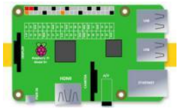
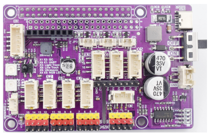
Lesson 8 How to Control DC Motor

In this lesson, we will learn how to control DC Motor.

8.1 Components used in this course

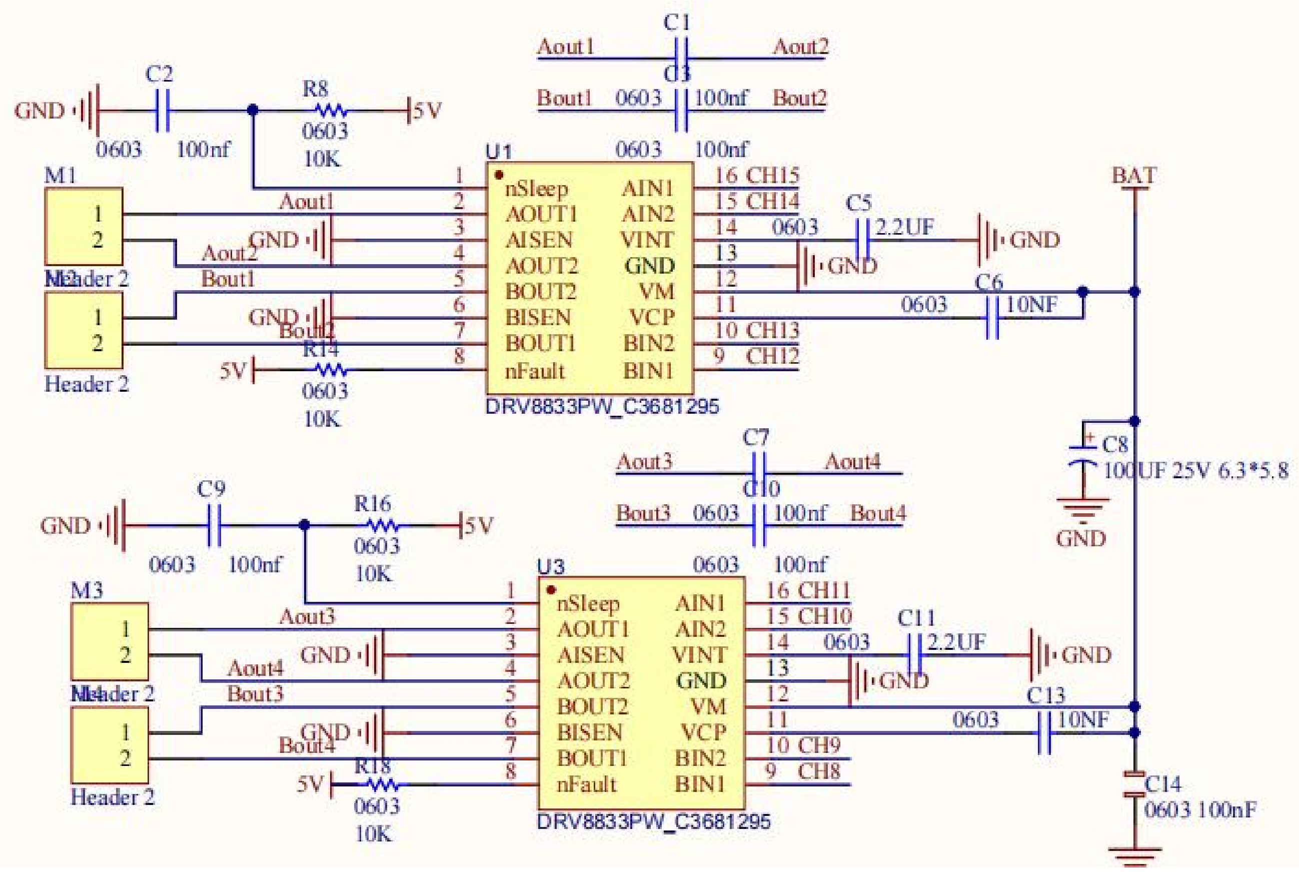
Components	Quantity	Picture
Raspberry Pi	1	
Adeept Robot HAT V3.0	1	
DC Motor	1	

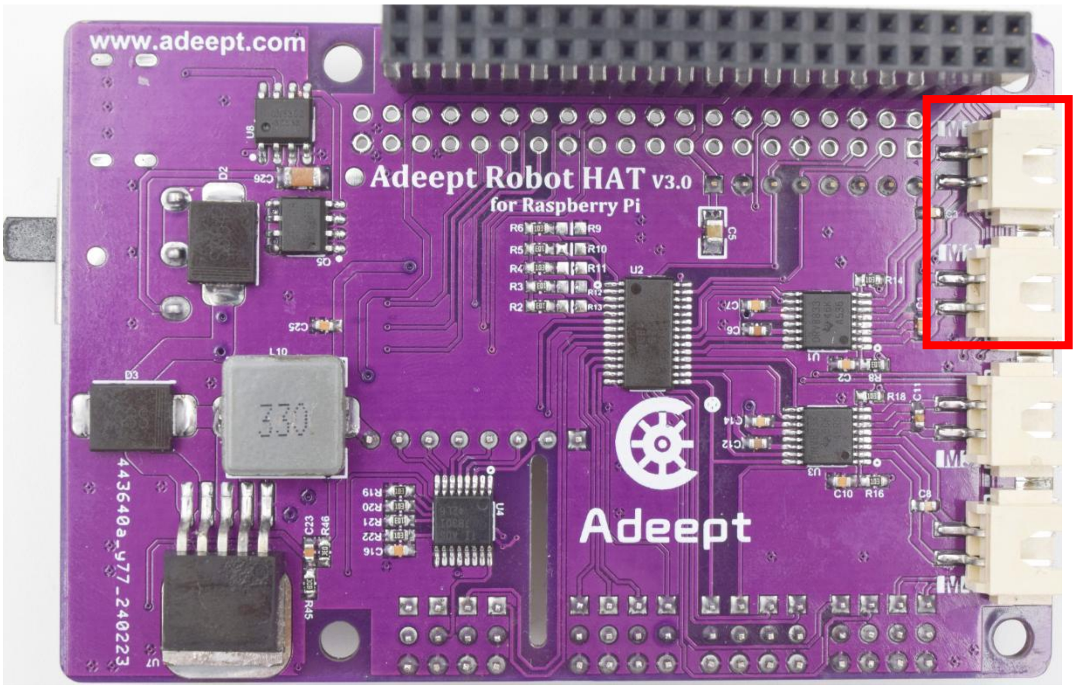
8.2 The introduction of DC Motor

PiCar-B robot products use DC motor as a power device. DC motor is a device that converts DC electrical energy into mechanical energy. It is widely used to drive various equipment, such as electric fans, remote control cars, electric windows, etc. The DC motor is very suitable as the walking mechanism of the robot.

8.3 Wiring diagram (Circuit diagram)

When the DC Motor module is in use, it needs to be connected to the M1/M2 interface on the Adeept Robot HAT V3.0 expanse board.

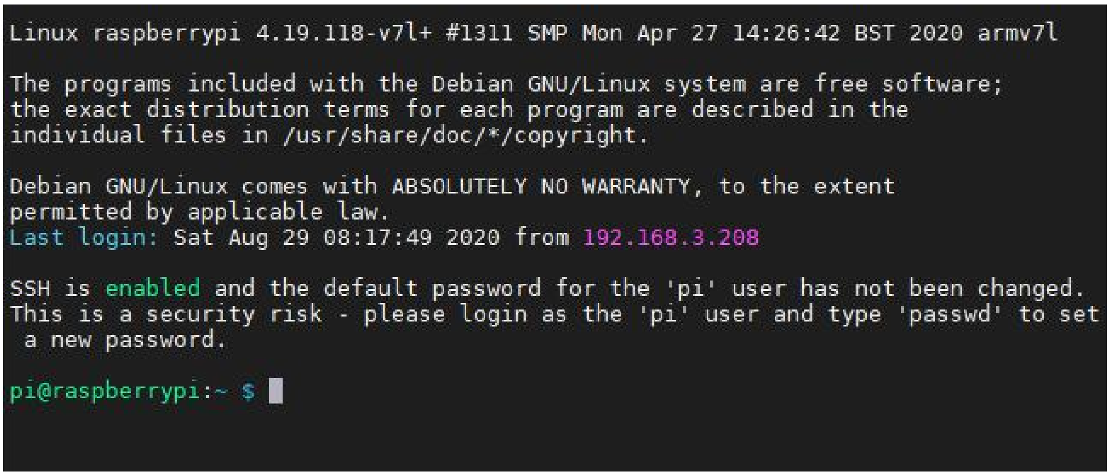




8.4 How to control Motor

Run the code

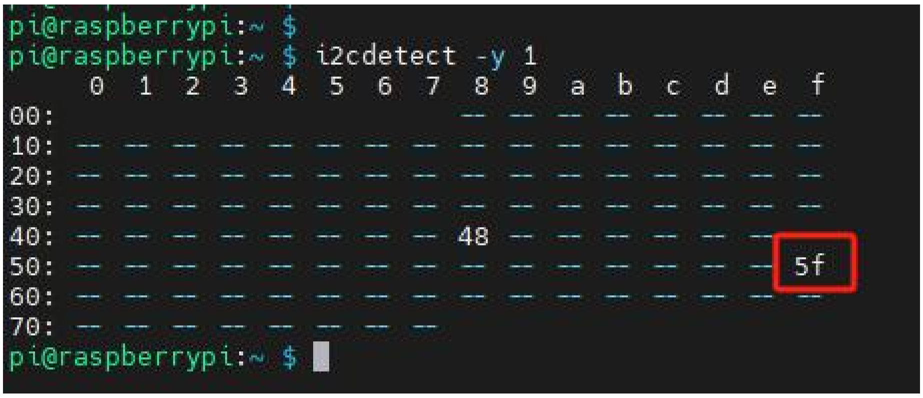
1. Remotely log in to the Raspberry Pi terminal.



```
Linux raspberrypi 4.19.118-v7l+ #1311 SMP Mon Apr 27 14:26:42 BST 2020 armv7l

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Sat Aug 29 08:17:49 2020 from 192.168.3.208

SSH is enabled and the default password for the 'pi' user has not been changed.
This is a security risk - please login as the 'pi' user and type 'passwd' to set
 a new password.

pi@raspberrypi:~ $ 
```

2. See the occupied I2C address.

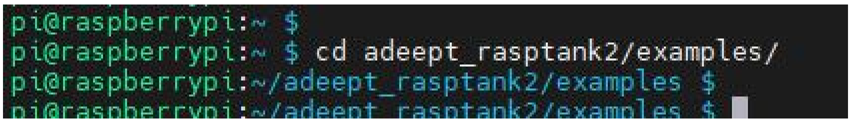
i2cdetect -y 1



```
pi@raspberrypi:~ $
pi@raspberrypi:~ $ i2cdetect -y 1
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:                         -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- 48 -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- 5f
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
pi@raspberrypi:~ $ 
```

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

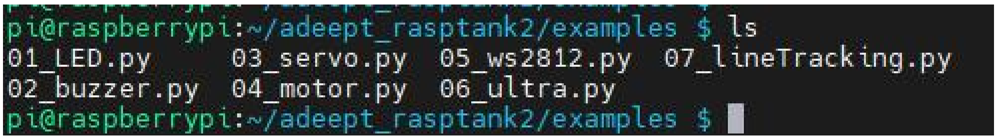
cd adeept_rasptank2/examples/



```
pi@raspberrypi:~ $
pi@raspberrypi:~ $ cd adeept_rasptank2/examples/
pi@raspberrypi:~/adeept_rasptank2/examples $
pi@raspberrypi:~/adeept_rasptank2/examples $ 
```

4. View the contents of the current directory file:

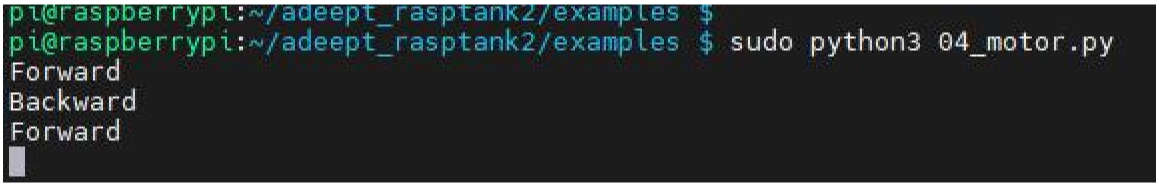
```
ls
```



```
pi@raspberrypi:~/adeept_rasptank2/examples $ ls
01_LED.py     03_servo.py  05_ws2812.py  07_lineTracking.py
02_buzzer.py  04_motor.py  06_ultra.py
pi@raspberrypi:~/adeept_rasptank2/examples $
```

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

```
sudo python3 04_motor.py
```



```
pi@raspberrypi:~/adeept_rasptank2/examples $
pi@raspberrypi:~/adeept_rasptank2/examples $ sudo python3 04_motor.py
Forward
Backward
Forward
```

6. After running the program successfully,You will observe that the motor will repeat the forward and reverse rotation ten times.

8.5 The main code program

For the complete code, please refer to the file **04_motor.py**.

```python
#!/usr/bin/env/python3
import time
from board import SCL, SDA
import busio
from adafruit_pca9685 import PCA9685
from adafruit_motor import motor

# motor_EN_A: Pin7  |  motor_EN_B: Pin11
# motor_A:  Pin8,Pin10    |  motor_B: Pin13,Pin12

MOTOR_M1_IN1 =  15      #Define the positive pole of M1
MOTOR_M1_IN2 =  14      #Define the negative pole of M1
MOTOR_M2_IN1 =  12      #Define the positive pole of M2
MOTOR_M2_IN2 =  13      #Define the negative pole of M2
MOTOR_M3_IN1 =  11      #Define the positive pole of M3
MOTOR_M3_IN2 =  10      #Define the negative pole of M3
MOTOR_M4_IN1 =  8       #Define the positive pole of M4
```

```
MOTOR_M4_IN2 =  9       #Define the negative pole of M4

Dir_forward   = 0
Dir_backward  = 1

left_forward  = 1
left_backward = 0

right_forward = 0
right_backward= 1

pwn_A = 0
pwm_B = 0

def map(x,in_min,in_max,out_min,out_max):
  return (x - in_min)/(in_max - in_min) *(out_max - out_min) +out_min


#def setup():
i2c = busio.I2C(SCL, SDA)
  #i2c = busio.I2C()
  # Create a simple PCA9685 class instance.
pwm_motor = PCA9685(i2c, address=0x5f) #default 0x40
pwm_motor.frequency = 1000

motor1 = motor.DCMotor(pwm_motor.channels[MOTOR_M1_IN1],pwm_motor.channels[MOTOR_M1_IN2] )
motor1.decay_mode = (motor.SLOW_DECAY)
motor2 = motor.DCMotor(pwm_motor.channels[MOTOR_M2_IN1],pwm_motor.channels[MOTOR_M2_IN2] )
motor2.decay_mode = (motor.SLOW_DECAY)
motor3 = motor.DCMotor(pwm_motor.channels[MOTOR_M3_IN1],pwm_motor.channels[MOTOR_M3_IN2] )
motor3.decay_mode = (motor.SLOW_DECAY)
motor4 = motor.DCMotor(pwm_motor.channels[MOTOR_M4_IN1],pwm_motor.channels[MOTOR_M4_IN2] )
motor4.decay_mode = (motor.SLOW_DECAY)
#  motorStop()

def Motor(channel,direction,motor_speed):
  if motor_speed > 100:
    motor_speed = 100
  elif motor_speed < 0:
    motor_speed = 0
  speed = map(motor_speed, 0, 100, 0, 1.0)
  if direction == -1:
    speed = -speed

  if channel == 1:
    motor1.throttle = speed
  elif channel == 2:
    motor2.throttle = speed
  elif channel == 3:
    motor3.throttle = speed
  elif channel == 4:
    motor4.throttle = speed


def motorStop():#Motor stops
    motor1.throttle = 0
    motor2.throttle = 0
    motor3.throttle = 0
    motor4.throttle = 0

def destroy():
  motorStop()
  pwm_motor.deinit()


if __name__ == '__main__':
  try:
    chann =  1
    # while True:
    for i in range(10):
      speed_set = 110
      Motor(chann, 1, speed_set)
      print("Forward")
      time.sleep(2)
      Motor(chann, -1 ,speed_set)
      print("Backward")
      time.sleep(2)
    destroy()
  except KeyboardInterrupt:
    destroy()
```