DIGITAL DESIGN LAB 2

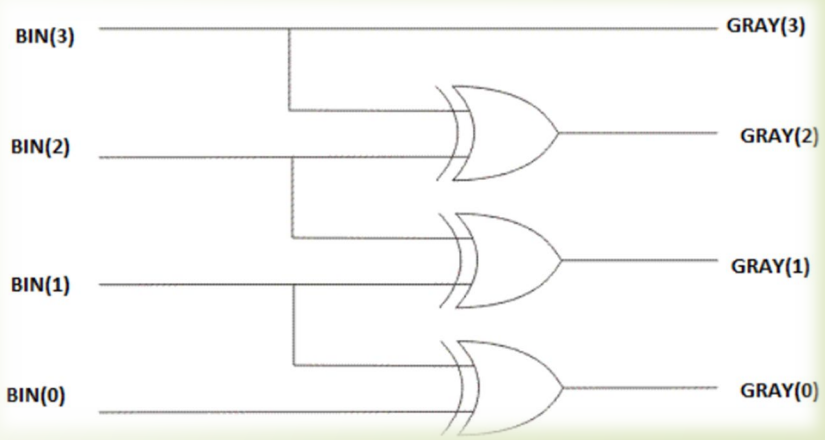
DIVYAM PATEL

B20EE082

Work 1: BINARY TO GRAY CODE CONVERTER

BINARY TO GRAY CODE CONVERTER TRUTH TABLE

INPUT				OUTPUT			
B4	B3	B2	B1	G4	G3	G2	G1
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	1
0	0	1	1	0	0	1	0
0	1	0	0	0	1	1	0
0	1	0	1	0	1	1	1
0	1	1	0	0	1	0	1
0	1	1	1	0	1	0	0
1	0	0	0	1	1	0	0
1	0	0	1	1	1	0	1
1	0	1	0	1	1	1	1
1	0	1	1	1	1	1	0
1	1	0	0	1	0	1	0
1	1	0	1	1	0	1	1
1	1	1	0	1	0	0	1
1	1	1	1	1	0	0	0



CODE

Project Summary | binary_to_gray.v | test_binary_to_gray.v

D:/YEAR 2/SEMESTER 2/DIGITAL DESIGN/LABS/LAB 2/binary_to_gray.v

```
1   `timescale 1ns / 1ps
2   //////////////////////////////////////////////////////////////////////////////...
21
22
23  module binary_to_gray(B,G);
24  input [4:1]B;      // binary input
25  output [4:1]G;     // gray code output
26
27  assign G[4] = B[4];
28  assign G[3] = B[4] ^ B[3];
29  assign G[2] = B[3] ^ B[2];
30  assign G[1] = B[2] ^ B[1];
31
32  endmodule
```


```
module binary_to_gray(B,G);
input [4:1]B;     // binary input
output [4:1]G;    // gray code output

assign G[4] = B[4];
assign G[3] = B[4] ^ B[3];
assign G[2] = B[3] ^ B[2];
assign G[1] = B[2] ^ B[1];

endmodule
```

Project Summary | test_binary_to_gray.v | binary_to_gray.v

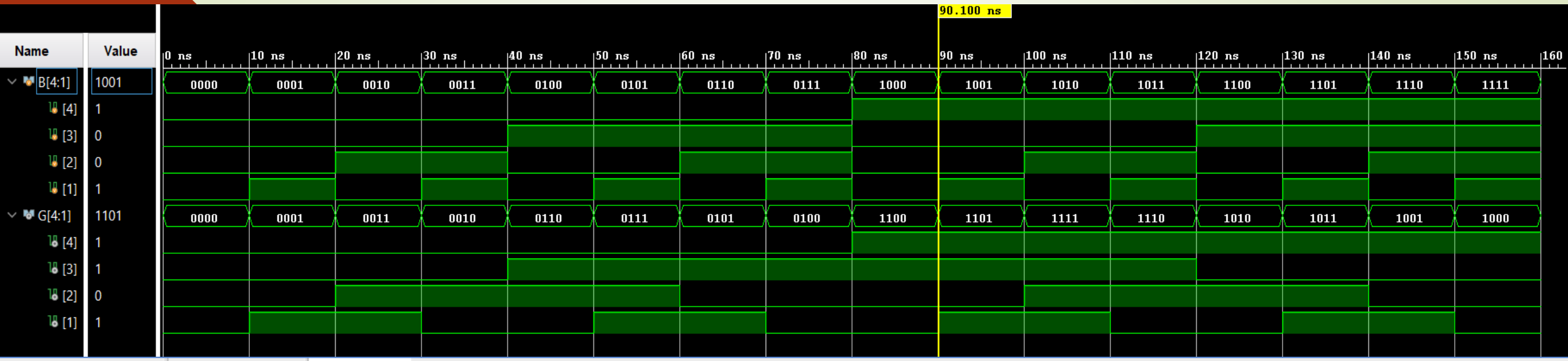
D:/YEAR 2/SEMESTER 2/DIGITAL DESIGN/LABS/LAB 2/test_binary_to_gray.v

