# Basic course ---9.IR control

**!Note:**

**1. In order to avoid the interference of sunlight on infrared sensor, we need to carry out this experiment indoors.**

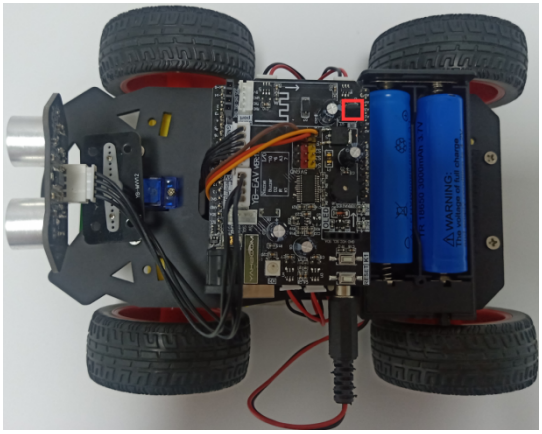
**2. During remote control, the infrared remote control needs to face the infrared receiver on the expansion board.**

## 1. Learning goal

In this lesson, we will learn how to control the car by infrared controller.

## 2. Preparation

2.1 The position of the infrared receiver on the robot car. As shown below.



2.2 The pin of UNO board is connected the infrared receiver.

## 3. Principle of experimental

3.1 From the hardware interface manual, we can know that infrared receiver are driven by Uno board.

| Classification | Function | The number of Drive chip PCA9685 | Drive Method | Connection with CPU | Uno board |
|---|---|---|---|---|---|
| Left Motor | Left front motor forward | LINB(13) | PCA9685 | I2C_SDA/I2C_SCL | A4/A5 |
| | Left front motor reverse | LINA(12) | | | |
| | Left rear motor forward | RINB(15) | | | |
| | Left rear motor reverse | RINA(14) | | | |
| Right Motor | Right front motor forward | LED10 | | | |
| | Right front motor reverse | LED11 | | | |
| | Right rear motor forward | LED8 | | | |
| | Right rear motor reverse | LED9 | | | |
| Servo | Control S1 | LED0 | | | |
| | Control S2 | LED1 | | | |
| | Control S3 | LED2 | | | |
| | Control S4 | S1(3) | | | |
| LOGO light | Control bluelight | LED7 | | | |
| Tarcking sensor | Left tracking sensor | Uno board drive directly | | | A0 |
| | Middle tracking sensor | | | | A1 |
| | Right tracking sensor | | | | A2 |
| Ultrasonic sensor | Ultrasonic Echo | | | | 12 |
| | Ultrasonic RGB light | | | | 11 |
| Key | K1 | | | | 7 |
| IR | IR control | | | | A3 |
| Bluetooth interface | RX | | | | 0 |
| | TX | | | | 1 |
| On board RGB Light | RGB Light on expansion board | | | | 6 |
| Buzzer | Control buzzer | | | | 10 |

### 3.2 Introduction of infrared remote controller :

This remote controller with a standard **38Khz** modulation frequency to accommodate a variety of infrared receivers on the market. It possess 0~9 number keys, 4 direction keys, acceleration and deceleration, left and right rotation, lighting, sound and other buttons, which can be easily applied to various smart cars and development boards. Built-in universal 3V button battery, long time and easy to replace.

### 3.3 Introduction of infrared receiver :

This infrared receiver comes with an iron-shell shield cover to prevent external interference(Some infrared receivers do not have). Built-in dedicated IC, with infrared remote control, can be applied to audio-visual equipment, home appliances, microcontroller learning, robot remote control and so on.

### 3.4 Introduction of infrared:

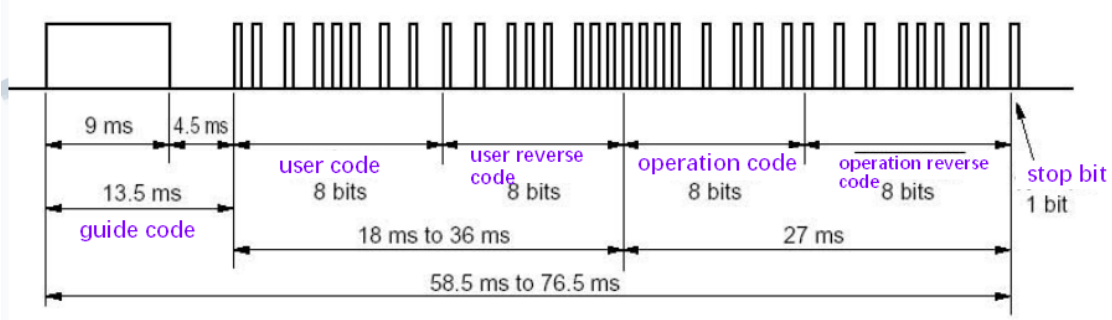
Electromagnetic waves having a wavelength from 760 nm to 400 um in the spectrum are called infrared rays, which is an invisible light. At present, almost all video and audio equipment can be remotely controlled by infrared remote

control, such as televisions, air conditioners, DVD players, etc., and the infrared remote control can be seen.

