Laboratory Report

Digital Logic Design (EET 1211)

Name: ……. KAUSHIK LAKHANI……..

Registration No.….2041012002……

Branch/Section: ……CSIT – D…….

INSTITUTE OF TECHNICAL EDUCATION AND RESEARCH (FACULTY OF ENGINEERING)
SIKSHA 'O' ANUSANDHAN (DEEMED TO BE UNIVERSITY)
BHUBANESWAR, ODISHA

CONTENTS

S. No.	Name Of The Experiment	Date of Experiment	Page No.	Signature
1	Examine the Operation of Logic Gates Using HDL	29.11.2021	01-08	
2	Examine & Analyze Advantages of Gate Level Minimization for Boolean Function Using HDL	01.12.2021	09-19	
3	Design, Construct & Test the Combinational Circuit to Solve a Given Problem Using HDL	08.12.2021	20-35	
4	Construct and Test various Binary Adder and Subtractor circuits Using HDL	15.12.2021	36-50	
5	DESIGN AND TEST VARIOUS CODE CONVERTER CIRCUITS USING HDL	22.12.2021	51-64	
6	DESIGNOF MAGNITUDE COMPARATOR, DECODER AND MULTIPLEXER CIRCUIT USING HDL	09.01.2022	65-86	
7	CONSTRUCT, TEST AND INVESTIGATE THE OPERATION OF VARIOUS FLIP-FLOP CIRCUITS USING HDL	02.02.2022	87-97	

DIGITAL LOGIC DESIGN LAB (EET1211)

LAB I: Examine the Operation of Logic Gates Using HDL

Siksha 'O' Anusandhan Deemed to be University, Bhubaneswar

Branch: CSIT (B.Tech)			Section: CSIT-D
S. No.	**Name**	**Registration No.**	**Signature**
01	KAUSHIK LAKHANI	2041012002	KaushikLakhani

Marks: 08.5/10

Remarks:

Dipti Patnaik
29/12/24

Teacher's Signature

I. OBJECTIVE:

1. Investigation of the logic behavior of various gates using HDL:
    a) 7400 quadruple two-input NAND gates
    b) 7402 quadruple two-input NOR gates
    c) 7404 hex inverters
    d) 7408 quadruple two-input AND gates
    e) 7432 quadruple two-input OR gates
    f) 7486 quadruple two-input XOR gates

2. Using a single 7400 IC, connect and implement a circuit using HDL that produces
    a) An inverter.
    b) A two-input AND.
    c) A two-input OR.
    d) A two-input XOR.

3. Construct & record the output of circuit using HDL that implements the Boolean function:

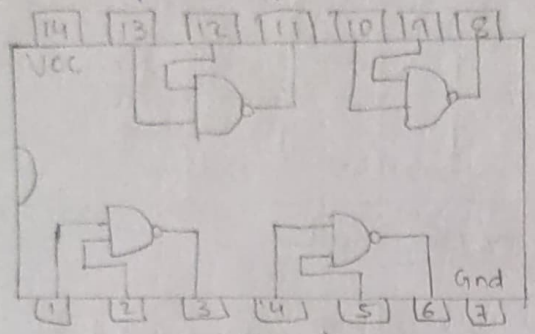
$$F=A\,(B+C)$$

a) Construct the circuit using Logic gates & verify the truth table.
b) Construct the circuit using NAND gates only & verify the truth table.

II. PRE-LAB

① Draw the IC Diagram & obtain truth table for obj. 1

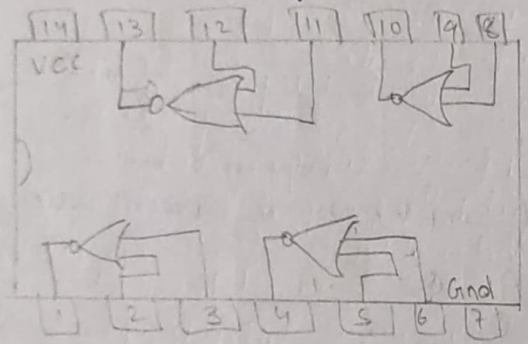
(a) 7400 quadruple two-input NAND Gates.



7400 Quad 2 Input NAND Gates

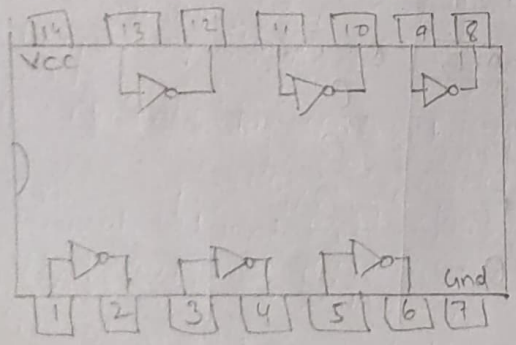
A	B	$X = \overline{A \cdot B}$
0	0	1
0	1	1
1	0	1
1	1	0

(b) 7402 quadruple two-input NOR Gates.



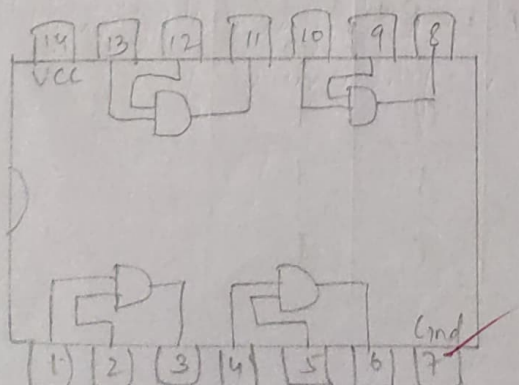
A	B	$X = \overline{A+B}$
0	0	1
0	1	0
1	0	0
1	1	0

(c) 7404 Hex one-input NOT Gate.



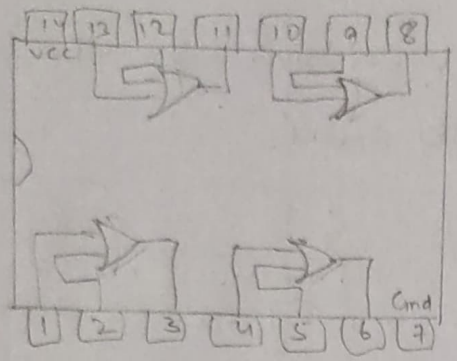
A	$X = \bar{A}$
0	1
1	0

(d) 7408 Quadruple two-input AND Gate.



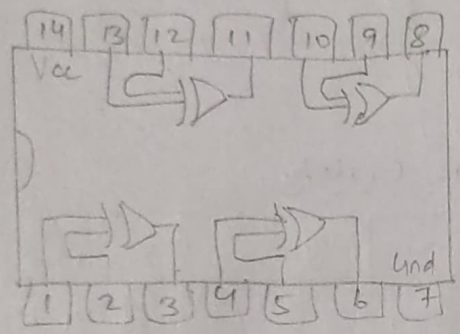
A	B	$X = A \cdot B$
0	0	0
0	1	0
1	0	0
1	1	1

(e) 7432 quadruple two-input OR Gate.



A	B	X = A + B
0	0	0
0	1	1
1	0	1
1	1	1

(f) 7486 quadruple two-input XOR Gate

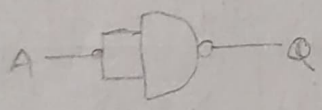


A	B	X = A⊕B
0	0	0
0	1	1
1	0	1
1	1	0

Objective 2:-

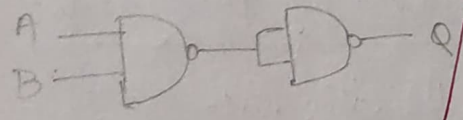
Using a single 7400 IC, connect and implement a circuit using HDL that produces

(a) An inverter



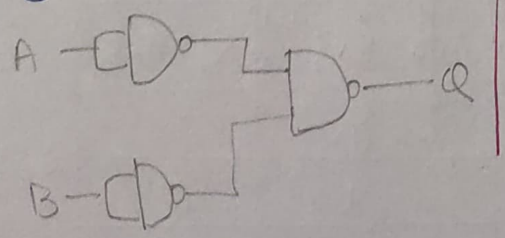
A	$Q = \overline{A \cdot A}$
0	1
1	0

(b) A two-input AND.



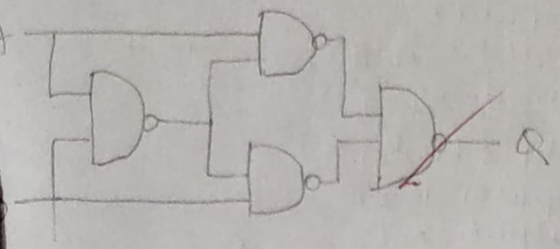
A	B	$C = \overline{A \cdot B}$	$Q = \overline{C \cdot C}$
0	0	1	0
0	1	1	0
1	0	1	0
1	1	0	1

(c) A two-input OR



A	B	$C = \overline{A \cdot A}$	$D = \overline{B \cdot B}$	$Q = \overline{C \cdot D}$
0	0	1	1	0
0	1	1	0	1
1	0	0	1	1
1	1	0	0	1

(d) A two-input XOR



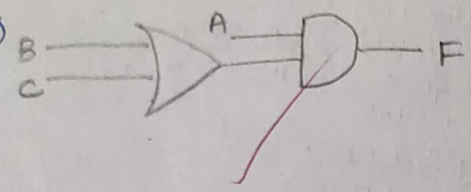
A	B	$C=\overline{A \cdot B}$	$D=\overline{A \cdot C}$	$E=\overline{B \cdot C}$	$Q=\overline{D \cdot E}$
0	0	1	1	1	0
0	1	1	1	0	1
1	0	1	0	1	1
1	1	0	1	1	0

OBJECTIVE → III

Construct & record the output of circuit using HDL that implements the Boolean Function :-

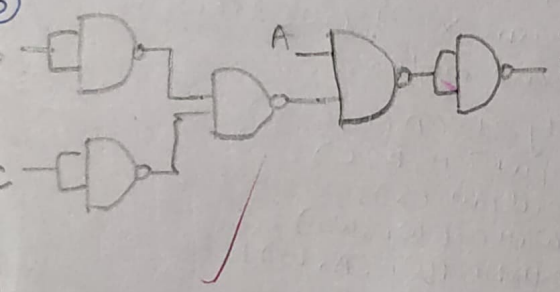
$F = A(B+C)$

(a)



A	B	C	B+C	F=A(B+C)
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	1	0
1	0	0	0	0
1	0	1	1	1
1	1	0	1	1
1	1	1	1	1

(b)



A	B	C	$X=\overline{B \cdot B}$	$Y=\overline{C \cdot C}$	$Z=\overline{X \cdot Y}$	$P=\overline{A \cdot Z}$	$F=\overline{P \cdot P}$
0	0	0	1	1	0	1	0
0	0	1	1	0	1	1	0
0	1	0	0	1	1	1	0
0	1	1	0	0	1	1	0
1	0	0	1	1	0	1	0
1	0	1	1	0	1	0	1
1	1	0	0	1	1	0	1
1	1	1	0	0	1	0	1

II. LAB :-

HDL Program :-

1 (a)
```
module objla(A,B,X);
  input (A,B);
  output(A,B);
  NAND(X,A,B);
endmodule
```

(b)
```
module objlb(A,B,X)
  input (A,B);
  output (X);
  NOR(X,A,B);
endmodule
```

(c)
```
module obj1c(A, X);
  input (A);
  output (X);
  NOT(X, A);
end module
```

(d)
```
module obj1d(A, B, X)
  input (A, B);
  output (X);
  AND(X, A, B);
endmodule
```

(e)
```
module obj1e(A, B, X);
  input (A, B);
  output (X);
  OR(X, A, B);
endmodule
```

(f)
```
module obj1f(A, B, X)
  input(A, B);
  output (X);
  XOR(X, A, B);
end module
```

(2) (a)
```
module obj2a(A, X);
  input (A);
  output (X);
  assign X = (AU A);
end module
```

(b)
```
module obj2b(A, B, X);
  input (A, B);
  output (X);
  wire (w);
  assign w = ! (A && B);
  assign x = ! (w, && w, );
  end module
```

(c)
```
module obj2c(A, B, X);
  input (A, B);
  output (X);
  write (w1, w2);
assign w1 = ! (A && A);
assign w2 = ! (B && B);
assign x = ! (w1 && w2);
  end module
```

(d)
```
module obj2d (A, B, X);
  input (A, B);
  output (X);
wire (w1, w2, w3);
assign w1 = ! (A && B);
assign w2 = ! (w2 && A, );
assign w3 = ! (w1 && B);
assign x = ! (w2 && w3);
  end module.
```

(3)

(a)
```
module obj3a(A, B, C, X);
  input (A, B, C);
  output (X);
  wire (w);
  OR (w, B, C);
AND (X, w, A);
end module
```

(b)
```
module obj3b (A, B, C, X);
input ( A, B, C);
  output (X);
  wire (w1, w2);
NAND (w1, A, B);
NAND (w2, A, C);
NAND (X, w1, w2);
end module
```

(4) COMPONENT ISSUE TABLE

- OBJECTIVE 1

SI No.	NAME	SPECIFICATION	QUANTITY
1	universal kit	nicholab	1
2	connecting wire	23 swg	1IC
3	7400	quad 2 input NAND	1 IC
4	7402	quad 2 input NOR	1 IC
5	7404	Hex input NOT	1IC
6	7408	quad 2 input AND	1IC
7	7432	quad 2 input OR	1IC
8	7486	quad 2 input XOR	1IC

OBJECTIVE 2 :-

Sl No	NAME	SPECIFICATION	QUANTITY
1	Universal Kit	micro lab	1
2	Connecting wire	23 SWG	as required
3	7400	Quad 2-input NAND	1 IC
1	Universal Kit ~~wire~~	microlab	1
2	wire	23 SWG	As required
3	7400	quad 2 input NAND	1 IC
4	7408	quad 2 input AND	1 IC
5	4432	quad 2 input OR	1 IC

OBSERVATION :-

OB1 → all the required gates are ~~blocking~~ working and gave output

OB2 → NAND can be successfully used to make the other gates.

OB3 → Output of circuit made of AND and OR gate shows the same output as the circuit purely made of NAND

CONCLUSION :-

OB1 → We conclude that the required gates/IC are working properly and giving the proper output.

OB2 → We conclude that NAND gate is universal gate since it can be used to make all the other gates and give proper output for those gates.

OB3 → We conclude that our conclusion from objective 2 is valid. We made a circuit for the same boolean expression on using the normal AND and OR gate and the other purely constructed by NAND gates. Both circuit give the same output.

IV. POST LAB

Q1 What is voltage range for operation of digital circuit?

Ans for low logic state, 0V → 0.8V and for high 2V → 5V.

Q2 What is the significance of ground and VCC connection?

Ans VCC is higher w.r.t. ground: VCC is the power input of a device. It may be positive/negative w.r.t. NAND but most of the world runs on ⊕ voltage like in a vehicle. GND is basically at 0V for a power supply and circuit. The ⊕ or ⊖ supply output are above or below the zero volt respectively.

Q3 Which gate is universal and why?

Ans NAND is universal gate and NOR gate, because they can be used to make any other gates.

Q4 What is min. no. of NAND Gates used to make XOR Gates.

Ans 4 is minimum no. of NAND Gate used to make XOR Gate.

DIGITAL LOGIC DESIGN LAB (EET1211)

LAB II: Examine & Analyze Advantages of Gate Level Minimization for Boolean Function Using HDL

Siksha 'O' Anusandhan Deemed to be University, Bhubaneswar

Branch: CSIT Section: CSIT - D / 2041010			
S. No.	**Name**	**Registration No.**	**Signature**
02	KAUSHIK LAKHANI	2041012002	Kaushik Lakhani

Marks: 9.5/10

Remarks:

→ pin configuration of the circuit missing

Teacher's Signature 3/1/22

I. OBJECTIVE:

1. Construct a circuit using basic gates that implements the Boolean function given below and record the output for all sets of input.

$$F = AB + AB'C$$

a. Simplify the Boolean function using minimization technique.

b. Construct the circuit for the simplified expression using basic gates & verify the truth table.

c. Write HDL code for the function F and for the simplified expression.

2. Simplify the following Boolean functions to a minimum number of literals and write HDLprogram for each functions.

a) F= XY +X'Z+YZ

b) F= (X'Y'+Z)'+Z+XY+WZ

3. Consider two Boolean functions in sum-of-min terms form:

F1 (A, B, C, D) = (0, 1, 2, 3, 4, 6, 8, 9, 10, 11)

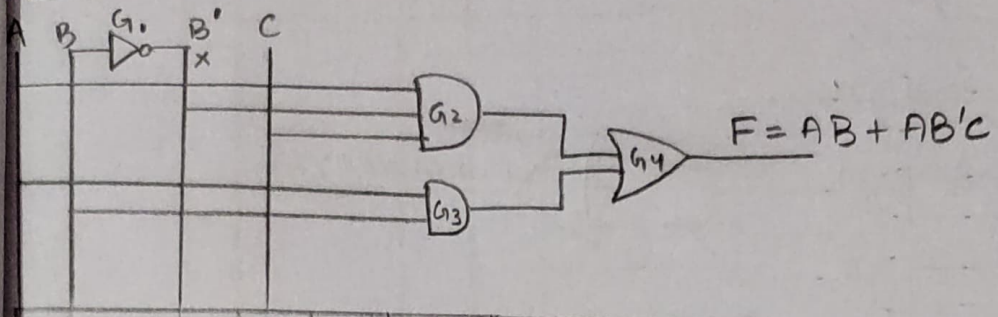
F2 (A, B, C, D) = (3, 5, 7, 8, 10, 11, 13, 15)

a. Implement both the functions using a minimum number of NAND ICs& verify the truth tables.

b.Write HDL code for the functions

PRE-LAB

For Objective 1 :-



A	B	C	B'	A·B	A·B'	A·B'·C	F = AB + AB'C
0	0	0	1	0	0	0	0
0	0	1	1	0	0	0	0
0	1	0	0	0	0	0	0
0	1	1	0	0	0	0	0
1	0	0	1	0	1	0	0
1	0	1	1	0	1	1	1
1	1	0	0	1	0	0	1
1	1	1	0	1	0	0	1

$F = (A \cdot B) + (A \cdot B' \cdot C)$

$A(B + (B' \cdot C))$ Taking A common from both

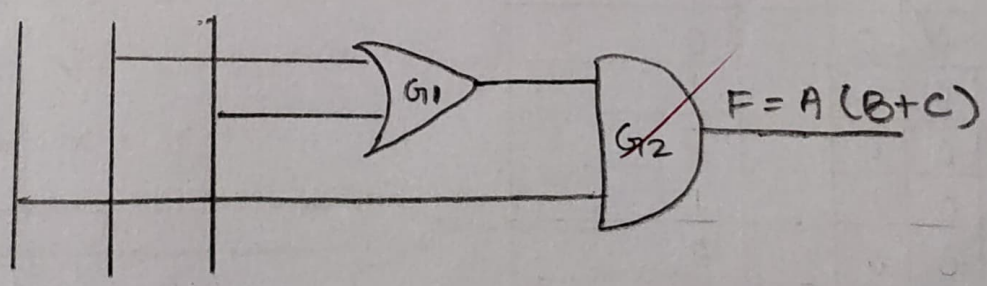
$+((B+B') \cdot (B+C))$ Distributive law

$A((1) \cdot (B+C))$ complement law

$A \cdot (B+C)$

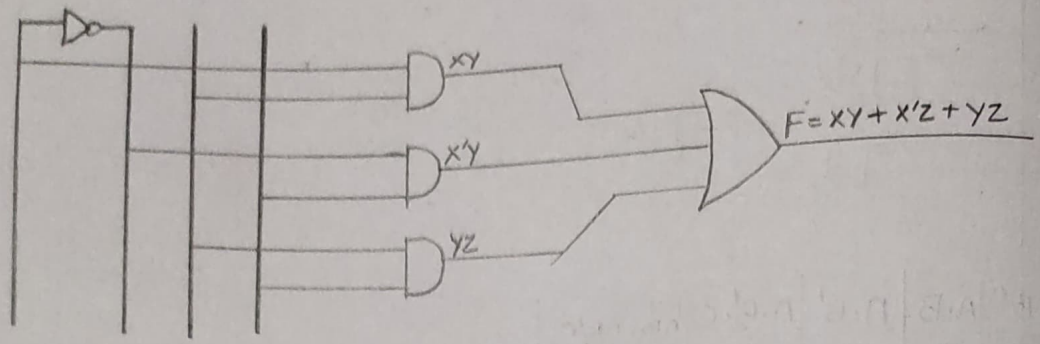
A	B	C	B+C	A(B+C)
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	1	0
1	0	0	0	0
1	0	1	1	1
1	1	0	1	1
1	1	1	1	1

Logic Diagram



2. For Objective 2 :-

(a) $F = XY + X'Z + YZ$



TRUTH TABLE ⇒

X	Y	Z	X'	XY	X'Z	YZ	F=XY+X'Z+YZ
0	0	0	1	0	0	0	0
0	0	1	1	0	1	0	1
0	1	0	1	0	0	0	0
0	1	1	1	0	1	1	1
1	0	0	0	0	0	0	0
1	0	1	0	0	0	0	0
1	1	0	0	1	0	0	1
1	1	1	0	1	0	1	1

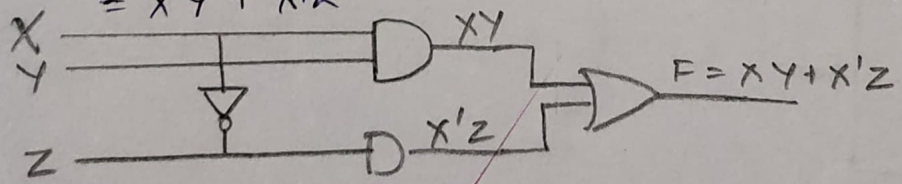
(b) $F = XY + X'Z + Y.Z$

$= XY + X'Z + YZ(X + X')$ (multiply 1 with Y.Z)

$= XY + X'Z + XYZ + X'YZ$ (Distributive Law)

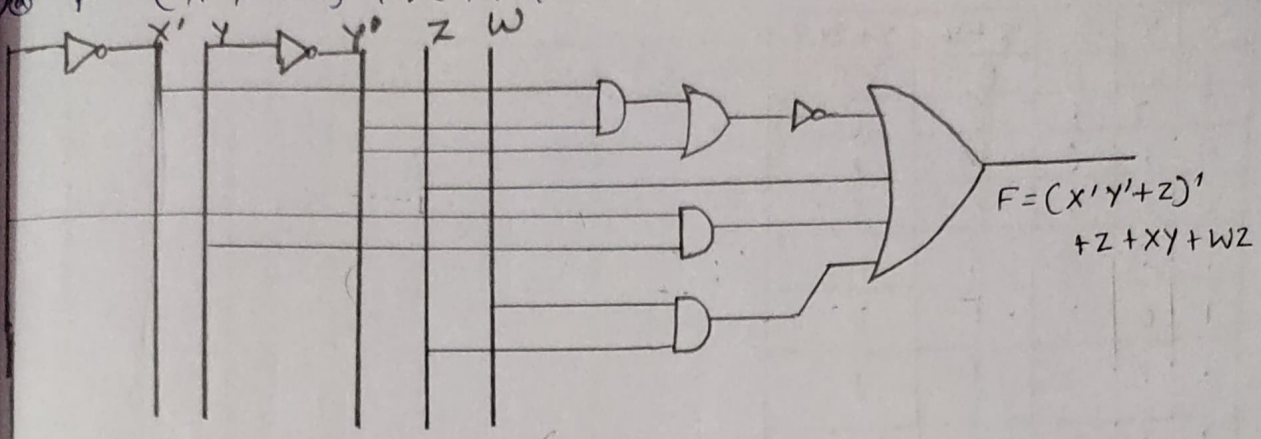
$= XY(1+Z) + X'Z(1+Y)$ (Taking XY and X'Z common)

$= XY + X'Z$ (Identity Law)



X	Y	Z	X'	XY	X'Z	F = XY + X'Z
0	0	0	1	0	0	0
0	0	1	1	0	1	1
0	1	0	1	0	0	0
0	1	1	1	0	1	1
1	0	0	0	0	0	0
1	0	1	0	0	0	0
1	1	0	0	1	0	1
1	1	1	0	1	0	1

(a) $F = (X'Y' + Z)' + Z + XY + WZ$



TH TABLE ⇒

X Y Z	X'	Y'	X'Y'	X'Y'+Z	(X'Y+Z)'	XY	WZ	(X'Y'+Z)'+Z+XY+WZ
0 0 0	1	1	1	1	0	0	0	0
0 0 1	1	1	1	1	0	0	0	1
0 1 0	1	0	0	0	1	0	0	1
0 1 1	1	0	0	1	0	0	0	1
1 0 0	0	1	0	0	1	0	0	1
1 0 1	0	1	0	1	0	0	0	1
1 1 0	0	0	0	0	1	1	0	1
1 1 1	0	0	0	1	0	1	0	1
0 0 0	1	1	1	1	0	0	0	0
0 0 1	1	1	1	1	0	0	1	1
0 1 0	1	0	0	0	1	0	0	1
0 1 1	1	0	0	1	0	0	1	1
1 0 0	0	1	0	0	1	0	0	1
1 0 1	0	1	0	1	0	0	1	1
1 1 0	0	0	0	0	1	1	0	1
1 1 1	0	0	0	1	0	1	1	1

$= (X'Y' + Z)' + Z + XY + WZ$

$= (X'Y' + Z)' + XY + Z(1 + W)$ (Taking Z common)

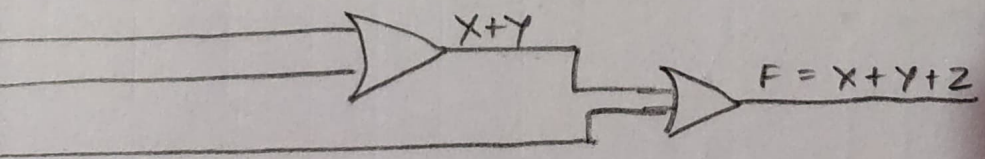
$= (X'Y' + Z)' + X \cdot Y + Z$ (Identity Law)

$= (X + Y) \cdot Z' + Z + XY$ (Commutative Law & De Morgan Thm)

$= (Z + Z')(X + Y + Z) + (X \cdot Y)$ (Distributive Law)

$= X + Y + Z + X \cdot Y$ (Distributive Law)

$= X + Y + Z$ (Absorption Law)



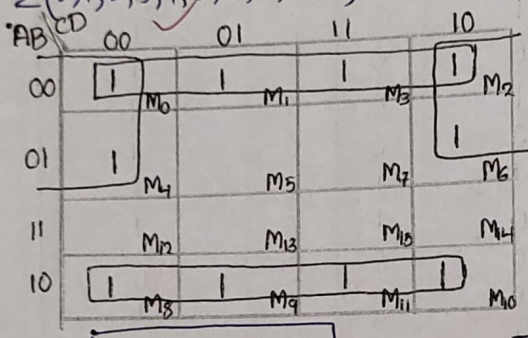
Truth Table

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

III. For Objective 3 :-

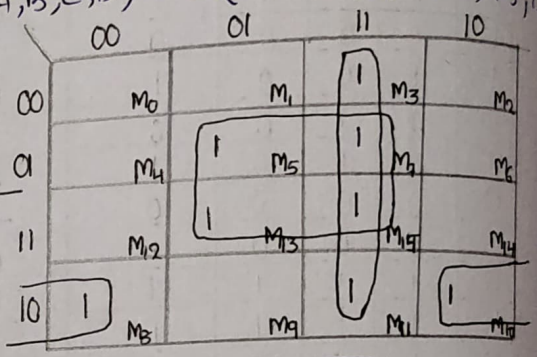
(a) Simplify the given expression using K-map :-

$F_1(A,B,C,D) = \Sigma(0,1,2,3,4,6,8,9,10,11)$



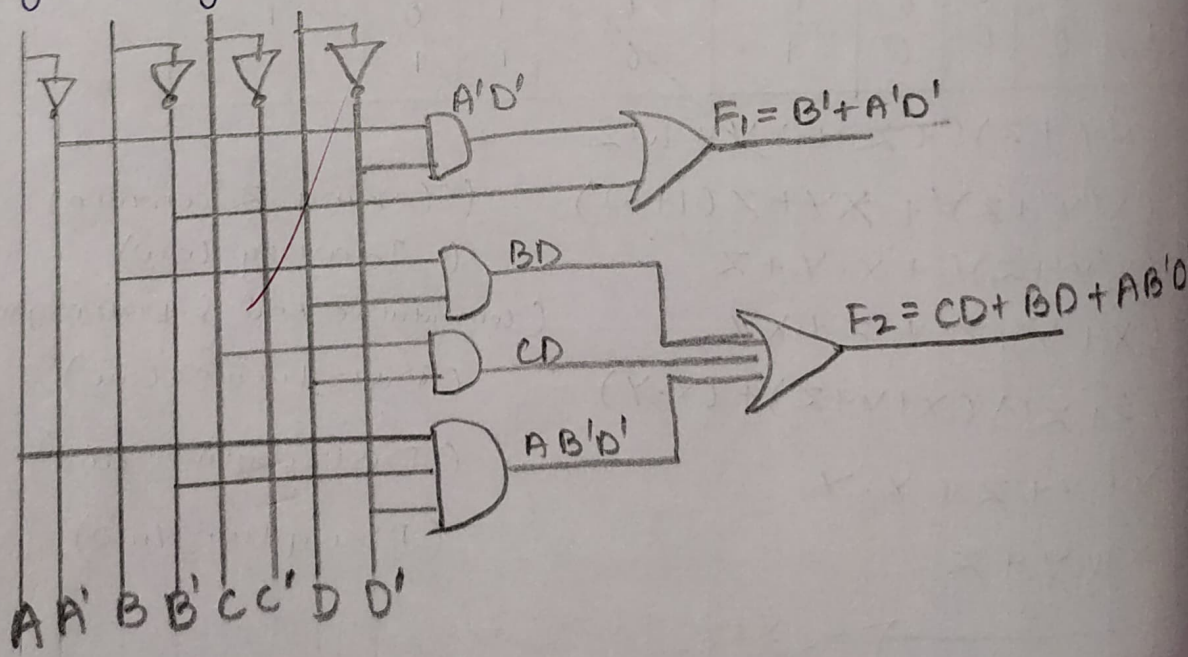
Simplified Expression $\Rightarrow$ $B' + A'D'$

$F_2(A,B,C,D) = \Sigma(3,5,7,8,10,11,13,15)$



$CD + BD + AB'D'$

(b) Logic Diagram :-



RUTH TABLE ⇒

A	B	C	D	$F_1 = B' + A'D'$	$F_2 = CD + BD + AB'D'$
0	0	0	0	1	0
0	0	0	1	1	0
0	0	1	0	1	0
0	0	1	1	1	1
0	1	0	0	1	0
0	1	0	1	0	1
0	1	1	0	0	0
0	1	1	1	1	1
1	0	0	0	1	0
1	0	0	1	1	1
1	0	1	0	1	0
1	0	1	1	1	1
1	1	0	0	0	0
1	1	0	1	0	1
1	1	1	0	0	0
1	1	1	1	0	1

LAB:

COMPONENTS REQUIRED:-

No.	Name of Component	Specification	Quantity
1	Universal Trainer Board	Microlab	1
2	Connecting Wires	23 SWG	as required
3	7400 IC	Quad 2 input NAND	1
4	7410 IC	3 input NAND	1
5	7404 IC	Hex inventor	1
6	7408 IC	Quad AND	1
7	7432 IC	Quad OR	1

L Program

objective 1:-

= AB + AB'C

dule function1(A,B,C,F);
tput F;
ut A,B,C;
re X,Y,Z;
t G1(X,B);
d G2(Y,A,X,C);
d G3(Z,A,B);
r G4(F,Y,Z);
dmodule

(b) F = A(B+C)

module funtion2(A,B,C,F);
output F;
input A,B,C;
wire X;
or G1(X,B,C);
and G2(F,X,A);
endmodule

- For objective 2a:-

(a) F = XY + X'Z + YZ

```
module function3(X,Y,Z,F);
output F;
input X,Y,Z;
wire w1,w2,w3;
not G1(w1, X);
and G2(w2,X,Y);
and G3(w3,w0,Z);
 or G4(F,w2,w3);
endmodule
```

(b) F = XY + X'Z

```
module function4(X,Y,Z,F);
output F;
input X,Y,Z;
wire w1;
or G1(w1,X,Y);
or G2(F,w1,Z);
endmodule
```

II For Objective 2b -

(a) F = (X'Y'+Z)' + Z + XY + WZ

```
module function5(W,X,Y,Z,F);
 output F;
 input W,X,Y,Z;
 wire w1,w2,w3,w4,w5,w6,w7,w8,w9
 not (w1,X);
 not (w2,Y);
 and (w3,X,Y);
 and (w4,W,Z);
 and (w5,w1,w2);
  or (w6,w3,w4);
  or (w7,w5,Z);
 not (w8,w7);
  or (w9,w8,Z);
  or (F,w9,w6);
  endmodule
```

(b) F = X + Y + Z

```
module function6(X,Y,Z,F);
 output F;
 input X,Y,Z;
 wire w1;
 or (w1,X,Y);
 or (F,Z,w1);
 endmodule
```

For Objective 3 :-

(a) F1 = B' + A'D'

```
module function7(A,B,C,D,F1);
  output F1;
 input A,B,D;
 wire w1,w2,w3;
 nand G1(w1,A,A);
 nand G2(w2,D,D);
 nand G3(w3,w1,w2);
 nand G4(F1,B,w3);
 endmodule
```

(b) F2 = BD + CD + AB'D'

```
module function8(A,B,C,D,X);
 output X;
 input A,B,C,D;
 wire w1,w2,w3,w4,w5;
 nand G1(w1,C,D);
 nand G2(w2,B,D);
 nand G3(w3,B,B);
 nand G4(w4,D,D);
 nand G5(w5,w3,w4);
 nand G6(F2,w1,w2,w5);
 endmodule
```

SERVATION ⇒

objective 1 :-

A	B	C	B+C	A(B+C)	Practical Output
0	0	0	0	0	0
0	0	1	1	0	0
0	1	0	1	0	0
0	1	1	1	0	0
1	0	0	0	0	0
1	0	1	1	1	1
1	1	0	1	1	1
1	1	1	1	1	1

objective 2 :-

X	Y	Z	X'	XY	X'Z	XY+X'Z	Practical Output
0	0	0	1	0	0	0	0
0	0	1	1	0	1	1	1
0	1	0	1	0	0	0	0
0	1	1	1	0	1	1	1
1	0	0	0	0	0	0	0
1	0	1	0	0	0	0	0
1	1	0	0	1	0	1	1
1	1	1	0	1	0	1	1

X	Y	Z	X+Y+Z	Practical Output
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	1	1
1	0	0	1	1
1	0	1	1	1
1	1	0	1	1
1	1	1	1	1

objective 3 :-

A	B	D	A'	B'	D'	A'D'	B'+A'D'	Practical Output
0	0	0	1	1	1	1	1	1
0	0	1	1	1	0	0	1	1
0	1	0	1	0	1	0	1	1
0	1	1	1	0	0	0	0	0
1	0	0	0	1	1	0	1	1
1	0	1	0	1	0	0	1	1
1	1	0	0	0	1	0	0	0
1	1	1	0	0	0	0	0	0

A	B	C	D	B'	D'	AB'D'	BD	CD	AB'D'+BD+CD	Practical Output
0	0	0	0	1	1	0	0	0	0	0
0	0	0	1	1	0	0	0	0	0	0
0	0	1	0	1	1	0	0	0	0	0
0	0	1	1	1	0	0	0	1	1	1
0	1	0	0	0	1	0	0	0	0	0
0	1	0	1	0	0	0	1	0	1	1
0	1	1	0	0	1	0	0	0	0	0
0	1	1	1	0	0	0	1	0	1	1
1	0	0	0	1	1	0	0	0	0	0
1	0	0	1	1	0	0	0	0	0	0
1	0	1	0	1	1	0	0	0	0	0
1	0	1	1	1	0	0	0	1	1	1
1	1	0	0	0	1	0	0	0	0	0
1	1	0	1	0	0	0	1	0	1	1
1	1	1	0	0	1	0	0	0	0	0
1	1	1	1	0	0	0	1	0	1	1

CONCLUSION ⇒

- Objective 1 :- From the boolean law, the given func CAB + AB'C) was simplified to A(B+C) and can also be reconstructed.
- Objective 2 :- Using the boolean law, the func can be simplified to minimum number of literals.
- Objective 3 :- It was concluded that using K-Map, the function can be simplified to minimum number of literal and can also be implemented using NAND ICs.

IV. POST-LAB ⇒

Q1. What is the advantage of using boolean laws?

Ans ① If we use boolean laws, we can save more gates and operations, which will make our design cheaper, more comprehensible & more serviceable.

② It allows logical steps quickly and repeatedly.

What is a K-Map? What are its advantages and disadvantages?

K-Map or Karnaugh maps take truth tables and provide a visual way to produce a much simpler formula for expressing the same logic.

