Varuvan Vadivelan Institute of Technology

Dharmapuri – 636 703

LAB MANUAL

Regulation : **2013**

Branch : ***B.E.*** **- EEE**

Year & Semester : **III Year / VI Semester**

EE6612-MICROPROCESSOR AND MICROCONTROLLER LAB

ANNA UNIVERSITY: CHENNAI

REGULATION - 2013

SYLLABUS

EE6612 MICROPROCESSOR AND MICROCONTROLLER LABORATORY

OBJECTIVES:

To provide training on programming of microprocessors and microcontrollers and understand the interface requirements.

LIST OF EXPERIMENTS:

1. Simple arithmetic operations: addition / subtraction / multiplication / division.
2. Programming with control instructions:
   - (i) Ascending / Descending order, Maximum / Minimum of numbers
   - (ii) Programs using Rotate instructions
   - (iii)Hex / ASCII / BCD code conversions.
3. Interface Experiments: with 8085
   - (i) A/D Interfacing. & D/A Interfacing.
4. Traffic light controller.
5. I/O Port / Serial communication
6. Programming Practices with Simulators/Emulators/open source
7. Read a key, interface display
8. Demonstration of basic instructions with 8051 Micro controller execution, including:
   - (i) Conditional jumps, looping
   - (ii) Calling subroutines.
9. Programming I/O Port 8051
   - (i) study on interface with A/D & D/A
   - (ii) study on interface with DC & AC motor .
10. Mini project development with processors.

TOTAL: 45 PERIODS

INDEX

EX.NO	DATE	NAME OF THE EXPERIMENT	STAFF SIGN	REMARKS
13		PROGRAM FOR INTERFACING OF ADC WITH 8085		
14		PROGRAM FOR INTERFACING OF DAC WITH 8085		
15		PROGRAM FOR INTERFACING OF SERIAL PORT COMMUNICATION WITH 8085		
16		PROGRAM FOR INTERFACING OF KEYBOARD AND DISPLAY WITH 8085		
17		PROGRAM FOR INTERFACING OF TRAFFIC LIGHT CONTROLLER WITH 8085		
18		PROGRAM FOR 8 BIT ADDITION USING 8051		
19		PROGRAM FOR 8 BIT SUBTRACTION USING 8051		
20		PROGRAM FOR 8 BIT MULTIPLICATION USING 8051		
21		PROGRAM FOR 8 BIT DIVISION USING 8051		
22		PROGRAM FOR INTERFACING OF ADC WITH 8051		
23		PROGRAM FOR INTERFACING OF DAC WITH 8051		
24		PROGRAM FOR INTERFACING OF STEPPER MOTOR WITH 8051		

Ex No : 1

Date :

PROGRAM FOR 8 BIT ADDITION USING 8085

AIM:

To write an assembly language program for addition of two 8 bit data using 8085 microprocessor.

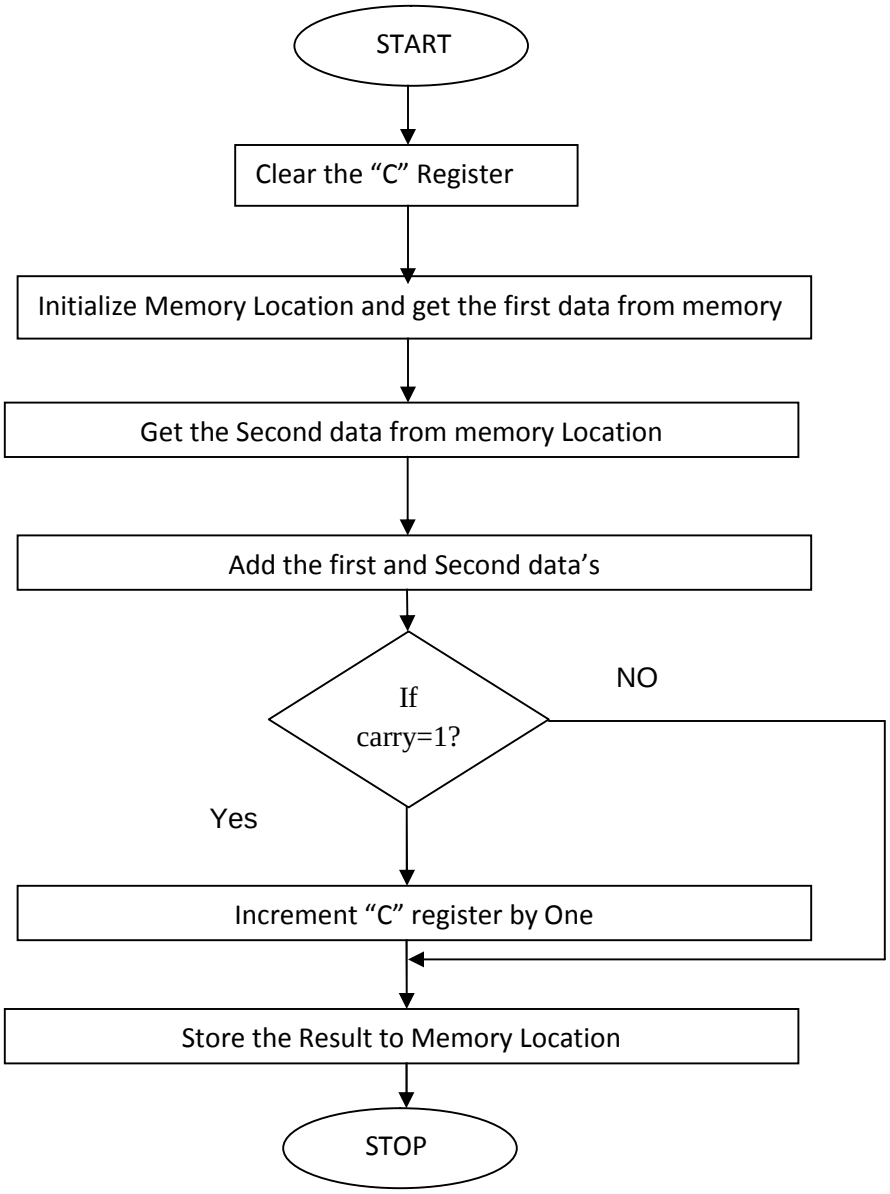
APPARATUS REQUIRED:

Sl.No	Name of the Apparatus	Qty
1	8085 Microprocessor kit	1
2	+5Volts Power Supply	1
3	Keyboard Connector	1

ALGORITHM:

1. Clear the register **C**
2. Initialize the memory pointer to data location.
3. Get the first Data from memory Location and move to register **A.**
4. Get the second data from memory location.
5. Add first and second data.
6. If carry the increment the register **C** by one.
7. Store the result to memory location.

FLOW CHART: (8 bit Addition)



PROGRAM:

ADDRESS	LABEL	MNEMONICS	OPCODE	COMMENTS
8000		*MVI C,00*		Clear the "C" Register
		LXI H,9200		Initialize the memory pointer
		MOV A,M		Move the First Data to A Register
		INX H		Increment the memory pointer
		ADD M		Add First and Second Data
		JNC LOOP1		Jump if No Carry to loop1
		INR C		Increment the "C" register by one
	Loop1	*STA 9500*		Store the Result
		MOV A,C		Move the Carry result to Register "A"
		STA 9501		Store the Carry Result
		HLT		Stop the program

Input:

Memory location	Data
9200	
9201	

Output:

Memory location	Data
9500	
9501	

Result :

Thus the addition of two 8 bit data's was executed using the 8085 microprocessor.

Ex No : 2

Date :

PROGRAM FOR 8 BIT SUBTRACTION USING 8085

AIM:

To write an assembly language program for subtraction of two 8 bit data using 8085 microprocessor.

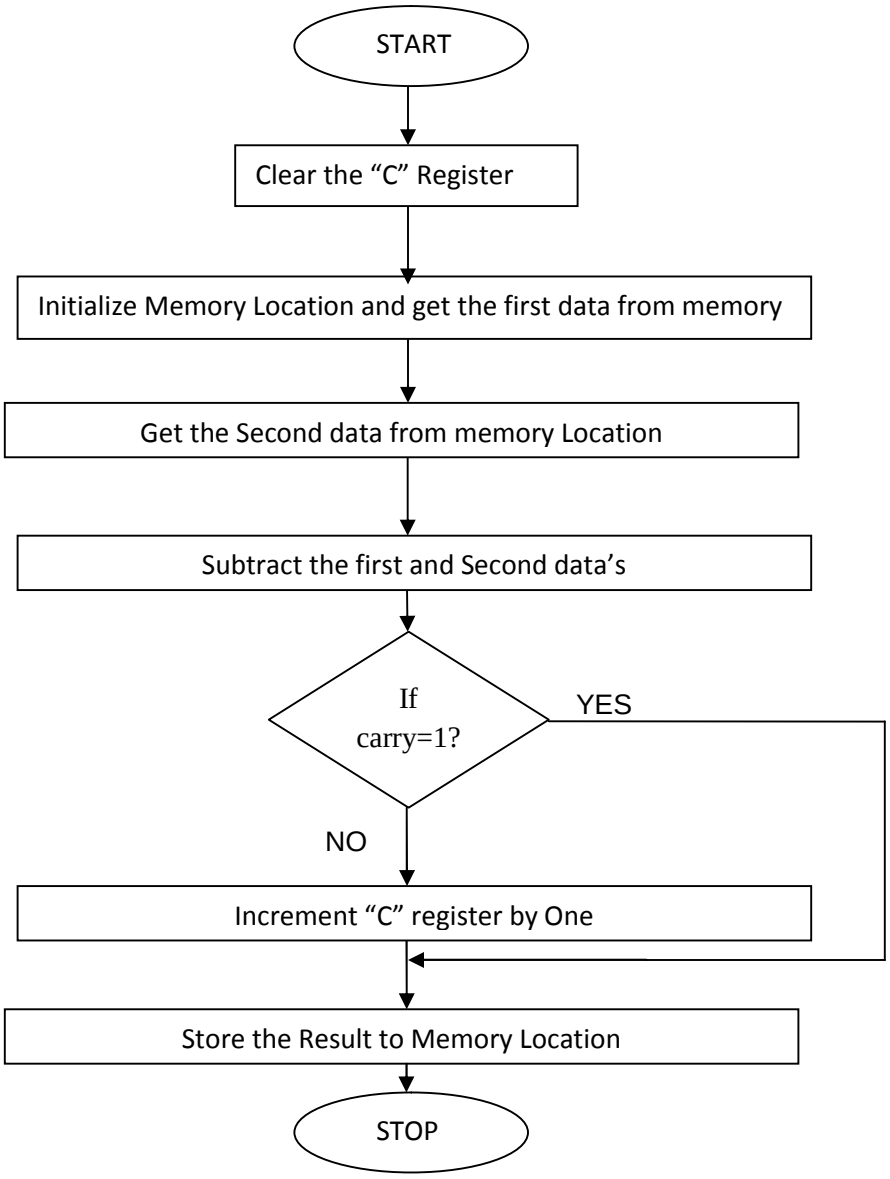
APPARATUS REQUIRED:

Sl.No	Name of the Apparatus	Qty
1	8085 Microprocessor kit	1
2	+5Volts Power Supply	1
3	Keyboard Connector	1

ALGORITHM:

1. Clear the register “C”
2. Initialize the memory pointer to data location.
3. Get the first Data from memory Location and move to register “A”.
4. Get the second data from memory location.
5. Subtract the first and second data.
6. If occur carry the increment the register ‘c’ by one.
7. Store the result to memory location.

FLOW CHART: (8 bit Subtraction)



PROGRAM:

ADDRESS	LABEL	MNEMONICS	OPCODE	COMMENTS
8000		*MVI C,00*		Clear the "C" Register
		LXI H,9200		Initialize the memory pointer
		MOV A,M		Move the First Data to A Register
		INX H		Increment the memory pointer
		SUB M		Subtract First and Second Data
		JNC LOOP1		Jump if No Carry to loop1
		INR C		Increment the **"C"** register by one
		CMA		Complement the Accumulator
		ADI 01		Add by one for two's complement
	Loop1	*STA 9500*		Store the Result
		MOV A,C		Move the Carry result to Register **"A"**
		STA 9501		Store the Carry Result
		HLT		Stop the program

Input:

Memory location	Data
9200	
9201	

Output:

Memory location	Data
9500	
9501	

RESULT :

Thus the subtraction of two numbers was performed using the 8085 microprocessor.

Exp No : 3
Date :

PROGRAM FOR 8 BIT MULTIPLICATION USING 8085

AIM:

To write an assembly language program for multiplication of two 8 bit data using 8085 microprocessor.

