

JSS SCIENCE AND TECHNOLOGY UNIVERSITY**Digital Electronics Circuits (EC37L)****Lab in-charge: Dr. Shankraiah****Course outcomes:** After the completion of laboratory the student will be able to,

1. Simplify, design and implement Boolean expression/half and full adders using basic/universal gates.
2. Design and implement the various combinational circuits using MSI components.
3. Implement and verify the truth tables of various flip-flops.
4. Design and implement the counters.
5. Design and implement the sequential circuits such as registers and sequence generator.

Lab experiments list:

1. Simplify the given Boolean expression and to realize them using logic gates/universal gates.
2. Design and implementation of half/full adder and subtracter using logic gates/universal gates.
3. Design and implementation of i) parallel adder/subtractor and ii) BCD-to- excess-3code converter and vice versa.
4. Design and implementation of code conversion from gray-to- binary and vice-versa.
5. Design and implementation of full adder/subtractor and code converters using i) multiplexer and ii) decoder IC's.
6. Design and implementation of one bit, two bit and magnitude comparators.
7. Implementation of i) priority encoders and ii) LED decoder driver circuit.
8. Implementation and verification of truth table for J-K flip-flop, Master-slave J-K flip-flop, D flip-flop and T flip-flop.
9. Design and implementation of Mod-N synchronous counter using J-K flip-flops.
10. Design and implementation of shift register to function as i) SISO, ii) SIPO, iii) PISO, iv) PIPO, v) shift left and vi) shift right operation.
11. Design and implementation of i) Ring counter and ii) Johnson counter using 4-bit shift register.
12. Design and implementation of sequence generator.

EXPERIMENT: 1

Simplify the given Boolean expression and to realize them using logic gates/universal gates.

AIM: To simplify the given expression and to realize it using basic gates and universal gates.

LEARNING OBJECTIVE:

- i) Simplify the Boolean expression and build the logic circuit.
- ii) For a given truth table derive the Boolean expressions and build the logic circuit to realize it.

COMPONENTS REQUIRED:

IC 7400, IC 7408, IC 7432, IC 7406, IC 7402, Patch cards and IC Trainer Kit.

THEORY:

A Karnaugh map (K-map) is a pictorial method used to minimize Boolean expressions without having to use Boolean algebra theorems and equation manipulations. A K-map can be thought of as a special version of a truth table. Using a K-map, expressions with two to four variables are easily minimized.

Canonical Forms (Normal Forms): Any Boolean function can be written in disjunctive normal form (sum of min-terms) or conjunctive normal form (product of max-terms). A Boolean function can be represented by a Karnaugh map in which each cell corresponds to a minterm. The cells are arranged in such a way that any two immediately adjacent cells correspond to two minterms of distance 1. There is more than one way to construct a map with this property.

Karnaugh Maps

For a function of two variables, say, $f(x, y)$,

	X'	X
y'	f(0,0)	f(1,0)
y	f(0,1)	f(1,1)

For a function of three variables, say, $f(x, y, z)$

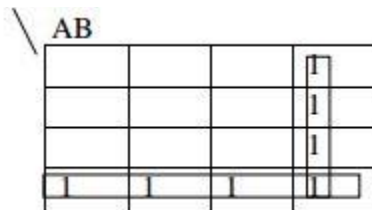
	X'Y'	X'Y	XY	XY'
Z'	f(0,0,0)	f(0,1,0)	f(1,1,0)	f(1,0,0)
Z	f(0,0,1)	f(0,1,1)	f(1,1,1)	f(1,0,1)

For a function of four variables: $f(w, x, y, z)$

	W'X	WX	WX'	WX
Y'Z'	0	4	12	8
Y'Z	1	5	13	9
YZ	3	7	15	11
YZ'	2	6	14	10

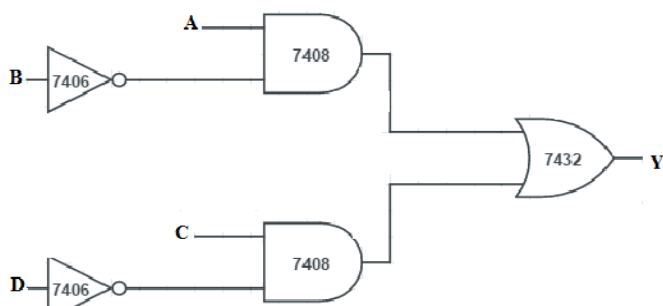
Realization of Boolean expression:

$$Y = A'B'CD' + A'BCD' + ABCD' + AB'CD' + AB'C'D' + AB'C'D + AB'CD$$

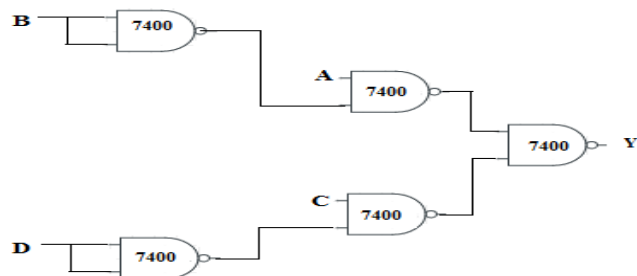


After simplifying using K-Map method we get

Realization using Basic gates



Realization using NAND gates

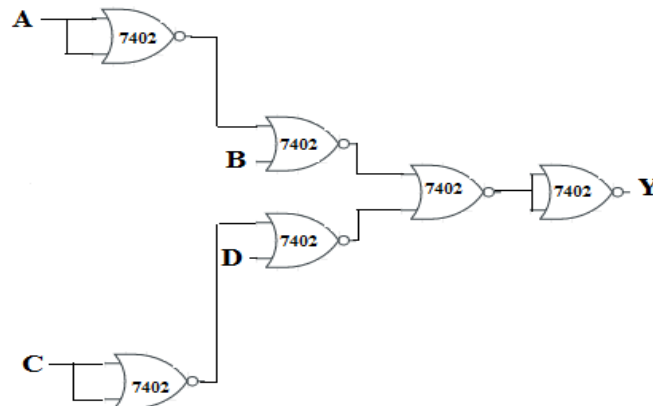


$$Y = AB' + CD'$$

INPUTS				OUTPUT
A	B	C	D	Y
0	0	0	0	0
0	0	0	1	0
0	0	1	0	1
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	1
0	1	1	1	0
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	1
1	1	0	0	0
1	1	0	1	0
1	1	1	0	1
1	1	1	1	0

Realization using NOR gates

Realization using NOR gates



2) For the given Truth Table, realize a logical circuit using basic gates and NAND gates

Inputs				Output
A	B	C	D	Y
0	0	0	0	1
0	0	0	1	1
0	0	1	0	0
0	0	1	1	0
0	1	0	0	1
0	1	0	1	1
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	1
1	1	1	0	0
1	1	1	1	1

PROCEDURE:

- Check the components for their working.
- Insert the appropriate IC into the IC base.
- Rig up the circuit as shown in the logic circuit diagram.
- Apply various input data to the logic circuit via the input logic switches.
- Note down the corresponding output and verify the truth table.

Note: Write the pin numbers of each gate and also write the intermediate expressions.

RESULT:

EXPERIMENT: 2**Design and implementation of half/full adder and subtractor using logic gates/universal gates****AIM:** To design and verify

- i. Half adder and Full adder
- ii. Half subtractor and Full subtractor using basic and NAND gates.

LEARNING OBJECTIVE:

- To design, realize and verify the adder and subtractor circuits using basic gates and universal gates.
- To design, realize and verify full adder using two half adders.
- To design, realize and verify a full subtractor using two half subtractors.

COMPONENTS REQUIRED: IC 7400, IC 7408, IC 7486, and IC 7432, Patch cards and IC Trainer Kit.**THEORY:**

Half-Adder: A combinational logic circuit that performs the addition of two data bits, A and B, is called a half-adder. Addition will result in two output bits; one of which is the sum bit, S, and the other is the carry bit, C. The Boolean functions describing the half-adder are:

$$S = A \oplus B$$

$$C = A B$$

Full-Adder: The half-adder does not take the carry bit from its previous stage into account. This carry bit from its previous stage is called carry-in bit. A combinational logic circuit that adds two data bits, A and B, and a carry-in bit, Cin, is called a full-adder. The Boolean functions describing the full-adder are:

$$S = (x \oplus y) \oplus \text{Cin}$$

$$C = xy + \text{Cin} (x \oplus y)$$

Half Subtractor: Subtracting a single-bit binary value B from another A (i.e. A -B) produces a difference bit D and a borrow out bit B-out. This operation is called half subtraction and the circuit to realize it is called a half subtractor. The Boolean functions describing the halfSubtractor are:

$$S = A \oplus B \quad C = A' B$$

Full Subtractor: Subtracting two single-bit binary values, B, Cin from a single-bit value A produces a difference bit D and a borrow out Br bit. This is called full subtraction. The Boolean functions describing the full-subtractor are:

$$D = (x \oplus y) \oplus \text{Cin}$$

$$\text{Br} = A' B + A' (\text{Cin}) + B (\text{Cin})$$

I. TO REALIZE HALF ADDER

TRUTH TABLE

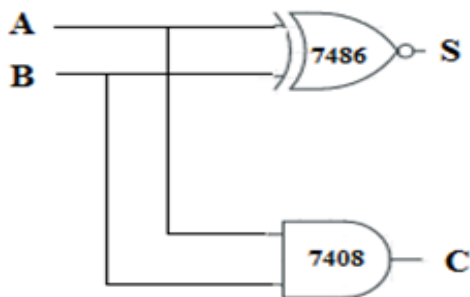
INPUTS		OUTPUTS	
A	B	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

BOOLEAN EXPRESSIONS:

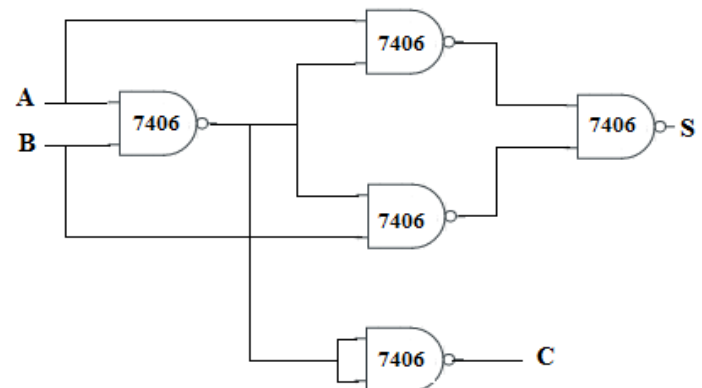
$$S = A \oplus B$$

$$C = A B$$

Basic gates



NAND gates



II. FULL ADDER

TRUTH TABLE

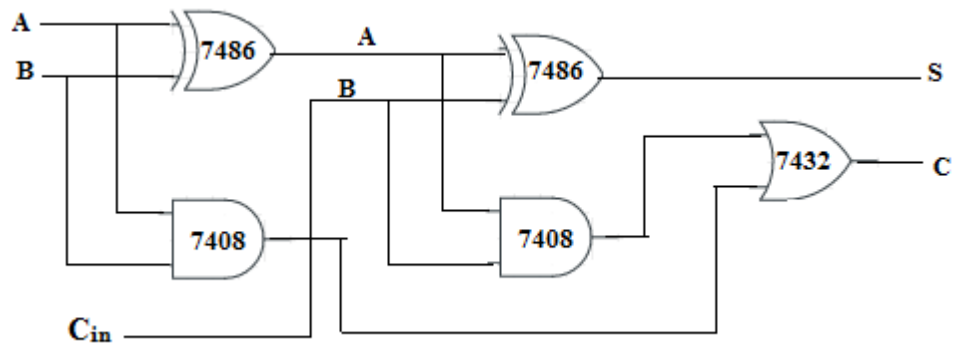
INPUTS			OUTPUTS	
A	B	Cin	S	C
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

BOOLEAN EXPRESSIONS:

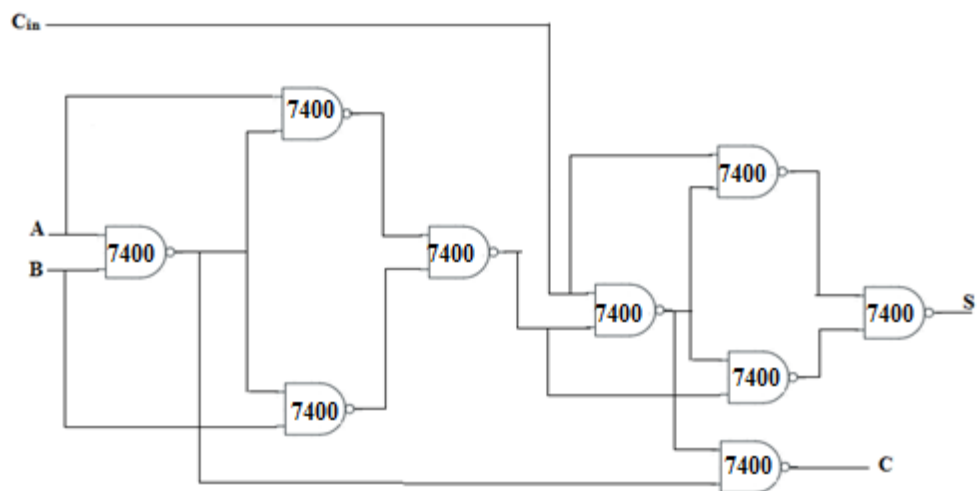
$$S = A \oplus B \oplus C$$

$$C = A B + B C_{in} + A C_{in}$$

BASIC GATES



i) NAND GATES



III. HALF SUBTRACTOR

TRUTH TABLE

INPUTS		OUTPUT	
A	B	D	Br
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

BOOLEAN EXPRESSIONS:

$$D = A \oplus B$$

$$Br = \bar{A}B$$

IV. FULL SUBTRACTOR

TRUTH TABLE

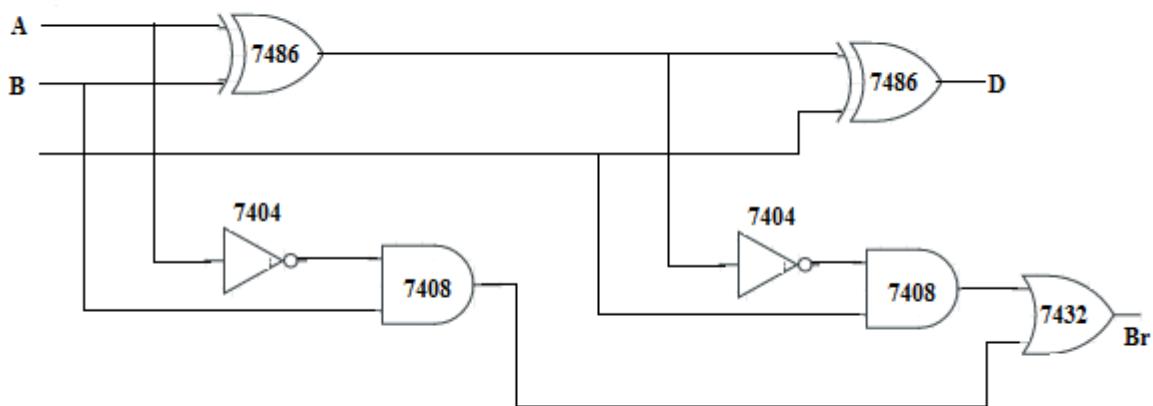
INPUTS			OUTPUTS	
A	B	Cin	D	Br
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

BOOLEAN EXPRESSIONS:

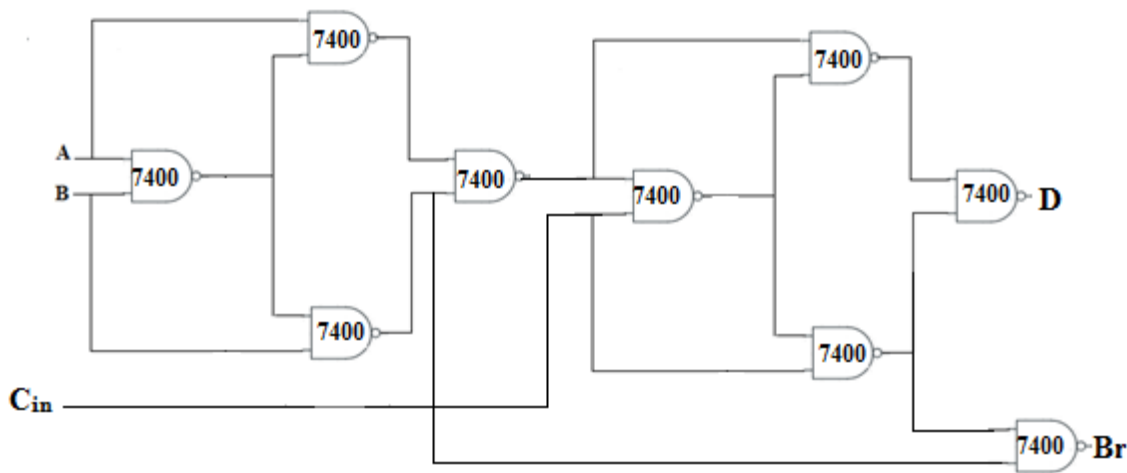
$$D = A \oplus B \oplus C$$

$$Br = \bar{A} B + B Cin + \bar{A} Cin$$

BASIC GATES



To realize the FULL subtractor using NAND Gates only



PROCEDURE:

- Check the components for their working
- Insert the appropriate IC into the IC base
- Make connections as shown in the circuit diagram.
- Verify the Truth Table and observe the outputs.

