



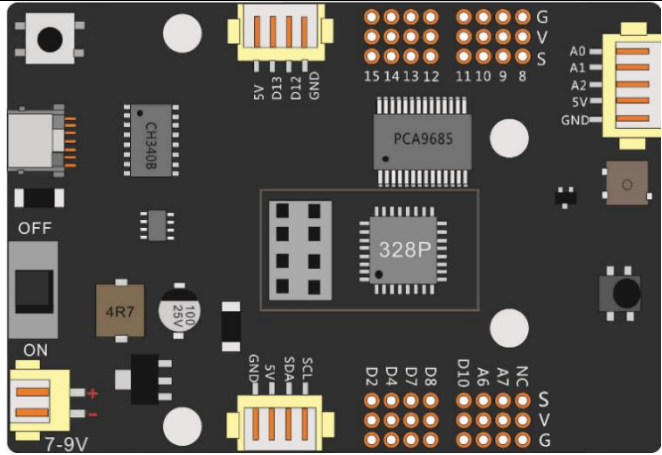
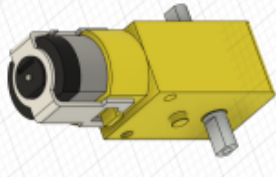
Lesson 10 Test the DC Motors

Table

1.What do you need to prepare	1
2. Introducing DC Motors.....	2
3. Introducing DRV8833	4
4. Introducing PCA9685.....	5
5. Principle.....	8
6. Wiring.....	9
7. Upload the code and test	10
8. Code.....	13
9.Any questions and suggestions are welcome	15

1. What do you need to prepare

Components	Quantity	Picture
Battery box with 2pcs 18650 batteries	1	
USB Cable	1	

Control board	1	
DC Motor	4	

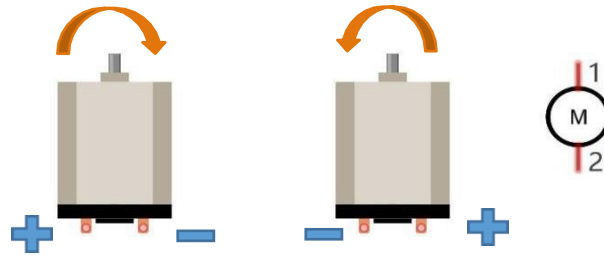
Remak: Two 18650 batteries are not included in this kit, please prepared by yourself.

2. Introducing DC Motors

Our products use DC motor as a power device. A DC motor is a device that converts DC electrical energy into mechanical energy. Widely used to drive various equipment, such as electric fans, remote control cars, electric windows, etc. The DC motor is very suitable as the moving mechanism of the trolley.

When motor is connected to the power supply, it will rotate in one direction. Reverse the polarity of power supply, the motor will rotate in the opposite direction.

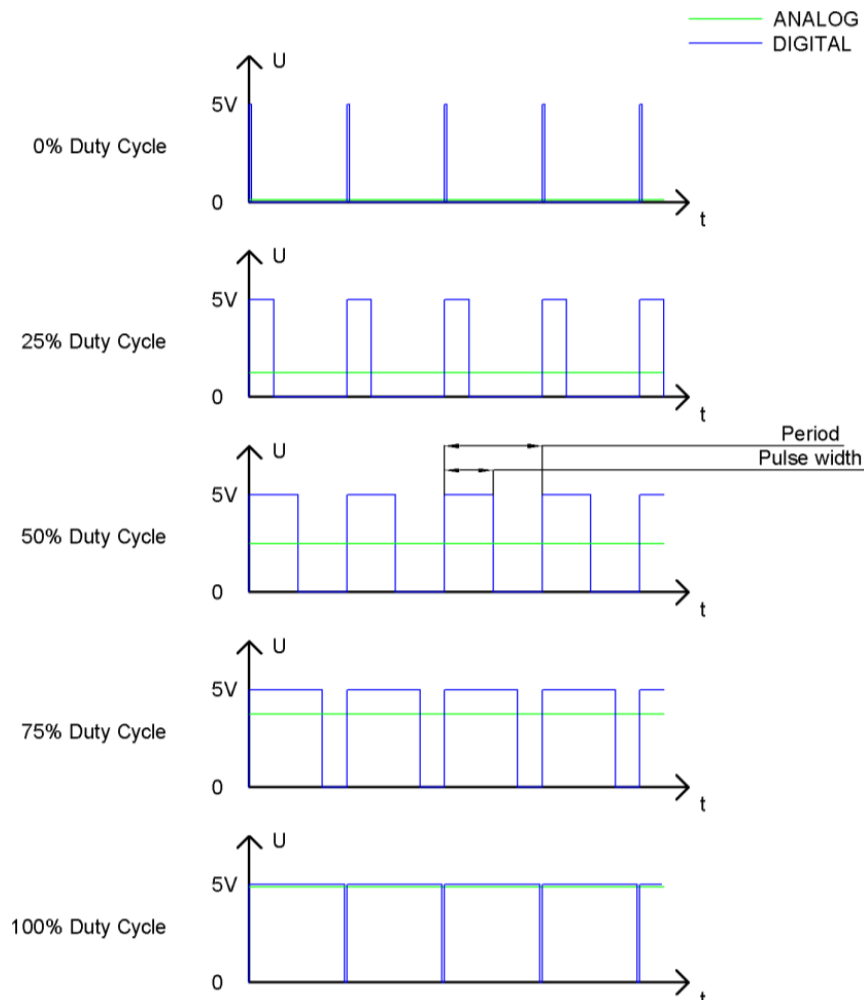
And the speed of motor depends on the voltage between two ends. The larger the voltage, the larger the speed.



