COMP311-5: Logic Circuit Design

Spring 2019, Prof. Taigon Song

Final Exam: June 18, 15:00 – 16:30pm [Total: 120 points]

Guidelines:

1. Read the questions carefully and pay attention to the special instructions.
2. Show all your mark to receive full credit.
3. State any assumptions you make on your solution.
4. Total number of pages in this exam: 5 (including this cover page.)
5. Write your name at ALL pages.
6. For each page with unwritten name or student ID: -5 points

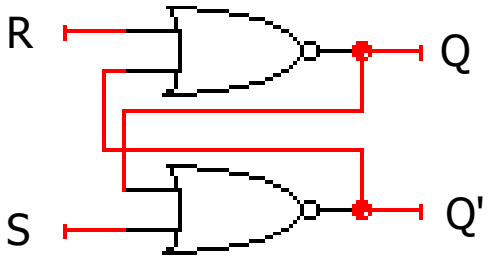
#ID / Name: _____

Prob. 1 (15 pts)	
Prob. 2 (10 pts)	
Prob. 3 (20 pts)	
Prob. 4 (20 pts)	
Prob. 5 (20 pts)	
Prob. 6 (20 pts)	
Prob. 7 (10 pts)	
Bonus (5 pts)	
Name penalty (-25 pts)	
Total (120 pts)	

Student ID#: Name:

1. Design an R-S latch based on the schematic below using primitive gates. [15 points]

```
module rslatch (input r, s, output q, qb);









endmodule
```



2. What does 'RTL' represent? Please decrypt the acronym. [10 points]

3. Is the source code below correct? If so, please say "correct" and do nothing. If not, fix the source code so that it would run in the intended purpose [Total: 20 points].

3.1)

```
module comp(in_a, in_b, out_sm, out_eq,
out_bg);
  localparam BITS = 9;
  input [BITS:0] in_a, in_b;
  output reg out_sm, out_eq, out_bg;

  always @(*) begin
    if(in_a < in_b) begin
       out_sm <= 1'b1;
           out_eq <= 1'b0;
           out_bg <= 1'b0;
    end
        else if(in_a == in_b) begin
       out_sm <= 1'b0;
           out_eq <= 1'b1;
           out_bg <= 1'b0;
        end
        else begin
       out_sm <= 1'b0;
           out_eq <= 1'b0;
           out_bg <= 1'b1;
        end
  end
endmodule
```

```
`timescale 1ns/1ns
module comp4b_tb();
  reg[12:0] in_a, in_b;
  wire out_sm, out_eq, out_bg;

  comp #(.BITS(12))
u1(.in_a(in_a), .in_b(in_b), .out_sm(out_sm),
           .out_eq(out_eq), .out_bg(out_bg));

  initial begin
    in_a = 0; in_b = 0;
  end

  always begin
    #4.1 in_a = $random; in_b = $random;
  end

endmodule
```

3.2)

```
module vcounter(
  input [3:0] d,
  input rst, clk, ld, up,
  output reg [3:0] q,
  output reg cp, cn);

  always @(posedge clk, negedge rst) fork
    if(~rst)
      q <= 0;
    else if (ld)
      q <= d;
    else if (up)
      if(q < 9) fork
        q <= q+1;
              cp <= 0;
          join
          else fork
            q <= 0;
              cp = 1;
          join
    else
      if(q > 0) fork
        q <= q-1;
              cn = 0;
      join
          else fork
        q = 9;
              cn = 1;
      join
  join
endmodule
```

4. Please choose the correct answer [Total 20 points, each 2 points]

A function can enable another function but not another task.	O	X
UDPs can have multiple output terminals	O	X
UDPs do not handle 'x' values	O	X
Tasks must have at least one input argument. They can have more than one input.	O	X
UDPs (user-defined primitives) can take vector input terminals	O	X
Functions may contain delay, event, or timing control statements	O	X
UDPs can be defined inside other modules	O	X
Functions may execute in non-zero simulation time	O	X
Vertical partitioning of a design is partitioning my huge module based on bit slices	O	X
Tasks always return a single value.	O	X

5. Please describe what the values of tmp_a and tmp_div be would be when the source code run is complete. (Note, a1 = 16'b1101_1110_1111_0000; b1 = 16'b101;) [Total: 20 points]

```
module div16b(input [15:0] a1, b1, output reg [15:0] out1, output reg underflow);
  integer i;
  reg [15:0] tmp_div, tmp_a;

  always @(*) begin
    if(b1 > a1) underflow = 1'b1;    // underflow condition
        else           underflow = 1'b0;
    tmp_a = a1;
    i=0;
    while (i != 5) begin
      tmp_div = b1;
      if (tmp_div < (tmp_a >> 15-i)) begin
        out1[15-i]=1'b1;
                tmp_div = tmp_div << (15-i);
                tmp_a = tmp_a - tmp_div;
                end
      else
            out1[15-i]=1'b0;
      i = i+1;
    end
  end
endmodule
```

6. Analyze the following source code and describe what is preferred than the other ($random or $random(seed)). If there is no preference over the other, you may write "no preference". Briefly reason why you chose your answer. [20 points]

```
module mux41(output reg [3:0] y, input
[3:0] a0, a1, a2, a3, [1:0] sel);
always @(*) begin
    case (sel)
          2'b00: y <= a0;
          2'b01: y <= a1;
          2'b10: y <= a2;
          default: y <= a3;
        endcase
  end
```

```
module mux41_tb();

  wire [3:0] y;
  reg [3:0] a0, a1, a2, a3;
  reg [1:0] sel;
  integer i;
  integer seed;

  mux41 tgmux41(y, sel, a0, a1, a2, a3);

  initial begin
    seed = 2;
    sel <= 2'b00;
      a0 <= $random; // or "$random(seed)"
      a1 <= $random; // or "$random(seed)"
      a2 <= $random; // or "$random(seed)"
      a3 <= $random; // or "$random(seed)"

  for (i=0; i <4; i=i+1)
    #5 sel <= i;
  end

endmodule
```

7. A statement exists that is very similar to $display, but it is executed after all other statements on the same time are executed. What is this statement? [10 points]