Birla Vishvakarma Mahavidyalaya
Engineering College

Subject: - Digital System Design (3EL42)

Prof. Chintan Patel

Name: - Vrushti Mehta

ID Number: - 21EL093

Division: - 11-B

Year: - 2023-24

Assignment 1

Q-1 Write a Verilog code for 2*4 decoder.

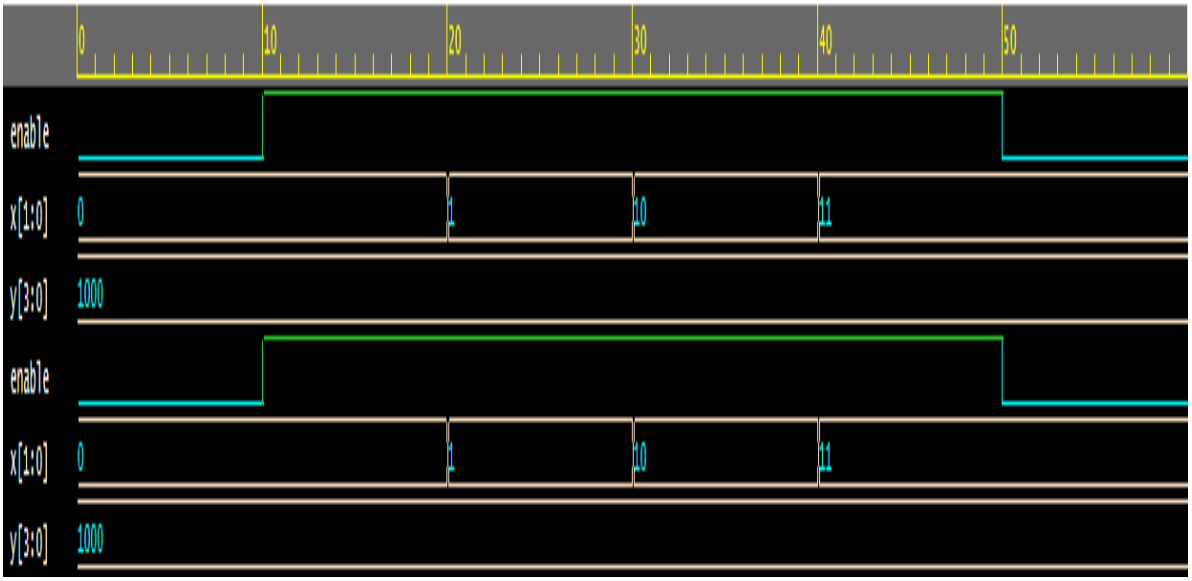
CODE :

```
1   module decoder_24(
2       input [1:0] x,
3       output reg [3:0] y
4       );
5
6         always @ (*)
7        begin : mux
8        y=4'b0000;
9        case(x)
10
11       2'b00  :  y[0] = 1'b1;
12       2'b01  :  y[1] = 1'b1;
13       2'b10  :  y[2] = 1'b1;
14       2'b11  :  y[3] = 1'b1;
15
16       endcase
17       end
18
19  endmodule
```

TESTBENCH :

```
module decoder_24_tb;

reg [1:0]x;
  wire [3:0]y;

  decoder_24 uut(x,y);

  initial begin

      $monitor($time |"x0= %b | x1= %b | y1= %b | y2= %b | y3= %b | y4= %b ",x[0],x[1],y[1],y[2],y[3],y[4]);

  end
    //y = 4'b0000;
   // decoder_24 uut(x,y);
    initial begin
     //  y =4'b0000;

    #10 x[0]=0  ;x[1]=0;
    #10 x[0]=0  ;x[1]=1;
    #10 x[0]=1  ;x[1]=0;
    #10 x[0]=1  ;x[1]=1;

    end

    initial begin

      $dumpfile("dump.vcd");
      $dumpvars(0);

    end

endmodule
```

OUTPUT :



Q-2 Write a Verilog code for full subtractor.

CODE :

```
module full_subtractor(
    input x,
    input y,
    input z,
    output diff,
    output borrow
    );

    assign diff = x^y^z;
    assign borrow = ~x^y | ~x^z | y^z;
endmodule
```

TESTBENCH :

```
module full_subractor_tb;

  reg x,y,z;
  wire diff,borrow;

  initial begin

     $monitor($time |"x = %b | y = %b | z = %b | diff = %b | borrow = %b ",x,y,z,diff,borrow);

  end

   full_subtractor uut(x,y,z,diff,borrow);


  initial begin

    #000 x=0; y=0; z=0;
    #100 x=0; y=0; z=1;
    #100 x=0; y=1; z=0;
    #100 x=0; y=1; z=1;
    #100 x=1; y=0; z=0;
    #100 x=1; y=0; z=1;
    #100 x=1; y=1; z=0;
    #100 x=1; y=1; z=1;

    #100 $finish;

  end

  initial begin

    $dumpfile("dump.vcd");
    $dumpvars(0);

  end
```

OUTPUT:

```
At time 0: a=0 b=0, Bin=0, difference=0, borrow=0
At time 1: a=0 b=0, Bin=1, difference=1, borrow=1
At time 2: a=0 b=1, Bin=0, difference=1, borrow=1
At time 3: a=0 b=1, Bin=1, difference=0, borrow=1
At time 4: a=1 b=0, Bin=0, difference=1, borrow=0
At time 5: a=1 b=0, Bin=1, difference=0, borrow=0
At time 6: a=1 b=1, Bin=0, difference=0, borrow=0
At time 7: a=1 b=1, Bin=1, difference=1, borrow=1
```

Q-3 Write a Verilog code for 2-bit comparator.

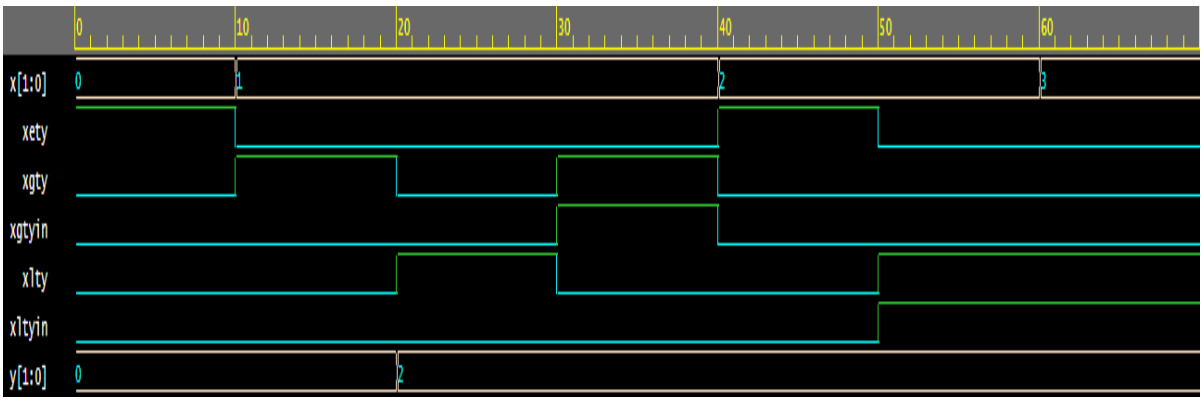
CODE :

```
module comparator_2bit(
    input x1,
    input x0,
    input y1,
    input y0,
    output x_greater_than_y,
    output x_less_than_y,
    output x_equal_to_y
    );
    wire a,b;
    assign a=(x1^y1);
    assign b=(x0^y0);
    assign x_greater_than_y =  x1*(~y1) |( ~a*x0*(~y0));
    assign x_less_than_y = (~x1)*y1 |(~a*(~x0)*y0);
    assign x_equal_to_y =  ~b*~a;

endmodule
```

TESTBENCH :

```
module comparator_2bit_tb;

  reg x0,x1,y0,y1;
  wire x_greater_than_y, x_less_than_y,x_equal_to_y;

  initial begin

     $monitor($time |"x0 = %b | x1 = %b | y0 = %b | y1 = %b | x_greater_than_y = %b | x_equal_to_y = %b | x

  end

   comparator_2bit uut(x0,x1,y0,y1,x_greater_than_y,x_less_than_y,x_equal_to_y);


  initial begin

    #000 x0=1'b0; x1=1'b0; y0=1'b0; y1=1'b0;
    #100 x0=1'b0; x1=1'b1; y0=1'b0; y1=1'b1;
    #100 x0=1'b1; x1=1'b0; y0=1'b1; y1=1'b0;
    #100 x0=1'b1; x1=1'b1; y0=1'b1; y1=1'b1;

    #100 $finish;

  end

  initial begin
    $dumpfile("dump.vcd");
    $dumpvars(0);
  end

endmodule
```

OUTPUT:-



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

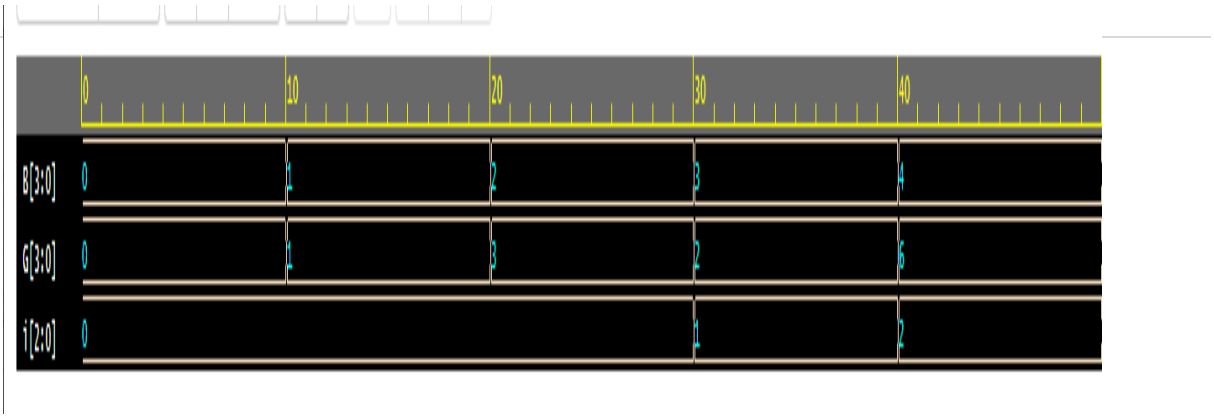
CODE :

```
1   module grey_3(
2       input a,
3       input b,
4       input c,
5       output x,
6       output y,
7       output z
8       );
9
10      assign x = a;
11      assign y= a^b;
12      assign z= b^c;
13
14  endmodule
```

TESTBENCH :

```
1   module grey_3_tb;
2
3     reg a,b,c;
4     wire x,y,z;
5
6     initial begin
7
8         $monitor($time |"a = %b | b = %b | c = %b | x = %b | y = %b | z = %b ",a,b,c,x,y,z);
9
10      end
11
12      grey_3 uut(a,b,c,x,y,z);
13
14      initial begin
15
16          #0   a=0; b=0; c=0;
17          #100 a=0; b=0; c=1;
18          #100 a=0; b=1; c=0;
19          #100 a=0; b=1; c=1;
20          #100 a=1; b=0; c=0;
21          #100 a=1; b=0; c=1;
22          #100 a=1; b=1; c=0;
23          #100 a=1; b=1; c=1;
24
25          #100 $finish;
26      end
27
28      initial begin
29          $dumpfile("dump.vcd");
30          $dumpvars(0);
31        end
32  endmodule
```

OUTPUT :



Q-5 Write a Verilog code for BCD to excess 3 convertor.

CODE :

```
1   module bcd_excess3(
2       input w,
3       input x,
4       input y,
5       input z,
6       output a,
7       output b,
8       output c,
9       output d
10      );
11
12      assign a = w | (x*y) | (x*z);
13      assign b = ~x*y | ~x*z | x*(~y)*(~z);
14      assign c = ~(y^z);
15      assign d = ~z;
16
17  endmodule
```

TESTBENCH :

```
1
2   module bcd_excess3_tb;
3
4       reg w,x,y,z;
5       wire a,b,c,d;
6
7       initial begin
8
9           $monitor($time |"w = %b | x = %b | y = %b | z = %b | a = %b | b = %b | c = %b | d = %b",w,x,y,z,a,b,c,d);
10
11        end
12
13
14      bcd_excess3 uut(w,x,y,z,a,b,c,d);
15
16      initial begin
17
18          #000 w=0; x=0; y=0; z=0;
19          #100 w=0; x=0; y=0; z=1;
20          #100 w=0; x=0; y=1; z=0;
21          #100 w=0; x=0; y=1; z=1;
22          #100 w=0; x=1; y=0; z=0;
23          #100 w=0; x=1; y=0; z=1;
24          #100 w=0; x=1; y=1; z=0;
25          #100 w=0; x=1; y=1; z=1;
26          #100 w=1; x=0; y=0; z=0;
27          #100 w=1; x=0; y=0; z=1;
28          #100 $finish;
29
30      end
31
32      initial begin
33          $dumpfile("dump.vcd");
34          $dumpvars(0);
35        end
36
37  endmodule
```

OUTPUT :