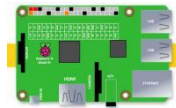




## Lesson 9 Control the Servo to Rotate

### 9.1 Overview

A servo is a position (angle) servo driver designed for control systems that require precise and maintainable angle changes. It's widely used in high - end remote - control toys like airplanes, submarine models, and remote - control robots. In this lesson, we use a 180° servo, which can move within the range of 0° to 180°

### 9.2 Required Components

Components	Quantity	Picture
Raspberry Pi	1	
Adept Robot HAT V3.2	1	
Servo	1	

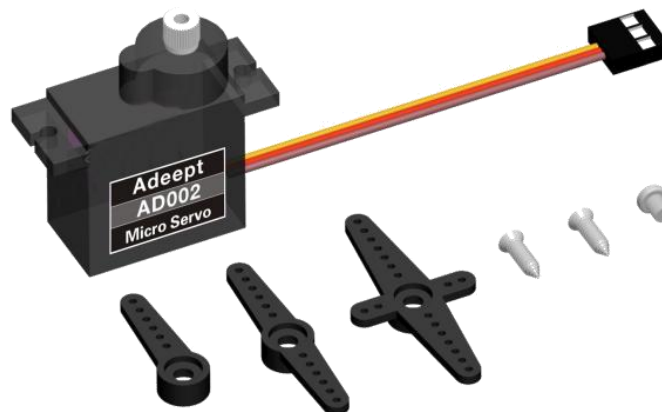
### 9.3 Principle Introduction

The 180° servo uses Pulse - Width Modulation (PWM) signals to control its rotation angle. The relationship between the pulse width and the rotation angle is as follows:

Pulse Width	Corresponding servo angle
500 $\mu$ s	0°
1500 $\mu$ s	90°
2400 $\mu$ s	180°

This linear relationship allows us to precisely control the servo's angle by adjusting the width of the PWM signal.

On the Adeept Robot HAT V3.2 Raspberry Pi driver board, there is a PCA9685 chip dedicated to servo control. The Raspberry Pi communicates with the PCA9685 via the I2C (Inter - Integrated Circuit) protocol. The microcontroller sends pulse signals to the PCA9685, which then forwards these signals to the servo, determining its movement.



The PCA9685 chip is connected to the I2C interface of the Raspberry Pi. It can control the 16 - pin signals extended by the PCA9685 module through its I2C address. The I2C address of Adeept Robot HAT V3.2 is 0x5f.