### 3.5 Introduction to the principle of infrared communication

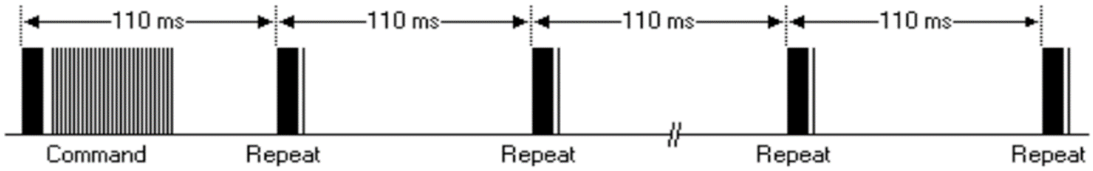

Infrared sending and receiving codes are divided into: **guide code**, **user code**, **user reverse code**, **operation code(data code)**, **operation reverse code(data reverse code).**
Below we will introduce these kinds of encoded waveform:
For the part of infrared transmission, the infrared remote control we provided has been set up. Therefore, we do not explain the knowledge about infrared transmission here, but mainly explain some points of infrared reception.

We mainly use the **NCE** protocol:

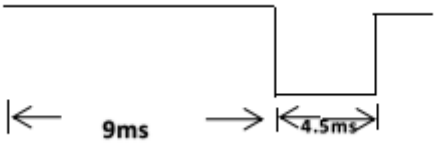



The protocol stipulates that the lower bits are transmitted first.
A string of information first sends a 9ms high pulse of AGC (Automatic Gain Control) . Then, send a 4.5ms low level. Next, send four-byte address code and a command code. These four bytes are : Address code, address reverse code; command code; command reverse code.

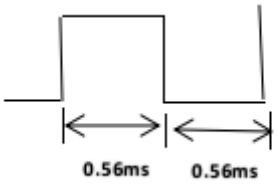
Note:If you keep pressing that button, a series of messages can only be sent once. If you keep pressing, the repeating code is sent with a period of 110ms. Waveform as shown below.

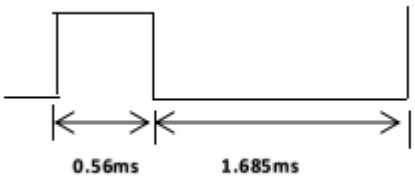



1)**Guide code:**The guide code defined by the uPD6121G of the NEC protocol is 9ms high level+ 4.5ms low level, waveform as shown in the figure below:



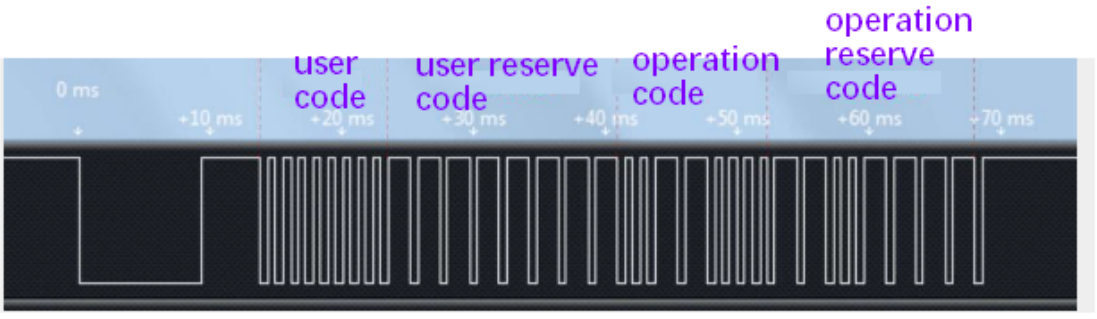

2)**User code and operation code**: The user code and operation code defined by the uPD6121G of the NEC protocol is:
"Logic 0" : 0.56ms high level + 0.565ms low level;
"Logic1" : 0.56ms high level + 1.685ms low level;
Waveform as shown below :




