Microprocessors Lab Manual

Department of Instrumentation Engineering

Jorhat Engineering College

Marks distribution:

Categories		Marks
End semester exam :		
Flowchart	:	05
Write the program	:	10
Result and discussion	:	05
Program execution	:	05
Viva (External)	:	10
Total (semester)	:	35
Internal assessment:		
Attendance	:	05
Experiments (Continuous assessment)	:	05
Lab Journal	:	05
Total (IA)	:	15

Total (End semester + AI)	**:**	**50**

Course outcomes (CO):

CO1 : Build an operational idea of typical microprocessor (8085) trainer kit with the students

CO2 : Develop programming strategies, identify proper mnemonics and run their program on the training board

CO3 : Examine interfacing I/O devices with 8085 microprocessor kit

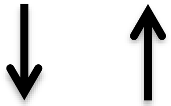
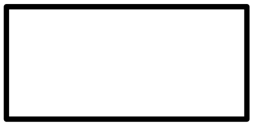
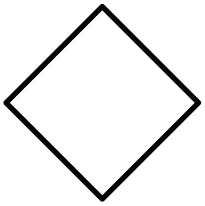
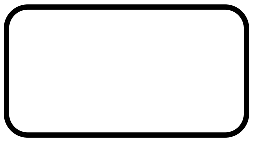
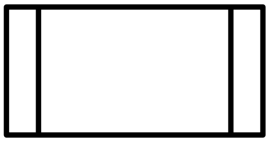
CO4 : Make use of team-based laboratory activities, students will able to interact with fellow students, and complete assignments

CO-PO mapping:

Course Outcome	Program Outcome												PS01	PS02
	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12		
CO1	2	1	1											1
CO2	2	1	2	1	1							1		
CO3	2	2	2	2	1	1						1		
CO4						1	1	1	2	1		1		

List of the experiments:

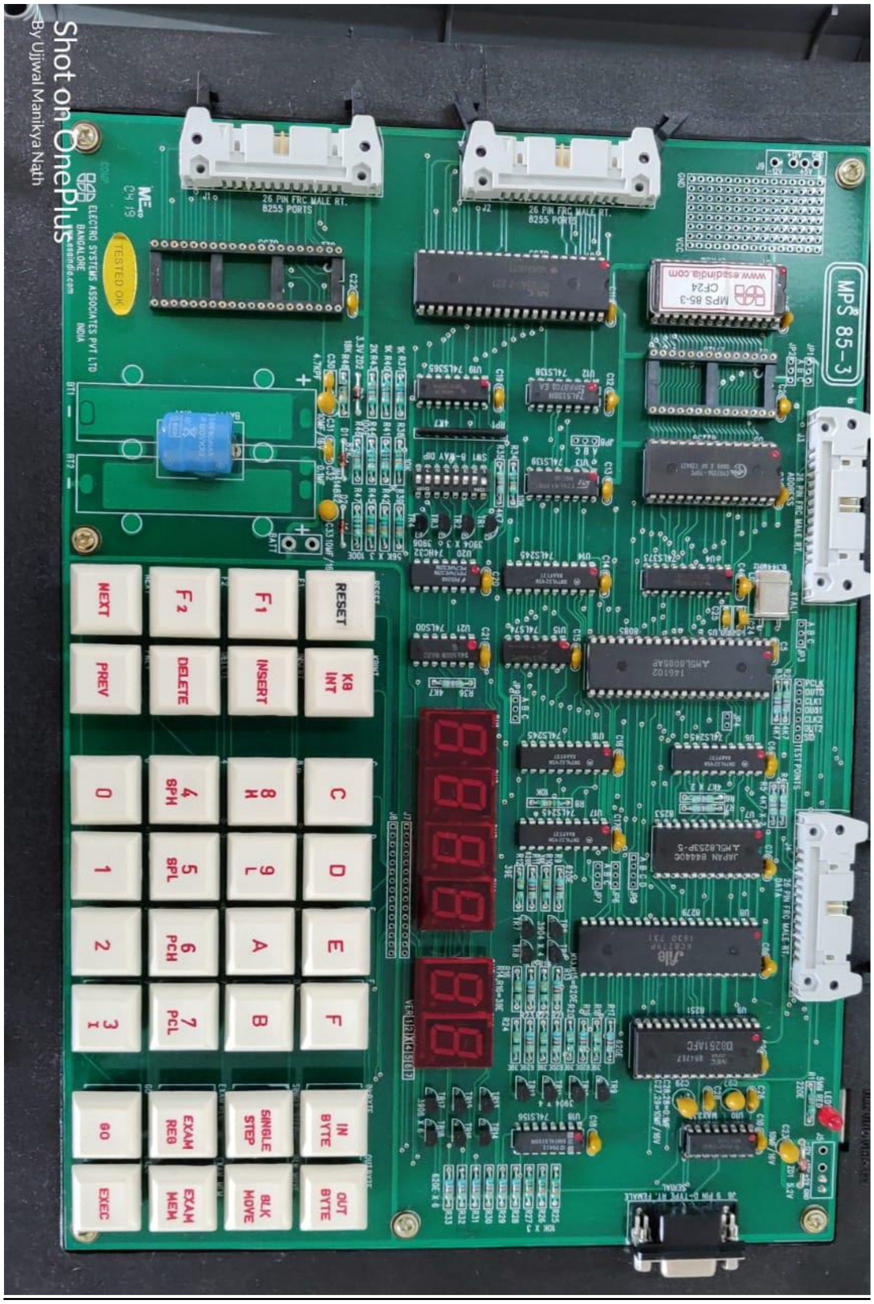
Basic blocks of a flow chat

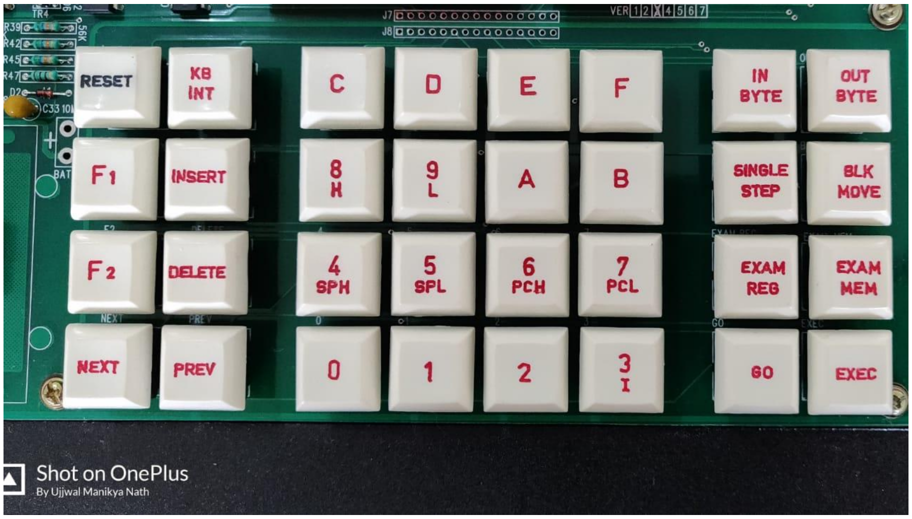
	Direction of program execution
	Process or an operation
	Decision making block
	Beginning or end of the program
	Subroutine

How to write a program in lab journal

Address	Mnemonics	Machine code (Hex)		Remarks
8000	MVI A, 80H	3E	Opcode	% Move 80H to Acc %
8001		80	Operand	
8002	MVI B, 02H	06	Opcode	% Move 02H to Reg. B %
8003		02	Operand	
8004	ADD B	80	Opcode	% Add content of Reg. B with content of Acc. %
8005	STA 8050	32	Opcode	% Store the content of Acc. To memory location 8050 %
8006		50	Operand	
8007		80	Operand	
8008	HLT	76	Opcode	% Execute the program %

Real-time Snapshoot of microprocessor kit





To write the program

- Rest
- Exam mem
- 8000
- Next
- Next
- …

Execute the program

- Reset
- Go
- 8000
- Exec

To see the output

- Reset
- Exam mem
- Storing address (8050)
- Next

Exp. 1: 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    8150    Load the value to Accumulator.
        MOV    B, A    Move the content of Accumulator to B register.
        LDA    8151    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    8152    Store the value of Accumulator (SUM).
        MOV    A, C    Move content of register C to Acc.
        STA    8153    Store the value of Accumulator (CARRY)
        HLT            Halt the program.
```

OBSERVATION:

Input:	82 (8150)
	96 (8251)
Output:	18 (8152)
	01 (8153)

RESULT:

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

Exp. 2: 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    8150     Load the value to Acc.
         MOV    B, A     Move the content of Acc to B  register.
         LDA    8151     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    8152     Store the value of A-reg to memory address.
         MOV    A, C     Move contents of register C to Accumulator.
         STA    8153     Store the value of Accumulator memory address.
         HLT             Terminate the program.
```

OBSERVATION:

Input: 06 (8150)
02 (8251)
Output: 04 (8152)
01 (8153)

RESULT:

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

Exp. 3: 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, 8150
         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   8152     Store the result in Memory
         MOV   A, D
         STA   8153     Store the MSB of result in Memory
         HLT            Terminate the program.
```

OBSERVATION:

Input:	FF (8150)
	FF (8151)
Output:	01 (8152)
	FE (8153)

RESULT:

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

Exp. 4: 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, 8150
          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     8152        Store the remainder in Memory
          MOV     A, C
           STA    8153        Store the quotient in memory
           HLT                Terminate the program.
```

