**ECE324 Lab 7: Traffic Light**

Name(s):

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

|  |  |  |
| --- | --- | --- |
| **Exercise** | **Course outcome** | **Grade** |
| Lab7 Demo | 2.a, 2.d, 5.c, 7.b | /15 |
| Lab7 Report | 2.a, 5.c, 7.b | /25 |
|  | **TOTAL:** | /40 |

2.a. Define engineering problems from specified needs for digital systems including implementation on FPGAs using HDL programming.

2.d. Produce FPGA designs that meet specified needs.

5.c. Collaborate with individuals with diverse backgrounds, skills and perspectives.

7.b. Employ appropriate learning strategies such as communicating with an expert, using external resources, experimentation, simulation, etc.

# Learning objectives:

1. Design a finite-state machine (FSM) sequential logic circuit using SystemVerilog HDL.
2. Learn how to implement a finite-state machine sequential logic circuit using Xilinx Vivado software, and demonstrate it on an FPGA prototype board.
3. Verify that the completed FSM implementation meets timing constraints.

# Procedure:

Use SystemVerilog HDL, Xilinx Vivado, and your FPGA prototype board to design, implement, and verify the traffic light enhancements specified below, using the following provided files. Files free\_run\_shift\_reg.sv disp\_mux.sv, hex\_to\_sseg.sv, and risingEdgeDetector.sv are identical to what have been used in previous labs. File db\_fsm.sv contains Pong Chu’s debouncing circuit, and pwm\_basic.sv contains Pong Chu’s basic pulse width modifier circuit. The only files you need to change are Lab7\_TrafficLight.sv and Lab7\_TrafficLight.xdc.

Part A: You are given a working design of a simple traffic light controller for an intersection in the United States[[1]](#footnote-1) of road A (east-west road) and road B (north-south road), with no turn lanes nor signals. The lights on opposite sides of each road have the identical values for their green, yellow, and red lights. The Nexys4DDR emulates the traffic lights using the RGB LEDs to the left of the buttons, the left one (LD17) for road A, and the right one (LD16) for road B. Use the 16 switches to set the duty cycle of the PWM controlling the brightness of the RGB LEDs to 6.25% = 1/16 = 2-4 (the radix point of the switches is to the left of SW15). Also, because some people are color-blind[[2]](#footnote-2), the position of the red (top), yellow (middle), and green (bottom) lights are shown on the rightmost two 7-segment displays. In addition, for your convenience in verification, another 7-segment display shows the number of seconds in which each state has been. Synthesize this design and understand its operation. You do not need to demonstrate this.

Part B: To better regulate traffic flow, four cameras have now been installed at the intersection, with Nexys4DDR’s BTNL and BTNR emulating the detection of a vehicle on the west and east of road A, and BTNU and BTND for the north and south of road B. Use free\_run\_shift\_reg’s to perform synchronization and handle metastability (suggestion: first “or” BTNL with BTNR, and BTNU with BTND, so you will only need two sensor signals), but since the buttons are to be sampled in the state machine only every second, you don’t need to debounce them. Each green state is always on for a minimum of 6 seconds and a maximum of 9 seconds, regardless of vehicle motion. If the greenA state has been active for at least 6 seconds and a vehicle is being detected on road B but not road A, then switch to the yellowA state, thus giving priority to the busier road A. If the greenB state has been active for at least 6 seconds and a vehicle is being detected on road A or not on road B, then switch to the yellow state. The duration of the yellow and red states are not affected by the cameras. To save time, you are welcome to implement part C before synthesizing and demonstrating your design.

Part C: When traffic lights are first turned on (such as after a power outage), or when construction is nearby, often a mode is implemented which flashes between the red lights being on and the red lights being off, which in the United States is to be treated the same as a four way stop sign. Add to the state machine a flashRedOn state (which should become the power-up state) and a flashRedOff state, with a normal duration of one second for each. Pushing the center button for any duration should immediately toggle from either of the two flash states to the yellowB state, or from any of the six non-flash states to the flashRedOn state. In addition to using a free\_run\_shift\_reg for synchronization and metastability, you’ll also need to use the debounce and edge detection modules, since you will only want to toggle once per push of the center button. Synthesize this design, verify its operation, and demonstrate it to the professor or lab assistant, including how you can increase the RGB LED brightness by increasing to a 25% duty cycle.

# Deliverables:

1. Demonstrate operation of your completed FPGA design to the instructor or lab assistant.
2. Write a brief (1-2 pages) lab report including the following items:
   1. Cover sheet with names, course number, assignment number, grade box and ABET outcomes.
   2. A written description of your SystemVerilog code: describe the overall function of your design and the operation of each module, including any salient details.
   3. Report the number of Flip-flops and LUTs used. Also report the utilization percentage for flip-flops and LUTs.
   4. Report the design’s minimum clock period, and calculate and report the maximum possible clock frequency.
3. Document any SystemVerilog code you modified. Your SystemVerilog code ***must include header comments stating your names, date and class number.*** Any changes to the SystemVerilog code ***must also include comments explaining the operation of the code***.
4. Upload your written lab report ***in .pdf format***, upload all of the .sv and .xdf text files used for your solution (zipped together if you like), and then hit the submit button just once.

1. In other areas of the world, the sequence is often different, such as the yellow and green lights being on at the same time, or flashing lights before they change. [↑](#footnote-ref-1)
2. Approximately 1 out of every 200 female people have some form of color-blindness, whereas approximately 1 out of every 12 male people do. So it’s unlikely that any female students in this class are color-blind, but it’s likely that at least one male student in this class has some form of color-blindness. [↑](#footnote-ref-2)