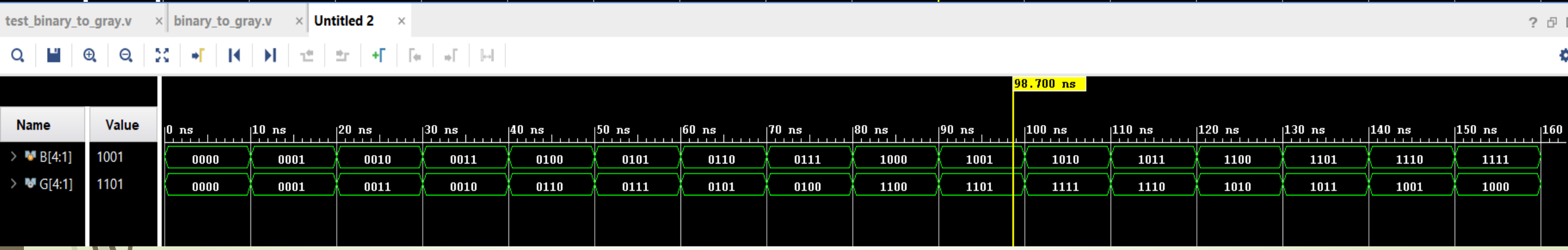
```
19  //
20  //////////////////////////////////////////////////////////////////
21
22
23  module test_binary_to_gray;
24  reg [4:1]B;
25  wire [4:1]G;
26  binary_to_gray G1(B,G);
27
28  initial
29  begin
30  B = 4'b0000;
31  #10 B = 4'b0001;
32  #10 B = 4'b0010;
33  #10 B = 4'b0011;
34  #10 B = 4'b0100;
35  #10 B = 4'b0101;
36  #10 B = 4'b0110;
37  #10 B = 4'b0111;
38  #10 B = 4'b1000;
39  #10 B = 4'b1001;
40  #10 B = 4'b1010;
41  #10 B = 4'b1011;
42  #10 B = 4'b1100;
43  #10 B = 4'b1101;
44  #10 B = 4'b1110;
45  #10 B = 4'b1111;
46  end
47
48  initial #160 $finish;
49  endmodule
```

TEST BENCH

```
module test_binary_to_gray;
reg [4:1]B;
wire [4:1]G;
binary_to_gray G1(B,G);

initial
begin
B = 4'b0000;
#10 B = 4'b0001;
#10 B = 4'b0010;
#10 B = 4'b0011;
#10 B = 4'b0100;
#10 B = 4'b0101;
#10 B = 4'b0110;
#10 B = 4'b0111;
#10 B = 4'b1000;
#10 B = 4'b1001;
#10 B = 4'b1010;
#10 B = 4'b1011;
#10 B = 4'b1100;
#10 B = 4'b1101;
#10 B = 4'b1110;
#10 B = 4'b1111;
end

initial #160 $finish;
endmodule
```

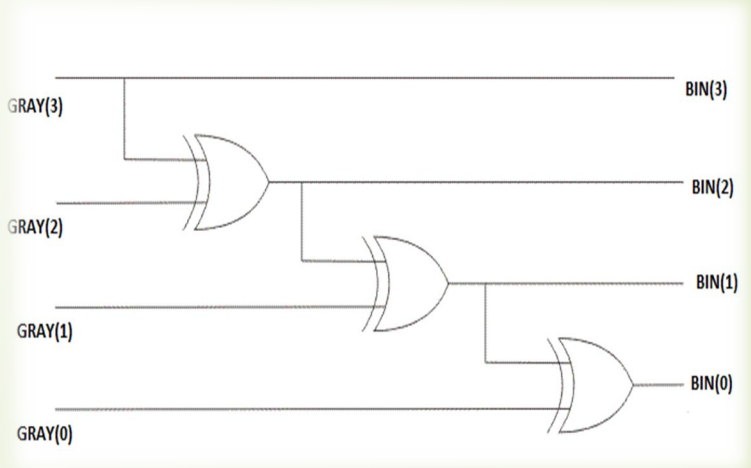
SIMULATION


90.100 ns
Name
Value
B[4:1]
1001
[4]
1
[3]
0
[2]
0
[1]
1
G[4:1]
1101
[4]
1
[3]
1
[2]
0
[1]
1
0 ns
10 ns
20 ns
30 ns
40 ns
50 ns
60 ns
70 ns
80 ns
90 ns
100 ns
110 ns
120 ns
130 ns
140 ns
150 ns
160
0000 0001 0010 0011 0100 0101 0110 0111 1000 1001 1010 1011 1100 1101 1110 1111
0000 0001 0011 0010 0110 0111 0101 0100 1100 1101 1111 1110 1010 1011 1001 1000


test_binary_to_gray.v
binary_to_gray.v
Untitled 2
98.700 ns
Name
Value
B[4:1]
1001
G[4:1]
1101
0 ns
10 ns
20 ns
30 ns
40 ns
50 ns
60 ns
70 ns
80 ns
90 ns
100 ns
110 ns
120 ns
130 ns
140 ns
150 ns
160
0000 0001 0010 0011 0100 0101 0110 0111 1000 1001 1010 1011 1100 1101 1110 1111
0000 0001 0011 0010 0110 0111 0101 0100 1100 1101 1111 1110 1010 1011 1001 1000

Work 1: GRAY TO BINARY CODE CONVERTER

GRAY TO BINARY CODE CONVERTER TRUTH TABLE							
INPUT				OUTPUT			
G4	G3	G2	G1	B4	B3	B2	B1
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	1	0	0	1	0
0	0	1	0	0	0	1	1
0	1	1	0	1	1	0	0
0	1	1	1	1	1	0	1
0	1	0	1	1	1	1	0
0	1	0	0	1	1	1	1
1	1	0	0	0	1	0	0
1	1	0	1	0	1	0	1
1	1	1	1	0	1	1	0
1	1	1	0	0	1	1	1
1	0	1	0	1	0	0	0
1	0	1	1	1	0	0	1
1	0	0	1	1	0	1	0
1	0	0	0	1	0	1	1