ADVANTAGES ⇒

(i) Reduces the number of logic gates.
(ii) It reduces the chance of error.
(iii) Reduces the cost of circuit/design.
(iv) Less no. of steps when compared to algebraic minimization techniq

DISADVANTAGES ⇒

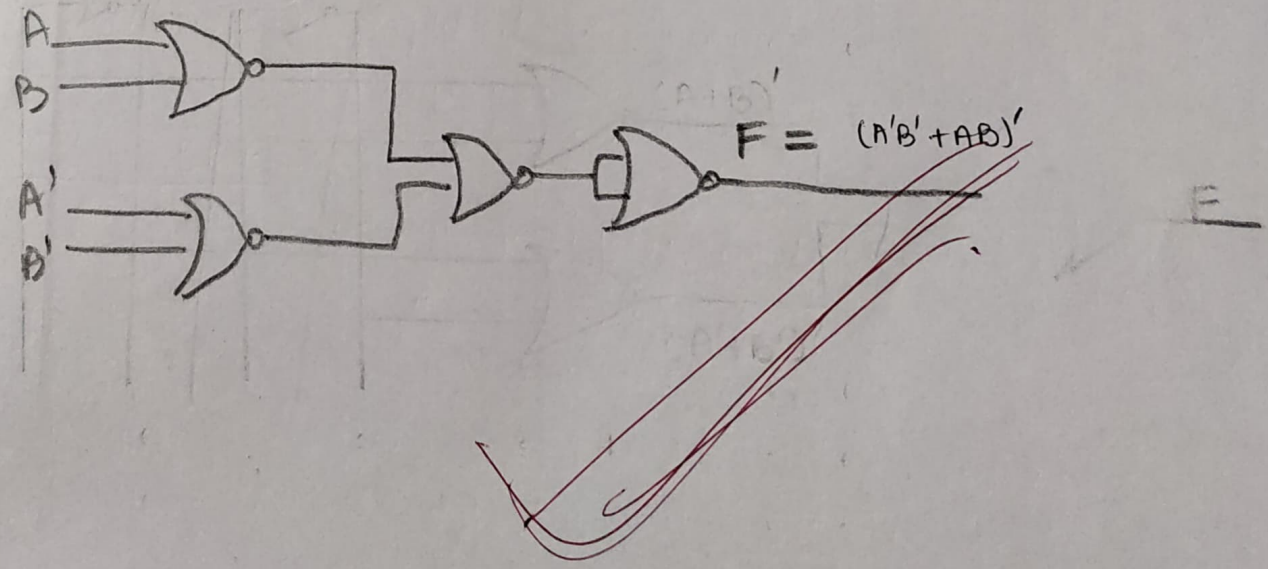
It is tedious for more than 5 variables.
It is not suitable for computer reduction.
(iii) Don't care condition should be taken care of.

What ways can boolean expression be represented?

We can represent Boolen Expression using 2 forms :-
(i) Standard Form (ii) Canonical Form.

Implement two input Ex-NOR Gate using minimum number of 2 input NOR Gates.

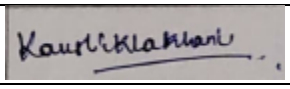
$F = A'B' + AB = A \odot B$



DIGITAL LOGIC DESIGN LAB (EET1211)

LAB III: Design, Construct & Test the Combinational Circuit to Solve a Given Problem Using HDL

Siksha 'O' Anusandhan Deemed to be University, Bhubaneswar

Branch: CSIT			**Section:** D
S. No.	**Name**	**Registration No.**	**Signature**
1	Kaushik Lakhani	2041012002	

Marks: _____/10

Remarks:

Teacher's Signature

I. OBJECTIVE:

1. Design a combinational circuit with four inputs A, B, C, and D and one output F. F is to be equal to 1 when (A = 1 and B = 0), or when (A=0, B = 1, provided that either C or D is equal to 1). F is also equal to 1 when (A=0, B=1, C=1 and D=1). Otherwise, the output is to be equal to 0.

2. Design and test a 3-input majority circuit using NAND gates with a minimum number of ICs. A majority logic is a digital circuit whose output is equal to 1 if the majority of the inputs are 1's. The output is 0 otherwise.

3. Design, construct, and test a circuit that generates an even parity bit from four message bits.

4. Design a combinational circuit that compares two 2-bit numbers A and B to check if they are equal or not.

II. PRE - LAB

For objective 1 :

a.) Obtain the truth table for output F as a function of 4 inputs (A, B, C, D) based on the given logic.

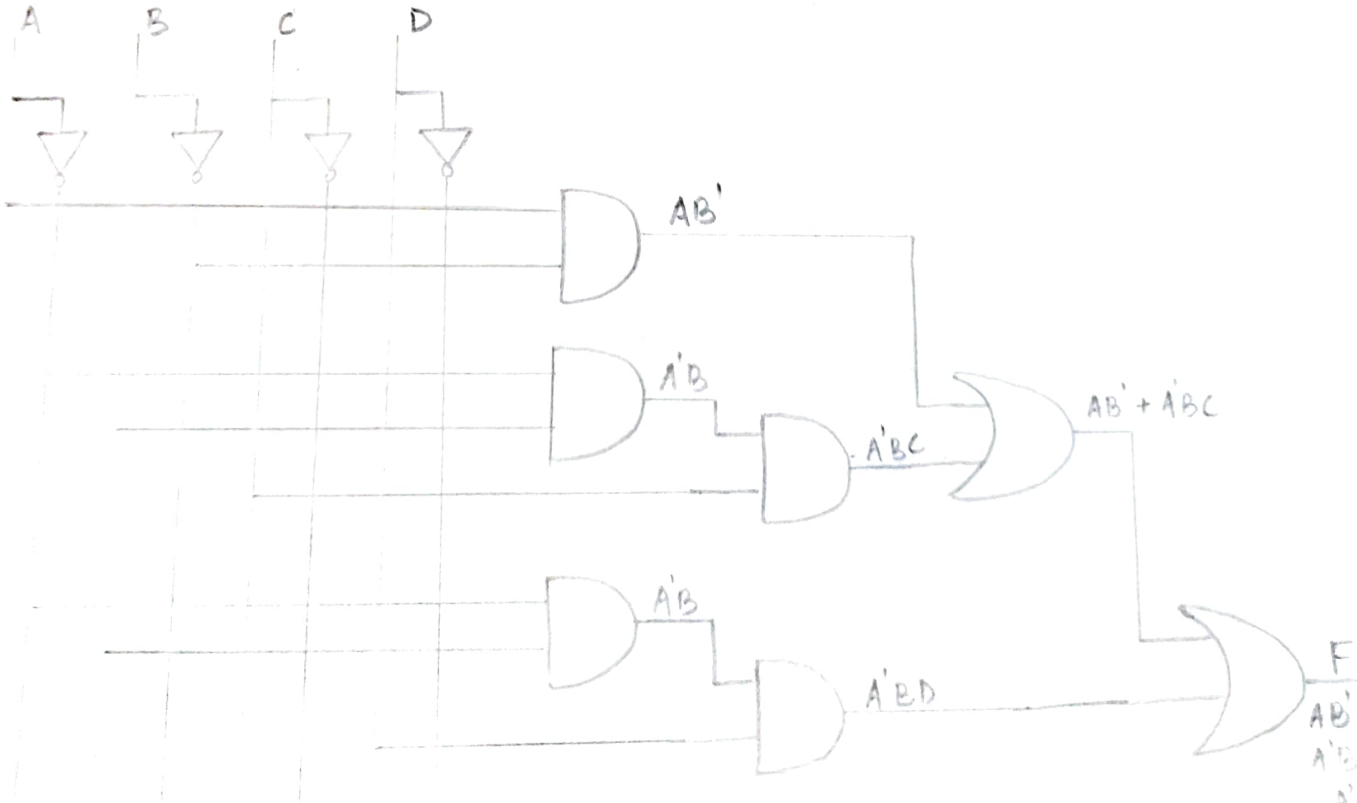
A	B	C	D	F
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	1
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	0

b.) Derive a simplified Boolean expression for F that satisfies the truth table.

AB \ CD	C'D'	C'D	CD	CD'
A'B'				
A'B		1	1	1
AB				
AB'	1	1	1	1

Simplified expression : $AB' + A'BD + A'BC$

C.) Draw the logic diagram of simplified Boolean expression.



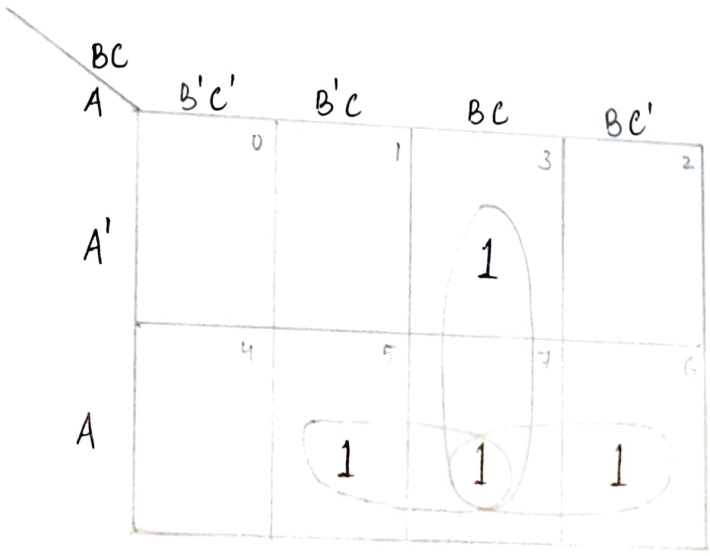
For objective 2 :

a) Draw the truth table for 3 input majority circuits.

A	B	C	F
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

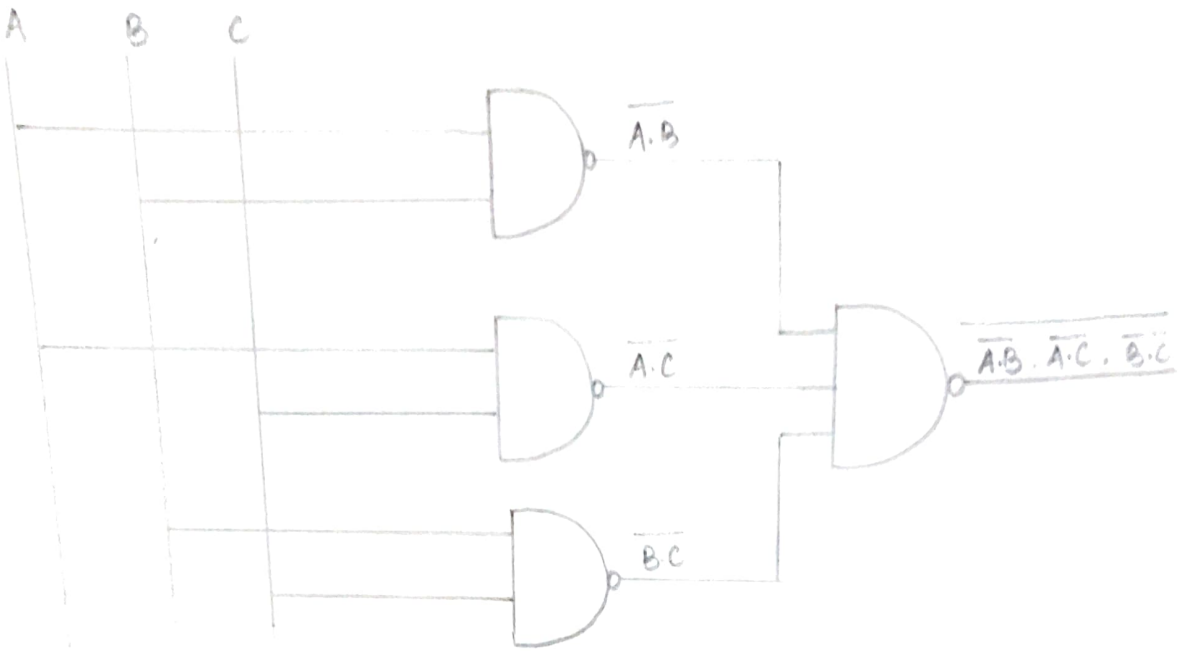
b) Write a simplified boolean expression for the output of the given 3 input majority circuit.

$F(A, B, C) = \Sigma (3, 5, 6, 7)$



Simplified Expression : $AB + AC + BC$

Draw the circuit diagram using NAND gates according to the simplified Boolean expression.



For Objective 3 :

a) Obtain the truth table for the parity bit as output of the circuit corresponding to 4 input message bits.

A	B	C	D	F
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	0
0	1	0	0	1
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	1
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	1
1	1	1	1	0

b) Derive a simplified Boolean function that defines the value of parity bit for given combination of 4 messages bits.

$$F(A,B,C,D) = \Sigma\ (1,2,4,7,8,11,13,14)$$

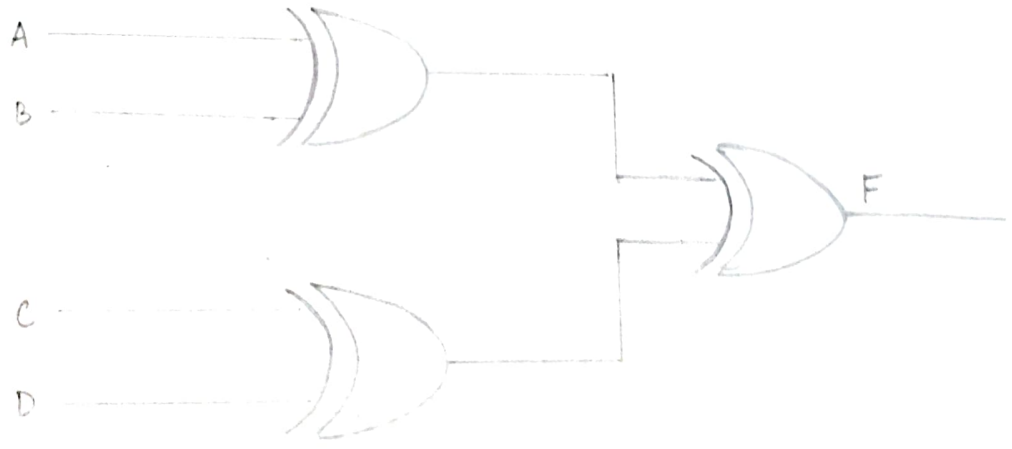
AB \ CD	C'D'	C'D	CD	CD'
A'B'		1		1
A'B	1		1	
AB		1		1
AB'	1		1	

$$= A'B'(C'D + CD') + A'B(C'D' + CD) + AB(C'D + CD') + AB'(C'D' + CD)$$
$$= A'B'(C \oplus D) + A'B(C \odot D) + AB(C \oplus D) + AB'(C \odot D)$$
$$= (C \oplus D)(A'B' + AB) + (C \odot D)(A'B + AB')$$
$$= (C \oplus D)(A \odot B) + (C \odot D)(A \oplus B)$$

Let $(A \oplus B)$ & $(C \oplus D)$ be X & Y respectively therefore equation will be $XY' + X'Y$ that equals to $X \oplus Y$.

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

c) Draw the logic diagram using logic gates according to derived boolean function.



For Objective 4 :

a) Obtain the truth table to know for which set of data bits the output of the circuit is high / low.

A	B	C	D	F
0	0	0	0	1
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	1
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

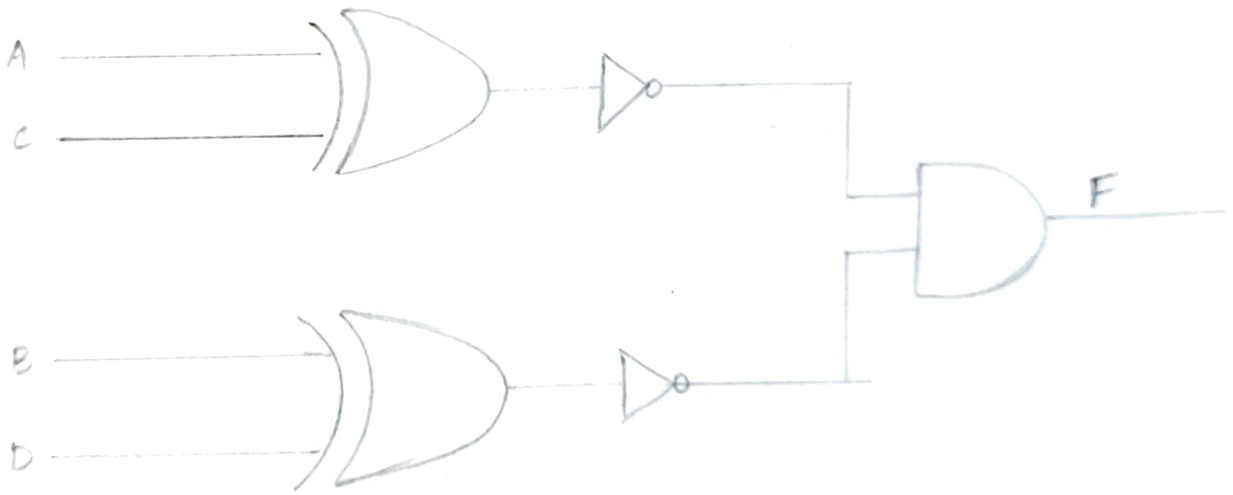
b) Derive a simplified Boolean expression that justifies the given logic.

$F(A,B,C,D) = \Sigma(0,5,10,15)$

AB \ CD	C'D'	C'D	CD	CD'
A'B'	1			
A'B		1		
AB			1	
AB'				1

$$
\begin{aligned}
F &= A'B'C'D' + A'BC'D + ABCD + AB'CD' \\
&= A'C'(B'D' + BD) + AC(BD + B'D') \\
&= A'C'(B \oplus D)' + AC(B \oplus D)' \\
&= (B \oplus D)'(A'C' + AC) \\
&= (A \oplus C)'(B \oplus D)'
\end{aligned}
$$

c) Draw the logic Diagram using logic gates according to derived boolean function.



III. LAB

COMPONENTS REQUIRED

S.NO.	Name of the Component	Specification	Quantity
01	Universal trainer kit	Microlab	1
02	Connecting Wires	23 SWG	As required
03	IC 7408	Quad 2 input AND Gate	1
04	IC 7432	Quad 2 input OR Gate	1
05	IC 7404	Hex inverter NOT Gate	1
06	IC 7400	Quad 2 input NAND Gate	1
07	IC 7410	3 input NAND Gate IC	1
08	IC 7486	Quad 2 input XOR Gate	1

HDL PROGRAM

For Objective 1 :

For $F = AB' + A'BD + A'BC$

```
module lab3_Obj1 (A,B,C,D,X);
 Output X ;
 Input A,B,C,D ;
 Wire w1,w2,w3,w4,w5,w6,w7 ;
 not (w1, A) ;
 not (w2, B) ;
 and (w3, A, w2) ;
 and (w4, B, w1) ;
and (w5, C, w4) ;
 and (w6, D, w4) ;
 OR (w7, w3, w5);
 OR (X, w7, w6);
 endmodule
```

For Objective 2 :

For $F = AB + AC + BC$

```
module lab3_Obj2 (A,B,C,X);
 Output X ;
 input A,B,C ;
wire w1,w2,w3 ;
 nand (w1, A, B);
 nand (w2, A, C);
 nand (w3, B, C);
 nand (X, w1, w2, w3) ;
 endmodule
```

For Objective 3 :

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

```
module lab3_Obj3 (A,B,C,D,X);
 Output X ;
 input A,B,C,D ;
 wire w1 , w2 ;
 XOR ( w1 , A, B) ;
 XOR (w2 , C, D) ;
 XOR ( X , w1, w2) ;
 endmodule
```

For objective 4 :

$F = (A \oplus C)' (B \oplus D)'$

```
module lab3_Obj4 (A,B,C,D,X);
 Output X ;
 input A,B,C,D ;
 wire w1, w2, w3, w4 ;
 XOR (w1, A, C) ;
 XOR (w2 , B, D) ;
 not (w3 , w1) ;
 not (w4 , w2) ;
 and ( X , w3, w4) ;
endmodule
```

OBSERVATION :

Objective 1

A	B	C	D	F
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	1
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	0

Objective 2

A	B	C	F
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

Objective 3 :

A	B	C	D	F
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	0
0	1	0	0	1
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	1
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	1
1	1	1	1	0

Objective 4 :

A	B	C	D	F
0	0	0	0	1
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	1
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

CONCLUSION

Objective 1 :

From this objective it can be concluded that for getting the output as per the instruction the combinational circuit leads to the function.

$$AB' + A'BD + A'BC$$

Objective 2 :

From this objective it can be concluded that 3-input majority circuit leads to the function

$$AB + AC + BC$$

Objective 3 :

From this objective it can be concluded that even parity bit from four message bits leads to the function

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

Objective 4 :

From this objective it can be concluded that equality condition leads to the XNOR function.

IV. POST LAB

1) What do you understand by the term 'majority logic'?

Ans. A majority logic is a digital circuit whose output is equal to 1 if the majority of the inputs are 1's i.e (>50%). The output is 0 otherwise.

2) Suggest a suitable modification to be made in existing even parity circuit that can be used to generate bit for odd parity.

Ans. Take XOR of it with high input i.e 1.

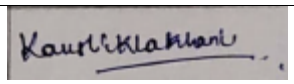
3) What is the function of a magnitude comparator circuit?

Ans. The function of a magnitude comparator circuit is to determine whether one number is greater than, less than or equal to the other number.

DIGITAL LOGIC DESIGN LAB (EET1211)

LAB IV: Construct and Test various Binary Adder and Subtractor circuits Using HDL

Siksha 'O' Anusandhan Deemed to be University, Bhubaneswar

Branch: CSIT			**Section:** D
S. No.	**Name**	**Registration No.**	**Signature**
1	Kaushik Lakhani	2041012002	

Marks: _____/10

Remarks:

Teacher's Signature

I. OBJECTIVE:

1. Design, construct and test a Full Adder circuit using two ICs, 7486(XOR) and 7400(NAND).

2. Design, construct, and test a Half Subtractor circuit using two ICs, 7486(XOR) and 7400(NAND).

3. Design, construct, and test a 2 bit Parallel Adder circuit.

II. PRE - LAB

For Objective 1 :

a) Write the truth table for full adder logic.

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

b) Derive the Boolean expression for sum and carry using XOR and NAND operation respectively.

A \ BC	B'C'	B'C	BC	BC'
A'		1		1
A	1		1	

Sum :- $A'(B'C + BC') + A(B'C' + BC)$

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

∴ Let $B \oplus C$ be X therefore equation will be $A'X + AX'$ that equals to $A \oplus X$

$= A \oplus B \oplus C$

IV.

1) Wh

Ans. A
ec
i

2) Sug
pas

ns. Tak

A \ BC	B'C'	B'C	BC	BC'
A'			1	
A		1	1	1

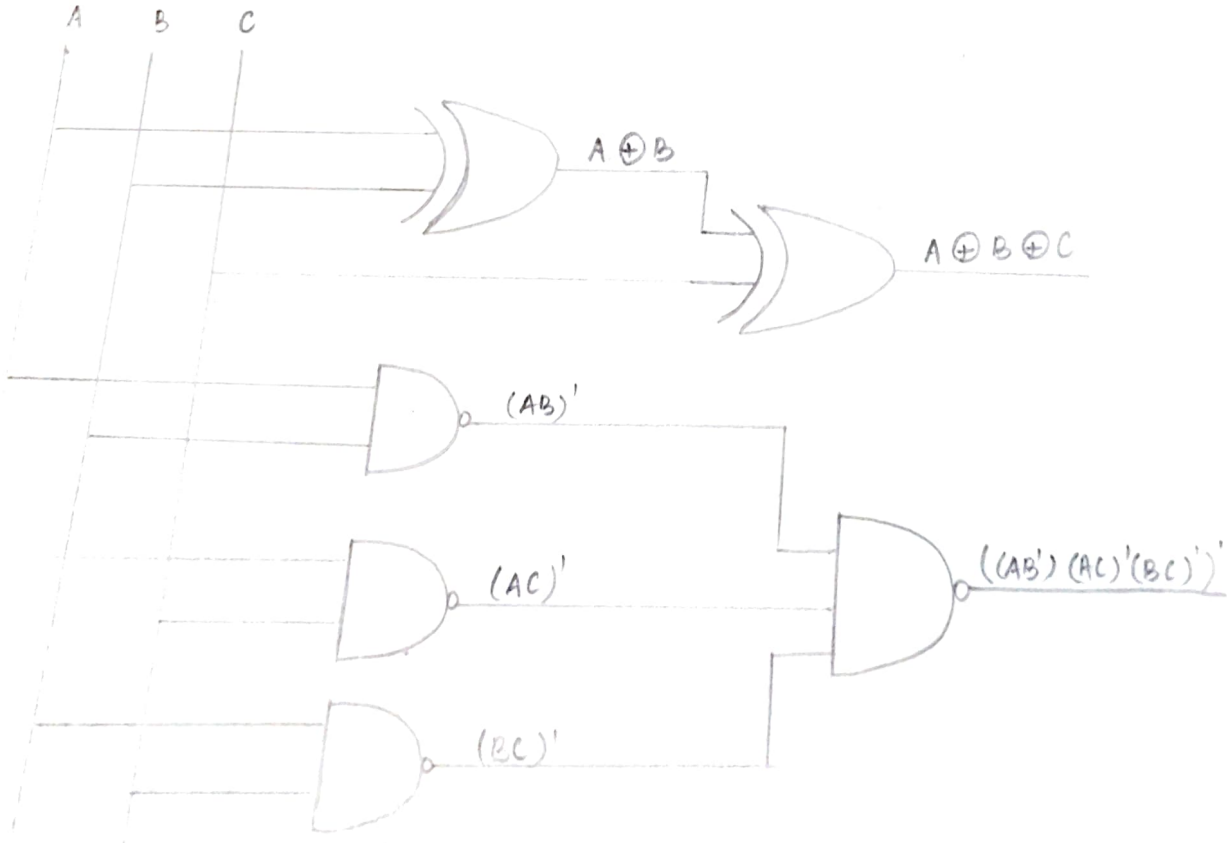
Carry :- $AB + AC + BC$

$= ((AB)'(AC)'(BC)')'$

c) Draw the logic diagram for sum and carry as output of the full adder circuit.

What

The
whet
the



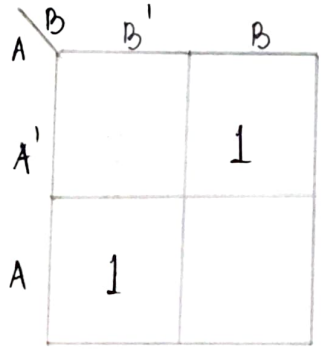
For objective 2 :

a) Write the truth table for Half subtractor logic.

A	B	Diff	Borrow
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

b) Derive the Boolean expression for difference and borrow using XOR and NAND operation respectively.

$$Diff(A,B') = \Sigma(1,2)$$



$$Diff :- \quad A'B + AB'$$

$$= A \oplus B$$

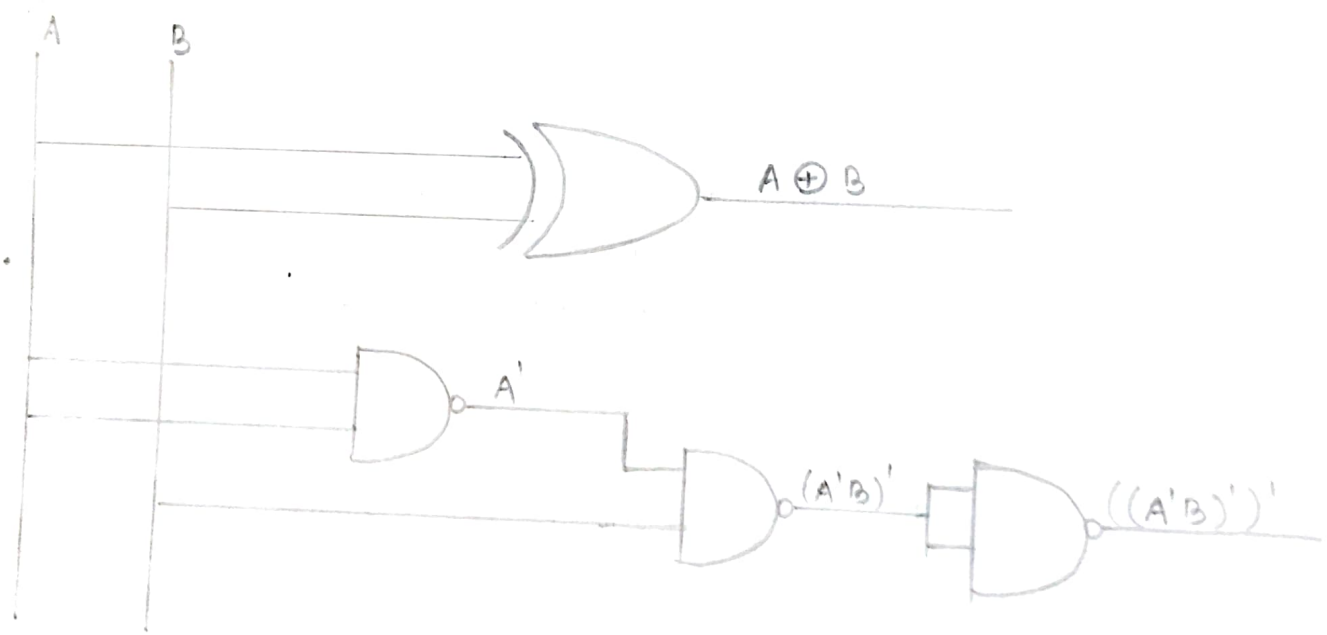
Borrow(A, B) = Σ(1)

A \ B	B'	B
A'		1
A		

Borrow :- A'B

= ((A'B)')'

c) Draw the logic diagram for difference and borrow as output of the half substractor circuit.



For objective 3 :

a) Write the truth table for 2-bit parallel adder.

A1	A0	B1	B0	C0	S0	S1	C1
0	0	0	0	0	0	0	0
0	0	0	1	0	1	0	0
0	0	1	0	0	0	1	0
0	0	1	1	0	1	1	0
0	1	0	0	0	1	0	0
0	1	0	1	1	0	1	0
0	1	1	0	0	1	1	0
0	1	1	1	1	0	0	1
1	0	0	0	0	0	1	0
1	0	0	1	0	1	1	0
1	0	1	0	0	0	0	1
1	0	1	1	0	1	0	1
1	1	0	0	0	1	1	0
1	1	0	1	1	0	0	1
1	1	1	0	0	1	0	1
1	1	1	1	1	0	1	1

b) Derive the Boolean expression for sum bit and carry bit using XOR and NAND operation respectively..

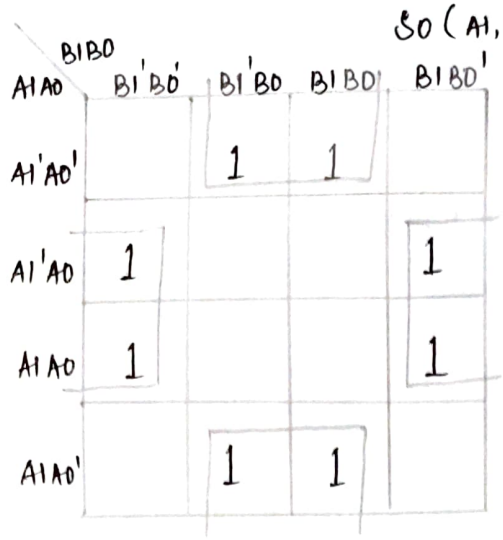
$$C0\,(A1, A0, B1, B0) = \Sigma(5, 7, 13, 15)$$

A1A0 \ B1B0	B1B0'	B1'B0	B1B0	B1B0'
A1'A0'				
A1'A0		1	1	
A1 A0		1	1	
A1 A0'				

C0 :- A0.B0

C0 :- ((A0. B0)')'

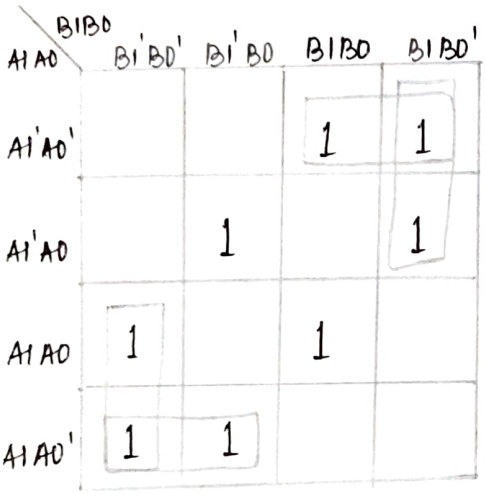
$S0(A1,A0,B1,B0) = \Sigma(1,3,4,6,9,11,12,14)$



$S0 :- A0'B0 + A0B0'$

$S0 :- A0 \oplus B0$

$S1(A1,A0,B1,B0) = \Sigma(2,3,5,6,8,9,12,15)$



$S1 :- A1'A0B1'B0 + A1A0B1B0 + A1B1'B0' + B1'A1A0' + A1'A0'B1 + A1'B1B0'$

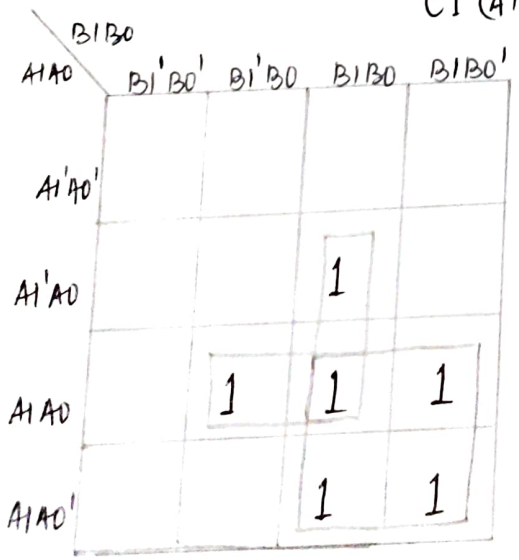
$S1 :- A0B0(A1'B1' + A1B1) + A1B1'(B0' + A0') + A1'B1(B0' + A0')$

$S1 :- A0B0 \overset{(A1 \oplus B1)' +}{\wedge} (A0' + B0')(A1 \oplus B1)$

$S1 :- A0B0 \overset{(A1 \oplus B1)' +}{\wedge} (A0 \cdot B0)'(A1 \oplus B1)$

$S1 :- C0 \oplus A1 \oplus B1$

$C1(A1,A0,B1,B0) = \Sigma(7,10,11,13,14,15)$

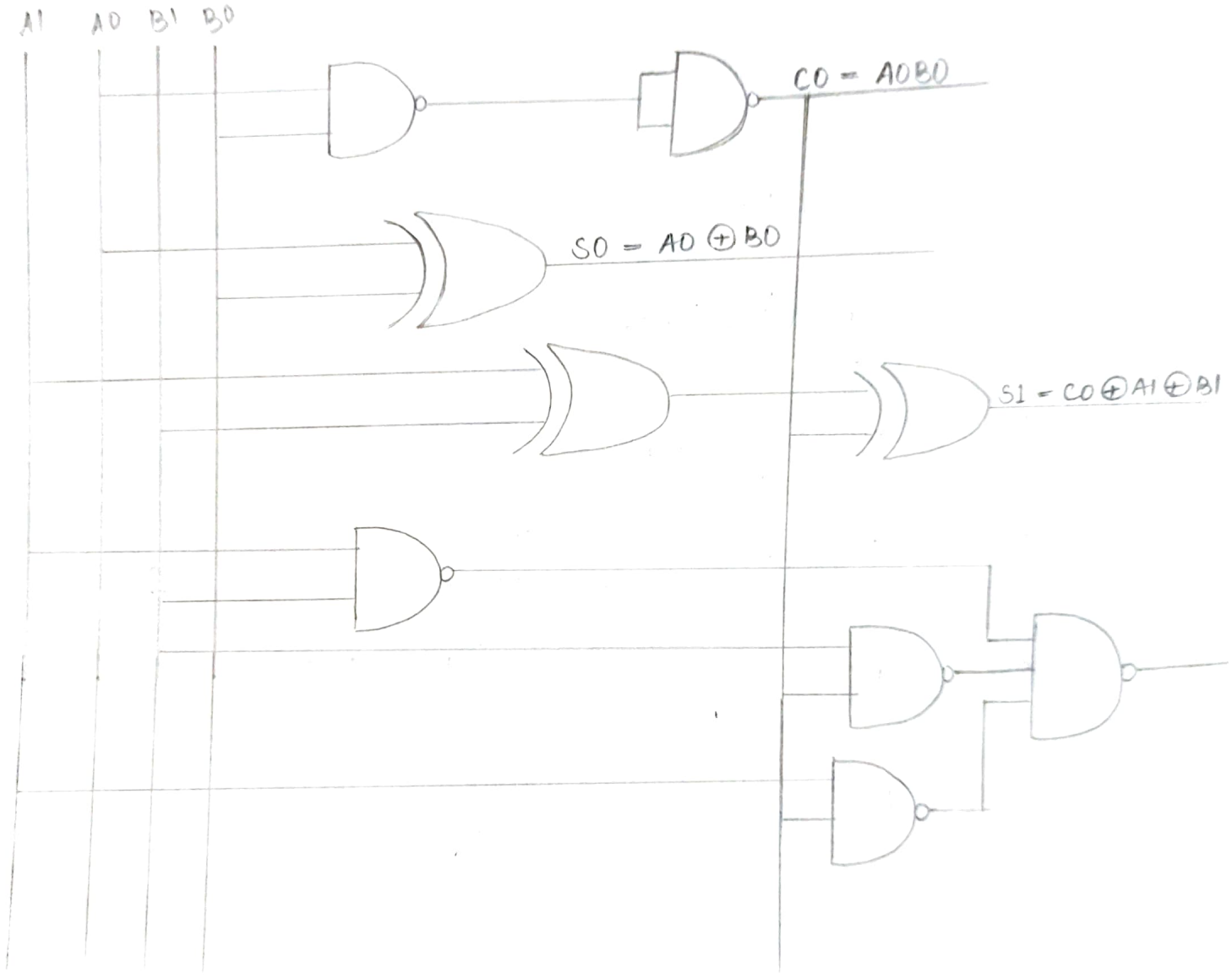


$C1 :- A1B1 + A0B1B0 + A1A0B0$

$C1 :- A1B1 + B1C0 + A1C0$

$C1 :- ((A1 \cdot B1)'(B1C0)'(A1C0)')'$

c) Draw the logic diagram based on boolean expression.



