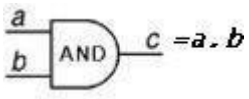
LIST OF EXPERIMENTS

1. Verification of Logic Gates and Boolean theorems.
2. Design and implementation of adder and subtractor
3. Design and implementation of Encoder and Decoder.
4. Design and implementation of Multiplexer and Demultiplexer.
5. Design and implementation of Parity generator/checker.
6. Design and implementation of Shift registers
7. Simple 8085 Assembly Language Programs: 8 bit Arithmetic operations
8. Interfacing and Programming of 8255- Programmable Peripheral Interface
9. Interfacing of a stepper motor with 8085 microprocessor
10. Interfacing of a traffic light controller with 8085 microprocessor

2-Input AND gate:

LOGIC DIAGRAM



TRUTH TABLE

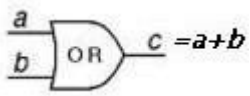
INPUTS		OUTPUT
a	b	c
0	0	0
0	1	0
1	0	0
1	1	1

OUTPUT TABLE

INPUTS		OUTPUT
a	b	c
0	0	
0	1	
1	0	
1	1	

2-Input OR gate:

LOGIC DIAGRAM



TRUTH TABLE

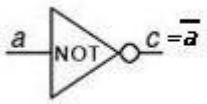
INPUTS		OUTPUT
a	b	c
0	0	0
0	1	1
1	0	1
1	1	1

OUTPUT TABLE

INPUTS		OUTPUT
a	b	c
0	0	
0	1	
1	1	
1	1	

NOT gate:

LOGIC DIAGRAM



TRUTH TABLE

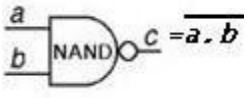
INPUTS	OUTPUT
a	c
0	1
1	0

OUTPUT TABLE

INPUTS	OUTPUT
a	c
0	
1	

2- Input NAND gate:

LOGIC DIAGRAM



TRUTH TABLE

INPUTS		OUTPUT
a	b	c
0	0	1
0	1	1
1	0	1
1	1	0

OUTPUT TABLE

INPUTS		OUTPUT
a	b	c
0	0	
0	1	
1	0	
1	1	

Expt. No.: 1	**VERIFICATION OF LOGIC GATES AND BOOLEAN THEOREMS**
Date:	

Aim

To verify the truth table of AND gate, OR gate, NOT gate, NAND gate, NOR gate, Exclusive-OR gate and Exclusive-NOR gate and Boolean theorems.

Components Required

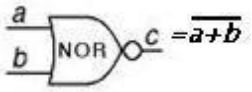
S. No.	Description	Specification	Quantity
1.	2-input AND Gate	IC 7408/4081	1
2.	2-input OR Gate	IC 7432/4071	1
3.	Hex Inverter (NOT Gate)	IC 7404/4069	1
4.	2-input NAND Gate	IC 7400/4011	1
5.	2-input NOR Gate	IC 7402/4001	1
6.	2-input X-OR Gate	IC 7486/4030	1
7.	Digital Trainer Kit	-	1
8.	Bread board	-	1
9.	Power Supply	5V	1
10.	Connecting wires	-	As Required

Procedure:

(i) Make the connections as per the logic diagram.
(ii) Switch on the power supply.
(iii) Apply various combinations of inputs using the switches according to the truth table and observe the condition of output LEDs.
(iv) Note down the output as ‘1’ when the LED is ‘ON’ and as ‘0’ when the LED is ‘OFF’.

2- Input NOR gate:

LOGIC DIAGRAM



TRUTH TABLE

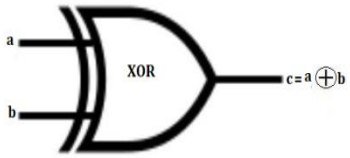
INPUTS		OUTPUT
a	b	c
0	0	1
0	1	0
1	0	0
1	1	0

OUTPUT TABLE

INPUTS		OUTPUT
a	b	c
0	0	
0	1	
1	0	
1	1	

2- Input XOR gate:

LOGIC DIAGRAM



TRUTH TABLE

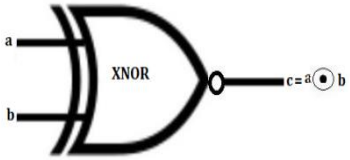
INPUTS		OUTPUT
a	b	c
0	0	0
0	1	1
1	0	1
1	1	0

OUTPUT TABLE

INPUTS		OUTPUT
a	b	c
0	0	
0	1	
1	0	
1	1	

2- Input XNOR gate:

LOGIC DIAGRAM



TRUTH TABLE

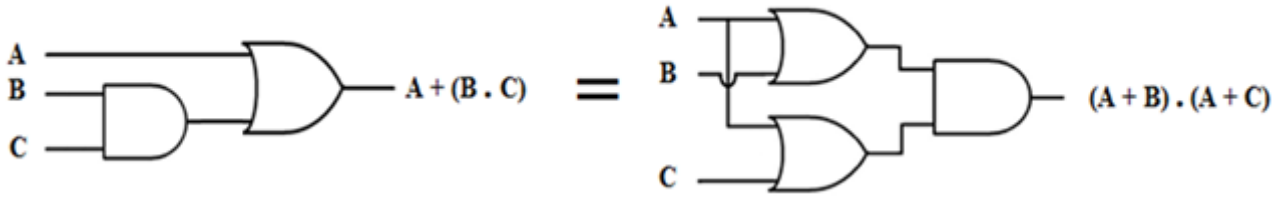
INPUTS		OUTPUT
a	b	c
0	0	1
0	1	0
1	0	0
1	1	1

OUTPUT TABLE

INPUTS		OUTPUT
a	b	c
0	0	
0	1	
1	0	
1	1	

Distributive Theorem:

A + (B.C) = (A + B) . (A + C)



Output Table:

INPUTS			OUTPUTS				
A	**B**	**C**	**B.C**	**A + (B.C)**	**A + B**	**A + C**	**(A + B).(A + C)**
0	0	0					
0	0	1					
0	1	0					
0	1	1					
1	0	0					
1	0	1					
1	1	0					
1	1	1					

Result

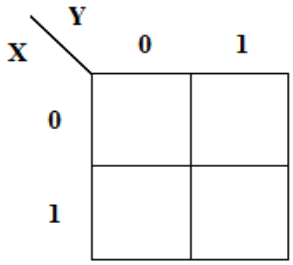
PREPARATION	**30**	
LAB PERFORMANCE	**30**	
REPORT	**40**	
TOTAL	**100**	
INITIAL OF FACULTY		

HALF ADDER:

Truth Table:

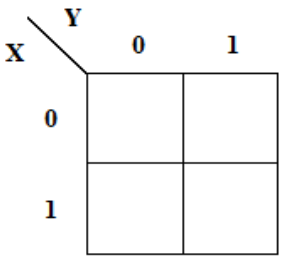
INPUTS		OUTPUTS	
X	Y	Carry	Sum
0	0		
0	1		
1	0		
1	1		

K-Map for Carry:



Carry =

K-Map for Sum:



Sum =

Logic diagram:

Output Table:

INPUTS		OUTPUTS	
X	Y	Carry	Sum
0	0		
0	1		
1	0		
1	1		

HALF SUBTRACTOR:

Truth Table:

INPUTS		OUTPUTS	
X	Y	Borrow	Difference
0	0		
0	1		
1	0		
1	1		

Expt. No.: 2	DESIGN AND IMPLEMENTATION OF ADDER AND SUBTRACTOR
Date:	

Aim:

To design and implement Half adder / subtractor and Full adder/subtractor using Digital logic gates.

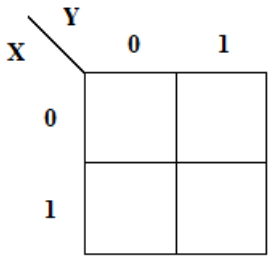
Components Required:

S. No.	Description	Specification	Quantity
1.	2-input AND Gate	IC 4081/7408	1
2.	2-input OR Gate	IC 4071/7432	1
3.	NOT Gate	IC 4069/7404	1
4.	2-input XOR Gate	IC 4030/7486	1
5.	Digital Trainer Kit	-	1
6.	Bread board	-	1
7.	Power Supply	5V	1
8.	Connecting wires	-	As Required

Procedure:

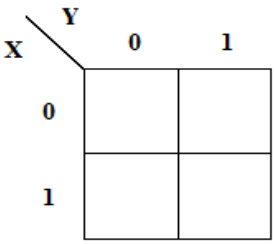
(i) Create the truth table according to the logic of the adder/subtractor.
(ii) Derive the expression for the output using k-map.
(iii) Draw the logic diagram.
(iv) Make the connections as per the logic diagram.
(v) Switch on the power supply.
(vi) Apply various combinations of inputs according to the truth table and observe the conditions of the output LEDs.
(vii) Note down the output for half adder/full adder (sum, carry) and half subtractor/full (difference, borrow) for different combinations of inputs.

K-Map for Borrow:



Borrow =

K-Map for Difference:



Difference =

Logic diagram:

Output Table:

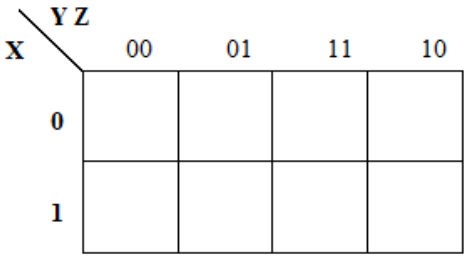
INPUTS		OUTPUTS	
X	Y	Borrow	Difference
0	0		
0	1		
1	0		
1	1		

FULL ADDER:

Truth Table:

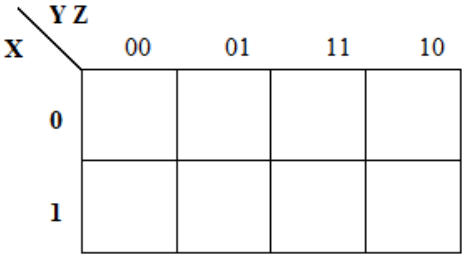
INPUTS			OUTPUTS	
X	Y	Z	Carry	Sum
0	0	0		
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

K-Map for Carry:



Carry =

K-Map for Sum:



Sum =

Logic diagram:

Output Table:

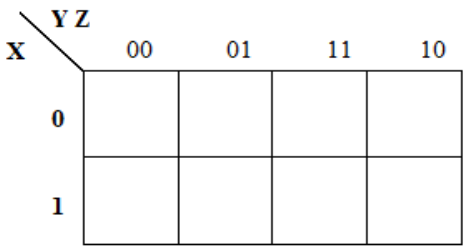
INPUTS			OUTPUTS	
X	**Y**	**Z**	**Carry**	**Sum**
0	0	0		
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

FULL SUBTRACTOR:

Truth Table:

