****

**Department of Electrical and Computer Engineering**

**EGC320 - Digital System Design**

**Final Project: Traffic Control System**

**December 8th, 2020**

|  |  |
| --- | --- |
| **Group Members** | **Department** |
| **Nikiforos Fokas** | **EE** |

**Instructor:  Dr. Damu Radhakrishnan**

# Abstract

Since the invention of the automobile, directing the flow of traffic has become of utmost importance. The fictional road system contains a two-way vehicle intersection in addition to pedestrian crossings on all four sides. The road system mimics a very real possibility and one which may be seen in the real world. Verilog code was used to control the flow of traffic and pedestrians crossing through the intersections.

Table of Contents

[Abstract 1](#_Toc58185229)

[Introduction 3](#_Toc58185230)

[Design Procedure 3](#_Toc58185231)

[Road System 4](#_Toc58185232)

[Steps Towards Completion 4](#_Toc58185233)

[Traffic Light 7](#_Toc58185234)

[Crosswalk 8](#_Toc58185235)

[Block Diagram 8](#_Toc58185236)

[Verification 9](#_Toc58185237)

[Conclusions 14](#_Toc58185238)

[Appendix 15](#_Toc58185239)

# Introduction

The purpose of this project was to use previous knowledge of traffic control systems in Microcontrollers and implement it in Verilog to broaden my skillset. It was a smart idea to use a project where there is familiarity to learn a new platform.

# Design Procedure

The following design procedure was used to implement the Traffic Control System to the FPGA DE10 Lite board using Verilog code in Intel’s Quartus Prime Lite software. Using the System Builder tool from Terasic, I was able to configure the clock, the 7 segments displays, the switches, the onboard LED’s, the buttons (keys), and the IO pins. The System Builder also generated a skeleton code with all the module ports and their associated pins.

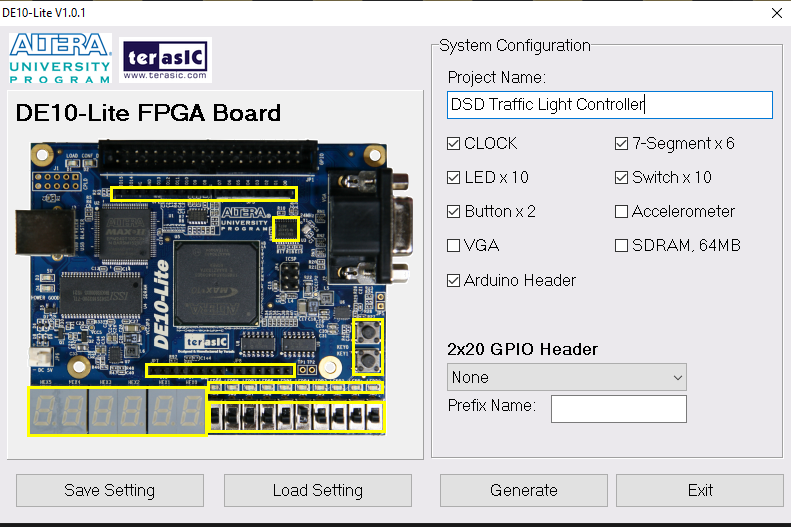
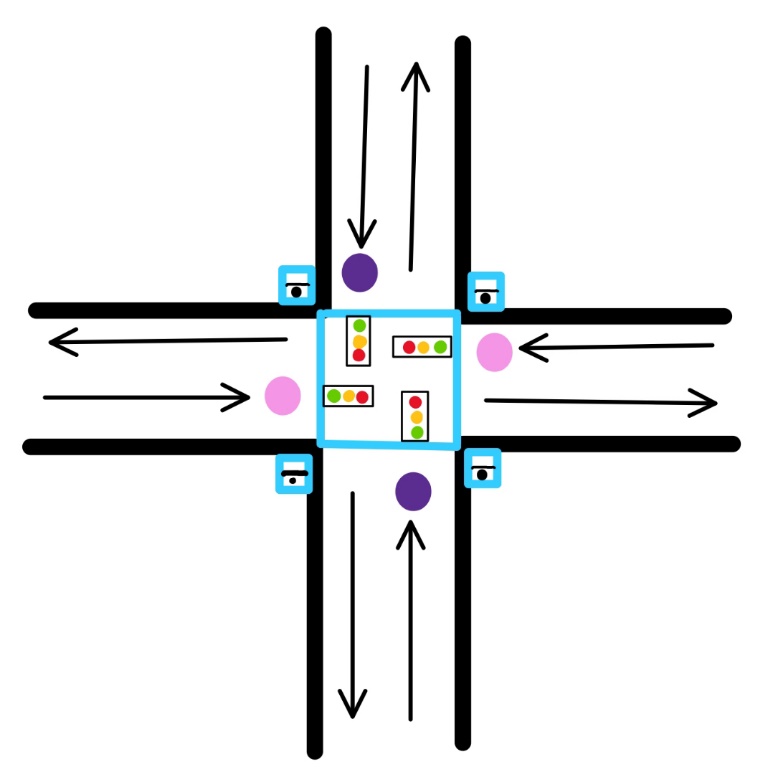


