EEL2020 DIGITAL DESIGN
LAB 2

Name: Dhruv
Roll No.: B20EE016

Work 1: (a) Binary to Gray Code Converter

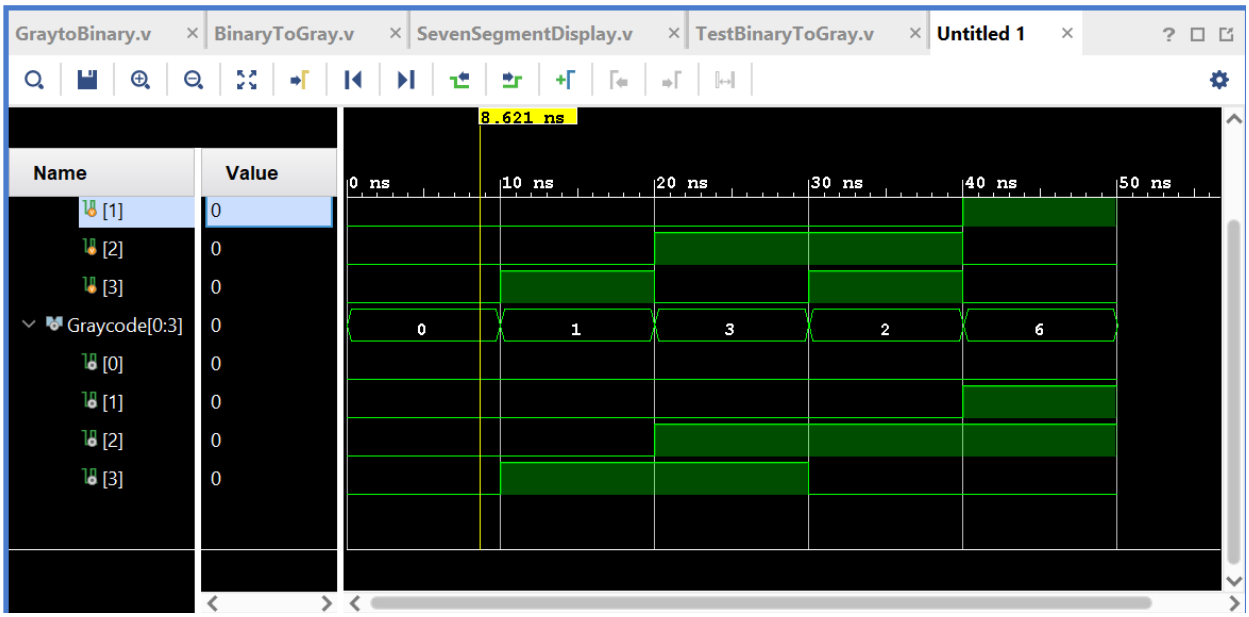
Binary to Gray Code Converter Verilog Code:

```
`timescale 1ns / 1ps
module BinaryToGray(Binary,Graycode);
   input [0:3] Binary;
   output [0:3] Graycode;
   assign Graycode[0]=Binary[0];
   assign Graycode[1]= Binary[1]^Binary[0];
   assign Graycode[2]=Binary[2]^Binary[1];
   assign Graycode[3]=Binary[3]^Binary[2];
endmodule
```

Binary to Gray Code Converter Test bench Verilog Code:

```
`timescale 1ns / 1ps
module TestBinaryToGray();
reg [0:3] Binary;
wire [0:3] Graycode;
   BinaryToGray bg( .Binary(Binary),.Graycode(Graycode));
   initial
   begin
   #0 Binary = 4'b0000;
   #10 Binary = 4'b001;
   #10 Binary = 4'b0010;
   #10 Binary = 4'b0011;
   #10 Binary = 4'b0100;
   end
   initial #50 $finish;
endmodule
```

Binary to Gray Code Converter Simulation:



Work 1: (b) Gray code to Binary code Converter

Gray code to Binary code Converter Verilog Code:

```
`timescale 1ns / 1ps

module GraytoBinary(Gray,Bcd);
    input [0:3] Gray;
    output [0:3] Bcd;
    assign Bcd[0] = Gray[0];
    assign Bcd[1] = Gray[0]^Gray[1];
    assign Bcd[2] = Gray[2]^Gray[1]^Gray[0];
    assign Bcd[3] = Gray[3]^Gray[2]^Gray[1]^Gray[0];
```

```
endmodule
```