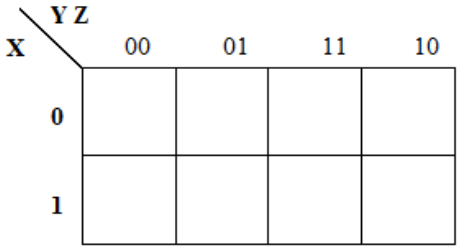
INPUTS			OUTPUTS	
X	**Y**	**Z**	**Borrow**	**Difference**
0	0	0		
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

K-Map for Borrow:



Borrow =

K-Map for Difference:



Difference =

Logic diagram:

Output Table:

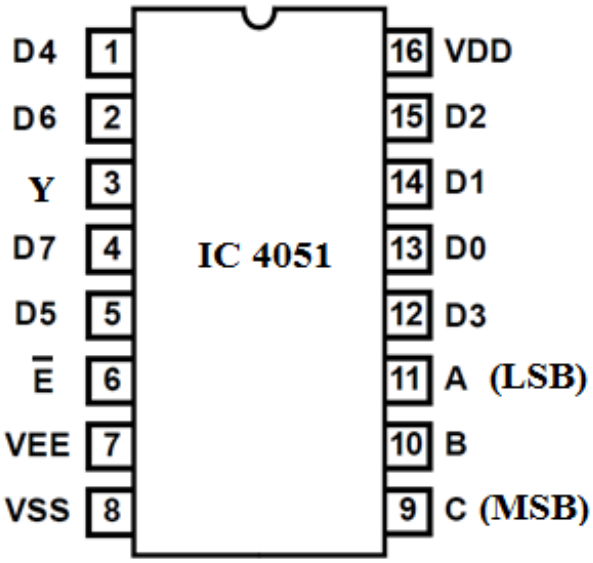
INPUTS			OUTPUTS	
X	**Y**	**Z**	**Borrow**	**Difference**
0	0	0		
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

Result:

PREPARATION	**30**	
LAB PERFORMANCE	**30**	
REPORT	**40**	
TOTAL	**100**	
INITIAL OF FACULTY		

4:1 MULTIPLEXER:

Pin Diagram:

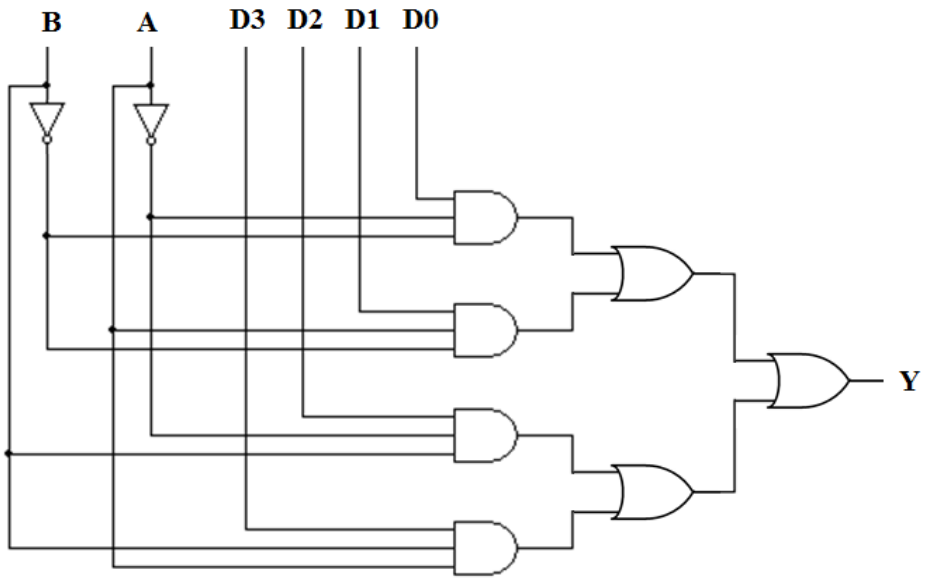


Truth Table:

SELECT LINES		OUTPUT
B	**A**	**Y**
0	0	D0
0	1	D1
1	0	D2
1	1	D3

Y = B'.A'.D0 + B'.A.D1 + B.A'.D2 + B.A.D3
DATA INPUTS : D0, D1, D2, D3

Internal Connection of IC 4051:



Expt. No.: 3	DESIGN AND IMPLEMENTATION OF MULTIPLEXER AND DEMULTIPLEXER
Date:	

Aim:

To design and implement Multiplexer and Demultiplexer using Digital IC 4051 and IC 4555 respectively.

Components Required:

S. No.	Description	Specification	Quantity
1.	Multiplexer	IC 4051	1
2.	Decoder/Demultiplexer	IC 4555/4556	1
3.	Digital Trainer Kit	-	1
4.	Bread board	-	1
5.	Power Supply	5V	1
6.	Connecting wires	-	As Required

Procedure:

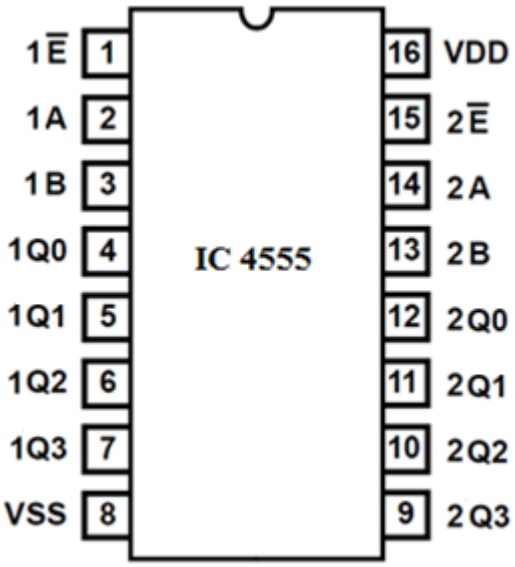
(i) Make the connections as per the logic diagram.
(ii) Switch on the power supply.
(iii) Apply various combinations of inputs according to the truth table and observe condition of output LEDs.
(iv) Note down the outputs of multiplexer and Demultiplexer for different combinations of input.

Output Table:

SELECT LINES		INPUTS				OUTPUT
B	A	D3	D2	D1	D0	Y
0	0	0	0	0	0	
		0	1	1	1	
0	1	1	0	0	0	
		0	1	1	0	
1	0	0	0	1	0	
		0	1	0	0	
1	1	1	0	0	0	
		0	1	0	1	

1:4 DEMULTIPLEXER:

Pin Diagram:



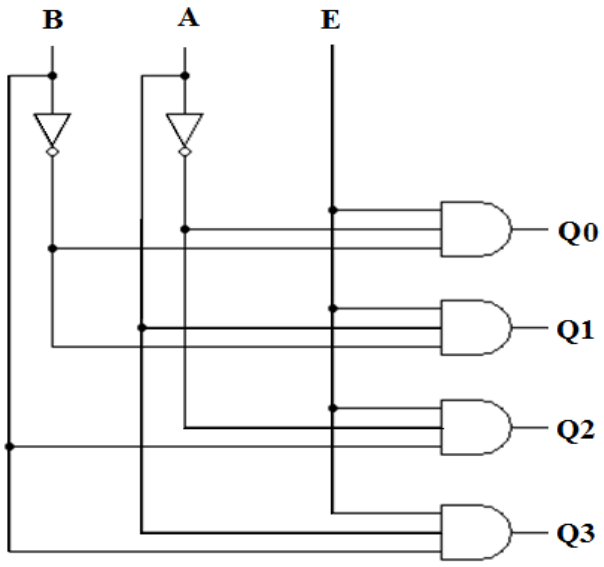
Note: The Data input D must be inverted and given at either $\overline{1E}$ or $\overline{2E}$

Truth Table:

DATA INPUT	SELECT LINES		OUTPUTS			
	B	A	Q3	Q2	Q1	Q0
D	0	0	0	0	0	D
D	0	1	0	0	D	0
D	1	0	0	D	0	0
D	1	1	D	0	0	0

Q0 = D**.**B'**.**A' Q1 = D**.**B'**.**A Q2 = D**.**B**.**A' Q3 = D**.**B**.**A

Internal Connection of IC 4555:



Output Table:

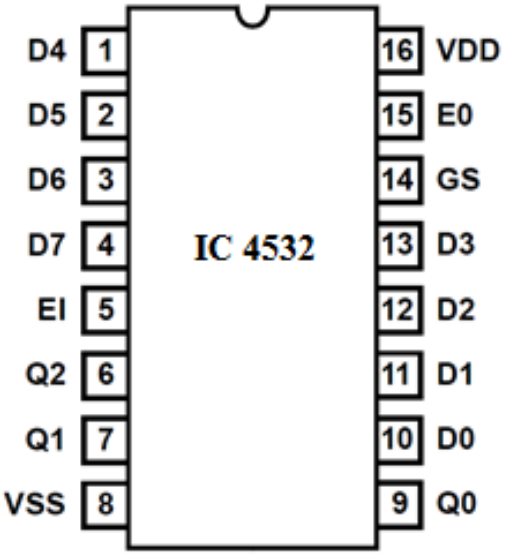
SELECT LINES		INPUT	OUTPUTS			
B	A	D	Q3	Q2	Q1	Q0
0	0	0				
		1				
0	1	0				
		1				
1	0	0				
		1				
1	1	0				
		1				

Result:

PREPARATION	**30**	
LAB PERFORMANCE	**30**	
REPORT	**40**	
TOTAL	**100**	
INITIAL OF FACULTY		

8-TO-3 LINE ENCODER:

Pin Diagram:

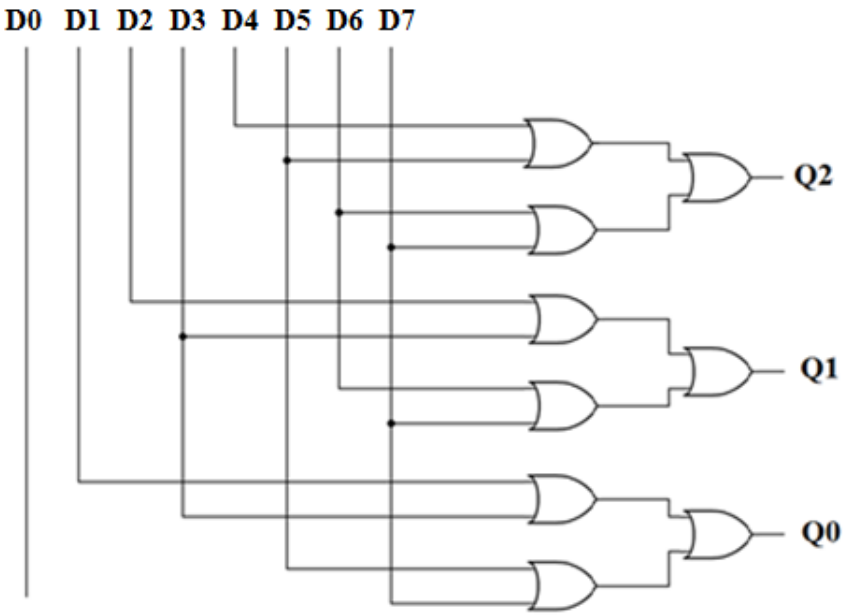


Truth Table:

INPUTS								OUTPUTS		
D7	D6	D5	D4	D3	D2	D1	D0	Q2	Q1	Q0
0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	1	0	0	0	1
0	0	0	0	0	1	0	0	0	1	0
0	0	0	0	1	0	0	0	0	1	1
0	0	0	1	0	0	0	0	1	0	0
0	0	1	0	0	0	0	0	1	0	1
0	1	0	0	0	0	0	0	1	1	0
1	0	0	0	0	0	0	0	1	1	1

Q2 = D4+D5+D6+D7 Q1 = D2+D3+D6+D7 Q0 = D1+D3+D5+D7

Internal Connection of IC 4532:



Expt. No.: 4	DESIGN AND IMPLEMENTATION OF ENCODER AND DECODER
Date:	

Aim

To design and implement Encoder and Decoder using Digital IC 4532 and IC 4555 respectively.