OBSERVATION:

Input: FF (8150)
FF (8251)

Output: 01 (8152) ----- Remainder
FE (8153) ---- Quotient

RESULT:

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

Exp. 5: 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,8200     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     8300       Store the largest value at 8300
         HLT
```

OBSERVATION:

Input: 05 (8200) ----- Array Size
0A (8201)
F1 (8202)
1F (8203)
26 (8204)
FE (8205)

Output: FE (8300)

RESULT:

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

Exp. 6: 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,8200    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     8300      Store the largest value at 8300
          HLT
```

OBSERVATION:

Input: 05 (8200) ----- Array Size
0A (8201)
F1 (8202)
1F (8203)
26 (8204)
FE (8205)

Output: 0A (8300)

RESULT:

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

Exp. 7: 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 8200 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,8200
            MOV     C,M
            DCR     C
REPEAT:     MOV     D,C
            LXI     H,8201
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:	8200	05 (Array Size)
	8201	05
	8202	04
	8203	03
	8204	02
	8205	01

Output:	8200	05(Array Size)
	8201	01
	8202	02
	8203	03
	8204	04
	8205	05

RESULT:

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

Exp. 8: 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 8200 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,8200
                MOV         C,M
                DCR         C
REPEAT:         MOV         D,C
                LXI         H,8201
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:	8200	05 (Array Size)
	8201	01
	8202	02
	8203	03
	8204	04
	8205	05

Output:	8200	05(Array Size)
	8201	05
	8202	04
	8203	03
	8204	02
	8205	01

RESULT:

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

Exp. 9: 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 8150 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,8150     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     8151       Store the Least Significant Byte
            MOV     A,D
            STA     8152       Store the Most Significant Byte
            HLT
```

OBSERVATION:

Input:	8150 : FF
Output:	8151 : 55 (LSB) 8152 : 02 (MSB)

RESULT:

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

Exp 10: 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,8125       Initialsie Look up table address
          LDA   8150         Get the data
          CPI   0A           Check input > 9
          JC    AFTER        if yes error
          MVI   A,FF         Error Indication
          STA   8151
          HLT
AFTER:    MOV   C,A     Add the desired Address
          MVI   B,00
          DAD   B
          MOV   A,M
          STA   8151    Store the result
          HLT           Terminate the program
```

LOOKUP TABLE:

8125	01
8126	04
8127	09
8128	16
8129	25
8130	36
8131	49
8132	64
8133	81

OBSERVATION:

Input:	8150:	05
Output:	8151	25 (Square)

Input :	8150:	11
Output:	8151:	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.

EXP. 11: INTERFACING ADC WITH 8085 PROCESSOR

AIM:

To write a program to initiate ADC and to store the digital data in memory

PROGRAM:

```
        MVI     A,10
        OUT     C8
        MVI     A,18
        OUT     C8
        MVI     A,10
        OUT     D0
        XRA     A
        XRA     A
        XRA     A
        MVI     A,00
        OUT     D0
LOOP:   IN      D8
        ANI     01
        CPI     01
        JNZ     LOOP
        IN      C0
        STA     8150
        HLT
```

OBSERVATION:

Compare the data displayed at the LEDs with that stored at location 8150

RESULT:

Thus the ADC was initiated and the digital data was stored at desired location

EXP. 12: INTERFACING 8279 KEYBOARD/DISPLAY CONTROLLERWITH 8085 MICROPROCESSOR

AIM:

To interface 8279 Programmable Keyboard Display Controller to 8085 Microprocessor.

APPARATUS REQUIRED:

1) 8085 Microprocessor toolkit.
2) 8279 Interface board.
3) VXT parallel bus.
4) Regulated D.C power supply.

PROGRAM:

```
START:      LXI       H,8130H
            MVI       D,0FH ;Initialize counter.
            MVI       A,10H
            OUT       C2H    ;Set Mode and Display.
            MVI       A,CCH;Clear display.
            OUT       C2H
            MVI       A,90H ;Write Display
            OUT       C2H
LOOP:       MOV        A,M
            OUT       C0H
            CALL      DELAY
            INX       H
            DCR       D
            JNZ       LOOP
            JMP       START

DELAY:      MVI       B, A0H
LOOP2:      MVI       C, FFH
LOOP1:      DCR       C
            JNZ       LOOP1
            DCR       B
            JNZ       LOOP2
            RET
```

Pointer equal to 8130 .FF repeated eight times.

```
8130    - FF
8131    –FF
8132    –FF
8133    –FF
8134    –FF
8135    –FF
8136    –FF
8137    –FF
8138   –98
8139   –68
813A    -7C
813B    -C8
813C    -1C
813D    -29
813E    -FF
813F     -FF
```

RESULT:

Thus 8279 controller was interfaced with 8085 and program for rolling display was executed successfully.