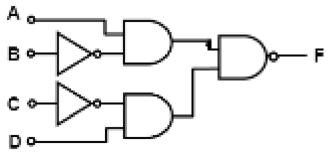
SAVEETHA SCHOOL OF ENGINEERING

SAVEETHA INSTITUTE OF MEDICAL AND TECHNICAL SCIENCES
INSTITUTE OF ELECTRONICS AND COMMUNICATION ENGINEERING
ECA47 – Principles of Digital System Design

Engineer to Excel

LIST OF EXPERIMENTS

1.	Assume the input to the logic circuit be A, B. Verify Demorgan's law using basic gates with the help of Truth table.	HARDWARE	1	2
2.	Verify the truth table of AND, OR, NOT, NAND, NOR and EXOR Logic Gates using Digital Trainer Kit	HARDWARE	1	2
3.	Realize the Boolean Theorems using Logic gates and verify its truth table	HARDWARE	1	3
4.	3. A logic gate circuit with Three inputs A, B and C with output F as shown a) Write the logic expression for output F  b) Determine the output of all four logic gates in the circuit c) Complete the truth table of the function F	HARDWARE	1	4
5.	Realize the Boolean Expression AB + BC + AC using Logic gates and verify its truth table.		1	3
6.	Design and implement any one code converter (Binary to Gray and Gray to Binary) and verify its truth table.	HARDWARE	2	4
7.	Design and implement Half and Full adder and implement the same with logic gates and verify its truth table.	HARDWARE	2	4
8.	Design and implement Half and Full Subtractor and implement the same with logic gates and verify its truth table.	HARDWARE	2	4
9.	Implement 4 to 1 Multiplexer and 1 to 4 De-Multiplexer using Logic gates and verify its truth table	HARDWARE	2	4
10.	Design and Implement 2 to 4 Decoder and 4 to Encoder. verify its truth table.	HARDWARE	2	4
11.	Design and Implement D and SR Flip Flop using Logic gates and also verify their truth table	HARDWARE	3	4
12.	Design and Implement T and JK Flip Flop using Logic gates and also verify their truth table.	HARDWARE	3	4
13.	Design and Implement 2 Bit Up Counter using D flipflops and verify with its truth table.	HARDWARE	3	4
14.	Design a circuit which two inputs and which counts the numbers in reverse order and implement the same using appropriate Flipflops	HARDWARE	3	4
15.	Implement SISO and SIPO shift registers using D flipflop and verify the same with the truth table.	HARDWARE	4	3
16.	Write a Verilog program for all Logic gates and verify its truth table	HARDWARE	4	3
17.	Design and implement the truth table of Half Adder and Full Adder using Modelsim Software	HARDWARE	4	4
18.	Design and implement the truth table of Half Subtractor and Full Subtractor using Modelsim Software	HARDWARE	4	4
19.	Design and implement the truth table of 4 to 1 Multiplexer & 1 to 4 De-Multiplexer using Modelsim Software	SIMULATION	5	1
20.	Design and implement the truth table of 2 to 4 Decoder & 4 to 2 Encoder using Modelsim Software	SIMULATION	5	4

INDEX

NAME OF THE STUDENT :

DEPARTMENT :

ROLL NO : REGISTER NO. :

LAB INCHARGE :

S. No	DATE	NAME OF THE EXPERIMENT	DATE OF SUBMISSION	PAGE NO	MARK	STAFF SIGN
1						
2						
3						
4						
5						
6						
7						
8						
9						
10						
11						
12						
13						
14						
15						
16						
17						
18						
19						
20						

EXP NO:1 Assume the input to the logic circuit be A, B. Verify Demorgan’s law using basic gates with the help of Truth table. **DATE :**

AIM:

To verify Demorgan’s Law using 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	BREAD BOARD	-	1
5.	CONNECTING WIRES	-	10

THEORY:

Boolean algebra has postulates and identities. We can often use these laws to reduce expressions or put expressions in to a more desirable form. One of these laws is the De- Morgan's law

De-Morgan's law has two conditions, or conversely, there are two laws called De-Morgan's Laws.

First Condition or First law:

The compliment of the product of two variables is equal to the sum of the compliment of eachvariable. Thus according to De-Morgan's laws or De-Morgan's theorem if A and B are the two variablesor Boolean numbers. Then accordingly

(A.B)’ = A’ + B’

Second Condition or Second law:

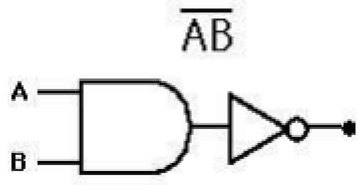
The compliment of the sum of two variables is equal to the product of the compliment of eachvariable.

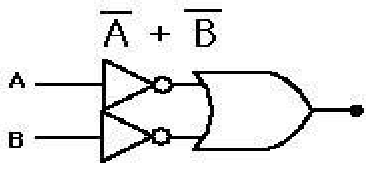
Thus according to De Morgan’s theorem if A and B are the two variables then.

(A+B)’ = A’ . B’

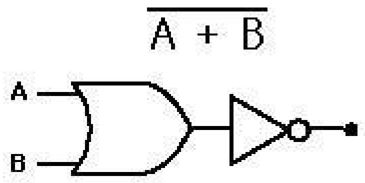
LOGIC DIAGRAM:

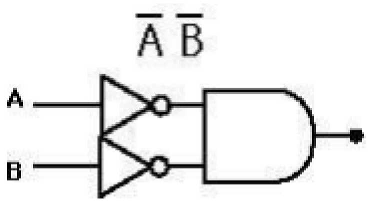
(A.B)’ = A’ + B’





(A+B)’ = A’ . B’





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.

TRUTH TABLE:

A	B	(A.B)’	A’ . B’	(A+B)’	A’ + B’
0	0				
0	1				
1	0				
1	1				

RESULT:

The Demorgan’s Theorem is verified using truth table.

EXP NO:2 Verify the truth table of AND, OR, NOT, NAND, NOR and EXOR Logic Gates using Digital Trainer Kit **DATE :**

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
8.	IC TRAINER KIT	-	1
9.	CONNECTING WIRES		

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 and NOR 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:

(iv) Connections are given as per circuit diagram.

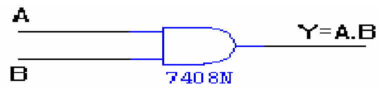
(v) Logical inputs are given as per circuit diagram.

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

AND GATE:

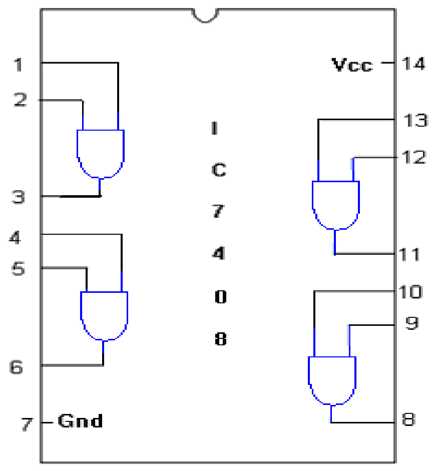
SYMBOL:

PIN DIAGRAM:



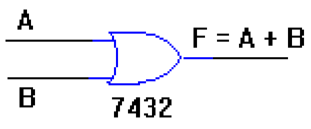
TRUTH TABLE

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

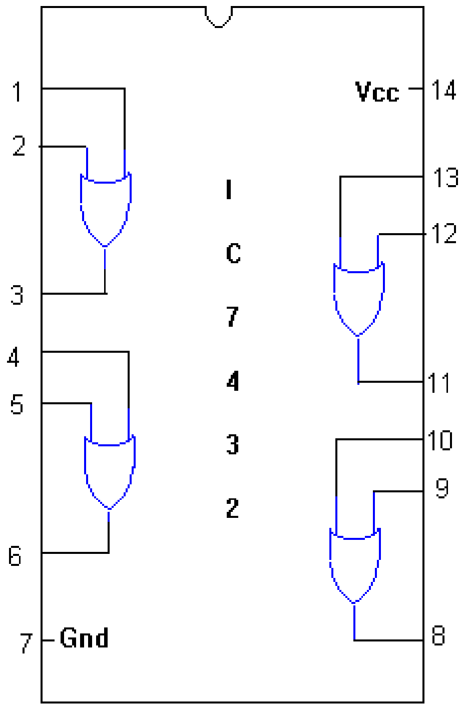


OR GATE:

SYMBOL :



PIN DIAGRAM :