CODE

Project Summary | test_binary_to_gray.v | binary_to_gray.v | test_gray_to_binary.v | gray_to_bina

D:/YEAR 2/SEMESTER 2/DIGITAL DESIGN/LABS/LAB 2/gray_to_binary.v

```
1   `timescale 1ns / 1ps
2   //////////////////////////////////////////////////////////////////////////////...
21
22
23  module gray_to_binary(G,B);
24  input [4:1]G;      // gray code input
25  output [4:1]B;     // binary output
26
27  assign B[4] = G[4];
28  assign B[3] = G[4] ^ G[3];
29  assign B[2] = G[4] ^ G[3] ^ G[2];
30  assign B[1] = G[4] ^ G[3] ^ G[2] ^ G[1];
31
32  endmodule
```


```
module gray_to_binary(G,B);
input [4:1]G;     // gray code input
output [4:1]B;     // binary output

assign B[4] = G[4];
assign B[3] = G[4] ^ G[3];
assign B[2] = G[4] ^ G[3] ^ G[2];
assign B[1] = G[4] ^ G[3] ^ G[2] ^ G[1];

endmodule
```

Project Summary | test_binary_to_gray.v | binary_to_gray.v | test_gray_to_binary.

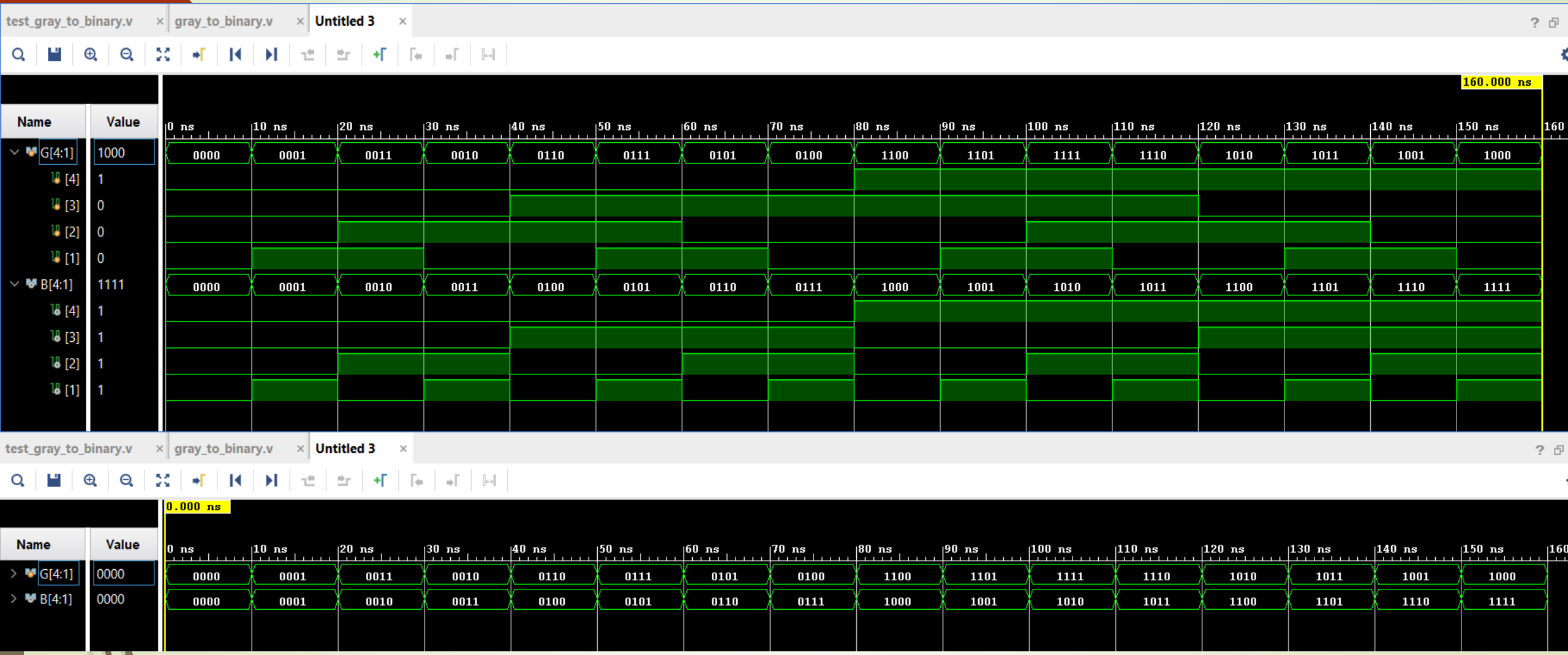
D:/YEAR 2/SEMESTER 2/DIGITAL DESIGN/LABS/LAB 2/test_gray_to_binary.v

```
1  `timescale 1ns / 1ps
2  //////////////////////////////////////////////////////////////////////////////////
21
22
23 module test_gray_to_binary;
24 reg [4:1]G;
25 wire [4:1]B;
26 gray_to_binary B1(G,B);
27
28 initial
29 begin
30 G = 4'b0000;
31 #10 G = 4'b0001;
32 #10 G = 4'b0011;
33 #10 G = 4'b0010;
34 #10 G = 4'b0110;
35 #10 G = 4'b0111;
36 #10 G = 4'b0101;
37 #10 G = 4'b0100;
38 #10 G = 4'b1100;
39 #10 G = 4'b1101;
40 #10 G = 4'b1111;
41 #10 G = 4'b1110;
42 #10 G = 4'b1010;
43 #10 G = 4'b1011;
44 #10 G = 4'b1001;
45 #10 G = 4'b1000;
46 end
47
48 initial #160 $finish;
49 endmodule
```

TEST BENCH

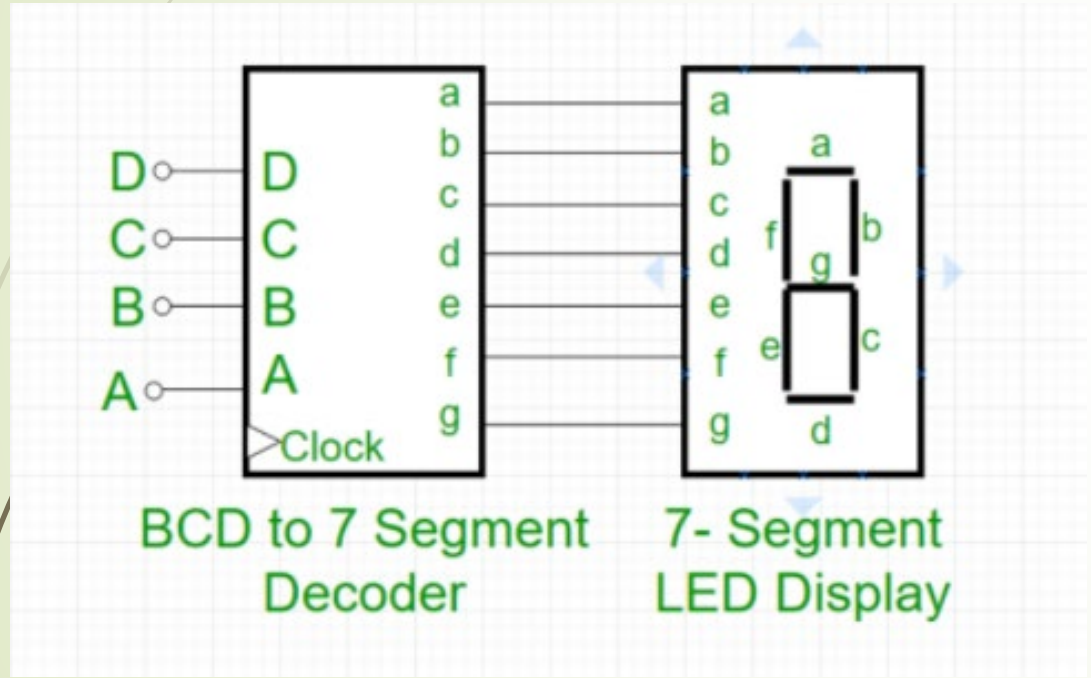
```
module test_gray_to_binary;
reg [4:1]G;
wire [4:1]B;
gray_to_binary B1(G,B);

initial
begin
G = 4'b0000;
#10 G = 4'b0001;
#10 G = 4'b0011;
#10 G = 4'b0010;
#10 G = 4'b0110;
#10 G = 4'b0111;
#10 G = 4'b0101;
#10 G = 4'b0100;
#10 G = 4'b1100;
#10 G = 4'b1101;
#10 G = 4'b1111;
#10 G = 4'b1110;
#10 G = 4'b1010;
#10 G = 4'b1011;
#10 G = 4'b1001;
#10 G = 4'b1000;
end

initial #160 $finish;
endmodule
```

SIMULATION


test_gray_to_binary.v
gray_to_binary.v
Untitled 3
Name
Value
G[4:1]
1000
[4]
1
[3]
0
[2]
0
[1]
0
B[4:1]
1111
[4]
1
[3]
1
[2]
1
[1]
1
160.000 ns
0 ns
10 ns
20 ns
30 ns
40 ns
50 ns
60 ns
70 ns
80 ns
90 ns
100 ns
110 ns
120 ns
130 ns
140 ns
150 ns
160
0000 0001 0011 0010 0110 0111 0101 0100 1100 1101 1111 1110 1010 1011 1001 1000
0000 0001 0010 0011 0100 0101 0110 0111 1000 1001 1010 1011 1100 1101 1110 1111

test_gray_to_binary.v
gray_to_binary.v
Untitled 3
Name
Value
G[4:1]
0000
B[4:1]
0000
0.000 ns
0 ns
10 ns
20 ns
30 ns
40 ns
50 ns
60 ns
70 ns
80 ns
90 ns
100 ns
110 ns
120 ns
130 ns
140 ns
150 ns
160
0000 0001 0011 0010 0110 0111 0101 0100 1100 1101 1111 1110 1010 1011 1001 1000
0000 0001 0010 0011 0100 0101 0110 0111 1000 1001 1010 1011 1100 1101 1110 1111

Work 2 : BCD to Seven Segment Display Decoder



Inputs				Segments							
A	B	C	D	a	b	c	d	e	f	g	
0	0	0	0	1	1	1	1	1	1	0	For display 0
0	0	0	1	0	1	1	0	0	0	0	For display 1
0	0	1	0	1	1	0	1	1	0	1	For display 2
0	0	1	1	1	1	1	1	0	0	1	For display 3
0	1	0	0	0	1	1	0	0	1	1	For display 4
0	1	0	1	1	0	1	1	0	1	1	For display 5
0	1	1	0	1	0	1	1	1	1	1	For display 6
0	1	1	1	1	1	1	0	0	0	0	For display 7
1	0	0	0	1	1	1	1	1	1	1	For display 8
1	0	0	1	1	1	1	1	0	1	1	For display 9
1	0	1	0	1	1	1	0	1	1	1	For display A
1	0	1	1	0	0	1	1	1	1	1	For display b
1	1	0	0	1	0	0	1	1	1	0	For display C
1	1	0	1	0	1	1	1	1	0	1	For display d
1	1	1	0	1	0	0	1	1	1	1	For display E
1	1	1	1	1	0	0	0	1	1	1	For display F

