ECE 310 - DN Cplx Dig Sys
Fall 2024
Dr. Mihail Cutitaru

Name: ______________________________

Signature: ______________________________

Midterm Exam
10/09/2024
Time Limit: 75 Minutes

Please put your name and signature in the fields above. By signing this page you agree to abide by the NC State University Code of Student Conduct.

This exam contains 5 questions and 8 pages (including this cover page). Total number of possible points is 100. Please **pay attention to the allocation of points for each question** and **plan your time accordingly**.

Answer the questions in the space provided on the question sheets. If you run out of room for an answer, leave a note and continue on the back of the page. Please write legibly.

Please make sure to turn your phones off or to silence them. If a phone goes off during the exam, you will be deducted 5 points from your score.

Question	Points	Score
1	18	
2	15	
3	15	
4	40	
5	12	
Total:	100	

1. (18 points) Theory Questions. Be concise in your answers.

   (a) (2 points) What is Verilog?

   (b) (4 points) List 1 advantage and 1 disadvantage of a Ripple-Carry Adder (RCA).

   (c) (4 points) Explain when may there be an inferred latch generated as a result of writing Verilog code.

   (d) (4 points) Explain the difference between gate-level (structural) modeling and dataflow modeling.

   (e) (4 points) Explain the difference between blocking (`=`) and non-blocking assignments (`<=`).

2. (15 points) General coding questions.

   (a) (5 points) Write the full **behavioral** code (module header and description/implementation) for a positive-edge/rising-edge triggered D flip-flop (module name `DFF`) with an input named `clk`, an input named `D`, and an output named `Q`. No other signals are necessary.

   (b) (5 points) What, if anything, is wrong with the code snippet below? Suggest a fix if one is needed, any correct fix is acceptable. Make sure the syntax is correct and all variables are properly declared.

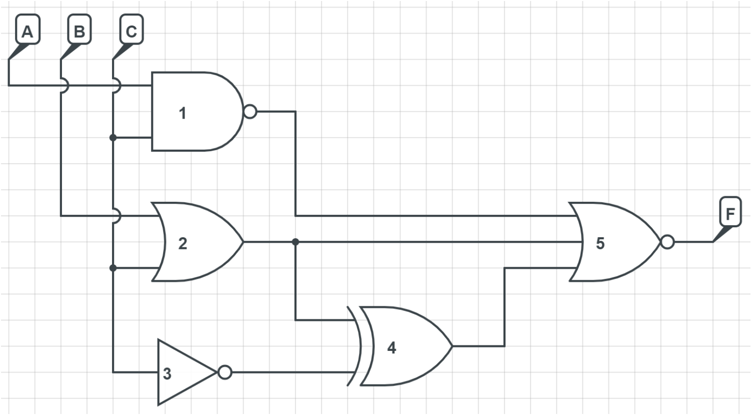
```
reg [1:0] k, m, n;
always@(k, m)
    case (m)
        0: n = |k;
        1: n = &k;
    endcase;
```

   (c) (5 points) What is the hardware generated as a result of running the Verilog code below? Assume all signals that are not explicitly declared have already been declared earlier.

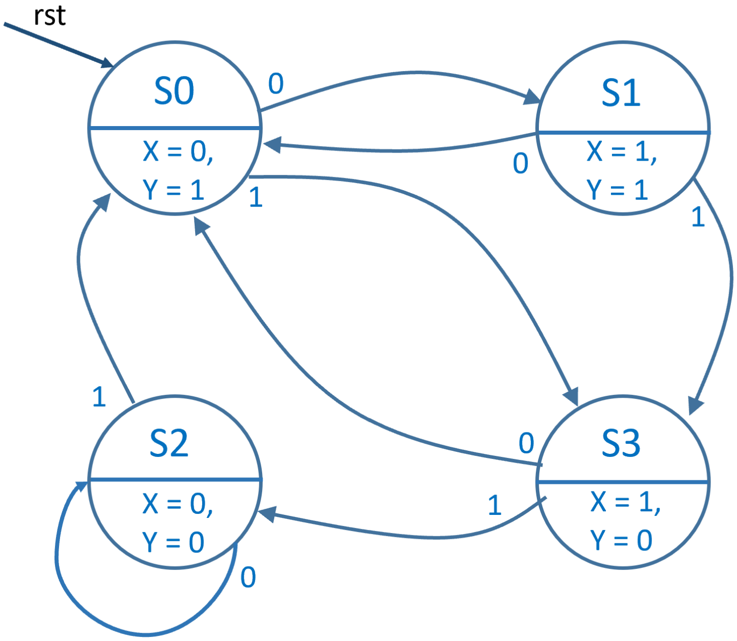
```
reg q, p, r, s, t;
always@(posedge clock)
begin
    r <= q & t;
    s <= r | p;
end
```

3. (15 points) Given the circuit below, answer the following questions:

- List the ports to the module.
- How many nets are there?
- How many symbols (different modules)? What are they?
- How many instances?
- Write a **dataflow** snippet of code for this circuit. Assume all ports have been defined.
- Write a **gate-level** snippet of code for this circuit. Assume all ports have been defined.



4. (40 points) Given the state transition diagram below, fill in the Verilog descriptions as necessary and answer the questions.
Input: `A` (1-bit)
Outputs: `X, Y` (1-bit each)



(a) (2 points) How many bits are needed for state encodings?

(b) (2 points) Is this a Mealy or a Moore FSM?

(c) (8 points) Fill in the next state table and output table for this FSM.

Next State Table

Q_1	Q_0	A	Q_1^+	Q_0^+
0	0	0		
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

Output Table

Q_1	Q_0	X	Y
0	0		
0	1		
1	0		
1	1		

(d) (4 points) Write the module header for this FSM, name it `MidtermFSM`. Assume the clock input is named `clk` and the synchronous reset signal is named `rst`. Use any other signals from previous page as necessary.

(e) (4 points) Write a list of state encodings using the `parameter` keyword using minimum number of bits necessary. Assign encodings to match the state number as a binary number.

(f) (6 points) Declare variables for present state (`presentState`) and next state (`nextState`). Write code to assign the next state to current state on the rising edge of the clock. If the `rst` signal is asserted, current state becomes `S0`.

(g) (10 points) Write code to update the next state based on current state and input. Use a `case` statement.

(h) (4 points) Write **dataflow** code to update the outputs.

5. (12 points) Answer the following general Verilog questions:

   (a) (8 points) What is the result of performing the following operations? Write the result as an **8-bit hex** number.

   - `6'b100110 + 8'b10001001`
   - `8'b11001100 & 8'b10100101`
   - `7'b0100101 ^ 8'b01001011`
   - `~(8'b10000110 | 8'b10101001)`

   (b) (2 points) Briefly explain what exhaustive testing is in a testbench.

   (c) (2 points) Briefly explain what `#10` does in a testbench.

This page is to be used as scratch paper. **DO NOT PUT ANY FINAL ANSWERS HERE.**