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



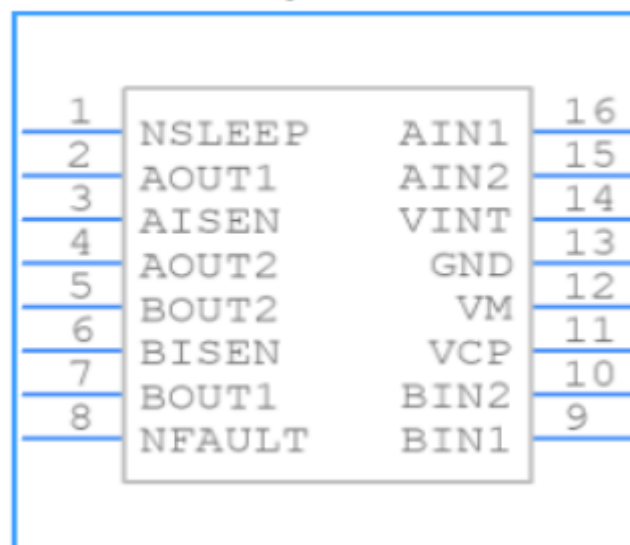
The longer the PWM duty cycle is, the higher the output power will be. Now that we understand this relationship, we can use PWM to control the brightness of an LED or the speed of DC motor and so on.



3. Introducing DRV8833

The DRV8833PWPR is a driver for 2A low voltage Dual brush DC or single bipolar stepper motor (PWM Ctrl). it provides a dual bridge motor driver solution for toys, printers, and other mechatronics applications. The device features two H-bridge drivers that can drive two DC brush motors, a bipolar stepper motor, solenoids or other inductive loads. The output driver module of each H-bridge consists of N-channel power MOSFETs configured as H-bridges to drive the motor windings. Each H-bridge includes a circuit to regulate or limit the winding current. Internal shutdown function with fault output pins provides overcurrent protection, short circuit protection, undervoltage locking and overheat protection. A low-power sleep mode is also provided.

Symbol

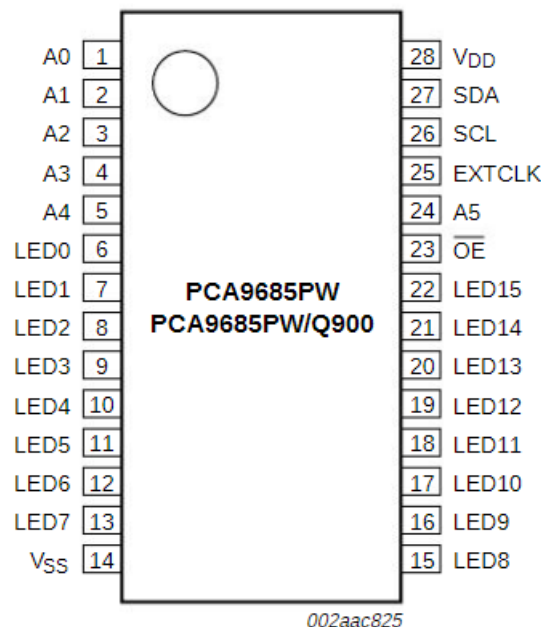


4. Introducing PCA9685

The PCA9685 is an I2C-bus controlled 16-channel LED controller optimized for Red/Green/ Blue/ Amber(RGBA)color backlighting applications.Each LED output has its own 12-bit resolution (4096 steps)fixed frequency individual PWM controller that operates at a programmable frequency from a typical of 24 Hz to 1526 Hz with a duty cycle that is adjustable from 0%to 100%to allow the LED to be set to a specific brightness value.All outputs are set to the same PWM frequency.

