# 4-Way Traffic Light Controller

Rafil Yousif, Yuan Wei, Joe Kocenda
Electrical and Computer Engineering Department
School of Engineering and Computer Science
Oakland University, Rochester, MI
e-mails: rafilyousif@oakland.edu, yuanwei@oakland.edu, jjkocenda@oakland.edu

***Abstract*— This written report demonstrates the design and simulation of a simple 4-way traffic light controller, with 2 different modes for regular and light traffic. During regular traffic, the traffic light operates normally cycling through all signals. In light traffic, flashing yellow is displayed for the high-traffic road, and flashing red for the low-traffic road. For this project, counters, multiplexers, and finite state machines offered the best method of implementation. The goal of this project was to be able to comprehend or understand one possible method of replicating a 4-way traffic light, using the knowledge gained from this course, and other courses as well.**

## I. Introduction

Traffic lights play an important role in today's world as society depends on traffic lights to function properly and ensure the general safety of daily commuters or travelers whether it's by automobile, bike, walking or any other type of transportation. In general, traffic signals are meant to maintain an orderly flow of traffic in multiple directions.

The purpose of this project is to design a functioning 4-way traffic light controller with no left or right-hand turns utilizing VHDL digital logic tools, such as finite-state machines (FSM), counters, multiplexers, and synchronous circuits. The knowledge regarding counters, multiplexers and finite-state machines acquired from this course, along with independent research was essential in designing this circuit. This traffic light system demonstrated one possible method of how the device may function in real-life through implementation on the field-programmable gate array (FPGA) using Vivado.

## II. Methodology

### *A. Clock and Counter Utilization*

To have all the components function in unison, manipulation of the 100 MHz master clock was necessary. Manipulation of the master clock was accomplished by implementing 4 total counters, each counting to a different number of seconds. To have a properly functioning traffic light, each light must remain illuminated for a specific number of seconds to correctly control the flow of traffic, explaining the importance of 4 counters to differentiate between states. These counters will operate successively, meaning once the first counter has reached its limit, the next counter will begin counting and so forth. For example, states 0 and 3 operate with a 15 second counter, and the succeeding states operate with a 3 second counter once 15 seconds have passed.

### *B. Regular Traffic Finite State Machine*

Finite state machines were the main driving force of this entire system, thus it is extremely important to understand how they are being utilized in this digital design system. The regular traffic FSM consists of 6 states, one for each possible scenario the traffic light may produce. Along with this FSM are 3 counters mentioned previously that work cohesively but have their own separate amount of time to count (15, 3, and 1 second). The finite state machine works as such: the traffic light will begin at state 0 displaying red in the North and South directions, and green in the East and West directions. This state will continue to occur until 15 seconds have been counted (first counter). Once 15 seconds have passed, the counter is cleared, and the FSM transitions to state 1 displaying red in the North and South directions and yellow in the East and West directions. Per regular traffic light rules regarding the yellow light, it should only stay on for a short period. As a result, state 1 will only occur for 3 seconds (second counter). After the counter finishes counting and is cleared, the system will move to state 2, displaying red in all 4 directions. Again, per traffic light rules, this state will occur for no more than 1 second (third counter). It will then move on to state 3, showing green in the North and South directions and red in the East and West directions (vice versa of state 0). It will remain at state 3 until it has counted 15 seconds (first counter) and then shifts to state 4. State 4 displays yellow in the North and South directions and red in the East and West directions for 3 seconds (second counter). Once 3 seconds have passed, it will jump to state 5 displaying red in all directions for 1 second (third counter). Finally, it returns to state 0 after 1 second, and the entire process repeats again. The full algorithmic state machine (ASM) for the regular traffic mode is shown in Figure 1.