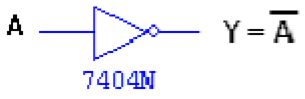


TRUTH TABLE

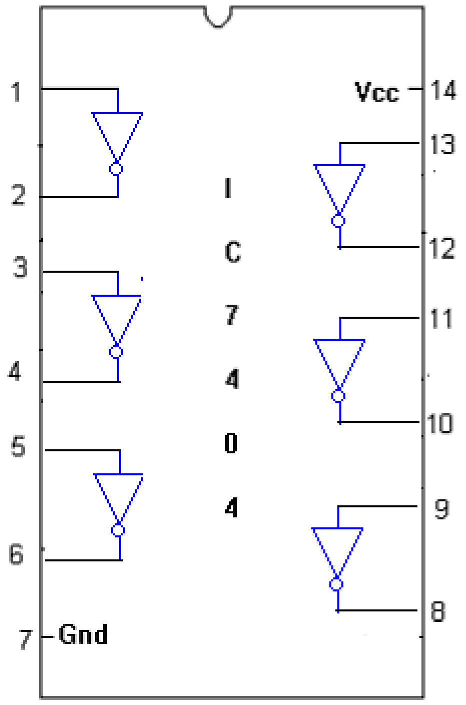
A	B	A+B
0	0	0
0	1	1
1	0	1
1	1	1

NOT GATE:

SYMBOL:



PIN DIAGRAM:

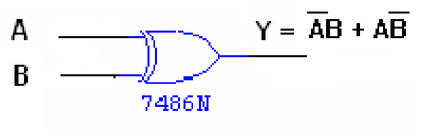


TRUTH TABLE :

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

X-OR GATE :

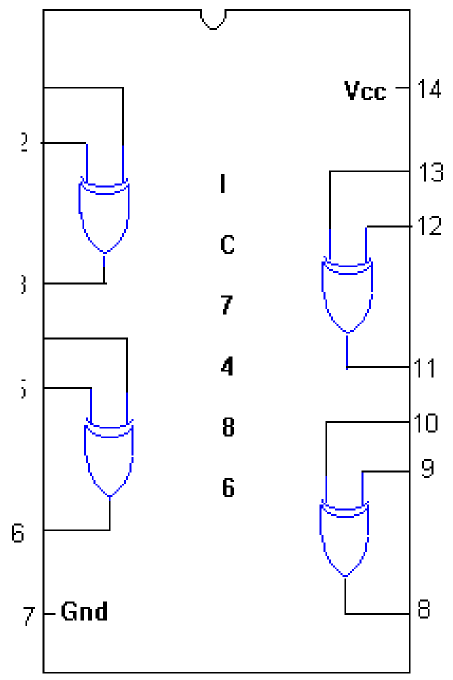
SYMBOL :



TRUTH TABLE :

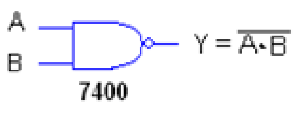
A	B	$\overline{A}B + A\overline{B}$
0	0	0
0	1	1
1	0	1
1	1	0

PIN DIAGRAM :



2-INPUT NAND GATE:

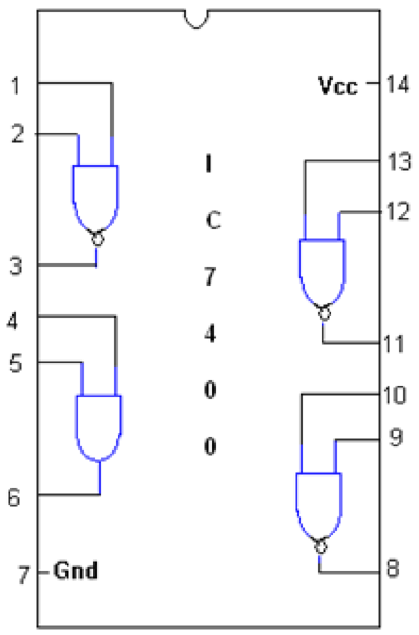
SYMBOL:



TRUTH TABLE

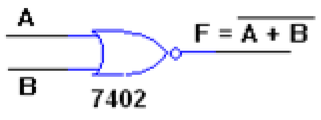
A	B	$\overline{A\cdot B}$
0	0	1
0	1	1
1	0	1
1	1	0

PIN DIAGRAM:



NOR GATE:

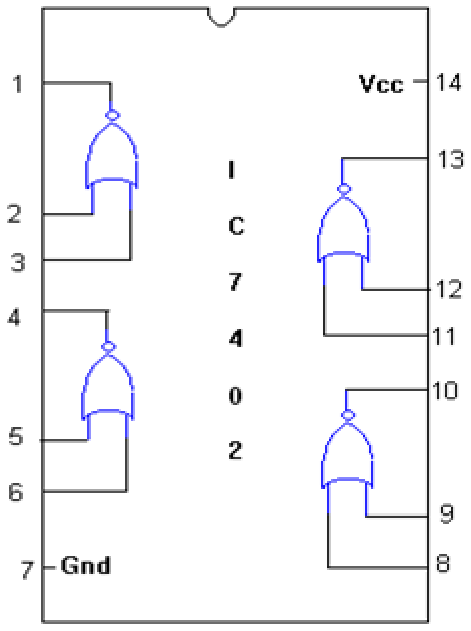
SYMBOL :



TRUTH TABLE

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

PIN DIAGRAM :



RESULT:

Thus the gates AND, OR, NOT, NAND, NOR, XOR and XNOR GATES are studied and verified using Truth table.

EXP NO:3 Realize the Boolean Theorems using Logic gates and verify its truth table

DATE :

AIM:

To verify i) Commutative Law ii) Distributive Law iii) Identity Law iv) Annulment Law v) Absoprption Law using 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	BREAD BOARD	-	1
5.	CONNECTING WIRES	-	10

THEORY:

Boolean algebra has postulates and identities. We can often use these laws to reduce expressions or put expressions in to a more desirable form.

i) Commutative Law

A + B = B+ A

ii) Distributive Law

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

iii) Identity Law

1 . A = A; 0 + A = A;

iv) Annulment Law

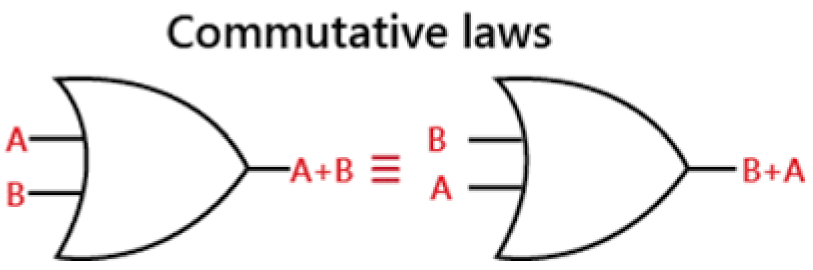
A . 0 = A

v) Absorption Law

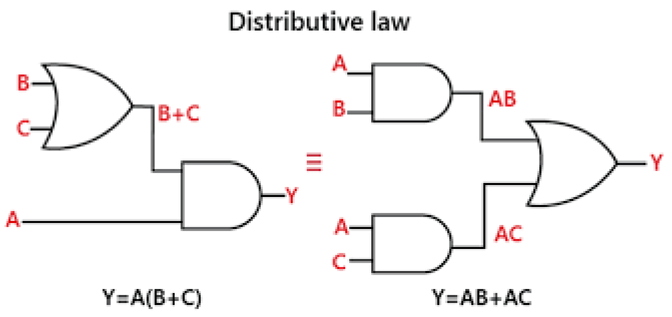
X + (X . Y) = X

LOGIC DIAGRAM:

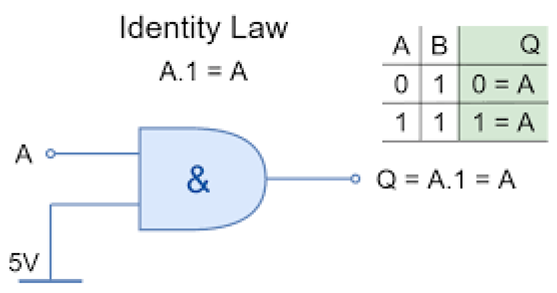
i) Commutative Law



ii) Distributive Law

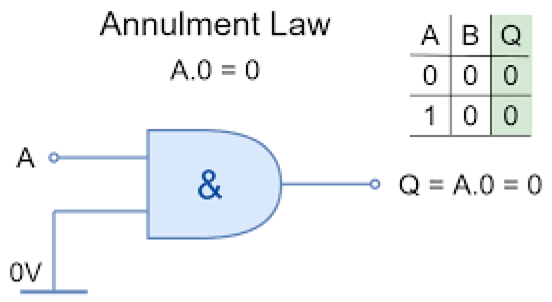


iii) Identity Law



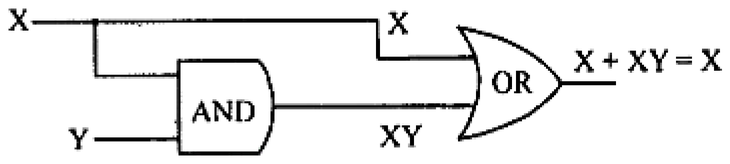
A	B	Q
0	1	0 = A
1	1	1 = A

iv) Annulment Law



A	B	Q
0	0	0
1	0	0

v) Absorption Law



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.

TRUTH TABLE:

A	B	A + B	B + A	A.(B + C)	AB + AC	1 . A	0 + A	A . 0
0	0							
0	1							
1	0							
1	1							

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

RESULT:

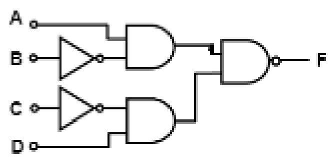
The Boolean Theorems are verified using truth table.

EXP NO:4 Identification of Boolean Expression using Logic Diagram and Verification of Truth Table

DATE :

AIM:

A logic gate circuit with Three inputs A, B and C with output F as shown



a) To Write the logic expression for output F

b) To determine the output of all four logic gates in the circuit

c) To complete the truth table of the function F

APPARATUS REQUIRED:

SL No.	COMPONENT	SPECIFICATION	QTY
1.	AND GATE	IC 7408	1
2.	NAND GATE	IC 7400	1
3.	NOT GATE	IC 7404	1
4	BREAD BOARD	-	1
5.	CONNECTING WIRES	-	10

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.

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.

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.

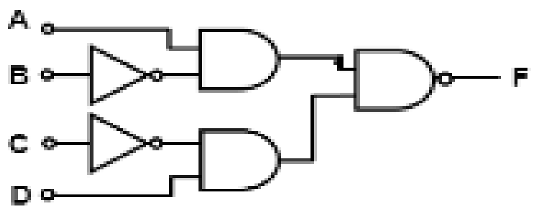
PROCEDURE:

(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.

LOGIC DIAGRAM:



F = ((A.B') . (C'.D))'

TRUTH TABLE:

A	**B**	**C**	**D**	**(A.B')**	**(C'.D)**	**((A.B') . (C'.D))'**
0	0	0				
0	0	1				
0	1	0				
0	1	1				
1	0	0				
1	0	1				
1	1	0				
1	1	1				

RESULT:

The Boolean expression is identified and the truth table is verified.

EXP NO:5 Realize the Boolean Expression F = AB+BC+AC using Logic gates and verify its truth table.

DATE :

AIM:

To realize the Boolean expression and verify its truth tables.

APPARATUS REQUIRED:

SL No.	COMPONENT	SPECIFICATION	QTY
1.	AND GATE	IC 7408	1
2.	OR GATE	IC 7432	1
3.	BREAD BOARD	-	1
4.	CONNECTING WIRES	-	10

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.

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.

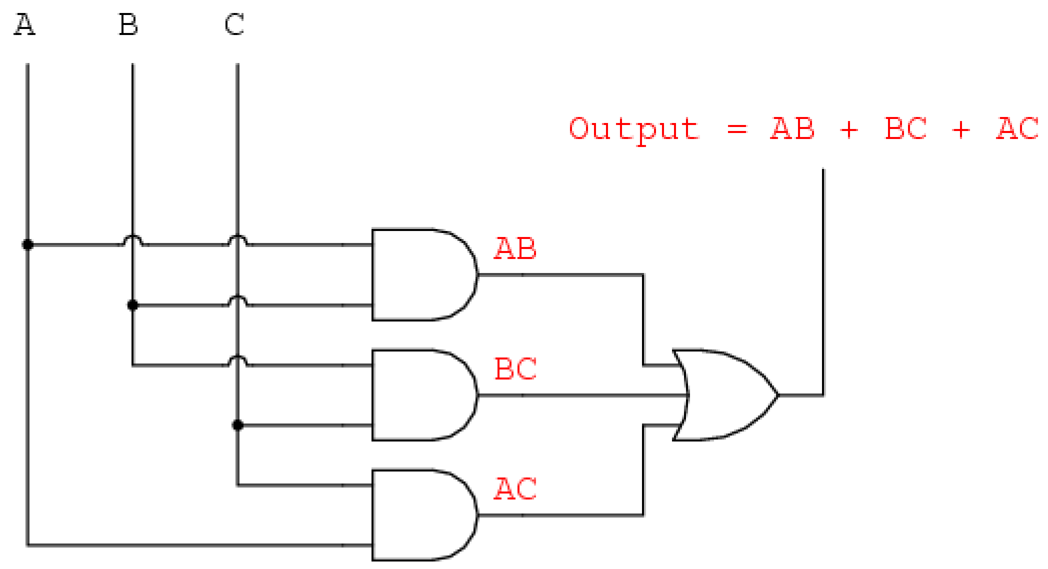
PROCEDURE:

(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.

LOGIC DIAGRAM:



TRUTH TABLE:

TRUTH TABLE:

A	**B**	**C**	**AB**	**BC**	**AC**	**F**
0	0	0				
0	0	1				
0	1	0				
0	1	1				
1	0	0				
1	0	1				
1	1	0				
1	1	1				

RESULT:

The Boolean expression is realized using logic gates and the truth table is verified.

EXP NO:6 Design and implement any one code converter (Binary to Gray and Gray to Binary) and verify its truth table. **DATE :**

AIM:

To design and implement 4-bit Binary to gray code converter.

APPARATUS REQUIRED:

S.No.	COMPONENT	SPECIFICATION	QTY.
1.	X-OR GATE	IC 7486	1
2.	IC TRAINER KIT	-	1
3.	CONNECTING WIRES	-	10

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. Since each code uses four bits to represent a decimal digit, there are four inputs and four outputs.

Binary to GRAY Converter:

By representing the ten decimal digits with a four bit gray code, we have another form of BCD code. The gray code however can be extended to any number of bits and conversion between

binary code and gray code is sometimes useful. The following rules apply for conversion:

1. The MSB in the gray code is the same as the corresponding bit in the binary number.

2. Going from left to right, add each adjacent pair of binary bits to get the next gray code bit. Discard carries.

PROCEDURE:

(i) Give the connections as per the logic diagram (Binary to Gray).

(ii) Switch on the power supply.

(iii) Apply the binary inputs at the appropriate terminal and observe the corresponding output.

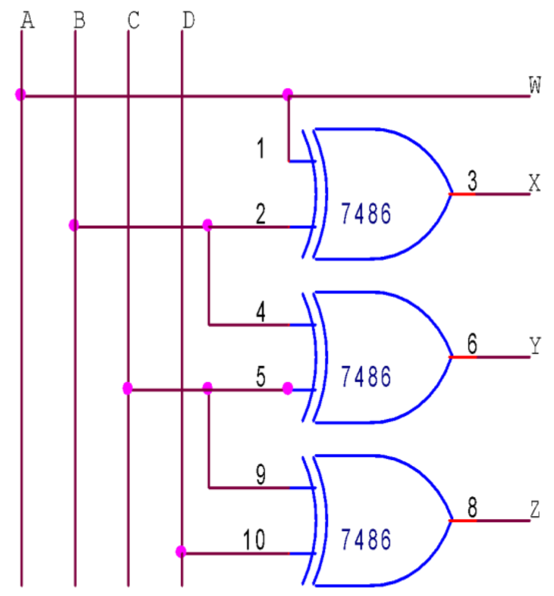
(iv) Verify the truth table and repeat the above procedure for other converters.

TRUTHTABLE: (Binary to Gray) **TRUTH TABLE:(Gray to Binary)**

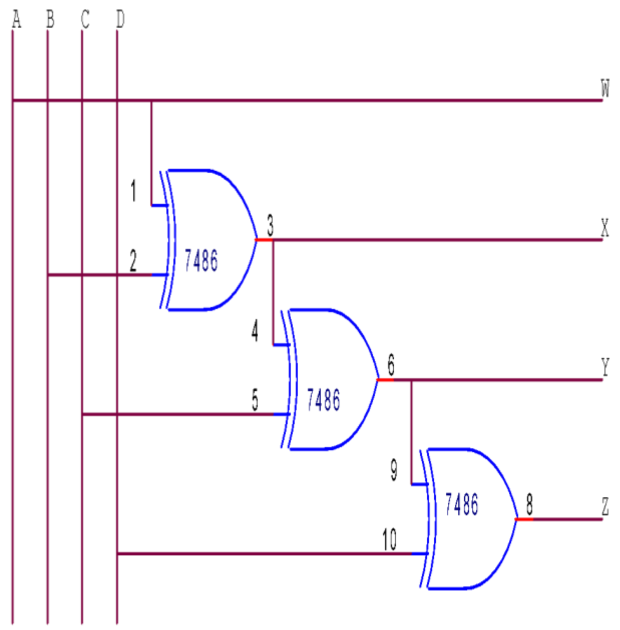
BINARYCODE				GRAYCODE			
A	B	C	D	W	X	Y	Z
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	1
0	0	1	1	0	0	1	0
0	1	0	0	0	1	1	0
0	1	0	1	0	1	1	1
0	1	1	0	0	1	0	1
0	1	1	1	0	1	0	0
1	0	0	0	1	1	0	0
1	0	0	1	1	1	0	1
1	0	1	0	1	1	1	1
1	0	1	1	1	1	1	0
1	1	0	0	1	0	1	0
1	1	0	1	1	0	1	1
1	1	1	0	1	0	0	1
1	1	1	1	1	0	0	0

GRAYCODE				BINARYCODE			
A	B	C	D	W	X	Y	Z
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

CIRCUIT DIAGRAM : (Binary to Gray)



CIRCUIT DIAGRAM : (Gray to Binary)



RESULT:

Thus, code converters are studied and verified.

EXP NO:7 Design and implement Half and Full adder and implement the same with logic gates and verify its truth table. **DATE :**

AIM:

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

APPARATUS REQUIRED:

S.No.	COMPONENT	SPECIFICATION	QTY.
1.	AND GATE	IC 7408	1
2.	X-OR GATE	IC 7486	1
3.	NOT GATE	IC 7404	1
4.	OR GATE	IC 7432	1
5.	IC TRAINER KIT	-	1
6.	CONNECTING WIRES	-	few

THEORY:

Half adder:

The logic circuit for the addition of two one bit numbers is referred to as a half adder. Here A and B are the two inputs and S (sum) and C (carry) are two outputs. The two outputs, the sum and carry equations are carried from the truth table of addition of two numbers. They are

$S = A'B + AB'$ ---------- (1)

$C = AB$ ------------- (2)

Full adder:

A half adder has only two inputs and there is no provision to add a carry coming from the lower order bits when multibit additions are performed. For this purpose, a third input terminal is added and this circuit is used to add A, B and C, where A_n and B_n are the nth order bits of the numbers respectively and C is the carry generated from the addition of bits. This circuit is referred to as full adder. The sum and carry equations are carried from the truth table of addition of three numbers. They are:

$S = A \oplus B \oplus C$ ---------------- (3) **$C = AB + C(A \oplus B)$** ---------------- (4)

PROCEDURE:

(i) Connections are given as per circuit diagram.

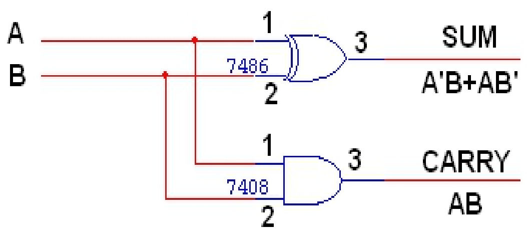
(ii) Logical inputs are given as per circuit diagram.

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

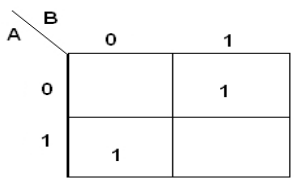
LOGIC DIAGRAM:

HALF ADDER TRUTH TABLE:

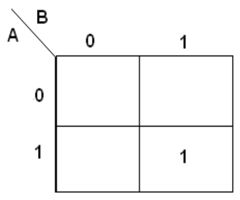
A	B	CARRY	SUM
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0



K-Map for SUM:



K-Map for CARRY:



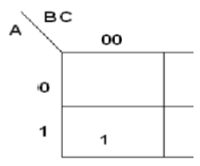
SUM = A’B + AB’

=A ⊕ B

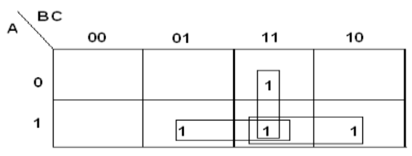
CARRY = AB

FULL ADDER

K-Map for SUM:



K-Map for CARRY:



SUM = A’B’C + A’BC’ + ABC’ + ABC

= A’ (B’C+BC’) + A(BC’+BC)

= A’(B ⊕ C) + A(B ⊕ C)

= A ⊕ B ⊕ C

CARRY = A’BC+ABC+ABC’+AB’C

= AB+A’BC+AB’C

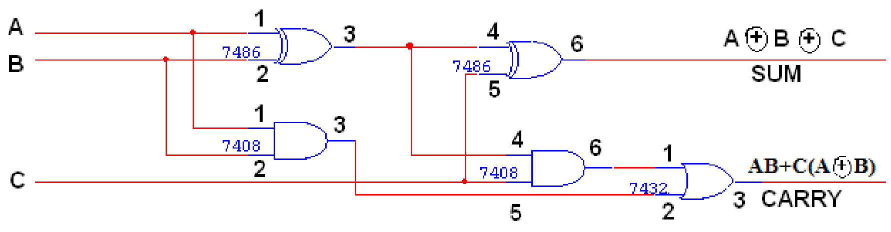
= AB+C(A ⊕ B)

(or) from K-Map= AB+BC+AC

TRUTH TABLE:

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

FULL ADDER USING TWO HALF ADDER



.

RESULT:

Thus , Half adder, full adder are studied and verified.

EXP NO:8 Design and implement Half and Full Subtractor and implement the same with logic gates and verify its truth table. **DATE :**

AIM:

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

APPARATUS REQUIRED:

S.No.	COMPONENT	SPECIFICATION	QTY.
1.	AND GATE	IC 7408	1
2.	X-OR GATE	IC 7486	1
3.	NOT GATE	IC 7404	1
4.	OR GATE	IC 7432	1
5.	IC TRAINER KIT	-	1
6.	CONNECTING WIRES	-	few

THEORY:

Half subtractor:

A logic circuit for the subtracting the one bit subtrahend (B) from one bit minuend (A) is referred as a half subtractor. The two outputs, the difference (D) and borrow (B) equations are given below

D = A’B + AB’ ------------- (5) **B =A’B** ------------- (6)

Full subtractor:

Just like a full adder circuit, full subtractor circuit performs multiple subtraction. A full subtractor will have three inputs, A (minuend), B (subtrahend) and C (borrow from previous stage) and two outputs D (difference) and C (borrow). The difference and carry equations are shown below.

D = A ⊕ B ⊕ C -------------- (7) **B = A’ B + C (A⊕ B)’** ------- (8)

PROCEDURE:

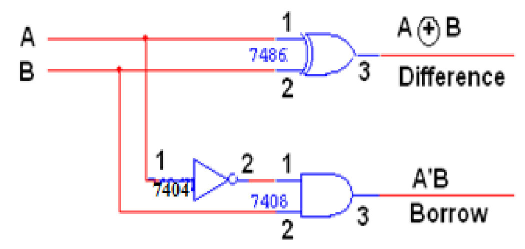
(i) Connections are given as per circuit diagram.

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

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

HALF SUBTRACTOR TRUTH TABLE:

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



K-Map for DIFFERENCE:

A \ B	0	1
0		1
1	1	

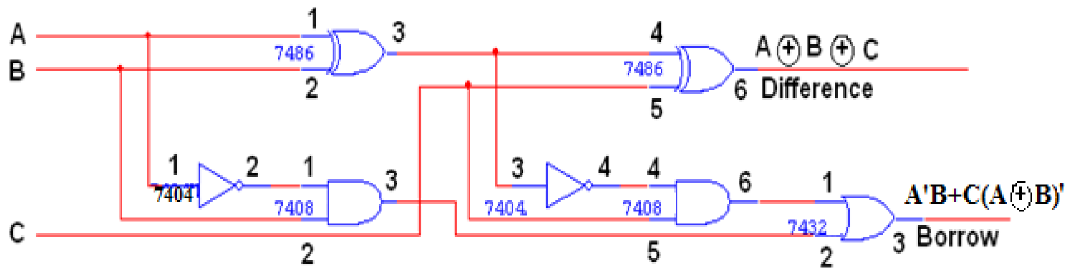
K-Map for BORROW:

A \ B	0	1
0		1
1		

DIFFERENCE = A'B + AB'

BORROW = A'B

FULL SUBTRACTOR USING TWO HALF SUBTRACTOR:



TRUTH TABLE:

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

K-Map for Difference:

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

K-Map for Borrow:

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

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

$= A'(B'C+BC') + A(B'C'+BC)$

$= A'(B\oplus C)+A(B\oplus C)$

$= A\oplus B\oplus C.$

Borrow = $A'B'C + A'BC' + A'BC + ABC$

$= A'B + C(A'B' + AB)$

$= A'B + C(A \oplus B)'$

(OR) from K-Map: Borrow = $A'B + BC + A'C$

RESULT:

Thus half subtractor and full subtractor are studied and verified.