Logic 0 --waveform Logic 1--waveform

The wave output after passing through the infrared receiver, as shown below:




The process of infrared receiving and controlling the car is as follows:
1) Generate a falling edge, enter the interrupt function of external interrupt 0, and check whether the IO port is still low level after delay time. If it is low level, wait for the low level of 9ms to pass. Wait for the 9ms low level to pass, and

then wait for the 4.5ms high level to pass.
2) Start receiving the 4 sets of data transmitted, wait for the low level of 560us to pass, and detect the duration of the high level. If it exceeds 1.12ms, it is high level (the duration of the high level is 1.69ms, the low level Duration is 5.65ms.)
3) Detect whether the received data is the same as the data reverse code (operation reverse code), and wait for the same data. If it is the same, it is proved that the corresponding data sent by the infrared remote control is received. Then, the car moves according to the function set in the program.

**4. About code**

For the code of this course, please refer to: **IR_Control_Car.ino** in the **IR_Control_Car** folder.

```
#include <Arduino.h>
#include <Adafruit_PWMServoDriver.h>
#include <Adafruit_NeoPixel.h>
#include <IRremote.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#include "RGBLed.h"

#define RGB_GREEN       0xFF0000       //Define different
color(green,red,blue)
#define RGB_RED     0x00FF00
#define RGB_BLUE       0x0000FF
#define RGB_YELLOW    0xFFFF00
#define RGB_PURPLE    0x00FFFF
#define RGB_WHITE     0xFFFFFF
#define RGB_CYAN     0xFF00FF
#define RGB_OFF    0x000000
#define SERVOMIN 150          // this is the 'minimum' pulse length count (out
of 4096)
#define SERVOMAX 600          // this is the 'maximum' pulse length count (out
of 4096)
const int RgbPin = 11;       //Define pin of Ultrasonic RGB light
RGBLed mRgb(RgbPin,2);
#define PIN 6                //Define the pins of the RGB light
#define MAX_LED 1       //Just one RGB light on the car
Adafruit_NeoPixel strip = Adafruit_NeoPixel(MAX_LED, PIN, NEO_RGB +
NEO_KHZ800);
Adafruit_PWMServoDriver pwm = Adafruit_PWMServoDriver(0x40);

/*Car running status enumeration*/
const typedef enum {
```

