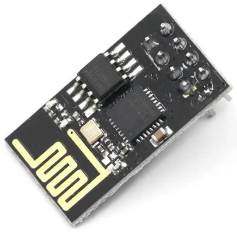
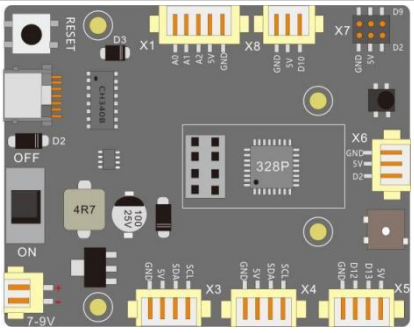
Lesson 12 APP Controlled with the 4WD Control Board

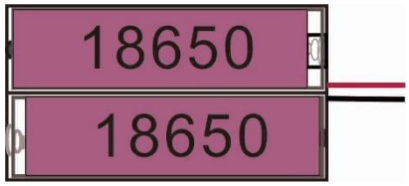
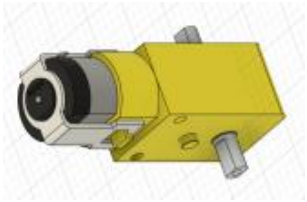
Preface

This Arduino 4WD control board has reserved the ESP-8266 interface, and the corresponding APP installation package is included in the tutorial we provide. Therefore, all you need to do is prepare an ESP-8266 module and some modules that you want to use the APP to control functions, such as motors, LED modules can be equipped with an APP controlled experiment. This lesson will provide examples of how to use the Arduino 4WD control board and ESP-8266 module to control the rotation of the motor based on the APP and the on/off of the green LED module.

1. Using the app to control motor operation

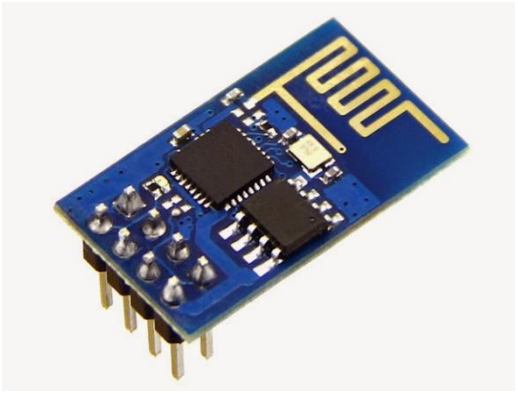
1.1 What do you need to prepare

Components	Quantity	Picture	Remark
USB cable	1		
ESP-8266 module	1		Not included in this kit,just for example.you need prepared by yourself
Control board	1		

Battery box with 2pcs 18650 batteries	1		18650 batteries are not included in this kit, please prepared by yourself.
DC Motor	4		TT motors are not included in this Kit, just for example,you can prepared what you want.

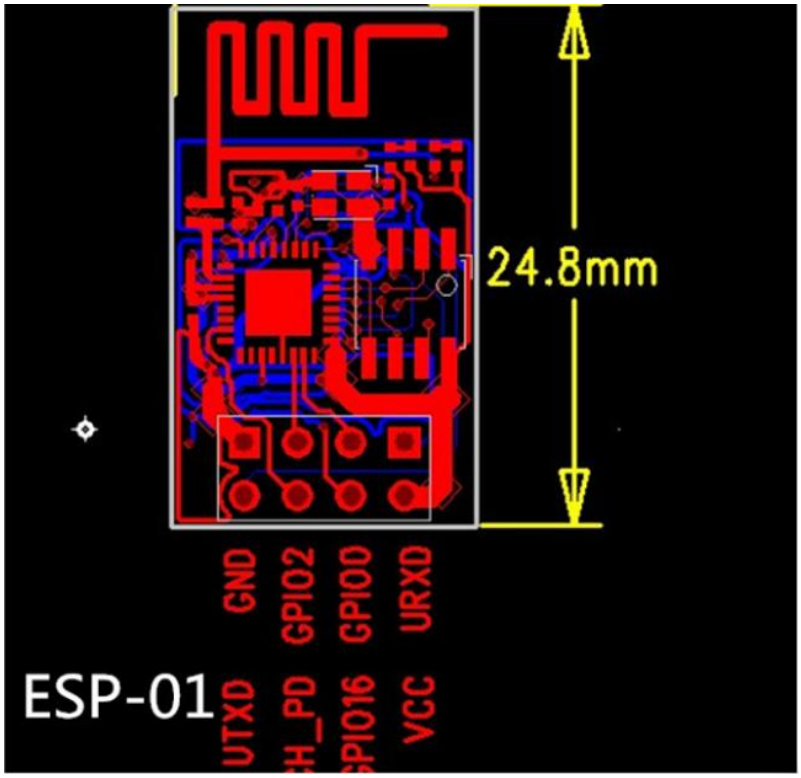
1.2 ESP8266-01 Module

The ESP8266-01 is a Wi-Fi module that allows microcontrollers access to a Wi-Fi network.



ESP8266-01 MODULE(hereinafter referred to as ESP-01)

ESP-01 pins



PIN	Description
URXD	UART_RXD，receive
UTXD	UART_TXD，send
GPIO	External Reset signal, reset when low level, work when high level (default high)
GND	GND
VCC	3.3V
GPIO 0	Working mode selection: floating: FlashBoot, working mode; pull down: UARTDownload,
CH_PD	Work at high level; power off at low level
GPIO 2	(1) It must be high level when powering on, and the hardware pull-down is prohibited; (2) Internal default pull-up

1.2.1 Introduction to AT commands of ESP-01 module

Basic instructions

command	description
AT	Test AT boot
AT+GMR	View version information
AT+CWMODE	Select WIFI application mode
AT+RST	restart module
client mode	
AT+CWLAP	List currently available router access points
AT+CWJAP	Join access point
AT+CWQAP	exit access point
AT+CIPSTART	Establish TCP, connect to server
AT+CIPCLOSE	Close TCP
AT+CIFSR	Get local IP address
AT+CIPMODE	Set module transfer mode
AT+CIPSEND	Send data
server mode	
AT+ CWSAP	Query and set the WIFI name, password and encryption method in AP (server) mode
AT+ CWLIF	View the IP address of the connected device
AT+CIPMUX	Start multiple connections
AT+CIPSERVER	Configured as server default port 333
AT+CIPSTO	Set server timeout
AT+ CIPSTATUS	Get connection status

1.2.2 Working mode and commands of ESP-01 module

ESP-01 module acts as a client (transparent transmission)

1. AT: Test AT development mode start
2. AT+GMR: View firmware version information
3. AT+CWMODE=1: Set WIFI application mode

（1） Station mode

（2） AP mode
（3） AP and Station mode, AP refers to as an access point, station refers to as a client station

