Department of Electronics Engineering

Lab Manual
Of
Microprocessor and Microcontroller Lab

Course code: ECC 211

New Academic Complex, 6th Floor Room no 622

IIT (ISM) Dhanbad

Do’s

- Be punctual.
- Maintain discipline & silence.
- Keep the Laboratory clean and tidy.
- Enter Laboratory with shoes.
- Handle instruments with utmost care.
- Come prepared with circuit diagrams, writing materials and calculator.
- Follow the procedure that has been instructed.
- Return all the issued equipment’s properly.
- Get the signature on experiment result sheet daily.
- For any clarification contact faculty/staff in charge only.
- Shut down the power supply after the experiment.
- Don’t take components along with you.

Don’ts

- Avoid unnecessary chat or walk.
- Playing mischief in the laboratory is forbidden.
- Disfiguring of furniture is prohibited.
- Do not start the experiment without instructions.
- Avoid using cell phones.
- Avoid late submission of laboratory reports.

List of Experiments

.

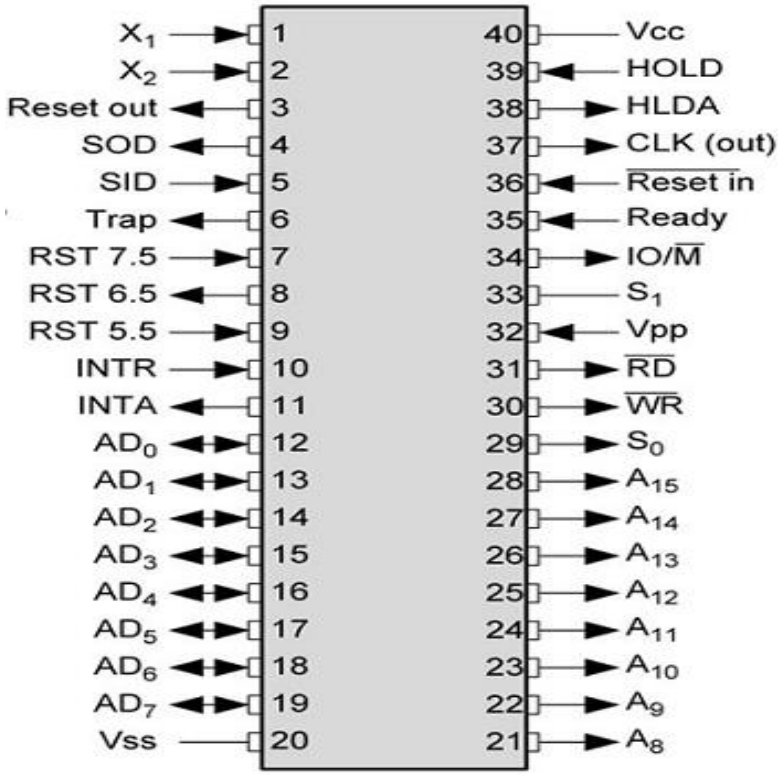
Experiment 1

Brief Introduction of 8085 Microprocessor and its Trainer Kit.

Aim: Introduction of 8085 Microprocessor and Its Trainer kit**.**

Theory: 8085 is an **8-bit microprocessor** as it operates on 8 bits at a time. It is a 40 pin dual-in-line chip.

Pin diagram of 8085 Microprocessor −



The pins of an 8085 microprocessor can be classified into seven Groups −

Address Bus: A15-A8, it carries the most significant 8-bits of memory I/O address.

Data bus: AD7-AD0, it carries the least significant 8-bit address and data bus.

Control and status signals: These signals are used to identify the nature of operation. There are 3 control signal and 3 status signals**.**

Three control signals are RD, WR & ALE.

- **RD** − This signal indicates that the selected IO or memory device is to be read and is ready for accepting data available on the data bus.
- **WR** − This signal indicates that the data on the data bus is to be written into a selected memory or IO location.
- **ALE** − It is a positive going pulse generated when a new operation is started by the microprocessor. When the pulse goes high, it indicates address. When the pulse goes down it indicates data.

Three status signals are IO/M, S0 & S1:-

IO/M - This signal is used to differentiate between IO and Memory Operations, i.e., when it is high indicates IO operation and when it is low then it indicates memory operation.

S1 & S0: These signals are used to identify the type of current operation.

Power supply: There are 2 power supply signals − VCC & VSS. VCC indicates +5v Power supply and VSS indicates ground signal.

Clock signals: There are 3 clock signals, i.e. X1, X2, CLK OUT.

- **X1, X2** − A crystal (RC, LC N/W) is connected at these two pins and is used to set frequency of the internal clock generator. This frequency is internally divided by 2.
- **CLK OUT** − This signal is used as the system clock for devices connected with the microprocessor.
- **Interrupts & externally initiated signals**: Interrupts are the signals generated by external devices to request the microprocessor to perform a task. There are 5 interrupt signals, i.e., TRAP, RST 7.5, RST 6.5, RST 5.5, and INTR.
- **INTA** − It is an interrupt acknowledgment signal.
- **RESET IN** − This signal is used to reset the microprocessor by setting the program counter to zero.
- **RESET OUT** − This signal is used to reset all the connected devices when the microprocessor is reset.
- **READY** − This signal indicates that the device is ready to send or receive data. If READY is low, then the CPU has to wait for READY to go high.
- **HOLD** − this signal indicates that another master is requesting the use of the address and data buses.

- **HLDA (HOLD Acknowledge)** – It indicates that the CPU has received the HOLD request and it will relinquish the bus in the next clock cycle. HLDA is set to low after the HOLD signal is removed.
- **Serial I/O signals**: There are 2 serial signals, i.e., SID and SOD and these signals are used for serial communication.
- **SOD** (Serial output data line) – The output SOD is set/reset as specified by the SIM instruction.
- **SID** (Serial input data line) – The data on this line is loaded into accumulator whenever a RIM instruction is executed**.**

Architecture of 8085 Microprocessor:

1. General purpose register: It is an 8 bit register i.e. B, C, D, E, H, L. The combination of 8 bit register is known as register pair, which can hold 16 bit data. The HL pair is used to act as memory pointer is accessible to program**.**
2. Accumulator: It is an 8 bit register which hold one of the

   Data to be processed by ALU and stored the result of the operation.
3. Program counter (PC): It is a 16 bit pointer which maintain the address of a byte entered to line stack.
4. Stack pointer (SP): It is a 16 bit special purpose register which is used to hold line memory address for line next instruction to be executed.
5. Arithmetic and logical unit: It carries out arithmetic and logical operation by 8 bit address it uses the accumulator content as input the ALU result is stored back into accumulator.
6. Temporary register**:** It is an 8 bit register associated with ALU hold data, entering an operation, used by the microprocessor and not accessible to programs.
7. Flags: Flag register is a group of fire, individual flip flops line content of line flag register will change after execution of arithmetic and logic operation.

The line status flags are:

Carry flag

Parity flag (P)

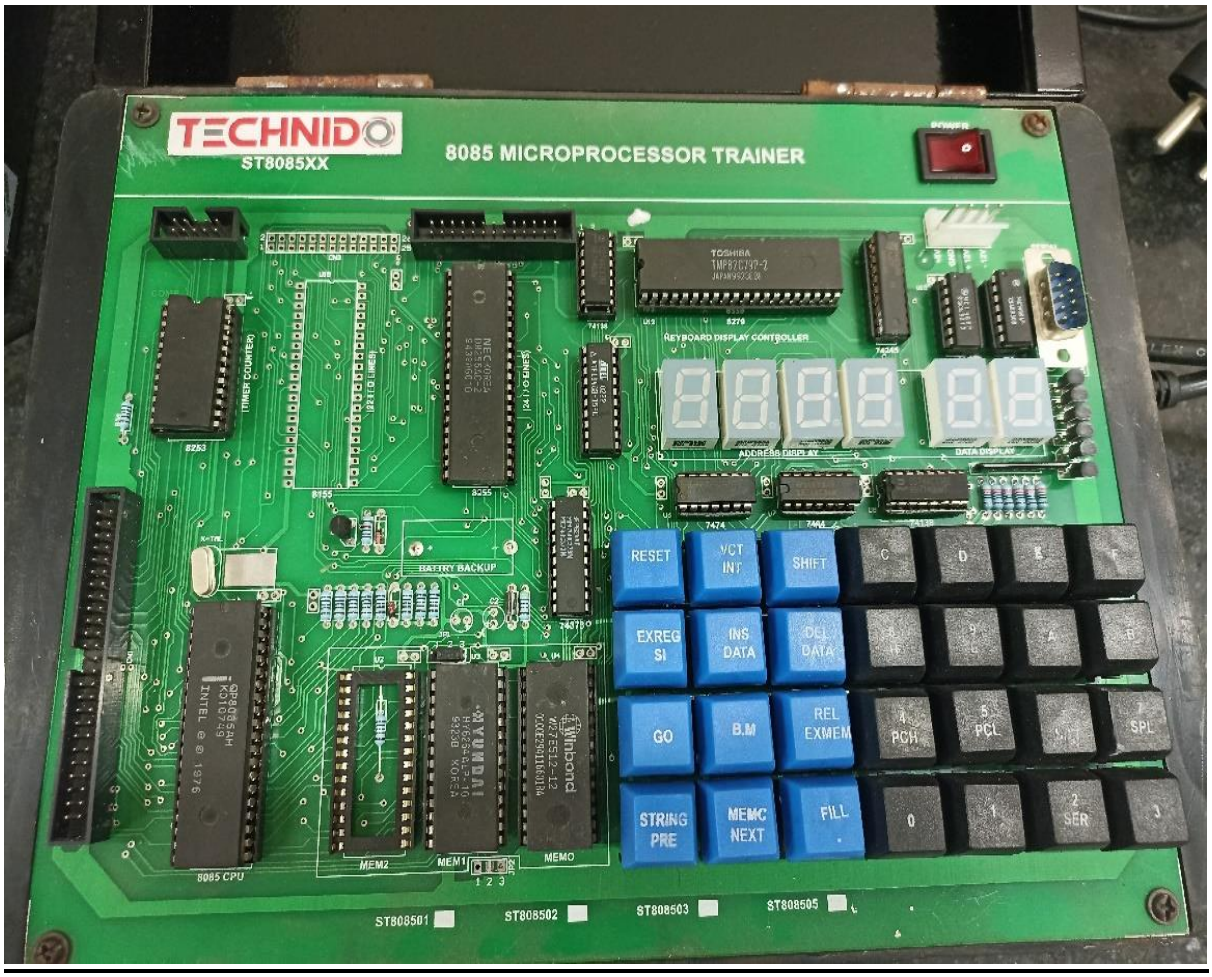
Zero flag (Z)