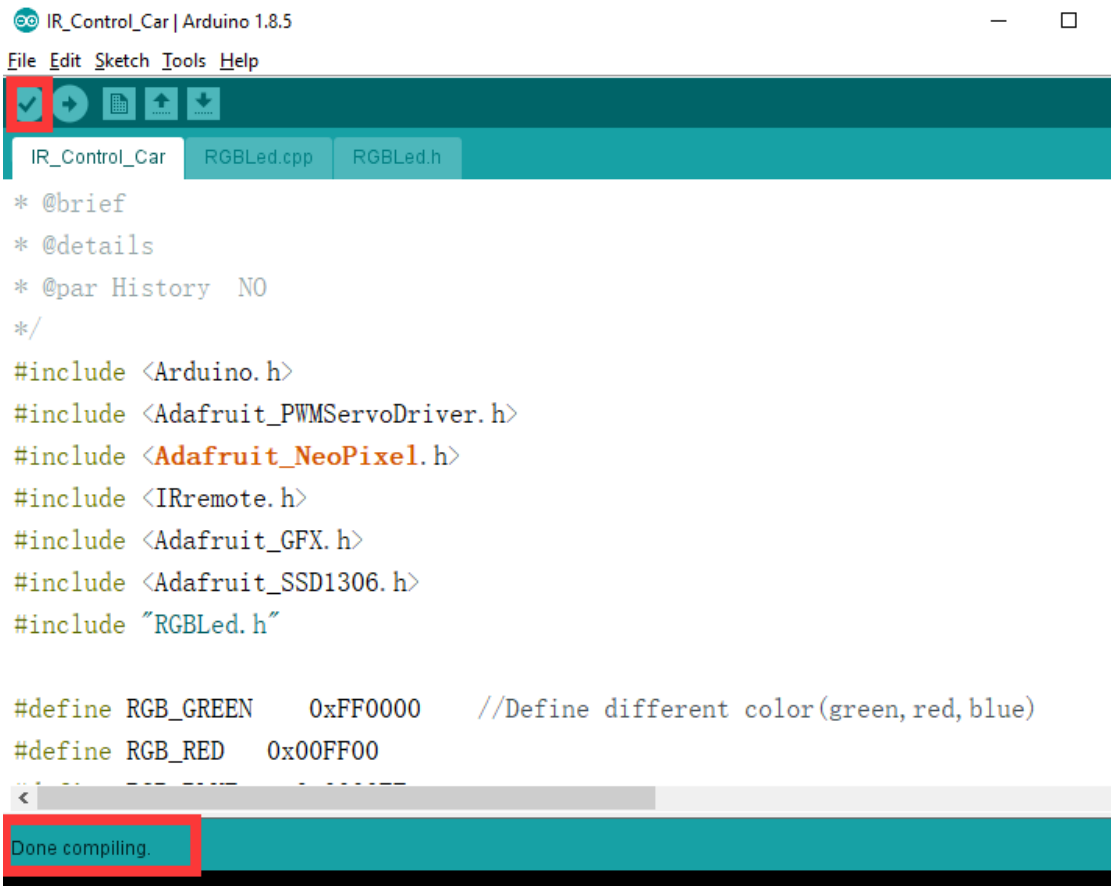
```
    enRUN = 1,
    enBACK,
    enLEFT,
    enRIGHT,
    enSPINLEFT,
    enSPINRIGHT,
    enSTOP
} enCarState;

const typedef enum {
    enServoL = 1,
    enServoR,
    enServoS
} enServoState;

const char enServo[] = {0, 1, 2, 3};

const int RECV_PIN = A3; //Infrared integrated receiver is connected to the A3
on the UNO
IRrecv irrecv(RECV_PIN);
decode_results results;
unsigned long last = millis();
int Servo_LR = 90;
static int CarSpeedControl = 100;
static int g_carstate = enSTOP;
static int g_ServoState = enServoS; //1Left 2Right 3Stop
const int buzzer = 10; //Buzzer is connected to the 10 on the UNO
.....
void loop()
{
  // put your main code here, to run repeatedly:
  if (irrecv.decode(&results))
  {
    switch (results.value)
    {
    //Power switch button
    case 0X00FF00FF:
                        Onboard_RGB_OFF();
                        Ultrasonic_RGB_OFF();
                        break;
    //Light button
    case 0x00FF40BF:
                        Onboard_RGB(255,255,255);
                        mRgb.setColor(0,RGB_WHITE);
```

```
                                Rgb.show();
                              break;
.....
      //Beep button
      case 0x00FFA05F: whistle();     break;
      //Num 7 button
      case 0x00ff18e7: g_ServoState = enServoL;    break;
      //Num 8 button
      case 0x00ff9867:    break;
      //Num 9 button
      case 0x00ff58a7: g_ServoState = enServoR;    break;
      //Advance button
      case 0x00FF807F: g_carstate = enRUN;    break;
      //Left button
      case 0x00FF20DF: g_carstate = enLEFT; break;
      //Right button
      case 0x00FF609F: g_carstate = enRIGHT; break;
      //Back button
      case 0x00FF906F: g_carstate = enBACK; break;
      //Spin left button
      case 0x00FF10EF: g_carstate = enSPINLEFT; break;
      //Spin right button
      case 0x00FF50AF: g_carstate = enSPINRIGHT;break;
      default:    break;
      }
      last = millis();
      irrecv.resume();       //Receive the next code
   }
   else if (millis() - last > 120)
   {
      g_carstate = enSTOP;
      g_ServoState = enServoS;
      last = millis();
   }
   switch (g_carstate)
   {
   //case enSTOP: brake(); break;
   case enRUN: advance(CarSpeedControl);    break;
   case enLEFT: left(CarSpeedControl);    break;
   case enRIGHT: right(CarSpeedControl); break;
   case enBACK: back(CarSpeedControl); break;
   case enSPINLEFT: spin_left(CarSpeedControl); break;
   case enSPINRIGHT: spin_right(CarSpeedControl); break;
   default: brake(); break;
```

