18-240: Structure and Design of Digital Systems

HW4 [9 problems, 64 points]

Covers lectures L08 – L10 *Due: 9 October 2023*

Homework sets are due at 5:00PM on the due date. Upload your answers, to Gradescope by then. No late homework will be accepted. Remember, we let you drop two homework assignments over the semester.

By now, you know how to use **`handin240`** to create the files for the Gradescope upload. Refer to the course wiki or to previous homework if you need refresher instructions.

Discussions about homework in small groups are encouraged — think of this as giving hints, not solutions, to each other. However, homework must be written up individually (no copying is allowed). If you discussed your homework solutions with someone else, either as the giver or receiver of information, your write-up must explicitly identify the individuals and the manner information was shared.

You must show details of your work. There is no credit for just writing down an answer.

Drill Problems [34 points]

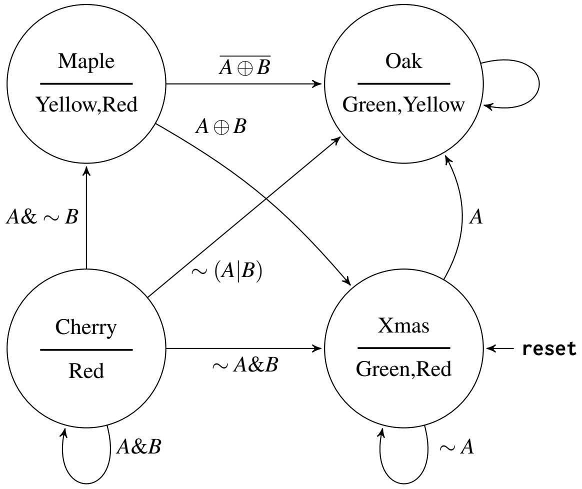
1. [6 points, Lecture 8] Draw the State Transition Diagram for a Moore FSM that detects the sequence `111` on its single input. It signals such a sequence with a `1` on its single output, as soon as it can after seeing the last '1' of the sequence. It should not detect overlapping sequences.

2. [8 points, Lecture 9] Finish the design and implementation of the Moore-style sequence detector from problem 1. Draw the state transition table and write the next-state and output generator equations. Finally, draw the schematic for the circuit.

3. [6 points, Lecture 8] Draw the State Transition Diagram for a Mealy FSM that detects the sequence `111` on its single input. It signals such a sequence with a `1` on its single output, as soon as it can after seeing the last '1' of the sequence. It should not detect overlapping sequences.

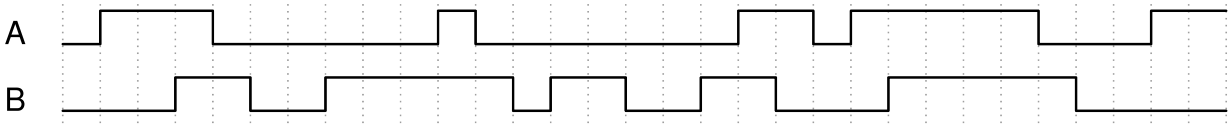
4. [8 points, Lecture 9] Finish the design and implementation of the Mealy-style sequence detector from problem 3. Point out the qualitative differences in the state transition table, equations and schematics (i.e. I'm not looking for detailed "this is a one, where that is a zero" type differences. Instead, show me the big differences.)

5. [6 points, Lecture 8] Provide the State Transition Table for this State Transition Diagram.



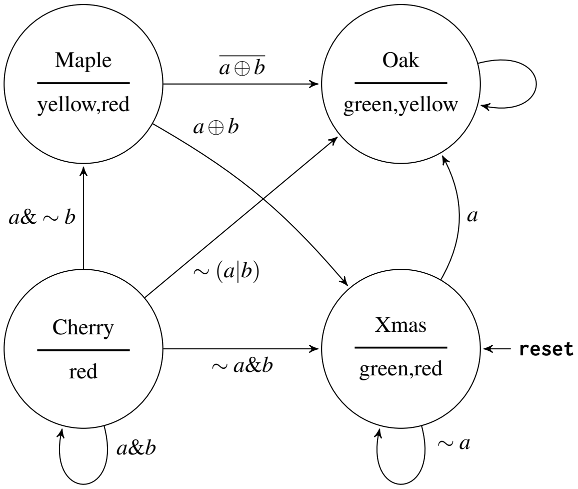
Non-Drill Problems [30 points]

6. [6 points, Lecture 8] Given the input waveform below, sketch the output (**Q**) of a D Flip Flop with **A** connected to the clock input and **B** connected to the **D** input. Then, reverse the connections (**B** ⇒ **clock**, **A** ⇒ **D**) and sketch again.



7. [6 points, Lecture 9] Write the SystemVerilog implementation of this STD. Your implementation should follow the style of Lecture 9, slide 21. Use the following header:

```
module hw4prob7
  (input  logic clock, reset, a, b,
   output logic green, yellow, red);
```



▷ **Submit your work as a file named `hw4prob7.sv`.**

8. [10 points, Lecture 9] You are to develop a state diagram for a washing machine. You have four inputs: **`coin`**, **`double`**, **`time`**, **`lid`**. After **`reset`**, the machine waits until a coin is deposited (i.e. **`coin`** = 1). It then sequences through the following stages: soak, wash, rinse and spin. If **`double`** = 1 at the end of the first rinse phase, that means a "double wash" has been requested and the sequence should be: soak, wash, rinse, wash, rinse and spin.

   The clock running your FSM is very fast, so a timer will set **`time`**=1 for one clock period, indicating your FSM should transition to the next stage. If the lid is raised (i.e. **`lid`** is asserted) during the spin cycle, the machine stops spinning until the lid is closed. The timer will keep ticking while the lid is raised.

   Draw a state diagram that implements this finite state machine. Your state will be the only necessary output.

9. [8 points, Lecture 9] Draw the State Transition Diagram for a Mealy machine with two inputs **`(A, B)`** and two outputs **`(F, G)`**. **`F`** is active anytime **`A`** is a one and **`B`** hasn't been a one for two clocks (i.e., the two most recent clock edges). **`G`** is active any time **`A`** is active and **`B`** was active at the last clock.