EXP NO:9 Implement 4 to 1 Multiplexer using Logic gates and verify its truth table.

DATE :

AIM:

To design and implement multiplexer and demultiplexer using logic gates.

APPARATUS REQUIRED:

S.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.	IC TRAINER KIT	-	1
5.	CONNECTING WIRES	-	10

THEORY:

MULTIPLEXER:

Multiplexer means transmitting a large number of information units from many users over a single channel. 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 to be selected.

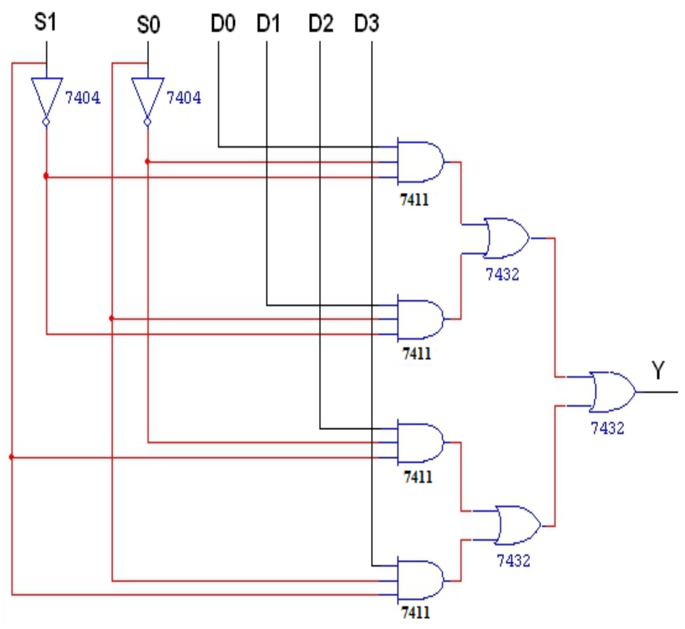
PROCEDURE:

(i) Connections are given as per circuit diagram.

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

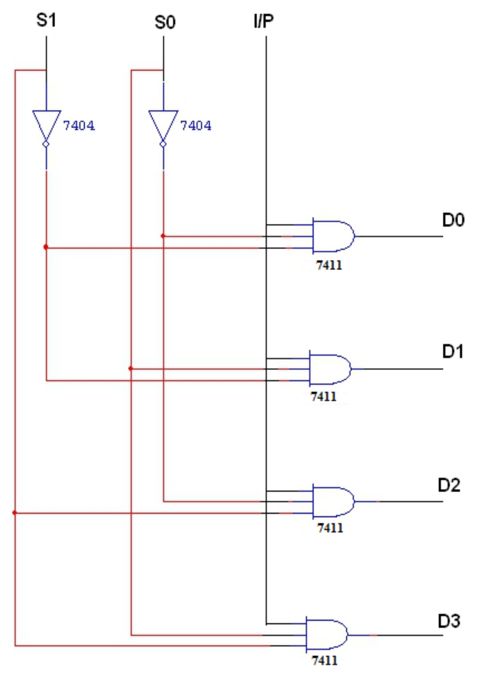
(iii) Observe the output and verify the truth table.

LOGIC DIAGRAM:



MULTIPLEXER TRUTH TABLE:

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



DEMULTIPLEXER TRUTH TABLE:

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

RESULT:

Thus, Multiplexer(MUX) and De- Multiplexer(DEMUX) are studied and verified.

EXP NO:10 Design and Implement 2 to 4 Decoder and 4 to 2 Encoder. verify its truth table.

DATE :

AIM:

To design and implement encoder and decoder using logic gates.

APPARATUS REQUIRED:

S.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.	IC TRAINER KIT	-	1
5.	CONNECTING WIRES	-	10

THEORY:

The combinational circuits that modify the binary data into N output lines are known as Encoders. The combinational circuits that convert the binary data into 2N output lines are called Decoders.

PROCEDURE:

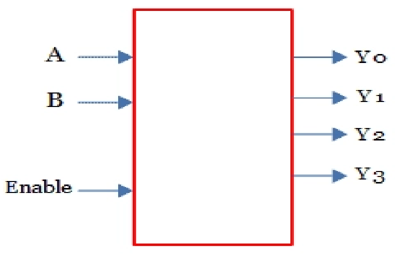
(i) Connections are given as per circuit diagram.

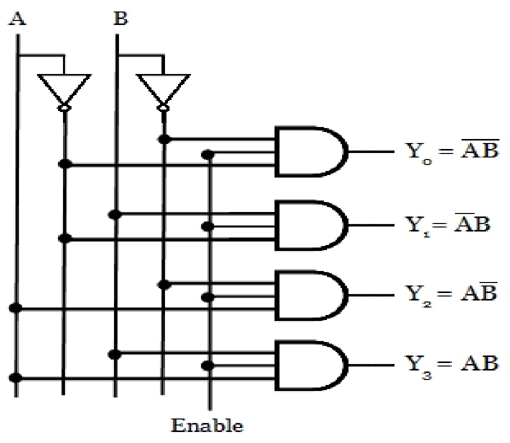
(ii) Logical inputs are given as per circuit diagram.

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

LOGIC DIAGRAM

DECODER





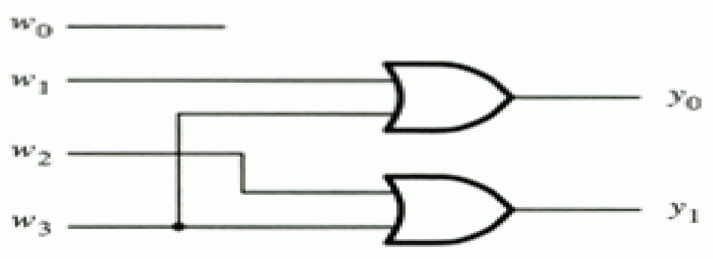
TRUTH TABLE:

Inputs			Outputs			
EN	A	B	Y_3	Y_2	Y_1	Y_0
0	×	×	0	0	0	0
1	0	0	0	0	0	1
1	0	1	0	0	1	0
1	1	0	0	1	0	0
1	1	1	1	0	0	0

ENCODER

w_3	w_2	w_1	w_0	y_1	y_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1

(a) Truth table



(b) Circuit

RESULT:

Thus, Encoder and Decoder are studied and verified.

EXP NO:11 Design and Implement SR and D Flip Flop using Logic gates and also verify their truth table

DATE :

AIM:

T0 design and implement SR and D Flip Flop using Logic gates and also verify their truth table.

APPARATUS REQUIRED:

S.No.	COMPONENT	SPECIFICATION	QTY
1.	NAND GATE	IC 7400	1
2.	IC TRAINER KIT	-	1
3.	CONNECTING WIRES	-	10

THEORY:

In digital circuits, a Flip-Flop is a term referring to an electronic circuit (a bistable multivibrator) that has two stable states and thereby is
capable of serving as one bit of memory. A flip-flop is usually controlled by one or
two control signals and /or a gate or clock signal. The output often includes the complement as well as the normal output.

SR Flip-Flop:

The fundamental latch is the simple SR flip-flop, where S and R stand for set and reset respectively. It can be constructed from a pair of cross-coupled NOR logic gates. The
stored bit is present on the output marked Q.

Normally, in storage mode, the S and R inputs are both low, and feedback maintains the outputs in a constant state, with Q and the complement of Q. If S (Set) is given with high while R is held low, then the Q output is forced high, and stays high even after S returns
low; similarly, if R (Reset) is given with high while S is held low, then the Q output is forced low, and stays low even after R returns low.

