Lab Report #4

Digital Logic Design

Submitted By:

Haider Ali (FA21-BEE-053)

Malik Ammar Murtaza (FA21-BEE-078)

Muhammad Ahmad (FA21-BEE-095)

Submitted to:

Dr. Adnan Qureshi

Lab#4 Design and Implementation of Boolean Functions by standard Forms using ICs/Verilog

Equipment Required:

- Breadboard
- Xilinx
- Logic Gate IC's

Background Theory:

For the design and implementation of Sum of Products and Product of Sums first the truth tables for min terms and max terms are created along with all of its possible outcomes.

Min terms are made in such a way that outcome will be 1 if all bits are 1 by taking product of these bits and if we sum all min terms in an expression then Sum of Product is obtained.

On the other hand, max terms are made in such a way that outcome should be 0 if all bits are 0 by taking sum of these bits and if take product of all min terms in an expression then Product of Sums is obtained.

The equations can be simplified and implemented using Xilinx.

Lab Tasks:

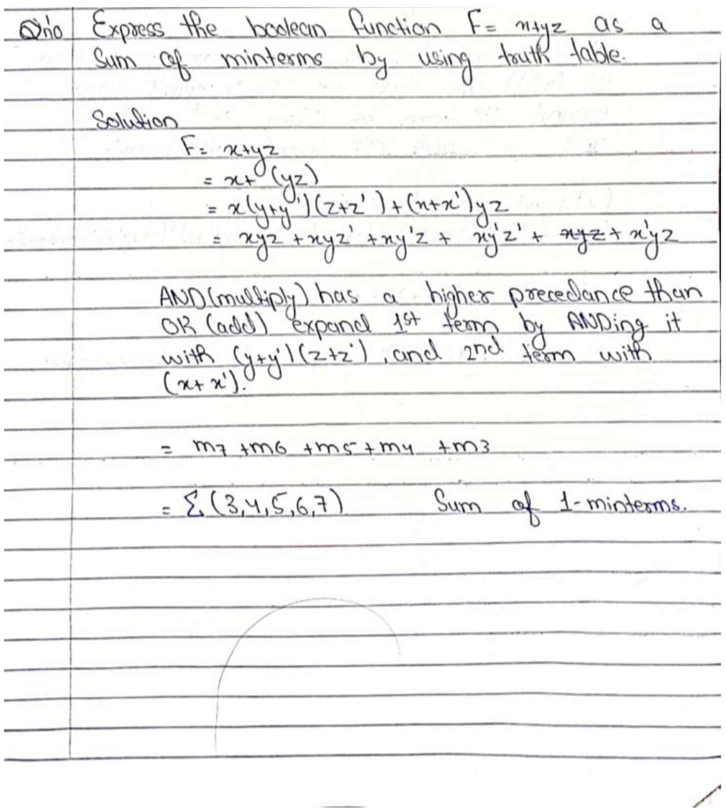
Boolean expression from truth table:

Procedure:

- Create truth table.
- Consider 1's in output as min terms and create SOP equation from table.
- Similarly create a POS equation from 0's.

- ➢ Simplify equations using k-map method.

Pre lab Tasks:



Q.no Express the boolean function $F = x+yz$ as a sum of minterms by using truth table.

Solution

$F = x+yz$

$= x + (yz)$

$= x(y+y')(z+z') + (x+x')yz$

$= xyz + xyz' + xy'z + xy'z' +$ ~~xyz~~ $+ x'yz$

AND(multiply) has a higher precedance than OR (add) expand 1st term by ANDing it with $(y+y')(z+z')$, and 2nd term with $(x+x')$.

$= m_7 + m_6 + m_5 + m_4 + m_3$

$= \Sigma(3,4,5,6,7)$ Sum of 1-minterms.

Qno^2 Express $F' = (x+yz)'$ as a product of maxterms.

Solution

$$F' = (x+yz)'$$
$$= (x+(yz))'$$
$$= x'(y'+z')$$

AND (multiply) has a higher precedance than OR (add) use dual or De Morgan's Law expand 1st term by ORing it with yy' and zz', and 2nd term with xx'.

$(x'+yy'+zz')(xx'+y'+z')$
$(x'+y+z)(x'+y+z')(x'+y'+z)(x'+y'+z')(x+y'+z')$~~$(x'+y'+z')$~~

$M_4 \cdot M_6 \cdot M_5 \cdot M_7 \cdot M_3$

$= \Pi(3,4,5,6,7)$ product of 1-maxterms.

Q no 3 Given the function as defined in the truth table (Table 4.4), express F using sum of minterms and product of maxterms.

Minterms F:-

$\Rightarrow \underset{m_1}{x'y'z} + \underset{m_3}{x'yz} + \underset{m_4}{xy'z'}$

$= \sum(m_1, m_3, m_4)$
$= \sum(1,3,4)$

Now reducing

$x'z(y+y') + xy'z'$ $\quad \because x+x'=1$
$= x'z + xy'z'$ $\quad x+x' = x$

Maxterms F:-

$\Rightarrow \underset{M_0}{(x+y+z)} \underset{M_2}{(x+y'+z)} \underset{M_5}{(x'+y+z')} \underset{M_6}{(x'+y'+z)} \underset{M_7}{(x'+y'+z')}$

$= \Pi(M_0 \cdot M_2 \cdot M_5 \cdot M_6 \cdot M_7)$
$\Pi(0, 2, 5, 6, 7)$

In lab Tasks:

Circuit Implementation

1. First, make the circuit diagram of the given task .
2. Select appropriate logic gate IC's which are needed.
3. Make connections according to the circuit diagram you made.
4. Connect the input to data switches and output to the logic indicator
5. Follow the input sequence and record the output.

TASK: Implement the circuit for the given function "F". Function's output is given in the table 4.5. Find its Boolean expression in SOP and POS forms.

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

Boolean Equations:

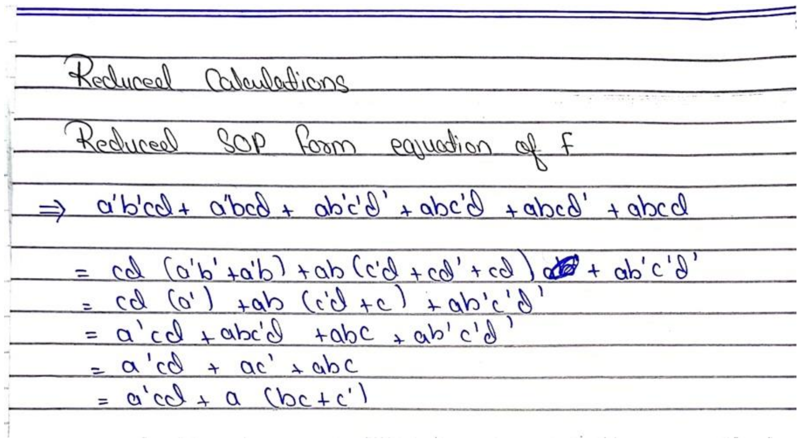
Sum of min terms equation of F: a’b’cd + a’bcd + ab’c’d’+ abc’d + abcd’+ abcd

Reduced SOP form equation of F: a’cd + acd’+abc

Product of max-terms equation of F:
(a+b+c+d)(a+b+c+d’)(a+b+c’+d)(a+b’+c+d)(a+b’+c+d’)(a+b’+c’+d)(a’+b+c+d’)(a’+b+c’+d)(a’+b+c’+d’)(a’+b’+c+d)

Reduced POS form equation of F: (A+C)(B+C)(A+D)

Reduced Form Calculation:



Reduceel Calculations

Reduceel SOP form equation of F

$\Rightarrow$ $a'b'cd + a'bcd + ab'c'd' + abc'd + abcd' + abcd$

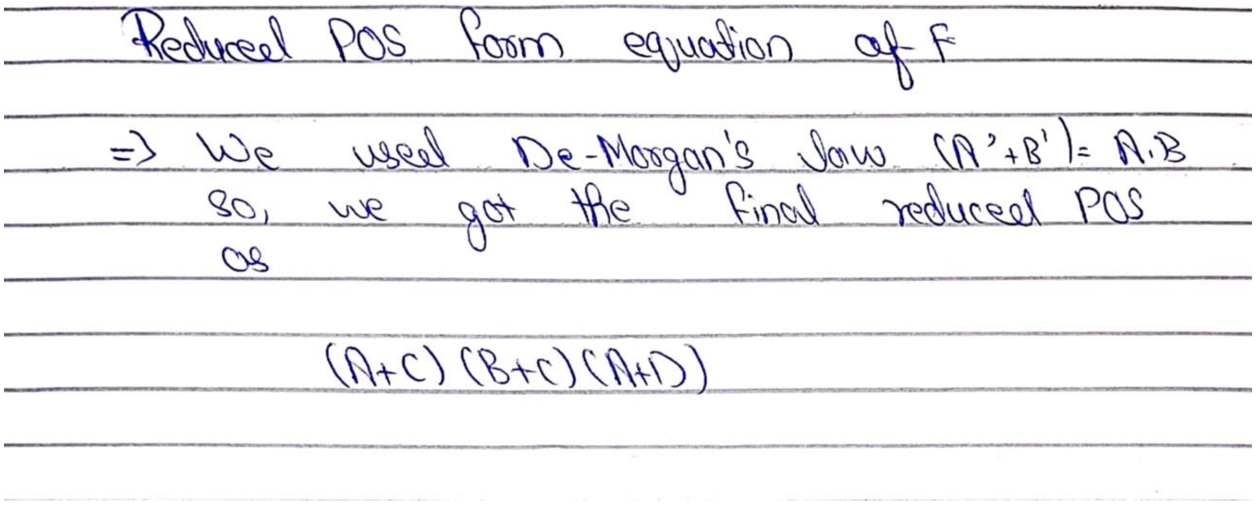
$= cd\,(a'b' + a'b) + ab\,(c'd + cd' + cd)$ + $ab'c'd'$