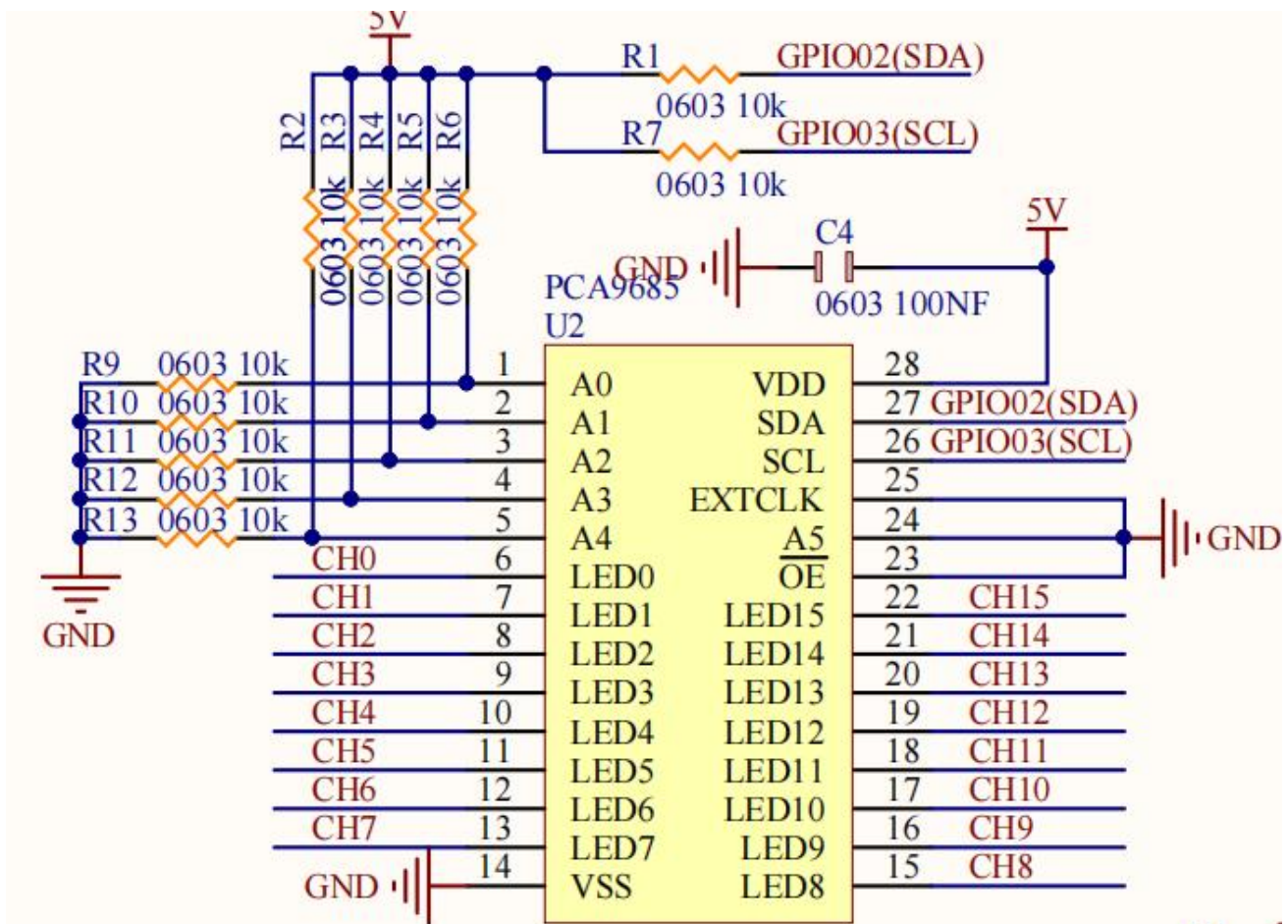
To view all the I2C addresses on the Raspberry Pi, you can enter the following command in the Raspberry Pi command line:

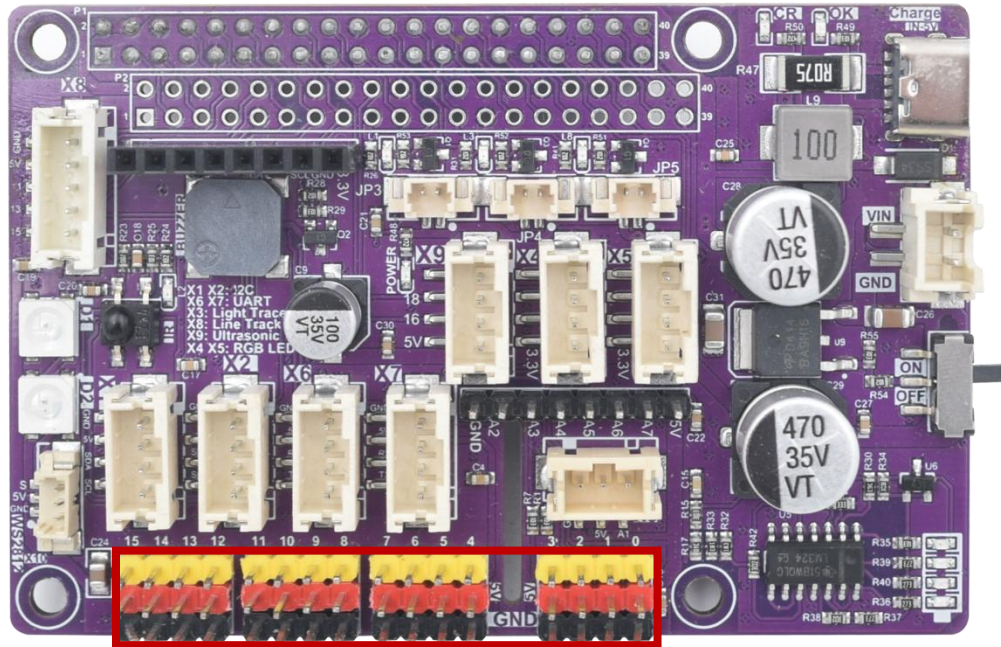
```
i2cdetect -y 1
```

## 9.4 Wiring Diagram

Note: Some of the pins on the PCA9685 extension are shared between the servo and the motor.

