Brac University
EEE 412/ ECE 412/ CSE 460
Verilog Design Laboratory

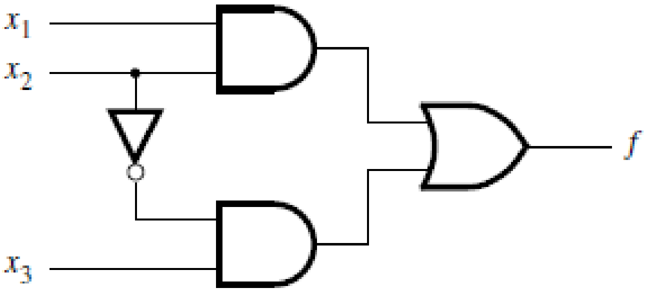
Experiment Title: Familiarization with Multiplexer-Demultiplexer, Encoder-Decoder and Different Types of Counters using Blocking and Non-blocking statements.

Blocking and Non-Blocking Statements

The "<=" operator in Verilog is another aspect of its being a hardware description language as opposed to a normal procedural language. This is known as a **non-blocking assignment**. Its action does not register until after the always block has executed. This means that the order of the assignments is irrelevant and will produce the same result.

The other assignment operator, "=", is referred to as a **blocking assignment**. When "=" assignment is used, for the purposes of logic, the target variable is updated immediately.

Example 1: Construct a 2 to 1 Mux using if-else conditions in Verilog HDL and verify the output using the truth table.



Truth Table

x3	x2	x1	f
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

```
module mux2_1(w,s,f);
input [0:1]w;
input s;
output f;
reg f;

always @(w or s)
      if (s==0)
            f=w[0];
      else if (s==1)
            f=w[1];
endmodule
```

Example 2: Construct a 4 to 1 Mux using case statement in Verilog HDL and verify the output from the timing diagram.

```
module mux4_1(w,s,f);
      input [0:3]w;
      input [0:1]s;
      output reg f;

      always @(w,s)
            case(s)
            0: f=w[0];
            1: f=w[1];
            2: f=w[2];
            3: f=w[3];
            default: f=1'bx;
            endcase
endmodule
```

Example 3: Write down the behavioral representation of Verilog HDL code for a priority encoder with default priority (3>2>1>0).

```
module encoder4_2(w,y);

input [3:0]w;
output reg[1:0]y;

always @(w)
      casex (w)
            4'b1xxx: y=3;
            4'b01xx: y=2;
            4'b001x: y=1;
            4'b0001: y=0;
      endcase
endmodule
```

Example 4: Write down a Verilog HDL code for an up counter with reset operation.

```
module upcounter(rst,clk,q);

input rst,clk;
output reg[3:0]q;

always @(posedge clk, posedge rst)
      begin
            if (rst)
                  q<=0;
            else
                  q<=q+1;
      end
endmodule
```

Example 5: Write down a Verilog HDL code for a 4 bit end around shift register with load operation.

```
module shiftreg(d,load,clk,q);

input [3:0]d;
input load,clk;
output reg[3:0]q;


      always @(posedge clk)
            if (load)
                  q<=d;
            else
                  //q[3:0]<={q[0],q[3:1]};
                  begin
                        q[3]<=q[0];
                        q[2]<=q[3];
                        q[1]<=q[2];
                        q[0]<=q[1];
                  end
endmodule
```

Class Work

1. Design an 8 to 1 MUX using case statement in Verilog HDL.
2. Write down a Verilog HDL code for a priority encoder for the priority 2>1>3>0.
3. Write down a Verilog HDL code for a 10 to 2 down counter with reset operation.

Home Work

Write down the behavioral representation of Verilog HDL code for the following:

1. 1 to 4 Demultiplexer
2. 3 to 8 Decoder
3. Up-Down Counter
4. Ring Counter
5. Johnson Counter