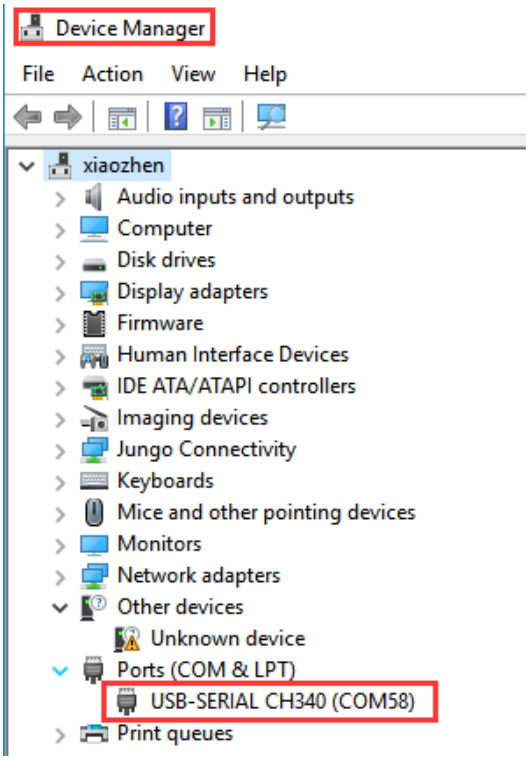
```
  }
  switch (g_ServoState)
  {
  case enServoL:
     Servo_LR++;
     if (Servo_LR > 180)
     {
        Servo_LR = 180;
     }
     Servo180(1, Servo_LR);
     delay(5);
     break;
  case enServoR:
     Servo_LR--;
     if (Servo_LR < 0)
     {
        Servo_LR = 0;
     }
     Servo180(1, Servo_LR);
     delay(5);
     break;
  default:
     break;
  }
}
```

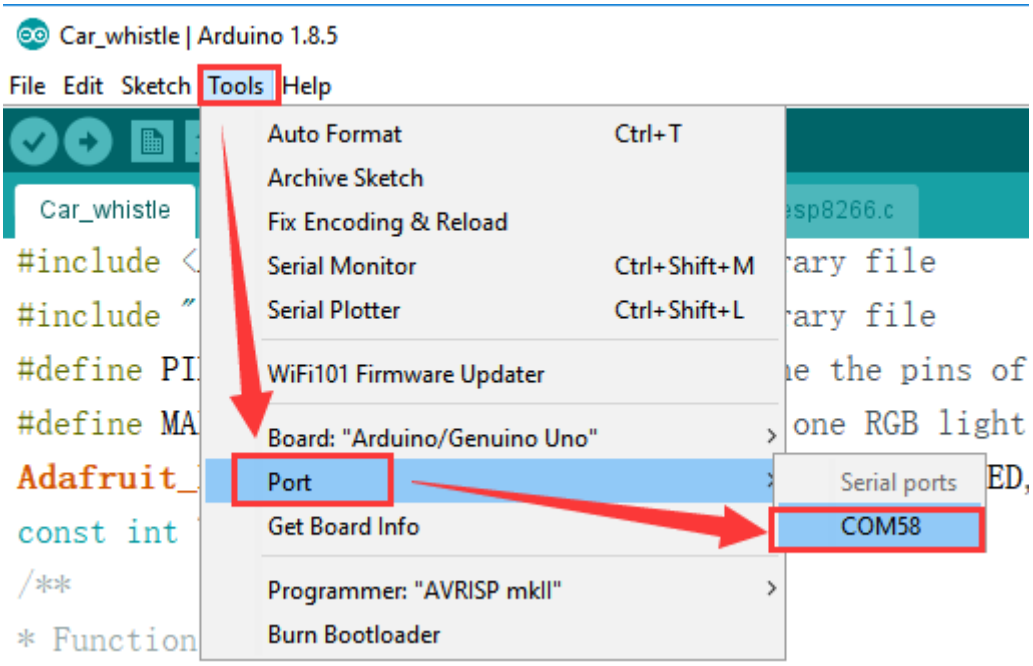
**5. Compiling and downloading code**

5.1 We need to open the **IR_Control_Car.ino** file by Arduino IDE software. Then click“√”under the menu bar to compile the code, and wait for the word "**Done compiling** " in the lower left corner, as shown in the figure below.

