DIGITAL SYSTEM DESIGN (3EL42)

ASSIGNMENT – 1

Q-1: Write a Verilog code for 2X4 decoder.

CODE :

testbench.sv

```
//2*4 decoder (21EL026)
module tb;
  reg a,b,en;
  wire [3:0]y;
  decoder2_4 dut(en,a,b,y);

  initial
    begin
      $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
```

design.sv

SV/Verilog Design

```
module decoder2_4(en,a,b,y);
   input en,a,b;

   output reg [3:0]y;

   always @(en,a,b)
     begin
       if(en==0)
         begin
           if(a==1'b0 & b==1'b0) y=4'b1110;
           else if(a==1'b0 & b==1'b1) y=4'b1101;
           else if(a==1'b1 & b==1'b0) y=4'b1011;
           else if(a==1 & b==1) y=4'b0111;
           else y=4'bxxxx;
         end
       else
        y=4'b1111;
     end
endmodule
```

OUTPUT :

Log Share

```
[2022-08-07 05:34:07 EDT] iverilog '-Wall' design.sv testbench.sv  && unbuffer vvp a.out
en=1 a=x b=x y=1111
en=0 a=0 b=0 y=0000
en=0 a=0 b=1 y=0001
en=0 a=1 b=0 y=0101
en=0 a=1 b=1 y=1000
Done
```

Q-2: Write a Verilog code for Full subtractor.

CODE :

testbench.sv

```
//full_sub (21EL026)
module tb_top;
  reg a, b, Bin;
  wire D, Bout;

  full_subtractor(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
```

design.sv

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

OUTPUT :

Log Share

```
[2022-08-07 05:47:04 EDT] iverilog '-Wall' design.sv testbench.sv  && unbuffer vvp a.out
a=0 b=0 cin=0 d=0 bout=0
a=0 b=1 cin=0 d=1 bout=1
a=1 b=0 cin=0 d=1 bout=0
a=1 b=1 cin=0 d=0 bout=0
a=1 b=1 cin=1 d=1 bout=1
Done
```

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

CODE:

testbench.sv

```
//2_bit comparator (21EL026)
module testbench;
 reg [1:0]a,b;
 wire equal,greater,lower;
 comparator_2bit DUT(a,b,equal,greater,lower);
 initial
 begin
 $monitor("a[0]=%b a[1]=%b b[0]=%b b[1]=%b greater =%b
lower =%b equal=%b",a[0],a[1],b[0],b[1],greater,lower,equal);

 a=2'b00;b=2'b01;#5
 a=2'b11;b=2'b10;#5
 a=2'b00;b=2'b00;#5

 $finish;
 end

endmodule
```

design.sv

```
module comparator_2bit(a,b,equal,greater,lower);
output equal ;
output greater ;
output lower ;
input [1:0] a;
input [1:0] b;
assign equal = (a==b) ? 1 : 0;
assign greater = (a>b) ? 1 : 0;
assign lower = (a<b) ? 1 : 0;
endmodule
```

OUTPUT:

Log Share

```
[2022-08-07 05:47:04 EDT] iverilog '-Wall' design.sv testbench.sv  && unbuffer vvp a.out
a=0 b=0 cin=0 d=0 bout=0
a=0 b=1 cin=0 d=1 bout=1
a=1 b=0 cin=0 d=1 bout=0
a=1 b=1 cin=0 d=0 bout=0
a=1 b=1 cin=1 d=1 bout=1
Done
```

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

CODE:

testbench.sv

```
//3_bit binary to gray (21EL026)
module testbench();
 reg [2:0] cin;
 wire [2:0] d;
 bin_to_gray DUT(cin,d);

 initial
 begin
 $monitor("cin = %b d = %b",cin,d);

 cin = 3'b001; #10
 cin = 3'b101; #10
 cin = 3'b111; #10
 cin = 3'b110; #10
 $finish;
 end
endmodule
```

design.sv

SV/Verilog Design

```
module bin_to_gray(input [2:0]cin,output [2:0]d);
assign d[0] = cin[1] ^ cin[0];
assign d[1] = cin[2] ^ cin[1];
assign d[2] = cin[2];
endmodule
```

OUTPUT:

Log Share

```
[2022-08-07 16:23:16 EDT] iverilog '-Wall' design.sv testbench.sv  && unbuffer vvp a.out
cin = 001 d = 001
cin = 101 d = 111
cin = 111 d = 100
cin = 110 d = 101
Done
```

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

CODE:

testbench.sv

```
//BCD to excess 3 convertor (21EL026)
module testbench();
 reg a,b,c,d;
 wire x,y,z,w;
 bcd_to_excess3 DUT(a,b,c,d,x,y,z,w);

 initial
 begin
 $monitor("a = %b b = %b c = %b d = %b, x = %b
y = %b z = %b w = %b",a,b,c,d,x,y,z,w);

 a = 1'b0; b = 1'b1; c = 1'b0; d = 1'b0;#50
 a = 1'b1; b = 1'b1; c = 1'b0; d = 1'b0;#50
 a = 1'b0; b = 1'b0; c = 1'b0; d = 1'b1;#50
 a = 1'b1; b = 1'b0; c = 1'b0; d = 1'b1;#50
 $finish;
 end
endmodule
```

design.sv

```
module bcd_to_excess3(input a,b,c,d,output x,y,z,w
assign x = a | (b & c) | b & d;
assign y = (~b & c) | (~b & d) | (b & ~c & ~d);
assign z = (c & d) | (~c & ~d);
assign w = ~d;
endmodule
```

OUTPUT:

Log Share

```
[2022-08-07 16:39:34 EDT] iverilog '-Wall' design.sv testbench.sv  && unbuffer vvp a.out
a = 0 b = 1 c = 0 d = 0, x = 0 y = 1 z = 1 w = 1
a = 1 b = 1 c = 0 d = 0, x = 1 y = 1 z = 1 w = 1
a = 0 b = 0 c = 0 d = 1, x = 0 y = 1 z = 0 w = 0
a = 1 b = 0 c = 0 d = 1, x = 1 y = 1 z = 0 w = 0
Done
```