- The extended pins 0 - 7 of PCA9685 are solely for servo use.
- The extended pins 8 - 15 can be used for both the motor and the servo. If you need to use the motor while controlling the servo, it's recommended to use pins 0 - 7 for servo control and pins 8 - 15 for motor control. If the motor is not required, you can use pins 0 - 15 for servo control.





## 9.5 Demonstration

1. **Remotely log:** Remotely log in to the Raspberry Pi terminal.
2. **Navigate to the Program Folder:** Enter the following command in the terminal and press Enter to access the folder where the program is located:

```
cd Adeept_AWR-V3/Examples/03_Servo/
```

```
pi@raspberrypi:~ $ cd Adeept_AWR-V3/Examples/03_Servo/  
pi@raspberrypi:~/Adeept_AWR-V3/Examples/03_Servo $
```

3. **View Directory Contents:** Type "ls" in the terminal and press Enter. This will display all the files in the current directory, ensuring that the "Servo180Degree.py" file is present:

```
ls
```

```
pi@raspberrypi:~/Adeept_AWR-V3/Examples/03_Servo $ ls
Servo100Degree.py
```

4. **Run the Program:** Enter the command below and press Enter to start the servo control program:

```
sudo python3 Servo100Degree.py
```

```
pi@raspberrypi:~/Adeept_AWR-V3/Examples/03_Servo $ sudo python3 Servo100Degree.py
Servo on channel 0 starts to rotate 100 degrees.
```

5. **Observation and Termination:** Once the program runs successfully, you'll observe the specified servo motor rotating back and forth between 0° and 100°. To stop the running program, simply press the "**Ctrl + C**" shortcut on the keyboard. This action will set the servo to 90° and release the PCA9685 resources.

## 9.6 Code

Complete code refer to [Servo100Degree.py](#)

```
01  #!/usr/bin/env/python3
02  # File name   : Servo100Degree.py
03  # Website    : www.Adeept.com
04  # Author     : Adeept
05  # Date      : 2025/08/11
06  '''
07  # SPDX-License-Identifier: MIT
08  # Import the PCA9685 module. Available in the bundle and here:
09  #   https://github.com/adafruit/Adafruit_CircuitPython_PCA9685
10  # sudo pip3 install adafruit-circuitpython-motor
11  # sudo pip3 install adafruit-circuitpython-pca9685
12  '''
13  import time
14  from board import SCL, SDA
15  import busio
16  from adafruit_motor import servo
17  from adafruit_pca9685 import PCA9685
18
19  i2c = busio.I2C(SCL, SDA)
20  # Create a simple PCA9685 class instance.
21  pca = PCA9685(i2c, address=0x5f) # default 0x40
22
23  pca.frequency = 50
24
25
26  def set_angle(ID, angle):
27      servo_angle = servo.Servo(pca.channels[ID], min_pulse=500, max_pulse=2400, actuation_range=180)
28      servo_angle.angle = angle
29
```

```
30
31 def test(channel):
32     for i in range(100): # The servo turns from 0 to 180 degrees.
33         set_angle(channel, i)
34         time.sleep(0.01)
35     time.sleep(0.5)
36     for i in range(100): # The servo turns from 180 to 0 degrees.
37         set_angle(channel, 180 - i)
38         time.sleep(0.01)
39     time.sleep(0.5)
40
41
42 if __name__ == "__main__":
43     channel = 0
44     try:
45         print(f"Servo on channel {channel} starts to rotate 180 degrees.")
46         while True:
47             test(channel)
48     except KeyboardInterrupt:
49         print("Ctrl + C detected. Setting servo to 90 degrees.")
50         set_angle(channel, 90)
51         pca.deinit() # Release PCA9685 resources
52
```

## Code explanation

### Initialization Stage:

Connect PCA9685 module via I2C (address 0x5F), Set 50Hz PWM frequency (servo standard).

### Loop Control Process:

Stage 1: Servo  $0^\circ \rightarrow 100^\circ$  for 0.5 second

Stage 2: Servo  $100^\circ \rightarrow 0^\circ$  for 0.5 second

On **Ctrl+C**, Reset to  $90^\circ$  neutral position, Release hardware resources, Exit safely