JK-Flip-Flop:

The JK flip-flop augments the behavior of the SR flip-flop (J = Set, K = Reset) by interpreting the S = R = 1 condition as a “flip“ or toggle command. Specifically, the combination J = 1, K = 0 is a command to set the flip-flop; the combination J = 0, K = 1 is a command to reset the flip-flop; and the combination J = K = 1 is a command to toggle the flip-flop, i.e., change
its output to the logical complement of its current value.

D-Flip-Flop:

The Q output always takes on the state of the D input at the moment of a rising clock edge. (or falling edge if the clock input is active low) It is called the D flip-flop for this reason, since the output takes the value of the D input or Data input, and Delays it by one clock count. The D flip-flop can be interpreted as a primitive memory cell, zero-order hold, or delay line.

T-Flip-Flop:

If the T input is high, the T flip-flop changes state (“toggles“) whenever the clock input is strobed. If the T input is low, the flip-flop holds the previous value. This behavior is described by the characteristic equation: A T flip-flop can also be built using a JK flip-flop (J & K pins are connected together and act as T) or D flip-flop.

PROCEDURE:

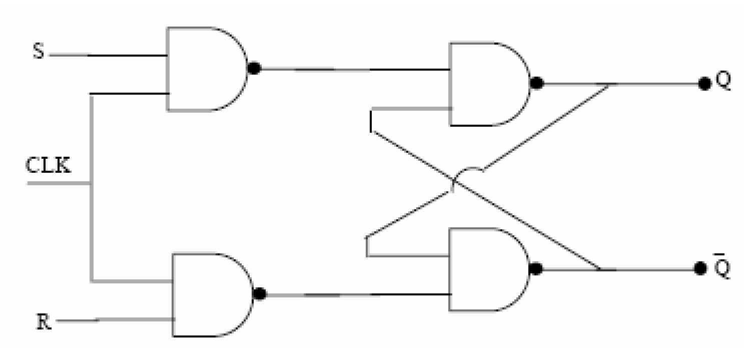
(i) Connections are given as per circuit diagram.

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

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

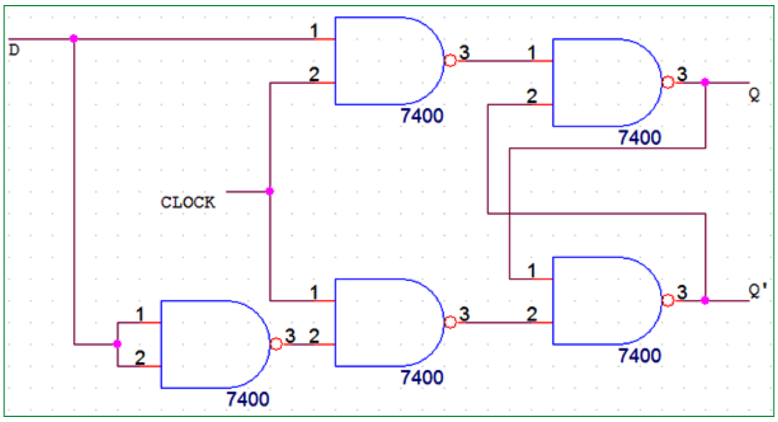
LOGIC DIAGRAM:

SR FLIPFLOP



Inputs			Outputs
R	S	CLK	Q
0	0	↑	Q_0
0	1	↑	1
1	0	↑	0
1	1	↑	Indeterminate state

D FLIPFLOP



Outputs		
D	CLK	Q
0	↑	0
1	↑	1

RESULT:

The sequential circuit was implemented and truth table is verified.

EXP NO:11 Design and Implement SR and D Flip Flop using Logic gates and also verify their truth table

DATE :

AIM:

T0 design and implement SR and D Flip Flop using Logic gates and also verify their truth table.

APPARATUS REQUIRED:

S.No.	COMPONENT	SPECIFICATION	QTY
4.	NAND GATE	IC 7400	1
5.	IC TRAINER KIT	-	1
6.	CONNECTING WIRES	-	10

THEORY:

In digital circuits, a Flip-Flop is a term referring to an electronic circuit (a bistable multivibrator) that has two stable states and thereby is capable of serving as one bit of memory. A flip-flop is usually controlled by one or two control signals and /or a gate or clock signal. The output often includes the complement as well as the normal output.

SR Flip-Flop:

The fundamental latch is the simple SR flip-flop, where S and R stand for set and reset respectively. It can be constructed from a pair of cross-coupled NOR logic gates. The stored bit is present on the output marked Q.

Normally, in storage mode, the S and R inputs are both low, and feedback maintains the outputs in a constant state, with Q and the complement of Q. If S (Set) is given with high while R is held low, then the Q output is forced high, and stays high even after S returns low; similarly, if R (Reset) is given with high while S is held low, then the Q output is forced low, and stays low even after R returns low.

D-Flip-Flop:

The Q output always takes on the state of the D input at the moment of a rising clock edge. (or falling edge if the clock input is active low) It is called the D flip-flop for this reason, since the output takes the value of the D input or Data input, and Delays it by one clock count. The D flip-flop can be interpreted as a primitive memory cell, zero-order hold, or delay line.

PROCEDURE:

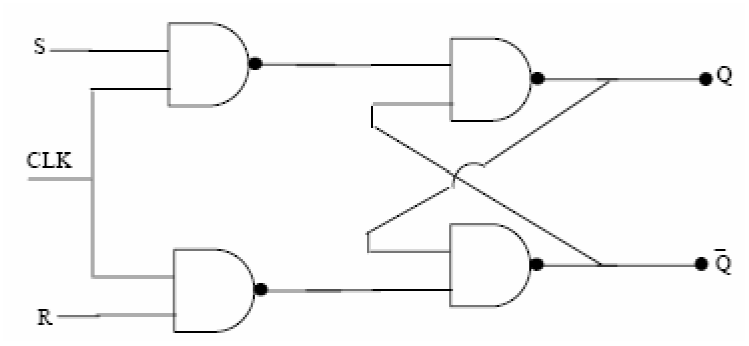
(iv) Connections are given as per circuit diagram.

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

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

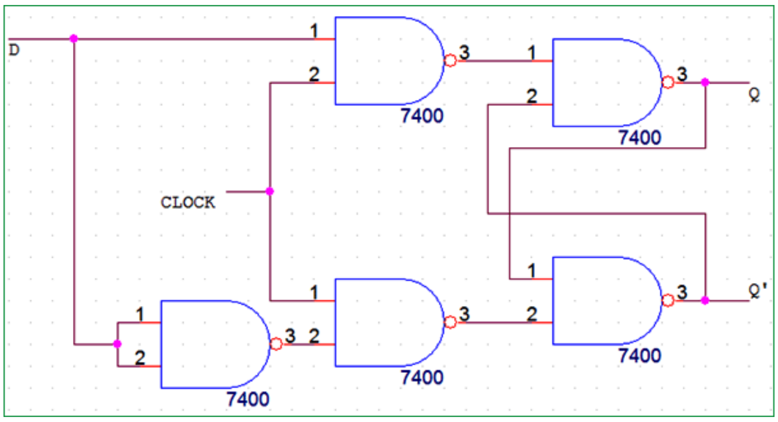
LOGIC DIAGRAM:

SR FLIPFLOP



Inputs			Outputs
R	S	CLK	Qt+1
0	0	⬆	$Q_{t(\text{Previous State})}$
0	1	⬆	1
1	0	⬆	0
1	1	⬆	Indeterminate state

D FLIPFLOP



Outputs		
D	CLK	Q
0	⬆	0
1	⬆	1

RESULT:

The sequential circuit was implemented and truth table is verified.

EXP NO:12 Design and Implement JK and T Flip Flop using Logic gates and also verify their truth table

DATE :

AIM:

T0 design and implement JK and T Flip Flop using Logic gates and also verify their truth table.

APPARATUS REQUIRED:

S.No.	COMPONENT	SPECIFICATION	QTY
7.	NAND GATE	IC 7400	1
8.	IC TRAINER KIT	-	1
9.	CONNECTING WIRES	-	10

THEORY:

In digital circuits, a Flip-Flop is a term referring to an electronic circuit (a bistable multivibrator) that has two stable states and thereby is
capable of serving as one bit of memory. A flip-flop is usually controlled by one or
two control signals and /or a gate or clock signal. The output often includes the complement as well as the normal output.