CODE

test_seven_seg.v | seven_seg.v | Untitled 5

D:/YEAR 2/SEMESTER 2/DIGITAL DESIGN/LABS/LAB 2/seven_seg.v

```
1   `timescale 1ns / 1ps
2   //////////////////////////////////////////////////////////////////////////////////
21
22
23  module seven_seg(hex,led);
24  input [3:0] hex;
25  output [6:0] led;
26  reg [6:0] led;
27  always @(hex)
28      begin
29          case (hex)
30              0 : led = 7'b1111110;
31              1 : led = 7'b0110000;
32              2 : led = 7'b1101101;
33              3 : led = 7'b1111001;
34              4 : led = 7'b0110011;
35              5 : led = 7'b1011011;
36              6 : led = 7'b1011111;
37              7 : led = 7'b1110000;
38              8 : led = 7'b1111111;
39              9 : led = 7'b1111011;
40              10 : led = 7'b1110111;
41              11 : led = 7'b0011111;
42              12 : led = 7'b1001110;
43              13 : led = 7'b0111101;
44              14 : led = 7'b1001111;
45              15 : led = 7'b1000111;
46              default : led = 7'b1111111;
47          endcase
48      end
49
50  endmodule
```


```
module seven_seg(hex,led);
input [3:0] hex;
output [6:0] led;
reg [6:0] led;
always @(hex)
    begin
        case (hex)
            0 : led = 7'b1111110;
            1 : led = 7'b0110000;
            2 : led = 7'b1101101;
            3 : led = 7'b1111001;
            4 : led = 7'b0110011;
            5 : led = 7'b1011011;
            6 : led = 7'b1011111;
            7 : led = 7'b1110000;
            8 : led = 7'b1111111;
            9 : led = 7'b1111011;
            10 : led = 7'b1110111;
            11 : led = 7'b0011111;
            12 : led = 7'b1001110;
            13 : led = 7'b0111101;
            14 : led = 7'b1001111;
            15 : led = 7'b1000111;
            default : led = 7'b1111111;
        endcase
    end

endmodule
```

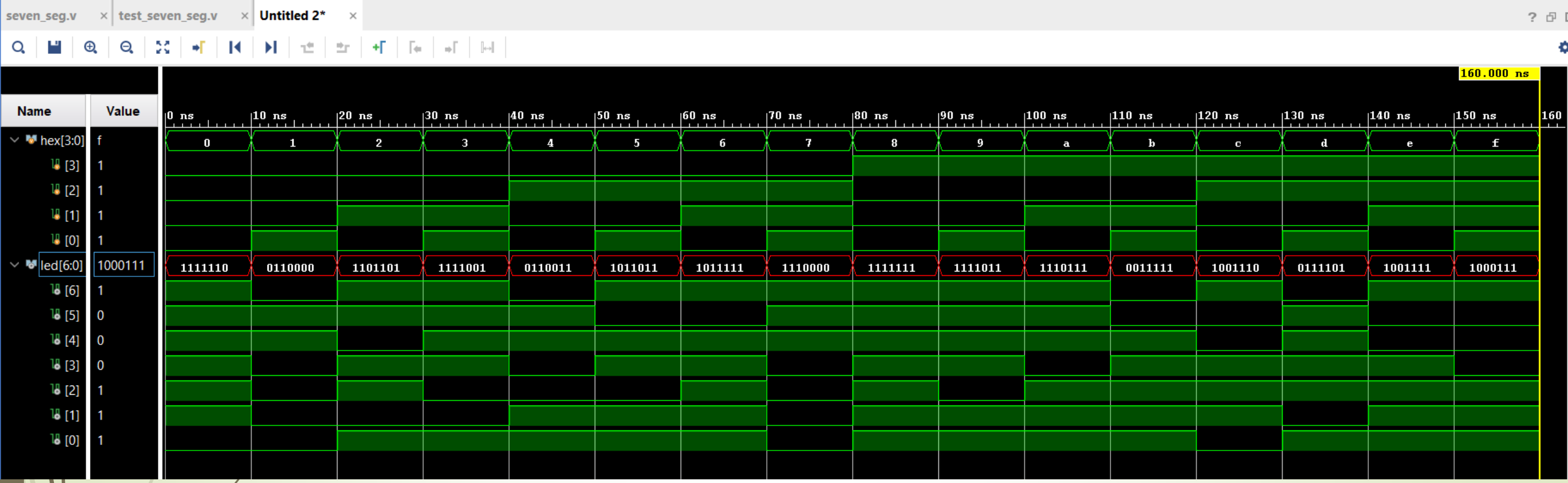
TESTBENCH

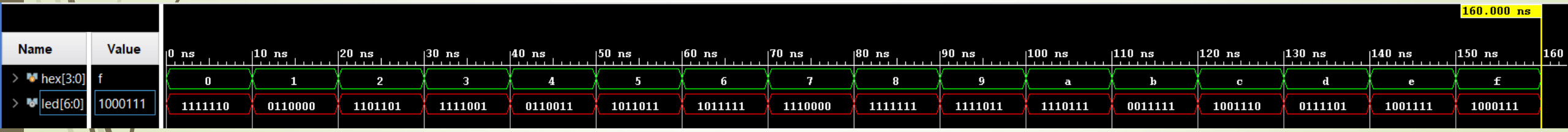
Project Summary | test_seven_seg.v

D:/YEAR 2/SEMESTER 2/DIGITAL DESIGN/LABS/LAB 2/test_seven_seg.v

```
1   `timescale 1ns / 1ps
2   //////////////////////////////////////////////////////////////////////////////////
21
22
23  module test_seven_seg;
24  reg [3:0] hex;
25  wire [6:0] led;
26  seven_seg S1(hex,led);
27
28  initial
29  begin
30      hex = 0;
31      #10 hex = 1;
32      #10 hex = 2;
33      #10 hex = 3;
34      #10 hex = 4;
35      #10 hex = 5;
36      #10 hex = 6;
37      #10 hex = 7;
38      #10 hex = 8;
39      #10 hex = 9;
40      #10 hex = 10;
41      #10 hex = 11;
42      #10 hex = 12;
43      #10 hex = 13;
44      #10 hex = 14;
45      #10 hex = 15;
46  end
47
48  initial #160 $finish;
49  endmodule
```

```
module test_seven_seg;
reg [3:0] hex;
wire [6:0] led;
seven_seg S1(hex,led);

initial
begin
    hex = 0;
    #10 hex = 1;
    #10 hex = 2;
    #10 hex = 3;
    #10 hex = 4;
    #10 hex = 5;
    #10 hex = 6;
    #10 hex = 7;
    #10 hex = 8;
        #10 hex = 9;
        #10 hex = 10;
        #10 hex = 11;
        #10 hex = 12;
        #10 hex = 13;
        #10 hex = 14;
        #10 hex = 15;
end

initial #160 $finish;
endmodule
```


seven_seg.v
test_seven_seg.v
Untitled 2*
Name
Value
hex[3:0]
f
[3]
1
[2]
1
[1]
1
[0]
1
led[6:0]
1000111
[6]
1
[5]
0
[4]
0
[3]
0
[2]
1
[1]
1
[0]
1
160.000 ns
0 ns
10 ns
20 ns
30 ns
40 ns
50 ns
60 ns
70 ns
80 ns
90 ns
100 ns
110 ns
120 ns
130 ns
140 ns
150 ns
160
0
1
2
3
4
5
6
7
8
9
a
b
c
d
e
f
1111110
0110000
1101101
1111001
0110011
1011011
1011111
1110000
1111111
1111011
1110111
0011111
1001110
0111101
1001111
1000111


Name
Value
hex[3:0]
f
led[6:0]
1000111
160.000 ns
0 ns
10 ns
20 ns
30 ns
40 ns
50 ns
60 ns
70 ns
80 ns
90 ns
100 ns
110 ns
120 ns
130 ns
140 ns
150 ns
160
0
1
2
3
4
5
6
7
8
9
a
b
c
d
e
f
1111110
0110000
1101101
1111001
0110011
1011011
1011111
1110000
1111111
1111011
1110111
0011111
1001110
0111101
1001111
1000111

SIMULATION

Work 3

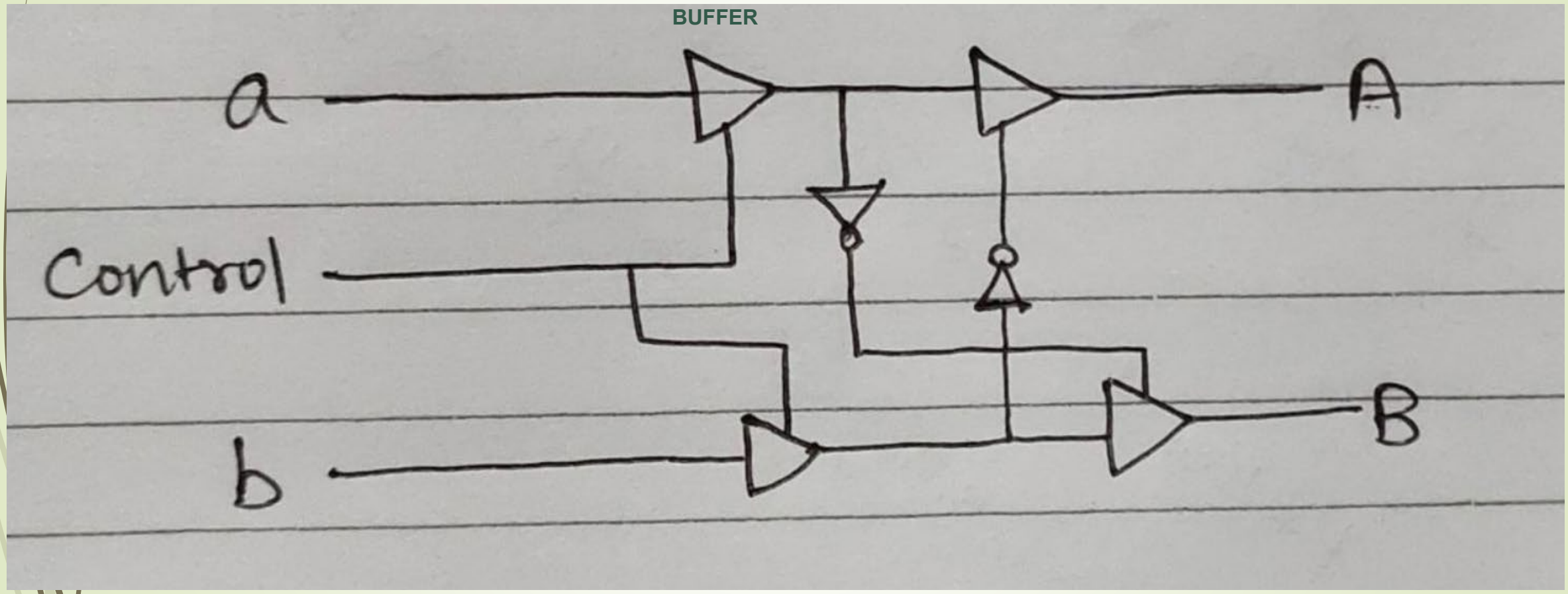
Implement a Buzzer system in Verilog. Write test bench and verify various cases.

