MIT AI2 204

IoT with MIT App Inventor

Fundamental

Xincheng Tang

Sending Temperature and Humidity To your android device

Prepare the environment:

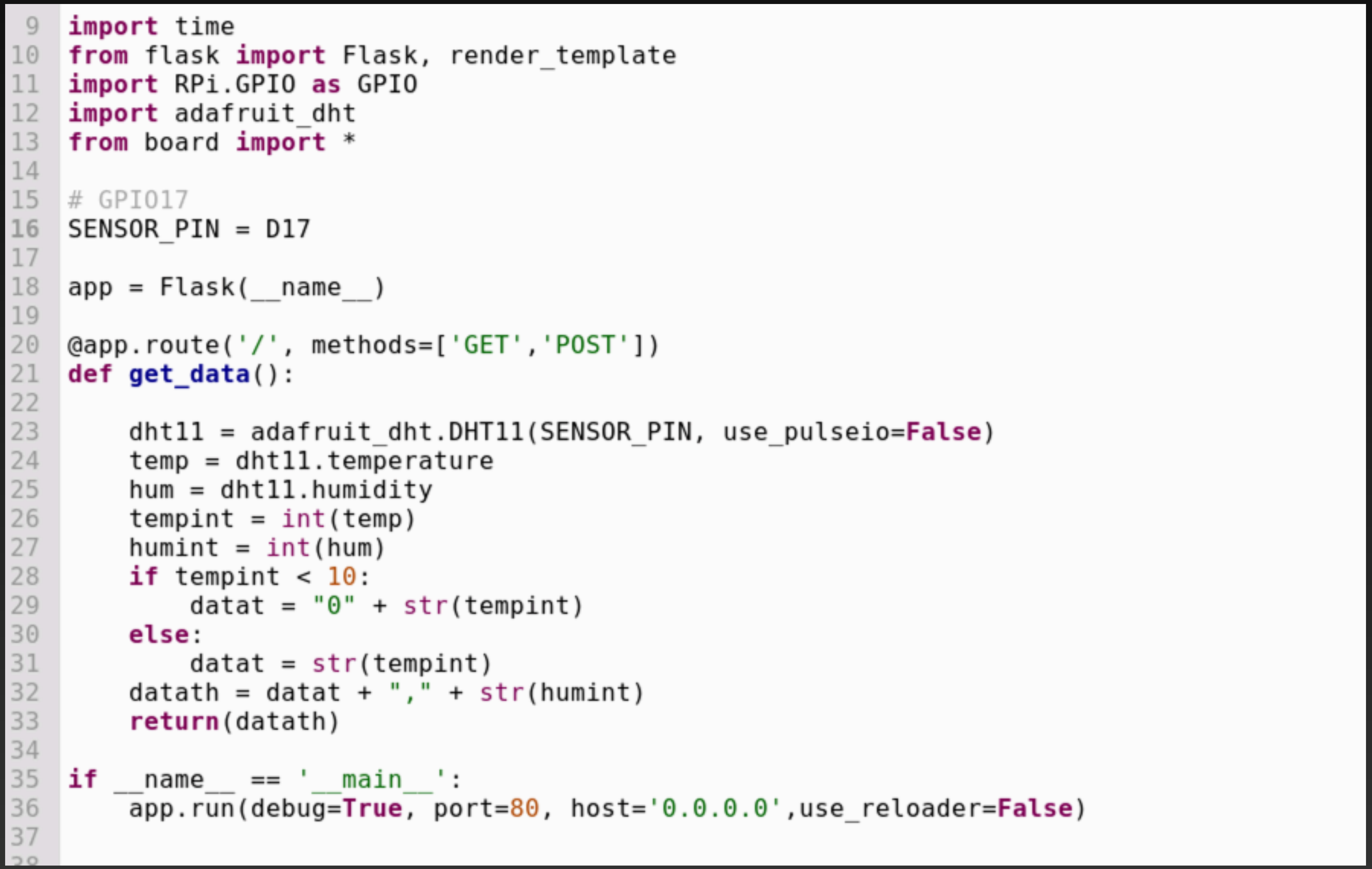
```
sudo pip3 install adafruit-circuitpython-dht
```

There is new updates to the library:

https://github.com/adafruit/Adafruit_CircuitPython_DHT

Sending Temperature and Humidity To your android device

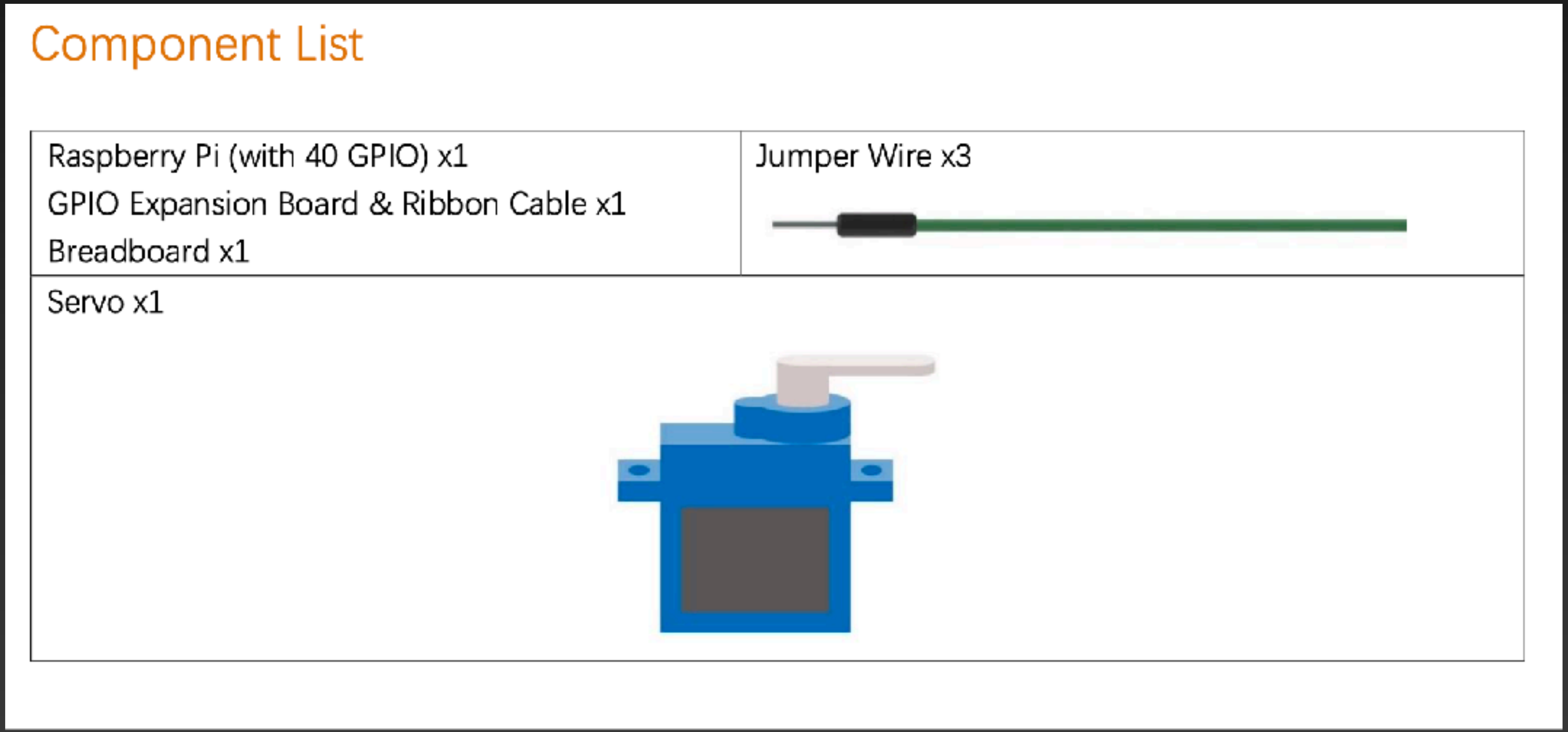
Write the pythons control program



```python
import time
from flask import Flask, render_template
import RPi.GPIO as GPIO
import adafruit_dht
from board import *

# GPIO17
SENSOR_PIN = D17

app = Flask(__name__)

@app.route('/', methods=['GET','POST'])
def get_data():

    dht11 = adafruit_dht.DHT11(SENSOR_PIN, use_pulseio=False)
    temp = dht11.temperature
    hum = dht11.humidity
    tempint = int(temp)
    humint = int(hum)
    if tempint < 10:
        datat = "0" + str(tempint)
    else:
        datat = str(tempint)
    datath = datat + "," + str(humint)
    return(datath)

if __name__ == '__main__':
    app.run(debug=True, port=80, host='0.0.0.0',use_reloader=False)
```

Servo Motor Control

We will learn how to make motor rotate.

