ECPE 174 – Lab 4
Asynchronous Design cont.

I. Objectives

Design and implement an asynchronous finite state machine that controls a vending machine. Explore synchronization issues.

II. Problem Statement

A vending machine accepts nickels and dimes. It dispenses merchandise when 20 cents is deposited; it does not give change if 25 cents is deposited.

- Coins are deposited one at a time.
- Input signal D = 1 when a dime is deposited; input signal N = 1 when a nickel is deposited.
- When 20 cents or greater has been deposited, *candy* = 1, the candy is dispensed, and the machine returns to the initial state.

Create the controller for this lock using an asynchronous FSM. Connect the input keys to D=KEY1, and N=KEY0. Output *candy* on LEDG0.

III. Pre-Lab

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

1. Generate a state machine diagram. It is your choice whether to use Mealy or Moore. Clearly indicate which you are designing and why.
2. Implement the design using behavioral VHDL. Synchronize inputs to a clock that will ensure inputs last longer than the clock period. Ensure the design is hierarchical with separate modules for the synchronizers and state machine.
3. Simulate your code using the waveform editor and functional simulation. Ensure simulation covers 75% or greater of the test cases and use simulation to verify VHDL matches your state machine.

IV. In-Lab

Complete the following steps in lab:

1. Implement the VHDL design on the Cyclone II.
2. Demonstrate it for check-off of the lab.
3. Modify the clock cycle in the functional simulation. What happens when the input changes exactly on the rising edge of the clock? When the input pulse is shorter than the clock cycle?
4. Re-run with the timing simulation.
5. Demonstrate for check-off.

V. Post-Lab

Discuss the following your lab report:

- What happened with the functional and timing simulations for the range of input conditions? How did this compare to your expectations?
- What is a reasonable pulse length for a real world application? What is the clock cycle to ensure input pulses are longer than one clock cycle? Is this reasonable? Why or why not?
- How did Quartus implement the system? Does this match your expectations? Why or why not? What, if anything, is unique or interesting about the design? Hint: Check out the RTL viewer under Tools -> Netlist Viewers to see exactly what hardware was developed.