Chettinad
College of Engineering & Technology
Approved by AICTE-New Delhi and Affiliated to Anna University-Chennai.

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

EC3352-DIGITAL SYSTEM DESIGN LAB

ACADEMIC YEAR: 2023-2024

II YEAR/III TH Semester Lab Manual

LIST OF EXPERIMENTS:

INDEX

S.No	Name of the Experiment	Page No	Marks Obtained	Signature of the faculty member.
1	**Design of code converter**			
2	**Design of adder and subtractors**			
3	**Design of Magnitude comparator**			
4	**Design of multiplexer and Demultiplexer**			
5	**Design and implement of shift registers**			
6	**Design of implementation of counters flipflop**			

Ex.No.-1 CODE CONVERTOR

AIM:

To design and implement 4-bit

(i) Binary to gray code converter
(ii) Gray to binary code converter
(iii) BCD to excess-3 code converter
(iv) Excess-3 to BCD code converter

APPARATUS REQUIRED:

SL.NO.	COMPONENT	SPECIFICATION	QTY.
1.	X-OR GATE	IC 7486	1
2.	AND GATE	IC 7408	1
3.	OR GATE	IC 7432	1
4.	NOT GATE	IC 7404	1
5.	IC TRAINER KIT	-	1
6.	PATCH CORDS	-	35

THEORY:

The availability of large variety of codes for the same discrete elements of information results in the use of different codes by different systems. A conversion circuit must be inserted between the two systems if each uses different codes for same information. Thus, code converter is a circuit that makes the two systems compatible even though each uses different binary code.

The bit combination assigned to binary code to gray code. Since each code uses four bits to represent a decimal digit. There are four inputs and four outputs. Gray code is a non-weighted code.

The input variable are designated as B3, B2, B1, B0 and the output variables are designated as C3, C2, C1, Co. from the truth table, combinational circuit is designed. The Boolean functions are obtained from K-Map for each output variable.

A code converter is a circuit that makes the two systems compatible even though each uses a different binary code. To convert from binary code to Excess-3 code, the input lines must supply the bit combination of elements as specified by code and the output lines generate the corresponding bit combination of code. Each one of the four maps represents one of the four outputs of the circuit as a function of the four input variables.

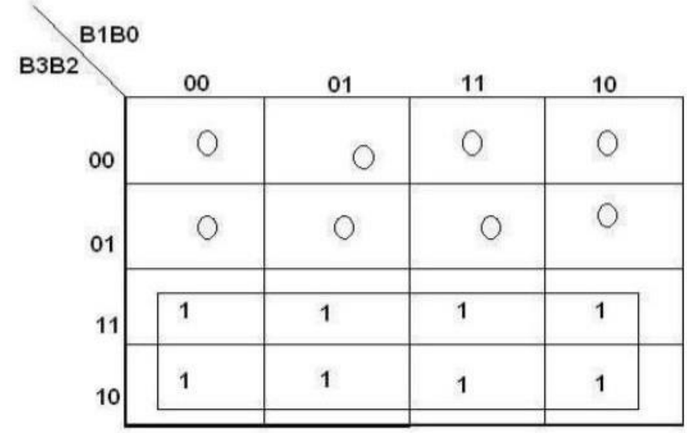
A two-level logic diagram may be obtained directly from the Boolean expressions derived by the maps. These are various other possibilities for a logic diagram that implements this circuit. Now the OR gate whose output is C+D has been used to implement partially each of three outputs.

BINARY TO GRAY CODE CONVERTOR

TRUTH TABLE:

Binary Input				Gray Code Output			
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

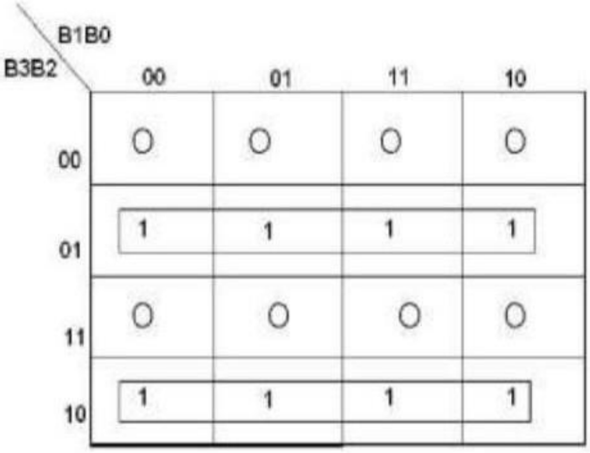
K-Map for G_3



B3B2 \ B1B0	00	01	11	10
00	O	O	O	O
01	O	O	O	O
11	1	1	1	1
10	1	1	1	1

$G_3 = B_3$

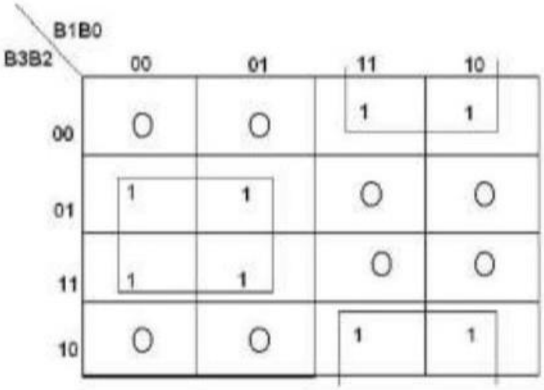
K-Map for G_2



B3B2 \ B1B0	00	01	11	10
00	O	O	O	O
01	1	1	1	1
11	O	O	O	O
10	1	1	1	1

G2 = B3 ⊕ B2

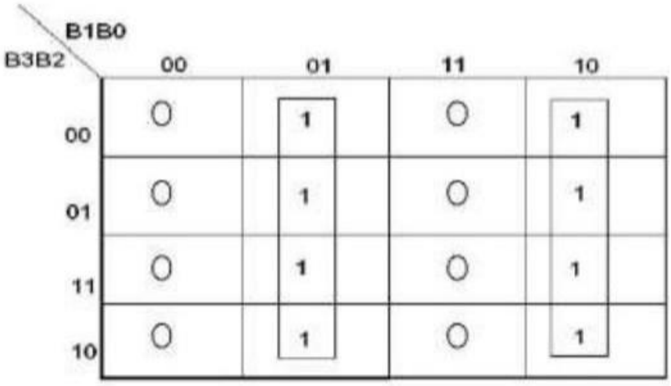
K-Map for G_1



B3B2 \ B1B0	00	01	11	10
00	O	O	1	1
01	1	1	O	O
11	1	1	O	O
10	O	O	1	1

G1 = B1 ⊕ B2

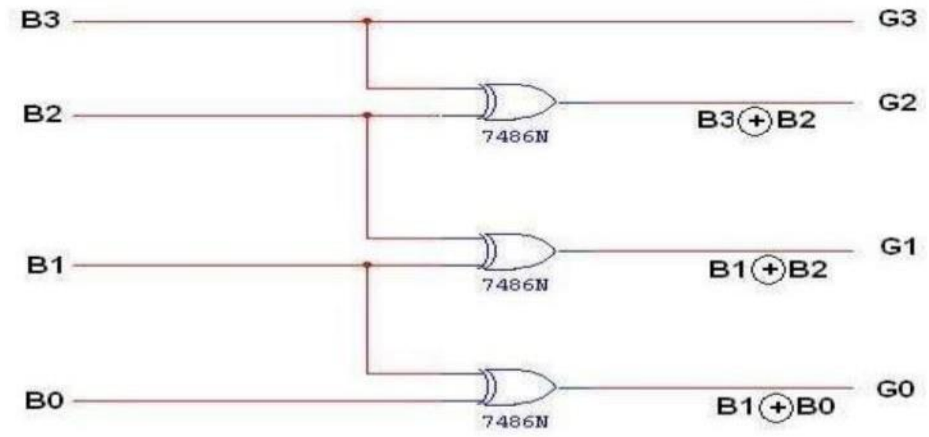
K-Map for G_0



B3B2 \ B1B0	00	01	11	10
00	O	1	O	1
01	O	1	O	1
11	O	1	O	1
10	O	1	O	1

G0 = B1 ⊕ B0

LOGIC DIAGRAM:

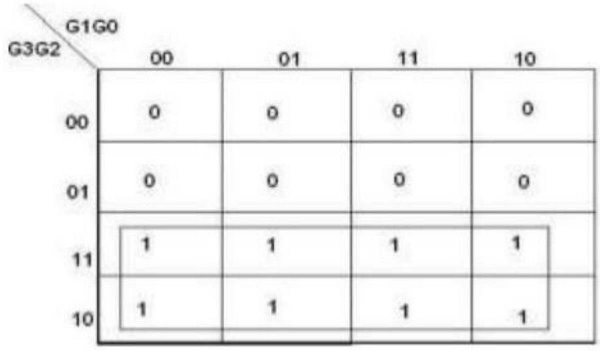


GRAY CODE TO BINARY CONVERTOR

TRUTH TABLE:

GRAY CODE				BINARY CODE			
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	1	0	0	1	0
0	0	1	0	0	0	1	1
0	1	1	0	0	1	0	0
0	1	1	1	0	1	0	1
0	1	0	1	0	1	1	0
0	1	0	0	0	1	1	1
1	1	0	0	1	0	0	0
1	1	0	1	1	0	0	1
1	1	1	1	1	0	1	0
1	1	1	0	1	0	1	1
1	0	1	0	1	1	0	0
1	0	1	1	1	1	0	1
1	0	0	1	1	1	1	0
1	0	0	0	1	1	1	1

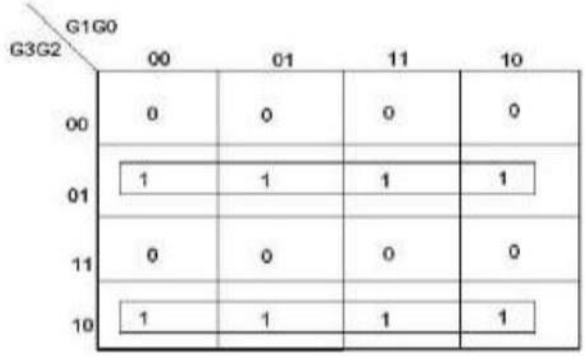
K-Map for B_3:



G3G2 \ G1G0	00	01	11	10
00	0	0	0	0
01	0	0	0	0
11	1	1	1	1
10	1	1	1	1

B3 = G3

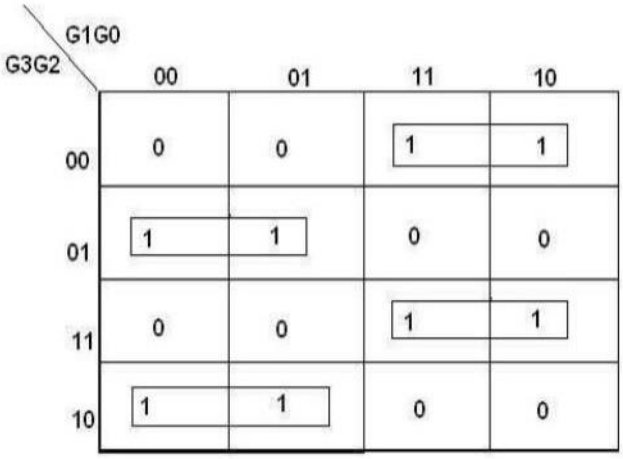
K-Map for B_2:



G3G2 \ G1G0	00	01	11	10
00	0	0	0	0
01	1	1	1	1
11	0	0	0	0
10	1	1	1	1

$B2 = G3 \oplus G2$

K-Map for B1:



G3G2 \ G1G0	00	01	11	10
00	0	0	1	1
01	1	1	0	0
11	0	0	1	1
10	1	1	0	0

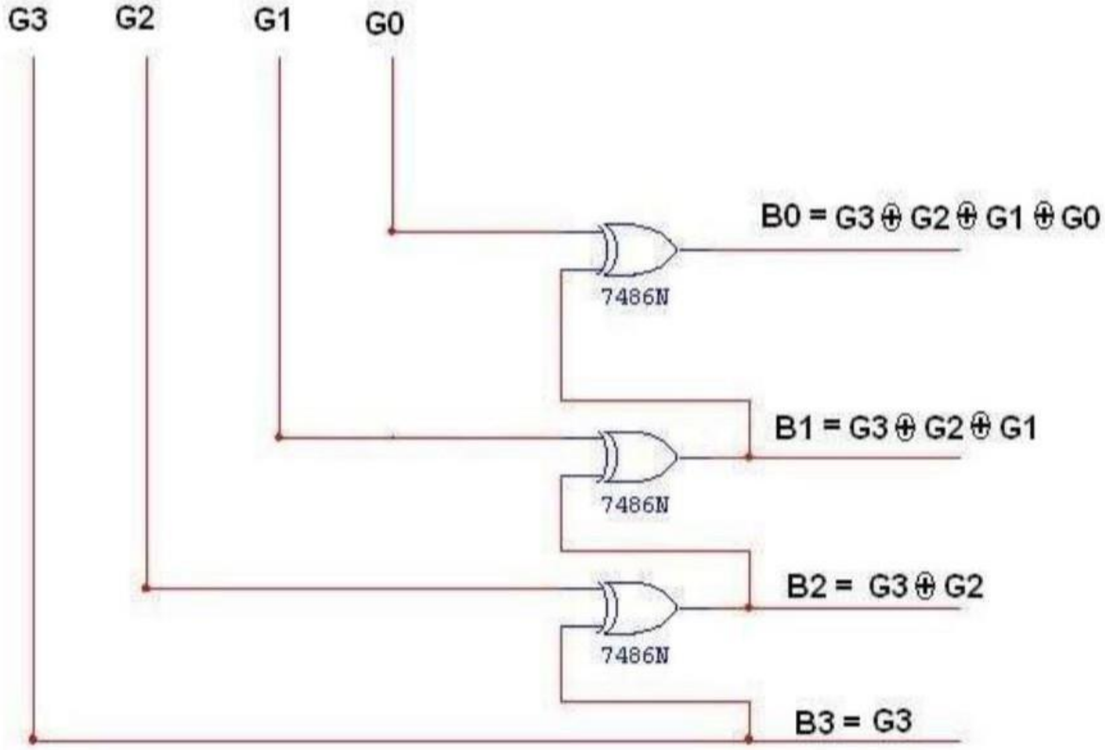
$B1 = G3 \oplus G2 \oplus G1$

K-Map for B0:

G3G2 \ G1G0	00	01	11	10
00	0	①	0	①
01	①	0	①	0
11	0	①	0	①
10	①	0	①	0

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

LOGIC DIAGRAM:

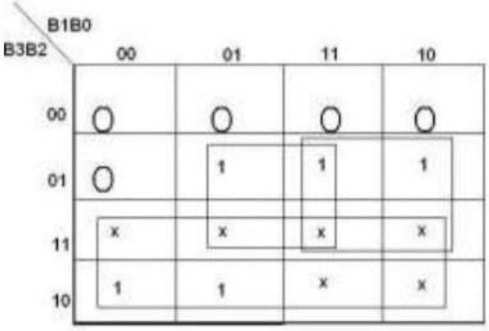


TRUTH TABLE: **BCD TO EXCESS-3 CONVERTOR**

| BCD input | Excess – 3 output |

B3	B2	B1	B0	G3	G2	G1	G0
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
1	0	1	0	x	x	x	x
1	0	1	1	x	x	x	x
1	1	0	0	x	x	x	x
1	1	0	1	x	x	x	x
1	1	1	0	x	x	x	x
1	1	1	1	x	x	x	X

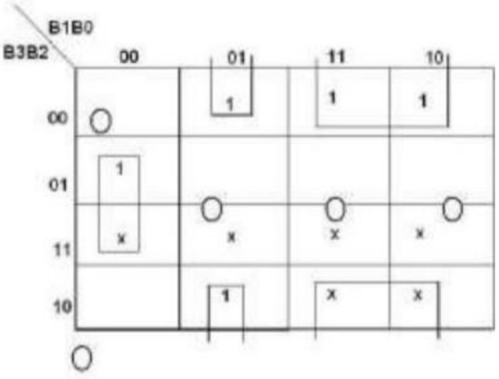
K-Map for E$_3$:



B3B2 \ B1B0	00	01	11	10
00	O	O	O	O
01	O	1	1	1
11	x	x	x	x
10	1	1	x	x

E3 = B3 + B2 (B0 + B1)

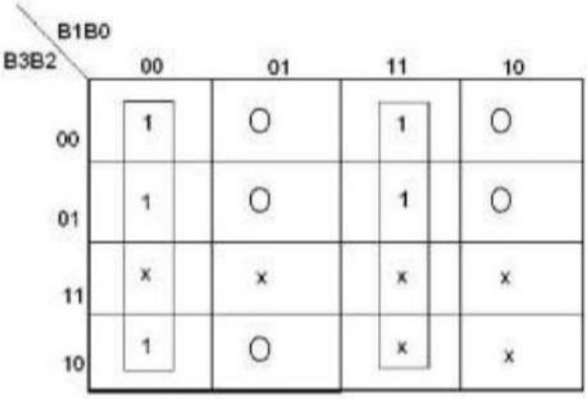
K-Map for E_2:



B3B2 \ B1B0	00	01	11	10
00	O	1	1	1
01	1	O	O	O
11	x	x	x	x
10	O	1	x	x

$E2 = B2 \oplus (B1 + B0)$

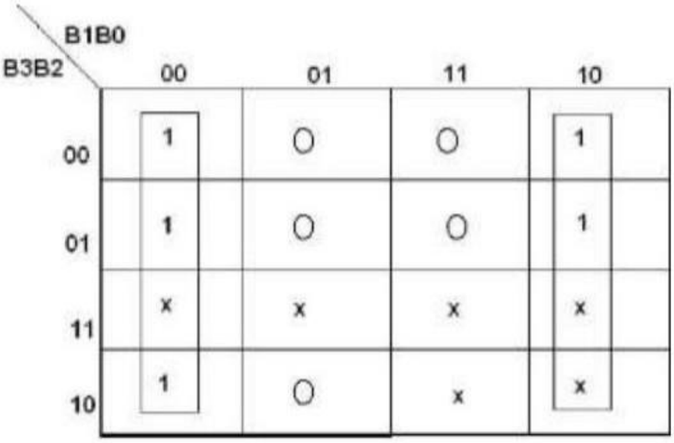
K-Map for E_1:



B3B2 \ B1B0	00	01	11	10
00	1	O	1	O
01	1	O	1	O
11	x	x	x	x
10	1	O	x	x

$E1 = B1 \overline{\oplus} B0$

K-Map for E_0:



B3B2 \ B1B0	00	01	11	10
00	1	O	O	1
01	1	O	O	1
11	x	x	x	x
10	1	O	x	x

$E0 = \overline{B0}$

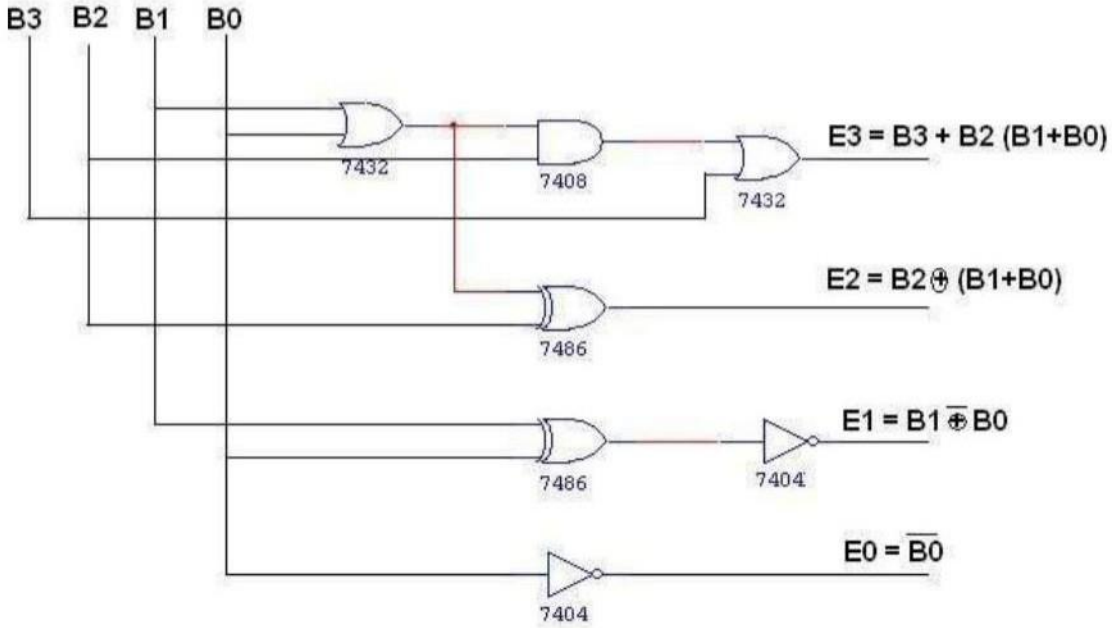
EXCESS-3 TO BCD CONVERTOR

TRUTH TABLE:

| Excess – 3 Input | BCD Output |

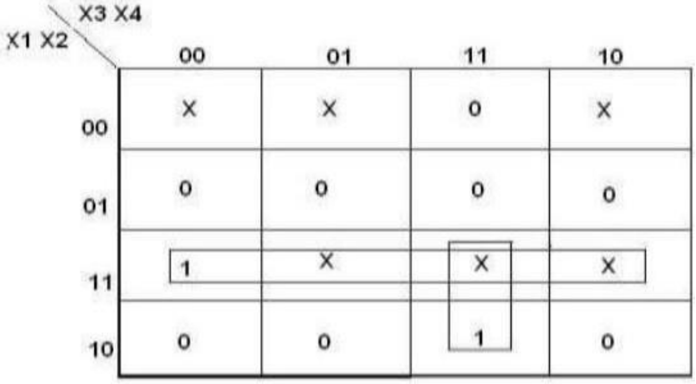
B3	B2	B1	B0	G3	G2	G1	G0
0	0	1	1	0	0	0	0
0	1	0	0	0	0	0	1
0	1	0	1	0	0	1	0
0	1	1	0	0	0	1	1
0	1	1	1	0	1	0	0
1	0	0	0	0	1	0	1
1	0	0	1	0	1	1	0
1	0	1	0	0	1	1	1
1	0	1	1	1	0	0	0
1	1	0	0	1	0	0	1

LOGIC DIAGRAM:



EXCESS-3 TO BCD CONVERTOR

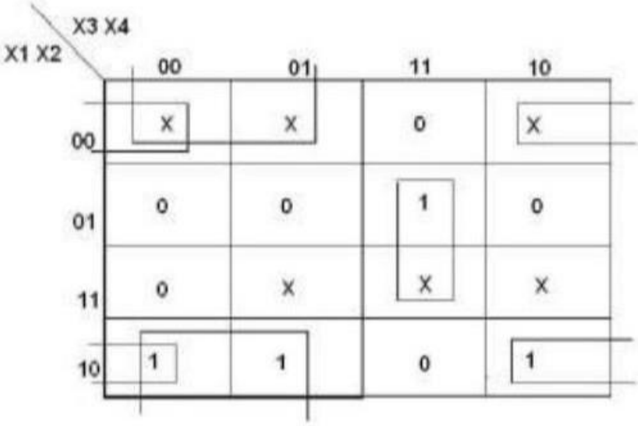
K-Map for A:



X1 X2 \ X3 X4	00	01	11	10
00	X	X	0	X
01	0	0	0	0
11	1	X	X	X
10	0	0	1	0

A = X1 X2 + X3 X4 X1

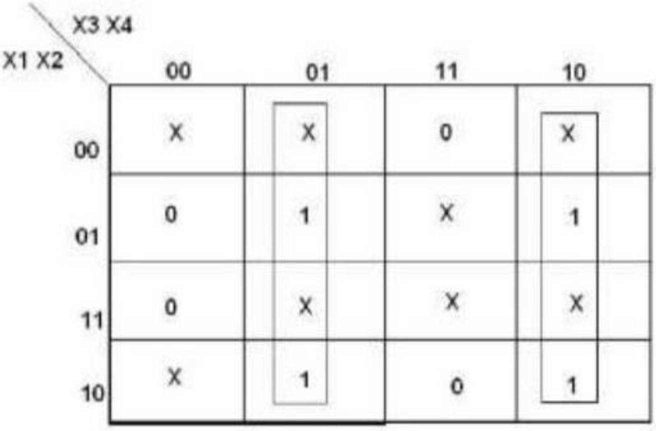
K-Map for B:



X1 X2 \ X3 X4	00	01	11	10
00	X	X	0	X
01	0	0	1	0
11	0	X	X	X
10	1	1	0	1

$$B = X2 \oplus (\overline{X3} + \overline{X4})$$

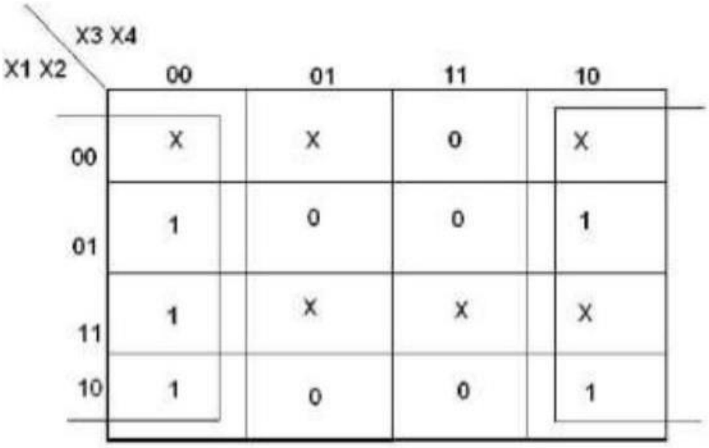
K-Map for C:



X1 X2 \ X3 X4	00	01	11	10
00	X	X	0	X
01	0	1	X	1
11	0	X	X	X
10	X	1	0	1

$$C = X3 \oplus X4$$

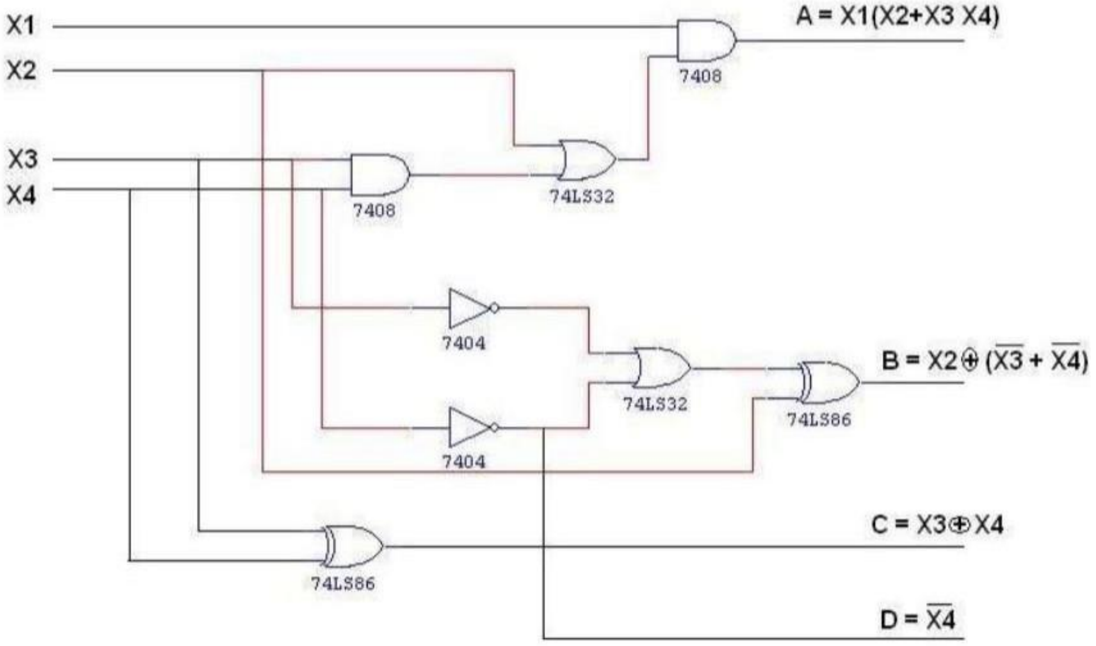
K-Map for D:



X1 X2 \ X3 X4	00	01	11	10
00	X	X	0	X
01	1	0	0	1
11	1	X	X	X
10	1	0	0	1

$$D = \overline{X4}$$

EXCESS-3 TO BCD CONVERTOR



PROCEDURE:

(i) Connections were given as per circuit diagram.
(ii) Logical inputs were given as per truth table
(iii) Observe the logical output and verify with the truth tables.

RESULT:

Thus the following 4-bit converters are designed and constructed.

(i) Binary to gray code converter
(ii) Gray to binary code converter
(iii) BCD to excess-3 code converter
(iv) Excess-3 to BCD code converter

Ex.No.-2 ADDER AND SUBTRACTOR

AIM:

To design and construct half adder, full adder, half subtractor and full subtractor circuits and verify the truth table using logic gates.