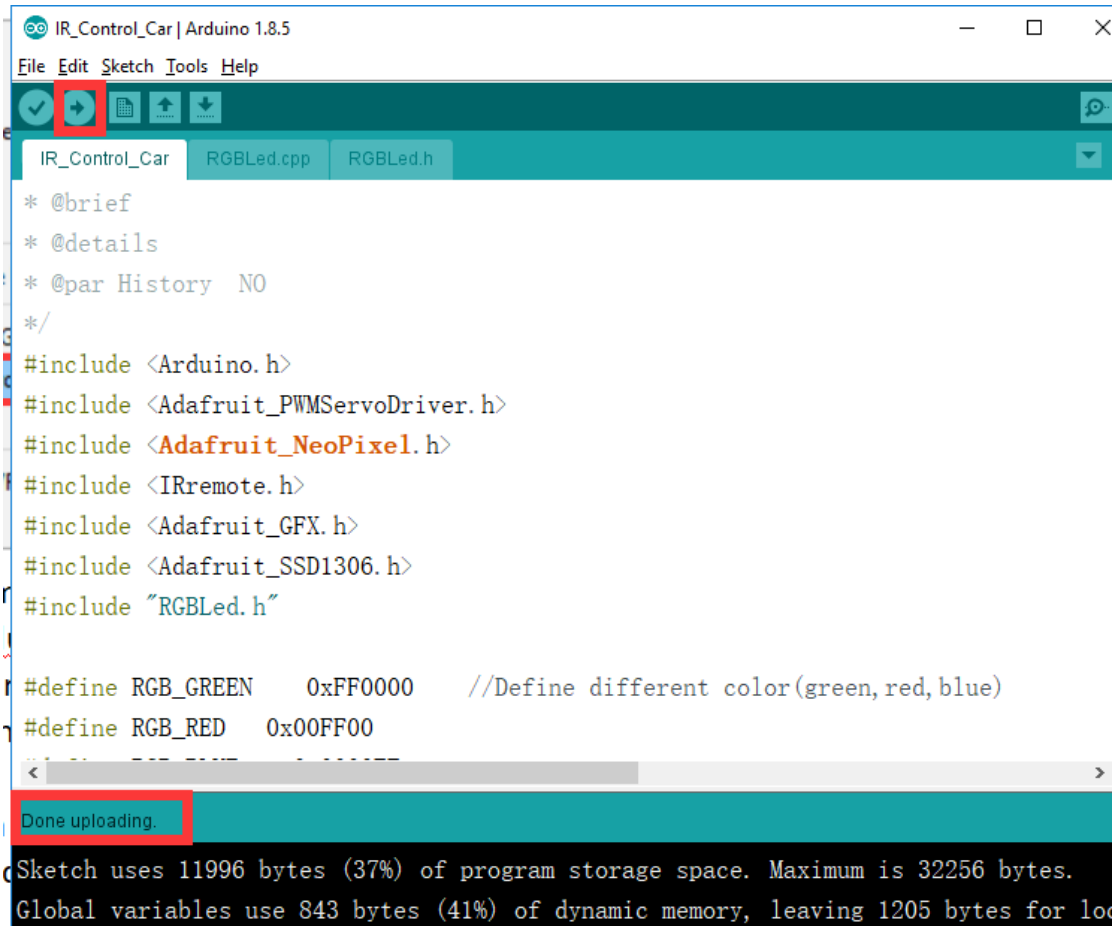



5.2 In the menu bar of Arduino IDE, we need to select 【Tools】---【Port】---selecting the port that the serial number displayed by the device manager just now, as shown in the figure below.



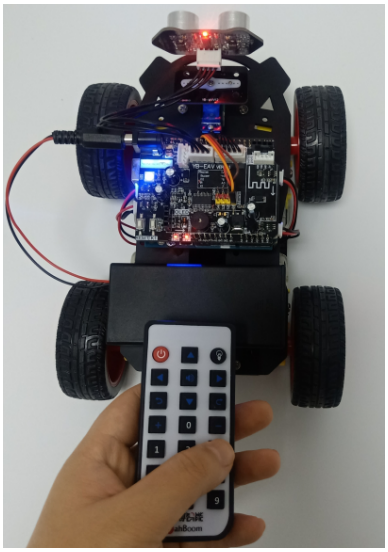


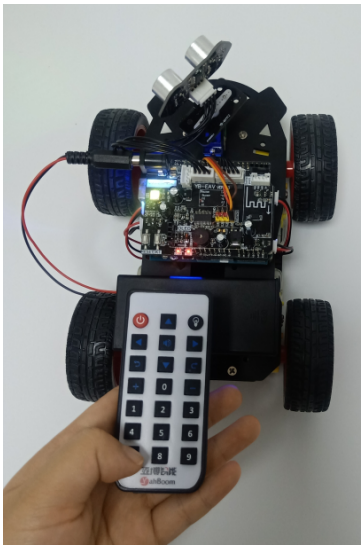

5.3 After the selection is completed, you need to click "→"under the menu bar to upload the code to the UNO board. When the word "**Done uploading**" appears in the lower left corner, the code has been successfully uploaded to the UNO board, as shown in the figure below.



## 6. Experimental phenomena

After the program is downloaded, we can control robot car by infrared controller. As shown below.





The following functions are controlled by the infrared remote controller.