APPARATUS REQUIRED:

Sl.No	Name of the Apparatus	Qty
1	8085 Microprocessor kit	1
2	+5Volts Power Supply	1
3	Keyboard Connector	1

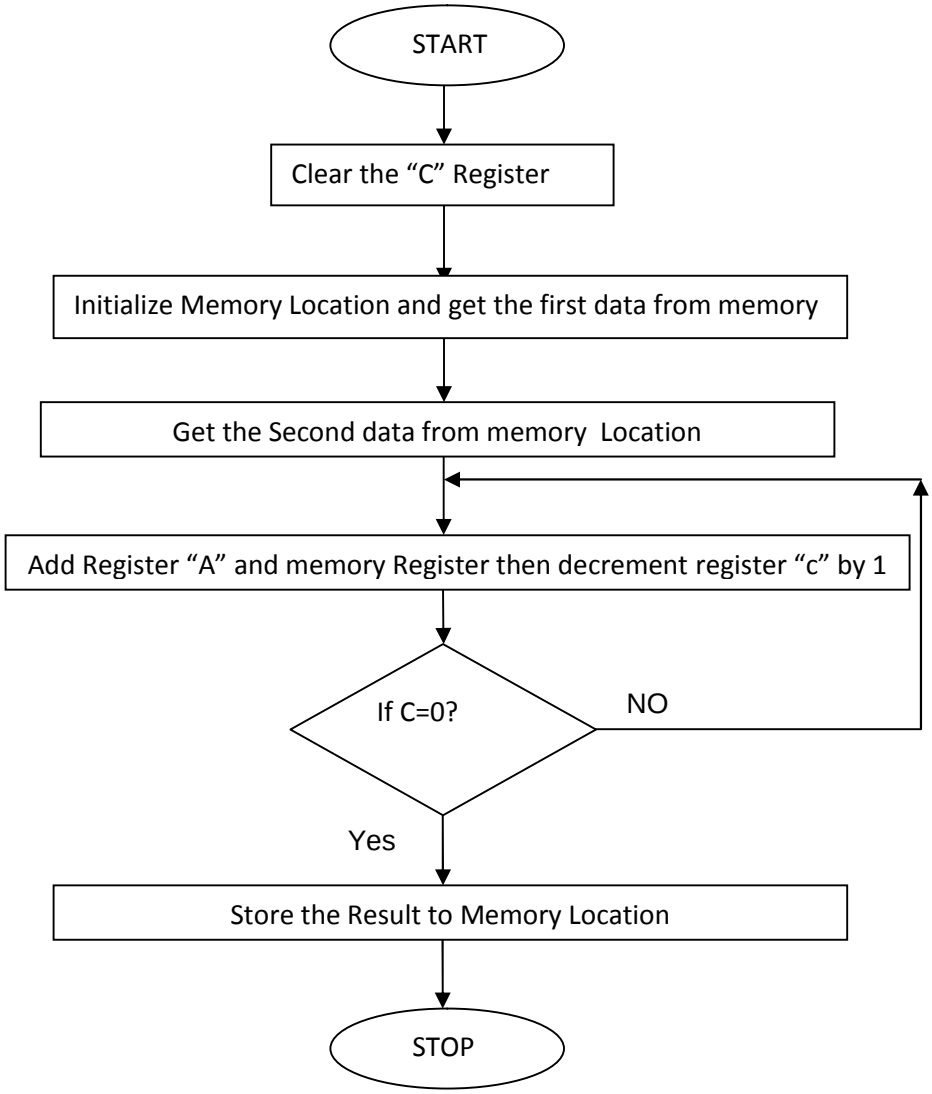
ALGORITHM:

1. Clear the register "A"
2. Initialize the memory pointer to data location.
3. Get the first Data from memory Location and move to register "C".
4. Get the second data from memory location.
5. Add the Memory Data and A Register.
6. Decrement the register 'c' by one
7. Check the register "C' is Zero otherwise repeat step 5.
8. Store the result to memory location.

PROGRAM :

ADDRESS	LABEL	MNEMONICS	OPCODE	COMMENTS
8000		*MVI A,00*		Clear the "A" Register
		LXI H,9200		Initialize the memory pointer
		MOV C,M		Move the First Data to C Reg
		INX H		Increment the memory pointer
	Loop1	*ADD M*		Add First and Second Data
		DCR C		Decrement C register by one
		JNZ LOOP1		Jump if No Zero to loop1
		STA 9500		Store the Result
		HLT		Stop the program

FLOW CHART: (8 bit Multiplication)



Input:

Memory location	Data
9200	
9201	

Output:

Memory location	Data
9500	

RESULT:

Thus the multiplication of two numbers was performed using the 8085 microprocessor.

Exp No : 4
Date :

PROGRAM FOR 8 BIT DIVISION USING 8085

AIM:

To write an assembly language program for division of two 8 bit data using 8085 microprocessor.

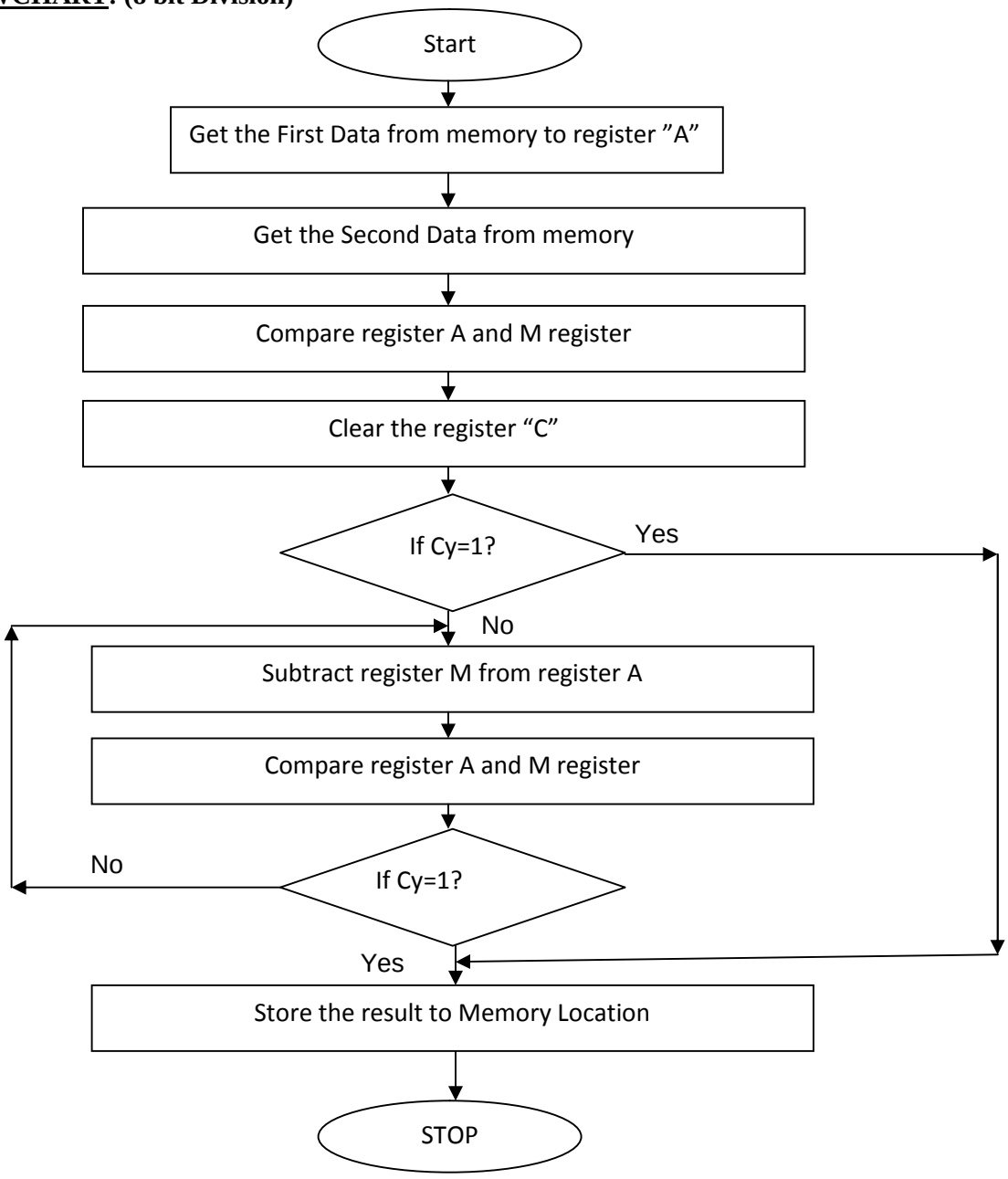
APPARATUS REQUIRED:

Sl.No	Name of the Apparatus	Qty
1	8085 Microprocessor kit	1
2	+5Volts Power Supply	1
3	Keyboard Connector	1

ALGORITHM:

1. Clear the Register “C”.
2. Initialize the Memory Pointer.
3. Get the First Data from memory to Accumulator (Dividend)
4. Get the Second Data from memory (Divisor)
5. Compare Register “A” and “M”
6. If No carry, Subtract Divisor from Dividend [(A)-(M)]
7. Increment Register “C” by one.
8. Compare Register “A” and “M”
9. If No Carry go to step 6
10. Store the result to memory location.

FLOWCHART: (8 bit Division)



PROGRAM:

ADDRESS	LABEL	MNEMONICS	OPCODE	COMMENTS
8000		*MVI C,00*		Clear the "C" Register
		LXI H,9200		Initialize the memory pointer
		MOV A,M		Move the First Data to A Register
		INX H		Increment the memory pointer
		CMP M		Compare register A and M data's
		JC Loop1		If carry jump to loop1
	Loop2	*SUB M*		Subtract register A and M Data
		INR C		Increment C register by one
		CMP M		Compare register A and M data's
		JNC Loop2		Jump if No Carry to loop2
		STA 9500		Store the Quotient Result
		MOV A,C		Move register C to A
		STA 9501		Store the Remainder Result
		HLT		Stop the program
	Loop1	*MVI A,00*		Clear the accumulator
		STA 9500		Store the result
		STA 9501		Store the result
		HLT		stop

Input:

Memory location	Data
9200	
9201	

Output:

Memory location	Data
9500	
9501	

RESULT :

Thus the division of two numbers was performed using the 8085 microprocessor.

Exp No : 5
Date :

LARGEST NUMBER OF PROGRAM IN A GIVEN ARRAY

AIM:

To write an assembly language program to find the largest number in a given array using 8085 microprocessor.

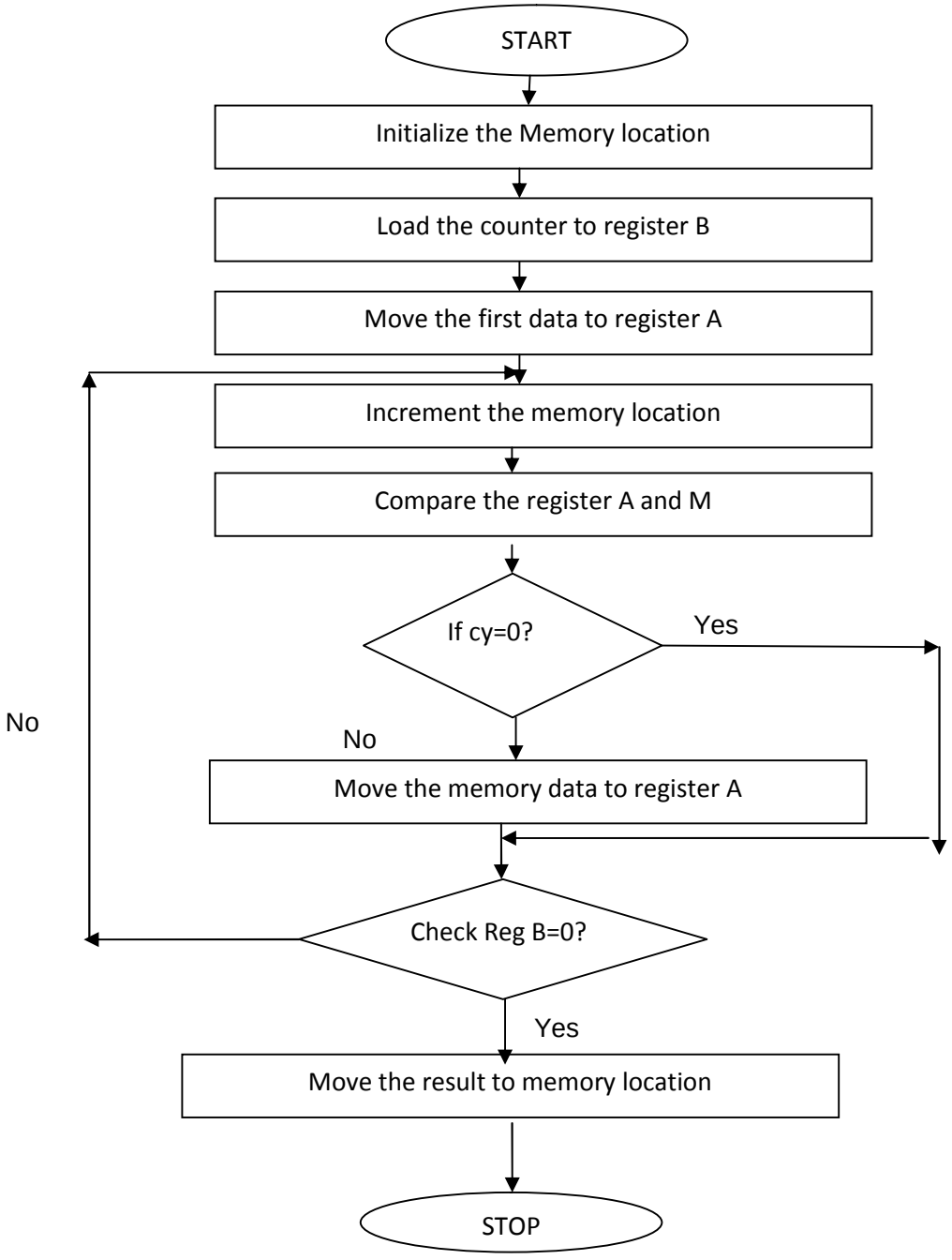
APPARATUS REQUIRED:

Sl.No	Name of the Apparatus	Qty
1	8085 Microprocessor kit	1
2	+5Volts Power Supply	1
3	Keyboard Connector	1

ALGORITHM:

1. Initialize the Memory Pointer.
2. Load the number of array to B register.
3. Move the first data to register A.
4. Increment the Memory pointer.
5. Compare the register A and M .
6. If No carry jump to loop2.
7. Move the register M to A.
8. Decrement the register B by one.
9. Check register B is Zero otherwise go to step 4.
10. Store the Largest value to memory location.
11. Stop.

FLOWCHART: (**LARGEST NUMBER)**



PROGRAM:

ADDRESS	LABEL	MNEMONICS	OPCODE	COMMENTS
8000		*LXI H,9100*		Initialize the memory pointer
		MVI B,04		Load the counter to register B
		MOV A,M		Move the First Data to A Register
	Loop1	*INX H*		Increment the memory pointer
		CMP M		Compare register A and M datas
		JNC Loop2		If No carry jump to loop2
		MOV A, M		Move register M to A
	LOOP2	*DCR B*		Decrement B register by one
		CMP M		Compare register A and M data's
		JNZ Loop1		Jump if No Zero to loop1
		STA 9500		Store the Result to memory
		HLT		Stop

Input:

Memory location	Data
9100	
9101	
9102	
9103	

Output:

Memory location	Data
9500	

RESULT:

Thus the program to find the largest number in a given array using 8085 microprocessor was executed.

Exp No : 6
Date :

SMALLEST NUMBER OF PROGRAM IN A GIVEN ARRAY

AIM:

To write an assembly language program to find the smallest number in a given array using 8085 microprocessor.

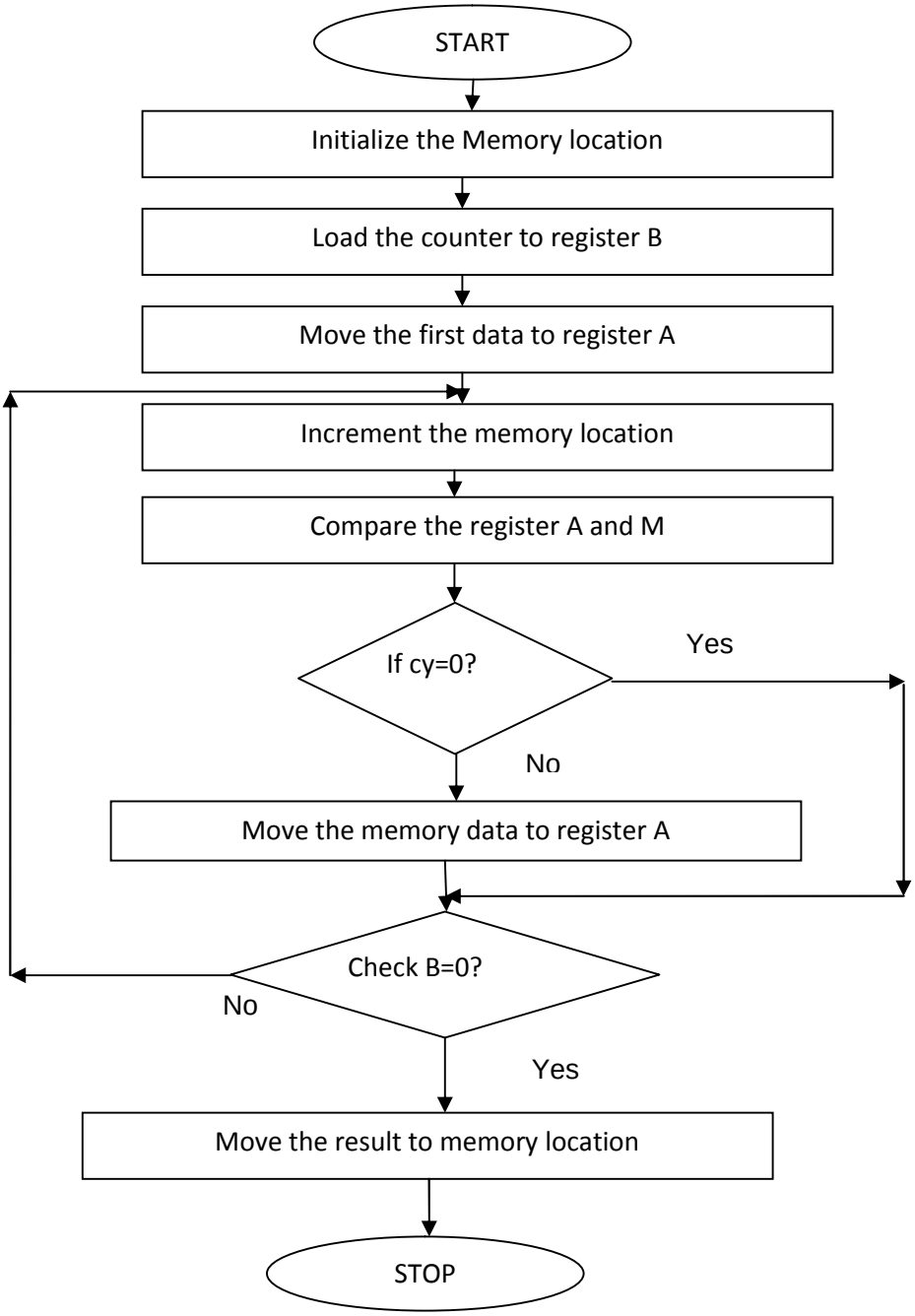
APPARATUS REQUIRED:

Sl.No	Name of the Apparatus	Qty
1	8085 Microprocessor kit	1
2	+5Volts Power Supply	1
3	Keyboard Connector	1

ALGORITHM:

1. Initialize the Memory Pointer.
2. Load the number of array to B register.
3. Move the first data to register A.
4. Increment the Memory pointer.
5. Compare the register A and M .
6. If carry jump to loop2.
7. Move the register M to A.
8. Decrement the register B by one.
9. Check register B is Zero otherwise go to step 4.
10. Store the Largest value to memory location.
11. Stop.

FLOWCHART: (Smallest Number)



PROGRAM:

ADDRESS	LABEL	MNEMONICS	OPCODE	COMMENTS
8000		*LXI H,9100*		Initialize the memory pointer
		MVI B,04		Load the counter to register B
		MOV A,M		Move the First Data to A Register
	Loop1	*INX H*		Increment the memory pointer
		CMP M		Compare register A and M datas
		JC Loop2		If carry jump to loop2
		MOV A, M		Move register M to A
	Loop2	*DCR B*		Decrement B register by one
		CMP M		Compare register A and M data's
		JNZ Loop1		Jump if No Zero to loop1
		STA 9500		Store the Result to memory
		HLT		Stop

Input:

Memory location	Data
9100	
9101	
9102	
9103	

Output:

Memory location	Data
9500	

RESULT:

Thus the program to find the smallest number in a given array using 8085 microprocessor was executed.

Exp No : 7
Date :

PROGRAM SORT ASCENDING OREDR IN A GIVEN ARRAY

AIM:

To write an assembly language program to sort ascending order in a given array using 8085 microprocessor.

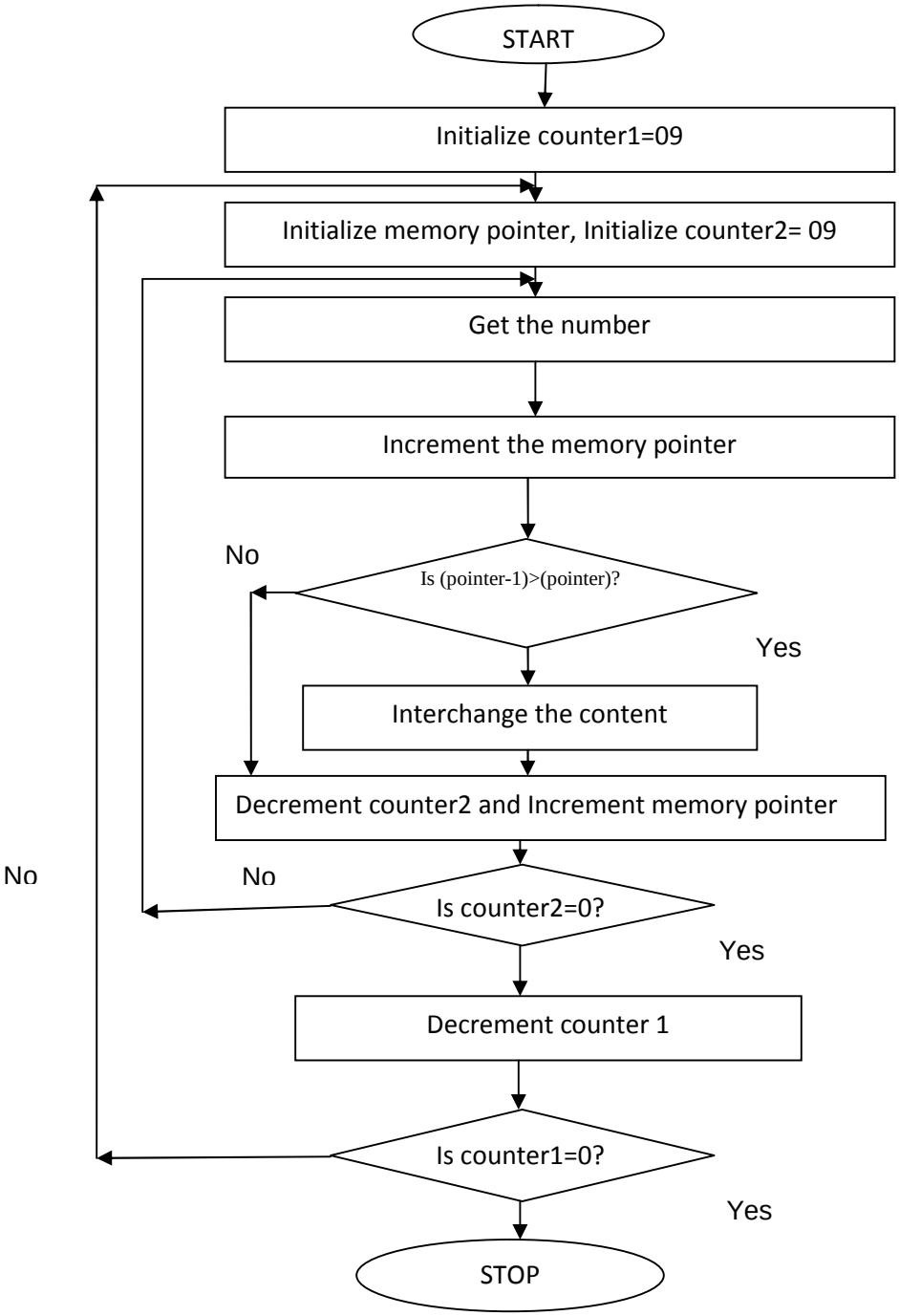
APPARATUS REQUIRED:

Sl.No	Name of the Apparatus	Qty
1	8085 Microprocessor kit	1
2	+5Volts Power Supply	1
3	Keyboard Connector	1

ALGORITHM:

1. Initialize the Memory Pointer.
2. Load the number of array to B register.
3. Move the first data to register A.
4. Increment the Memory pointer.
5. Compare the register A and M .
6. If carry jump to loop2.
7. Move the register M to A.
8. Decrement the register B by one.
9. Check register B is Zero otherwise go to step 4.
10. Store the Largest value to memory location.
11. Stop.