RESULT:

Experiment 3

PARALLEL ADDER / SUBTRACTOR AND CODE CONVERTORS

Aim: To design and set up the following:

1. 4 bit binary adder and Subtractor.
2. Code conversion BCD to Excess-3 and vice versa.

Learning objective:

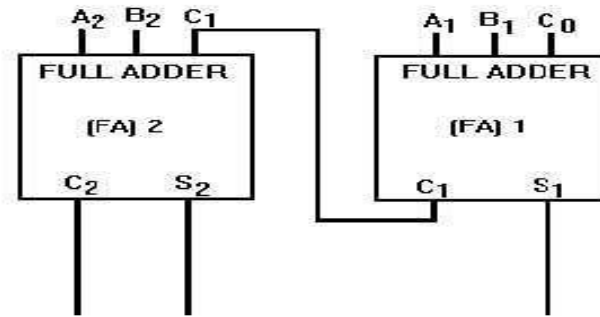
- To learn about IC 7483 and its internal structure.
- To realize a Subtractor using adder IC 7483.

Components required: IC 7483, IC 7486 trainer kit, patch cords.

Theory:

The Full adder can add single-digit binary numbers and carries. The largest sum that can be obtained using a full adder is 11_2 . Parallel adders can add multiple-digit numbers. If full adders are placed in parallel, we can add two- or four-digit numbers or any other size desired. Figure below uses standard symbols to show a parallel adder capable of adding two; two-digit binary numbers. The addend would be on A inputs, and the augend on the B inputs. For this explanation

we will assume there is no input to C_0 (carry from a previous circuit)



To add 10_2 (addend) and 01_2 (augend), the addend inputs will be 1 on A_2 and 0 on A_1 . The augend inputs will be 0 on B_2 and 1 on B_1 . Working from right to left, as we do in normal addition, let's calculate the outputs of each full adder. With A_1 at 0 and B_1 at 1, the output of adder1 will be a sum (S_1) of 1 with no carry (C_1). Since A_2 is 1 and B_2 is 0, we have a sum (S_2) of 1 with no carry (C_2) from adder1. To determine the sum, read the outputs (C_2 , S_2 , and S_1) from left to right. In this case, $C_2 = 0$, $S_2 = 1$, and $S_1 = 1$. The sum, then, of 10_2 and 01_2 is 011_2 .

Circuit implementation:

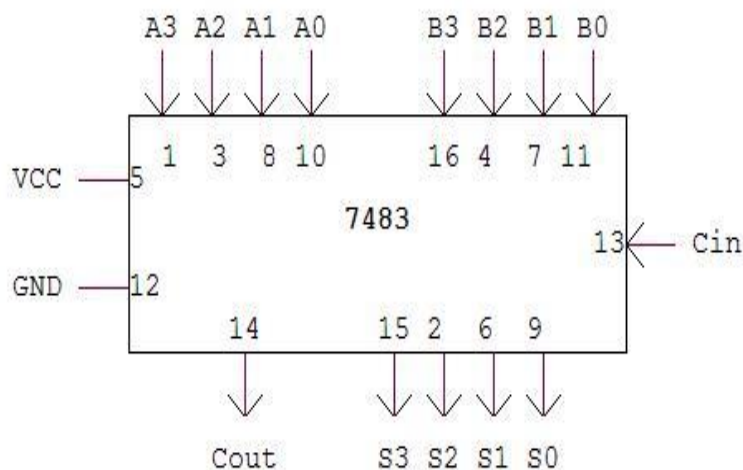
4 Bit Binary Adder:

An Example: $7+2=11$ (1001)

7 is realized at $A_3 A_2 A_1 A_0 = 0111$

2 is realized at $B_3 B_2 B_1 B_0 = 0010$

Sum = 1001



Truth table:

Inputs A				Inputs B				Sum			
A4	A3	A2	A1	B4	B3	B2	B1	S4	S3	S2	S1

4 Bit Binary Subtractor:

Subtraction is carried out by adding 2's complement of the subtrahend. Example: $8 - 3 = 5$ (0101)

8 is realized at $A_3 A_2 A_1 A_0 = 1000$

3 is realized at $B_3 B_2 B_1 B_0$ through X-OR gates $= 0011$

Output of X-OR gate is 1's complement of 3 $= 1100$

2's Complement can be obtained by adding $C_{in} = 1$

Therefore

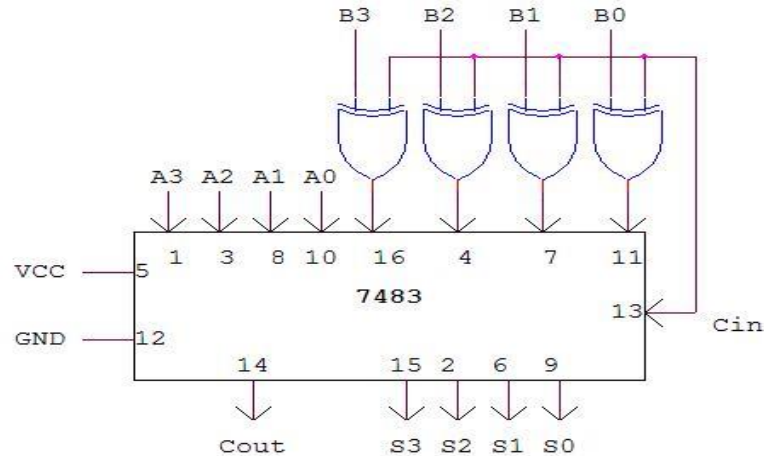
$C_{in} = 1$

$A_3 A_2 A_1 A_0 = 1000$

$B_3 B_2 B_1 B_0 = 1100$

$S_3 S_2 S_1 S_0 = 0101$

$C_{out} = 1$ (Ignored)



Truth table:

Inputs A				Inputs B				Difference			
A4	A3	A2	A1	B4	B3	B2	B1	S4	S3	S2	S1

Procedure:

- Check the components for their working.
- Insert the appropriate IC into the IC base.
- Rig up the circuit as shown in the logic circuit diagram.
- Apply various input data to the logic circuit via the input logic switches.
- Note down the corresponding output and verify the truth table.

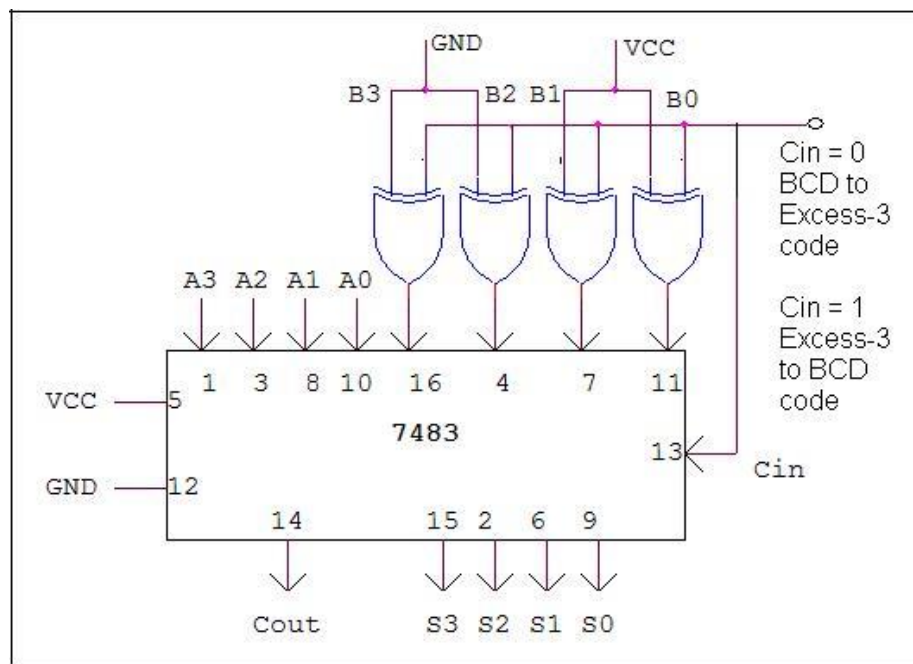
Result:**BCD to Excess 3 Code Conversion:****Learning objective:**

- To learn to realize BCD to Excess-3 code using adder IC 7483.
- To learn to realize Excess-3 to BCD Code using adder IC 7483.

Theory:

Code converter is a combinational circuit that translates the input code word into a new corresponding word. The excess-3 code digit is obtained by adding three to the corresponding BCD digit. To Construct a BCD-to-excess-3-code converter with a 4-bit adder feed BCD-code to the 4-bit adder as the first operand and then feed constant 3 as the second operand. The output is the corresponding excess-3 code.

To make it work as a excess-3 to BCD converter, we feed excess-3 code as the first operand and then feed 2's complement of 3 as the second operand. The output is the BCD code.

Circuit Implementation:

Truth table:

BCD code				Excess 3 Code			
A4	A3	A2	A1	S4	S3	S2	S1
0	0	0	0	0	0	1	1
0	0	0	1	0	1	0	0
0	0	1	0	0	1	0	1
0	0	1	1	0	1	1	0
0	1	0	0	0	1	1	1
0	1	0	1	1	0	0	0
0	1	1	0	1	0	0	1
0	1	1	1	1	0	1	0
1	0	0	0	1	0	1	1
1	0	0	1	1	1	0	0

Excess 3 to BCD Code Conversion:

Truth table:

Excess 3 Code				BCD code			
A4	A3	A2	A1	S4	S3	S2	S1
0	0	1	1	0	0	0	0
0	1	0	0	0	0	0	1
0	1	0	1	0	0	1	0
0	1	1	0	0	0	1	1
0	1	1	1	0	1	0	0
1	0	0	0	0	1	0	1
1	0	0	1	0	1	1	0
1	0	1	0	0	1	1	1
1	0	1	1	1	0	0	0
1	1	0	0	1	0	0	1

Procedure:

- Check the components for their working.
- Insert the appropriate IC into the IC base.
- Rig up the circuit as shown in the logic circuit diagram.
- Apply various input data to the logic circuit via the input logic switches.
- Note down the corresponding output and verify the truth table.

Result:

Experiment 4

BINARY TO GRAY CODE CONVERSION AND VICE VERSA

Aim: To realize binary to gray and gray to binary code converters.

Learning objective:

- To learn the importance of non-weighted code.
- To learn to generate gray code.

Components required: IC 7486, trainer kit, patch cords.

Theory:

The logical circuit which converts binary code to equivalent gray code is known as **binary to gray code converter**. The gray code is a non-weighted code. The successive gray code differs in one bit position only that means it is a unit distance code. It is also referred as cyclic code. It is not suitable for arithmetic operations. It is the most popular of the unit distance codes. It is also a reflective code. An n-bit Gray code can be obtained by reflecting an n-1 bit code about an axis after 2^{n-1} rows, and putting the MSB of 0 above the axis and the MSB of 1 below the axis.

Circuit implementation:

Binary to gray code converter:

0	0	1	1
0	0	1	1
0	0	1	1
0	0	1	1

$$G3 = B3$$

0	1	1	0
0	1	1	0
1	0	0	1
1	0	0	1

$$G1 = B1 \oplus B2$$

0	1	0	1
0	1	0	1
0	1	0	1
0	1	0	1

$$G2 = B3 \oplus B2$$

0	0	0	0
1	1	1	1
0	0	0	0
1	1	1	1

$$G0 = B1 \oplus B0$$

Truth table:

Binary inputs				Gray outputs			
B3	B2	B1	B0	G3	G2	G1	G0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	1
0	0	1	1	0	0	1	0
0	1	0	0	0	1	1	0
0	1	0	1	0	1	1	1
0	1	1	0	0	1	0	1
0	1	1	1	0	1	0	0
1	0	0	0	1	1	0	0
1	0	0	1	1	1	0	1
1	0	1	0	1	1	1	1
1	0	1	1	1	1	1	0
1	1	0	0	1	0	1	0
1	1	0	1	1	0	1	1
1	1	1	0	1	0	0	1
1	1	1	1	1	0	0	0

Gray to binary code converter:

0	0	1	1
0	0	1	1
0	0	1	1
0	0	1	1

$$B3 = G3$$

0	1	0	1
0	1	0	1
0	1	0	1
0	1	0	1

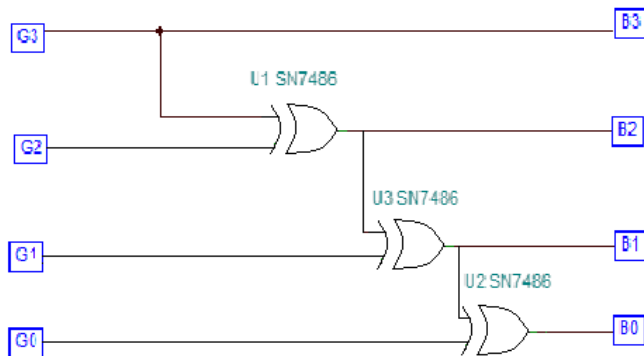
$$B2 = G3 \oplus G2$$

0	1	0	1
0	1	0	1
1	0	1	0
1	0	1	0

$$B1 = G3 \oplus G2 \oplus G1$$

0	1	0	1
1	0	1	0
0	1	0	1
1	0	1	0

$$B0 = G3 \oplus G2 \oplus G1 \oplus G0$$



Truth table:

Gray inputs				Binary outputs			
G3	G2	G1	G0	B3	B2	B1	B0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	1
0	0	1	1	0	0	1	0
0	1	0	0	0	1	1	1
0	1	0	1	0	1	1	0
0	1	1	0	0	1	0	0
0	1	1	1	0	1	0	1
1	0	0	0	1	1	1	1
1	0	0	1	1	1	1	0
1	0	1	0	1	1	0	0
1	0	1	1	1	1	0	1
1	1	0	0	1	0	0	0
1	1	0	1	1	0	0	1
1	1	1	0	1	0	1	1
1	1	1	1	1	0	1	0

Procedure:

- Check the components for their working.
- Insert the appropriate IC into the IC base.
- Rig up the circuit as shown in the logic circuit diagram.
- Apply various input data to the logic circuit via the input logic switches.
- Note down the corresponding output and verify the truth table.