- **There are 2 participant with a push button each.- Indicate when to press the button. (This is the host control)**
- **Indicate who is pressing the button first (who wins the chance)**
- **Disable the button of the other participant.**

INPUT			OUTPUT	
a	**b**	**control**	**A**	**B**
0	0	0	X	X
0	0	1	0	0
0	1	0	X	X
1	0	0	X	X
0	1	1	Z	1
1	0	1	1	Z
1	1	1	Z	Z

TRUTH TABLE

TRI STATE
BUFFER



CIRCUIT DIAGRAM

Project Summary | Two_Player_Buzzer.v * | Test_Bench_Two_Player_Bu

D:/YEAR 2/SEMESTER 2/DIGITAL DESIGN/LABS/LAB 2/Two_Player_Buzzer.v

```
1  `timescale 1ns / 1ps
2  //////////////////////////////////////////////////////////////////////////////////
3  // Company:
4  // Engineer:
5  //
6  // Create Date: 20.01.2022 19:26:39
7  // Design Name:
8  // Module Name: Two_Player_Buzzer
9  // Project Name:
10 // Target Devices:
11 // Tool Versions:
12 // Description:
13 //
14 // Dependencies:
15 //
16 // Revision:
17 // Revision 0.01 - File Created
18 // Additional Comments:
19 //
20 //////////////////////////////////////////////////////////////////////////////////
21
22 module Two_Player_Buzzer(a,b,control,A,B);
23 input a,b,control;
24 output A,B;
25 wire p,q;
26
27 assign p = control ? a : 'bz;
28 assign q = control ? b : 'bz;
29 assign A= ~q ? p : 'bz;
30 assign B= ~p ? q : 'bz;
31
32 endmodule
```

CODE

```
module Two_Player_Buzzer(a,b,control,A,B);
input a,b,control;
output A,B;
wire p,q;

assign p = control ? a : 'bz;
assign q = control ? b : 'bz;
assign A= ~q ? p : 'bz;
assign B= ~p ? q : 'bz;

endmodule
```

TESTBENCH

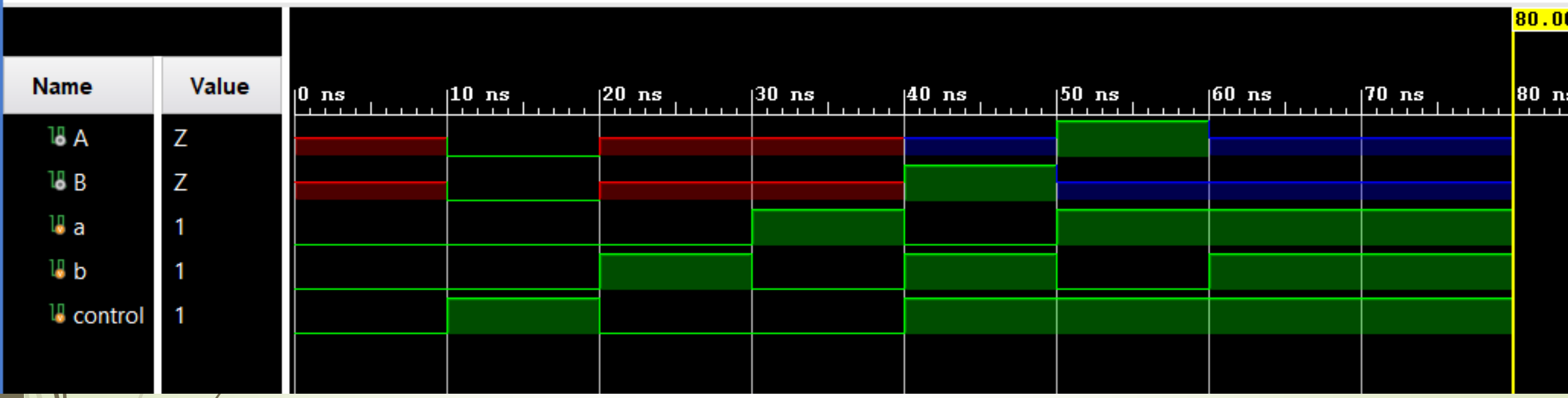
Project Summary | Two_Player_Buzzer.v | **Test_Bench_Two_Player_Buzzer.v**

