ASSIGNMENT-1

Digital System Design

Name: - Nirbhay Thakkar

ID: - 21EL092

Course Code: - 3EL41

Q1) Write a Verilog code for 2X4 decoder

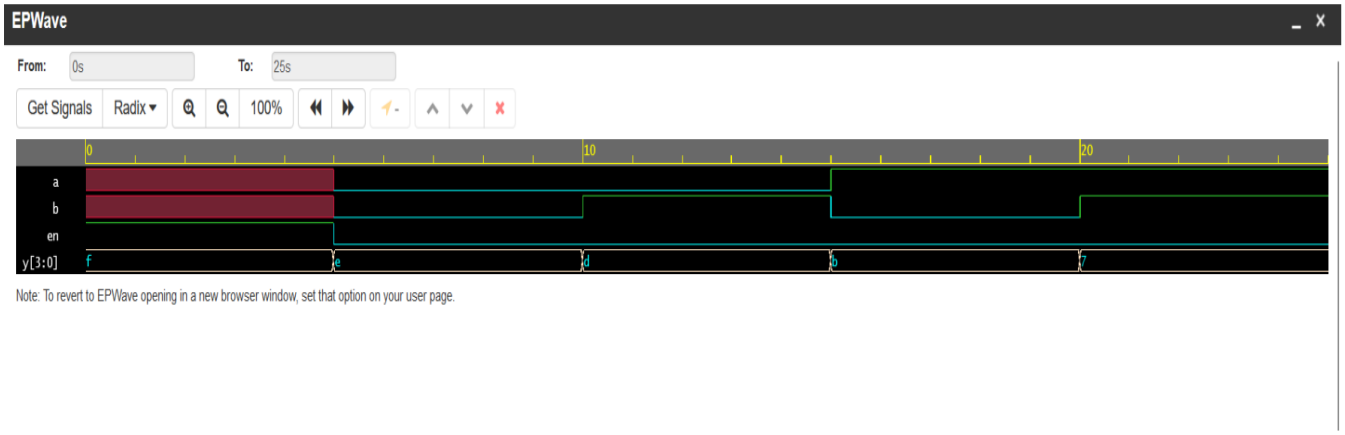
Design :-

```
module decoder(
    input a,b,en;
    output[3:0] y;
);
    wire enb,na,nb;

    assign enb = ~en;
    assign na = ~a;
    assign nb = ~b;
    assign y[0] = ~(na&nb&enb);
    assign y[1] = ~(na&b&enb);
    assign y[2] = ~(a,nb,enb);
    assign y[3] = ~(a,b,enb);

endmodule
```

Testbench :-

```
module tb;

    reg a,b,en;
    wire [3:0]y;

    decoder24_assign dut(en,a,b,y);
    initial begin
    $dumpvars;
    $dumpfile("dump.vcd");
    $monitor("en=%b a=%b b=%b y=%b",en,a,b,y);

    en=1;a=1'bx;b=1'bx;#5
    en=0;a=0;b=0;#5
    en=0;a=0;b=1;#5
    en=0;a=1;b=0;#5
    en=0;a=1;b=1;#5

    $finish;
    end
endmodule
```

Output :-


EPWave
From:
0s
To:
25s
Get Signals
Radix
100%
0
10
20
a
b
en
y[3:0]
f
e
d
b
7
Note: To revert to EPWave opening in a new browser window, set that option on your user page.

Q2) Write a Verilog code for Full subtractor

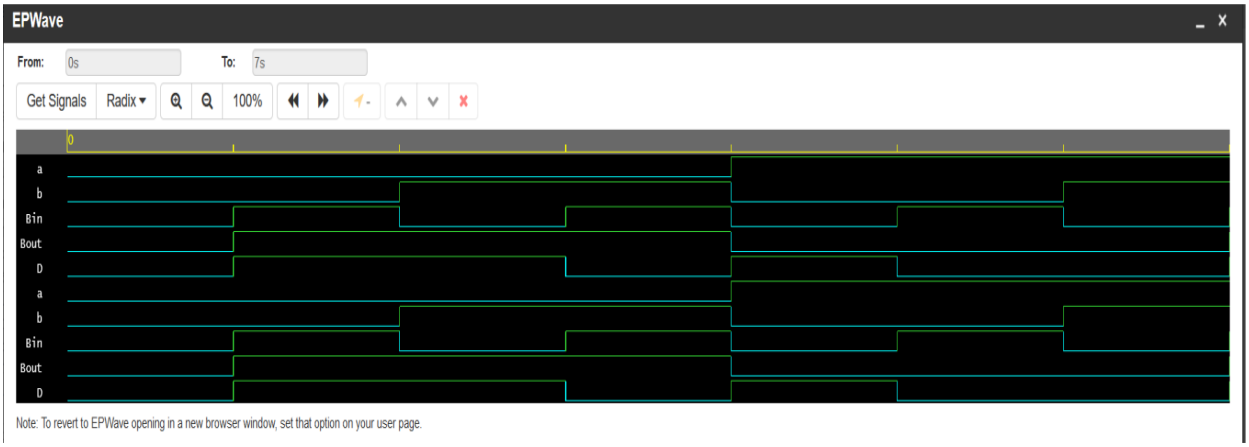
Design :-

```
module full_subtractor(input a, b, Bin, output D, Bout);
    assign D = a ^ b ^ Bin;
    assign Bout = (~a & b) | (~(a ^ b) & Bin);
endmodule
```

