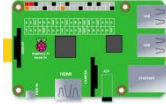
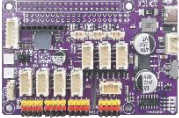
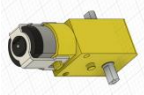


Lesson 10 Control the DC Motor to Work for Ordinary Wheels

10.1 Overview

In this lesson, we'll learn to control a DC motor with a Raspberry Pi and Adeept Robot HAT V3.2. We'll cover components, principles, wiring, run a Python program for forward - backward rotation, and analyze the code. By the end, you can control the motor's direction and speed.

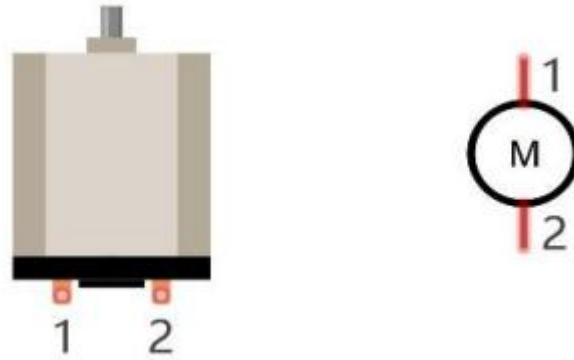
10.2 Required Components

Components	Quantity	Picture
Raspberry Pi	1	
Adeept Robot HAT V3.2	1	
DC Motor	4	

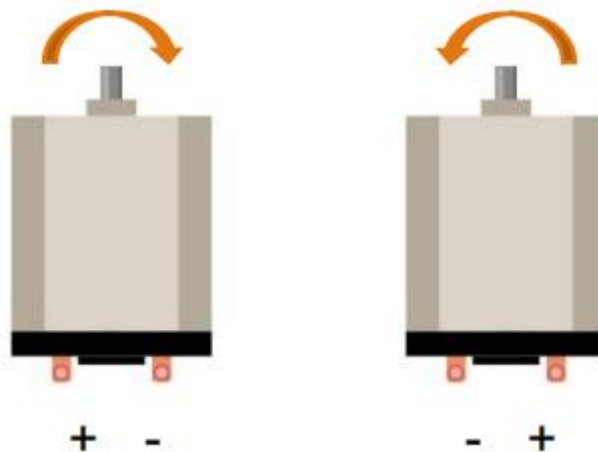
10.3 Principle Introduction

Our products use DC motor as a power device. A motor is a device that converts electrical energy into mechanical energy. Motor consists of two parts: stator and rotor. When motor works, the

stationary part is stator, and the rotating part is rotor. Stator is usually the outer case of motor, and it has terminals to connect to the power. Rotor is usually the shaft of motor, and can drive other mechanical devices to run. The schematic below is a small DC motor with two pins.



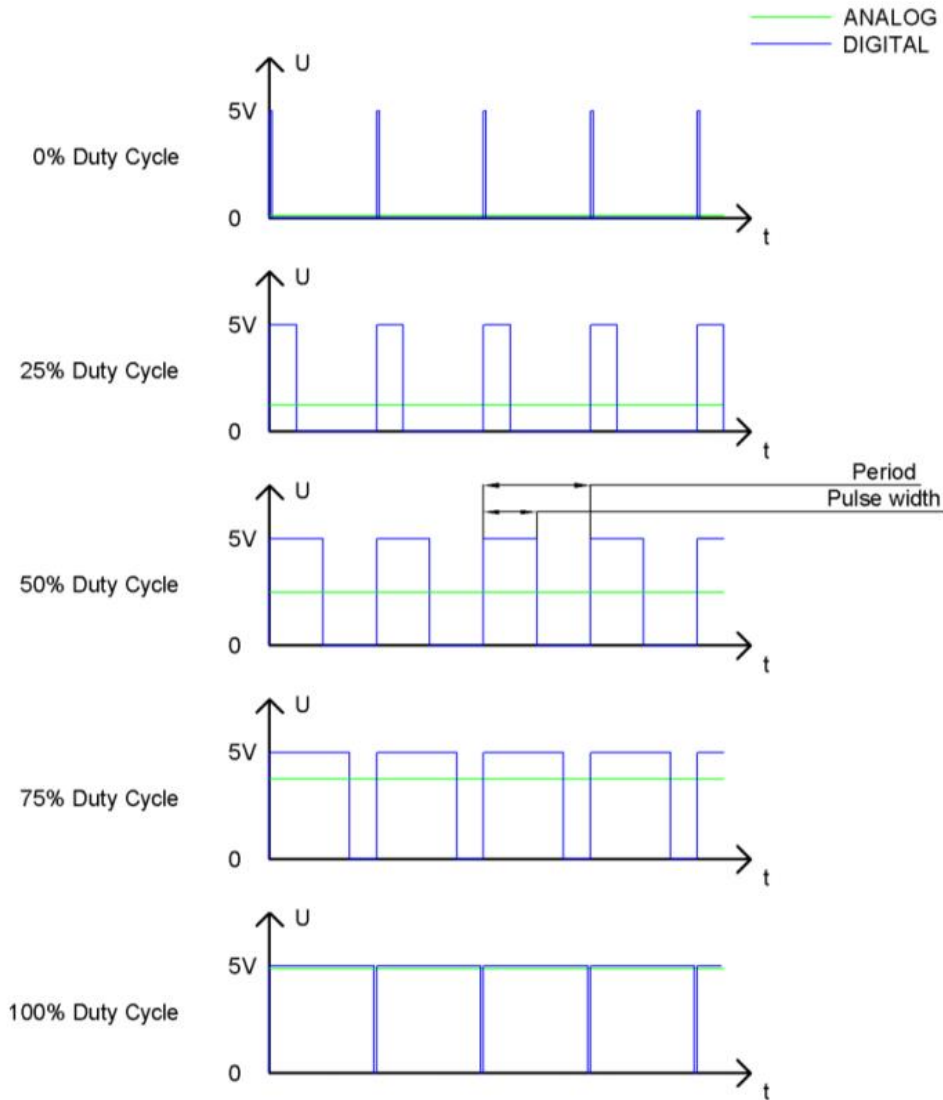
When a motor gets connected to the power supply, it will rotate in one direction. Reverse the polarity of power supply, then the motor rotates in opposite direction.



PWM

PWM, Pulse Width Modulation, uses digital pins to send certain frequencies of square waves, that is, the output of high levels and low levels, which alternately last for a while. The total time for each set of high levels and low levels is generally fixed, which is called the period (the reciprocal of the period is frequency). The time of high level outputs are generally called "pulse width", and the duty cycle is the percentage of the ratio of pulse duration, or pulse width (PW) to the total period (T) of the waveform.

