ECPE 174 – Lab 2
One-Hot Design and Mealy Finite State Machines

I. Objectives

Design and implement a One-Hot Moore finite state machine and a Mealy-based finite state machine that mimics the drive circuitry for the taillights of an old car. Analyze the use of Moore, One-Hot Moore, and Mealy FSMs.

II. Problem Statement

Three input signals control the operation of the taillights. The input signals are LEFT, RIGHT, and HAZARD. The output signals LA, LB, and LC drive the left taillights, and the signals RA, RB, and RC drive the right taillights. All lights should be off (idle state) when none of the inputs is HIGH. When HAZARD=1, all six lights should flash on and off in unison. When LEFT=1, the left side taillights should flash according to the repeating pattern LB on, LA and LC on, all lights on, all lights off, and then back to LB on. When RIGHT=1, the right side tail-lights should flash according to the repeating pattern RB on, RA and RC on, all lights on, all lights off, and then back to RB on.

Both LEFT and RIGHT cannot be 1 at the same time. If a switch from LEFT to RIGHT (or vice-versa) occurs in the middle of a flashing sequence, the network should go to the idle state (all lights off) and then begin the new sequence. HAZARD takes precedence over LEFT or RIGHT if either of them is asserted at the same time as HAZARD.

The state machine should be driven by a clock signal that runs at a frequency equal to the desired flashing rate (which will be given in class).

Just as in Lab 1, create the controller for this lock. This time use a One-Hot Moore FSM and a Mealy FSM. Connect the input keys to SW0, SW1, and SW2. Output the LEFT lights using LEDR0-LEDR2; output the RIGHT lights using LEDG0-LEDG2.

III. Pre-Lab

Complete the following steps before lab and turn in by 2pm:

1. Modify last week's lab to use a One-Hot design. Generate a state table and state machine diagram.
2. Generate a state assignment table.
3. Implement the design using behavioral VHDL (see 8.29 on pg 509 as a guideline). You need to explicitly enumerate the states using the one-hot encoding instead of variable names or creating constants defining the variable names.
4. Simulate your code using the waveform editor and functional simulation. Turn in image of simulation waveform. Please use buses to group related signals instead of individual wires.

5. Repeat steps 1-4 using a Mealy FSM. You can use the usual state enumeration for the VHDL.

Submit electronically via Sakai if possible. Otherwise submit a clean, handwritten copy by 2pm in class.

IV. In-Lab

Complete the following steps in lab:

1. Starting with the One-Hot design:
   a. Ensure Quartus State Machine Processing is set to "Auto."
   b. Implement the VHDL design on the Cyclone II.
   c. Demonstrate it for check-off of the lab.
2. Repeat for Mealy design. Ensure the Quartus State Machine Processing is set to "Minimal Bits."

V. Post-Lab

Discuss the following your lab report:

- How does the state machine diagram that Quartus generates compare to your state machine diagram for the One-Hot design? For the Mealy?
- How do all three designs differ? How does Quartus implement each one? What hardware is used and what clocking rate did each achieve?
- Which design makes the most sense for solving this problem? Why?