University of Washington – Computer Science & Engineering

Autumn 2016 Instructor: Justin Hsia 2016-11-22

CSE 369 QUIZ 2

Name: ______________________________

UWNetID: ______________________________

Please do not turn the page until 10:30.

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 20 minutes to complete this quiz.

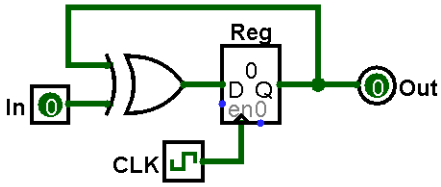
Advice

- 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	8	
(2) FSM Implementation	10	
(3) FSM Design	9	
Total:	**27**	

Question 1: Sequential Logic & Timing [8 pts]

Consider the following circuit diagram with a clock period of **500 ps** (10^{-12} s), setup time of **80 ps**, hold time of **50 ps**, and clock-to-q delay of **100 ps**. Fill in your answers in the boxes below.



(A) If the input `In` changes exactly on clock triggers, what are the limits on the XOR gate delay that ensure proper behavior? Write "**n/a**" if no such limit exists. ***Include units!*** [4 pts]

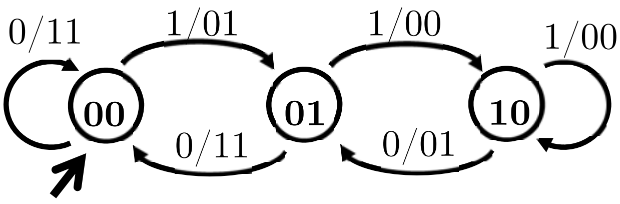
Max t_{XOR} =	Min t_{XOR} =

(B) We choose a gate with t_{XOR} = **100 ps** and complicate the input logic so that the input `In` changes t_{in} after each clock trigger. Within each clock cycle (between 0 and 500 ps) for what ranges of t_{in} will we get proper behavior? Answer using interval notation: $[t_{start}, t_{end}]$. [4 pts]

[0 , ________] ps and [________ , 500] ps

Question 2: Finite State Machine Implementation [10 pts]

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



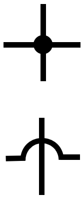
PS_1	PS_0	In	NS_1	NS_0	Out_1	Out_0
0	0	0	0			1
0	0	1	0			1
0	1	0	0			1
0	1	1	1			0
1	0	0	0	1	0	1
1	0	1	1	0	0	0
1	1	0	X	X	X	X
1	1	1	X	X	X	X

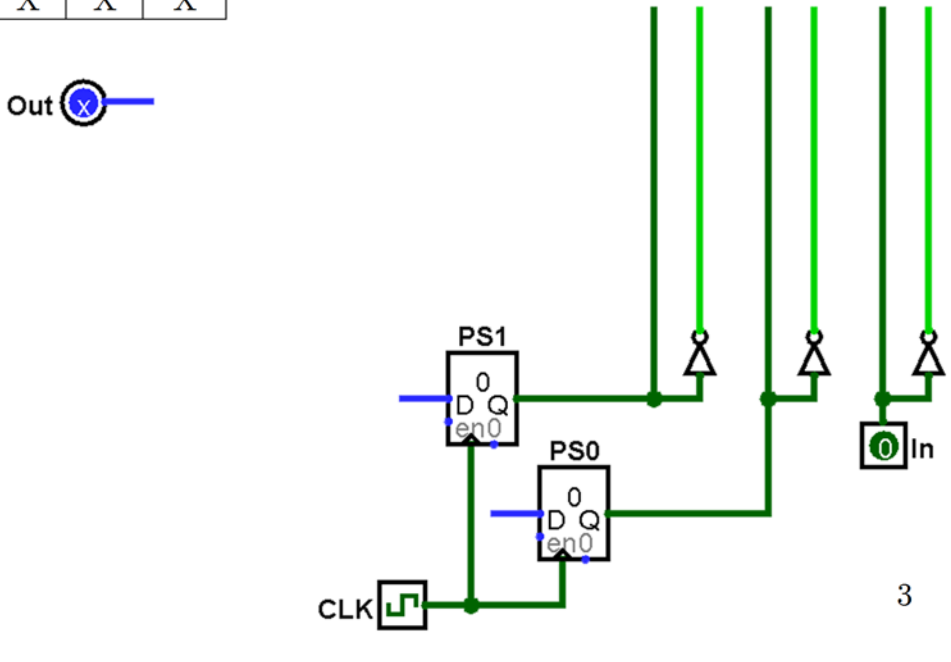
(B) Complete the circuit diagram below using *minimal logic* based on the truth table shown below. You are welcome to use 2- and 3-input logic gates. [8 pts]

PS_1	PS_0	In	NS_1	NS_0	Out
0	0	0	0	1	0
0	0	1	0	0	0
0	1	0	0	0	0
0	1	1	1	0	1
1	0	0	1	0	1
1	0	1	0	0	0
1	1	0	X	X	X
1	1	1	X	X	X

Wire connection:

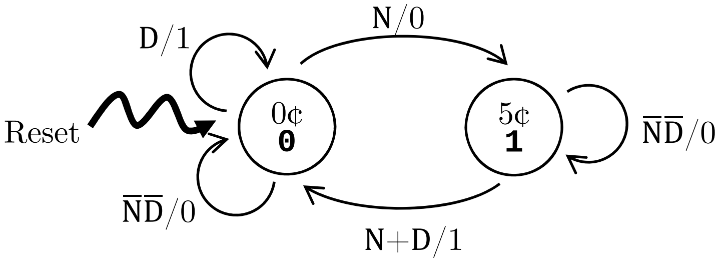
Wire crossing:





Question 3: Finite State Machine Design [9 pts]

Recall the 10¢ gumball-dispensing, no-change-giving vending machine FSM from lecture:



(A) Complete the testbench `initial` block to ***thoroughly*** test the FSM. Even though they may be unnecessary, please fill in all blanks. Don't worry about situations we don't expect to see during normal operation. [4 pts]

```
initial begin
                      N <= 0;     D <= 0;
     @(posedge clk);  N <= 1;     D <= 0;
     @(posedge clk);  N <= 1;     D <= 0;

     @(posedge clk);  N <= ____;  D <= ____;

     @(posedge clk);  N <= ____;  D <= ____;

     @(posedge clk);  N <= ____;  D <= ____;

     @(posedge clk);  N <= ____;  D <= ____;
     @(posedge clk);
     $stop();
end
```

(B) Due to inflation, we decided to make the gumballs 25¢, how many states and state bits would we need? [2 pts]

States:	State bits:

(C) If we kept the cost of gumballs at 10¢ but got greedy and also accepted quarters (25¢), draw the new state diagram below. *Use as few arrows as possible.* [3 pts]