Result:

EXPERIMENT: 5**MULTIPLEXER AND DEMULTIPLEXER**

AIM: To design and set up the following circuit

- 1) To verify the various functions of IC 74153(MUX) and IC 74139(DEMUX).
- 2) Half/Full Adder and Half/Full Subtractor using IC 74153 and IC 74139.

LEARNING OBJECTIVE:

- To learn about various applications of multiplexer and de-multiplexer
- To learn and understand the working of IC 74153 and IC 74139, and realize Half/Full Adder /Subtractor function using Multiplexer/De-multiplexer.

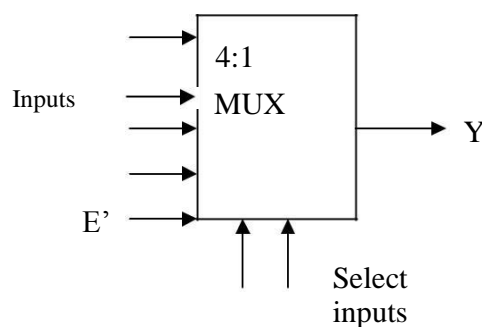
THEORY:

Multiplexers are very useful components in digital systems. They transfer a large number of information units over a smaller number of channels, (usually one channel) under the control of selection signals. Multiplexer means many to one. A multiplexer is a circuit with many inputs but only one output. By using control signals (select lines) we can select any input to the output. Multiplexer is also called as data selector because the output bit depends on the input data bit that is selected. The general multiplexer circuit has 2^n input signals, n control/select signals and 1 output signal.

De-multiplexers perform the opposite function of multiplexers. They transfer a small number of information units (usually one unit) over a larger number of channels under the control of selection signals. The general de-multiplexer circuit has 1 input signal, n control/select signals and 2^n output signals. De-multiplexer circuit can also be realized using a decoder circuit with enable.

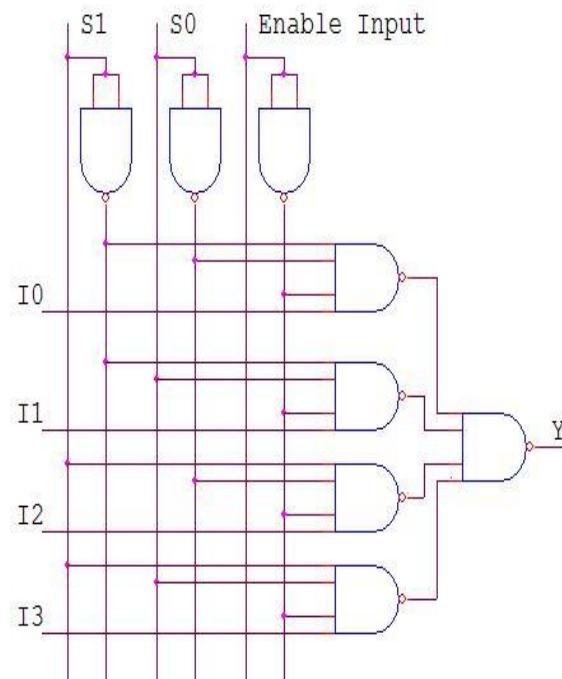
COMPONENTS REQUIRED:

IC 7400, IC 7410, IC 7420, IC 7404, IC 74153, IC 74139, Patch Cords & IC Trainer Kit.

i) 4:1 MULTIPLEXER

$$\text{Output } Y = E'S_1'S_0'I_0 + E'S_1'S_0'I_1 + E'S_1S_0'I_2 + E'S_1S_0'I_3$$

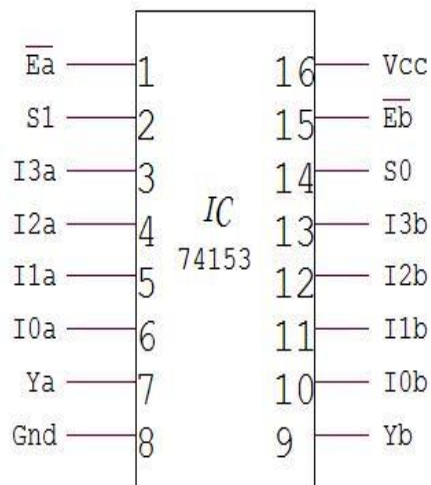
REALIZATION USING NAND GATES



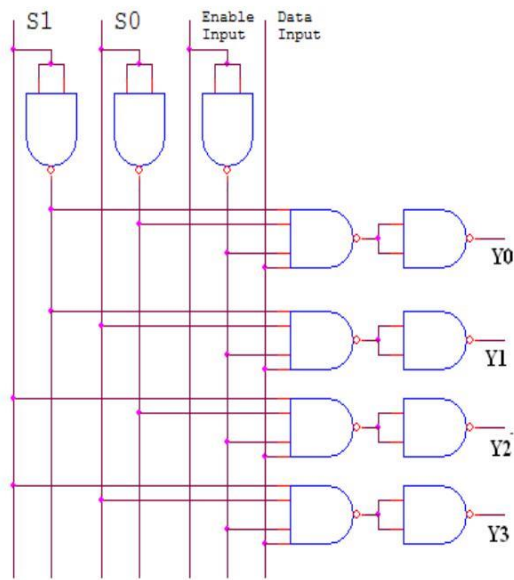
TRUTH TABLE

Select Inputs		Enable Input	Inputs				Out puts
S_1	S_0	E	I_0	I_1	I_2	I_3	Y
X	X	1	X	X	X	X	0
0	0	0	0	X	X	X	0
0	0	0	1	X	X	X	1
0	1	0	X	0	X	X	0
0	1	0	X	1	X	X	1
1	0	0	X	X	0	X	0
1	0	0	X	X	1	X	1
1	1	0	X	X	X	0	0
1	1	0	X	X	X	1	1

VERIFY IC 74153 MUX (DUAL 4:1 MULTIPLEXER)



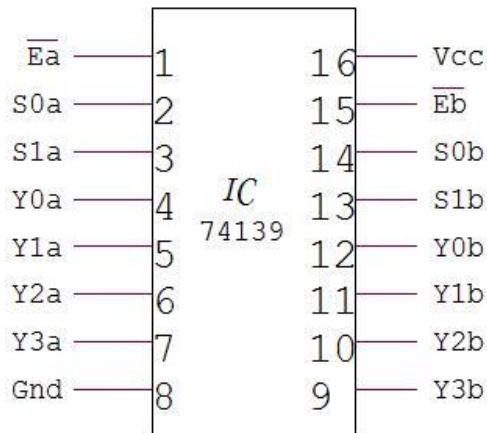
ii) DE-MUX USING NAND GATES



Enable Inputs	Data Input	Select Inputs		Outputs			
E	D	S ₁	S ₀	Y ₃	Y ₂	Y ₁	Y ₀
1	0	X	X	X	X	X	X
0	1	0	0	0	0	0	1
0	1	0	1	0	0	1	0
0	1	1	0	0	1	0	0
0	1	1	1	1	0	0	0

VERIFICATION OF IC 74139 (DEMUX)

TRUTH TABLE



Inputs			Outputs			
Ea	S ₁	S ₀	Y ₃	Y ₂	Y ₁	Y ₀
1	X	X	1	1	1	1
0	0	0	1	1	1	0
0	0	1	1	1	0	1
0	1	0	1	0	1	1
0	1	1	0	1	1	1

HALF ADDER USING MUX:

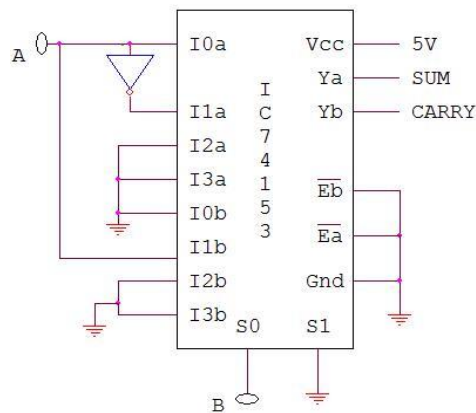
DESIGN:

SUM		CARRY	
I0	I1	I0	I1
0	1	0	1
2	3	2	3
A	A'	0	A

TRUTH TABLE

Input s		Outputs	
A	B	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

CIRCUIT:



FULL ADDER USING MUX:

DESIGN:

SUM			
I0	I1	I3	I3
0	1	2	3
4	5	6	7
A	A'	A'	A

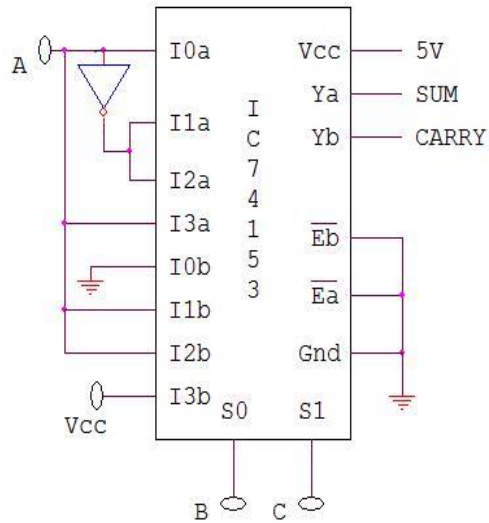
CARRY

I0	I1	I3	I3
0	1	2	3
4	5	6	7
0	A	A	1

TRUTH TABLE

Inputs			Outputs	
A	B	C	S	C
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

FULL ADDER CIRCUIT



HALF SUBTRACTOR USING MUX:

DESIGN:

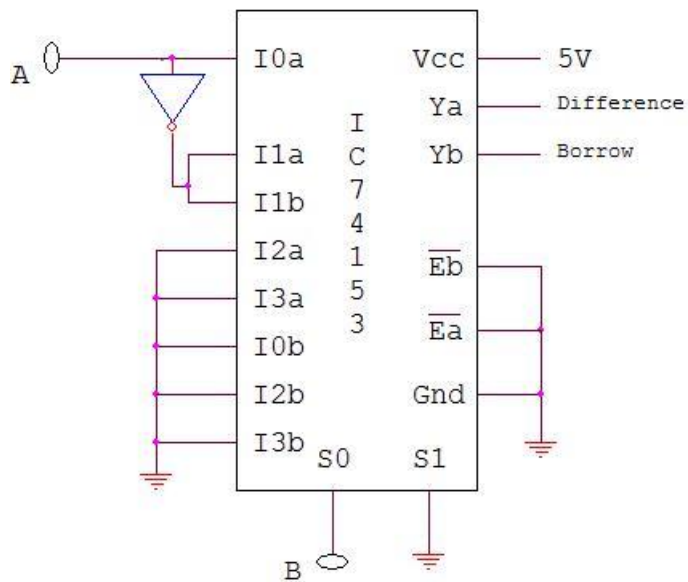
DIFFERENCE

I0	I1
0	1
2	3
A	A'

BORROW

I0	I1
0	1
2	3
0	A'

CIRCUIT:



TRUTH TABLE

Inputs		Outputs	
A	B	D	Br
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

FULL SUBTRACTOR USING MUX:

DESIGN:

DIFFERENCE

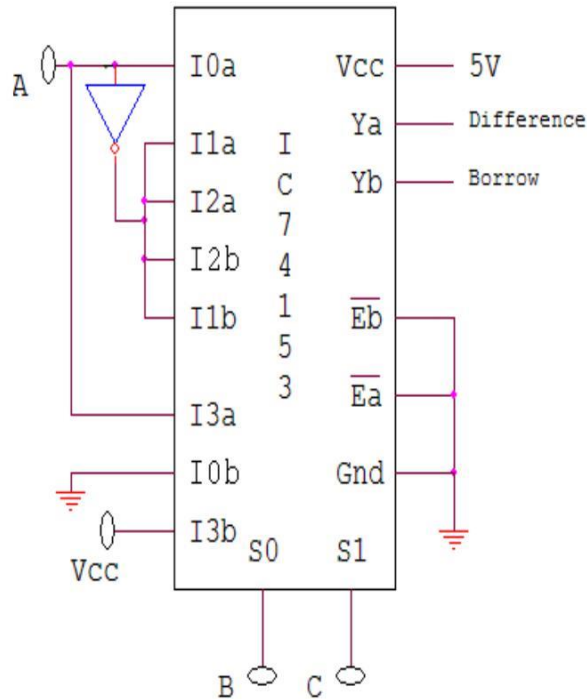
I0	I1	I2	I3
0	1	2	3
4	5	6	7
A	A'	A'	A

BORROW

I0	I1	I2	I3
0	1	2	3
4	5	6	7
0	A'	A'	1

TRUTH TABLE

Inputs			Outputs	
A	B	C	D	Br
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1



PROCEDURE:

- Check all the components for their working.
- Insert the appropriate IC into the IC base.
- Make connections as shown in the circuit diagram.
- Verify the Truth Table and observe the outputs.

RESULT:

EXPERIMENT: 6**COMPARATORS**

AIM: To realize One & Two Bit Comparator and study of 7485 magnitude comparator.

LEARNING OBJECTIVE:

- To learn about various applications of comparator
- To learn and understand the working of IC 7485 magnitude comparator To learn to realize 8-bit comparator using 4-bit comparator

THEORY:

Magnitude Comparator is a logical circuit, which compares two signals A and B and generates three logical outputs, whether $A > B$, $A = B$, or $A < B$. IC 7485 is a high speed 4-bit Magnitude comparator, which compares two 4-bit words. The $A = B$ Input must be held high for proper compare operation.

COMPONENTS REQUIRED:

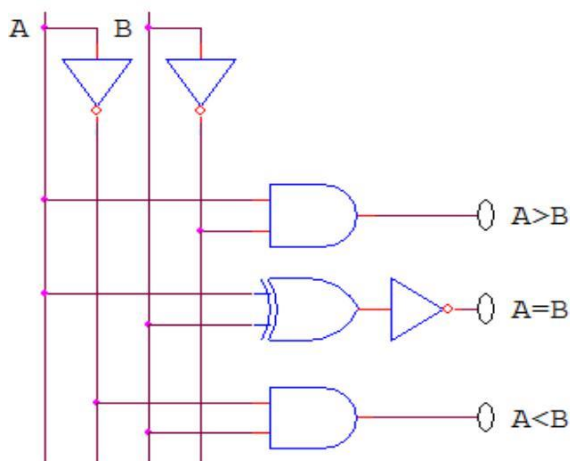
IC 7400, IC 7410, IC 7420, IC 7432, IC 7486, IC 7402, IC 7408, IC 7404, IC 7485, Patch Cords & IC Trainer Kit.

