

Lesson 13 Infrared Remote Controlled Car

Table

1. Preface.....	1
2. What do you need to prepare.....	2
3. Upload the code.....	2
4. Control the robot with an infrared remote control.....	3
5. Troubleshooting.....	4
6. Code.....	5
7. Any questions and suggestions are welcome.....	14

1. Preface

In the previous lessons, we have learned the basic knowledge of buzzer, servo, ultrasonic module, WS2812 LED, line tracking sensor, LCD1602 display, DC Motor and the IR remote. We also learned how to use the control board to control them or read the data of them. Through these example experiments, we also tested and confirmed whether the modules in this suite can work normally.

In the last lesson, we have assembled the robot. Next let's combine these modules to work with the control board, uploading more complex code to command the robot to make it complete various tasks.

2. What do you need to prepare

Assembled robot x 1

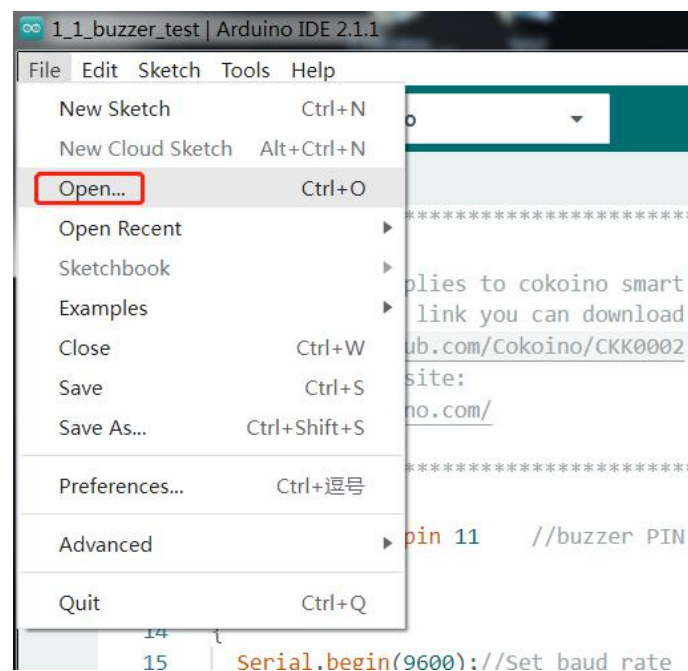
(How to assemble the robot? Please refer to "Lesson12 Assemble the Smart Robot Car")

USB cable x 1

18650 battery X 2

3. Upload the code

3.1 Click "File" --- "open" on the Arduino IDE





3.2 the code used in this lesson [11_1_Infrared_Remote_Controlled_Car](#) is placed in this folder:

[E:\CKK0002-master\Tutorial\sketches\11_1_Infrared_Remote_Controlled_Car](#)

Click to open it

3.3 Select the board "Arduino UNO" and Port "COM3"

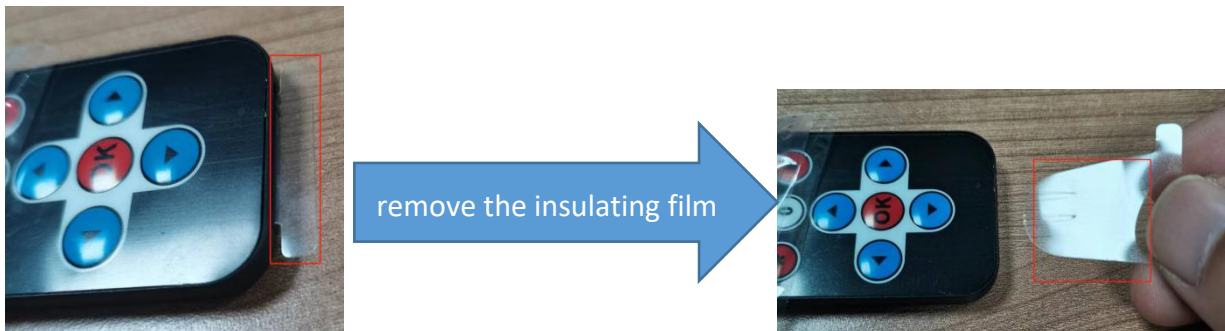
3.4 Click compile button , successfully compiled the code will display "Done compiling"

3.5 Click upload button , successfully uploading the code will display “Done uploading”. When code is uploaded successfully, the program starts to run.

