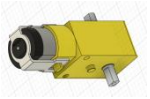
Lesson 9 Control the DC Motor to Work for Mecanum Wheels

9.1 Overview

This course focuses on DC motor control and aims to guide learners in mastering the methods of using Adeept Robot Control Board and related components to achieve DC motor operation control. It covers hardware connection, principle analysis, code writing and debugging, and helps learners to deeply understand and practice the application of DC motors in practical projects.

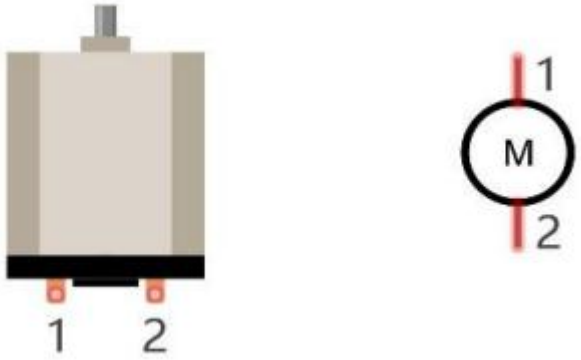
9.2 Required Components

Components	Quantity	Picture
Adeept Robot Control Board	1	
Type-C USB Cable	1	
DC Motor	1	

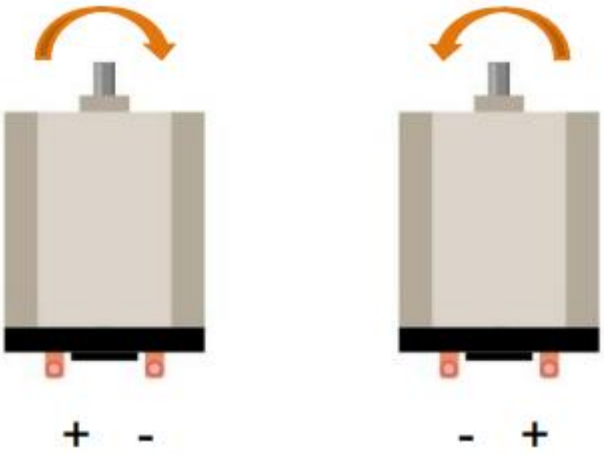
9.3 Principle Introduction

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.



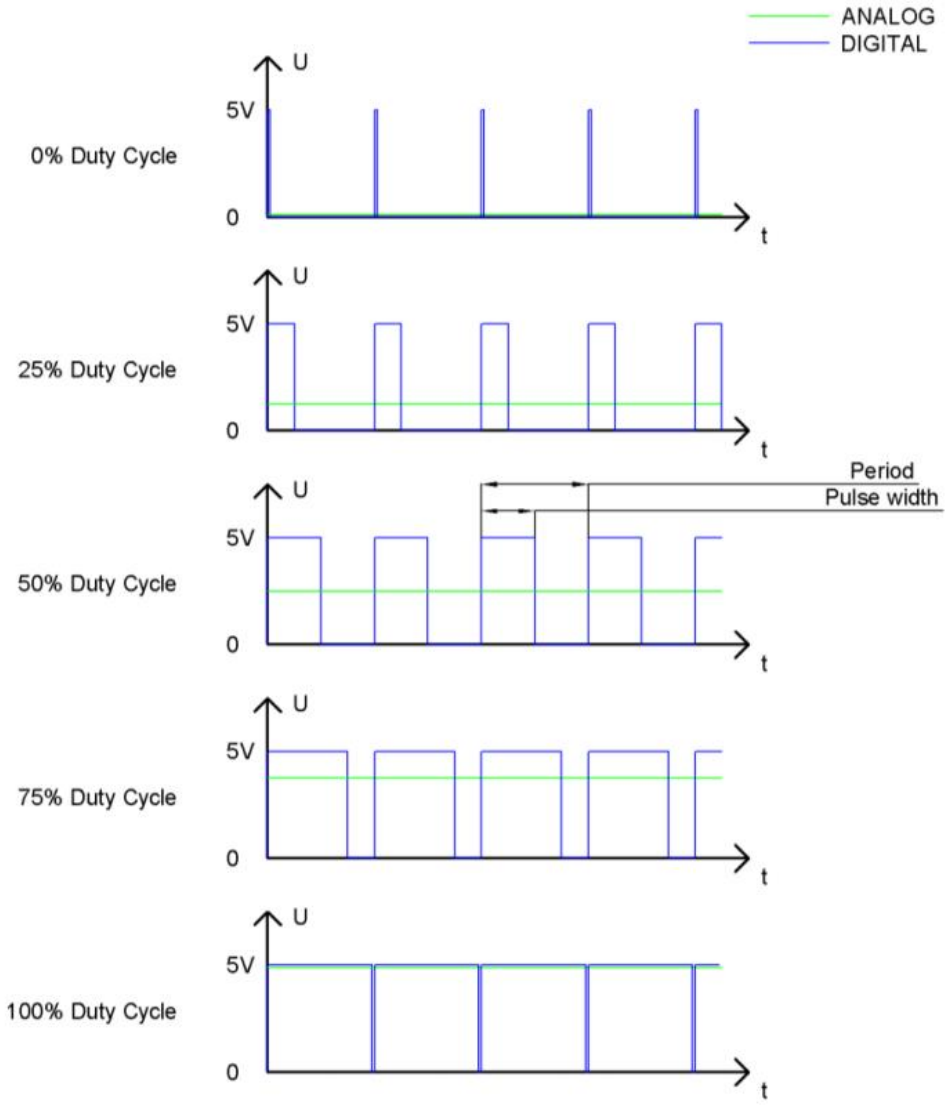
When a motor gets connected to the power supply, it will rotate in one direction. Reverse the polarity of power supply, then the motor rotates in opposite direction.



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.

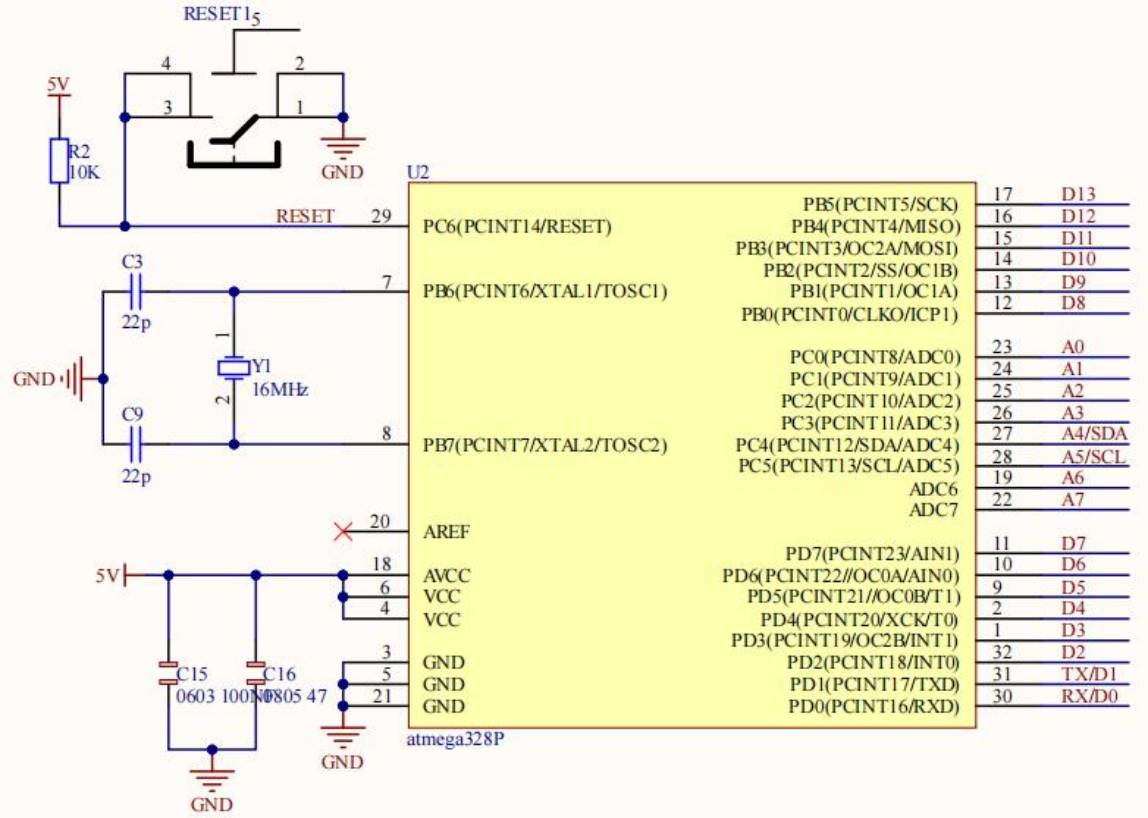
The longer the output of high levels last, the larger the duty cycle and the higher the corresponding voltage in analog signal will be. The following figures show how the analogs signal voltage vary between 0V-5V(high level is 5V) corresponding to the pulse width 0%-100%:

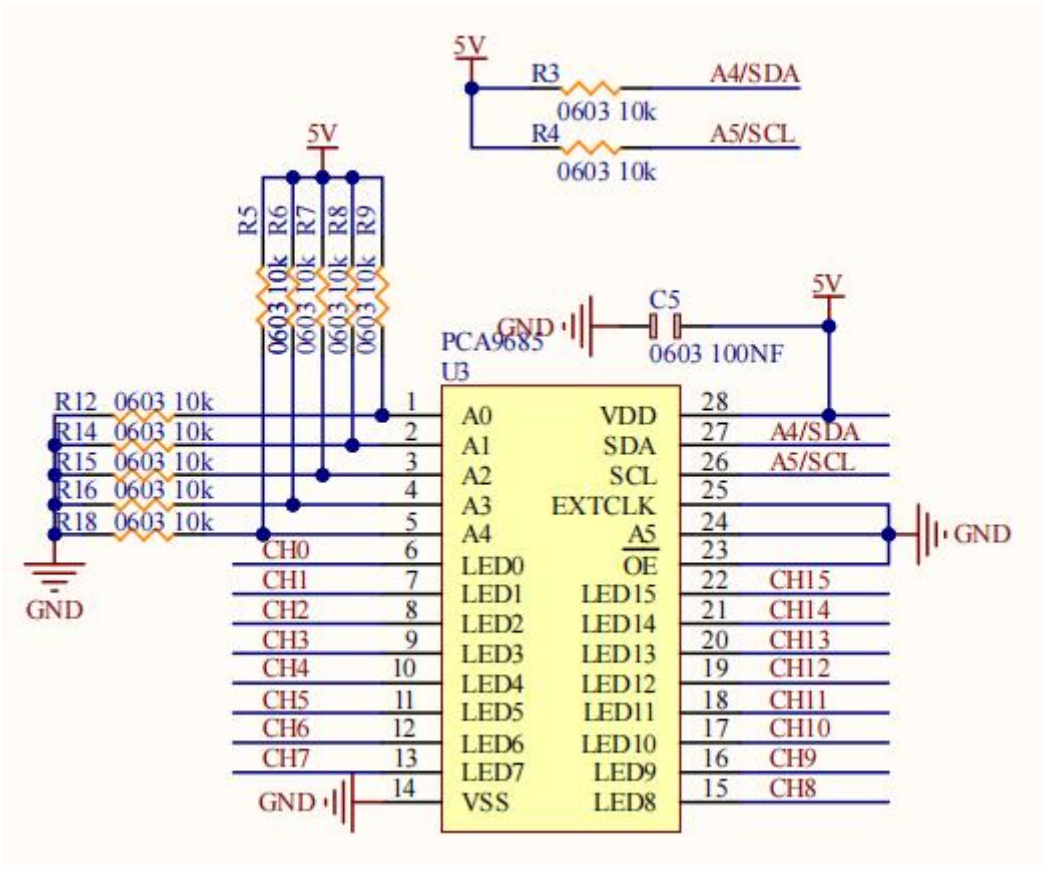


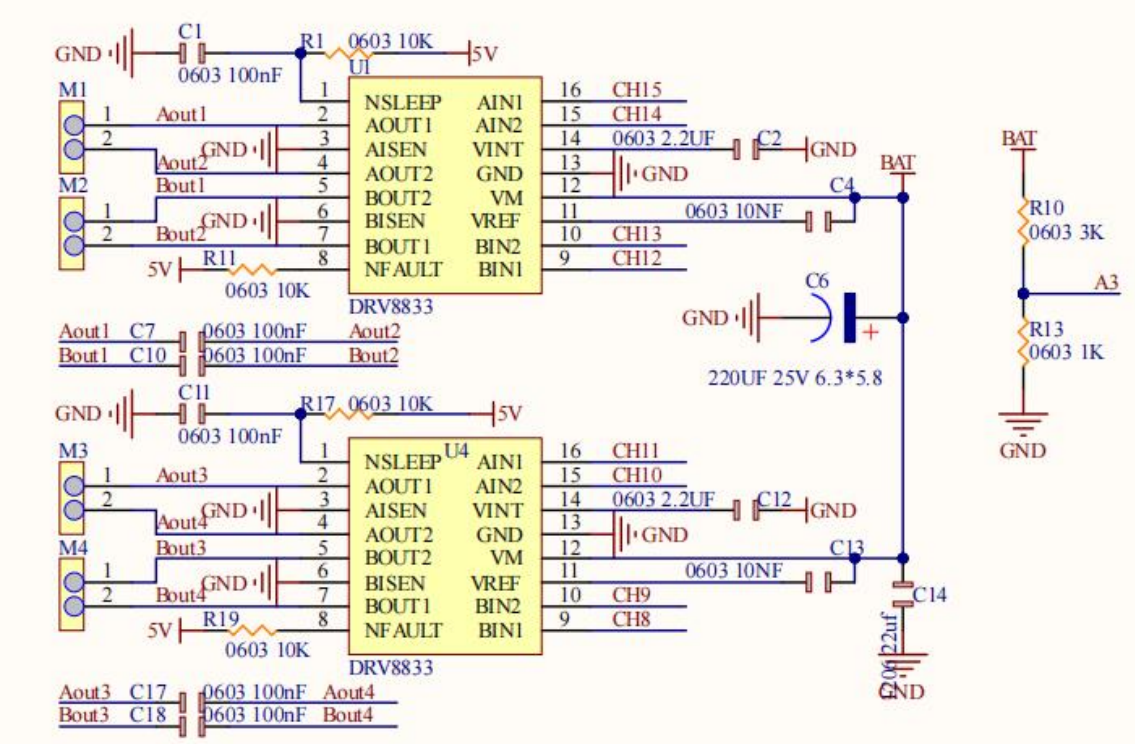
The longer the PWM duty cycle is, the higher the output power will be. Now that we understand this relationship, we can use PWM to control the brightness of an LED or the speed of DC motor and so on.