Testbench :-

```
module tb_top;
    reg a, b, Bin;
    wire D, Bout;
    full_subtractor fs(a, b, Bin, D, Bout);
    initial begin
    $monitor("At time %0t: a=%b b=%b, Bin=%b, difference=%b, borrow=%b",$time,
        a,b,Bin,D,Bout)
    a = 0; b = 0; Bin = 0; #1;
    a = 0; b = 0; Bin = 1; #1;
    a = 0; b = 1; Bin = 0; #1;
    a = 0; b = 1; Bin = 1; #1;
    a = 1; b = 0; Bin = 0; #1;
    a = 1; b = 0; Bin = 1; #1;
    a = 1; b = 1; Bin = 0; #1;
    a = 1; b = 1; Bin = 1;
    end
endmodule
```

Output :-



Q3) Write a Verilog code for 2-bit comparator

Design: -

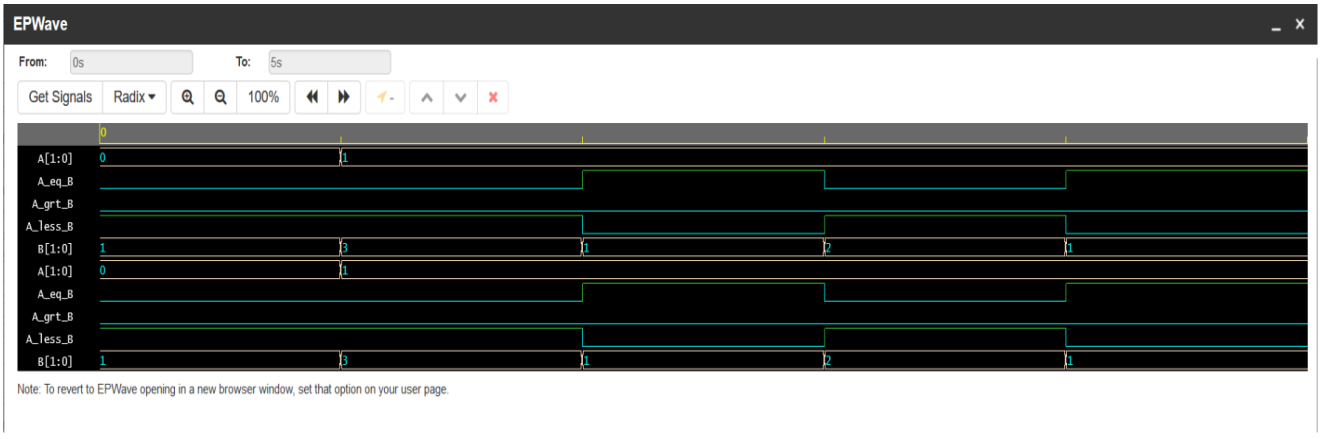
```
module comparator(
    input [1:0] A, B,
    output reg A_grt_B, A_less_B, A_eq_B);
    always@(*) begin
        A_grt_B = 0; A_less_B = 0; A_eq_B = 0;
        if(A>B) A_grt_B = 1'b1;
        else if(A<B) A_less_B = 1'b1;
        else A_eq_B = 1'b1;
end
endmodule
```

Testbench: -

SV/Verilog Testbench

```
module tb;
    reg [1:0] A, B;
    wire A_grt_B, A_less_B, A_eq_B;
    comparator comp(A, B, A_grt_B, A_less_B, A_eq_B);
    initial begin
    $dumpvars;
    $dumpfile("dump.vcd");
      $monitor("A = %0h, B = %0h -> A_grt_B = %0b, A_less_B = %0b, A_eq_B = %0b",
      A, B, A_grt_B,A_less_B,A_eq_B);
        repeat(5)
        begin
            A=$random; B=$random; #1;
        end
    end
endmodule
```

Output: -



4) Write a Verilog code for 3 bit binary to gray convertor.

Design: -

```
module Binary_to_Gray(
    input [3:0] b,
    output [3:0] g
);
    assign g[0]=b[1]^b[0];
    assign g[1]=b[2]^b[1];
    assign g[2]=b[3]^b[2];
    assign g[3]=b[3];

endmodule
```

Testbench: -

```
module Binary_to_Gray_tb;
    reg [3:0]b;
    wire [3:0]g;
    Binary_to_Gray uut (b,g);
    initial begin
    $dumpvars;
    $dumpfile("dump.vcd");
    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
endmodule
```

Output: -


EPWave
From:
0s
To:
150s
Get Signals
Radix
100%
0
20
40
60
80
100
120
140
b[3:0]
g[3:0]
b[3:0]
g[3:0]
Note: To revert to EPWave opening in a new browser window, set that option on your user page.

Q5) Write a Verilog code for BCD to excess 3 convertors.

Design: -

```
module Bcd_excess3(b,e);
    input [3:0] b;
    output [3:0] e;
    assign e[3]=b[3]|b[2]&b[1]|b[2]&b[0];
    assign e[2]=~b[2]&b[1]|~b[2]&b[0]|b[2]&~b[1]&~b[0];
    assign e[1]=b[1]&b[0]|~b[1]&~b[0];
    assign e[0]=~b[0];
endmodule
```

Testbench: -

```
module Bcd_excess3_tb;

    reg [3:0] b;
    wire [3:0] e;

    Bcd_excess3 uut (
    .b(b),
    .e(e)
    );
    initial begin
    $dumpvars;
    $dumpfile("dump.vcd");

    b=3'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;
    end
endmodule
```

Output: -


EPWave
From:
0s
To:
90s
Get Signals
Radix
100%
0
10
20
30
40
50
60
70
80
b[3:0]
e[3:0]
b[3:0]
e[3:0]
Note: To revert to EPWave opening in a new browser window, set that option on your user page.