Components Required

S. No.	Description	Specification	Quantity
1.	Encoder	IC 4532	1
2.	Decoder/Demultiplexer	IC 4555/4556	1
3.	Digital Trainer Kit	-	1
4.	Bread board	-	1
5.	Power Supply	5V	1
6.	Connecting wires	-	As Required

Procedure

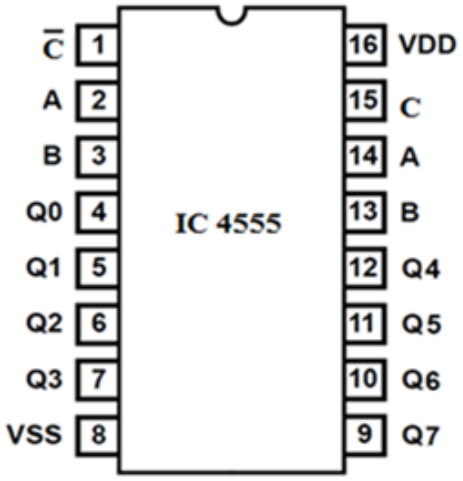
(i) Make the connections as per the logic diagram.
(ii) Switch on the power supply.
(iii) Apply various combinations of inputs according to the truth table and observe condition of output LEDs.
(iv) Note down the outputs of encoder and decoder for different combinations of input.

Output Table:

INPUTS									OUTPUTS				
EI	**D7**	**D6**	**D5**	**D4**	**D3**	**D2**	**D1**	**D0**	**Q2**	**Q1**	**Q0**	**GS**	**E0**
0	X	X	X	X	X	X	X	X					
1	0	0	0	0	0	0	0	0					
1	0	0	0	0	0	0	0	1					
1	0	0	0	0	0	0	1	0					
1	0	0	0	0	0	1	0	0					
1	0	0	0	0	1	0	0	0					
1	0	0	0	1	0	0	0	0					
1	0	0	1	0	0	0	0	0					
1	0	1	0	0	0	0	0	0					
1	1	0	0	0	0	0	0	0					

3-TO-8 LINE DECODER:

Pin Diagram:



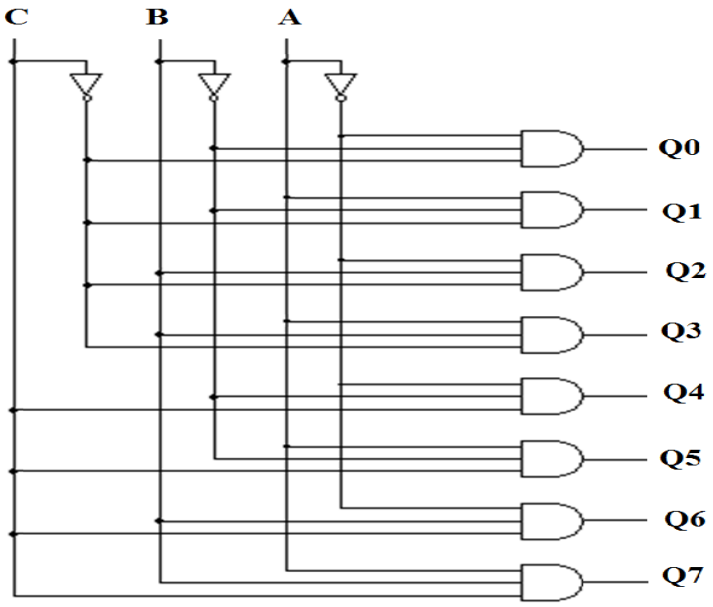
Truth Table:

INPUTS			OUTPUTS							
C	**B**	**A**	**Q7**	**Q6**	**Q5**	**Q4**	**Q3**	**Q2**	**Q1**	**Q0**
0	0	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	1	0	0	0	0	0	0	1	0	0
0	1	1	0	0	0	0	1	0	0	0
1	0	0	0	0	0	1	0	0	0	0
1	0	1	0	0	1	0	0	0	0	0
1	1	0	0	1	0	0	0	0	0	0
1	1	1	1	0	0	0	0	0	0	0

Q0 = C'.B'.A' Q1 = C'.B'.A Q2 = C'.B.A' Q3 = C'.B.A

Q4 = C.B'.A' Q5 = C.B'. A Q6 = C.B.A' Q7 = C.B.A

Internal Connection of IC 4555:



Output Table:

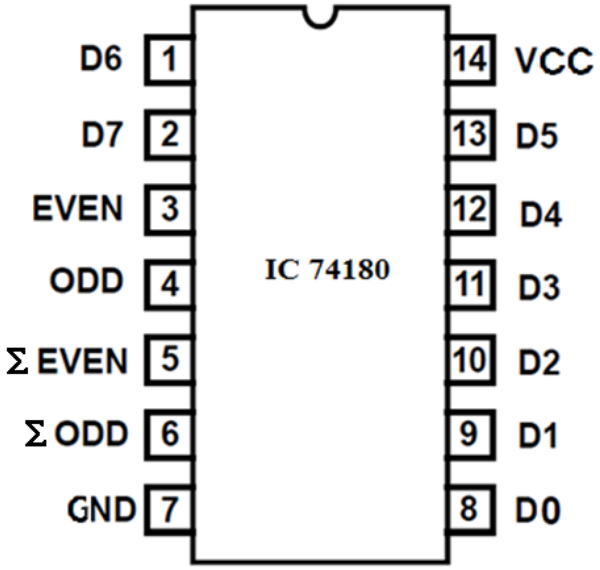
INPUTS			OUTPUTS							
C	B	A	Q7	Q6	Q5	Q4	Q3	Q2	Q1	Q0
0	0	0								
0	0	1								
0	1	0								
0	1	1								
1	0	0								
1	0	1								
1	1	0								
1	1	1								

Result

PREPARATION	**30**	
LAB PERFORMANCE	**30**	
REPORT	**40**	
TOTAL	**100**	
INITIAL OF FACULTY		

PARITY GENERATOR/CHECKER:

Pin Diagram:



Truth Table:

INPUTS			OUTPUTS	
EVEN	**ODD**	**∑ of 1's at D0 THRU D7**	**∑ EVEN**	**∑ ODD**
1	0	**EVEN**	1	0
		ODD	0	1
0	1	**EVEN**	0	1
		ODD	1	0
1	1	**X**	0	0
0	0	**X**	1	1

EVEN = 1, ODD = 0 ⟶ Parity checker
EVEN = 0, ODD = 1 ⟶ Parity generator

Output Table:

CONTROL INPUT		DATA BITS								OUTPUTS	
EVEN	**ODD**	**D7**	**D6**	**D5**	**D4**	**D3**	**D2**	**D1**	**D0**	**∑ EVEN**	**∑ ODD**
1	0	0	0	0	0	1	0	0	0		
1	0	0	1	0	0	1	0	0	0		
0	1	0	1	1	0	1	0	0	0		
0	1	0	1	1	0	1	1	0	0		

Expt. No.: 5	DESIGN AND IMPLEMENTATION OF PARITY GENERATOR/ CHECKER
Date:	

Aim

To design and implement Parity Generator/Checker using Digital IC 74180.

Components Required

S. No.	Description	Specification	Quantity
1.	Parity Generator/Checker	IC 74180	1
2.	Digital Trainer Kit	-	1
3.	Bread board	-	1
4.	Power Supply	5V	1
5.	Connecting wires	-	As Required

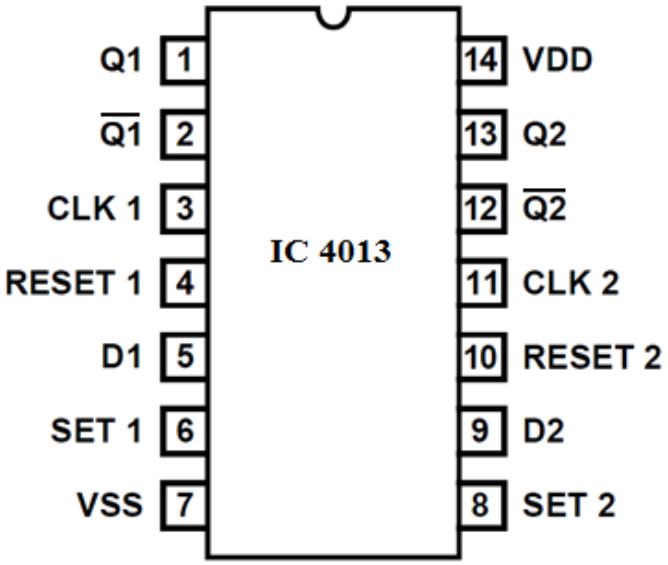
Procedure

(i) Make the connections as per the logic diagram.
(ii) Switch on the power supply.
(iii) Apply various combinations of inputs according to the truth table and observe condition of output LEDs.
(iv) Note down the outputs of Parity Generator/Checker for different combinations of input.