Each LED output can be off or on(no PWM control),or set at its individual PWM controller value.The LED output driver is programmed to be either open-drain with a 25 mA current sink capability at 5 V or totem pole with a 25 mA sink,10 mA source capability at 5 V.The PCA9685 operates with a supply voltage range of 2.3 V to 5.5 V and the inputs and outputs are 5.5 V tolerant.LEDs can be directly connected to the LED output (up to 25 mA,5.5 V)or controlled with external drivers and a minimum amount of discrete components for larger current or higher voltage LEDs.The PCA9685 is in the new Fast-mode Plus (Fm+) family. Fm+ devices offer higher frequency (up to 1 MHz) and more densely populated bus operation (up to 4000 pF).

PCA9685 pin



Based on Arduino, we expand PCA9685 on Atmega328p of the control board through I2C connection to make it control servos, LEDs, motors and other equipment.

The I2C address of PCA9685 defaults to 0x40.Arduino Atmega328p----->PCA9685

5V -----> VCC

A4/SDA -----> SDA

A5/SCL -----> SCL

GND -----> GND

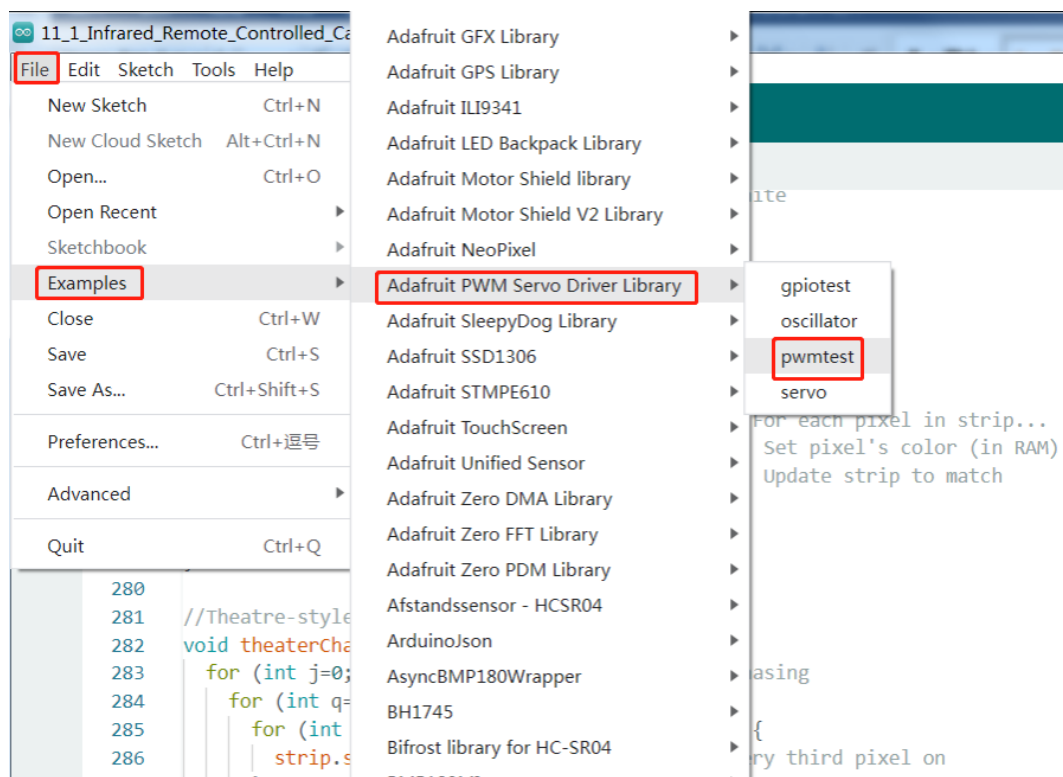
The library corresponding to the PCA9685 module is an external library provided by Adafruit :

Adafruit PWM Servo Driver Library

To install the Adafruit PWM library, please refer to "How to install the library file Servo.h in Lesson4"

After installing the library, click on the toolbar menu: "File" ---"Examples" ---"Adafruit PWM Servo Driver Library" ---"pwm test"

You will see a Built-in Example called [pwm test](#) in the list. Click to open it.





```
31 Serial.println("16 channel PWM test!");
32
33 pwm.begin();
34
35 pwm.setOscillatorFrequency(27000000);
36 pwm.setPWMFreq(1600); // This is the maximum PWM frequency
37
38 // if you want to really speed stuff up, you can go into 'fast 400khz I2C'
39 // some i2c devices dont like this so much so if you're sharing the bus,
40 // out for this!
41 Wire.setClock(400000);
42 }
43
44 void loop() {
45     // Drive each PWM in a 'wave'
46     for (uint16_t i=0; i<4096; i += 8) {
47         for (uint8_t pwmnum=0; pwmnum < 16; pwmnum++) {
48             pwm.setPWM(pwmnum, 0, (i + (4096/16)*pwmnum) % 4096 );
49         }
50     }
51 }
```

There are three main functions in this Arduino sample program:

`pwm.begin();pwm.setPWMFreq(SERVO_FREQ); pwm.setPWM(pwmnum, on, off)`

The `pwm.begin()` function is mainly to initialize the configuration of the IIC pins, and reset the PCA9685 after the configuration is completed, that is, write 0x00 on the MODE1 address. This step is critical, without which the PCA9685 will not work properly.

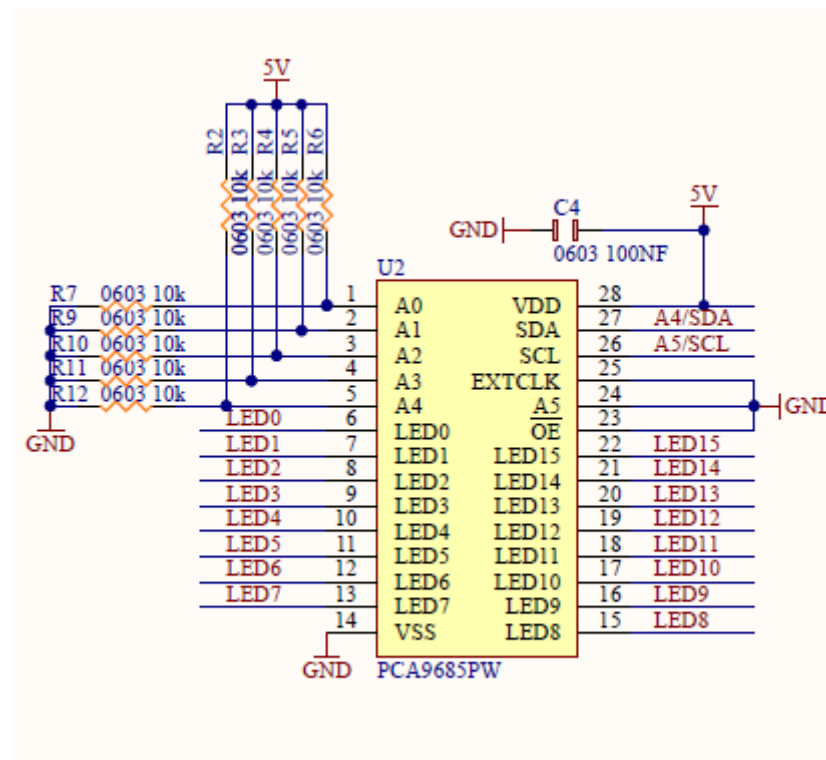
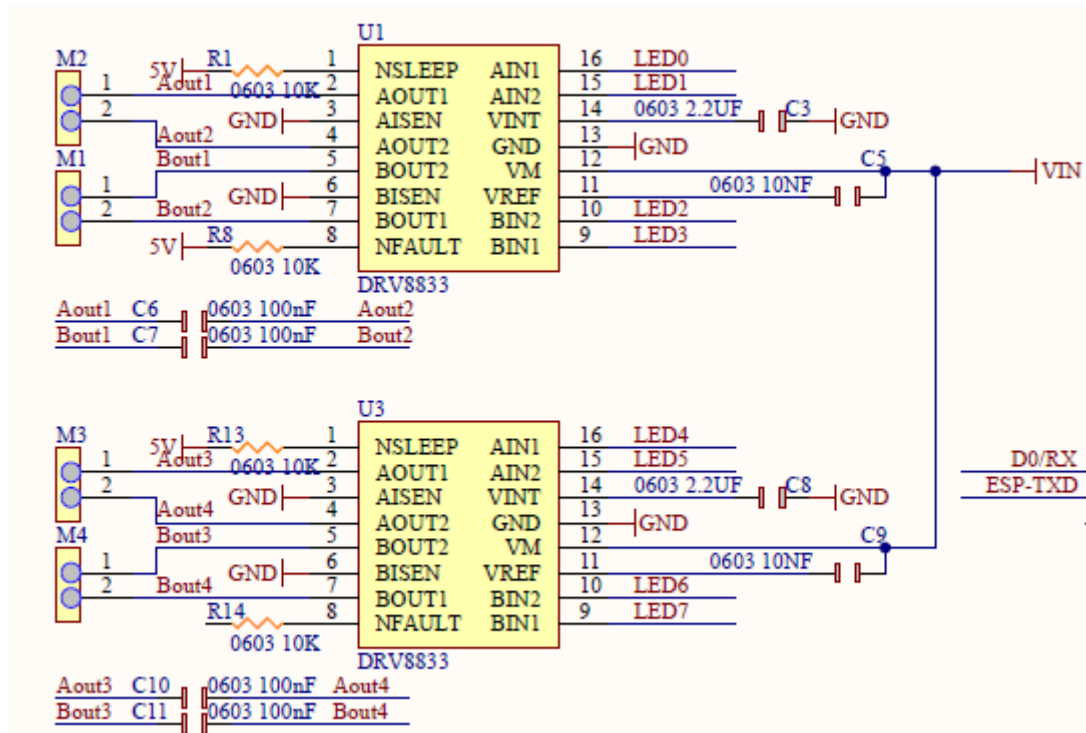
The `pwm.setPWMFreq(SERVO_FREQ)` function is mainly to set the output frequency of PCA9685.