1) 1- BIT COMPARATOR**TRUTH TABLE**

$$A > B = A \bar{B}$$

$$A < B = \bar{A} B$$

$$A = B = \bar{A} \bar{B} + AB$$



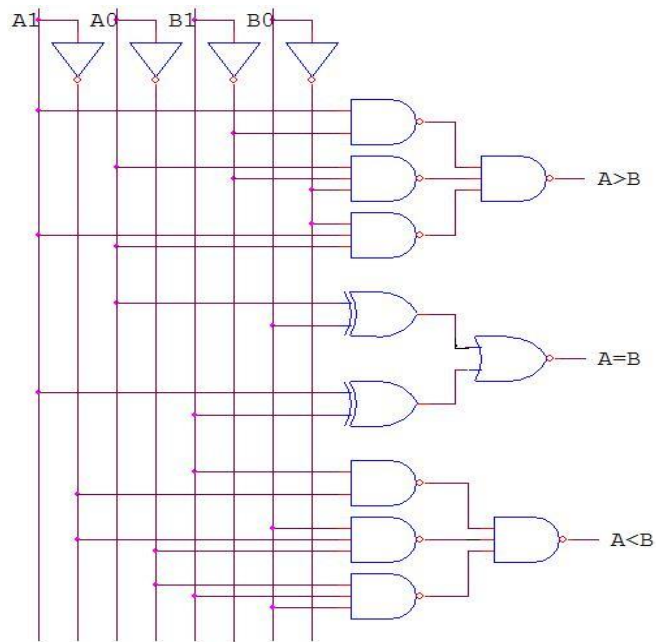
INPUTS		OUTPUTS		
A	B	A > B	A = B	A < B
0	0	0	1	0
0	1	0	0	1
1	0	1	0	0
1	1	0	1	0

2) 2- BIT COMPARATOR

$$(A > B) = A_1 \bar{B}_1 + A_0 \bar{B}_1 \bar{B}_0 + \bar{B}_0 A_1 A_0$$

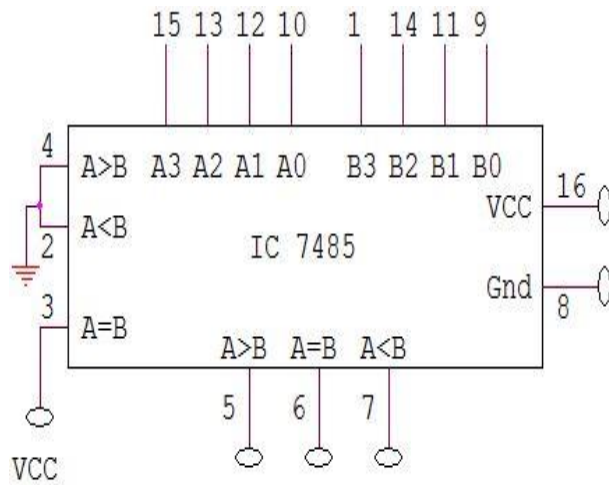
$$(A = B) = (A_0 \oplus B_0) (A_1 \oplus B_1)$$

$$(A < B) = B_1 A_1 + B_0 \bar{A}_1 \bar{A}_0 + \bar{A}_0 B_1 B_0$$

**TRUTH TABLE**

INPUTS				OUTPUTS		
A ₁	A ₀	B ₁	B ₀	A > B	A = B	A < B
0	0	0	0	0	1	0
0	0	0	1	0	0	1
0	0	1	0	0	0	1
0	0	1	1	0	0	1
0	1	0	0	1	0	0
0	1	0	1	0	1	0
0	1	1	0	0	0	1
0	1	1	1	0	0	1
1	0	0	0	1	0	0
1	0	0	1	1	0	0
1	0	1	0	0	1	0
1	0	1	1	0	0	1
1	1	0	0	1	0	0
1	1	0	1	1	0	0
1	1	1	0	1	0	0
1	1	1	1	0	1	0

3) TO COMPARE THE GIVEN DATA USING 7485 CHIP.



A				B				Result
A3	A2	A1	A0	B3	B2	B1	B0	
0	0	0	1	0	0	0	0	A > B
0	0	0	1	0	0	0	1	A = B
0	0	0	0	0	0	0	1	A < B

PROCEDURE:

- Check all the components for their working.
- Insert the appropriate IC into the IC base.
- Make connections as shown in the circuit diagram.
- Verify the Truth Table and observe the outputs.

RESULT:

EXPERIMENT 7 DECODER AND PRIORITY ENCODER

AIM:

Implement the decoder circuit to drive LED and also implement the priority encoder circuit.

LEARNING OBJECTIVE:

- i) To learn about working principle of decoder and encoder
- ii) To learn about various applications of decoder and encoder
- iii) To learn and understand the working of IC 7447, IC74147.

COMPONENTS REQUIRED:

IC-7447, 74147, LED, Digital IC trainer kit and Patch chords.

THEORY:

IC – 74147 is a decimal to BCD priority encoder. The input decimal number 0 through 9 are converted to its equivalent Binary Coded Decimal (BCD) number.

Encoders are inverse of decoders. That is, generally decoder has n -inputs and 2^n outputs and generally encoder has 2^n inputs and n outputs. E.g. When input $n = 3$ for decoder, $2^n = 8$ outputs, hence it is called 3 to 8 line decoder, and BCD to seven segment decoder has 4 – inputs and 10 (0-9) outputs, Octal – to – binary encoder has $2^3 = 8$ inputs and only 3 binary outputs, and decimal – to –BCD encoder has 10 (0-9) inputs and 4- BCD outputs, which has application in digital systems.

The Priority encoder is one, the output of which always corresponds to the highest order inputs.e.g. If the input is given simultaneously are 4 and 6, the output corresponds to 6 only, because 6 has higher priority than 4. One can verify this from the truth table of decimal – to BCD encoder.

When all the inputs are HIGH, active LOW outputs also HIGH and is equal to BCD 0, to obtain normal BCD output, use inverter as shown in the circuit diagram. When the last input i.e. Input (is LOW the output of encoder is BCD = 1 0 0 1, equivalent of decimal 9 (Inverted Output).

i)Decoder : IC-7447BCD to 7 – Segment Decoder:

IC - 7447: Pin diagram

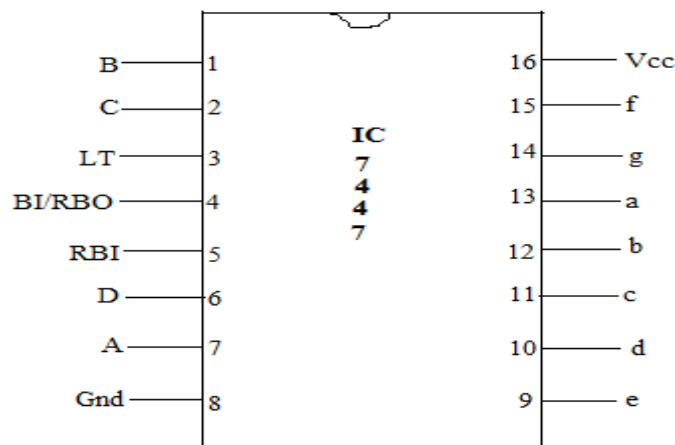


Fig (7.1) Pin diagram of 7447

Where

A,B,C,D \longrightarrow BCD inputs.

a-f \longrightarrow Active LOW outputs of decoder that drives LED.

IC-7447/46 is a BCD to 7 – Segment Decoder, this type of decoder takes BCD code as input and provides output to drive seven segment display devices in order to produce

decimal data.

Circuit Implementation:

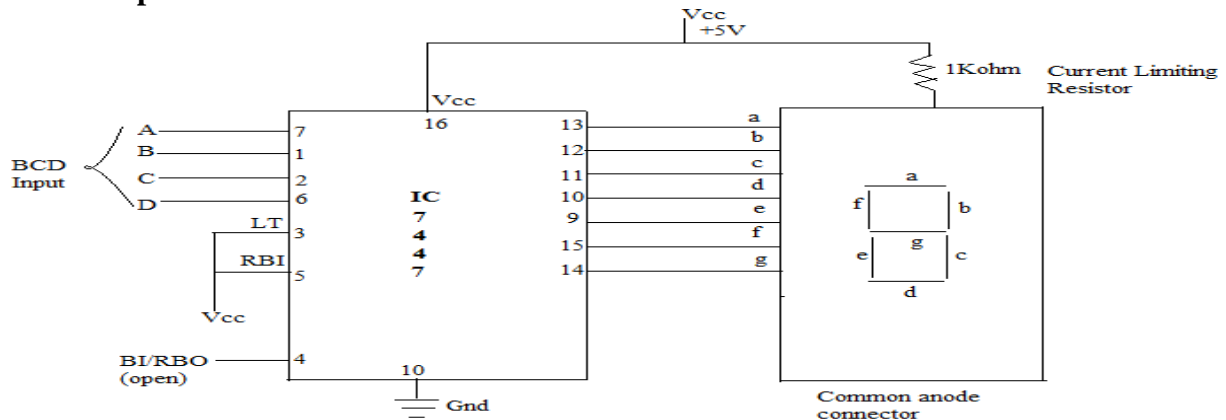


Fig (7.2) IC-7447 driving a 7 –segment LED display

Where

- LT → Lamp Test; Lamp test when LT = 0
- LT → Ripple blanking; when RBI = 0 generate all data except 0 (zero)
- RBO → Ripple blanking output; Used for cascading multiple LED's
- BI → Blanking input

For normal decoder operation connect LT = RBI = Vcc. And BR/RBO keep open.

The IC – 7447 decoder has 4 – inputs lines for BCD data and 7 – output lines to drive a seven segment display. The 7 output lines ‘a’ through ‘f’ of decoder are to be connected to ‘a’ through ‘f’ lines of LED display respectively.

Following table shows the IC – 7447 decoder output for 4 – bit BCD input to the common – anode connection mode of LED.

B C D Inputs				IC – 7447 decoder output							Decimal Number Display
D	C	B	A	a	b	c	d	e	f	g	
0	0	0	0	0	0	0	0	0	0	1	0
0	0	0	1	1	0	0	1	1	1	1	1
0	0	1	0	0	0	1	0	0	1	0	2
0	0	1	1	0	0	0	0	1	1	0	3
0	1	0	0	1	0	0	1	1	0	0	4
0	1	0	1	0	1	0	0	1	0	0	5
0	1	1	0	1	1	0	0	0	0	0	6
0	1	1	1	0	0	0	1	1	1	1	7
1	0	0	0	0	0	0	0	0	0	0	8
1	0	0	1	0	0	0	1	1	0	0	9

Seven segment displays has 7 Light Emitting Diodes (LED). Each diode or segment emits light when current flows through that diode.

For common – anode mode, all the seven diode anode is connected to the +5v, and cathodes are connected to the output of IC -7447 decoder as shown below, when cathode is at LOW (0) level. LED is forward biased and current flows through it, and it emits light (glows). Common anode connection is as follows.

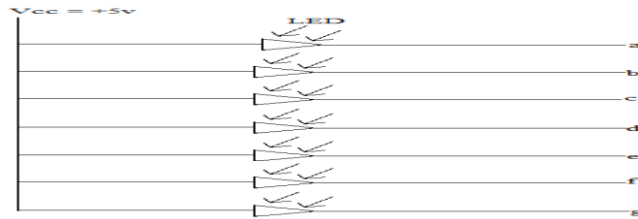


Fig (7.3) Common – anode connection for seven segments LED

For example to display 8, all the seven segment i.e. all seven LED's should glow. To make these LED's to glow (conduct), it must be forward biased. All the LED's conducts when $a = b = c = d = e = f = g = 0V$. See table

ii) Priority Encoder:

IC – 74147: Decimal - to - BCD Priority Encoder

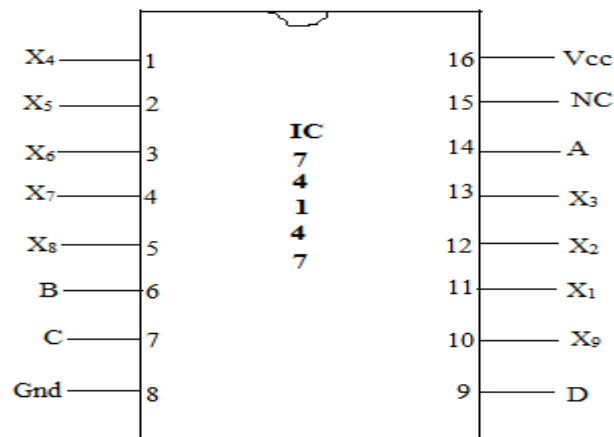


Fig (7.4) Pin Diagram of 74147

Where

- $X_1 - X_9 \longrightarrow$ Active LOW inputs
- $A, B, C, D \longrightarrow$ Active LOW outputs
- NC \longrightarrow no connection

Circuit Implementation:

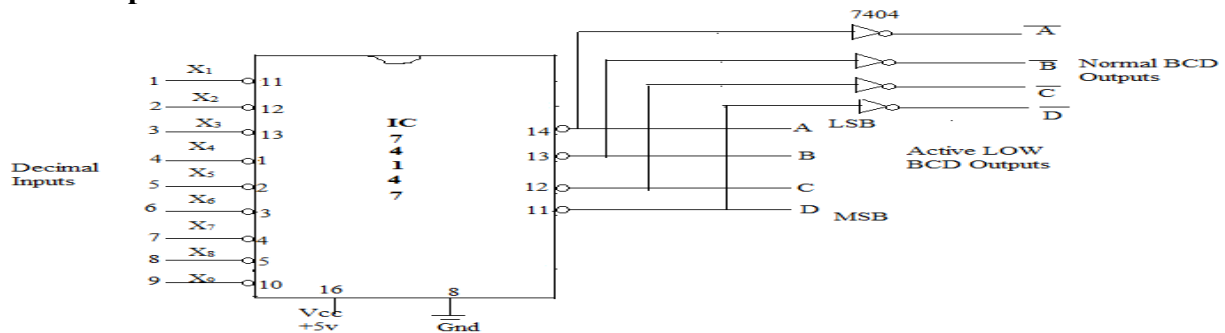


Fig (7.5) Decimal to BCD Converter

Truth Table:

Active –Low Decimal Inputs									Active – Low BCD outputs				Normal BCD outputs			
1	2	3	4	5	6	7	8	9	D	C	B	A				
1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0
0	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	1

X	0	1	1	1	1	1	1	1	1	1	1	0	1	0	0	1	0
X	X	0	1	1	1	1	1	1	1	1	1	0	0	0	0	1	1
X	X	X	0	1	1	1	1	1	1	1	0	1	1	0	1	0	0
X	X	X	X	0	1	1	1	1	1	1	0	1	0	0	1	0	1
X	X	X	X	X	0	1	1	1	1	1	0	0	1	0	1	1	0
X	X	X	X	X	X	0	1	1	1	1	0	0	0	0	1	1	1
X	X	X	X	X	X	X	0	1	1	0	1	1	1	1	0	0	0
X	X	X	X	X	X	X	X	X	0	0	1	1	0	1	0	0	1

PROCEDURE:

1. Connections are made as per the circuit diagram.
2. Apply inputs as shown in the truth table and verify the corresponding BCD outputs.
Apply the BCD inputs and verify the corresponding decimal outputs for decoder.
3. Switch off – power supply and disconnect the circuit.

