Kennedy Anukam
CPE 406
Assignment 6
April 12, 2021

Assignment Description:

In the first portion of the lab, I will complete project 9.1 described in the Freenove tutorial. In this portion I will wire a circuit that includes a photoresistor and an ADC. With the values read by the ADC, the code will translate that to a duty cycle to control the brightness of the LED. In this section, I hope to learn how a photoresistor can be used with an ADC to control the brightness of an LED.

Building off this portion, I will change the code to reverse the circuit operation. What this means is that the LED should become dimmer when there is a higher ambient light. I hope to learn how the ADC value read can be altered to achieve the modified functionality.

The next portion of the same project section is modifying the circuit to achieve the same functionality as the modified code described in the previous section. With the modified circuit, the LED should become dimmer when there is a higher ambient light. The code will be the original code and not the modified code. I hope to learn how the circuit can be changed to achieve this functionality.

The next part of this lab assignment is wiring up Project 10.1 described in the Freenove tutorial. In this portion I will wire a circuit that includes a thermistor and an ADC. With the wired circuit, the code should display the ADC value, temperature, and voltage in the terminal output. I hope to learn in more depth how a thermistor works.

Building off the previous part is altering the code to read the ADC value of the thermistor and mapping it to a duty cycle to control the brightness of an LED. I hope to learn how the ADC value can be altered in the code to make the temperature difference more discrete for the LED mapping.

Problems Encountered:

In this lab, a problem I encountered was a slight mistake in circuit wiring for project 9.1. I forgot to connect the ADC to ground initially and that was causing I2C to not be detected. I fixed this error by inspecting my circuit to identify that it was not wired correctly.

Lessons Learned:

A lesson I learned from this lab is how a photoresistor. works. I learned that a potentiometer decreases resistance when receiving light on its surface. I learned how an ADC can be utilized with a photoresistor to control the brightness of an LED.

Another lesson that I learned from this lab is how a thermistor works. I learned that it works by changing its resistance with a change in temperature. With this property of a thermistor, the exact temperature can be identified.

Description of Completed Lab:

Project 9.1

In this part of the lab, I wired the circuit and ran the code to control the brightness of the LED with a photoresistor. An ADC is used to achieve this.

Circuit Wiring

Figure 1 shows the snippet of the circuit containing the LED.

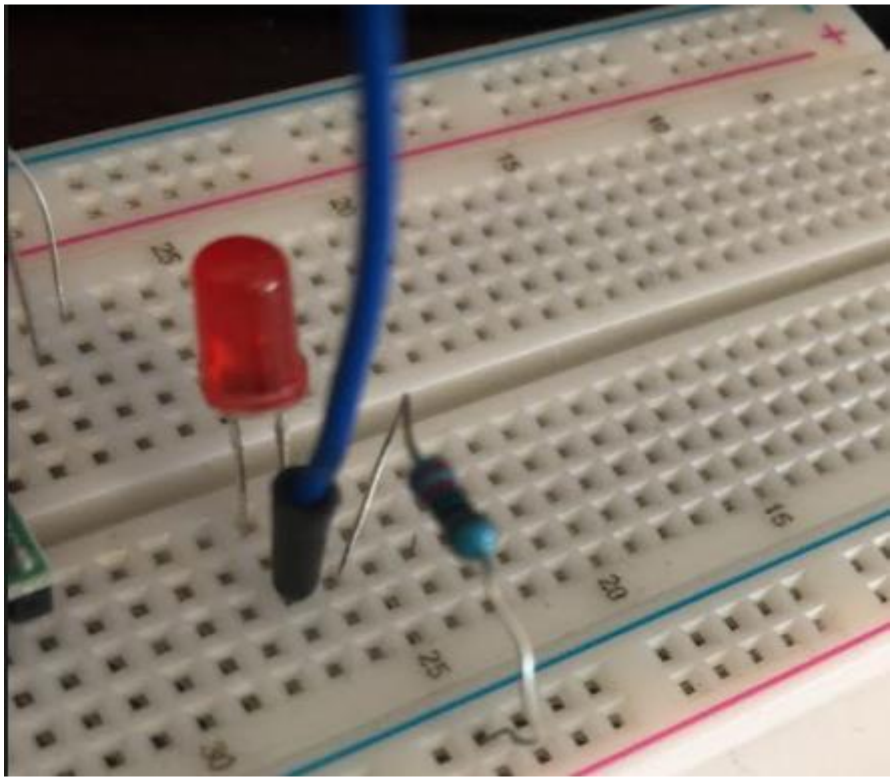

Figure 1: A subsection of the circuit containg the LED. A 220 Ohm resistor is utilized alongside the LED.

Figure 2 shows a subsection of the circuit containg the ADC and the photoresistor.

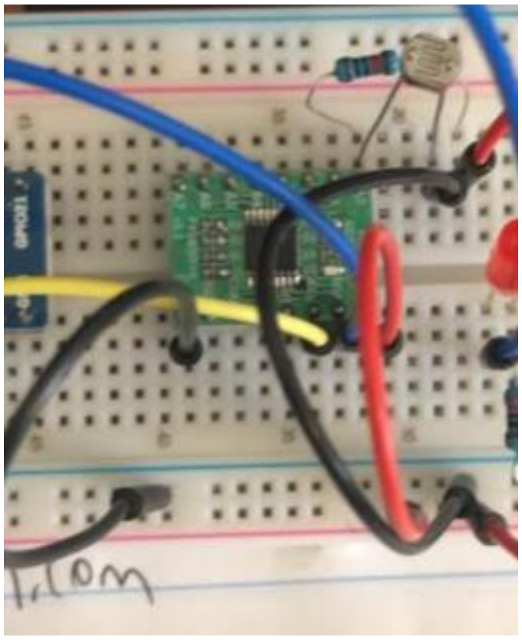

Figure 2: A subsection of the circuit containg the photoresistor and ADC. A 10k Ohm resistor is shown alongside the other components..

Figure 3 shows the entire circuit for Project 9.1 of the tutorial.

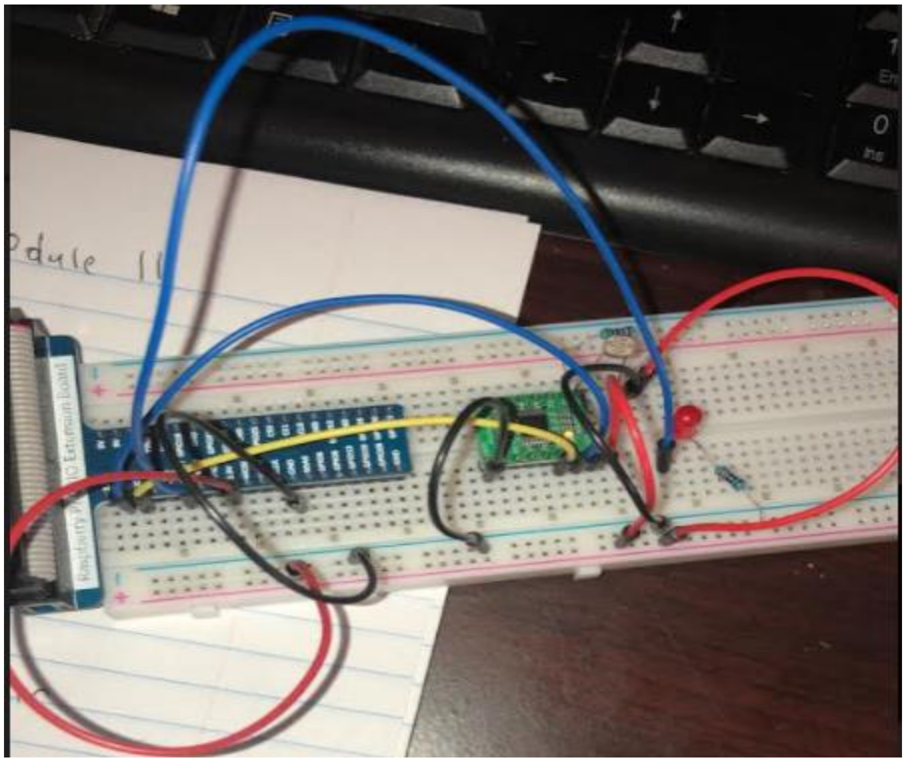

Figure 3: The circuit described in Project 9.1 of the guide. A photoresistor and ADC are utilized to control the brightness of an LED.

Code & Results

```
#!/usr/bin/env python3
#############################################################################
# Filename    : Nightlamp.py
# Description : Control LED with Photoresistor
# Author      : www.freenove.com
# modification: 2020/03/09
########################################################################
# modified by K. Anukam rev 1.2 2021/03/12
import RPi.GPIO as GPIO
import time
from ADCDevice import *

ledPin = 11  # define ledPin
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!
\n"
              "Program Exit. \n");
        exit(-1)
    global p
    GPIO.setmode(GPIO.BOARD)
    GPIO.setup(ledPin, GPIO.OUT)  # set ledPin to OUTPUT mode
    GPIO.output(ledPin, GPIO.LOW)

    p = GPIO.PWM(ledPin, 1000)  # set PWM Frequence to 1kHz
    p.start(0)


def loop():
    print("Kennedy Anukam HW #06")
    while True:
        value = adc.analogRead(0)  # read the ADC value of channel 0
        p.ChangeDutyCycle(value * 100 / 255)
        voltage = value / 255.0 * 3.3
        print('ADC Value : %d, Voltage : %.2f' % (value, voltage))
        time.sleep(0.01)


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


if __name__ == '__main__':  # Program entrance
```

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

Video

Project 9.1 (Modified Code)

In this part of the lab, I modified the code to make the LED become dimmer when there is a higher ambient light. I achieved this by making a slight modification to the code:

```
while True:
value = adc.analogRead(0)    # read the ADC value of channel 0
# Modification, subtraction
value = 255 - value
p.ChangeDutyCycle(value*100/255)
voltage = value / 255.0 * 3.3
print ('ADC Value : %d, Voltage : %.2f'%(value,voltage))
time.sleep(0.01)
```

The modification that was made was changing the value of the ADC value read to be 255 – the value. This achieved the desired functionality of the light becoming dimmer upon receiving a higher ambient light.

Video

Project 9.1 (Modified Circuit)

In this part of the lab, I modified the circuit to achieve the same functionality as described with the modified code. That is, the LED become dimmer when there is a higher ambient light. The original code was used for this portion.

Circuit Wiring

Figure 4 shows the modified circuit constructed.

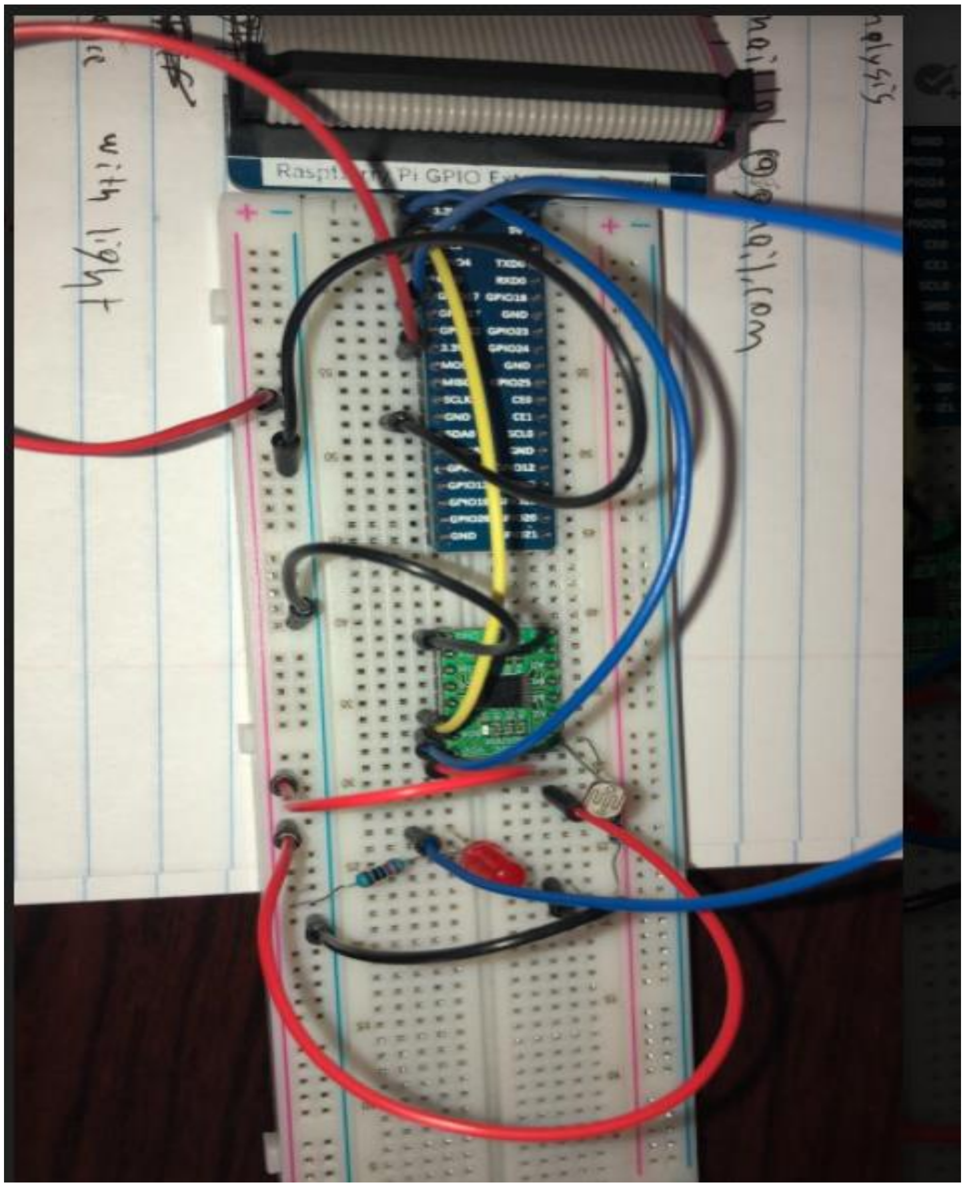

Figure 4: The modified circuit to achieve the functionality of the LED becoming brighter with a higher ambient light.

Code & Results

```
#!/usr/bin/env python3
#############################################################################
# Filename    : Nightlamp.py
# Description : Control LED with Photoresistor
# Author      : www.freenove.com
# modification: 2020/03/09
########################################################################
# modified by K. Anukam rev 1.2 2021/03/12
import RPi.GPIO as GPIO
import time
from ADCDevice import *

ledPin = 11  # define ledPin
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!
\n"
              "Program Exit. \n");
        exit(-1)
    global p
    GPIO.setmode(GPIO.BOARD)
    GPIO.setup(ledPin, GPIO.OUT)  # set ledPin to OUTPUT mode
    GPIO.output(ledPin, GPIO.LOW)

    p = GPIO.PWM(ledPin, 1000)  # set PWM Frequence to 1kHz
    p.start(0)


def loop():
    print("Kennedy Anukam HW #06")
    while True:
        value = adc.analogRead(0)  # read the ADC value of channel 0
        p.ChangeDutyCycle(value * 100 / 255)
        voltage = value / 255.0 * 3.3
        print('ADC Value : %d, Voltage : %.2f' % (value, voltage))
        time.sleep(0.01)


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


if __name__ == '__main__':  # Program entrance
```

```
    print('Program is starting ... ')
    setup()
    try:
        loop()
    except KeyboardInterrupt:  # Press ctrl-c to end the program.
#!/usr/bin/env python3
#############################################################################
# Filename    : Nightlamp.py
# Description : Control LED with Photoresistor
# Author      : www.freenove.com
# modification: 2020/03/09
########################################################################
# modified by K. Anukam rev 1.2 2021/03/12
import RPi.GPIO as GPIO
import time
from ADCDevice import *

ledPin = 11  # define ledPin
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!
\n"
              "Program Exit. \n");
        exit(-1)
    global p
    GPIO.setmode(GPIO.BOARD)
    GPIO.setup(ledPin, GPIO.OUT)  # set ledPin to OUTPUT mode
    GPIO.output(ledPin, GPIO.LOW)

    p = GPIO.PWM(ledPin, 1000)  # set PWM Frequence to 1kHz
    p.start(0)


def loop():
    print("Kennedy Anukam HW #06")
    while True:
        value = adc.analogRead(0)  # read the ADC value of channel 0
        p.ChangeDutyCycle(value * 100 / 255)
        voltage = value / 255.0 * 3.3
        print('ADC Value : %d, Voltage : %.2f' % (value, voltage))
        time.sleep(0.01)


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

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

Video

Project 10.1

In this part of the lab, I wired the circuit described in Project 10.1 to read the temperature values with an ADC and thermistor.

Circuit Wiring

Figure 5 shows the circuit wiring for the entire circuit.

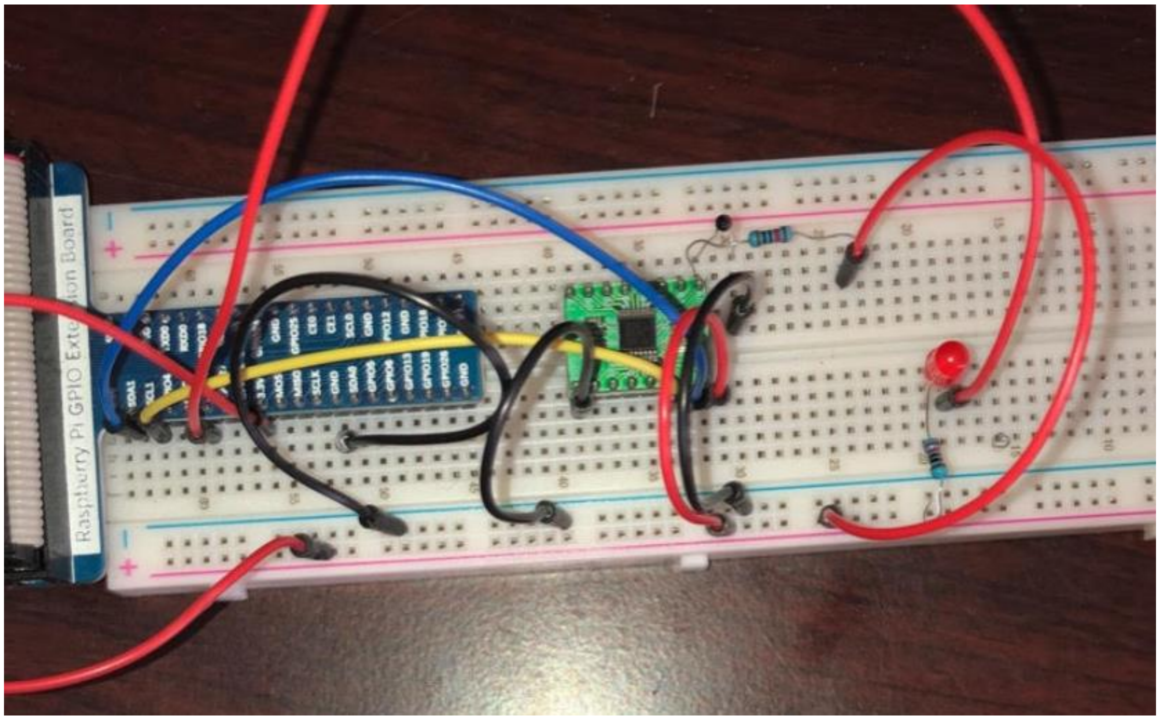

Figure 5: The complete circuit utilizing a thermistor, ADC, LED, and resistors.

Code & Results

```
#!/usr/bin/env python3
#############################################################################
# Filename    : Thermometer.py
# Description : DIY Thermometer
# Author      : www.freenove.com
# modification: 2019/03/09
########################################################################
# modified by Kennedy Anukam rev 1.1 2020/05/12
import RPi.GPIO as GPIO
import time
import math
from ADCDevice import *

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!
\n"
              "Program Exit. \n");
        exit(-1)


def loop():
    print("Kennedy Anukam HW06")
    while True:
        value = adc.analogRead(0)  # read ADC value A0 pin
        voltage = value / 255.0 * 3.3  # calculate voltage
        Rt = 10 * voltage / (3.3 - voltage)  # calculate resistance value of
thermistor
        tempK = 1 / (1 / (273.15 + 25) + math.log(Rt / 10) / 3950.0)  #
calculate temperature (Kelvin)
        tempC = tempK - 273.15  # calculate temperature (Celsius)
        print('ADC Value : %d, Voltage : %.2f, Temperature : %.2f' % (value,
voltage, tempC))
        time.sleep(0.01)


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


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

Video

Project 10.1 (Modified Code)

In this part of the lab, I changed the code to read the ADC value of the thermistor to map it to a duty cycle to control the brightness of the LED. I used ranges of ADC values to change the values to make the differences in temperature more noticeable for the LED.

Code & Results

```
#!/usr/bin/env python3
#############################################################################
# Filename    : Thermometer.py
# Description : DIY Thermometer
# Author      : www.freenove.com
# modification: 2019/03/09
########################################################################
# modified by Kennedy Anukam rev 1.2 2020/05/12
import RPi.GPIO as GPIO
import time
import math
from ADCDevice import *

adc = ADCDevice()  # Define an ADCDevice class object
ledPin = 17


def setup():
    global adc
    global p
    GPIO.setmode(GPIO.BCM)
    GPIO.setup(ledPin, GPIO.OUT)
    p = GPIO.PWM(ledPin, 1000)
    p.start(0)
    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!
\n"
              "Program Exit. \n");
        exit(-1)


def loop():
    print("Kennedy Anukam HW06")
    while True:
        value = adc.analogRead(0)  # read ADC value A0 pin
        voltage = value / 255.0 * 3.3  # calculate voltage
        Rt = 10 * voltage / (3.3 - voltage)  # calculate resistance value of
thermistor
```

```
        tempK = 1 / (1 / (273.15 + 25) + math.log(Rt / 10) / 3950.0)  # 
calculate temperature (Kelvin)
        tempC = tempK - 273.15  # calculate temperature (Celsius)
        print('ADC Value : %d, Voltage : %.2f, Temperature : %.2f' % (value, 
voltage, tempC))
        if value < 100:
            value = 40
        elif value in range(100, 110):
            value = 80
        elif value in range(110, 120):
            value = 120
        elif value in range(120, 130):
            value = 160
        elif value in range(130, 140):
            value = 200
        else:
            value = 255
        p.ChangeDutyCycle(value * 100 / 255)  # Mapping to PWM duty cycle
        time.sleep(0.03)


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


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

Video