FOWCHART: (Ascending Order)



PROGRAM:

ADDRESS	LABEL	MNEMONICS	OPCODE	COMMENTS
8000		*MVI B,09*		Load the counter to register B
	L3	*LXI H,9100*		Initialize the memory pointer
		MVI C,09		Load the counter value
	L2	*MOV A,M*		Move the First Data to A Register
		INX H		Increment the memory pointer
		CMP M		Compare register A and M data's
		JC L1		If carry jump to loop1
		MOV D, M		Move register M to D
		MOV M,A		Move register M to A
		DCX H		Decrement memory pointer
		MOV M,D		Move register M to D
		INX H		Increment memory pointer
	L1	*DCR C*		Decrement B register by one
		JNZ L2		Jump if No Zero to loop2
		DCR B		Decrement B register by one
		JNZ L3		Jump if No Zero to loop3
		HLT		Stop

Input:

Memory location	Data
9100	
9101	
9102	
9103	
9104	
9105	
9106	
9107	
9108	
9109	

Output:

Memory location	Data
9100	
9101	
9102	
9103	
9104	
9105	
9106	
9107	
9108	
9109	

RESULT:

Thus the program of sort the ascending order in given array was executed by using 8085.

Exp No : 8
Date :

PROGRAM SORT DESCENDING OREDR IN A GIVEN ARRAY

AIM:

To write an assembly language program to sort descending order in a given array using 8085 microprocessor.

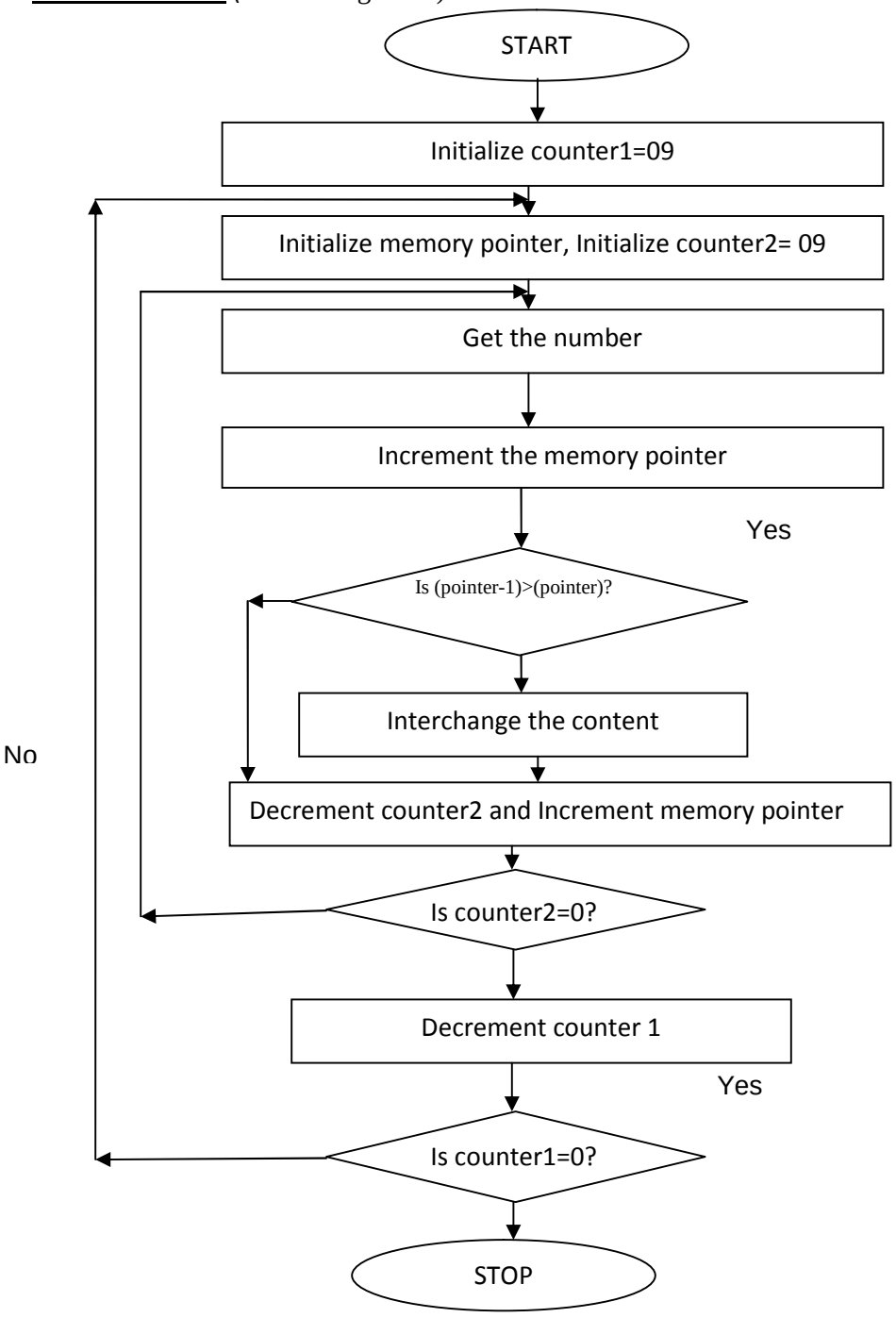
APPARATUS REQUIRED:

Sl.No	Name of the Apparatus	Qty
1	8085 Microprocessor kit	1
2	+5Volts Power Supply	1
3	Keyboard Connector	1

ALGORITHM:

1. Initialize the Memory Pointer.
2. Load the number of array to B register.
3. Move the first data to register A.
4. Increment the Memory pointer.
5. Compare the register A and M .
6. If no carry jump to loop2.
7. Move the register M to A.
8. Decrement the register B by one.
9. Check register B is Zero otherwise go to step 4.
10. Store the Largest value to memory location.
11. Stop.

FLOWCHART: (Descending order)



PROGRAM:

ADDRESS	LABEL	MNEMONICS	OPCODE	COMMENTS
8000		*MVI B,04*		Load the counter to register B
	L3	*LXI H,9100*		Initialize the memory pointer
		MVI C,04		Load the counter value
	L2	*MOV A,M*		Move the First Data to A Register
		INX H		Increment the memory pointer
		CMP M		Compare register A and M datas
		JNC L1		If carry jump to loop1
		MOV D, M		Move register M to D
		MOV M,A		Move register M to A
		DCX H		Decrement memory pointer
		MOV M,D		Move register M to D
		INX H		Increment memory pointer
	L1	*DCR C*		Decrement B register by one
		JNZ L2		Jump if No Zero to loop2
		DCR B		Decrement B register by one
		JNZ L3		Jump if No Zero to loop3
		HLT		Stop

Input:

Memory location	Data
9100	
9101	
9102	
9103	
9104	
9105	
9106	
9107	
9108	
9109	

Output:

Memory location	Data
9100	
9101	
9102	
9103	
9104	
9105	
9106	
9107	
9108	
9109	

RESULT:

Thus the program of sort the descending order in given array was executed by using 8085.

EX.NO:9

Date :

PROGRAM FOR CODE CONVERSION – DECIMAL TO HEX

AIM:

To write an assembly language program to convert a given decimal number into hexadecimal number using 8085 microprocessor.

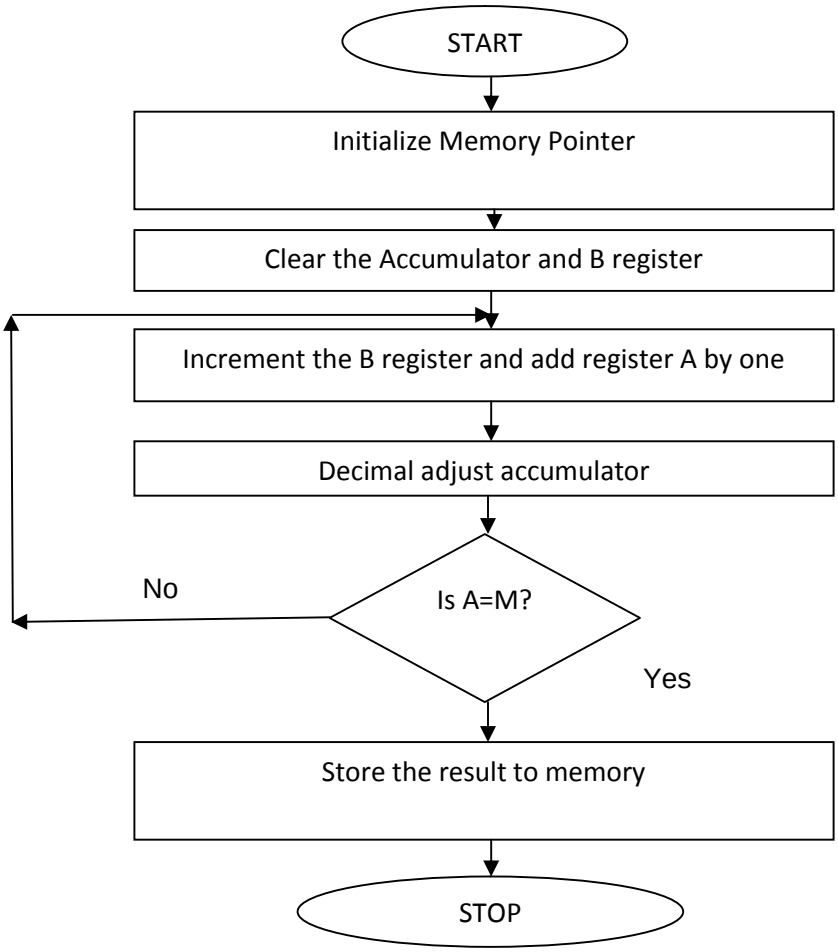
APPARATUS REQUIRED:

Sl.No	Name of the Apparatus	Qty
1	8085 Microprocessor kit	1
2	+5Volts Power Supply	1
3	Keyboard Connector	1

ALGORITHM:

1. Initialize the Memory Pointer.
2. Increment B register.
3. Increment accumulator by one and adjust to decimal every time.
4. Compare the given decimal number with accumulator value.
5. When both matches, the equivalent hexadecimal value is in B register.
6. Store the resultant in memory location.

FLOWCHART : (DECIMAL TO HEX)



PROGRAM:

ADDRESS	LABEL	MNEMONICS	OPCODE	COMMENTS
8000		*LXI H,9100*		Initialize the memory pointer
		MVI A,00		Clear the Accumulator
		MVI B,00		Clear the B register
	Loop1	*INR B*		Increment the B register
		ADI 01		Increment the A register
		DAA		Decimal Adjust Accumulator
		CMP M		Compare A and M register
		JNZ Loop1		Jump if No Zero to loop1
		MOV A,B		Move register B to A
		STA 9500		Store the result to memory
		HLT		Stop the program

Input:

Memory location	Data
9100	

Output:

Memory location	Data
9500	

RESULT:

Thus the program of conversion of decimal to hex given data was executed by using 8085.

Exp No:10
Date :

PROGRAM FOR CODE CONVERSION – HEXA DECIMAL TO DECIMAL

AIM:

To write an assembly language program to convert a given hexadecimal number into decimal number using 8085 microprocessor.

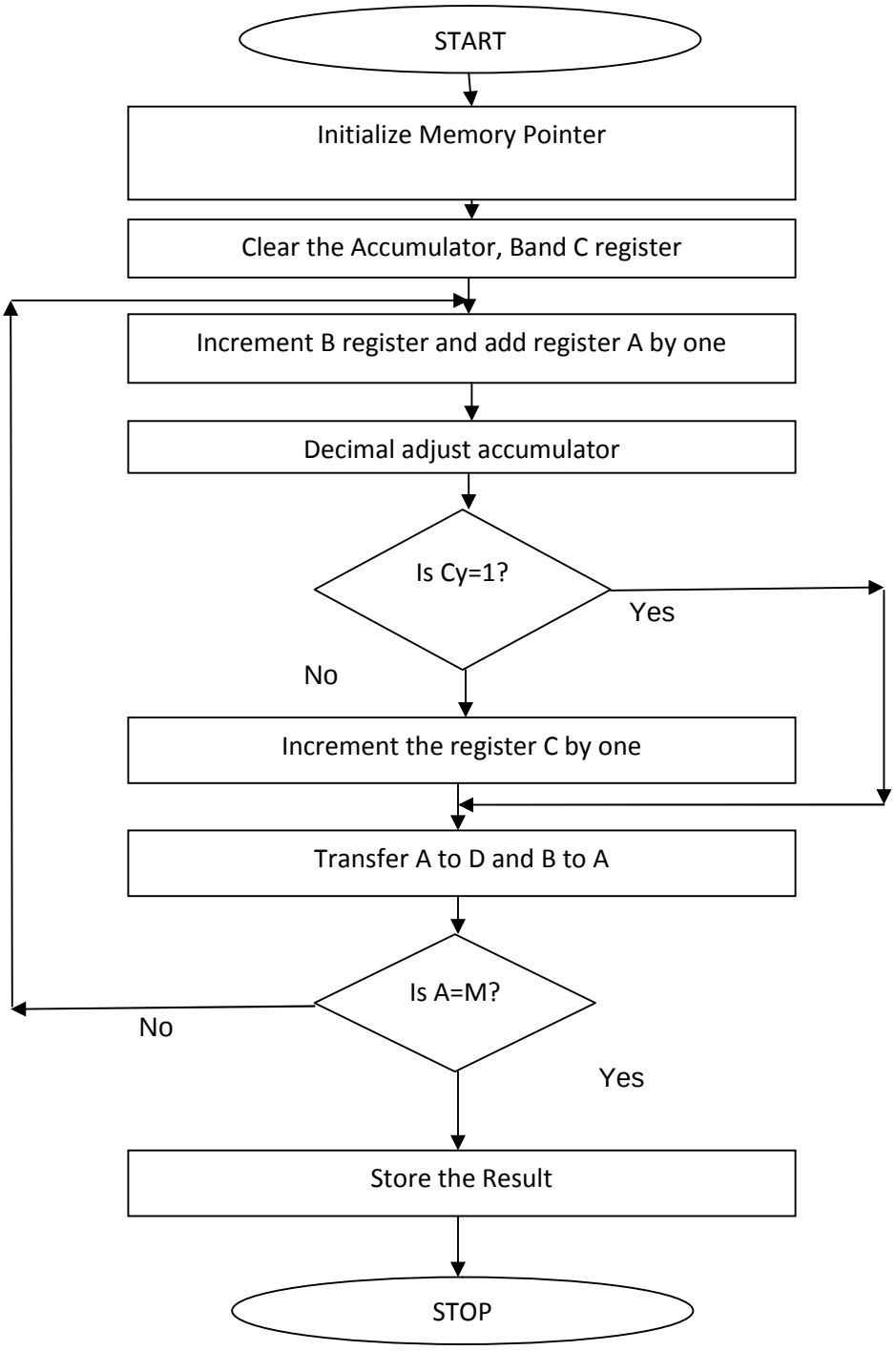
APPARATUS REQUIRED:

Sl.No	Name of the Apparatus	Qty
1	8085 Microprocessor kit	1
2	+5Volts Power Supply	1
3	Keyboard Connector	1

ALGORITHM:

1. Initialize the Memory Pointer.
2. Increment B register.
3. Increment accumulator by one and adjust to decimal every time.
4. Compare the given decimal number with B register value.
5. When both match, the equivalent decimal value is in A register.
6. Store the resultant in memory location.

FLOWCHART : (Decimal To Hex)



PROGRAM:

ADDRESS	LABEL	MNEMONICS	OPCODE	COMMENTS
8000		*LXI H,9100*		Initialize the memory pointer
		MVI A,00		Clear the Accumulator
		MVI B,00		Clear the B register
		MVI C,00		Clear the C register
	Loop1	*INR B*		Increment the B register
		ADI 01		Increment the A register
		DAA		Decimal Adjust Accumulator
		JNC NEXT		If no carry go to next loop
		INR C		Increment register C by one
	NEXT	*MOV D,A*		Transfer A to D
		MOV A,B		Transfer B to A
		CMP M		Compare A and M register
		MOV A,D		Transfer D to A
		JNZ Loop1		Jump if No Zero to loop1
		STA 9500		Store the result to memory
		MOV A,C		Move register B to A
		STA 9501		Store the result to memory
		HLT		Stop the program

Input:

Memory location	Data
9100	

Output:

Memory location	Data
9500	

RESULT:

Thus the program of conversion of hexadecimal to decimal given data was executed by using 8085.

EX.NO:11

DATE :

PROGRAM FOR CODE CONVERSION – BINARY TO ASCII

AIM:

To write an assembly language program to convert a given binary number into ASCII number using 8085 microprocessor.

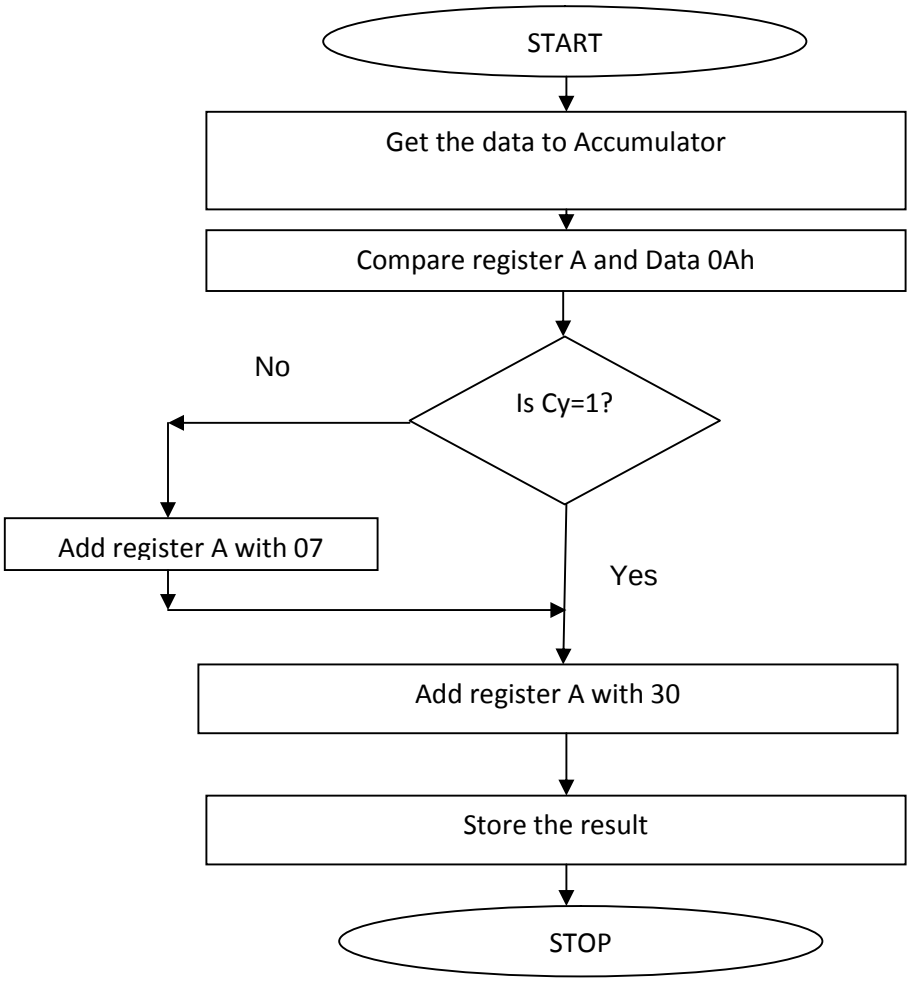
APPARATUS REQUIRED:

Sl.No	Name of the Apparatus	Qty
1	8085 Microprocessor kit	1
2	+5Volts Power Supply	1
3	Keyboard Connector	1

ALGORITHM:

1. Start the program
2. Load the data from address 9100 to A
3. Compare register A and data 0Ah(decimal 10).
4. If no carry add 37h with register A.
5. If Carry add 30h with register A.
6. Store the result to memory location.

FLOWCHART : (Binary to ASCII)



PROGRAM:

ADDRESS	LABEL	MNEMONICS	OPCODE	COMMENTS
8000		*LDA 9100*		Get the data to Accumulator
		MVI A,00		Clear the Accumulator
		CPI 0A		Compare register A and data 10
		JC LOOP1		If carry jump to loop1
		ADI 07H		Add A with 07
	LOOP1	*ADI 30H*		Add A with 30
		STA 9500		Store the result to memory
		HLT		Stop the program

Input:

Memory location	Data
9100	

Output:

Memory location	Data
9500	

RESULT:

Thus the program of conversion of binary to ASCII given data was executed by using 8085.

Exp No : 12

DATE :

PROGRAM FOR CODE CONVERSION – ASCII TO BINARY

AIM:

To write an assembly language program to convert a given ASCII number into binary number using 8085 microprocessor.

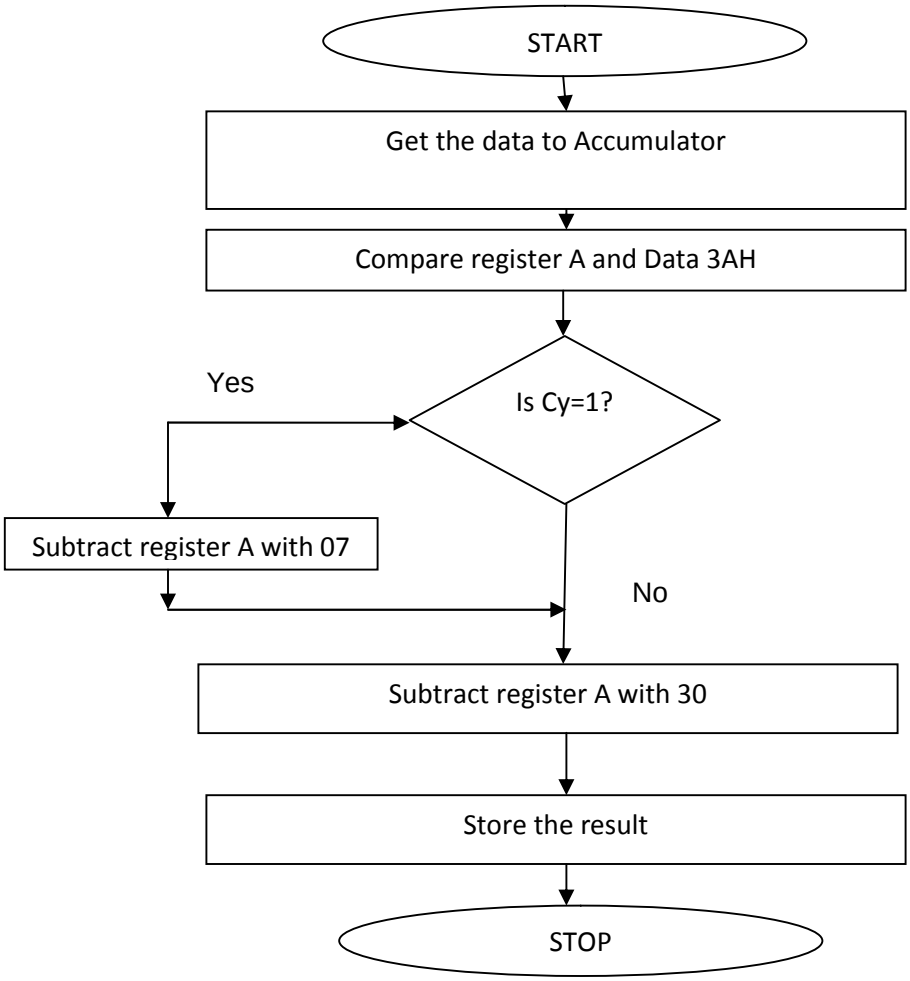
APPARATUS REQUIRED:

Sl.No	Name of the Apparatus	Qty
1	8085 Microprocessor kit	1
2	+5Volts Power Supply	1
3	Keyboard Connector	1

ALGORITHM:

1. Start the program
2. Load the data from address 9100 to A
3. Compare register A and data 3Ah.
4. If no carry subtract 37h with register A.
5. If carry subtract 30h with register A.
6. Store the result to memory location.

FLOWCHART : (ASCII TO BINARY)



PROGRAM:

ADDRESS	LABEL	MNEMONICS	OPCODE	COMMENTS
8000		*LDA 9100*		Get the data to Accumulator
		MVI A,00		Clear the Accumulator
		CPI 3A		Compare register A and data 3Ah
		JC LOOP1		If carry jump to loop1
		ADI 07H		Add A with 07
	LOOP1	*ADI 30H*		Add A with 30
		STA 9500		Store the result to memory
		HLT		Stop the program

Input :

Memory location	Data
9100	

Output:

Memory location	Data
9500	

RESULT:

Thus the program of conversion of ASCII to binary given data was executed by using 8085.

EX.NO:13
DATE :

PROGRAM FOR INTERFACING OF ADC WITH 8085

AIM:

To write an assembly language program to convert an analog signal into a digital signal using an ADC interfacing with 8085 microprocessor.

APPARATUS REQUIRED:

Sl. No	Name of the Apparatus	Qty
1	8085 Microprocessor kit	1
2	+5Volts Power Supply	1
3	Keyboard Connector	1
4	ADC Interfacing kit	1
5	Multimeter	1

Working procedure for ADC:

- Connect the 5v power supply to trainer kit and adc chord.
- Connect the 26-pin connector from the kit to adc chord.
- Connect the card from keyboard into the socket provided in the kit.
- Switch on the power supply.
- Assemble your program.
- Vary the pot in the adc chord.
- Execute it and view the output count in the register A which will be displayed in LCD display.
- Repeat the above steps for different inputs in the pots.

8085 ADDRESS:

8255 control register address – 23H

8255 port A address -20H

8255 port B address -21 H

8255 port C address -22H

PROGRAM:

ADDRESS	LABEL	MNEMONICS	OPCODE	COMMENTS
9000		*MVI A,90*		Control Word for port A as input and port c as output
		OUT 23		Out in control RE
		MVI A,01		Select input for MUX
		OUT 21		
		MVI A,FF		
		OUT 22		Port C is enable
		MVI A,00		Start of Conversion(SOC)
		OUT 22		
		MVI A,FF		
		OUT 22		
		CALL 9100		Delay subroutine
		IN20		End of conversion(EOC)
		RST 1		Break point
9100	delay	*MVI B,0F*		Delay Count
	Loop1	*MVI A,FF*		Load data FF to register A
	Loop2	*NOP*		No operation
		NOP		No operation
		DCR A		Decrement register A
		JNZ loop2		If no zero jump to loop2
		DCR B		Decrement register B by one
		JNZ loop1		If no zero jump to loop1
		RET		Return to main Program

Analog Input	Digital output

RESULT :

Thus the conversion of a analog signal into and digital signal was executed using interfacing of ADC with 8085.

EX.NO:14
DATE :

PROGRAM FOR INTERFACING OF DAC WITH 8085

AIM:

To write an assembly language program to convert a digital signal into a analog signal using an DAC interfacing with 8085 microprocessor.

APPARATUS REQUIRED:

Sl.No	Name of the Apparatus	Qty
1	8085 Microprocessor kit	1
2	+5Volts Power Supply	1
3	Keyboard Connector	1
4	DAC Interfacing kit	1
5	CRO	1

Working procedure for DAC:

- Connect the 5v power supply to trainer kit and dac(0800) chord.
- Connect the 26-pin connector from the kit to dac(0800) chord.
- Connect the card from keyboard into the socket provided in the kit.
- Switch on the power supply.
- Assemble your program.Give the digital data in the software program itself.
- Execute it and view the output count in the register A which will be displayed in CRO
- Repeat the above steps for different digital inputs.

i)SQUARE WAVE FORM
ALGORITHM:

1. Load the initial value (00) to Accumulator and move it to DAC.
2. Call the delay program
3. Load the final value (FF) to accumulator and move it to DAC.
4. Call the delay program.
5. Repeat steps 2 to 5.
6. Execute the program and using a CRO, verify that the waveform at the DAC2 output is a square-wave. Modify the frequency of the square-wave, by varying the time delay.

8085 ADDRESS:

8255 control register address – 23H

8255 port A address -20H

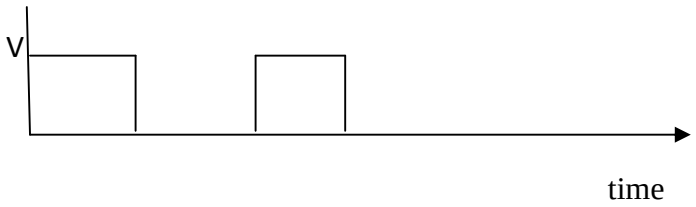
8255 port B address -21 H

8255 port C address -22H

PROGRAM: SQUARE WAVE FORM

ADDRESS	LABEL	MNEMONICS	OPCODE	COMMENTS
9000		*MVI A,80*		Control Word
		OUT 23		Out in control REG
	LOOP1	*MVI A,FF*		High Input
		OUT 21		output in port B
		CALL 9100		Delay subroutine
		MVI A,00		Low Input
		OUT 21		output in port B
		CALL 9100		Delay subroutine
		JMP LOOP1(9004)		Jump To Start
9100	delay	*MVI C,FF*		Delay Count
	Loop1	*MVI B,FF*		Load data FF to register A
	Loop2	*NOP*		No operation
		NOP		No operation
		DCR B		Decrement register B
		JNZ loop2		If no zero jump to loop2
		DCR C		Decrement register C by one
		JNZ loop1		If no zero jump to loop1
		RET		Return to main Program

SQUARE WAVE FORMS ;



II) SAW TOOTH GENERATION:

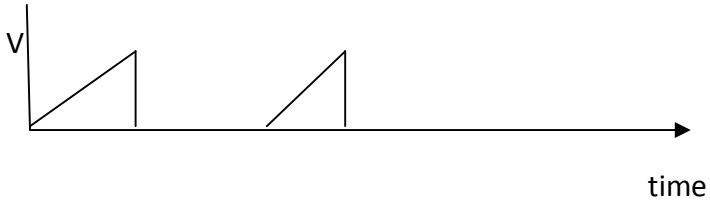
ALGORITHM:

1. Load the initial value (00) to Accumulator
2. Move the accumulator content to DAC.
3. Increment the accumulator content by 1.
4. Repeat steps 3 and 4.
5. Output digital data from 00 to FF constant steps of 01 to DAC1 repeat this sequence again and again. As a result a saw – tooth wave will be generated at DAC1 output.

PROGRAM: SAW TOOTH WAVE FORM

ADDRESS	LABEL	MNEMONICS	OPCODE	COMMENTS
9000		*MVI A,80*		Control Word
		OUT 23		Out in control REG
	START	*MVI A,00*		Low Input
	LOOP1	*OUT 21*		output in port B
		INR A		Increment register A by one
		JNZ LOOP1		If zero jump to loop1
		JMP START		Jump To Start

SAW TOOTH WAVE FORM :



(iii) TRIANGULAR WAVE GENERATION:

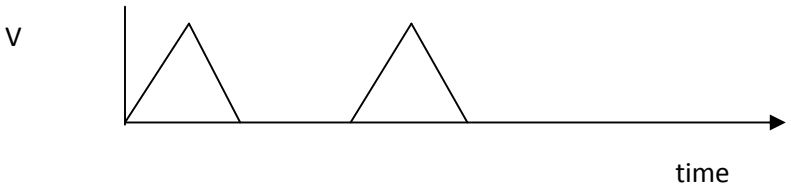
ALGORITHMS:

1. Load the initial value (00) to Accumulator.
2. Move the accumulator content to DAC
3. Increment the accumulator content by 1.
4. If accumulator content is zero proceed to next step. Else go to step 3.
5. Load value (FF) to accumulator.
6. Move the accumulator content to DAC.
7. Decrement the accumulator content by 1.
8. If accumulator content is zero go to step 2. Else go to step 2.

PROGRAM: TRIANGULAR WAVE FORM

ADDRESS	LABEL	MNEMONICS	OPCODE	COMMENTS
9000	START	*MVI L,00*		Transfer 00 to register L
	LOOP1	*MOV A,L*		Transfer L to A
		OUT 21		Output in control register
		INR L		Increment register L
		JNZ LOOP1		If no zero jump to loop1
		MVI L,FF		Transfer FF to register L
	LOOP2	*MOVA,L*		Transfer L to A
		OUT 21		Output in control register
		DCR L		Decrement register L
		JNZ LOOP2		If no zero jump to loop2
		JMP START		Jump to start loop

TRIANGULAR WAVE FORM :



RESULT :

Thus the conversion of a digital signal into an analog signal was executed using interfacing of DAC with 8085.

EXP.No :15

DATE :

PROGRAM FOR KEYBOARD AND DISPLAY INTERFACING WITH 8085

AIM:

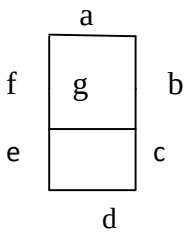
To interface 8279 Programmable Keyboard Display Controller with 8085 Microprocessor.

APPARATUS REQUIRED:

Sl.No	Name of the Apparatus	Qty
1	8085 Microprocessor kit	1
2	+5Volts Power Supply	1
3	Keyboard Connector	1
4	Keyboard and Display Interfacing kit	1

PROGRAM: 7 SEGMENT DISPLAY

ADDRESS	LABEL	MNEMONICS	OPCODE	COMMENTS
9000		MVI C,BA		7 segment code for 2
9002		MVI A,12		control word
9004		OUT 71		Control port
9006		MVI A,3E		Frequency division
9008		OUT 71		Control register
900A		MVI A,A0		Display / write inhibit
900C		OUT 71		Control register
900E		MVI B,08		
9010		MVI A,00		Clear display
9012		OUT 70		
9014		DCR B		
9015		JNZ 9012		
9018		MOV A,C		Take the character to display
9019		OUT 70		
901B		JMP 9019		



Char	D	C	B	A	E	F	G	H	HEX CODE
0	1	1	1	1	1	1	0	0	FC
1	0	1	1	0	0	0	0	0	60

RESULT:

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

EXP.No: 16

DATE :

PROGRAM FOR INTERFACING - TRAFFIC LIGHT CONTROLLER WITH 8085

AIM:

To write an assembly language program to simulate the traffic light at an intersection using a traffic light interface with 8085 microprocessor.

APPARATUS REQUIRED:

Sl.No	Name of the Apparatus	Qty
1	8085 Microprocessor kit	1
2	+5Volts Power Supply	1
3	Keyboard Connector	1
4	Traffic Light control Interfacing kit	1

WORKING PROCEDURE:

- Connect the 5V supply to trainer kit.
- Connect the 26 pin FRC from the kit.
- Switch on the power supply.
- Assemble the program.
- Execute it and output display by LED.

8085 ADDRESS:

PORT	ADDRESS
CWR	23
PORT A	20
PORT B	21
PORT C	22

PROGRAM: TRAFFIC LIGHT CONTROLLER

ADDRESS	LABEL	MNEMONICS	OPCODE	COMMENTS
8500		MVI A,80		All port as output
8502		Out 23		
For starting right turn in N-S sides & pedestrian stopping				
8504		MVI A,0F		For pedestrian
8506		Out 21		Signal
8508		MVI A,4D		For green LEDs in N-S
850A		OUT 20		Direction
850C		CALL DELAY(8569)		Sequence Delay
850F		CALL AMBER(855F)		Amber delay
For stopping vehicles in N-S direction & starting E-W direction				
8512		MVI A,8B		For stopping N-S sides
8514		OUT 20		For starting E-W sides
8516		CALL DELAY		Sequence Delay
8519		CALL AMBER		Amber delay
For starting right turn in N-S sides & stopping E-W sides				
851C		MVI A,49		For free left in all sides
851E		OUT 20		For stopping E-W sides
8520		MVI A,01		For right turn in N-S sides
8522		OUT 22		
8524		CALL DELAY		
8527		MVI A,07		
8529		OUT 22		
852B		CALL AMBER		
Stopping Right Turn In N-S Sides & Starting Turn In E-W Sides				
852E		MVI A,89		
8530		OUT 20		
8532		MVI A,02		
8534		OUT 22		
8536		CALL DELAY		
8539		MVI A,00		
853B		OUT 22		
853D		MVI A,30		
853F		OUT 20		
8541		MVI C,04		
8543		CALL DELAY		
For starting Pedestrian				

8546		*MVI A,C0*		
8548		*OUT 20*		
854A		*MVI A,F0*		
854C		*OUT 21*		
854E		*MVI C,10*		
8550		*CALL DELAYSUB*		
8553		*MVI A,30*		
8555		*OUT 20*		
8257		*MVI C,08*		
8259		*CALL DELAYSUB*		
825C		*JMP CONTINUE*		
855F	AMBER	*MVI A,39*		
8561		*OUT 20*		
8563		*MVI C,08*		
8568		*CALL DELAYSUB*		
8568		*RET*		
8569	DELAY	*MVI C,40*		
856B		*CALL DELAYSUB*		
856E		*RET*		
856F	DELAYSUB			
	BACK2	*MVI D,FF*		
	BACK1	*MVI A,FF*		
	BACK	*NOP*		
		DCR A		
		JNZ BACK		
		DCR D		
		JNZ BACK1		
		MOV A,C		
		JZ OUT		
		DCR C		
		JNZ BACK2		
	OUT	*RET*		

RESULT:

Thus an assembly language program to simulate the traffic light at an intersection using a traffic light was written and implemented.

EXP.No : 17

DATE:

PROGRAM FOR I/O PORT SERAIL COMMUNICATION WITH 8085

AIM:

To write a program to initiate 8251 and to check the transmission and reception of character.

APPARATUS REQUIRED:

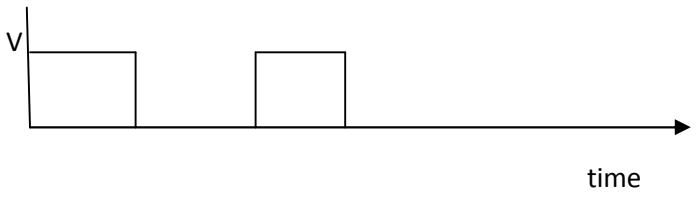
Sl.No	Name of the Apparatus	Qty
1	8085 Microprocessor kit	1
2	+5Volts Power Supply	1
3	Keyboard Connector	1
4	8251/8253 Interfacing kit	1

ALGORITHM:

1. Initialize timer (8253) IC
2. Move the Mode command word (36H) to A reg.
3. Output it port address CE
4. Move the command instruction word (37H) to A reg.
5. Output it to port address C8

PROGRAM: SQUARE WAVE GENERATION

ADDRESS	LABEL	MNEMONICS	OPCODE	COMMENTS
9000		*MVI A,36*		Channel 0 in mode 3
		OUT CE		Send mode control word
		MVI A,0A		LSB of count
		OUT C8		Write count to register
		MVI A,00		MSB of count
		OUT C8		Write count to register
		HLT		Stop the program



RESULT:

Thus the program to initiate 8251 was written and executed.

Exp No : 18
Date :

PROGRAM FOR 8 BIT ADDITION USING 8051

AIM:

To write an assembly language program for addition of two 8 bit data using 8051 microcontroller.

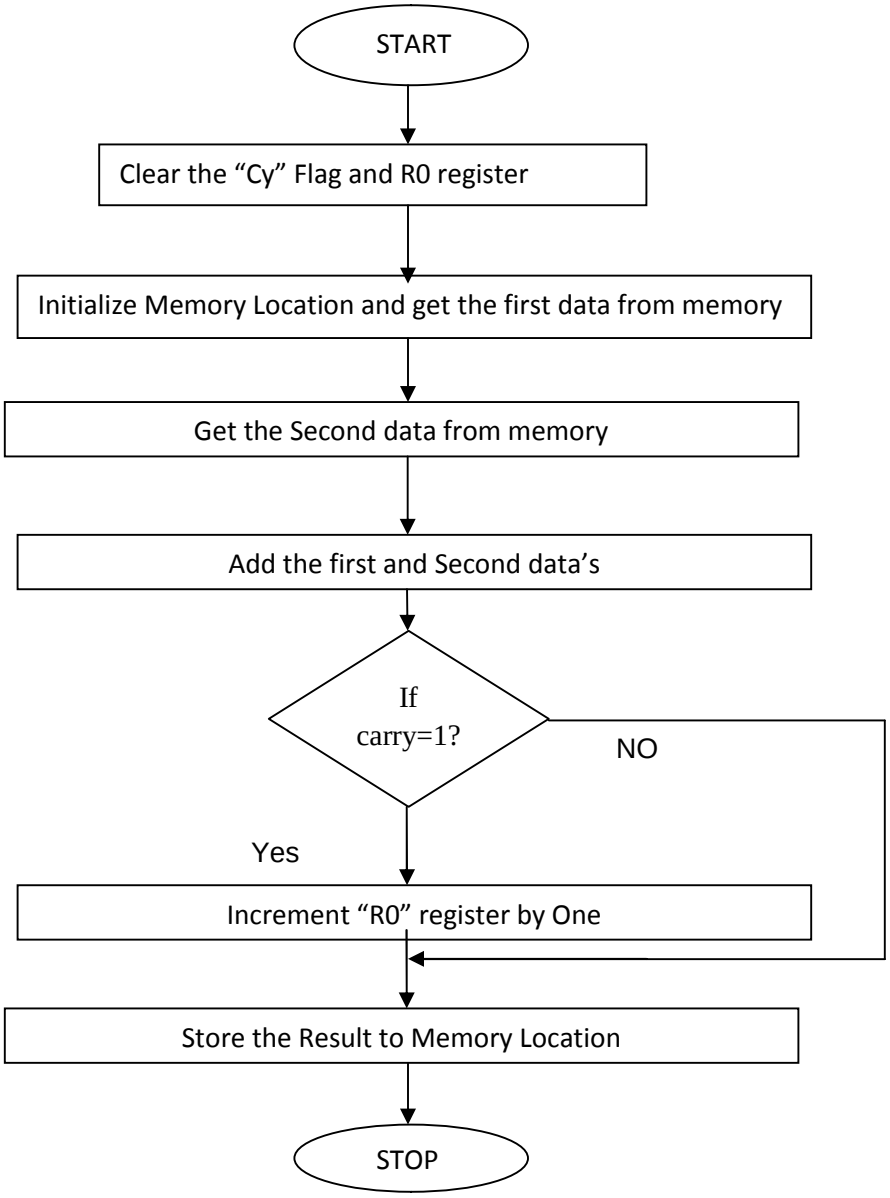
APPARATUS REQUIRED:

Sl.No	Name of the Apparatus	Qty
1	8051 Microcontroller kit	1
2	+5Volts Power Supply	1
3	Keyboard Connector	1

ALGORITHM:

1. Clear the register “C”
2. Initialize the memory pointer to data location.
3. Get the first Data from memory Location and move to register “A”.
4. Get the second data from memory location.
5. Add first and second data.
6. If carry the increment the register ‘C’ by one.
7. Store the result to memory location.

FLOW CHART: (8 bit Addition)



PROGRAM:

ADDRESS	LABEL	MNEMONICS	OP CODE	COMMENTS
8000		*CLR C*		Clear the "Carry" Flag
		MOV R0,#00		
		MOV DPTR,#9200		Initialize the memory pointer
		MOVX A,@DPTR		Move the First Data to A Reg
		MOV B,A		Transfer Data To B
		INC DPTR		Increment the memory pointer
		MOVX A,@DPTR		Get the second data from memory
		ADD A,B		Add First and Second Data
		JNC Loop1		Jump if No Carry to loop1
		INR R0		Increment the "C" register by one
	Loop1	*MOV DPTR,#9500*		Initialize the memory pointer
		MOVX @DPTR,A		Store The Result
		MOV A,R0		Move the Carry result to Reg "A"
		INC DPTR		Increment the memory pointer
		MOVX @DPTR,A		Store the Carry Result
	HLT	*SJMP : HLT*		Stop the program

Input:

Memory location	Data
9200	
9201	

Output:

Memory location	Data
9500	
9501	

RESULT :

Thus the addition of two numbers was performed using the 8085 microprocessor.

Exp No : 19
Date :

PROGRAM FOR 8 BIT SUBTRACTION USING 8051

AIM:

To write an assembly language program for subtraction of two 8 bit data using 8051 microcontroller.

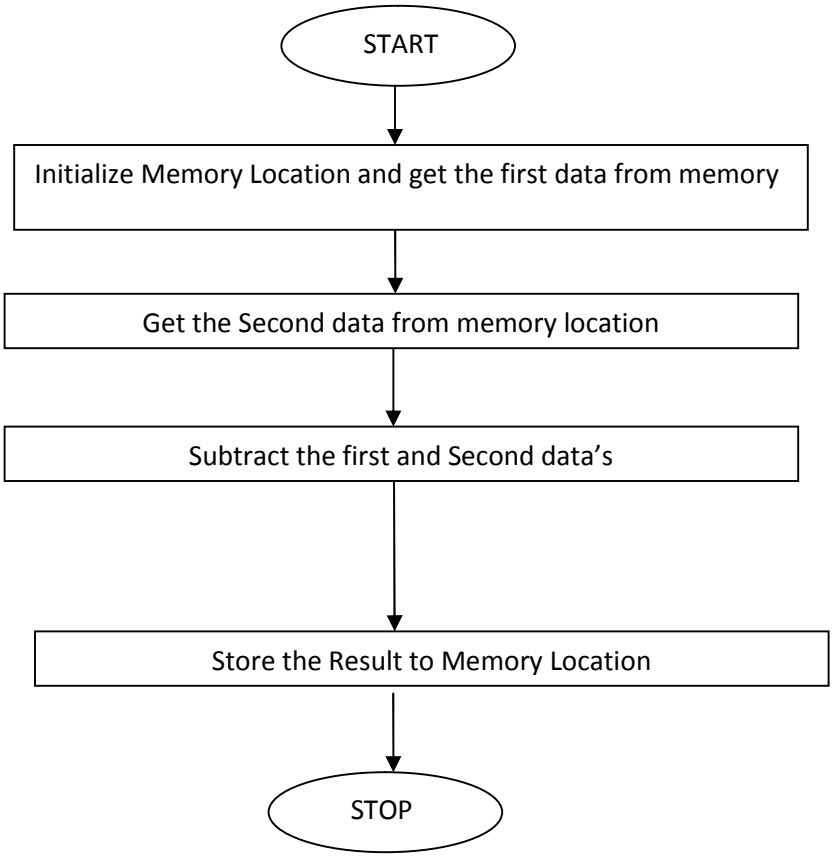
APPARATUS REQUIRED:

Sl.No	Name of the Apparatus	Qty
1	8051 Microcontroller kit	1
2	+5Volts Power Supply	1
3	Keyboard Connector	1

ALGORITHM:

1. Initialize the memory pointer to data location.
2. Get the first Data from memory and move to register A**.**
3. Get the second data from memory location.
4. Subtract the first and second data.
5. Store the result to memory location.

FLOW CHART: (8 bit Subtraction)



PROGRAM:

ADDRESS	LABEL	MNEMONICS	OPCODE	COMMENTS
8000		*MOV DPTR,#9200*		Initialize the memory pointer
		MOVX A,@DPTR		Move the First Data to A Reg
		MOV B,A		Transfer Data To B
		INC DPTR		Increment the memory pointer
		MOVX A,@DPTR		Get the second data from memory
		SUBB A,B		Add First and Second Data
		MOV DPTR,#9500		Initialize the memory pointer
		MOVX @DPTR,A		Store The Result
	HLT	*SJMP : HLT*		Stop the program

Input:

Memory location	Data
9200	
9201	

Output:

Memory location	Data
9500	

RESULT :

Thus the subtraction of two numbers was performed using the 8085 microprocessor.

Exp No : 20
Date :

PROGRAM FOR 8 BIT MULTIPLICATION USING 8051

AIM:

To write an assembly language program for multiplication of two 8 bit data using 8051 microcontroller.

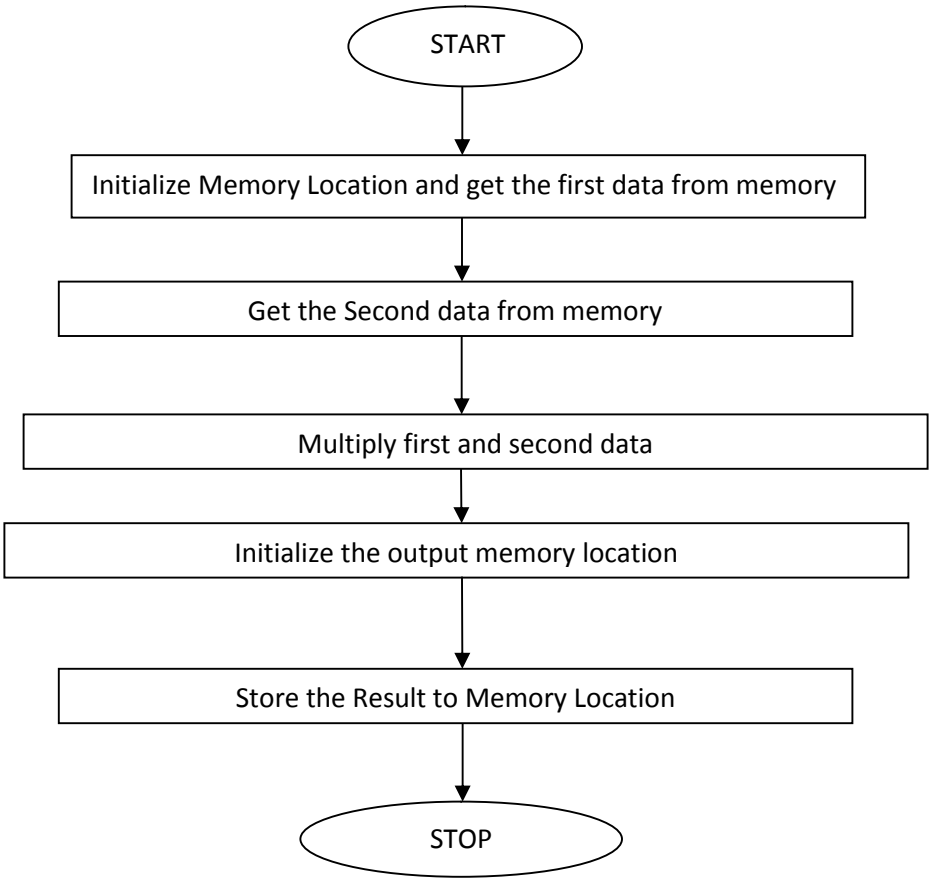
APPARATUS REQUIRED:

Sl.No	Name of the Apparatus	Qty
1	8051 Microcontroller kit	1
2	+5Volts Power Supply	1
3	Keyboard Connector	1

ALGORITHM:

1. Initialize the memory pointer to data location.
2. Get the first Data from memory and move to register A.
3. Get the second data from memory location.
4. Multiply the first and second data.
5. Store the result to memory location.

FLOW CHART: (8 bit Multiplication)



PROGRAM:

ADDRESS	LABEL	MNEMONICS	OPCODE	COMMENTS
8000		*MOV DPTR,#9200*		Initialize the memory pointer
		MOVX A,@DPTR		Move the First Data to A Register
		MOV B,A		Transfer Data To B
		INC DPTR		Increment the memory pointer
		MOVX A,@DPTR		Get the second data from memory
		MUL AB		Multiply First and Second Data
		MOV DPTR,#9500		Initialize the memory pointer
		MOVX @DPTR,A		Store The LSB Result
		INC DPTR		Increment the data pointer
		MOV B,A		Transfer B to A
		MOVX @DPTR,A		Store the MSB result
	HLT	*SJMP : HLT*		Stop the program

Input:

Memory location	Data
9200	
9201	

Output:

Memory location	Data
9500	
9501	

RESULT:

Thus the multiplication of two numbers was performed using the 8085 microprocessor.

Exp No : 21
Date :

PROGRAM FOR 8 BIT DIVISION USING 8051

AIM:

To write an assembly language program for division of two 8 bit data using 8051 microcontroller.

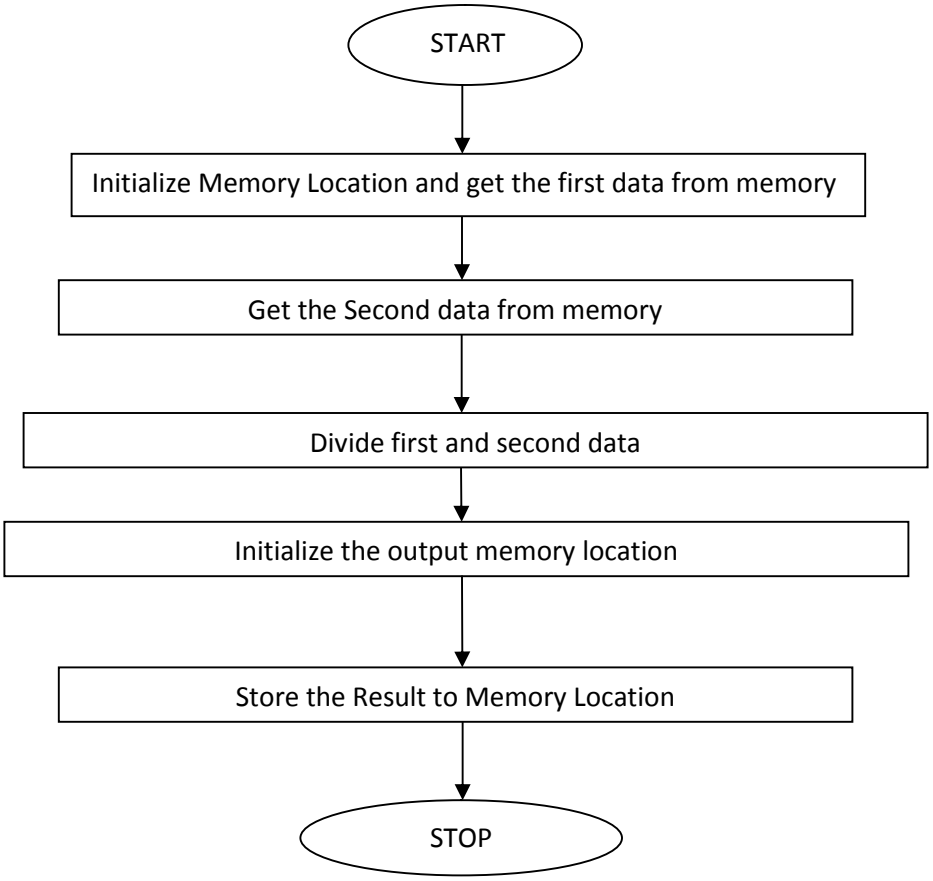
APPARATUS REQUIRED:

Sl.No	Name of the Apparatus	Qty
1	8051 Microcontroller kit	1
2	+5Volts Power Supply	1
3	Keyboard Connector	1

ALGORITHM:

1. Initialize the memory pointer to data location.
2. Get the first Data from memory and move to register A.
3. Get the second data from memory location.
4. Divide the first and second data.
5. Store the result to memory location.

FLOW CHART: (8 bit Division)



PROGRAM:

ADDRESS	LABEL	MNEMONICS	OPCODE	COMMENTS
8000		*MOV DPTR,#9200*		Initialize the memory pointer
		MOVX A,@DPTR		Move the First Data to A Reg
		MOV B,A		Transfer Data To B
		INC DPTR		Increment the memory pointer
		MOVX A,@DPTR		Get the second data from memory
		DIV AB		Multiply First and Second Data
		MOV DPTR,#9500		Initialize the memory pointer
		MOVX @DPTR,A		Store The Remainder Result
		INC DPTR		Increment the data pointer
		MOV B,A		Transfer B to A
		MOVX @DPTR,A		Store the Quotient result
	HLT	*SJMP : HLT*		Stop the program

Input:

Memory location	Data
9200	
9201	

Output:

Memory location	Data
9500	
9501	

Result:

Thus the division of two numbers was performed using the 8085 microprocessor.

EX.NO:22
DATE :

PROGRAM FOR INTERFACING OF ADC WITH 8051

AIM:

To write an assembly language program to convert an analog signal into a digital signal using an ADC interfacing with 8051 microcontroller.

APPARATUS REQUIRED:

Sl. No	Name of the Apparatus	Qty
1	8051 Microcontroller kit	1
2	+5Volts Power Supply	1
3	Keyboard Connector	1
4	ADC Interfacing kit	1
5	Multi-meter	1

Working procedure for ADC:

- Connect the 5v power supply to trainer kit and adc chord.
- Connect the 26-pin connector from the kit to adc chord.
- Connect the card from keyboard into the socket provided in the kit.
- Switch on the power supply.
- Assemble your program.
- Vary the pot in the adc chord.
- Execute it and view the output count in the register A which will be displayed in LCD display.
- Repeat the above steps for different inputs in the pots.

8085 ADDRESS:

8255 control register address – 6003H

8255 port A address -6000H

8255 port B address -6001 H

8255 port C address -6002H

PROGRAM:

ADDRESS	LABEL	MNEMONICS	OPCODE	COMMENTS
9000		*MOV DPRT,#6003*		Control Word for port A as input and port c as output
		MOV A,#90		
		MOVX @DPTR,A		
		MOV DPTR,#6002		
		MOV A,#FF		SOC(start of conversion)
		MOVX @DPTR,A		
		MOV DPRT,#6002		
		MOV A,#00		
		MOVX @DPTR,A		
		MOV DPTR,#6002		
		MOV A,#FF		
		MOVX @DPTR,A		
		LCALL DELAY		
		MOV DPTR,#6000		
		MOV A,#FF		
		MOVX @DPTR,A		
		LCALL 00BB		
	DELAY	*MOV R1,#FF*		
	LOOP	*NOP*		
		NOP		
		DJNZ R1,LOOP		
		RET		Return to main Program

Analog Input	Digital output

RESULT:

Thus the conversion of a analog signal into and digital signal was executed using interfacing of ADC with 8051.

EX.NO:23
DATE :

PROGRAM FOR INTERFACING OF DAC WITH 8051

AIM:

To write an assembly language program to convert an digital signal into a analog signal using an DAC interfacing with 8051 microcontroller.

APPARATUS REQUIRED:

Sl.No	Name of the Apparatus	Qty
1	8051 Microcontroller kit	1
2	+5Volts Power Supply	1
3	Keyboard Connector	1
4	DAC Interfacing kit	1
5	CRO	1

Working procedure for DAC:

- Connect the 5v power supply to trainer kit and dac(0800) chord.
- Connect the 26-pin connector from the kit to dac(0800) chord.
- Connect the card from keyboard into the socket provided in the kit.
- Switch on the power supply.
- Assemble your program.
- Give the digital data in the software program itself.
- Execute it and view the output count in the register A which will be displayed in CRO
- Repeat the above steps for different digital inputs.

i)SQUARE WAVE FORM

ALGORITHM:

1. Load the initial value (00) to Accumulator and move it to DAC.
2. Call the delay program
3. Load the final value (FF) to accumulator and move it to DAC.
4. Call the delay program.
5. Repeat steps 2 to 5.
6. Execute the program and using a CRO, verify that the waveform at the DAC2 output is a square-wave. Modify the frequency of the square-wave, by varying the time delay.

8085 ADDRESS:

8255 control register address – 6003H

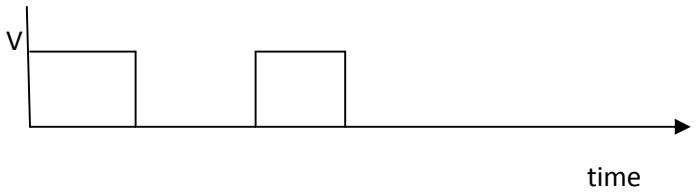
8255 port A address -6000H

8255 port B address -6001 H

8255 port C address -6002H

PROGRAM: SQUARE WAVE FORM

ADDRESS	LABEL	MNEMONICS	OPCODE	COMMENTS
8000	LOOP1	*MOV P1,#00*		Low Input
		LCALL DELAY		Call Delay
		MOV P1,#99		High Input
		LCALL DELAY		Call Delay
		SJMP START		Jump To Start
	DELAY	*MOV R0,#FF*		
	RPT	*DJNZ R0,RPT*		
		RET		Return to main Program



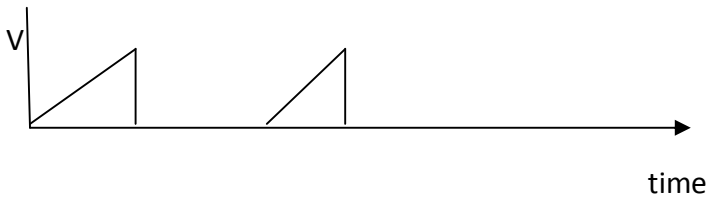
II) SAW TOOTH GENERATION:

ALGORITHM:

1. Load the initial value (00) to Accumulator
2. Move the accumulator content to DAC.
3. Increment the accumulator content by 1.
4. Repeat steps 3 and 4.
5. Output digital data from 00 to FF constant steps of 01 to DAC1 repeat this sequence again and again. As a result a saw – tooth wave will be generated at DAC1 output.

PROGRAM: SAW TOOTH WAVE FORM

ADDRESS	LABEL	MNEMONICS	OPCODE	COMMENTS
9000	LOOP	*MOV P1,R1*		
		LCALL DELAY		
		INC R1		
		SJMP LOOP		
	DELAY	*MOV R0,#FF*		
	RPT	*DJNZ,R0,RPT*		
		RET		



RESULT :

Thus the conversion of a digital signal into an analog signal was executed using interfacing of DAC with 8051.

EX.NO:24
DATE :

PROGRAM FOR INTERFACING STEPPER MOTOR USING 8051 MICROCONTROLLER

AIM:

To interface a stepper motor with 8051 microcontroller and operate it.

THEORY:

A motor in which the rotor is able to assume only discrete stationary angular position is a stepper motor. The rotary motion occurs in a step-wise manner from one equilibrium position to the next. Stepper Motors are used very wisely in position control systems like printers, disk drives, process control machine tools, etc.

The basic two-phase stepper motor consists of two pairs of stator poles. Each of the four poles has its own winding. The excitation of any one winding generates a North Pole. A South Pole gets induced at the diametrically opposite side. The rotor magnetic system has two end faces. It is a permanent magnet with one face as South Pole and the other as North Pole.

The Stepper Motor windings A1, A2, B1, B2 are cyclically excited with a DC current to run the motor in clockwise direction. By reversing the phase sequence as A1, B2, A2, B1, anticlockwise stepping can be obtained.

2-PHASE SWITCHING SCHEME:

In this scheme, any two adjacent stator windings are energized. The switching scheme is shown in the table given below. This scheme produces more torque.

A1	A2	B1	B2	HEX	CLOCK WISE	ANTI CLOCKWISE
1	0	0	1	09	↓	↑
0	1	0	1	05		
0	1	1	0	06		
1	0	1	0	0A		

PROCEDURE:

1. Enter the above program starting from location 4100.and execute the same.
2. The stepper motor rotates.
3. Varying the count at R4 and R5 can vary the speed.
4. Entering the data in the look-up TABLE in the reverse order can vary direction of rotation.

PROGRAM:

ADDRESS	LABEL	MNEMONICS	OPCODE	COMMENTS
8000	START	*MOV DPTR,#TABLE*		Initialize the memory pointer
		MOV R0,#04		Load the counter
	LOOP1	*MOVX A,@DPTR*		Move the First Data to A Reg
		PUSH DPH		Transfer DPH to SP
		PUSH DPL		Transfer DPL to SP
		MOV DPTR,#0FFCO		Initialize the port address
		MOVX @DPTR,A		Send the value to port address
		MOV R4,#0FF		Delay subroutine
	D1	*MOV R5,#0FF*		
	D2	*DJNZ R5,D2*		
		DJNZ R4,D1		
		POP DPL		
		POP DPH		
		INC DPTR		Next data
		SJMP : START		Stop the program
	LOOKUP	*09,06,05,0A*		Data's

RESULT:

Thus a stepper motor was interfaced with 8051 and run in forward and reverse directions at various speeds.