Auxiliary carry flag (AC)

Sign flag (S)

8. **Timing and control unit:** Synchronous all microprocessor, operation with the clock and generator and control signal from it necessary to communicate between controller and peripherals.
9. **Instruction registers and decoder:** Instruction is fetched from line memory and stored in line instruction register decoder the stored information.
10. **Register Array:** These are used to store 8-bit data during execution of some instruction.

8085 Microprocessor Trainer



8279 (Keyboard and Display Controller)

8279 is a general-purpose programmable keyboard and display i/o interface device designed for use with the 8085 microprocessor.it provides a scanned interface to 28 contact key matrix provided in VMC-850X and scanned interface for the six seven segment display.8279 has got 16 X 8 display RAM which can be loaded or interrogated by the CPU. When a key is pressed, its corresponding code is entered in the FIFO queue of 8279 and can now be read by the microprocessor.

8255 (Programmable and Peripheral Interface)

8255 is a Programmable and Peripheral Interface designed to use with 8085 Microprocessor. It has got three Input/output ports of 8 lines each (PORT –A, PORT-B, PORT-C). Port c can be divided into two ports of 4 lines each named as port C upper and port C lower.

8253 (Programmable Internal Timer)

This chip is a programmable interval Timer/Counter and can be used for the generation of accurate time delays.

Display

VMC-850x provides 6 digits of seven segment display .Four digits are for displaying the address of any location or name of any register, whereas the rest of the two digits are meant for displaying the contents of a memory location or of a register. All the six digits of the display ae n hexadecimal notation.

Some commonly used Command keys:

Reset	Reset the system
VCT INT	Hardware interrupt via keyboard RST 7.5.
SHIFT	Provides a second level command to all keys.
GO	To execute the program.
SI	To execute the program in single step mode.
EXREG	Examine Register; allows user to examine and modify the contents of different registers.
EXMEM	Examine Memory; allows user to examine any memory location and modify any RAM location
PRE	Previous is used as an intermediate terminator in case of Examine Memory. It decrements the PC contents and writes the contents of data fields to the address displayed in the

	address location.
Next	Increment is used as an intermediate terminator in case of Examine Memory, Examine Register etc. It increments the PC Contents and writes the data lying in data field at the location displayed in address field.
DEL	Delete the part of program or data, with relocation by one or more bytes.
INS	Inserts the part of the program or data with relocation, by one or more bytes.
B.M.	Allows user to move a block of memory to any RAM area.
FILL	Allows user to fill RAM area with a constant.
REL	Relocates a program written for some memory area and to be transferred to other memory area.
INSDATA	Inserts one or more data bytes in the user's program/data area.
DELDATA	Deletes one or more data bytes from the user's program/data area.
STRING	Finds out the string of data lying at a particular address or addresses.
MEMC	Memory Compare: Compares two blocks of memory for equality.
0-F	Hexadecimal Keys.

For Entering Program into 8085 Trainer Kit

1. Enter the Starting address, press NEXT, the contents of the entered address is displayed.
2. On pressing NEXT, one instruction is executed and the address of the next instruction will be displayed with its data in the data field.
3. If one wants to terminate and command at any stage, one can do that using (.) key, a'-'is displayed in the address field.

Result:

Thus 8085 microprocessors were studied successfully.

Experiment No 2

Write an assembly language program to add, subtract, multiply and divide two 8 bit numbers.

Aim: Performing assembly language Programming of addition, Subtraction, multiplication, division using 8085 Microprocessors.

Apparatus Required: 8085 Trainer kit.

Objective:

1. To learn and understand the assembly instructions

2. To develop an 8085-assembly language program to add, sub,

Multiplication and divide.

Program:

Addition of two 8-bit number

Address	Opcode	Mnemonics
2000	21 09 20	LXI H,2009
2003	7E	MOV A, M
2004	23	INX H
2005	86	ADD M
2006	23	INX H
2007	77	MOV M, A
2008	EF	RST 5
2009	DATA	02
200A	DATA	03
200B	RESULT	05

Procedure: -

1. Press Reset Key.
2. Press 'REL EXMEM' Key
3. Type starting address of the program (e.g.: - 2000)

4. Press ‘Next’ key
5. Type the Opcode of the program from starting to End by Pressing Next Key.

(Next Key is used to go one by one-by-one memory location.)

6. Press Reset key
7. Press ‘GO’ then type starting address. (i.e., 2000)
8. Press ‘Fill’ key then press reset key.
9. Press ‘REL EXMEM’ key then type the o/p address (i.e., 200B)
10. Press ‘Next’ key
11. Data will be displayed on seven segment.

Subtraction of Two Number

Address	Mnemonics	Hex code	Remarks
0000	MVI A,23H	3E	Load accumulator with 23H
0001		23	
0002	MVI B,12H	06	Load register bb with 12H
0003		12	
0004	SUB B	90	Subtract contents of B from accumulator
0005	HLT	76	Halts Program execution

Final value stored in Register

Accumulator	Register B
11	12

Multiplication of two numbers

Address	Label	Mnemonics	Hex code	Remarks
0000		MVI A,00	3E	Load accumulator with 00H
0001			00	
0002		MVI B,05	06	Load register bb with 05H
0003			05	
0004	LOOP	ADD B	80	Add contents of B to A
0005		DCR C	0D	Decrement c by 1
0006		JNZ LOOP	C2	Jump to loop statement if c is not zero
0007			04	
0008			00	
0009		HLT	76	

Division of two number

Address	Label	Mnemonics	Hex code	Remarks
0000		MVI A,14H	3E	Load accumulator with 14H
0001			14	
0002		MVI B,04	06	Load register B with 04H
0003			04	
0004		MVI C,00	0E	Load register B with 00H
0005			00	
0006	BACK	SUB B	90	
0007		INR C	0C	Increment c by 1
0008		CMP B	B8	
0009		JC END	DA	
000A			0F	
000B			00	
000C		JMP BACK	C3	
000D			06	
000E			00	
000F	END	HLT	76	

Conclusion: The assembly language program for 8 bit addition, subtraction, multiplication and division of two numbers was executed successfully by using 8085 microprocessor kit.

