National University of Sciences and Technology (NUST)
School of Electrical Engineering and Computer Science

Department of Electrical Engineering

Faculty Member: Miss Rafia **Dated: 17-02-2025**

Semester: 2nd **Section: BSDS-2A**

Group No.:

EE-221: Digital Logic Design

Lab 4: Design and Implementation of Simple Practical Circuits with Full Adder

		PLO4/CLO4	PLO4/CLO4	PLO5/CLO5	PLO8/CLO6	PLO9/CLO7	
Name	**Reg. No**	**Viva / Lab Performance** 5 Marks	**Analysis of data in Lab Report** 5 Marks	**Modern Tool Usage** 5 Marks	**Ethics and Safety** 5 Marks	**Individual and Team Work** 5 Marks	**Total marks Obtained** 25 Marks
AILYA ZAINAB	523506						
IMAN NAEEM	525378						
LAIBA NASIR	510419						
LUQMAN SHEHZAD	507599						

Lab 4: Design and Implementation of Simple Practical Circuits

This Lab has been divided into two parts.

In first part you are required to design and implement simple practical circuits.
In the second part you are required to write Verilog code for the practical circuit

Objectives:

- ✓ Design the Simple practical circuits using digital logic
- ✓ Hardware Implementation of Basic Logic Circuits
- ✓ Verilog coding of the practical circuit

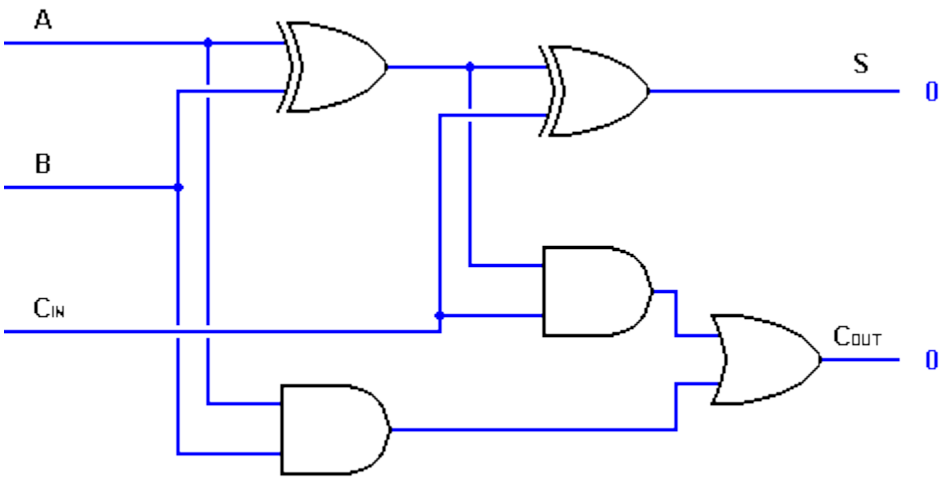
Lab Instructions

- ✓ This lab activity comprises three parts, namely Pre-lab, Lab tasks, and Post-Lab Viva session.
- ✓ The lab report will be uploaded on LMS three days before scheduled lab date. The students will complete the Pre-lab task before coming to the lab and deposit it with teacher/lab engineer for necessary evaluation.
- ✓ The students will start lab task and demonstrate design steps separately for step-wise evaluation (course instructor/lab engineer will sign each step after ascertaining functional verification)
- ✓ Remember that a neat logic diagram with pins numbered coupled with nicely patched circuit will simplify trouble-shooting process.
- ✓ After the lab, students are expected to unwire the circuit and deposit back components before leaving.
- ✓ The students will complete lab task and submit complete report to Lab Engineer before leaving lab
- ✓ There are related questions at the end of this activity. Give complete answers.

.

Lab Task 1: (5 Marks)

Implement the Full Adder Circuit given below, make its truth table and run the circuit on Proteus.

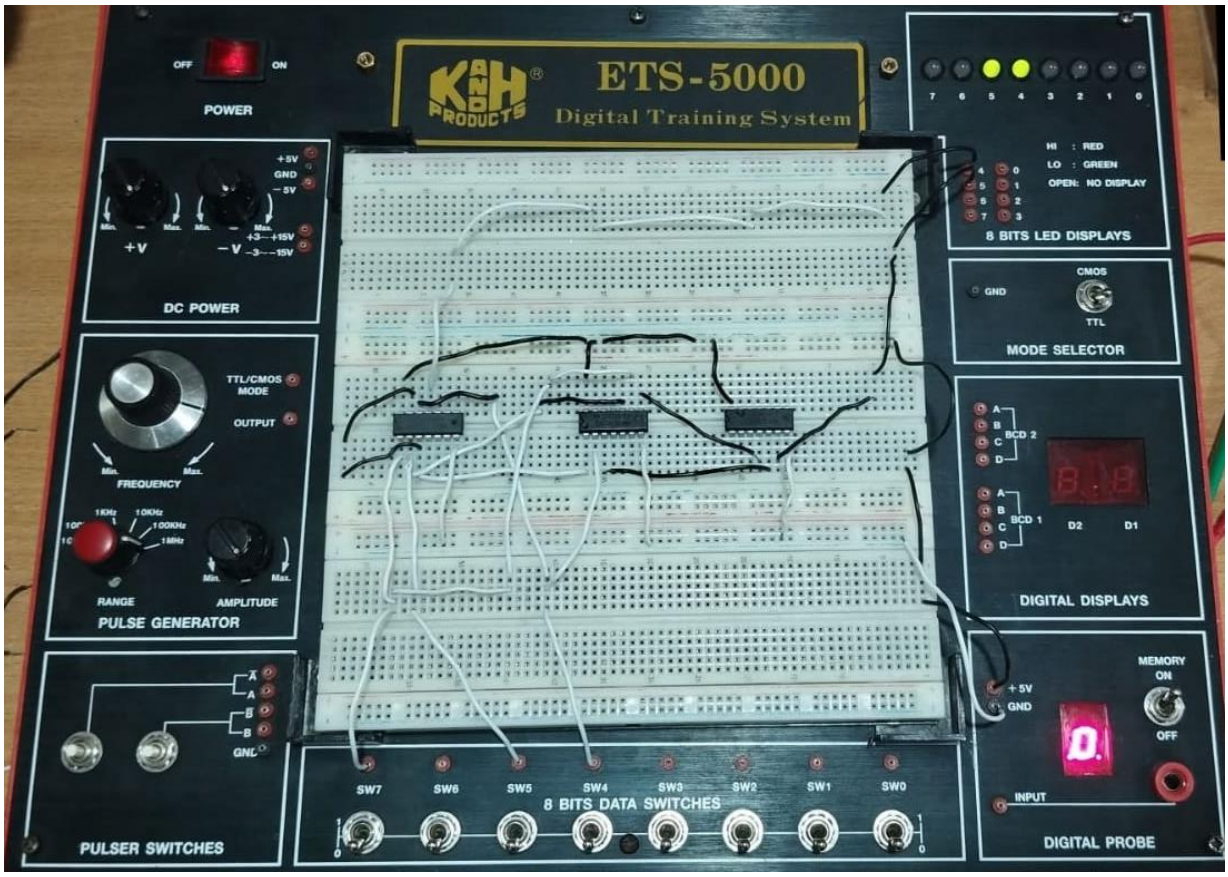


Truth Table

A	B	Cin	Cout	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

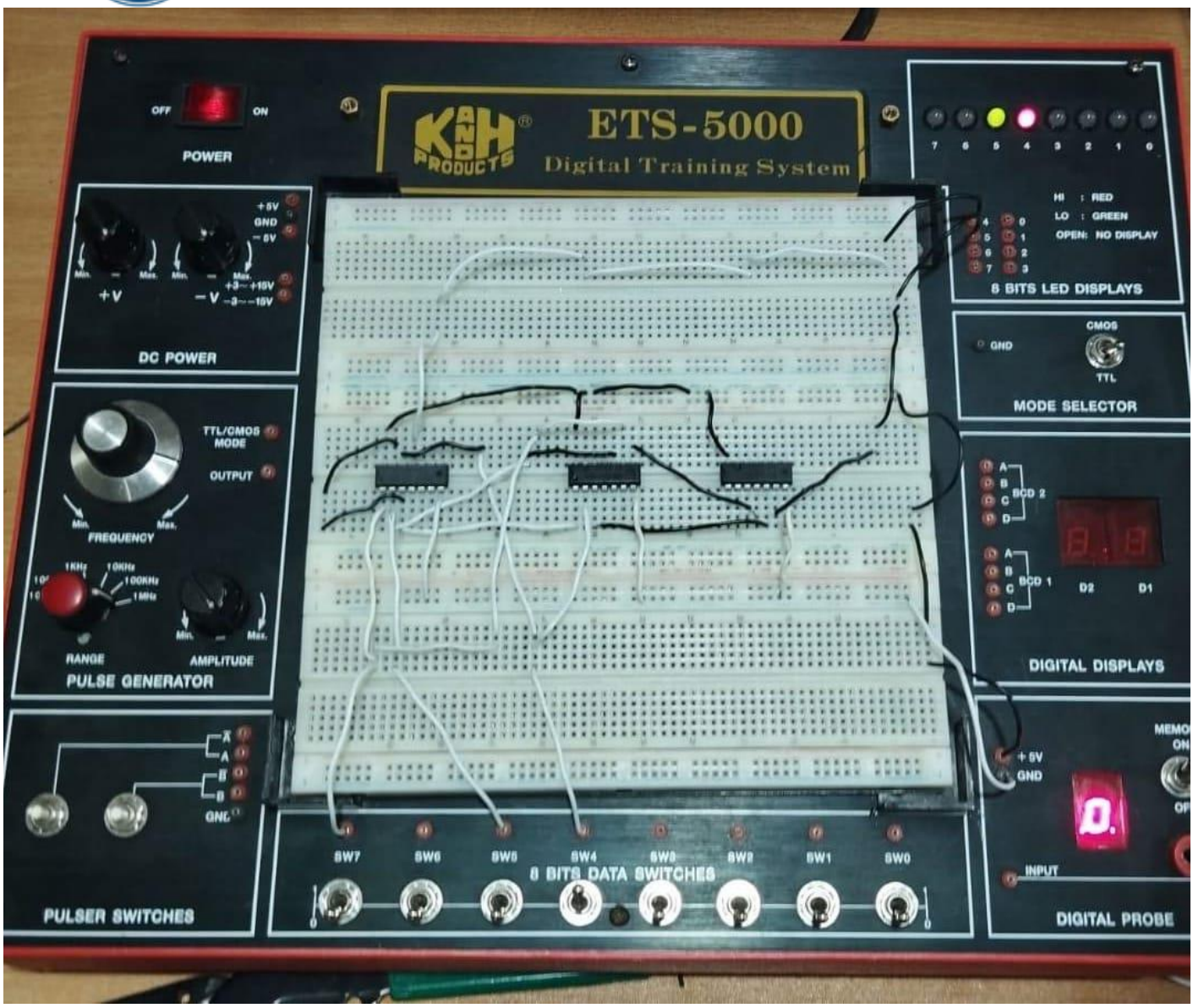
Which are the results we would expect from a FULL ADDER circuit, and to ensure integrity, we have implemented the given schematic on Proteus as well

Hardware Implementation:



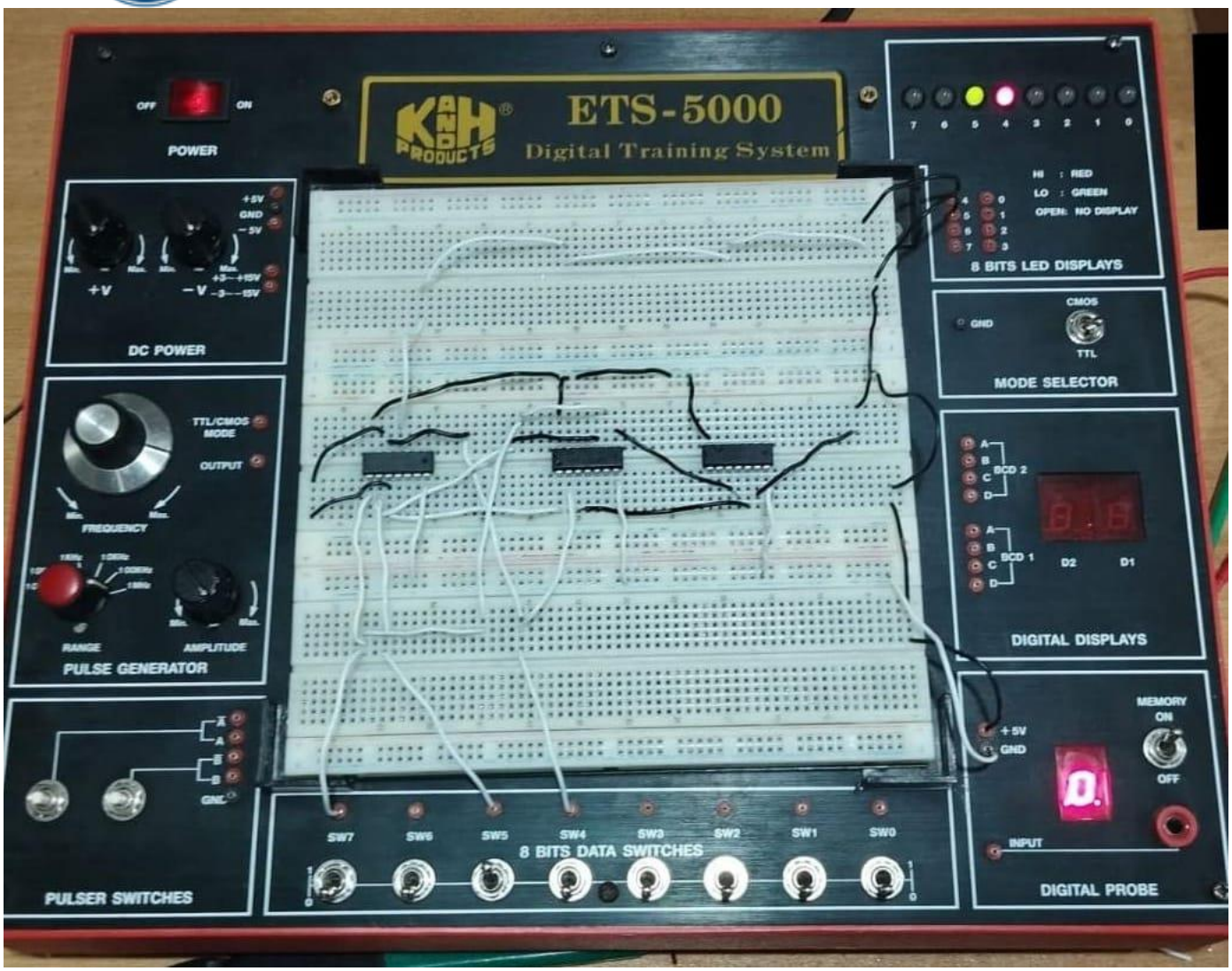
INPUT: 000

OUTPUT: 00



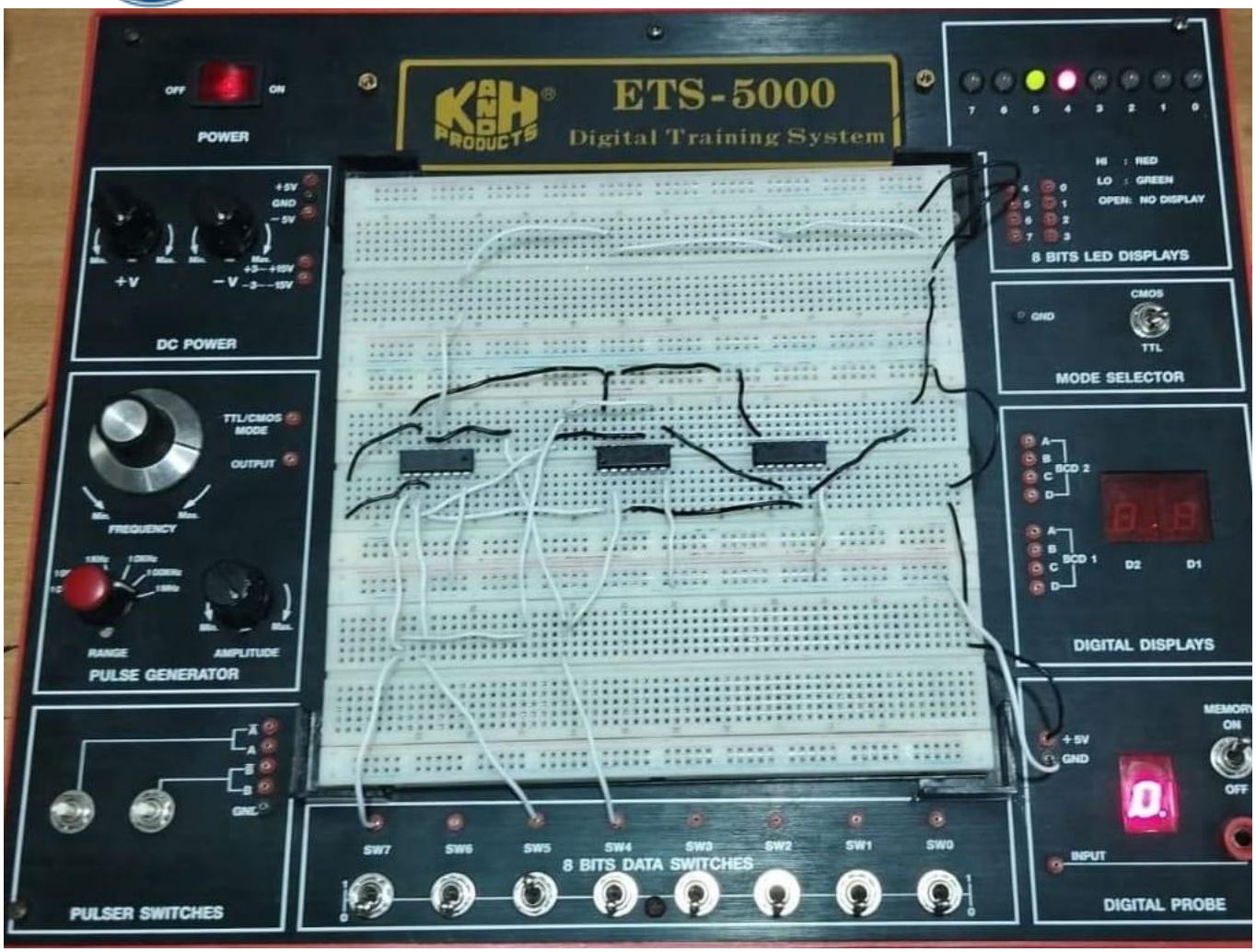
INPUT: 001

OUTPUT: 01