III. LAB

COMPONENTS REQUIRED

S.No.	Name of the component	Specification	Quantity
01	Universal Trainer kit	Micro lab	1
02	Connecting Wires	23 SWG	As required
03	IC 7400	Quad 2 input NAND Gate	2
04	IC 7410	3 input NAND Gate IC	1
05	IC 7486	Quad 2 input XOR Gate	1

HDL PROGRAM

For Objective 1 :

```
module lab4_Obj1 (A,B,C,S,CO);
 Output S,CO;
 input A,B,C;
 wire W1,W2,W3,W4;
 XOR (W1,A,B);
 XOR (S, W1,C);
 nand (W2,A,B);
 nand (W3, A,C);
 nand (W4, B,C);
 nand (CO, W2,W3,W4);
 endmodule
```

For objective 2 :

```
module lab4_Obj2 (A,B, BO,D);
  Output BO,D;
  input A,B;
  wire W1,W2;
  XOR (D,A,B);
  nand (W1,A,A);
  nand (W2,B,W1);
  nand (BO, W2,W2);
  endmodule
```

For Objective 3 :

```
module lab4_Obj3 ( A1, A0, B1, B0, C0, S0, S1, C1);
 Output  C0, S0, S1, C1 ;
 input   A1, A0, B1, B0 ;
 wire  w1, w2, w3, w4, w5 ;
 nand  (w1, A0, B0);
 nand  ( C0, w1, w1);
  XOR ( S0, A0, B0);
  XOR  (w2, A1, B1);
  XOR  ( S1, w2, C0);
 nand  (w3, A1, B1);
  nand  (w4, B1, C0);
 nand ( w5, A1, C0);
 nand ( C1, w3, w4, w5 );
 endmodule
```

OBSERVATION

Objective 1 :

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

Objective 2 :

A	B	Diff	Borrow
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

Objective 3 :

A1	A0	B1	B0	C0	S0	S1	C1
0	0	0	0	0	0	0	0
0	0	0	1	0	1	0	0
0	0	1	0	0	0	1	0
0	0	1	1	0	1	1	0
0	1	0	0	0	1	0	0
0	1	0	1	1	0	1	0
0	1	1	0	0	1	1	0
0	1	1	1	1	0	0	1
1	0	0	0	0	0	1	0
1	0	0	1	0	1	1	0
1	0	1	0	0	0	0	1
1	0	1	1	0	1	0	1
1	1	0	0	0	1	1	0
1	1	0	1	1	0	0	1
1	1	1	0	0	1	0	1
1	1	1	1	1	0	1	1

CONCLUSION

Objective 1 :

It can be concluded that a full adder can add 3 binary bit using 7486 and 7400 ICs.

Objective 2 :

It can be concluded that a half subtractor can subtract 2 binary bit using 7486 and 7400 ICs.

Objective 3 :

It can be concluded that a 2 bit parallel adder can add 2 2 binary bit binary number using 7486 and 7400 ICs.

IV. POST LAB

1. A Half-adder is characterized by

a. Two inputs and two outputs.

b. Three inputs and two outputs.

c. Two inputs and three outputs.

d. Two inputs and one outputs.

Ans. a) Two inputs and two outputs.

2. A 4-bit parallel adder can add.

a. Two 4-bit binary numbers

b. Two 2 bit binary numbers

c. Four bits at a time

d. Four bits at a time

Ans. a) Two 4-bit binary numbers

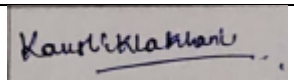
3. Two four bit numbers can be added using two full adders. Yes or NO ? Justify your answer.

Ans. No because to add two 4 bit numbers to produce 4 bit sum with possible carry four full adders are required.

DIGITAL LOGIC DESIGN LAB (EET1211)

LAB V: DESIGN AND TEST VARIOUS CODE CONVERTER CIRCUITS USING HDL

Siksha 'O' Anusandhan Deemed to be University, Bhubaneswar

Branch: CSIT			**Section:** D
S. No.	**Name**	**Registration No.**	**Signature**
1	Kaushik Lakhani	2041012002	

Marks: _____/10

Remarks:

Teacher's Signature

I. OBJECTIVE

1. Design a combinational circuit with four input lines that represent a decimal digit in BCD and four output lines that generate the 9's complement of the input digit.

2. Design a combinational circuit with four inputs and four outputs that converts a 4bit binary number into the equivalent 4bit Gray code.

3. Design a combinational circuit that accepts a 2-bit number and generates an output binary number equal to the square of the input number.

II. PRE-LAB

For objective-1:

Design a combinational circuit with four input lines that represent a decimal digit in BCD and four output lines that generate the 9's complement of the input digit.

a) Write the truth table for the circuit:

decimal digit	A	B	C	D		W	X	Y	Z
0	0	0	0	0		1	0	0	1
1	0	0	0	1		1	0	0	0
2	0	0	1	0		0	1	1	1
3	0	0	1	1		0	1	1	0
4	0	1	0	0		0	1	0	1
5	0	1	0	1		0	1	0	0
6	0	1	1	0		0	0	1	1
7	0	1	1	1		0	0	1	0
8	1	0	0	0		0	0	0	1
9	1	0	0	1		0	0	0	0

[10 – 15 are unused bit. They are considered as don't care conditions]

b) Derive the minimized boolean expression for each output of the circuit.

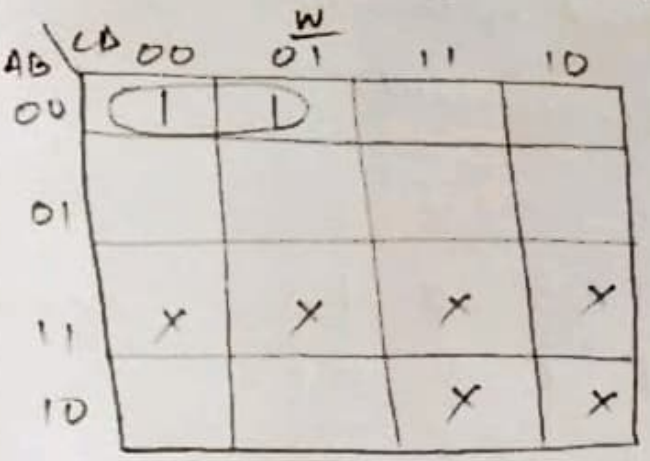
$W = \Sigma m(0,1)$

$X = \Sigma m(2,3,4,5)$

$Y = \Sigma m(2,3,6,7)$

$Z = \Sigma m(0,2,4,6,8)$

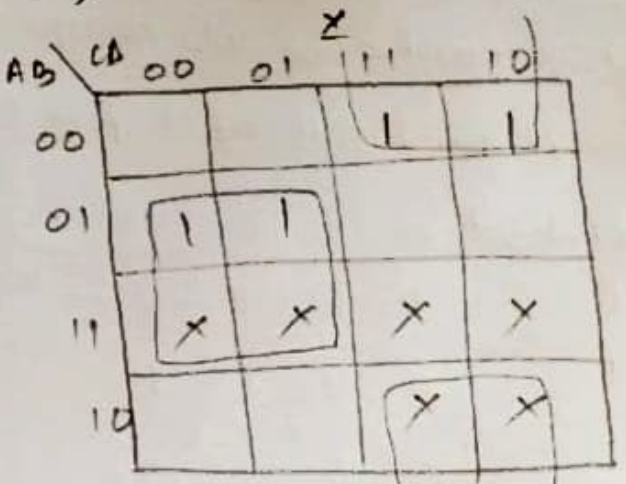
W



1 pair

$W = A'B'C'$

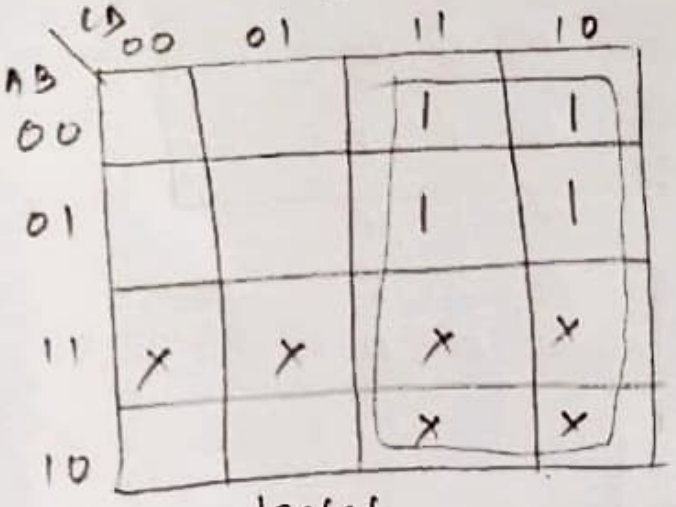
X



2 quads

$X = B \oplus C$

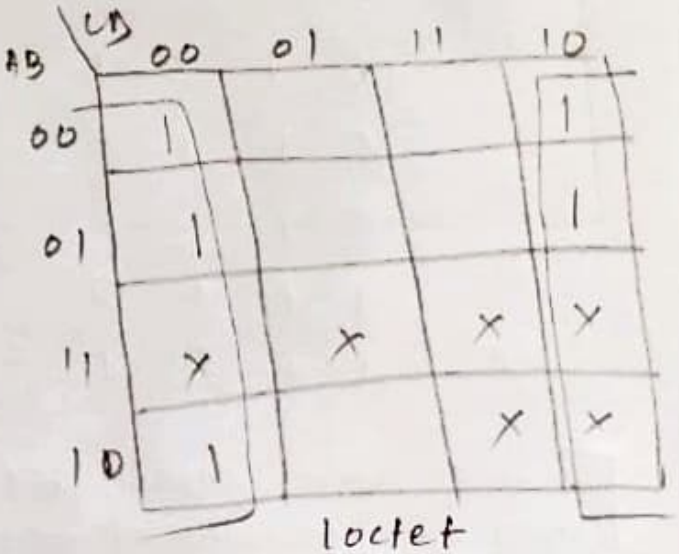
Y



1 octet

$Y = C$

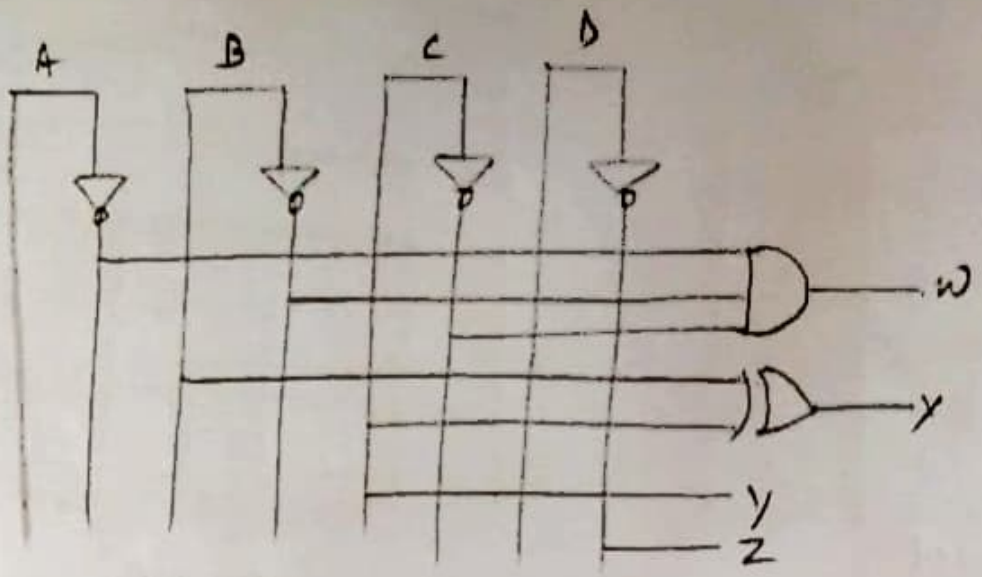
Z



1 octet

$Z = D'$

c) Draw the circuit diagram for the above.



For objective 2

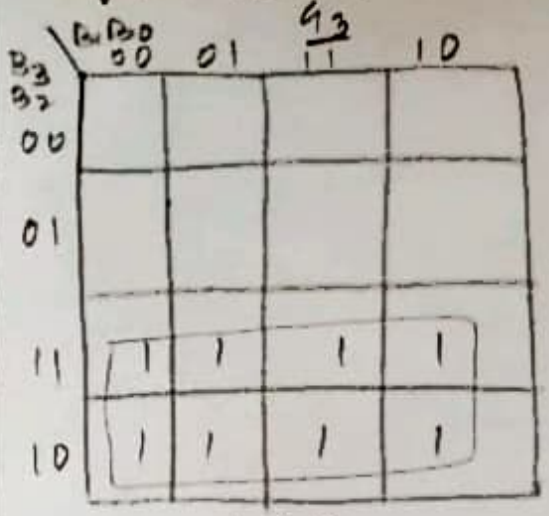
Design a combinational circuit with four input and four outputs that converts a 4bit binary number into the equivalent 4bit Gray code.

a) Write the truth table

B_3	B_2	B_1	B_0	G_3	G_2	G_1	G_0
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	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	1	1	0	0	0

b) Derive the minimized boolean expression for each output of the circuit.

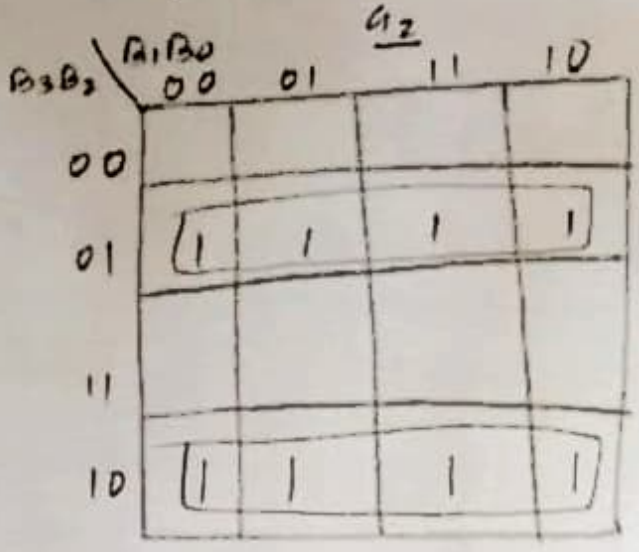
G_3



locfef

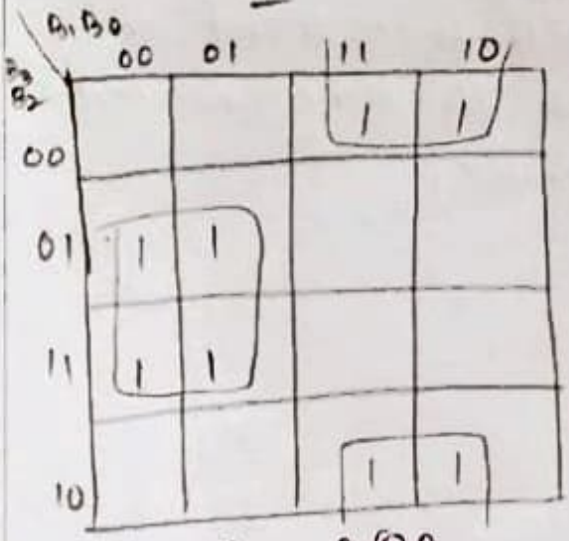
$G_3 = B_3$

G_2



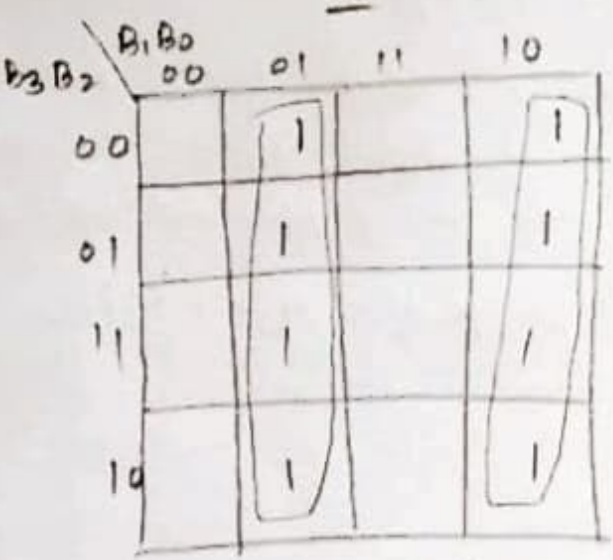
$G_2 = B_2 \oplus B_3$

G_1



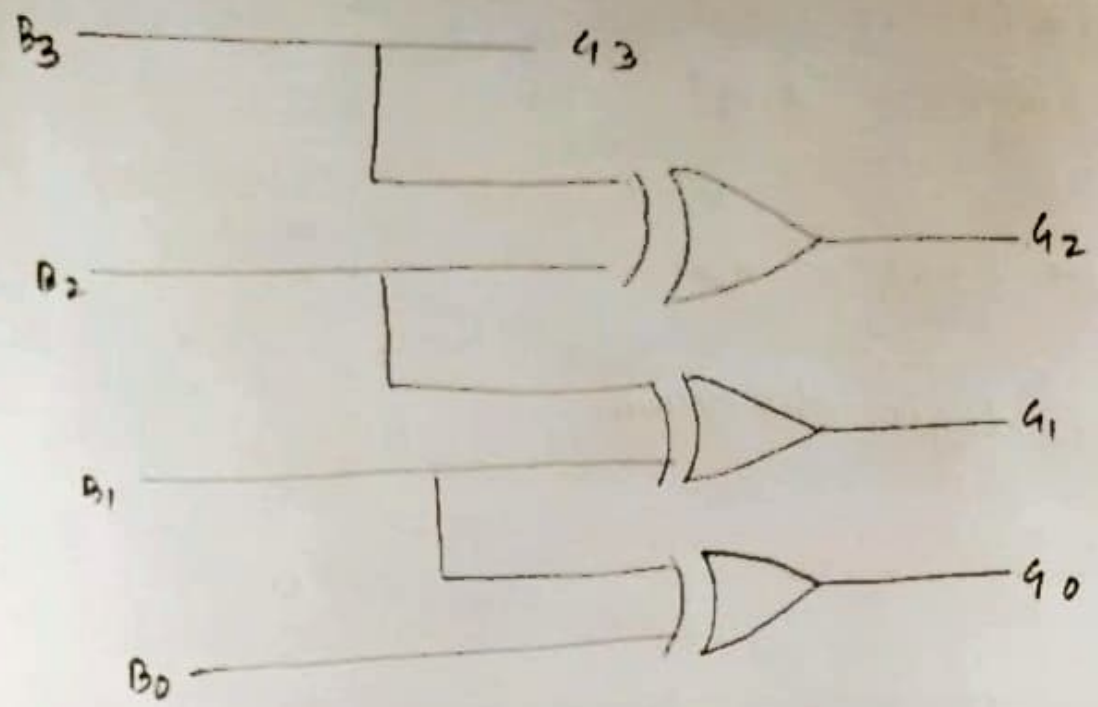
$G_1 = B_1 \oplus B_2$

G_0



$G_0 = B_0 \oplus B_1$

c) Draw the logic diagram.



For objective 3

Design a combinational circuit that accepts a 2-bit number and generates an output number equal to the square of the input number.

a) Write the truth table.

A_1	A_0	W	X	Y	Z
0	0	0	0	0	0
0	1	0	0	0	1
1	0	0	1	0	0
1	1	1	0	0	1

For objective 4.