The 16-channel PWM output frequency of PCA9685 is consistent, so it is impossible to achieve different frequencies for different pins.

The `pwm.setPWM(pwmnum, on, off)` function is mainly to adjust the output PWM duty cycle. Usually on is set to 0, changing the value of off can change the output PWM duty cycle. Because PCA9685 has 12-bit resolution, the value of off 0~4096 represents the duty cycle 0-100.

5. Principle

One DRV8833 can control two TT motor, the schematic diagram like below. The driving signal of DRV8833 is provided by PCA9685, and the two-channel PWM signal of PCA9685 is connected to AIN1/AIN2 of DRV8833 to control the speed and direction of the motor.



6. Wiring

There are 4 DC motor ports on the control board: M1, M2, M3, M4. We define the M1 port drive the right back wheel of the robot car, define the M2 port drive the left back wheel of the robot car, define the M3 port drive the right front wheel of the robot car, define the M4 port drive the left front wheel of the robot car.

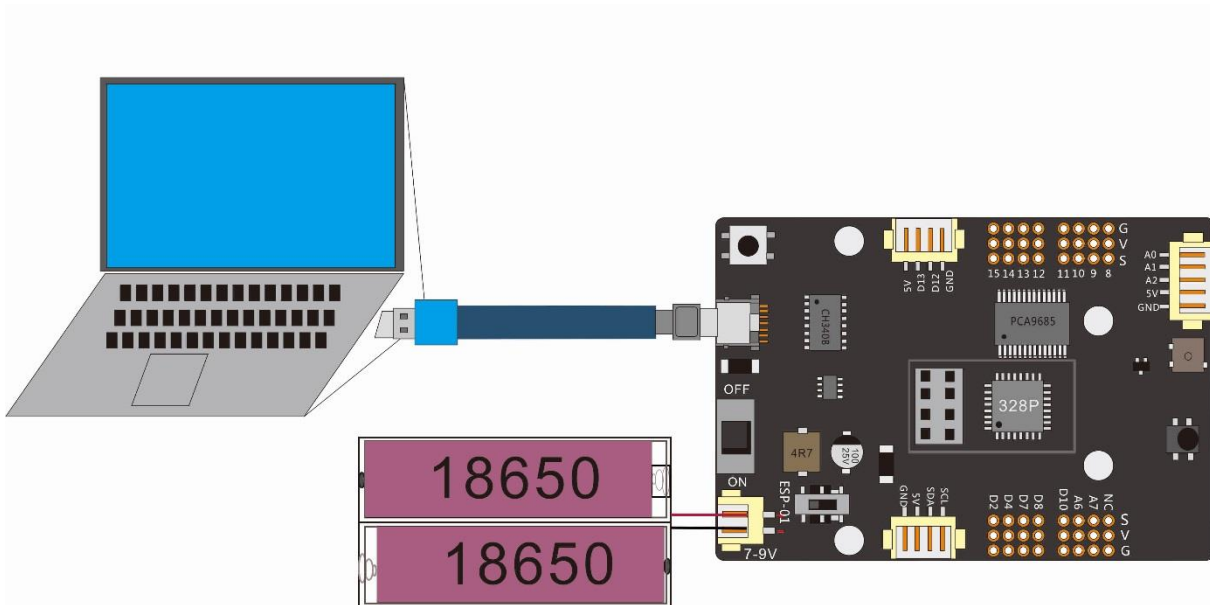
Please connect the circuit as shown below:

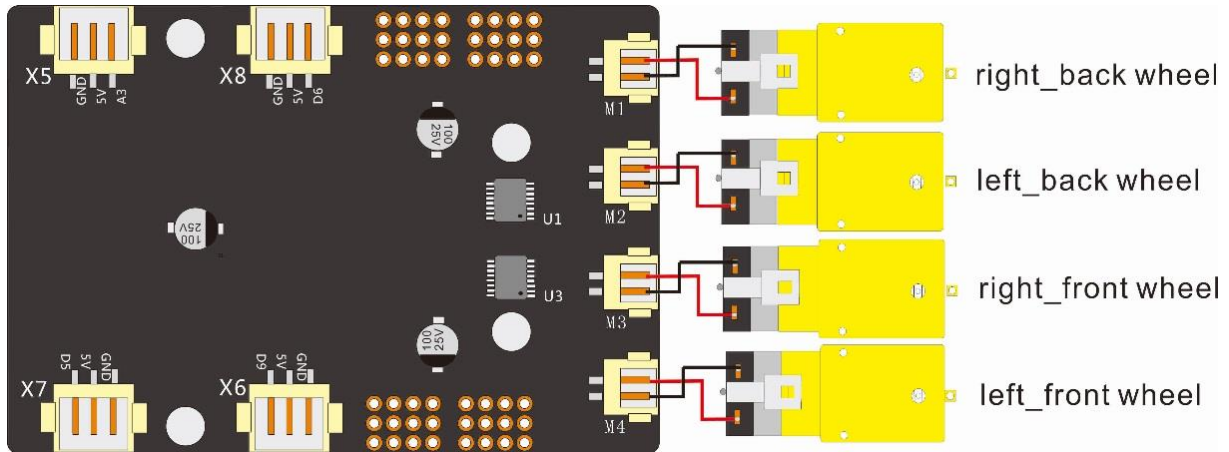
Turn the power switch on the control board to OFF

Connect the four TT motors to the M1,M2,M3, M4 ports of the control board respectively

Install two 18650 batteries in the battery box (the batteries need to be prepared by yourself), and connect them to the power port on the control board