Experiment 3

Sorting of 10 numbers in ascending order

Aim: To perform sorting of 10 numbers in 8085 microprocessor

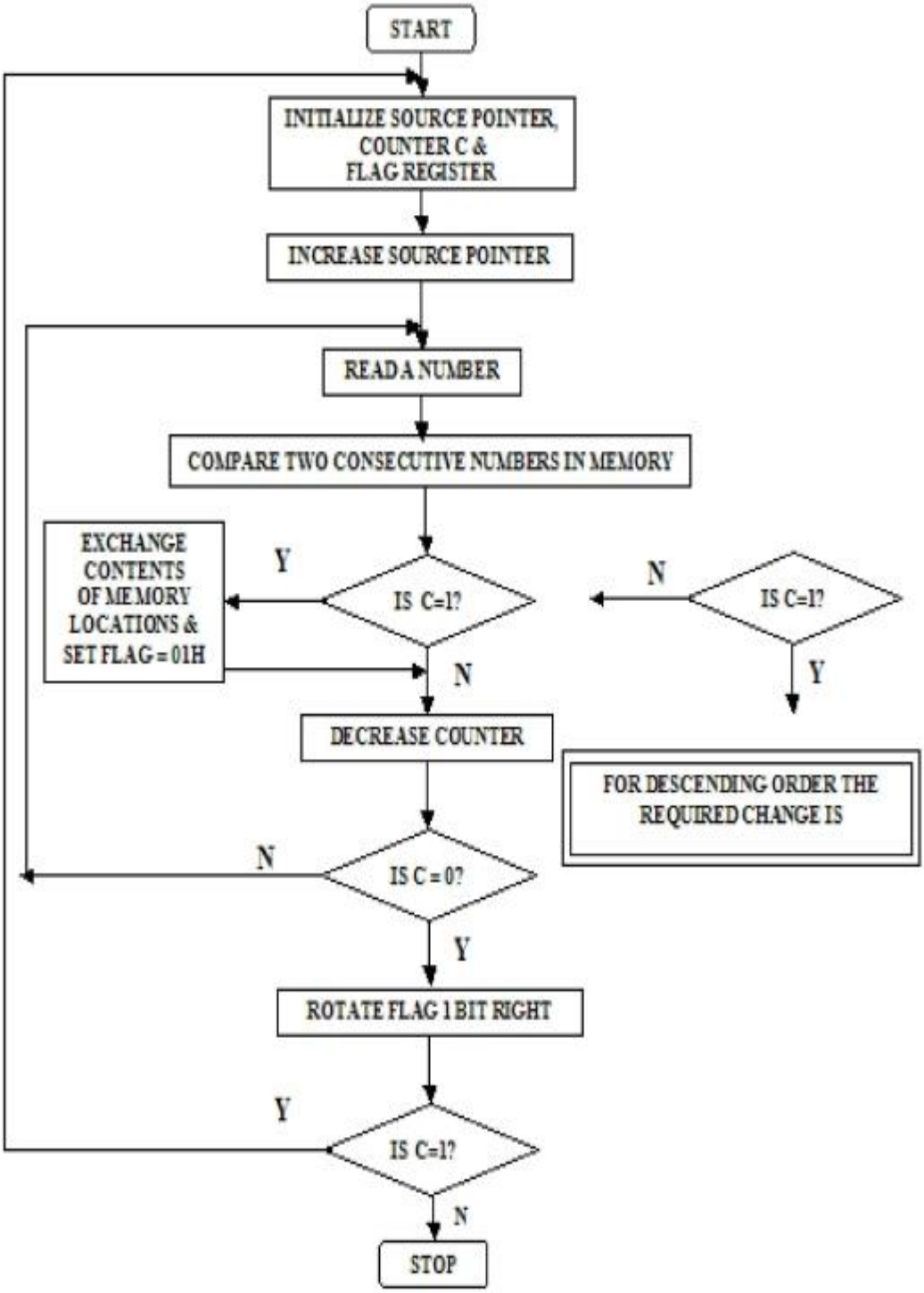
Components Required: 8085 Microprocessor kit, Simulator, Breadboard and Jumper Wires

Theory: In this program we will arrange the numbers in bubble sorting technique. In this sorting technique, it will be executed in different pass. In each pass the largest number is stored at the end of the list. Here we are taking the numbers from location 8040H to 8046H. The array size is stored at8040H.

Let us assume the inputs of the program are-

Address	Data
8040	06
8041	06
8042	05
8043	04
8044	01
8045	02
8046	03

Steps of Experiment:



Program-

Address	Labels	Mnemonics	Comments
8000	START	LXI H, 8040H	Pointer to the IN-BUFFER
8003		MVI D, 00H	The D register is used as a flag register
8005		MOV C, M	Initialize reg. C with data count
8006		DCR C	Set Reg. C for comparison count
8007		INX H	Pointing to the next location
8008	CHECK	MOV A, M	Get the number
8009		INX H	Go to next location
800A		CMP M	Compare the contents of the current memory location with the contents of the accumulator
800B		JC	If (A) < second byte,

Address	Labels	Mnemonics	Comments
		NEXTBYT	do not exchange
800E		MOV B, M	Get second byte for exchange
800F		MOV M, A	Store first byte in second location
8010		DCX H	Point to first location
8011		MOV M, B	Store second byte in first location
8012		INX H	Get ready for next comparison
8013		MVI D, 01H	Load 1 in D as a remainder for exchange
8015	NEXTBYT	DCR C	Decrement comparison count
8016		JNZ CHECK	If comparison count not 0, go back
8019		MOV A, D	Get flag bit in A
801A		RRC	Place flag bit D0in carry

Address	Labels	Mnemonics	Comments
801B		JC START	If flag is 1, exchange occurred
801E		HLT	Terminate the program

Output of the experiment-

Address	Data
8041	01
8042	02
8043	03
8044	04
8045	05
8046	06

Precautions:

1. Memory is set to be clear after the end of the program.
2. Memory should be interred carefully otherwise its shows program has some error or it will interrupt the program differently and executed accordingly which cannot be mitigated.
3. After editing the code, the assembly button must be preset before executing.

Experiment 4

Find the factorial of a given number and searching the smallest number in a given array

---

Aim: To find the factorial of a given number and searching the smallest number in a given array in 8085 microprocessors

Components Required: 8085 Microprocessor, Simulator, Breadboard, and Jumper wires.

Experiment 4 (a) - To find the factorial of a given number.

Theory- Factorial is a mathematical function that multiplies the number by every natural number below it. Some important points should be considered before performing experiment.

- We only calculate factorial of a Natural number.
- Natural number does not include negative and zero.
- The factorial of a negative number does not exist.
- Factorial of zero is 1.
- Example1- Factorial of 5= 5!=5*4*3*2*1=120=78H
- Example2- 4!=18H

Steps of Experiment/ Algorithm:

Step 1: Load the data into register B
Step 2: To start multiplication set D to 01H
Step 3: Jump to step 7
Step 4: Decrements B to multiply previous number
Step 5: Jump to step 3 till value of B>0
Step 6: Take memory pointer to next location and store result
Step 7: Load E with contents of B and clear accumulator
Step8: Repeatedly add contents of D to accumulator E times
Step9: Store accumulator content to D
Step10: Go to step 4

Program:

Address	Label	Mnemonics	Remarks
4000 H	Data		Data byte
4001 H	Result		Result
4002 H		LXIH,4000H	Loading data from memory
4005 H		MOV B,M	Load data to B register
4006 H		MVID,01H	Set D register with 1
4008 H	LOOP1	CALL LOOP2	Subroutine call for loop2
400B H		DCR B	Decrement B
400C H		JNZ LOOP1	Call loop1 till B becomes 0
400F H		INX H	Increment memory
4010 H		MOV M,D	Store result in memory
4011 H		HLT	Halt
4100 H	LOOP2	MOV E,B	
4101 H		MVI A,00H	Clear Accumulator
4103 H	LOOP3	ADD D	Add contents of D to A
4104 H		DCR E	Decrement E
4105 H		JNZ LOOP3	Call LOOP3 till E becomes zero
4108 H		MOV D,A	Transfer contents of A to D
4109 H		RET	return from subroutine

Experiment 4 (b) - Searching the smallest number in a given array

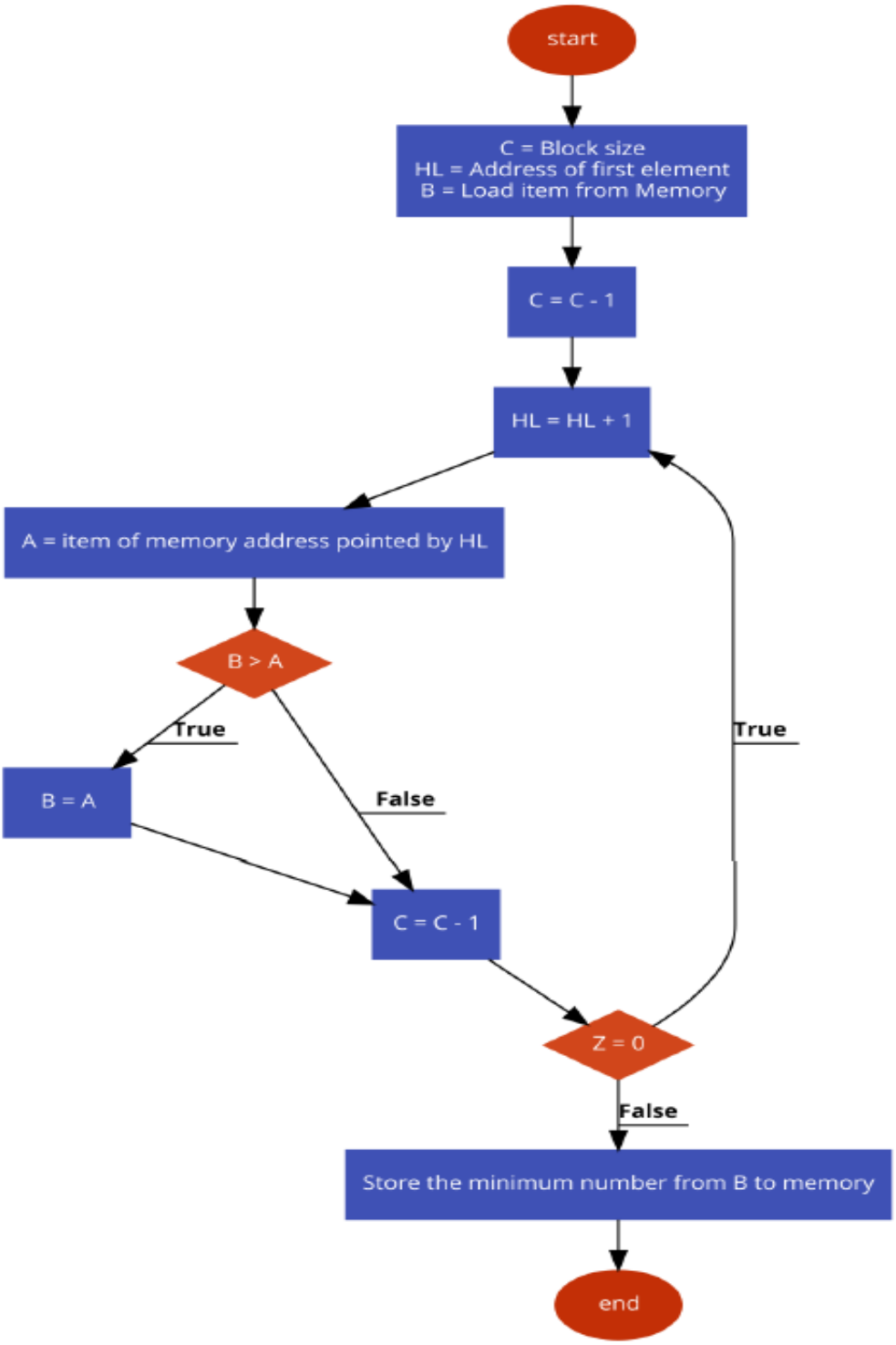
Let us assume that the inputs are-

Address	Data
8000	06
8001	55
8002	22
8003	44
8004	11
8005	33
8006	66

Steps of the experiment/Algorithm-

Step 1: Load the address of the first element of the array in HL pair
Step 2: Move the count to C
Step 3: Increment the pointer
Step 4: Get the first data in B
Step 5: Decrement the count
Step 6: Increment the pointer H
Step 7: Compare the content of memory addressed by HL pair with that of A
Step8: If carry = 1, go to step 10 or if Carry = 0 go to step 9.
Step9: Move the content of memory addressed by HL to A
Step10: Decrement the count
Step11: Check for Zero of the count. If ZF = 0, go to step 6, or if ZF = 1 go to next step
Step 12: Store the smallest data in memory.
Step 13: Terminate the program.