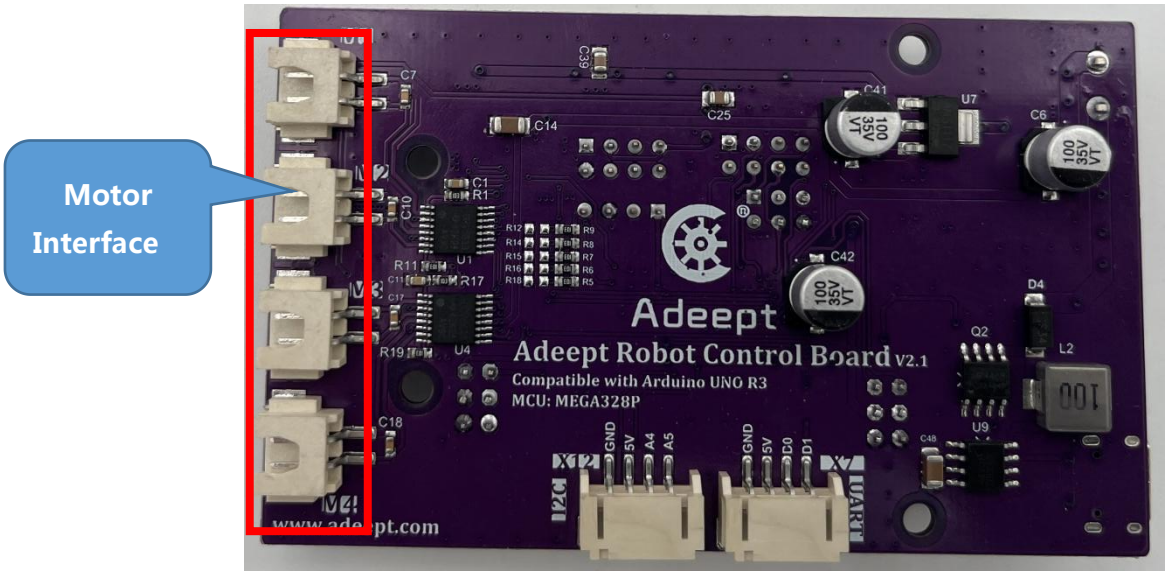
9.4 Wiring Diagram

Extend the Arduino pin interface using PCA9685. Use DRV8833 as the motor driver chip. Circuit schematic:


RESET
5V
R2
10K
GND
U2
PC6(PCINT14/RESET)
PB6(PCINT6/XTAL1/TOSC1)
PB7(PCINT7/XTAL2/TOSC2)
AREF
AVCC
VCC
VCC
GND
GND
GND
C3
22p
C9
22p
Y1
16MHz
C15
0603 100NF
C16
0805 47
atmega328P
PB5(PCINT5/SCK)
PB4(PCINT4/MISO)
PB3(PCINT3/OC2A/MOSI)
PB2(PCINT2/SS/OC1B)
PB1(PCINT1/OC1A)
PB0(PCINT0/CLKO/ICP1)
PC0(PCINT8/ADC0)
PC1(PCINT9/ADC1)
PC2(PCINT10/ADC2)
PC3(PCINT11/ADC3)
PC4(PCINT12/SDA/ADC4)
PC5(PCINT13/SCL/ADC5)
ADC6
ADC7
PD7(PCINT23/AIN1)
PD6(PCINT22//OC0A/AIN0)
PD5(PCINT21//OC0B/T1)
PD4(PCINT20/XCK/T0)
PD3(PCINT19/OC2B/INT1)
PD2(PCINT18/INT0)
PD1(PCINT17/TXD)
PD0(PCINT16/RXD)
D13
D12
D11
D10
D9
D8
A0
A1
A2
A3
A4/SDA
A5/SCL
A6
A7
D7
D6
D5
D4
D3
D2
TX/D1
RX/D0