Result

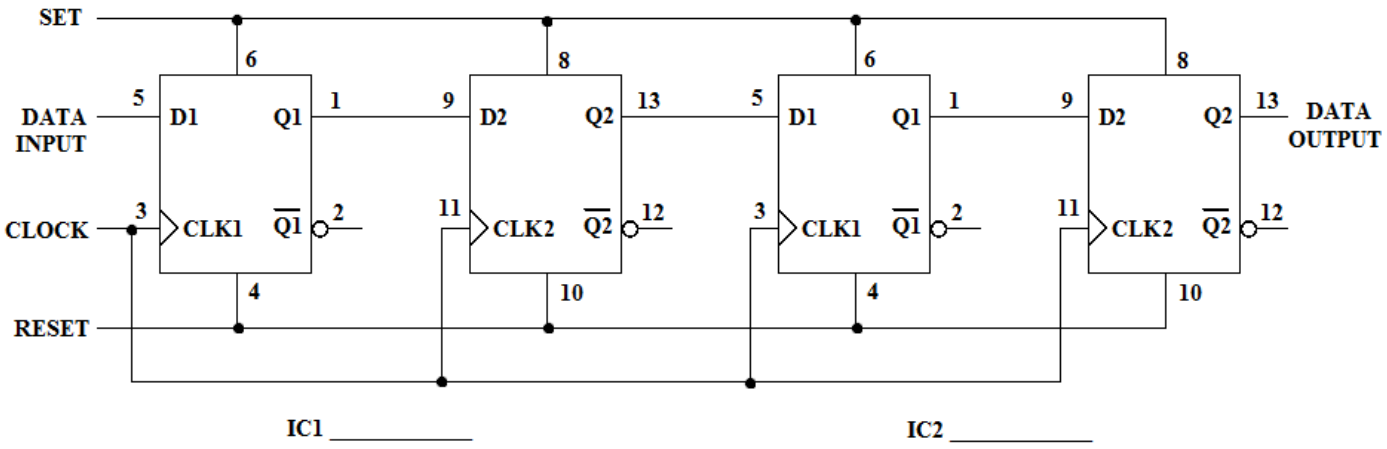
PREPARATION	**30**	
LAB PERFORMANCE	**30**	
REPORT	**40**	
TOTAL	**100**	
INITIAL OF FACULTY		

Pin Diagram:



SERIAL IN SERIAL OUT SHIFT REGISTER:

Logic Diagram:



Output Table:

CLK	DATA INPUT	DATA OUTPUT
0	1	
1	0	
2	1	
3	0	
4	0	
5	1	

Expt. No.: 6	DESIGN AND IMPLEMENTATION OF SHIFT REGISTERS
Date:	

Aim:

To design and implement Shift Registers using Digital IC 4013.

Components Required:

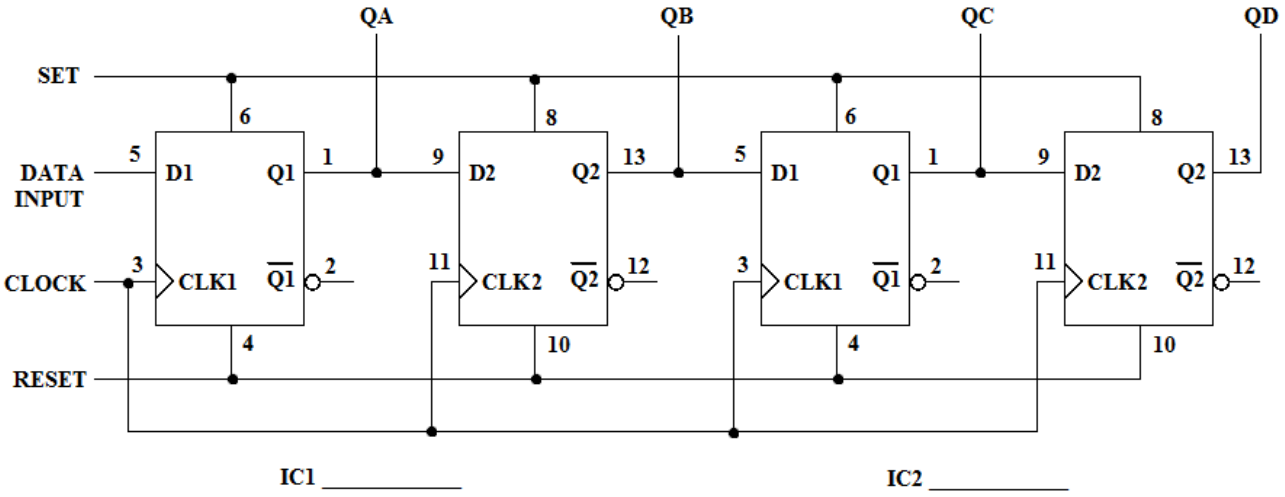
S. No.	Description	Specification	Quantity
1.	D Flip Flop	IC 4013	2
2.	Digital Trainer Kit	-	1
3.	Bread board	-	1
4.	Power Supply	5V	1
5.	Connecting wires	-	As Required

Procedure:

(i) Make the connections as per the logic diagram.
(ii) Switch on the power supply.
(iii) Apply various combinations of inputs according to the truth table and observe condition of output LEDs.
(iv) Observe the ouputs of Shift Registers by providing consecutive clock pulses.

SERIAL IN PARALLEL OUT SHIFT REGISTER:

Logic Diagram:



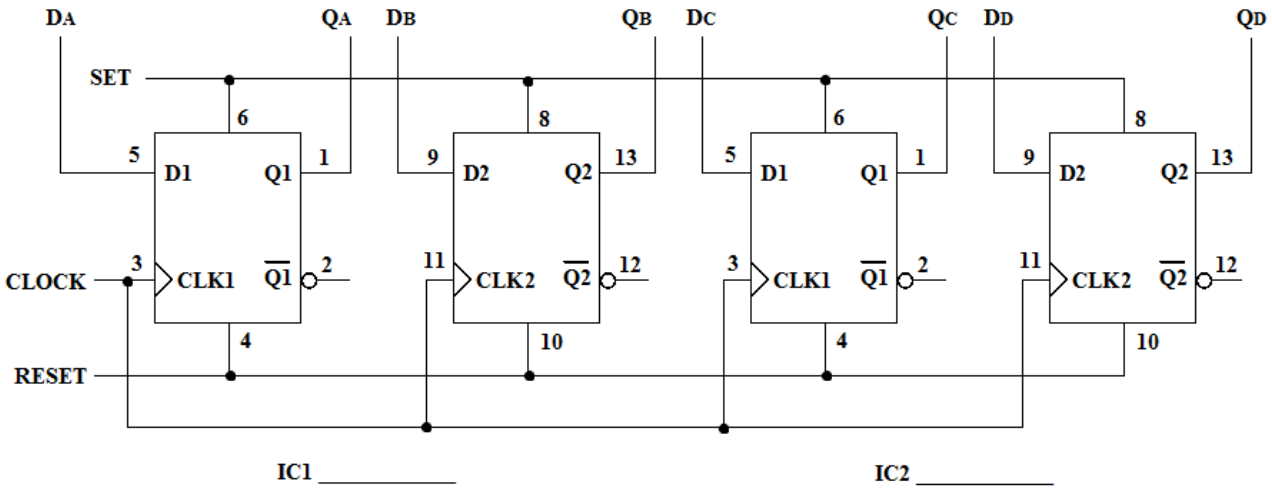
DATA OUTPUTS : QA, QB, QC,QD

Output Table:

CLK	DATA INPUT	DATA OUTPUTS			
		QA	QB	QC	QD
0	1				
1	1				
2	0				
3	0				
4	0				

PARALLEL IN PARALLEL OUT SHIFT REGISTER:

Logic Diagram:



DATA INPUTS : DA, DB, DC,DD DATA OUTPUTS : QA, QB, QC, QD

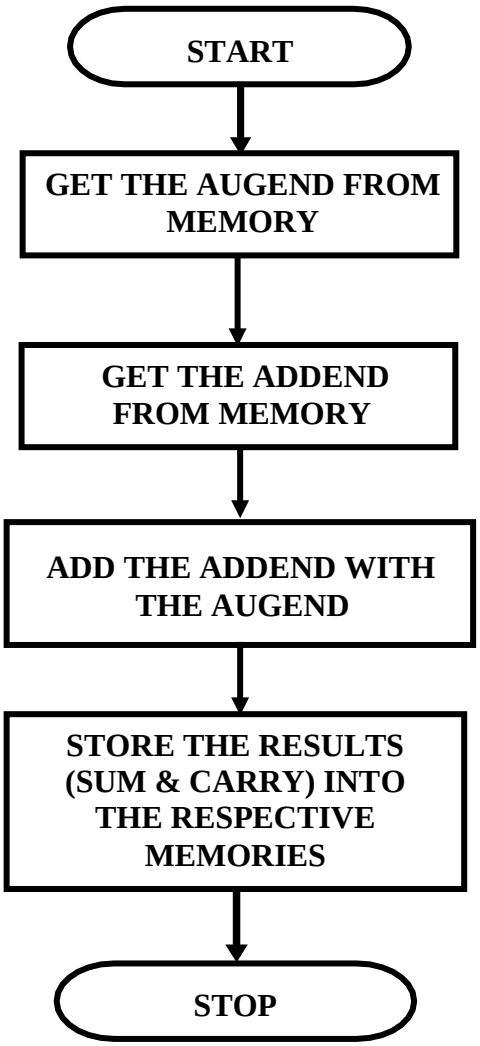
Output Table:

CLK	DATA INPUTS				DATA OUTPUTS			
	DA	DB	DC	DD	QA	QB	QC	QD
0	1	0	1	1				
1	1	0	0	1				
2	1	0	1	0				
3	1	1	0	0				
4	0	1	1	0				

Result:

PREPARATION	**30**	
LAB PERFORMANCE	**30**	
REPORT	**40**	
TOTAL	**100**	
INITIAL OF FACULTY		

FLOW CHART:


START
GET THE AUGEND FROM MEMORY
GET THE ADDEND FROM MEMORY
ADD THE ADDEND WITH THE AUGEND
STORE THE RESULTS (SUM & CARRY) INTO THE RESPECTIVE MEMORIES
STOP

Expt. No.:7a	SIMPLE 8085 ASSEMBLY LANGUAGE PROGRAM: 8 BIT ARITHMETIC OPERATIONS (ADDITION)
Date:	

Aim

To develop an 8085 Assembly Language Program (ALP) to test the addition of two 8-bit data available in the memory.

Problem Statement

1. Consider two 8-bit data X (Augend) and Y (Addend).
2. The Augend, X is in memory address 8500H and the Addend, Y is in memory address 8501H.
3. Find X+Y and store the results (Sum & Carry) in memory address 9000H and 9001H respectively.

Apparatus Required

S. No	Apparatus name	Quantity
1	8085 microcomputer trainer kit	1
2	Power Supply (5V)	1

Algorithm

Step1: Get the Augend from memory

Step2: Get the Addend from the next memory

Step3: Perform addition

Step4: Store the results of addition (Sum and Carry) to memory

Procedure

1. Connect +5V power supply to the 8085 microcomputer trainer kit.
2. Enter the opcode in the trainer kit using the matrix keyboard.
3. Check the opcode before execution of the program.
4. Enter the input data in the respective memories.
5. Execute the program.
6. Verify the result in the respective memories.