Verify the operation of 7447IC to display 0 to 9 in a seven segment display device (LT 542).

a) Draw the picture of digits (0 to 9) as it appears on a 7-segment display device.

B_3	B_2	B_1	B_0	A	B	C	D	E	F	G
0	0	0	0	0	0	0	0	0	0	1
0	0	0	1	1	0	0	1	1	1	1
0	0	1	0	0	0	1	0	0	1	0
0	0	1	1	0	0	0	0	1	1	0
0	1	0	0	1	0	0	1	1	0	0
0	1	0	1	0	1	0	0	1	0	0
0	1	1	0	0	1	0	0	0	0	0
0	1	1	1	0	0	0	1	1	1	1
1	0	0	0	0	0	0	0	0	0	0
1	0	0	1	0	0	0	0	1	0	0

b) Derive the minimized Boolean expression for each output of the circuit.

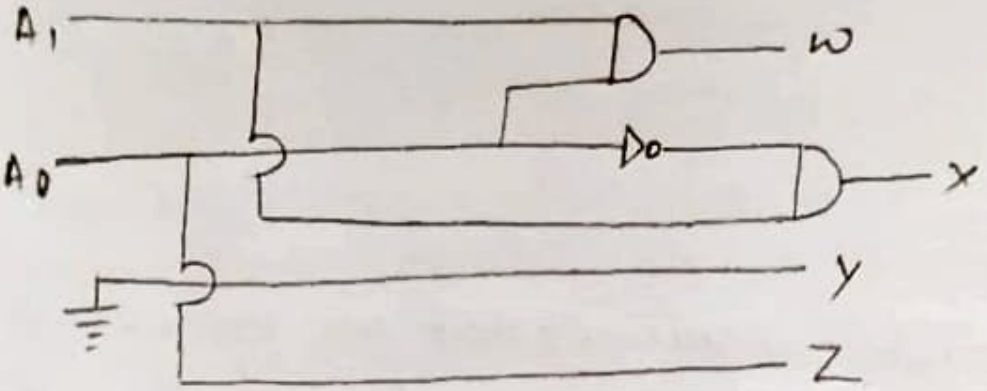
$W = \Sigma m(3) = A_1 A_0$

$X = \Sigma m(2) = A_1 A_0'$

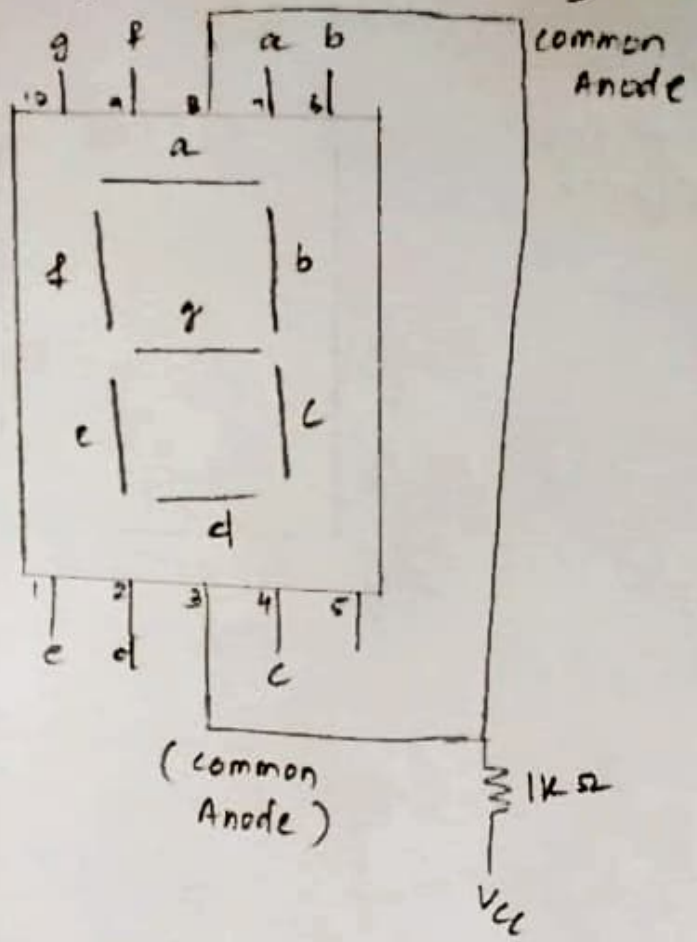
$Y = 0$

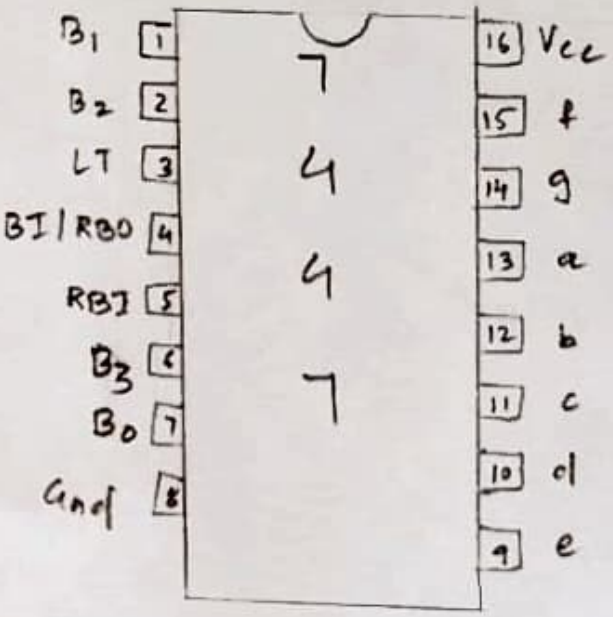
$Z = \Sigma m(1,3) = A_0$

c) Draw the logic diagram.



b) Draw the pin diagram of 7447IC and LT542 display device indicating the functions.





Conclusion

III. LAB

Components Required

SL No	components Name	specification	Quantity
1	IC-7408	quad 2 input AND gate	1
2	IC-7486	quad 2 input XOR gate	1
3	IC-7432	quad 2 input OR gate	1
4	IC-7404	hex inverter	1
5	IC-7447		1
6	LT-524		1
7	IC-7411	3 input AND gate	1
8	universal Trainer kit	microlab	1
9	wires	sw q	Ac. per Requirement

Observations

HDL Program

1) module first-objective (A, B, C, D, w, x, y, z);
output w, x, y, z;
input A, B, C, D;
wire w_1, w_2, w_3;
assign w = ((!A) && (!B) && (!C));
assign x = B ^ C;
assign y = C;
assign z = (!D);
endmodule

2) module second - objective ($B_3, B_2, B_1, B_0, g_3, g_2, g_1, g_0$);

output g_3, g_2, g_1, g_0 ;

input B_3, B_2, B_1, B_0 ;

~~wire~~ assign $g_3 = B_3$;

assign $g_2 = B_2 \wedge B_3$;

assign $g_1 = B_1 \wedge B_2$;

assign $g_0 = B_0 \wedge B_1$;

endmodule

3) module third - objective (A_1, A_0, w, x, y, z);

output w, x, y, z ;

input A_1, A_0 ;

wire w_1 ;

assign $w = (A_1 \,||\, A_0)$;

assign $x = (A_1 \,||\, (!A_0))$;

assign $y = 0$;

assign $z = A_0$;

endmodule

...re circu... ...node.

Conclusion

objective 1:

It can be concluded that to 9's complement of BCD input digit we need 3 AND gates and NOT Gates, 2 buffers and 1XOR Gate and circuit codes to the functions.

W: A'B'C'

X = B⊕C

Y = C

Z = D'

objective 2:

It can be concluded the required function is generated by taking the XOR of consecutive binary bits and a total of 3 XOR gates and a buffer is required to make the circuit.

~~W = AB~~

~~X = 1~~

objective 3:

It can be concluded that circuit which accept 2-bit number and generates an output binary number equal to the square of the input number leads to function.

W = A1A0;

X = A1A0';

Y = 0

Z = A0;

objective 4:
It is concluded that for a 7-segment display IC 7447 and LT524 is required.

POSTLAB:

1) Using the circuit you have designed to get 9's complement of 4 bit BCD input find the 9's complement of $(26)_{10}$.

Ans) $(26)_{10}$ = ~~0010 0110~~ $(11010)_2$

in BCD = 0010 0110

complement for 0110 as per circuit 0011

complement fore 0010 as per circuit 0111

Final expression of BCD is 0111 0011 = $(73)_{10}$

9's complement of $(26)_{10}$ is $(73)_{10}$

2) What is the advantage of gray code?

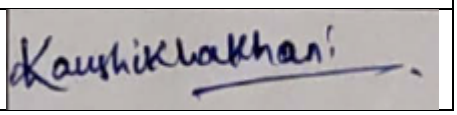
ans) In gray code, if we go from one decimal number to next, only one bit of the gray code changes. Because of this feature, an amount of switching is minimized and the reliability of the switching systems is improved.

Advantage of gray code over binary is only one bit changes for each step. This will be usefull in circuits that are sensitive to glitches.

DIGITAL LOGIC DESIGN LAB (EET1211)

LAB VI: DESIGN OF MAGNITUDE COMPARATOR, DECODER AND MULTIPLEXER CIRCUIT USING HDL

Siksha 'O' Anusandhan Deemed to be University, Bhubaneswar

Branch: Computer Science and Engineering			**Section:** D
S. No.	**Name**	**Registration No.**	**Signature**
6	Kaushik Lakhani	2041012002	

Marks: _____/10

Remarks:

Teacher's Signature

I. OBJECTIVE:

1. Design a combinational circuit for a 1 bit magnitude comparator.

2. Design a combinational circuit for a 2 X 1 Multiplexer that will select the binary information from one of the two input lines and direct it to a single output line based on the value of a selection line.

3. Design a combinational circuit for a full adder using 3 to 8 line decoder and external OR gates.

II. PRE-LAB

For Obj. 1:

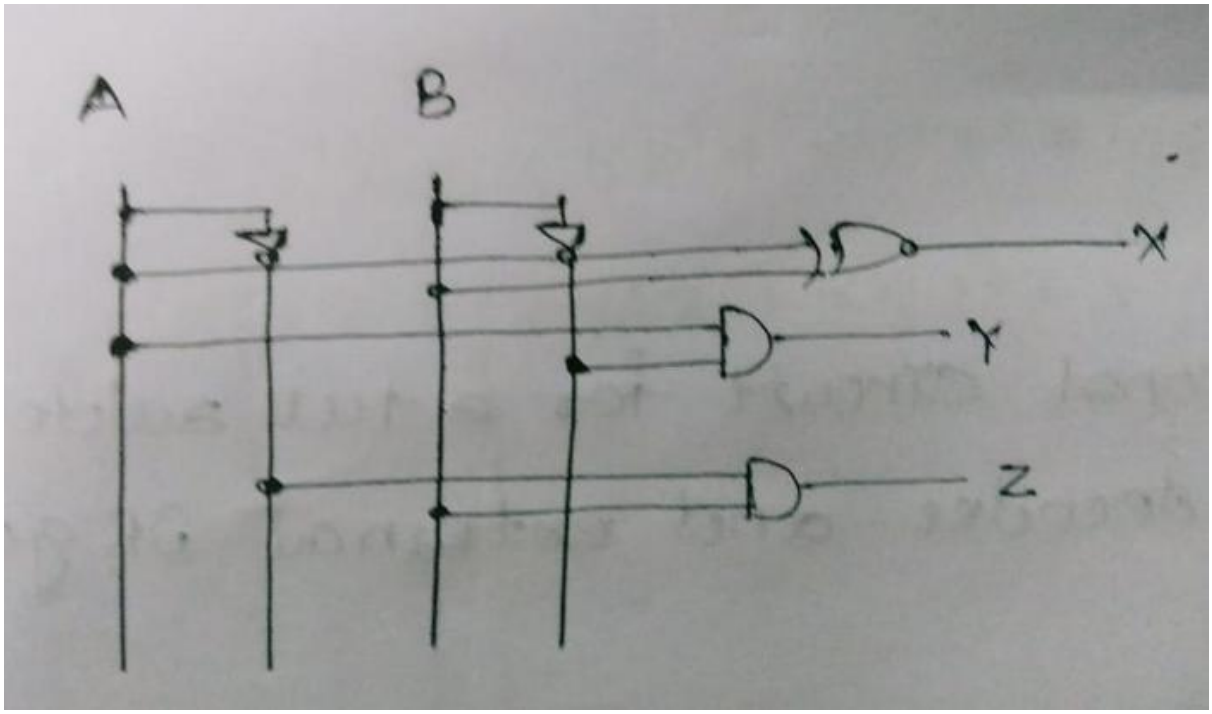
a) Write the truth table for the circuit.

A	B	X	Y	Z
0	0	1	0	0
0	1	0	0	1
1	0	0	1	0
1	1	1	0	0

b) Derive the Minimized Boolean expression for each output of the circuit.

Simplified Expression: **X = A ⊙ B**, **Y = AB'**, **Z = A'B**

c) Draw the logic diagram for the circuit.



d) Write HDL code.

design.sv:

```
`default_nettype none
module mod (
  input A,
  input B,
    output X,
  output Y,
    output Z
);
 // dataflow model
 assign X=~(A^B);
 assign Y= A&&~B;
 assign Z= ~A&&B;

 // gate-level model
 xnor(X,A,B);
```

and a(Y,A,~B),

a1(Z,~A,B);

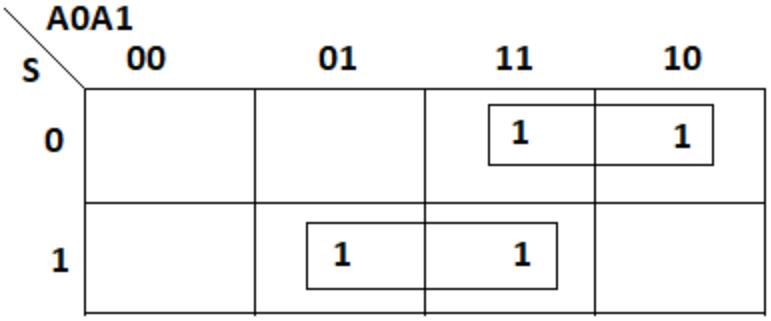
endmodule

For Obj. 2:

a) Write the truth table for the circuit.

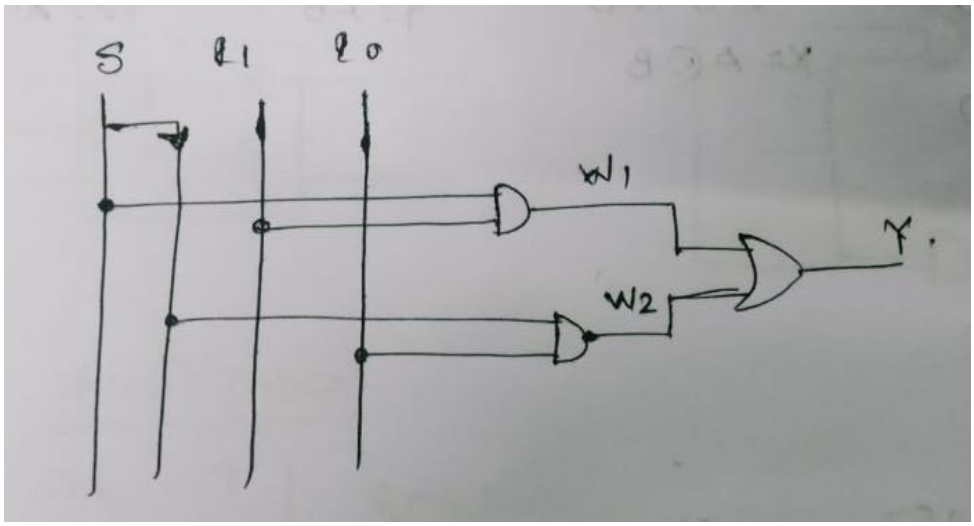
S	A0	A1	Y
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

b) Derive the Minimized Boolean expression for each output of the circuit.



Simplified Expression: **Y = S'A0 + SA1**

c) Draw the logic diagram for the circuit.



d) Write HDL code.

design.sv:

```
`default_nettype none
module mod (
  input S,A0,A1,
  output Y
);

 wire W1,W2;

 // dataflow model
 assign Y= ~S&&A0||S&&A1;

 // gate-level model
 and a(W1,~S,A0),
 a1(W2,S,A1);
```

or(Y,W1,W2);

endmodule

For Obj. 3:

a) Write the truth table for the circuit.

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

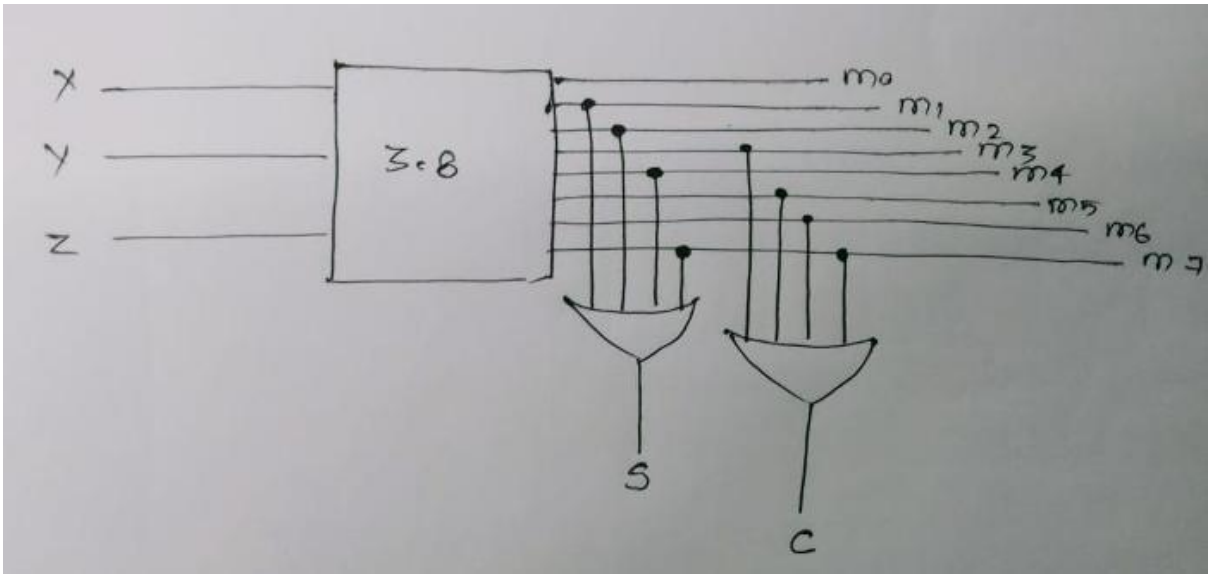
b) Derive the Minimized Boolean expression for each output of the circuit.