5V
R3
0603 10k
A4/SDA
R4
0603 10k
A5/SCL
R5
R6
R7
R8
R9
0603 10k
C5
0603 100NF
GND
PCA9685
U3
R12 0603 10k
R14 0603 10k
R15 0603 10k
R16 0603 10k
R18 0603 10k
A0
A1
A2
A3
A4
LED0
LED1
LED2
LED3
LED4
LED5
LED6
LED7
VSS
VDD
SDA
SCL
EXTCLK
A5
OE
LED15
LED14
LED13
LED12
LED11
LED10
LED9
LED8
CH0
CH1
CH2
CH3
CH4
CH5
CH6
CH7
CH15
CH14
CH13
CH12
CH11
CH10
CH9
CH8





9.5 About Mecanum Wheels

Mecanum Wheel is a patent of the Swedish Mecanum Company, which is often used in the field of robotics to achieve omnidirectional movement. The Mecanum wheel consists of two parts: the hub and the roller. The hub is the main support for the entire wheel, and the rollers

are the drums mounted on the hub. The hub axle is at a 45° angle to the roller axis. (The angle between the hub axle of the omnidirectional wheel and the roller is 90 degrees)

Mecanum wheels, like conventional wheels, can be mounted on axes parallel to each other. Wheat wheels are generally used in groups of four, two left-handed wheels and two right-handed wheels. The difference between left-handed and right-handed wheels is shown in the figure below

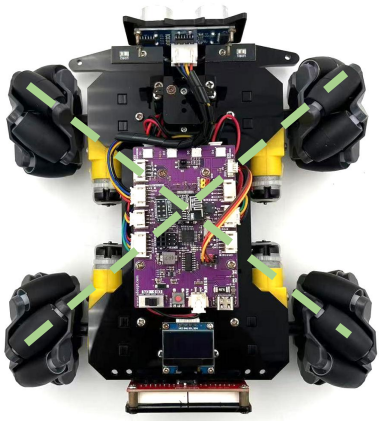
The Mecanum wheel car uses Mecanum wheels on the basis of ordinary cars, and each wheel can be controlled independently.

There are also many ways to install Mecanum wheels on cars. Mainly divided into:

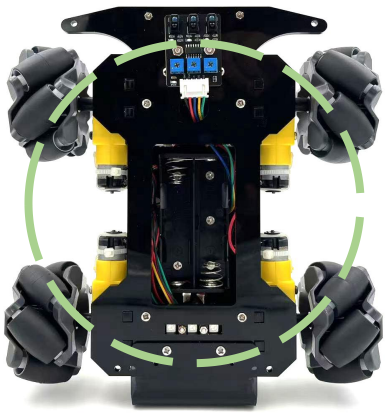
X-square (X-square), X-rectangle (X-rectangle), O-square (O-square), O-rectangle (O-rectangle).

Where X and O represent the figure formed by the rollers in contact with the ground of the four wheels; square and rectangle refer to the shape enclosed by the contact points of the four wheels with the ground.

The installation method of our Mecanum wheel car products is the common O-rectangular installation method.



X

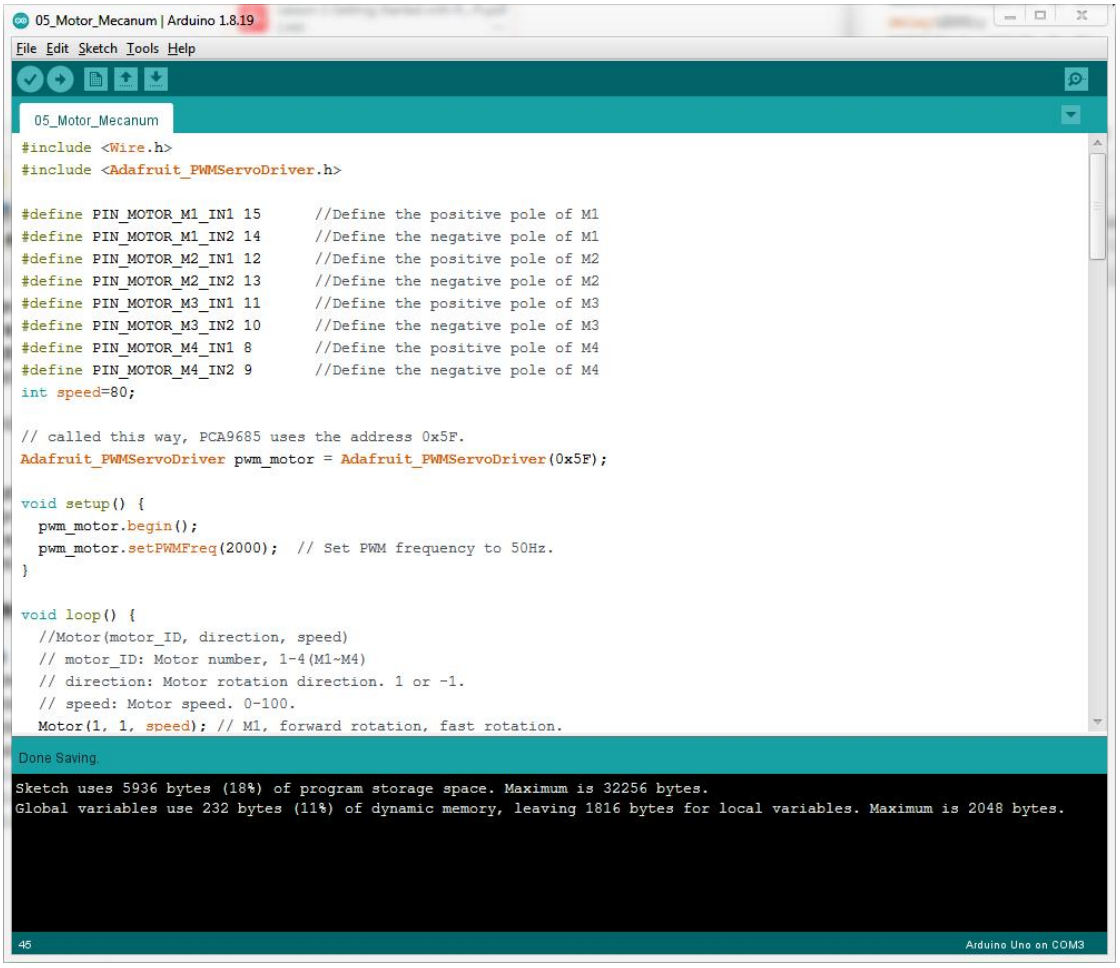


O

Note: The Mecanum wheel car adopts the O-rectangle assembly method. The actual wheel is "X" when viewed from above, and is "O" when it is actually in contact with the ground. (looking up from below the Mecanum wheel car)

9.6 Demonstration

1. Connect your computer and Adeept Robot Control Board (Arduino Board) with a USB cable.
2. Open "**05_Motor_Mecanum**" folder in "/Code", double-click "05_Motor_Mecanum.ino".

