Lab 1 – Raspberry Pi

Online Link:

This lab is available as part of my online portfolio at: https://github.com/Brianhayden7/LED-Stoplight

Objective

The purpose of this lab is to learn the basic principles of the raspberry pi GPIO manipulation 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 python

Materials

I used the following materials to accomplish this lab:

- Personal computer
- Raspberry Pi
- 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

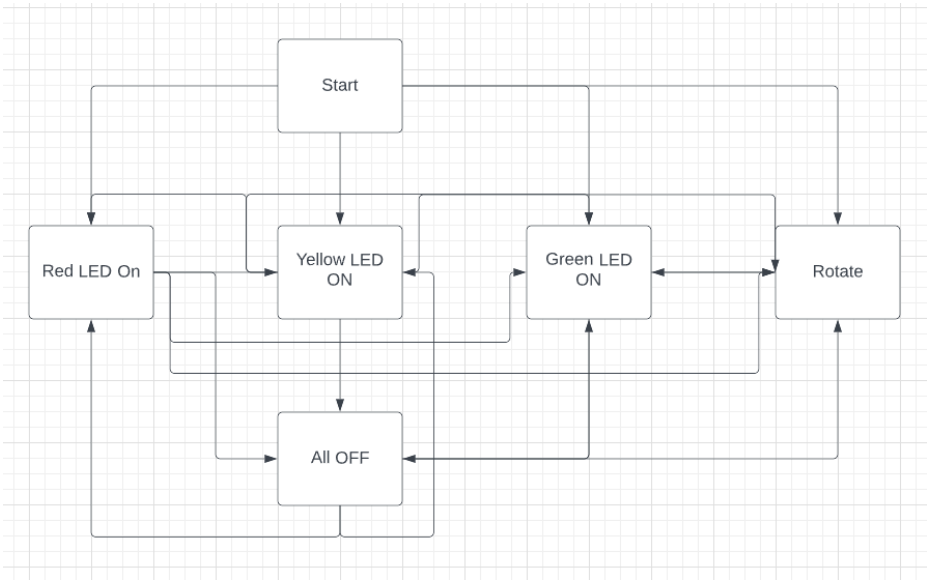
References

I used the following resources in this lab:

- https://docs.python.org/3/library/asyncio-task.html This is the asyncio library I spent a lot of time trying to get working but never did.
- https://stackoverflow.com/questions/40652793/how-to-kill-python-script-with-bash-script This is where I got my system command to kill a python script, specifically used to kill the loop script
- https://www.hackster.io/adhyoksh/controlling-gpio-pins-of-raspberry-pi-with-web-page-2d5bdc This is how I got my web to GPIO connection working as well as serve up a webpage with python
- https://stackoverflow.com/questions/28549641/run-multiple-python-scripts-concurrently This is where I got a bash script to concurrently run multiple python scripts.

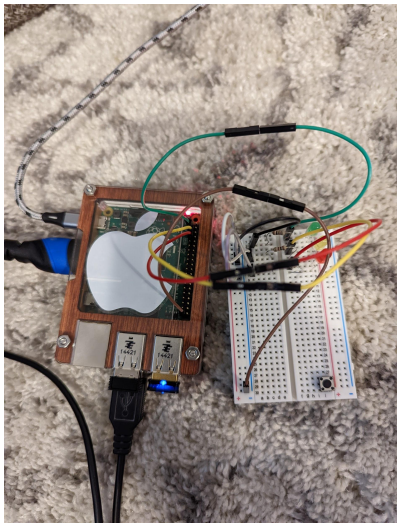
Procedures:

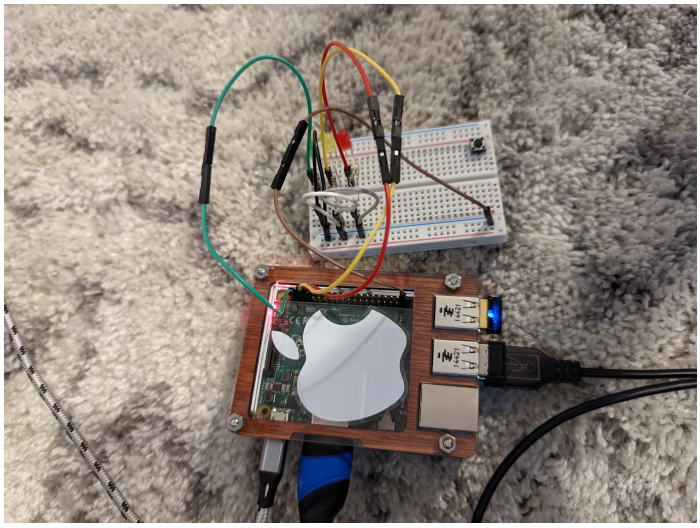
1. Download and install the raspberry pi raspbian flash tool to get an install of the OS onto your microSD card.
2. Get familiar with the GPIO pins, numbering methods, and how to find the ones you need.
3. Start coding up the python script. Using Flask you can serve up a web page in python and using routes you can essentially create functions for buttons.
4. 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. *Component Diagram* and *Schematic Diagram* – This system is represented by the photos below.





5. 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. You can test your code by calling "python3 python_code.py"
6. In order to get the rotating lights to work
   a. create a separate python script with the looping function in it
   b. in the rout for the loop button, use os.system() to call that script
   c. then in every other route, use os.system("pkill -f python.py") to stop that script from running.

Observations

I came into this lab with some GPIO experience with raspberry pi so I had an idea of how to make certain things work right away. I didn't know how to make a website work with these pins though so that was something new that I had to look up and research. It took some time to get all the different parts working together and also took a while to get the loop function to be interrupted while letting me interact with the website.

There were some issues I ran into trying to make a python script, asyncio, work but in the end had to use some system commands to just run and kill a separate python script. I wish I could learn more about the concurrency that library offers but maybe in another lab I can dive more into that.

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.

Overall I really enjoyed making all these things work together.

Thought Questions

What are some practical applications for a device similar to this? What enhancements do you think would make this more valuable?

This can obviously be used for some sort of traffic control. It's easy to see how this can be improved as well. Obviously you wouldn't want to manually control all the lights in most real world situations. The automatic looping is also useful only to some extent as there might be more traffic one way than others. This is where having some sort of other sensor to measure how busy the stoplight is would help improve performance.

How secure do you feel this device is? How can you improve its security posture?

This project currently is very insecure aside from the part that it is localized, meaning you have to be on the same network that is usually encrypted. For something like this that is actually being used, you would want to have it on a secure webpage on port 443 over https. On top of this you would want some sort of authentication to the system so that only authorized personnel can view and use it. You wouldn't want some bad actor to get a hold of such a system.

Estimate the time you spent on this lab (from start to finish)?

I probably spent a total of 5 hours on this lab. I got it to a pretty close working solution in about 2ish hours but then spent way too long trying to figure out how to use the asyncio library in python to run concurrent code to fulfill the looping method. That used up a lot of my time.

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: Python Code

(available at https://github.com/Brianhayden7/LED-Stoplight)

Main Function

```
from flask import Flask
from flask import render_template
import RPi.GPIO as rpi
import time
import os
import asyncio
import subprocess

app= Flask(__name__)

led1,led2,led3= 3,5,7

rpi.setwarnings(False)
rpi.setmode(rpi.BOARD)
rpi.setup(led1, rpi.OUT)
rpi.setup(led2, rpi.OUT)
rpi.setup(led3, rpi.OUT)
rpi.output(led1, 0)
rpi.output(led2, 0)
rpi.output(led3, 0)
looping = True
print("Done")

#loop = asyncio.new_event_loop()
#asyncio.set_event_loop(loop)

async def loopy():
```

```
    while looping:
        rpi.output(led1,0)
        rpi.output(led2,0)
        rpi.output(led3,1)
        time.sleep(1)
        rpi.output(led3,0)
        rpi.output(led1,1)
        time.sleep(1)
        rpi.output(led1,0)
        rpi.output(led2,1)
        time.sleep(1)




@app.route('/')
def index():
    os.system("pkill -f Stoplight.py")
    return render_template('webpage.html')

@app.route('/A')
def led1on():
    os.system("pkill -f Stoplight.py")
    looping = False
    rpi.output(led3,0)
    rpi.output(led2,0)
    rpi.output(led1,1)
    return render_template('webpage.html')

@app.route('/B')
def led2on():
    os.system("pkill -f Stoplight.py")
    looping = False
    rpi.output(led1,0)
    rpi.output(led3,0)
    rpi.output(led2,1)
    return render_template('webpage.html')

@app.route('/E')
def led2off():
    looping = True
    subprocess.Popen(["/home/briguy97/Desktop/CODE/stop.sh"])
    #asyncio.run(loopy())
    #asyncio.ensure_future(loopy())
    #loop.run_forever()
    return render_template('webpage.html')

@app.route('/C')
def led3on():
    os.system("pkill -f Stoplight.py")
    looping = False
    rpi.output(led1,0)
```

```
    rpi.output(led2,0)
    rpi.output(led3,1)
    return render_template('webpage.html')

@app.route('/D')
def led3off():
    os.system("pkill -f Stoplight.py")
    #looping = False
    #loop.stop()
    rpi.output(led1,0)
    rpi.output(led2,0)
    rpi.output(led3,0)
    return render_template('webpage.html')

if __name__=="__main__":
    print("Start")

    app.run(debug=True, host='192.168.33.162')
```

Loop Script

```
from flask import Flask
from flask import render_template
import RPi.GPIO as rpi
import time
import os
import asyncio
import subprocess

app= Flask(__name__)

led1,led2,led3= 3,5,7

rpi.setwarnings(False)
rpi.setmode(rpi.BOARD)
rpi.setup(led1, rpi.OUT)
rpi.setup(led2, rpi.OUT)
rpi.setup(led3, rpi.OUT)
rpi.output(led1, 0)
rpi.output(led2, 0)
rpi.output(led3, 0)
looping = True
print("Done")

while True:
    rpi.output(led1,0)
    rpi.output(led2,0)
    rpi.output(led3,1)
    time.sleep(1)
    rpi.output(led3,0)
    rpi.output(led1,1)
    time.sleep(1)
```

```
    rpi.output(led1,0)
    rpi.output(led2,1)
    time.sleep(1)
```

Bash Script

```
#!/bin/bash
python3 Stoplight.py &
```

Web Page

```
<html>
  <head>
    <title>Rpi Stoplight</title>
  </head>
  <body>
  <h1>Stoplight Controller</h1>
  <br>
  <a href=\A><button>Green</button></a><br>
  <a href=\B><button>Yellow</button></a><br>
  <a href=\C><button>Red</button></a><br>
  <a href=\D><button>OFF</button></a><br>
  <a href=\E><button>Loop</button></a>
  </body>
</html>
```