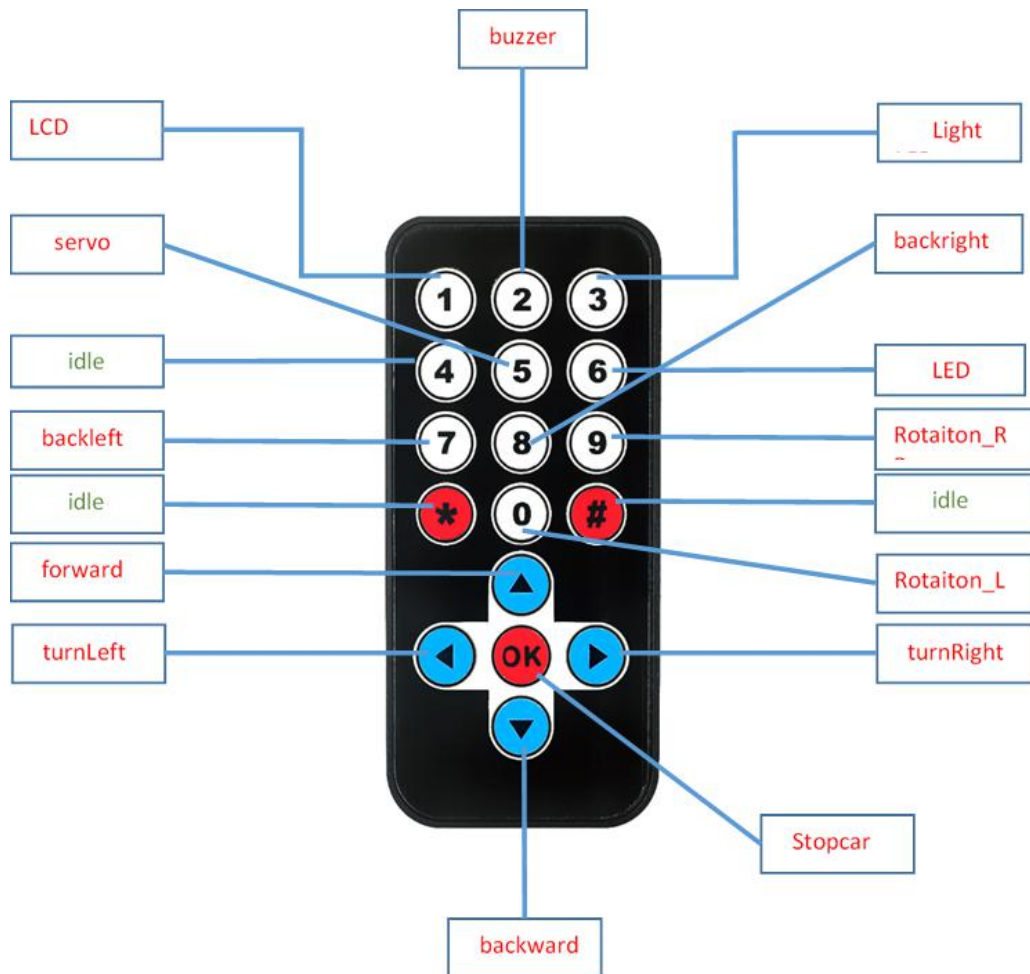
4. Control the robot with an infrared remote control

After the code is uploaded to the control board, turn the power switch of the control board to ON. Place the Smart Robot Car on a flat ground, point the transmitter end of the infrared remote control to the infrared receiver on the car, press the button on the infrared remote control, and the car will execute different commands.

Note: The built-in button battery of the infrared remote control is pasted with an insulating film. Be sure to remove the insulating film of the battery before using it, otherwise the infrared remote control will not work normally.