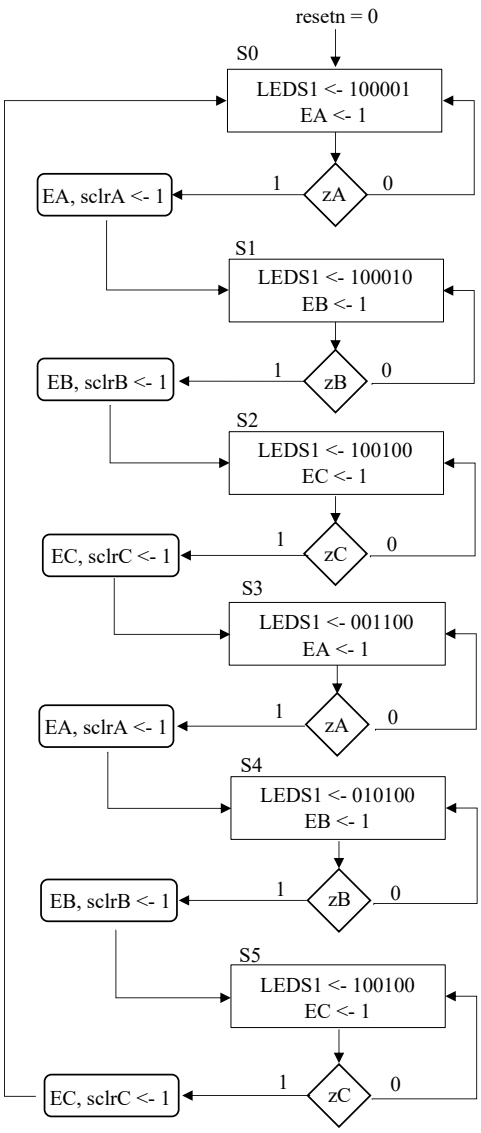


Figure 1: Regular Traffic Mode ASM State Diagram

### C. *Light Traffic Finite State Machine*

Continuing, the light traffic mode acts differently from the regular traffic mode, as the yellow and red lights flash (yellow for high traffic road, red for low traffic) on and off repeatedly second by second (on for 1 second, then off for 1 second). For this finite state machine there are only 2 total states. State 0 displays every light as off and the next state displays only the yellow and red lights (in opposite directions) as on. Fortunately, this FSM only requires the use of 1 counter (1 second) as that counter can be used for both states. This finite state machine works as such: the traffic light begins at state 0 and remains at state 0 until the counter reaches 1 second. Once the counter has reached 1 second, it will move to state 1 where the yellow and red lights will begin to flash. The yellow and red lights will remain lit until the same previous counter reaches 1 second again. The FSM will continue to loop the same system using that single counter, hence the "flashing" feature. The full algorithmic state machine (ASM) for the light traffic mode is shown in Figure 2.

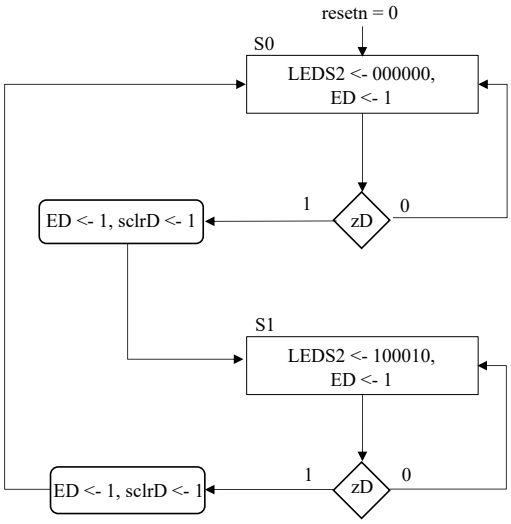


Figure 2: Light Traffic Mode ASM State Diagram

### D. *Selecting with a Multiplexer*

To differentiate between which traffic mode is being displayed, a 2-to-1 multiplexer was implemented into the design. This multiplexer simply allows the user to choose between which traffic mode to make active depending on the time of day, using a single switch.

### E. *Block Diagram*

In summary, the functionality of two different traffic modes, required the design to consist of 4 counters, 2 FSMs, and 1 multiplexer. The multiplexer determined which mode the system displayed and 2 FSM's, to control the regular and light traffic modes. To be specific, the multiplexor was controlled by a switch on the Nexys A7-100T board. When the switch was off, the default regular traffic mode was enabled, and the light traffic mode was enabled when the switch was on. The utilization of two different FSMs allowed each mode to work properly, without having to worry of an overlap between modes resulting in the system displaying the wrong signals. The block diagram of the entire system is shown in Figure 3.

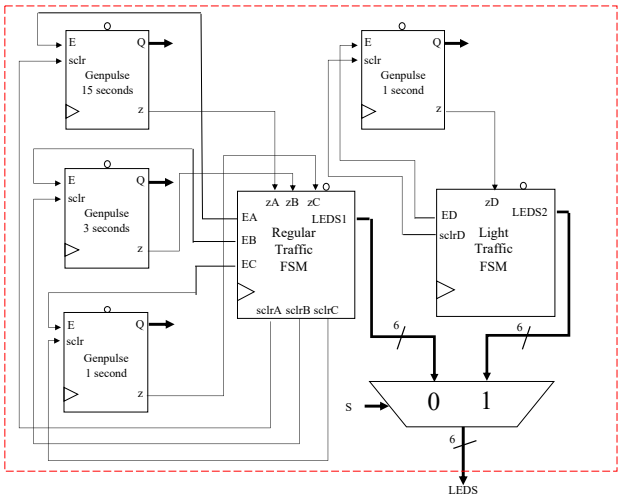


Figure 3: Block Diagram

## III. Experimental Setup

To properly demonstrate the functionality of our traffic light controller, 12 light-emitting diodes (LED) and their corresponding 220 ohm current-limiting resistors were placed on a breadboard and connected to a Nexys A7-100T board (3 LEDs for each direction). The connection between the breadboard and the Nexys board was accomplished using the PMOD port JA on the Nexys board. This port allows for the connection between peripheral modules and the FPGA by sending programmed signals from the board. Finally, one of the provided switches on the FPGA was used to ensure the proper traffic mode was being displayed. A photo of the peripheral traffic light system is shown in Figure 4.

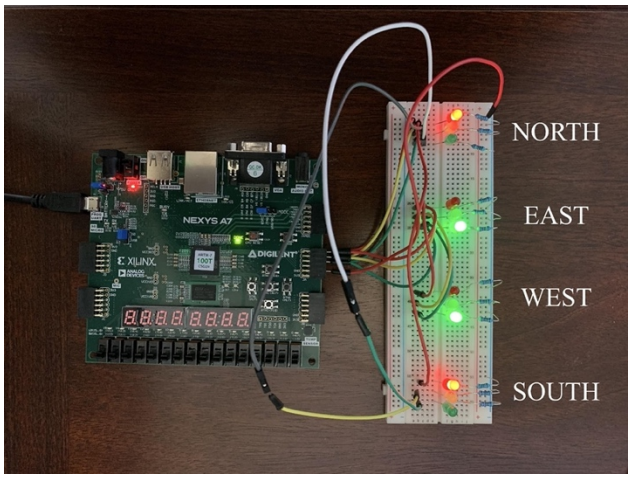


Figure 4: Peripheral Traffic Light System

## IV. Results

The results of this project are in line with expectations as the program was implemented seamlessly. This digital design system accurately simulates the functionality of a four-way traffic light with 2 different modes for regular and light traffic. Initially upon implementation, the state machines were not transitioning between states because the counter was counting to large values. Essentially, the counters were set for such a long time that the shifting between states was noticeable. However, after fixing the issues with the counter values, the traffic light correctly switched between states in the correctly timely manner. If for some reason the traffic light didn't operate correctly, it could easily be reset by pressing the reset button on the Nexys board. Figures 5 and 6 display the state transitions for both traffic modes.

