Lab 7: Example State Machine: Traffic-Light Controller

OBJECTIVES

The objective of this lab is to design and implement a state machine using a CPLD. You will use an ASM to design a traffic light controller and implement this state machine on your proto-board using your UF-3701 CPLD board, switch circuits and LED circuits.

MATERIALS

- Your entire lab kit (including your DAD)
- Debounced switch circuit (from previous labs), switch circuits, LED circuits, and UF-3701 CPLD board.
- Read the file *Creating Graphical Components* from the Software/Docs page of the website, under the Quartus heading. Download this document onto your computer.

SPECIFICATIONS

Design a state machine to control the traffic lights at an intersection in Gainesville's midtown, the intersection of West University Avenue and NW 17th Street. To reduce the number of LEDs needed in lab, you only need to control the light outputs for West University Avenue, not NW 17th Street (Buckman Drive). The traffic light controller has three **active-high** outputs (`Red`, `Yellow`, `Green`), and two inputs: **c**ar **w**aiting at 17th Street [`CW(H)`] and **e**mergency **v**ehicle approaching [(`EV(L)`].

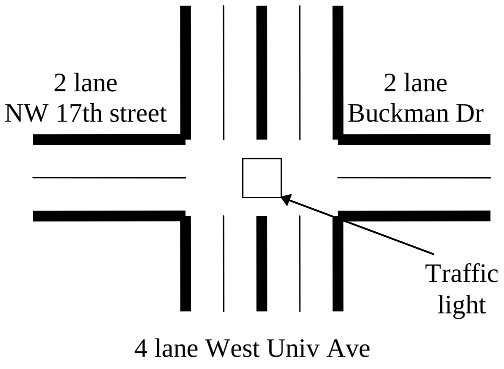

Figure 1. Traffic Light

TIMING:

1. The University Avenue green light normally stays on for five cycles. However, if an emergency vehicle comes up to the light (while traveling down the 17th Street), `EV` will go true and this can occur during any of the five green cycles. If this occurs, the light should **immediately** turn yellow and stay on for at least one clock cycle (but no more than two) and then proceed on to step #4. Otherwise, if `EV` is not true after the five cycles, move to step #2.
2. After the University Avenue green light has been on for five cycles as described in step 1, check if there is a car waiting (`CW` is true). If so, then continue to step #3; else go back up to step #1.
3. `Green` turns off and `Yellow` turns `on`. This lasts for two clock cycles.
4. `Yellow` turns off and `Red` turns on for three clock cycles.
5. If EV is not true, go back to step #1. Otherwise, as long as EV remains true, keep the light Red.

A few notes about our traffic lights and traffic light controller are listed below.

1. There should **never** be more than one traffic light on at a time.
2. There should **always** be a single traffic light on.
3. You can assume that an emergency vehicle will take at least one clock cycle to approach an intersection.

PRE-LAB REQUIREMENTS

1. Draw an ASM chart for your controller. Note: You should **never** have a situation when two lights are on at the same time, i.e., Green and Yellow should never be on at the same time.
    - The simplest way to create a multiple clock delay (e.g., step 1 in the Timing section) is to have successive states. I suggest that you do this rather than designing a counter for this lab. (In EEL 4712: *Digital Design*, you will design counters, but use VHDL to make this a relatively simple procedure.)
2. Create a next-state truth table showing the current state, input, next state, flip flop inputs and system outputs (i.e., red, green or yellow) for each state in your ASM diagram.
3. During previous labs during the semester, you designed your circuits using logic gates (by drawing the circuit elements in Quartus). This is called *schematic entry*. You **could** create a Quartus schematic entry design using D-Flip Flops and SSI logic gates using (as you have all semester, and recently, for implementation in the CPLD); but you do **NOT** need to do this for this lab. (Do this only if you want to; there will be

no credit for doing this part.) In part 5 you will use VHDL to replace the combinatorial **part** (not the flip-flops) of your schematic entry design.

Note: Simplification of the equations is helpful, but not required in this lab. Quartus will simplify for you! After compiling your design, make a Quartus equation file by selecting Processing | Start | Start Equation Writer (Post-Synthesis). The generated file will have an ".eqn" extension. Open this file to observe the equations. Some of the operators in Quartus equation files are as follows: & = AND, ! = NOT, # = OR, and $ = XOR (exclusive OR).

4. If you want to, simulate this design in Quartus. There will be **no** credit for doing this part.

5. Repeat the design, again using D-Flip Flops, but this time replace the combinatorial part of your design (that used logic gates) with a VHDL program. The design should be named `Lab7_Traf_Cont`. Output the state bits (the outputs of the Flip Flops) so that you always know the present state of your system. These state bit outputs greatly simplify the debugging process. (See the *Creating graphical components* file on the website for information on how to incorporate VHDL code in a bdf file.)

6. Simulate this design in Quartus. As always, annotate your design simulation.

7. Repeat the design for the ASM in part 1, using a T flip-flop for the most significant bit, a JK-flop for the least significant state bit, and a D flip-flop(s) for the remaining state bits. More columns should be added to the next-state truth table to deal with the new choice of flip-flops. You can use the techniques of part 3 (schematic entry) or part 5 (VHDL) to complete this design.

8. Simulate this design in Quartus. As always, annotate your design simulation.

9. Program your CPLD and design your first VHDL plus schematic entry design (from Pre-Lab part 5) using the UF-3701 CPLD board. Use your DAD for the non-clock inputs and outputs, but also use LED circuits for the outputs. Use the debounced switch for your clock input (as discussed used in previous labs).

   Note: You could use your DAD for your debounced clock input too, but I want you to get some more experience building a debounced switch circuit, so do **NOT** do this for this lab.

   Warning: Always Tri-state all unused CPLD pins, as described in the Quartus Tutorial.

10. Reprogram your CPLD to the second VHDL plus schematic entry design (from Pre-Lab part 7) (to the CPLD). **Use the same CPLD inputs and CPLD outputs pins as the first design.**

11. You must adhere to the Lab Rules and Policies document for every lab. Re-read, if necessary. Documents must be submitted by email and on paper for every lab. All pre-lab material is to be emailed **BEFORE** the beginning of your lab. As always (with a few new things starting with this lab), put your design (ASM, next-state truth table, derived equations, and VHDL files) annotated simulation, and other requested items in your lab document and email this file along with your design archives, and summary document. Also as usual, bring a printout of your *Summary* document to lab.

Note: The simplest way to create a multiple clock delay is to have successive states, n-states for an n-clock delay. I suggest that you do this rather than designing a counter for this lab.

IN-LAB

1. Demonstrate your first VHDL plus schematic entry design (from Pre-Lab part 9) for your TA.

2. After successfully demonstrating your first design, download the second VHDL plus schematic entry design (from Pre-Lab part 10) (to the CPLD). **Use the same inputs and outputs as the first design.** Demonstrate your second design to your TA.