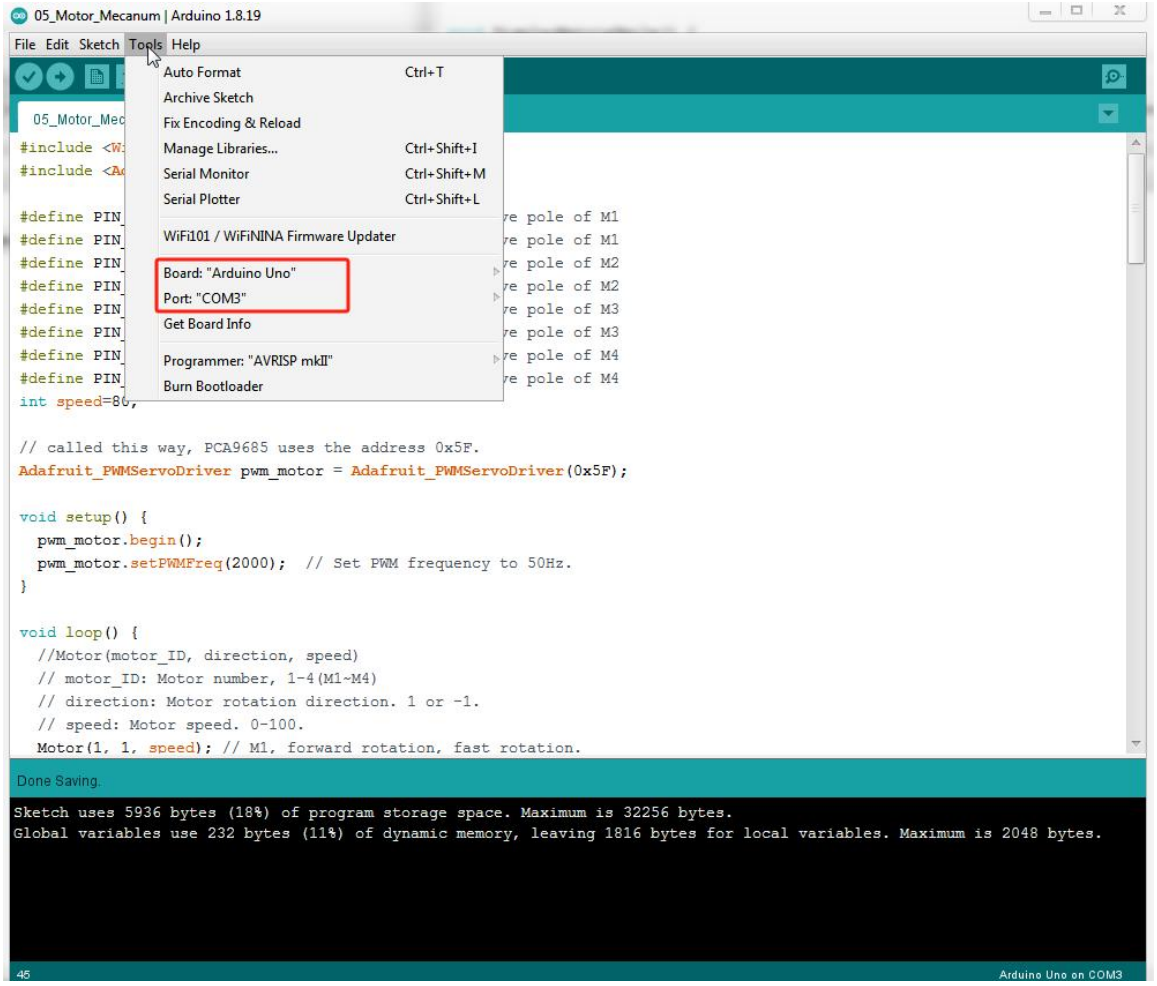


3. Select development board and serial port.

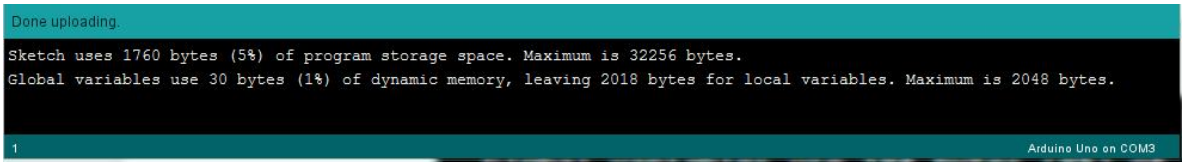
Board: Tools--->Board--->Arduino AVR Boards--->Arduino Uno

Port: Tools --->Port--->COMx

Note: The port number will be different in different computers.



4. After opening, click  to upload the code program to the Arduino. If there is no error warning in the console below, it means that the Upload is successful.



5. The DC motor operates according to the program settings, which sequentially execute the actions of forward full speed rotation for 1 second, stop for 1 second, reverse full speed rotation for 1 second, and stop for 1 second.

9.7 Code

Complete code refer to 05_Motor_Mecanum.ino

```
#include <Wire.h>
#include <Adafruit_PWMServoDriver.h>

#define PIN_MOTOR_M1_IN1 15      //Define the positive pole of M1
#define PIN_MOTOR_M1_IN2 14      //Define the negative pole of M1
#define PIN_MOTOR_M2_IN1 12      //Define the positive pole of M2
#define PIN_MOTOR_M2_IN2 13      //Define the negative pole of M2
#define PIN_MOTOR_M3_IN1 11      //Define the positive pole of M3
#define PIN_MOTOR_M3_IN2 10      //Define the negative pole of M3
#define PIN_MOTOR_M4_IN1 8       //Define the positive pole of M4
#define PIN_MOTOR_M4_IN2 9       //Define the negative pole of M4
int speed=80;

// called this way, PCA9685 uses the address 0x5F.
Adafruit_PWMServoDriver pwm_motor = Adafruit_PWMServoDriver(0x5F);

void setup() {
  pwm_motor.begin();
  pwm_motor.setPWMFreq(2000);
}

void loop() {
  //Motor(motor_ID, direction, speed)
  // motor_ID: Motor number, 1-4(M1~M4)
  // direction: Motor rotation direction. 1 or -1.
  // speed: Motor speed. 0-100.
  Motor(1, 1, speed); // M1, forward rotation, fast rotation.
  Motor(2, 1, speed);  // M2, forward rotation,low rotation.
  Motor(3, 1, speed); // M3, forward rotation, fast rotation.
  Motor(4, 1, speed); // M4, forward rotation, fast rotation.
  delay(2000);

  Motor(1,-1, speed);  //backward
  Motor(2,-1, speed);
  Motor(3,-1, speed);
  Motor(4,-1, speed);
  delay(2000);

  Motor(1,-1, speed);  //turn left
  Motor(2,-1, speed);
  Motor(3,1, speed);
  Motor(4,1, speed);
  delay(2000);

  Motor(1,1, speed);  //turn right
  Motor(2,1, speed);
  Motor(3,-1, speed);
  Motor(4,-1, speed);
  delay(2000);

  Motor(1, -1, speed);   //drift left
  Motor(2, 1, speed);
  Motor(3, -1, speed);
  Motor(4, 1, speed);
  delay(2000);