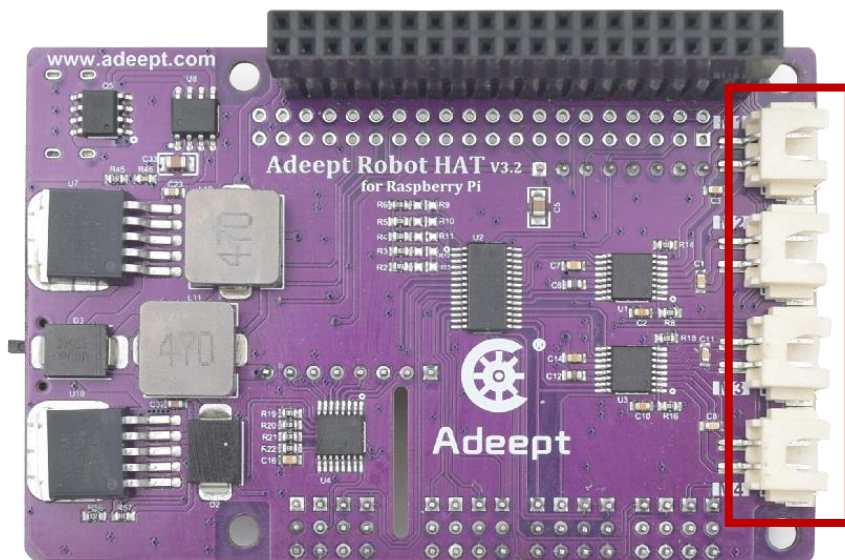
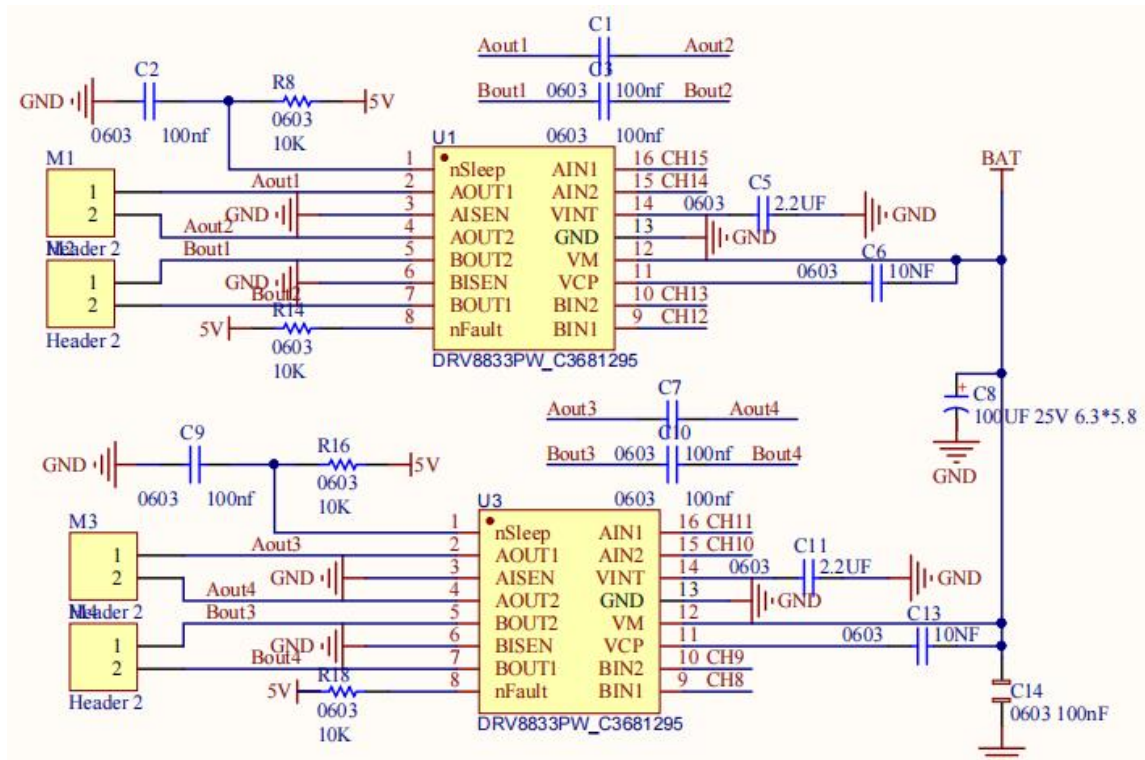
The longer the output of high levels last, the larger the duty cycle and the higher the corresponding voltage in analog signal will be. The following figures show how the analog signal voltage vary between 0V-5V(high level is 5V) corresponding to the pulse width 0%-100%:



The longer the PWM duty cycle is, the higher the output power will be. Now that we understand this relationship, we can use PWM to control the brightness of an LED or the speed of DC motor and so on.

10.4 Wiring Diagram

When the DC Motor module is in use, it needs to be connected to the M1 interface on the Adept Robot HAT V3.2 expansion board.



10.5 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_4WD_Smart_Car_for_RPi/Examples/04_Motor/
```

```
pi@raspberrypi:~ $ cd Adeept_4WD_Smart_Car_for_RPi/Examples/04_Motor/  
pi@raspberrypi:~/Adeapt_4WD_Smart_Car_for_RPi/Examples/04_Motor $
```

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 "**MotorCtrl.py**" file is present:

```
ls
```

```
pi@raspberrypi:~/Adeapt_4WD_Smart_Car_for_RPi/Examples/04_Motor $ ls  
Motor_Mecanum.py  Motor_Ordinary.py
```

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

```
sudo python3 Motor_Ordinary.py
```

```
pi@raspberrypi:~/Adeapt_4WD_Smart_Car_for_RPi/Examples/04_Motor $ sudo python3 Motor_Ordinary.py  
Forward  
Backward
```

5. **Observation and Termination:** Once the program runs successfully, you'll observe the DC motor rotating forward and backward. The program first sets the motor to rotate forward at a speed of 100 (scaled to the appropriate duty - cycle value) for 2 seconds, then rotates it backward at the same speed for 2 seconds. This process repeats 10 times. To stop the running program, simply press the "**Ctrl + C**" shortcut on the keyboard. This action will stop the motor and release the PCA9685 resources.

10.6 Code

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

```
01 #!/usr/bin/env/python3  
02 # File name   : Motor_Ordinary.py  
03 # Website    : www.Adeept.com
```

