D0	8400	10
20D1	8401	10
20D2	8402	0
20D3	8403	0
20D4	8404	0
20D5	8405	0
20D6	8406	0
20D7	8407	0
20D8	8408	0
20D9	8409	0
20DA	8410	0
20DB	8411	0

Line No Assembler Message
0 Program assembled successfully

OUTPUT:

The screenshot shows an 8085 assembler interface. On the left, a text editor displays assembly code with line numbers 1 through 15. The code includes instructions like LXI, MOV, MVI, MOV, INX, CONT, JNC, INR, SKIP, JNZ, STA, MOV, STA, and HLT. On the right, a 'Memory' window is open, showing a table of memory addresses (hex) and their corresponding data. The 'Start' address is set to 100. The memory dump shows addresses from 0064 to 006F, with data values ranging from 5 to 0. Below the memory dump, a 'Line No Assembler Message' section shows a successful assembly message.

Address (Hex)	Address	Data
0064	100	5
0065	101	5
0066	102	25
0067	103	0
0068	104	0
0069	105	0
006A	106	0
006B	107	0
006C	108	0
006D	109	0
006E	110	0
006F	111	0

Line No Assembler Message
0 Program assembled successfully

RESULT: Thus the program was executed successfully using 8085 processor simulator.

EXP NO: 4

AIM: To write an assembly language program to implement 8-bit division using 8085 processor.

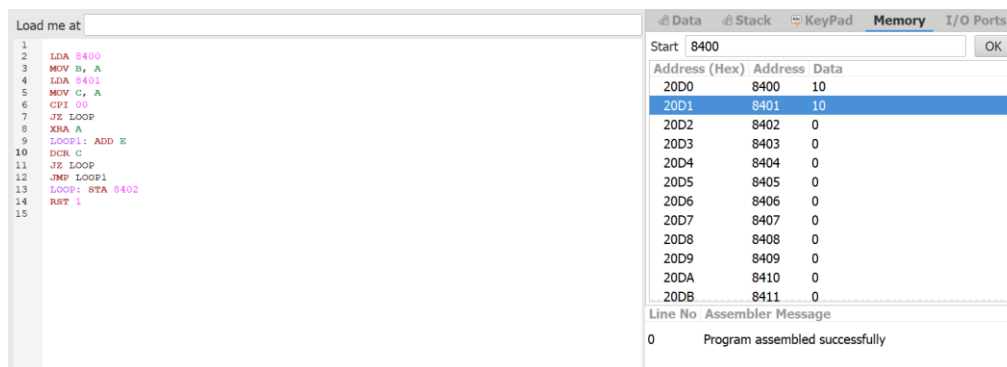
ALGORITHM:

- 1) Start the program by loading a register pair with the address of memory location.
- 2) Move the data to a register.
- 3) Get the second data and load it into the accumulator.
- 4) Subtract the two register contents.
- 5) Increment the value of the carry.
- 6) Check whether the repeated subtraction is over.
- 7) Store the value of quotient and the remainder in the memory location.
- 8) Halt.

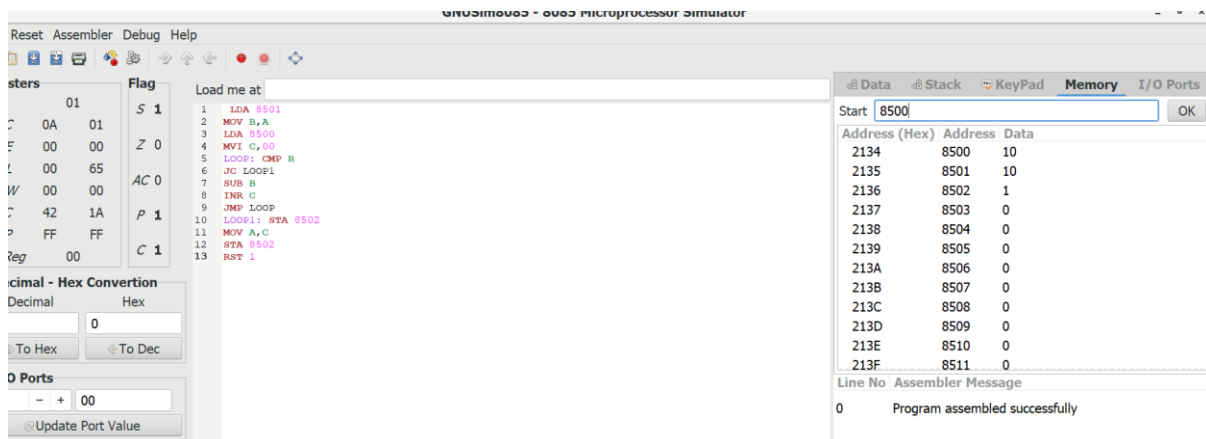
PROGRAM:

```
LDA 8501
MOV B,A
LDA 8500
MVI C,00
LOOP: CMP B
JC LOOP1
SUB B
INR C
JMP LOOP
LOOP1: STA 8502
MOV A,C
STA 8502
RST 1
```

