University of Washington – Computer Science & Engineering
Winter 2023 Instructor: Chris Thachuk 2023-02-21

CSE 369 QUIZ 2

Name: _Perry_Perfect____________________

Student ID Number: _1234567______

Please do not turn the page until 12:20.

Instructions

- This quiz contains 4 pages, including this cover page. You may use the backs of the pages for scratch work.
- Please clearly indicate (box, circle) your final answer.
- The quiz is closed book and closed notes.
- Please silence and put away all cell phones and other mobile or noise-making devices.
- Remove all hats, headphones, and watches.
- You have 30 (+5) minutes to complete this quiz.

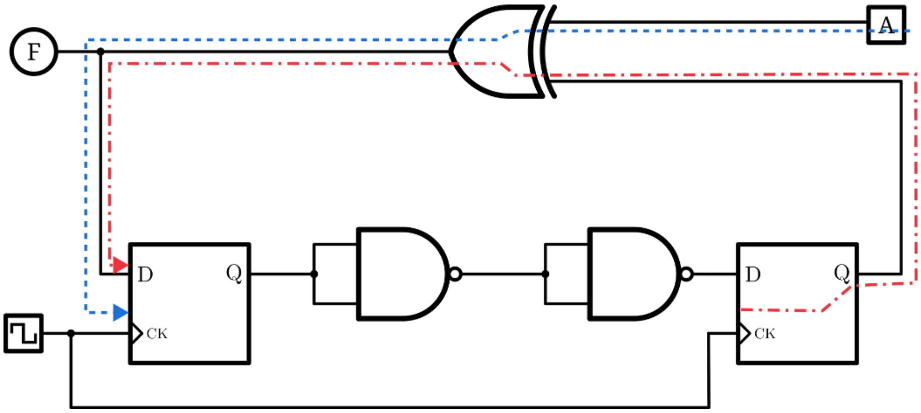
Advice

- Read questions carefully before starting. Read *all* questions first and start where you feel the most confident to maximize the use of your time.
- There may be partial credit for incomplete answers; please show your work.
- Relax. You are here to learn.

Question	Points	Score
(1) SL & Timing	6	6
(2) FSM Implementation	10	10
(3) FSM Design	12	12
Total:	**28**	**28**

Question 1: Sequential Logic & Timing [6 pts]

Consider the following circuit diagram with a clock period of **85 ns** (10^{-9} s), $t_{\text{C2Q}} =$ **25 ns**, $t_{\text{XOR}} =$ **25 ns**, and $t_{\text{NAND}} =$ **10 ns**. Assume that A changes **15 ns** after every clock trigger.



(A) Calculate the **maximum setup time** that will allow the circuit to function correctly. Make sure to *include units.* [3 pts]

$$\boxed{t_{setup} \leq 35\ ns}$$

$t_{C2Q} + t_{XOR} \leq T_{period} - t_{setup}$

$t_{setup} \leq T_{period} - t_{C2Q} - t_{XOR}$

$= 85\ ns - 25\ ns - 25\ ns$

(red path with dash-dotted line)

(B) Calculate the **maximum hold time** that will allow the circuit to function correctly. Make sure to *include units.* [3 pts]

$$\boxed{t_{hold} \leq 40\ ns}$$

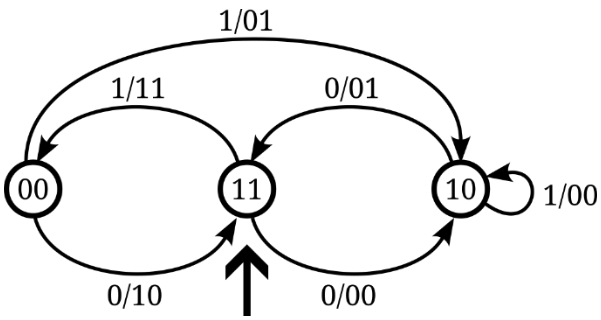
$t_{hold} \leq t_{input} + t_{XOR}$

$= 15\ ns + 25\ ns$

(blue path with dashed line)

Question 2: Finite State Machine Implementation [10 pts]

(A) Fill in the provided truth table based on the FSM shown. [2 pts]



PS_1	PS_0	In	NS_1	NS_0	Out_1	Out_0
0	0	0	1	1	1	0
0	0	1	1	0	0	1
0	1	0	X	X	X	X
0	1	1	X	X	X	X
1	0	0	1	1	0	1
1	0	1	1	0	0	0
1	1	0	1	0	0	0
1	1	1	0	0	1	1

(B) Complete the circuit diagram below using *minimal logic* based on the truth table shown below. **Use only 2-input logic gates.** [8 pts]

PS	In_1	In_0	NS	Out
0	0	0	0	0
0	0	1	1	X
0	1	0	0	0
0	1	1	0	1
1	0	0	1	0
1	0	1	X	1
1	1	0	X	X
1	1	1	1	X

Wire connection: ┼

NS

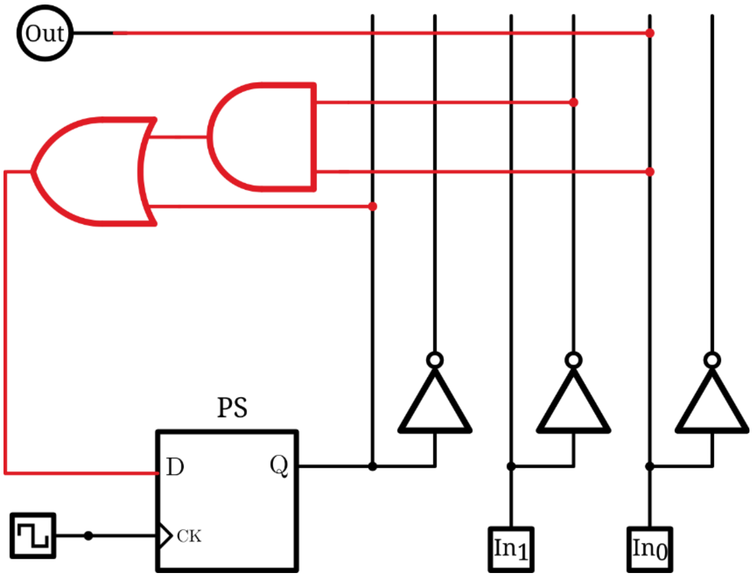
PS \ In_1In_0	00	01	11	10
0	0	1	0	0
1	1	X	1	X

NS = PS + $\overline{In_1}$•In_0

Out

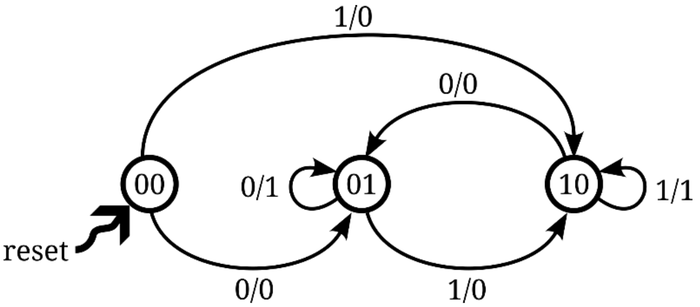
PS \ In_1In_0	00	01	11	10
0	0	X	1	0
1	0	1	X	X

Out = In_0



Question 3: Finite State Machine Design [12 pts]

The following FSM outputs a modified version of its stream of inputs:



(A) Assume we pass the following stream of inputs (left-to-right) immediately after resetting the FSM. What is the corresponding stream of outputs? [4 pt]

```
Input:    1  0  1  1  1  0  0  0
Output:    0  0  0  1  1  0  1  1
```

(B) As *briefly* as you can, describe how this FSM manipulates its input stream. [3 pt]

Outputs 1 when two consecutive inputs agree, and 0 otherwise.

(C) Complete the testbench `initial` block to test the state diagram as *thoroughly as possible*. You need to fill in all bolded blanks. You are welcome to fill out the Verilog comments to help you keep track of state, but these will not be graded. [4 pts]

```
initial begin
                      In <= 0;      // state: 00
     @(posedge clk);  In <= _0__;   // state: _01_
     @(posedge clk);  In <= _0__;   // state: _01_
     @(posedge clk);  In <= _1__;   // state: _10_
     @(posedge clk);  In <= _1__;   // state: _10_
     @(posedge clk);  In <= 0;      // state: _01_
     @(posedge clk);  $stop();
end                 // (input sequence 0 1 1 0 also optimal)
```

(D) What transition of the state diagram were you not able to test in the above testbench? [1 pt]

00 → 10