Flow Chart


start
C = Block size
HL = Address of first element
B = Load item from Memory
C = C - 1
HL = HL + 1
A = item of memory address pointed by HL
B > A
True
False
B = A
C = C - 1
Z = 0
True
False
Store the minimum number from B to memory
end

Program

Address	Labels	Mnemonics	Remarks
F000		LXI H,8000H	Point to get array size
F003		MOV C, M	Get the size of array
F004		INX H	Point to actual array
F005		MOV B, M	Load the first number into B
F006		DCR C	Decrease C
F007	LOOP	INX H	Point to next location
F008		MOV A, M	Get the next number from memory to Acc
F009		CMP B	Compare Acc and B
F00A		JNC SKIP	if B <= A, then skip
F00D		MOV B, A	If CY is 1, update B
F00E	SKIP	DCR C	Decrease C
F00F		JNZ LOOP	When count is not 0, go to LOOP
F012		LXI H,9000H	Point to destination address
F015		MOV M, B	Store the minimum number
F016		HLT	Terminate the program

Precautions:

1. Memory is set to be clear after the end of the program.
2. Memory should be interred carefully otherwise its shows program has some error or it will interrupt the program differently and executed accordingly which cannot be mitigated.
3. After editing the code, the assembly button must be preset before executing.

Experiment 5

Find the sum of a series of numbers

Aim: Find the sum of a series of numbers in 8085 microprocessor.

Objective:

1. To learn and understand the assembly instructions
2. To draw flow chart for addition of a series of a number
3. To develop an 8085 assembly language program to add series of a number

Components Required: 8051 Microprocessor, Jumper wires and Breadboard, Simulator.

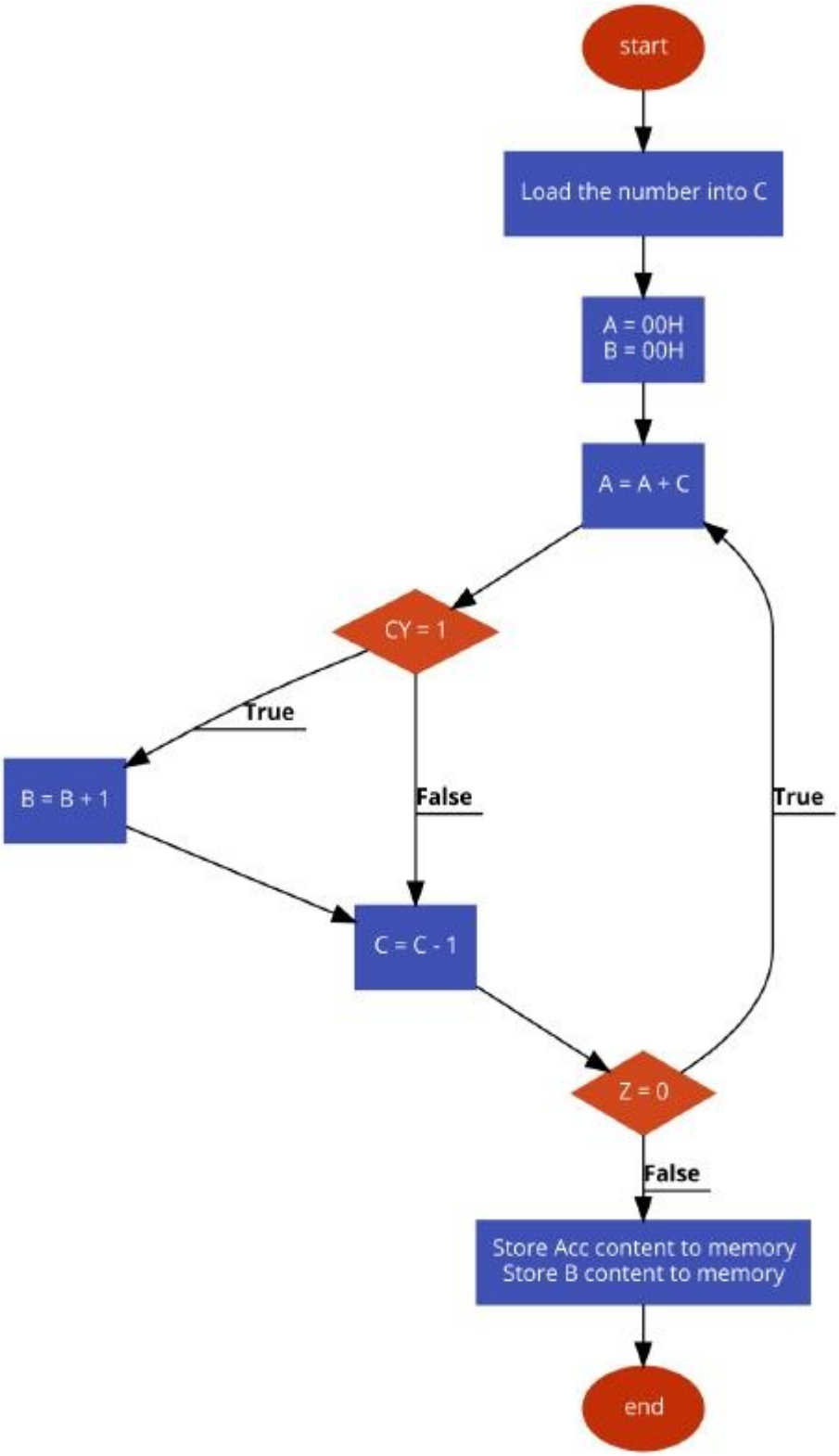
Theory: We are getting the value of N from memory location 8000H. We are using the number N as count variable, in each step we are calculating (A + Count) value, and store them into A. after adding them, the count value is decreased, thus the total series is completed.

If the number is 23H (35D), then the sum will be (35*36)/2 = 630 (276H).

Input:

Address	Data
.	.
.	.
.	.
8000	23
.	.
.	.
.	.

Flow Diagram:



Program:

Address	HEX Codes	Labels	Mnemonics	Comments
F000	21, 00, 80		LXI H,8000H	Point to get the upper limit
F003	4E		MOV C, M	Load the upper limit to C
F004	AF		XRA A	Clear the A register
F005	47		MOV B, A	Also clear B register
F006	81	LOOP	ADD C	Add C with A
F007	D2, 0B, F0		JNC SKIP	If CY = 0,Skip next step
F00A	04		INR B	Increase B ifCY = 1
F00B	0D	SKIP	DCR C	Decrease C by1
F00C	C2, 06, F0		JNZ LOOP	Until Z = 1,go to LOOP
F00F	21, 50, 80		LXI H, 8050H	Point to the destination address
F012	77		MOV M, A	Store the acc content
F013	23		INX H	Point to next location

F014	70		MOV M, B	Store the MSbyte
F015	76		HLT	Terminate the program

Output:

Address	Data
.	.
.	.
.	.
8050	76
8051	02
.	.
.	.
.	.

Precautions:

1. Memory is set to be clear after the end of the program.
2. Memory should be interred carefully otherwise its shows program has some error or it will interrupt the program differently and executed accordingly which cannot be mitigated.
3. After editing the code, the assembly button must be preset before executing.

Conclusion: The program iterates over all the given (n number) and adds repeatedly. Thus, if we conclude that time taken to perform one addition is constant, then time complexity of the program is of O (n).

