CS8382 DIGITAL SYSTEMS LABORATORY

LIST OFEXPERIMENTS

1. Verification of Boolean Theorems using basic gates.
2. Design and implementation of combinational circuits using basic gates for arbitrary functions, code converters.
3. Design and implement Half/Full Adder and Subtractor.
4. Design and implement combinational circuits using MSI devices:
   - 4 – bit binary adder / subtractor
   - Parity generator / checker
   - Magnitude Comparator
   - Application using multiplexers
5. Design and implement shift-registers.
6. Design and implement synchronous counters.
7. Design and implement asynchronous counters.
8. Coding combinational circuits using HDL.
9. Coding sequential circuits using HDL.
10. Design and implementation of a simple digital system (Mini Project).

CONTENT BEYOND THE SYLLABUS

Design of traffic light controller using Verilog HDL

Ex No.: 1

Date: **STUDY OF LOGIC GATES**

AIM:

To study about logic gates and verify their truth tables.

APPARATUS REQUIRED:

S.No.	COMPONENT	SPECIFICATION	QTY
1.	ANDGATE	IC7408	1
2.	ORGATE	IC7432	1
3.	NOTGATE	IC7404	1
4.	NAND GATE 2I/P	IC7400	1
5.	NORGATE	IC7402	1
6.	X-ORGATE	IC7486	1
7.	NAND GATE 3I/P	IC7410	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:

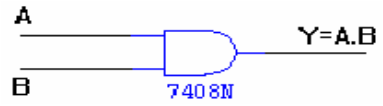
The AND gate performs a logical multiplication commonly known as AND function. The output is high when both the inputs are high. The output is low level when any one of the inputs is low.

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.

AND GATE:

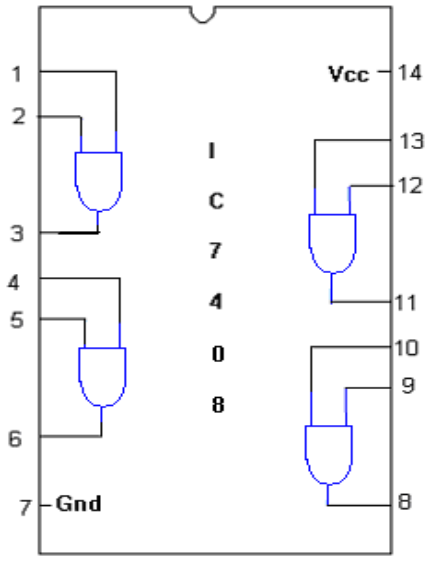
SYMBOL:



TRUTH TABLE

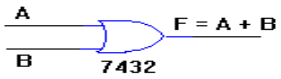
A	B	A.B
0	0	0
0	1	0
1	0	0
1	1	1

PIN DIAGRAM:

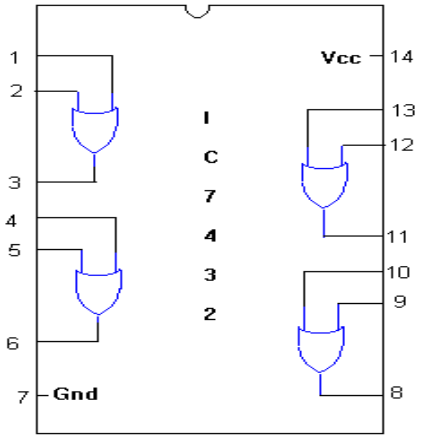


OR GATE

SYMBOL :



PIN DIAGRAM :



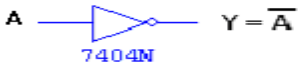
TRUTH TABLE

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

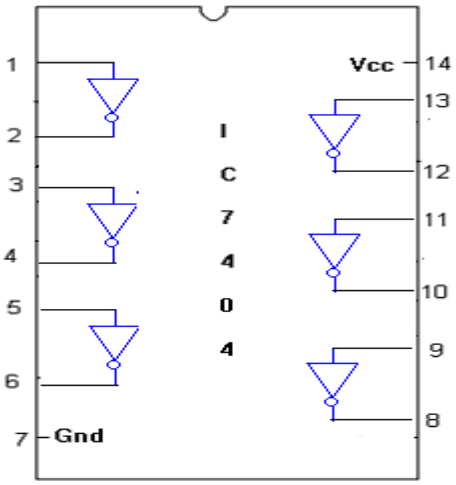
:

NOT GATE:

SYMBOL:



PIN DIAGRAM:

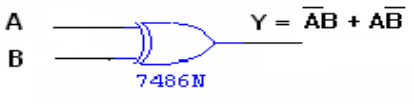


TRUTH TABLE :

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

X-OR GATE:

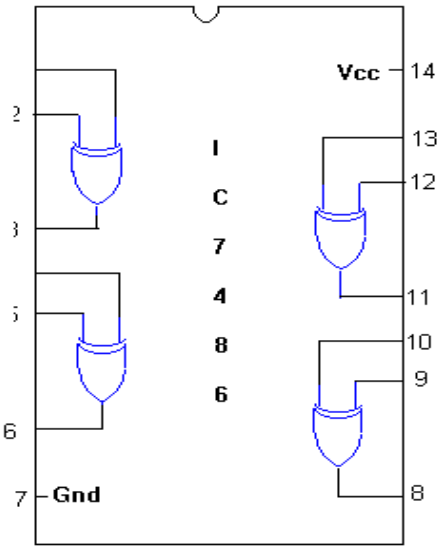
SYMBOL:



TRUTH TABLE :

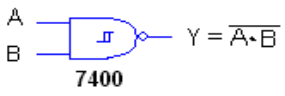
A	B	$\overline{A}B + A\overline{B}$
0	0	0
0	1	1
1	0	1
1	1	0

PIN DIAGRAM:



2- INPUT NAND GATE:

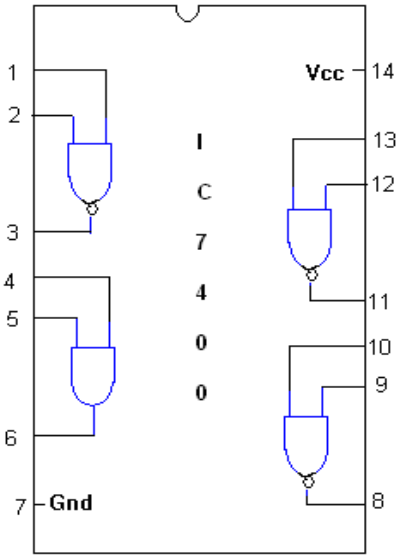
SYMBOL:



TRUTH TABLE

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

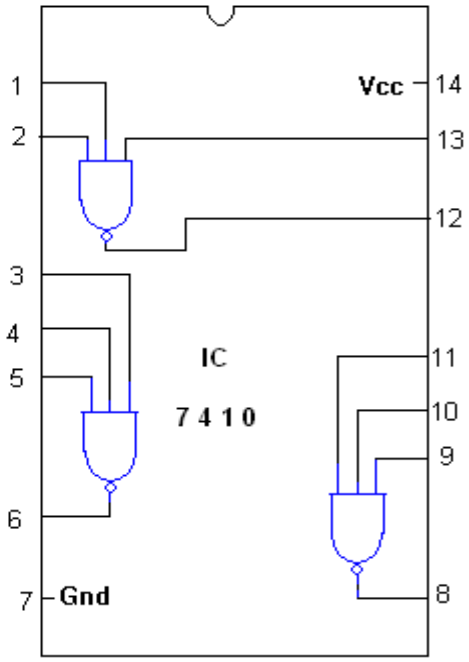
PIN DIAGRAM:



3-INPUT NAND GATE:

SYMBOL :

PIN DIAGRAM :

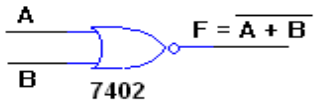


TRUTH TABLE

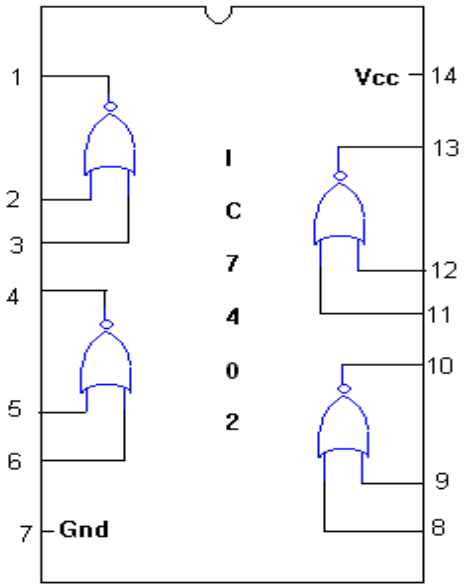
A	B	C	$\overline{A.B.C}$
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

NOR GATE:

SYMBOL :



PIN DIAGRAM :



TRUTH TABLE

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

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-ORGATE:

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.

RESULT:

Thus, the logic gates are studied and verified using truth tables.

EXPT NO. 2 : **VERIFICATION OF BOOLEAN THEOREMS USING DIGITAL LOGIC GATES**

DATE :

AIM:

To verify the Boolean Theorems using logic gates.

APPARATUS REQUIRED:

S.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

BOOLEAN THEOREM:

Theorem:1. x + x = x;x. x = x

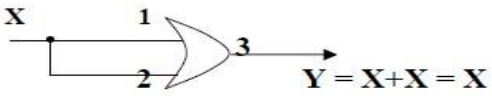
2. x + 1 = 1;x. 0 = 0
3. $(x')' = x$ (Involution)
4. Associative x + (y + z) = (x + y) + z x. (y. z) = (x. y). z
5. De Morgan's $(x + y)' = x'. y'$

   $(x. y)' = x' + y'$
6. Absorption x + yd = x

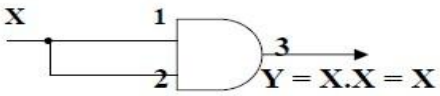
   x. (x + y) = x

CIRCUIT DIAGRAM:

THEOREM : 1



IC 7432



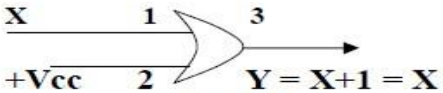
IC 7408

Truth Table:

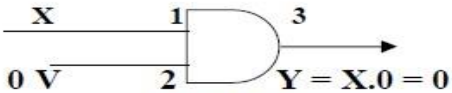
X	X	Y = X+X
0	0	0
1	1	1

X	X	Y = X.X
0	0	0
1	1	1

THEOREM: 2



IC 7432



IC 7408

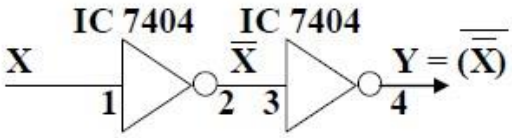
Truth Table

X	1	Y=X+1=1
0	1	1
1	1	1

X	0	Y=X.0=0
0	0	0
1	0	0

THEOREM: 3 (INVOLUTION THEOREM)

Truth Table:

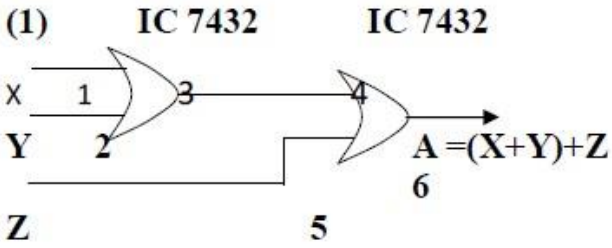


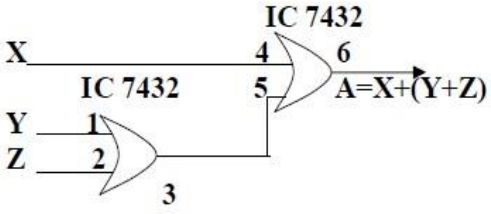
X	$\overline{X}$	$Y = (\overline{X})$
0	1	0
1	0	1