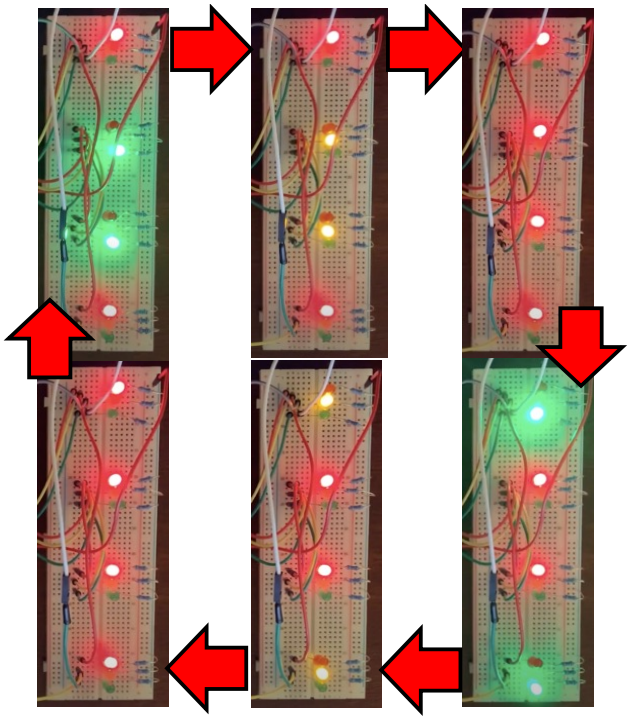

Figure 5: Regular Traffic Mode

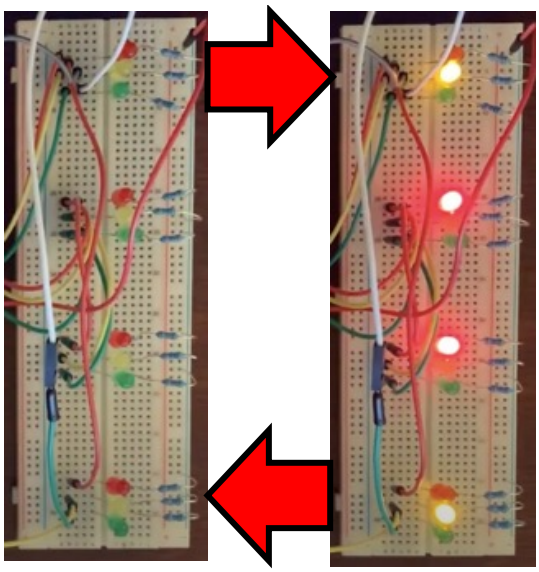

Figure 6: Light Traffic Mode

## CONCLUSIONS

This project allowed students to dive deeper into the programming of FPGAs using VHDL, and obtain a stronger understanding of how digital design systems are created. The issues that arose during the process of programming gave students hands-on experience with troubleshooting and the type of things that can occur when working with a system of this complexity. There were also numerous design issues that were faced. The use of the PMOD ports on the Nexys board was also something brand new to students, as they were forced to learn the functionality of those ports independently. Of course, there are numerous ways to improve this system as this design replicated a simple traffic light. Some improvements that could be made include adding the ability for oncoming traffic to turn left or right. A crosswalk system could also be implemented to help control the flow of traffic for pedestrians. Despite the issues and improvements stated, the four-way traffic light created still functions proficiently and accurately.

## REFERENCES


[1] Llamocca, Daniel. VHDL Coding for FPGAs, www.secs.oakland.edu/~llamocca/VHDLforFPGAs.html

[2] Brown, Arthur. "Nexys A7 Reference Manual." *Nexys A7 Reference Manual [Digilent Documentation],* Digilent, reference.digilentinc.com/reference/programmable-logic/nexys-a7/reference-manual.