INPUT:



OUTPUT:



RESULT: Thus the program was executed successfully using 8085 processor simulator.

EXP NO: 5

AIM: To write an assembly language program to implement 16-bit addition using 8085 processor.

ALGORITHM:

- 1) Start the program by loading a register pair with address of 1st number.
- 2) Copy the data to another register pair.
- 3) Load the second number to the first register pair.
- 4) Add the two register pair contents.
- 5) Check for carry.
- 6) Store the value of sum and carry in memory locations.
- 7) Terminate the program.

PROGRAM:

LDA 3050

MOV B,A

LDA 3051

ADD B

STA 3052

LDA 3053

MOV B,A

LDA 3054

ADC B

STA 3055

HLT

INPUT:

The screenshot shows the 8085 processor simulator interface. On the left, the assembly code is loaded at address 3050. The code consists of 11 lines: LDA 3050, MOV B,A, LDA 3051, ADD B, STA 3052, LDA 3053, MOV B,A, LDA 3054, ADC B, STA 3055, and HLT. On the right, the Memory window is open, displaying a table of memory addresses and their contents. The table shows addresses from 0BEA to 0BF5. The data at 0BEA is 34, at 0BEB is 12, at 0BEC is 0, at 0BED is 78, at 0BEE is 56, at 0BEF is 0, at 0BF0 is 0, at 0BF1 is 0, at 0BF2 is 0, at 0BF3 is 0, at 0BF4 is 0, and at 0BF5 is 0. Below the memory dump, the assembler message window shows the message "Program assembled successfully".

Address (Hex)	Address	Data
0BEA	3050	34
0BEB	3051	12
0BEC	3052	0
0BED	3053	78
0BEE	3054	56
0BEF	3055	0
0BF0	3056	0
0BF1	3057	0
0BF2	3058	0
0BF3	3059	0
0BF4	3060	0
0BF5	3061	0

Line No Assembler Message
0 Program assembled successfully

OUTPUT:

The screenshot shows the 8085 processor simulator interface. On the left, the assembly code is loaded at address 3050. The code consists of 11 lines: LDA 3050, MOV B,A, LDA 3051, ADD B, STA 3052, LDA 3053, MOV B,A, LDA 3054, ADC B, STA 3055, and HLT. On the right, the Memory window is open, displaying a table of memory addresses and their contents. The table shows addresses from 0BEA to 0BF5. The data at 0BEA is 34, at 0BEB is 12, at 0BEC is 46, at 0BED is 78, at 0BEE is 56, at 0BEF is 134, at 0BF0 is 0, at 0BF1 is 0, at 0BF2 is 0, at 0BF3 is 0, at 0BF4 is 0, and at 0BF5 is 0. Below the memory dump, the assembler message window shows the message "Program assembled successfully".

Address (Hex)	Address	Data
0BEA	3050	34
0BEB	3051	12
0BEC	3052	46
0BED	3053	78
0BEE	3054	56
0BEF	3055	134
0BF0	3056	0
0BF1	3057	0
0BF2	3058	0
0BF3	3059	0
0BF4	3060	0
0BF5	3061	0

Line No Assembler Message
0 Program assembled successfully

RESULT: Thus the program was executed successfully using 8085 processor simulator.

EXP NO: 6

AIM: To write an assembly language program to implement 16-bit subtraction using 8085 processor.

ALGORITHM:

- 1) Start the program by loading a register pair with address of 1st number.
- 2) Copy the data to another register pair.
- 3) Load the second number to first register pair.
- 4) Subtract the two register pair contents.
- 5) Check for borrow.
- 6) Store the value of difference and borrow in memory locations.
- 7) End.