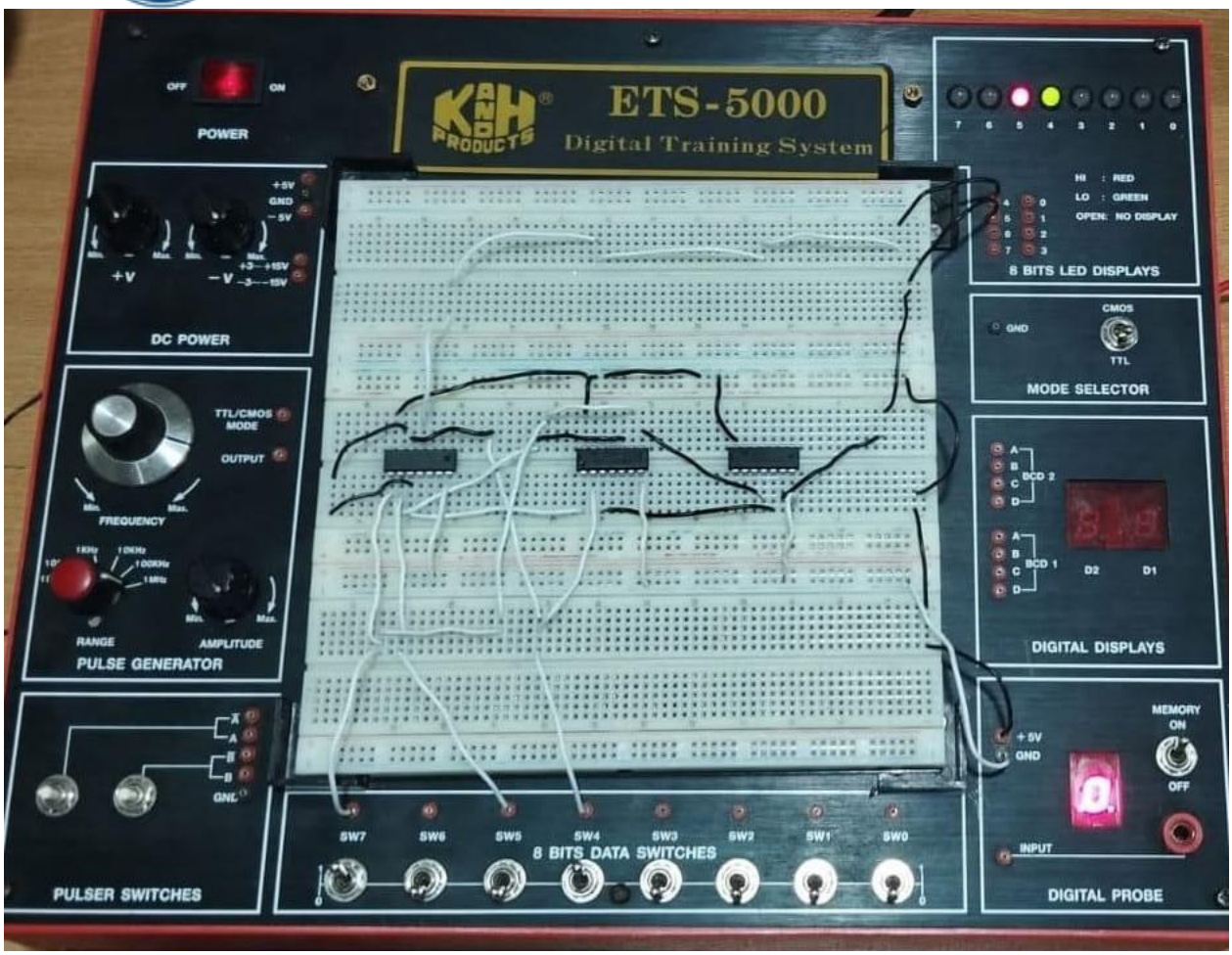
INPUT: 010

OUTPUT: 01



INPUT: 011

OUTPUT: 10

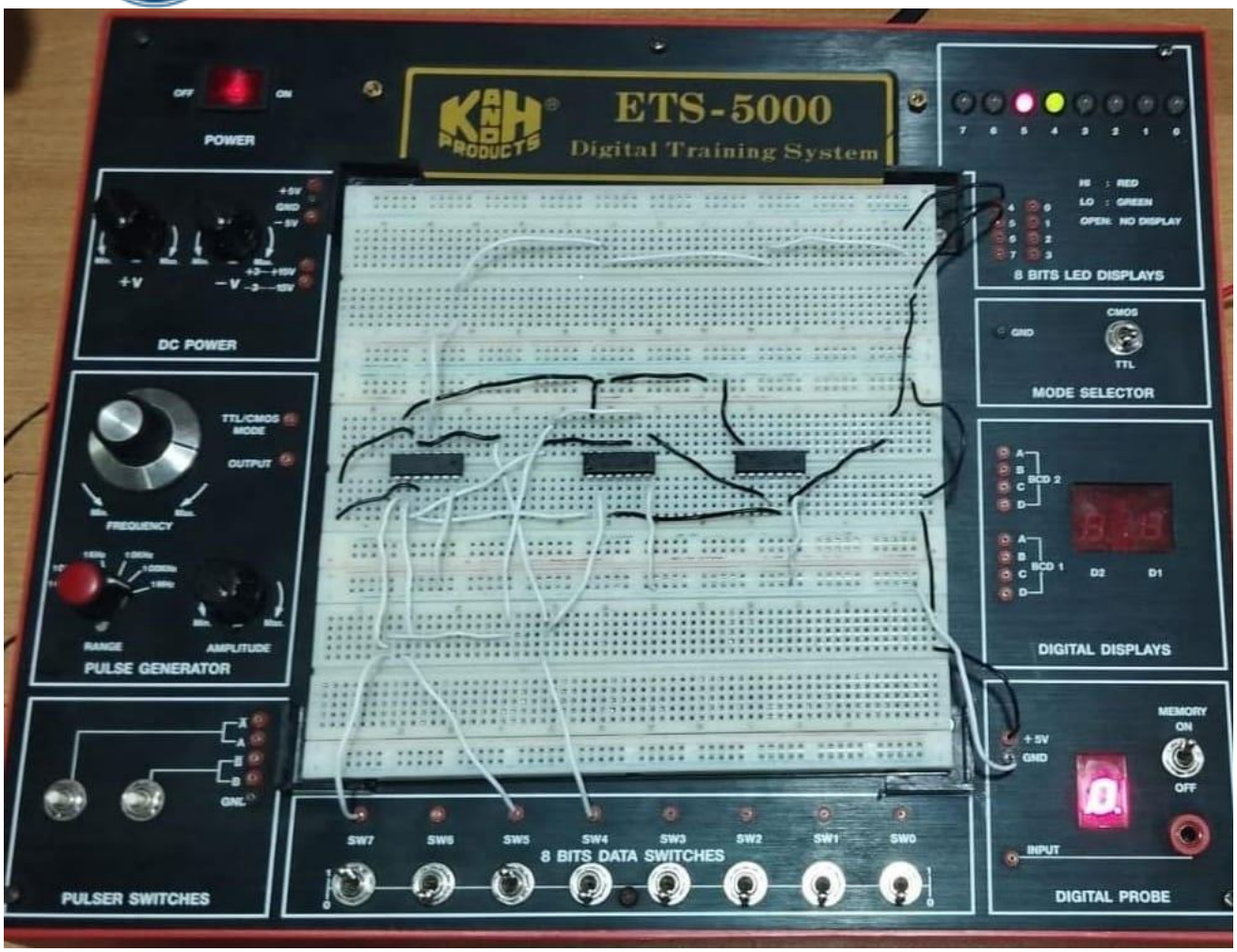


INPUT: 100

OUTPUT: 01