```

```
  Motor(1, 1, speed);    //drift right
  Motor(2, -1, speed);
  Motor(3, 1, speed);
  Motor(4, -1, speed);
  delay(2000);

  Motor(1,1, 0);    //drift front-left
  Motor(2,1, speed);
  Motor(3,1, 0);
  Motor(4,1, speed);
  delay(2000);

  Motor(1,-1, 0);   //drift rear-right
  Motor(2,-1, speed);
  Motor(3,-1, 0);
  Motor(4,-1, speed);
  delay(2000);

  Motor(1,1, speed);        //drift front-right
  Motor(2,1, 0);
  Motor(3,1, speed);
  Motor(4,1, 0);
  delay(2000);

  Motor(1,-1, speed);        //drift rear-left
  Motor(2,-1, 0);
  Motor(3,-1, speed);
  Motor(4,-1, 0);
  delay(2000);

}

// Convert motor speed to PWM value.
void motorPWM(int channel, int motor_speed){
  motor_speed = constrain(motor_speed, 0, 100);
  int motor_pwm = map(motor_speed, 0, 100, 0, 4095);
  if (motor_pwm == 4095){
    pwm_motor.setPWM(channel, 4096, 0);
  }
  else if (motor_pwm == 0){
    pwm_motor.setPWM(channel, 0, 4096);
  }
  else{
    pwm_motor.setPWM(channel, 0, motor_pwm);
    // pwm_motor.setPWM(channel, 0, 4095 - motor_pwm);
  }
}

// Control motor rotation.
void Motor(int Motor_ID, int dir, int Motor_speed){
  if(dir > 0){dir = 1;}
  else {dir = -1;}

  if (Motor_ID == 1){
    if (dir == 1){
      motorPWM(PIN_MOTOR_M1_IN1, 0);
```

```
113         motorPWM(PIN_MOTOR_M1_IN2, Motor_speed);
114       }
115       else{
116         motorPWM(PIN_MOTOR_M1_IN1, Motor_speed);
117         motorPWM(PIN_MOTOR_M1_IN2, 0);
118         }
119     }
120     else if (Motor_ID == 2){
121       if (dir == 1){
122         motorPWM(PIN_MOTOR_M2_IN1, 0);
123         motorPWM(PIN_MOTOR_M2_IN2, Motor_speed);
124       }
125       else{
126         motorPWM(PIN_MOTOR_M2_IN1, Motor_speed);
127         motorPWM(PIN_MOTOR_M2_IN2, 0);
128         }
129     }
130     else if (Motor_ID == 3){
131       if (dir == 1){
132         motorPWM(PIN_MOTOR_M3_IN1, 0);
133         motorPWM(PIN_MOTOR_M3_IN2, Motor_speed);
134       }
135       else{
136         motorPWM(PIN_MOTOR_M3_IN1, Motor_speed);
137         motorPWM(PIN_MOTOR_M3_IN2, 0);
138         }
139     }
140     else if (Motor_ID == 4){
141       if (dir == 1){
142         motorPWM(PIN_MOTOR_M4_IN1, 0);
143         motorPWM(PIN_MOTOR_M4_IN2, Motor_speed);
144       }
145       else{
146         motorPWM(PIN_MOTOR_M4_IN1, Motor_speed);
147         motorPWM(PIN_MOTOR_M4_IN2, 0);
148         }
149     }
150   }
```

Code explanation

Initialization Stage (setup Function)

Library Loading and Hardware Connection

Load Wire.h (I2C communication library) and Adafruit_PWMServoDriver.h (PCA9685 PWM control library).

Define the positive and negative pins of 4 motors (8 GPIO pins in total) for controlling motor rotation direction.

Main Loop Stage (loop Function)

The loop sequentially executes 10 movement modes, each lasting 2 seconds. The process is as follows:

- Forward movement: All motors rotate forward
- Backward movement: All motors rotate backward
- Left turn: Left motors (M1, M2) rotate backward, right motors (M3, M4) rotate forward
- Right turn: Left motors rotate forward, right motors rotate backward
- Left lateral movement: M1/M3 rotate backward, M2/M4 rotate forward
- Right lateral movement: M1/M3 rotate forward, M2/M4 rotate backward
- Front-left drift: M2/M4 rotate forward, M1/M3 stop
- Rear-right drift: M2/M4 rotate backward, M1/M3 stop
- Front-right drift: M1/M3 rotate forward, M2/M4 stop
- Rear-left drift: M1/M3 rotate backward, M2/M4 stop