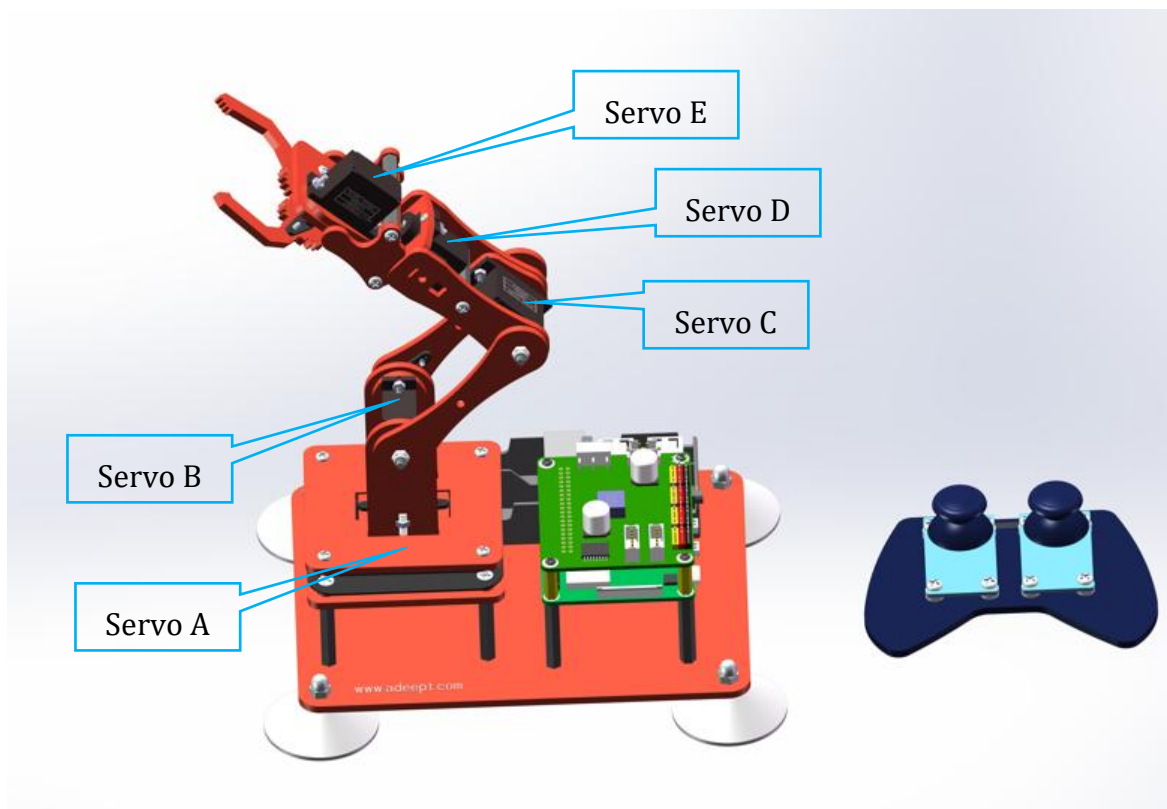
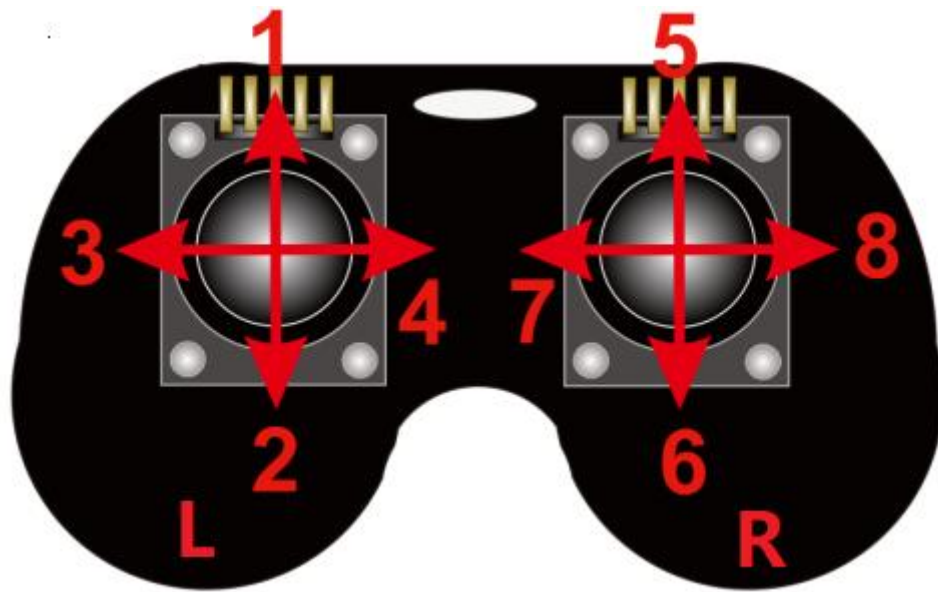


Lesson 8 Joystick Control Robotic Arm

8.1 Manipulate the robotic arm

Description of the joystick handle:





control method:

- 1, 2: Control servo **A** forward and reverse rotation.
- 3, 4: Control servo **B** to rotate forward and backward.
- 5, 6: Control servo **C** to rotate forward and backward.
- 7, 8: Control servo **E** forward and reverse rotation.
- After pressing the L joystick button, enter the servo **D** control mode,, 7,8: Control servo **D** forward and reverse rotation.
- After pressing the R joystick button, exit the servo D control mode.

8.2 Run the code

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 joystickControl.py
```

```
adeept@raspberrypi:~/Adeept_Robotic_Arm_for_RPi/Server$ sudo python3 joystickControl.py
home
L-up
L-right
L-up
home
L-down
home
L-left
home
L-right
home
```

5. After running the program successfully, You can control the robotic arm through the joystick.

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

8.3 The main code program

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

```
001  #!/usr/bin/env python3
002  import gpiozero
003  import PCF8591 as ADC
004  import time
005  from adafruit_motor import servo
006  from adafruit_pca9685 import PCA9685
007  import busio
008  from board import SCL, SDA
009
010  i2c = busio.I2C(SCL, SDA)
011  pca = PCA9685(i2c, address=0x40) # default 0x40
012  pca.frequency = 50
013
014  L_BTN_BCM = 17
015  R_BTN_BCM = 18
016  btn_L = gpiozero.Button(L_BTN_BCM, pull_up=True)
017  btn_R = gpiozero.Button(R_BTN_BCM, pull_up=True)
018
019  def set_angle(ID, angle):
020      servo_angle = servo.Servo(pca.channels[ID], min_pulse=500, max_pulse=2400, actuation_range=180)
021      servo_angle.angle = angle
022
023  angle = [90, 90, 90, 90, 90]
024  speed = 1
025  forward = 1
026  reverse = -1
027
028  mark = None
029  state_num = None
030  state_mark = None
031  servoD_mark = 0
```

```
032
033 def setup():
034     ADC.setup(0X48)
035
036     set_angle(0, 90)
037     set_angle(1, 90)
038     set_angle(2, 90)
039     set_angle(3, 90)
040     set_angle(4, 90)
041
042 def ctrl_range(raw, max_genout, min_genout):
043     if raw > max_genout:
044         raw_output = max_genout
045     elif raw < min_genout:
046         raw_output = min_genout
047     else:
048         raw_output = raw
049     return int(raw_output)
050
051 def rotation(ID, direction, speed):
052     global angle
053     if ID is None:
054         pass
055     else:
056         if direction == 1:
057             angle[ID] += speed
058         else:
059             angle[ID] -= speed
060         angle[ID] = ctrl_range(angle[ID], 180, 0)
061         set_angle(ID, angle[ID])
062
063 def move_servo(value):
064     if value == 1:
065         rotation(0, forward, speed)
066     elif value == -1:
067         rotation(0, reverse, speed)
068     elif value == 2:
069         rotation(1, forward, speed)
070     elif value == -2:
071         rotation(1, reverse, speed)
072     elif value == 3:
073         rotation(2, forward, speed)
```

```
074     elif value == -3:
075         rotation(2, reverse, speed)
076     elif value == 4:
077         rotation(4, forward, speed)
078     elif value == -4:
079         rotation(4, reverse, speed)
080     elif value == 5:
081         rotation(3, forward, speed)
082     elif value == -5:
083         rotation(3, reverse, speed)
084     else:
085         rotation(None, reverse, speed)
086
087 def joystick():
088     global state_num, state_mark, servoD_mark
089     state = ['home', 'L-pressed', 'L-down', 'L-up', 'L-right', 'L-left',
090             'R-home', 'R-pressed', 'R-down', 'R-up', 'R-left', 'R-right']
091     value = None
092
093     if btn_L.is_pressed:
094         value = -6
095         state_num = 1
096         servoD_mark = 1
097     elif btn_R.is_pressed:
098         value = 6
099         state_num = 7
100         servoD_mark = 0
101     else:
102         value = 0
103         state_num = 0
104
105     if ADC.read(0) <= 30: # servo A
106         value = 1
107         state_num = 2
108     elif ADC.read(0) >= 210: # servo A
109         value = -1
110         state_num = 3
111
112     if ADC.read(1) >= 210: # servo B
113         value = 2
114         state_num = 4
115     elif ADC.read(1) <= 30:
```

```
116     value = -2
117     state_num = 5
118
119     if ADC.read(2) <= 30: # servo C
120         value = 3
121         state_num = 8
122     elif ADC.read(2) >= 210: # servo C
123         value = -3
124         state_num = 9
125
126     if servoD_mark == 1:
127         if ADC.read(3) <= 30: # servo D
128             value = 5
129             state_num = 10
130         elif ADC.read(3) >= 210:
131             value = -5
132             state_num = 11
133     else:
134         if ADC.read(3) <= 30: # servo E
135             value = 4
136             state_num = 10
137         elif ADC.read(3) >= 210:
138             value = -4
139             state_num = 11
140     if state_mark != state_num: # print state.
141         print(state[state_num])
142     state_mark = state_num
143     return value
144
145 def loop():
146     global mark
147     value = joystick()
148     move_servo(value)
149     if mark != value:
150         mark = value
151     time.sleep(0.01)
152
153 def destroy():
154     btn_L.close()
155     btn_R.close()
156     pca.deinit()
157
```

```
158 if __name__ == '__main__':  
159     setup()  
160     try:  
161         while True:  
162             loop()  
163     except KeyboardInterrupt:  
164         destroy()  
165         print("\nProgram exited, resources cleaned up")
```