EE1100 Introduction to Electrical and Computer Engineering

Lab Report #2: Traffic Control Systems
Laboratory Team Members: Mahir Rahman and Lecture Section #001
Instructor: James M. Florence
Date: November 3, 2020

Abstract

In this report, we describe our design of an Arduino controlled system that controlled a traffic light system with a timer, and another Arduino controlled system that controlled a pedestrian crosswalk traffic light system. In TinkerCad, the programs were designed and tested. The hardware set up for both systems were successfully completed and the systems were simulated in TinkerCad. The results that were observed matched the expected outcome.

Introduction

The first traffic light system has different LEDs (Green, Yellow, and Red) that glow when programmed to. When the red light at East-West intersection glows, the green light will glow at North-South intersection. After sometime, the yellow light glows and green light stops glowing at N-S intersection while red light remains on at the E-W intersection. This is followed by only red light at both intersections. This part is needed to clear cars from the intersection before the new green light allows traffic to start moving. Then, the process repeats with green light in E-W intersection and red light at N-S intersection. For the second traffic light system, the process described above takes place for this system too. Along with that, if any pedestrian presses and holds the button to request crossing in North-South direction, a blue LED will turn on when green light for N-S traffic light set is on. The same happens in E-W direction. When button in the E-S direction is pressed and held, the blue LED glows when E-W traffic light set has green LED on. The traffic light system at an intersection is a vital and basic element in our day-to-day lives. Without proper traffic control, there would be more congestion of vehicles on roads and more accidents. The idea of a traffic light system was first put into practice in London in 1868. It was designed to manage the overflow of vehicles (mainly carriages and wagons driven by horses). They were especially useful to control traffic at night. The Washington State University writes that, “At night, gas-lit red and green lights were used, but still changed by a police officer. The lights became a safety hazard as they sometimes exploded and injured police officers.” This shows that the traffic lights were gas-lit, manual and a safety concern. In the 20th century, the system upgraded to a four-sided metal box with words “Stop” and “Go”. Wikipedia writes that the words were painted on a green background. They had green and red lenses which would glow brightly due to placement of kerosene lamps. However, it was still manually controlled by a police-officer who would also blow a whistle to alert travelers just before the lights were switched. In 1912, the first electric traffic lights with two lights (green and red) were put into use. This developed to the first four-way, three color traffic light system by 1920. Yellow lights were introduced to notify drivers that the signal would change soon. The traffic light systems made another huge leap with the introduction of computers. Pressure plate detection rose into prominence. This system would detect number of cars waiting at a signal. By the 1990s, we finally see that countdown timers for traffic light systems appear. Timers allow drivers to know how long they have to wait at a signal, and how much time pedestrians have to cross streets. The motivation for such a system stems from its importance and urgency. Without traffic lights, it would put an immense burden on the traffic control department as more officers will need to be assigned at each intersection. The likelihood of quarrels, congestion, and accidents may increase. Countdown timers for traffic light systems enable the whole process to be digitalized and efficient. Therefore, the development of traffic light systems is a very important and necessary aspect in traffic control.

Procedure and Design

For the first system, there are two sets of traffic lights at an intersection. The first set controls traffic going North-South while the other set coordinates traffic going East-West. In total, there are six LEDS, with three different colors (Red, Green, and Yellow) at each set. When the green LED at N-S traffic light set is on, the red light at E-W traffic light set is on. After 3.5 seconds, the yellow light at N-S turns on and red light turns off. At the other set, red light will continue to be illuminated. Then, yellow light turns off after a second, and red light on both sets turn on. After 1 second, red light at E-W set turns off and green light turns on while the red light at N-S remains on for 3.5 seconds. The cycle repeats and alternates between the two sets. We will be designing our system in TinkerCad, and we will need to access Arduino Uno R3 from the component section in TinkerCad. Along with that, we will need connecting wires, a breadboard, six LEDs (two green, two yellow, and two red), and six 220 Ω resistors. The figure below demonstrates the set-up:

Timer controlled traffic light system using Arduino

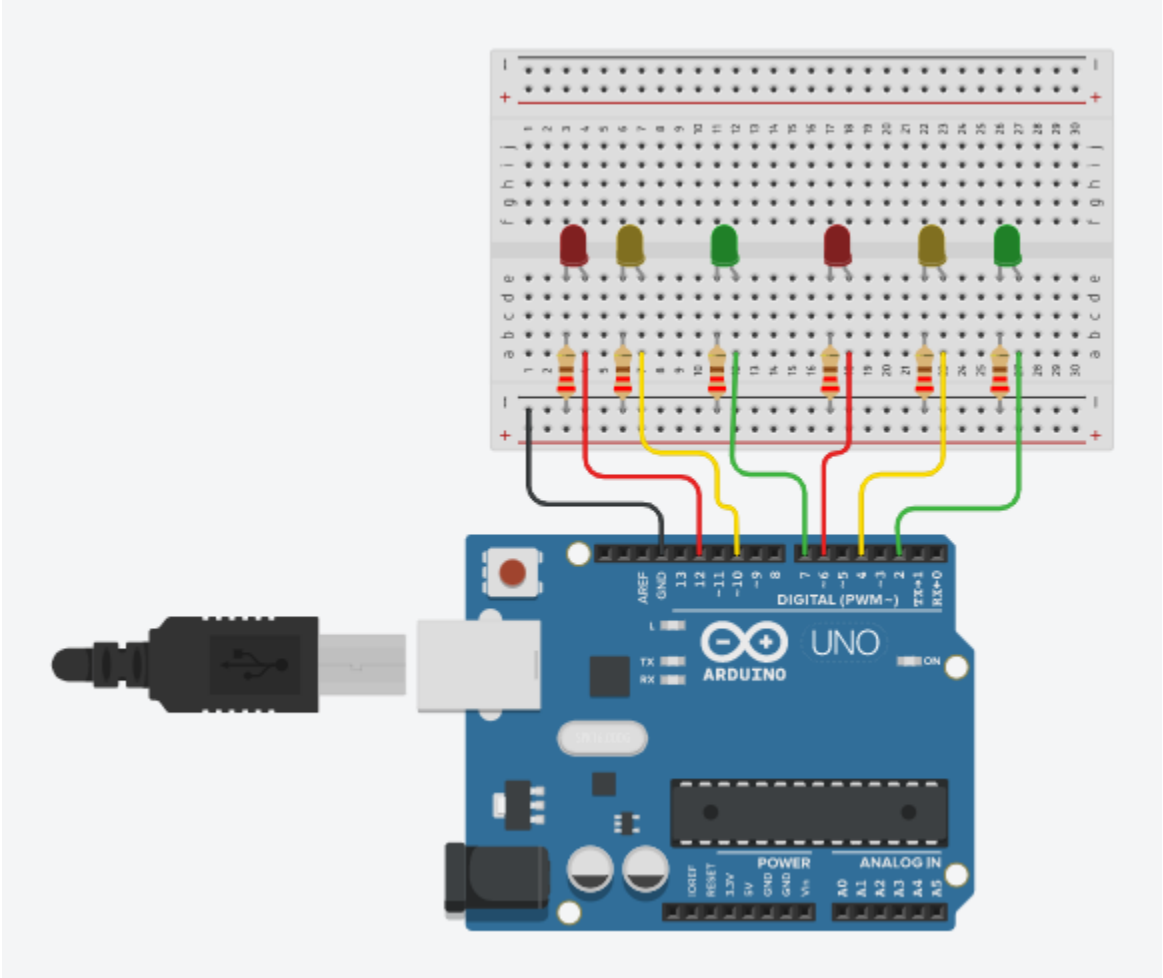

Figure 1: A TinkerCad layout including six LEDS

From Figure 1, we observe a breadboard beside an Arduino Uno R3. A black connecting wire connects the GND pin of Arduino to the negative terminal of breadboard. Right beside that, a red wire connects pin 12 to the anode of red LED present on the left hand side and a resistor of 220 Ω connected to cathode of LED to complete the circuit. The resistor prevents excessive current flow through the LED and damaging the LED. It is connected to cathode of all six LEDS. The connection to pins on Arduino allows a voltage across the LEDS. A yellow wire connects the anode of yellow LED on the left hand side to pin 10 and a green wire connects the anode of green LED on the left hand side to pin 7. Arduino pins 6, 4, and 2 connect the right hand side green, yellow, and red LED respectively. All the pins have digital output and so the LEDs have either a high voltage across it or 0 V.

Now, in the program, the left hand side (N-S traffic lights) red, yellow, and green LEDs shown in Figure 1 have been assigned variables and their names ledPin1, ledPin2, and ledPin3 respectively. The right hand side (E-W traffic lights) red, yellow, and green LEDs have been named ledPin4, ledPin5, and ledPin6 respectively. pinMode allows us to assign components as either INPUT or OUTPUT. All these LEDS will be used as OUTPUT as they will display different colors depending on existing conditions. After that, a loop is created that will continue to run as long as we want. The program starts with ledPin1 and ledPin6 set to high using digitalWrite statements while all the other LEDS are set to low. To keep them lit up for 3.5 seconds, as described in the logic of our design, a delay statement is used. This statement is given a value of 3500 µs, and helps us delay the program from moving on to the next set of programming statements for as long as we like. Next, digitalWrite statements are used to set ledPin5 to high and ledPin6 to low (ledPin1 remains high). A delay statement is used after that to keep yellow light on for 1 second. Then, ledPin4 is set to high and ledPin5 to low. When this happens, both red lights at N-S and E-W are on. There is also a delay of 1 second for both red lights to allow cars to clear from the intersection. Next, ledPin3 is set to high and ledPin1 to low. This process repeats for the N-S direction and the loop continues on until we stop the simulation.

In our second traffic light system, it is similar to the first traffic light system except the second system has two pedestrian crosswalks. If a pedestrian presses and keeps on holding the N-S crosswalk button for N-S direction while the process for the first traffic light system continues, a blue light will turn on when the N-S traffic light stand's green LED is on. This indicates to the pedestrians that it is safe to cross the street in the N-S direction. The same process happens in the E-W direction. A pedestrian wanting to cross in the E-W direction needs to press and keep on holding the E-W button until green light at E-W traffic stand turns on. This will turn on a separate blue LED to notify pedestrians it is safe to cross in E-W direction. Both blue LEDs turn off when green LEDs turn off. This system is also designed in TinkerCad and we will need to access Arduino Uno R3 from the component section in TinkerCad. Along with that, we will need connecting wires, two pushbuttons, a breadboard, eight LEDs (two green, two yellow, two blue, and two red), two 10,000 Ω resistors, and eight 220 Ω resistors. The figure below demonstrates the set-up:

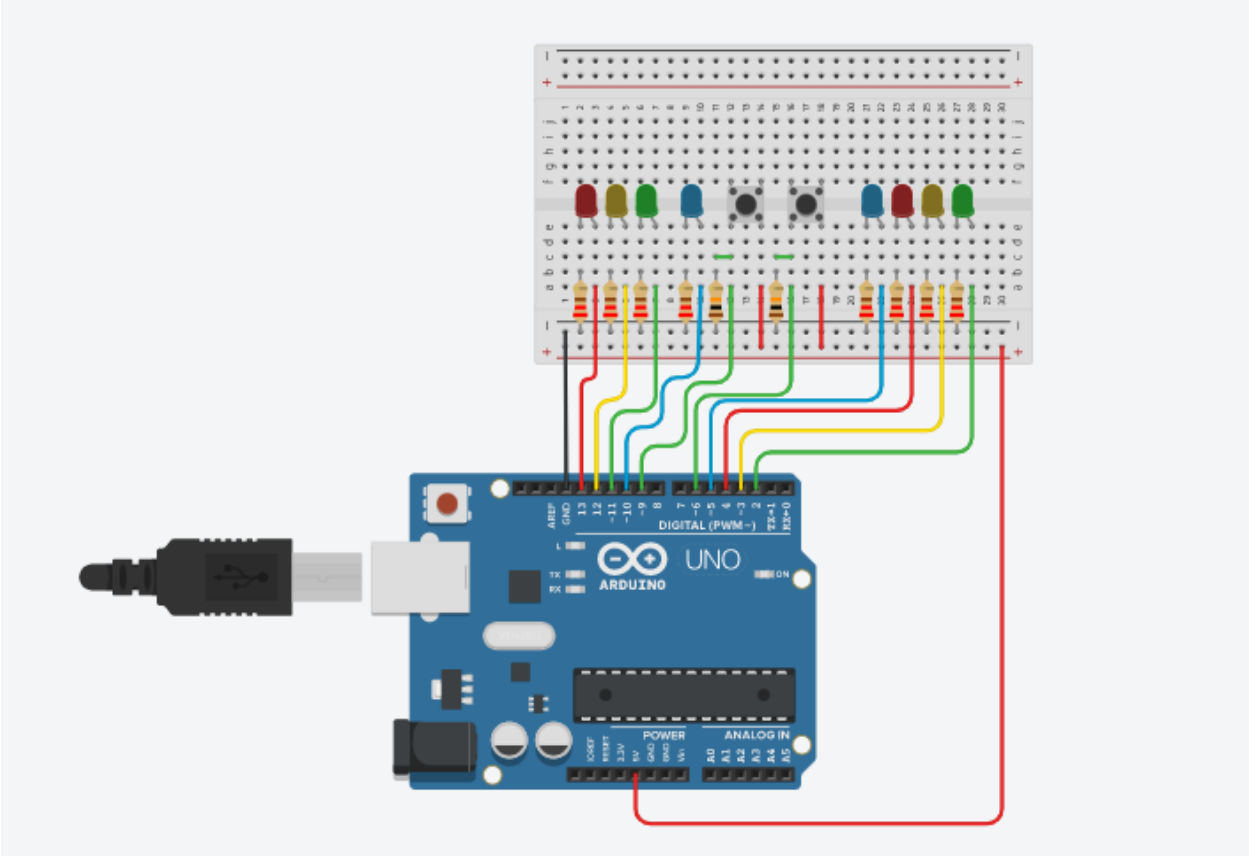

Figure 2: A TinkerCad layout including eight LEDs and two controlling buttons

From Figure 2, we observe a breadboard beside an Arduino Uno R3 just like Figure 1. A black connecting wire connects GND pin of Arduino to negative terminal of breadboard. A red wire connects 5 V pin of Arduino to positive terminal of breadboard. All LEDs have their anode connected to a specific pin, ranging from pin 13 to 2, and each of their cathodes are connected to 220 Ω resistors and then to negative terminal of breadboard. The left hand side red, yellow, green, and blue (N-S traffic lights) are connected to pins 13, 12, 11, and 10 while the right hand side (E-W traffic lights) red, yellow, red, and blue LEDs are connected to pins 4, 3, 2, and 5 respectively. Both pushbuttons have their terminal 2a connected to positive terminal of breadboard that has a 5 V volt supply running across it. Terminal 1a of both pushbuttons are connected to 10,000 Ω resistors which connects it to negative terminal of breadboard. The 10 kΩ resistors act as “pull-up” resistors. The left hand side push button is connected to pin 9 while the right hand side push button is connected to pin 6 of the Arduino. All the pins have digital output and so the LEDs have either a high voltage across it or 0 V.

Now, in the program, the left hand side (N-S traffic lights) red, yellow, green, and blue LEDs shown in Figure 2 have been assigned variables and their names ledPin1, ledPin2, ledPin3, and ledPin7 respectively. The right hand side (E-W traffic lights) red, yellow, green, and blue LEDs have been named ledPin4, ledPin5, ledPin6, and ledPin8 respectively. The left hand side button is assigned as buttonPin2 and right hand side button is assigned to buttonPin1. Each buttons also

has another variable associated with them. They are named the buttonState1 (for buttonPin1) and buttonState2 (for buttonPin2), and are used to check whether buttons have been pressed and held or not. pinMode allows us to assign components as either INPUT or OUTPUT. All these LEDS will be used as OUTPUT as they will display different colors depending on existing conditions. The buttons are set up to be used as INPUT using pinMode. After that, a loop is created that will continue to run as long as we want. The program starts with reading the value of buttonState1. If buttonState1 is high (right-hand side button pressed and held), ledPin1, ledPin8 (blue LED), and ledPin6 set to high using digitalWrite statements while all the other LEDS are set to low. To keep them lit up for 3.5 seconds, as described in the logic of our design, a delay statement is used. This statement is given a value of 3500 µs, and helps us delay the program from moving on to the next set of programming statements for as long as we like. If right hand side button is not pressed and held, the program executes the else portion of the program where only ledPin1 and ledPin6 are set to high. Next, digitalWrite statements are used to set ledPin5 to high and ledPin6 to low (ledPin1 remains high). A delay statement is used after that to keep yellow light on for 1 second. Then, ledPin4 is set to high and ledPin5 to low. When this happens, both red lights at N-S and E-W are on. There is also a delay of 1 second for both red lights to allow cars to clear from the intersection. The program then reads in the value of buttonState2 (left-hand side button). If left-hand side button was pressed and held, ledPin3and ledPin7 (blue LED) are set to high and ledPin1 to low. It remains on for 3.5 seconds, and then ledPin3 and ledPin7 are set to low and ledPin2 is set to high (left hand side yellow LED glows). If buttonState2 was low, only ledPin7 (blue LED) is not turned on. Then, it goes in the same way as in first traffic light system. The ledpin2 is set to low after 1 second, and ledPin1 is set to high for1 second. This process repeats as the loop begins all over again, and continues on until we stop the simulation.

Results

Here is a hyperlink to my first TinkerCad timer controlled traffic light system without the pedestrian crosswalks:

https://www.tinkercad.com/things/2bPp9GVZHom-timer-controlled-traffic-light-system/editel?sharecode=pS1Hb4KTNB5yupFDklsN64HnFF1qjmMEmaBKv08iJ20

There are no buttons. Once simulation is started, the traffic lights will automatically switch colors at the appropriate times.

Here is a hyperlink to my TinkerCad pedestrian crosswalk traffic system:

https://www.tinkercad.com/things/eeajeWRsojg-traffic-light-system-with-pedestrian-crosswalk/editel?sharecode=jG4QY-P6__eYte5nkha0ZqWprbIJzQKHAwVpFo9sqjg

The two buttons can be used to send request to allow pedestrian crossing. When the blue LED is on, it is safe to cross road.

Discussion and Conclusions

The objectives of the lab exercise were met. The traffic light systems follow the logic discussed in the above sections and the LEDs glow at the appropriate times. The project was successfully designed and implemented the required hardware and functionality in TinkerCad. Regarding conceptual insight, I learnt more about the programming language C and about the different functions of an Arduino. There were a few difficulties such as adding the pedestrian crosswalk to the system and writing the program to control the blue lights; however, both systems work without any errors now. If I were to design this work again, I would add a feature that detects presence of cars. If there are no cars in front of a set of traffic lights that has green LED on, the traffic light would automatically switch to red and allow the road with the most cars to go instead.

References/Appendices

I worked alone on this project. I used two outside sources for my lab report. A listing of those sources in MLA format is given below:

Clark, Larry. "Traffic signals: A brief history." *Washington State Magazine*, wsu.edu/web-extra/traffic-signals-a-brief-history/. Accessed 29 October 2020.

"Traffic Light." *Wikipedia*, Wikimedia Foundation, en.wikipedia.org/wiki/Traffic_light. Accessed 29 October 2020.

Copy of my Arduino program that controls the first traffic light system with a timer is given below:

```
/*************************************************************

  Author:     Mahir Rahman

  Course:     CE1100.001

  Date:       10/25/2020

  Assignment: Traffic Light Lab Report Part 1

  Compiler:   Arduino 1.8.13



  Description:

  This program is used to control traffic lights at an

  intersection.
```

```
**********************************************************/

//These variables store the pin number to which each
//LED is connected.
const int ledPin1 = 12;
const int ledPin2 = 10;
const int ledPin3 = 7;
const int ledPin4 = 6;
const int ledPin5 = 4;
const int ledPin6 = 2;

void setup() {
  // initialize all digital pins assigned above as an output.
  pinMode(ledPin1, OUTPUT);
  pinMode(ledPin2, OUTPUT);
  pinMode(ledPin3, OUTPUT);
  pinMode(ledPin4, OUTPUT);
  pinMode(ledPin5, OUTPUT);
  pinMode(ledPin6, OUTPUT);

}

// the loop function runs over and over again forever
```

```
void loop()
{
  //This portion turns on the green light at the
  //East-West intersection, and red light on North-
  //South intersection for 3.5 seconds
  digitalWrite(ledPin1, HIGH);
  digitalWrite(ledPin2, LOW);
  digitalWrite(ledPin3, LOW);
  digitalWrite(ledPin4, LOW);
  digitalWrite(ledPin5, LOW);
  digitalWrite(ledPin6, HIGH);
  delay(3500);
  //The following statements turn off green light and turn
  //on yellow light at E-W for 1 second.
  digitalWrite(ledPin5, HIGH);
  digitalWrite(ledPin6, LOW);
  delay(1000);
  //This part turns off yellow light and turns on
  //red light at E-W
  digitalWrite(ledPin4, HIGH);
  digitalWrite(ledPin5, LOW);
  delay(1000);
  /*The following statements turn on green light and turn
```

```
   off red light at N-S intersection*/
  digitalWrite(ledPin1, LOW);
  digitalWrite(ledPin3, HIGH);
  delay(3500);
  /*This part turns on yellow light at N-S and turns red light
  off*/
  digitalWrite(ledPin2, HIGH);
  digitalWrite(ledPin3, LOW);
  delay(1000);
  //Turns on red light at both intersections
  digitalWrite(ledPin2, LOW);
  digitalWrite(ledPin1, HIGH);
  delay(1000);



} //end void loop()
```

Copy of my Arduino program that controls the second traffic light system and processes requests to allow pedestrian crossing is given below:

```
/*********************************************************
  Author:     Mahir Rahman
  Course:     CE1100.001
  Date:       11/3/2020
  Assignment: Traffic Light Lab Report
```

```
  Compiler:   Arduino 1.8.13


  Description:
  This program is used to control traffic lights at an
  intersection and accept requests to allow pedestrians
  to cross.
**********************************************************/


//These variables store the pin number to which each
//LED is connected.
const int ledPin1 = 13;
const int ledPin2 = 12;
const int ledPin3 = 11;
const int ledPin4 = 4;
const int ledPin5 = 3;
const int ledPin6 = 2;
const int ledPin7 = 10;
const int ledPin8 = 5;
//These variables store the pin number to which
//each button is connected
const int buttonPin1 = 6;
const int buttonPin2 = 9;
```

```
//This declares and initializes button states
//to 0
int buttonState1 = 0;
int buttonState2 = 0;

void setup() {
  // initialize all digital pins assigned above as an output.
  pinMode(ledPin1, OUTPUT);
  pinMode(ledPin2, OUTPUT);
  pinMode(ledPin3, OUTPUT);
  pinMode(ledPin4, OUTPUT);
  pinMode(ledPin5, OUTPUT);
  pinMode(ledPin6, OUTPUT);
  pinMode(ledPin7, OUTPUT);
  pinMode(ledPin8, OUTPUT);

  //initialize all button pins assigned above as input
  pinMode(buttonPin1, INPUT);
  pinMode(buttonPin2, INPUT);

}  //end void setup()

// the loop function runs over and over again forever
```

```
void loop()
{
  buttonState1 = digitalRead(buttonPin1);

  //This portion turns on the green light at the
  //East-West intersection and red light on North-
  //South intersection for 3.5 seconds
  //If button 1 was pressed and held, blue LED turns
  //on at E-W intersection for 3.5 seconds
  if (buttonState1 == HIGH)
  {
    digitalWrite(ledPin1, HIGH);
    digitalWrite(ledPin2, LOW);
    digitalWrite(ledPin3, LOW);
    digitalWrite(ledPin4, LOW);
    digitalWrite(ledPin5, LOW);
    digitalWrite(ledPin6, HIGH);
    digitalWrite(ledPin8, HIGH);
    delay(3500);
    digitalWrite(ledPin8, LOW);
  }
  else
  {
```

```
digitalWrite(ledPin1, HIGH);
digitalWrite(ledPin2, LOW);
digitalWrite(ledPin3, LOW);
digitalWrite(ledPin4, LOW);
digitalWrite(ledPin5, LOW);
digitalWrite(ledPin6, HIGH);
delay(3500);
}  //end if (buttonState1 == HIGH)
//The following statements turn off green light and turn
//on yellow light at E-W for 1 second.
digitalWrite(ledPin5, HIGH);
digitalWrite(ledPin6, LOW);
delay(1000);
//This part turns off yellow light and turns on
//red light at E-W
digitalWrite(ledPin4, HIGH);
digitalWrite(ledPin5, LOW);
delay(1000);
/*The following statements turn on green light and turn
  off red light at N-S intersection*/
//If button 2 was pressed and held, blue LED turns
//on at N-S intersection for 3.5 seconds
buttonState2 = digitalRead(buttonPin2);
```

```
if (buttonState2 == HIGH)
{
  digitalWrite(ledPin1, LOW);
  digitalWrite(ledPin7, HIGH);
  digitalWrite(ledPin4, HIGH);
  digitalWrite(ledPin3, HIGH);
  delay(3500);
  digitalWrite(ledPin7, LOW);
}
else
{
  digitalWrite(ledPin1, LOW);
       digitalWrite(ledPin3, HIGH);
       digitalWrite(ledPin4, HIGH);
       delay(3500);
}  //end if (buttonState2 == HIGH)
/*This part turns on yellow light at N-S and turns red light
off*/
digitalWrite(ledPin2, HIGH);
digitalWrite(ledPin3, LOW);
delay(1000);
//Turns on red light at both intersections
digitalWrite(ledPin2, LOW);
```

```
  digitalWrite(ledPin1, HIGH);
  delay(1000);

}  //end void loop()
```

END