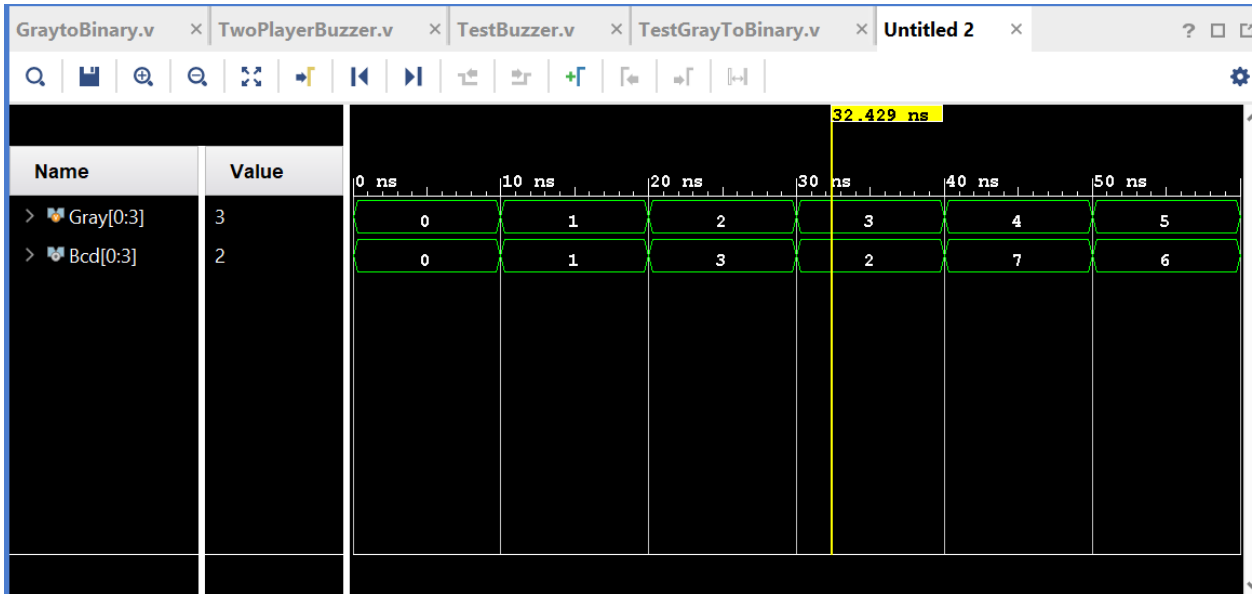
Gray code to Binary code Converter Test bench Verilog Code:

```
`timescale 1ns / 1ps

module TestGrayToBinary();
   reg [0:3] Gray;
   wire [0:3] Bcd;
   GraytoBinary B(.Gray(Gray),.Bcd(Bcd));
   initial
   begin
   #0  Gray = 4'b0000;
   #10 Gray = 4'b0001;
   #10 Gray  = 4'b0010;
   #10 Gray  = 4'b0011;
   #10 Gray  = 4'b0100;
   #10 Gray  = 4'b0101;
   #10 Gray = 4'b0110;
   end
   initial #60 $finish;


endmodule
```

Gray code to Binary code Converter Simulation:



Work 2: Write a program to implement a BCD to 7-segment display decoder.

BCD to 7-segment display decoder Verilog Code:

```
`timescale 1ns / 1ps
module SevenSegmentDisplay(Hex,Led);
   input [0:3] Hex;
   output reg [1:7] 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;
            default: Led = 7'b0000000;
         endcase
      end
endmodule
```

Work 3: Implement a Buzzer system in Verilog. Write a test bench and verify various cases.
- There are 2 participants 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.

Buzzer Verilog Code:

```
`timescale 1ns / 1ps
module TwoPlayerBuzzer(A,B,WinnerA,WinnerB,Control);
   input A,B,Control;
   output WinnerA,WinnerB;
   wire tempa,tempb;
   assign tempa=A&&Control;    // when control is 0 host has not aksed the q yet so tempa and tempv is 0
   assign tempb=B&&Control;

   bufif0 C(WinnerA,tempa,tempb);
   bufif0 D(WinnerB,tempb,tempa);
endmodule
```

Buzzer test bench Verilog Code:

```
`timescale 1ns / 1ps
wire WinnerA,WinnerB;
   reg  A,B,Control;

   TwoPlayerBuzzer c(.A(A),.B(B),.WinnerA(WinnerA),.WinnerB(WinnerB),.Control(Control));
   initial
   begin
   A = 0; B= 0; Control = 0;
   #10 A = 1; B= 1; Control = 0;
   #10 A = 1; B= 0; Control = 0;
```

```
    #10 A = 0; B= 1; Control = 0;
    #10 A = 1; B= 0; Control = 1;
    #10 A = 0; B= 1; Control = 1;
    end
    initial #60 $finish;

endmodule
```

Buzzer Simulation: