SYLLABUS

1. Design and implementation of Adders and Subtractors using logic gates.
2. Design and implementation of code converters using logic gates
   (i) BCD to excess-3 code and vice versa
   (ii) Binary to gray and vice-versa
3. Design and implementation of 4 bit binary Adder/ subtractor and BCD adder using IC 7483
4. Design and implementation of 2 Bit Magnitude Comparator using logic gates 8 Bit Magnitude Comparator using IC 7485
5. Design and implementation of 16 bit odd/even parity checker /generator using IC74180.
6. Design and implementation of Multiplexer and De-multiplexer using logic gates and study of IC74153 and IC 74139
7. Design and implementation of encoder and decoder using logic gates and study of IC7445 and IC74147
8. Construction and verification of 4 bit Asynchronous (ripple) counter.
9. Construction and verification of 4 bit Asynchronous (ripple) counter.
10. Construction and verification of Mod 10 and Mod 12 counter.
11. Implementation of SISO, SIPO, PISO and PIPO shift registers using Flip- flops.
12. Design of expts 1,6,8,10 using NI MultiSim Software.

LIST OF EXPERIMENTS

1. Study of logic gates.
2. Design and implementation of adder
3. Design and implementation of subtractor using logic gates.
4. Design and implementation of code converters using logic gates.
5. Design and implementation of 4-bit binary adder/subtractor
6. Study and verify the performance of BCD adder using IC 7483.
7. Design and implementation of 2-bit magnitude comparator using logic gates and IC7485
8. Design and implementation of multiplexer and demultiplexer using logic gates and study of IC 74153 and IC 74139.
9. Design and implementation of encoder and decoder using logic gates and study of IC 7442 and IC 74147.
10. Study and verify the performance of SR, JK and D flip-flops using logic gates.
11. Construct and verify the truth table of 4-bit Asynchronous (Ripple) counter.
12. Construct and verify the truth table of 4-bit Synchronous counter.
13. Design and implementation of 3-bit synchronous up/down counter.
14. Construct and verify the truth table of Asynchronous Mod-10/Mod-12 counter.
15. Construct and verify the truth table of Johnson counter
16. Implementation of SISO, SIPO, PISO and PIPO shift registers using flip-flops.

INDEX

EXP. NO	DATE	NAME OF THE EXPERIMENT	PAGE NO	MARKS	SIGNATURE

Date: Expt. No.:	STUDY OF LOGIC GATES

AIM:

To study about logic gates and verify their truth tables.

APPARATUS REQUIRED:

SL No.	COMPONENT	SPECIFICATION	QTY
1.	AND GATE	IC 7408	1
2.	OR GATE	IC 7432	1
3.	NOT GATE	IC 7404	1
4.	NAND GATE 2 I/P	IC 7400	1
5.	NOR GATE	IC 7402	1
6.	X-OR GATE	IC 7486	1
7.	NAND GATE 3 I/P	IC 7410	1
8.	IC TRAINER KIT	-	1
9.	PATCH CORD	-	14

THEORY:

Circuit that takes the logical decision and the process are called logic gates. Each gate has one or more input and only one output.

OR, AND and NOT are basic gates. NAND, NOR and X-OR are known as universal gates. Basic gates form these gates.

AND GATE:

The AND gate performs a logical multiplication commonly known as AND function. The output is high when both the inputs are high. The output is low level when any one of the inputs is low.

OR GATE:

The OR gate performs a logical addition commonly known as OR function. The output is high when any one of the inputs is high. The output is low level when both the inputs are low.

NOT GATE:

The NOT gate is called an inverter. The output is high when the input is low. The output is low when the input is high.

NAND GATE:

The NAND gate is a contraction of AND-NOT. The output is high when both inputs are low and any one of the input is low .The output is low level when both inputs are high.

NOR GATE:

The NOR gate is a contraction of OR-NOT. The output is high when both inputs are low. The output is low when one or both inputs are high.

X-OR GATE:

The output is high when any one of the inputs is high. The output is low when both the inputs are low and both the inputs are high.

PROCEDURE:

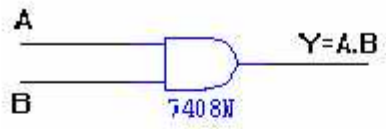
(i) Connections are given as per circuit diagram.

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

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

AND GATE:

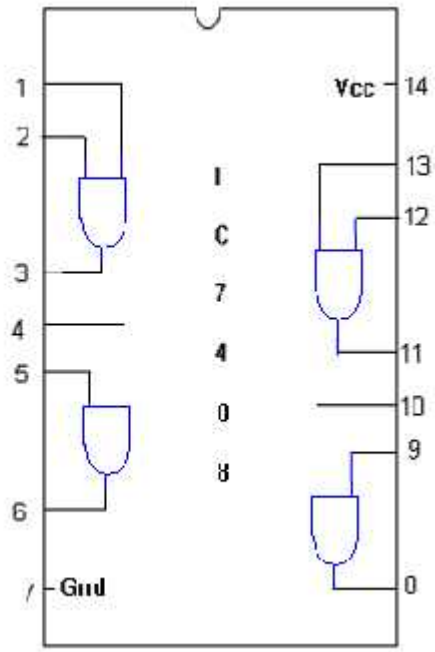
SYMBOL:



TRUTH TABLE

A	B	A.B
0	0	0
0	1	0
1	0	0
1	1	1

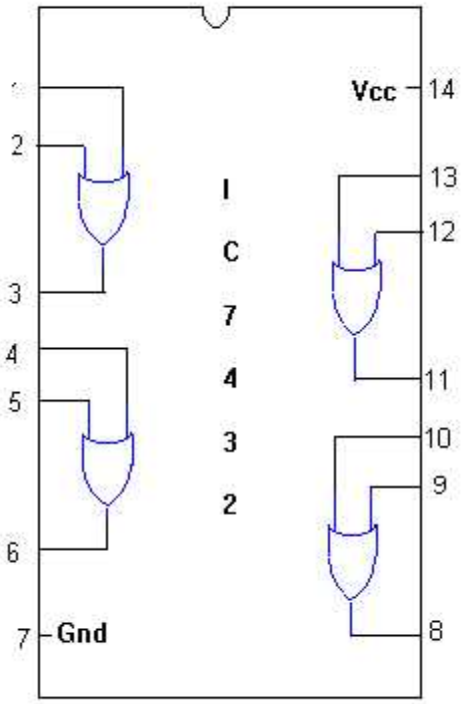
PIN DIAGRAM:



OR GATE:

SYMBOL :

PIN DIAGRAM :



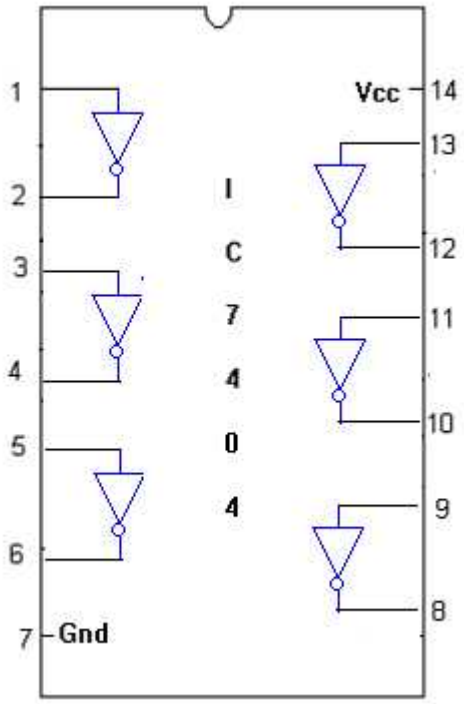
TRUTH TABLE

A	B	A+B
0	0	0
0	1	1
1	0	1
1	1	1

NOT GATE:

SYMBOL:

PIN DIAGRAM:

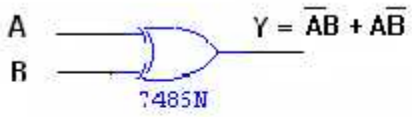


TRUTH TABLE :

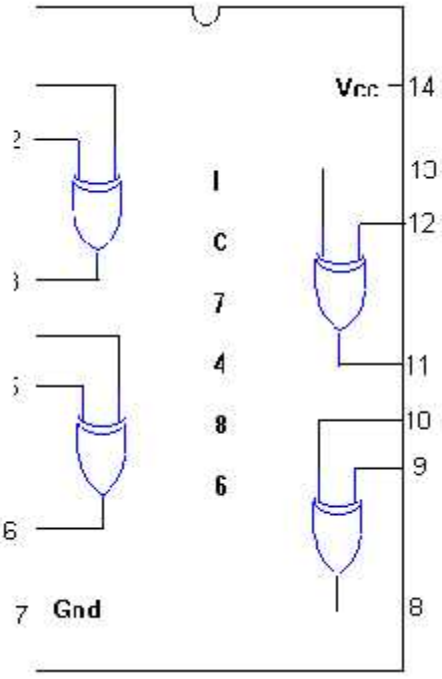
A	$\overline{A}$
0	1
1	0

X-OR GATE :

SYMBOL :



PIN DIAGRAM :

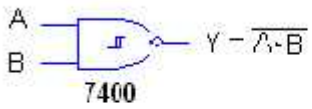


TRUTH TABLE :

A	B	$\overline{A}B + A\overline{B}$
0	0	0
0	1	1
1	0	1
1	1	0

2-INPUT NAND GATE:

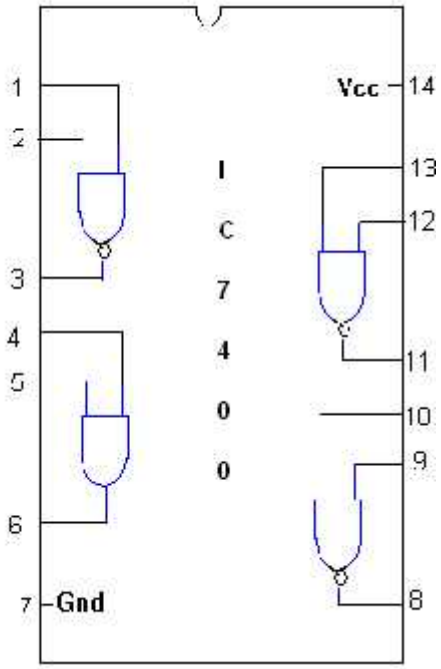
SYMBOL:



TRUTH TABLE

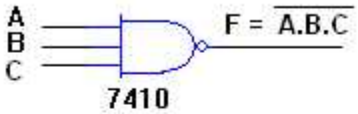
A	B	$\overline{A \cdot B}$
0	0	1
0	1	1
1	0	1
1	1	0

PIN DIAGRAM:

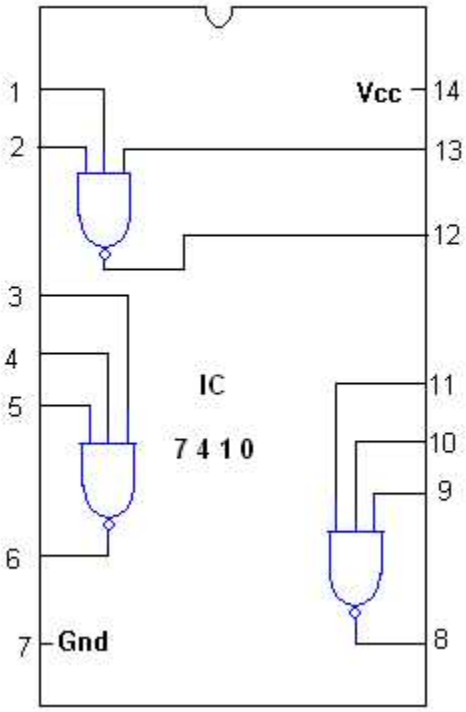


3-INPUT NAND GATE :

SYMBOL :



PIN DIAGRAM :

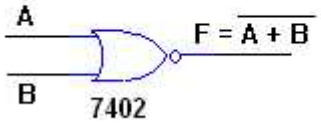


TRUTH TABLE

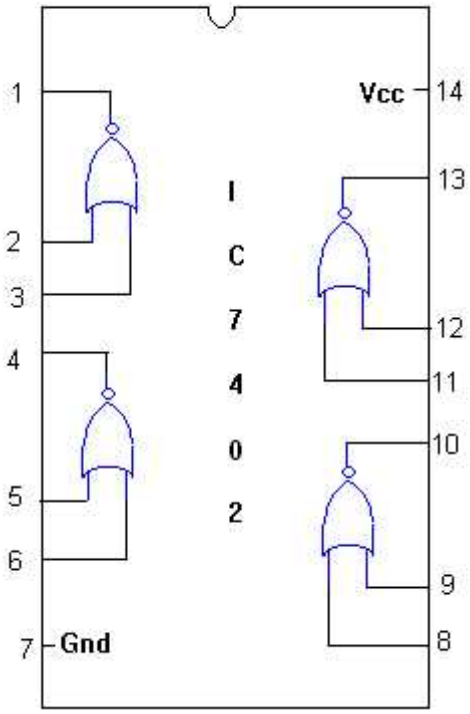
A	B	C	$\overline{A.B.C}$
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

NOR GATE:

SYMBOL :



PIN DIAGRAM :



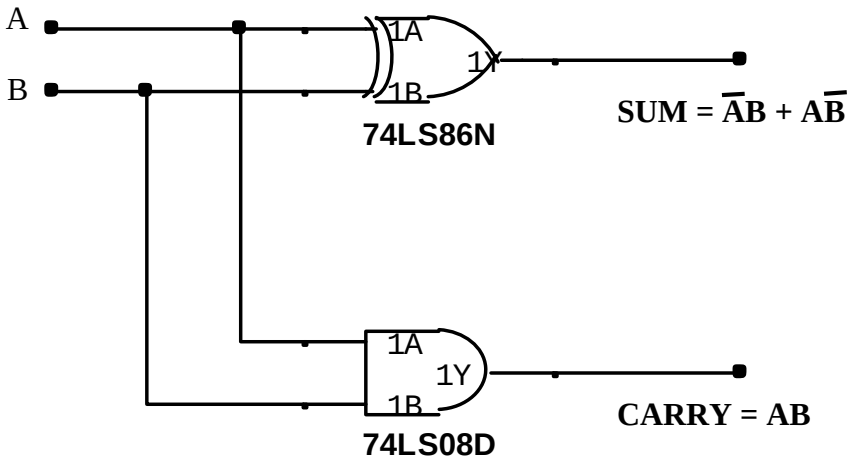
TRUTH TABLE

A	B	$\overline{A+B}$
0	0	1
0	1	1
1	0	1
1	1	0

RESULT:

LOGIC DIAGRAM:

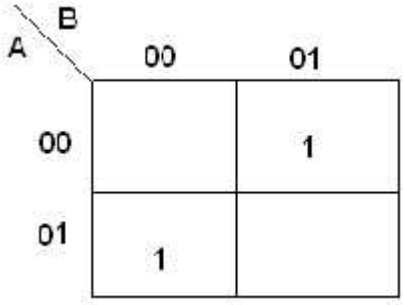
HALF ADDER



TRUTH TABLE:

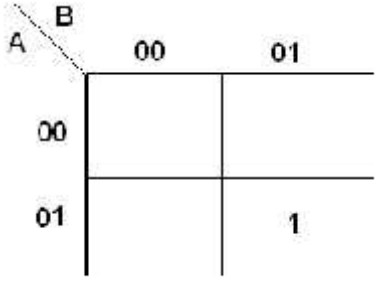
A	B	CARRY	SUM
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

K-Map for SUM:



SUM = $\overline{A}B + A\overline{B}$

K-Map for CARRY:



CARRY = AB

Date: Expt. No.:	DESIGN OF HALF ADDER AND FULL ADDER

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.	EX-OR GATE	IC 7486	1
3.	OR GATE	IC 7432	1
4.	IC TRAINER KIT	-	1
5.	PATCH CORDS	-	Adequate

THEORY:

HALF ADDER:

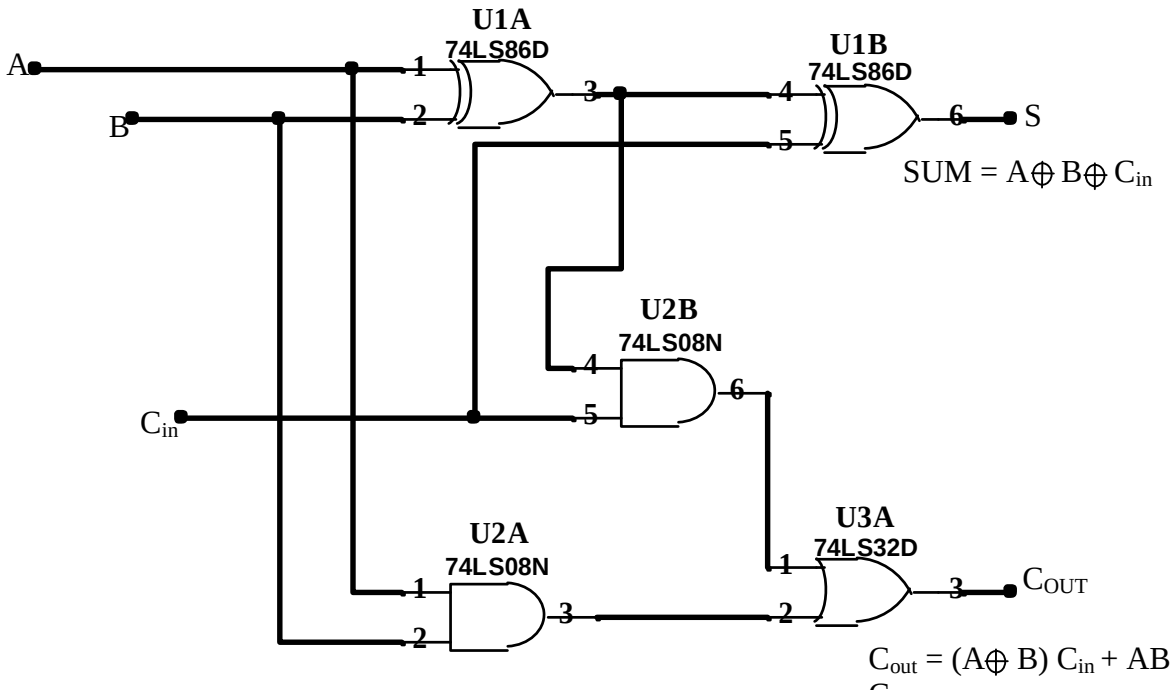
Half adder is a combinational circuit which consists of two binary input variables called augend and addend and two binary output variables called sum and carry. In the addition result, the lower significant bit is called as sum and the higher significant bit is called as carry. The truth table of the half adder is given below, in that the sum becomes logic ‘1’ when any one of the input is maximum and it is equal to logic ‘0’ when both inputs are equal. The carry is equal to logic ‘1’ when both inputs are equal to logic ‘1’ unless it is equal to logic ‘0’.

FULL ADDER:

Full adder is a combinational circuit which consists of three binary input variables called augends and addends and two binary output variables called sum and carry out. In the addition result, the lower significant bit is called as sum and the higher significant bit is called as carry out. The truth table of the full adder describes all the eight possible input variations. The full adder results the outputs are equal to logic ‘0’ when all the applied inputs are equal to logic ‘0’ and the outputs are equal to logic ‘1’ when all the inputs are equal to logic ‘1’. The sum is equal to 1 when odd numbers of inputs are equal to 1 from the applied three inputs. The carry out is equal to 1 if more than one applied inputs are equal to 1.

LOGIC DIAGRAM:

FULL ADDER USING TWO HALF ADDER



TRUTH TABLE:

Inputs			Outputs	
A	**B**	**C_{in}**	**C_{out}**	**S**
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_{in}	00	01	11	10
0	0	1	0	1
1	1	0	1	0

Sum (S) = $\bar{A}\bar{B}C_{in} + \bar{A}B\bar{C}_{in} + A\bar{B}\bar{C}_{in} + ABC_{in}$

$= (\bar{A}\bar{B} + AB)C_{in} + (\bar{A}B + A\bar{B})\bar{C}_{in}$

$= (A \odot B)\, C_{in} + (A \oplus B)\, \bar{C}_{in}$

$= \overline{(A \oplus B)}\, C_{in} + (A \oplus B)\, \bar{C}_{in}$

SUM (S) = $\mathbf{A \oplus B \oplus C_{in}}$

K-Map for C_{out}:

A \ BC_{in}	00	01	11	10
0	0	0	1	0
1	0	1	1	1

$C_{out} = BC_{in} + AC_{in} + ABC_{in} + AB$

$= (A + \bar{A})\, BC_{in} + A\,(B + \bar{B})C_{in} + AB$

$= ABC_{in} + \bar{A}BC_{in} + ABC_{in} + A\bar{B}C_{in} + AB$

$= (\bar{A}B + A\bar{B})\, C_{in} + ABC_{in} + AB$

C_{out} = $\mathbf{(A \oplus B)C_{in} + AB}$

PROCEEDURE:

(i) Verify the truth table of the given Logic Gates.

(ii) Connection to be made as per the circuit diagram.

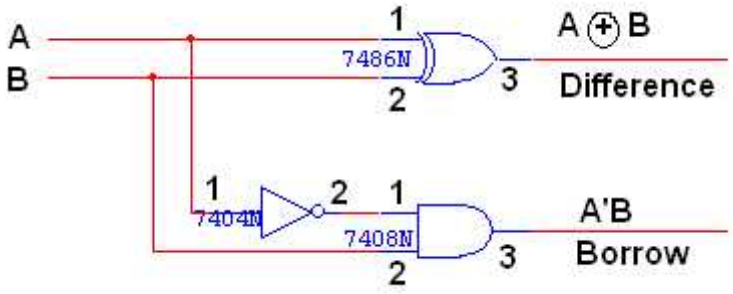
(iii) All the possible input variations are to be given.

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

RESULT:

LOGIC DIAGRAM:

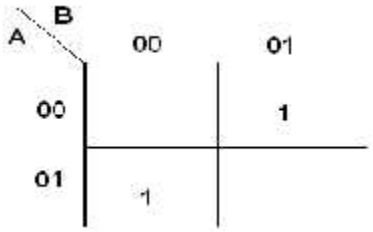
HALF SUBTRACTOR



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:



DIFFERENCE = $\overline{A}B + A\overline{B}$

K-Map for BORROW:

BORROW = $\overline{A}B$

Date: Expt. No.:	DESIGN OF HALF SUBTRACTOR AND FULL 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.	EX-OR GATE	IC 7486	1
3.	OR GATE	IC 7432	1
4.	NOT GATE	IC7040	1
5.	IC TRAINER KIT	-	1
6.	PATCH CORDS	-	Adequate

THEORY:

HALF SUBTRACTOR:

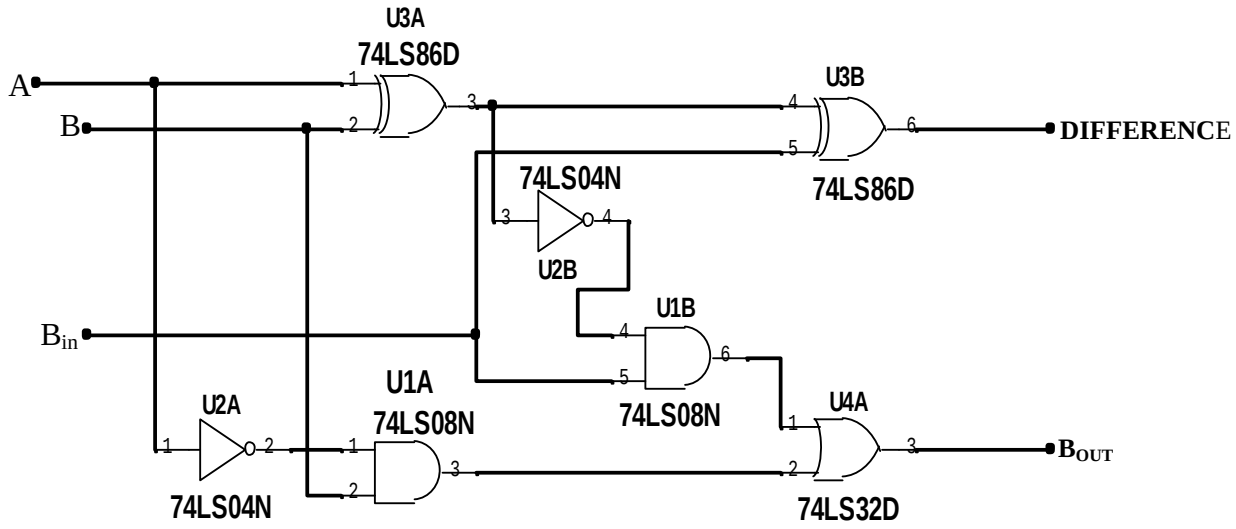
Half subtractor is a combinational circuit which consists of two binary input variables called minuend and subtrahend, and two binary output variables called difference and borrow. In the two bit subtraction result, the lower significant bit is called as difference and higher significant bit is called as borrow. The truth table of the subtractor is given below in that the difference becomes logic ‘1’ when both inputs are different each other and it is equal to logic ‘0’ when both inputs are equal. And borrow is equal to logic ‘1’ when minuend is smaller than subtrahend.

FULL SUBTRACTOR:

Full subtractor is a combinational circuit which consists of three binary input variables called minuend and subtrahend, and two binary output variables called difference and borrow out. In the subtraction result, the lower significant bit is called as difference and the higher significant bit is called as barrow out. The truth table of the full subtractor describes all the eight possible input variations. The full subtractor results the outputs are equal to logic ‘0’ when all the applied inputs are equal to logic ‘0’

LOGIC DIAGRAM:

FULL SUBTRACTOR USING TWO HALF SUBTRACTOR



TRUTH TABLE

Inputs			Outputs	
A	**B**	**B_{in}**	**B_{out}**	**D**
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**

or most significant bit and any one of the least significant bit is equal to logic '1', and the outputs are equal to logic '1' when all the inputs are equal to logic '1' or any one of the subtrahend is equal to logic '1'. The difference is equal to 1 when odd numbers of inputs are equal to 1 from the applied three inputs. The barrow out is equal to 1 if any one of the subtrahend or all the applied inputs are equal to logic '1'.

PROCEEDURE:

(i) Verify the truth table of the given Logic Gates.
(ii) Connection to be made as per the circuit diagram.
(iii) All the possible input variations are to be given.
(iv) Observe the output and verify the truth table.

K-Map for Difference:

A \ BB_{in}	00	01	11	10
0	0	1	0	1
1	1	0	1	0

Sum (S) = $\bar{A}\bar{B}B_{in} + \bar{A}B\bar{B}_{in} + A\bar{B}\bar{B}_{in} + ABC_{in}$

$= (\bar{A}\bar{B} + AB)B_{in} + (\bar{A}B + A\bar{B})\bar{B}_{in}$

$= (A \odot B)\, B_{in} + (A \oplus B)\, \bar{B}_{in}$

$= \overline{(A \oplus B)}\, B_{in} + (A \oplus B)\, \bar{B}_{in}$

SUM (S) = $\mathbf{A \oplus B \oplus B_{in}}$

K-Map for Borrow (B_{out}):

A \ BB_{in}	00	01	11	10
0	0	1	1	1
1	0	0	1	0

BORROW (B_{out}) = $\bar{A}B + \bar{A}B_{in} + BB_{in}$

$= \bar{A}B + \bar{A}\,(B + \bar{B})B_{in} + (\bar{A} + A)\, BB_{in}$

$= \bar{A}B + \bar{A}BB_{in} + \bar{A}\bar{B}B_{in} + \bar{A}BB_{in} + ABB_{in}$

$= \bar{A}B + \bar{A}BB_{in} + \bar{A}\bar{B}B_{in} + ABB_{in}$

$= \bar{A}B + \bar{A}BB_{in} + (\bar{A}\bar{B} + AB)\, B_{in}$

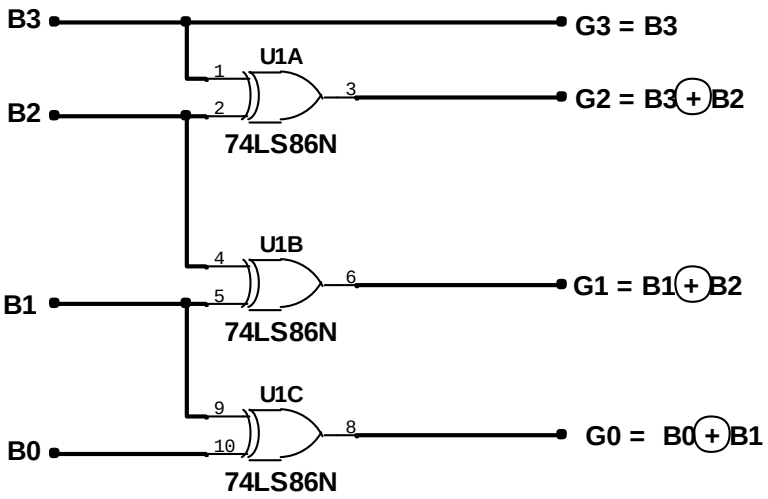
$= \bar{A}B + (A \odot B)\, B_{in}$

$\mathbf{(B_{out}) = \bar{A}B + \overline{(A \oplus B)}\, B_{in}}$

RESULT:

LOGIC DIAGRAM:

BINARY TO GRAY CODE CONVERTOR



TRUTH TABLE:

Inputs (BINARY CODE)				Outputs (GRAY CODE)			
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

Date: Expt. No.:	**DESIGN AND IMPLEMENTATION OF 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:

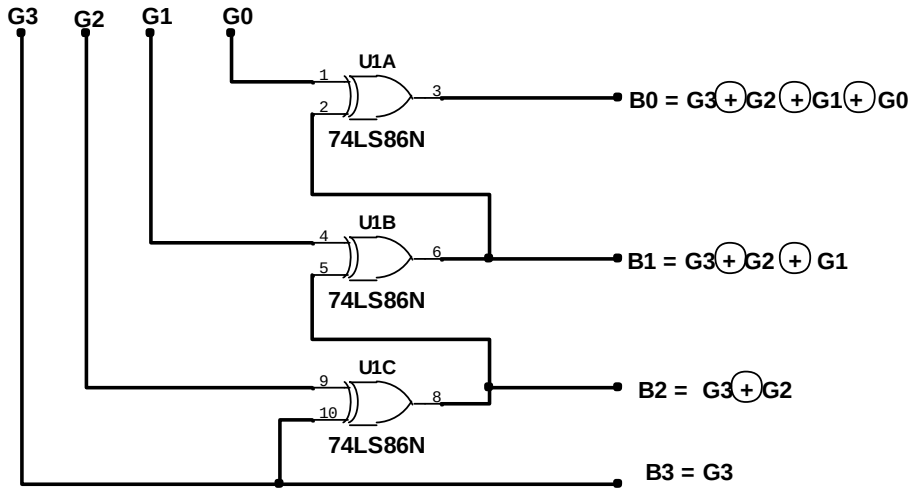
BINARY CODES: A group of binary bits that used to represent the characters, numbers and symbols is defined as binary codes. Binary codes are used in the digital computer to represent, store and transmit various data.

BCD NUMBERS: BCD numbers are straight binary representation for decimal numbers. The decimal numbers ate directly represented with the weightages of 8421 in BCD code. This is popularly used in decimal addition, subtraction, etc. the BCD code represents the decimal number 0 to 9 with the binary representation 0000 to 1001. In the 4-bit binary representation last six assignments are discarded for BCD number representation.

EXCESS – 3 CODE: The 4-bit excess – 3 code is obtained by adding 3(0011) with BCD code. 8421 and 2421 weighted codes provide the self-complement number of excess – 3 code in the binary representation. The self-complement property of excess – 3 code helps to perform the arithmetic operation in digital system design.

LOGIC DIAGRAM:

GRAY TO BINARY CODE CONVERTOR



TRUTH TABLE:

Inputs (GRAY CODE)				Outputs (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	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

BINARY TO GRAY CODE CONVERTOR – K-MAP

K-map for G3:

B3B2 \ B1B0	00	01	11	10
00	0	0	0	0
01	0	0	0	0
11	1	1	1	1
10	1	1	1	1

G3 = B3

K-map for G2:

B3B2 \ B1B0	00	01	11	10
00	0	0	0	0
01	1	1	1	1
11	0	0	0	0
10	1	1	1	1

$$G2 = \overline{B3}B2 + B3\overline{B2} = B3 \oplus B2$$

K-map for G1:

B3B2 \ B1B0	00	01	11	10
00	0	0	1	1
01	1	1	0	0
11	1	1	0	0
10	0	0	1	1

$$G1 = \overline{B1}B2 + B1\overline{B2} = B1 \oplus B2$$

K-map for G0:

B3B2 \ B1B0	00	01	11	10
00	0	1	0	1
01	0	1	0	1
11	0	1	0	1
10	0	1	0	1

$$G0 = \overline{B1}B0 + B1\overline{B0} = B1 \oplus B0$$

GRAY TO BINARY CODE CONVERTOR – K-MAP

K-map for B3:

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

G3 = B3

K-map for B2:

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

$$B2 = \overline{G3}G2 + G3\overline{G2} = G3 \oplus G2$$

GRAY CODE: This code is an un-weighted binary code. A gray code is often used in the translation of an analog quantity, such as a shaft position in to digital form. The four bit gray code can be used to represent the decimal number from 0 to 15. In this representation the last and first entry of gray code consequently differs only in one bit position (MSB bit). So this is also called reflective code.

CODE CONVERTERS: The presence of different codes in digital system for the same discrete elements of binary information results the requirement of code conversion. Code converter is a logic circuit that converts one type of binary code into another type of binary code.

PROCEDURE:

1. Construct the truth table to convert one form of code to another form.
2. Verify the Boolean expression using K-map for the output variables.
3. Rig the circuit diagram for simplified Boolean expressions.
4. Verify the truth table.

GRAY TO BINARY CODE CONVERTOR – K-MAP

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

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

BCD TO EXCESS-3 CODE CONVERTOR - K – MAP:

K-map for E3:

B3B2 \ B1B0	00	01	11	10
00	0	0	0	0
01	0	1	1	1
11	X	X	X	X
10	1	1	X	X

$E3 = B3 + B2\,(B1 + B0)$

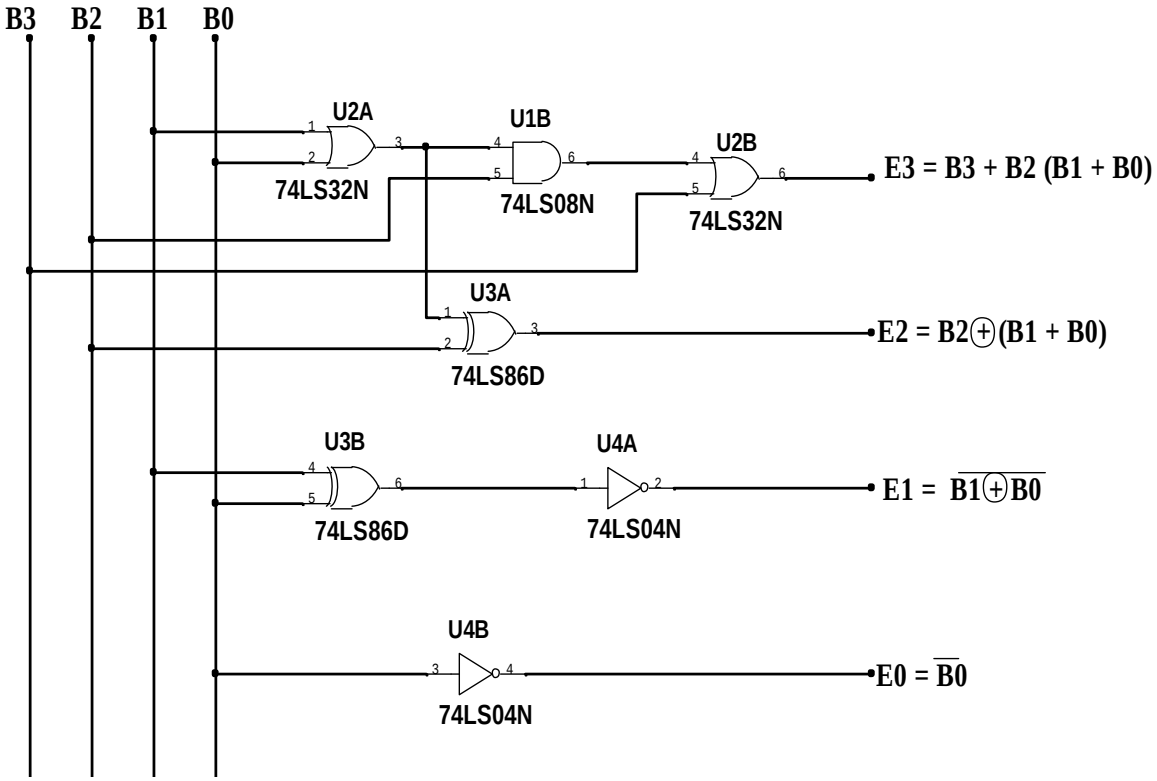
K-map for E2:

B3B2 \ B1B0	00	01	11	10
00	0	1	1	1
01	1	0	0	0
11	X	X	X	X
10	0	1	X	X

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

LOGIC DIAGRAM:

BCD TO EXCESS-3 CODE CONVERTOR

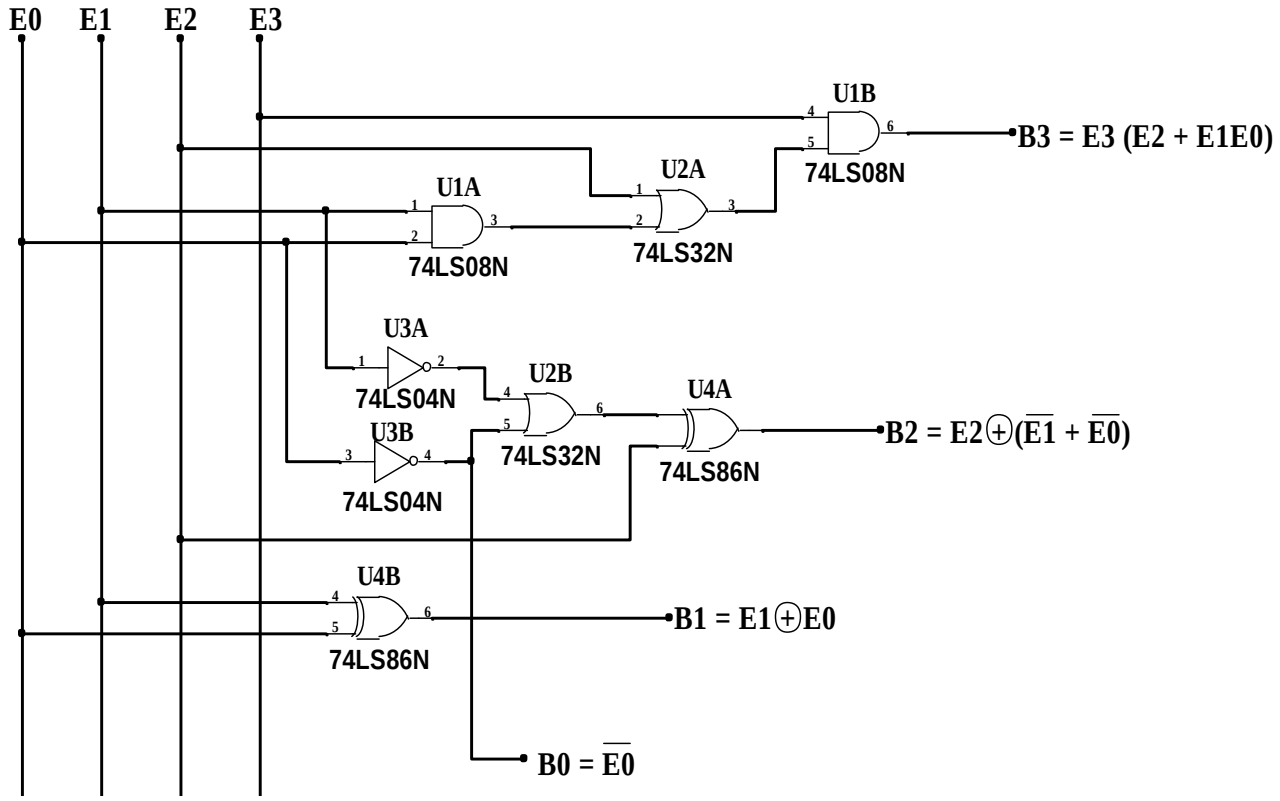


TRUTH TABLE:

Inputs (BCD CODE)				Outputs (EX-3 CODE)			
B3	**B2**	**B1**	**B0**	**E3**	**E2**	**E1**	**E0**
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

LOGIC DIAGRAM:

EXCESS-3 TO BCD CODE CONVERTOR



TRUTH TABLE:

Inputs (EX-3 CODE)				Outputs (BCD CODE)			
E3	**E2**	**E1**	**E0**	**B3**	**B2**	**B1**	**B0**
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

BCD TO EXCESS-3 CODE CONVERTOR - K – MAP:

K-map for E1:

B3B2 \ B1B0	00	01	11	10
00	1	0	1	0
01	1	0	1	0
11	X	X	X	X
10	1	0	X	X

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

K-map for E0:

B3B2 \ B1B0	00	01	11	10
00	1	0	0	1
01	1	0	0	1
11	X	X	X	X
10	1	0	X	X

$E0 = \overline{B0}$

EXCESS-3 TO BCD CODE CONVERTOR - K – MAP:

K-map for B3:

E3E2 \ E1E0	00	01	11	10
00	X	X	0	X
01	0	0	0	0
11	1	X	X	X
10	0	0	1	0

$B3 = E3 (E2 + E1E0)$

K-map for B2:

E3E2 \ E1E0	00	01	11	10
00	X	X	0	X
01	0	0	1	0
11	0	X	X	X
10	1	1	0	1

$B2 = E2 \oplus (\overline{E1} + \overline{E0})$

K-map for B1:

E3E2 \ E1E0	00	01	11	10
00	X	X	0	X
01	0	1	0	1
11	0	X	X	X
10	0	1	0	1

$B1 = E1 \oplus E0$

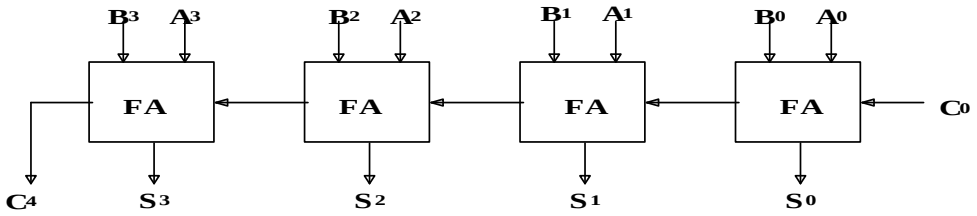
K-map for B0:

E3E2 \ E1E0	00	01	11	10
00	X	X	0	X
01	1	0	0	1
11	1	X	X	X
10	1	0	0	1

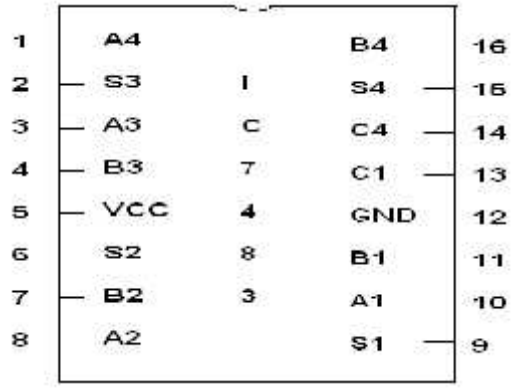
$B0 = \overline{E0}$

RESULT:

BLOCK DIAGRAM - BINARY ADDER

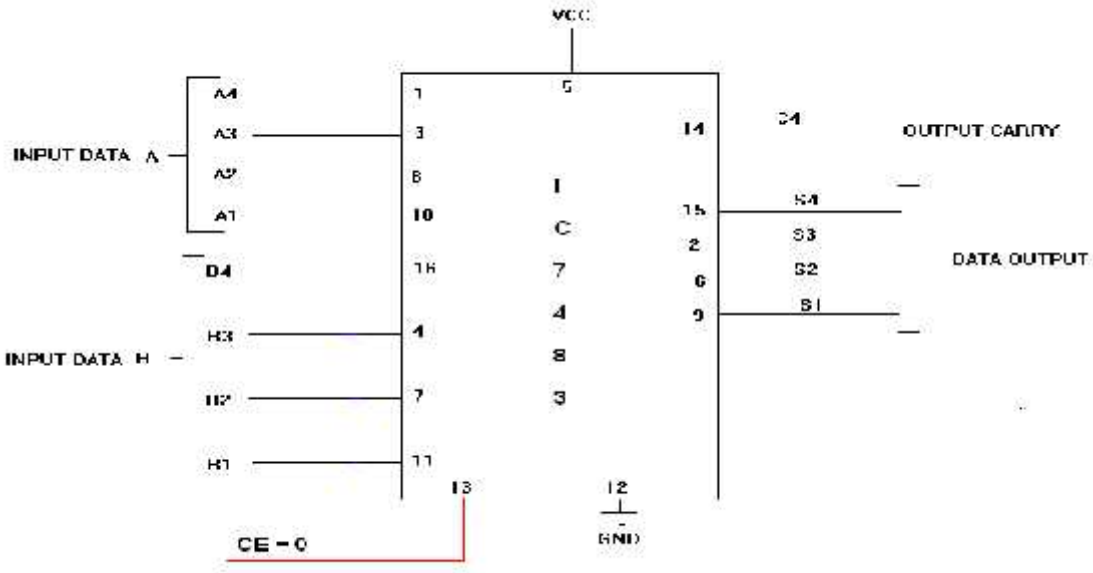


PIN DIAGRAM OF IC 7483:



LOGIC DIAGRAM:

4-BIT BINARY ADDER



Date: Expt. No.:	**DESIGN OF 4-BIT ADDER AND SUBTRACTOR**

AIM:

To design and implement 4-bit adder and subtractor using IC 7483.

APPARATUS REQUIRED:

Sl.No.	COMPONENT	SPECIFICATION	QTY.
1.	IC	IC 7483	1
2.	EX-OR GATE	IC 7486	1
3.	NOT GATE	IC 7404	1
3.	IC TRAINER KIT	-	1
4.	PATCH CORDS	-	Adequate

THEORY:

BINARY ADDER:

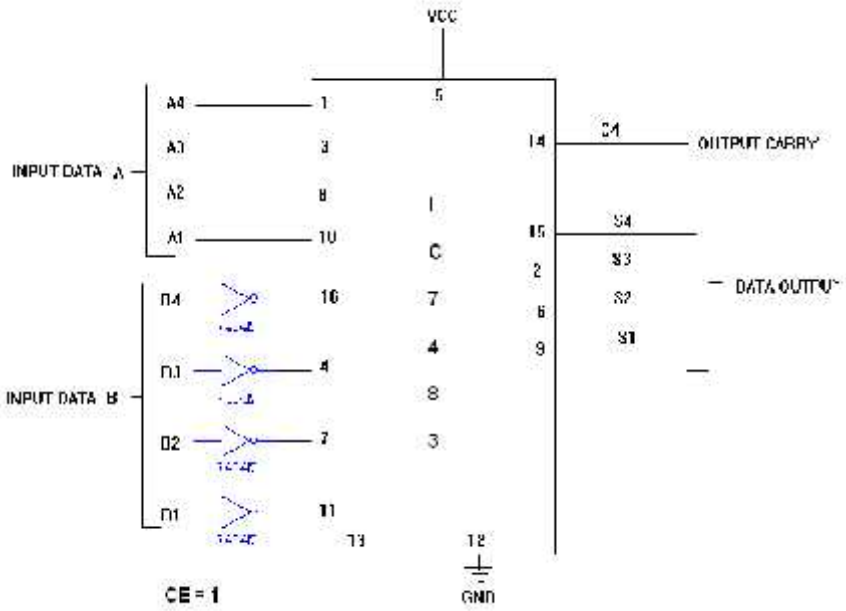
A binary adder is a digital circuit that produces the arithmetic sum of two binary numbers. It can be constructed with full adders connected in cascade, with the output carry from each full adder connected to the input carry of next full adder in the chain. The augend bits of 'A' and the addend bits of 'B' are designated by subscript numbers from right to left, with subscript '0' denoting the least significant bit. The carries are connected in chain through the full adders. The input carry to the adder is C_0 and it ripples through the full adders to the output carry C_4. The 'S' outputs generate the required sum bits.

BINARY SUBTRACTOR:

The subtraction of unsigned binary numbers can be done most conveniently by means of complements. The subtraction A – B can be done by taking 2's complement of B and adding it to A. The 2's complement can be obtained by taking 1's complement and adding 1 to the least significant pair of bits. The 1's complements can be implemented with inverters, and a 1 can be added to the sum through the input carry. The input carry C_0 must be equal to 1 when performing subtraction.

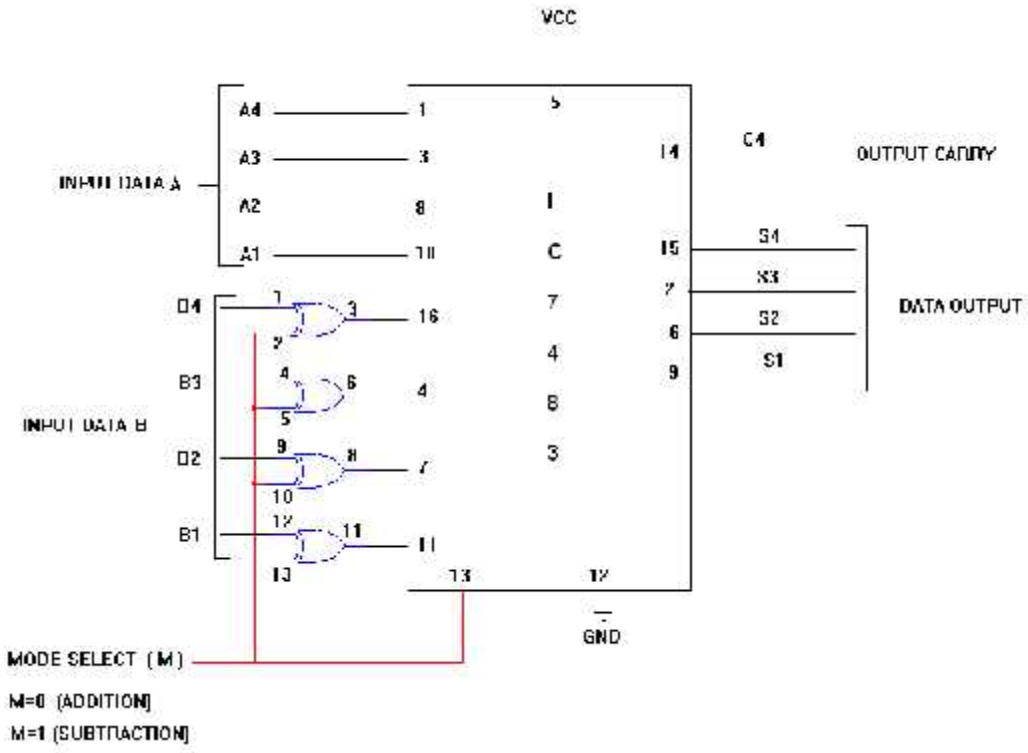
LOGIC DIAGRAM:

4-BIT BINARY SUBTRACTOR



LOGIC LOGIC DIAGRAM:

4-BIT BINARY ADDER/SUBTRACTOR



BINARY ADDER/SUBTRACTOR:

The addition and subtraction operation can be combined into one circuit with one common binary adder. This is done by including an exclusive – OR gate with each full-adder. The mode input M controls the operation of the circuit. When M=0, the circuit is an adder and when M=1, the circuit becomes a subtractor.

PROCEDURE:

1. Rig the circuit as per the circuit diagram.
2. Apply the given binary input data to the respective input pins
3. Verify the truth table.

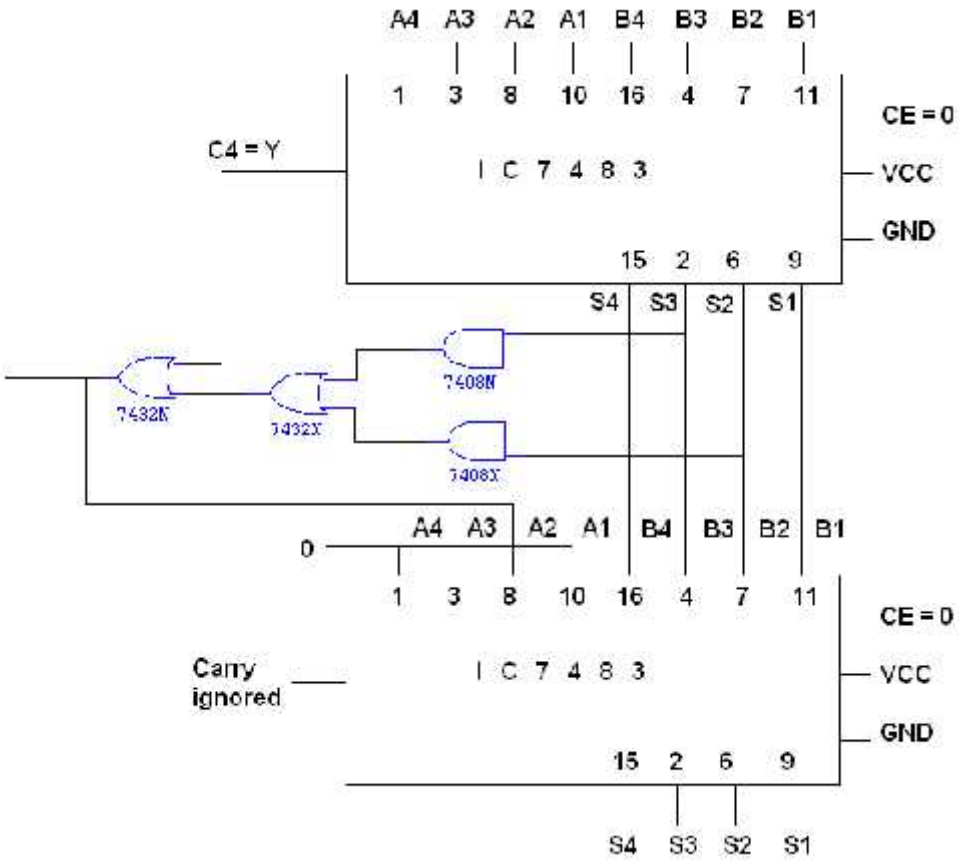
TRUTH TABLE:

Input Data A				Input Data B				Addition					Subtraction				
A4	A3	A2	A1	B4	B3	B2	B1	C	S4	S3	S2	S1	B	D4	D3	D2	D1
1	0	0	0	0	0	1	0	0	1	0	1	0	1	0	1	1	0
1	0	0	0	1	0	0	0	1	0	0	0	0	1	0	0	0	0
0	0	1	0	1	0	0	0	0	1	0	1	0	0	1	0	1	0
0	0	0	1	0	1	1	1	0	1	0	0	0	0	1	0	1	0
1	0	1	0	1	0	1	1	1	0	0	1	0	0	1	1	1	1
1	1	1	0	1	1	1	1	1	1	0	1	0	0	1	1	1	1
1	0	1	0	1	1	0	1	1	0	1	1	1	0	1	1	0	1

RESULT:

LOGIC DIAGRAM:

BCD ADDER



K-map for C:

Z8 Z4 \ Z2 Z1	00	01	11	10
00	0	0	0	0
01	0	0	0	0
11	1	1	1	1
10	0	0	1	1

Date: Expt. No.:	**DESIGN OF BCD ADDER**

AIM:

To design and implement BCD adder IC 7483.

APPARATUS REQUIRED:

Sl.No.	COMPONENT	SPECIFICATION	QTY.
1.	IC	IC 7483	2
2.	EX-OR GATE	IC 7486	1
3.	NOT GATE	IC 7408	1
3.	IC TRAINER KIT	-	1
4.	PATCH CORDS	-	Adequate

THEORY:

BCD ADDER: BCD adder is a circuit that performs the addition of two BCD numbers in parallel. BCD additions are performed in 4-bit binary form so there is a possibility of increasing binary number greater than 9 that results wrong output. To avoid this, in BCD addition correction logic I included as described below,

1. If the binary sum is equal or less than 9 with carry 0, then that binary sum is correct BCD sum.
2. I the binary sum is equal or less than 9 with carry 1, then that binary sum is an incorrect BCD sum. To get the correct BCD sum add 0110 with least significant binary sum digits.
3. If the binary number is greater than 9, then that binary sum is an incorrect BCD sum. To get the correct BCD sum add 0110 with binary sum digits.

BCD adder can be constructed with three blocks such as two binary adders and the correction logic circuit. Initially in the BCD adders, the four bit binary numbers are added using parallel binary adder and then, the binary output is checked to correct as BCD number. The correction logic generates the correction code based on the binary output values. When we get the incorrect binary output as per the condition described above, the correction code is added with the binary output to get the correct BCD number through second binary adder.

PROCEDURE:

1. Connections are made as per the circuit diagram.
2. Apply the logical Input data and verify the corresponding output.

TRUTH TABLE:

Binary sum					BCD sum					Decimal
K	Z_8	Z_4	Z_2	Z_1	C	S_8	S_4	S_2	S_1	
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	1	1
0	0	0	1	0	0	0	0	1	0	2
0	0	0	1	1	0	0	0	1	1	3
0	0	1	0	0	0	0	1	0	0	4
0	0	1	0	1	0	0	1	0	1	5
0	0	1	1	0	0	0	1	1	0	6
0	0	1	1	1	0	0	1	1	1	7
0	1	0	0	0	0	1	0	0	0	8
0	1	0	0	1	0	1	0	0	1	9
0	1	0	1	0	1	0	0	0	0	10
0	1	0	1	1	1	0	0	0	1	11
0	1	1	0	0	1	0	0	1	0	12
0	1	1	0	1	1	0	0	1	1	13
0	1	1	1	0	1	0	1	0	0	14
0	1	1	1	1	1	0	1	0	1	15
1	0	0	0	0	1	0	1	1	0	16
1	0	0	0	1	1	0	1	1	1	17
1	0	0	1	0	1	1	0	0	0	18
1	0	0	1	1	1	1	0	0	1	19

RESULT:

Date: Expt. No.:	**DESIGN AND IMPLEMENTATION OF MAGNITUDE COMPARATOR**

AIM:

To design and implement

(i) 2 – bit magnitude comparator using basic gates.

(ii) 4 – bit magnitude comparator using IC 7485.

APPARATUS REQUIRED:

Sl.No.	COMPONENT	SPECIFICATION	QTY.
1.	AND GATE	IC 7408 / IC7411	2 / 1
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 operation that determines whether 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 determines their relative magnitude. The outcome of the comparator is specified by three binary variables that indicate whether A>B, A=B (or) A<B.

$A = A_3\ A_2\ A_1\ A_0$

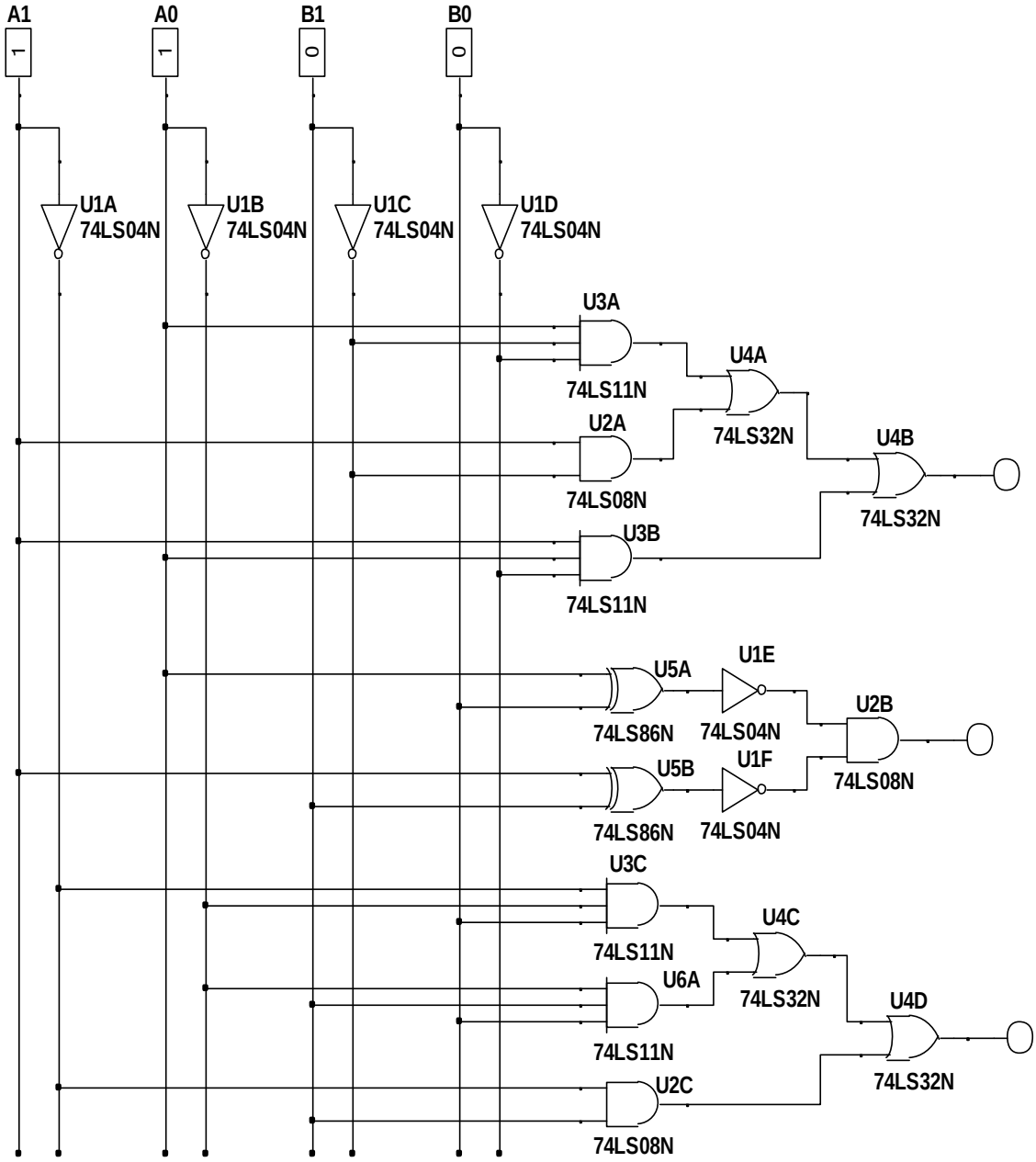
$B = B_3\ B_2\ B_1\ B_0$

The equality of the two numbers A 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.

LOGIC DIAGRAM:

2 BIT MAGNITUDE COMPARATOR



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

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

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

The same circuit can be used to compare the relative magnitude of two BCD digits.

Where, A = B is expanded as,

$A = B = (A_3 + B_3)\ (A_2 + B_2)\ (A_1 + B_1)\ \ (A_0 + B_0)$

$x_3 \quad x_2 \quad x_1 \quad x_0$

2-BIT MAGNITUDE COMPARATOR – K-MAP

K-map for A>B:

A1 A0 \ B1B0	00	01	11	10
00	0	0	0	0
01	1	0	0	0
11	1	1	0	1
10	1	1	0	0

A>B = $\overline{B1}$ A1 + $\overline{B1}$ $\overline{B0}$ A0 + $\overline{B0}$ A1 A0

K-map for A=B:

A1 A0 \ B1B0	00	01	11	10
00	1	0	0	0
01	0	1	0	0
11	0	0	1	0
10	0	0	0	1

A=B = (B0 ⊙ A0) (B1⊙ A1)

K-map for A<B:

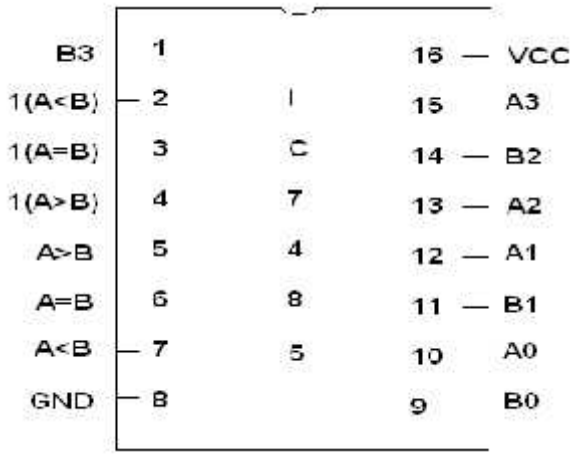
A1 A0 \ B1B0	00	01	11	10
00	0	1	1	1
01	0	0	1	1
11	0	0	0	0
10	0	0	1	0

A<B = B1 $\overline{A1}$ + $\overline{A1}$ $\overline{A0}$ B0 + B0 B1 $\overline{A0}$

TRUTH TABLE – 2 - BIT MAGNITUDE COMPARATOR

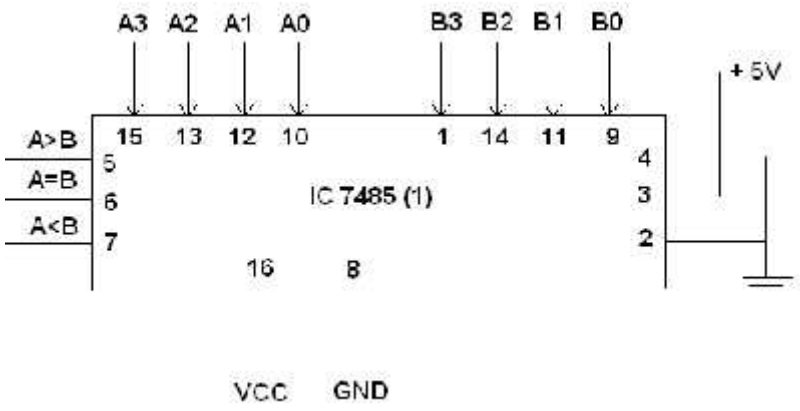
INPUTS				OUTPUTS		
A		B		A>B	A=B	A<B
A1	A0	B1	B0			
0	0	0	0	0	1	0
0	0	0	1	0	0	1
0	0	1	0	0	0	1
0	0	1	1	0	0	1
0	1	0	0	1	0	0
0	1	0	1	0	1	0
0	1	1	0	0	0	1
0	1	1	1	0	0	1
1	0	0	0	1	0	0
1	0	0	1	1	0	0
1	0	1	0	0	1	0
1	0	1	1	0	0	1
1	1	0	0	1	0	0
1	1	0	1	1	0	0
1	1	1	0	1	0	0
1	1	1	1	0	1	0

PIN DIAGRAM OF IC 7485:



LOGIC DIAGRAM:

4 BIT MAGNITUDE COMPARATOR



TRUTH TABLE:

Comparing Inputs				Cascading Inputs			OUPUTS		
A_3 B_3	A_2 B_2	A_1 B_1	A_0 B_0	$I_{A>B}$	$I_{A<B}$	$I_{A=B}$	A>B	A<B	A=B
$A_3 > B_3$	X	X	X	X	X	X	1	0	0
$A_3 < B_3$	X	X	X	X	X	X	0	1	0
$A_3 = B_3$	$A_2 > B_2$	X	X	X	X	X	1	0	0
$A_3 = B_3$	$A_2 < B_2$	X	X	X	X	X	0	1	0
$A_3 = B_3$	$A_2 = B_2$	$A_1 > B_1$	X	X	X	X	1	0	0
$A_3 = B_3$	$A_2 = B_2$	$A_1 < B_1$	X	X	X	X	0	1	0
$A_3 = B_3$	$A_2 = B_2$	$A_1 = B_1$	$A_0 > B_0$	X	X	X	1	0	0
$A_3 = B_3$	$A_2 = B_2$	$A_1 = B_1$	$A_0 < B_0$	X	X	X	0	1	0
$A_3 = B_3$	$A_2 = B_2$	$A_1 = B_1$	$A_0 = B_0$	1	0	0	1	0	0
$A_3 = B_3$	$A_2 = B_2$	$A_1 = B_1$	$A_0 = B_0$	0	1	0	0	1	0
$A_3 = B_3$	$A_2 = B_2$	$A_1 = B_1$	$A_0 = B_0$	0	0	1	0	0	1
$A_3 = B_3$	$A_2 = B_2$	$A_1 = B_1$	$A_0 = B_0$	X	X	1	0	0	1
$A_3 = B_3$	$A_2 = B_2$	$A_1 = B_1$	$A_0 = B_0$	1	1	0	0	0	0
$A_3 = B_3$	$A_2 = B_2$	$A_1 = B_1$	$A_0 = B_0$	0	0	0	1	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:

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

AIM:

To design and implement multiplexer and de-multiplexer using logic gates and study of IC 74153 and IC 74139.

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
4.	MULTIPLEXER IC	IC 71LS153	1
5.	DEMULTIPLEXER IC	IC 74LS139	1
6.	IC TRAINER KIT	-	1
7.	PATCH CORDS	-	32

THEORY:

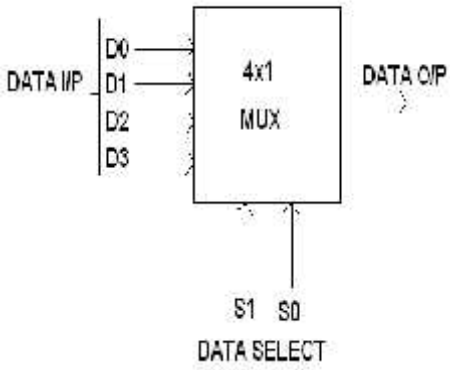
MULTIPLEXER:

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.

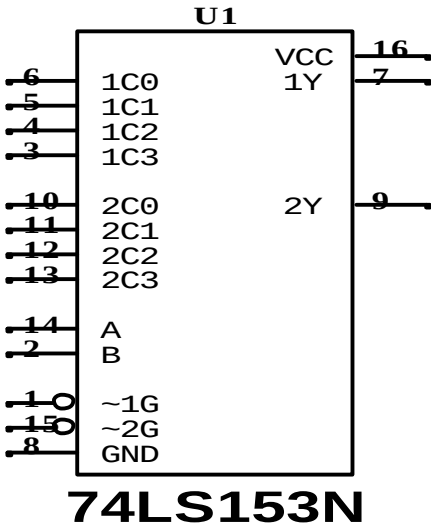
DEMULTIPLEXER:

A demultiplexer is a circuit that receives information on a single line and transmits this information on one of 2^n possible output lines. The selection of specific output line is controlled by the values of n selection lines. The single input variable D_{in} has a path to all four outputs, but the input information is directed to only one of the output lines.

BLOCK DIAGRAM OF 4:1 MULTIPLEXER:



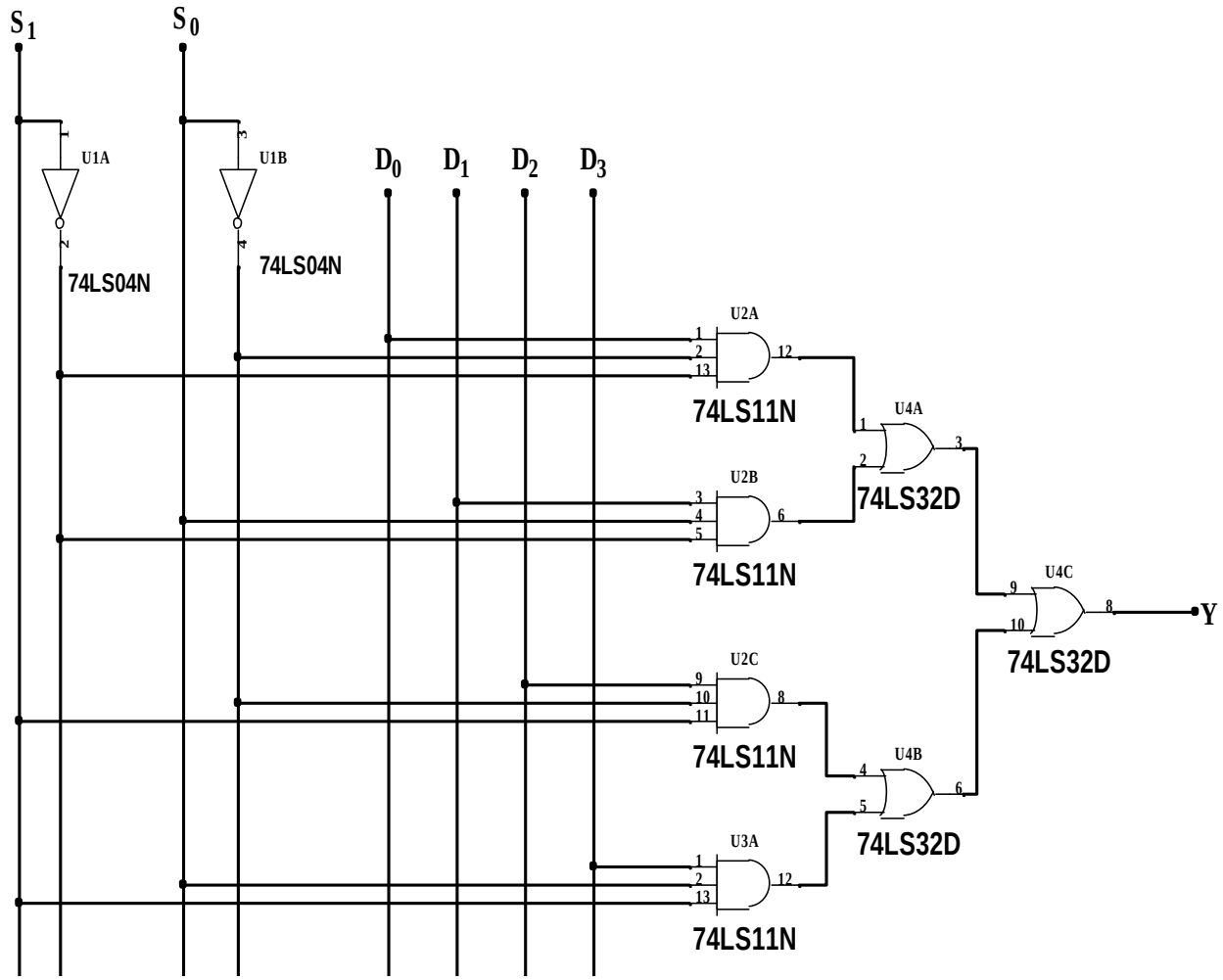
PIN DIAGRAM



FUNCTION TABLE:

Select Inputs		Inputs (1 or 2)					Output
S_0	S_1	$\overline{E}$	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
1	0	0	X	0	X	X	0
1	0	0	X	1	X	X	1
0	1	0	X	X	0	X	0
0	1	0	X	X	1	X	1
1	1	0	X	X	X	0	0
1	1	0	X	X	X	1	1

CIRCUIT DIAGRAM OF MULTIPLEXER:

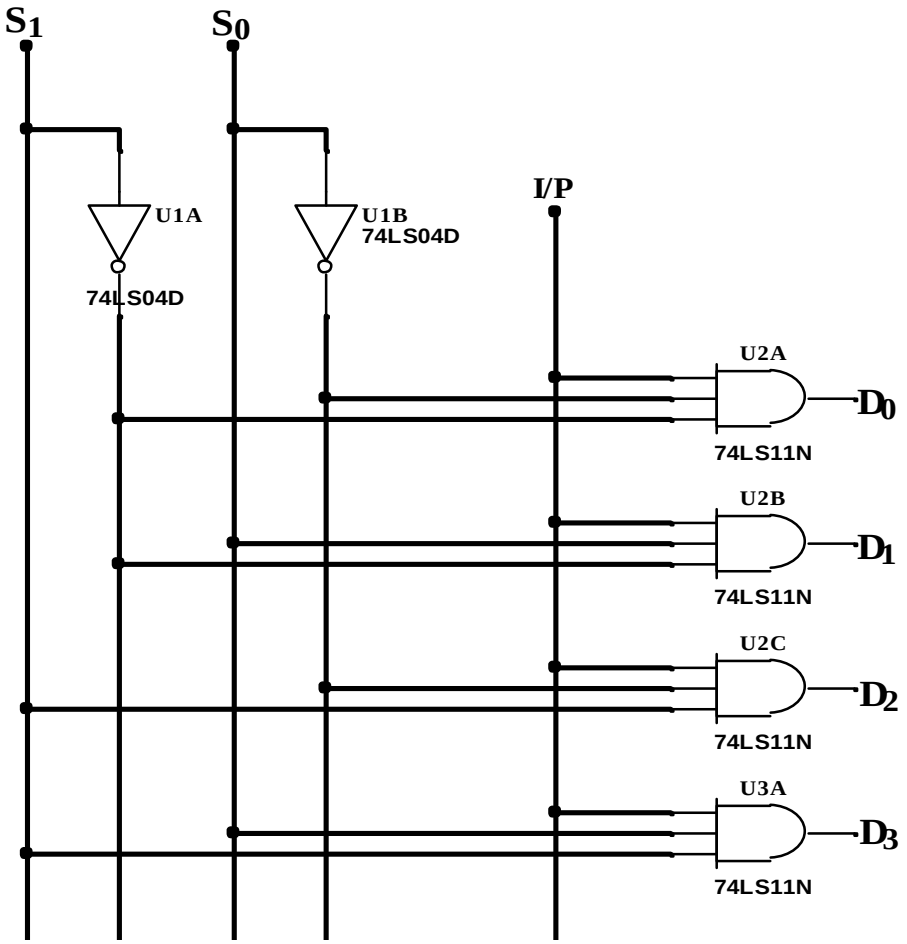


TRUTH TABLE:

Select Inputs		Output
S_0	S_1	Y
0	0	D0
0	1	D1
1	0	D2
1	1	D3

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

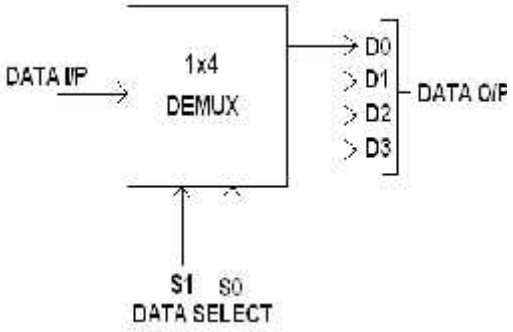
LOGIC DIAGRAM OF DEMULTIPLEXER:



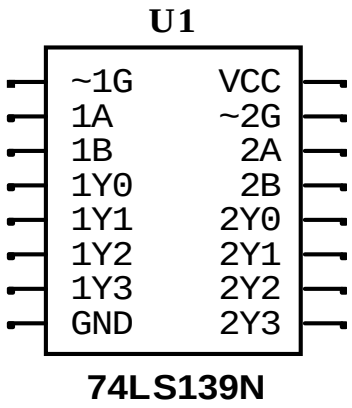
TRUTH TABLE:

Inputs			Outputs			
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

BLOCK DIAGRAM OF 1:4 DEMULTIPLEXER:



PIN DIAGRAM



FUNCTION TABLE:

Inputs			Outputs			
$\overline{E}$	A	B	Y_0	Y_1	Y_2	Y_3
1	X	X	1	1	1	1
0	0	0	0	1	1	1
0	1	0	1	0	1	1
0	0	1	1	1	0	1
0	1	1	1	1	1	0

PROCEDURE:

(i) Connections are given as per circuit diagram.

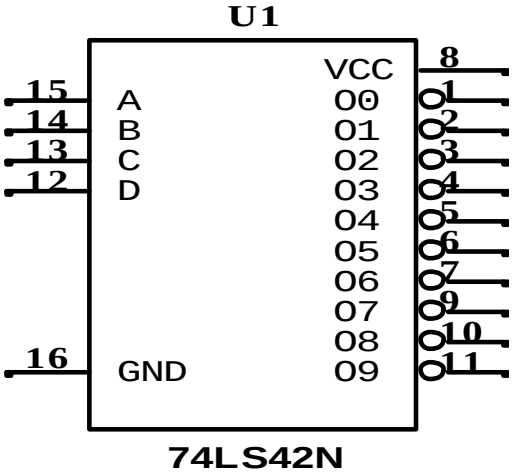
(ii) Logical inputs are given as per circuit diagram.

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

RESULT:

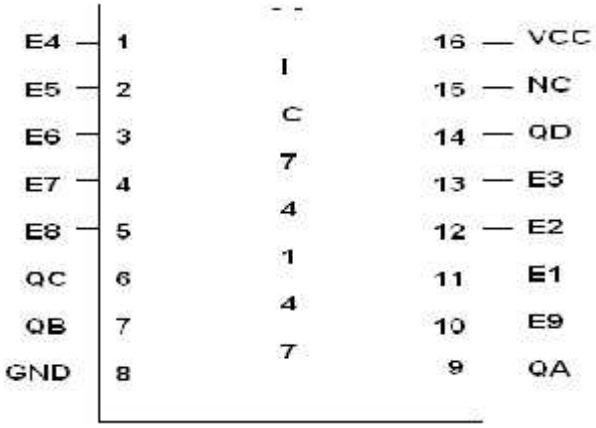
PIN DIAGRAM OF IC 7442:

BCD TO Decimal Decoder:

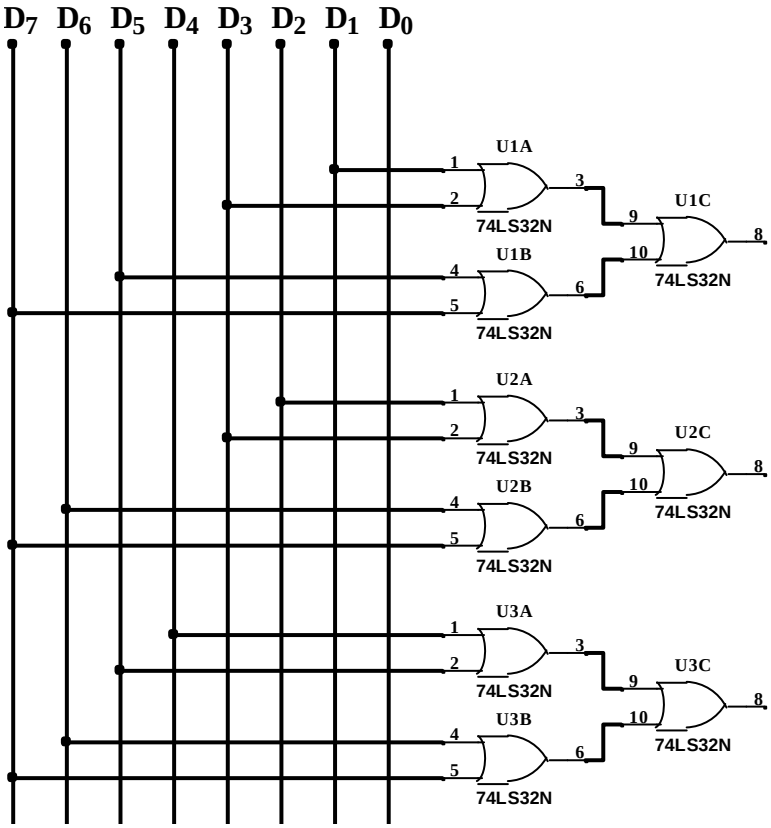


PIN DIAGRAM OF IC 74147:

10 Line to 4 Line Priority Encoder:



LOGIC DIAGRAM OF 8 – TO – 3 LINE ENCODER:



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

AIM:

To design and implement encoder and decoder using logic gates and study of IC 7442 and IC 74147.

APPARATUS REQUIRED:

Sl.No.	COMPONENT	SPECIFICATION	QTY.
1.	3 I/P NAND GATE	IC 7410	2
2.	OR GATE	IC 7432	3
3.	NOT GATE	IC 7404	
4.	DECODER	IC 7441	1
5.	ENCODER	IC 74147	1
6.	IC TRAINER KIT	-	1
7.	PATCH CORDS	-	27

THEORY:

ENCODER:

An encoder is a digital circuit that perform inverse operation of a decoder. An encoder has 2^n input lines and n output lines. In encoder the output lines generates the binary code corresponding to the input value. In octal to binary encoder it has eight inputs, one for each octal digit and three output that generate the corresponding binary code. In encoder it is assumed that only one input has a value of one at any given time otherwise the circuit is meaningless. It has an ambiguity that when all inputs are zero the outputs are zero. The zero outputs can also be generated when D0 = 1.

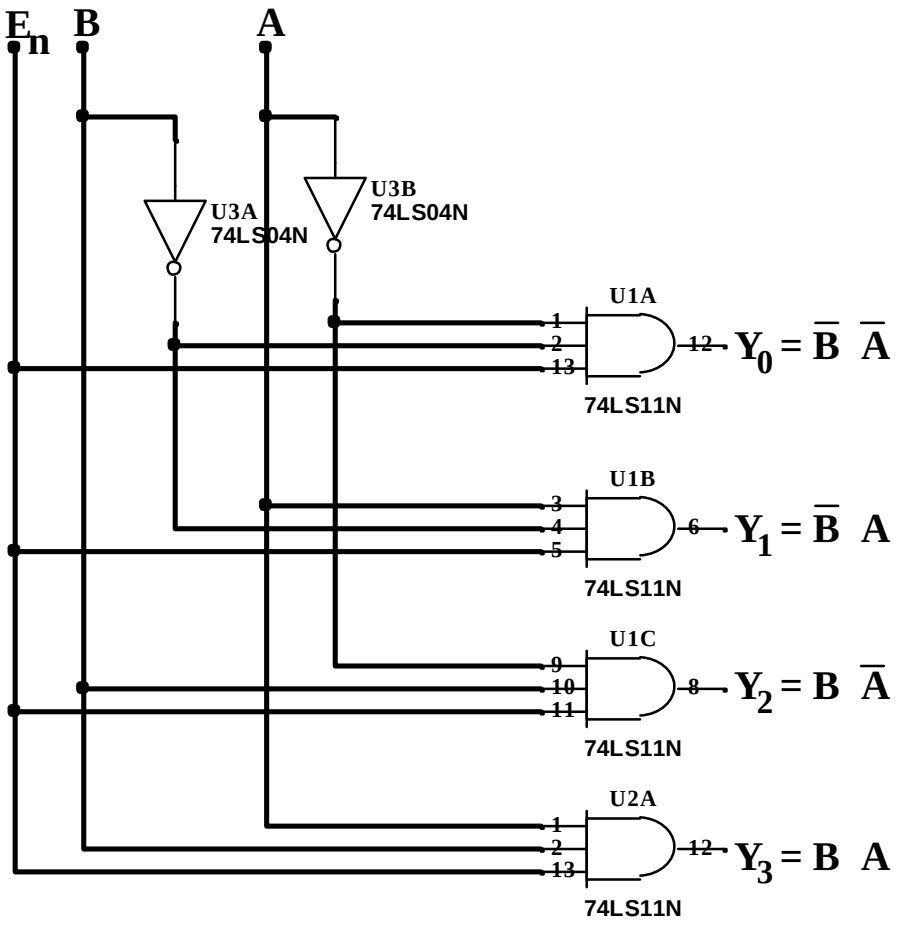
DECODER:

Decoder is a multiple input multiple output combinational digital circuit that converts n number of coded binary inputs in to 2^n number of coded binary outputs. In the decoder, the combination of input information lines define the logic output of any one output line as logic high at a time and rest of the output lines are being fixed to logic 0. When the combination of input binary information changes the logic 1 output line also be changes. Usually decoder produces unique output corresponding to each input pattern. Therefore, the n to 2^n decoder is also called as simple minterm generator with each output corresponding to exactly one minterm. The enable input used in the diagram acts as a controller of decoder. To operate the decoder the enable input must set as active high.

TRUTH TABLE – 8 – TO –3 – line Encoder:

Inputs								Outputs		
D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	C	B	A
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

LOGIC DIAGRAM OF 2-TO-4 LINE DECODER:



TRUTH TABLE OF 2-TO-4 LINE DECODER:

Enable	INPUTS		OUTPUTS			
E_n	B	A	Y_0	Y_1	Y_2	Y_3
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	0	1

PROCEDURE:

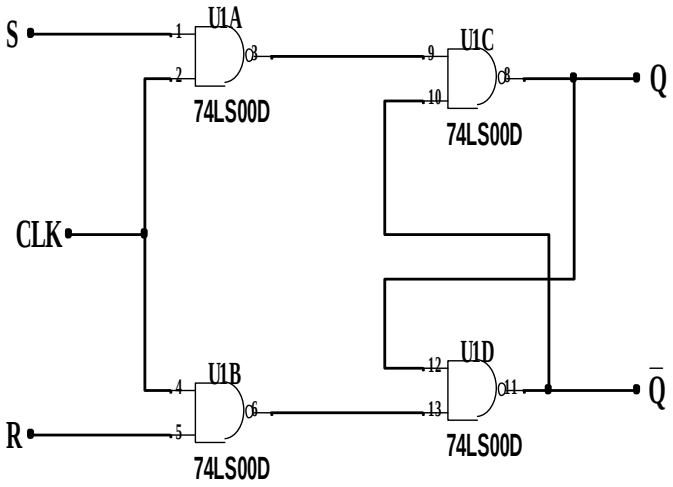
(i) Connections are given as per circuit diagram.

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

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

RESULT:

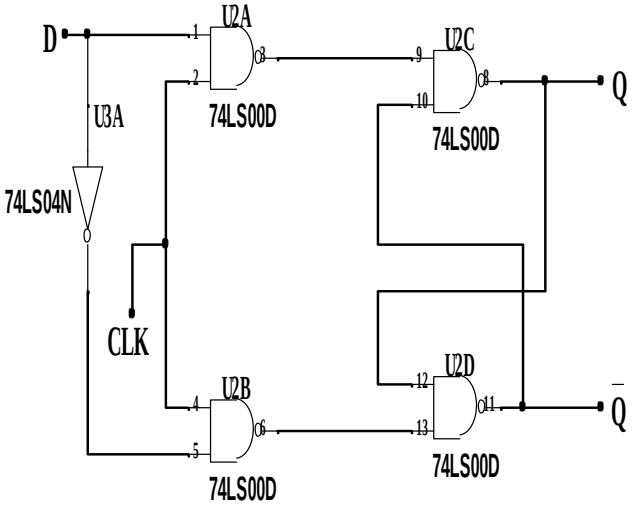
LOGIC DIAGRAM OF SR FLIP-FLOP:



TRUTH TABLE:

INPUTS			OUTPUTS	
CLK	S	R	Q	$\bar{Q}$
0	0	0	NC	NC
0	0	1	NC	NC
0	1	0	NC	NC
0	1	1	NC	NC
1	0	0	NC	NC
1	0	1	0	1
1	1	0	1	0
1	1	1	0/1	0/1

LOGIC DIAGRAM OF D FLIP-FLOP:



TRUTH TABLE:

INPUTS		OUTPUTS	
CLK	D	Q	$\bar{Q}$
0	X	NC	NC
1	0	0	1
1	1	1	0

Date: Expt. No.:	**STUDY OF FLIP-FLOPS**

AIM:

To construct and verify the truth table of a given flip-flops.

1. SR flip-flop
2. D flip-flop
3. JK flip-flop

APPARATUS REQUIRED:

Sl.No.	COMPONENT	SPECIFICATION	QTY.
	NAND GATE	IC 7400	1
	NOT GATE	IC 7404	1
	IC TRAINER KIT	-	1
	PATCH CORDS	-	30

THEORY:

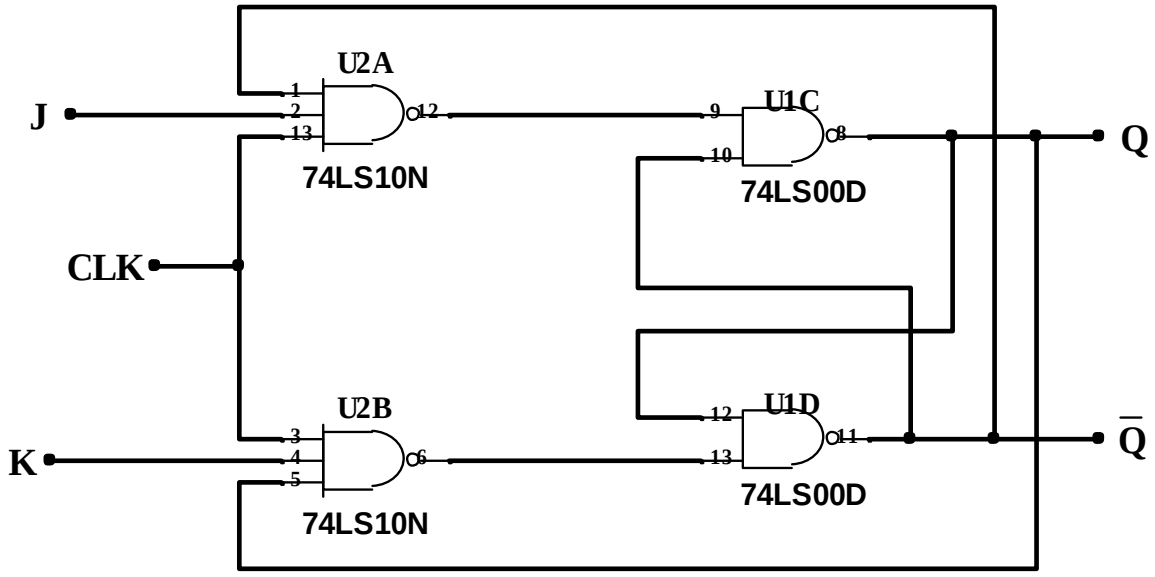
FLIP-FLOP:

Flip-Flop is an edge triggered synchronous sequential logic circuit that is capable of storing single bit binary information. A flip-flop consists of two outputs one for normally stored binary bit and another one for complement of stored binary bit. It can maintain the stored binary state indefinitely until driven by another input signal to switch other state.

SR FLIP-FLOP:

SR (Set-Reset) flip-flop is a clocked sequential circuit which is controlled by edge triggered CLK control signal. The change of clock signal from either each edge of logic 0 to logic 1 changes or each edge of logic 1 to logic 0 changes updates the output for a response of change in input signal. The circuit output is only respond to the CLK not for the change in inputs S and R.

LOGIC DIAGRAM OF JK FLIP-FLOP:



TRUTH TABLE:

INPUTS			OUTPUTS		STATES
CLK	J	K	Q	$\overline{Q}$	
0	0	0	NC	NC	No Change
0	0	1	NC	NC	No Change
0	1	0	NC	NC	No Change
0	1	1	NC	NC	No Change
1	0	0	NC	NC	No Change (Present State = Next State)
1	0	1	0	1	Reset
1	1	0	1	0	Set
1	1	1	$\overline{Q}$	Q	Toggles (complement of present state is equal to next state)

D FLIP-FLOP:

D flip-flop is used to eliminate the undesirable conditions that are present in the SR flip-flop. It define only the set and reset condition of SR flip-flop. The logic diagram of D flip-flop which satisfies the above consideration. The D flip-flop has only two inputs such as data input D and control input CLK. The input D is Connected directly to the SR flip-flop S and connected with an inverter to R. When an input D = 1, the flip-flop represent the set condition while an input D = 0, the flip-flop represent the reset condition. When the control input CLK is applied to the D flip-flop the input is applied to develop the output for the response of change in input. When the applied clock pulse CLK is equal to logic 1 the input D is stored in the flip-flop and the output Q is equal to the input D. When the applied clock pulse CLK is equal to logic 0 the input D does not applied to the flip-flop that maintains the data that stored previously on it and the output Q is also equal to previous state output.

JK FLIP-FLOP:

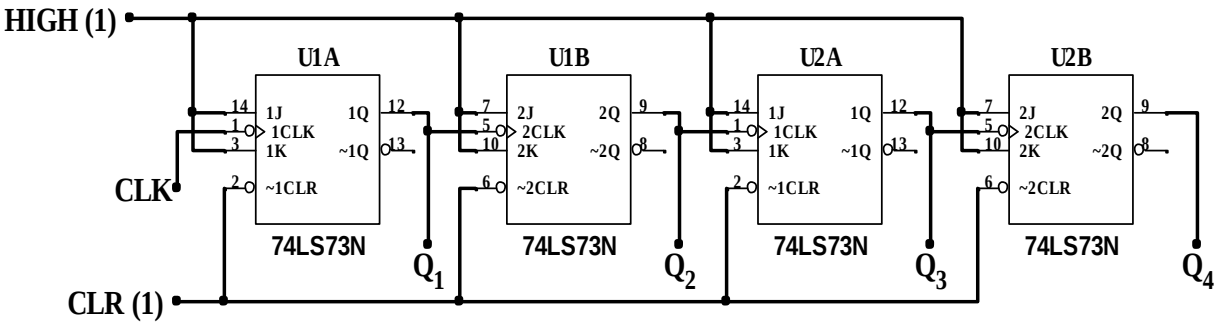
JK flip-flop is a refinement of SR flip-flop in which the indeterminate state of SR flip-flop is defined. The inputs of JK flip-flop J and K are behave like inputs of SR flip-flop S (set) and R (reset) respectively. The input J is used to set the flip-flop, while K is used to reset the flip-flop. When both inputs are high the output complements the previous output that obtained from flip-flop. The previous output is called as present state input while the present output is called as next state of flip-flop.

PROCEDURE:

1. Connections are given as per the logic diagram.
2. Apply the input data and verify the truth table.

RESULT:

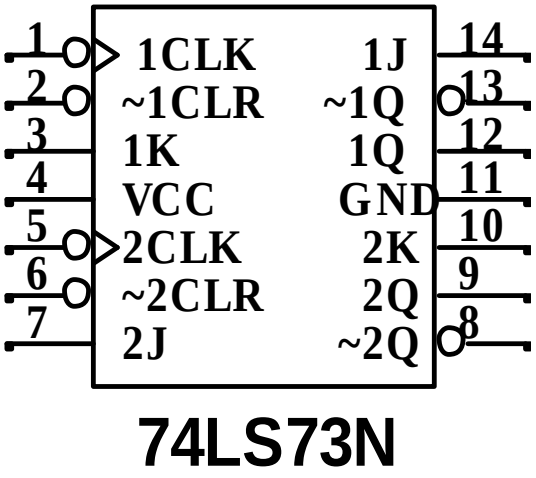
LOGIC DIAGRAM OF 4-BIT ASCHNCRONOUS (RIPPLE) UP-COUNTER:



TRUTH TABLE:

INPUT	OUTPUTS			
CLK	Q_4	Q_3	Q_2	Q_1
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

PIN DIAGRAM OF IC 7473



Date: Expt. No.:	VERIFICATION OF 4 BIT ASYNCHRONOUS (RIPPLE) COUNTER

AIM:

To construct and verify 4 bit ripple counter.

APPARATUS REQUIRED:

Sl. No.	COMPONENT	SPECIFICATION	QTY.
1.	JK FLIP FLOP	IC 7473	2
2.	IC TRAINER KIT	-	1
3.	PATCH CORDS	-	30

THEORY:

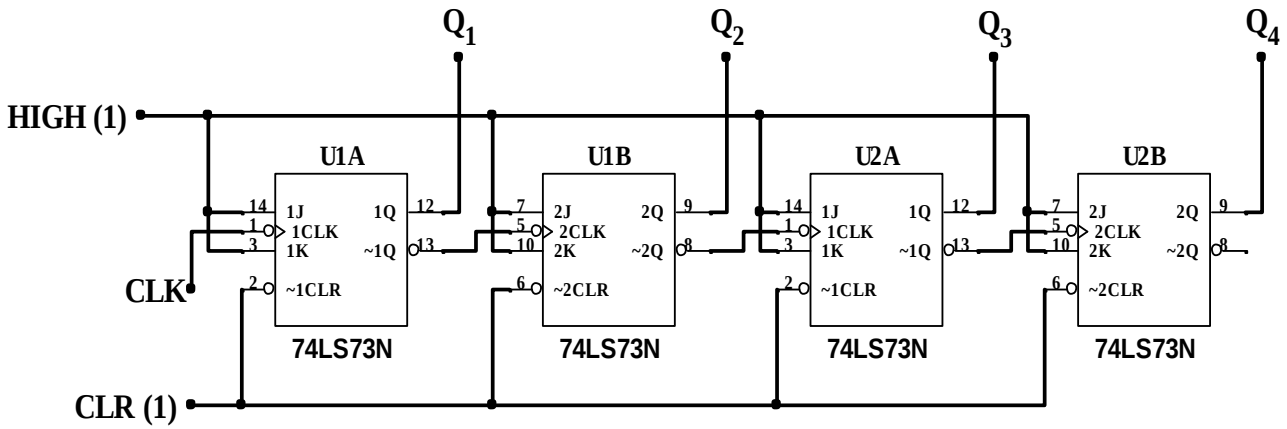
UP-COUNTER:

Asynchronous counters are the circuit that used to count the binary numbers in prescribed sequence. In asynchronous counter the flip-flops are not triggered with common clock pulse. Except the first (least significant) flip-flop others are triggered by the output of previous flip-flop while the first one is triggered by the clock pulse. Hence asynchronous counter is also called as ripple counter. When inputs set into logic high the JK flip-flops are continuously present in the toggle condition which complements the output continuously. This cause to prevent the circuit from triggering of two adjacent flip-flops simultaneously.

DOWN-COUNTER:

Asynchronous down counter performs the reverse operation of up-counter which counts the binary number by decreasing one when the flip-flops are activated by the clock pulse. First flip-flop triggered by clock pulse the remaining flip-flips are triggered by the inverted output of previous flip-flop. It is an only difference from up-counter.

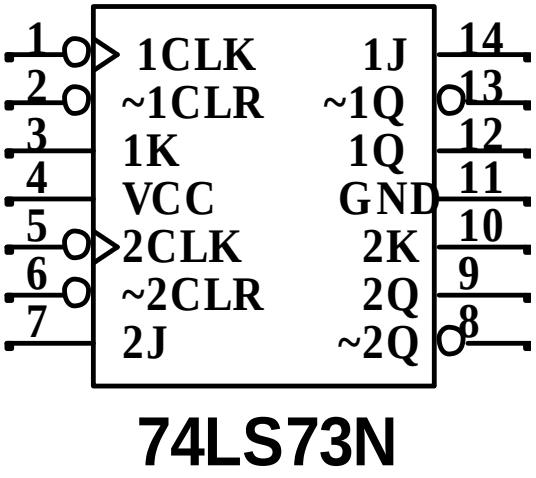
LOGIC DIAGRAM OF 4-BIT ASCHNCRONOUS (RIPPLE) DOWN-COUNTER:



TRUTH TABLE:

INPUT	OUTPUTS			
CLK	Q_4	Q_3	Q_2	Q_1
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

PIN DIAGRAM OF IC 7473

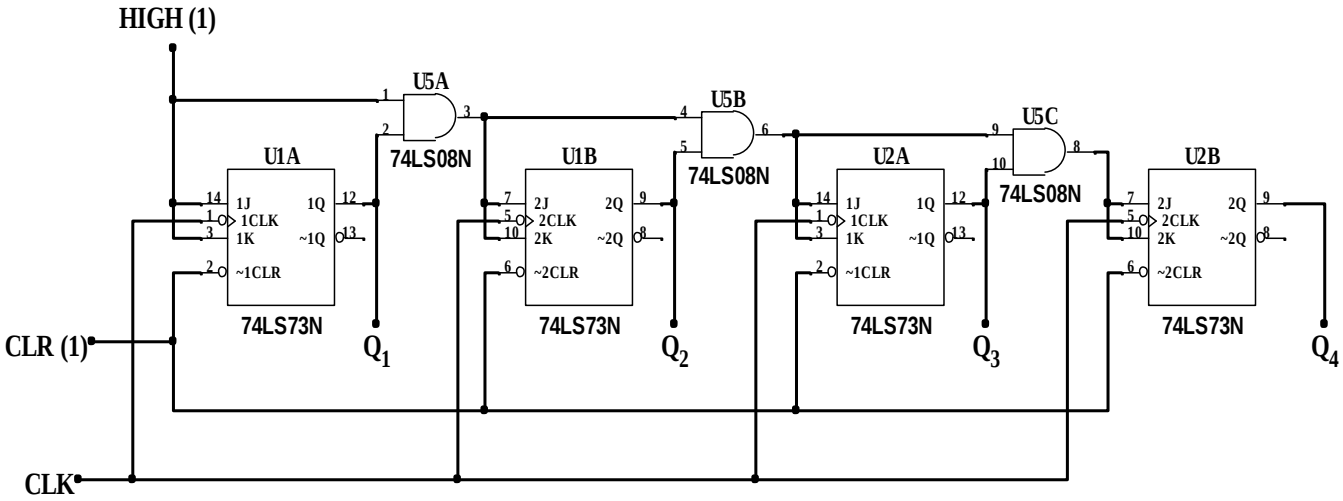


PROCEDURE:

1. Connections are given as per the logic diagram.
2. Apply the clock pulse and verify the truth table.

RESULT:

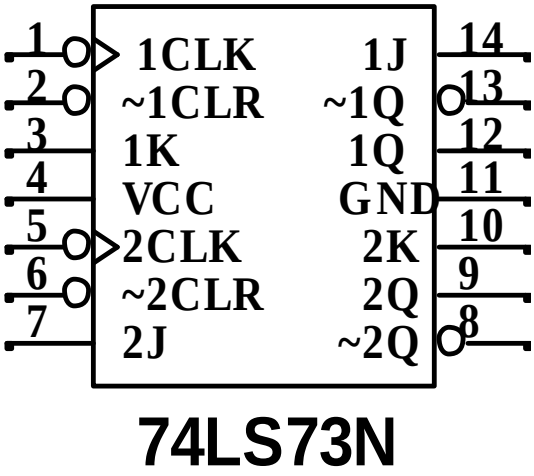
LOGIC DIAGRAM OF 4-BIT SCHNCRONOUS UP-COUNTER:



TRUTH TABLE:

INPUT	OUTPUTS			
CLK	Q_4	Q_3	Q_2	Q_1
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

PIN DIAGRAM OF IC 7473



Date: Expt. No.:	VERIFICATION OF 4 BIT SYNCHRONOUS UP-COUNTER

AIM:

To construct and verify 4 bit Synchronous counter.

APPARATUS REQUIRED:

Sl.No.	COMPONENT	SPECIFICATION	QTY.
1.	JK FLIP FLOP	IC 7473	2
2.	IC TRAINER KIT	-	1
3.	PATCH CORDS	-	30

THEORY:

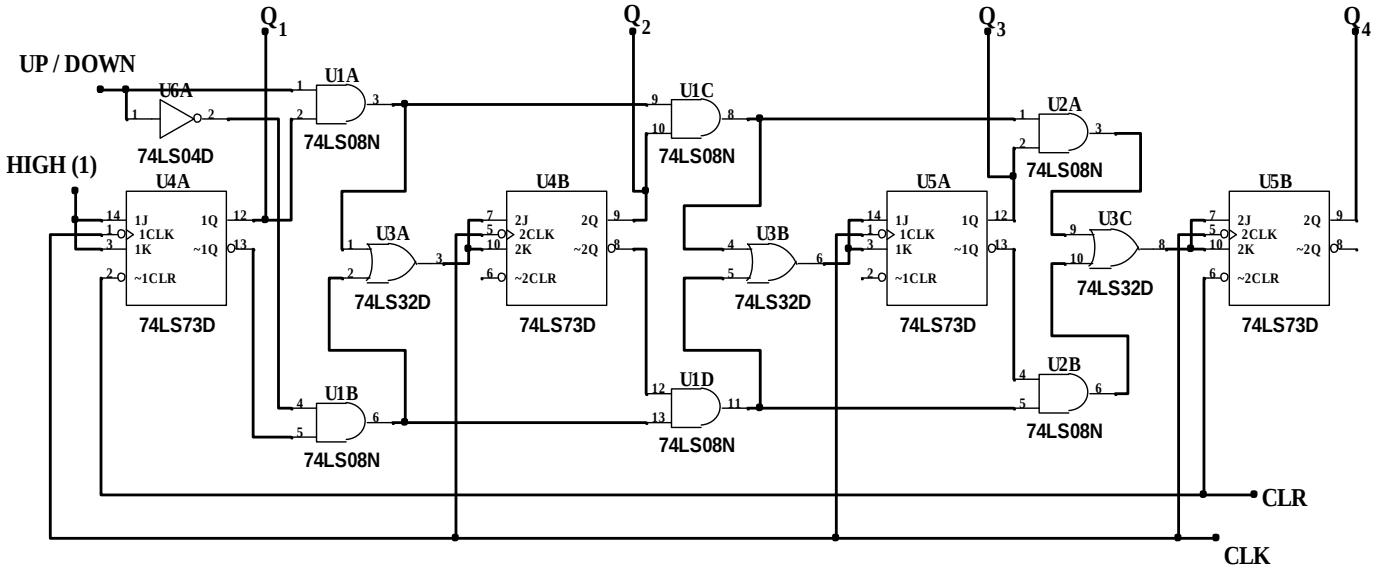
UP COUNTER: Synchronous counter is a circuit in that all flip-flops are triggered by the common clock pulse. The common clock pulse is applied to all the flip-flops simultaneously to update the output for the change of input by triggering. In the logic diagram of 4-bit synchronous up counter four flip-flops are connected in series with common clock pulse. The logic high input is applied to LSB flip-flop (1). The AND output of first stage flip-flop Q_1 and input is used to drive the J and K inputs os flip- flop (2). The J and K inputs of flip-flop (3) is driven by the AND output of flip-flops (1) and (2). Similarly, J and K inputs of flip-flop (4) is driven by the AND output of (2) and (3). The Jk flip-flop employed in the synchronous counter operates only in two conditions such as no change condition and toggle condition.

PROCEDURE:

1. Connections are given as per the logic diagram.
2. Apply the clock pulse and verify the truth table.

RESULT:

LOGIC DIAGRAM OF 4-BIT SCHNCRONOUS UP / DOWN COUNTER:



TRUTH TABLE:

Control Inputs		OUTPUTS			
CLK	UP/DOWN	Q_4	Q_3	Q_2	Q_1
0	Initial state	0	0	0	0
1	0	1	1	1	1
2	0	1	1	1	0
3	0	1	1	0	1
4	0	1	1	0	0
5	0	1	0	1	1
6	0	1	0	1	0
7	0	1	0	0	1
8	0	1	0	0	0
9	0	0	1	1	1
10	0	0	1	1	0
11	0	0	1	0	1
12	0	0	1	0	0
13	0	0	0	1	1
14	0	0	0	1	0
15	0	0	0	0	1
16	0	0	0	0	0

Control Inputs		OUTPUTS			
CLK	UP/DOWN	Q_4	Q_3	Q_2	Q_1
17	1	0	0	0	1
18	1	0	0	1	0
19	1	0	0	1	1
20	1	0	1	0	0
21	1	0	1	0	1
22	1	0	1	1	0
23	1	0	1	1	1
24	1	1	0	0	0
25	1	1	0	0	1
26	1	1	0	1	0
27	1	1	0	1	1
28	1	1	1	0	0
29	1	1	1	0	1
30	1	1	1	1	0
31	1	1	1	1	1
32	1	0	0	0	0

Date: Expt. No.:	**VERIFICATION OF 4 BIT SYNCHRONOUS UP/DOWN COUNTER**

AIM:

To construct and verify 3 bit Synchronous UP / DOWN counter.

APPARATUS REQUIRED:

Sl.No.	COMPONENT	SPECIFICATION	QTY.
1.	JK FLIP FLOP	IC 7473	2
2.	IC TRAINER KIT	-	1
3.	PATCH CORDS	-	30

THEORY:

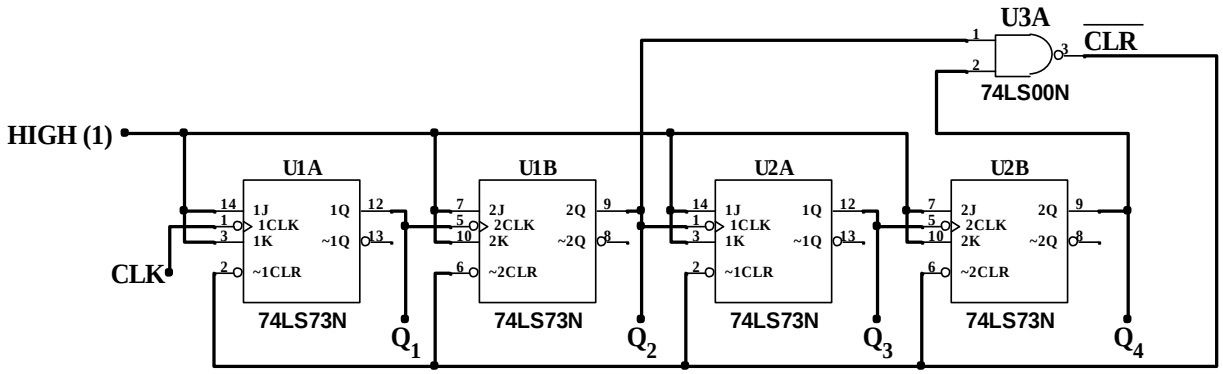
SYNCHRONOUS UP/DOWN COUNTER: Up/Down counter is a circuit which perform the logic count either up or down by increasing or decreasing a number by 1. Synchronous Up/Down counter is a circuit which executes the counting operation either up or down with a commonly clocked flip-flops. In a counter the progress of Up and Down counting operation s are selected by the control signal. After selecting the counting operation by enforcing the clock pulses to the flip-flops desired counting operation is executed. If the control signal is logic low (0) then the counter counts in the decreasing order that is down counter. When the control signal is logic high (1) then the counter counts in the increasing order that is up counter.

PROCEDURE:

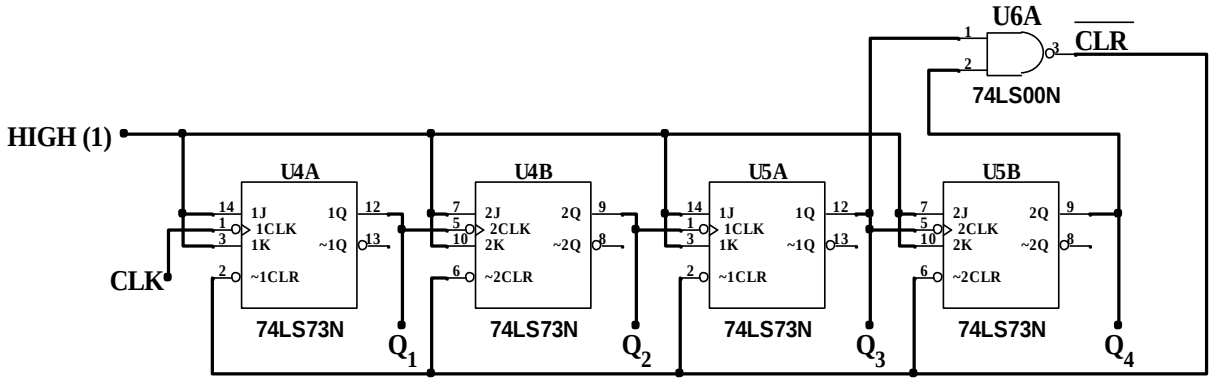
1. Connections are given as per the logic diagram.
2. Apply the clock pulse and verify the truth table.

RESULT:

LOGIC DIAGRAM FOR MOD - 10 RIPPLE COUNTER:



LOGIC DIAGRAM FOR MOD - 12 RIPPLE COUNTER:



TRUTH TABLE of MOD-10:

Input	OUTPUTS			
CLK	Q_4	Q_3	Q_2	Q_1
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	0	0	0	0

TRUTH TABLE of MOD-12:

Input	OUTPUTS			
CLK	Q_4	Q_3	Q_2	Q_1
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	0	0	0	0

Date: Expt. No.:	**VERIFICATION OF MODULO – N COUNTER**

AIM:

To construct and verify the circuit of MOD 10 and MOD 12 ripple counter.

APPARATUS REQUIRED:

Sl.No.	COMPONENT	SPECIFICATION	QTY.
4.	JK FLIP FLOP	IC 7473	2
5.	NAND GATE	IC7400	1
6.	IC TRAINER KIT	-	1
7.	PATCH CORDS	-	30

THEORY:

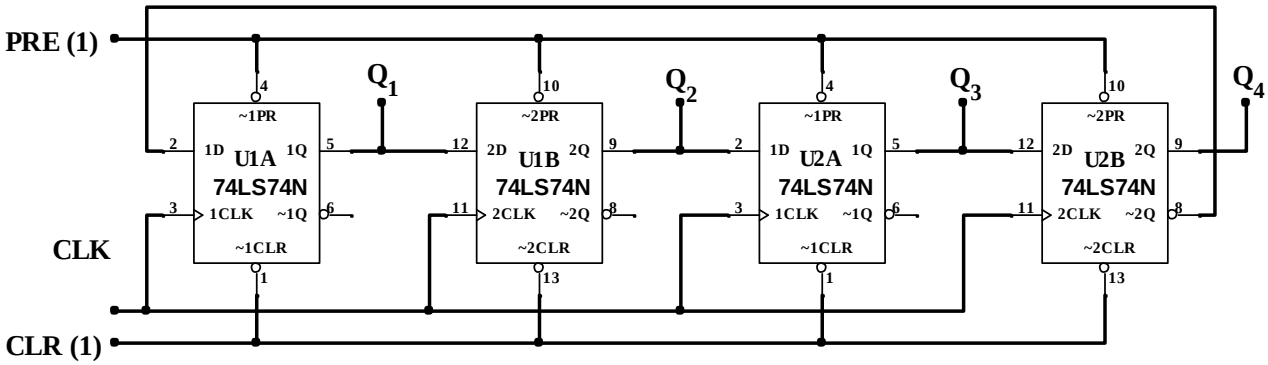
MODULO - N COUNTER: Modulo – N (mod – N) counter is a type of counter that counts the N count sequence repeatedly by the control of clock pulse. The logic circuit of mod-10 and 12 in its basic form is a BCD counter. However the presence of NAND gate will alter this sequence as follows: the NAND gate output is connected to the asynchronous RESET inputs of each flip-flops. As long as the NAND gate output is high, it will have no effect on the counter. When it goes low, it will reset all the flip-flops so that counter immediately goes to the 0000 state.

PROCEDURE:

1. Connections are given as per the logic diagram.
2. Apply the clock pulse and verify the truth table.

RESULT:

LOGIC DIAGRAM FOR JOHNSON COUNTER:



TRUTH TABLE OF JOHNSN COUNTER

Input	OUTPUTS			
CLK	Q_1	Q_2	Q_3	Q_4
0	0	0	0	0
1	1	0	0	0
2	1	1	0	0
3	1	1	1	0
4	1	1	1	1
5	0	1	1	1
6	0	0	1	1
7	0	0	0	1

Date: Expt. No.:	VERIFICATION OF JOHNSON COUNTER

AIM:

To construct and verify the circuit of MOD 10 and MOD 12 ripple counter.

APPARATUS REQUIRED:

Sl.No.	COMPONENT	SPECIFICATION	QTY.
1.	JK FLIP FLOP	IC 7474	2
2.	IC TRAINER KIT	-	1
3.	PATCH CORDS	-	30

THEORY:

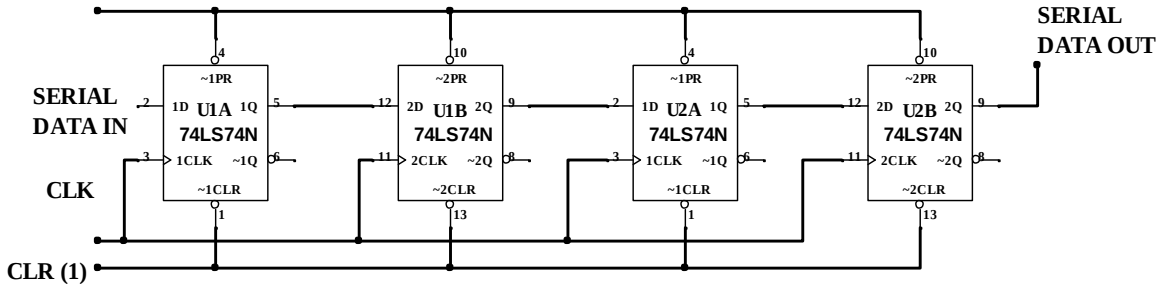
JOHNSON COUNTER: Johnson counter is a modified ring counter that the inverted output of last flip-flop is applied input for first flip-flop. Hence it is also called as twisted ring counter or switch tail ring counter. The diagram is constructed with series connected common clocked four negative edge triggered flip-flops. Commonly clocked negative edge triggered flip-flop shifts the binary data by one bit when the clock pulse is switched from logic 1 to logic 0. The count sequence of Johnson counter is obtained by shifting the numbers in circle for the triggering of each clock pulse.

PROCEDURE:

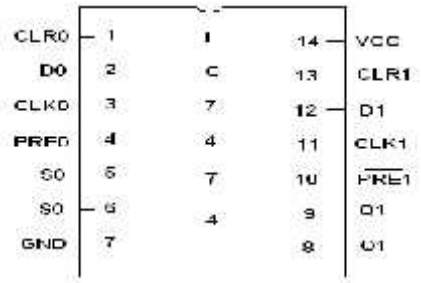
1. Connections are given as per the logic diagram.
2. Apply the clock pulse and verify the truth table.

RESULT:

LOGIC DIAGRAM: SERIAL-IN-SERIAL-OUT SHIFT REGISTER



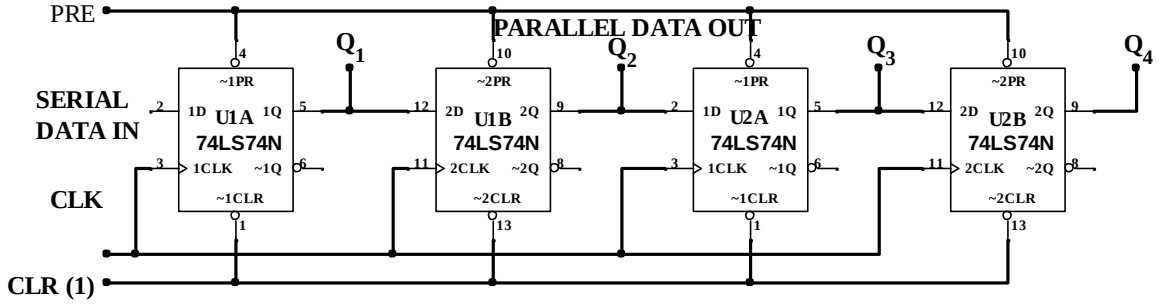
PIN DIAGRAM:



TRUTH TABLE OF SERIAL-IN-SERIAL-OUT SHIFT REGISTER

Clock Pulse	Serial Input	Binary data stored in Register				Serial Output
	Data In	Q_1	Q_2	Q_3	Q_4	Data out = Q_4
Initial Vaue	1 1 1 1	0	0	0	0	0
After 1st Clk	1 1 1	1	0	0	0	0
After 2nd Clk	11	1	1	0	0	0
After 3rd Clk	1	1	1	1	0	0
After 4th Clk	-	1	1	1	1	1

LOGIC DIAGRAM: SERIAL-IN-PARALLEL-OUT SHIFT REGISTER



Date: Expt. No.:	DESIGN AND IMPLEMENTATION OF SHIFT REGISTER

AIM:

To design and implement

(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:

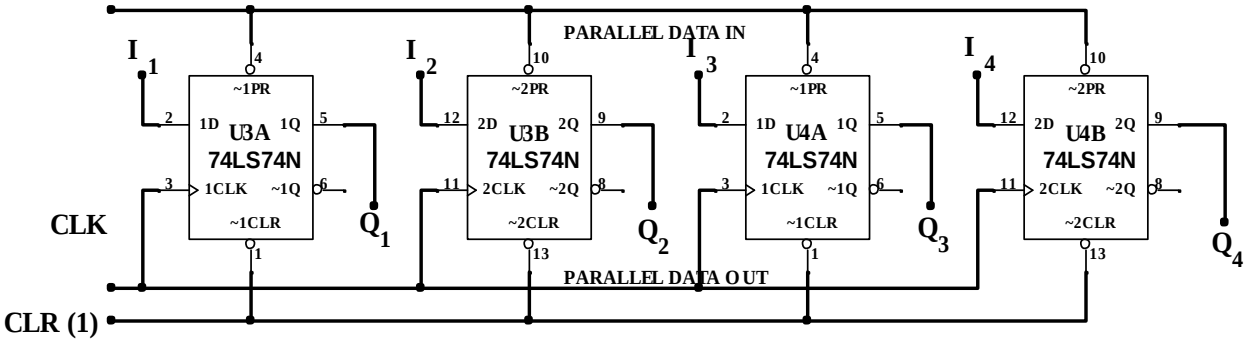
SHIFT REGISTER: Shift register is a group of flip- flops that has the capability of storing and shifting the binary information. In digital system the binary datum are required to shift in the register from one position to next position. Shift register performs the logic operation shifting of binary data from one flip-flop to next flip-flop. In the shift register, the shifting operation is controlled by common clock pulse. All the flip-flops employed with shift register receives the clock pulse that helps to shift the data from one position to next.

SERIAL-IN-SERIAL-OUT SHIFT REGISTER: This Shift register is constructed in the way of connecting the output of one flip-flop to the input of next flip –flop. All the flip-flops are connected with a common clock pulse. The binary inputs are applied in the input terminal of first flip-flop in series and the outputs are obtained from the output terminal of last flip-flop in series.

TRUTH TABLE OF SERIAL-IN-PARALLEL-OUT SHIFT REGISTER

Clock Pulse	Serial Input	Parallel Output			
	Data In	Q_1	Q_2	Q_3	Q_4
Initial Vaue	1011	0	0	0	0
After 1st Clk	1 1 1	1	0	0	0
After 2nd Clk	11	1	1	0	0
After 3rd Clk	1	0	1	1	0
After 4th Clk	-	1	0	1	1

LOGIC DIAGRAM: PARALLEL-IN-PARALLEL-OUT SHIFT REGISTER



TRUTH TABLE OF PARALLEL-IN-PARALLEL-OUT SHIFT REGISTER

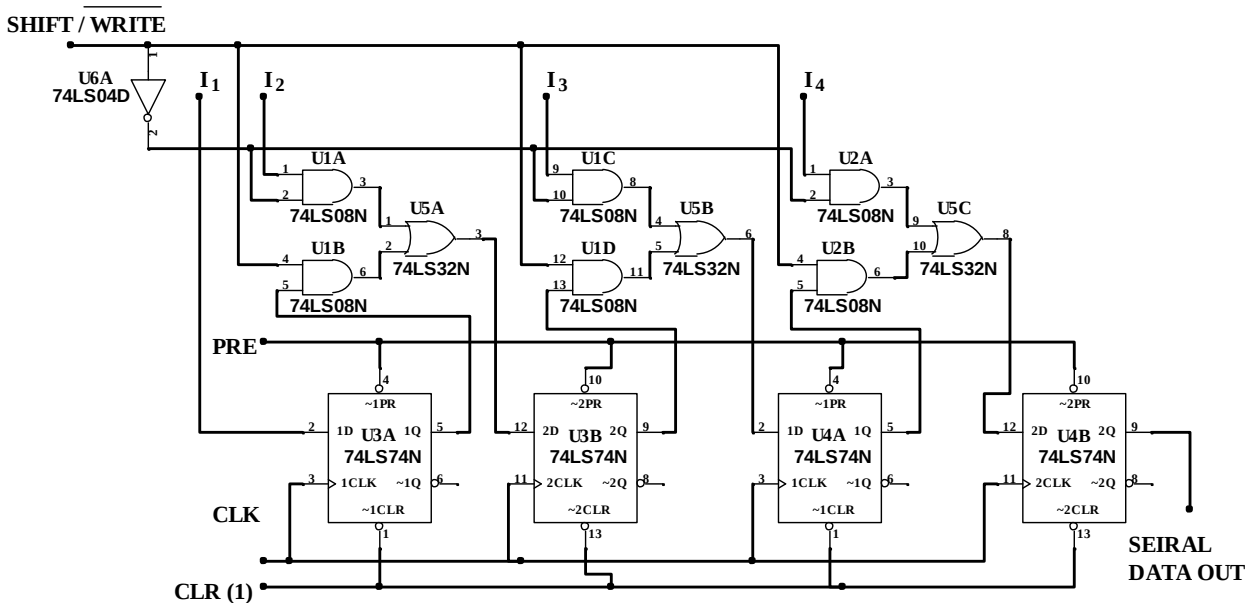
Clock Pulse	Parallel Input				Parallel Output			
	I_1	I_2	I_3	I_4	Q_1	Q_2	Q_3	Q_4
Initial Value	0	0	0	0	0	0	0	0
After 1st Clk	1	0	0	0	1	0	0	0
After 2nd Clk	1	1	0	0	1	1	0	0
After 3rd Clk	0	1	1	0	0	1	1	0
After 4th Clk	1	0	1	1	1	0	1	1

SERIAL-IN-SERIAL-OUT SHIFT REGISTER: This register accepts the binary data in series to provide the output in parallel. In logic diagram the flip-flops are connected similar to SISO shift register that logic diagram is constructed in the way of connecting the output of one flip-flop to the input of next flip-flop. The binary inputs are applied in the input terminal of first flip-flop in series and the outputs are obtained in parallel from the output terminal of each flip-flop employed with register. Hence this configuration is called serial-in-serial-out shift register.

PARALLEL-IN-PARALLEL-OUT SHIFT REGISTER: This register accepts the binary input in parallel to provide the output in parallel. All the flip-flops present in the register are triggered with common clock pulse. The binary inputs are applied in parallel to all the flip-flops and the outputs are obtained in parallel from the output terminal of each flip-flop employed with register. Hence this configuration is called parallel-in-parallel-out shift register.

PARALLEL-IN-SERIAL-OUT SHIFT REGISTER: This register accepts the binary input in parallel to provide the output in series form right most flip-flop. The logic diagram is constructed by flip-flop and combination of logic gates. The combinational logic gates are functioning as a control circuit to load the input of shift the stored binary information. A control signal ‘shift’ is an active high signal i.e., when the control signal line is activated with logic high input a shift register performs the shifting operation. A control signal $\overline{\text{'Write'}}$ is an active low signal i.e. when the control signal line is activated with logic low input a shift register loads the newly appeared input signals into the flip-flops in parallel. These logic operations are controlled by the combinational circuits.

LOGIC DIAGRAM: PARALLEL-IN-SERIAL-OUT SHIFT REGISTER



TRUTH TABLE OF PARALLEL-IN-PARALLEL-OUT SHIFT REGISTER

Control signals		Parallel Input				Serial Output
Clock Pulse	SHIFT / $\overline{\text{WRITE}}$	I_1	I_2	I_3	I_4	Q_4
0	X	0	0	0	0	No Change (Initial state)
1	0	0	0	0	1	1
1	0	1	1	0	0	0
1	1	1	1	0		0
1	1	1	1			0
1	1	1				1
1	1	-				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:

Date: Expt. No.:	**STUDY OF ICs - REGISTER, DECADE COUNTER AND RIPPLE COUNTER**

AIM:

To study the given ICs (IC 7495, IC7490 and IC7493)

APPARATUS REQUIRED:

SL.NO.	COMPONENT	SPECIFICATION	QTY.
1.	REGISTER IC	IC 7495	1
2.	DECADE COUNTER IC	IC 7490	1
3.	RIPPLE COUNTER IC	IC 7493	1
4.	IC TRAINER KIT	-	1
5.	PATCH CORDS	-	35

THEORY:

IC-74LS95 chip Study:

Description: The IC-74LS95 is a 4-bit shift register with serial and parallel synchronous operating modes. It has serial data (D_S) and four parallel data ($D_0 - D_3$) inputs and four parallel outputs ($Q_0 - Q_3$). The serial or parallel mode of operation is controlled by a mode select input (S) and two clock ($\overline{CP_1}$ and $\overline{CP_2}$) inputs. The serial (shift right) or parallel data transfers occur synchronously with the HIGH – to – LOW transition of the selected clock input.

When the mode select input (S) is $\overline{CP_2}$ HIGH, is enabled. A HIGH – to – LOW transition on $\overline{CP_2}$ enabled loads parallel data from the ($D_0 - D_3$) inputs into the $\overline{CP_1}$ $\overline{CP_1}$ register. When (S) is low, is enabled. A HIGH – to – LOW transition on enabled shifts the data from serial input D_S to Q_0 and transfers the data from Q_0 to Q_1, Q_1 to Q_2 and Q_2 to Q_3 respectively (shift right). Shift left is accomplished by externally connecting Q_3 to D_2, Q_2 to D_1 and Q_1 to D_0 and operating the IC-74LS95 in the parallel mode (S = HIGH)