8085 MICROPROCESSOR PROGRAMS

ADDITION OF TWO 8 BIT NUMBERS

AIM:

To perform addition of two 8 bit numbers using 8085.

ALGORITHM:

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

PROGRAM:

```
        MVI     C, 00     Initialize C register to 00
        LDA     4150      Load the value to Accumulator.
        MOV     B, A      Move the content of Accumulator to B register.
        LDA     4151      Load the value to Accumulator.
        ADD     B         Add the value of register B to A
        JNC     LOOP      Jump on no carry.
        INR     C         Increment value of register C
LOOP:   STA     4152      Store the value of Accumulator (SUM).
        MOV     A, C      Move content of register C to Acc.
        STA     4153      Store the value of Accumulator (CARRY)
        HLT               Halt the program.
```

OBSERVATION:

Input:	80 (4150)
	80 (4251)
Output:	00 (4152)
	01 (4153)

RESULT:

Thus the program to add two 8-bit numbers was executed.

SUBTRACTION OF TWO 8 BIT NUMBERS

AIM:

To perform the subtraction of two 8 bit numbers using 8085.

ALGORITHM:

1. Start the program by loading the first data into Accumulator.
2. Move the data to a register (B register).
3. Get the second data and load into Accumulator.
4. Subtract the two register contents.
5. Check for carry.
6. If carry is present take 2's complement of Accumulator.
7. Store the value of borrow in memory location.
8. Store the difference value (present in Accumulator) to a memory
9. location and terminate the program.

PROGRAM:

```
        MVI     C, 00     Initialize C to 00
        LDA     4150      Load the value to Acc.
        MOV     B, A      Move the content of Acc to B register.
        LDA     4151      Load the value to Acc.
        SUB     B
        JNC     LOOP      Jump on no carry.
        CMA               Complement Accumulator contents.
        INR     A         Increment value in Accumulator.
        INR     C         Increment value in register C
LOOP:   STA     4152      Store the value of A-reg to memory address.
        MOV     A, C      Move contents of register C to Accumulator.
        STA     4153      Store the value of Accumulator memory address.
        HLT               Terminate the program.
```

OBSERVATION:

Input: 06 (4150)
02 (4251)
Output: 04 (4152)
01 (4153)

RESULT:

Thus the program to subtract two 8-bit numbers was executed.

MULTIPLICATION OF TWO 8 BIT NUMBERS

AIM:

To perform the multiplication of two 8 bit numbers using 8085.

ALGORITHM:

1) Start the program by loading HL register pair with address of memory location.
2) Move the data to a register (B register).
3) Get the second data and load into Accumulator.
4) Add the two register contents.
5) Check for carry.
6) Increment the value of carry.
7) Check whether repeated addition is over and store the value of product and carry in memory location.
8) Terminate the program.

PROGRAM:

```
        MVI   D, 00     Initialize register D to 00
        MVI   A, 00     Initialize Accumulator content to 00
        LXI   H, 4150
        MOV   B, M      Get the first number in B - reg
        INX   H
        MOV   C, M      Get the second number in C- reg.
LOOP:   ADD   B         Add content of A - reg to register B.
        JNC   NEXT      Jump on no carry to NEXT.
        INR   D         Increment content of register D
NEXT:   DCR   C         Decrement content of register C.
        JNZ   LOOP      Jump on no zero to address
        STA   4152      Store the result in Memory
        MOV   A, D
        STA   4153      Store the MSB of result in Memory
        HLT             Terminate the program.
```

OBSERVATION:

Input:	FF (4150)
	FF (4151)
Output:	01 (4152)
	FE (4153)

RESULT:

Thus the program to multiply two 8-bit numbers was executed.

DIVISION OF TWO 8 BIT NUMBERS

AIM:

To perform the division of two 8 bit numbers using 8085.

ALGORITHM:

1) Start the program by loading HL register pair with address of memory location.
2) Move the data to a register(B register).
3) Get the second data and load into Accumulator.
4) Compare the two numbers to check for carry.
5) Subtract the two numbers.
6) Increment the value of carry .
7) Check whether repeated subtraction is over and store the value of product and carry in memory location.
8) Terminate the program.

PROGRAM:

	LXI	H, 4150	
	MOV	B, M	Get the dividend in B – reg.
	MVI	C, 00	Clear C – reg for qoutient
	INX	H	
	MOV	A, M	Get the divisor in A – reg.
NEXT:	CMP	B	Compare A - reg with register B.
	JC	LOOP	Jump on carry to LOOP
	SUB	B	Subtract A – reg from B- reg.
	INR	C	Increment content of register C.
	JMP	NEXT	Jump to NEXT
LOOP:	STA	4152	Store the remainder in Memory
	MOV	A, C	
	STA	4153	Store the quotient in memory
	HLT		Terminate the program.

OBSERVATION:

Input:	FF (4150) FF (4251)
Output:	01 (4152) ----- Remainder FE (4153) ---- Quotient

RESULT:

Thus the program to divide two 8-bit numbers was executed.

LARGEST NUMBER IN AN ARRAY OF DATA

AIM:

To find the largest number in an array of data using 8085 instruction set.

ALGORITHM:

1) Load the address of the first element of the array in HL pair
2) Move the count to B – reg.
3) Increment the pointer
4) Get the first data in A – reg.
5) Decrement the count.
6) Increment the pointer
7) Compare the content of memory addressed by HL pair with that of A - reg.
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 – reg.
10) Decrement the count
11) Check for Zero of the count. If ZF = 0, go to step 6, or if ZF = 1 go to next step.
12) Store the largest data in memory.
13) Terminate the program.

PROGRAM:

	LXI	H,4200	Set pointer for array
	MOV	B,M	Load the Count
	INX	H	
	MOV	A,M	Set 1st element as largest data
	DCR	B	Decrement the count
LOOP:	INX	H	
	CMP	M	If A- reg > M go to AHEAD
	JNC	AHEAD	
	MOV	A,M	Set the new value as largest
AHEAD:	DCR	B	
	JNZ	LOOP	Repeat comparisons till count = 0
	STA	4300	Store the largest value at 4300
	HLT		

OBSERVATION:

Input:	05 (4200) ----- Array Size 0A (4201) F1 (4202) 1F (4203) 26 (4204) FE (4205)
Output:	FE (4300)

RESULT:

Thus the program to find the largest number in an array of data was executed

SMALLEST NUMBER IN AN ARRAY OF DATA

AIM:

To find the smallest number in an array of data using 8085 instruction set.

ALGORITHM:

1) Load the address of the first element of the array in HL pair
2) Move the count to B – reg.
3) Increment the pointer
4) Get the first data in A – reg.
5) Decrement the count.
6) Increment the pointer
7) Compare the content of memory addressed by HL pair with that of A - reg.
8) If carry = 1, go to step 10 or if Carry = 0 go to step 9
9) Move the content of memory addressed by HL to A – reg.
10) Decrement the count
11) Check for Zero of the count. If ZF = 0, go to step 6, or if ZF = 1 go to next step.
12) Store the smallest data in memory.
13) Terminate the program.

PROGRAM:

	LXI	H,4200	Set pointer for array
	MOV	B,M	Load the Count
	INX	H	
	MOV	A,M	Set 1st element as largest data
	DCR	B	Decrement the count
LOOP:	INX	H	
	CMP	M	If A- reg < M go to AHEAD
	JC	AHEAD	
	MOV	A,M	Set the new value as smallest
AHEAD:	DCR	B	
	JNZ	LOOP	Repeat comparisons till count = 0
	STA	4300	Store the largest value at 4300
	HLT		

OBSERVATION:

Input:	05 (4200) ----- Array Size 0A (4201) F1 (4202) 1F (4203) 26 (4204) FE (4205)
Output:	0A (4300)

RESULT:

Thus the program to find the smallest number in an array of data was executed

ARRANGE AN ARRAY OF DATA IN ASCENDING ORDER

AIM:

To write a program to arrange an array of data in ascending order

ALGORITHM:

1. Initialize HL pair as memory pointer
2. Get the count at 4200 into C – register
3. Copy it in D – register (for bubble sort (N-1) times required)
4. Get the first value in A – register
5. Compare it with the value at next location.
6. If they are out of order, exchange the contents of A –register and Memory
7. Decrement D –register content by 1
8. Repeat steps 5 and 7 till the value in D- register become zero
9. Decrement C –register content by 1
10. Repeat steps 3 to 9 till the value in C – register becomes zero

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
```

OBSERVATION:

Input:	4200	05 (Array Size)
	4201	05
	4202	04
	4203	03
	4204	02
	4205	01

Output:	4200	05(Array Size)
	4201	01
	4202	02
	4203	03
	4204	04
	4205	05

RESULT:

Thus the given array of data was arranged in ascending order.

ARRANGE AN ARRAY OF DATA IN DESCENDING ORDER

AIM:

To write a program to arrange an array of data in descending order

ALGORITHM:

1. Initialize HL pair as memory pointer
2. Get the count at 4200 into C – register
3. Copy it in D – register (for bubble sort (N-1) times required)
4. Get the first value in A – register
5. Compare it with the value at next location.
6. If they are out of order, exchange the contents of A –register and Memory
7. Decrement D –register content by 1
8. Repeat steps 5 and 7 till the value in D- register become zero
9. Decrement C –register content by 1
10. Repeat steps 3 to 9 till the value in C – register becomes zero

PROGRAM:

```
        LXI     H,4200
        MOV     C,M
        DCR     C
REPEAT: MOV     D,C
        LXI     H,4201
LOOP:   MOV     A,M
        INX     H
        CMP     M
        JNC     SKIP
        MOV     B,M
        MOV     M,A
        DCX     H
        MOV     M,B
        INX     H
SKIP:   DCR     D
        JNZ     LOOP
        DCR     C
        JNZ     REPEAT
        HLT
```

OBSERVATION:

Input:	4200	05 (Array Size)
	4201	01
	4202	02
	4203	03
	4204	04
	4205	05

Output:	4200	05(Array Size)
	4201	05
	4202	04
	4203	03
	4204	02
	4205	01

RESULT:

Thus the given array of data was arranged in descending order.

BCD TO HEX CONVERSION

AIM:

To convert two BCD numbers in memory to the equivalent HEX number using 8085 instruction set

ALGORITHM:

1) Initialize memory pointer to 4150 H
2) Get the Most Significant Digit (MSD)
3) Multiply the MSD by ten using repeated addition
4) Add the Least Significant Digit (LSD) to the result obtained in previous step
5) Store the HEX data in Memory

PROGRAM:

```
LXI     H,4150
MOV     A,M     Initialize memory pointer
ADD     A       MSD X 2
MOV     B,A     Store MSD X 2
ADD     A       MSD X 4
ADD     A       MSD X 8
ADD     B       MSD X 10
INX     H       Point to LSD
ADD     M       Add to form HEX
INX     H
MOV     M,A     Store the result
HLT
```

OBSERVATION:

Input: 4150 : 02 (MSD)
4151 : 09 (LSD)

Output: 4152 : 1D H

RESULT:

Thus the program to convert BCD data to HEX data was executed.

HEX TO BCD CONVERSION

AIM:

To convert given Hexa decimal number into its equivalent BCD number using 8085 instruction set

ALGORITHM:

1) Initialize memory pointer to 4150 H
2) Get the Hexa decimal number in C - register
3) Perform repeated addition for C number of times
4) Adjust for BCD in each step
5) Store the BCD data in Memory

PROGRAM:

```
        LXI     H,4150      Initialize memory pointer
        MVI     D,00        Clear D- reg for Most significant Byte
        XRA     A           Clear Accumulator
        MOV     C,M         Get HEX data
LOOP2:  ADI     01          Count the number one by one
        DAA                 Adjust for BCD count
        JNC     LOOP1
        INR     D
LOOP1:  DCR     C
        JNZ     LOOP2
        STA     4151        Store the Least Significant Byte
        MOV     A,D
        STA     4152        Store the Most Significant Byte
        HLT
```

OBSERVATION:

Input:	4150 : FF
Output:	4151 : 55 (LSB) 4152 : 02 (MSB)

RESULT:

Thus the program to convert HEX data to BCD data was executed.

HEX TO ASCII CONVERSION

AIM:

To convert given Hexa decimal number into its equivalent ASCII number using 8085 instruction set.

ALGORITHM:

1. Load the given data in A- register and move to B – register
2. Mask the upper nibble of the Hexa decimal number in A – register
3. Call subroutine to get ASCII of lower nibble
4. Store it in memory
5. Move B –register to A – register and mask the lower nibble
6. Rotate the upper nibble to lower nibble position
7. Call subroutine to get ASCII of upper nibble
8. Store it in memory
9. Terminate the program.

PROGRAM:

```
        LDA     4200    Get Hexa Data
        MOV     B,A
        ANI     0F      Mask Upper Nibble
        CALL    SUB1    Get ASCII code for upper nibble
        STA     4201
        MOV     A,B
        ANI     F0      Mask Lower Nibble
        RLC
        RLC
        RLC
        RLC
        CALL    SUB1    Get ASCII code for lower nibble
        STA     4202
        HLT

SUB1:   CPI     0A
        JC      SKIP
        ADI     07
SKIP:   ADI     30
        RET
```

OBSERVATION:

Input:	4200	E4(Hexa data)
Output:	4201	34(ASCII Code for 4)
	4202	45(ASCII Code for E)

RESULT:

Thus the given Hexa decimal number was converted into its equivalent ASCII Code.

ASCII TO HEX CONVERSION

AIM:

To convert given ASCII Character into its equivalent Hexa Decimal number using 8085 instruction set.

ALGORITHM:

1. Load the given data in A- register
2. Subtract 30 H from A – register
3. Compare the content of A – register with 0A H
4. If A < 0A H, jump to step6. Else proceed to next step.
5. Subtract 07 H from A – register
6. Store the result
7. Terminate the program

PROGRAM:

```
        LDA   4500
        SUI   30
        CPI   0A
        JC    SKIP
        SUI   07
SKIP:   STA   4501
        HLT
```

OBSERVATION:

Input:	4500	31
Output:	4501	0B

RESULT:

Thus the given ASCII character was converted into its equivalent Hexa Value.

SQUARE OF A NUMBER USING LOOK UP TABLE

AIM:

To find the square of the number from 0 to 9 using a Table of Square.

ALGORITHM:

1. Initialize HL pair to point Look up table
2. Get the data .
3. Check whether the given input is less than 9.
4. If yes go to next step else halt the program
5. Add the desired address with the accumulator content
6. Store the result

PROGRAM:

```
        LXI   H,4125      Initialsie Look up table address
        LDA   4150        Get the data
        CPI   0A          Check input > 9
        JC    AFTER       if yes error
        MVI   A,FF        Error Indication
        STA   4151
        HLT
AFTER:  MOV   C,A     Add the desired Address
        MVI   B,00
        DAD   B
        MOV   A,M
        STA   4151    Store the result
        HLT           Terminate the program
```

LOOKUP TABLE:

4125	01
4126	04
4127	09
4128	16
4129	25
4130	36
4131	49
4132	64
4133	81

OBSERVATION:

Input:	4150:	05
Output:	4151	25 (Square)

Input :	4150:	11
Output:	4151:	FF (Error Indication)

RESULT:

Thus the program to find the square of the number from 0 to 9 using a Look up table was executed.