Experiment 6

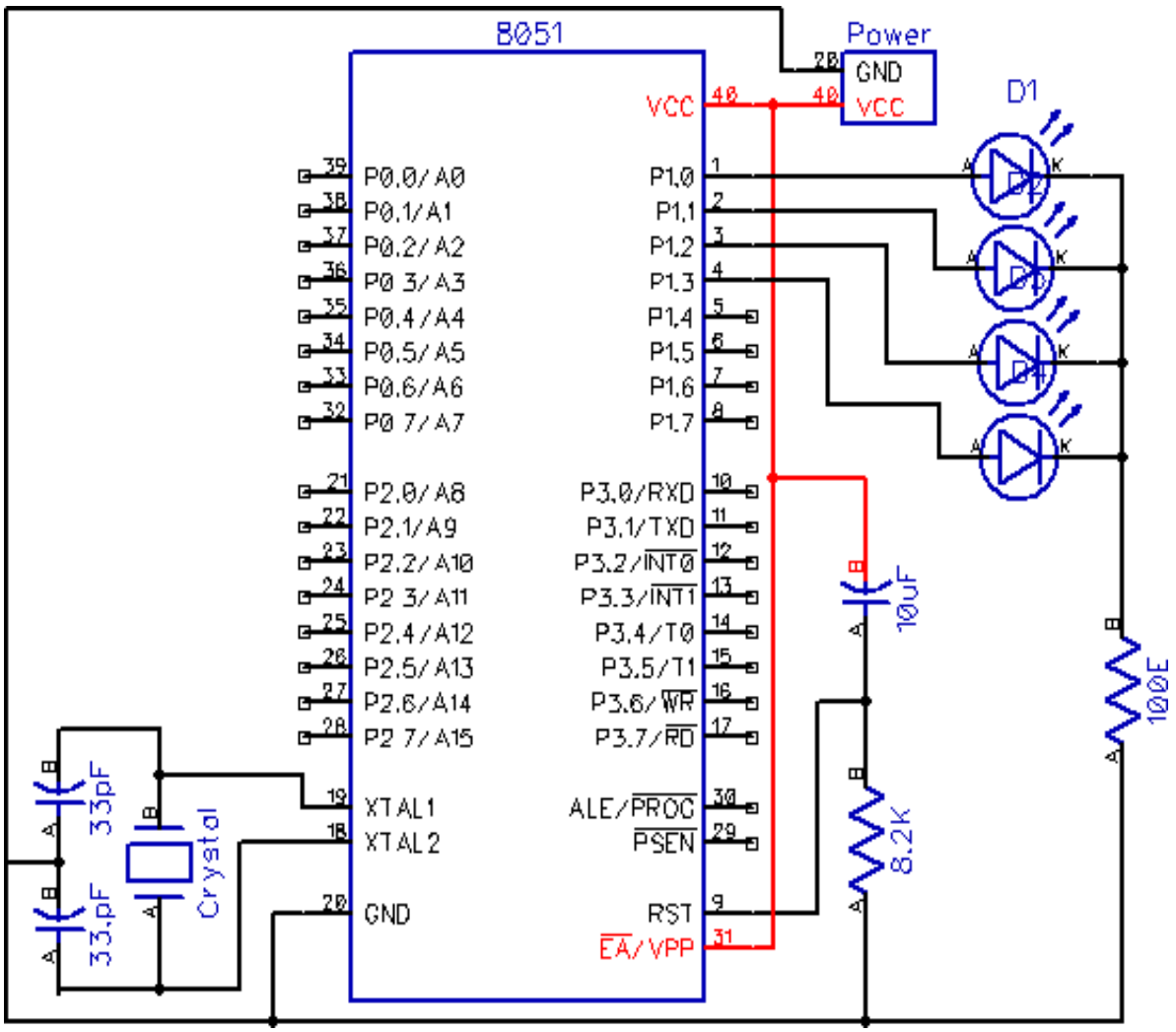
Design and Implementation of 4 Bit Binary Counter Using 8051 Microcontroller

---

Aim: Design an 8051 µC based circuit and its associated program to implement 4 bit binarycounter using LEDs.

Components Required: 8051 Microcontroller, Crystal Oscillator, Capacitor, Resistor,LEDs, and Breadboard.

Circuit Diagram:



Steps of Experiment:

Step 1: Set any port as an output port. Here port 1 has been used and initialized its value as FFH.

Step 2: Provide delay using internal timer or by making use of processing delay during jump operations and decreasing general purpose register (GPR) values. To increase delay nesting may be done.

Step 3: Put this calculated delay values in the GPR (here R0 and R1).

a) Put the calculated delay value in any GPR.
b) Decrease the GPR value.
c) Repeat step 3.b till the GPR value is 00H.

Step 4: Decrement the output port (port 1).

Step 5: Same as Step 3 to provide delay.

Step 6: Jump to Step 4 for continuous decrement.

Program with Hex Codes:

Address	Code	Mnemonics	Comments
0000	75 90 00	MOV P1, #0F	Move 0FH to port 1
0003	78 FF	MOV R0, #FF	Move FFH to GPR R0
0005	79 FF	MOV R1, #FF	Move FFH to GPR R1
0007	D9 FE	DJNZ R1, 0007	Decrease R1 and jump if R1 not zero to location 0007H
0009	D8 FA	DJNZ R0, 0005	Decrease R0 and jump if R0 not zero to location 0005H
000B	15 90	DEC P1	Decrease P1
000D	02 00 03	LJMP 0003	Jump to location 0003H

Experiment 7

Implementation of Up/Down Counter Using Following Input Specifications

(a). Two General Purpose I/O Pins as Control Inputs

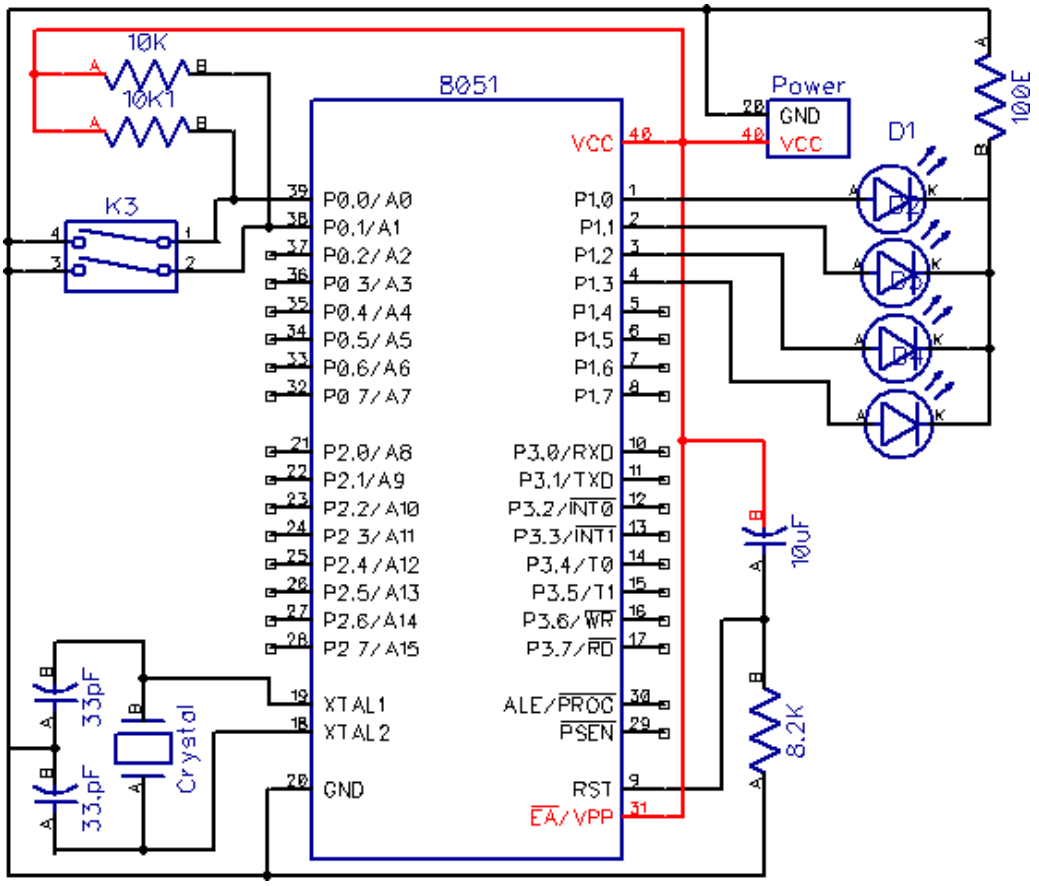
(b). One Interrupt Pin as a Control Pin

Aim: Design an 8051 µC based circuit and its associated program to implement 4 bit binaryup/down counter using LEDs with the stated specifications.

Components Required: 8051 Microcontroller, Crystal Oscillator, Capacitor, Resistor,LEDs, Breadboard, and Switches.

Experiment 7 (a):

Circuit Diagram:



Steps of Experiment:

Step 1: Set P1 to input by putting data FFH and P2 as output by putting data 00H.

Step 2: Apply a nested loop delay using register R0 and R1.

Step 3: Check the content of P1.1 and P1.0 by ANDING P1 with 03H.

(a) If value is P1.1 is not set we jump to Step 5.

(b) Else we continue with Step 4.

Step 4: Decrease the content of output port P2 and jump to step 2.

Step 5: Check the content of P1.0.

a) If value of P1.0 is not set jump to Step 7

b) Else continue with Step 6.

Step 6: Increase content of output port P2 and jump to step 2.

Step 7: If none of the above condition is met we jump to step 2 from here (no change inoutput port P2).

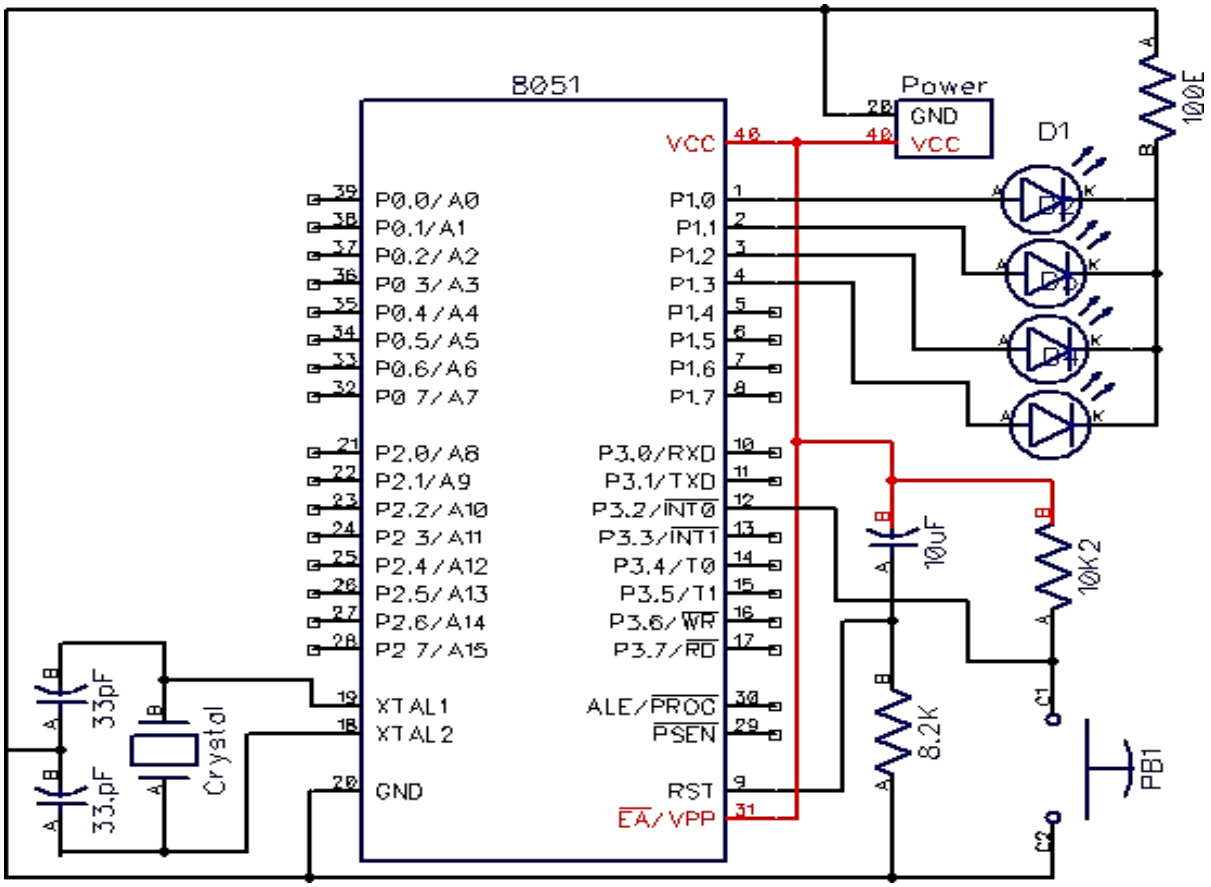
Program with Hex Codes:

Address	Code	Mnemonics	Remarks
0000	75 90 FF	MOV P1,#FF	MOVE FF DATA TO P1
0003	75 A0 00	MOV P2,#00	MOVE 00 DATA TO P2
0006	AA A0	MOV R2,P2	COPY DATA FROM P2 TO R2
0008	79 FF	MOV R1,#FF	STORE DATA FF IN R1
000A	78 FF	MOV R0,#FF	STORE FF DATA IN R0
000C	D8 FE	DJNZ R0,000C	DECREASE R0 AND JUMP TO 000C IF R0 IS NOT ZERO
000E	D9 FA	DJNZ R1,000A	DECREAS R1 AND JUMP TO 000A IF R1 IS NOT ZERO
0010	74 D3	MOV A,#03	MOVE IMMEDIATE DATA 03H TO ACCUMULATOR
0012	52 90	ANL P1,A	AND P1 TO ACCUMULATOR
0014	E5 90	MOV A, P1	`MOVE DATA P1 TO ACCUMULATOR

0016	B4 00 05	CJNE A,#10,001E	COMPARE ACCUMULATOR CONTENT WITH 10H AND JUMP TO ADDRESS 001EIF NOT EQUAL
0019	15 A0	DEC P2	DECREMENT REGISTOR P2
001B	02 00 06	LJMP 0006	JUMP TO ADDRESS 0006
001E	B4 01 05	CJNE A,#01,0026	CAMPARE ACCUMULATOR CONTENT WITH 01H AND JUMP TO ADDRESS 0026IF NOT EQUAL
0021	05 A0	INC P2	INCREMENET REGISTER P2
0023	02 00 06	LJMP 0006	JUMP TO ADDRESS 0006
0026	B4 10 08	CJNE A,#10,0031	CAMPARE ACCUMULATOR CONTENT WITH 10H AND JUMPTO ADDRESS 0031IF NOT EQUAL
0029	7A 00	MOV R2 ,#00	MOVE DATA 00H TO R2
002B	75 A0 00	MOV P2, #00	MOVE DATA 00H TO P2
002E	02 00 06	LJMP 0006	JUMP TO ADDRESS 0006
0031	8A AD	MOV P2, R2	MOVE CONTENT OF R2 IN P2
0033	02 00 06	LJMP 0006	JUMP TO ADDRESS 0006

Experiment 7 (b)

Circuit Diagram:



Steps of Experiment:

Step 1: Set P1 to input by putting data FFH and P2 as output by putting data 00H.

Step 2: Apply a nested loop delay using register R0 and R1.

Step 3: Check the content of P1.1 and P1.0 by ANDING P1 with 03H.

(a) If value is P1.1 is not set we jump to step 5.

(b) Else we continue with step 4.

Step 4: Decrease the content of output port P2 and jump to step 2.

Step 5: Check the content of P1.0.

(a) If value of P1.0 is not set jump to step 7

(b) Else continue with step 6.

Step 6: Increase content of output port P2 and jump to step 2.

Step 7: If none of the above condition is met we jump to step 2 from here (no change inoutput port P2).

Step 8: In case if there is an interrupt the pointer goes to 0000H. The accumulator is used analternator.

a) Accumulator is complemented.

b) If accumulator is not equal to zero then jump to 0068H

c) Else, push 0065H to stack.

d) Return from interrupt. Stack pops up new address becomes 0065H.

e) At 0065H, jump to 0065H again. Infinite loop.

f) At 0068H POP the stack it now has the location where interrupt was pressed.Return from interrupt.

Program with Hex Codes:

Address	Code	Mnemonics	Remarks
0000	02 00 50	LJMP 0050	JUMP TO 0050
0003	74 81	MOV A,#81	MOVE 81 TO ACCUMULATOR
00 05	F5 A8	MOV IE,A	MOVE A TO INTERRUPT ENABLE
00 07	74 01	MOV A, #01	MOVE 01 TO A
00 09	F5 B8	MOV IP, A	MOVE A TO INTERRUPT PRIORITY
000B	74 FF	MOV A, #FF	MOV FF TO A
0010	75 90 FF	MOV P1,#FF	MOVE FF DATA TO P1
0013	75 A0 00	MOV P2,#00	MOVE 00 DATA TO P2
0016	AA A0	MOV R2,P2	COPY DATA FROM P2 TO R2
0018	79 FF	MOV R1,#FF	STORE DATA FF IN R1
001A	78 FF	MOV R0,#FF	STORE FF DATA IN R0
001C	D8 FE	DJNZ R0,001C	DECREASE R0 AND JUMP TO 000C IF R0 IS NOT ZERO
001E	D9 FA	DJNZ R1,001A	DECREAS R1 AND JUMP TO 000A IF R1 IS NOT ZERO

0020	74 D3	MOV A,#03	MOVE IMMEDIATE DATA 03H TO ACCUMULATOR
0022	52 90	ANL P1,A	AND P1 TO ACCUMULATOR
0024	E5 90	MOV A, P1	`MOVE DATA P1 TO ACCUMULATOR
0026	B4 00 05	CJNE A,#10, 002E	COMPARE ACCUMULATOR CONTENT WITH 10H AND JUMP TO ADDRESS 001EIF NOT EQUAL
0029	15 A0	DEC P2	DECREMENT REGISTOR P2
002B	02 00 06	LJMP 0006	JUMP TO ADDRESS 0006
002E	B4 01 05	CJNE A,#01, 0036	CAMPARE ACCUMULATOR CONTENT WITH 01H AND JUMP TO ADDRESS 0026IF NOT EQUAL
0031	05 A0	INC P2	INCREMENET REGISTER P2
0033	02 00 06	LJMP 0016	JUMP TO ADDRESS 0006
0036	B4 10 08	CJNE A,#10,0041	CAMPARE ACCUMULATOR CONTENT WITH 10H AND JUMPTO ADDRESS 0031IF NOT EQUAL
0039	7A 00	MOV R2 ,#00	MOVE DATA 00H TO R2
003B	75 A0 00	MOV P2, #00	MOVE DATA 00H TO P2
003E	02 00 06	LJMP 0016	JUMP TO ADDRESS 0006
0041	8A AD	MOV P2, R2	MOVE CONTENT OF R2 IN P2
0043	02 00 06	LJMP 0006	JUMP TO ADDRESS 0006
0050	F4	CPL A	COMPLEMENT A
0051	B4 00 0C	CJNE A,#00,0068	COMPARE A TO 00 AND JUMP TO 0060 IF NOT EQUAL
0054	C0 00	PUSH 00	PUSH 00 INTO STACK POINTER
0056	C0 65	PUSH 65	PUSH 30 INTO STACK POINTER
0058	32	RETI	RETURN FROM INTERRUPT
0065	00 02 65	LJMP 0065	JUMP INFINITELY
0068	D0 81	POP SP	POP ADDRESS TO STACK POINTER
006A	D0 81	POP SP	POP ADDRESS TO STACK POINTER
006C	32	RETI	RETURN FROM INTERRUPT