The action of the car corresponding to the button of the infrared remote control is as follows:



5. Troubleshooting

5.1 Unable to upload code successfully

Before uploading the code, please check whether the ESP-01 switch on the control board is turned to the side away from the "ESP-01" silk screen

5.2 Press the infrared remote control button, the car does not move

Check whether the insulating film of the built-in battery of the infrared remote control is removed?

Is there any obstacle between the infrared remote control and the car?

5.3 Nothing displays on the LCD display

Check whether the 4pin wires of the LCD and the 4pin wires of the ultrasonic module are mixed?

5.4 The servo does not turn

Check whether the servo is connected to the D10 pin header of the control board

Check whether the wiring sequence of the servo connected to the D10 pin is connected incorrectly

5.5 When running the car, the wheels do not rotate or rotate very slowly.

Please check if the battery level is low? If the battery level is below 7V, it is recommended to charge it before use. If the battery level is between 7~7.4V, you can try increasing the motor speed in the code. The battery level matched with the motor speed in the example code is between 7.4V and 8.4V.

6. Code

Usually, there are two basic main functions for Arduino code, `void setup()` and `void loop()`.

```
void setup(){ }
```

The `setup()` function is called when a sketch starts, which is used to initialize variables, pin modes, start using libraries, etc.

The `setup()` function will *only run once*, after each *power up* or *reset* of the Arduino board.

```
void loop(){ }
```

This function will loop consecutively. Code in this function will be executed again and again...

There are some other functions integrated by Arduino.

```
pinMode(pin, mode)
```

Configures the specified pin to behave either as an input or an output.

Parameters

pin: Arduino pin number to set the mode of.

mode: INPUT, OUTPUT, or INPUT_PULLUP.

```
digitalWrite(pin, value)
```

Write a HIGH or a LOW value to a digital pin.

If the pin has been configured as an OUTPUT with pinMode(), its voltage will be set to the corresponding

value: 5V (or 3.3V on 3.3V boards) for HIGH, 0V (ground) for LOW.

Parameters

pin: Arduino pin number.

value: HIGH or LOW.

analogWrite(pin, value)

Writes an analog value (PWM wave) to a pin.

You do not need to call pinMode() to set the pin as an output before calling analogWrite().

Parameters

pin: Arduino pin to write to. Allowed data types: int.

value: the duty cycle: between 0 (always off) and 255 (always on). Allowed data types: int.

For more details, please refer to: <https://www.arduino.cc/reference/>

11_1_Infrared_Remote_Controlled_Car.ino

```
/******
```

```
* This code applies to cokoino smart robot car kit
* Through this link you can download the source code:
* https://github.com/Cokoino/CKK0002
* Company web site:
* http://cokoino.com/
```

```
*****/
```

```
/******
```