RESULT:

EXPERIMENT 8

STUDY OF FLIPFLOPS

AIM: To implement the circuit and to verify the truth table of the following Flip-Flops.

LEARNING OBJECTIVE:

- i) To learn about various Flip-Flops.
- ii) To learn and understand the working of Master slave FF.

COMPONENTS REQUIRED:

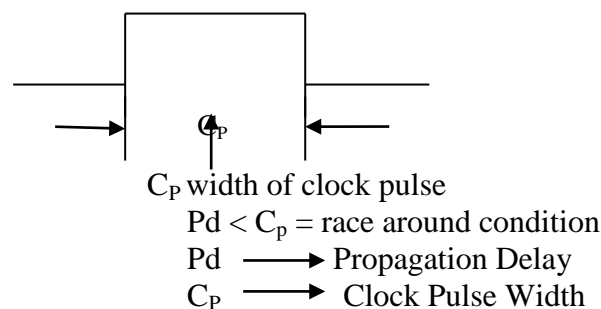
7485, 7408, 8411, 7421, 7432, digital IC Trainer Kit, Patch Chord.

THEORY:

Basically Flip-Flops are the bistable multivibrators that stores logic 1 and logic 0. Shift registers, memory, and counters are built by using Flip – Flops. Any complex sequential machines are build using Flip – Flops. Sequential circuit (machine) output depends on the present state and input applied at that instant.

Mealy Machine is one whose output depends on both the present state and the input. Moore machines one whose output depends only on the present state of the sequential circuit. Note that the truth table of J – K Flip – Flop is same as the Master – Slave.

J – K Flip Flop and they must be remain same because IC – 7476 is –ve edge triggered flip – flop and we know that race around condition is eliminated by edge triggered flip – flop. Another way of eliminating race around condition is by using Master – Slave J – K Flip – Flop. When $J = K = 1$ (logic HIGH), J – K Flip – Flop changes output many times for single clock pulse, it is Smaller than width of the clock pulse.



Race around condition is eliminated by using edge triggered clock pulse and using Master – Slave J - K Flip Flops.

i) Implementation of J.K.Flip-Flop

Design:

IC – 74LS76: Dual –ve edge triggered J – K Flip – Flop

Inputs			Outputs
Q	J	K	Q_{t+1}
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

Truth Table of JK Flip – Flop:

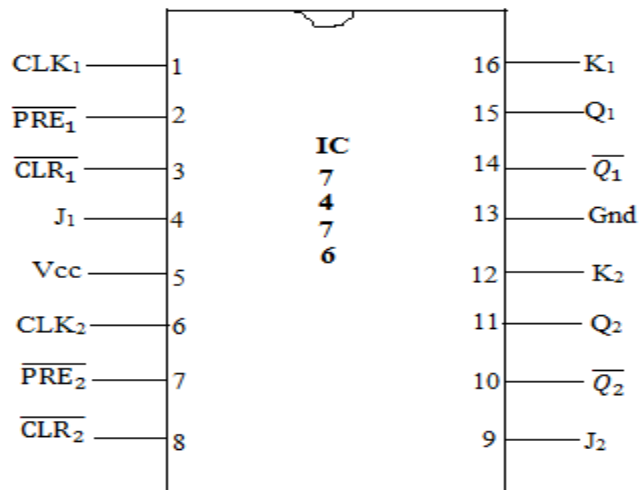


Fig (8.1) Pin diagram of 7476

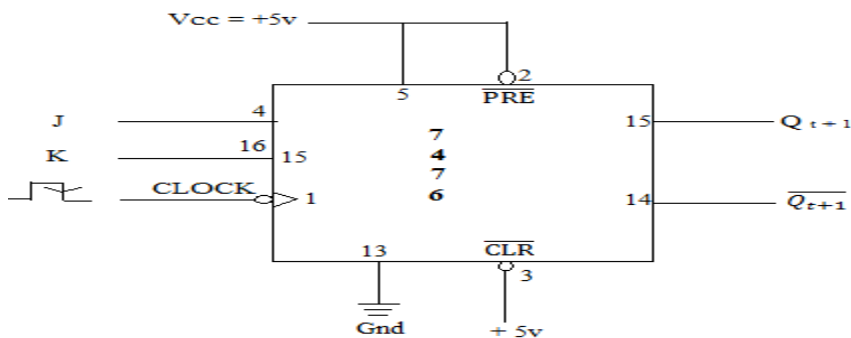
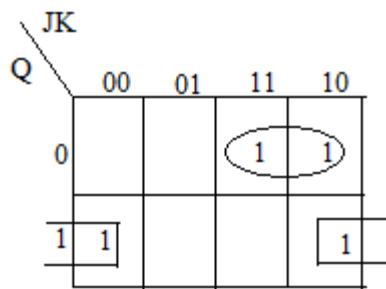


Fig (8.2) J – K Flip – Flop Circuit

Where

Q → Present State
 Q_{t+1} → Next State



Characteristic eqn $Q_{t+1} = J Q + K Q$

ii) Master Slave J K Flip – Flop:

IC – 74107: Dual – Master – Slave J-K Flip - Flop

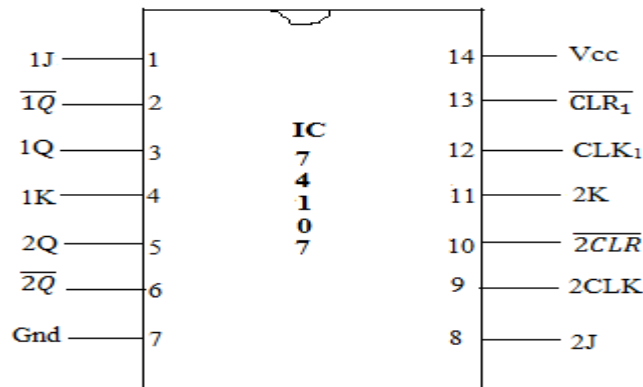


Fig (8.3) Pin diagram

Circuit Implementation:

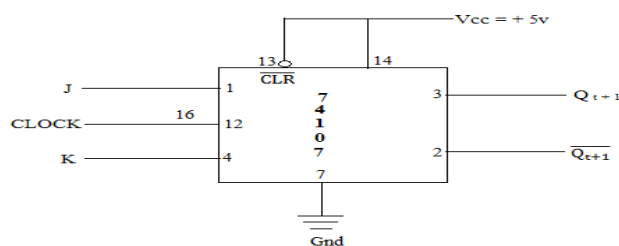
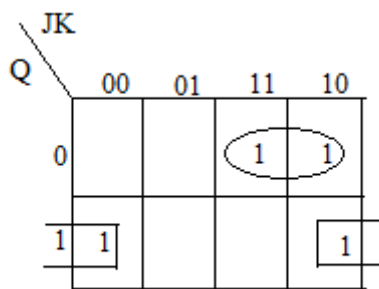


Fig (8.4) Master – Slave J –K Flip – Flop Circuit.

Truth Table of Master – Slave – JK Flip – Flop:

Inputs			Outputs
Q	J	K	Q_{t+1}
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

Where
 $Q \longrightarrow$ Present state
 $Q_{t+1} \longrightarrow$ Next state



Characteristic eqn $Q_{t+1} = J \bar{Q} + K Q$

iii) D – Flip – Flop:

IC – 7474: Dual + ve edge triggered D- Flip Flop :

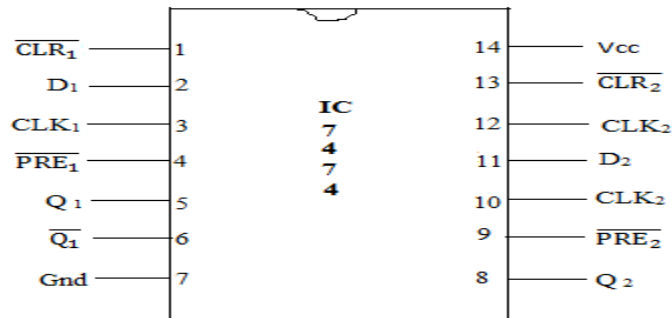


Fig (8.5) Pin Diagram

Circuit Implementation:

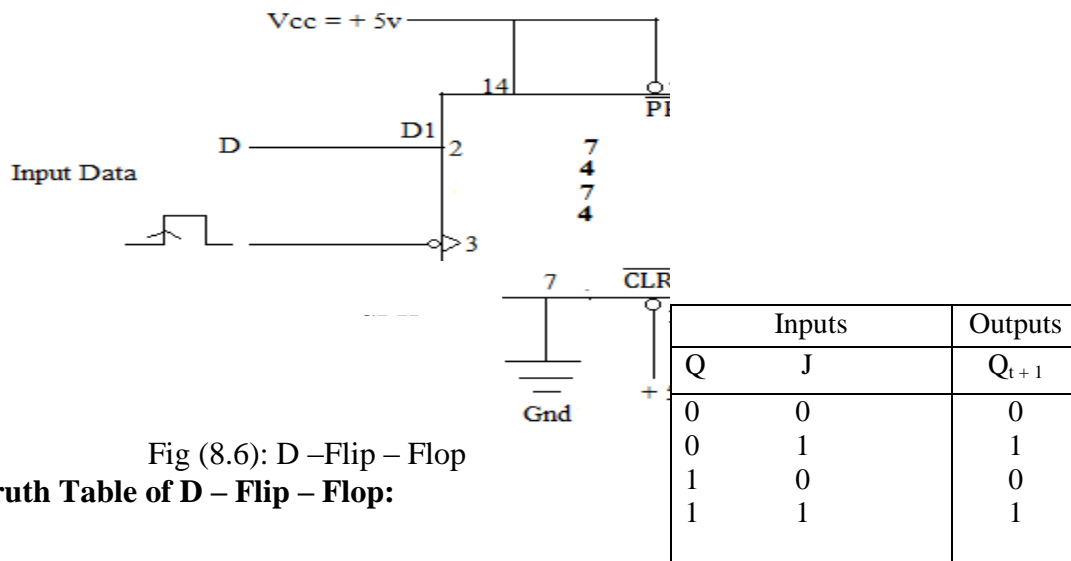
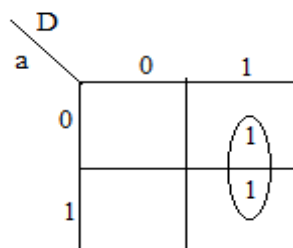


Fig (8.6): D –Flip – Flop

Truth Table of D – Flip – Flop:

Where

$Q \longrightarrow$ Present State
 $D \longrightarrow$ Data Input
 $Q_{t+1} \longrightarrow$ Next State



Characteristic eqn $Q_{t+1} = D$

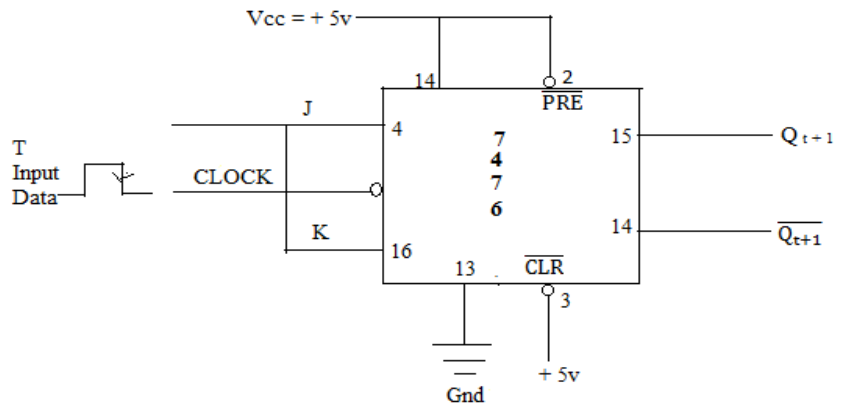
$Q_{t+1} = D$

From Characteristic equation it is clear that next state Q_{t+1} is equal to the input data D

i.e. $Q_{t+1} = D$

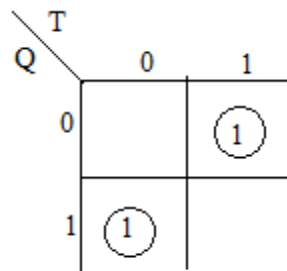
iv) T – Flip – Flop: Using J –K Flip – Flop

Truth Table of T – Flip – Flop:



Inputs		Outputs
Q	J	Q_{t+1}
0	0	0
0	1	1
1	0	0
1	1	1

Fig (8.7) : T – Flip - Flop Circuit
 Where
 $Q \longrightarrow$ Present State
 $D \longrightarrow$ Data Input



Characteristic eqn $Q_{t+1} = DT + Q \bar{T}$ (Ex – OR)

The next state of T Flip Flop is equal to Ex –OR of Present State (Q) and T input.

PROCEDURE:

- 1) Connections are made as per the circuit diagram.
- 2) Apply the –ve edge triggered, +ve edge triggered and level sensitive clock pulses as required.
- 3) Verify the truth table of all the Flip – Flops.
- 4) Switch - off the power supply and disconnect the circuit.

RESULT:

EXPERIMENT: 9

STUDY OF COUNTERS

I. STUDY OF ASYNCHRONOUS COUNTER

AIM: To design and test 3-bit binary asynchronous counter using flip-flop IC 7476 for the given sequence.

LEARNING OBJECTIVE:

To learn about Asynchronous Counter and its application

To learn the design of asynchronous up counter and down counter

COMPONENTS REQUIRED:

IC 7476, Patch Cords & IC Trainer Kit

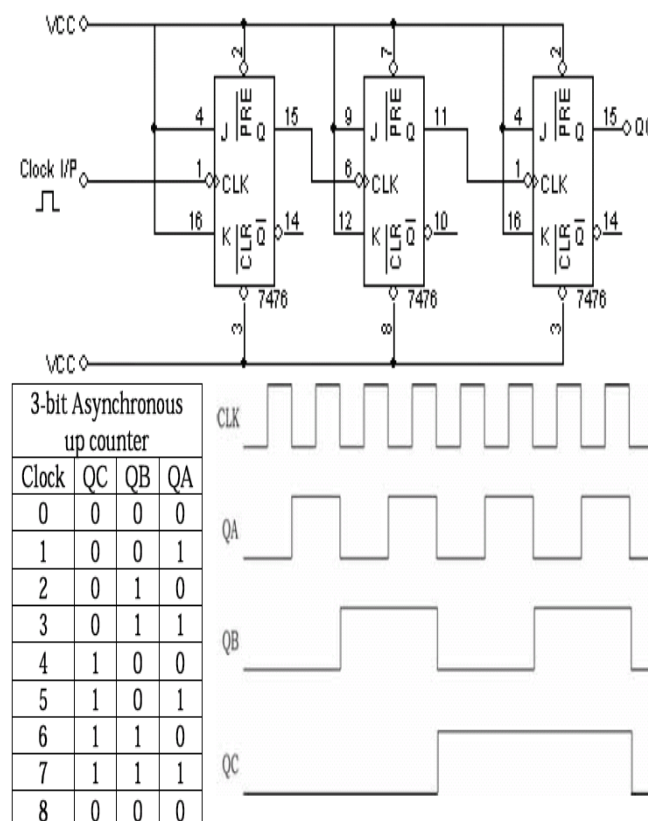
THEORY:

A counter in which each flip-flop is triggered by the output goes to previous flip-flop. As all the flip-flops do not change state simultaneously spike occur at the output. To avoid this, strobe pulse is required. Because of the propagation delay the operating speed of asynchronous counter is low. Asynchronous counter are easy and simple to construct.

Design:

MOD-8 UP COUNTER

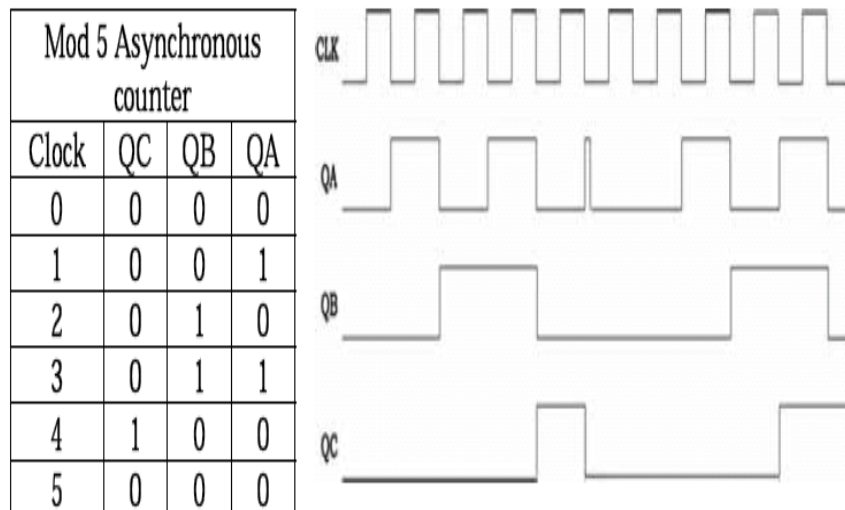
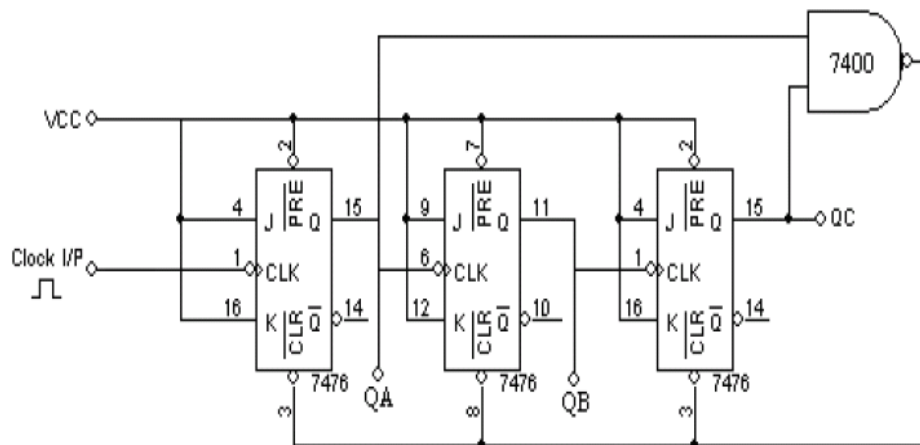
Circuit Diagram: - 3-Bit Asynchronous Up Counter



MOD_5 UP COUNTER

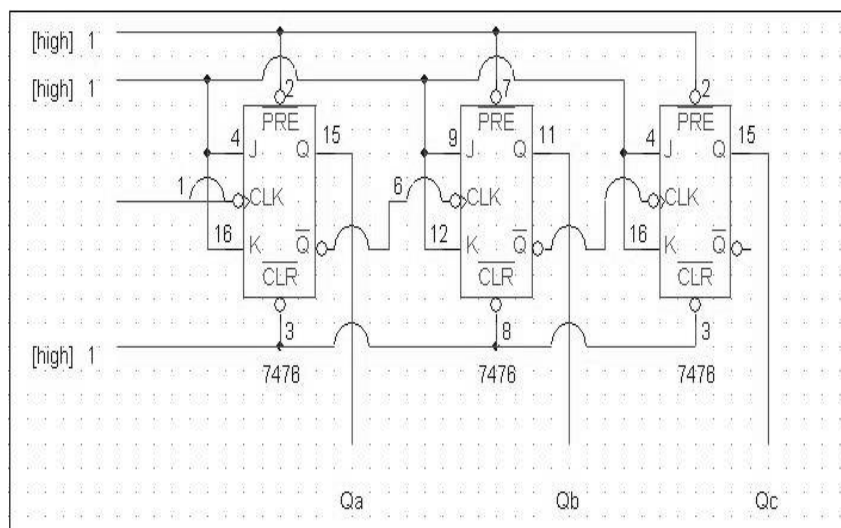
CIRCUIT DIAGRAM:

Mod 5 Asynchronous Counter:-



MOD-8 DOWN COUNTER

CIRCUIT DIAGRAM:



TRUTH TABLE

CLK	Q _C	Q _B	Q _A
0	1	1	1
1	1	1	0
2	1	0	1
3	1	0	0
4	0	1	1
5	0	1	0
6	0	0	1
7	0	0	0
8	1	1	1

PROCEDURE:

1. Check the components for their working.
2. Insert the appropriate IC into the IC base.
3. Rig up the circuit as shown in the logic circuit diagram.
4. Apply various input data to the logic circuit via the input logic switches.
5. Note down the corresponding output and verify the truth table.

Note: Write the pin numbers of each gate and also write the intermediate expressions.

RESULT:

II. SYNCHRONOUS COUNTERS

AIM:

To design and test 3-bit binary synchronous counter using flip-flop IC 7476 for the given sequence.

LEARNING OBJECTIVE:

To learn about synchronous Counter and its application
To learn the design of synchronous counter

COMPONENTS REQUIRED:

IC 7476, Patch Cords & IC Trainer Kit

THEORY:

A counter in which each flip-flop is triggered by the output goes to previous flip-flop. As all the flip-flops do not change states simultaneously in asynchronous counter, spike occur at the output. To avoid this, strobe pulse is required. Because of the propagation delay the operating speed of asynchronous counter is low. This problem can be solved by triggering all the flip-flops in synchronous with the clock signal and such counters are called synchronous counters.

Design:

MOD 5 COUNTERS:

TRUTH TABLE:

Q _C	Q _B	Q _A
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
0	0	0

Present count

Q _C	Q _B	Q _A
0	0	0
0	0	1
0	1	0
0	1	1
0	1	1
1	0	0

next count

Q _C	Q _B	Q _A
0	0	1
0	1	0
0	1	1
1	0	0
0	0	0

JK FF excitation table:

Q	Q ⁺	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

1	X	X	1
0	X	X	X

$$J_A = \overline{Q_C}$$

X	1	X	X
X	X	X	X

$$K_A = 1$$

0	1	X	X
0	X	X	X

$$J_B = Q_A$$

X	X	1	0
X	X	X	X

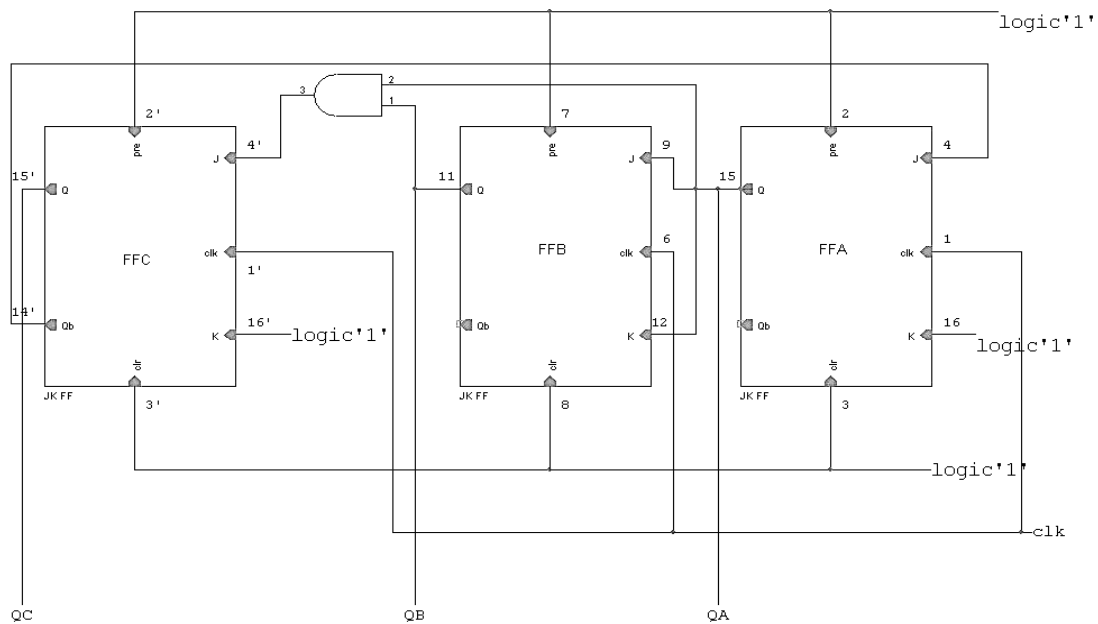
$$K_B = Q_A$$

0	0	1	0
X	X	X	X

$$J_C = Q_B Q_A$$

X	X	X	X
1	X	X	X

$$K_C = 1$$



MOD 8 COUNTERS:

TRUTH TABLE:

Q _C	Q _B	Q _A
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
1	1	0
1	1	1
0	0	0

Present count

Q _C	Q _B	Q _A
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
1	1	0
1	1	1

Next count

Q _C	Q _B	Q _A
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
1	1	0
1	1	1
0	0	0

JK FF excitation table:

Q	Q ⁺	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

DESIGN:

1	X	X	1
1	X	X	1

$$J_A = 1$$

X	1	1	X
X	1	1	X

$$K_A = 1$$

0	1	X	X
X	1	X	X

$$J_B = Q_A$$

X	X	1	0
X	X	1	0

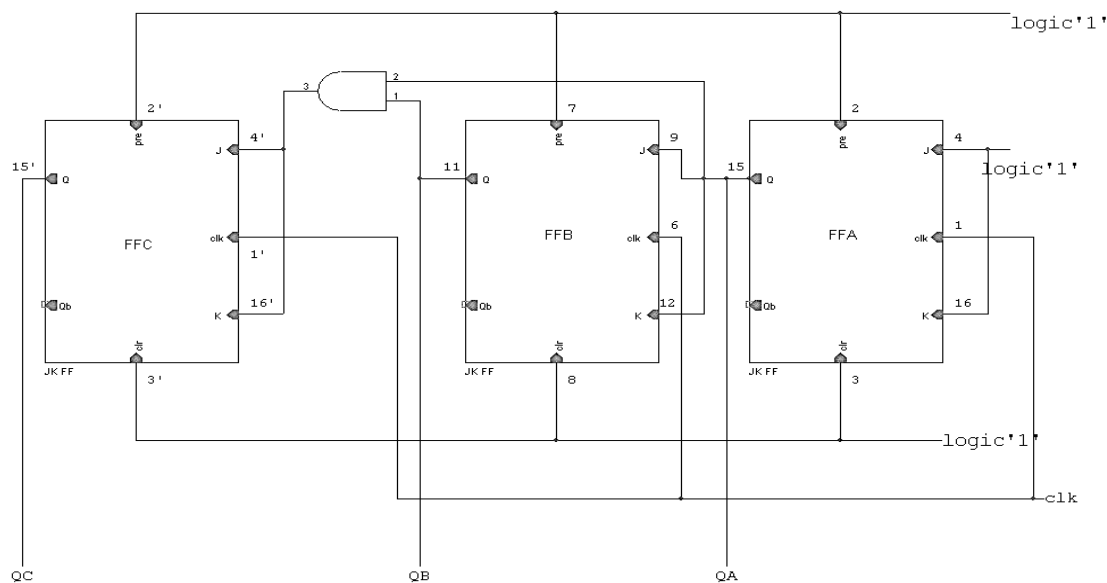
$$K_B = Q_A$$

0	0	1	0
X	X	X	X

$$J_C = Q_B Q_A$$

X	X	X	X
0	0	1	0

$$K_C = Q_B Q_A$$



PROCEDURE:

1. Check the components for their working.
2. Insert the appropriate IC into the IC base.
3. Rig up the circuit as shown in the logic circuit diagram.
4. Apply various input data to the logic circuit via the input logic switches.
5. Note down the corresponding output and verify the truth table.

Note: Write the pin numbers of each gate and also write the intermediate expressions.

RESULT:

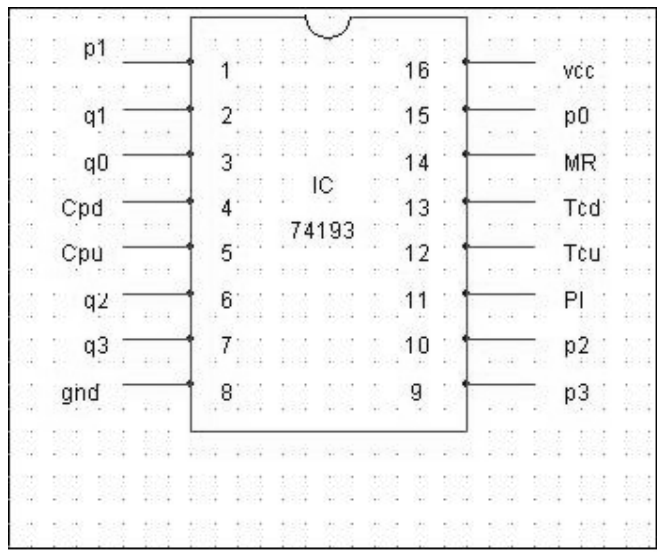
III. PRESETTABLE 4-BIT BINARY UP/DOWN COUNTER

AIM: To design IC 74193 as a up/down counter

COMPONENTS REQUIRED:

IC 74193, Patch Cords & IC Trainer Kit

PIN DETAILS OF IC 74193



1. P1,P2,P3 and P0 are parallel data inputs
2. Q0,Q1,Q2 and Q3 are flip-flop outputs
3. MR: Asynchronous master reset
4. PL: Asynchronous parallel load(active low) input
5. TCd : Terminal count-down output
6. TCu : Terminal count-up output

Up counter

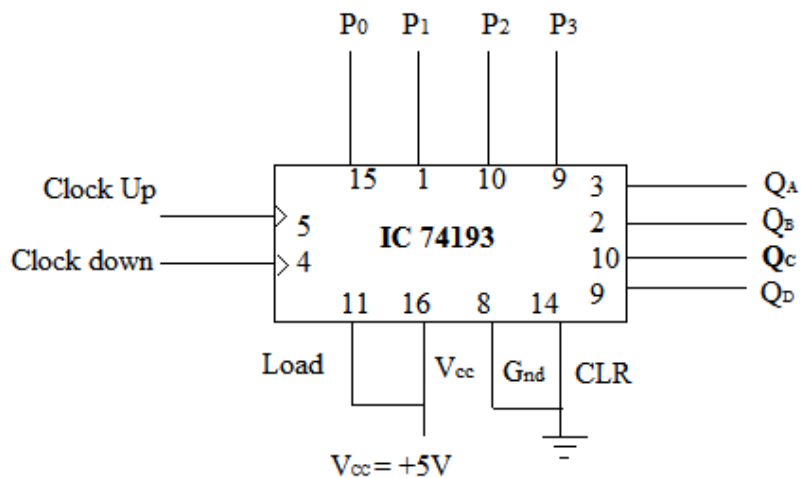
CLK	Q _D	Q _C	Q _B	Q _A
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	1	0	1	0
11	1	0	1	1
12	1	1	0	0
13	1	1	0	1
14	1	1	1	0
15	1	1	1	1
16	0	0	0	0

Down counter

CLK	Q _D	Q _C	Q _B	Q _A
0	1	1	1	1
1	1	1	1	0
2	1	1	0	1
3	1	1	0	0
4	1	0	1	1
5	1	0	1	0
6	1	0	0	1
7	1	0	0	0
8	0	1	1	1
9	0	1	1	0
10	0	1	0	1
11	0	1	0	0
12	0	0	1	1
13	0	0	1	0
14	0	0	0	1
15	0	0	0	0
16	1	1	1	1

Design up/ down counter using presettable counter

CIRCUIT DIAGRAM



PROCEDURE:

1. Check the components for their working.
2. Insert the appropriate IC into the IC base.
3. Rig up the circuit as shown in the logic circuit diagram.
4. Apply the parallel input to p_0 to p_3 and Give the Load pin to logic LOW.
5. To start counting connect Load input to logic HIGH.
6. Apply the clock pulse to observe the output.(For up-counter make clock down to be at logic 1 and give the clock input to Clock up input, for down counter make clock up to be at logic 1 and give the clock input to Clock down input)

Note:

1. For up-counter make clock down to be at logic 1 and give the clock input to Clock up input.
2. For down counter make clock up to be at logic 1 and give the clock input to Clock down input
3. Write the pin numbers of each gate and also write the intermediate expressions.