THEOREM:4 (ASSOCIATIVE)

Truth Table:

X	Y	Z	X+Y	A
0	0	0	0	0
0	1	0	1	1
1	0	1	1	1
1	1	1	1	1

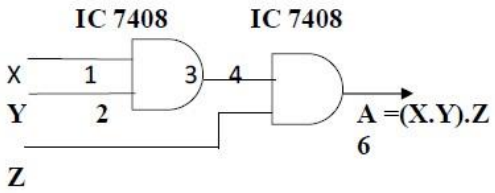




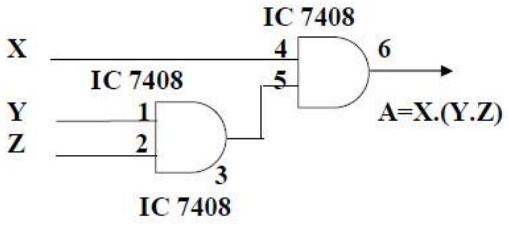
X	Y	Z	Y+Z	A
0	0	0	0	0
0	1	0	1	1
1	0	1	1	1
1	1	1	1	1

(2) (X.Y).Z =X.(Y.Z)

Truth Table:



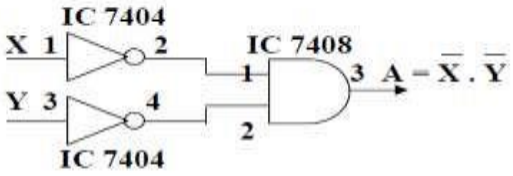
X	Y	Z	X.Y	A
0	0	0	0	0
0	1	0	0	0
1	0	1	0	0
1	1	1	1	1



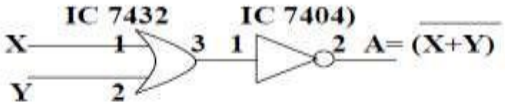
X	Y	Z	Y.Z	A
0	0	0	0	0
0	1	0	0	0
1	0	1	0	0
1	1	1	1	1

THEOREM: 5 DE-MORGAN'S LAW:

(I) X. Y = (X+Y))



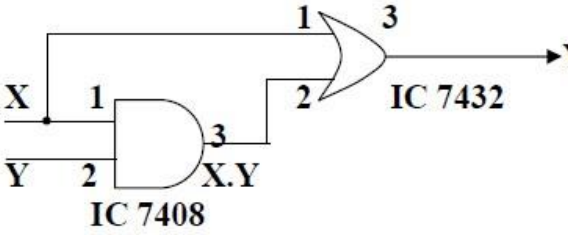
X	Y	$\bar{X}$	$\bar{Y}$	$A=\bar{X}.\bar{Y}$
0	0	1	1	1
0	1	1	0	0
1	0	0	1	0
1	1	0	0	0



Truth Table:

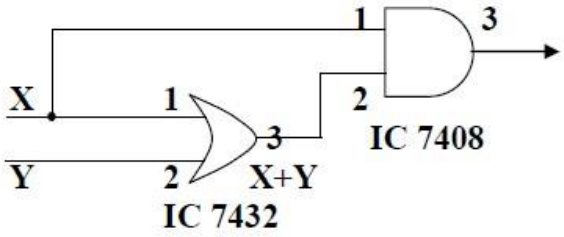
X	Y	X+Y	$A=\overline{(X+Y)}$
0	0	0	1
0	1	1	0
1	0	1	0
1	1	1	0

THEOREM: 6 (ABSORPTION)



Truth Table:

X	Y	X+Y	Y = X + X.Y	X
0	0	0	0	0
0	1	0	0	0
1	0	0	1	1
1	1	1	1	1



Truth Table:

X	Y	X+Y	Y = X.(X+Y)	X
0	0	0	0	0
0	1	0	0	0
1	0	0	1	1
1	1	1	1	1

PROCEDURE:

1. The connections are made as per the circuit diagram.
2. Give the logical inputs as per the truth table.
3. The corresponding output is verified with their truth table.

RESULT: Thus, the Boolean Theorems were verified using logic gates.

EXP NO.: 3 **DESIGN OF ADDER AND SUBTRACTOR**

DATE:

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:

S.No.	COMPONENT	SPECIFICATION	QTY.
1.	ANDGATE	IC7408	1
2.	X-ORGATE	IC7486	1
3.	NOTGATE	IC7404	1
4.	ORGATE	IC7432	1
3.	ICTRAINERKIT	-	1
4.	PATCHCORDS	-	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:

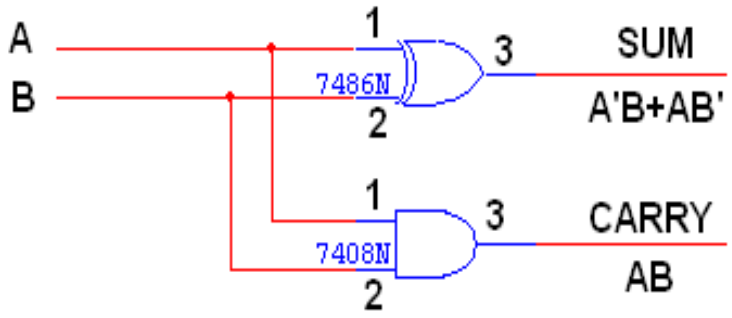
A full adder is a combinational circuit that forms the arithmetic sum of input; it consists of three inputs and two outputs. A full adder is useful to add three bits at a time but a half adder cannot do so. In full adder sum output will be taken from X-OR Gate, carry output will be taken from OR Gate.

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 implemented

using an AND Gate and an inverter.

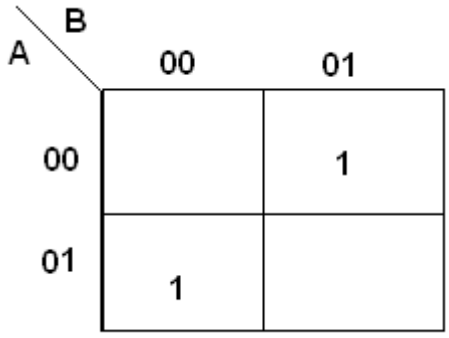
HALFADDER



TRUTHTABLE:

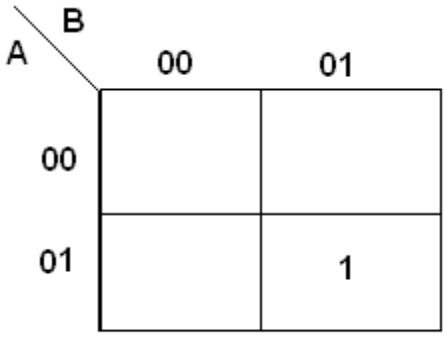
A	B	CARRY	SUM
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

K-MapforSUM:



SUM=A'B+AB'

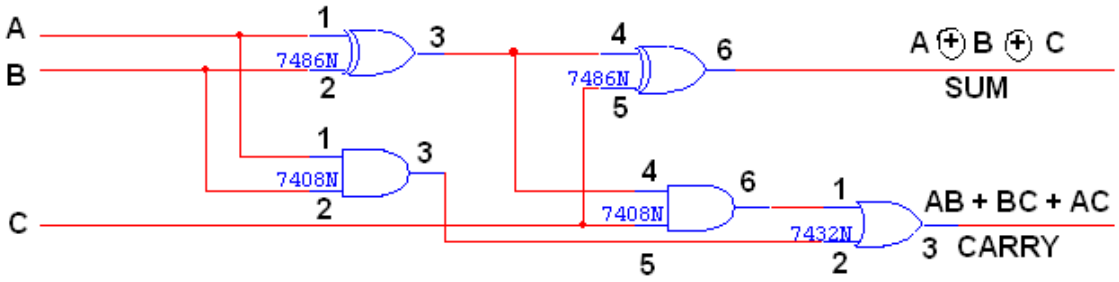
K-MapforCARRY:



CARRY =AB

FULLADDER

FULLADDERUSINGTWOHALFADDER



TRUTHTABLE:

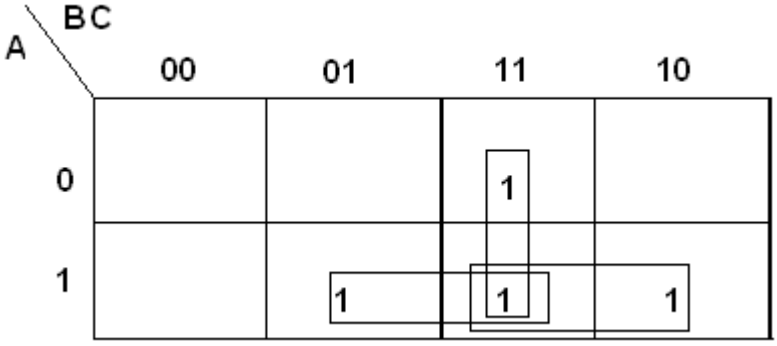
A	B	C	CARRY	SUM
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

K-MapforSUM:

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

SUM=A'B'C+A'BC'+ABC'+ABC

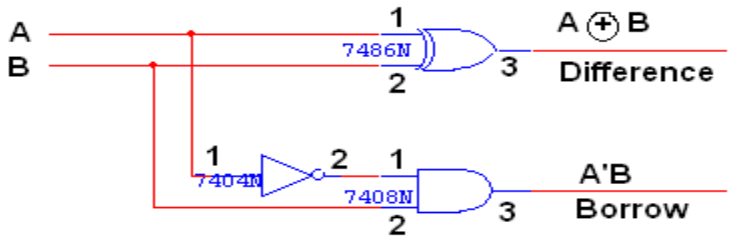
K-Map for CARRY:



CARRY =AB + BC+ AC

LOGIC DIAGRAM:

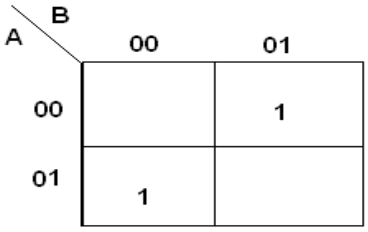
HALF SUBTRACTOR



TRUTH TABLE:

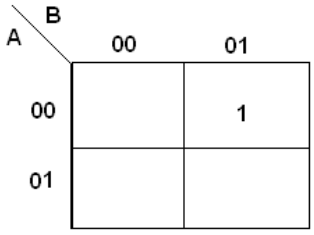
A	B	BORROW	DIFFERENCE
0 1 1	1 0 1	1 0 0	1 1 0

K-Map for DIFFERENCE:



DIFFERENCE=A'B+AB'

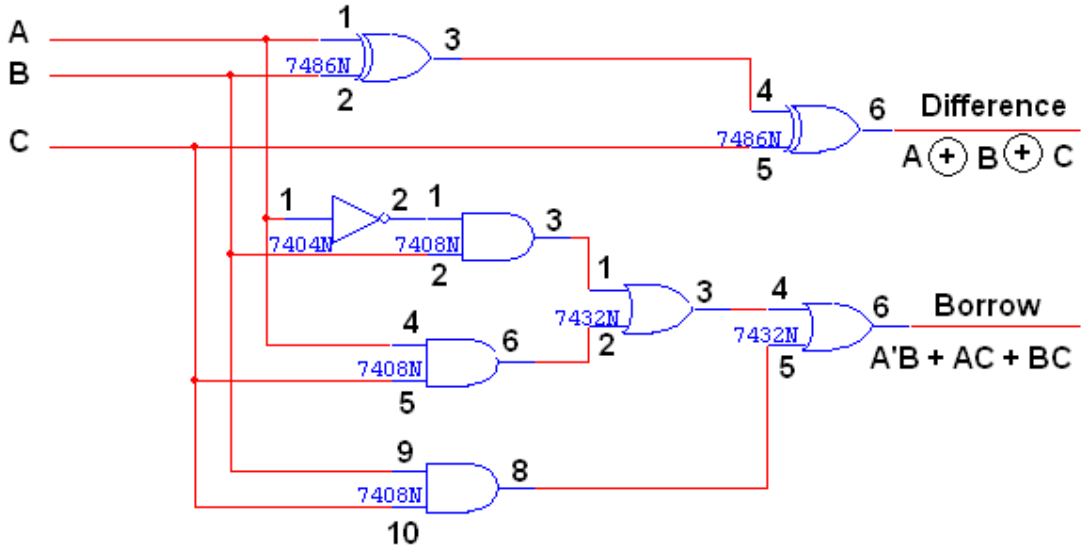
K-Map for BORROW:



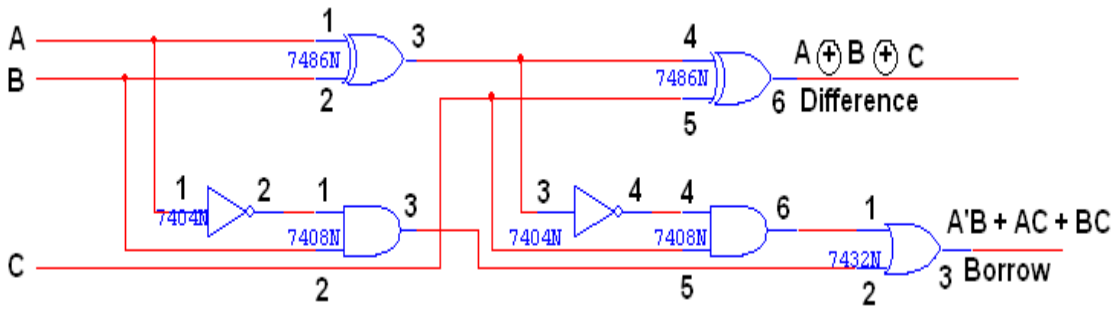
BORROW=A'B

Logic Diagram:

Full Subtractor:



FULL SUBTRACTOR USING TWO HALF SUBTRACTOR:



TRUTHTABLE:

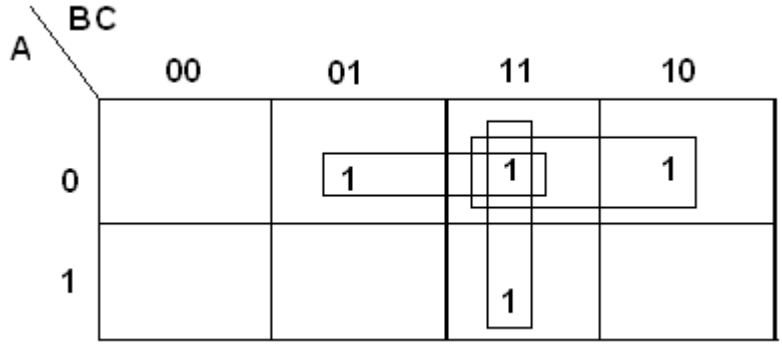
A	B	C	BORROW	DIFFERENCE
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	1	0
1	0	0	0	1
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

K-Map for Difference:

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

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

K-Map for Borrow:



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 AB. 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.

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 Subtractors circuits are designed and constructed and verified using the truth table.

EX.NO: 4

DATE: **DESIGN OF 4-BIT ADDER AND SUBTRACTOR**

AIM:

To design and implement 4-bit adder and subtractor using IC7483.

APPARATUS REQUIRED:

S.No.	COMPONENT	SPECIFICATION	QTY.
1.	IC	IC7483	1
2.	EX-ORGATE	IC7486	1
3.	NOTGATE	IC7404	1
3.	ICTRAINERKIT	-	1
4.	PATCHCORDS	-	40

THEORY:

4- BIT 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 chain. The augends bits of 'A 'andthe addend bits of 'B' are designated by subscript numbers from right to left, with subscript 0 denoting the least significant bits. The carries are connected in chain through the full adder. The input carry to the adder is C_0 and it ripples through the full adderto the output carry C4.

4-BIT BINARYSUBTRACTOR:

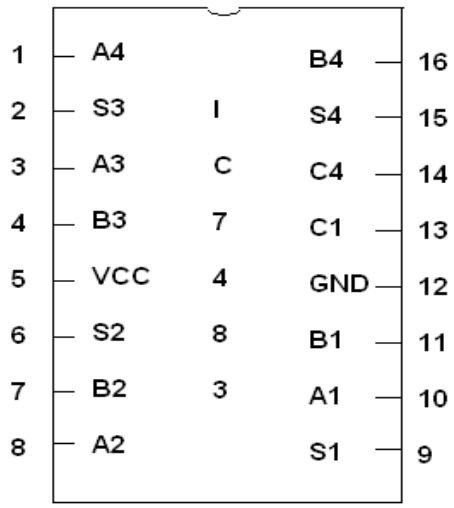
The circuit for subtracting A - B consists of an adder with inverters, placed between each data inputs'' and the corresponding input of full adder. The input carry C_0 must be equal to1 when performing subtraction.

4-BIT BINARY ADDER/SUBTRACTOR:

The addition and subtraction operation can be combined into one circuit with one common binary adder. The mode input M controls the operation. When

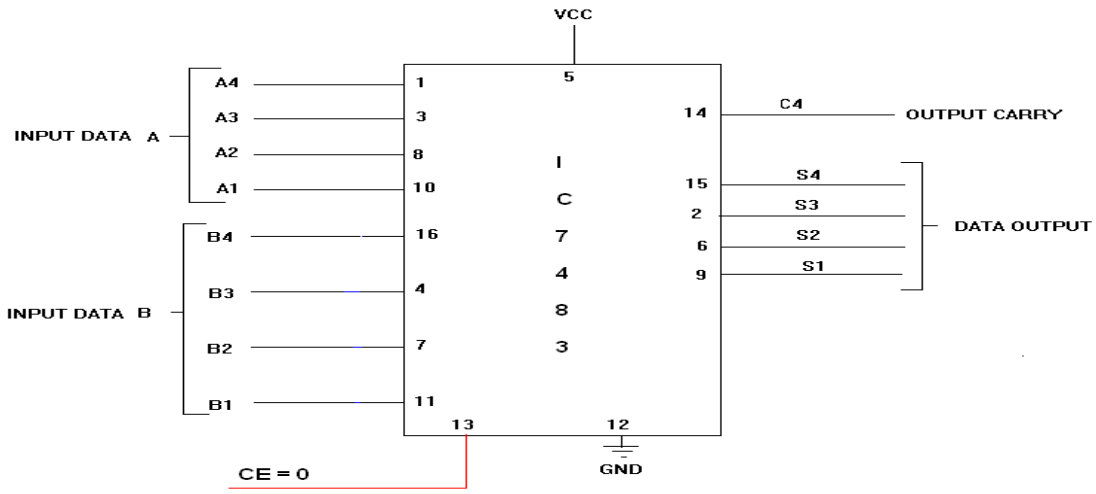
M=0, the circuit is adder circuit. When M=1, it becomes subtractor.

PIN DIAGRAM FOR IC7483:



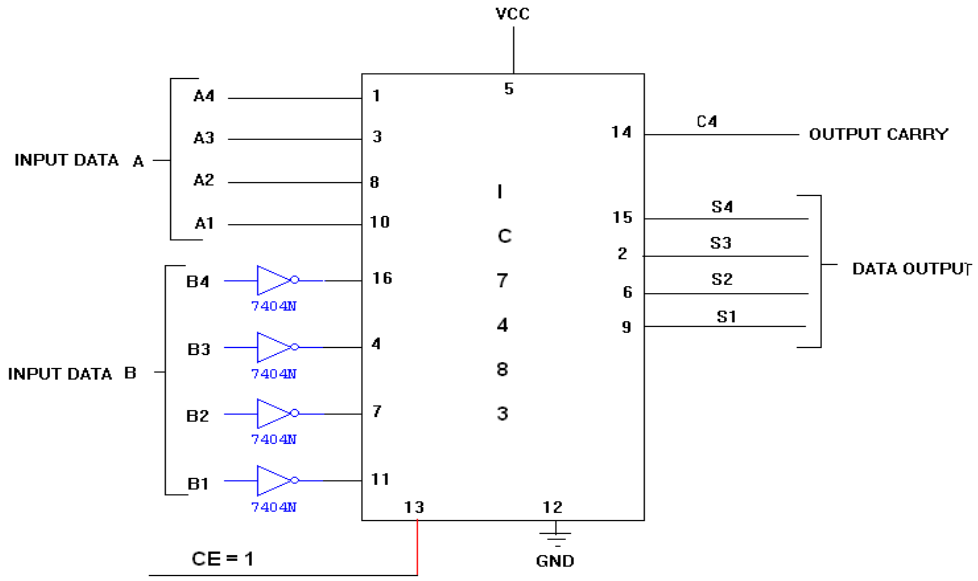
LOGIC DIAGRAM:

4-BIT BINARY ADDER



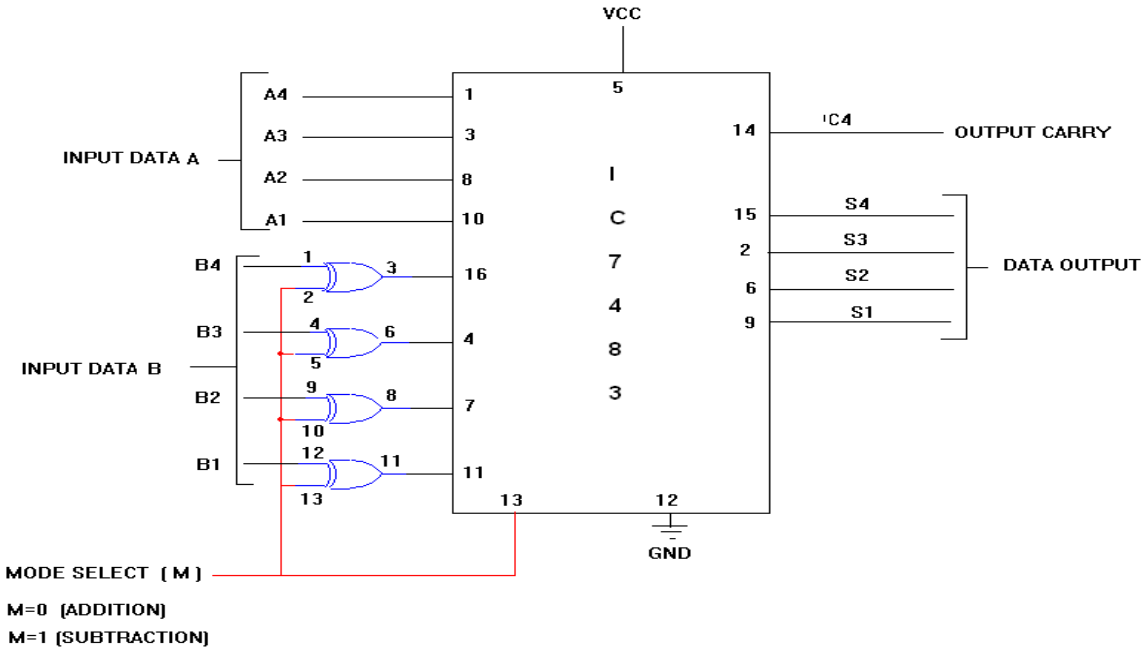
LOGIC DIAGRAM:

4-BIT BINARY SUBTRACTOR



LOGIC DIAGRAM:

4-BIT BINARY ADDER/SUBTRACTOR



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

4-BIT BCD ADDER:

Consider the arithmetic addition of two decimal digits in BCD, together with an input carry from a previous stage. Since each input digit does not exceed 9, the output sum cannot be greaterthan19, the 1 in the sum being an input carry. The output of two decimal digits must be represented in BCD and should appear in the form listed in the columns.

A BCD adder that adds 2 BCD digits and produce a sum digit in BCD. The 2 decimal digits, together with the input carry, are first added in the top 4-bitadder to produce the binary sum.

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 4-bit adder and subtractor are designed and implemented using IC7483.

EX.NO :5 DESIGN AND IMPLEMENTATION OF CODE CONVERTORS

DATE:

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:

S.No.	COMPONENT	SPECIFICATION	QTY.
1.	X-ORGATE	IC7486	1
2.	ANDGATE	IC7408	1
3.	ORGATE	IC7432	1
4.	NOTGATE	IC7404	1
5.	ICTRAINERKIT	-	1
6.	PATCHCORDS	-	35

THEORY:

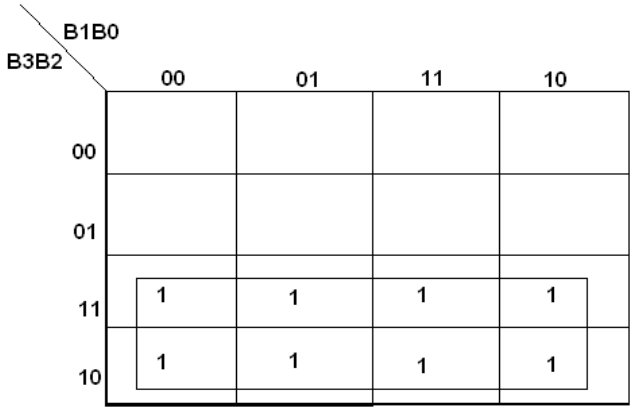
The availability of large variety of codes for the same discrete elements of information results in the use of different codes by different systems. A conversion circuit must be inserted between the two systems if each uses different codes for same information. Thus, code converter is a circuit that 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. Theinput variable is designated as B3, B2, B1, B0 and the output variables are designated asC3, C2, C1, Co from the truth table, Combinational circuit is designed. 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-3code, the input lines must supply the bit combination of elements as specified by code and the output lines generate the correspondingbit combination of code. Each one of the four maps representone 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. The various other possibilities for a logic diagram that implements this circuit. Now the OR gate whose output isC+D has been used to implement partially each of three outputs.

TRUTH TABLE:

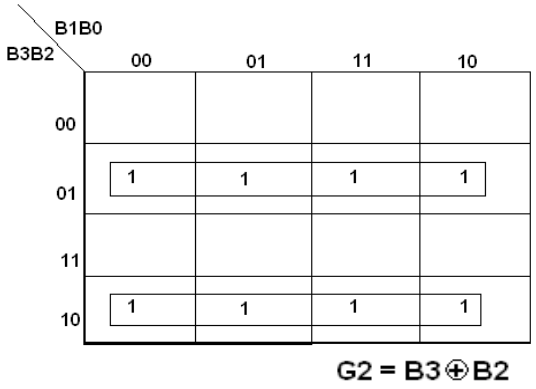
Binary input				**Gray code output**			
B3	**B2**	**B1**	**B0**	**G3**	**G2**	**G1**	**G0**
0	**0**	**0**	**1**	**0**	**0**	**0**	**1**
0	**0**	**1**	**0**	**0**	**0**	**1**	**1**
0	**0**	**1**	**1**	**0**	**0**	**1**	**0**
0	**1**	**0**	**0**	**0**	**1**	**1**	**0**
0	**1**	**0**	**1**	**0**	**1**	**1**	**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 G3:

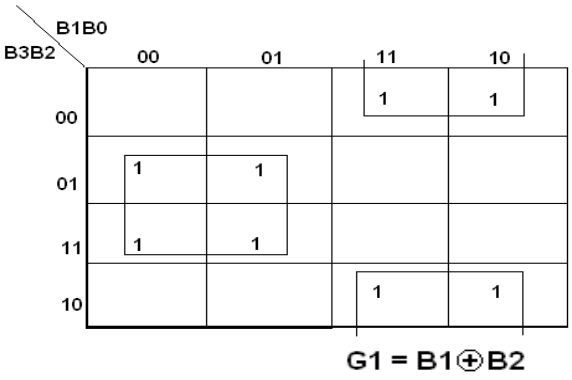


$G_3=B_3$

K-Map for G2:



K-Map for G1:



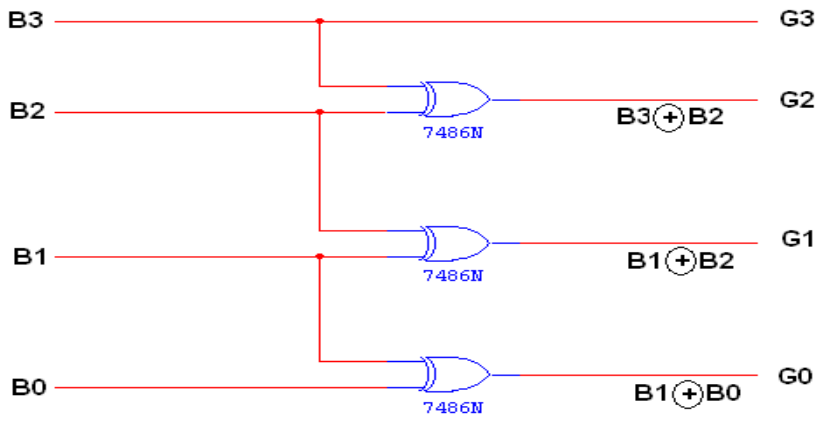
K-Map for G0:

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

$$G0 = B1 \oplus B0$$

LOGIC DIAGRAM:

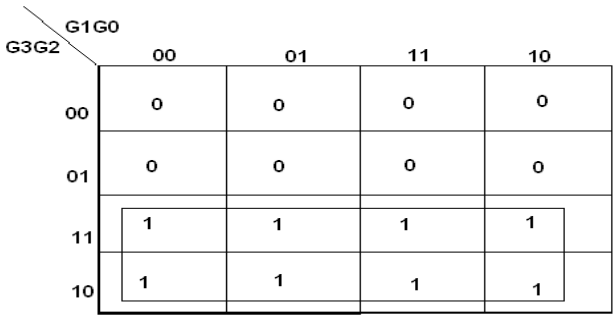
BINARY TO GRAY CODE CONVERTOR



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 B3:



$B_3=G_3$

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

$B2 = G3 \oplus G2$

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

$B1 = G3 \oplus G2 \oplus G1$

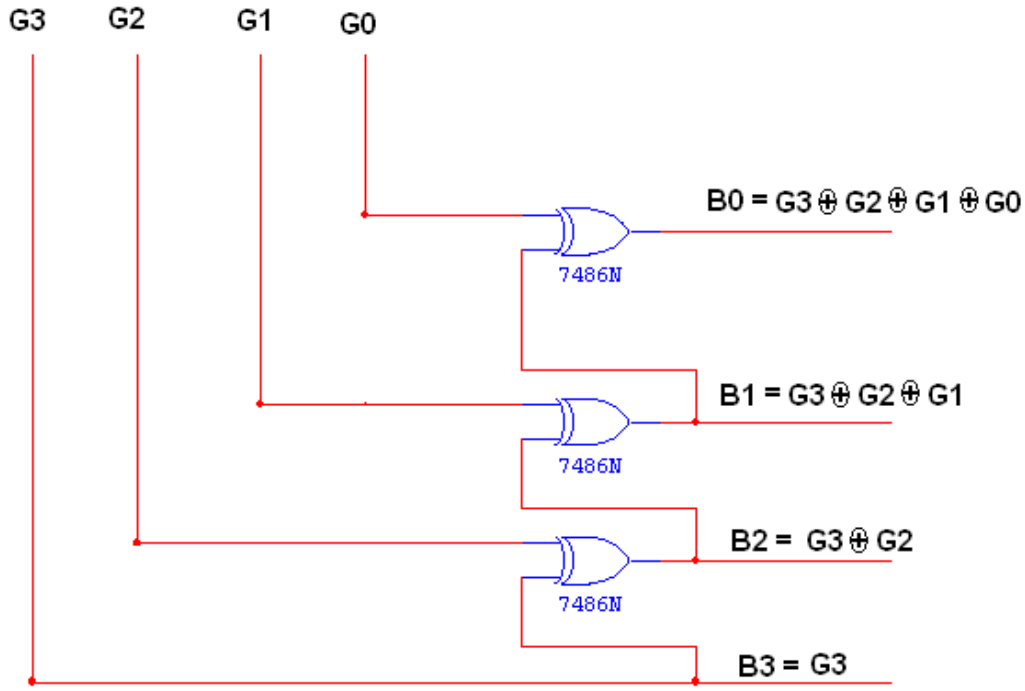
K-Map for B0:

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:

GRAY CODE TO BINARY 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 code convertors are designed and verified using truth table.

EXP NO:6 DESIGN AND IMPLEMENTATION OF MULTIPLEXER AND DEMULTIPLEXER

AIM:

To design and implement multiplexer and demultiplexer using logic Gates.

APPARATUS REQUIRED:

S.No.	COMPONENT	SPECIFICATION	QTY.
1.	3I/P ANDGATE	IC7411	2
2.	ORGATE	IC7432	1
3.	NOTGATE	IC7404	1
2.	ICTRAINERKIT	-	1
3.	PATCHCORDS	-	32

THEORY:

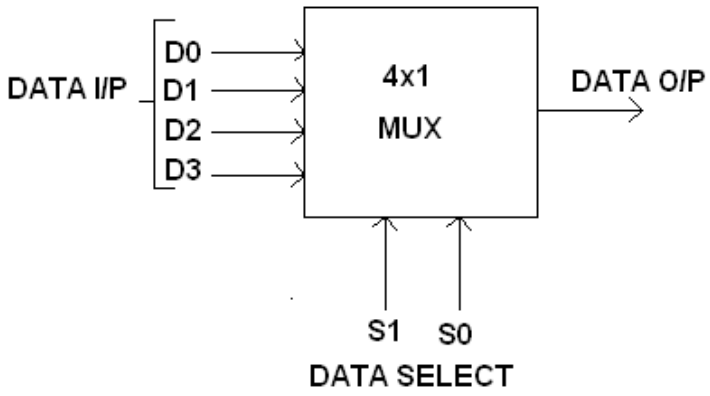
MULTIPLEXER:

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 2n 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 1:4demultiplexer circuit, the data input line goes to all of the AND gates. The data select lines enable only one gate at time and the data on the data input line will pass through the selected gate to the associated data output line.

BLOCK DIAGRAM FOR 4:1 MULTIPLEXER:

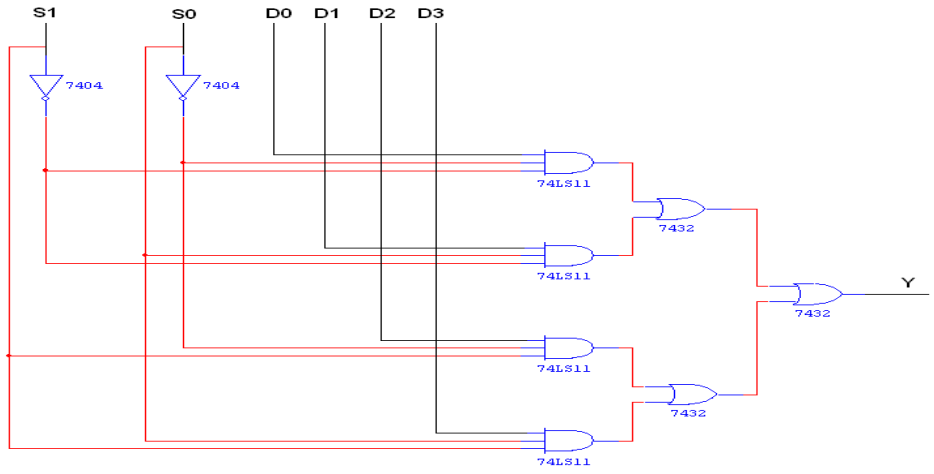


FUNCTIONTABLE:

S1	S0	INPUTSY
0	0	D0 → D0S1' S0'
0	1	D1 → D1 S1' S0
1	0	D2 → D2 S1 S0'
1	1	D3 → D3 S1 S0

Y= D0 S1' S0' + D1S1' S0 + D2 S1 S0' + D3 S1S0

CIRCUIT DIAGRAM FOR MULTIPLEXER:

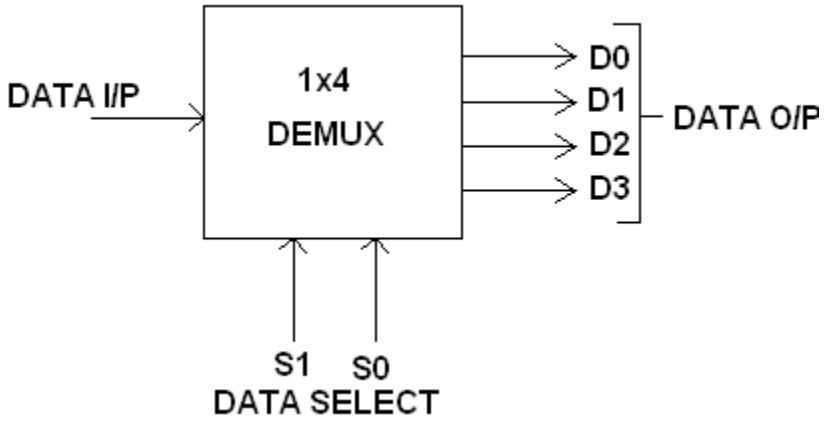


TRUTHTABLE:

S1	S0	Y=OUTPUT
0	0	D0

0	1	D1
1	0	D2
1	1	D3

BLOCK DIAGRAM FOR 1:4 DEMULTIPLEXER:

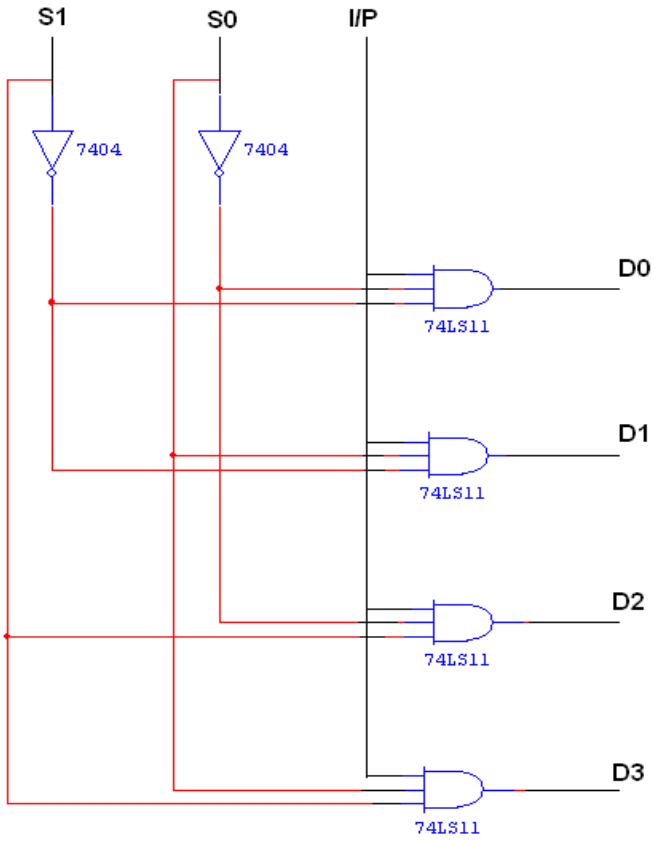


FUNCTIONTABLE:

S1	S0	INPUT
0	0	X→ D0 =XS1' S0'
0	1	X→ D1 =XS1' S0
1	0	X→ D2 =XS1 S0'
1	1	X→ D3 =XS1 S0

Y= XS1' S0' +X S1' S0 + XS1 S0' + XS1S0

LOGIC DIAGRAM FOR DEMULTIPLEXER:



TRUTHTABLE:

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

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 are designed and implemented using logic Gates.

EXP NO: 7 CONSTRUCTION AND VERIFICATION OF 4BIT RIPPLECOUNTER AND MOD10COUNTER(Asynchronous)

DATE:

AIM:

To design and verify 4bit ripple counter mod 10 counter.

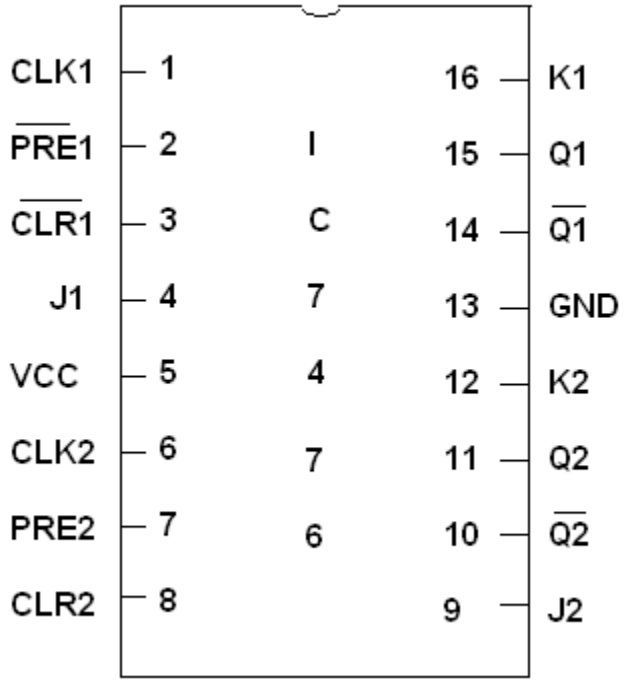
APPARATUSREQUIRED:

S.No.	COMPONENT	SPECIFICATION	QTY.
1.	JKFLIPFLOP	IC7476	2
2.	NANDGATE	IC7400	1
3.	ICTRAINERKIT	-	1
4.	PATCHCORDS	-	30

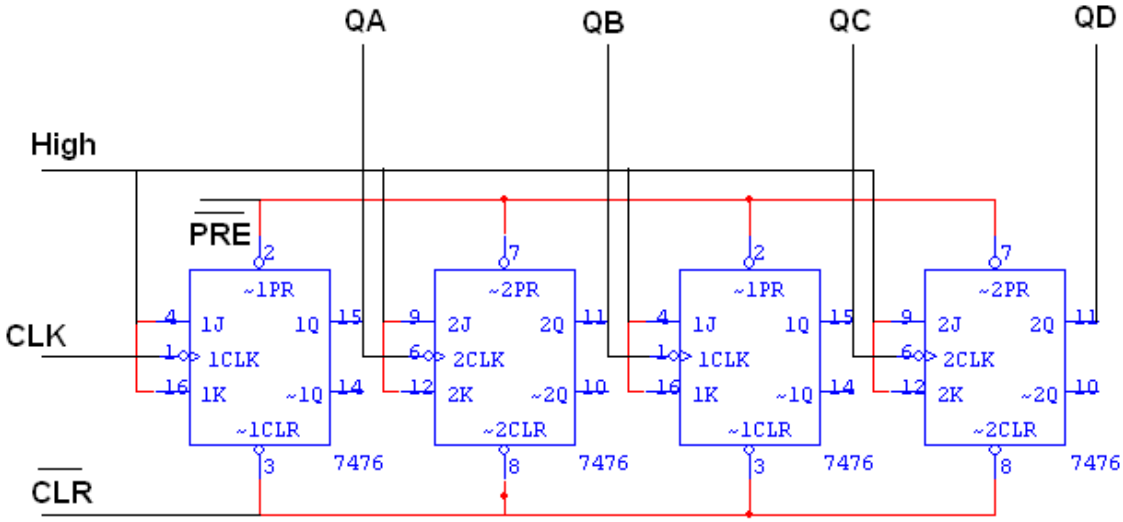
THEORY:

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 flops 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 IC7476:



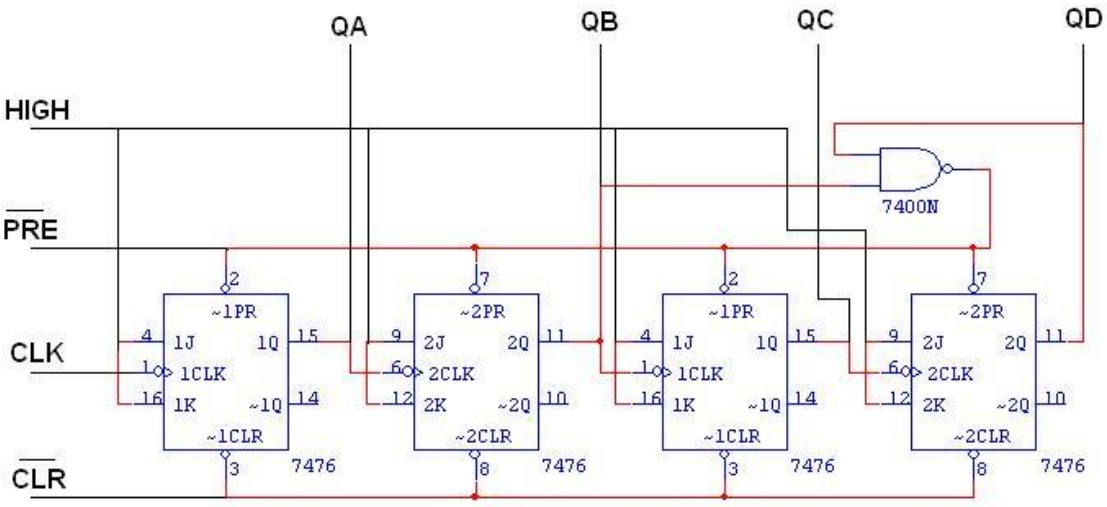
LOGIC DIAGRAM FOR 4 –BIT RIPPLE COUNTER:



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	1	0	1
11	1	1	0	1

LOGIC DIAGRAM FOR MOD – 10 RIPPLECOUNTERS:



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

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 asynchronous counter is designed and verified using truth table.

EXP. NO.:8 DESIGNS AND IMPLEMENTATION OF 3 BIT SYNCHRONOUS UP/DOWN COUNTER

DATE :

AIM:

To design and implement 3 bit synchronous up/down counter.

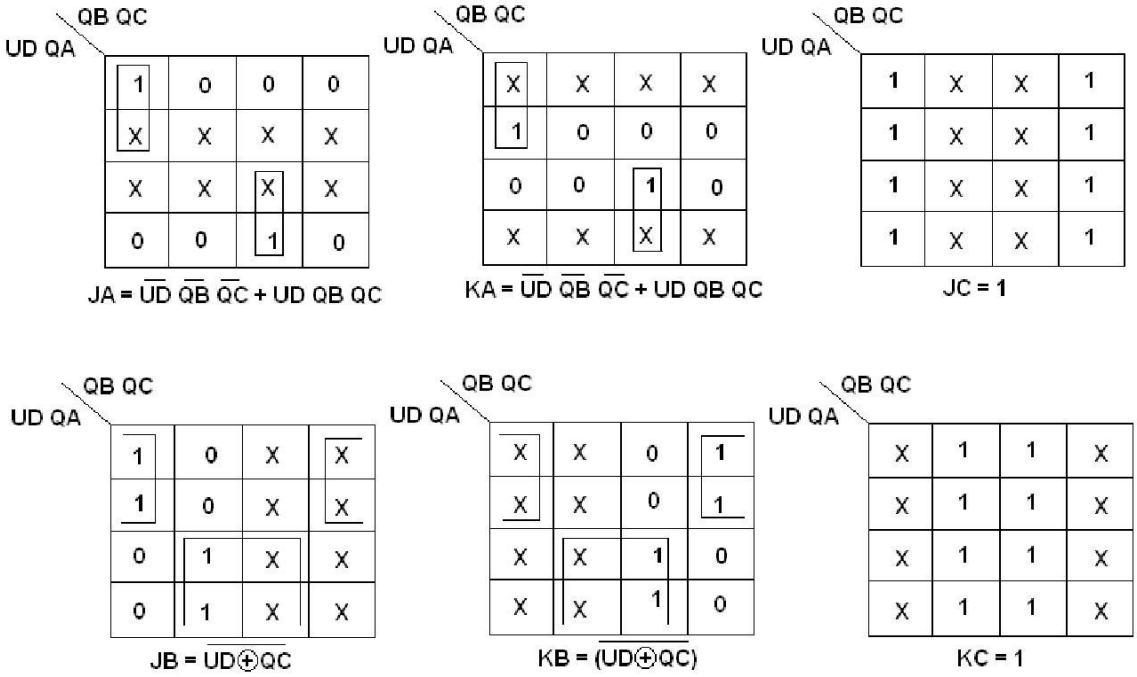
APPARATUS REQUIRED:

S.No.	COMPONENT	SPECIFICATION	QTY.
1.	JK FLIP FLOP	IC 7476	2
2.	3 I/P AND GATE	IC 7411	1
3.	OR GATE	IC 7432	1
4.	XOR GATE	IC 7486	1
5.	NOT GATE	IC 7404	1
6.	IC TRAINER KIT	-	1
7.	PATCH CORDS	-	35

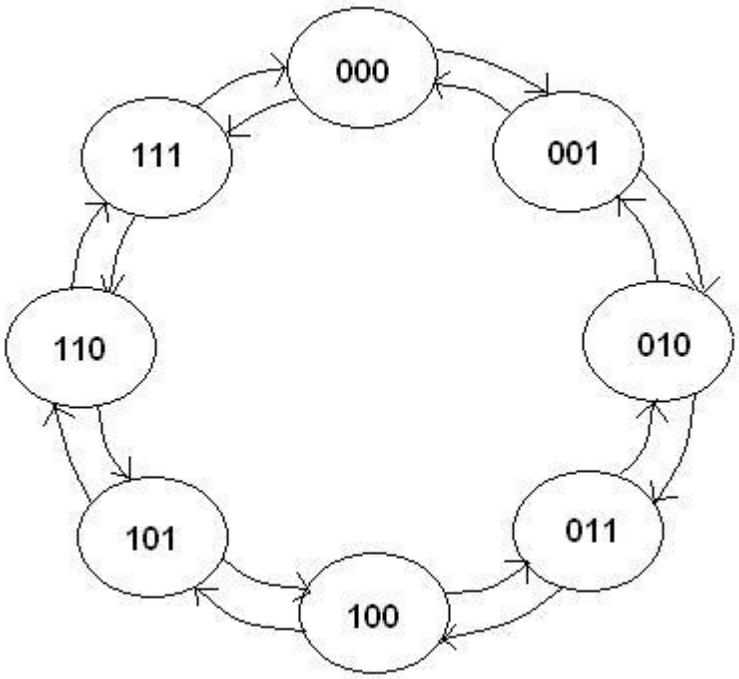
THEORY:

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. An up/down counter is one that is capable of progressing in increasing order or decreasing order through a certain sequence. An up/down counter is also called bidirectional counter. Usually up/down operation of the counter is controlled by up/down signal. When this signal is high counter goes through up sequence and when up/down signal is low counter follows reverse sequence.

K MAP



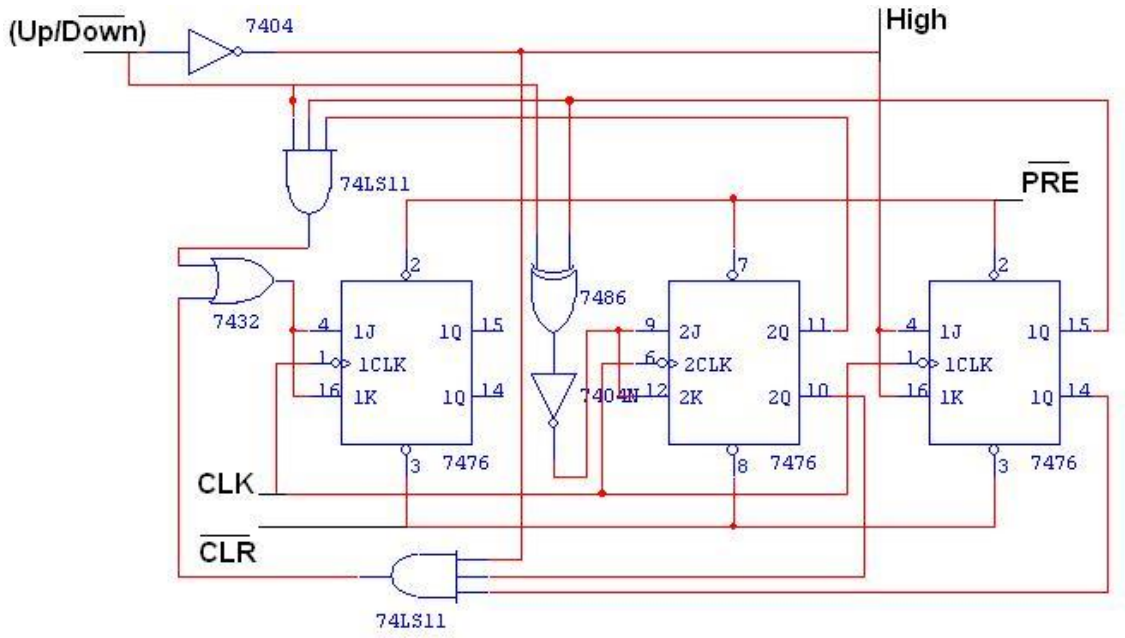
STATE DIAGRAM:



CHARACTERISTICS TABLE:

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

LOGIC DIAGRAM:



TRUTH TABLE:

Input	Present State			Next State			A		B		C	
				QA+1	Q	B+1	J					
Up/Down	QA	QB	QC	QC+1			A	KA	JB	KB	JC	KC
0	0	0	0	1	1	1	1	X	1	X	1	X
0	1	1	1	1	1	0	X	0	X	0	X	1
0	1	1	0	1	0	1	X	0	X	1	1	X
0	1	0	1	1	0	0	X	0	0	X	X	1
0	1	0	0	0	1	1	X	1	1	X	1	X
0	0	1	1	0	1	0	0	X	X	0	X	1
0	0	1	0	0	0	1	0	X	X	1	1	X
0	0	0	1	0	0	0	0	X	0	X	X	1
1	0	0	0	0	0	1	0	X	0	X	1	X
1	0	0	1	0	1	0	0	X	1	X	X	1
1	0	1	0	0	1	1	0	X	X	0	1	X
1	0	1	1	1	0	0	1	X	X	1	X	1
1	1	0	0	1	0	1	X	0	0	X	1	X
1	1	0	1	1	1	0	X	0	1	X	X	1
1	1	1	0	1	1	1	X	0	X	0	1	X
1	1	1	1	0	0	0	X	1	X	1	X	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 counter has been designed and verified.

EX NO. :9 DESIGN AND IMPLEMENTATION OF SHIFT REGISTERS

DATE:

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

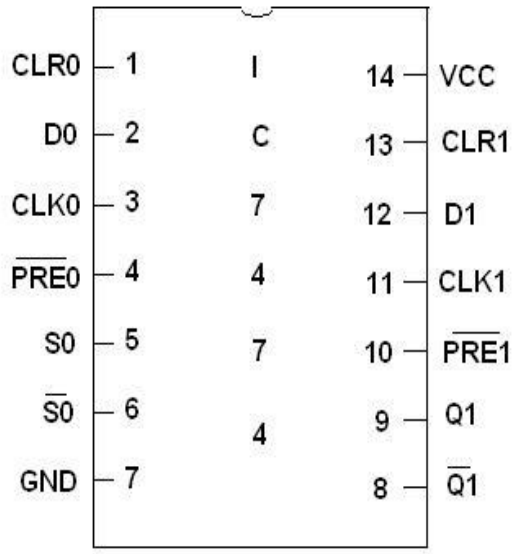
APPARATUS REQUIRED:

S.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:

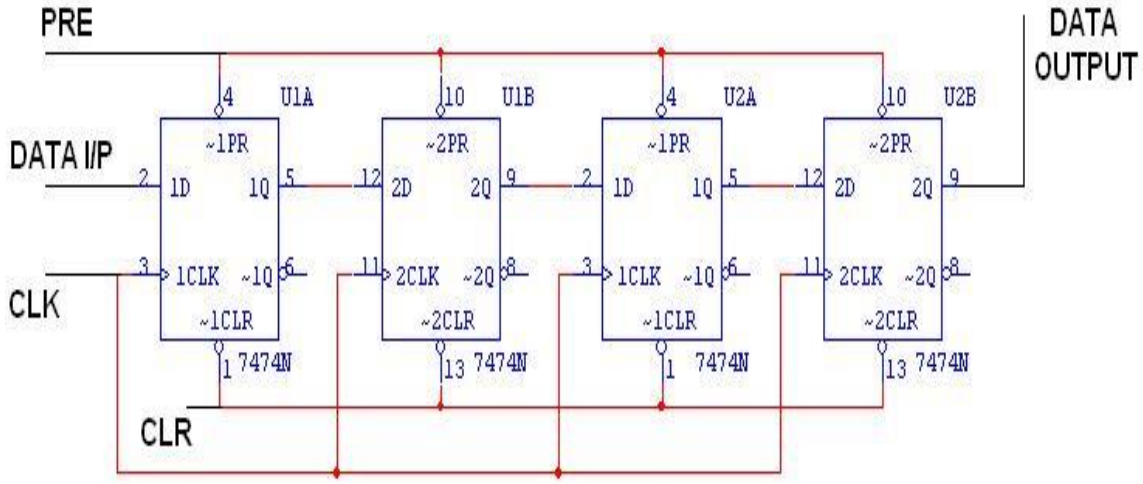
A register is capable of shifting its binary information in one or both directions is known as shift register. The logical configuration of shift register consists of a D-Flip flop cascaded with output of one flip flop connected to input of next flip flop. All flip flops receive common clock pulses which 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:



LOGIC DIAGRAM:

SERIAL IN SERIAL OUT:

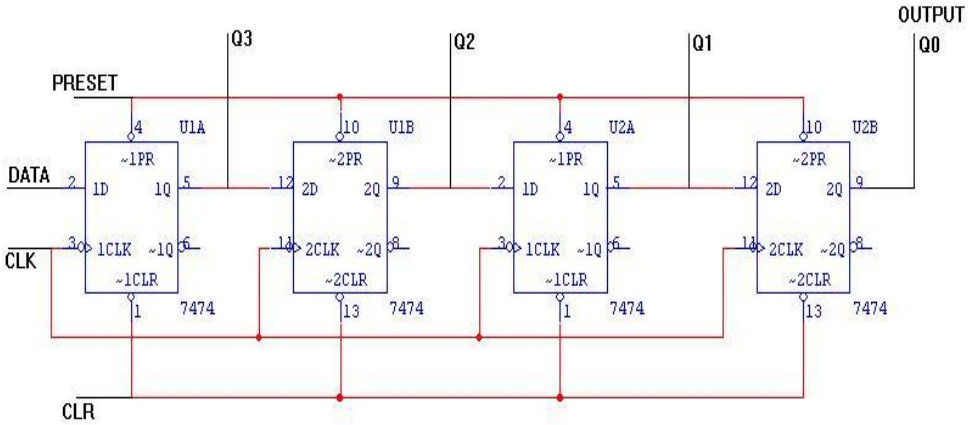


TRUTH TABLE:

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

LOGIC DIAGRAM:

SERIAL IN PARALLEL OUT:

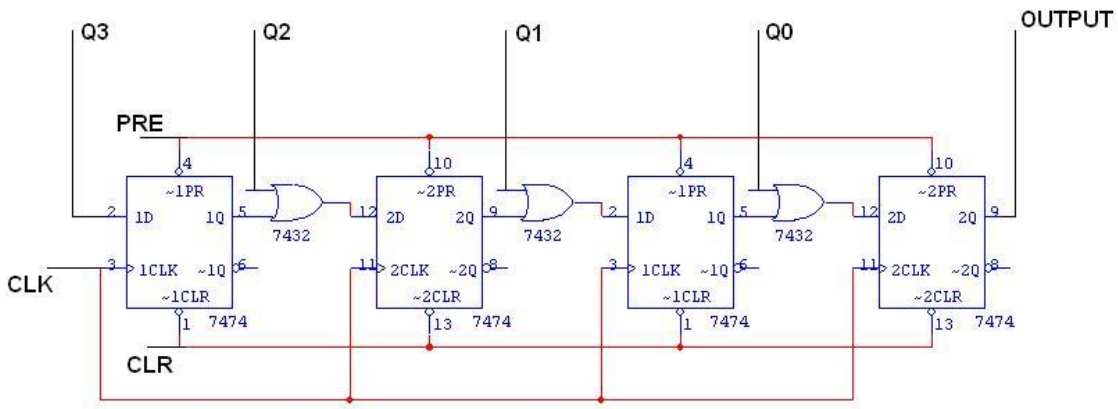


TRUTH TABLE:

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

LOGIC DIAGRAM:

PARALLEL IN SERIAL OUT:

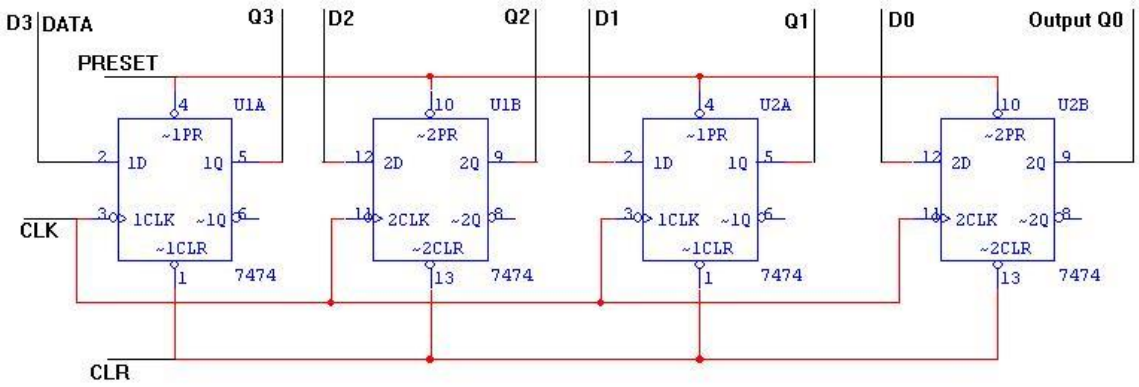


TRUTH TABLE:

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

LOGIC DIAGRAM:

PARALLEL IN PARALLEL OUT:



TRUTH TABLE:

CLK	DATA INPUT				OUTPUT			
	DA	DB	DC	DD	QA	QB	QC	QD
1	1	0	0	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 shift registers have beendesigned and verified using truth table.

EX.NO. :10 **DESIGN AND IMPLEMENTATION OF MAGNITUDE COMPARATOR**

DATE:

AIM:

To design and implement

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

(ii) 8–bit magnitude comparator using IC7485.

APPARATUSREQUIRED:

S.No.	COMPONENT	SPECIFICATION	QTY.
1.	ANDGATE	IC7408	2
2.	X-ORGATE	IC7486	1
3.	ORGATE	IC7432	1
4.	NOTGATE	IC7404	1
5.	4-BITMAGNITUDE COMPARATOR	IC7485	2
6.	ICTRAINERKIT	-	1
7.	PATCHCORDS	-	30

THEORY:

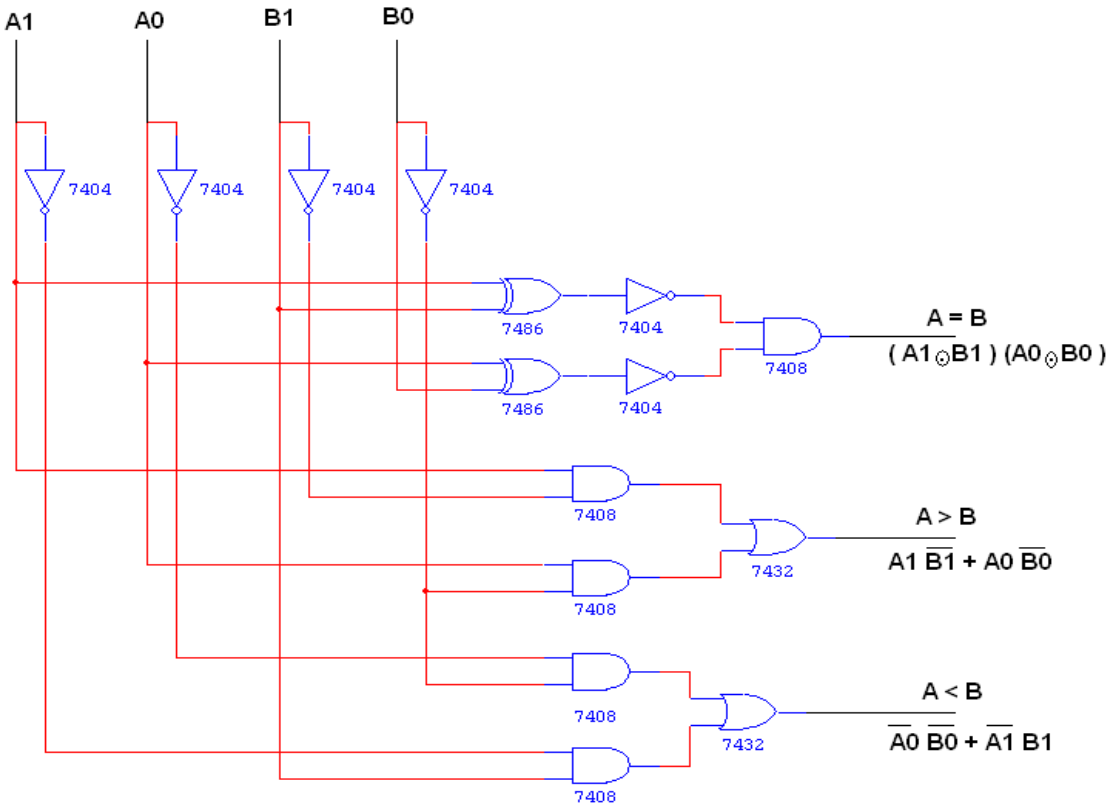
The comparison of two numbers is an operator that determines one number is greater than, less than (or) equal to the other number. A magnitude comparator is a combinational circuit that compares two numbers A and B and determines their relative magnitude. The outcome of the comparator is specifiedby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

A= A3 A2 A1 A0

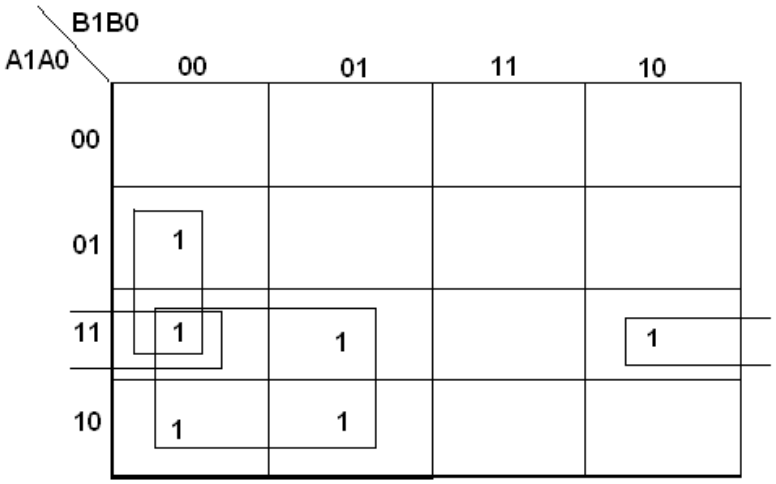
B = B3 B2 B1 B0

LOGIC DIAGRAM:

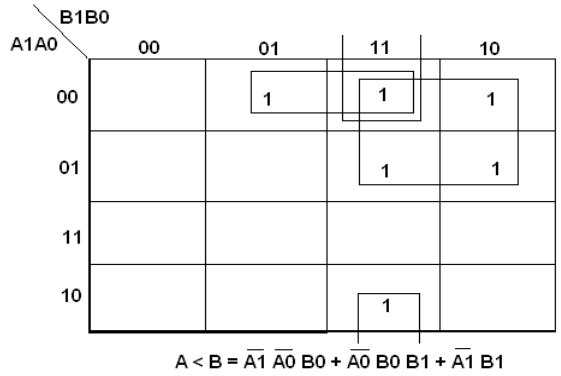
2-BIT MAGNITUDE COMPARATOR



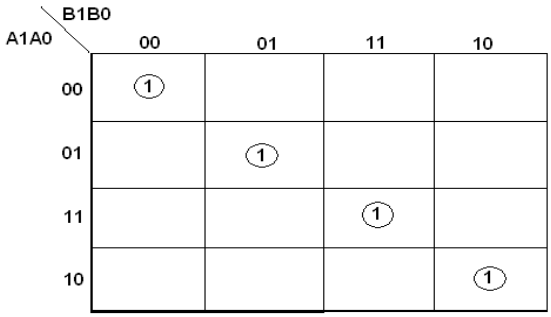
K- MAP



A > B = A0 B̄0 B1 + A1 B̄1 + A1 A0 B̄0



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

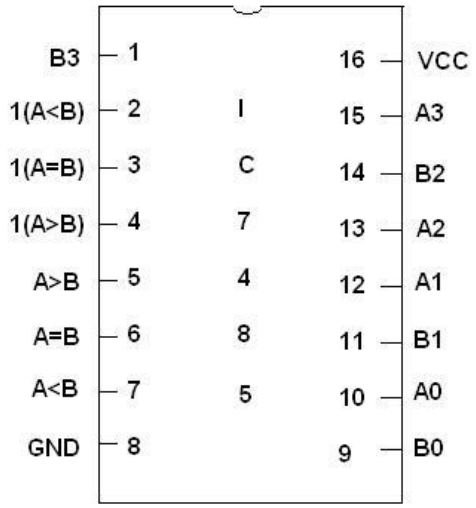


$A = B = (A0 \odot B0)(A1 \odot B1)$

TRUTH TABLE

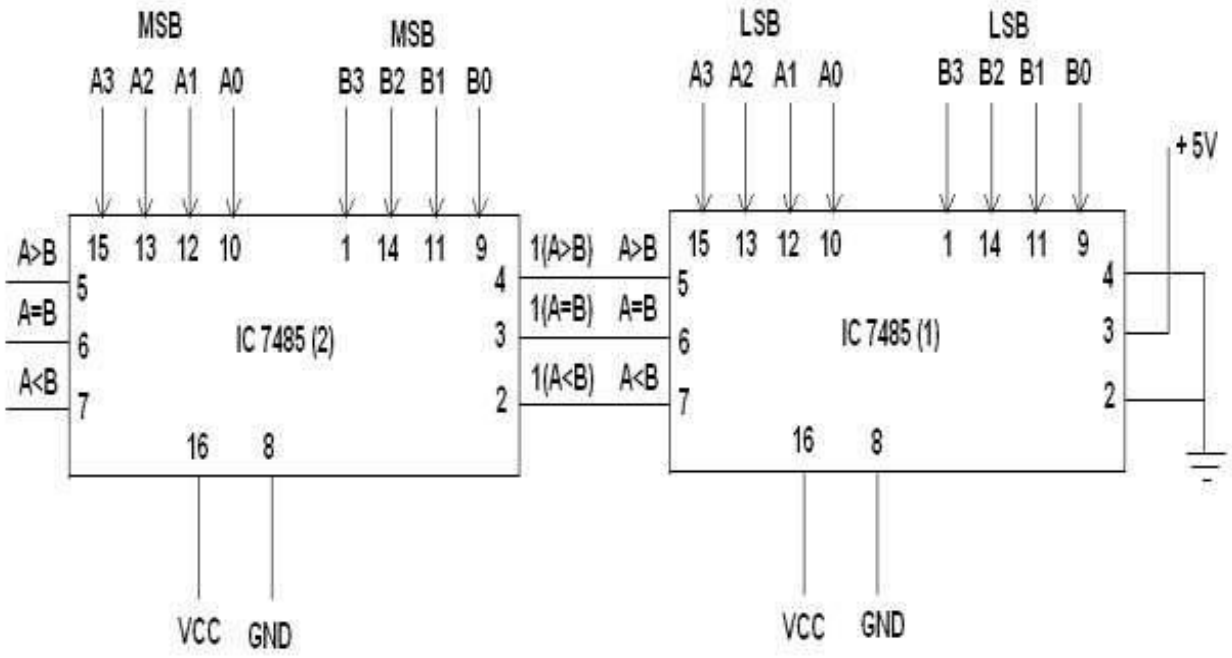
A1	A0	B1	B0	A>B	A=B	A<B
0	0	0	0	0	1	0
0	0	0	1	0	0	1
0	0	1	0	0	0	1
0	0	1	1	0	0	1
0	1	0	0	1	0	0
0	1	0	1	0	1	0
0	1	1	0	0	0	1
0	1	1	1	0	0	1
1	0	0	0	1	0	0
1	0	0	1	1	0	0
1	0	1	0	0	1	0
1	0	1	1	0	0	1
1	1	0	0	1	0	0
1	1	0	1	1	0	0
1	1	1	0	1	0	0
1	1	1	1	0	1	0

DIAGRAM FOR IC 7485:



LOGIC DIAGRAM:

8 BIT MAGNITUDE COMPARATOR



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 comparatorhas beendesigned and verified using truth table.

EX.NO. :11 -BIT ODD / EVEN PARITY CHECKER/GENERATOR

DATE:

AIM:

To design and implement 8 bit odd/even parity checker generator using IC74180.

APPARATUS REQUIRED:

S.No.	COMPONENT	SPECIFICATION	QTY.
1.	NOTGATE	IC7404	1
1.		IC74180	2
2.	ICTRAINERKIT	-	1
3.	PATCHCORDS	-	30

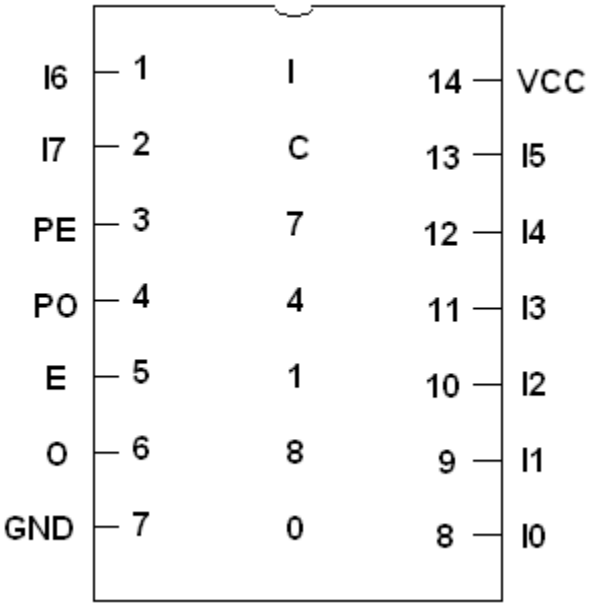
THEORY:

A parity bit is used for detecting errors during transmission of binary information. A parity bit is an extra bit included with a binary message to make the number is either even or odd. The message including the parity bit is transmitted and then checked at the receiver ends for errors. An error is detected if the checked parity bit doesn't correspond to the one transmitted. The circuit that generates the parity bit in the transmitter is called a 'parity generator' and the circuit that checks the parity in the receiver is called a 'parity checker'.

In even parity, the added parity bit will make the total number is even amount. In oddparity, the added parity bit will make the total number is odd amount. The parity checker circuit checks for possible errors in the transmission. If the information is passed in even parity, then the bits required must have an even number of 1's. An error occurs during

transmission,if the received bits have an odd number of 1's indicating that one bit has changed in value during transmission.

PIN DIAGRAM FOR IC74180:



FUNCTION TABLE:

INPUTS			OUTPUTS	
Number of High Data Inputs (I0 – I7)	PE	PO	∑E	∑O
EVEN	1	0	1	0
ODD	1	0	0	1
EVEN	0	1	0	1
ODD	0	1	1	0
X	1	1	0	0
X	0	0	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:

Thus, the parity generator /Checker has been designed and verified.

EX.

NO: 12 SIMULATION OF COMBINATIONAL CIRCUITS USING VERILOG HDL

DATE:

AIM:

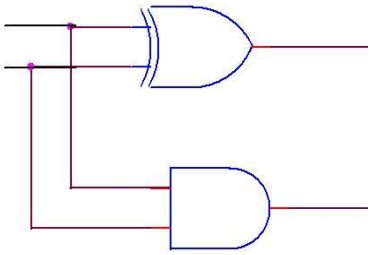
To write Verilog codes for half adder, full adder and multiplexer and simulate them

TOOLS REQUIRED:

Xilinx 9.2

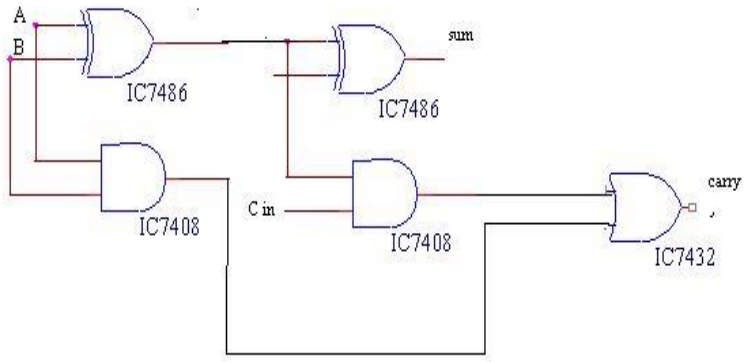
PROGRAM:

HALF ADDER



```
module halfadd(a, b, s, c);
    input a;
    input b;
    output s;
    output c;
    xor (s,a,b);
    and (c,a,b);
endmodule
```

FULL ADDER




```
module fulladd(a, b, c, s, ca);
    input a;
    input b;
    input c;
    output s;
    output ca;
    wire  s1,c1,c2;
    xor g1(s1,a,b);
    and g2(c1,a,b);
    xor g3 (s,s1,c);
    and g4(c2,s1,c);
    or g5(ca,c1,c2);
endmodule
```

MULTIPLEXER:

```
module mux1(y, s0, s1, i0, i1, i2, i3);
    output y;
    input s0;
    input s1;
    input i0;
    input i1;
    input i2;
    input i3;
    reg y;
always@(s0,s1)
begin
if(s0==1'b0&&s1==1'b0)
y=i0;
else if(s0==1'b0&&s1==1'b1)
y=i1;
else if(s0==1'b1&&s1==1'b0)
y=i2;
else
y=i3;
end
endmodule
```

RESULT:

Thus, the Verilog codes for half adder, full adder and multiplexer were written and simulated.

EX. NO: 13 SIMULATION OF SEQUENTIAL CIRCUITS USING VERILOG HDL

DATE:

AIM:

To write Verilog codes for RS, D, JK Flip Flop and up down counter and simulate them

TOOLS REQUIRED:

Xilinx 9.2

PROGRAM:

```
module sr1(clk, s, r, q);
    input clk;
    input s;
    input r;
    output reg q;
    always@(posedge clk or s or r)
    begin
    if(clk==1)
    if(s==1 && r==0)
    q<=1;
    else if(s==0 && r==0)
    q<=q;
    else
    q<=0;
    end
endmodule
```

```
module dff1(d, clk, rst, q);
    input d;
    input clk;
    input rst;
    output q;
    reg q;
    always@(posedge clk)
    if(rst)
    q<=0;
    else
    q<=d;
endmodule
```

```
module jkflip(clk, j, k, q);
    input clk;
    input j;
    input k;
    output reg q;
    always@(posedge clk or j or k)
    begin
    if(clk==1)
    if(j==1 && k==0)
    q<=1;
    else if(j==0 && k==0)
    q<=q;
    else if(j==0 && k==1)
    q<=0;
    else
    q<=~q;
    end
endmodule
```

```
module upcount(clk, N, a);
    input clk;
    input [3:0] N;
    output reg [3:0] a;
    initial a=4'b0000;
    always @(negedge clk)
    a=(a==N)?4'b0000: a+1'b1;
endmodule
```

RESULT:

Thus, the Verilog codes for RS, D, JK flip flop and up counter were written and simulated.

CONTENT BEYOND THE SYLLABUS

DESIGN OF TRAFFIC LIGHT CONTROLLER USING VERILOG HDL

DATE:

AIM:

To write a Verilog code for design of a traffic light controller and simulate it

TOOLS REQUIRED:

Xilinx 9.2

PROGRAM:

```
module traffic_light(clk, light);
input  clk;
output [0:2]  light;      reg  [0:2]  light;
parameter  S0=0, S1=1, S2=2;
parameter  RED=3'b100, GREEN=3'b010,
YELLOW=3'b001;
reg [0:1]  state;
always  @ (posedgeclk)
case  (state)
S0:  begin                    // S0 means RED
light  <=  YELLOW;
state <=  S1;
end
S1:  begin                  // S1 means YELLOW
light  <=  GREEN;
state <=  S2;
end
S2:  begin                  // S2 means GREEN
light  <=  RED;
state <=  S0;
end
default:  begin
light  <=  RED;
state <=  S0;
end
endcase
endmodule
```

RESULT:

Thus, the Verilog code for traffic light was written and simulated.