PROGRAM:

LHLD 2050

XCHG

LHLD 2052

MVI C,00

MOV A,E

SUB L

STA 2054

MOV A,D

SUB H

STA 2055

HLT

INPUT:

The screenshot displays an 8085 assembly simulator interface. On the left, a text area shows the assembly code with line numbers 1 through 11. The code is as follows:

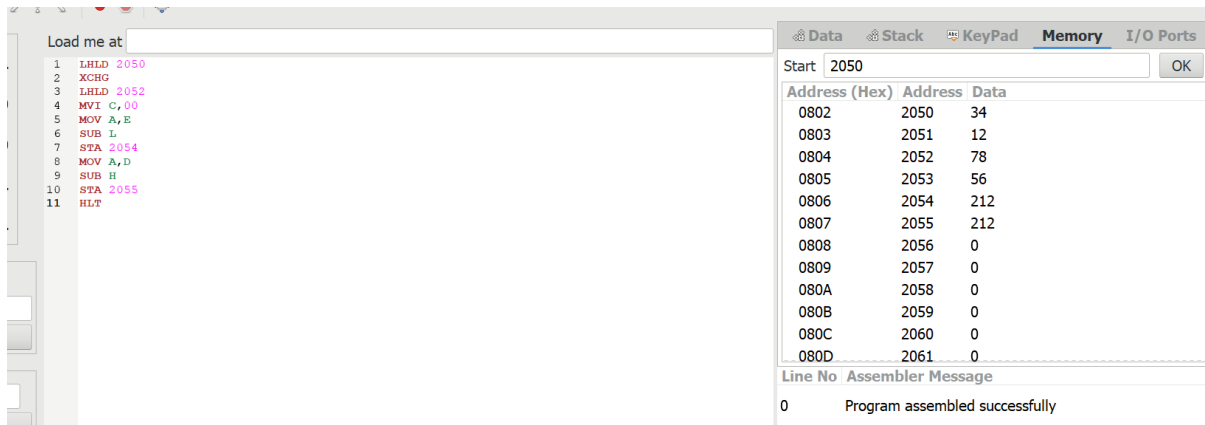
```
1 LHLD 2050
2 XCHG
3 LHLD 2052
4 MVI C,00
5 MOV A,E
6 SUB L
7 STA 2054
8 MOV A,D
9 SUB H
10 STA 2055
11 HLT
```

On the right, the 'Memory' tab is active, showing a memory dump starting at address 2050. The dump is as follows:

Address (Hex)	Address	Data
0802	2050	34
0803	2051	12
0804	2052	78
0805	2053	56
0806	2054	0
0807	2055	0
0808	2056	0
0809	2057	0
080A	2058	0
080B	2059	0
080C	2060	0
080D	2061	0

At the bottom, the 'Assembler Message' window shows the message: 'Program assembled successfully'.

OUTPUT:



RESULT: Thus the program was executed successfully using 8085 processor simulator.

EXP NO: 7

AIM: To write an assembly language program to implement 16-bit multiplication using 8085 processor.

ALGORITHM:

- 1) Load the first data in HL pair.
- 2) Move content of HL pair to stack pointer.
- 3) Load the second data in HL pair and move it to DE.
- 4) Make H register as 00H and L register as 00H.
- 5) ADD HL pair and stack pointer.
- 6) Check for carry if carry increment it by 1 else move to next step.
- 7) Then move E to A and perform OR operation with accumulator and register D.
- 8) The value of operation is zero, then store the value else go to step 3.

PROGRAM:

```
LHLD 2050
SPHL
LHLD 2052
XCHG
LXI H,0000H
LXI B,0000H
AGAIN: DAD SP
```

```

JNC START
INX B
START: DCX D
MOV A,E
ORA D
JNZ AGAIN
SHLD 2054
MOV L,C
MOV H,B
SHLD 2055
HLT

```

INPUT:

The screenshot shows the 8085 processor simulator interface. On the left, the assembly code is listed with line numbers 1 through 11. The code includes instructions like LHL, XCHG, MVI, MOV, SUB, STA, and HLT. On the right, the 'Memory' tab is selected, displaying a memory dump starting at address 2050. The dump shows addresses 0802 through 080D with their corresponding data values. Below the memory dump, the 'Line No Assembler Message' section shows '0 Program assembled successfully'.

Address (Hex)	Address	Data
0802	2050	34
0803	2051	12
0804	2052	78
0805	2053	56
0806	2054	0
0807	2055	0
0808	2056	0
0809	2057	0
080A	2058	0
080B	2059	0
080C	2060	0
080D	2061	0

OUTPUT:

The screenshot shows the 8085 processor simulator interface after execution. The assembly code is listed with line numbers 1 through 18. The code includes instructions like LHL, SPHL, XCHG, LXI, AGAIN, JNC, INX, START, MOV, ORA, JNZ, SHLD, and HLT. On the right, the 'Memory' tab is selected, displaying a memory dump starting at address 2050. The dump shows addresses 0802 through 080D with their corresponding data values. Below the memory dump, the 'Line No Assembler Message' section shows '0 Program assembled successfully'.

Address (Hex)	Address	Data
0802	2050	52
0803	2051	18
0804	2052	120
0805	2053	86
0806	2054	96
0807	2055	38
0808	2056	6
0809	2057	0
080A	2058	0
080B	2059	0
080C	2060	0
080D	2061	0

RESULT: Thus the program was executed successfully using 8085 processor simulator.

EXP NO: 8

AIM: To write an assembly language program to implement 16-bit divided by 8-bit using 8085 processor.

ALGORITHM:

- 1) Read dividend (16 bit)
- 2) Read divisor
- 3) count <- 8
- 4) Left shift dividend
- 5) Subtract divisor from upper 8-bits of dividend
- 6) If CS = 1 go to 9
- 7) Restore dividend
- 8) Increment lower 8-bits of dividend
- 9) count <- count - 1
- 10) If count = 0 go to 5
- 11) Store upper 8-bit dividend as remainder and lower 8-bit as quotient
- 12) Stop

PROGRAM:

```
LDA 8501
MOV B,A
LDA 8500
MVI C,00
LOOP: CMP B
JC LOOP1
SUB B
INR C
JMP LOOP
STA 8503
DCR C
MOV A,C
LOOP1: STA 8502
RST 1
```

INPUT:

The screenshot shows the 8085 processor simulator interface. On the left, the assembly code is displayed with line numbers 1 through 4. The code is as follows:

```
1 LDA 8501
2 MOV B,A
3 LDA 8500
4 MVI C,00
5 LOOP: CMP B
6 JC LOOP1
7 SUB B
8 INR C
9 JMP LOOP
0 STA 8503
1 DCR C
2 MOV A,C
3 LOOP1: STA 8502
4 RST 1
```

On the right, the Memory window is open, showing a table of memory addresses and their contents. The table is as follows:

Address (Hex)	Address	Data
2134	8500	5
2135	8501	10
2136	8502	0
2137	8503	0
2138	8504	0
2139	8505	0
213A	8506	0
213B	8507	0
213C	8508	0
213D	8509	0
213E	8510	0
213F	8511	0

Below the memory table, the Line No Assembler Message window shows the message: "Program assembled successfully".

OUTPUT:

The screenshot shows the 8085 processor simulator interface after execution. On the left, the assembly code is displayed with line numbers 1 through 14. The code is as follows:

```
1 LDA 8501
2 MOV B,A
3 LDA 8500
4 MVI C,00
5 LOOP: CMP B
6 JC LOOP1
7 SUB B
8 INR C
9 JMP LOOP
10 STA 8503
11 DCR C
12 MOV A,C
13 LOOP1: STA 8502
14 RST 1
```

On the right, the Memory window is open, showing a table of memory addresses and their contents. The table is as follows:

Address (Hex)	Address	Data
2134	8500	5
2135	8501	10
2136	8502	5
2137	8503	0
2138	8504	0
2139	8505	0
213A	8506	0
213B	8507	0
213C	8508	0
213D	8509	0
213E	8510	0
213F	8511	0

Below the memory table, the Line No Assembler Message window shows the message: "Program assembled successfully".

RESULT: Thus the program was executed successfully using 8085 processor simulator.

EXP NO: 9

AIM: To find the factorial of a given number using 8085 microprocessor.

ALGORITHM:

- 1) Load the data into register B
- 2) To start multiplication set D to 01H
- 3) Jump to step 7
- 4) Decrements B to multiply previous number
- 5) Jump to step 3 till value of B>0
- 6) Take memory pointer to next location and store result
- 7) Load E with contents of B and clear accumulator
- 8) Repeatedly add contents of D to accumulator E times
- 9) Store accumulator content to D

10) Go to step 4

PROGRAM:

```
LDA 2001
MOV B,A
MVI C,#01
MVI E,#01
LOOP: MOV D,C
MVI A,00H
LP: ADD E
DCR D
JNZ LP
MOV E,A
INR C
DCR B
JNZ LOOP
MOV A,E
STA 2010
HLT
```

INPUT:

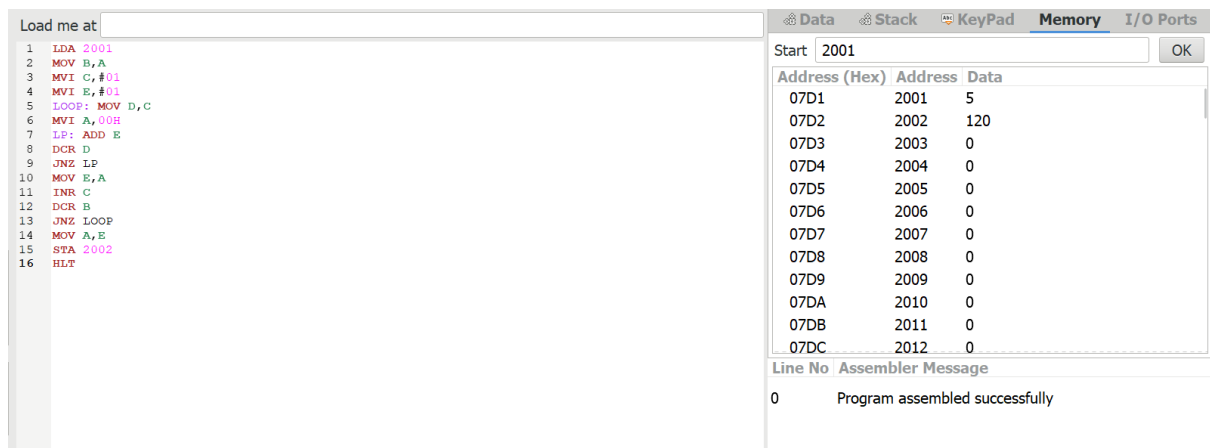
The screenshot shows an 8085 assembly simulator interface. On the left, the assembly code is listed with line numbers 1 through 16. The code includes instructions like LDA, MOV, MVI, LOOP, LP, DCR, JNZ, and STA. On the right, the 'Memory' tab is active, displaying a memory dump starting at address 2001. The dump shows addresses from 07D1 to 07DC with corresponding data values. At the bottom, a message box indicates 'Program assembled successfully'.

Address (Hex)	Address	Data
07D1	2001	5
07D2	2002	0
07D3	2003	0
07D4	2004	0
07D5	2005	0
07D6	2006	0
07D7	2007	0
07D8	2008	0
07D9	2009	0
07DA	2010	0
07DB	2011	0
07DC	2012	0

Line No Assembler Message

0 Program assembled successfully

OUTPUT:



RESULT: Thus the program was executed successfully using 8085 processor simulator.

EXP NO: 10

AIM: To find the largest number from an array using 8085 processor.

ALGORITHM:

- 1) Load the address of the first element of the array in HL pair.
- 2) Move the count to B register.
- 3) Increment the pointer.
- 4) Get the first data in A register.
- 5) Decrement the count.
- 6) Increment the pointer.
- 7) Compare the content of memory addressed by HL pair with that of A register.
- 8) If carry=0, go to step 10 or if carry=1 go to step 9
- 9) Move the content of memory addressed by HL to A register.
- 10) Decrement the count.

PROGRAM:

LXI H,2050

MOV C,M

DCR C

INX H

MOV A,M

LOOP1: INX H

```

CMP M
JNC LOOP
MOV A,M
LOOP: DCR C
JNZ LOOP1
STA 2058
HLT

```

INPUT:

The screenshot shows the 8086 processor simulator interface. The assembly code is loaded at address 2050. The memory dump shows the following data:

Address (Hex)	Address	Data
0802	2050	52
0803	2051	18
0804	2052	120
0805	2053	86
0806	2054	96
0807	2055	38
0808	2056	6
0809	2057	0
080A	2058	0
080B	2059	0
080C	2060	0
080D	2061	0

The assembler message shows: 0 Program assembled successfully

OUTPUT:

The screenshot shows the 8086 processor simulator interface. The assembly code is loaded at address 2050. The memory dump shows the following data:

Address (Hex)	Address	Data
0802	2050	52
0803	2051	18
0804	2052	120
0805	2053	86
0806	2054	96
0807	2055	38
0808	2056	6
0809	2057	0
080A	2058	120
080B	2059	0
080C	2060	0
080D	2061	0

The assembler message shows: 0 Program assembled successfully

RESULT: Thus the program was executed successfully using 8086 processor simulator.