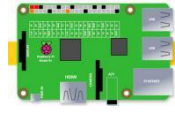
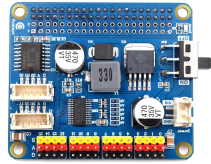



## Lesson 4 How to Control 180° Servo

In this lesson, we will learn how to control 180° Servo.

### 4.1 Components used in this course

Components	Quantity	Picture
Raspberry Pi	1	
Arm HAT	1	
180°Servo	1	

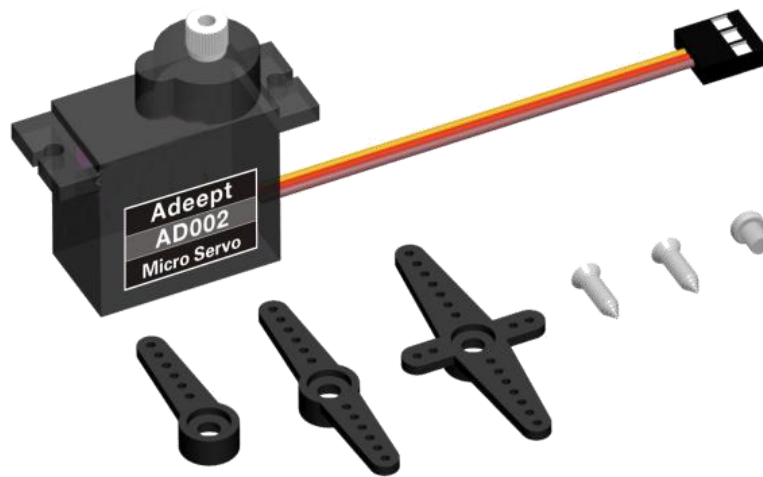
### 4.2 Introduction of 180° Servo

What is a servo?

The servo is a position (angle) servo driver, which is suitable for those control systems that require constant angle changes and can be maintained. It has been widely used in high-end remote control toys, such as airplanes, submarine models, and remote control robots.

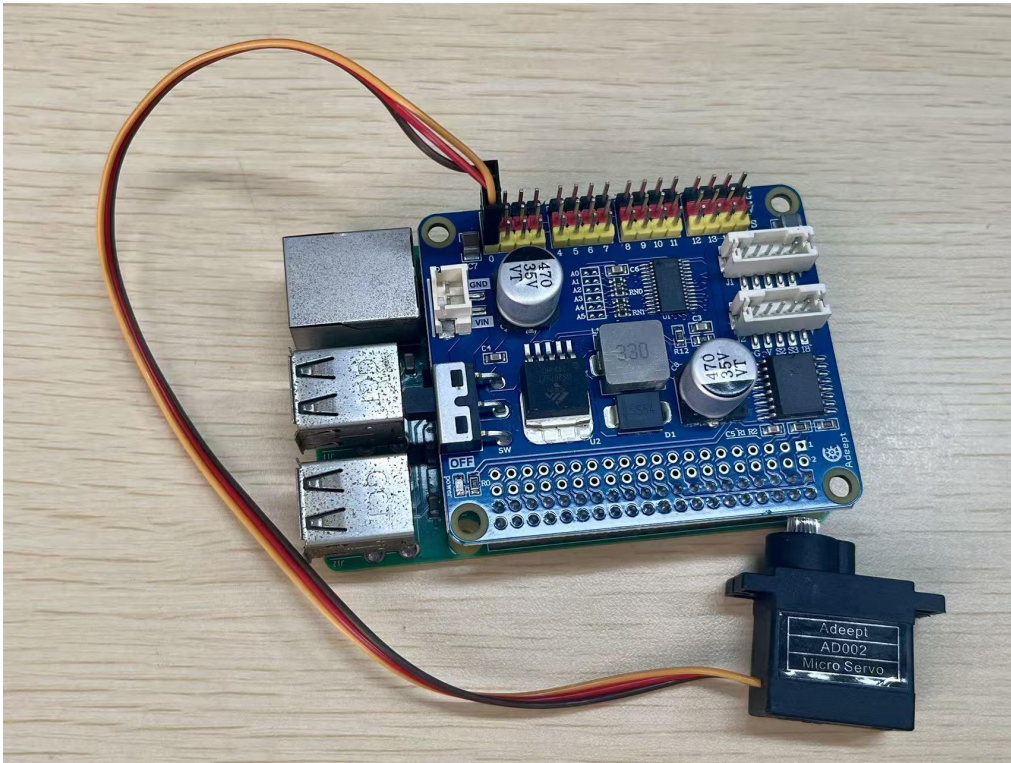
We use a 180° servo in this lesson, which can move between 0° and 180°. Since the 180° servo can use the PWM signal to control the rotation angle of a certain mechanism, it is a more commonly used module in robot products.

On the Raspberry Pi driver board Arm HAT, there is a PCA9685 chip specially used to control the servo. The Raspberry Pi uses I2C to communicate with the PCA9685. It controls the servo by sending pulse signals from the microcontroller. These pulses tell the servo mechanism of the servo where to move. The picture of the 180° servo is as follows:



### 4.3 Wiring diagram (Circuit diagram)

When the 180° Servo module is in use, it needs to be connected to the servo interface on the Arm HAT driver board. The yellow wire is connected to the yellow pin, the red wire is connected to the red pin, and the brown wire is connected to black pin, as shown below (connect to pin 0):



## 4.3 How to control 180°Servo

### Run the code

1. Remotely log in to the Raspberry Pi and use the following command to stop the auto-running program:

```
sudo killall python3
```

```
adeept@raspberrypi:~$ sudo killall python3
adeept@raspberrypi:~$
```

2. Enter the command and press Enter to enter the folder where the program is located:

```
cd Adeept_Robotic_Arm_for_RPi/Server/
```

```
adeept@raspberrypi:~$ cd Adeept_Robotic_Arm_for_RPi/Server/
adeept@raspberrypi:~/Adeept_Robotic_Arm_for_RPi/Server$
```

3. View the contents of the current directory file:

**ls**

```
adeept@raspberrypi:~/Adeept_Robotic_Arm_for_RPi/Server $ ls
app.py  dist  info.py  joystickControl.py  joystick.py  PCF8591.py  plan.json  __pycache__  RPiServo.py  servo.py  WebServer.py
adeept@raspberrypi:~/Adeept_Robotic_Arm_for_RPi/Server $
```

4. Enter the command and press Enter to run the program:

**sudo python3 servo.py**

```
adeept@raspberrypi:~/Adeept_Robotic_Arm_for_RPi/Server $ sudo python3 servo.py
```

5. After running the program successfully, you will observe that the servo will rotate regularly.

6. When you want to terminate the running program, you can press the shortcut key "Ctrl + C" on the keyboard.

## 4.4 The main code program

Complete code refer to [servo.py](#).

Control the servo to rotate to a certain angle.

```
1. # https://github.com/adafruit/Adafruit_CircuitPython_PCA9685
2. # sudo pip3 install adafruit-circuitpython-motor
3. # sudo pip3 install adafruit-circuitpython-pca9685
4.
5. import time
6. from board import SCL, SDA
7. import busio
8. from adafruit_motor import servo
9. from adafruit_pca9685 import PCA9685
10.
11.
12. i2c = busio.I2C(SCL, SDA)
13. # Create a simple PCA9685 class instance.
14. pca = PCA9685(i2c, address=0x40) #default 0x40
15.
16. pca.frequency = 50
```

```
17.  
18. def set_angle(ID, angle):  
19.     servo_angle = servo.Servo(pca.channels[ID], min_pulse=500, max_pulse=2400, actuation_range=180)  
20.     servo_angle.angle = angle  
21.  
22. def test(channel):  
23.     for i in range(180):  
24.         set_angle(channel, i)  
25.         time.sleep(0.01)  
26.         time.sleep(0.5)  
27.     for i in range(180):  
28.         set_angle(channel, 180-i)  
29.         time.sleep(0.01)  
30.         time.sleep(0.5)  
31.  
32. if __name__ == "__main__":  
33.     channel = 0  
34.     while True:  
35.         test(channel)
```

In the above code, `pca.frequency = 50` is used to set the PWM frequency to 50Hz. This setting depends on the model of the servo. The servo used by our robot products needs to be controlled by a 50Hz PWM signal. If you use For other servos, this value needs to be set by referring to the specific servo documentation.

`set_angle(ID, angle)` This method is used to control the rotation of a servo to a certain angle, where ID is the port number of the servo, which corresponds to the number marked on the Arm HAT driver board, but pay attention When the servo is connected to the drive board, do not insert the ground wire, VCC and signal wire in the reverse direction, brown to black, red to red, and yellow to yellow.