JK-Flip-Flop:

The JK flip-flop augments the behavior of the SR flip-flop (J = Set, K = Reset) by interpreting the S = R = 1 condition as a "flip" or toggle command. Specifically, the combination J = 1, K = 0 is a command to set the flip-flop; the combination J = 0, K = 1 is a command to reset the flip-flop; and the combination J = K = 1 is a command to toggle the flip-flop, i.e., change
its output to the logical complement of its current value.

T-Flip-Flop:

If the T input is high, the T flip-flop changes state ("toggles") whenever the clock input is strobed. If the T input is low, the flip-flop holds the previous value. This behavior is described by the characteristic equation: A T flip-flop can also be built using a JK flip-flop (J & K pins are connected together and act as T) or D flip-flop.

PROCEDURE:

(vii) Connections are given as per circuit diagram.

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

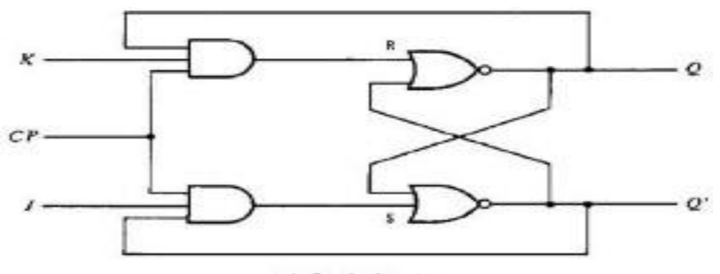
(ix) Observe the output and verify the truth table.

LOGIC DIAGRAM:

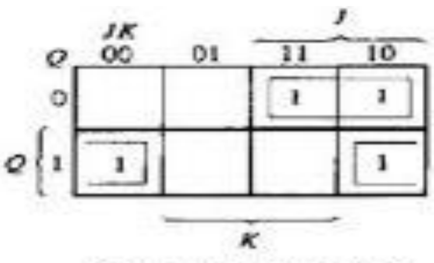
SR FLIPFLOP

Inputs			Outputs
R	S	CLK	Q
0	0	↑	Q_0
0	1	↑	1
1	0	↑	0
1	1	↑	Indeterminate state

JK FLIPFLOP



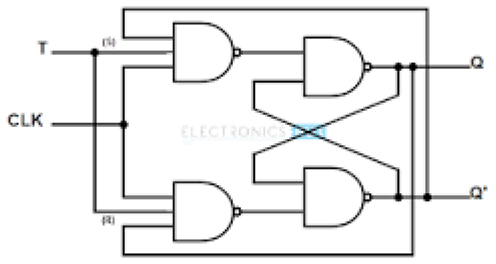
(a) Logic diagram



$Q(t+1) = JQ' + K'Q$

Truth Table			
Q	J	K	Q(t+1)
0	x	x	Qt(Previous State)
1	0	0	Qt(Previous State)
1	0	1	0
1	1	0	1
1	1	1	Qt'(Toggle)

T FLIP FLOP:



Q	T	Q(t+1)
0	x	Qt(Previous State)
1	0	Qt(Previous State)
1	1	Qt'(Toggle)

<u>RESULT:</u>

The sequential circuit was implemented and truth table is verified.

EXP NO:13 Design and Implement 2 Bit Up Counter using D flipflops and verify with its truth table.

DATE :

AIM:

To design and implement 2 Bit Up Counter using D flipflops and verify with its truth table.

APPARATUS REQUIRED:

S.No.	COMPONENT	SPECIFICATION	QTY
1.	D FLIPFLOP	IC 7474	1
2.	IC TRAINER KIT	-	1
3.	CONNECTING WIRES	-	10

THEORY:

In the **synchronous counter**, the same clock pulse is passed to the clock input of all the flip flops. The clock signals produced by all the flip flops are the same as each other. Below is the diagram of a 2-bit synchronous counter in which the inputs of the first flip flop, i.e., FF-A, are set to 1. So, the first flip flop will work as a toggle flip-flop. The output of the first flip flop is passed to both the inputs of the next JK flip flop.

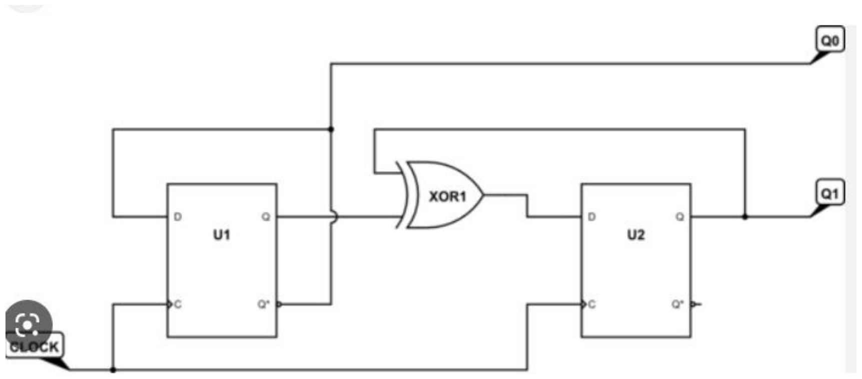
PROCEDURE:

(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.

LOGIC DIAGRAM:



TRUTH TABLE:

CLK	Q_A	Q_B
1	0	0
2	0	1
3	1	0
4	1	1

RESULT:

The sequential circuit was implemented and truth table is verified.

EXP NO:14 Design and Implement 2 Bit Down Counter using D flipflops and verify with its truth table.

DATE :

AIM:

To design and implement 2 Bit down Counter using D flipflops and verify with its truth table.

APPARATUS REQUIRED:

S.No.	COMPONENT	SPECIFICATION	QTY
1.	D FLIPFLOP	IC 7474	1
2.	IC TRAINER KIT	-	1
3.	CONNECTING WIRES	-	10

THEORY:

In the **synchronous counter**, the same clock pulse is passed to the clock input of all the flip flops. The clock signals produced by all the flip flops are the same as each other. Below is the diagram of a 2-bit synchronous counter in which the inputs of the first flip flop, i.e., FF-A, are set to 1. So, the first flip flop will work as a toggle flip-flop. The output of the first flip flop is passed to both the inputs of the next JK flip flop.

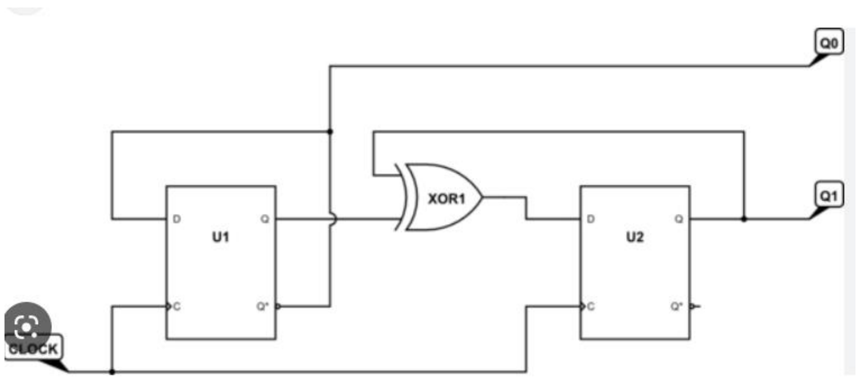
PROCEDURE:

(iv) Connections are given as per circuit diagram.

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

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

LOGIC DIAGRAM:



TRUTH TABLE:

CLK	Q_A	Q_B
1	1	1
2	1	0
3	0	1
4	0	0

RESULT:

The sequential circuit was implemented and truth table is verified.

EXP NO:15 Implement SISO and SIPO shift registers using D flipflop and verify the same with the truth table. **DATE :**

AIM:

To implement SISO and SIPO shift registers using D flipflop and verify the same with the truth table.

APPARATUS REQUIRED:

S.No.	COMPONENT	SPECIFICATION	QTY
1.	D FLIPFLOP	IC 7474	1
2.	IC TRAINER KIT	-	1
3.	CONNECTING WIRES	-	10

THEORY:

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

PROCEDURE:

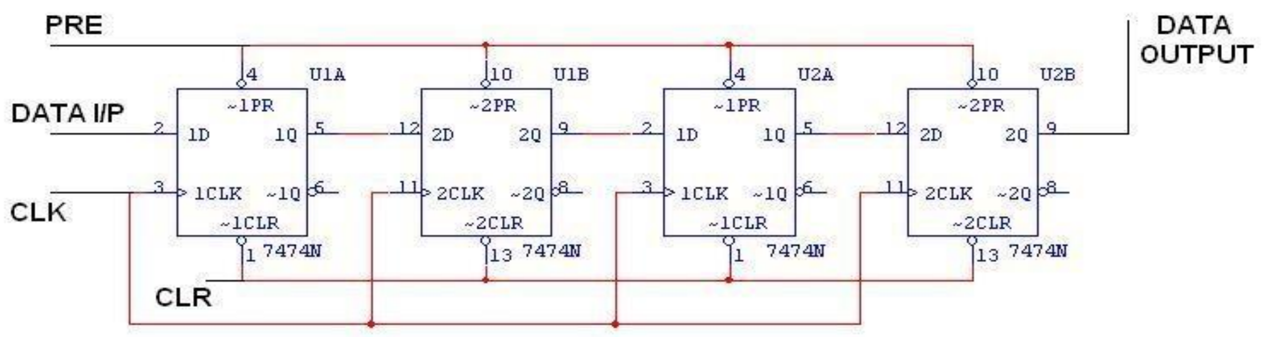
(i) Connections are given as per circuit diagram.

