Homework 9

ECE2504 CRN:82729

Jacob Abel

October 24, 2017

Question 1: (4 pts) Recall the structural Verilog description of a ripple carry adder you wrote in class. Compile and simulate your description in Quartus. Apply combinations that check out the rightmost full adder for all eight input combinations; this also serves as a check for the other full adders. Submit the output waveform. Note that you will also need to assign inputs for the other three full adders – all 9 inputs and 5 outputs should be shown in your simulation. Choose your input values to include all eight possible combinations for the least significant full adder.

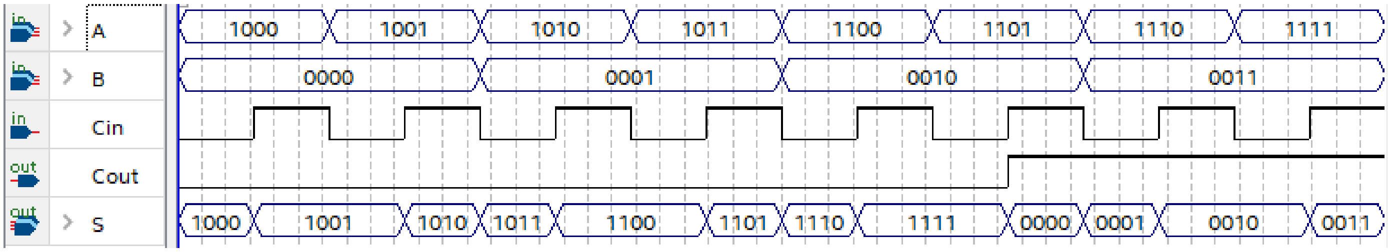

Figure 1: Ripple Carry Adder Simulation Waveform

```
module half_adder(a,b,s,cout);
    input a, b;
    output s, cout;

    xor g0(s, a, b);
    and g1(cout, a, b);
endmodule

module full_adder(a,b,cin,s,cout);
    input a, b, cin;
    output s, cout;
    wire has1, hac1, has2, hac2;
    half_adder g0(a,b,has1,hac1);
    half_adder g1(has1,cin,has2,hac2);
    or g2(cout, hac1, hac2);
    assign s=has2;
endmodule

module ripple_carry_adder(a,b,cin,s,cout);
    input [3:0]a;
    input [3:0]b;
    input cin;
    output [3:0]s;
    output cout;
    wire [2:0]c;
    full_adder fa0(a[0],b[0],cin,s[0],c[0]);
    full_adder fa1(a[1],b[1],c[0],s[1],c[1]);
    full_adder fa2(a[2],b[2],c[1],s[2],c[2]);
    full_adder fa3(a[3],b[3],c[2],s[3],cout);
endmodule
```

Listing 1: 4-bit Ripple Carry Adder

Question 2: (4 pts) Draw the logic diagram that corresponds to the following structural Verilog description.

```
// Combinational Circuit 1: Structural Verilog Description
module comb_ckt_1(f, x1, x2, x3, x4, x5);
    input x1, x2, x3, x4, x5;
    output f;

    wire n1, n2, n3, n4, n5;
    not
        g0(n4, n1),
        g1(n5, x4);
    and
        g2(n1, x1, x2),
        g3(n2, n4, x3),
        g4(n3, n4, n5);
    or
        g6(f, n1, n2, n3, x5);
endmodule
```

Listing 2: Combinational Circuit 1

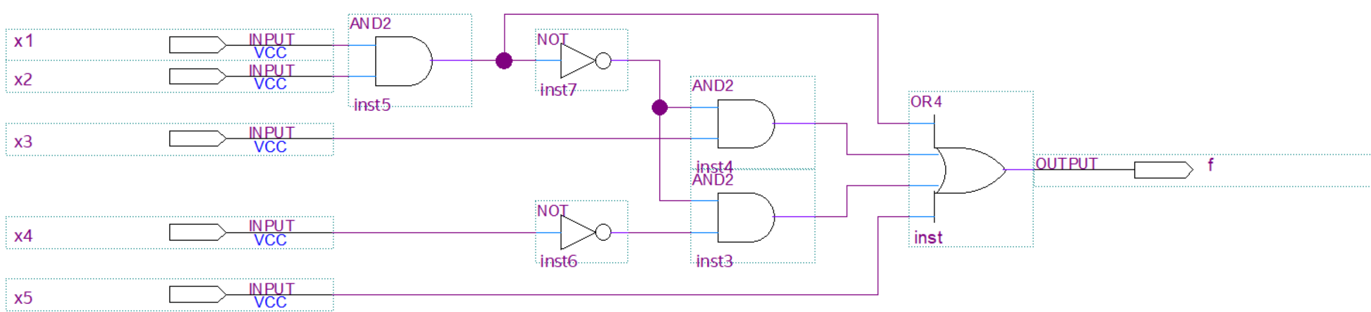

Figure 2: Combinational Circuit 1 Logic Diagram

Question 3: (8 pts)

a) Using the Verilog description in the previous problem as a framework, write a structural Verilog description of the circuit shown below.
b) Compile and simulate the circuit in Quartus. Submit the waveform showing all inputs and outputs.

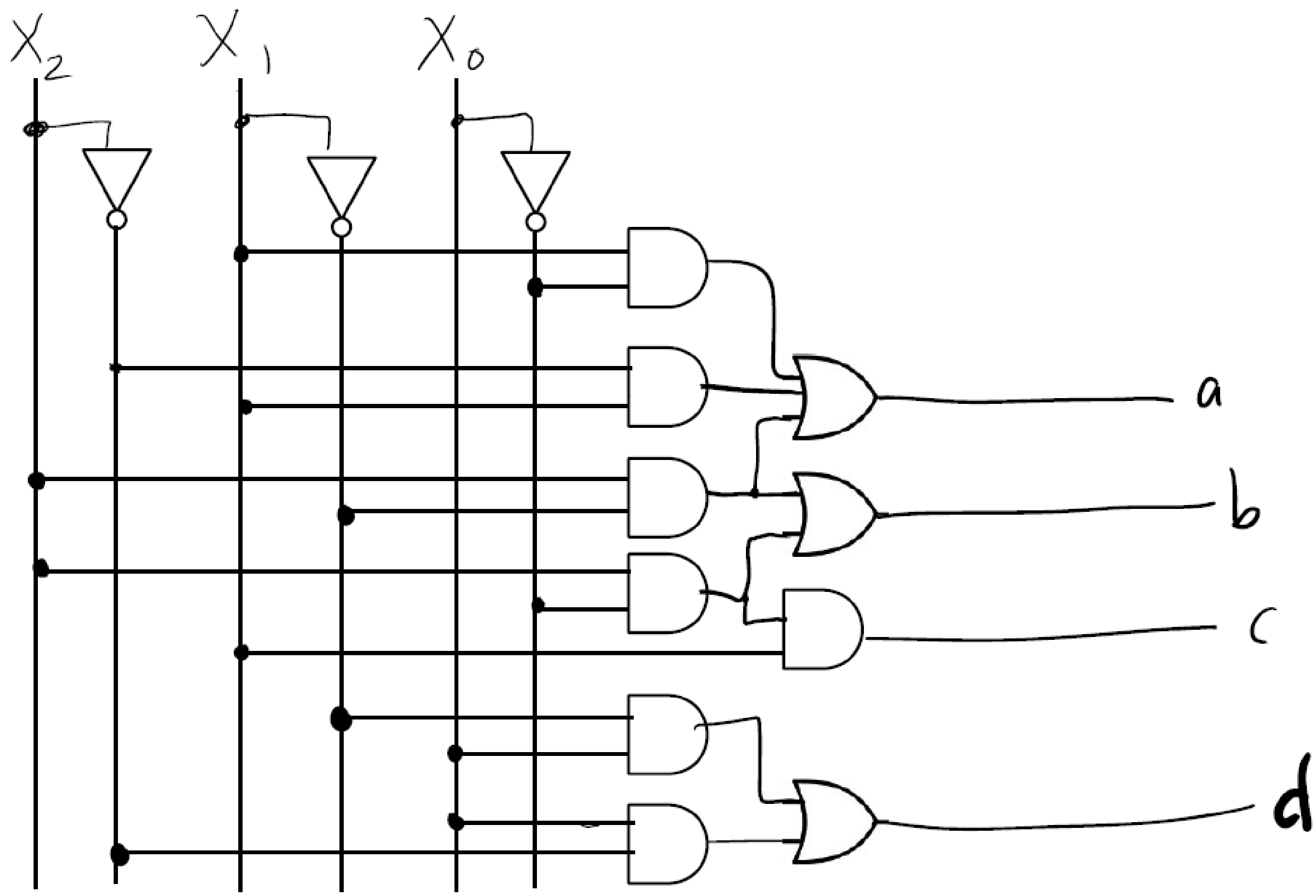

Figure 3: Problem 3 Logic Diagram

```verilog
module p3_comb_ckt (x2, x1, x0, a, b, c, d);
    input x2, x1, x0;
    output a, b, c, d;
    wire [2:0]nx;
    wire [5:0]lvl1o;

    not
        n0(nx[0], x0),
        n1(nx[1], x1),
        n2(nx[2], x2);
    and
        a0(lvl1o[0], x1, nx[0]),
        a1(lvl1o[1], x1, nx[2]),
        a2(lvl1o[2], x2, nx[1]),
        a3(lvl1o[3], x2, nx[0]),
        a4(lvl1o[4], x0, nx[1]),
        a5(lvl1o[5], x0, nx[2]),
        a6(c, lvl1o[3], x1);
    or
        o0(a, lvl1o[0], lvl1o[1], lvl1o[2]),
        o1(b, lvl1o[2], lvl1o[3]),
        o2(d, lvl1o[4], lvl1o[5]);
endmodule
```

Listing 3: Problem 3 Structural Verilog Description

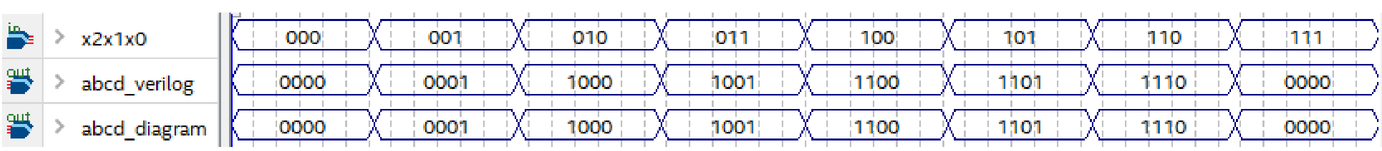

Figure 4: Problem 3 Simulation Waveform

Question 4:

a) (6 pts) Draw the logic diagram that represents minimum two-level logic needed to implement the following Verilog dataflow description.

b) (1 pt) what type of circuit is this ? (i.e. what does it do?)

```verilog
// Combinational Circuit 2: Dataflow Verilog Description
module comb_ckt_2(y, a, b, c, d, e, f);
    input a, b, c, d, e, f;
    output y;
    wire n1, n2, n3, n4, n5;

    assign n1 = (~e & ~f) & a;
    assign n2 = e | ~f;
    assign n3 = e & ~f & c;
    assign n4 = ~d + ~e + ~f;
    assign n5 = ~n2 & b;
    assign y = n1 | n3 | ~n4 | n5;
endmodule
```

Listing 4: Combinational Circuit 2

ab \ cd	00	01	11	10
00	0	1	0	1
01	1	0	1	0
11	1	0	1	1
10	0	1	0	1

$ef = 00$

ab \ cd	00	01	11	10
00	0	1	0	1
01	1	1	1	0
11	1	1	1	1
10	0	1	0	1

$ef = 01$

ab \ cd	00	01	11	10
00	1	1	0	1
01	1	0	1	0
11	1	0	1	1
10	1	1	0	1

$ef = 10$

ab \ cd	00	01	11	10
00	1	1	0	1
01	1	1	1	0
11	1	1	1	1
10	1	1	0	1

$ef = 11$

$$(a' + b + c + e)(b + c' + d')(b' + c + d' + f)(a + b' + c' + d)$$

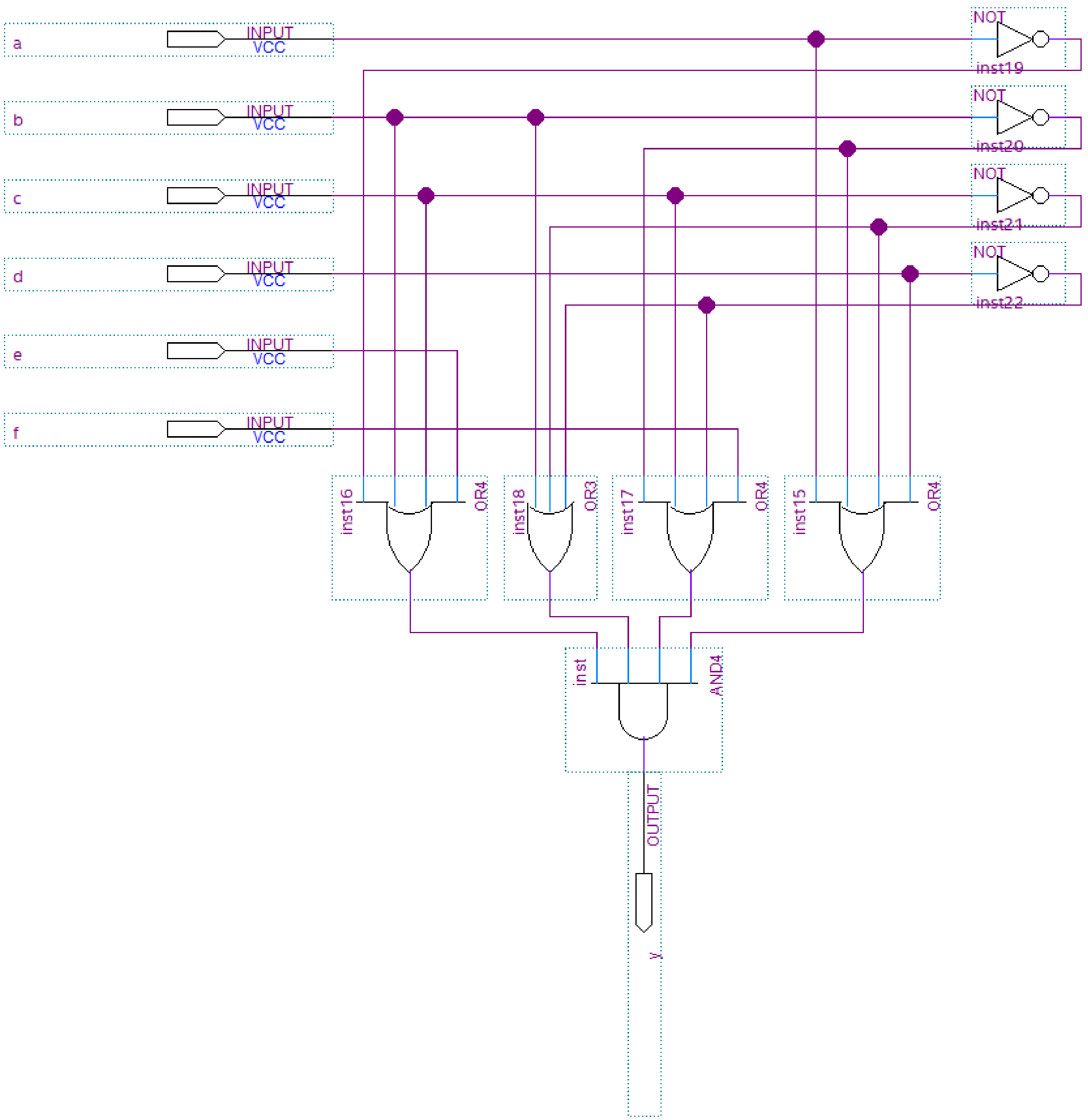

Figure 5: Combinational Circuit 2 Logic Diagram

Question 5: (3 pt) Determine the (2-input) gate count and propagation delay for circuit diagram you found for the previous problem. Use a form similar to HW8, Problem 1 (where t_{pdNOT} = propagation delay of an inverter, t_{pdAND} = propagation delay of a 2-input AND gate, and t_{pdOR} = propagation delay of a 2-input OR gate).

Question 6: (3 pt) Determine the (2-input) gate count and propagation delay for circuit diagram described in Figure 3-33 (below). Use a form similar to HW8, Problem 1 (where t_{pdNOT} = propagation delay of an inverter, t_{pdAND} = propagation delay of a 2-input AND gate, and t_{pdOR} = propagation delay of a 2-input OR gate).

```verilog
// 4-to-1-Line Multiplexer: Dataflow Verilog Description
module multiplexer_4_to_1_cf_v(S, I, Y);
    input [1:0] S;
    input [3:0] I;
    output Y;

   assign Y = (S ==2'b00) ? I[0] :
               (S ==2'b01) ? I[1] :
               (S ==2'b10) ? I[2] :
               (S ==2'b11) ? I[3] : 1'bx;
endmodule
```

Listing 5: 4-to-1-Line Multiplexer

Question 7: (8 pts) Using the conditional dataflow concept from Figure 3-33 (above), complete the following Verilog dataflow description for an 3x8 decoder using the conditional operator. Compile and simulate your description with a set of inputs that are a good test for the selection function it performs. Submit the waveform.

```
module decoder_3_to_8 (D, A);
    input [2:0] A;
    output [7:0] D;

    assign D[7:0] = (A == 3'b000) ? 8'b00000001 :
                    (A == 3'b001) ? 8'b00000010 :
                    (A == 3'b010) ? 8'b00000100 :
                    (A == 3'b011) ? 8'b00001000 :
                    (A == 3'b100) ? 8'b00010000 :
                    (A == 3'b101) ? 8'b00100000 :
                    (A == 3'b110) ? 8'b01000000 :  8'b10000000;
endmodule
```

Listing 6: 3x8 Decoder

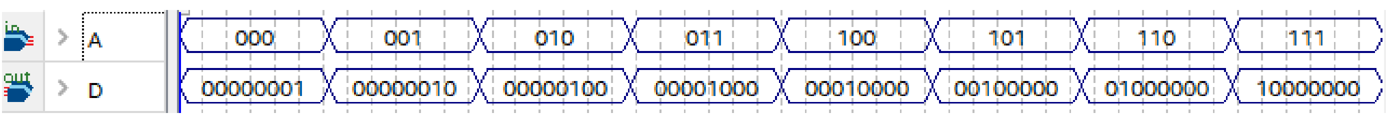

Figure 6: 3x8 Decoder Simulation Waveform

Question 8: (8 pts) Using the module definitions from Problem 1 (4-bit ripple carry adder) and Problem 6 (Figure 3-33), implement the arithmetic circuit shown in the Case Study 2 Sample Solution. You will need to instantiate the ripple carry adder module and as many muxes as you need into your Verilog model. Compile and simulate the circuit in Quartus. Submit the waveform showing all inputs and outputs. Use the following inputs.

R_1R_0	c_0	X	Y
00	0		
00	1		
01	0		
01	1	0100	1101
10	0		
10	1		
11	0		

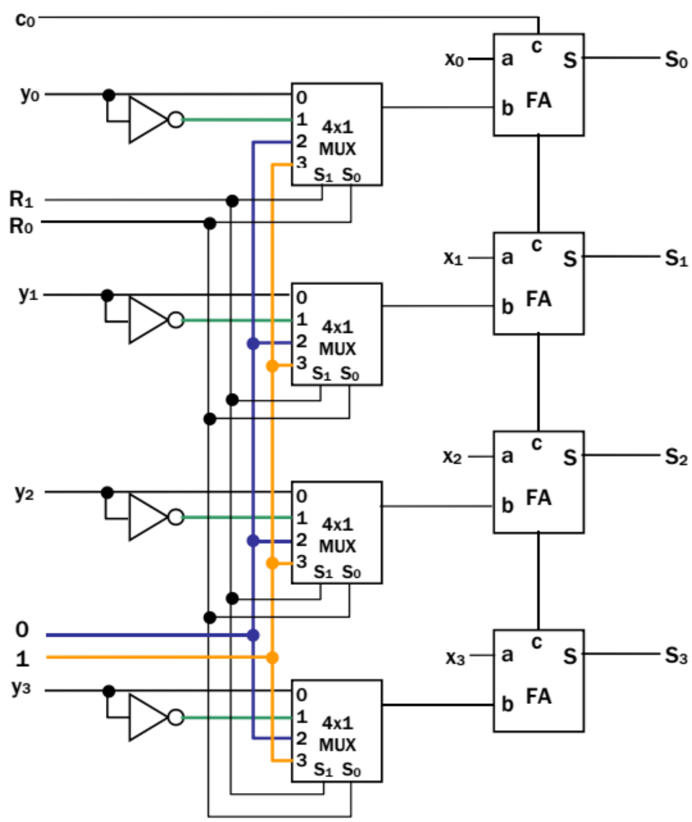

Figure 7: Case Study 2 Logic Diagram

```
module case_study_2 (R, x, y, cin, S);
    input cin;
    input [2:0]R;
    input [3:0]x, y;
    output [3:0]S;
    wire [3:0]muxo;

    multiplexer_4_to_1_cf_v
        m0(R, {y[0], ~y[0],1'b0,1'b1}, muxo[0]),
        m1(R, {y[1], ~y[1],1'b0,1'b1}, muxo[1]),
        m2(R, {y[2], ~y[2],1'b0,1'b1}, muxo[2]),
        m3(R, {y[3], ~y[3],1'b0,1'b1}, muxo[3]);
    ripple_carry_adder
        rca(x, muxo, cin, S,);
endmodule
```

Listing 7: Case Study 2

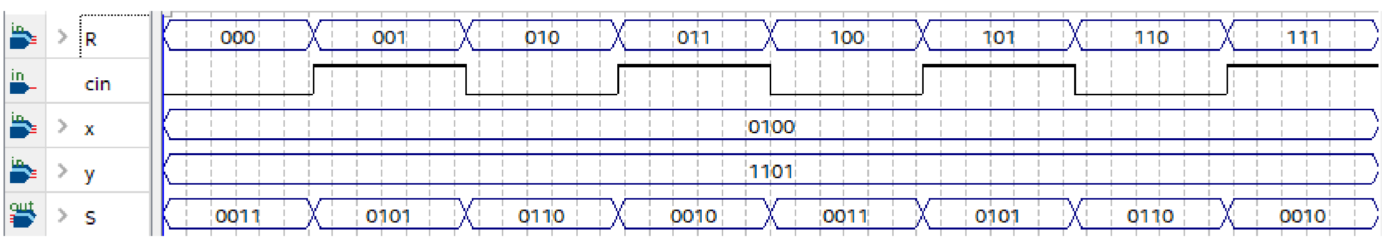

Figure 8: Case Study 2 Simulation Waveform

Question 9: (8 pts) Determine the (2-input) gate count and propagation delay for

a) the circuit you designed in the previous problem.
b) the circuit shown below.

Use a form similar to HW8, Problem 1 (where t_{pdNOT} = propagation delay of an inverter, t_{pdAND} = propagation delay of a 2-input AND gate, and t_{pdOR} = propagation delay of a 2-input OR gate).

```
// Combinational Circuit 3
module comb_ckt_3(S,X,Y,R);
    input [3:0]X,Y;
    input [2:0]R;
    output [3:0]S;
    wire [3:0]n1, n2, n3, n4, n5;

    assign n1 = X + Y;
    assign n2 = X + Y + 1;
    assign n3 = X + ~Y;
    assign n4 = X + ~Y + 1;
    assign n5 = X;
    assign n6 = X + 1;
    assign n7 = X - 1;
    assign S = (R == 3'b000) ? n1 :
               (R == 3'b001) ? n2 :
               (R == 3'b010) ? n3 :
               (R == 3'b011) ? n4 :
               (R == 3'b100) ? n5 :
               (R == 3'b101) ? n6 :
               (R == 3'b110) ? n7 : 4'bx;
endmodule
```

Listing 8: Combinational Circuit 3

GRADING SCALE

Total: 53 pts

Pts	0	6	13	19	26	32	39	46
Letter Grade	D-	D	C-	C	B-	B	A-	A