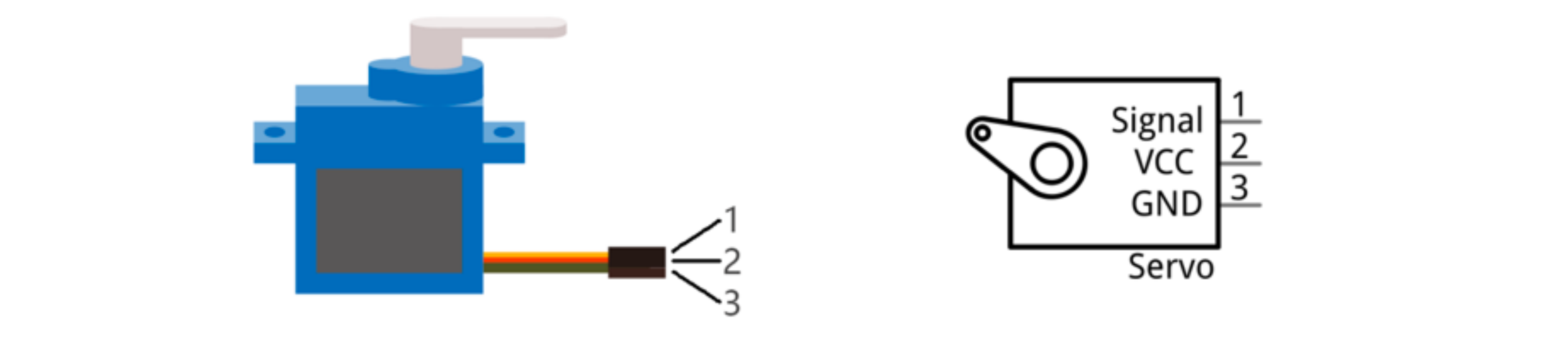


Component List

Raspberry Pi (with 40 GPIO) x1 GPIO Expansion Board & Ribbon Cable x1 Breadboard x1	Jumper Wire x3
Servo x1	

Servo Motor Control

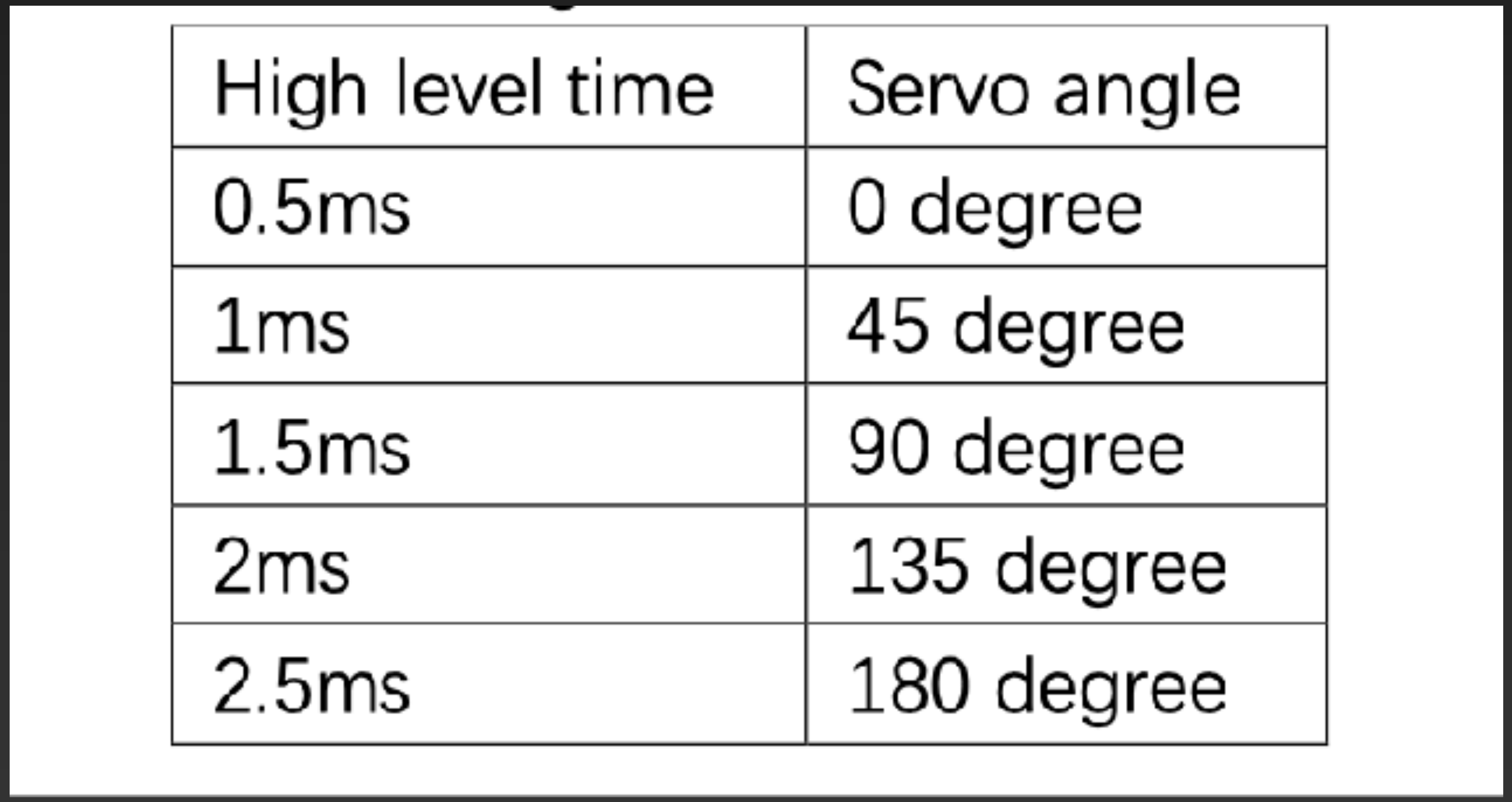
Servo is a compact package which consists of a DC Motor, a set of reduction gears to provide torque, a sensor and control circuit board. Most Servos only have a 180-degree range of motion via their "horn". Servos can output higher torque than a simple DC Motor alone and they are widely used to control motion in model cars, model airplanes, robots, etc. Servos have three wire leads which usually terminate to a male or female 3-pin plug. Two leads are for electric power: Positive (2-VCC, Red wire), Negative (3-GND, Brown wire), and the signal line (1-Signal, Orange wire) as represented in the Servo provided in your Kit.