D:/YEAR 2/SEMESTER 2/DIGITAL DESIGN/LABS/LAB 2/Test_Bench_Two_Player_Buzzer.v

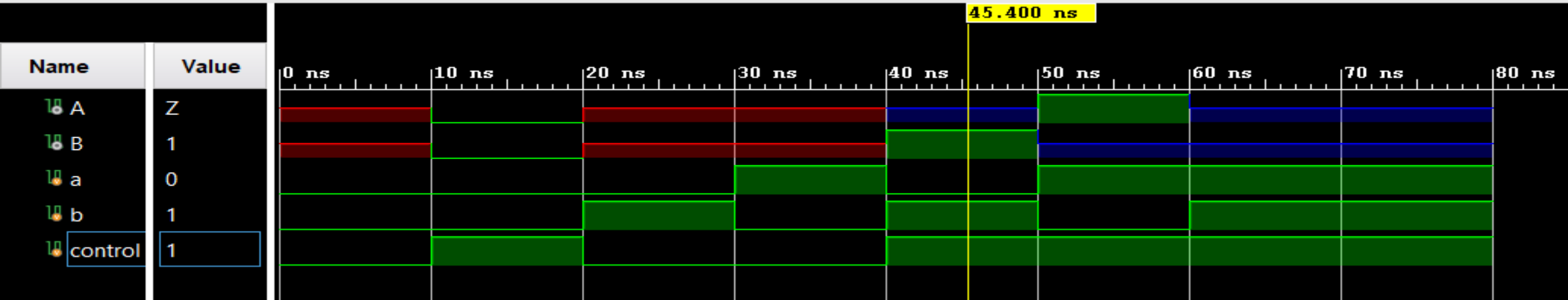
```
1   `timescale 1ns / 1ps
2   ////////////////////////////////////////////////////////////////////////////////
21
22
23  module Test_Bench_Two_Player_Buzzer;
24
25  wire A,B;
26  reg a,b,control;
27
28  Two_Player_Buzzer G(a,b,control,A,B);
29  initial
30  begin
31
32  a = 1'b0; b = 1'b0; control = 1'b0;
33  #10;
34  a = 1'b0; b = 1'b0; control = 1'b1;
35  #10;
36  a = 1'b0; b = 1'b1; control = 1'b0;
37  #10;
38  a = 1'b1; b = 1'b0; control = 1'b0;
39  #10;
40  a = 1'b0; b = 1'b1; control = 1'b1;
41  #10;
42  a = 1'b1; b = 1'b0; control = 1'b1;
43  #10;
44  a = 1'b1; b = 1'b1; control = 1'b1;
45  #10;
46  end
47  initial #80 $finish;
48  endmodule
```

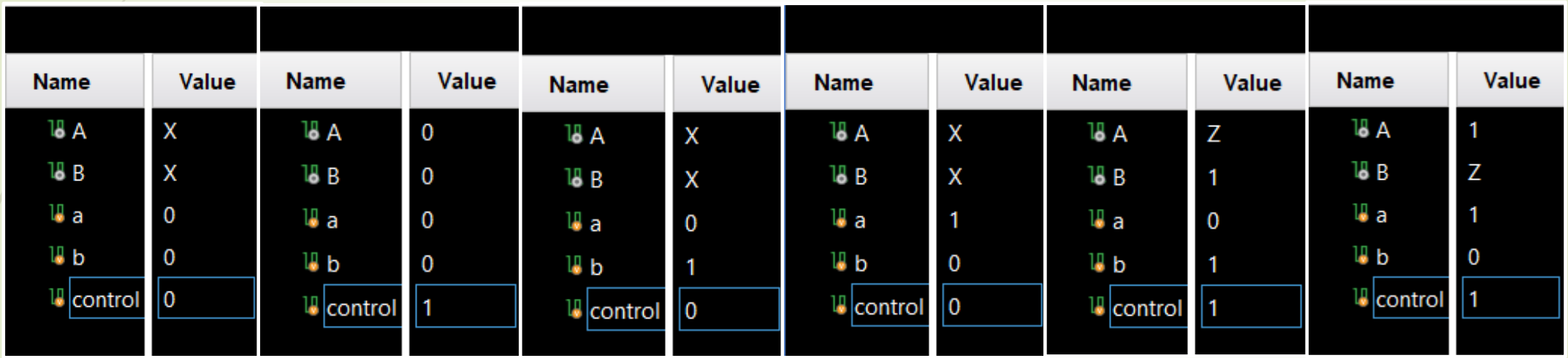
```
module Test_Bench_Two_Player_Buzzer;

wire A,B;
reg a,b,control;

Two_Player_Buzzer G(a,b,control,A,B);
initial
begin

a = 1'b0; b = 1'b0; control = 1'b0;
#10;
a = 1'b0; b = 1'b0; control = 1'b1;
#10;
a = 1'b0; b = 1'b1; control = 1'b0;
#10;
a = 1'b1; b = 1'b0; control = 1'b0;
#10;
a = 1'b0; b = 1'b1; control = 1'b1;
#10;
a = 1'b1; b = 1'b0; control = 1'b1;
#10;
a = 1'b1; b = 1'b1; control = 1'b1;
#10;
end
initial #80 $finish;
endmodule
```

Two_Player_Buzzer.v | Test_Bench_Two_Player_Buzzer.v | Untitled 2



SIMULATION




Name
Value
A
X
B
X
a
0
b
0
control
0
Name
Value
A
0
B
0
a
0
b
0
control
1
Name
Value
A
X
B
X
a
0
b
1
control
0
Name
Value
A
X
B
X
a
1
b
0
control
0
Name
Value
A
Z
B
1
a
0
b
1
control
1
Name
Value
A
1
B
Z
a
1
b
0
control
1

THANK YOU