RESULT:

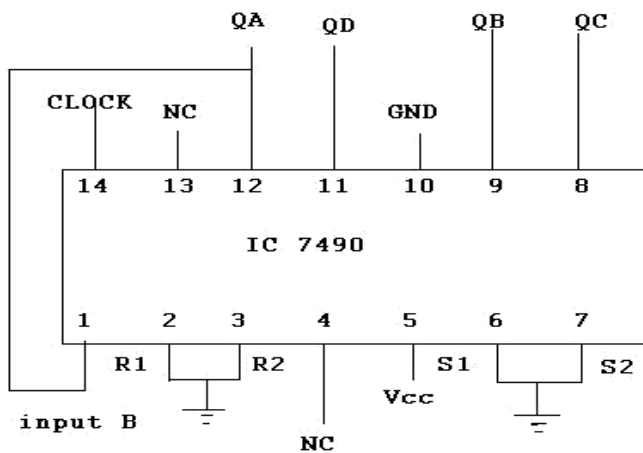
IV. STUDY OF 7490 BCD COUNTER

AIM: To design IC 7490 as a decade counter with BCD count sequence

COMPONENTS REQUIRED:

IC 7490, Patch Cords & IC Trainer Kit

DECADE COUNTER:



TRUTH TABLE:

Q _D	Q _C	Q _B	Q _A
0	0	0	0
0	0	0	1
0	0	1	0
0	0	1	1
0	1	0	0
0	1	0	1
0	1	1	0
0	1	1	1
1	0	0	0
1	0	0	1
0	0	0	0

PROCEDURE:

1. Check the components for their working.
2. Insert the appropriate IC into the IC base.
3. Rig up the circuit as shown in the logic circuit diagram.
4. Apply various input data to the logic circuit via the input logic switches.
5. Note down the corresponding output and verify the truth table.

Note: Write the pin numbers of each gate and also write the intermediate expressions.

RESULT:

EXPERIMENT: 10

SHIFT REGISTERS

AIM: To realize and study of Shift Register.

- 1) SISO (Serial in Serial out)
- 2) SIPO (Serial in Parallel out)
- 3) PIPO (Parallel in Parallel out)
- 4) PISO (Parallel in Serial out)

LEARNING OBJECTIVES:

To illustrate the operation of shift registers

To study different shift register configurations

COMPONENTS REQUIRED: IC 7495, IC 7474, Patch Cords & IC Trainer Kit.

Theory:

Shift registers are a type of sequential logic circuit, mainly for storage of digital data. They are a group of flip-flops connected in a chain so that the output from one flip-flop becomes the input of the next flip-flop. All the flip-flops are driven by a common clock, and all are set or reset simultaneously.

The serial in/serial out shift register accepts data serially – that is, one bit at a time on a single line. It produces the stored information on its output also in serial form.

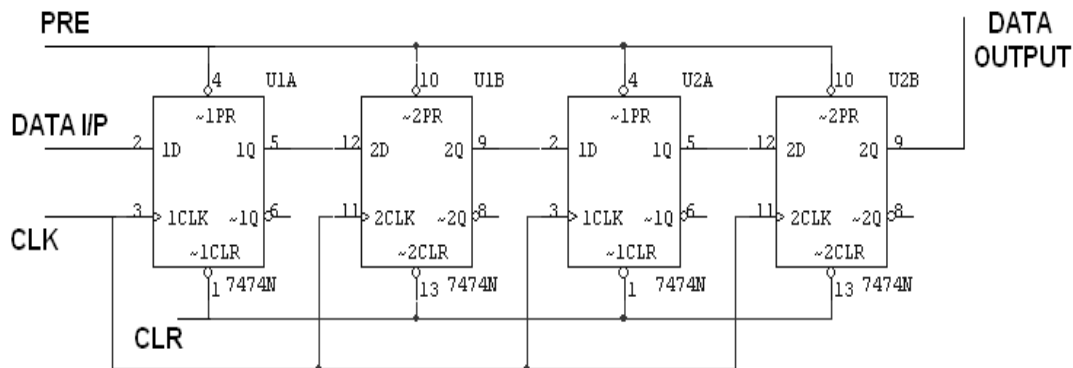
The serial in/parallel out shift register accepts data serially – that is, one bit at a time on a single line. It produces the stored information on its output in parallel form.

The parallel in/serial out shift register accepts data in parallel. It produces the stored information on its output also in serial form.

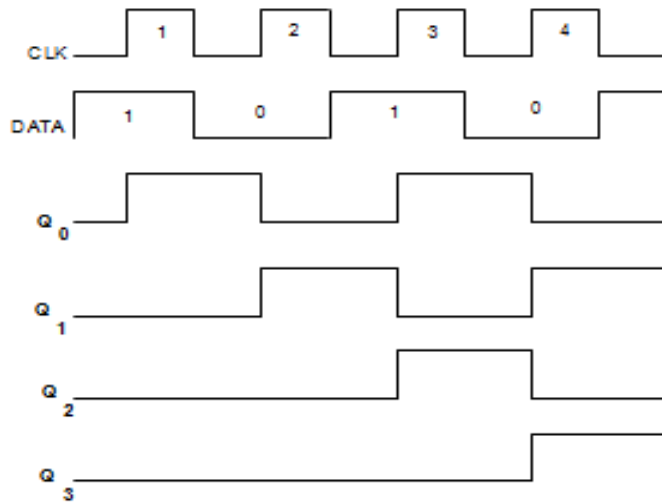
The parallel in/parallel out shift register accepts data in parallel. It produces the stored information on its output in parallel form.

SERIAL IN SERIAL OUT:

LOGIC DIAGRAM:



OUTPUT WAVEFORM:



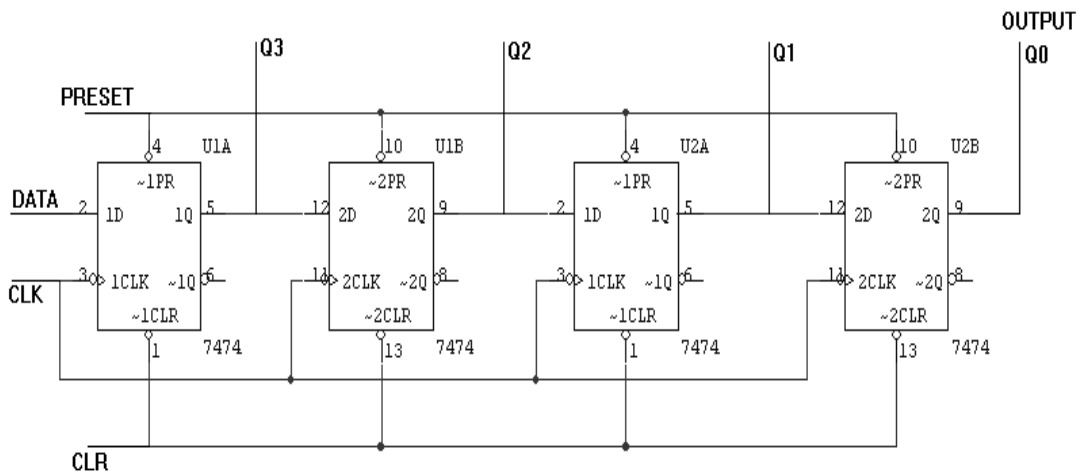
Output Waveforms of 4-bit Serial-in Serial-out Register

TRUTH TABLE:

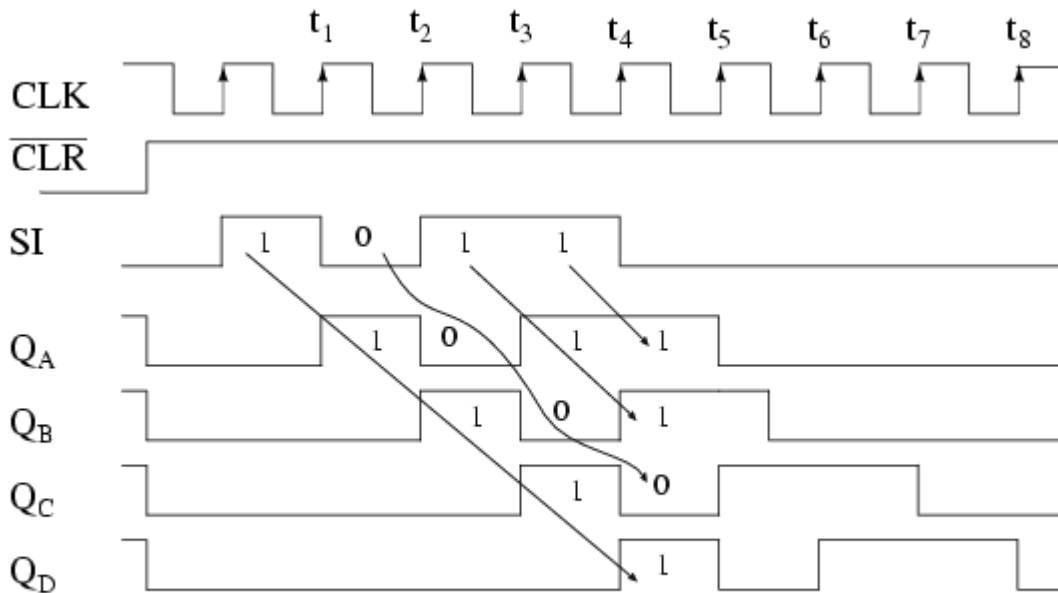
CLK	Serial in	Serial out
1	1	0
2	0	0
3	0	0
4	1	1
5	X	0
6	X	0
7	X	1

SERIAL IN PARALLEL OUT:

LOGIC DIAGRAM:



OUTPUT WAVEFORM:



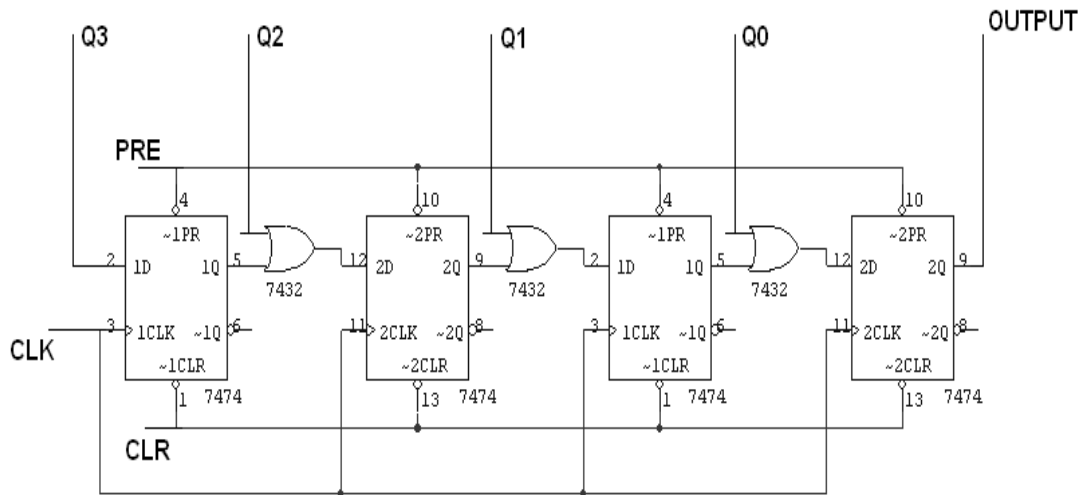
Serial-in/ parallel-out shift register waveforms

TRUTH TABLE:

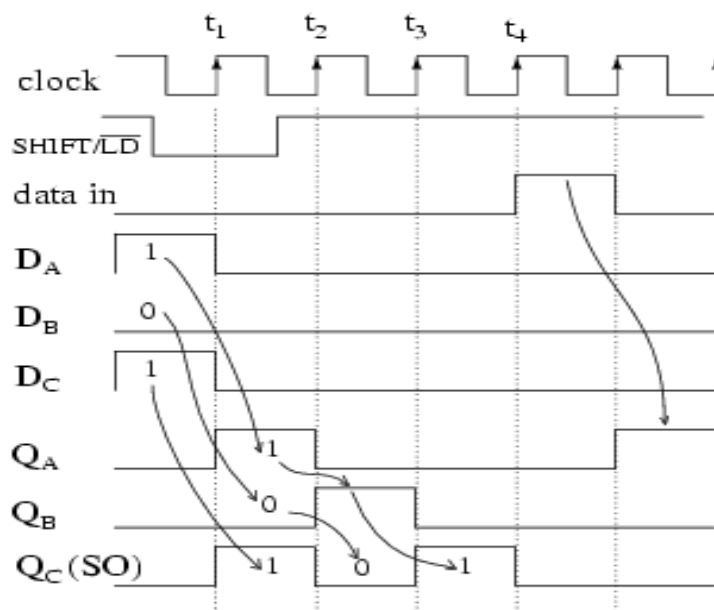
CLK	DATA	OUTPUT			
		QA	QB	QC	QD
1	1	1	0	0	0
2	0	0	1	0	0
3	0	0	0	1	1
4	1	1	0	0	1

PARALLEL IN SERIAL OUT:

LOGIC DIAGRAM:



OUTPUT WAVEFORM:



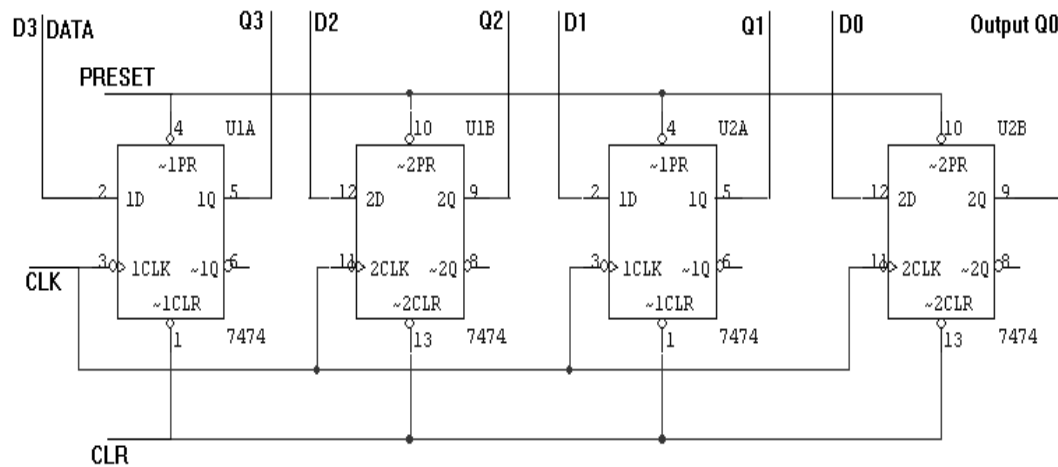
Parallel-in/ serial-out shift register load/shift waveforms

TRUTH TABLE:

CLK	Q3	Q2	Q1	Q0	O/P
0	1	0	0	1	1
1	0	0	0	0	0
2	0	0	0	0	0
3	0	0	0	0	1

PARALLEL IN PARALLEL OUT:

LOGIC DIAGRAM:



TRUTH TABLE:

CLK	DATA INPUT				OUTPUT			
	D _A	D _B	D _C	D _D	Q _A	Q _B	Q _C	Q _D
1	1	0	0	1	1	0	0	1
2	1	0	1	0	1	0	1	0

PROCEDURE:

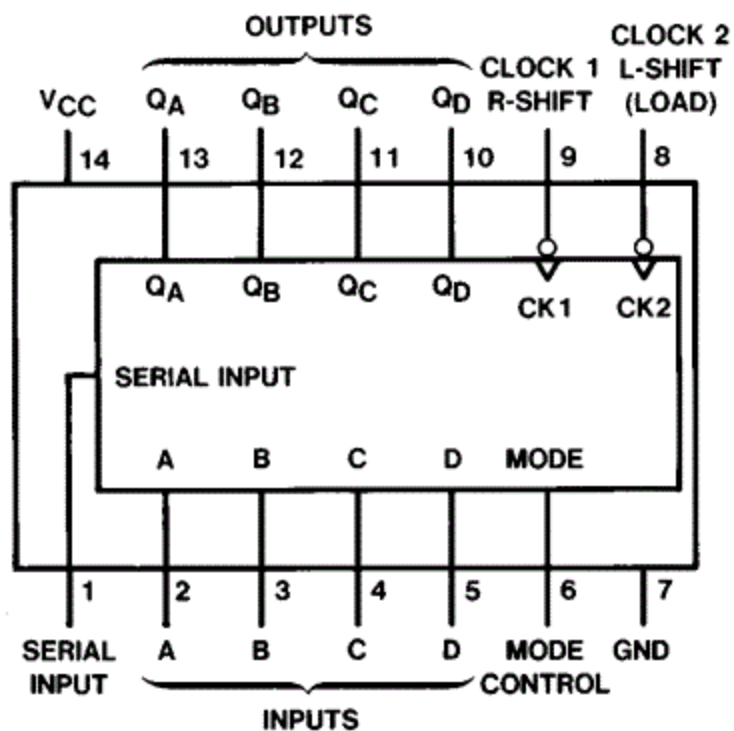
1. Check the components for their working.
2. Insert the appropriate IC into the IC base.
3. Rig up the circuit as shown in the logic circuit diagram.
4. Apply various input data to the logic circuit via the input logic switches.
5. Note down the corresponding output and verify the truth table.

Note: Write the pin numbers of each gate and also write the intermediate expressions.

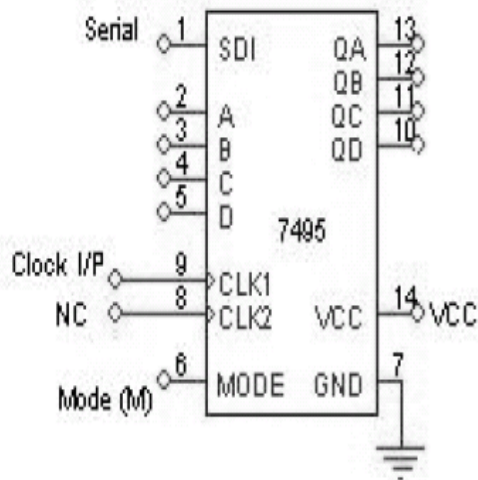
RESULT:

Shift Registers using IC 7495

PIN DIAGRAM of IC 7495:

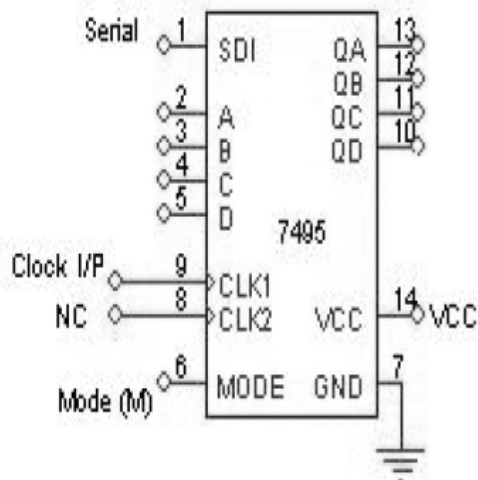


SIPO (Right Shift):-



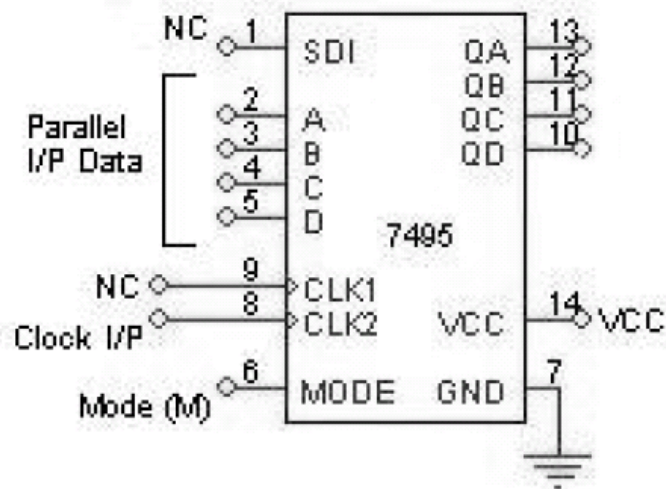
Clock	Serial i/p	QA	QB	QC	QD
1	0	0	X	X	X
2	1	1	0	X	X
3	1	1	1	0	X
4	1	1	1	1	0

SISO:-



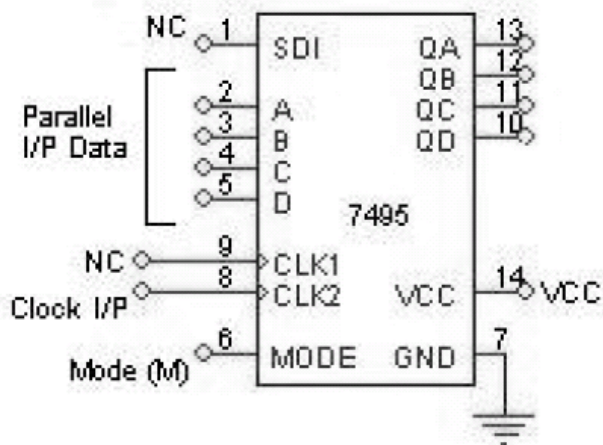
Clock	Serial i/p	QA	QB	QC	QD
1	do=0	0	X	X	X
2	d1=1	1	0	X	X
3	d2=1	1	1	0	X
4	d3=1	1	1	1	0=do
5	X	X	1	1	1=d1
6	X	X	X	1	1=d2
7	X	X	X	X	1=d3

PISO:



Mode	Clock	Parallel i/p				Parallel o/p			
		A	B	C	D	QA	QB	QC	QD
1	1	1	0	1	1	1	0	1	1
0	2	X	X	X	X	X	1	0	1
0	3	X	X	X	X	X	X	1	0
0	4	X	X	X	X	X	X	X	1

PIPO:-



Clock	Parallel i/p				Parallel o/p			
	A	B	C	D	QA	QB	QC	QD
1	1	0	1	1	1	0	1	1

PROCEDURE:**Serial In Parallel Out (SIPO):-**

1. Connections are made as per circuit diagram.
2. Keep the mode control in logic 0
3. Apply the data at serial input.
4. Apply one clock pulse at clock 1 observe this data at Q_A .
5. Apply the next data at serial input.
6. Apply one clock pulse at clock 2, observe that the data on Q_A will shift to Q_B and the new data applied will appear at Q_A .
7. Repeat steps 2 and 3 till all the 4 bits data appear at the output of shift register.

Serial In Serial Out(SISO):-

1. Connections are made as per circuit diagram.
2. Keep the mode control in logic 0
3. Load the shift register with 4 bits of data one by one serially.
4. At the end of 4th clock pulse the first data ' d_0 ' appears at Q_D .
5. Apply another clock pulse; the second data ' d_1 ' appears at Q_D and so on.
6. Thus the data applied serially at the input comes out serially at Q_D

Parallel In Serial Out (PISO):-

1. Connections are made as per circuit diagram.
2. Apply the desired 4 bit data at A, B, C and D.
3. Keeping the mode control $M=1$ apply one clock pulse. The data applied at A, B, C and D will appear at Q_A , Q_B , Q_C and Q_D respectively.
4. Now mode control $M=0$. Apply clock pulses one by one and observe the Data coming out serially at Q_D

Parallel In Parallel Out (PIPO):-

1. Connections are made as per circuit diagram.
2. Apply the 4 bit data at A, B, C and D.
3. Apply one clock pulse at Clock 2 (Note: Mode control $M=1$).
4. The 4 bit data at A, B, C and D appears at Q_A , Q_B , Q_C and Q_D respectively.

RESULT:

EXPERIMENT- 11**RING AND JHONSON COUNTERS**

AIM: To design 3-bit Ring counter and Johnson counter.

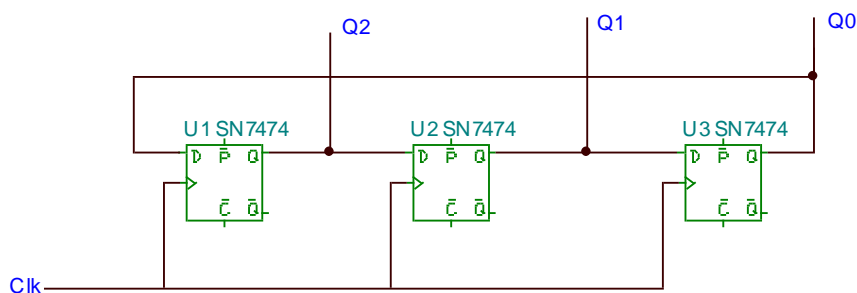
COMPONENTS REQUIRED:

7474 (D flip flop), 7404.

THEORY:

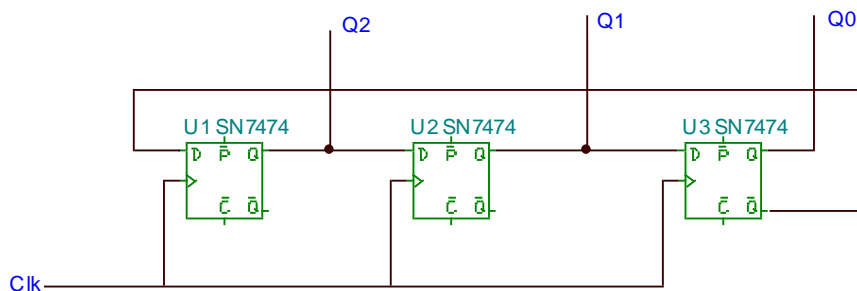
Ring counter is a basic register with direct feedback such that the contents of the register simply circulate around the register when the clock is running. Here the last output that is QD in a shift register is connected back to the serial input.

A basic ring counter can be slightly modified to produce another type of shift register counter called Johnson counter. Here complement of last output is connected back to the not gate input and not gate output is connected back to serial input. A four bit Johnson counter gives 8 state output.

CIRCUIT DIAGRAM:**RING COUNTER:**

Clk	Q2	Q1	Q0
0	1	0	0
1	0	1	0
2	0	0	1
3	1	0	0

JOHNSON COUNTER:



TRUTH TABLE:

Clk	Q2	Q1	Q0
0	0	0	0
1	1	0	0
2	1	1	0
3	1	1	1
4	0	1	1
5	0	0	1
6	0	0	0

PROCEDURE:

- Check the D flip flop for its working.
- Rig up the circuit as Ring counter.
- For ring counter set the first flip flop to logic 1(using preset), second and third to logic 0 (using clear) to set the data as 1 0 0 and then leave the preset and clear open.
- Observe the output and verify the truth table.
- For Johnson counter set the data as 0 0 0 (using clear) and observe the output and verify the truth table.

RESULT:

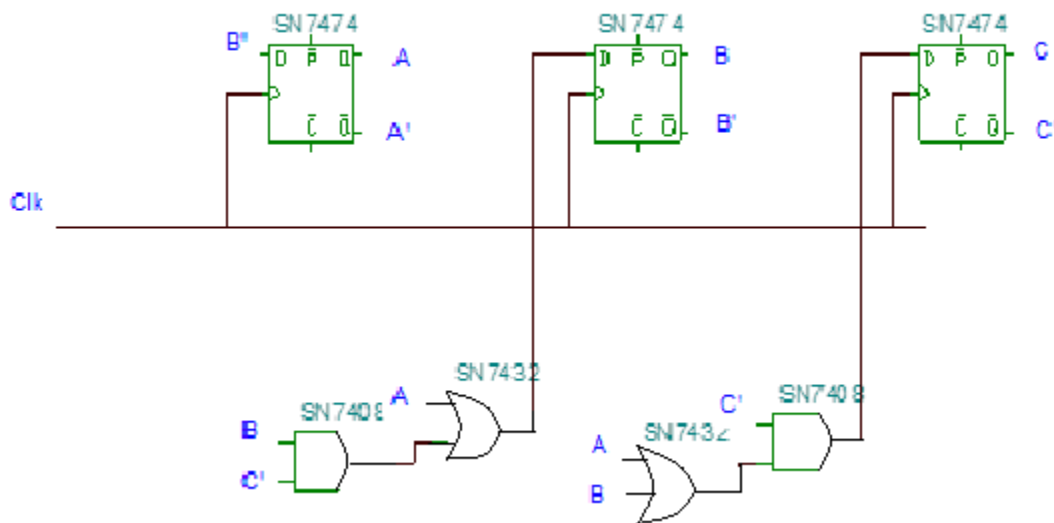
EXPERIMENT 12

SEQUENCE GENERATOR USING D FLIP FLOP(IC 7474)

AIM: Design and testing of sequence generator using D flip flop

COMPONENTS REQUIRED: IC 7474, 7408, 7432, Trainer kit.

CIRCUIT DIAGRAM:



TRUTH TABLE:

Clk	QC	QB	QA
0	0	0	0
1	1	0	0
2	1	1	1
3	0	1	0
4	0	1	1

PROCEDURE:

- Check the D flip flop for its working.
- Design the circuit for sequence 0, 4, 7, 2, 3.
- Connect the designed circuit for the given sequence.
- Apply the clock pulse and verify the sequence.

RESULT: