Lab 2 – Arduino Stoplight

Online Link:

This lab is available as part of my online portfolio at: https://github.com/Brianhayden7/arduino-led-stoplight

Objective

The purpose of this lab is to learn the basic principles of the arduino development platform and GPIO by creating a wifi-controlled stoplight. These principles include:

- Identifying GPIO pins and referencing them in code.
- Become familiar with representing the system in different views (ie state diagram, flow chart and components)
- Become familiar with coding in c using the arduino IDE

Materials

I used the following materials to accomplish this lab:

- Personal computer
- Arduino wemos d1 mini
- MicroUSB Cable
- 1 x breadboard
- 1 x Red LED
- 1 x Yellow LED
- 1 x Green LED
- 3 x 100 Ω Resistor (BrBlBrGold)
- Jumper Wires

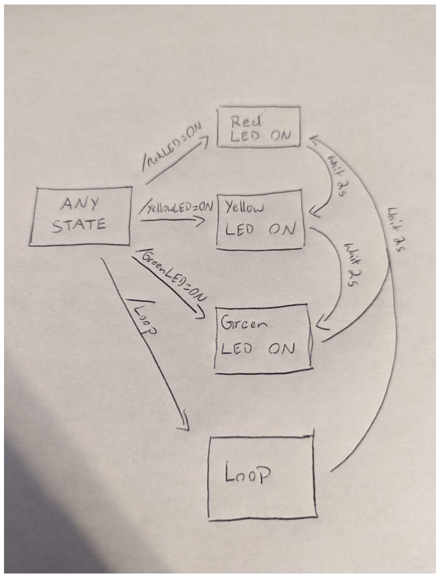
References

I used the following resources in this lab:

- https://github.com/Brianhayden7/LED-Stoplight My previous led stoplight implantation using the raspberry pi
- https://github.com/todddb/example-lab The example lab that had info on how to run a web server on an arduino and connect to wifi

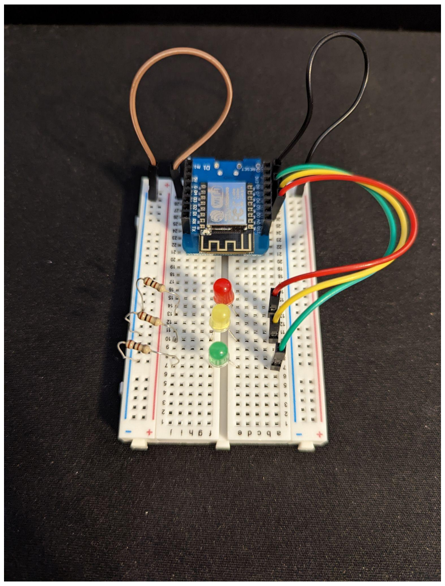
Procedures:

1. Download and install the arduino IDE.
2. Search for the library that includes the esp8266 boards.
3. Configure the arduino IDE to use the wemos d1 mini and select the correct port.
4. Start coding up the same logic used before in the raspberry pi version of this project.
5. Model the functionality, logical flow and components of the system. Include a schematic diagram.
    a. ***Functionality*** - This system is a basic state machine, represented by the following diagram:



- Upon startup, it will start it the off position. You essentially can get to any state from any other state in the diagram so the logic needs to account for this.

b. *<u>Component Diagram</u>* and *<u>Schematic Diagram</u>* – This system is represented by the photos below.



6. Program the logic behind your webpage
   a. Set the GPIO pins you want to use to output so you can send them power.
   b. Define each route you will use and make sure the lights do what you want.
   c. Upload code to arduino and it will start to run automatically when powered
7. In order to get the rotating lights to work
   a. Using a non-blocking loop and a flag, I set up 3 IF statements within the main loop() function that use the millis() function and a boolean variable to check if the loop route has been visited and rotate through the lights until a different route is chosen.

Client Interactions

The web page uses routes in order to initiate procedures coded into the arduino. When they go to the webpage, there are several buttons to choose from. Each of these buttons uses and aref to go to one of these routes. They are as follows.

/redLED=ON

/yellowLED=ON

/greenLED=ON

/loop

These are currently just called by using the buttons but could be called outside of the web page as well in the future by using GET requests.

Observations

I've used the arduino platform before so getting things set up and coding with it right away was no issue. Especially since we walked through adding the board in during class it was easy to just get going. There were a few things I used to get the whole project put together. I used my last raspberry pi project to get most of the logic, the example lab for how to set up the web server on arduino, and some googling to debug some errors.

I feel like this project wasn't a whole lot different from the raspberry pi lab other than the web server and the way that I handled the loop. I think in both regards, it was actually easier. The solutions for both of these implementations were actually simpler on the arduino compared to the hack that I used on the raspberry pi.

The circuitry wasn't super complex but making it look kind of simple and clean was kind of fun. Somebody looking at it that doesn't know a lot about these things would definitely be impressed.

Thought Questions

1. What are some key differences between developing this lab on a Raspberry Pi, and developing on Arduino?

For one thing, developing on the raspberry pi offers a lot more flexibility in terms of languages and other tools for interacting to with the physical components. Arduino is very tied in to its IDE and coding in the C language. This kind of makes things simpler in a way because of how few options this leaves.

2. What are the strengths and trade-offs of each of these platforms?

The strengths of the raspberry pi are it's versatility like I stated above but this is also a bad thing sometimes. There are a lot of ways you can get things done on the raspberry pi but that makes it hard to do things clean or efficiently sometimes. The arduino is bad because of the lack of options it provides but this leads to simple implementations that work well.

3. How familiar were you with the Arduino platform prior to this lab?

I started messing around with the Arduino platform back in highschool when I bought my first board but never did much with it. I've messed around with them here and there since then but only did one real project with them about a year ago so I know what resources to look for and how to work the system.

4. What was the biggest challenge you overcame in this lab?

One challenge was getting the loop to work well without interruptions and unnecessary delays. Learning not only how to use non-blocking loops to make it interruptible, but also how to use a flag to enable those loops. This is how I went about adding in a routine that runs continuously until I want it to stop without any time delay or funny business.

5. Please estimate the total time you spent on this lab and report.

I spent about 3 hours coding up the lab and another 2 hours on this report.

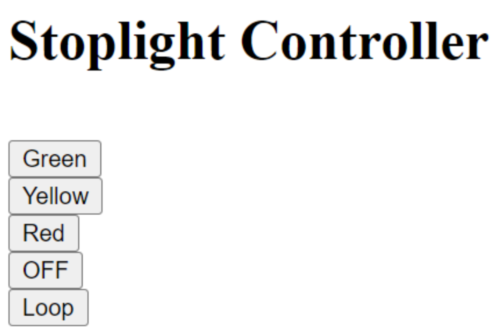
Certification of Work

I certify that the solution presented in this lab represents my own work. In the case where I have borrowed code or ideas from another person, I have provided a link to the author's work in the references and included a citation in the comments of my code.

--Brian Hayden

Appendix

Appendix 1: System Interface - Web Page



Appendix 2: Arduino Code

(available at: https://github.com/Brianhayden7/arduino-led-stoplight)

```
#include <ESP8266WiFi.h>

const char* wifiSSID = "Brian's Pixel"; // In order for this to work, you MUST specify the SSID for your wifi
const char* wifiPSK = "00000007"; // And the preshared key (wifi password)

int mytime = 0;
int redLED = D5;
int yellowLED = D6;
int greenLED = D7;
bool isLOOP = false;
bool firstLoop = true;

WiFiServer server(80); // This sets which port our server will listen on

//WiFiClient client = server.available(); // Create a new client object for available connections

void setup() {
```

```
// put your setup code here, to run once:
Serial.begin(9600);
// int myTime = 0;
pinMode(redLED, OUTPUT);
pinMode(yellowLED, OUTPUT);
pinMode(greenLED, OUTPUT);
digitalWrite(greenLED, LOW);
digitalWrite(yellowLED, LOW);
digitalWrite(redLED, LOW);

// ** Connect to WiFi network - Adapted from http://www.esp8266learning.com/wemos-webserver-example.php

Serial.print("Connecting to "); // Display debugging connection info

Serial.println(wifiSSID); // Print the configured SSID to the Serial Monitor

WiFi.begin(wifiSSID, wifiPSK); // Use the provided SSID and PSK to connect

while (WiFi.status() != WL_CONNECTED) { // If not connected to wifi

delay(500); // Pause

Serial.print("."); // Print a dot each loop while trying to connect

}

Serial.println("");

Serial.println("WiFi connected"); // Print "connected" message to the Serial Monitor

server.begin(); // Start the web server
```

```
  Serial.println("Server started");


  Serial.print("Use this URL : "); // Print the connected IP address to the Serial Monitor


  Serial.print("http://");


  Serial.print(WiFi.localIP());


  Serial.println("/");


  // ** End Adapted Code - This is the end of the code that was adapted from www.esplearning.com
}


void loop() {
  // put your main code here, to run repeatedly:
  WiFiClient client = server.available(); // Create a new client object for available connections


  int myTime = millis();
  //if(firstLoop){
  //  myTime = 0;
  //}
  if((myTime / 2000) % 3 == 0 && isLOOP){
    // Serial.println("0");
    digitalWrite(greenLED, HIGH);
    digitalWrite(yellowLED, LOW);
    digitalWrite(redLED, LOW);
    firstLoop = false;
  }
  if((myTime / 2000) % 3 == 1 && isLOOP){
    // Serial.println("1");
    digitalWrite(greenLED, LOW);
    digitalWrite(redLED, LOW);
    digitalWrite(yellowLED, HIGH);
    firstLoop = false;
  }
  if((myTime / 2000) % 3 == 2 && isLOOP){
    // Serial.println("2");
    digitalWrite(yellowLED, LOW);
    digitalWrite(redLED, HIGH);
    digitalWrite(greenLED, LOW);
    firstLoop = false;
  }

  if (client) { // If a client is connected, wait until it sends some data
    //while (!client.available()) { // If the client hasn't sent info, wait for it
    //  delay(10);
    //}

    String request = client.readStringUntil('\r'); // read the first line of the request
    Serial.println(request); // Echo the request to the Serial Monitor for debug

    client.flush(); // Wait until the buffers are clear

    if (request.indexOf("/loop") != -1) { // If the request is for the page "/led=on"
      isLOOP = true;
      firstLoop = true;
    }

    if (request.indexOf("/led=OFF") != -1) { // If the request is for the page "/led=off"
      isLOOP = false;
      digitalWrite(yellowLED, LOW);
      digitalWrite(redLED, LOW);
      digitalWrite(greenLED, LOW);
    }
    if (request.indexOf("/ledGREEN=ON") != -1) { // If the request is for the page "/led=off"
      isLOOP = false;
      digitalWrite(yellowLED, LOW);
      digitalWrite(redLED, LOW);
      digitalWrite(greenLED, HIGH);
    }
    if (request.indexOf("/ledYELLOW=ON") != -1) { // If the request is for the page "/led=off"
      isLOOP = false;
      digitalWrite(yellowLED, HIGH);
      digitalWrite(redLED, LOW);
      digitalWrite(greenLED, LOW);
    }
    if (request.indexOf("/ledRED=ON") != -1) { // If the request is for the page "/led=off"
      isLOOP = false;
      digitalWrite(yellowLED, LOW);
      digitalWrite(redLED, HIGH);
      digitalWrite(greenLED, LOW);
    }
    // ** End Adapted Code - This is the end of the code that was adapted from www.esplearning.com

    // Return the response

    client.println("HTTP/1.1 200 OK");
    client.println("Content-Type: text/html");
    client.println("");
    client.println("<!DOCTYPE HTML>");
    client.println("<html>");
    client.println("<head></head>");
```

```
    client.println("<body>");
    client.println("<h1>Stoplight Controller</h1>");
    client.println("<br>");
    client.println("<a href=\ledGREEN=ON><button>Green</button></a><br>");
    client.println("<a href=\ledYELLOW=ON><button>Yellow</button></a><br>");
    client.println("<a href=\ledRED=ON><button>Red</button></a><br>");
    client.println("<a href=\led=OFF><button>OFF</button></a><br>");
    client.println("<a href=\loop><button>Loop</button></a>");
    client.println("</body>");
    client.println("</html>");
  }

  //delay(100); // This introduces a little pause in each cycle. Probably helps save some power.

}
```