INPUT: 101

OUTPUT: 10



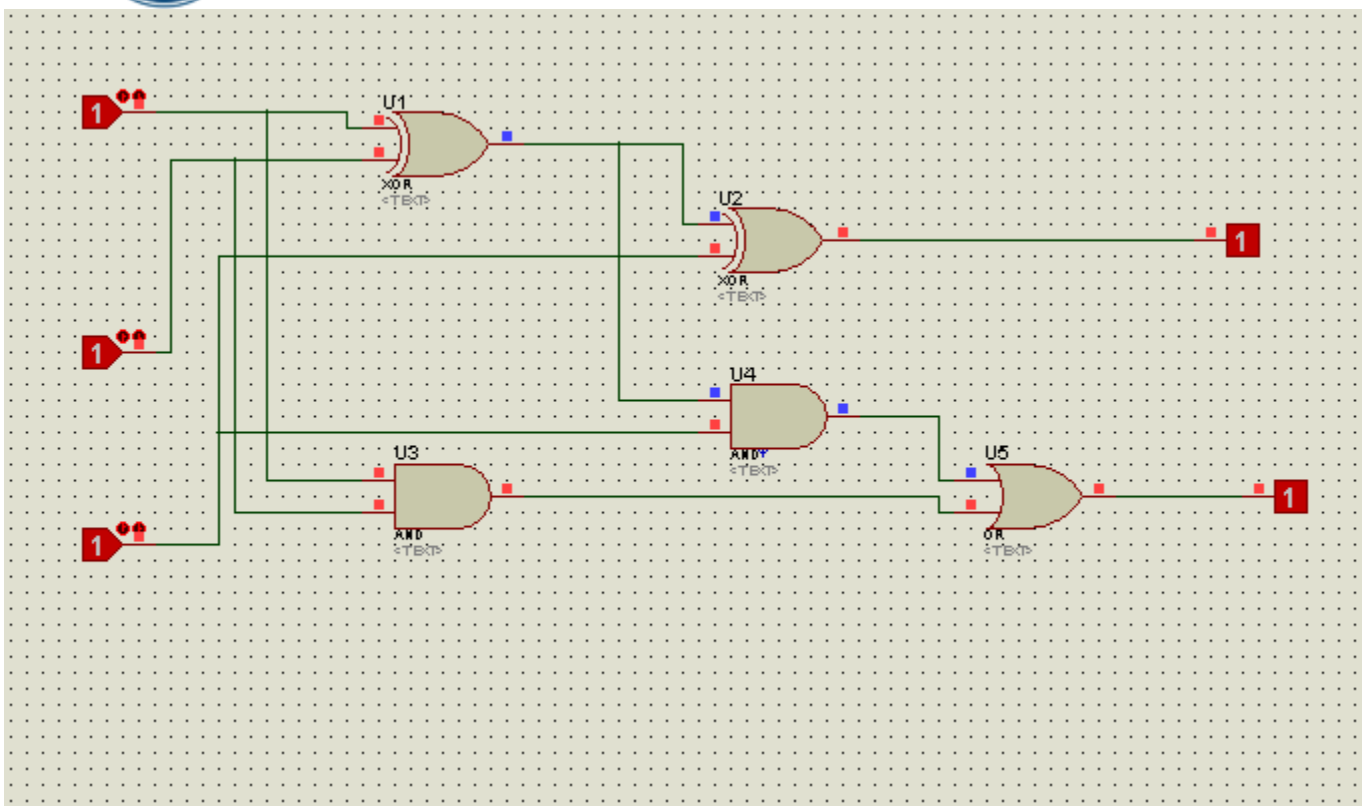
INPUT: 110

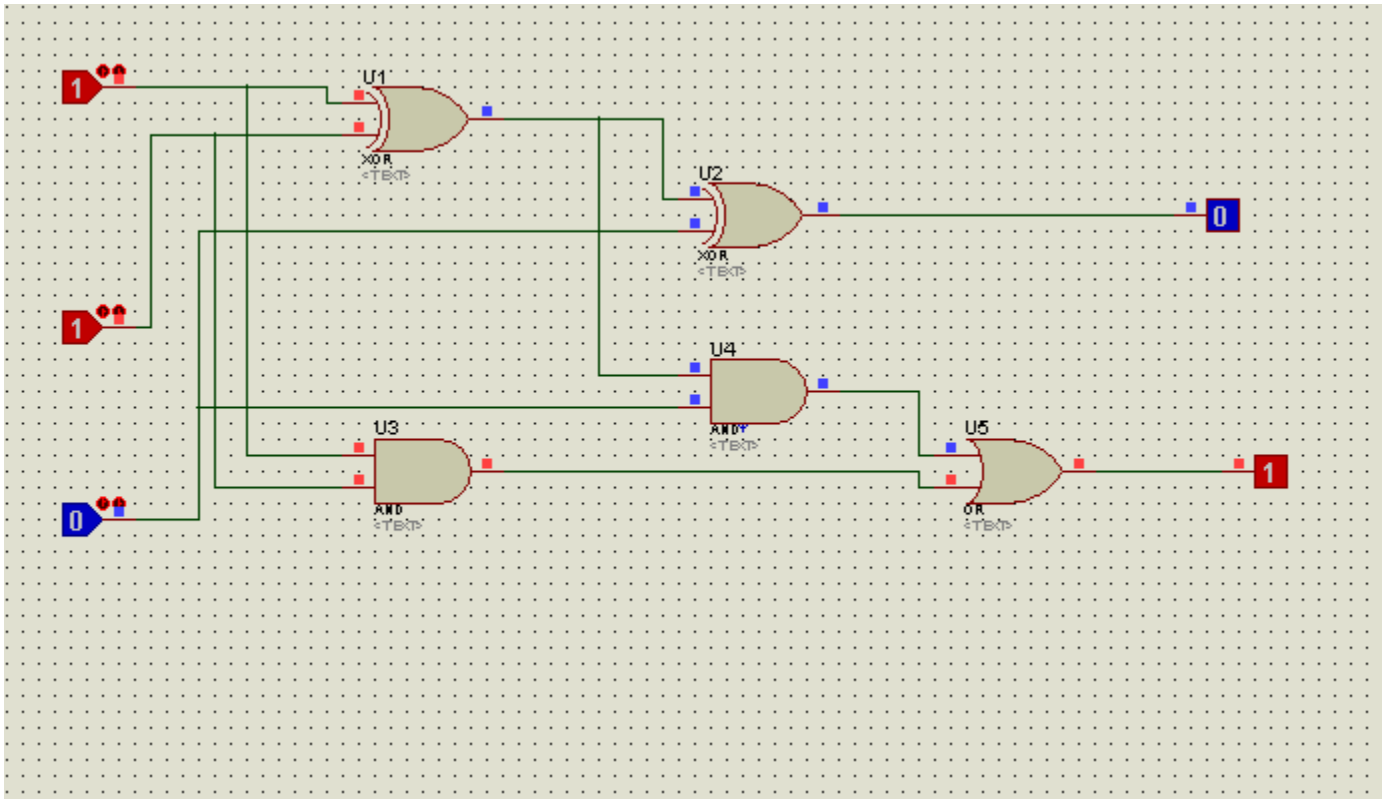
OUTPUT: 10