Figure 1: TERASIC SYSTEM BUILDER- CONFIGURES ON BOARD DEVICES AND GENERATES A PROJECT SKELETON

## Road System

To implement a traffic control system in Verilog, a road system needed to be designed to base the design from. To implement a traffic control system, we must understand the components for how the system will control vehicle traffic and foot traffic. The intersection will be controlled via a traffic light, with external LED’s telling the vehicles when it is safe to proceed. Additionally, the intersection will have pedestrian crossings from corner to corner, with a countdown shown by seven segment displays telling the pedestrians when it is safe to cross.



Pink dots indicate EW traffic light sensor (Key 1 on DE10 board)

Purple dots indicate NS traffic light sensor (Key 0 on DE10 board)

Blue lines indicate a crosswalk

Crosswalk switches (SW 0,1,8,9 on DE10 board)

Figure 2: ROAD SYSTEM DRAWING

## Steps Towards Completion

A few things that needed to be done before the project could be completed was to create a case for the North-South light to change, create a case for the East-West light to change, and create a case for the cross walk. Figure 5 shows the state diagram of the entire traffic control system.

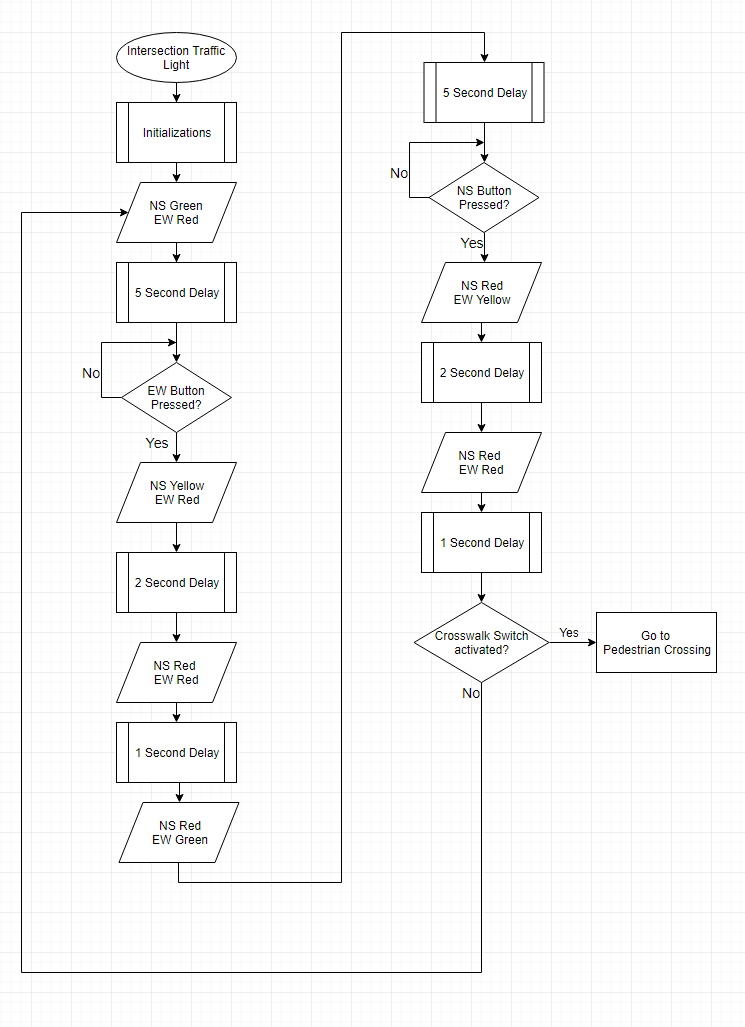


Figure : TRAFFIC LIGHT FLOWCHART

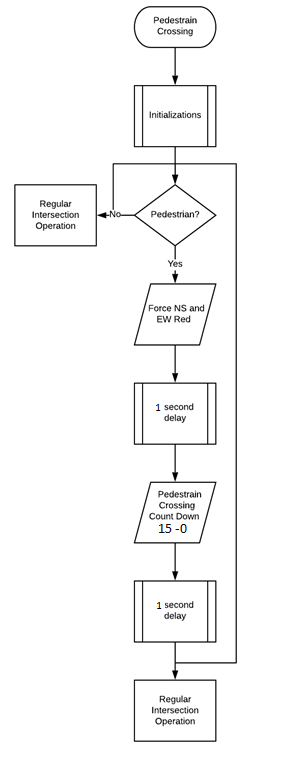


Figure : CROSSWALK FLOWCHART

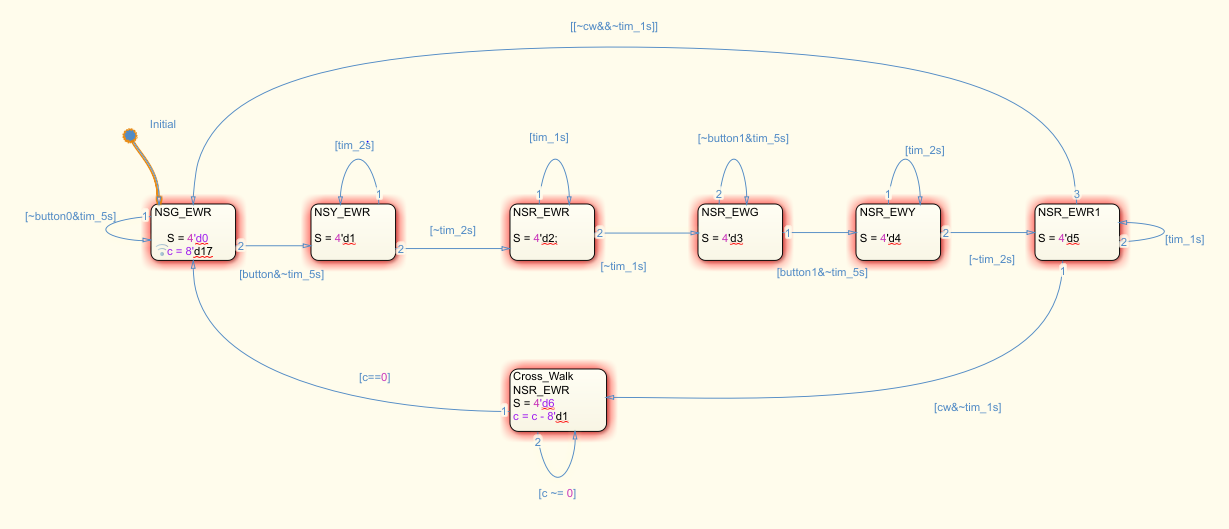


Figure : STATE DIAGRAM

### Traffic Light

The traffic light flow chart in Figure 3 shows the sequential steps of how the traffic light process works. Upon initialization, the North-South direction is Green, the East- West is red, and the cross walk says STOP. There is a five second delay and then it checks to see if the East-West sensor (Key 0 = 1) is active. If it is active, the North-South direction turns yellow, the East-West direction is still red, and the cross walk still says STOP. Then there is a two second delay before the North-South turns red and the East-West direction turns green. At this point, the cross walk has yet to change from STOP. There is another 5 second delay allowing the East-West side to be green long enough so vehicles can pass through. From here, it waits for the North-South sensor (Key 1=1) to be active. Once the North-South sensor is active, the North-South direction stays red, the East-West direction turns yellow, and the cross walk reads STOP. There is a two second delay before the North-South direction turns Green and the East-West direction turns red. At this point, the program has looped around to the beginning where it was for initialization.

### Crosswalk

The crosswalk flow chart in Figure 4 shows the sequential steps of how the crosswalk process works. Upon initialization, it checks if there is a pedestrian at any of the 4 cross walks (Switch 0,1,8, or 9 = 1). If there is no pedestrian, the intersection operates normally switching between the traffic lights. If there is a pedestrian, it forces the North-South direction and the East-West direction to change to red. Then there is a two second delay before the cross walk count down starts. This two second delay provides a safe environment for the pedestrians to walk, god forbid a vehicle runs the red light. After the two second safety delay, the seven segment displays show GO and a countdown starting at 15, counting down to 0. Once the counter hits 0, there is a two second delay and then it loops back around to initiate the regular traffic light operation.

## Block Diagram and Schematic

Figure 6 shows the block diagram with all the inputs and outputs mapped in the board. Figure 7 shows the circuit for the external LED’s and the board input pins.

Diagram

Description automatically generated

Figure : BLOCK DAIGRAM OF INOUTS MAPPED

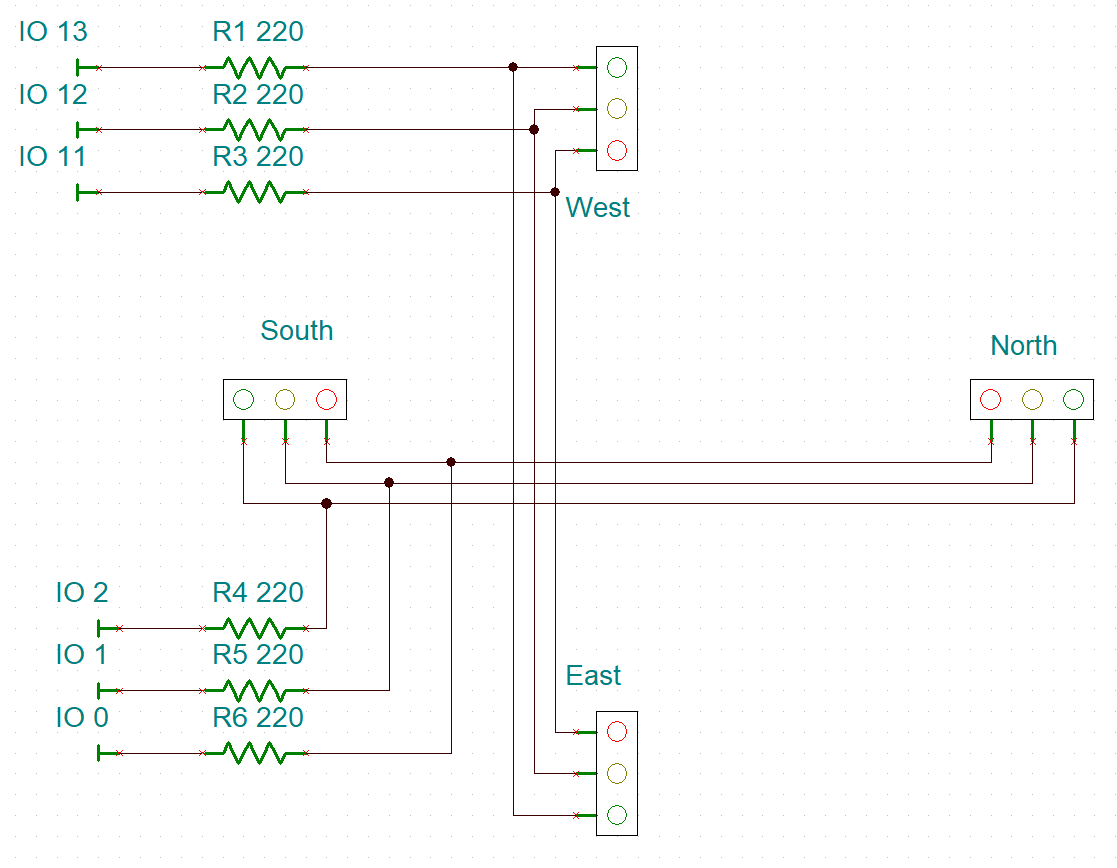


Figure : CIRCUIT OF EXTERNAL LEDS

# Verification

The following photos are provided to verify the design was successful. Figure 8 shows the initialization state where the North-South direction is red, and the East-West direction is red.

