COURSE CODE: CSA1261
COURSE NAME: Computer Architecture for Data Processing

NAME: k.pavani
REGISTER NUMBER:192425288

8-BIT ADDITION

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 8050
MOV B, A
LDA 8051
ADD B
STA 8052
HLT
```

INPUT:

ADDRESS	DATA
8050	1
8051	2

OUTPUT:

ADDRESS	DATA
8052	3

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

8-BIT SUBTRACTION

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
HLT
```

INPUT:

ADDRESS	DATA
8000	4
8001	5

OUTPUT:

ADDRESS	DATA
8002	1

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

8-BIT MULTIPLICATION

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:

```
LDA 2200
MOV E, A
MVI D, 00
LDA 2201
MOV C, A
LXI H, 0000

BACK: DAD D
DCR C
JNZ BACK

SHLD 2202
HLT
```

INPUT:

ADDRESS	DATA
2200	4
2201	2

OUTPUT:

ADDRESS	DATA
2202	8

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

8-BIT DIVISION

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 reminder in the memory location.

8) Halt.

PROGRAM:

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

INPUT:

ADDRESS	DATA
8500	2
8501	6

OUTPUT:

ADDRESS	DATA
8502	0
8503	3

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

16-BIT ADDITION

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:-

Address	Data
3050	2
3051	3
3053	5
3054	5

OUTPUT:-

Address	Data
3052	5
3055	10

RESULT:-

Thus the program was executed successfully using 8085 processor simulator.

16-BIT SUBTRACTION

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 the first register pair.

4) sub the two register pair contents.

5) Check for carry.

6) Store the value of sum and carry in memory locations.

7) End.

PROGRAM:-

```
LHLD 2050
XCHG
LHLD 2052
MOV A, L
SUB E
STA 2054
MOV A, H
SBB D
STA 2055
HLT
```

INPUT:-

Address	Data
2050	2
2052	3

OUTPUT:-

Address	Data
2054	1
2055	1

RESULT:-

Thus the program was executed successfully using 8085 processor simulator.

16-BIT MULTIPLICATION

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

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:-

Address	Data
2050	10
2052	5

OUTPUT:-

Address	Data
2054	50
2055	5

RESULT:-

Thus the program was executed successfully using 8085 processor simulator.

16-BIT DIVISION

EXP NO: 8

AIM:-

To write an assembly language program to implement 16-bit division 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 8500
MOV B,A
LDA 8501
MVI C,00
LOOP: CMP B
JC LOOP1
SUB B
INR C
JMP LOOP
LOOP1: STA 8502
MOV A,C
STA 8503
HLT
```

INPUT:-

Address	Data
8051	20
8050	2

OUTPUT:-

Address	Data
8502	10
8503	2

RESULT:-

Thus the program was executed successfully using 8085 processor simulator

16-BIT ADDITION

EXP NO: 9

AIM :- To write an assembly language program to implement 16-Bit addition using 8086 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.

4-Add the two register pair contents.

5-Check for carry.

6-Store the value of sum and carry in memory location. Result stored in AX. 7-

Terminate the program.

PROGRAM :

```
MOV AX,[1100H]
MOV BX,[1102H]
ADD AX,BX
MOV [1200H], AX
HLT
```

INPUT :-

REGISTER	MEMORY	DATA
AX	32	1100
BX	45	1102

OUTPUT :-

REGISTER	MEMORY	DATA
AX	77	1200

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

16 BIT SUBTRACTION

EXP NO: 10

AIM:

To write an assembly language program to implement 16 bit subtraction using 8086 processor.

ALGORITHM:

1] Start the program by loading a register pair with address of first number.

2] Copy the data to another register pair.

3] Laod the second number to first register pair.

4] Subtract the two register pair contacts.

5] Check for borrow.

6] Store the value of difference and borrow in memory location.

7] End.

PROGRAM:

MOV AX,[1100H]

MOV BX,[1102H]

SUB AX,BX

MOV [1200H], AX

HLT

INPUT:

ADDRESS	DATA
1100	30
1102	15

OUTPUT:

ADDRESS	DATA
1200	15

RESULT:

Thus the program was executed successfully using 8086 processor simulator.

16-bit multiplication

EXP NO: 11

Aim: To write an assembly language program to implement 16-bit multiplication on 8086 processer.

ALGORITHM:

1. Load the first data in HL pair
2. Move content of HL pair to stack pointer
3. Load the second data in the HL pair and move it to DE
4. Make H register as OH and L register OH
5. Add HL pair and stack pointer
6. Check for carry if carry increment by 1 else move to next step
7. Then move E to A and perform or operation with accumulation and register D
8. The value of operation is zero the solve the value else go to step 3

PROGRAM:

MOV AX, [1100H]

MOV BX, [1102H]

MUL BX

MOV [1200H], AX

MOV [1202H], DX

HLT

INPUT:

ADDRESS	**DATA**
1100	20
1202	3

OUT PUT:

ADDRESS	**DATA**
1200	60
1202	3

RESULT:

Thus the program was executed successfully using 8086 emulator.

16 BIT DIVISION

EXP NO: 12

AIM:

To write an assemble language program to implement 16 bit divided using 8086 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:

MOV AX, [1100H]

MOV BX, [1102H]

DIV BX

MOV [1200H], AX

MOV [1202H], DX

HLT

INPUT:

ADDRESS	**DATA**
1100	10

1102	10

OUTPUT:

ADDRESS	**DATA**
1200	1
1202	10

RESULT:

Thus the program was executed successfully using 8086 processor simulator.

Greatest of 2 numbers

EXP NO: 13

Exp. No :-

AIM:-

To write an Assembly Language Program to find the smallest number in an array using 8085 Microprocessor in GNUSim.

SOFTWARE USED:-

GNUSim8085

ALGORITHM:-

1. Initialize the count
2. Get the input numbers
3. Compare the content of Accumulator(A) with HL pair for all input numbers
4. Stores the smallest number in the output register
5. End the program

PROGRAM:-

LDA 2050

MOV B, A

LDA 2051

CMP B

JNC STORE

MOV A, B

STORE: STA 2052

HLT

Input

Address	Data
2050	29
2051	22

Output

Address	Data
2052	29

RESULT:

Thus the Assembly Language Program to find the smallest number in an array using 8085 Microprocessor in GNUSim is performed.

Smallest of 2 numbers

EXP NO: 14

AIM:-

To write an Assembly Language Program to find the smallest number in an array using 8085 Microprocessor in GNUSim.

SOFTWARE USED:-

GNUSim8085

ALGORITHM:-

1. Initialize the count
2. Get the input numbers
3. Compare the content of Accumulator(A) with HL pair for all input numbers
4. Stores the smallest number in the output register
5. End the program

PROGRAM:-

```
LXI H,8050
MOV C, M
INX H
MOV B, M
DCR C
LOOP: INX H
MOV A, M
CMP B
JNC SKIP
MOV B, A
SKIP: DCR C
JNZ LOOP
LXI H,8500
MOV M, B
HLT
```

Input

Address	Data
8000	10

Output

Address	Data
8010	3

RESULT:

Thus the Assembly Language Program to find the smallest number in an array using 8085 Microprocessor in GNUSim is performed.

SWAPING OF TWO 8-BIT DATA

EXP NO: 15

AIM:

To Write an assembly language program to swap two 8-bit data using 8085 processor.

ALGORITHM:

1. Load the contents of memory address 1100 into accumulator A.
2. Move the contents of accumulator A into register B.
3. Load the contents of memory address 1101 into accumulator A.
4. Move the contents of accumulator A into register C.
5. Store the contents of accumulator A (which is the original value at 1101) into memory address 1102.
6. Move the contents of register B (which is the original value at 1100) into accumulator A.
7. Store the contents of accumulator A into memory address 1103.

PROGRAM:

```
LDA 1100
MOV B, A
LDA 1101
MOV C, A
STA 1102
MOV A, B
STA 1103
HLT
```

INPUT:

ADDRESS	DATA
1100	6
1101	4

OUTPUT:

ADDRESS	DATA
1103	4
1104	6

RESULT:

Thus the program was executed successfully using 8085 processor simulator.

1’s COMPLIMENT

EXP NO: 16

AIM :

To write assembly language to find 1’s COMPLIMENT by using 8085 microprocessor Simulator

ALGORITHM:

1. Loads the value from memory address 8000 into accumulator A.

2. Complements the bits of the value in accumulator A using the CMA (Complement Accumulator) instruction. This means that all 1s become 0s and all 0s become 1s.

3. Stores the complemented value into memory address 8001.

4. Halts the program execution.

PROGRAM:

LDA 8000

CMA

STA 8001

HLT

INPUT:

ADDRESS	**DATA**
8000	**6**

OUTPUT:

ADDRESS	**DATA**
8001	**249**

RESULT: THIS PROGRAM WAS EXECUTED SUCCESSFULLY BY USING 8085 MICROPROCESSOR SIMULATOR

2’S COMPLEMENT

EXP NO: 17

AIM:

To write an assembly language program to find 2’s complement of 8-bit number

ALGORITHM:

1) Start with the binary number:
2) If the number is positive, simply write its binary equivalent.
3) If the number is negative, begin with the binary equivalent of its positive value.
4) Invert all the bits (1's complement):
5) Flip every 0 to 1 and every 1 to 0
6) Add 1 to the result:
7) Add 1 to the least significant bit (rightmost bit) of the inverted number.
8) The final result is the 2's complement representation of the number.

PROGRAM:

```
LDA 3000
CMA
STA 3001
ADI 3002
HLT
```

INPUT:

ADDRESS	DATA
3000	8

OUTPUT:

ADDRESS	DATA
3001	247
3002	0

RESULT: Thus the PROGRAM WAS EXECUTED SUCCESSFULLY USING 8085 PROCESSOR SIMULATER

ODD OR EVEN – 8085 MICROPROCESSOR

EXP NO: 18

AIM:

To write an assembly language program to find the number is odd or even using 8085 Microprocessor in GNUSim8085

SOFTWARE USED:-

GNUSim8085

ALGORITHM:-

1. **I**nitialize the number in the accumulator
2. Perform the AND operation with accumulator by 01
3. If the result is '0', it means it is even number (indicates as 22)
4. If the result is non zero , it means the given number is odd (indicates as 11)
5. Stores the out put in the register
6. End the program

Program :-

```
LDA 8050H
ANI 80H
JZ POS
MVI A,11
JMP STO
POS: MVI A,22
STO: STA 8051H
HLT
```

Input :

Address	Data
8050	20

Address	Data
8050	19

Output:

Address	Data
8051	22

Address	Data
8051	11

RESULT:-

Thus the assembly Language Program to find the ODD OR EVEN is performed using 2050H Microprocessor in GNUSim8085

POSITIVE AND NEGATIVE

EXP.NO : 19

AIM:

To write an assembly language program to find the number is POSITIVE AND NEGATIVE using 8085 Microprocessor in GNUSim

SOFTWARE USED:-

GNUSim 8085

ALGORITHM:-

1. **I**nitialize the number in the accumulator
2. Perform the AND operation with accumulator by 01
3. If the result is '0', it means it is even number (indicates as 22)
4. If the result is non zero , it means the given number is odd (indicates as 11)
5. Stores the out put in the register
6. End the program

Program :-

```
LDA 8050
ANI 01H
JZ LOOP1
MVI A, 11
JMP LOOP2
LOOP1: MVI A, 22
LOOP2: STA 8051
HLT
```

OUT PUT :-

Input :

Address	Data
8050	0

Address	Data
8050	15

Output:

Address	Data
8051	22

Address	Data
8051	11

RESULT:-

Thus the assembly Langua6g36e5 885250Program to find the positive or negative is performed using 2050H Microprocessor in GNUSim

ASCENDING ORDER – 8085 MICROPROCESSOR

EXP NO: 20

AIM:

To write an assembly language program to find the ascending order of numbers using 8085 Microprocessor in GNUSim8085

SOFTWARE USED:-

GNUSim8085

ALGORITHM:-

1. **I**nitialize the count
2. Get the input numbers
3. compare content accumulator [A] with HL pair for all input numbers
4. stores the ascending numbers in the output registers
5. end the program

Program :-

```
LXI H,8000
MOV C,M
DCR C
LOOP3: MOV D,C
LXI H,8001
LOOP2: MOV A,M
INX H
CMP M
JC LOOP1
MOV B,M
MOV M,A
DCX H
MOV M,B
INX H
LOOP1: DCR D
JNZ LOOP2
DCR C
JNZ LOOP3
```

HLT

Input :

Address	Data
8000	3
8001	4
8002	18

Output:

Address	Data
8001	3
8002	4
8003	18

RESULT: Thus the assembly Language Program to find the Ascending order of numbers is performed using 8085 Microprocessor in GNUSim8085

DESCENDING ORDER

EXP NO: 21

AIM:-

To write an assembly language program to implement descending order using 8085 processor.

ALGORITHM:-

1) Load the number of elements in the array (N) into a register.

2) Use nested loops:

Outer loop: Decrease the range of comparison in each iteration.

Inner loop: Compare adjacent elements and swap if needed.

3) Repeat until the array is sorted in descending order.

PROGRAM:-

```
LXI H,8050
MOV C,M
DCR C
LOOP3: MOV D,C
LXI H,8051
LOOP2: MOV A,M
INX H
CMP M
JNC LOOP1
MOV B,M
MOV M,A
DCX H
MOV M,B
INX H
LOOP1: DCR D
JNZ LOOP2
DCR C
JNZ LOOP3
HLT
```

INPUT:-

Address	Data
2001	2
2002	6
2003	3
2004	2
2005	5

OUTPUT:-

Address	Data
2010	120

RESULT:-

Thus the program was executed successfully using 8085 processor simulator.

LARGEST NUMBER IN AN ARRAY

EXP NO: 22

AIM:

To write an Assembly Language Program to find the largest number in an array using 8085 Microprocessor in GNUSim.

SOFTWARE USED:

GNUSim8085

ALGORITHM:

1. Initialize the count
2. Get the input numbers
3. Compare the content of Accumulator(A) with HL pair for all input numbers
4. Stores the largest number in the output register
5. End the program

PROGRAM:

```
LXI H,8050
MOV C, M
INX H
MOV B, M
DCR C
LOOP: INX H
MOV A, M
CMP B
JC SKIP
MOV B, A
SKIP: DCR C
JNZ LOOP
LXI H,8500
MOV M, B
HLT
```

Input

Address	Data
8050 (Counter)	5

Address	Data
8051	5
8052	2
8053	6
8054	8
8055	9

Output:

Address	Data
8500	9

RESULT:

Thus the Assembly Language Program to find the largest number in an array using 8085 Microprocessor in GNUSim is performed.

SMALLEST NUMBER IN AN ARRAY

EXP NO: 23

AIM:

To write an Assembly Language Program to find the smallest number in an array using 8085 Microprocessor in GNUSim.

SOFTWARE USED:

GNUSim8085

ALGORITHM:

1. Initialize the count
2. Get the input numbers
3. Compare the content of Accumulator(A) with HL pair for all input numbers
4. Stores the smallest number in the output register
5. End the program

PROGRAM:

```
LXI H, 2050H
MOV C, M
INX H
MOV A, M
DCR C
LOOP: INX H
CMP M
JC NEXT
MOV A, M
NEXT: DCR C
JNZ LOOP
STA 3052H
HLT
```

Address	Data
8050 (Counter)	5

Address	Data
8051	2
8052	4
8053	7
8054	5
8055	9

Output:

Address	Data
8500	2

RESULT:

Thus the Assembly Language Program to find the smallest number in an array using 8085 Microprocessor in GNUSim is performed.

LCM – 8085 MICROPROCESSOR

EXP NO: 24

AIM:

To write an assembly language program to find the LCM of numbers using 8085 Microprocessor in GNUSim8085

SOFTWARE USED:-

GNUSim8085

ALGORITHM:-

1. start the program.
2. Load A into the accumulator.
3. Move A to R1.
4. Load B into the accumulator.
5. Move B to R2.
6. Call the GCD subroutine (the GCD subroutine is already implemented using the Euclidean algorithm).
7. Compute Product:
8. Multiply A and B to get the product.
9. Store this product temporarily.
10. Divide the Product by GCD to get the LCM.
11. Store the LCM at a memory location (e.g., 6009).
12. Halt the program

PROGRAM:-

Input :

Address	Data
8000	60
8001	45

Output:

Address	Data
8011	225

RESULT: Thus the assembly Language Program to find the LCM of numbers is performed using 8085 Microprocessor in GNUSim8085

GCD

EXP NO: 25

AIM:
To write an assembly language program to implement GCD using 8085 processor.

ALGORITHM:

1) **Start the program** by loading the first number (A) into the accumulator.
2) **Move the first number (A)** to a register (R1) to store it temporarily.
3) **Get the second number (B)** and load it into the accumulator.
4) **Compare** if B is greater than 0 (i.e., check if the divisor is non-zero).
5) **Perform division** of A by B and calculate the remainder.
6) **If the remainder is 0**, the **GCD is B**. Store the result in memory (at a designated location).
7) **If the remainder is not 0**, move B to register R1 and load the remainder into the accumulator.
8) **Repeat the steps** from step 4 (looping back) until the remainder becomes 0.
9) **Store the result** (GCD) when the loop terminates and the remainder is 0.
10) **Halt the program** after completing the process.

PROGRAM:

INPUT:

ADDRESS	DATA
6000	38
6001	19

OUTPUT:

ADDRESS	DATA
6009	19

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

Factorial

EXP NO: 26

AIM :
To Write an assembly language program to find factorial of n in the given

ALGORITHM:

1. Load the address 8050H into the HL register pair.
2. Move the value from the memory location (8050H) into the B register.
3. Load the value 01H into the D register to serve as an accumulator for the factorial result.
4. Call the subroutine MUL to multiply the current value of D (partial factorial) by B.
5. Decrement the B register to move to the next value in the factorial computation.
6. Check if B is zero. If not, jump back to the label FACT.
7. Increment the HL register to point to the next memory location (8051H).
8. Store the result from the D register at the memory location pointed to by HL.
9. Halt the program.
10. Move the current value of B into the E register (as a multiplier).
11. Clear the A register (set it to 0) to use as a running total for the multiplication.
12. Perform repeated addition (ADD D) E times to compute the product.
13. Decrement the E register after each addition and check if E is zero.
14. When E becomes zero, move the result from A to D and return.

PROGRAM:

```
LXI H,8000
MOV C,M
MVI B,00
INX H
MOV B,M
CMA
MOV E,A
MVI D,00FH
MOV A,B
CMA
MOV D,A
INX D
LXI H,0000
NEXT: DAD B
SHLD 8010
LOOP: DAD D
JNC SKIP
MOV A,H
ORA L
JZ EXIT
JMP LOOP
SKIP: LHLD 8010
JMP NEXT
EXIT: LHLD 8010
HLT
```

INPUT:

ADDRESS	DATA
8050	5

OUTPUT:

ADDRESS	DATA
8051	120

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

DECIMAL TO HEXA DECIMAL

EXP NO: 27

AIM: Write a program to convert Decimal number to Hexadecimal number

SOFTWARE : GNUSIM 8085

ALGORITHM:

1. Initialize Registers:
2. Store the decimal number in a register (e.g., register B).
3. Perform repeated division of the decimal number by 16 to obtain the hexadecimal digits.
4. Store the quotient in a register (e.g., B or C).
5. Store the remainder (hex digit) separately.
6. If the remainder is greater than 9, convert it to its corresponding ASCII representation for A-F (e.g., add 7 to the remainder).
7. Store the hexadecimal digits (remainders) in reverse order in memory.
8. If the quotient is zero, the conversion is complete. Otherwise, repeat the division step with the quotient as the new dividend.
9. Use the stored hexadecimal digits to display the result.

PROGRAM:

INPUT:

ADDRESS	**DATA**
2050	**34**

OUTPUT:

ADDRESS	**DATA**
2051	**0**
2052	**84**

RESULT: Thus the PROGRAM WAS EXECUTED SUCCESSFULLY USING 8085 PROCESSOR SIMULATER