CS 2110 Timed Lab 2: Finite State Machines

Corey, Nick, Saahir, Brian, Pranav

Fall 2021

Contents

Please take the time to read the entire document before starting the assignment. It is your responsibility to follow the instructions and rules.

1 Timed Lab Rules - Please Read

You are allowed to submit this timed lab starting from the moment your assignment is released until your individual period is over. You have 75 minutes to complete the lab, unless you have accommodations that have already been discussed with your professor. Gradescope submissions will remain open for several days, but you are not allowed to submit after the lab period is over. **You are responsible for watching your own time. Submitting or resubmitting after your due date may constitute an honor code violation.**

If you have questions during the timed lab, you may ask the TAs for clarification, though you are ultimately responsible for what you submit. The information provided in this Timed Lab document takes precedence. If you notice any conflicting information, please indicate it to your TAs.

The timed lab is open-resource. You may reference your previous homeworks, class notes, etc., but your work must be your own. Contact in any form with any other person besides a TA is absolutely forbidden. **No collaboration is allowed for timed labs.**

2 Overview

In this timed lab, you will implement a **One-Hot State Machine** in CircuitSim. This Finite State Machine will take in one 1-bit input (D), and it will output three 1-bit outputs (C, E, L). The state machine is a Moore State Machine, where your output is based solely on the current state. Diagrams and detailed instructions are provided below.

Dr. TwenTee OneTun is a roboticist and an avid nature photographer. Combining his passions, he decided to make a tiny robotic bunny with a camera to get beautiful pictures of animals in the local forest and he needs your help with the creation of his robot, the Little Cottontail MK.4. He has given you these specifications:

- The robotic bunny will start out hiding and turned off (State: 0000).
- The robotic bunny turns on but stays hiding in standby mode (State: 0001) until it senses danger (D) outside, using its robo-ears (E) to listen for threats. If the robotic bunny does not sense danger (D), it will enter foraging mode (State: 0100), activating the camera (C) to take photographs of nearby animals.
- While in foraging mode (State: 0100), the bunny will remain in foraging mode as long as it does not sense danger (D).
- If the robotic bunny is in foraging (State: 0100) or standby (State: 0001) and it *does* sense danger (D), then it will enter alert mode (State: 1000), turning on both its camera (C) and robo-ears (E) to evaluate the perceived threat. If the danger (D) disappears, the bunny will return to standyby mode (State: 0001)
- Should the danger (D) persist, the robotic bunny will enter defense mode (State: 0010), leaving its camera (C) on and turning on its laser defense system (L). The bunny will remain in this state until the threat dissipates, at which point it will move back to standby (State: 0001).

3 Instructions

3.1 State Transition Diagram

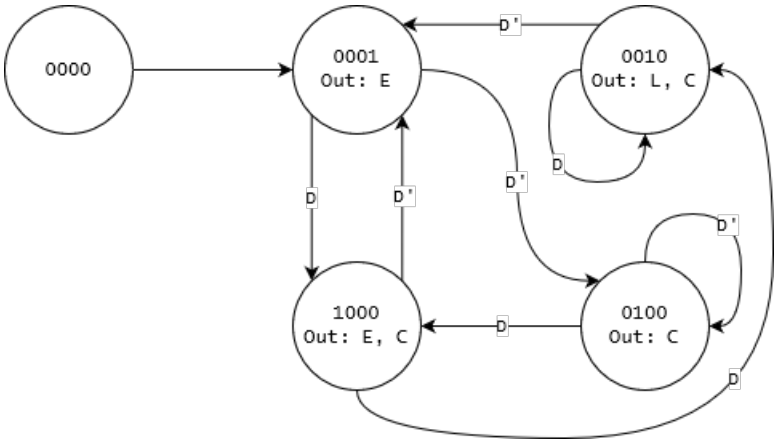

Figure 1: Transition diagram.

3.2 Building the Circuit

Use CircuitSim to build your circuit in the `tl2.sim` file provided in the assignment files. Complete the circuit so that the logic matches the diagram above.

3.3 Restrictions

The input / output pins we have given you in the skeleton file must not be renamed. Do not add any additional input / output pins other than the ones we have given you. Do not rename the sub-circuit we have given you. If you have issues, check out the "Common Errors" section below.
If you have any questions on what you may not use then assume you can't use it and ask a TA.

You are only allowed to use the following components in CircuitSim:

- Basic logic gates (NAND, NOR, AND, OR, NOT)
- Registers (for this assignment, **exactly ONE**).
- Wires, splitters/joiners, tunnels, constants, plexers

4 Hints

- Connect your clock and reset first.
- Be careful while using the splitter component to separate your register output or combine register inputs. Remember to first set the bitsize, then the number of fan-outs and finally assign each bit to each fan-out (Bit 0 refers to the least significant bit).
- Remember that due to this FSM being a One-Hot FSM, you do not need to create minimized/reduced boolean expressions and can implement your circuit with simple sum-of-products expressions.
- Don't forget to provide **ALL** necessary inputs to your register as well as specifying its bitsize so that it functions appropriately.

5 Common Errors

Use the autograder's output to determine where you have gone wrong. The names of the tests you fail should usually (but not always) point you in the right direction.

Some common errors and their remedies:

1. **Be careful that the bits on your splitters are ordered correctly**. A common error is that the register inputs are combined in the opposite order, causing your state transitions to be wrong.
2. Make sure you haven't added extra any input/output pins to your circuit. It is common to confuse constants with input pins, and probes with output pins.
3. Make sure you have not renamed input/output pins, or else the autograder won't be able to find them.
4. Make sure you haven't changed the name of your sub-circuit.
5. A common cause of short-circuits is two pins (on an AND gate, splitter, etc) being unintentionally connected. These are often hard to spot, since the pins are so close to each other, but if you zoom into your circuit you may find such an error and fix it.
6. Make sure the names on your tunnels match. Tunnel labels are case-sensitive, and they do not trim whitespace. If two tunnels look the same but for some reason they aren't connecting, there may be a "space" hidden in the tunnel's label.

6 Autograder/Grading

To run the autograder locally, navigate to the directory containing your timed lab and the tester and run the following command:

```
java -jar tl02-tester.jar
```

Make sure to run the local autograder out of your docker container.

The output of the autograder is an approximation of your score on this timed lab, so that you can evaluate how much of the assignment expectations your submission fulfills. We reserve the right to change the autograder test cases before finalizing grades. Timed labs are not manually graded or reviewed by the TAs, and there will not be opportunities for partial credit beyond the autograder. Submissions that are not runnable will receive a 0.

7 Deliverables

Please upload the following files to **Gradescope**:

1. `tl2.sim`

Note: if you were granted an extension, you will still turn in the assignment over Gradescope.

Download and test your submission to make sure you submitted the right files