Expression:

Sum = **∑ m (1,2,4,7)**

Carry = **∑ m (3,5,6,7)**

c) Draw the logic diagram for the circuit.



d) Write HDL code.

design.sv:

```
`default_nettype none
module lab6 (A,B,C,Sum,Carry);
 input A,B,C;
 output Sum,Carry,D0,D1,D2,D3,D4,D5,D6,D7;

 wire P,Q,R,S,T,U,V;

 // dataflow model
 assign D0=~A&&~B&&~C;
 assign D1=~A&&~B&&C;
 assign D2=~A&&B&&~C;
 assign D3=~A&&B&&C;
 assign D4=A&&~B&&~C;
 assign D5=A&&~B&&C;
 assign D6=A&&B&&~C;
 assign D7=A&&B&&C;
 assign Sum=~A&&~B&&C||~A&&B&&~C||A&&~B&&~C||A&&B&&C;
```

```
assign Carry= B&&C||A&&C||A&&B;

// gate-level model
and a1(D0,~A,~B,~C),
a2(D1,~A,~B,C),
a3(D2,~A,B,~C),
a4(D3,~A,B,C),
a5(D4,A,~B,~C),
a6(D5,A,~B,C),
a7(D6,A,B,~C),
a8(D7,A,B,C),
a9(P,~A,~B,C),
a10(Q,~A,B,~C),
a11(R,A,~B,~C),
a12(S,A,B,C),
a13(T,B,C),
a14(U,A,C),
a15(V,A,B);
or o1(Sum,P,Q,R,S),
o2(Carry,T,U,V);

endmodule
```

III. LAB:

1) Design a combinational circuit for a 1 bit magnitude comparator.

HDL Program:

design.sv:

```
`default_nettype none
```

```
module mod (
   input A,
   input B,
        output X,
   output Y,
        output Z
);

 // dataflow model
 assign X=~(A^B);
 assign Y= A&&~B;
 assign Z= ~A&&B;

 // gate-level model
 xnor(X,A,B);
 and a(Y,A,~B),
 a1(Z,~A,B);

endmodule
```

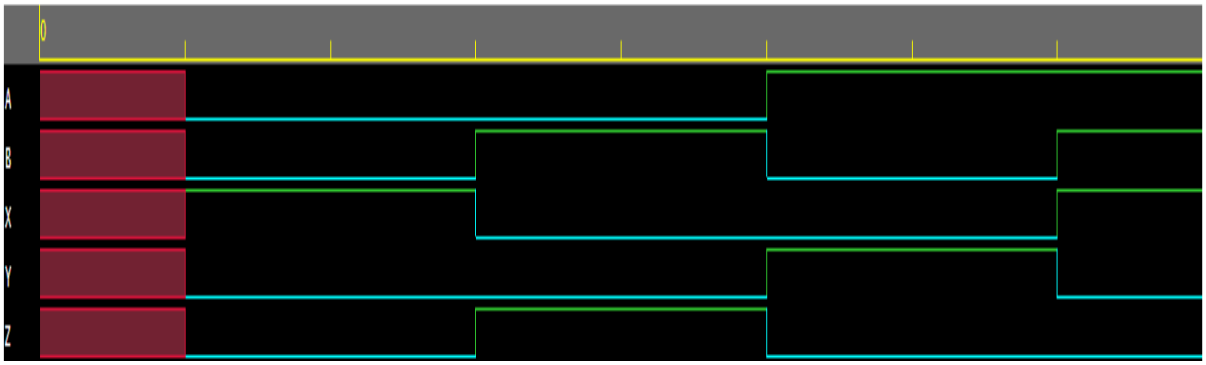
testbench.sv:

```
`default_nettype none
module tb_mod;
   reg a, b;
   wire x,y,z;
 mod h_dut(a,b,x,y,z);
   initial
      begin
         $dumpfile("dump.vcd");
         $dumpvars(0, h_dut);
```

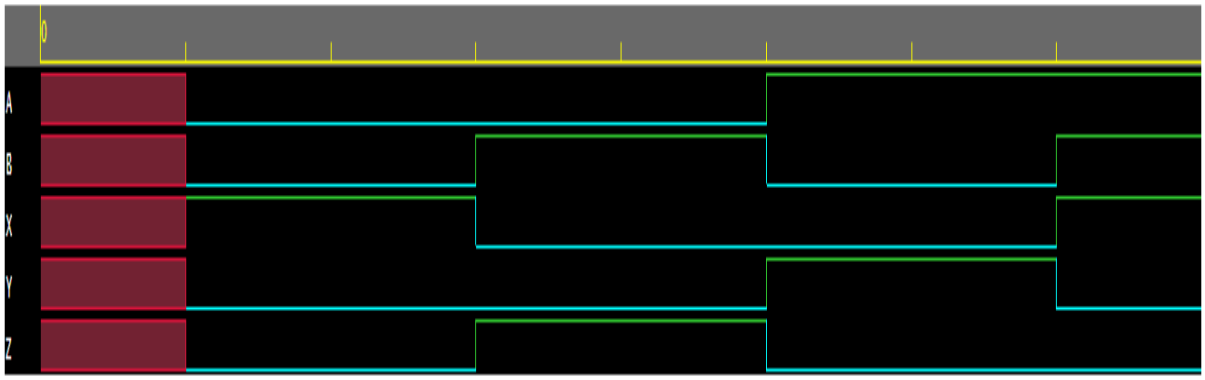
```
        $display("Lab 6 Obj 1");
        #1
        a <= 0;
        b <= 0;
        #1
        #1
        a <= 0;
        b <= 1;
        #1
        #1
        a <= 1;
        b <= 0;
        #1
        #1
        a <= 1;
        b <= 1;
        #1
        $finish();
    end
endmodule
```

Links: https://www.edaplayground.com/x/nGTL

EP Waveform:



Dataflow model



Gate-level model

Observation:

The following Truth table was obtained from the above EP Waveform.

A	B	X	Y	Z
0	0	1	0	0
0	1	0	0	1
1	0	0	1	0
1	1	1	0	0

2) Design a combinational circuit for a 2 X 1 Multiplexer that will select the binary information from one of the two input lines and direct it to a single output line based on the value of a selection line.

HDL Program:

design.sv:

```
`default_nettype none
module mod (
  input S,A0,A1,
  output Y
);

 wire W1,W2;

 // dataflow model
 assign Y= ~S&&A0||S&&A1;

 // gate-level model
 and a(W1,~S,A0),
 a1(W2,S,A1);
 or(Y,W1,W2);

endmodule
```

testbench.sv:

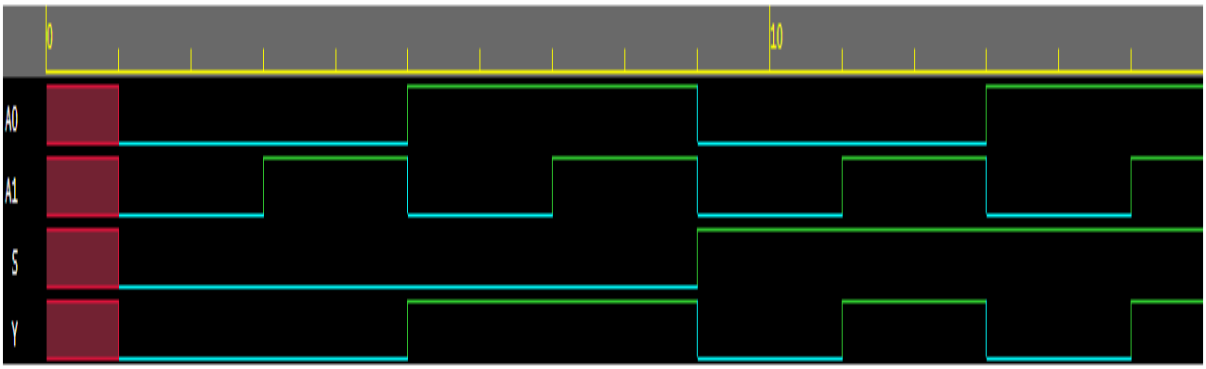
```
`default_nettype none
module tb_mod;
  reg S,A0,A1;
  wire Y;
 mod h_dut(S,A0,A1,Y);
  initial
    begin
      $dumpfile("dump.vcd");
      $dumpvars(0, h_dut);
```

```
$display("Lab 6 Obj 2");

 #1
 S <= 0;
A0 <= 0;
A1 <= 0;
 #1

 #1
 S <= 0;
 A0 <= 0;
A1 <= 1;
 #1

                #1
 S <= 0;
 A0 <= 1;
A1 <= 0;
 #1

 #1
 S <= 0;
 A0 <= 1;
A1 <= 1;
 #1

 #1
 S <= 1;
A0 <= 0;
```

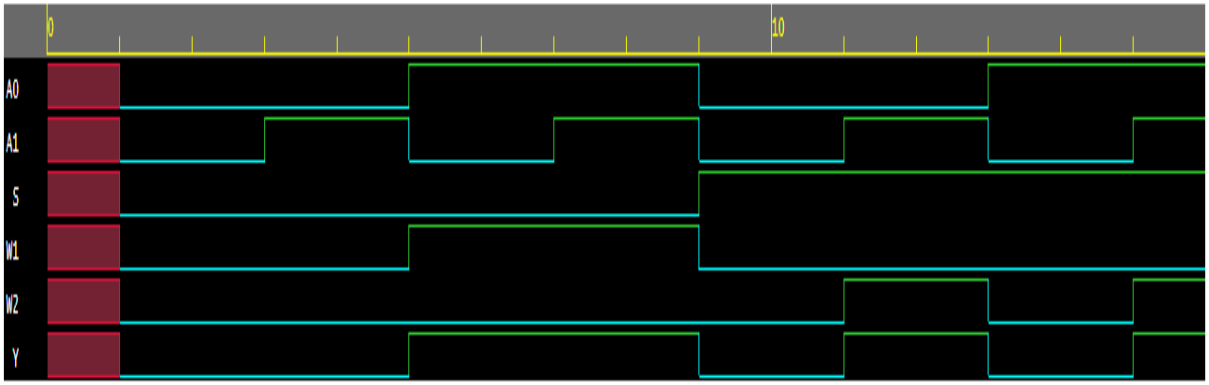
```
        A1 <= 0;
        #1

        #1
        S <= 1;
        A0 <= 0;
        A1 <= 1;
        #1

        #1
        S <= 1;
        A0 <= 1;
        A1 <= 0;
        #1

        #1
        S <= 1;
        A0 <= 1;
        A1 <= 1;
        #1

        $finish();
    end
endmodule
```

Links: https://www.edaplayground.com/x/CMsS

EP Waveform:



Dataflow model



Gate-level model

Observation:

The following Truth table was obtained from the above EP Waveform.

S	A0	A1	Y
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0

1	1	1	1

3) Design a combinational circuit for a full adder using 3 to 8 line decoder and external OR gates.

HDL Program:

design.sv:

```
`default_nettype none
module lab6 (A,B,C,Sum,Carry);
 input A,B,C;
 output Sum,Carry,D0,D1,D2,D3,D4,D5,D6,D7;

 wire P,Q,R,S,T,U,V;

 // dataflow model
 assign D0=~A&&~B&&~C;
 assign D1=~A&&~B&&C;
 assign D2=~A&&B&&~C;
 assign D3=~A&&B&&C;
```

```
    assign D4=A&&~B&&~C;
    assign D5=A&&~B&&C;
    assign D6=A&&B&&~C;
    assign D7=A&&B&&C;
    assign Sum=~A&&~B&&C||~A&&B&&~C||A&&~B&&~C||A&&B&&C;
    assign Carry= B&&C||A&&C||A&&B;

    // gate-level model
    and a1(D0,~A,~B,~C),
    a2(D1,~A,~B,C),
    a3(D2,~A,B,~C),
    a4(D3,~A,B,C),
    a5(D4,A,~B,~C),
    a6(D5,A,~B,C),
    a7(D6,A,B,~C),
    a8(D7,A,B,C),
    a9(P,~A,~B,C),
    a10(Q,~A,B,~C),
    a11(R,A,~B,~C),
    a12(S,A,B,C),
    a13(T,B,C),
    a14(U,A,C),
    a15(V,A,B);
    or o1(Sum,P,Q,R,S),
    o2(Carry,T,U,V);

  endmodule
```

testbench.sv:

```
  `default_nettype none
```

```
module dl_lab6;
   reg a,b,c;
   wire sum,carry;
  lab6 h_dut(a,b,c,sum,carry);
   initial
      begin
         $dumpfile("dump.vcd");
         $dumpvars(0, h_dut);
        $display("Lab 6 Obj 3");
         #1
         a <= 0;
         b <= 0;
         c <= 0;
         #1

         #1
         a <= 0;
         b <= 0;
         c <= 1;
         #1

         #1
         a <= 0;
         b <= 1;
         c <= 0;
         #1

         #1
         a <= 0;
```

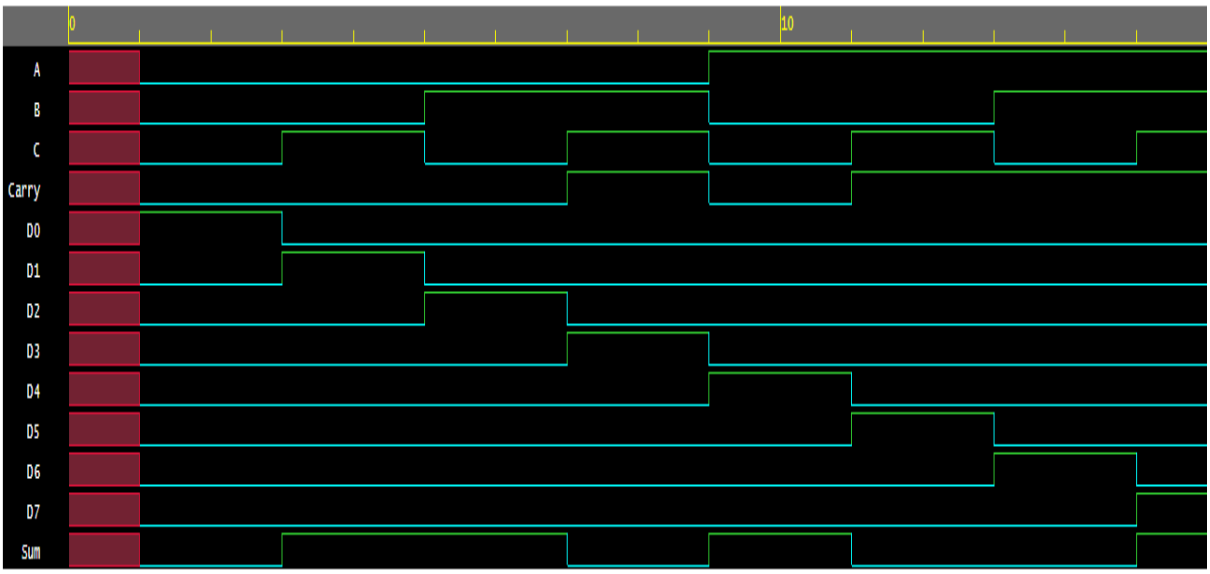
```
b <= 1;
c <= 1;
#1

#1
a <= 1;
b <= 0;
c <= 0;
#1

#1
a <= 1;
b <= 0;
c <= 1;
#1

#1
a <= 1;
b <= 1;
c <= 0;
#1


#1
a <= 1;
b <= 1;
c <= 1;
#1
$finish();
```

end

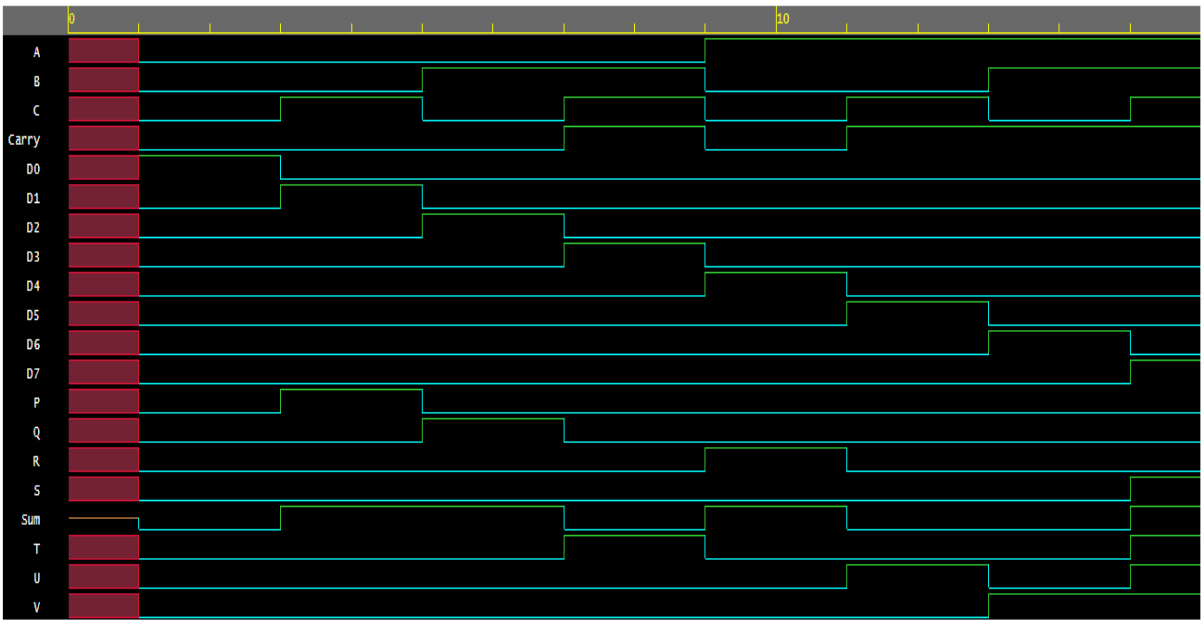
endmodule

Links: https://www.edaplayground.com/x/LN3N

EP Waveform:



Dataflow model



Gate-level model

Observation:

The following Truth table was obtained from the above EP Waveform.

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

Conclusion:

Objective 1: It can be concluded that to design a combinational circuit for a 1 - bit magnitude comparator 1 xnor and 2 and and not are required and circuit leads to the functions

X = A ⊙ B

Y = AB'

Z = A'B

Objective 2: It can be concluded that to design a combinational circuit for a 2 X 1 Multiplexer 2 and and 1 or gates are required and circuit leads to the function

Y=S'A0+SA1

Objective 3: It can be concluded that to implement a full adder using 3-to-8-line decoder and external OR gates a decoder and 2 or gates are required and circuit leads to function

Sum= ∑m (1,2,4,7)

Carry= ∑m (3,5,6,7)

IV. POST LAB:

1) **Logically derive the Boolean expressions for the output variables of a 2-bit magnitude comparator.**

Ans: - **A > B:** A1B1’ + A0B1’B0’ + A1A0B0’

A=B: (A0 ⊙ B0) (A1 ⊙ B1)

A<B: A1’B1 + A0’B1B0 + A1’A0’B0

2) **Why is a multiplexer known as data selector?**

Ans: - Multiplexer known as data selector because it selects which data input is to be sent.