Assembly Language Program

Address	Opcode	Label	Mnemonics	Comments

Input & Output Data:

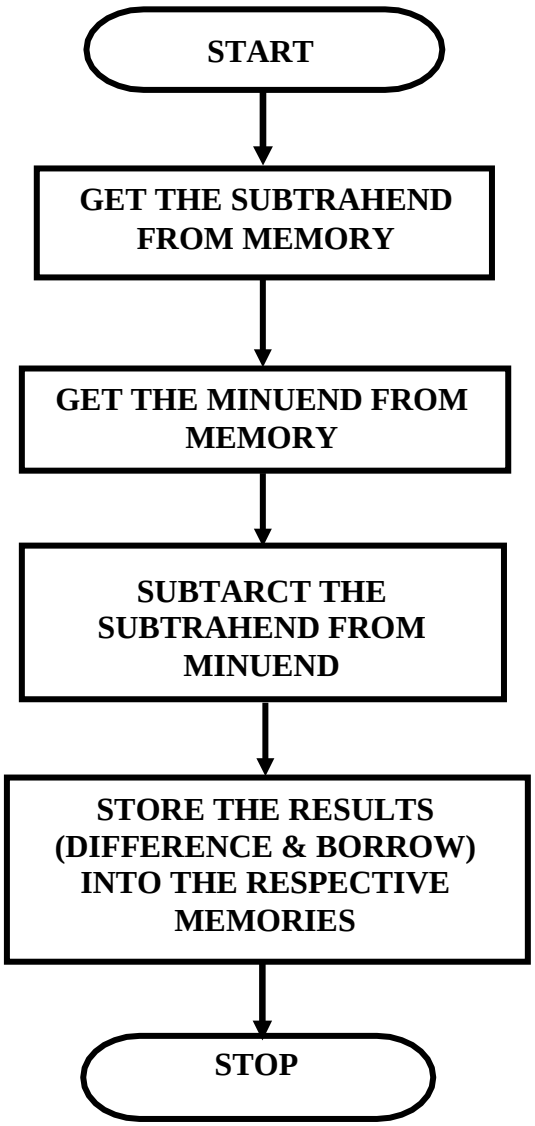
Input	
Address	**Data**
8500H (Augend)	
8501H (Addend)	

Output	
Address	**Data**
9000H (Sum)	
9001H (Carry)	

Result:

PREPARATION	**30**	
LAB PERFORMANCE	**30**	
REPORT	**40**	
TOTAL	**100**	
INITIAL OF FACULTY		

FLOW CHART:


START
GET THE SUBTRAHEND FROM MEMORY
GET THE MINUEND FROM MEMORY
SUBTARCT THE SUBTRAHEND FROM MINUEND
STORE THE RESULTS (DIFFERENCE & BORROW) INTO THE RESPECTIVE MEMORIES
STOP

Expt. No.:7b	SIMPLE 8085 ASSEMBLY LANGUAGE PROGRAM: 8 BIT ARITHMETIC OPERATIONS (SUBTRACTION)
Date:	

Aim

To develop an 8085 assembly language program (ALP) to test the subtraction of two 8-bit data available in the memory.

Problem Statement

1. Consider two 8 bit data X (Minuend) and Y (Subtrahend).
2. The Minuend, X is in memory address 8600H and Subtrahend, Y is in memory address 8601H.
3. Find X-Y and store the results (Difference & Borrow) in memory address 9000H and 9001H respectively.

Apparatus Required

S. No	Apparatus Name	Quantity
1	8085 microcomputer trainer kit	1
2	Power Supply (5V)	1

Algorithm

Step1: Get the Subtrahend from memory

Step2: Get the Minuend from memory

Step3: Perform subtraction

Step4: Store the results of subtraction (Difference and Borrow) to memory

Procedure

1. Connect +5V power supply to the 8085 microcomputer trainer kit.
2. Enter the opcode in the trainer kit using the matrix keyboard.
3. Check the opcode before execution of the program.
4. Enter the input data in the respective memories.
5. Execute the program.
6. Verify the result in the respective memories.

Assembly Language Program

Address	Opcode	Label	Mnemonics	Comments

Input & Output Data:

Input	
Address	**Data**
8600H (Minuend)	
8601H (Subtrahend)	

Output	
Address	**Data**
9000H (Difference)	
9001H (Borrow)	

Result

PREPARATION	**30**	
LAB PERFORMANCE	**30**	
REPORT	**40**	
TOTAL	**100**	
INITIAL OF FACULTY		

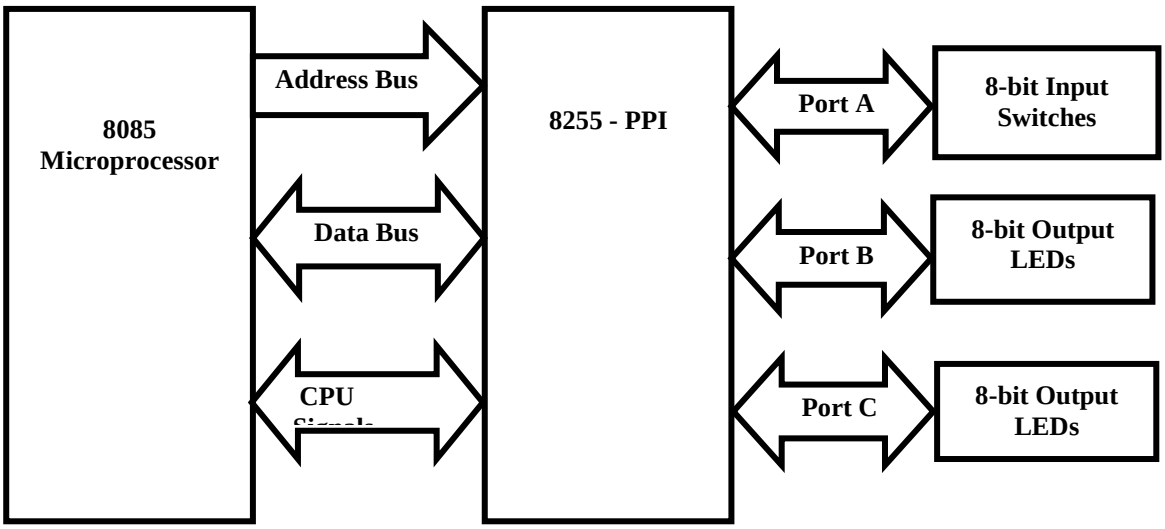
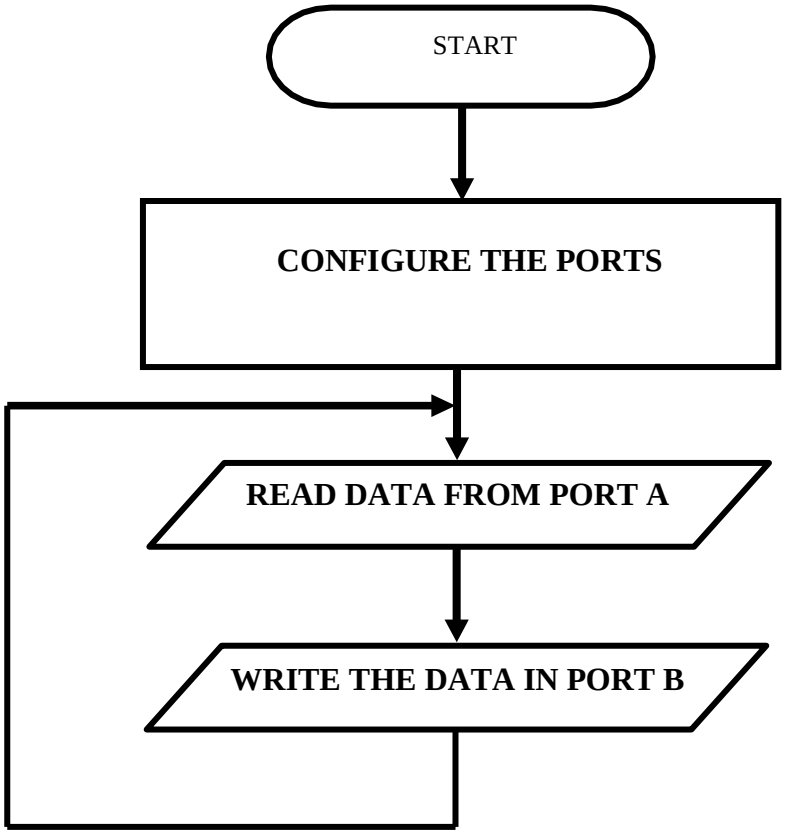


Fig 8.a: Block Diagram - 8255 with 8085 microprocessor

Flow Chart: (Mode 0)



Expt. No.: 8	**INTERFACING AND PROGRAMMING OF PROGRAMMABLE**
Date:	**PERIPHERAL INTERFACE (8255) WITH 8085 MICROPROCESSOR**

Aim

To develop an 8085 ALP to interface the 8255 - Programmable Peripheral Interface (PPI) with 8085 microprocessor for the following modes of operations:

a. Mode 0 b. BSR Mode

Problem statement

a. Mode 0:
1. Configure Port A as an input port in mode 0 and Port B as an output port in mode 0.
2. Read the 8-bit data from port A and write it in Port B.

b. BSR Mode:
1. Set Port C bits PC0 & PC7 using BSR mode.

Apparatus required:

S. No	Apparatus Name	Quantity
1	8085 microcomputer trainer kit	1
2	Power Supply (5V)	1
3	8255 – Interface Card	1
4	26 pin FRC Cable	2

Algorithm:

a. Mode 0:

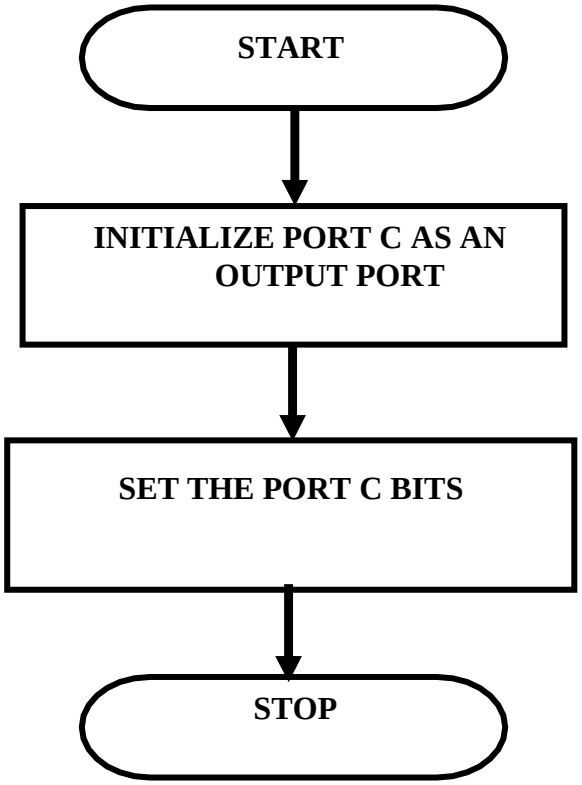
Step1: Initialize port A as input port in mode 0 and port B as an output port in mode 0
Step2: Read the 8-bit data from port A and Write it in port B
Step3: Go to step 2

