SER232 - Assignment 6

[10 Points]

Description

In this assignment you will translate a system description into a FSM state diagram and then transform it into a FSM controller circuit in Logisim by utilizing a truth table and Boolean equations.

It is heavily recommended to watch the supplemental video and work on the optional assignment before you attempt this assignment. The video shows step-by-step how this process works using a complex example. Watching and understanding this video, will help you greatly with this assignment.

System Description & State Diagram

First task: Transform the following system description into a FSM state diagram (you can use any tool to create this diagram, which allows you to draw circles and connect arrows between them; hand-drawn diagrams will not be accepted). Make sure you are using the correct notation for the state diagram. Use the given labels for everything, don't pick your own labels. Everything is defined and you don't have to choose your own labels at any point during this assignment.

Use minterms for every transition event in the state diagram (each event must be a specific configuration of all inputs) to avoid incomplete and non-exclusive transitions. If the state does not change for certain events, make sure to model this behavior with an appropriate transition.

This system represents a simplified version of a battery charger, which has three functions: charging a battery, maintaining its charge and a diagnostic mode to check if the charger hardware still works. The charger has two inputs which are used to control everything: A battery input will indicate if a battery was inserted into the charger, and a charge request input indicates if the user wants to charge the battery. The charge request input can also be used to enter the diagnostic mode, while no battery is inserted. The controller developed in this assignment will control an external circuit to apply the correct voltage to the battery (high voltage to charge, low voltage to maintain charge) and enable the diagnostic mode.

Below is the detailed system description, which will be used to create the state diagram of this system.

- The system has two 1-bit inputs:
  1. *Battery Inserted* (label: b)
  2. *Charging Requested* (label: c)
- The system has three 1-bit outputs:

1. *Diagnostic* (label: D), which will enable the diagnostic mode with a 1, which is then performed by an external circuit.
2. *Low Voltage* (label: L), which will indicate with a 1 to an external circuit that a low voltage should be applied to the battery in order to maintain its current charge level.
3. *High Voltage* (label: H), which will indicate with a 1 to an external circuit that a high voltage should be applied to the battery in order to charge it.

- The system has four states: *Idle* in which the charger is not active and waiting for the inputs to change. Here, the diagnostic mode is disabled, and neither low nor high voltage is applied. *Maintain* is used to apply a low voltage to the battery, while the diagnostic mode is disabled and high voltage is not applied. *Charge* will apply a high voltage, while the diagnostic mode is disable and low voltage is not applied. *Diagnostic* will test the charger hardware by enabling the diagnostic mode, and in order to test the hardware associated with generating the high and low voltage, the high and low voltage outputs are enabled in addition.

- The binary encoding of the states is as follows:
  - Idle: 00
  - Maintain: 01
  - Charge: 10
  - Diagnostic: 11

- If the system is in the *Idle* state, it will transition into *Maintain*, if a battery is inserted into the charger, independent of the charge request input. The system will stay in *Idle* if no battery is inserted and no charging is requested, and will only transition into *Diagnostic* if a charge request was made by the user without a battery inserted.

- If the system is in the *Maintain* state, it will stay in this state as long as a battery is inserted and no charge request has been given. Only if charging the battery has been requested while a battery is inserted, it will transition to *Charge*. As soon as the user removes the battery, independent of the charge request, the system will transition back to *Idle*.

- If the system is in the *Charge* state, it will keep charging as long as the battery is inserted and the charge request active. If the charge request is not active anymore and the battery is still inserted, the system will go back to the *Maintain* state to maintain the current battery charge. If the user removes the battery while charging the system will go back to *Idle*, independent of the charge request input.

- If the system is in the *Diagnostic* state, it will stay in diagnostic mode as long as the charge request without an inserted battery is active. In any other case, it will return back to *Idle*.

Converting to Circuit

Perform the following tasks after you transformed the previous system description into a state diagram:

1. Create the truth table to describe the complete behavior of the state diagram (all possible input combinations for all existing states must be covered). Use the given labels. Order columns for multi-bit inputs/outputs from MSB to LSB. Indicate which are inputs and which are outputs. Do not simplify the truth table with *don't care*-symbols.

   For the current state use label *c* (add index if multi-bit: *c0* for LSB, *c1*, etc.) and for the next state use label *n* (add index if multi-bit: *n0* for LSB, *n1*, etc.).

2. Derive the Boolean equations from the truth table for all outputs (next state and all system outputs). Use the same labels as in the truth table. Don't simplify the Boolean equations.

3. Create a working FSM controller circuit in Logisim. You can use the built-in register component of Logisim for your state register. Rename the "main" circuit to "BatteryCharger".

   Notes:

   - Make sure to test it properly and thoroughly to confirm it is working as described in the System Description above.
   - To test the FSM, it is recommended to manually toggle the clock. This way you are able to see each state change clearly.
   - In this system, not all system outputs are equal to an existing signal in the system. There are two possible approaches to generate system outputs: Create a combinational circuit based on the derived Boolean equations for the respective system outputs, or use a multiplexer to map the current state to the correct system outputs.
   - For the next state bits, you must include all system inputs including the current state. For the system outputs, since they only depend on the current state, you can derive these equations by just using the current state bits and can omit the system inputs.

Deliverables

Important:

- Hand-drawn state diagrams and hand-written truth tables will not be accepted. Use proper software to create them.
- Format your table properly (e.g. using the same content alignment for all cells, use of cell borders, make sure the table is not split and on a single page, etc.). Points will be deducted if the table is not properly formatted.
- Make sure to submit both files in one submission. Only the files of the last submission will be graded. If one is missing, you will receive 0 points for the missing part.

The deliverables must be submitted on Canvas before the due date as a single submission:

1. Logisim *.circ* file of the FSM controller circuit. Submit the file with the following naming convention: *lastname_a6.circ*
2. Design document as *.pdf* that includes the state diagram, truth table and Boolean equations of the FSM. Submit a single file with the following naming convention: *lastname_a6design.pdf*