Issa Odeh
CPE 406
Assignment # 5
04/1/2021

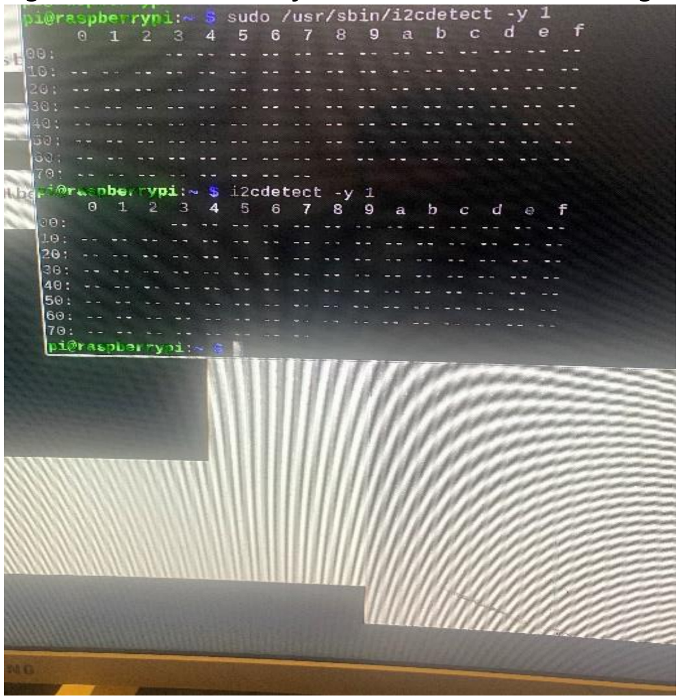
Assignment Description:

This assignment was about how to use ADC and PWM. Using these methods, we will learn how to control the brightness of an LED by using a potentiometer. We created a circuit that consists of one Potentiometer, an ADS7830, 1 LED, and a 220k ohm resistor. To use the ADS in code, the "from ADCDevice import *" was used so the code understands what the device is, Inside the code, we had to detect the ADS by using the address "0x4b". For the code to work, we must read the ADC value of potentiometer and map it to the duty cycle of PWM to control the brightness of the LED. In the code, we calculated the voltage by using the function "voltage = value / 255.0 * 3.3". To map PWM duty cycle, the function "p.ChangeDutyCycle(value*100/255)" was used. Both functions must be in a def loop.

Problems Encountered:

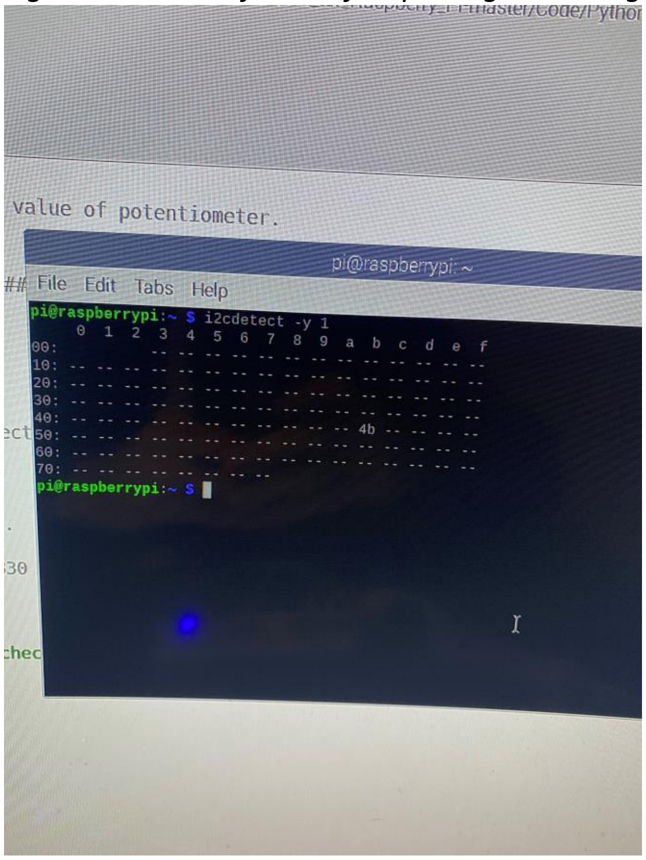
I had a couple of problems in this assignment. The first one was my raspberry pi was not detecting the address of the i2c address which I did not know how to fix. I brought it up in class to ask for help and a helpful classmate told me he was having the same issue and that I needed to double check my circuit wiring. And he was correct, 1 wire was put into the right locating. After I put the wire in the right place, my raspberry pi recognized my i2c address and the code worked and ran. My second problem was finding my ADC device. I had to go back to the starter kit I purchased and made sure the device even came with it. Luckily, it did but I lost it because I could not find it anywhere, but I had a classmate who let me barrow theirs for couple days till I finished the assignment. The worst part was when I was cleaning up my choir and I found the ADC device, so I did not even need to barrow one, I just did not look hard enough.

Fig 1: I2c address not found. Where I was having troubles with.



```
pi@raspberrypi:~ $ sudo /usr/sbin/i2cdetect -y 1
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:          -- -- -- -- -- -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
pi@raspberrypi:~ $ i2cdetect -y 1
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:          -- -- -- -- -- -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
pi@raspberrypi:~ $
```

Fig 2: i2c address found after placing wire in right place. (0x4b)



```
pi@raspberrypi:~ $ i2cdetect -y 1
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:          -- -- -- -- -- -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- -- -- -- 4b -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
pi@raspberrypi:~ $
```

Lessons Learned:

I enhanced my knowledge in building more circuits. I started off the semester needing to look at the picture of the circuit. Now I only use the diagrams to connect everything as it is better to know how to build a circuit without just copying the picture. I now how to make a code on how to adjust the voltage with the correct formulas. Also, mapping will be extremely helpful method to remember for the future. I have a better understanding of a duty cycle and how each percentage works.

Description of Completed Lab:

As we see a below, I have a picture of circuit and a video link of me testing the code and circuit to see if I can control the brightness of the LED. While running the code, if the potentiometer is all the way to the right then the code should read 0 volts. If the potentiometer is moved counterclockwise the LED should turn on and the code should detect that there is more voltage being applied when the meter is moved. The maximum voltage should only go up to 3.3v as we have it connected to 3.3v and It can't go down more than 0.

Fig 3: Circuit I constructed.

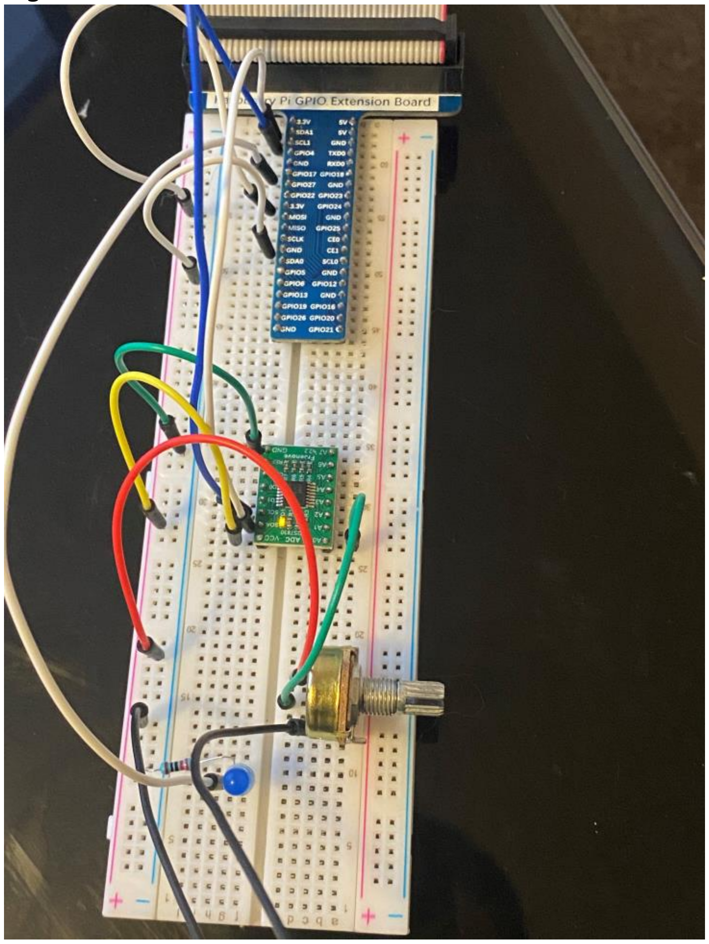
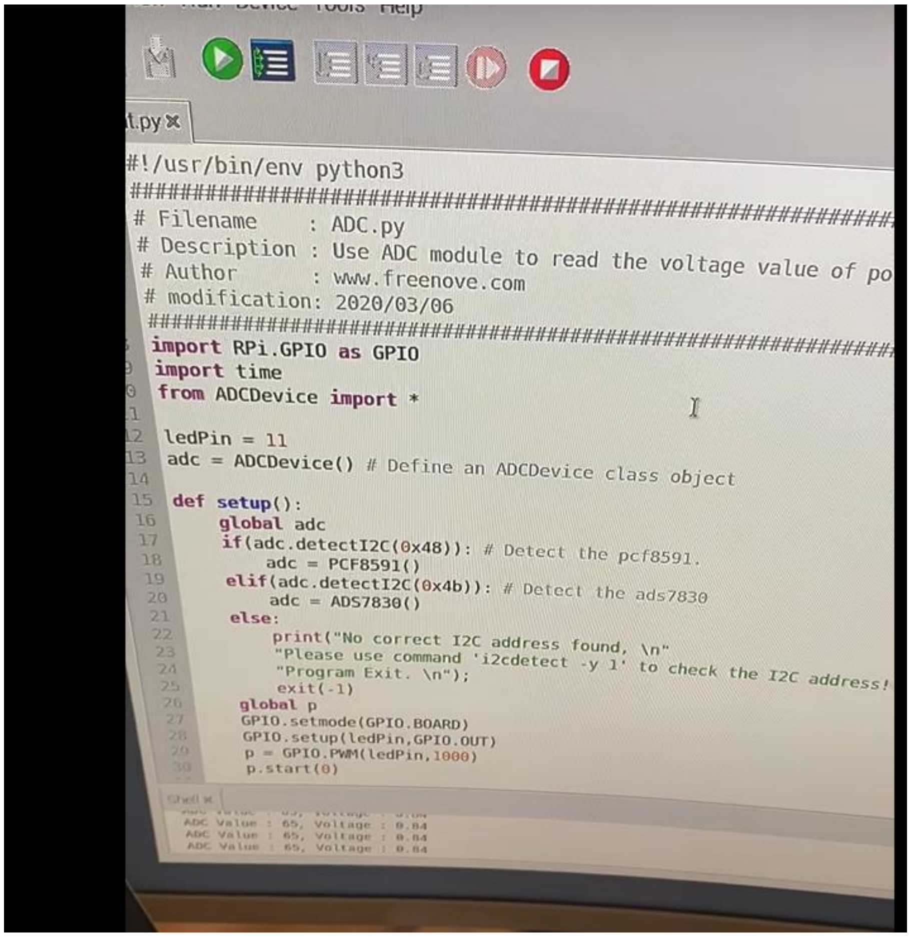


Fig 4: Part 1 of code I used. Helped get it from freenove.



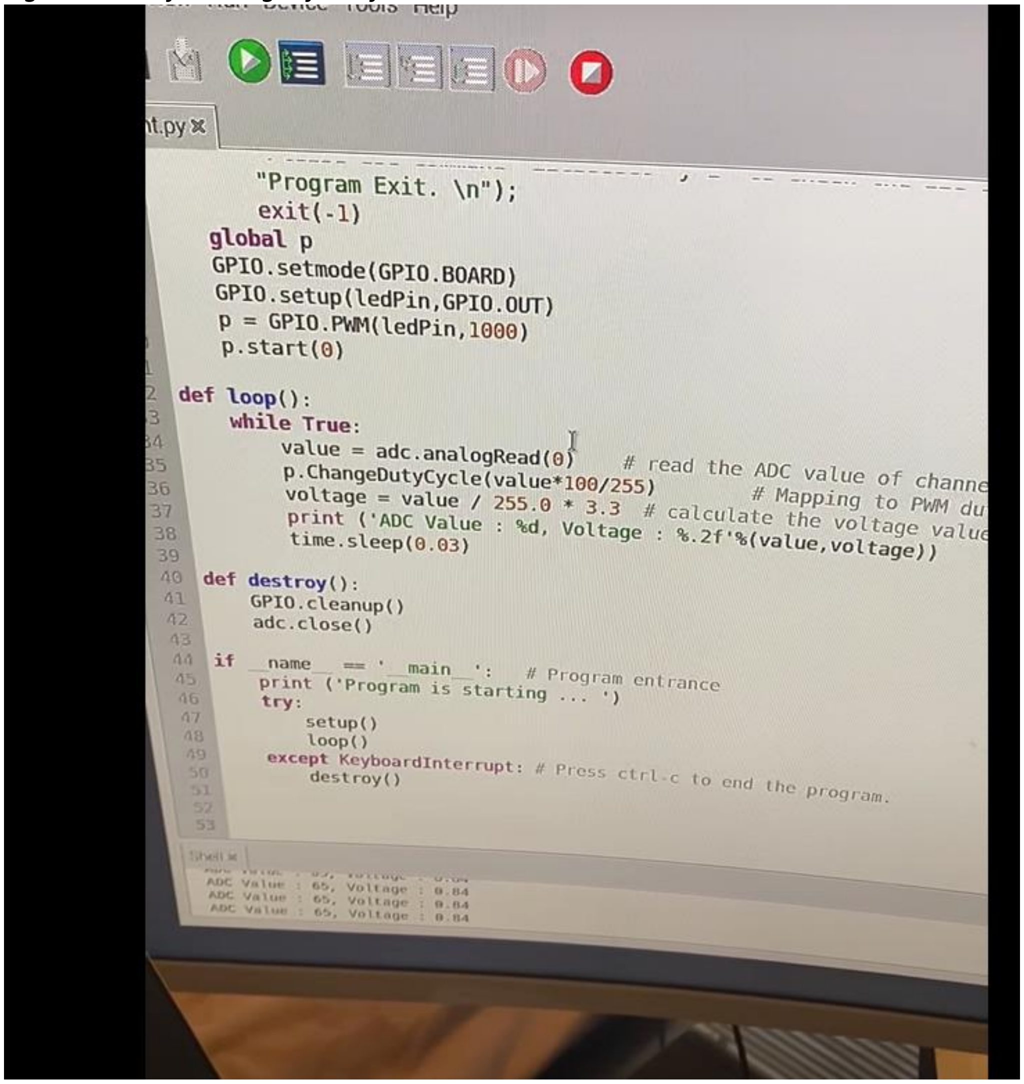
```
#!/usr/bin/env python3
########################################################################
# Filename    : ADC.py
# Description : Use ADC module to read the voltage value of po
# Author      : www.freenove.com
# modification: 2020/03/06
########################################################################
import RPi.GPIO as GPIO
import time
from ADCDevice import *

ledPin = 11
adc = ADCDevice() # Define an ADCDevice class object

def setup():
    global adc
    if(adc.detectI2C(0x48)): # Detect the pcf8591.
        adc = PCF8591()
    elif(adc.detectI2C(0x4b)): # Detect the ads7830
        adc = ADS7830()
    else:
        print("No correct I2C address found, \n"
        "Please use command 'i2cdetect -y 1' to check the I2C address!
        "Program Exit. \n");
        exit(-1)
    global p
    GPIO.setmode(GPIO.BOARD)
    GPIO.setup(ledPin,GPIO.OUT)
    p = GPIO.PWM(ledPin,1000)
    p.start(0)
```

```
ADC Value : 65, Voltage : 0.84
ADC Value : 65, Voltage : 0.84
ADC Value : 65, Voltage : 0.84
```

Fig 5: Part 2 of code I got from freenove



```
        "Program Exit. \n");
        exit(-1)
    global p
    GPIO.setmode(GPIO.BOARD)
    GPIO.setup(ledPin,GPIO.OUT)
    p = GPIO.PWM(ledPin,1000)
    p.start(0)

def loop():
    while True:
        value = adc.analogRead(0)    # read the ADC value of channe
        p.ChangeDutyCycle(value*100/255)        # Mapping to PWM du
        voltage = value / 255.0 * 3.3  # calculate the voltage value
        print ('ADC Value : %d, Voltage : %.2f'%(value,voltage))
        time.sleep(0.03)

def destroy():
    GPIO.cleanup()
    adc.close()

if __name__ == '__main__':   # Program entrance
    print ('Program is starting ... ')
    try:
        setup()
        loop()
    except KeyboardInterrupt: # Press ctrl-c to end the program.
        destroy()
```

```
ADC Value : 65, Voltage : 0.84
ADC Value : 65, Voltage : 0.84
ADC Value : 65, Voltage : 0.84
```