Finally, connect the control board to the computer with a USB cable





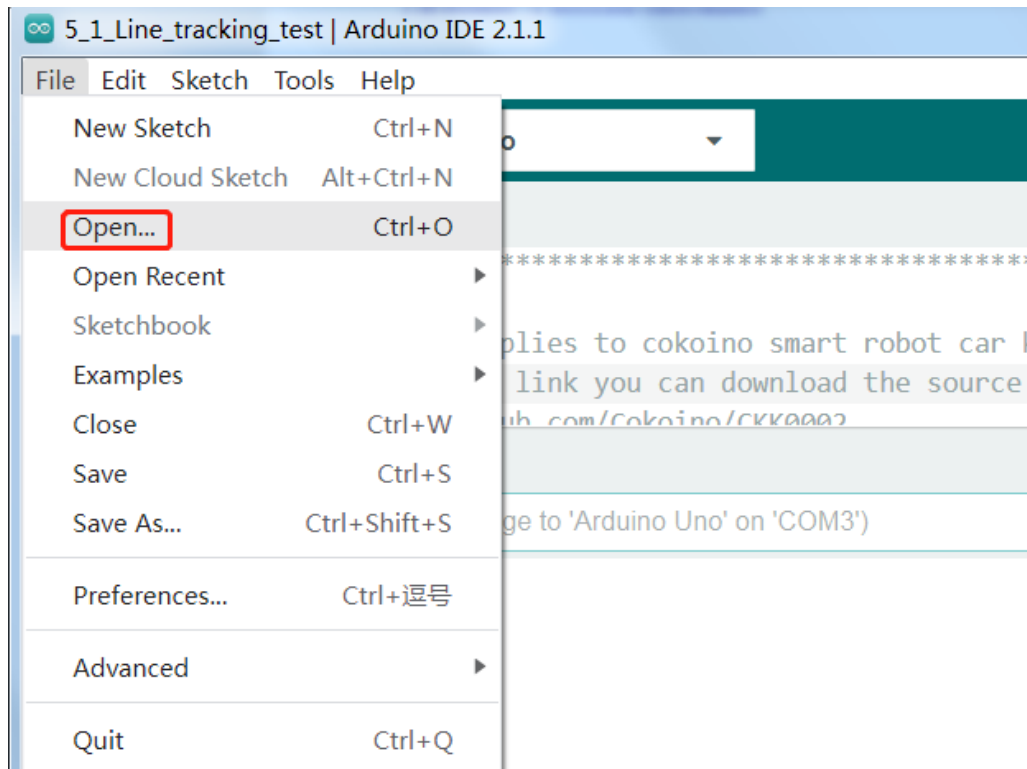
7. Upload the code and test

The code used in this lesson is placed in this folder: “ <E:\CKK0014-main\Tutorial\sketches>”

7.1 Double-click the Arduino IDE shortcut on the desktop to open it



7.2 Click "File" --- "open"

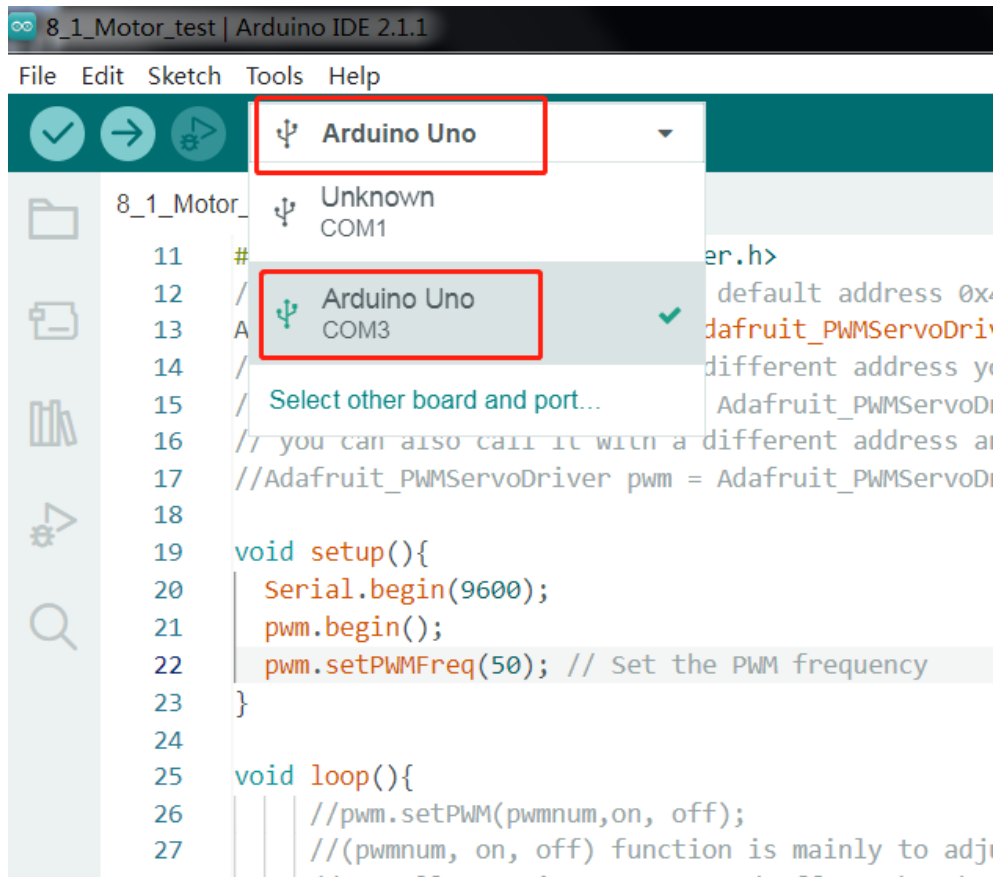


7.3 Select the code in the folder named **8_1_Motor_test**:

E:\CKK0014-main\Tutorial\sketches\8_1_Motor_test

Then click "open"

7.4 Select the board "Arduino UNO" and Port "COM3" (COM port is commonly known as an input output port for a device normally PC which enables communication between Arduino and PC. You can check your arduino com number in device manager, the com port of our arduino board is recognized as COM3 in this tutorial)



7.5 Install Adafruit_PWMServoDriver library

For the installation method, please refer to the method of installing the library Servo.h in Lesson 4

7.6 Click compile button , successfully compiled the code will display “Done compiling”

7.7 Before uploading the code, turn the ESP-01 switch on the control board to the side away from the "ESP-01" silk screen.

7.8 Click upload button , successfully uploading the code will display “Done uploading”.

When code is uploaded successfully, the program starts to run.

7.9 Turn the power switch on the control board to the ON state, and then you will see that the motor starts to rotate, forward for 2 seconds---reverse for 2 seconds---stop for 0.5 seconds, and then cycle.

8. Code

8_1_Motor_test.ino

```
#include <Wire.h>
#include <Adafruit_PWMServoDriver.h>
// called this way, it uses the default address 0x40
Adafruit_PWMServoDriver pwm = Adafruit_PWMServoDriver();
// you can also call it with a different address you want
//Adafruit_PWMServoDriver pwm = Adafruit_PWMServoDriver(0x41);
// you can also call it with a different address and I2C interface
//Adafruit_PWMServoDriver pwm = Adafruit_PWMServoDriver(0x40, Wire);

void setup(){
  Serial.begin(9600);
  pwm.begin();
  pwm.setPWMFreq(50); // Set the PWM frequency
}

void loop(){
  //pwm.setPWM(pwmnum,on, off);
  //(pwmnum, on, off) function is mainly to adjust the output PWM duty cycle.
  // Usually, on is set to 0 and off can be changed.
  // Because the PCA9685 is a 12-bit resolution
  // the value of 0 to 4096 off represents a duty cycle of 0 to 100.
  //*****
  // drive M1 Motror forward
  pwm.setPWM(2,0,4000); //set pwm signal to BIN2 of DRV8833
  pwm.setPWM(3,0,-4000); //set pwm signal to BIN1 of DRV8833
  // drive M2 Motror forward
  pwm.setPWM(0,0,-4000); //set pwm signal to BIN2 of DRV8833
  pwm.setPWM(1,0,4000); //set pwm signal to BIN1 of DRV8833
  // drive M3 Motror forward
  pwm.setPWM(6,0,4000); //set pwm signal to BIN2 of DRV8833
  pwm.setPWM(7,0,-4000); //set pwm signal to BIN1 of DRV8833
  //drive M4 Motror forward
  pwm.setPWM(4,0,-4000); //set pwm signal to AIN1 of DRV8833
  pwm.setPWM(5,0,4000); //set pwm signal to AIN2 of DRV8833
  delay(2000);
  // drive M1 Motror backward
  pwm.setPWM(2,0,-4000); //set pwm signal to BIN2 of DRV8833
  pwm.setPWM(3,0,4000); //set pwm signal to BIN1 of DRV8833
  // drive M2 Motror backward
```

```
pwm.setPWM(0,0,4000);//set pwm signal to BIN2 of DRV8833
pwm.setPWM(1,0,-4000);//set pwm signal to BIN1 of DRV8833
// drive M3 Motror backward
pwm.setPWM(6,0,-4000); //set pwm signal to BIN2 of DRV8833
pwm.setPWM(7,0,4000);//set pwm signal to BIN1 of DRV8833
// drive M4 Motror backward
pwm.setPWM(4,0,4000);//set pwm signal to AIN1 of DRV8833
pwm.setPWM(5,0,-4000);//set pwm signal to AIN2 of DRV8833
delay(2000);
//set M1/M2/M3/M4 Motor stop
pwm.setPWM(0,0,0);//set pwm signal to BIN2 of DRV8833
pwm.setPWM(1,0,0);//set pwm signal to BIN1 of DRV8833
pwm.setPWM(2,0,0);//set pwm signal to BIN2 of DRV8833
pwm.setPWM(3,0,0);//set pwm signal to BIN1 of DRV8833
pwm.setPWM(4,0,0);//set pwm signal to AIN1 of DRV8833
pwm.setPWM(5,0,0);//set pwm signal to AIN2 of DRV8833
pwm.setPWM(6,0,0);//set pwm signal to BIN2 of DRV8833
pwm.setPWM(7,0,0);//set pwm signal to BIN1 of DRV8833
delay(500);
//*****
```

}

9.Any questions and suggestions are welcome

Thank you for reading this document!

If you find any errors and omissions in the tutorial, or if you have any suggestions and questions, please feel free to contact us:
cokoino@outlook.com

We will do our best to make changes and publish revisions as soon as possible.

If you want to learn more about smart cars, robots, learning kits and other technology products from us, please bookmark and pay attention to our website:

<http://cokoino.com/>

We will continue to launch interesting, cost-effective, innovative, user-friendly products.

LK COKOINO