```
04 # Author      : Adeept
05 # Date        : 2025/03/6
06 import time
07 from board import SCL, SDA
08 import busio
09 from adafruit_pca9685 import PCA9685
10 from adafruit_motor import motor
11
12 MOTOR_M1_IN1 = 15      #Define the positive pole of M1
13 MOTOR_M1_IN2 = 14      #Define the negative pole of M1
14 MOTOR_M2_IN1 = 12      #Define the positive pole of M2
15 MOTOR_M2_IN2 = 13      #Define the negative pole of M2
16 MOTOR_M3_IN1 = 11      #Define the positive pole of M3
17 MOTOR_M3_IN2 = 10      #Define the negative pole of M3
18 MOTOR_M4_IN1 = 8        #Define the positive pole of M4
19 MOTOR_M4_IN2 = 9        #Define the negative pole of M4
20
21 def map(x,in_min,in_max,out_min,out_max):
22     return (x - in_min)/(in_max - in_min) *(out_max - out_min) +out_min
23
24 i2c = busio.I2C(SCL, SDA)
25 pwm_motor = PCA9685(i2c, address=0x5f)
26 pwm_motor.frequency = 50
27
28 motor1 = motor.DCMotor(pwm_motor.channels[MOTOR_M1_IN1],pwm_motor.channels[MOTOR_M1_IN2] )
29 motor1.decay_mode = (motor.SLOW_DECAY)
30 motor2 = motor.DCMotor(pwm_motor.channels[MOTOR_M2_IN1],pwm_motor.channels[MOTOR_M2_IN2] )
31 motor2.decay_mode = (motor.SLOW_DECAY)
32 motor3 = motor.DCMotor(pwm_motor.channels[MOTOR_M3_IN1],pwm_motor.channels[MOTOR_M3_IN2] )
33 motor3.decay_mode = (motor.SLOW_DECAY)
34 motor4 = motor.DCMotor(pwm_motor.channels[MOTOR_M4_IN1],pwm_motor.channels[MOTOR_M4_IN2] )
35 motor4.decay_mode = (motor.SLOW_DECAY)
36
37 def Motor(channel,direction,motor_speed):
38     if motor_speed > 100:
39         motor_speed = 100
40     elif motor_speed < 0:
41         motor_speed = 0
42     speed = map(motor_speed, 0, 100, 0, 1.0)
43     if direction == 1:
44         speed = -speed
45
46     if channel == 1:
47         motor1.throttle = speed
48     elif channel == 2:
49         motor2.throttle = speed
50     elif channel == 3:
51         motor3.throttle = speed
52     elif channel == 4:
53         motor4.throttle = speed
54
55 def motorStop():#Motor stops
56     motor1.throttle = 0
57     motor2.throttle = 0
58     motor3.throttle = 0
59     motor4.throttle = 0
```

```
60
61 def destroy():
62     motorStop()
63     pwm_motor.deinit()
64
65
66 if __name__ == '__main__':
67     try:
68         for i in range(10):
69             speed_set = 100
70             Motor(1, 1, speed_set)
71             Motor(2, 1, speed_set)
72             Motor(3, 1, speed_set)
73             Motor(4, 1, speed_set)
74             print("Forward")
75             time.sleep(2)
76             motorStop()
77             time.sleep(1)
78             Motor(1, -1 ,speed_set)
79             Motor(2, -1 ,speed_set)
80             Motor(3, -1 ,speed_set)
81             Motor(4, -1 ,speed_set)
82             print("Backward")
83             time.sleep(2)
84             motorStop()
85             time.sleep(1)
86         destroy()
87     except KeyboardInterrupt:
88         destroy()
```

Code explanation

Initialization Stage:

Connect the PCA9685 module (address 0x5F) via the I2C protocol. Set the PWM frequency to 50Hz and initialize 4 DC motors: M1 - M4 (bind them to GPIO pins respectively and configure them in slow decay mode).

Loop Control Process:

The main program loops 10 times:

Stage 1: Motor M1-M4 rotates forward at 100% speed → Run for 2 seconds.

Stage 2: Motor M1-M4 stops running → Run for 1 second.

Stage 3: Motor M1-M4 rotates backward at 100% speed → Run for 2 seconds.

Stage 4: Motor M1-M4 stops running → Run for 1 second.

Safety Mechanism:

motorStop(): Immediately stop all motors.

destroy(): Release the PCA9685 resources.

Press **Ctrl+C** to trigger a safe exit.