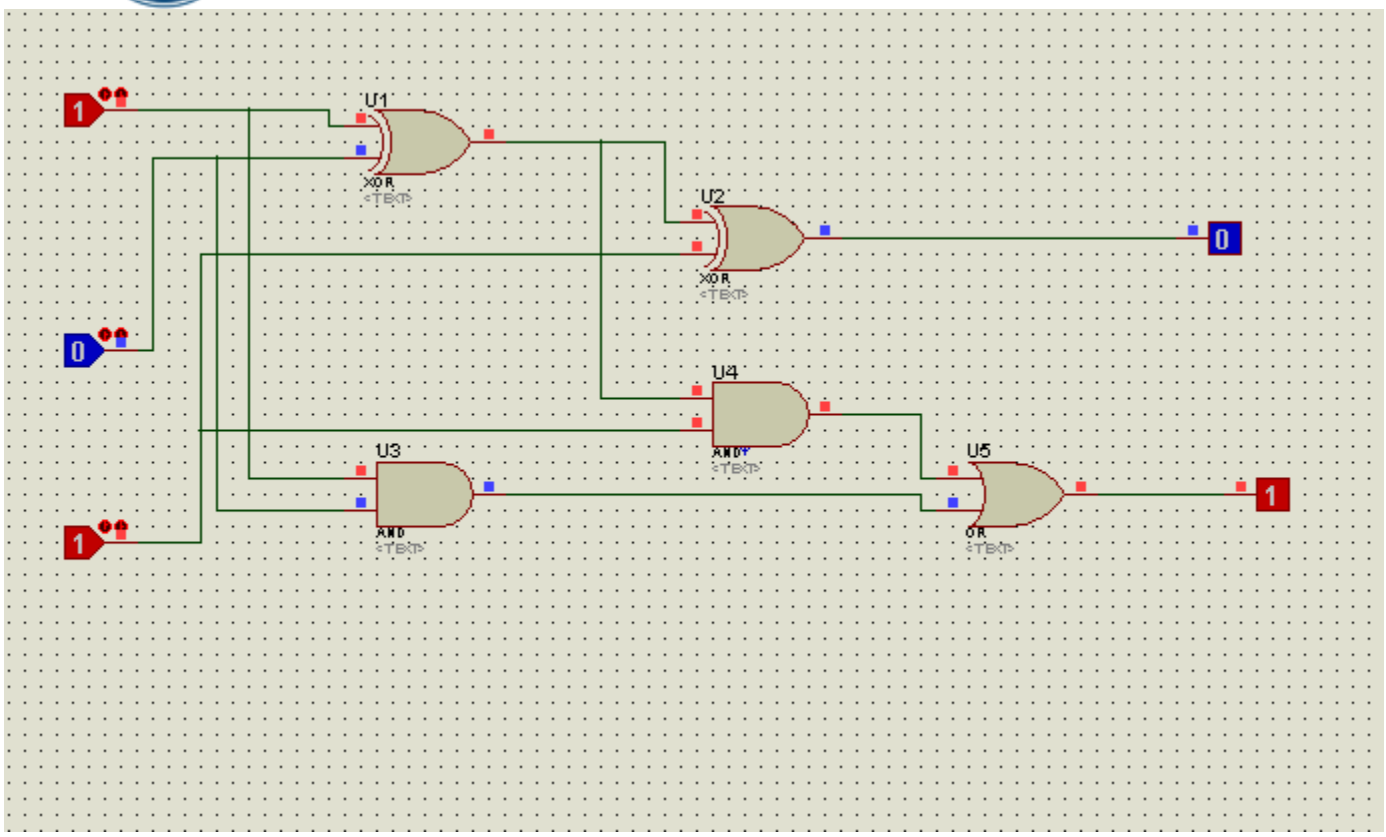
INPUT: 111

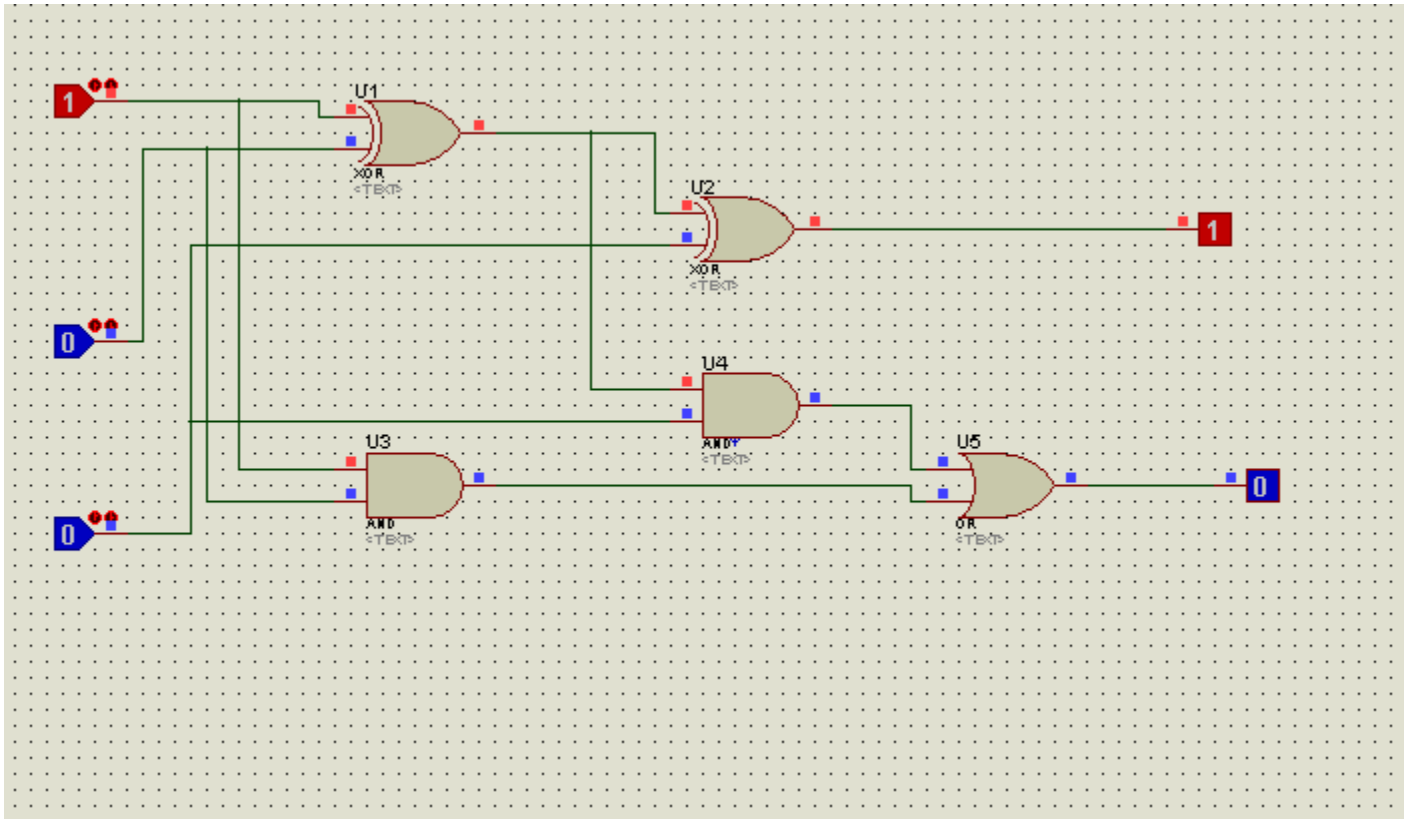
OUTPUT: 11

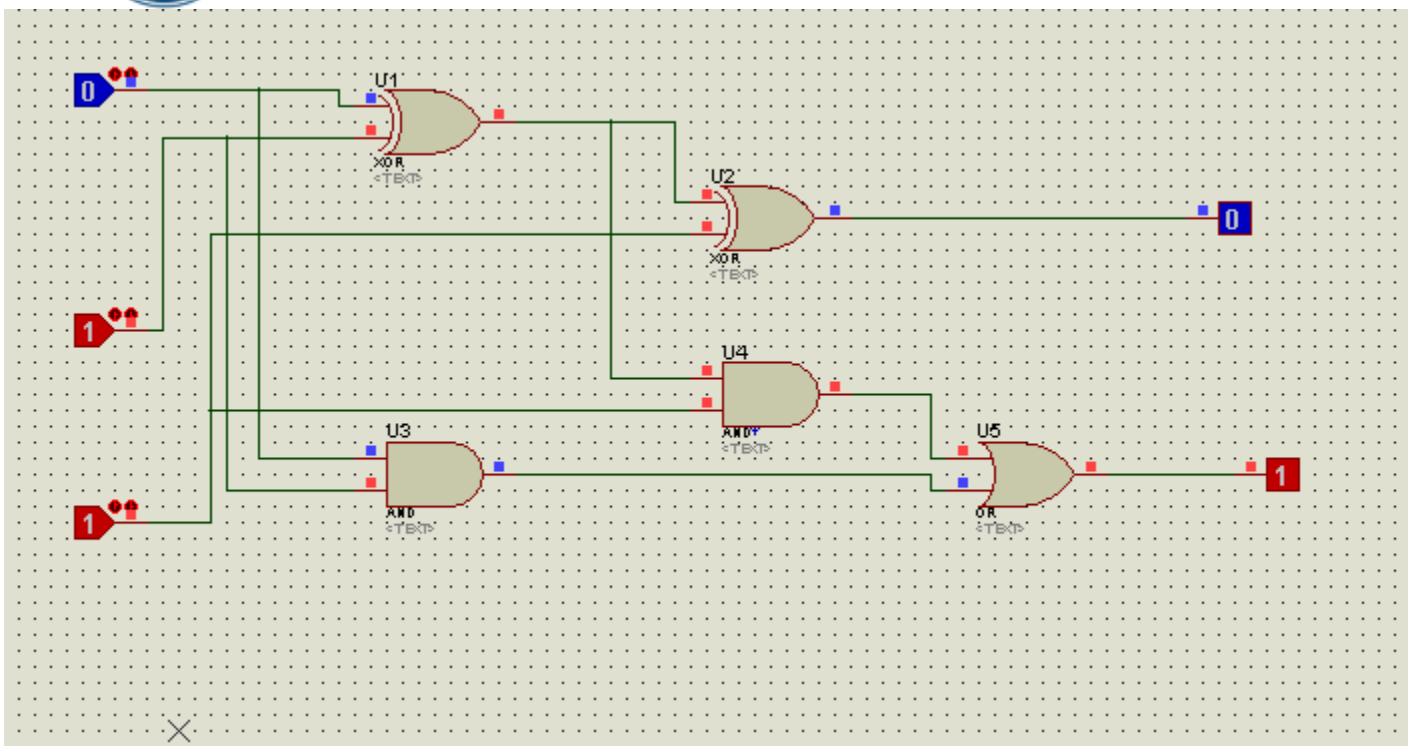
Proteus Simulation:

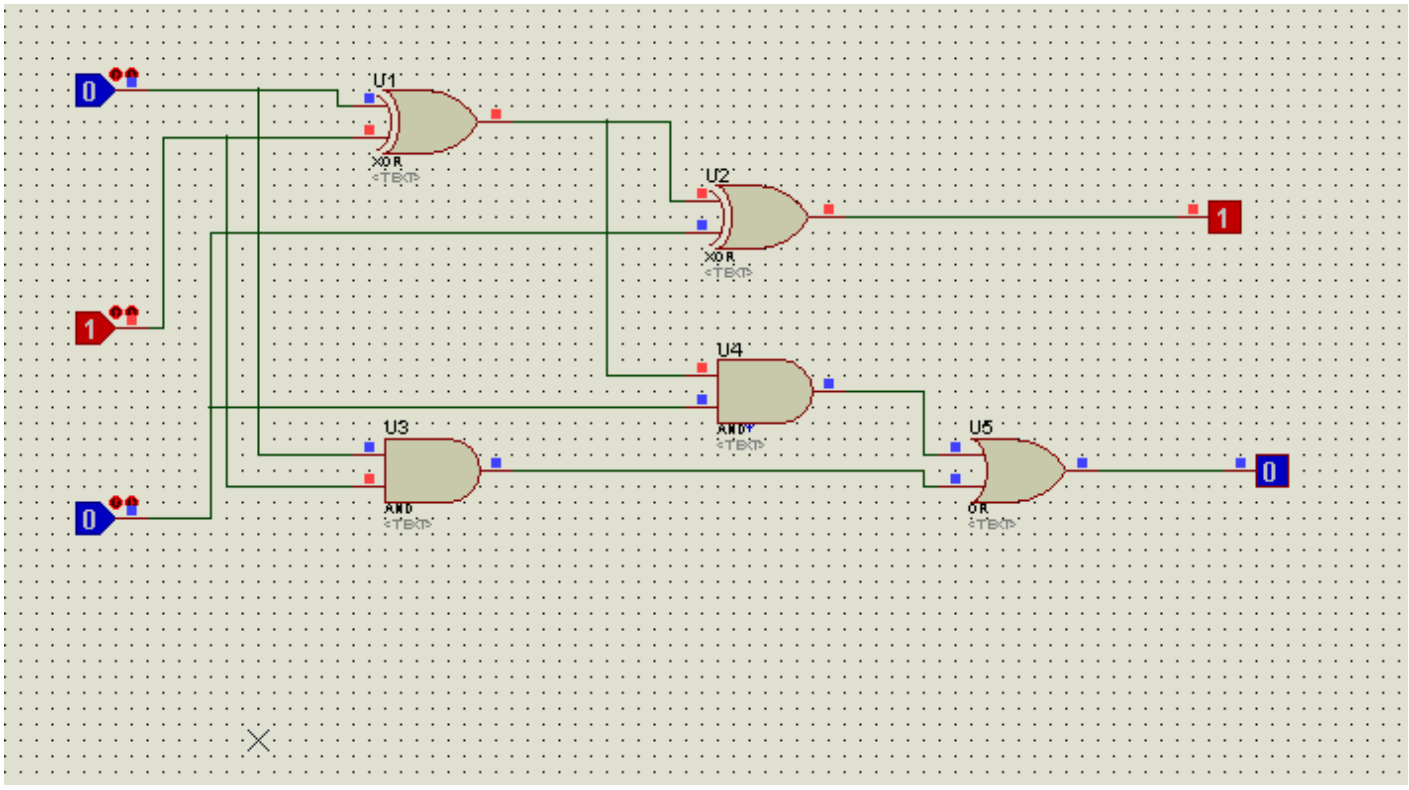

1
1
1
U1
XOR
<TEXT>
U2
XOR
<TEXT>
1
U4
AND
<TEXT>
U3
AND
<TEXT>
U5
OR
<TEXT>
1

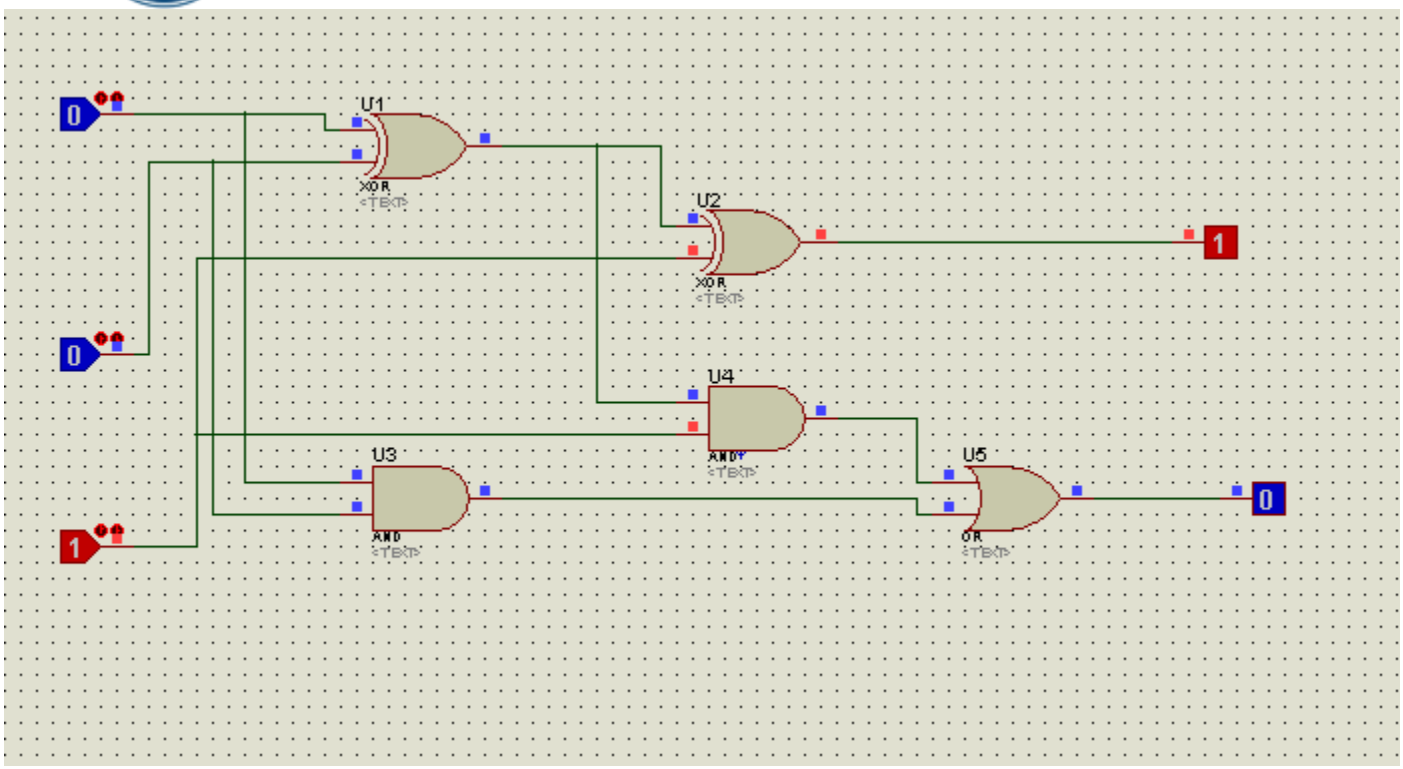

1
1
0
U1
XOR
<TEXT>
U2
XOR
<TEXT>
0
U4
AND
<TEXT>
U3
AND
<TEXT>
U5
OR
<TEXT>
1

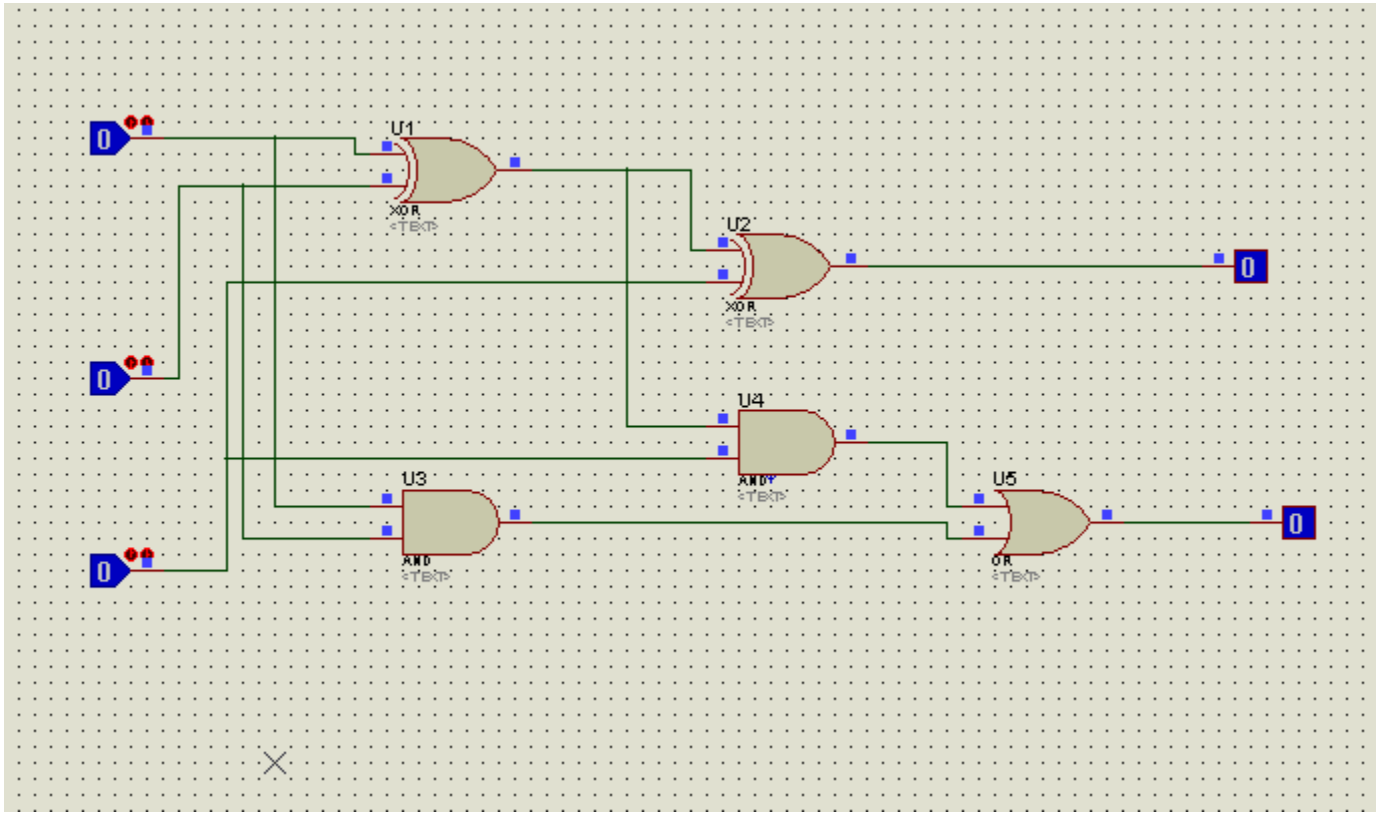

1
0
1
U1
XOR
<TEXT>
U2
XOR
<TEXT>
0
U4
AND
<TEXT>
U3
AND
<TEXT>
U5
OR
<TEXT>
1


1
0
0
U1
XOR
<TEXT>
U2
XOR
<TEXT>
1
U4
AND
<TEXT>
U3
AND
<TEXT>
U5
OR
<TEXT>
0


0
1
1
U1
XOR
U2
XOR
0
U4
AND
U3
AND
U5
OR
1


0
1
0
U1
XOR
U2
XOR
1
U4
AND
U3
AND
U5
OR
0


0
0
1
U1
XOR
<TEXT>
U2
XOR
<TEXT>
1
U4
AND
<TEXT>
U3
AND
<TEXT>
U5
OR
<TEXT>
0


0
0
0
U1
XOR
<TEXT>
U2
XOR
<TEXT>
0
U4
AND
<TEXT>
U3
AND
<TEXT>
U5
OR
<TEXT>
0

Lab Task 2:

(2.5 Marks)

Simulate the gate-level model of the circuit you patched in task 1. Give the code and resulted waveform in the space provided below.

CODE:

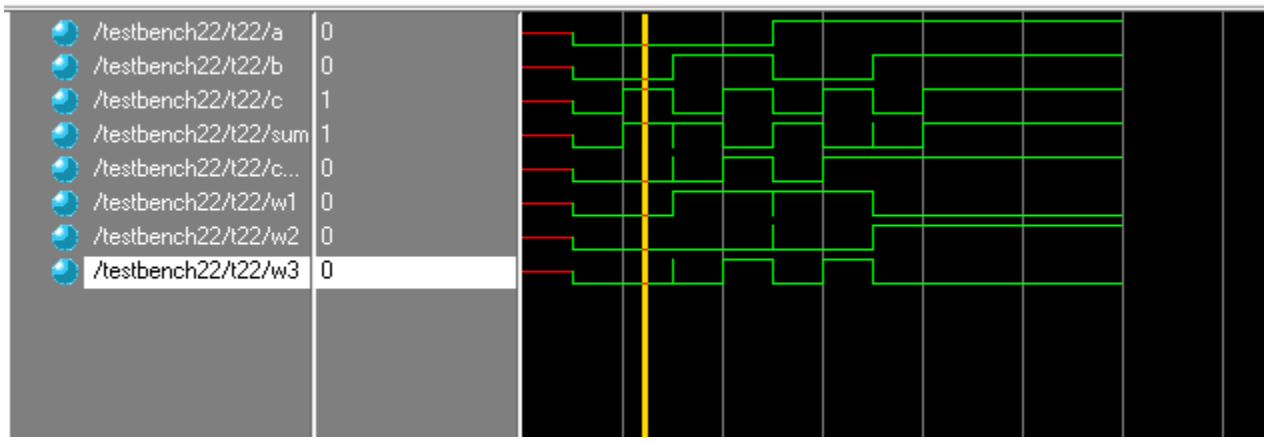
```
module task1(sum, carry, a, b, c);


    input a, b, c;

    output sum, carry;


    wire w1, w2, w3;


    xor x1(w1, a, b);

    xor x2(sum, w1, c);


    and a1(w2, a, b);

    and a2(w3, w1, c);


    or o1 (carry, w3, w2);


endmodule


module testbench22;

    reg A, B, C;

    wire SUM, CARRY;

    task1 t22(SUM, CARRY, A, B, C);

    initial
```

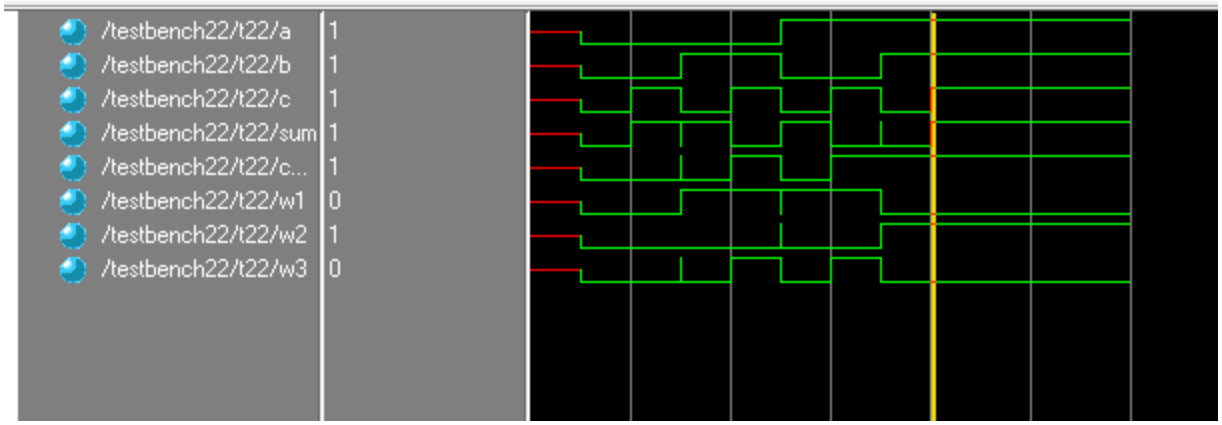
```
begin
    #100 A=1'b0; B=1'b0; C=1'b0;
    #100 A=1'b0; B=1'b0; C=1'b1;
    #100 A=1'b0; B=1'b1; C=1'b0;
    #100 A=1'b0; B=1'b1; C=1'b1;
    #100 A=1'b1; B=1'b0; C=1'b0;
    #100 A=1'b1; B=1'b0; C=1'b1;
    #100 A=1'b1; B=1'b1; C=1'b0;
    #100 A=1'b1; B=1'b1; C=1'b1;
end

Endmodule
```

ln # C:/Modelte

```
 1 module task1(sum, carry, a, b, c);
 2
 3     input a, b, c;
 4     output sum, carry;
 5
 6     wire w1, w2, w3;
 7
 8     xor x1(w1, a, b);
 9     xor x2(sum, w1, c);
10
11     and a1(w2, a, b);
12     and a2(w3, w1, c);
13
14     or o1 (carry, w3, w2);
15
16 endmodule
17
18 module testbench22;
19     reg A, B, C;
20     wire SUM, CARRY;
21     task1 t22(SUM, CARRY, A, B, C);
22     initial
23     begin
24         #100 A=1'b0; B=1'b0; C=1'b0;
25         #100 A=1'b0; B=1'b0; C=1'b1;
26         #100 A=1'b0; B=1'b1; C=1'b0;
27         #100 A=1'b0; B=1'b1; C=1'b1;
28         #100 A=1'b1; B=1'b0; C=1'b0;
29         #100 A=1'b1; B=1'b0; C=1'b1;
30         #100 A=1'b1; B=1'b1; C=1'b0;
31         #100 A=1'b1; B=1'b1; C=1'b1;
32     end
33
34 endmodule
35
```

OUTPUT:





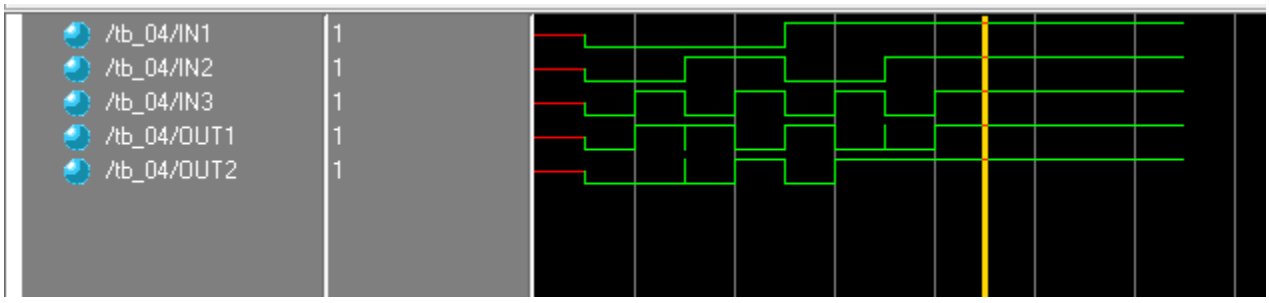
Lab Task 3:

(2.5 Marks)

Write Verilog code to implement full adder using two half adder circuits.

```
ln #                                                    C:/Modeltech_5.7f/lab4.
 1 module halfadd (h_s, h_c, in1, in2);
 2     input in1, in2;
 3     output h_s, h_c;
 4     xor x1(h_s, in1, in2);
 5     and a1(h_c, in1, in2);
 6 endmodule
 7
 8 module full_add (sum, carry, IN1, IN2, CIN);
 9     input IN1, IN2, CIN;
10     output sum, carry;
11     wire h_s, h_c1, h_c2;
12
13     half_add ha1(h_s, h_c1, IN1, IN2);
14     half_add ha2(sum, h_c2, h_s, CIN);
15     or o1(carry, h_c1, h_c2);
16 endmodule
17
18 module tb_04;
19     reg IN1, IN2, IN3;
20     wire OUT1, OUT2;
21
22     full_add lab4(OUT1, OUT2, IN1, IN2, IN3);
23
24     initial begin
25         #100 IN1 = 1'b0; IN2 = 1'b0; IN3 = 1'b0;
26         #100 IN1 = 1'b0; IN2 = 1'b0; IN3 = 1'b1;
27         #100 IN1 = 1'b0; IN2 = 1'b1; IN3 = 1'b0;
28         #100 IN1 = 1'b0; IN2 = 1'b1; IN3 = 1'b1;
29         #100 IN1 = 1'b1; IN2 = 1'b0; IN3 = 1'b0;
30         #100 IN1 = 1'b1; IN2 = 1'b0; IN3 = 1'b1;
31         #100 IN1 = 1'b1; IN2 = 1'b1; IN3 = 1'b0;
32         #100 IN1 = 1'b1; IN2 = 1'b1; IN3 = 1'b1;
33     end
34 endmodule
```

OUTPUT:



Observations/Comments:

In this lab, we deployed the concepts learned in Lab 2, *Implementation of a Half Adder circuit*, to implement a Full Adder. Two Half Adders concatenated together and through use of an additional OR gate give us a Full Adder circuit, essentially implying that:

$$\text{Full Adder} = \text{Half Adder} + \text{Half Adder}$$

We have also confirmed/verified the circuit's output as it conforms to the truth table laid out in the Pre – lab and that it serves the purpose for what it was designed conclusively.