smart robot car

```
-----| |-----
-----| |-----
      M1 (-----) M4
```

```
*****/
```

```
#include <SCoop.h> //Import multithread library
```

```
#include <IRremote.h> //IR library
#define RECV_PIN 3 //The infrared control is defined as D3
IRsend irsend;
IRrecv irrecv(RECV_PIN);
decode_results results;

#include <Wire.h>
#include <Adafruit_PWMServoDriver.h>
Adafruit_PWMServoDriver pwm = Adafruit_PWMServoDriver();

#include <Servo.h>
Servo carservo;
int pos=0;

#include <Adafruit_NeoPixel.h>
#ifdef __AVR__
#include <avr/power.h>
#endif
#define WS2812_PIN 6 //WS2812 PIN
#define WS2812_COUNT 12 // How many NeoPixels are attached to the Arduino?
#define BRIGHTNESS 10 // NeoPixel brightness, 0 (min) to 255 (max)
// Declare our NeoPixel strip object:
Adafruit_NeoPixel strip = Adafruit_NeoPixel(WS2812_COUNT, WS2812_PIN, NEO_GRB + NEO_KHZ800);

#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27,16,2);

#define Buzz 11 //buzzer PIN
#define led_R 9 //LED_R PIN
#define led_L 5 //LED_R PIN
defineTask(TaskOne); // Create subthread 1
defineTask(TaskTwo); // Create subthread 2

////////////////////////////////////
void TaskOne::setup() //Thread 1 setup
{
  Serial.begin(9600);
  pwm.begin();
  pwm.setPWMFreq(50);
  pwm.setPWM(2, 0, 0);
  pwm.setPWM(3, 0, 0);
  pwm.setPWM(4, 0, 0);
}
```

```
pwm.setPWM(5, 0, 0);

lcd.init();
lcd.backlight();
lcd.clear();
lcd.setCursor(3,0);
lcd.print("HELLO WORLD!");
lcd.setCursor(2,1);
lcd.print("HELLO COKOINO!");

pinMode(Buzz, OUTPUT);
pinMode(led_R, OUTPUT);
pinMode(led_L, OUTPUT);

carservo.attach(10);
carservo.write(65);

irrecv.enableIRIn(); // Start the receiver
}

void TaskTwo::setup() { // Thread 2 setup
    strip.begin();
    strip.show();
    strip.setBrightness(BRIGHTNESS);
}
void setup() {
    mySCoop.start();
}

////////////////////////////////////
void TaskOne::loop() //loop subthread 1
{
    if (irrecv.decode(&results))
    {
        Serial.println(results.value, HEX); //Serial print data
        delay(50);
        irrecv.resume(); // Receive the next value
        delay(10);
    }
    switch(results.value) //Jump to the position of the corresponding value
    {
        case 0xff18e7: forward(); results.value=0; break; //Up button of remote control
```



```
case 0xff4ab5 : backward();      results.value=0; break;      //down button of remote control
case 0xff10ef : turnLeft();      results.value=0; break;      //left button of remote control
case 0xff5aa5 : turnRight();     results.value=0; break;     //right button of remote control
case 0xff38c7 : Stopcar();       results.value=0; break;     //OK button of remote control
case 0xFFA25D : LCD();           results.value=0; break;     //1 button of remote control
case 0xFF629D : buzz();          results.value=0; break;     //2 button of remote control
case 0xFFE21D : LED();           results.value=0; break;     //3 button of remote control
case 0xFF02FD : servo();         results.value=0; break;     //5 button of remote control
case 0xFFC23D : LED_Blink();     results.value=0; break;     //6 button of remote control
case 0xFFE01F : backleft();      results.value=0; break;     //7 button of remote control
case 0xFFA857 : backright();     results.value=0; break;     //8 button of remote control
case 0xFF906F : Rotation_R();    results.value=0; break;     //9 button of remote control
case 0xFF9867 : Rotation_L();    results.value=0; break;     //0 button of remote control
default : break;
}
}
////////////////////////////////CONTROL FUNCIONES////////////////////////////////
//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.
void forward()
////The off value is based on the battery level ranging from 7.4V to 8.4V. //////////////////////////////////
////If your battery level is below 7.4V, you can increase the off value to allow the car to move normally.//
{
pwm.setPWM(2, 0, -650);
pwm.setPWM(3, 0, 650);