Servo Motor Control

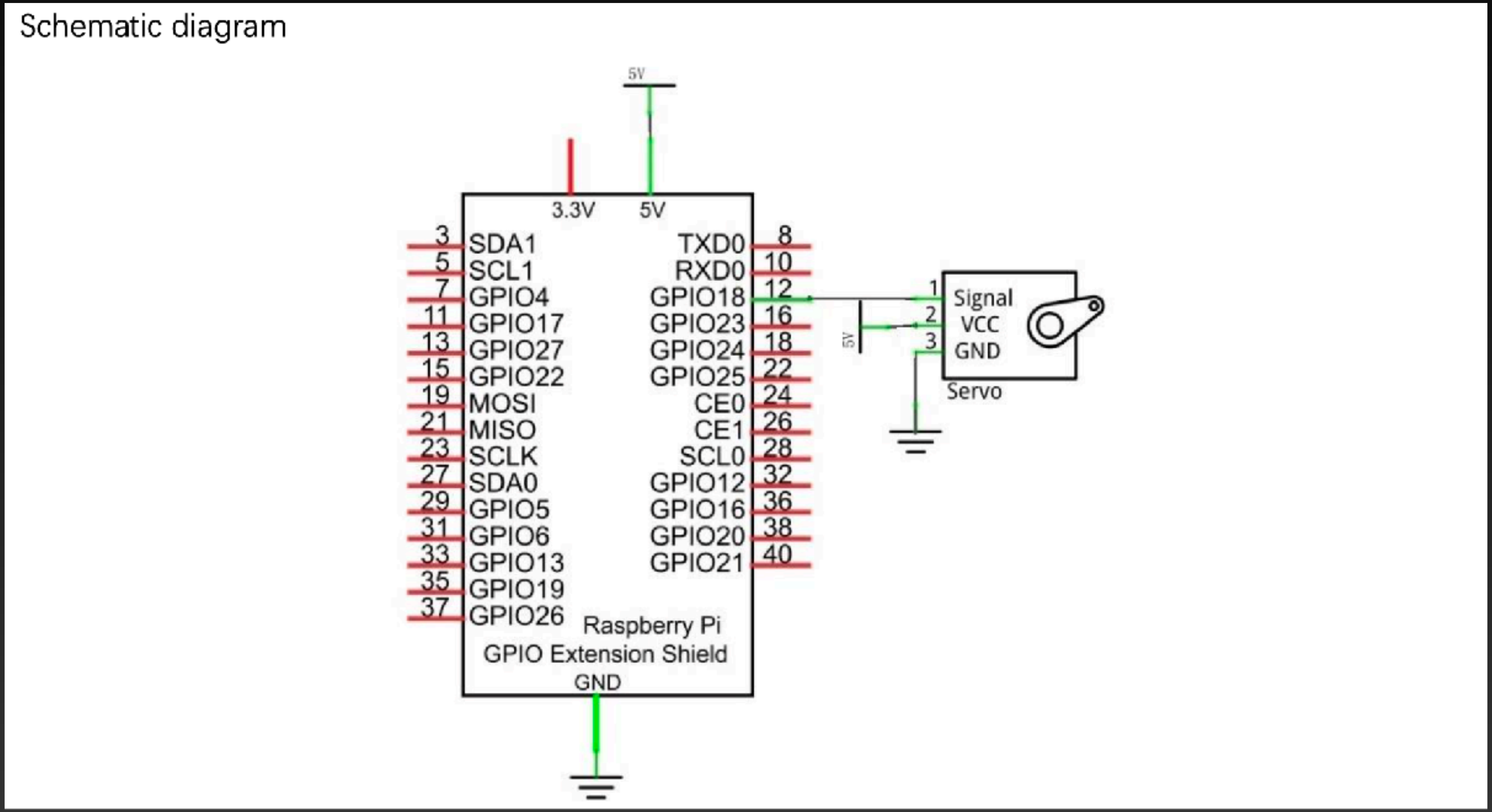
We will use a 50Hz PWM signal with a duty cycle in a certain range to drive the Servo. The lasting time 0.5ms- 2.5ms of PWM single cycle high level corresponds to the Servo angle 0 degrees - 180 degree linearly. Part of the corresponding values are as follows:

Note: the lasting time of high level corresponding to the servo angle is absolute instead of accumulating. For example, the high level time lasting for 0.5ms correspond to the 0 degree of the servo. If the high level time lasts for another 1ms, the servo rotates to 45 degrees.

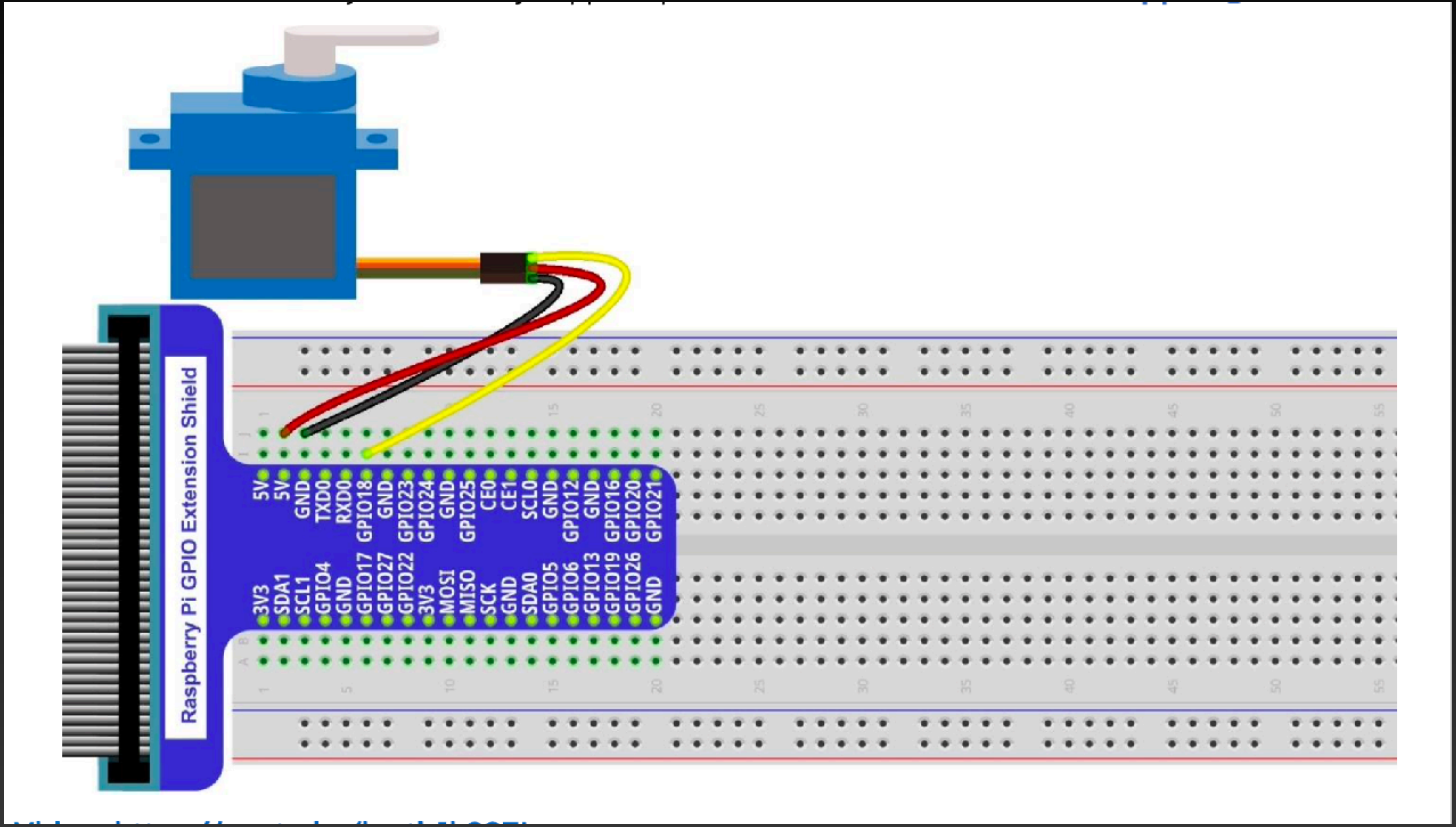


High level time	Servo angle
0.5ms	0 degree
1ms	45 degree
1.5ms	90 degree
2ms	135 degree
2.5ms	180 degree

Servo Motor Control



Servo Motor Control



Servo Motor Control

```
import RPi.GPIO as GPIO
import time

OFFSE_DUTY = 0.5  #define pulse offset of servo
SERVO_MIN_DUTY = 2.5 + OFFSE_DUTY  #define the minimum angle of servo
SERVO_MAX_DUTY = 12.5 + OFFSE_DUTY #define the maximum angle of servo
servoPin = 12

def map(value, fromLow, fromHigh, toLow, toHigh):
    return (toHigh-toLow)*(value-fromLow)/(fromHigh-fromLow) + toLow
```

Servo Motor Control

```
def setup():
    global p
    GPIO.setmode(GPIO.BOARD)   #use PHYSICAL GPIO numbering
    GPIO.setup(servoPin, GPIO.OUT) #set servoPin to OUTPUT mode
    GPIO.setup(servoPin, GPIO.LOW) #make servoPin output to LOW level

    p = GPIO.PWM(servoPin, 50) #set Freq to 50Hz
    p.start(0)                 #set initial duty cycle to 0
```

Servo Motor Control

```
def servoWrite(angle):

    if(angle<0):

        angle = 0

    elif(angle > 180):

        angle = 180

    p.ChangeDutyCycle(map(angle,0,180,SERVO_MIN_DUTY,SERVO_MAX_DUTY)) #map the
angle to duty cycle and output it
```

Servo Motor Control

```
def loop():
    while True:
        for dc in range(0, 181, 1):  # make servo rotate from 0 to 180
            servoWrite(dc)   #write dc value to servo
            time.sleep(0.001)
        time.sleep(0.5)

        for dc in range(180, -1, -1): # make the servo rotate from 180 to 0 deg
            servoWrite(dc)
            time.sleep(0.001)
        time.sleep(0.5)
```

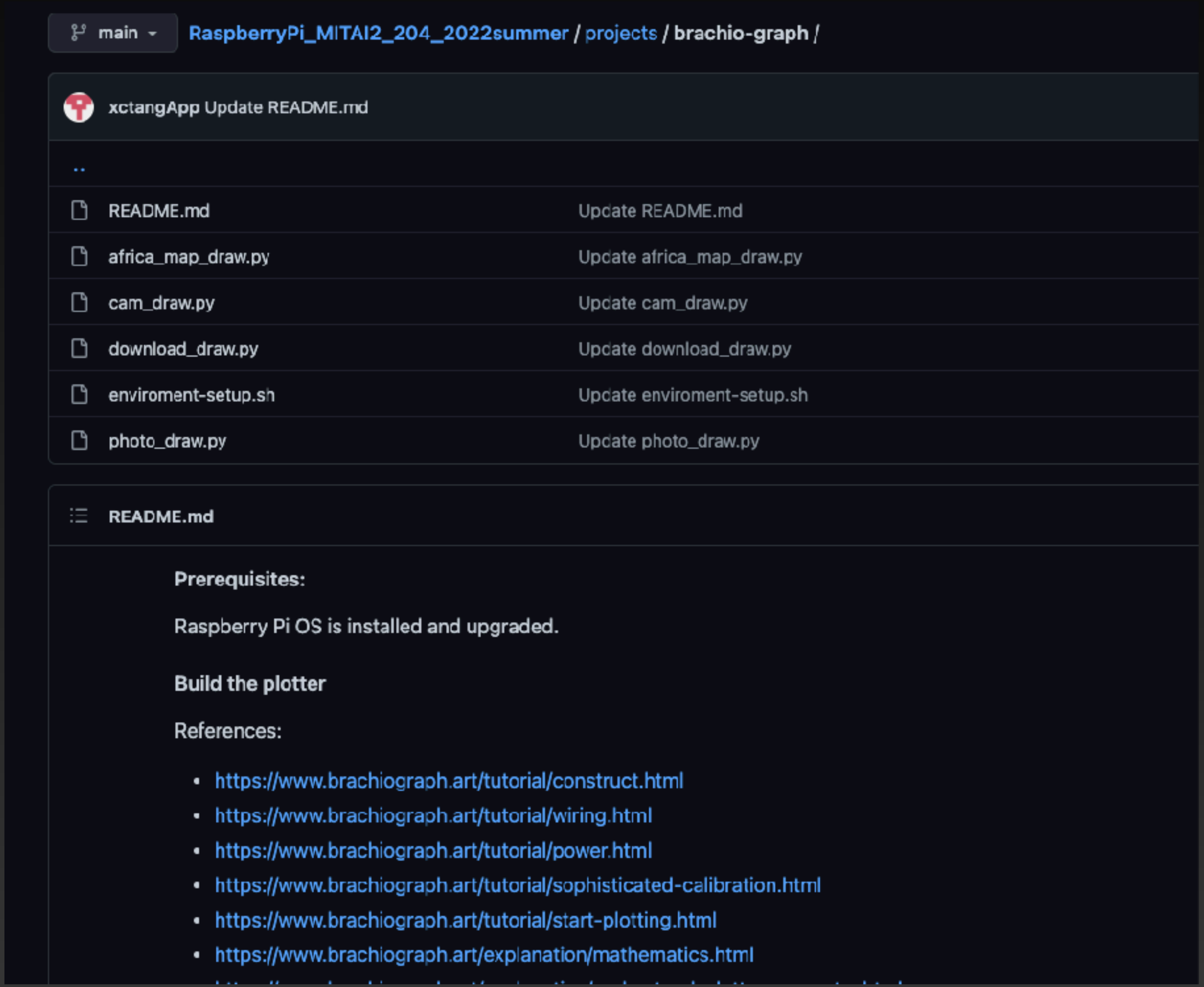
Servo Motor Control

```
def destroy():
    p.stop()
    GPIO.cleanup()

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

Capstone project - plotter machine

Project introduction will be in below link:

https://github.com/xctangApp/RaspberryPi_MITAI2_204_2022summer/tree/main/projects/brachio-graph



RaspberryPi_MITAI2_204_2022summer / projects / brachio-graph /

xctangApp Update README.md

..

README.md	Update README.md
africa_map_draw.py	Update africa_map_draw.py
cam_draw.py	Update cam_draw.py
download_draw.py	Update download_draw.py
enviroment-setup.sh	Update enviroment-setup.sh
photo_draw.py	Update photo_draw.py

README.md

Prerequisites:

Raspberry Pi OS is installed and upgraded.

Build the plotter

References:

- https://www.brachiograph.art/tutorial/construct.html
- https://www.brachiograph.art/tutorial/wiring.html
- https://www.brachiograph.art/tutorial/power.html
- https://www.brachiograph.art/tutorial/sophisticated-calibration.html
- https://www.brachiograph.art/tutorial/start-plotting.html
- https://www.brachiograph.art/explanation/mathematics.html

Capstone project - plotter machine

Read the part 1 of the tutorial to finish construct the plotter:

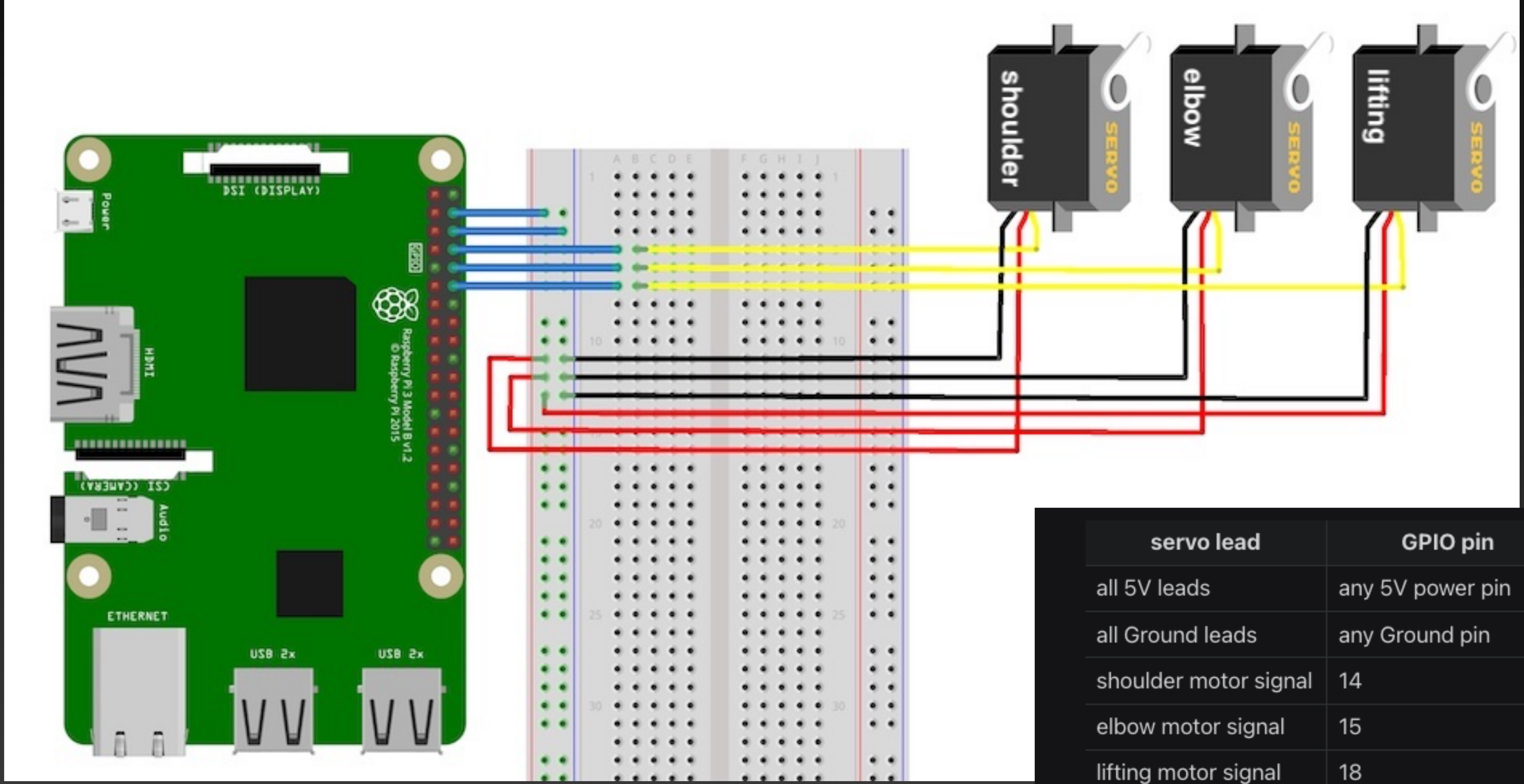
https://www.brachiograph.art/tutorial/construct.html

Video link:

https://youtu.be/7hI-9dHqTeg

Capstone project - plotter machine

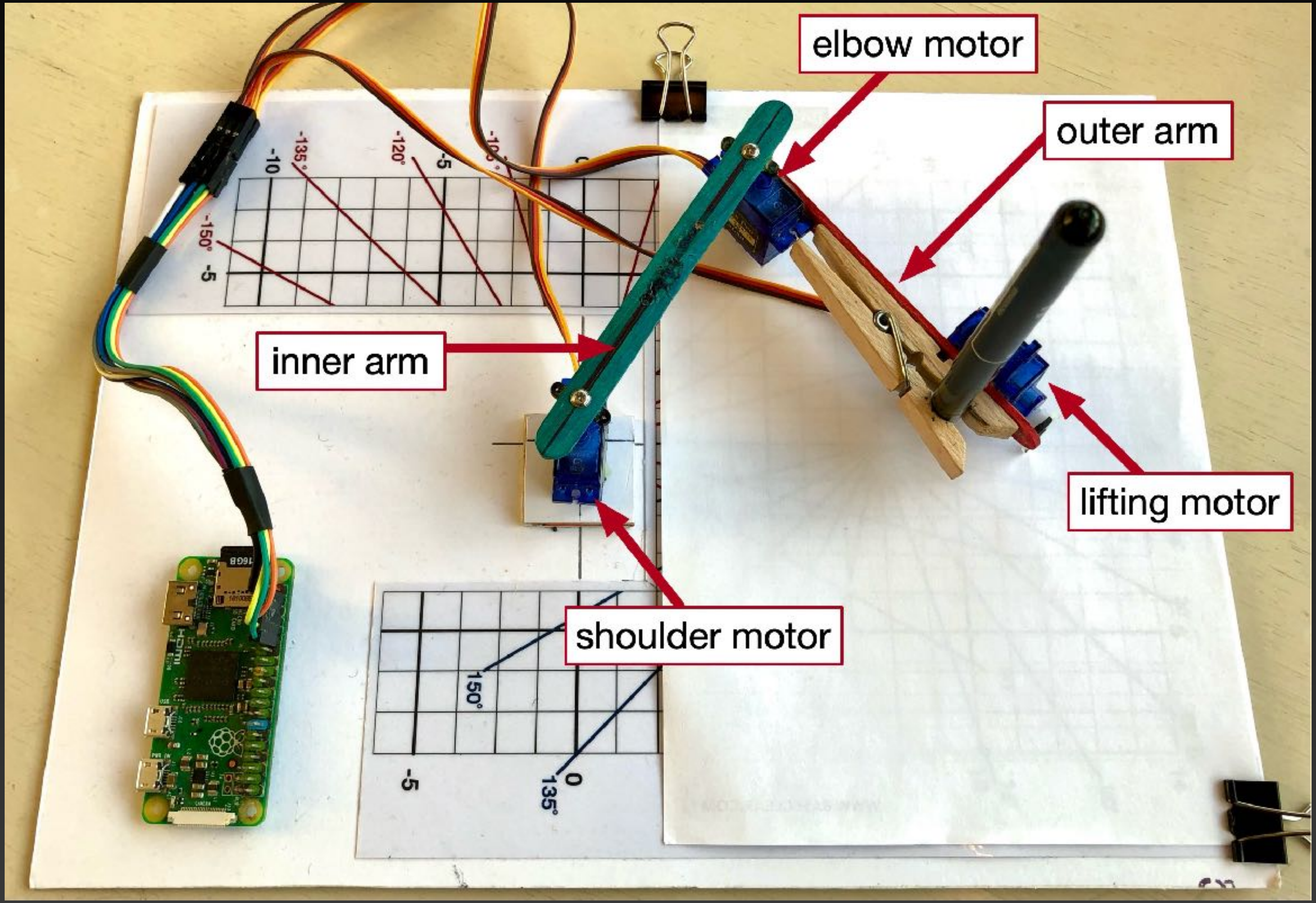
Wiring carefully, double check each connection all the way from the servo to the Raspberry Pi.



servo lead	GPIO pin	physical pin
all 5V leads	any 5V power pin	2 or 4
all Ground leads	any Ground pin	6, 9, 14, 20, 25, 30, 34, 36, 39
shoulder motor signal	14	8
elbow motor signal	15	10
lifting motor signal	18	12

Capstone project - plotter machine

Finish building your project.



Homework

Capstone project detailed instructions:

https://github.com/xctangApp/RaspberryPi_MITAI2_204_2022summer/blob/main/projects/brachio-graph/README.md

IMPORTANT!

Carefully go through the plotter machine construct link and finish assembling before tomorrow's class. We will focus on the software part of the project in class.

https://www.brachiograph.art/tutorial/construct.html