**Chapter 4 Exercise Questions**

Name: **Victor Marrujo**

**1. Explain the difference between blocking and nonblocking assignments in SystemVerilog. Give examples. (6 pts.):**

Blocking assignments are evaluated in the order in which they appear in the code, whilst nonblocking assignments are evaluated concurrently.

**Blocking example:** a and b start as 1 and 2 respectively, after we make it so a=b, and b=3. Our values would be a = 2 and b=3 as soon as they are assigned.

**Nonblocking example:** a and b start as 1 and 2 respectively, after we make it so a=b, and b=3. Our values would **NOT** be a = 2 and b=3 as soon as they are assigned, **INSTEAD** the change would only be applied once oour simulator hits and “end” time stamp.

**2. Correct the errors in the following SystemVerilog code (5 pts.):**

module example (a,b,x1,x2,x3,x4,x5);

input a, b;

output x1, x2, x3, x4, x5;

assign x1 <= a ^ b;

assign x2 <= ~(a & b);

assign x3 <= ~(a | b);

assign x4 <= a | b;

assign x5 <= a & b;

endmodule

**module example (a,b,y1,y2,y3,y4,y5);**

**input a, b;**

**output y1, y2, y3, y4, y5;**

**assign y1 = a ^ b;**

**assign y2 = ~(a & b);**

**assign y3 = ~(a | b);**

**assign y4 = a | b;**

**assign y5 = a & b;**

**endmodule**

3. **Correct the errors in the following SystemVerilog code (4 pts.):**

module example(a1, a2, w0, w1, w2, w3);

input a1;

input a2;

assign w0 = ~a1 & ~a2;

assign w1 = ~a1 & a2;

assign w2 = a1 & ~a2;

assign w3 = a1 & a2;

endmodule

**module example(a1, a2, w0, w1, w2, w3);**

**input a1, a2;**

**output w0, w1, w2, w3;**

**assign w0 = ~a1 & ~a2;**

**assign w1 = ~a1 & a2;**

**assign w2 = a1 & ~a2;**

**assign w3 = a1 & a2;**

**endmodule**

**4. Correct the errors in the following SystemVerilog code (4 pts.):**

module example(a,b,c, Q, R);

input a;

input b;

input c;

output Q

output R

assign Q = a & b’ | b’ & ~c;

assign R = ~a & ~b & ~c

endmodule

**module example(a,b,c, Q, R);**

**input a, b, c;**

**output Q, R;**

**assign Q = a & ~b | ~b & ~c;**

**assign R = ~a & ~b & ~c;**

**endmodule**

**5. Correct the errors in the following SystemVerilog code (6 pts.):**

module example(j, k, clk, q);

input j,k,clk;

output reg q=0;

case()

2'b00:q <= q

2'b01:q <= 0

2'b10:q <= 1

2'b11:q <= ~q

endcase

always@(posedge)

endmodule

**module example(j, k, clk, q);**

**input j,k,clk;**

**output reg q=0;**

**always@(posedge clk)**

**case({j,k})**

**2'b00:q <= q;**

**2'b01:q <= 0;**

**2'b10:q <= 1;**

**2'b11:q <= ~q;**

**endcase**

**endmodule**

**6. Correct the errors in the following SystemVerilog code (5 pts.):**

module example(a, clk, y);

input clk;

output reg y=0;

always(postedge clk)

if (a==1)

y=~y

else

y=y

endmodule

module example(a, clk, y);

input a, clk;

output reg y=0;

always@(posedge clk)

begin

if (a==1)

y=~y;

else

y=y;

end

endmodule

**7. Write the SystemVerilog module to simulate the circuit based on the following equation (5 pts):**

1. y = efg + e'f

module question7a(e, f, g, y );

input e, f, g;

output y;

assign y = e & f & g | ~e & f; // y = efg + e'f

endmodule

1. y = e'f' + ef

module question7b(e, f, y );

input e, f;

output y;

assign y = ~e & f | e & f; // y = e'f' + ef

endmodule