pwm.setPWM(4, 0, 650);
pwm.setPWM(5, 0, -650);
}

void backward()
{
pwm.setPWM(2, 0, 750);
pwm.setPWM(3, 0, -750);
pwm.setPWM(4, 0, -750);
pwm.setPWM(5, 0, 750);
}

void backright()
{
```

```
pwm.setPWM(2, 0, 1200);
pwm.setPWM(3, 0, -1200);
pwm.setPWM(4, 0, -650);
pwm.setPWM(5, 0, 650);
}
void backleft()
{
pwm.setPWM(2, 0, 650);
pwm.setPWM(3, 0, -650);
pwm.setPWM(4, 0, -1200);
pwm.setPWM(5, 0, 1200);
}
void turnRight()
{
pwm.setPWM(2, 0, -1200);
pwm.setPWM(3, 0, 1200);
pwm.setPWM(4, 0, 650);
pwm.setPWM(5, 0, -650);
}

void turnLeft()
{
pwm.setPWM(2, 0, -650);
pwm.setPWM(3, 0, 650);
pwm.setPWM(4, 0, 1200);
pwm.setPWM(5, 0, -1200);
}
void Rotation_R()
{
pwm.setPWM(2, 0, -1200);
pwm.setPWM(3, 0, 1200);
pwm.setPWM(4, 0, -1200);
pwm.setPWM(5, 0, 1200);
}
void Rotation_L()
{
pwm.setPWM(2, 0, 1200);
pwm.setPWM(3, 0, -1200);
pwm.setPWM(4, 0, 1200);
pwm.setPWM(5, 0, -1200);
}
```

```
void Stopcar()
{
  pwm.setPWM(2, 0, 0);
  pwm.setPWM(3, 0, 0);
  pwm.setPWM(4, 0, 0);
  pwm.setPWM(5, 0, 0);
}

void servo()
{
  for (pos = 0; pos <= 150; pos += 1) { // goes from 0 degrees to 150 degrees
    // in steps of 1 degree
    carservo.write(pos);           // tell servo to go to position in variable 'pos'
    delay(15);                     // waits 15 ms for the servo to reach the position
  }
  for (pos = 150; pos >= 0; pos -= 1) { // goes from 150 degrees to 0 degrees
    carservo.write(pos);           // tell servo to go to position in variable 'pos'
    delay(15);                     // waits 15 ms for the servo to reach the position
  }
  delay(50);
  carservo.write(65);
}

void LCD()
{
  lcd.setCursor(1,1);
  lcd.print("Robot car testing");
}

void buzz()
{
  for(int i = 0; i < 100; i++)
  {
    digitalWrite(Buzz, HIGH);
    delay(1);
    digitalWrite(Buzz, LOW);
    delay(1);
  }
  for(int j = 0; j < 180; j++)
  {
    digitalWrite(Buzz, HIGH);
```

```
    delay(2);
    digitalWrite(Buzz, LOW);
    delay(2);
  }
}

void LED()
{
  digitalWrite(led_R,HIGH);
  digitalWrite(led_L,HIGH);
}

void LED_Blink()
{
  for(int i = 0;i < 15; i++)
  {
    digitalWrite(led_R,HIGH);
    digitalWrite(led_L,HIGH);
    delay(50);
    digitalWrite(led_R,LOW);
    digitalWrite(led_L,LOW);
    delay(50);
    digitalWrite(led_R,HIGH);
    digitalWrite(led_L,HIGH);
    delay(50);
  }
}

//WS2812 TEST
void TaskTwo::loop() //loop subthread 2
{
  colorWipe(strip.Color(255, 0, 0), 10); // Red
  delay(800);
  colorWipe(strip.Color(255, 150, 0), 10); // yellow
  delay(800);
  colorWipe(strip.Color(0, 255, 0), 10); // Green
  delay(800);
  colorWipe(strip.Color(0, 255, 255), 10); // CYAN
  delay(800);
  colorWipe(strip.Color(0, 0, 255), 10); // Blue
  delay(800);
  colorWipe(strip.Color(180, 0, 255), 10); // purple
}
```

```
    delay(800);
    colorWipe(strip.Color(127, 127, 127), 10); // White
    delay(800);
    colorWipe(strip.Color(0, 0, 0), 30); // Clear
    delay(100);
}

void colorWipe(uint32_t c, uint8_t wait)
{
    for(uint16_t i=0; i<strip.numPixels(); i++) { // For each pixel in strip...
        strip.setPixelColor(i, c);                // Set pixel's color (in RAM)
        strip.show();                               // Update strip to match
        delay(wait);
    }
}

//Theatre-style crawling lights.
void theaterChase(uint32_t c, uint8_t wait) {
    for (int j=0; j<10; j++) { //do 10 cycles of chasing
        for (int q=0; q < 3; q++) {
            for (int i=0; i < strip.numPixels(); i=i+3) {
                strip.setPixelColor(i+q, c);    //turn every third pixel on
            }
            strip.show();

            delay(wait);

            for (int i=0; i < strip.numPixels(); i=i+3) {
                strip.setPixelColor(i+q, 0);    //turn every third pixel off
            }
        }
    }
}

void loop(){
    yield(); //loop multithread tsak
}
```

7. 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