APPARATUS REQUIRED:

SL.NO.	COMPONENT	SPECIFICATION	QTY.
1.	AND GATE	IC 7408	1
2.	X-OR GATE	IC 7486	1
3.	NOT GATE	IC 7404	1
4.	OR GATE	IC 7432	1
5.	IC TRAINER KIT	-	1
6.	PATCH CORDS	-	23

THEORY:

HALF ADDER:

A half adder has two inputs for the two bits to be added and two outputs one from the sum ' S' and other from the carry ' c' into the higher adder position. Above circuit is called as a carry signal from the addition of the less significant bits sum from the X-OR Gate the carry out from the AND gate.

FULL ADDER:

A full adder is a combinational circuit that forms the arithmetic sum of input; it consists of three inputs and two outputs. A full adder is useful to add three bits at a time but a half adder cannot do so. In full adder sum output will be taken from X-OR Gate, carry output will be taken from OR Gate.

HALF SUBTRACTOR:

The half subtractor is constructed using X-OR and AND Gate. The half subtractor has two input and two outputs. The outputs are difference and borrow. The difference can be applied using X-OR Gate, borrow output can be implemented using an AND Gate and an inverter.

FULL SUBTRACTOR:

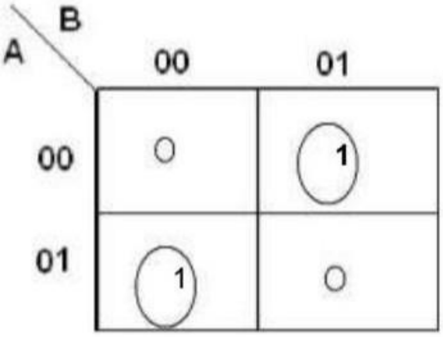
The full subtractor is a combination of X-OR, AND, OR, NOT Gates. In a full subtractor the logic circuit should have three inputs and two outputs. The two half subtractor put together gives a full subtractor .The first half subtractor will be C and A B. The output will be difference output of full subtractor. The expression AB assembles the borrow output of the half subtractor and the second term is the inverted difference output of first X-OR.

HALF ADDER

TRUTH TABLE:

A	B	CARRY	SUM
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

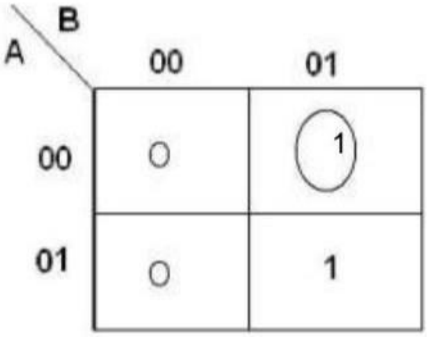
K-Map for SUM:



A \ B	00	01
00	0	1
01	1	0

SUM = A'B + AB'

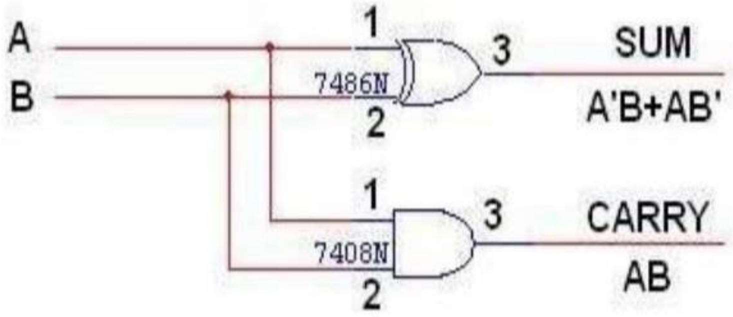
K-Map for CARRY:



A \ B	00	01
00	0	1
01	0	1

CARRY = AB

LOGIC DIAGRAM:

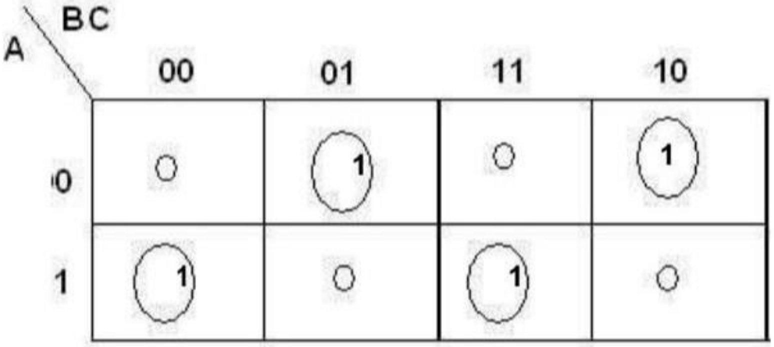


FULL ADDER

TRUTH TABLE:

A	B	C	CARRY	SUM
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

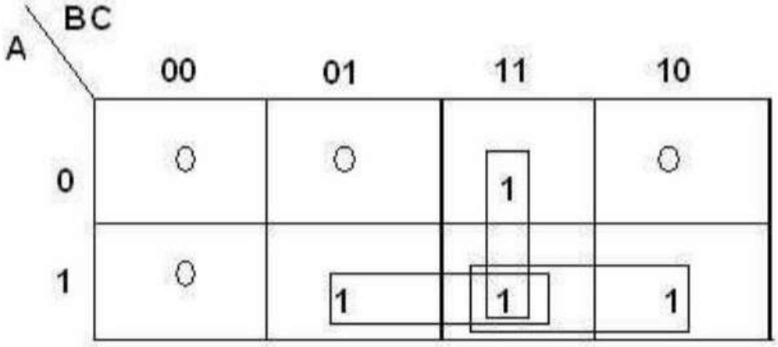
K-Map for SUM



A \ BC	00	01	11	10
0	0	1	0	1
1	1	0	1	0

SUM = A’B’C + A’BC’ + ABC’ + ABC

K-Map for CARRY

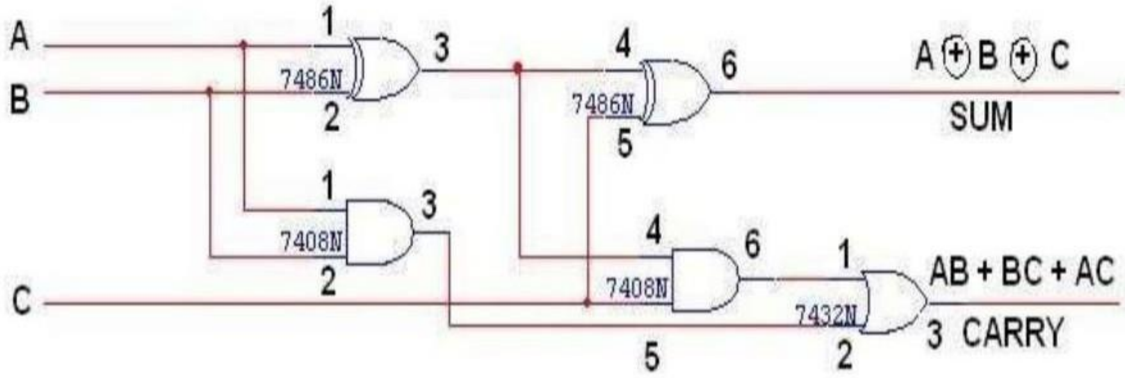


A \ BC	00	01	11	10
0	0	0	1	0
1	0	1	1	1

CARRY = AB + BC + AC

LOGIC DIAGRAM:

FULL ADDER USING TWO HALF ADDER

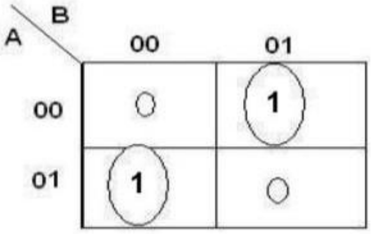


HALF SUBTRACTOR

TRUTH TABLE:

A	B	BORROW	DIFFERENCE
0	0	0	0
0	1	1	1
1	0	0	1
1	1	0	0

K-Map for DIFFERENCE

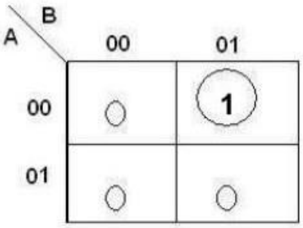


A \ B	00	01
00	0	1
01	1	0

DIFFERENCE = A'B + AB'

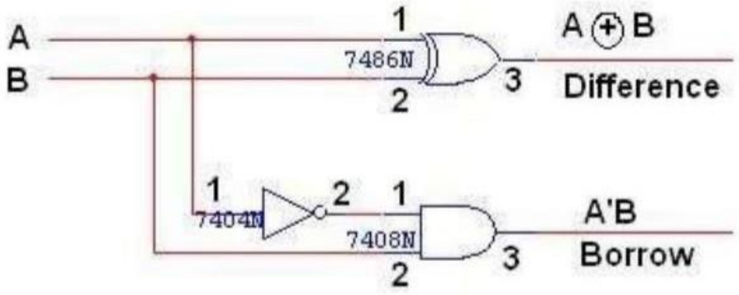
K-Map for BORROW

BORROW = A'B



A \ B	00	01
00	0	1
01	0	0

LOGIC DIAGRAM

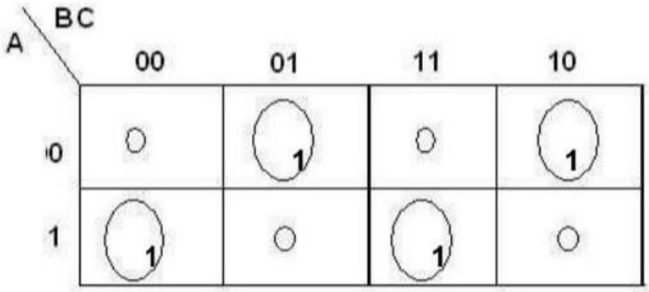


FULL SUBTRACTOR

TRUTH TABLE:

A	B	C	BORROW	DIFFERENCE
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	1	0
1	0	0	0	1
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

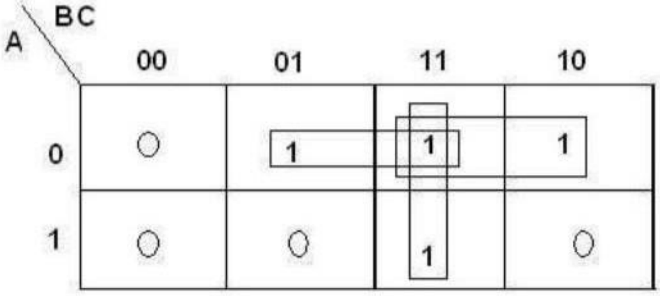
K-Map for Difference



A \ BC	00	01	11	10
0	0	1	0	1
1	1	0	1	0

Difference = A'B'C + A'BC' + AB'C' + ABC

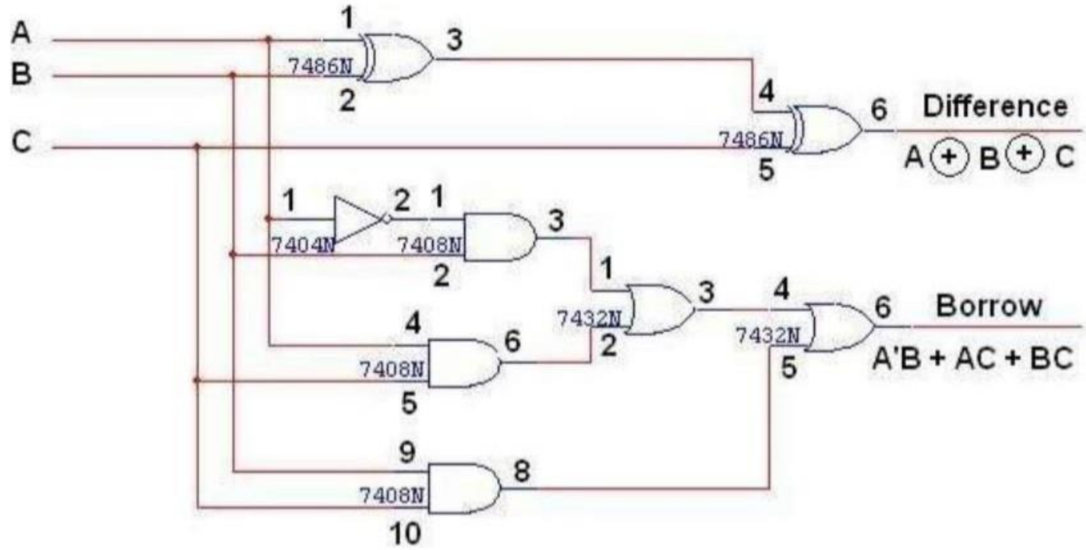
K-Map for Borrow



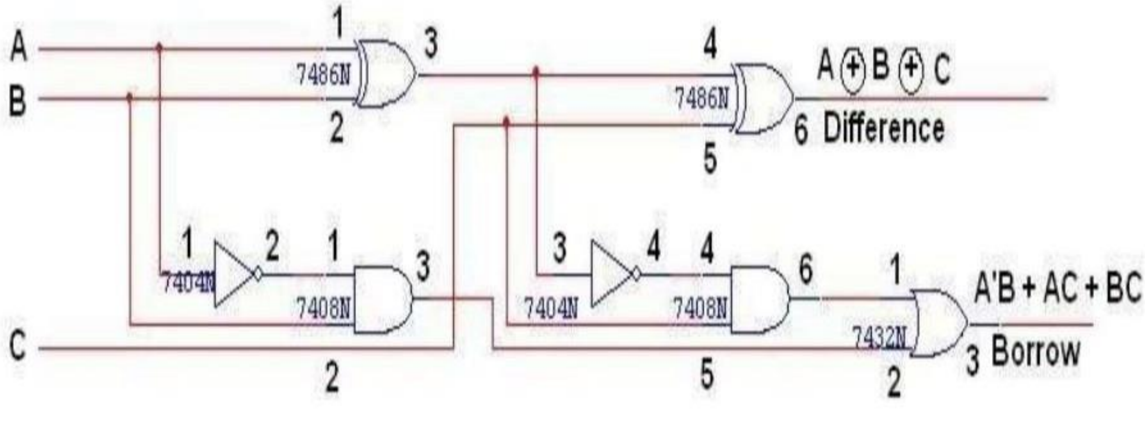
A \ BC	00	01	11	10
0	0	1	1	1
1	0	0	1	0

Borrow = A’B + BC + A’C

LOGIC DIAGRAM:



FULL SUBTRACTOR USING TWO HALF SUBTRACTOR



PROCEEDURE:

(i) Connections are given as per circuit diagram.

(ii) Logical inputs are given as per circuit diagram.

(iii) Observe the output and verify the truth table.

RESULT:

Thus, the half adder, full adder, half subtractor and full subtractor circuits are designed, constructed and verified the truth table using logic gates.

Ex.No. 3 MAGNITUDE COMPARATOR

AIM:

To design and implement the magnitude comparator using MSI device.

APPARATUS REQUIRED:

SL.NO.	COMPONENT	SPECIFICATION	QTY.
1.	AND GATE	IC 7408	2
2.	X-OR GATE	IC 7486	1
3.	OR GATE	IC 7432	1
4.	NOT GATE	IC 7404	1
5.	4-BIT MAGNITUDE COMPARATOR	IC 7485	2
6.	IC TRAINER KIT	-	1
7.	PATCH CORDS	-	30

THEORY:

The comparison of two numbers is an operator that determine one number is greater than, less than (or) equal to the other number. A magnitude comparator is a combinational circuit that compares two numbers A and B and determine their relative magnitude. The outcome of the comparator is specified by three binary variables that indicate whether A>B, A=B (or) A<B.

$A = A_3\ A_2\ A_1\ A_0$

$B = B_3\ B_2\ B_1\ B_0$

The equality of the two numbers and B is displayed in a combinational circuit designated by the symbol (A=B).

This indicates A greater than B, then inspect the relative magnitude of pairs of significant digits starting from most significant position. A is 0 and that of B is 0.

We have A<B, the sequential comparison can be expanded as

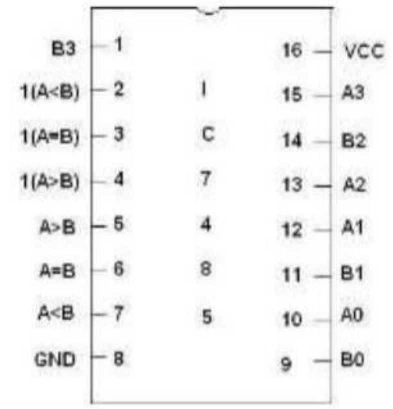
$$A>B = A3B_3{}^1 + X_3A_2B_2{}^1 + X_3X_2A_1B_1{}^1 + X_3X_2X_1A_0B_0{}^1$$

$$A<B = A_3{}^1B_3 + X_3A_2{}^1B_2 + X_3X2A_1{}^1B_1 + X_3X_2X_1A_0{}^1B_0$$

The same circuit can be used to compare the relative magnitude of two BCD digits. Where, A = B is expanded as,

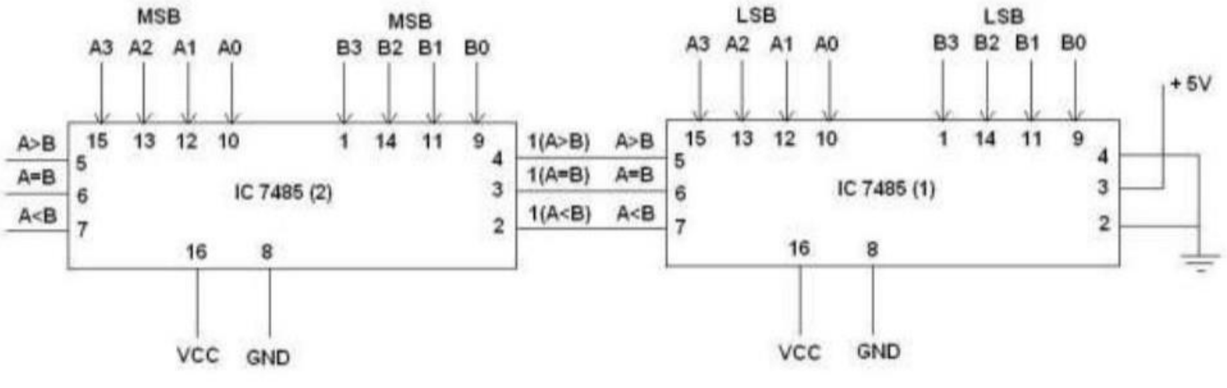
$$A = B = \underset{X3}{\underset{\downarrow}{(A_3 + B_3)}}\ \underset{X2}{\underset{\downarrow}{(A_2 + B_2)}}\ \underset{X1}{\underset{\downarrow}{(A_1 + B_1)}}\ \underset{X0}{\underset{\downarrow}{(A_0 + B_0)}}$$

PIN DIAGRAM FOR IC 7485:



8-BIT MAGNITUDE COMPARATOR

LOGIC DIAGRAM:



TRUTH TABLE:

A		B		A>B	A=B	A<B
0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	0	1	0
0 0 0 1	0 0 0 1	0 0 0 0	0 0 0 0	1	0	0
0 0 0 0	0 0 0 0	0 0 0 1	0 0 0 1	0	0	1

PROCEDURE:

(i) Connections are given as per circuit diagram.
(ii) Logical inputs are given as per circuit diagram.
(iii) Observe the output and verify the truth table.