$= cd\,(a') + ab\,(c'd + c) + ab'c'd'$

$= a'cd + abc'd + abc + ab'c'd'$

$= a'cd + ac' + abc$

$= a'cd + a\,(bc + c')$



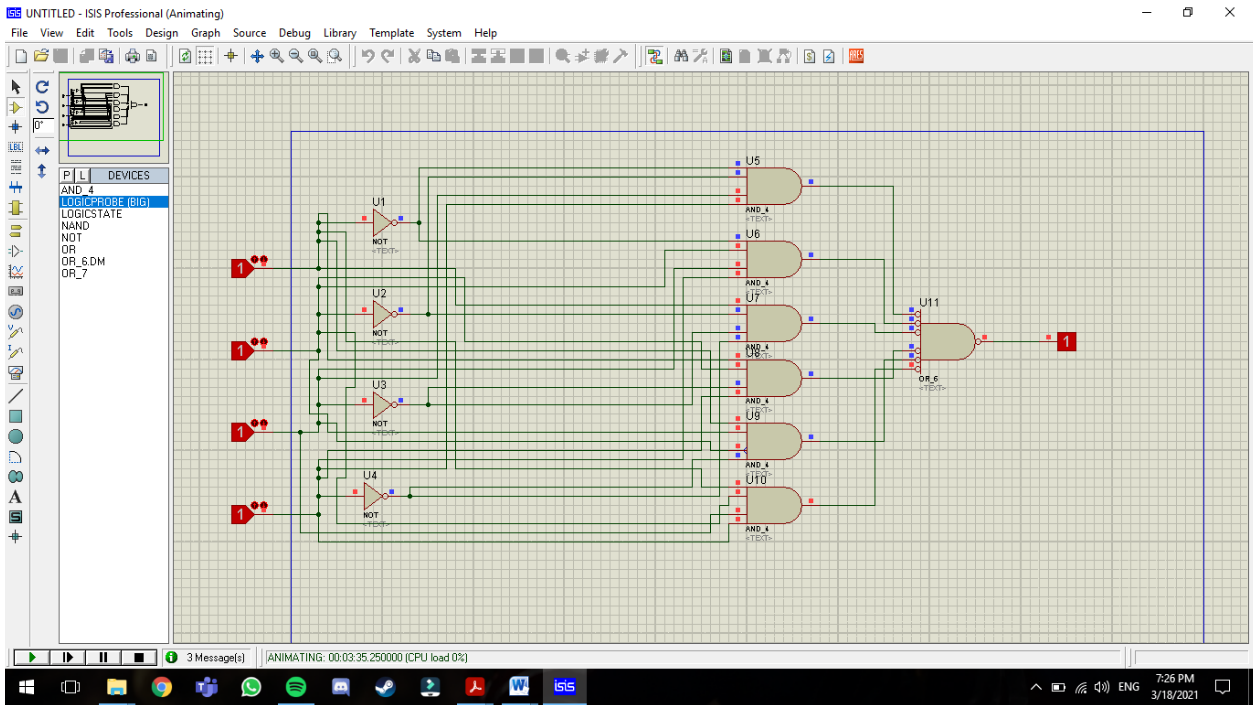
Reduceel POS form equation of F

$\Rightarrow$ We useel De-Morgan's law $(A' + B') = A.B$
So, we got the final reduceel POS as

$(A+C)(B+C)(A+D)$

Circuit Diagrams

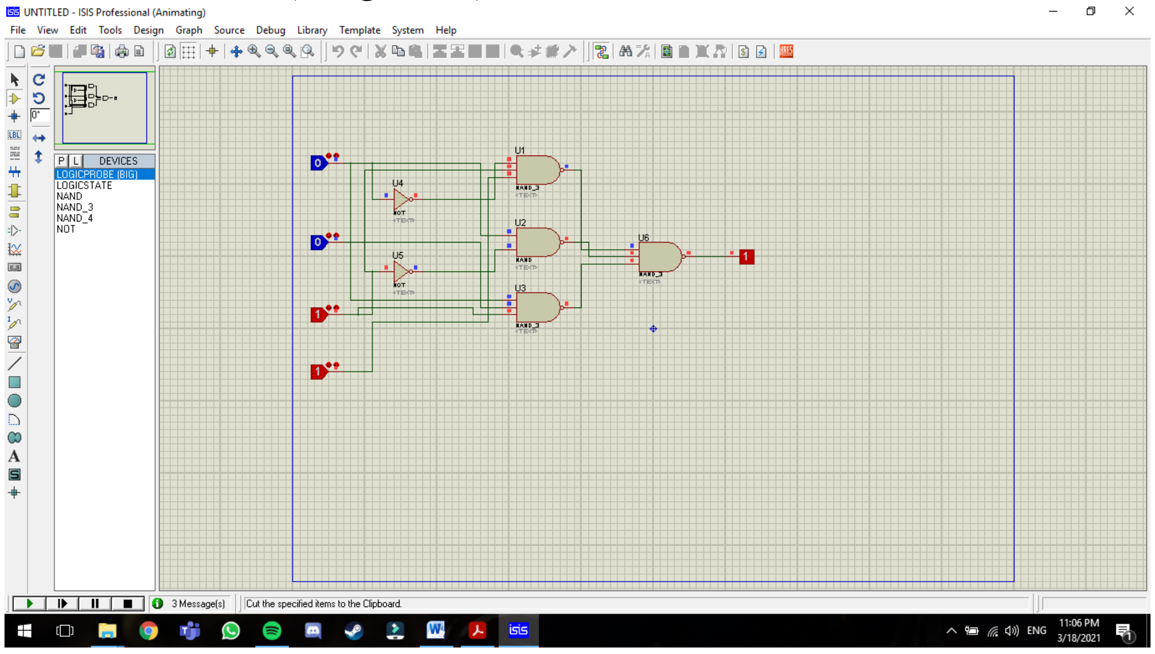
1. Sum of Min-Terms form:


UNTITLED - ISIS Professional (Animating)
File View Edit Tools Design Graph Source Debug Library Template System Help
DEVICES
AND_4
LOGICPROBE (BIG)
LOGICSTATE
NAND
NOT
OR
OR_6.DM
OR_7
U1
U2
U3
U4
NOT
U5
U6
U7
U8
U9
U10
AND_4
U11
OR_6
1
7:26 PM
3/18/2021
ENG

3 Message(s)

ANIMATING: 00:03:35.250000 (CPU load 0%)

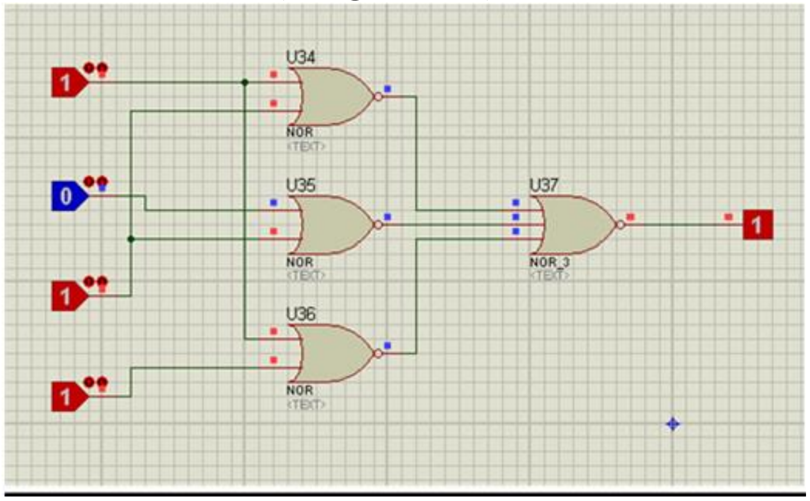
2. Reduced SOP Form (Using NAND):


UNTITLED - ISIS Professional (Animating)
File View Edit Tools Design Graph Source Debug Library Template System Help
DEVICES
LOGICPROBE (BIG)
LOGICSTATE
NAND
NAND_3
NAND_4
NOT
U1
U2
U3
U4
U5
U6
NAND_3
NOT
0
1
11:06 PM
3/18/2021
ENG

3 Message(s)

Cut the specified items to the Clipboard.