Experiment 8

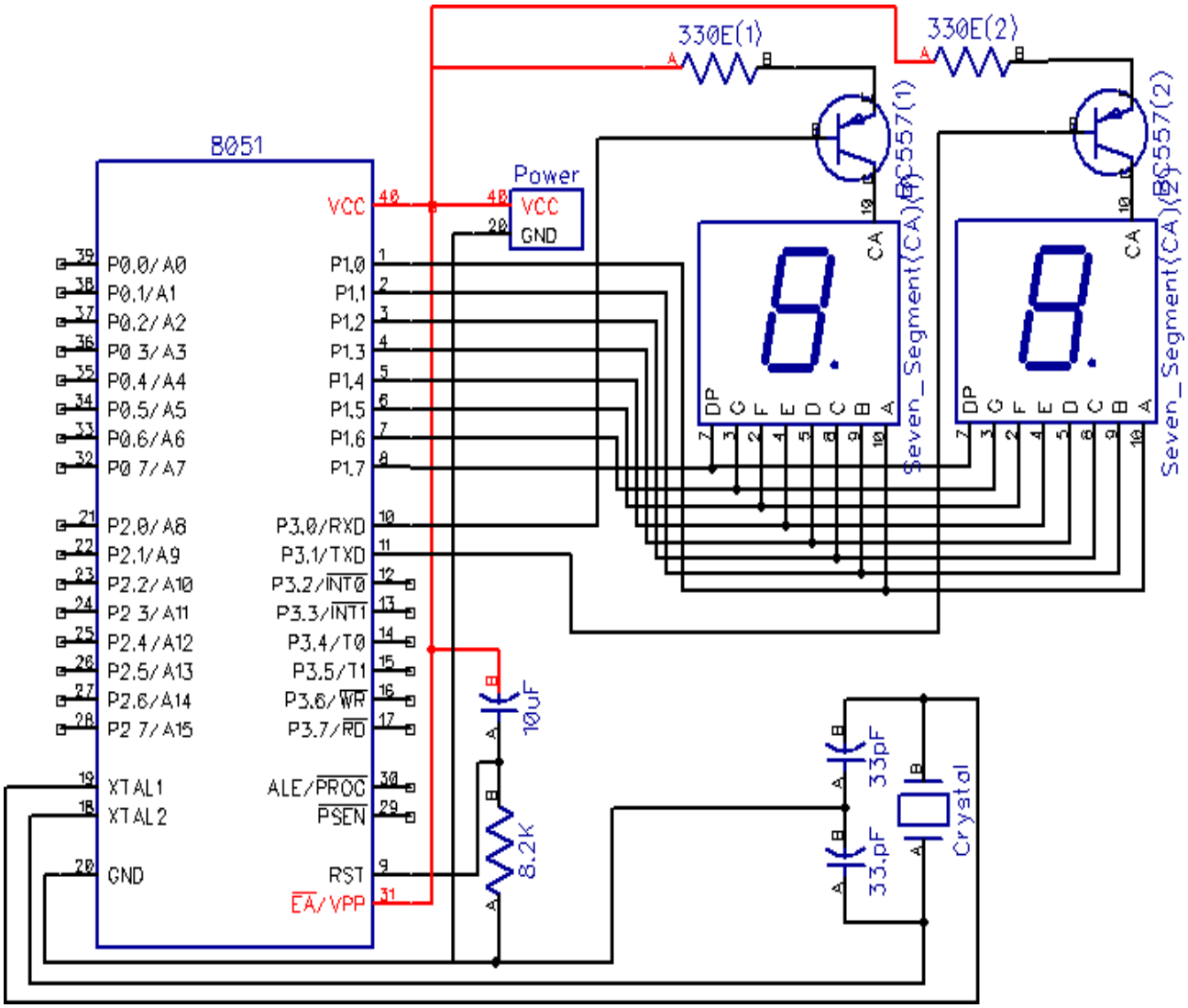
Interfacing Two Seven Segment Displays (in Multiplexing Modes) with an 8051 microcontroller

---

Aim: Design a 8051 µC based circuit and its associated program to interface two sevensegment displays.

Components Required: 8051 Microcontroller, Crystal Oscillator, Capacitor, Resistor,LEDs, and Breadboard, and two Seven Segment Displays.

Circuit Diagram:



Steps of Experiment:

Step 1: Save lookup table at location 0500 and initialize it at beginning of code here.
Step 2: Set value of Register R1
to 00H**Step 3:** Set R0 to 32H (60
in decimal) **Step 4:** Take units
place of R1 to A.
Step 5: Using this value find corresponding value in lookup table for that value
and save it inACC.
Step 6: Set P2 to print ACC value to 1st Seven Segment display (P2.0 = 0 and P2.1 = 1).
Step 7: Apply delay for some ms.
Step 8: Again take R1 value to A, take 10‟s places to A, and use this value to
savecorresponding value in lookup table in ACC.
Step9: Set P2 to print ACC value in 2nd Seven Segment display (P2.0 = 1 and P2.1 = 0).
Step 10: Apply 1 sec delay.
Step 11: Increment value of R1 and Put Decimally adjusted value in A and
then this value inR1.
Step 12: Compare R1 to 60, and jump to Step3 if not equal.
Step 13: If R is 60, Jump to Step 2.

Program with Hex Codes:

Address	Mnemonics	Remarks
0000	MOB DPTR, #0500	IN 0500 STARTING ADDRESS THE CODES FOR DISPLAY ARE ENTERED
0003	MOV R1,#00	
0005	MOV R0,#32	LOOP FOR ALTERNATE SENDING FOR ONE NUMBER
0007	MOV A,R1	

0008	ANL A,#0F	FOR KNOWING DIGIT AT UNITS PLACE
000A	MOVC A,A+DPTR	
000B	CLR P2.0	
000D	SETB P2.1	
0011	MOV R2,#FF	DELAY FOR SWITCHING PLACES
0013	DJNZ R2,0013	
0015	MOV A,R1	
0016	ANL A,#F0	FOR KNOWING DIGIT AT TENS PLACE
0018	SWAP A	
0019	MOVC A,@A+DPTR	
001A	SETB P2.0	
001C	CLR P2.1	
001E	MOV P1,A	
0020	MOV R2,#FF	
0022	DJNZ R2,0022	DELAY FOR 1SEC
0024	DJNZ R0,0007	
0026	INC R1	
0027	MOV A,R1	
0028	DA A	DECIMALLY ADJUSTING IT
0029	MOV R1,A	
002A	CJNE R1,#60,0005	TO STOP AFTER 59 AND START FROM 0
002B	LJMP 0003	

Experiment 9

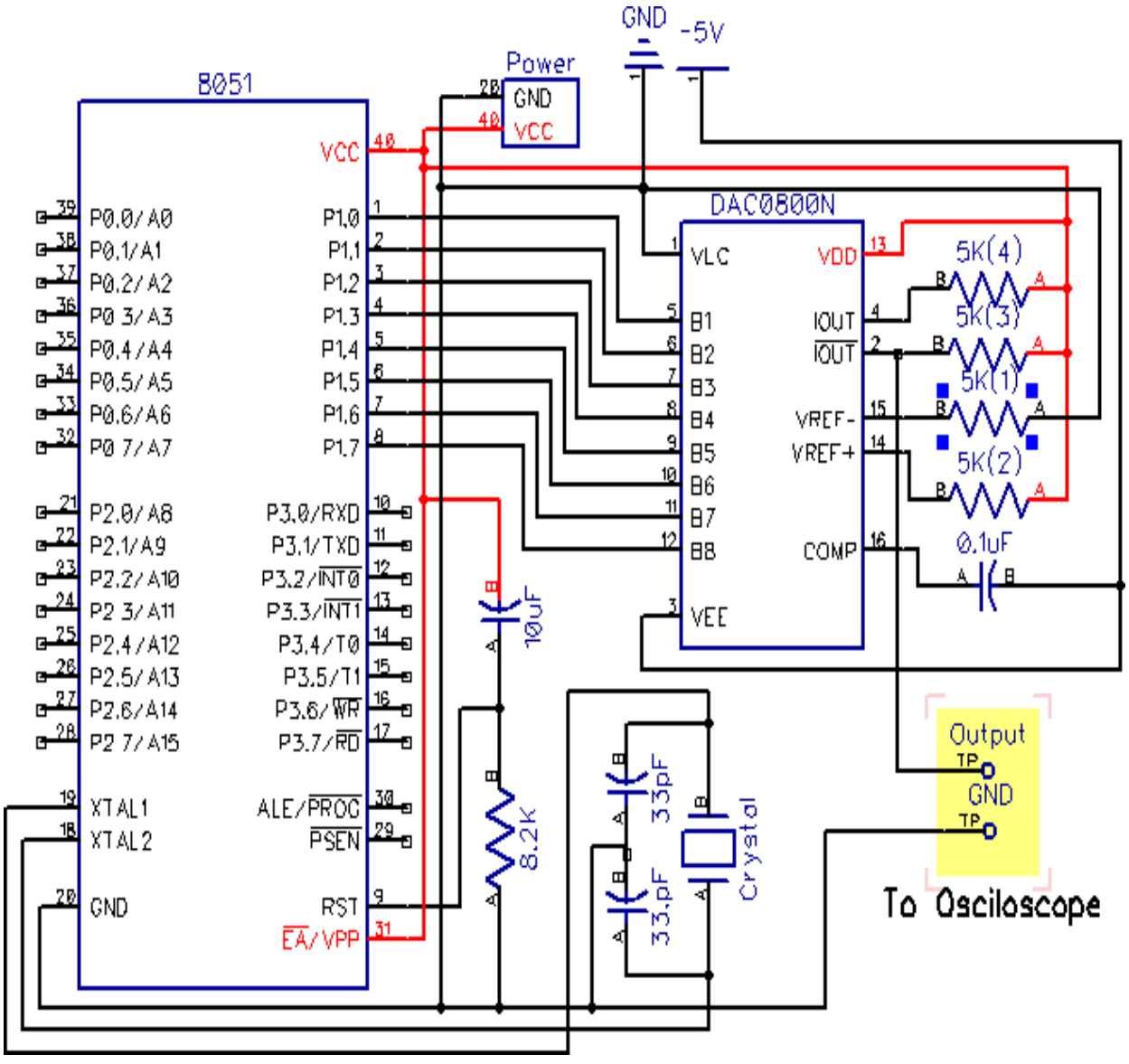
Interface a DAC with 8051 Microcontroller and Generate a RampOutput Through It.