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

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

LOGIC DIAGRAM:

SERIAL IN SERIAL OUT

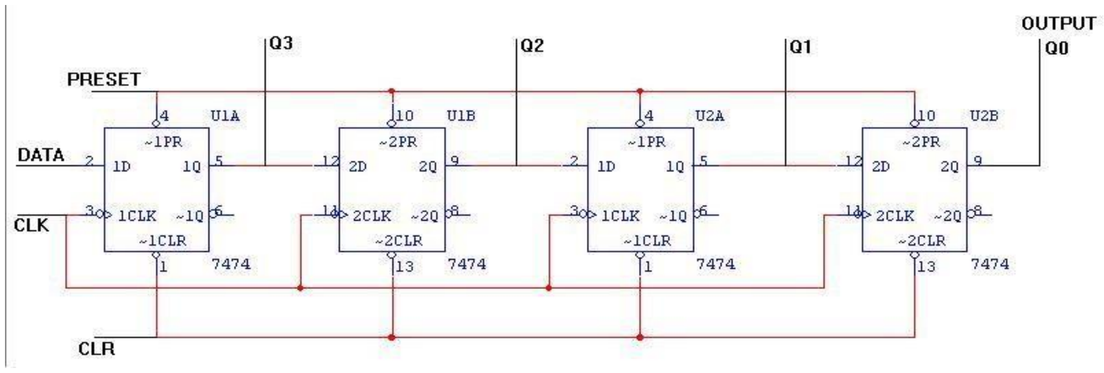


TRUTH TABLE:

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

SERIAL IN PARALLEL OUT

LOGIC DIAGRAM:



TRUTH TABLE:

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

RESULT:

The sequential circuit was implemented and truth table is verified.

EXP NO:16 Write a Verilog program for all Logic gates and verify its truth table

DATE :

AIM:

To write a Verilog program for all Logic gates and verify its truth table.

APPARATUS REQUIRED:

S.No.	COMPONENT	SPECIFICATION	QTY
1.	LAPTOP WITH MODEL SIM SOFTWARE	IC 7411	1

VERILOG CODE:

AND GATE

```
module and12(a,b,c);

input a;

input b;

output c;

assign c = a & b;

endmodule
```

OR GATE

```
module or12(a,b,d);

input a;

input b;

output d;

 assign d = a | b;

endmodule
```

NAND GATE

```
module nand12(a,b,e);

input a;

input b;

output e;

assign e = ~(a & b);

endmodule
```

XOR GATE

```
module xor12(a,b,h);

input a;

 input b;

output h;

assign h = a ^ b;

endmodule
```

XNOR GATE

```
module xnor12(a,b,i);

input a;

input b;

output i;

assign i = ~(a ^ b);

endmodule
```

NOR GATE

```
module nor12(a,b,f);

input a;

input b;

output f;

assign f = ~(a | b);

endmodule
```

NOT GATE

```
module not12(a,g);

input a;

output g;

assign g = ~a;

endmodule
```

RESULT:

The Verilog code was executed using MODEL SIM software and output was verified.

EXP NO:17 Design and implement the truth table of Half Adder and Full Adder using Modelsim Software

DATE :

AIM:

To design and implement the truth table of Half Adder and Full Adder using Modelsim Software

APPARATUS REQUIRED:

S.No.	COMPONENT	SPECIFICATION	QTY
1.	LAPTOP WITH MODEL SIM SOFTWARE	IC 7411	1

VERILOG CODE:

HALF ADDER

```
module hadd(a,b,s,c);

input a;

input b;

output s;

output c;

assign s = a ^ b;

assign c = a & b;

endmodule
```

FULL ADDER

```
module fadd(a,b,c,s,cout);

input a;

input b;

input c; output s;

output cout;

assign s = (a ^ b) ^ c;

assign cout = (a & b)|( b & c)|(c & a);

endmodule
```

RESULT:

The Verilog code was executed using MODEL SIM software and output was verified.

EXP NO:18 Design and implement the truth table of Half Substractor and Full Substractor using Modelsim Software **DATE :**

AIM:

To design and implement the truth table of Half Substractor and Full Substractor using Modelsim Software

APPARATUS REQUIRED:

S.No.	**COMPONENT**	**SPECIFICATION**	**QTY**
1.	LAPTOP WITH MODEL SIM SOFTWARE	IC 7411	1

VERILOG CODE:

HALF SUBSTRACTOR

```
module hsub(a,b,d,bor);

Input a;

Input b;

output d;

output bor;

assign d=)a^b);

assign bor = (~a&~b);

end module
```

FULL SUBSTRACTOR

```
module sub(a,b,c,d,b out);

 input a;

 input b;

 input c;

 output d;

 output bout;

 assign d = (a ^ b) ^ c;

 assign bout = (~a & b)|( b & c)|(c & ~a);

endmodule
```

RESULT:

The Verilog code was executed using MODEL SIM software and output was verified.

EXP NO:19 Design and implement the truth table of 4 to 1 Multiplexer & 1 to 4 De-Multiplexer using Modelsim Software **DATE :**

AIM:

To design and implement the truth table of 4 to 1 Multiplexer & 1 to 4 De-Multiplexer using Modelsim Software

APPARATUS REQUIRED:

S.No.	**COMPONENT**	**SPECIFICATION**	**QTY**
1.	LAPTOP WITH MODEL SIM SOFTWARE	IC 7411	1

VERILOG CODE:

4 TO 1 MULTIPLEXER

```
module mux4to1(Y, I0,I1,I2,I3, sel);
 output Y;
 input I0,I1,I2,I3;
 input [1:0] sel;
 reg Y;
always @ (sel or I0 or I1 or I2 or I3)
case (sel)
2'b00:Y=I0;
2'b01:Y=I1;
2'b10: Y=I2;
2'b11: Y=I3;
endcase
endmodule
```

1 TO 4 DE-MULTIPLEXER

```
module demux(S,D,Y);
Input [1:0] S;
Input D;
Output [3:0] Y; reg Y;
always @(S OR )
case({D,S})
3’b100: Y=4’b0001;
3’b101: Y=4’b0010;
 3’b110: Y=4’b0100;
 3’b111: Y=4’b1000;
```

```
default:Y=4'b0000;

endcase

endmodule
```

RESULT:

The Verilog code was executed using MODEL SIM software and output was verified.

EXP NO:20 Design and implement the truth table of 2 to 4 Decoder & 4 to 2 Encoder using Modelsim Software **DATE :**

AIM:

To design and implement the truth table of 4 to 1 Multiplexer & 1 to 4 De-Multiplexer using Modelsim Software

APPARATUS REQUIRED:

S.No.	COMPONENT	SPECIFICATION	QTY
1.	LAPTOP WITH MODEL SIM SOFTWARE	IC 7411	1

VERILOG CODE:

2 TO 4 DECODER

```
module decoder24_assign(en,a,b,y);
    // declare input and output ports
    input en,a,b;
    output [3:0]y;

    // supportive connection required
    wire enb,na,nb;
    assign enb = ~en;
    assign na = ~a;
    assign nb = ~b;

    // assign output value by referring to logic diagram
    assign y[0] = ~(enb&na&nb);
    assign y[1] = ~(enb&na&b);
    assign y[2] = ~(enb&a&nb);
    assign y[3] = ~(enb&a&b);

endmodule
```

4 TO 2 ENCODER

```
module priority_encoder_datafloe(A0,A1,Y0,Y1,Y2,Y3);
input Y0,Y1,Y2,Y3;
output A0,A1;
assign A1 = Y3 + Y2;
assign A0 = Y3 + ((~Y2)&Y1);
endmodule
```

RESULT:

The Verilog code was executed using MODEL SIM software and output was verified.