RESULT:

Thus the magnitude comparator using MSI device is designed and implemented.

Ex.No.-4 MULTIPLEXER AND DEMULTIPLEXER

AIM:

To design and implement the multiplexer and demultiplexer using logic gates and study of IC 74150 and IC 74154.

APPARATUS REQUIRED:

SL.NO.	COMPONENT	SPECIFICATION	QTY.
1.	3 I/P AND GATE	IC 7411	2
2.	OR GATE	IC 7432	1
3.	NOT GATE	IC 7404	1
2.	IC TRAINER KIT	-	1
3.	PATCH CORDS	-	32

THEORY:

MULTIPLEXER:

Multiplexer means transmitting a large number of information units over a smaller number of channels or lines. A digital multiplexer is a combinational circuit that selects binary information from one of many input lines and directs it to a single output line. The selection of a particular input line is controlled by a set of selection lines. Normally there are 2^n input line and n selection lines whose bit combination determine which input is selected.

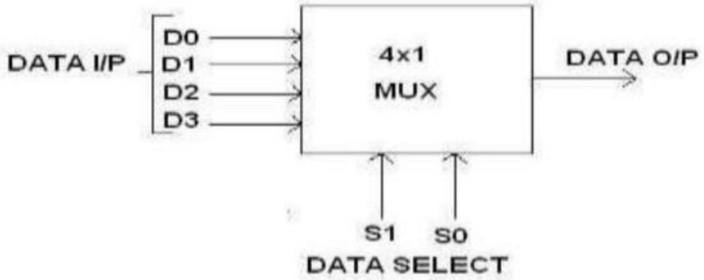
DEMULTIPLEXER:

The function of Demultiplexer is in contrast to multiplexer function. It takes information from one line and distributes it to a given number of output lines. For this reason, the demultiplexer is also known as a data distributor. Decoder can also be used as demultiplexer.

In the 1: 4 demultiplexer circuit, the data input line goes to all of the AND gates. The data select lines enable only one gate at a time and the data on the data input line will pass through the selected gate to the associated data output line.

4:1 MULTIPLEXER

BLOCK DIAGRAM FOR 4:1 MULTIPLEXER:



FUNCTION TABLE:

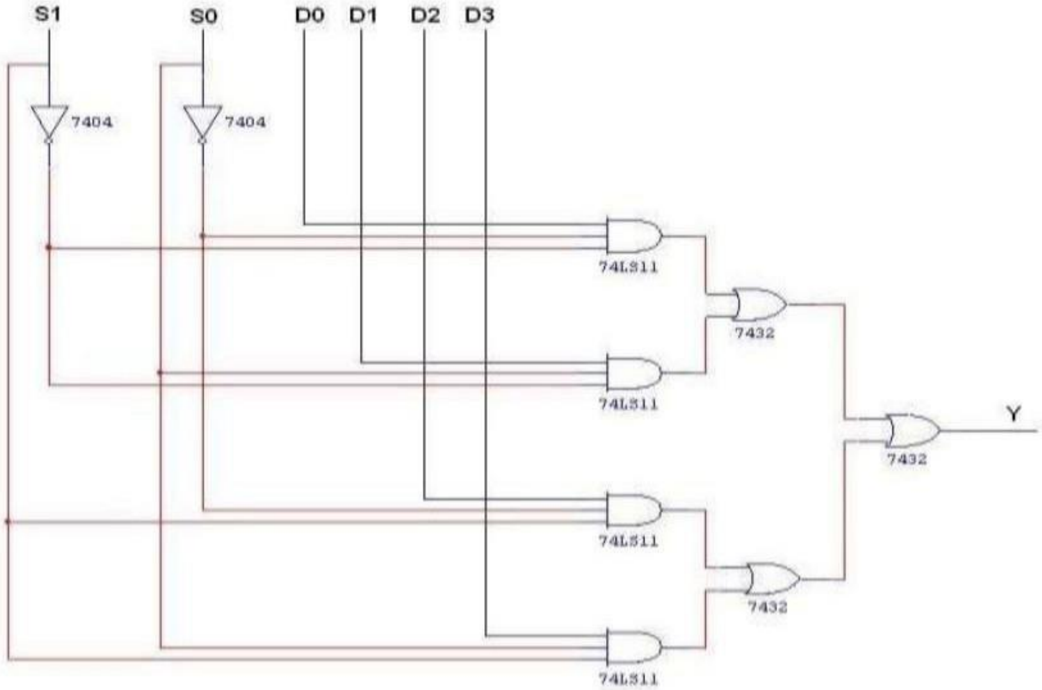
S1	S0	INPUTS Y
0	0	D0 → D0 S1' S0'
0	1	D1 → D1 S1' S0
1	0	D2 → D2 S1 S0'
1	1	D3 → D3 S1 S0

Y = D0 S1' S0' + D1 S1' S0 + D2 S1 S0' + D3 S1 S0

TRUTH TABLE:

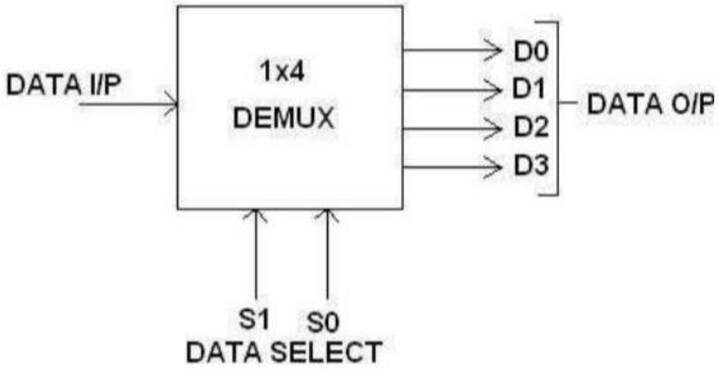
S1	S0	Y = OUTPUT
0	0	D0
0	1	D1
1	0	D2
1	1	D3

CIRCUIT DIAGRAM FOR MULTIPLEXER:



1:4 DEMULTIPLEXER

BLOCK DIAGRAM FOR 1:4 DEMULTIPLEXER:



FUNCTION TABLE:

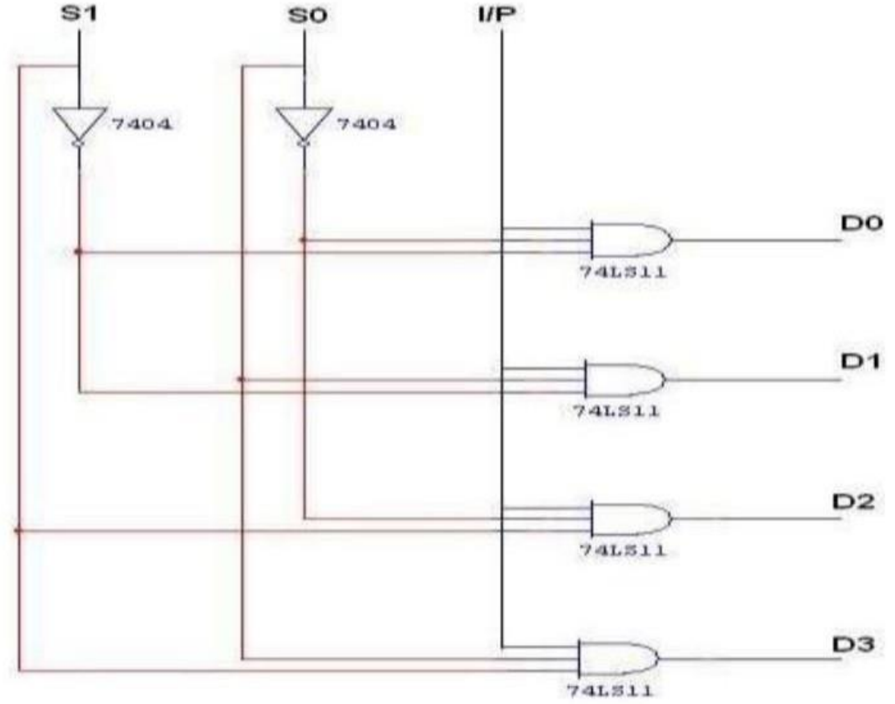
S1	S0	INPUT
0	0	X → D0 = X S1’ S0’
0	1	X → D1 = X S1’ S0
1	0	X → D2 = X S1 S0’
1	1	X → D3 = X S1 S0

Y = X S1’ S0’ + X S1’ S0 + X S1 S0’ + X S1 S0

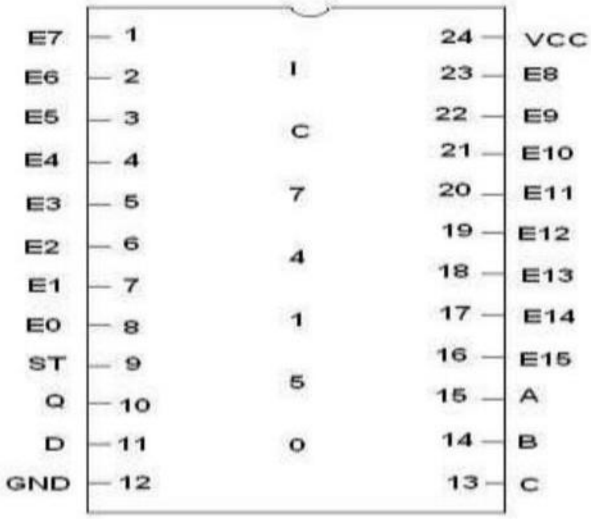
TRUTH TABLE:

INPUT			OUTPUT			
S1	**S0**	**I/P**	**D0**	**D1**	**D2**	**D3**
0	0	0	0	0	0	0
0	0	1	1	0	0	0
0	1	0	0	0	0	0
0	1	1	0	1	0	0
1	0	0	0	0	0	0
1	0	1	0	0	1	0
1	1	0	0	0	0	0
1	1	1	0	0	0	1

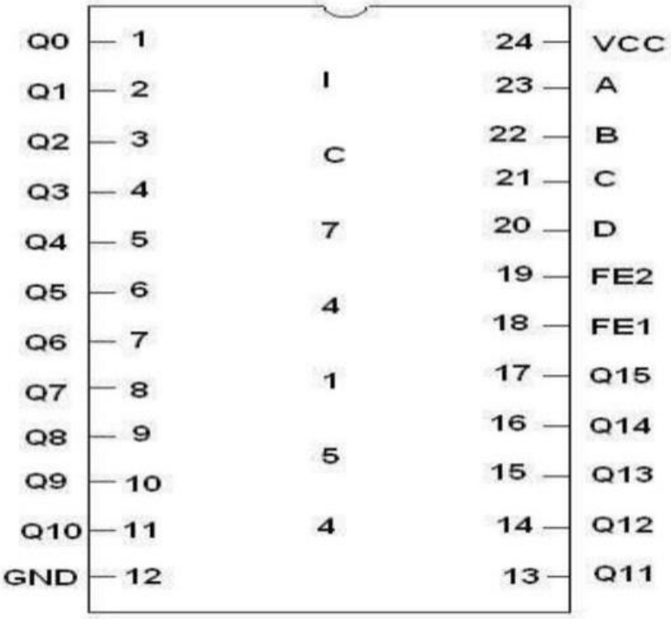
LOGIC DIAGRAM FOR DEMULTIPLEXER:



PIN DIAGRAM FOR IC 74150:



PIN DIAGRAM FOR IC 74154:



PROCEDURE:

(i) Connections are given as per circuit diagram.
(ii) Logical inputs are given as per circuit diagram.
(iii) Observe the output and verify the truth table.

RESULT:

Thus the multiplexer and demultiplexer using logic gates are designed and implemented.

Ex.No.-5 SHIFT REGISTER

AIM:

To design and implement the following shift registers

(i) Serial in serial out
(ii) Serial in parallel out
(iii) Parallel in serial out
(iv) Parallel in parallel out

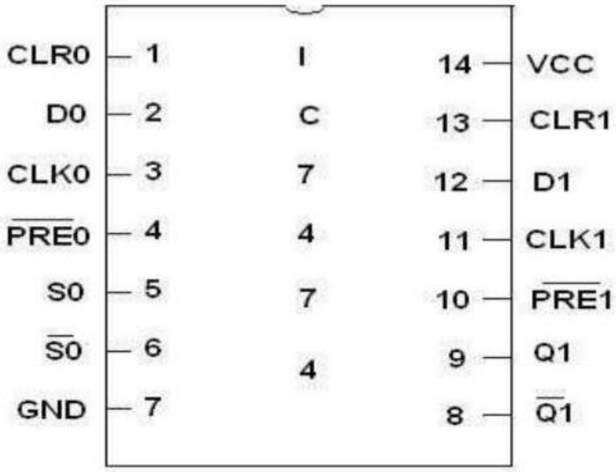
APPARATUS REQUIRED:

SL.NO.	COMPONENT	SPECIFICATION	QTY.
1.	D FLIP FLOP	IC 7474	2
2.	OR GATE	IC 7432	1
3.	IC TRAINER KIT	-	1
4.	PATCH CORDS	-	35

THEORY:

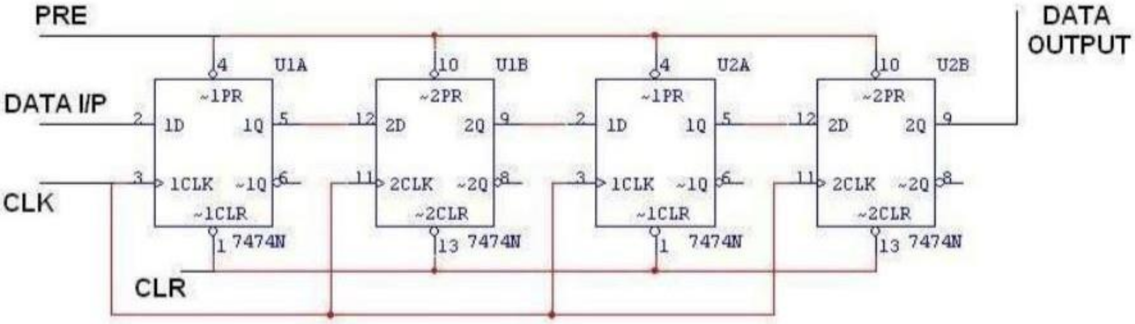
A register is capable of shifting its binary information in one or both directions is known as shift register. The logical configuration of shift register consist of a D-Flip flop cascaded with output of one flip flop connected to input of next flip flop. All flip flops receive common clock pulses which causes the shift in the output of the flip flop. The simplest possible shift register is one that uses only flip flop. The output of a given flip flop is connected to the input of next flip flop of the register. Each clock pulse shifts the content of register one bit position to right.

PIN DIAGRAM OF IC 7474:



SERIAL IN SERIAL OUT

LOGIC DIAGRAM:

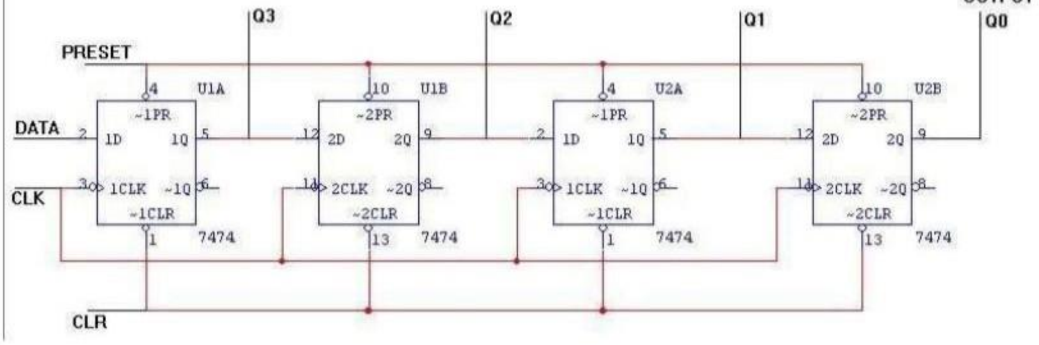


TRUTH TABLE:

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:

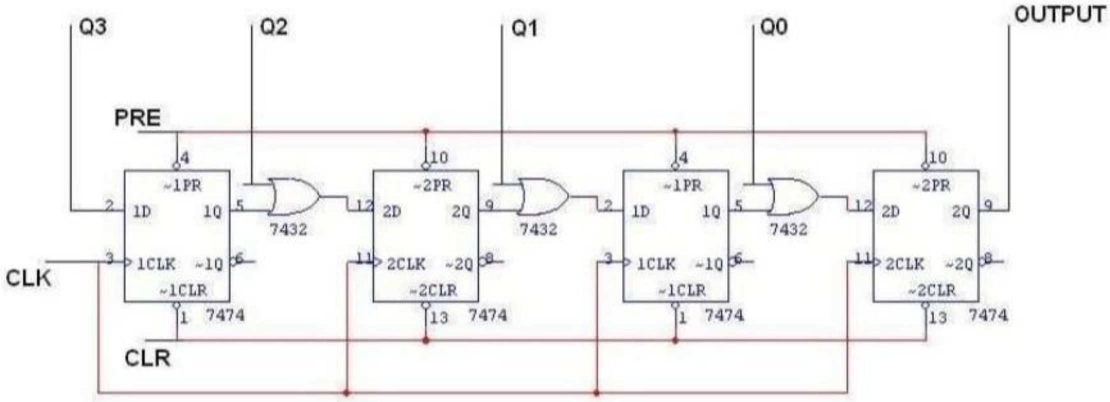


TRUTH TABLE:

CLK	DATA	OUTPUT			
		Q_A	Q_B	Q_C	Q_D
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:

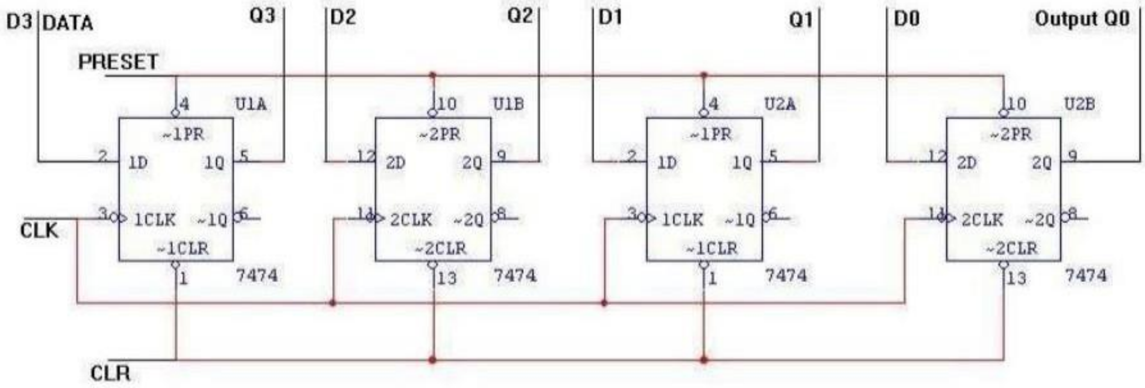


TRUTH TABLE:

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:

(i) Connections are given as per circuit diagram.
(ii) Logical inputs are given as per circuit diagram.
(iii) Observe the output and verify the truth table.

RESULT:

The Serial in serial out, Serial in parallel out, Parallel in serial out and Parallel in parallel out shift registers are designed and implemented.

Ex.No.-6 SYNCHRONOUS AND ASYNCHRONOUS COUNTER

AIM:

To design and implement synchronous and asynchronous counter.

APPARATUS REQUIRED:

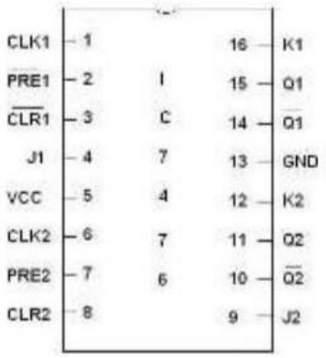
S.NO.	NAME OF THE APPARATUS	RANGE	QUANTITY
1.	Digital IC trainer kit		1
2.	JK Flip Flop	IC 7473	2
3.	D Flip Flop	IC 7473	1
4.	NAND gate	IC 7400	1
5.	Connecting wires		As required

THEORY:

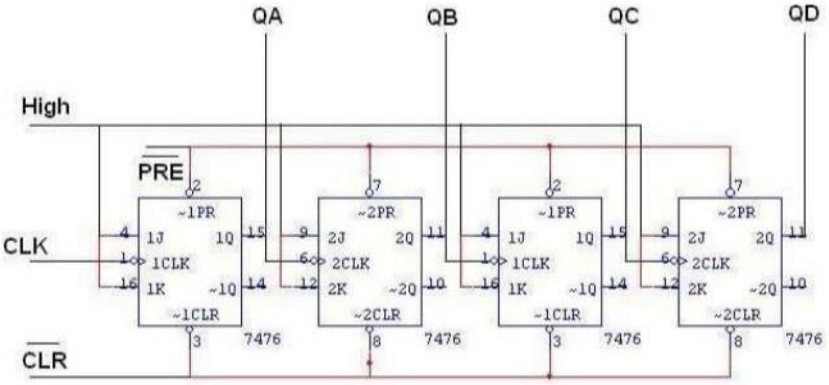
Asynchronous decade counter is also called as ripple counter. In a ripple counter the flip flop output transition serves as a source for triggering other flip flops. In other words the clock pulse inputs of all the flip flops are triggered not by the incoming pulses but rather by the transition that occurs in other flip flops. The term asynchronous refers to the events that do not occur at the same time. With respect to the counter operation, asynchronous means that the flip flop within the counter are not made to change states at exactly the same time, they do not because the clock pulses are not connected directly to the clock input of each flip flop in the counter.

A counter is a register capable of counting number of clock pulse arriving at its clock input. Counter represents the number of clock pulses arrived. A specified sequence of states appears as counter output. This is the main difference between a register and a counter. There are two types of counter, synchronous and asynchronous. In synchronous common clock is given to all flip flop and in asynchronous first flip flop is clocked by external pulse and then each successive flip flop is clocked by Q or Q output of previous stage. A soon the clock of second stage is triggered by output of first stage. Because of inherent propagation delay time all flip flops are not activated at same time which results in asynchronous operation.

PIN DIAGRAM FOR IC 7476:



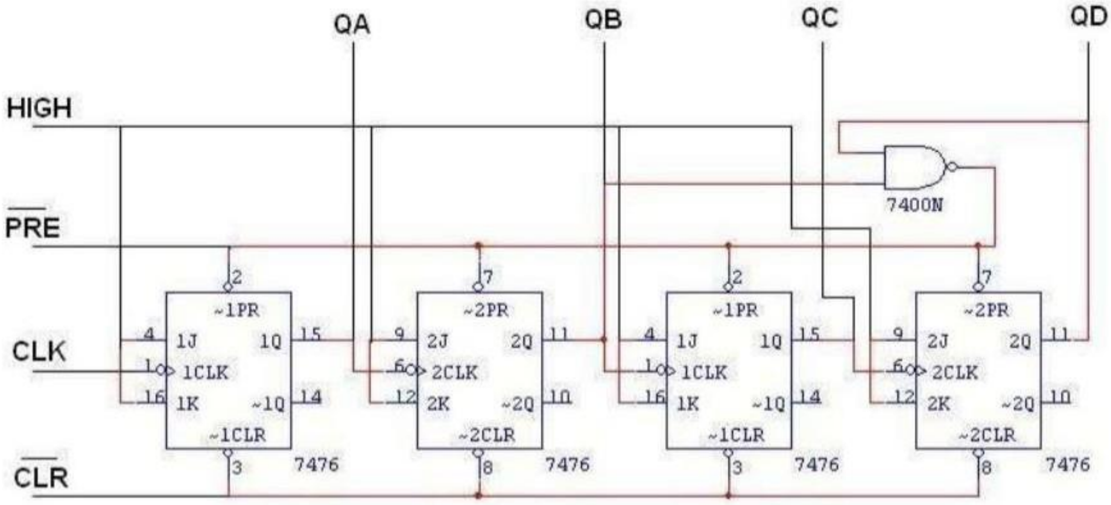
CIRCUIT DIAGRAM:



TRUTH TABLE:

CLK	QA	QB	QC	QD
0	0	0	0	0
1	1	0	0	0
2	0	1	0	0
3	1	1	0	0
4	0	0	1	0
5	1	0	1	0
6	0	1	1	0
7	1	1	1	0
8	0	0	0	1
9	1	0	0	1
10	0	1	0	1
11	1	1	0	1
12	0	0	1	1
13	1	0	1	1
14	0	1	1	1
15	1	1	1	1

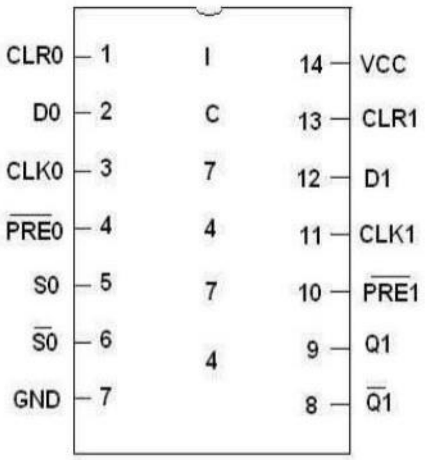
LOGIC DIAGRAM FOR MOD - 10 RIPPLE COUNTER:



TRUTH TABLE:

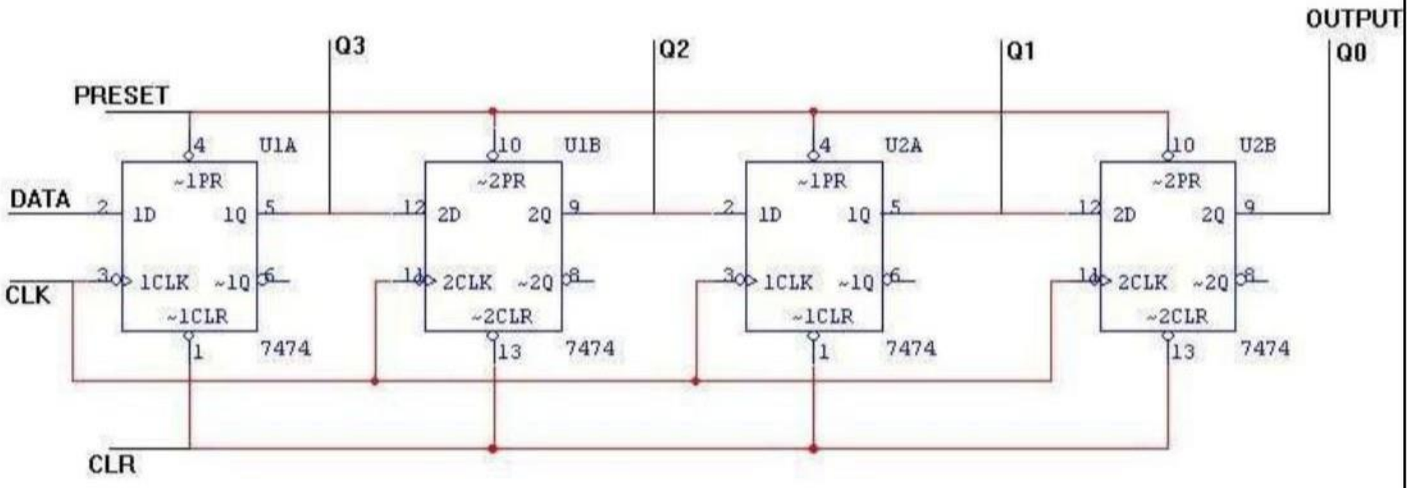
CLK	QA	QB	QC	QD
0	0	0	0	0
1	1	0	0	0
2	0	1	0	0
3	1	1	0	0
4	0	0	1	0
5	1	0	1	0
6	0	1	1	0
7	1	1	1	0
8	0	0	0	1
9	1	0	0	1
10	0	0	0	0

PIN DIAGRAM:


CLR0 — 1
D0 — 2
CLK0 — 3
$\overline{\text{PRE0}}$ — 4
S0 — 5
$\overline{\text{S0}}$ — 6
GND — 7
I
C
7
4
7
4
14 — VCC
13 — CLR1
12 — D1
11 — CLK1
10 — $\overline{\text{PRE1}}$
9 — Q1
8 — $\overline{\text{Q1}}$

SYNCHRONOUS COUNTER

LOGIC DIAGRAM:



TRUTH TABLE:

CLK	DATA	OUTPUT			
		Q_A	Q_B	Q_C	Q_D
1	1	1	0	0	0
2	0	0	1	0	0
3	0	0	0	1	1
4	1	1	0	0	1

PROCEDURE:

(i) Connections are given as per circuit diagram.
(ii) Logical inputs are given as per circuit diagram.
(iii) Observe the output and verify the truth table.

RESULT:

Thus the synchronous and asynchronous counter are designed and implemented.