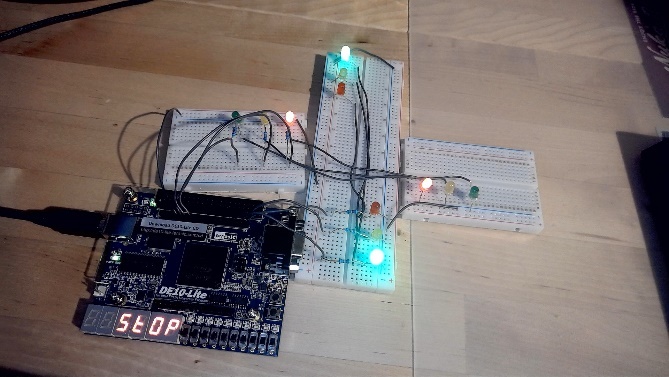


Figure 8: INITIALIZATION- NS=GREEN, EW=RED

Figure 9 shows the North-South direction turning yellow because the key 0 was pressed.

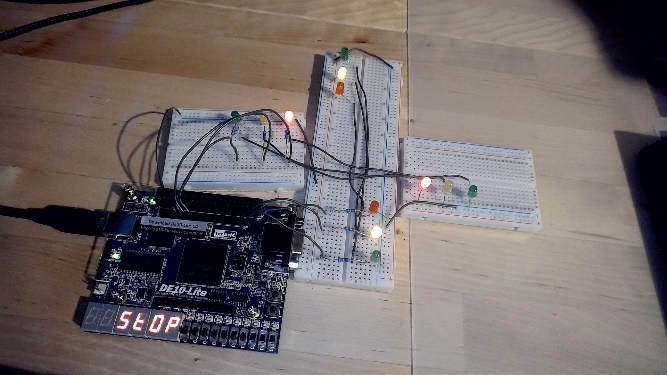


Figure 9: KEY 0 PRESSED- NS=YELLOW, EW=RED

Figure 10 shows all directions being red after the key 0 was pressed.

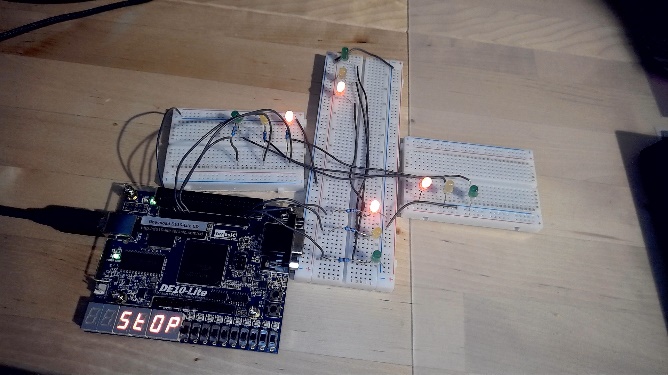


Figure 10: AFTER KEY 0 IS PRESSED- NS AND EW =RED

Figure 11 shows the East-West direction being green and the North-South direction being red because the key 0 was pressed.

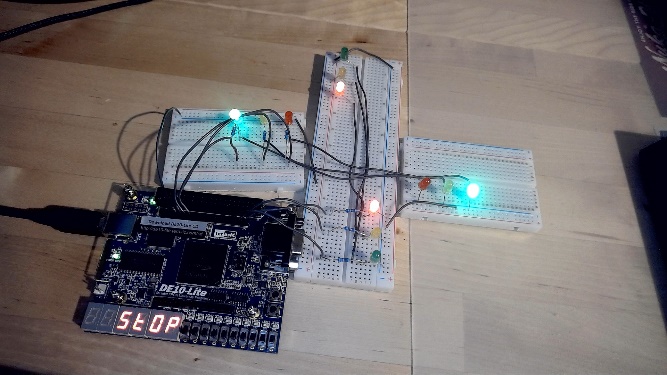


Figure 11: AFTER KEY 0 PRESSED- NS=RED, EW=GREEN

Figure 12 shows the East-West direction turning yellow and the North-South direction still being red.