4. AT+RST：restart

5. AT+CWLAP：list available access points
6. AT+CWJAP="wifiname","wifi passport"：join wifi

7. AT+CIFSR ：get local IP address

8. The PC connects to the router, and the network debugging assistant uses the computer IP address to create a server
9. AT+CIPSTART="TCP","192.168.101.110",8080 Establish a TCP connection with the server
10. AT+CIPMODE=1：Set the transparent transmission mode (you can send it all the time, otherwise you have to use AT+CIPSEND=4 to send the number of bytes; as a server mode, you cannot use the transparent transmission mode)
11. AT+CIPSEND ：Start transparent transmission, the serial port debugging assistant sends data, and the network debugging assistant sends data
12. Received data format: serial port debugging: +IPD, n:xxxxxxxxxx The length of the received data is n bytes, xxxxx is the data; network debugging: [Tcp client 192.168.1.108 2872] 123, TCP mode, client IP address, port number, 123 is the data

ESP-01 WIFI module as client (single connection)

1. AT+CWMODE=1: set WIFI application mode
   (1) Station mode
   (2) AP mode
   (3) AP and Station mode,
   AP refers to as an access point, station refers to as a client station
2. AT+CWJAP="wifiname","wifi password": join wifi
3. PC is connected to the router, and the network debugging assistant uses the computer IP address to create a server, and the IP setting is shown in Figure 2 above;
4. AT+CIPSTART="TCP","192.168.101.110",8080 ：Establish a TCP connection with the server
5. AT+CIPSEND=4: Serial port debugging sends four bytes of data, input the content of the four bytes to be sent, no need to press Enter. If the number of bytes sent exceeds the length n set by the command, it will respond busy, and send the first n bytes of data, and respond SEND OK after completion. Network debugging can be sent arbitrarily.

ESP-01 WIFI module as server

1.AT+CWMODE=2 ： set WIFI application mode
(1) Station mode
(2) AP mode
(3) AP and Station mode,
AP refers to as an access point, station refers to as a client station
2.AT+RST：restart
3.AT+CWSAP? ： Query and display the parameters in AP mode，
+CWSAP:"ESP_8266","12345678",11,3,4,0
4.AT+CWSAP="ESP_8266","12345678",11,3 ： access point name, password, channel number, encryption method. 11 is the channel number, it needs to be restarted after modification, and 3 is the encryption method

<ecn>	Encryption	0	OPEN，if set to open, it will not work even if a password is set
		1	WEP
		2	WPA_PSK
		3	WPA2_PSK
		4	WPA_WPA2_PSK

1. AT+CIPMUX=1:start multiple connections
2. AT+CIPSERVER=1:create a server, the default port is 333
3. AT+CIPSTO=300：set the server timeout from 0 to 28800, the unit is s, and the client will be kicked out when the timeout expires.
4. AT+CIFSR :obtain the local IP address in order to set up the network assistant. First, the PC needs to be connected to the hotspot of the WIFI module, and the network debugging assistant on the PC connects to the AP as a client.
5. AT+CWLIF:view connected devices

This lesson uses the ESP-01 module as the working mode of the server:

```
1.AT+RST\r\n       //In the Arduino code, the AT command must end with a carriage return and line feed character "\r\n"
2. AT+CWMODE=3\r\n    //set to soft AP+station mode
3. AT+CWSAP=\"Cokoino_ESP8266-01\",\"12345678\",11,0\r\n
//.   Cokoino_ESP8266-01        ----------WIFI access point name
//.   12345678                  ----------WIFI password
//.   11                        ----------Channel number
//.   0                         ----------Encryption mode    0-OPEN
```

4. AT+CIPMUX=1\r\n //start multiple connections
5. AT+CIPSERVER=1,3001\r\n //Create a server, the default port is 333, modify the port to 3001, consistent with APP
6. AT+CIPSTO=7000\r\n // Example Set the server timeout period to 7000 seconds

1.3 Install and learn the Cokoino Robot APP

1.3.1 Install Robot APP

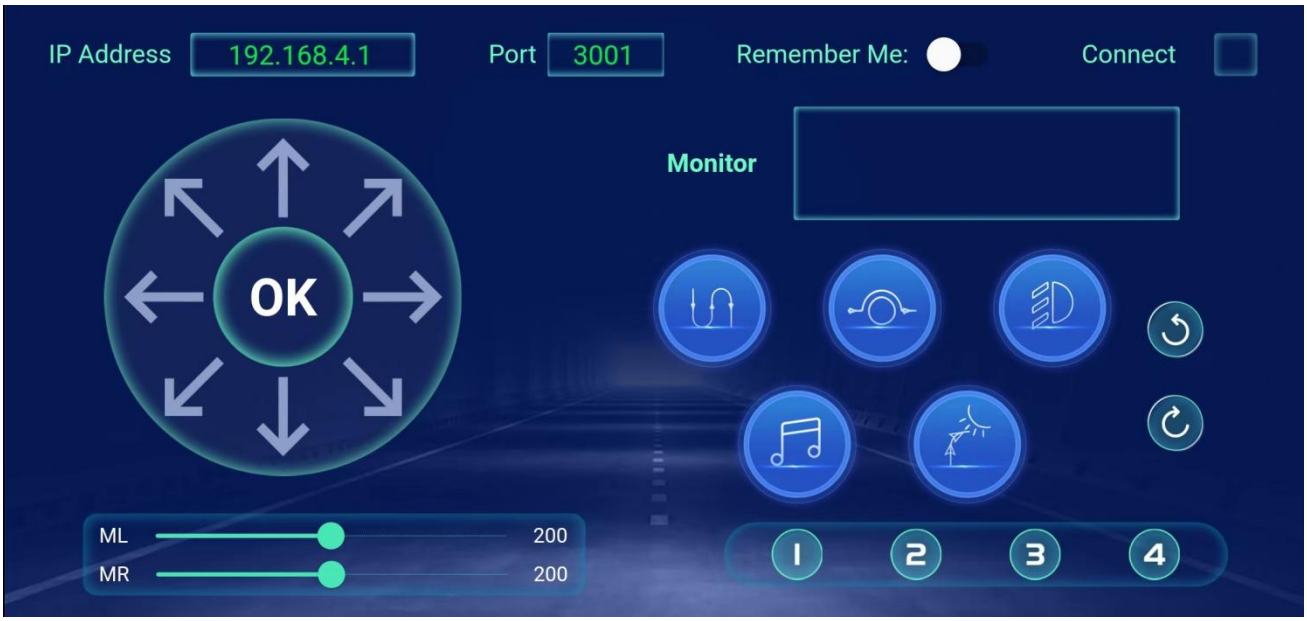
The APK file of the Robot APP is stored in this folder：E:\CKK0019-main\Robot apk\app-release.apk

Send the .apk file to the mobile phone for installation. Note: this APP is only compatible with Android phones

If a risk warning pops up during installation, please ignore the risk and choose to continue the installation. We guarantee that the APP is virus-free and risk-free. After the installation is complete, we will see the Robot APP icon as shown below on the phone

1.3.2 Introduction of Robot APP

Click the Robot APP icon on the mobile phone, the APP interface is as follows



1.3.3 Introduction to APP UI

It is the address of the ESP-01 module as a server. It is a fixed value and cannot be changed on the APP interface.

It is the port number of the ESP-01 module as a server, it is set to be a fixed value of 3001 in the app, so when you writing the Arduino code, be sure to use the AT command to set the port number of the ESP-01 module to 3001.

Connect Button. Click "Connect", you can connect the wifi to the ESP-01 module, when the connection is completed, the Monitor will display a successful connection message

This is the data monitoring window, which can simultaneously display the operation instructions and status of the APP

control command:“trk”.Usually we define it as function button: “following the line”. When the APP is connected to the ESP-01 module of the car, press this button, and the car will follow the line. You can also define its functions yourself.

 Control command“aod”.Usually we define it as function button:“Avoid obstacles”. When the APP is connected to the ESP-01 module of the car, press this button, and the car will drive automatically and avoid obstacles.You can also define its functions yourself.

 control command:“lgt”.Usually we define it as function button:“Light Show”， When the APP is connected to the ESP-01 module of the car, press this button, and the led light on the car will be turned on and change various colors.You can also define its functions yourself.

 control command:“muc”.Usually we define it as function button:“Music”. When the APP is connected to the ESP-01 module of the car, press this button, the buzzer on the car will start playing music with different melodies. You can also define its functions yourself.

 control command:“flt”,Usually we define it as function button:“Follow Light”. If you add a photosensitive sensor to the car, press this button, the robot will follow the light source in a dark environment. You can also define its functions yourself.

 control command:“rtl”.Usually we define it as function button:“Rotaiton Left”， when the APP is connected to the ESP-01 module of the car, press this button, the car will turn left in a circle. You can also define its functions yourself.

 control command:“rtr”.Usually we define it as function button:“Rotaiton Right”， when the APP is connected to the ESP-01 module of the car, press this button, the car will turn right in a circle.You can also define its functions yourself.

 control command:“bt1”.You can also define its functions yourself.

Function button:“Button 2”、 “Button 3”、 “Button 4”， control command: “bt2”、 “bt3” 、 “bt4”.Functions are undefined,you can define its functions yourself.

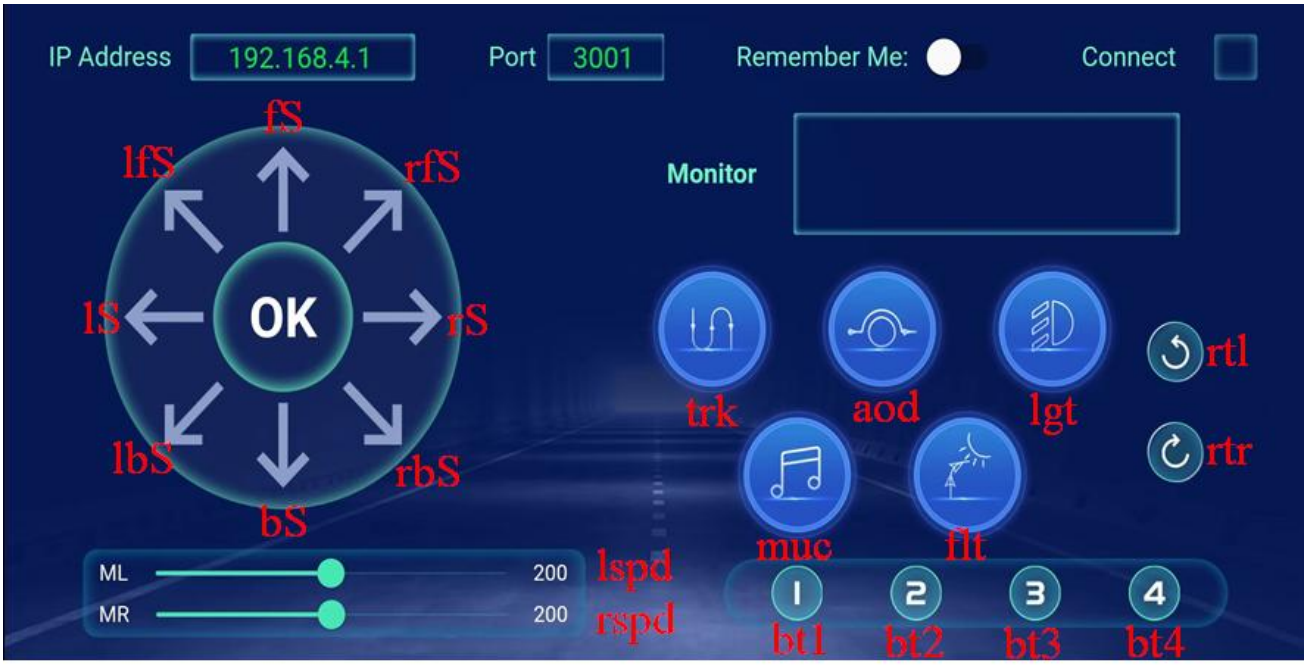
Drag-and-drop button:“Left Speed”、 “Right Speed”，when the APP is connected to the ESP-01 module of the car, dragging these two buttons will change the rotation speed of the left and right wheels of the car. control commands:“lspd”、 “rspd”

Move direction buttons,the arrow is the control button for the driving direction of the car, and there are 8 control directions in total, namely "forward", "left forward", "left", "left backward", "backward", "right backward", "right", "right forward" ", the middle "OK" button is defined as the stop button.When the APP is connected to the ESP-01 module of the car,You can control the direction of movement of the car with these arrow buttons.

1.3.4 Introduction to button commands on the APP interface

All function buttons on the APP interface have a fixed command, which is unique and invariable. Therefore, when writing Arduino codes, you need to pay attention to matching the judgment commands in the code with the commands of the APP function button, otherwise the APP will not control the car correctly.

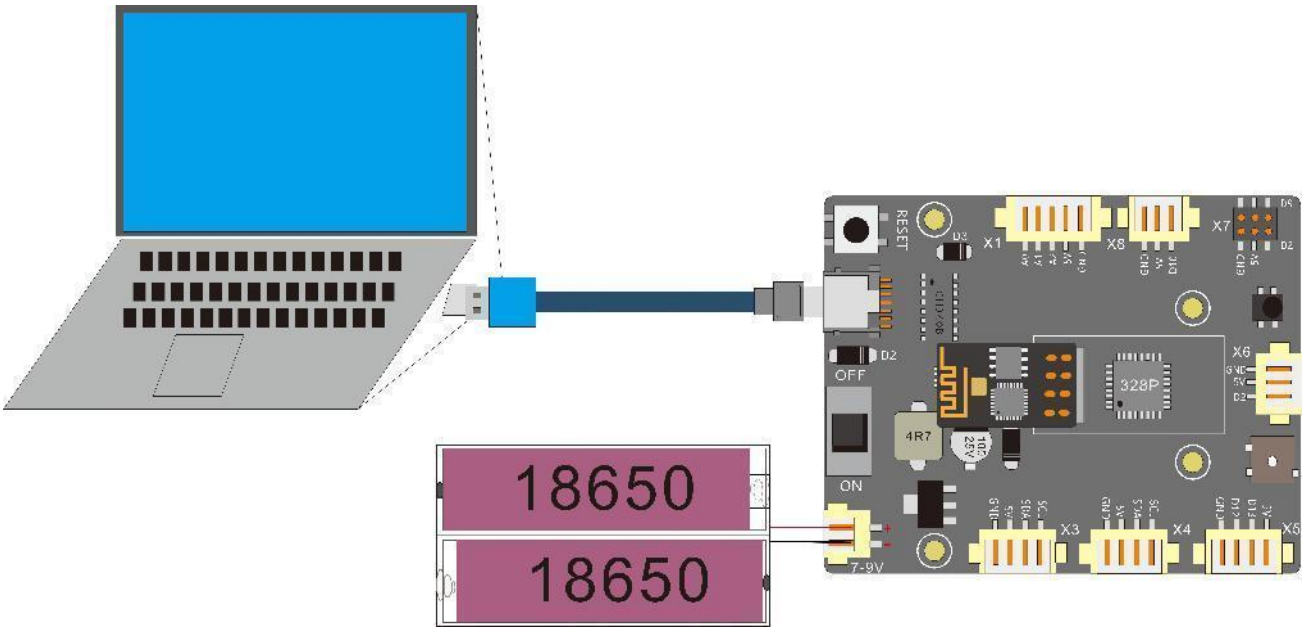
The commands of the function button on the APP interface are as follows:

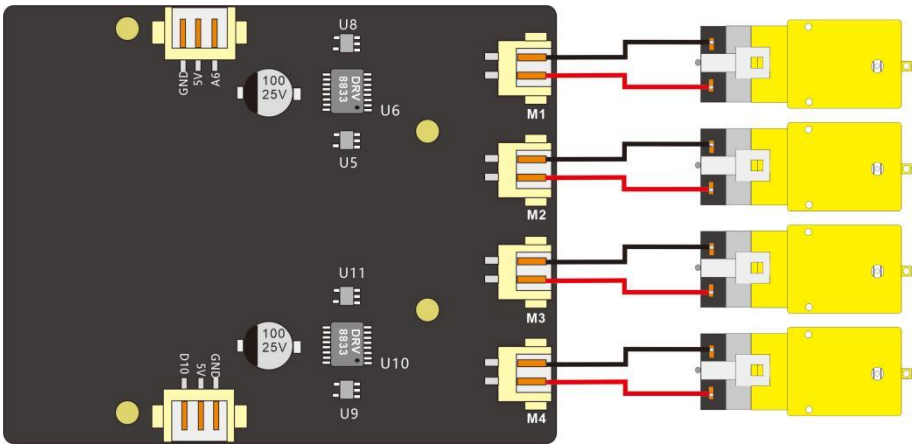


After you successfully upload the code and connect the APP to the ESP-01 module of the Arduino 4WD Control Board, press the button in the app, and the ESP-01module will receive the command sent by the APP button and convert it into a string signal.

1.4 Circuit Connection

Insert the ESP8266-01 module into the 4WD control board, pay attention to the direction, and do not insert it backwards. Connect the battery pack to the power input port on the 4WD control board, and connect 4 TT motors to M1, M2, M3, M4 interfaces on the 4WD control board. Connect the 4WD control board to the computer using a Typec USB cable.





1.5 Upload the code and test

The code used in this lesson is placed in this folder：

E:\CKK0019-main\Tutorial\sketches\10_1_Wifi_Controlled_Motors

Install Regexp library

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

Before uploading the code, Remove the ESP-01 module from the 4WD control board.

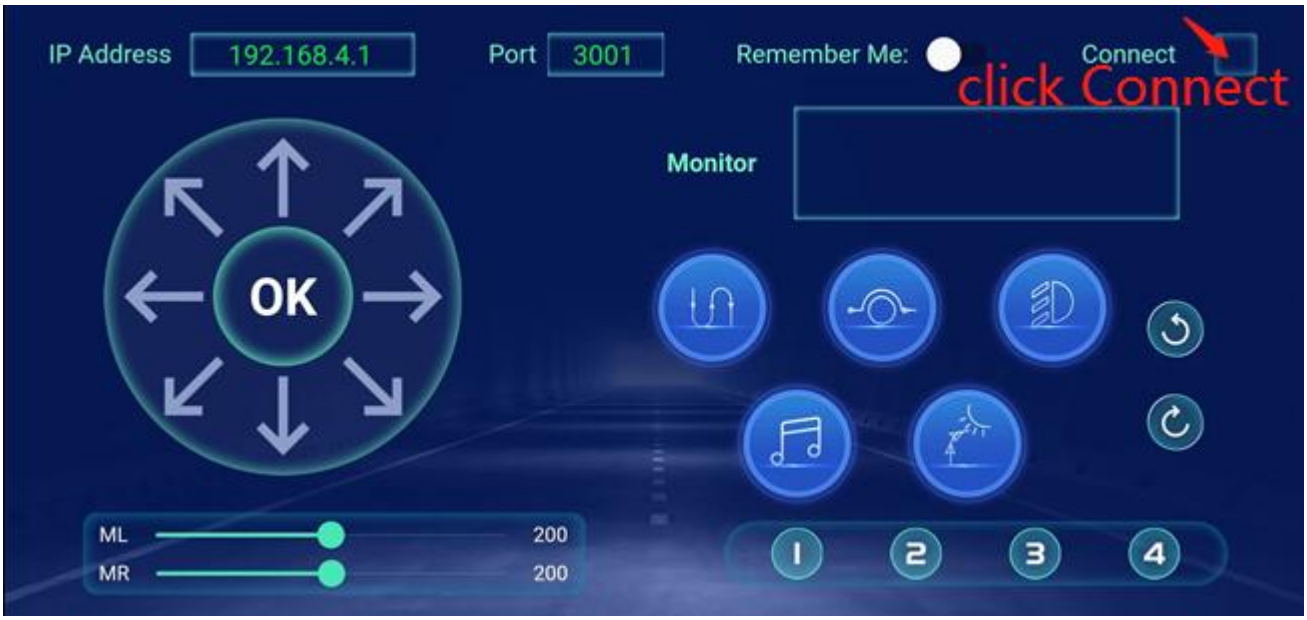
After uploading the code, unplug the USB cable, Insert the ESP8266-01 module back into the 4WD control board, turn on the power switch on the control board.

Click "Settings" on the mobile phone, and click "WLAN" on the setting interface to enter the WLAN interface. Then look for the "Cokoino_ESP8266-01" signal in the list of available WLANs

Click “Cokoino_ESP8266-01”WLAN，enter password 12345678，then click "connect"

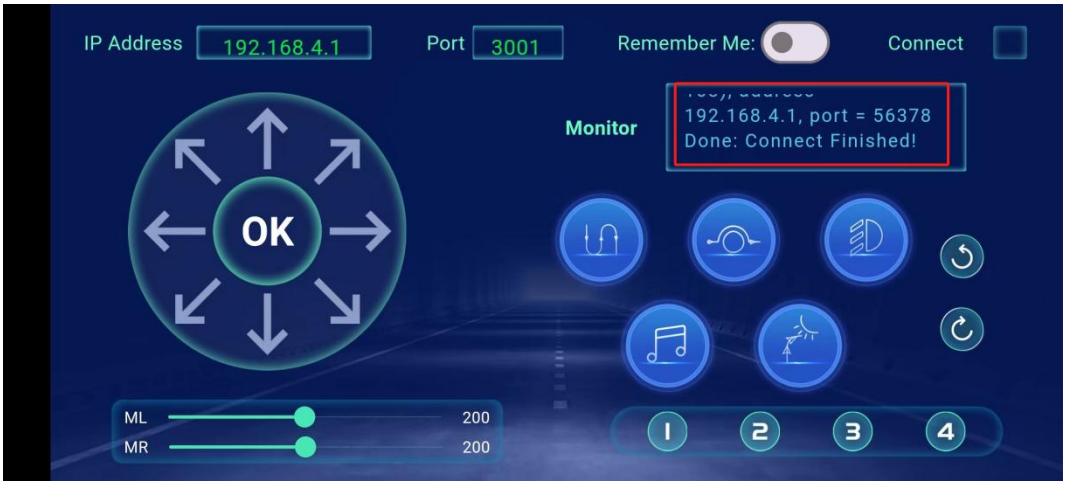
After the connection is successful, if the mobile phone pops up a window prompting that the current WLAN cannot access the Internet, whether to continue to use this WLAN, click "Use"

Open the Robot APP on the mobile phone and click "Connect"



After the connection is successful, the Monitor box will display "Received: Welcome to connect to ESP8266-01, ID: 0""or "Connect Finished".

If unable to connect, please exit the app, power on the control board again, reconnect the phone to the "Cokoino-ESP8266-01" WLAN signal, and then open the app and click on "Connect".Generally, a re operation can be successful.



Congratulations, the APP has been successfully connected to the ESP-01 module, You can control the motor operation through the mobile app now.

 In this example, we define it as forward. Press this button, TT motor rotates forward.

 In this example, we define it as backward. Press this button, TT motor reverse.

In this example, we define it as stopping. Press this button, The TT motor stops rotating.

 In this example, we define it as a buzzer. Press this button to activate the buzzer.

1.6 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/

10_1_Wifi_Controlled_Motors.ino:

```
/*************************************************
 * This code applies to cokoino Arduino 4WD Control Board
 * Through this link you can download the source code:
 * https://github.com/Cokoino/CKK0019
 * Company web site:
 * http://cokoino.com/
 *
*************************************************/
#include <Regexp.h>

const int NSLEEP1 = 6;      // define pin for PWM used to control rotational speed of M1,M2 motor
const int AIN1 = 8;         // define pin used to control rotational direction of M1 motor
const int BIN1 = 7;         // define pin used to control rotational direction of M2 motor
//SET THE PIN FOR DRV8833(U10)
const int NSLEEP2 = 5;      // define pin for PWM used to control rotational speed of M3,M4 motor
const int AIN2 = 4;         // define pin used to control rotational direction of M3 motor
const int BIN2 = A3;        // define pin used to control rotational direction of M4 motor

#define buzzer_pin 11     //buzzer PIN


// regular
MatchState ms;
/// Car driving direction control button on the app interface, a total of 8 direction buttons.
const String phone1 = "fS";  // forwardStart:
const String phone1_5 = "lfS";  // forward_left_Start
const String phone2 = "lS";    // leftStart
const String phone2_5 = "lbS";  // left_backward_Start
const String phone3 = "bS";    // backwardStart
```

```
31.     const String phone3_5 = "rbS";  // backward_right_Start
32.     const String phone4 = "rS";    // rightStart
33.     const String phone4_5 = "rfS";  // right_forward_Start
34.     /// The other function buttons on the app interface
35.     const String phone5 = "OK";//stop
36.     const String phone6 = "rtl";//rotation left
37.     const String phone7 = "rtr";//rotation right
38.     const String phone8 = "trk";//track line running
39.     const String phone9 = "aod";//Avoid obstacles
40.     const String phone10 = "lgt";//light show
41.     const String phone11 = "muc";//buzzer
42.     const String phone12 = "flt";//fllow light
43.     const String phone13 = "bt1";//button1
44.
45.     String comdata = "";//import the comdata string
46.     char judge = 0;//init the judge
47.
48.
49.     void setup() {
50.       Serial.begin(115200);
51.       delay(100);  // If the information printed out of the serial port is garbled, extend the delay time to solve th
    e problem.
52.       while (Serial.read() >= 0)
53.         continue;
54.       Serial.flush();
55.       ESP8266_ATCOMMAND();//esp-01 module AT instruction function
56.
57.       pinMode(buzzer_pin, OUTPUT);
58.       pinMode(NSLEEP1, OUTPUT);     // set to output mode
59.       pinMode(AIN1, OUTPUT);       // set to output mode
60.       pinMode(BIN1, OUTPUT);       // set to output mode
61.       pinMode(NSLEEP2, OUTPUT);     // set to output mode
62.       pinMode(AIN2, OUTPUT);       // set to output mode
63.       pinMode(BIN2, OUTPUT);       // set to output mode
64.     }
65.
66.     void loop() {
67.
68.       while (Serial.available() > 0) {
69.         comdata += char(Serial.read());
70.         delay(1);
71.       }
```

```
72.         judgement();
73.       }
74.
75.       // ESP8266 set the AT instructionS
76.       void ESP8266_ATCOMMAND() {
77.
78.         Serial.print(F("AT+RST\r\n"));  //F(): Store string constants in Flash flash to avoid memory depletion due to SRAM usage.
79.         delay(3000);
80.         Serial.print(F("AT+CWMODE=3\r\n"));  //set to softAP+station mode
81.         delay(300);
82.         Serial.print(F("AT+CWSAP=\"Cokoino_ESP8266-01\",\"12345678\",11,2\r\n"));//wifiname:Cokoino_ESP8266-01,wifipassword:12345678
83.         //channnel:11 Encryption mode:2 ;Encryption mode should not set to 1,otherwise the wifi can't set successed
84.         delay(200);
85.         Serial.print(F("AT+CIPMUX=1\r\n"));//Enable multiple connections
86.         delay(200);
87.         Serial.print(F("AT+CIPSERVER=1,3001\r\n"));//Create the server. The default port is 333. Change the port to 3001, which is consistent with the APP
88.         delay(200);
89.         Serial.print(F("AT+CIPSTO=7000\r\n"));//Example Set the server timeout period to 7000 seconds
90.         delay(2000);
91.       }
92.       void judgement() {
93.
94.         if (comdata.length() > 0) {
95.           comdata += "\n";  //This sentence must be added, otherwise the matched command character is one less, and the newline is used to assist in the complete match.
96.           char buf[comdata.length()];
97.           comdata.toCharArray(buf, comdata.length());
98.           ms.Target(buf);
99.           char result = ms.Match("%c*%+IPD, ?[0-9]+, ?[0-9]+: ?([^%c]+)%c*$");
100.          if (result > 0) {
101.            ms.GetCapture(buf, 0);
102.            comdata = String(buf);
103.            delay(100);
104.          } else {
105.            result = ms.Match("%c*%s?([0-9]),%s?([^%c]+)%c*$");  // esp8266 Multi-channel supports up to 5 connections (id:0-4)
106.            if (result > 0) {
107.              char buf0[1];  // esp8266 In multi-channel mode. id of the connection at this time
```

```
108.            ms.GetCapture(buf0, 0);
109.            ms.GetCapture(buf, 1);
110.            comdata = String(buf);
111.            if (comdata == "CONNECT")//The APP successfully connects to the wifi of ESP-01 module
112.            {
113.              String receiveOkMs = "Welcome to connect to ESP8266-01, ID: " + String(buf0) + " .";//A successf
    ul connection message is displayed
114.              Serial.println("AT+CIPSEND=" + String(buf0) + "," + receiveOkMs.length() + "\r\n");
115.              delay(10);
116.              Serial.print(receiveOkMs);
117.            }
118.          }
119.        }
120.        //comdata = "";
121.        //return; // When debugging communication with the APP, it needs to be commented out when normal u
    se
122.
123.         if (comdata == phone1) {
124.           judge = 1;
125.         }
126.         // else if (comdata == phone1_5) {
127.           //judge = 2;
128.         //} else if (comdata == phone2) {
129.           //judge = 3;
130.         //} else if (comdata == phone2_5) {
131.           //judge = 4;
132.         //}
133.           else if (comdata == phone3) {
134.           judge = 5;
135.         }
136.           //else if (comdata == phone3_5) {
137.           //judge = 6;
138.         //} else if (comdata == phone4) {
139.           //judge = 7;
140.         //} else if (comdata == phone4_5) {
141.           //judge = 8;
142.           //}
143.           else if (comdata == phone5) {
144.           judge = 9;
145.         }
146.           //else if (comdata == phone6) {
147.           //judge = 10;
```

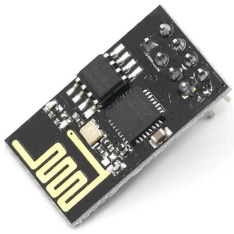
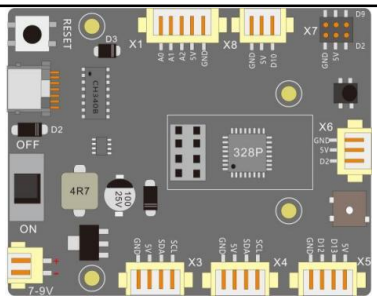
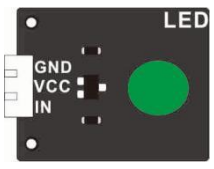
```
148.          //} else if (comdata == phone7) {
149.            //judge = 11;
150.          //} else if (comdata == phone8) {
151.            //judge = 12;
152.          //} else if (comdata == phone9) {
153.            //judge = 13;
154.          //}
155.            else if (comdata == phone10) {
156.            judge = 14;
157.          } else if (comdata == phone11) {
158.            judge = 15;
159.          }
160.            //else if (comdata == phone12) {
161.            //judge = 16;
162.          //} else if (comdata == phone13) {
163.            //judge = 17;
164.          //}
165.          else {
166.            judge = 9;
167.          }
168.          comdata = "";
169.        }
170.
171.        switch (judge) {
172.
173.          case 1:
174.            forward();
175.            break;
176.          case 5:
177.            backward();
178.            break;
179.          case 9:
180.            Stop();
181.            break;
182.          case 15:
183.            music();
184.            delay(2000);
185.            judge = 9;
186.            break;
187.          default: break;
188.        }
189.      }
```

```
190.
191.    void forward()
192.    {
193.     analogWrite(NSLEEP1,140); //Start M1,M2 motor and set speed
194.     //analogWrite(NSLEEP1,Num);Num range from 0 to 255,The larger the Num, the higher the output voltag
     e and the faster the motor speed.
195.     analogWrite(NSLEEP2,140); //Start M3,M4 motor and set speed
196.     //analogWrite(NSLEEP2,Num);Num range from 0 to 255,The larger the Num, the higher the output voltag
     e and the faster the motor speed.
197.     digitalWrite(AIN1,HIGH);  //drive the M1 motor foward rotation
198.     digitalWrite(BIN1,HIGH);  //drive the M2 motor foward rottion
199.     digitalWrite(AIN2,HIGH);  //drive the M3 motor foward rotation
200.     digitalWrite(BIN2,HIGH);  //drive the M4 motor foward rottion
201.    }
202.
203.    void backward()
204.    {
205.     analogWrite(NSLEEP1,140); //Start M1,M2 motor and set speed
206.     analogWrite(NSLEEP2,140); //Start M3,M4 motor and set speed
207.     digitalWrite(AIN1,LOW);  //drive the M1 motor reverse rotation
208.     digitalWrite(BIN1,LOW);  //drive the M2 motor reverse rottion
209.     digitalWrite(AIN2,LOW);  //drive the M3 motor reverse rotation
210.     digitalWrite(BIN2,LOW);  //drive the M4 motor reverse rottion
211.    }
212.
213.    void Stop()
214.    {
215.     analogWrite(NSLEEP1,0);//stop M1,M2 motor
216.     analogWrite(NSLEEP2,0);//stop M3,M4 motor
217.    }
218.    void music()
219.    {
220.      for(int i = 0;i < 80; i++)
221.      {
222.       //The high and low levels constitute a square-wave signal that triggers the passive buzzer to work
223.       digitalWrite(buzzer_pin, HIGH);
224.       delay(1);
225.       digitalWrite(buzzer_pin, LOW);
226.       delay(1);
227.      }
228.      for(int j = 0;j< 100; j++)
229.      {
```

```
230.        digitalWrite(buzzer_pin, HIGH);
231.        delay(2);
232.        digitalWrite(buzzer_pin, LOW);
233.        delay(2);
234.      }
235.    }
```

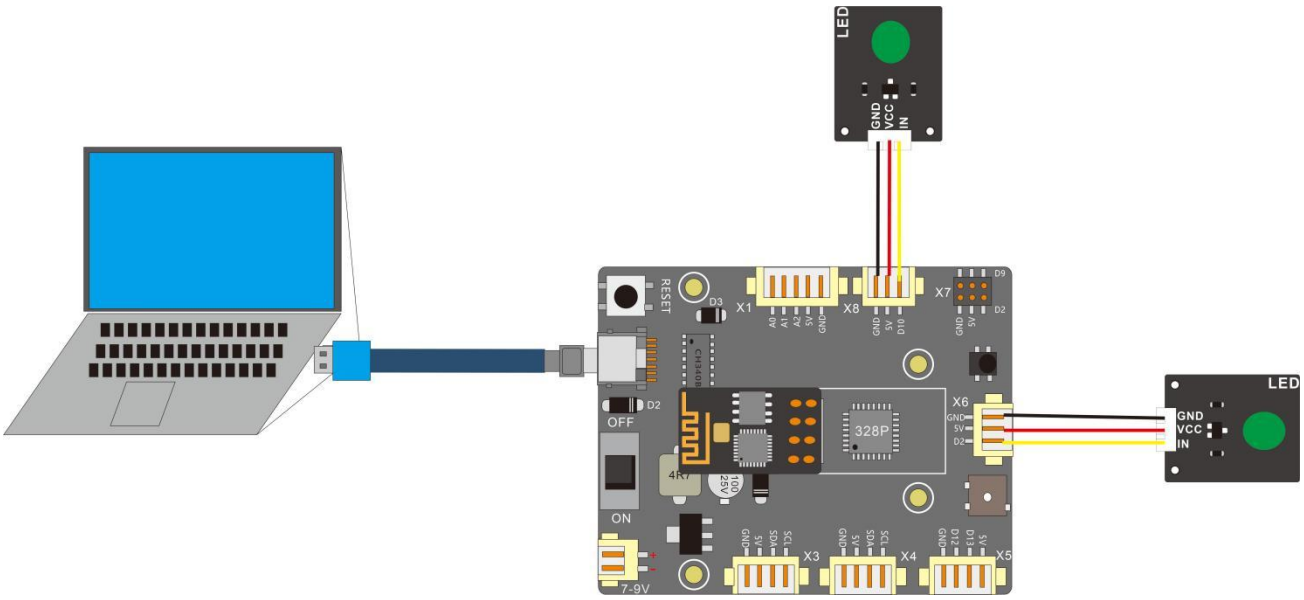
2. Using the app to control Green LED Module

2.1 What do you need to prepare

Components	Quantity	Picture	Remark
USB cable	1		
ESP-8266 module	1		Not included in this kit,just for example.you need prepared by yourself
Control board	1		
Green LED Module	2		Green LED Modules are not included in this kit, just for example,you can prepare what you want

2.2 Circuit Connection

Connect one green LED module to the x6 terminal of the 4WD control board using a DuPont cable, and the other to the x8 terminal of the 4WD control board. Insert the ESP8266-01 module into the corresponding interface of the WD control board, paying attention to the direction and not inserting it in the opposite direction.



2.3 Upload the code and test

The code used in this lesson is placed in this folder：

E:\CKK0019-main\Tutorial\sketches\10_2_Wifi_Controlled_LED-Modules

Before uploading the code, Remove the ESP-01 module from the 4WD control board.

After uploading the code, unplug the USB cable, Insert the ESP8266-01 module back into the 4WD control board, turn on the power switch on the control board.

Click "Settings" on the mobile phone, and click "WLAN" on the setting interface to enter the WLAN interface. Then look for the "Cokoino_ESP8266-01" signal in the list of available WLANs

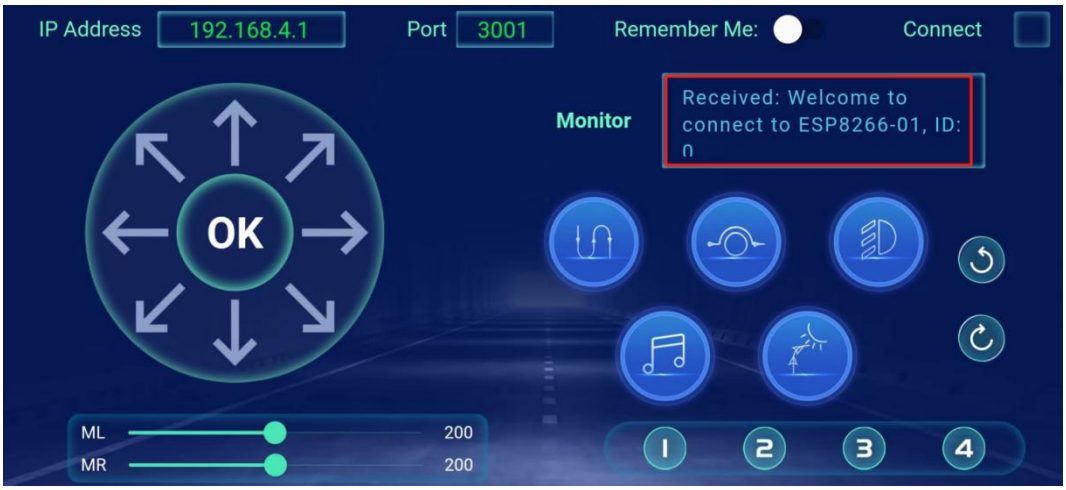
Click “Cokoino_ESP8266-01”WLAN，enter password 12345678，then click "connect"

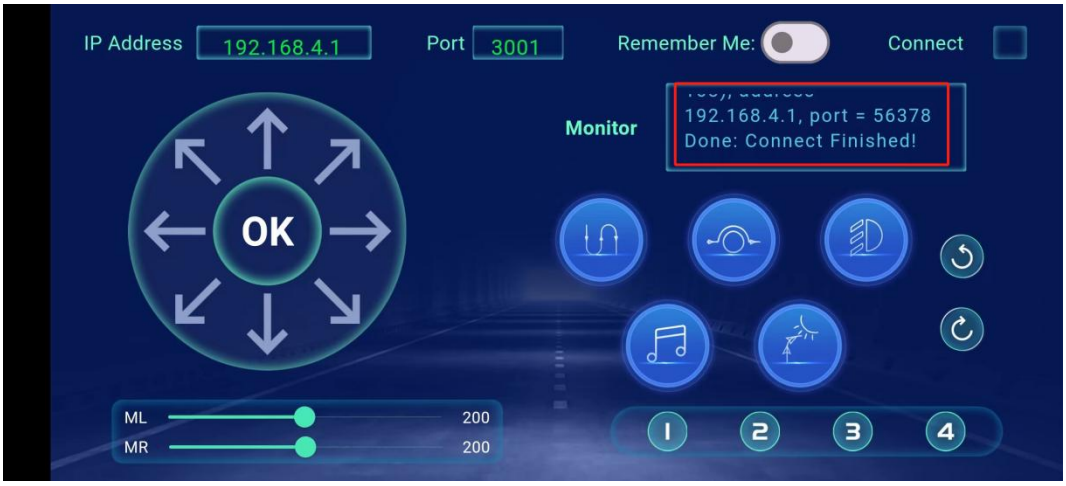
After the connection is successful, if the mobile phone pops up a window prompting that the current WLAN cannot access the Internet, whether to continue to use this WLAN, click "Use"

Open the Robot APP on the mobile phone and click "Connect"

After the connection is successful, the Monitor box will display "Received: Welcome to connect to ESP8266-01, ID: 0""or "Connect Finished".