V. HDL PROGRAM LINK:

Objective 1: https://www.edaplayground.com/x/nGTL

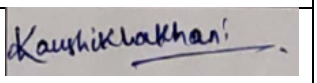
Objective 2: https://www.edaplayground.com/x/CMsS

Objective 3: https://www.edaplayground.com/x/LN3N

DIGITAL LOGIC DESIGN LAB (EET1211)

LAB VII: CONSTRUCT, TEST AND INVESTIGATE THE OPERATION OF VARIOUS FLIP-FLOP CIRCUITS USING HDL

Siksha 'O' Anusandhan Deemed to be University, Bhubaneswar

Branch: Computer Science and Information Technology **Section:** D			
S. No.	**Name**	**Registration No.**	**Signature**
7	**Kaushik Lakhani**	**2041012002**	

Marks: _____/10

Remarks:

Teacher's Signature

I. OBJECTIVE:

1. Design and test SR latch.

2. Design and test SR flip-flop.

3. Design and test JK flip-flop.

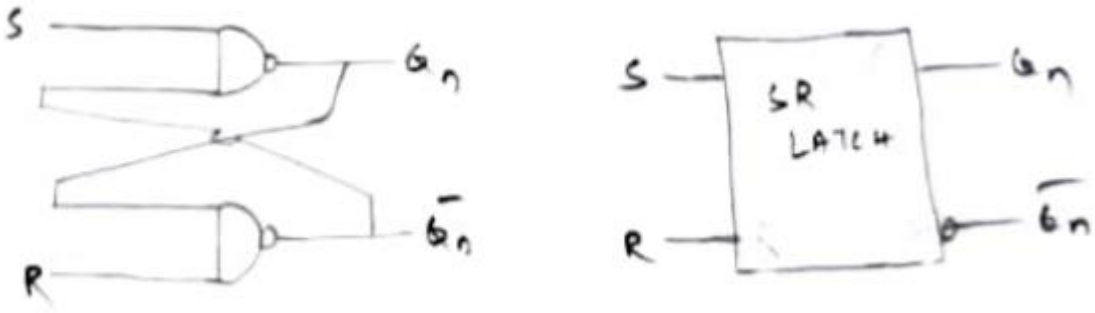
4. Realize the function of T flip flop using JK flip-flop.

II. PRE-LAB

For Obj. 1:

a. Obtain the characteristic table for the SR latch.
b. Draw the logic diagram for SR latch.
c. Write HDL code for SR latch.

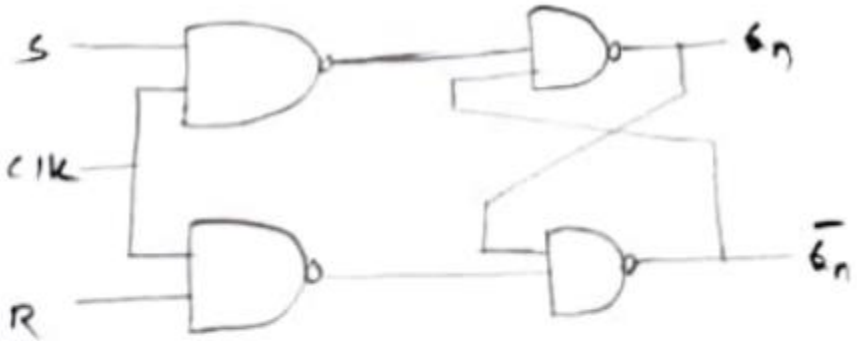
S	R	CLK	Q	Q'
0	0	0	X	X
0	1	1	0	1
1	0	1	1	0
1	1	1	0	0



For Obj. 2:
a. Obtain the characteristic table for the SR flip flop.
b. Draw the logic diagram for SR flip flop.

c. Write HDL code for SR flip flop.

Clk/En	Prev state (Qn)	S	R	Qn+1
□	*X*	*X*	*X*	Prev state maintained
↑	0	0	0	0(No change)
↑	0	0	1	0(Reset)
↑	0	1	0	1(Set)
↑	0	1	1	*X*($Q_{t+1} = Q'_{t+1}$)
↑	1	0	0	1(No change)
↑	1	0	1	0(Reset)
↑	1	1	0	1(Set)
↑	1	1	1	*X* (Race)

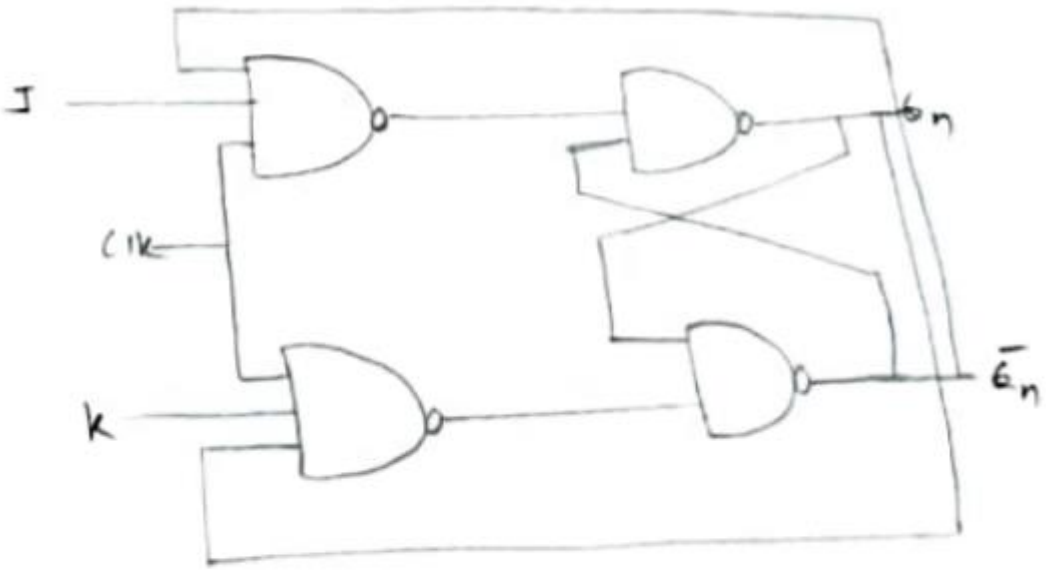


For Obj. 3:

a. Obtain the characteristic table for the JK flip flop.

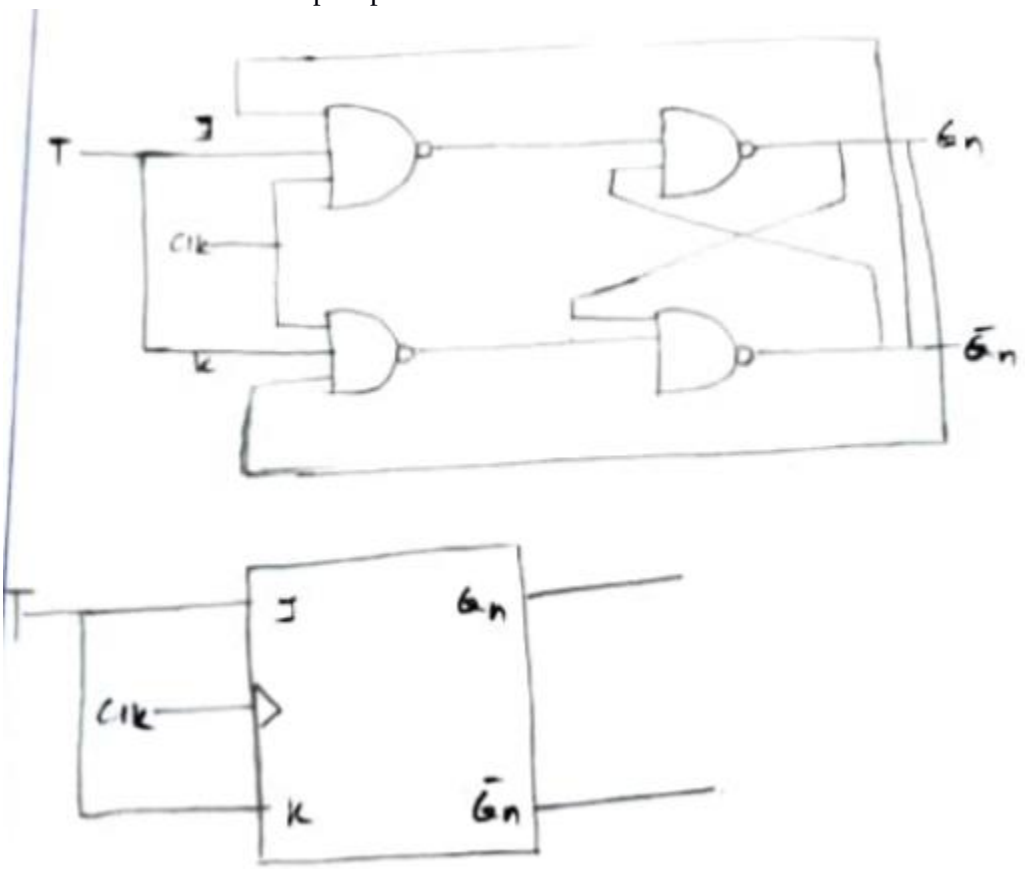
b. Draw the logic diagram for JK flip flop.
c. Write HDL code for JK flip flop.

CLK	J	K	Q_{t+1}
↑	0	0	Q_t (No change)
↑	0	1	0 (Reset)
↑	1	0	1 (Set)
↑	1	1	Q'_t (Toggle)



For Obj. 4:
a. Draw the logic diagram for T Flip flop using JK flip-flop.

b. Write HDL code for T flip flop.



c.

III. LAB:

HDL Program:

obj1:

```
module SR_LATCH(S, R, CLK, Q, Q');
                input S;
                input CLK; input R;

                input Q; inout Q'

                 assign Q=!(R||Q');

                  assign Q'=!(S||Q);

              endmodule
```

obj2:

HDL code for SR flip-flop:-

design.sv

```
`default_nettype none
module Obj2(output Q,Qbar,input S,R,clock,reset);
        reg Q;
        assign Qbar=!Q;
        always@(posedge clock or posedge reset)
    begin
        if(reset)
                Q = 0;
        else
                case({S,R})
                        2'b00: Q = Q;
                2'b01: Q = 0;
                2'b10: Q = 1;
                2'b11: Q = 1'bx;
                default: Q = 0;
                endcase
                end
endmodule
```

Obj 3

HDL Code for JK flip-flop:-

design.sv

```
`default_nettype none
module Obj3(output Q,Qbar,input J,K,clock,reset);
        reg Q;
        assign Qbar = !Q;
        always@(posedge clock or posedge reset) begin
   if(reset)
        Q = 0;
   else
        case({J,K})
        2'b00: Q = Q;
        2'b01: Q = 0;
        2'b10: Q = 1;
        2'b11: Q = !Q;
```

```
        default: Q = 0;
        endcase
        end
endmodule
```

Obj 4

HDL Code for T flip-flop using JK flip-flop:-

design.sv

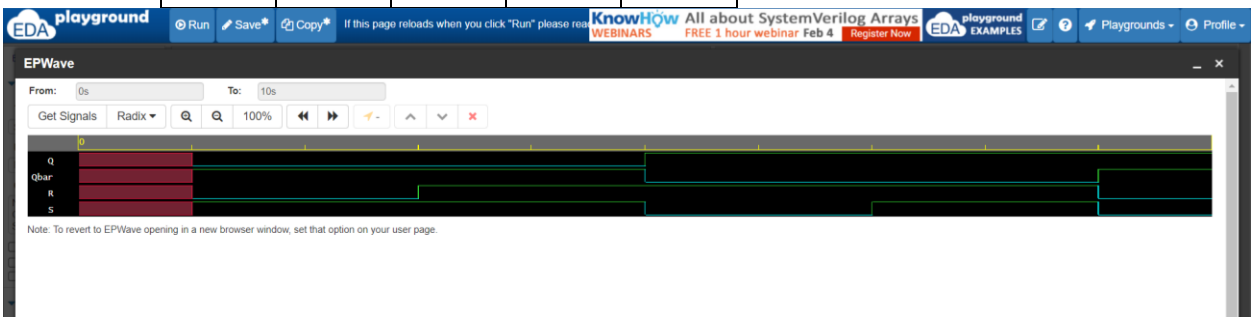
```
`default_nettype none
module Obj4(output Q,Qbar,input J,K,clock,reset);
        reg Q;
        assign Qbar = !Q;
        always@(posedge clock or posedge reset) begin
   if(reset)
        Q = 0;
   else
    case({J,K})
        2'b00: Q = 0;
        2'b01: Q = 1;
        2'b10: Q = 1;
        2'b11: Q = 0;
        default: Q = 0;
    endcase
 end
endmodule
```

Observation:

Objective 1

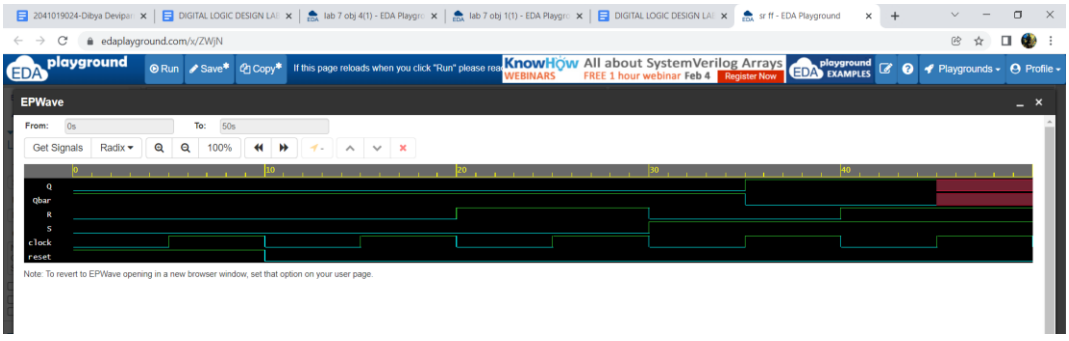
S	R	CLK	Q	Q’
0	0	0	X	X
0	1	1	0	1

1	0	1	1	0
1	1	1	0	0



Obj2:

R	S	CLK	Q	QBAR
0	0	1	X	X
0	1	1	1	0
1	0	1	0	1
1	1	1	1	0

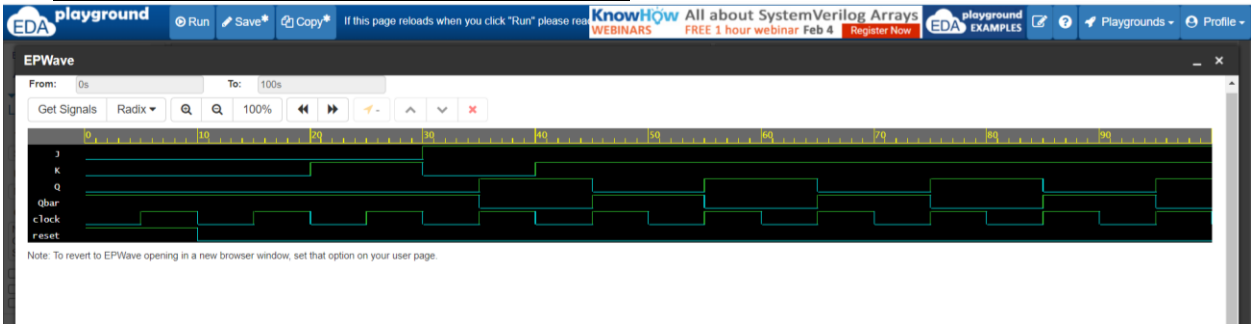


Objective 3

TRUTH-TABLE: -

J	K	CLK	Q	~Q

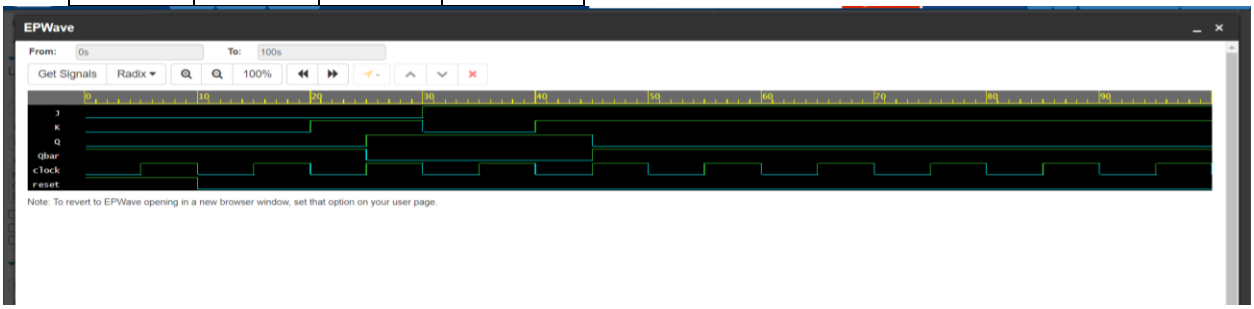
0	0	0	X	X
0	1	1	0	1
1	0	0	1	0
1	1	1	0	1



Objective 4

TRUTH-TABLE: -

T	CLK	Q	QBAR
0	**1**	**X**	**Z**
1	**1**	**X**	**Z**



Conclusion:

From the above experiment, it can be concluded that we have successfully designed, tested and investigated the working and function of SR latch, SR flip-flop, JK flip-flop and T flip-flop using JK flip-flop, all of these using HDL. We can use the above designed circuits in other circuits for memory storage and other different purposes.

IV. POST LAB:

1. Differentiate between a latch and a flip-flop.

LATCH	FLIP FLOP
These are building block of sequential circuit and these can be built from logic gates	These are also building blocks of sequential circuit but these can built from latches
It continuously check its input and changes its output correspondingly	It also does the same thing only at times determined by clocking signal
It is sensitive to the duration of the pulse and can send or receive data when switch is on	It is sensitive to signal change.They can transfer data only at the single instant and data cannot be changed until next signal change
It is based on enable function input	It works on the basis of clock pulses
It is a level triggered	It is an edge triggered

2. If both inputs of a SR NOR Latch are low, what will happen to the output?
   a) The output will reset.
   b) The output will toggle.
   c) The output will become unpredictable.
   d) The output will not be changed

   ANSWER:-(OPTION D) When both inputs of SR NOR latches are low the latch remains in its present state. There are no change in output.

3. If both inputs of a SR NAND Latch are low, what will happen to the output?
   a) The output will reset.
   b) The output will toggle.
   c) The output will become unpredictable.
   d) The output will not be changed

   ANSWER: - (OPTION C) When both inputs of SR NAND latches are low the latch the output will become unpredictable.

4. Which of the following describes the operation of a positive edge-triggered D-type flip-flop?
   a) If both inputs are high, the output will toggle.
   b) The input is toggled into the flip-flop on the leading edge of the clock and is passed to the output on the trailing edge of the clock.
   c) When both inputs are LOW, an invalid state exists.
   d) The output will follow the input on the leading edge of the clock.

   ANSWER: -
   (OPTION D) Edge-triggered flip-flop means the device will

change state during the rising or failing edge of the clock pulse. The main phenomenon of the D Flip-Flop is that the outputs will follow the input when the enable pin is high.

V. HDL PROGRAM LINK:

Objective 1:https://www.edaplayground.com/x/eYHh
Objective 2:https://www.edaplayground.com/x/ZWjN
Objective 3:https://www.edaplayground.com/x/ri6x
Objective 4:https://www.edaplayground.com/x/C9DE