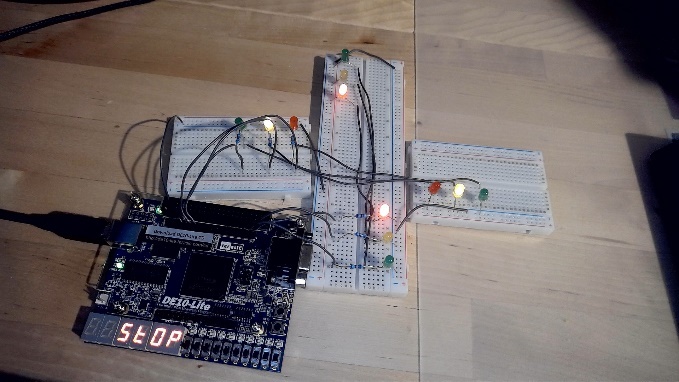


Figure 12: KEY 1 PRESSED- NS=RED, EW=YELLOW

Figure 13 shows all directions being red before it goes back to the initialization state.

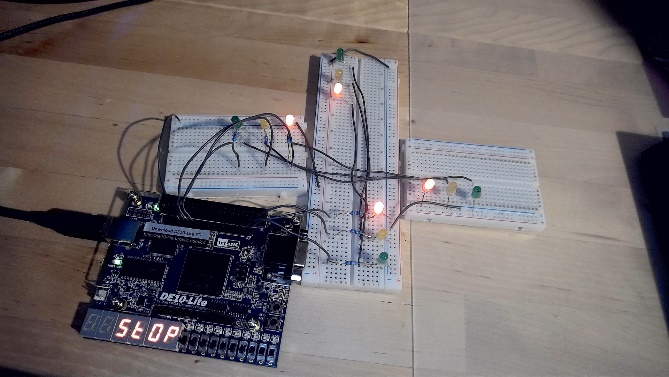


Figure 13: AFTER KEY 1 IS PRESSED- NS AND EW= RED

Figure 14 shows the North-South direction being green and the East-West being red just like initialization, except this time the four crosswalk switches are enabled indicating there is a pedestrian waiting to cross.

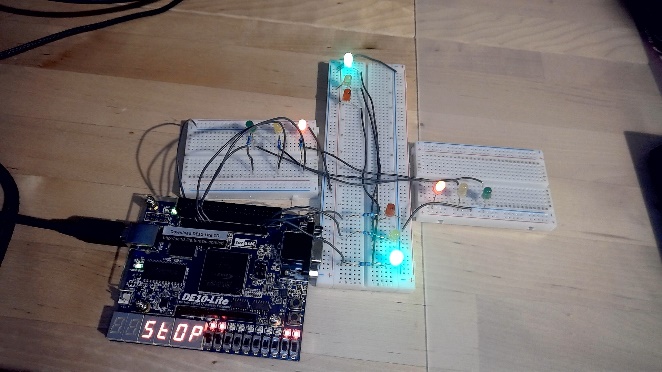


Figure 14: ANY OF THE FOUR SWITCHES =1 - INITIATE CROSSWALK

Figure 15 shows the start of the crosswalk countdown. After key 1 is pressed and both the North-South direction and the East-West direction have a chance to be green for 5 seconds, the countdown will begin. At the start of the crosswalk countdown, all directions are red.

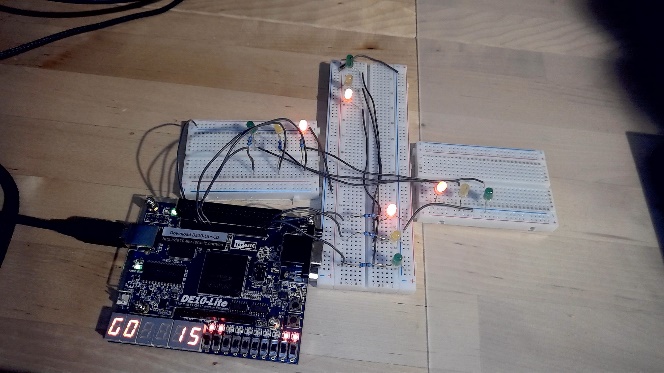


Figure 15: KEY 1 PRESSED- NS AND EW = RED, CROSSWALK STARTS COUNTDOWN AT 15

Figure 16 shows the countdown at 7 seconds to go until the countdown is over.

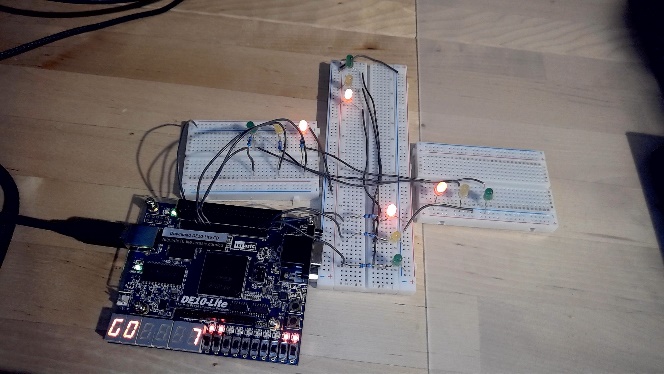


Figure 16: CROSSWALK COUNTDOWN AT 7, NS AND EW STILL = RED

Figure 17 shows the crosswalk countdown at the last second. Notice that all directions are still red.

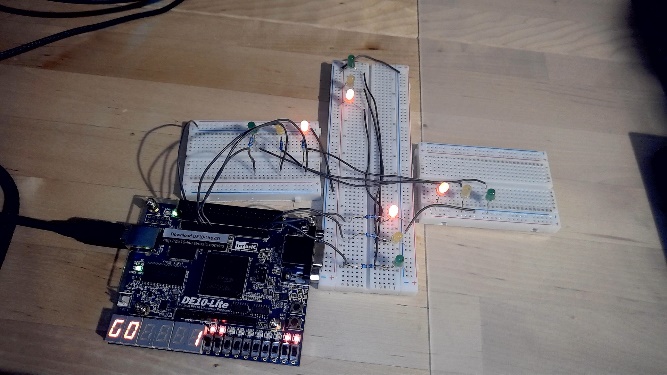


Figure 17: CROSSWALK AT 1, NS AND EW STILL = RED

Figure 18 shows the 2 second window after the crosswalk countdown is over before the North-South direction changes to green. All directions are red and the crosswalk says stop.

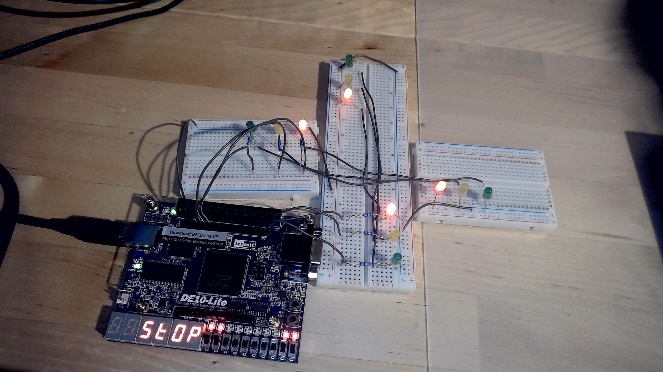


Figure 18: CROSSWALK COUNTDOWN ENDS, NS AND EW STILL = RED FOR SAFETY

Figure 19 shows the North-South direction as green and the East-West direction as red. At this point, the system is back at the initialization state and has been through a full cycle with a crosswalk countdown integrated in it.

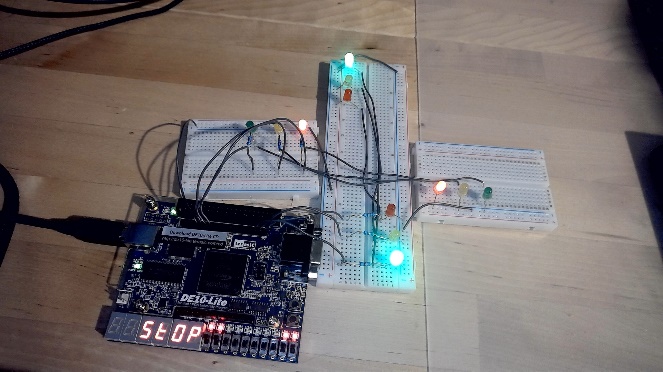


Figure 19: BACK TO INITIALIZATION STATE- NS = GREEN, EW=RED

# Conclusions

Throughout the semester, I have learned a lot about Verilog coding while implementing this project on a DE10 Lite FPGA board. The biggest problem I faced while working on this project was getting the crosswalk countdown to synchronize with the traffic light changing to red. The other problem I faced and overcame was slowing the internal clock down from 50 MHz to 1kHz by using a secondary clock. After overcoming these problems, I was able to time all the components correctly and have the crosswalk countdown initialize when all the directions were red.

# Appendix