Aim: Design a circuit associated program interface a DAC with 8051 microcontroller andgenerate a ramp output through it.

Components Required: 8051 Microcontroller, Crystal Oscillator**,** Capacitor**,** Resistor**,**LEDs**,** and Breadboard, and DAC

.

Circuit Diagram:



Steps of Experiment:

Step 1: Set P1 as output port by putting data 00h to P1.

Step 2: Apply delay using nested loops using register R0 and R1.

Step 3: Increase the value of output port P1 and jump to step 2.

Ramp output is obtained at DAC output port and with value proportional to input digitalvalue from micro-controller.

Program with Hex Codes:

Address	Code	Mnemonics	Remarks
0000	75 90 00	MOV P1,#00	STORE DATA 00H TO OUTPUT PORT P1
0003	78 FF	MOV R1,#FF	STORE DATA FFH TO REGISTER R1
0005	79 FF	MOVE R0,#FF	STORE DATA FFH TO REGISTER R0
0007	D9 88	DJNZ R0, 0007	DECREASE THE CONTENT OF REGISTERR0 AND JUMP TO ADDRESS 0007 IF NOT ZERO
0009	D8 FA	DJNZ R1, 0005	DECREASE THE CONTENT OF REGISTER R1 AND JUMP TO ADDRESS 0005 IF NOTZERO
000B		INC P1	INCRESE THE CONTENT OF OUTPUT PORT P1
000D		LJMP 0003	JUMP TO ADDRESS 0003

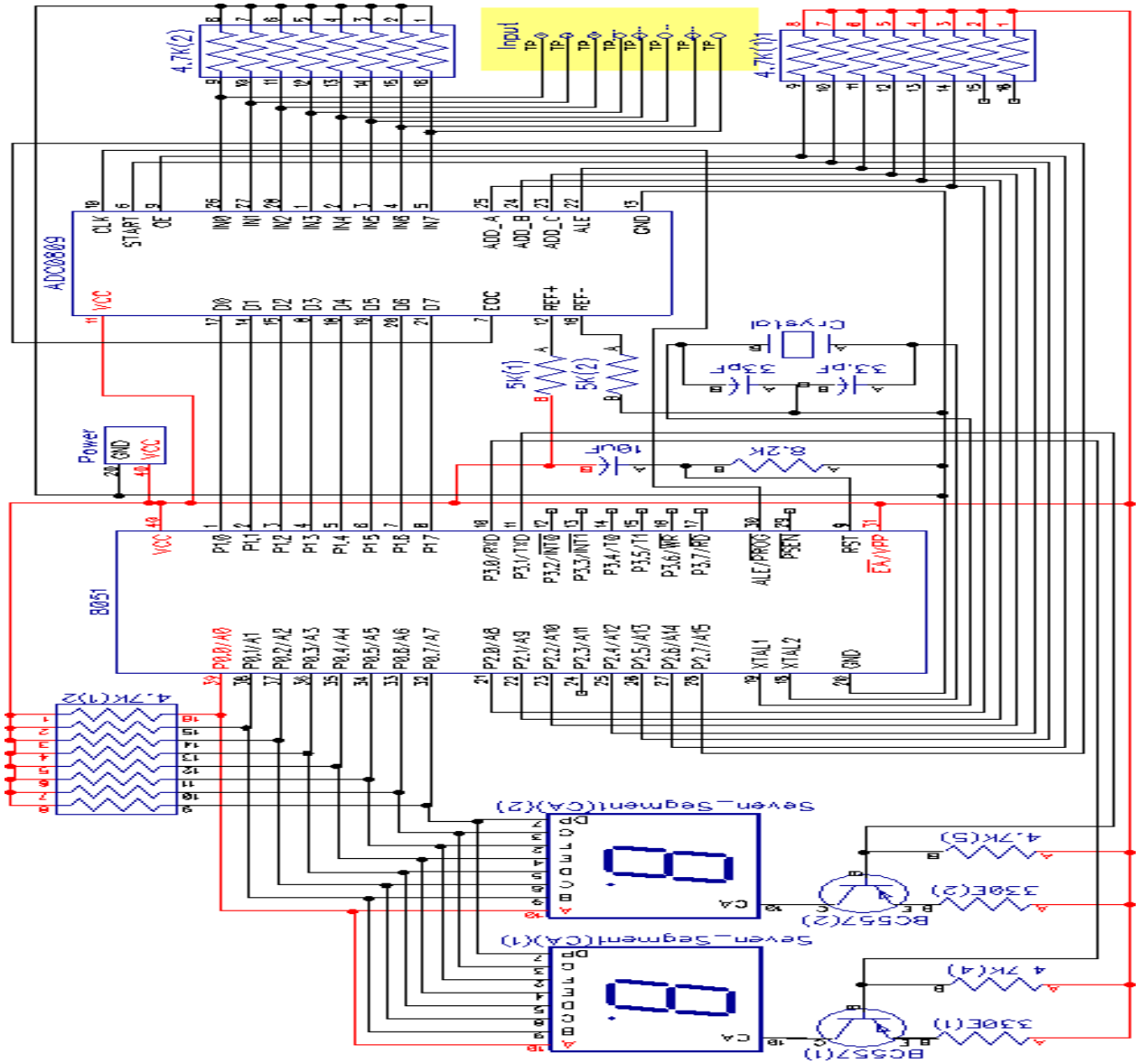
Experiment 10

Interface an ADC with 8051 Microcontroller and Show the Corresponding Output in Two Seven Segment Displays.

Aim: Design a circuit associated program interface an ADC with 8051 microcontroller andobtain the corresponding output in two seven segment displays.

Components Required: 8051 Microcontroller, Crystal Oscillator**,** Capacitor**,** Resistor**,**Breadboard, ADC, and Seven Segment Displays.

Circuit Diagram:



Steps of Experiment:

Step 1: Set +/- V_{ref}.

Step 2: Select Input signal by sending select lines in ADC pins A, B, C.

Step 3: Send Start of conversion signal to ADC 0809.

Step 4: When EOC signal is received, set output enable signal of ADC.

Step 5: Output is received at output port selected say P2.

Step 6: Put P2 value in R1.

Step 7: Save lookup table at location 0500 and initialize it at beginning of code here.

Step 8: Take upper nibble of R1 to A by ANDING it with F0.

Step 9: Swap upper and lower nibble of ACC.

Step 10: Using this value find corresponding value in lookup table for that value and save itin ACC.

Step 11: Set P2 to print ACC value to 1st Seven Segment display (P2.0 = 0 and P2.1 = 1).

Step 12: Apply delay for some ms.

Step 13: Take lower nibble of R1 to A by ANDING it with 0F, and use this value to savecorresponding value in lookup table in ACC.

Step 14: Set P2 to print ACC value in 2nd Seven Segment display (P2.0 = 1 and P2.1 = 0).

Step 15: Apply 0.3 sec delay.

Step 16: Jump to Step.

List of Experiments of Microprocessor and Microcontroller Lab
(Course code: ECC 211)

Sl. No	Name of the Experiment
01.	Introduction of 8085 Microprocessor and Its Trainer kit**.**
02.	Performing assembly language Programming of addition, Subtraction, multiplication, division using 8085 Microprocessors.
03.	To perform sorting of 10 numbers in 8085 microprocessor
04.	To find the factorial of a given number and searching the smallest number in a given array in 8085 microprocessors.
05.	Find the sum of a series of numbers in 8085 microprocessor
06.	Design an 8051 µC based circuit and its associated program to implement 4 bit binarycounter using LEDs
07.	Design an 8051 µC based circuit and its associated program to implement 4 bit binaryup/down counter using LEDs with the stated specifications.
08.	Design a 8051 µC based circuit and its associated program to interface two seven segment displays.
09.	Design a circuit associated program interface a DAC with 8051 microcontroller and generate a ramp output through it.
10.	Interface an ADC with 8051 Microcontroller and Showthe Corresponding Output in Two Seven Segment Displays

.