3. Reduced POS form (Using NOR IC):



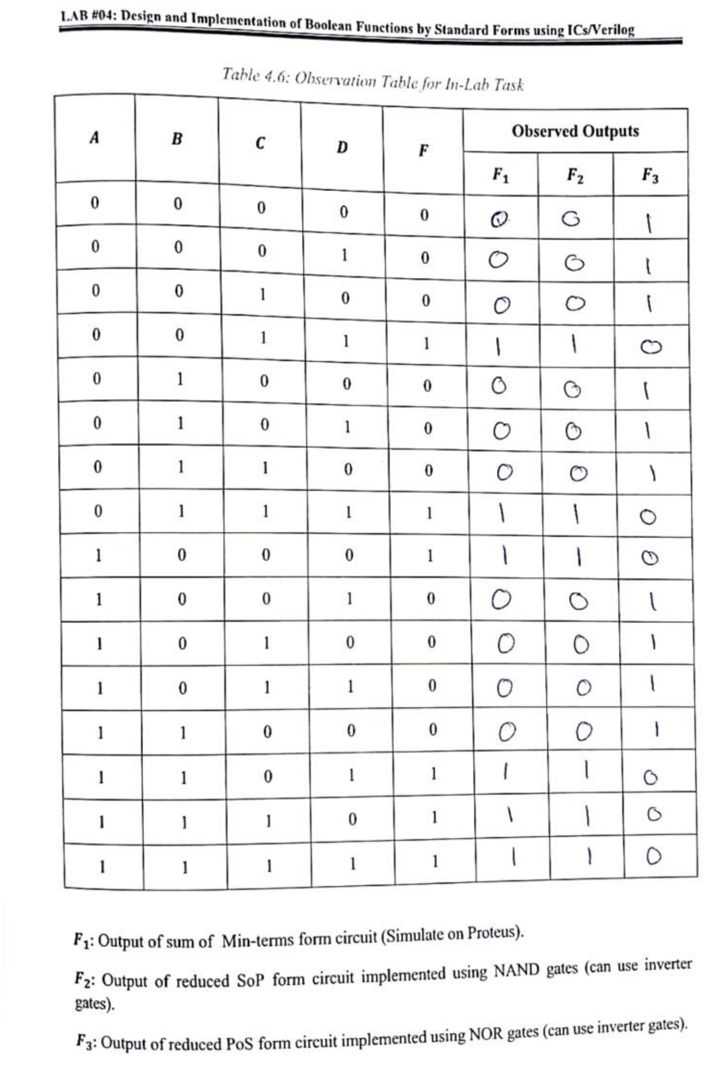


Table 4.6: Observation Table for In-Lab Task

A	B	C	D	F	Observed Outputs		
					F_1	F_2	F_3
0	0	0	0	0	0	0	1
0	0	0	1	0	0	0	1
0	0	1	0	0	0	0	1
0	0	1	1	1	1	1	0
0	1	0	0	0	0	0	1
0	1	0	1	0	0	0	1
0	1	1	0	0	0	0	1
0	1	1	1	1	1	1	0
1	0	0	0	1	1	1	0
1	0	0	1	0	0	0	1
1	0	1	0	0	0	0	1
1	0	1	1	0	0	0	1
1	1	0	0	0	0	0	1
1	1	0	1	1	1	1	0
1	1	1	0	1	1	1	0
1	1	1	1	1	1	1	0

F_1: Output of sum of Min-terms form circuit (Simulate on Proteus).

F_2: Output of reduced SoP form circuit implemented using NAND gates (can use inverter gates).

F_3: Output of reduced PoS form circuit implemented using NOR gates (can use inverter gates).

Implement Boolean expression using Xilinx:

Procedure:

- Create project in Xilinx.
- Specify inputs and outputs in main module.
- Implement the Boolean expression using gates.
- Give final output as output function variable
- Test wave form.

F1 Minterms Main method code:

```
module Main(
   input a,
   input b,
   input c,
   input d,
   output F1
   );
wire w1,w2,w3,w4,w5,w6,w7,w8,w9,w10,w11,w12,w13,w14;

              not n1(w1, a);
              not n2(w2, b);
              not n3(w3, a);
              not n4(w4, b);
              not n5(w5, c);
              not n6(w6, d);
              not n7(w7, c);
              not n8(w8, d);

              and a1(w9, w1,w2,c,d);
              and a2(w10, w3,b,c,d);
              and a3(w11, a,w4,w5,w6);
              and a4(w12, a,b,w7,d);
              and a5(w13, a,b,c,w8);
              and a6(w14, a,b,c,d);

endmodule
```

Test Bench:

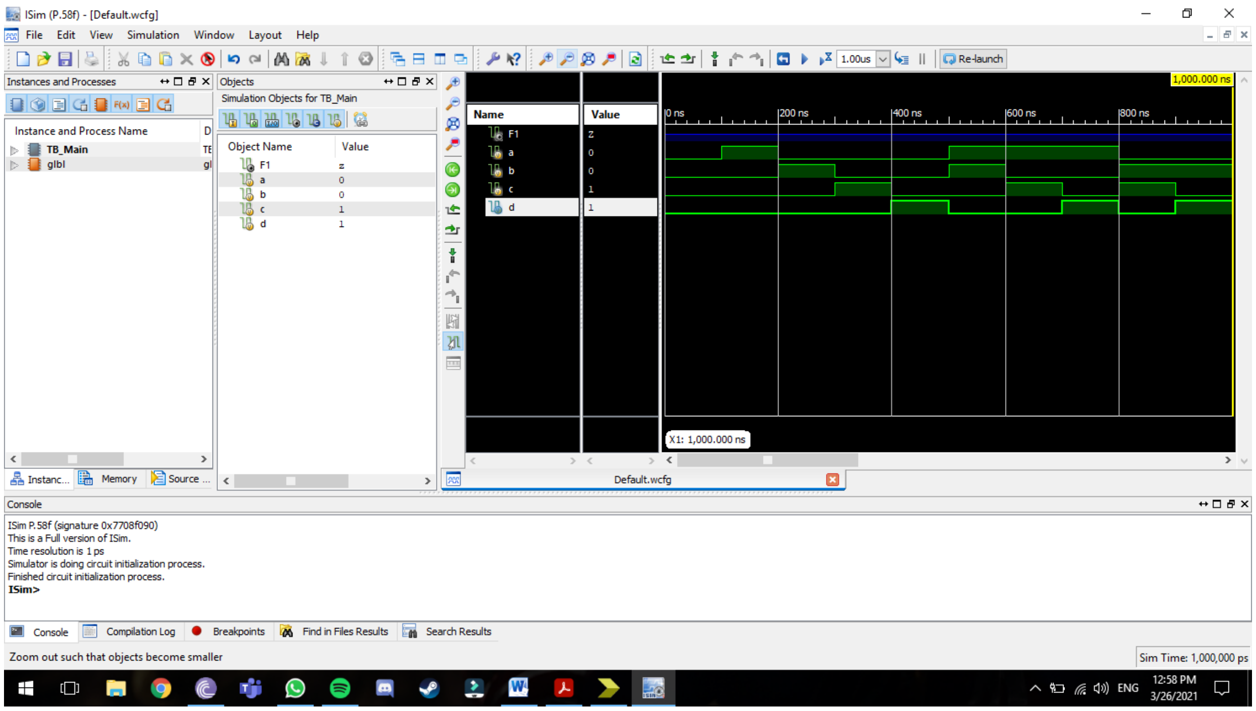
```
module TB_Main;

	// Inputs
	reg a;
	reg b;
	reg c;
	reg d;

	// Outputs
	wire F1;

	// Instantiate the Unit Under Test (UUT)
	Main uut (
		.a(a),
		.b(b),
		.c(c),
		.d(d),
		.F1(F1)
	);

	initial begin
		// Initialize Inputs

		  a = 0;
		b = 0;
		c = 0;
		d = 0;


		#100;
		a = 1;
		b = 0;
		c = 0;
		d = 0;
```

```
#100;
a = 0;
b = 1;
c = 0;
d = 0;

#100;
a = 0;
b = 0;
c = 1;
d = 0;

#100;
a = 0;
b = 0;
c = 0;
d = 1;

#100;
a = 1;
b = 1;
c = 0;
d = 0;

#100;
a = 1;
b = 0;
c = 1;
d = 0;

#100;
```

```
a = 1;

b = 0;

c = 0;

d = 1;


#100;

a = 0;

b = 1;

c = 1;

d = 0;


#100;

a = 0;

b = 1;

c = 0;

d = 1;


#100;

a = 0;

b = 0;

c = 1;

d = 1;


#100;

a = 1;

b = 1;

c = 1;

d = 0;


#100;

a = 1;
```

```
                b = 1;

                c = 0;

                d = 1;



                #100;

                a = 0;

                b = 1;

                c = 1;

                d = 1;



                #100;

                a = 1;

                b = 0;

                c = 1;

                d = 1;



                #100;

                a = 1;

                b = 1;

                c = 1;

                d = 1;



                #100;



        end


endmodule
```

Behavior Test:


ISim (P.58f) - [Default.wcfg]
File Edit View Simulation Window Layout Help
Re-launch
Instances and Processes
Objects
Simulation Objects for TB_Main
Default.wcfg
X1: 1,000,000 ns
1,000,000 ns

Instance and Process Name
TB_Main
glbl

Object Name	Value
F1	z
a	0
b	0
c	1
d	1

```
ISim P.58f (signature 0x7708f090)
This is a Full version of ISim.
Time resolution is 1 ps
Simulator is doing circuit initialization process.
Finished circuit initialization process.
ISim>
```

Zoom out such that objects become smaller

Sim Time: 1,000,000 ps

F3 SOP Method Code:

```
module Main(
    input a,
    input b,
    input c,
    input d,
    output F3
    );
                    wire w1,w2,w3,w4,w5,w6;
        not n1(w1,a);
        not n2(w2,c);

        and a1(w3,w1,c,d);
        and a2(w4,a,b,c);
        and a3(w5,a,w2,d);

        or o1(F3,w3,w4,w5);


endmodule
```

Test Bench:

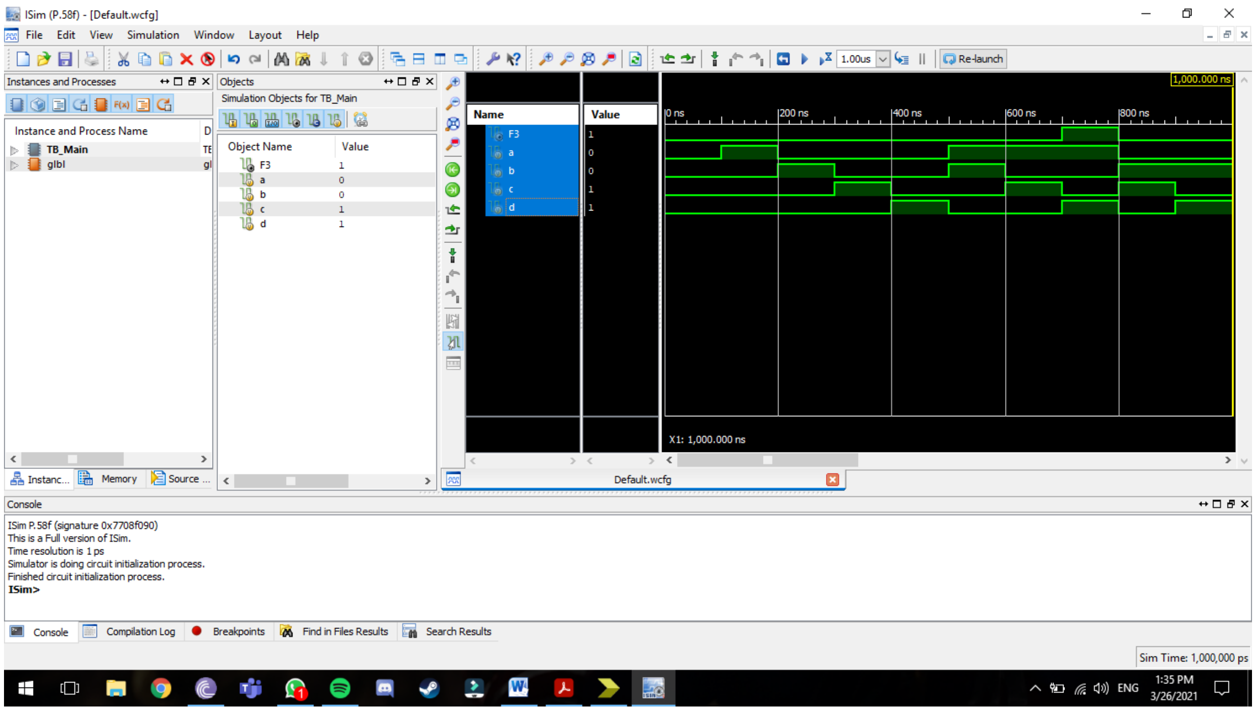
```
module TB_Main;


	// Inputs
	reg a;
	reg b;
	reg c;
	reg d;


	// Outputs
	wire F3;


	// Instantiate the Unit Under Test (UUT)
	Main uut (
		.a(a),
		.b(b),
		.c(c),
		.d(d),
		.F3(F3)
	);


	initial begin
		// Initialize Inputs
		  a = 0;
		b = 0;
		c = 0;
		d = 0;



		#100;
		a = 1;
		b = 0;
		c = 0;
		d = 0;
```

```
#100;

a = 0;

b = 1;

c = 0;

d = 0;


#100;

a = 0;

b = 0;

c = 1;

d = 0;


#100;

a = 0;

b = 0;

c = 0;

d = 1;


#100;

a = 1;

b = 1;

c = 0;

d = 0;


#100;

a = 1;

b = 0;

c = 1;

d = 0;


#100;
```

```
a = 1;

b = 0;

c = 0;

d = 1;


#100;

a = 0;

b = 1;

c = 1;

d = 0;


#100;

a = 0;

b = 1;

c = 0;

d = 1;


#100;

a = 0;

b = 0;

c = 1;

d = 1;


#100;

a = 1;

b = 1;

c = 1;

d = 0;


#100;

a = 1;

b = 1;
```

```
                c = 0;
                d = 1;


                #100;
                a = 0;
                b = 1;
                c = 1;
                d = 1;


                #100;
                a = 1;
                b = 0;
                c = 1;
                d = 1;


                #100;
                a = 1;
                b = 1;
                c = 1;
                d = 1;


                #100;



        end

endmodule
```

Behavior Test:


ISim (P.58f) - [Default.wcfg]
File Edit View Simulation Window Layout Help
Re-launch
Instances and Processes
Objects
Simulation Objects for TB_Main
Default.wcfg
X1: 1,000.000 ns

Object Name	Value
F3	1
a	0
b	0
c	1
d	1

```
ISim P.58f (signature 0x7708f090)
This is a Full version of ISim.
Time resolution is 1 ps
Simulator is doing circuit initialization process.
Finished circuit initialization process.
ISim>
```

Sim Time: 1,000,000 ps

F2 POS Method Code:

```
module Main(
    input a,
    input b,
    input c,
    input d,
    output F2
    );

            wire w1,w2,w3,w4,w5,w6,w7,w8,w9,w10,w11;

    not n1(w1,a);
            not n2(w2,c);
            not n3(w3,c);
            not n4(w4,a);
            not n5(w5,d);
            not n6(w6,a);
```

```
 not n7(w7,b);


 or o1(w8,a,c);
 or o2(w9,w1,b,w2);
 or o3(w10,a,w3,d);
 or o4(w11,w4,b,c,w5);
 or o5(w12,w6,w7,c,d);


 and a1(F2,w8,w9,w10,w11,w12);



endmodule
```

Test Bench:

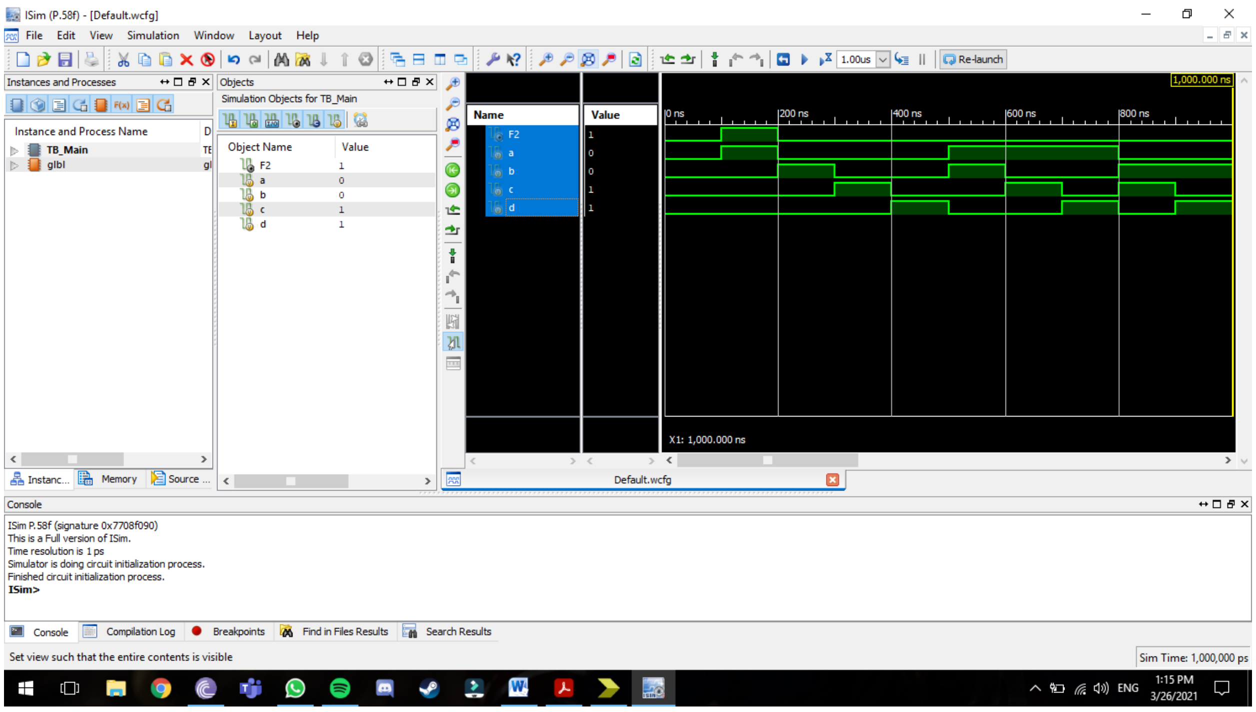
```
module TB_Main;


	// Inputs
	reg a;
	reg b;
	reg c;
	reg d;


	// Outputs
	wire F2;


	// Instantiate the Unit Under Test (UUT)
	Main uut (
		.a(a),
		.b(b),
		.c(c),
		.d(d),
		.F2(F2)
	);


	initial begin
		// Initialize Inputs
  a = 0;
b = 0;
c = 0;
d = 0;


#100;
a = 1;
b = 0;
c = 0;
d = 0;


#100;
a = 0;
b = 1;
c = 0;
d = 0;


#100;
a = 0;
b = 0;
c = 1;
d = 0;


#100;
a = 0;
b = 0;
c = 0;
d = 1;


#100;
a = 1;
```

```
b = 1;

c = 0;

d = 0;



#100;

a = 1;

b = 0;

c = 1;

d = 0;



#100;


a = 1;

b = 0;

c = 0;

d = 1;



#100;

a = 0;

b = 1;

c = 1;

d = 0;



#100;

a = 0;

b = 1;

c = 0;

d = 1;



#100;

a = 0;

b = 0;
```

```
c = 1;
d = 1;


#100;
a = 1;
b = 1;
c = 1;
d = 0;


#100;
a = 1;
b = 1;
c = 0;
d = 1;


#100;
a = 0;
b = 1;
c = 1;
d = 1;


#100;
a = 1;
b = 0;
c = 1;
d = 1;


#100;
a = 1;
b = 1;
c = 1;
d = 1;
```

```
                #100;



        end


endmodule
```

Behavior Test:



ISim (P.58f) - [Default.wcfg]

Object Name	Value
F2	1
a	0
b	0
c	1
d	1

```
ISim P.58f (signature 0x7708f090)
This is a Full version of ISim.
Time resolution is 1 ps
Simulator is doing circuit initialization process.
Finished circuit initialization process.
ISim>
```

Critical Analysis:

In this lab 4 we learned how to make circuits in proteus with the help of boolean functions equations and filled the required outputs given in the table. Also we simulate the min terms and max terms both in reduced and actual equations using proteus software. After this then comes the post labs tasks the same thing we did using the Proteus software we were ought to do same thing using XILINX software and write the Verilog code for all the functions F1,F2, and F3 and also write the test bench as well along with the behavior test of all the functions we attached the following requirements in this lab report as well.