Assembly language program:

a. Mode 0

ADDRESS	OPCODE	LABEL	MNEMONICS	COMMENTS

Flow Chart: (BSR Mode)



Algorithm:

b. BSR mode:

Step1: Configure Port C as an output port in mode 0

Step2: Set Port C bits using BSR mode

Assembly language program:

b. BSR mode

ADDRESS	OPCODE	LABEL	MNEMONICS	COMMENTS

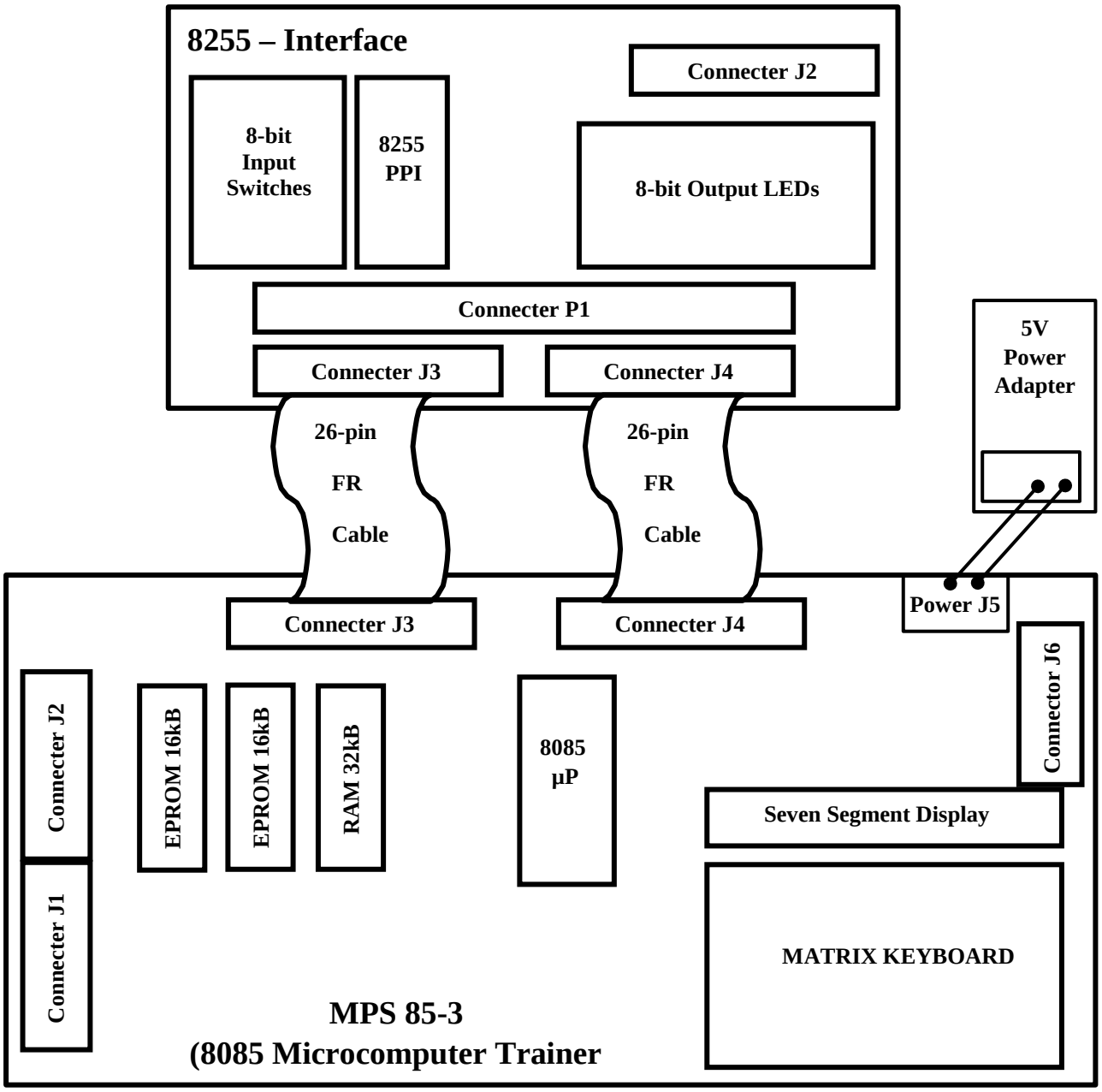


Fig 8.b: Interfacing diagram of 8255 with 8085 microprocessor

Procedure:

a. **Mode 0:**

1. Connect +5V power supply to 8085 microcomputer trainer kit
2. Connect the 8255 - Interface Card with 8085 microcomputer trainer kit as shown in fig 5.b.
3. Manually configure the Port setting (Port A as input and Port B as output) in the 8255 – Interface Card using 8-way DIP switches.
4. Enter the opcode in the 8085 microcomputer trainer kit using matrix keyboard.
5. Check the opcode before execution of the program.
6. Execute the program.
7. Provide Input data to Port A using 8-way DIP switch connected to port A and Verify the output displayed in the LEDs connected to Port B.

b. **BSR mode:**

1. Connect +5V power supply to 8085 microcomputer trainer kit
2. Connect the 8255 - Interface Card with 8085 microcomputer trainer kit as shown in fig 5.b.
3. Enter the opcode in the 8085 microcomputer trainer kit using matrix keyboard.
4. Check the opcode before execution of the program.
5. Execute the program.
6. Verify the output displayed in the LEDs connected to Port C.

Output

a. **For mode 0 Before execution:**

PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0

PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0

After execution:

PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0

PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0

b. **For BSR Mode**

Before execution:

PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0

After execution:

PC7	PC6	PC5	PC4	PC3	PC2	PC1	

Note: Fill "0" to indicate OFF state of LED and "1" to indicate ON state of LED in 8255 - Interface card.

RESULT

PREPARATION	**30**	
LAB PERFORMANCE	**30**	
REPORT	**40**	
TOTAL	**100**	
INITIAL OF FACULTY		

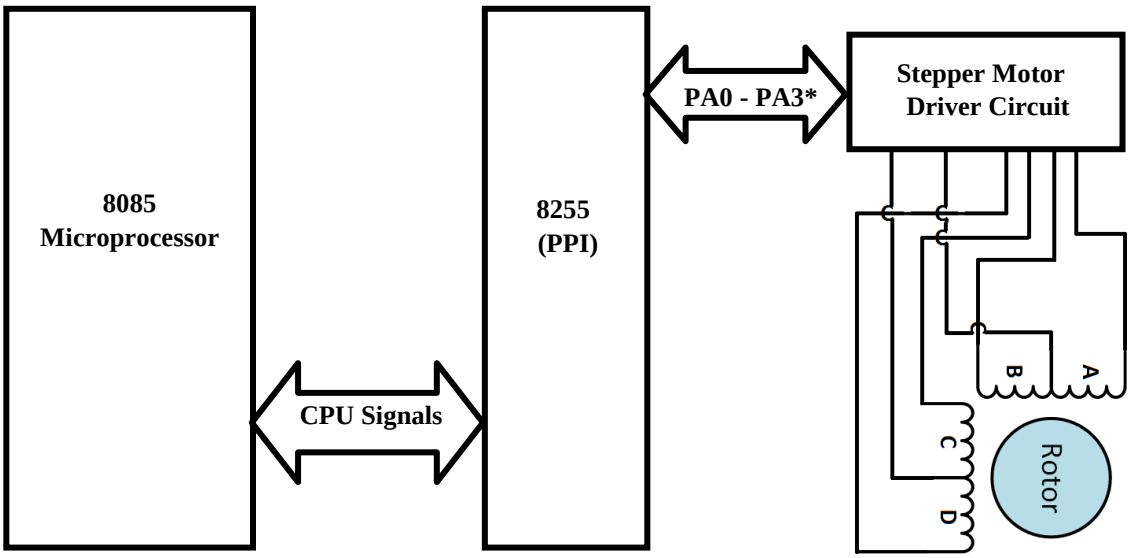


Fig 9.a: Block Diagram - Stepper motor with 8085 microprocessor through 8255

*The stepper motor coils are connected to the Port bits as shown in the table below

Coil	Port Bit
A	PA3
B	PA2
C	PA1
D	PA0

Expt. No.: 9	**INTERFACING OF STEPPER MOTOR WITH 8085 MICROPROCESSOR**
Date:	

Aim

To develop an 8085 ALP to interface the stepper motor with 8085 microprocessor and to rotate the stepper motor in the following sequences:

a. Full step sequence b. Half step sequence

Problem statement

a. Full step sequence:
Rotate the stepper motor in the clockwise direction in full step sequence.

b. Half step sequence:
Rotate the stepper motor in the clockwise direction in half step sequence.

Apparatus required

S. No	Apparatus Name	Quantity
1	8085 microprocessor trainer kit	1
2	Power Supply (+5V)	1
3	4 Phase Uni-polar Stepper Motor	1
4	Stepper Motor driver kit	1
5	26 pin FRC Cable	1
6	+5V Power Adapter for Stepper motor driver kit	1

Algorithm:

a. Full step sequence

Step1: Initialize Port A of 8255 as an output port
Step2: Write the bit pattern to port A to energize a coil **Step3:** Call delay routine
Step4: Rotate the bit pattern to energize the successive coil
Step5: Go to step 2

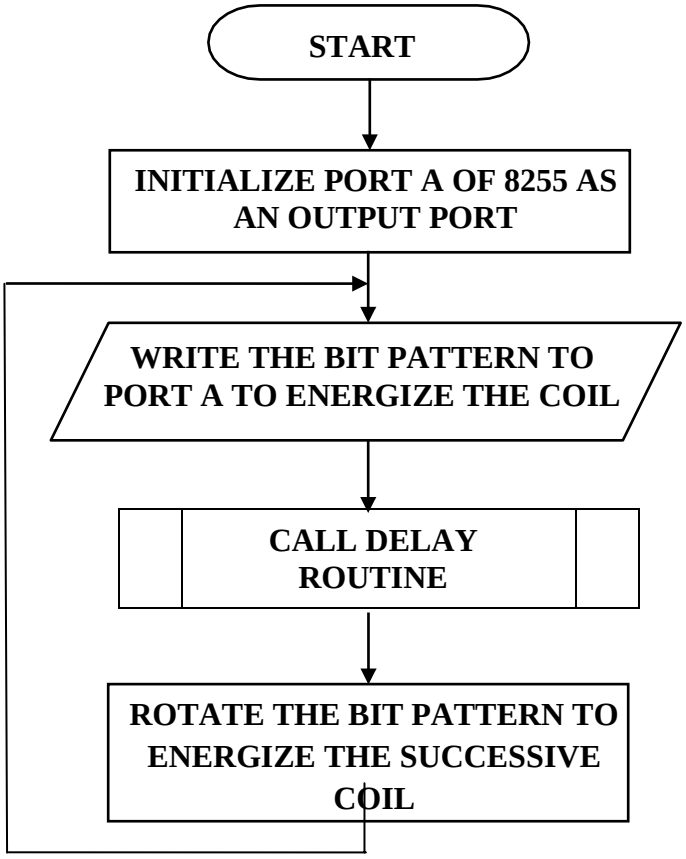
Algorithm:

b. Half step sequence

Step1: Initialize Port A of 8255 as an output port
Step2: Initialize the size (count) of the lookup table in a register
Step3: Point the starting address of the lookup table which has the bit pattern of half step sequence
Step4: Write the bit patterns available in the successive memory of lookup table to Port A with a delay
Step5: After writing all the bit pattern in the lookup table, go to step2

FLOWCHART:

FULL STEP IN CLOCKWISE DIRECTION



FULL STEP SEQUENCE PATTERN:

STEPS	COILS				D7	D6	D5	D4	D3	D2	D1	D0	HEX DATA
	A	B	C	D	-	-	-	-	PA3	PA2	PA1	PA0	
1	1	0	0	0	1	0	0	0	1	0	0	0	88H
2	0	1	0	0	0	1	0	0	0	1	0	0	44H
3	0	0	1	0	0	0	1	0	0	0	1	0	22H
4	0	0	0	1	0	0	0	1	0	0	0	1	11H

[OR]

STEPS	COILS				D7	D6	D5	D4	D3	D2	D1	D0	HEX DATA
	A	B	C	D	-	-	-	-	PA3	PA2	PA1	PA0	
1	1	1	0	0	1	1	0	0	1	1	0	0	CCH
2	0	1	1	0	0	1	1	0	0	1	1	0	66H
3	0	0	1	1	0	0	1	1	0	0	1	1	33H
4	1	0	0	1	1	0	0	1	1	0	0	1	99H

Assembly language program:

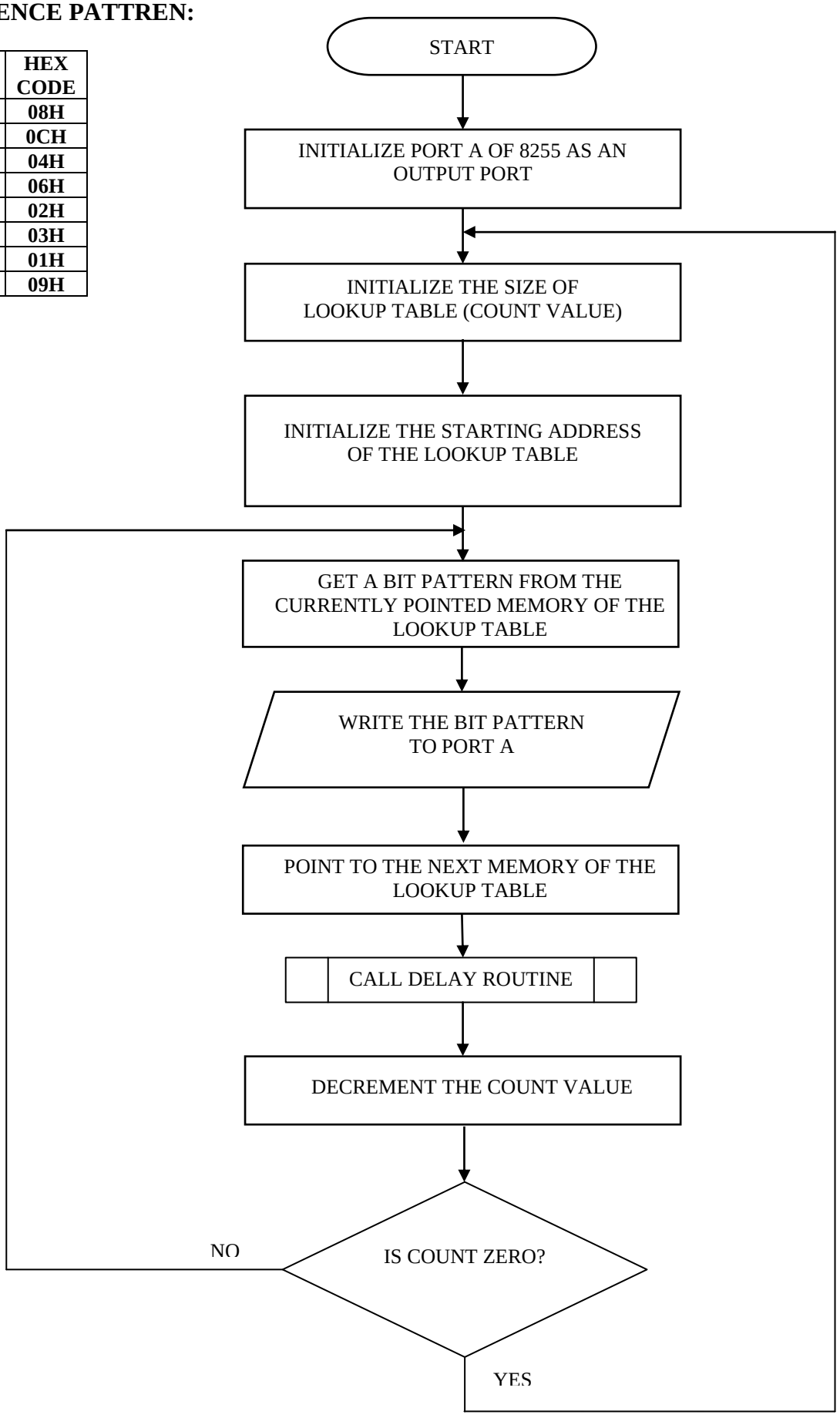
a. Full step sequence

ADDRESS	OPCODE	LABEL	MNEMONICS	COMMENTS

FLOWCHART: (HALF STEP IN CLOCKWISE DIRECTION)

HALF STEP SEQUENCE PATTREN:

STEP	A	B	C	D	HEX CODE
1	1	0	0	0	08H
2	1	1	0	0	0CH
3	0	1	0	0	04H
4	0	1	1	0	06H
5	0	0	1	0	02H
6	0	0	1	1	03H
7	0	0	0	1	01H
8	1	0	0	1	09H

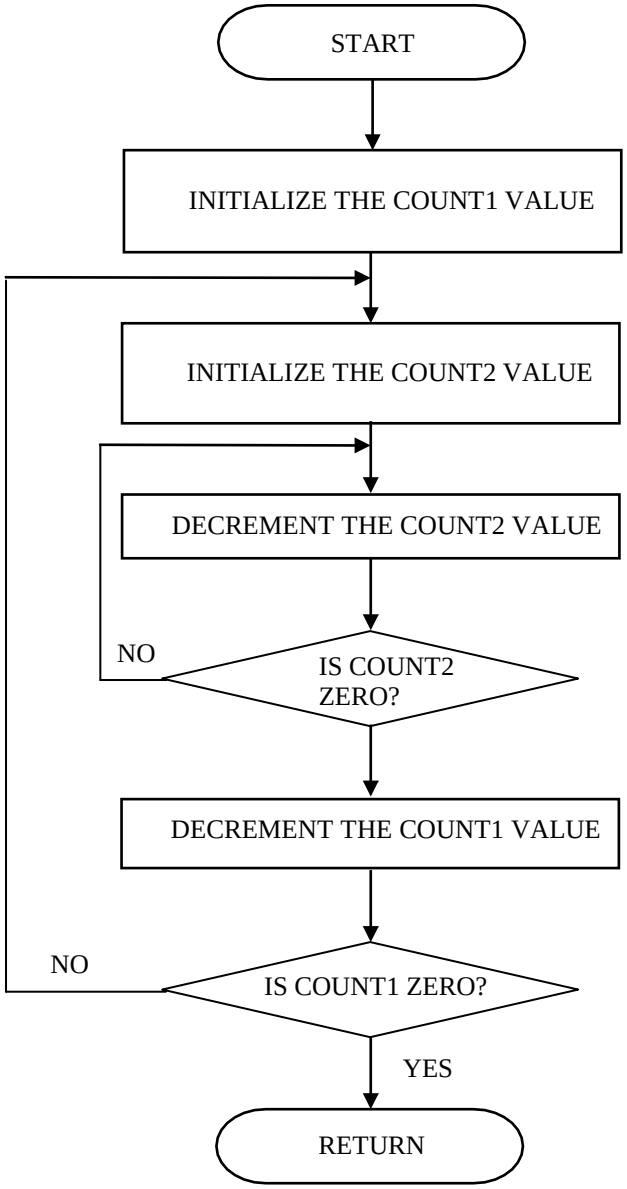


Assembly language program:

b. Half step sequence

ADDRESS	OPCODE	LABEL	MNEMONICS	COMMENTS

FLOWCHART: DELAY ROUTINE



Lookup table for half step sequence in Clockwise direction:

MEMORY LOCATIONS	DATA
	08H
	0CH
	04H
	06H
	02H
	03H
	01H
	09H

Algorithm:

Delay routine: (Delay ≈ 0.3sec)

Step1: Initialize the count1 value of delay routine
Step2: Initialize the count2 value of delay routine
Step3: Decrement the count2 value till it gets 00H
Step4: Decrement the count1 value and Go to step2 till the count1 value gets 00H
Step5: Return to the main program

Assembly language program:

Delay Routine

ADDRESS	OPCODE	LABEL	MNEMONICS	COMMENTS

Interfacing of Stepper motor with 8085 microprocessor

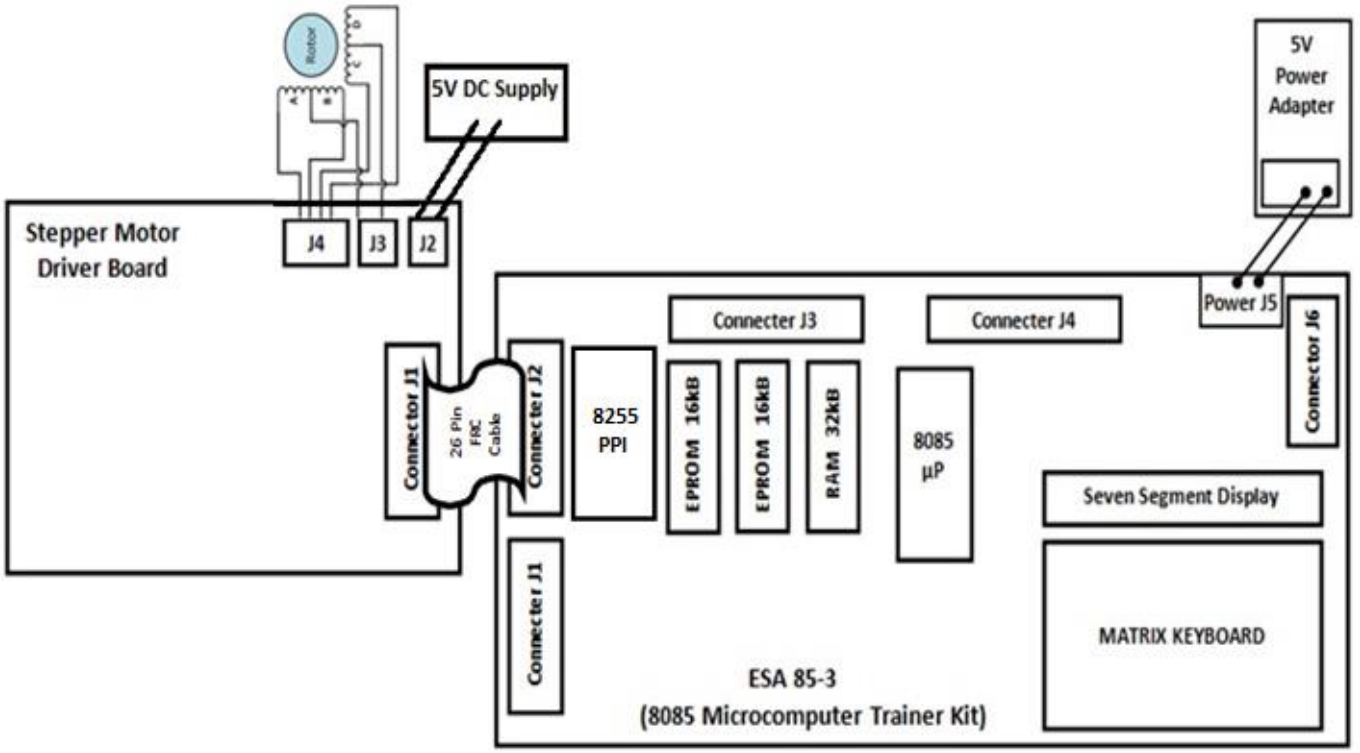


Fig 9.b: Interfacing of stepper motor with 8085 microprocessor using 8255 (Without 8255 IO Port Extension Board)

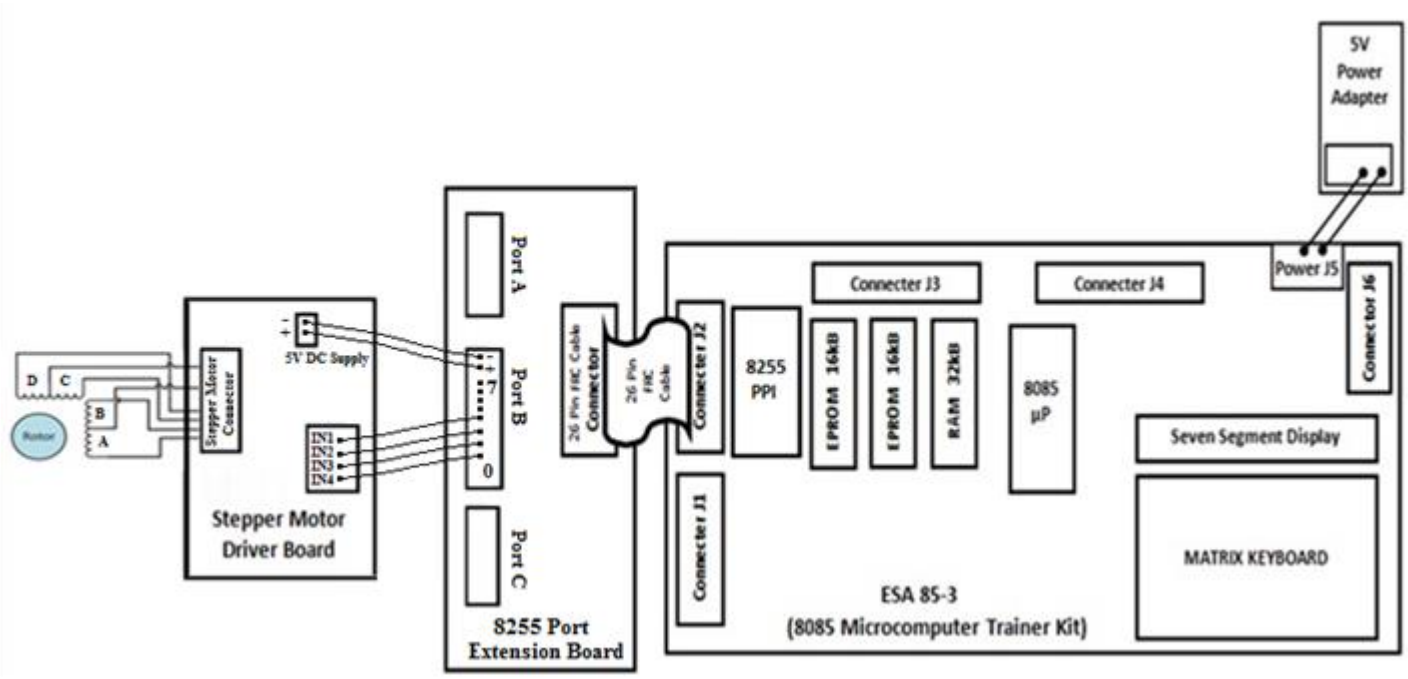


Fig 9.c: Interfacing of stepper motor with 8085 microprocessor using 8255 (With 8255 IO Port Extension Board)

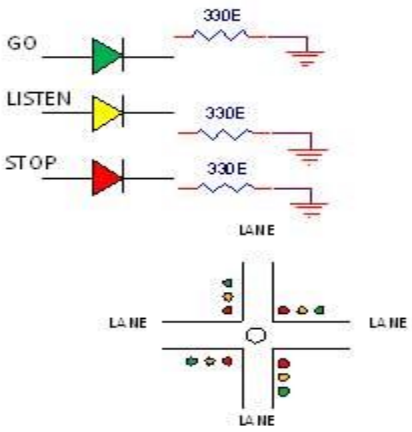
Procedure:

1. Connect +5V power supply to 8085 microcomputer trainer kit
2. Interface the Stepper motor with the driver board and interface the driver board with 8085 microprocessor through 8255 as shown in fig 9.b or 9.c
3. Enter the opcode in the 8085 microcomputer trainer kit using matrix keyboard
4. Check the opcode before execution of the program
5. Execute the program
6. Verify that the stepper motor starts rotating

RESULT

PREPARATION	**30**	
LAB PERFORMANCE	**30**	
REPORT	**40**	
TOTAL	**100**	
INITIAL OF FACULTY		

PIN ASSIGNMENT WITH 8085

LAN Direction	8255 PORT PINS	LEDs	Traffic Light Controller Card
SOUTH		GREEN	
		YELLOW	
		RED	
		PEDESTRIAN	
EAST		GREEN	
		YELLOW	
		RED	
		PEDESTRIAN	
NORTH		GREEN	
		YELLOW	
		RED	
		PEDESTRIAN	
WEST		GREEN	
		YELLOW	
		RED	
		PEDESTRIAN	
PWR		VCC	Supply form 8085 Microcomputer Trainer Kit at connector J2
		GND	

Make high to - LED On

Make low to – LED Off

CIRCUIT DIAGRAM TO INTERFACE TRAFFIC LIGHT WITH 8085

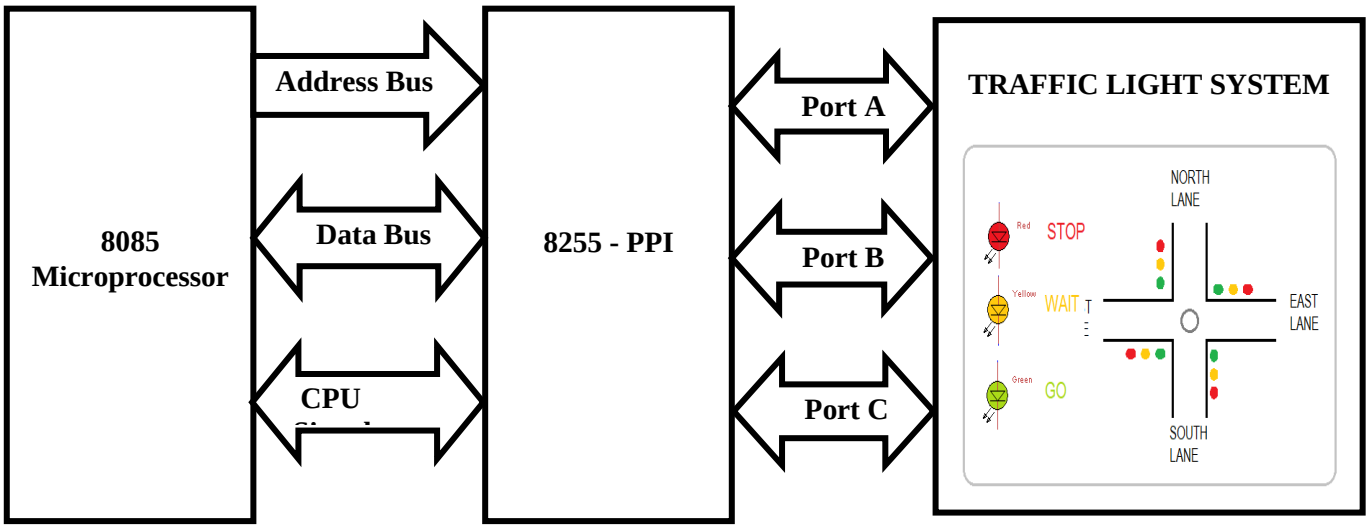


Fig 10: Interfacing of stepper motor with 8085 microprocessor using 8255

Expt. No.:10	**INTERFACING OF TRAFFIC LIGHT CONTROLLER WITH 8085 MICROPROCESSOR**
Date:	

Aim

To develop an 8085 ALP to interface traffic light controller and to generate a sequence of lighting to control the traffic.

Problem statement

Control the 4 lane Traffic light signal interfaced with 8085 microprocessor through 8255 by generating a proper sequence of lighting.

Apparatus required

S. No	Apparatus Name	Quantity
1	8085 microprocessor trainer kit	1
2	Traffic light controller module	1
3	+5V DC Power Adapter	1

Algorithm:

Step1: Initialize all ports of 8255 as output port

Step2: Write the bit pattern to generate the lighting sequence to control the traffic

Step3: Call delay routine

Step4: Go to step 2

Procedure:

1. Connect +5V power supply to 8085 microcomputer trainer kit
2. Interface the traffic light controller with 8085 microprocessor through 8255 as shown in fig 10
3. Enter the opcode in the 8085 microcomputer trainer kit using matrix keyboard
4. Check the opcode before execution of the program
5. Execute the program
6. Verify that the sequence of lighting generated to control the traffic

ADDRESS	OPCODE	LABEL	MNEMONICS	COMMENTS

ADDRESS	OPCODE	LABEL	MNEMONICS	COMMENTS

RESULT

PREPARATION	**30**	
LAB PERFORMANCE	**30**	
REPORT	**40**	
TOTAL	**100**	
INITIAL OF FACULTY		