If unable to connect, please exit the app, power on the control board again, reconnect the phone to the "Cokoino-ESP8266-01" WLAN signal, and then open the app and click on "Connect".Generally, a re operation can be successful.





Congratulations, the APP has been successfully connected to the ESP-01 module, You can control the motor operation through the mobile app now.

 In this example, we define it as LED function. Press this button, the green LED modules will light on and off.

 In this example, we define it as stopping. Press this button, The reen LED modules will light off.

2.4 Code

10_2_Wifi_Controlled_LED-Modules.ino:

```
/*************************************************
 * This code applies to cokoino Arduino 4WD Control Board
 * Through this link you can download the source code:
 * https://github.com/Cokoino/CKK0019
 * Company web site:
 * http://cokoino.com/
 *************************************************/
```

```
#include <Regexp.h>

const int NSLEEP1 = 6;      // define pin for PWM used to control rotational speed of M1,M2 motor
const int AIN1 = 8;         // define pin used to control rotational direction of M1 motor
const int BIN1 = 7;         // define pin used to control rotational direction of M2 motor
//SET THE PIN FOR DRV8833(U10)
const int NSLEEP2 = 5;      // define pin for PWM used to control rotational speed of M3,M4 motor
const int AIN2 = 4;         // define pin used to control rotational direction of M3 motor
const int BIN2 = A3;        // define pin used to control rotational direction of M4 motor

#define buzzer_pin 11     //buzzer PIN
#define LED1 2
#define LED2 10



// regular
MatchState ms;
/// Car driving direction control button on the app interface, a total of 8 direction buttons.
const String phone1 = "fS";  // forwardStart:
const String phone1_5 = "lfS";  // forward_left_Start
const String phone2 = "lS";    // leftStart
const String phone2_5 = "lbS";  // left_backward_Start
const String phone3 = "bS";    // backwardStart
const String phone3_5 = "rbS";  // backward_right_Start
const String phone4 = "rS";    // rightStart
const String phone4_5 = "rfS";  // right_forward_Start
/// The other function buttons on the app interface
const String phone5 = "OK";//stop
const String phone6 = "rtl";//rotation left
const String phone7 = "rtr";//rotation right
const String phone8 = "trk";//track line running
const String phone9 = "aod";//Avoid obstacles
const String phone10 = "lgt";//light show
const String phone11 = "muc";//buzzer
const String phone12 = "flt";//fllow light
const String phone13 = "bt1";//button1

String comdata = "";//import the comdata string
char judge = 0;//init the judge

```

```
52.     void setup() {
53.
54.       Serial.begin(115200);
55.       delay(100);  // If the information printed out of the serial port is garbled, extend the delay time to solve th
    e problem.
56.       while (Serial.read() >= 0)
57.         continue;
58.       Serial.flush();
59.       ESP8266_ATCOMMAND();//esp-01 module AT instruction function
60.
61.       pinMode(buzzer_pin, OUTPUT);  // set to output mode
62.       pinMode(LED1, OUTPUT);        // set to output mode
63.       pinMode(LED2, OUTPUT);        // set to output mode
64.       pinMode(NSLEEP1, OUTPUT);     // set to output mode
65.       pinMode(AIN1, OUTPUT);        // set to output mode
66.       pinMode(BIN1, OUTPUT);        // set to output mode
67.       pinMode(NSLEEP2, OUTPUT);     // set to output mode
68.       pinMode(AIN2, OUTPUT);        // set to output mode
69.       pinMode(BIN2, OUTPUT);        // set to output mode
70.     }
71.
72.     void loop() {
73.
74.       while (Serial.available() > 0) {
75.         comdata += char(Serial.read());
76.         delay(1);
77.       }
78.       judgement();
79.     }
80.
81.     // ESP8266 set the AT instructionS
82.     void ESP8266_ATCOMMAND() {
83.
84.       Serial.print(F("AT+RST\r\n"));  //F(): Store string constants in Flash flash to avoid memory depletion due
    to SRAM usage.
85.       delay(3000);
86.       Serial.print(F("AT+CWMODE=3\r\n"));  //set to softAP+station mode
87.       delay(300);
88.       Serial.print(F("AT+CWSAP=\"Cokoino_ESP8266-01\",\"12345678\",11,2\r\n"));//wifiname:Cokoino_ES
    P8266-01,wifipassword:12345678
89.       //channnel:11 Encryption mode:2 ;Encryption mode should not set to 1,otherwise the wifi can't set success
    ed
```

```
90.         delay(200);
91.         Serial.print(F("AT+CIPMUX=1\r\n"));//Enable multiple connections
92.         delay(200);
93.         Serial.print(F("AT+CIPSERVER=1,3001\r\n"));//Create the server. The default port is 333. Change the p
    ort to 3001, which is consistent with the APP
94.         delay(200);
95.         Serial.print(F("AT+CIPSTO=7000\r\n"));//Example Set the server timeout period to 7000 seconds
96.         delay(2000);
97.       }
98.       void judgement() {
99.
100.       if (comdata.length() > 0) {
101.         comdata += "\n";  //This sentence must be added, otherwise the matched command character is one less,
    and the newline is used to assist in the complete match.
102.         char buf[comdata.length()];
103.         comdata.toCharArray(buf, comdata.length());
104.         ms.Target(buf);
105.         char result = ms.Match("%c*%+IPD, ?[0-9]+, ?[0-9]+: ?([^%c]+)%c*$");
106.         if (result > 0) {
107.           ms.GetCapture(buf, 0);
108.           comdata = String(buf);
109.           delay(100);
110.         } else {
111.           result = ms.Match("%c*%s?([0-9]),%s?([^%c]+)%c*$");  // esp8266 Multi-channel supports up to 5 co
    nnections (id:0-4)
112.           if (result > 0) {
113.             char buf0[1];  // esp8266 In multi-channel mode. id of the connection at this time
114.             ms.GetCapture(buf0, 0);
115.             ms.GetCapture(buf, 1);
116.             comdata = String(buf);
117.             if (comdata == "CONNECT")//The APP successfully connects to the wifi of ESP-01 module
118.             {
119.               String receiveOkMs = "Welcome to connect to ESP8266-01, ID: " + String(buf0) + " .";//A successf
    ul connection message is displayed
120.               Serial.println("AT+CIPSEND=" + String(buf0) + "," + receiveOkMs.length() + "\r\n");
121.               delay(10);
122.               Serial.print(receiveOkMs);
123.             }
124.           }
125.         }
126.         //comdata = "";
```

```
127.        //return; // When debugging communication with the APP, it needs to be commented out when normal u
    se
128.
129.         if (comdata == phone1) {
130.           judge = 1;
131.         }
132.         // else if (comdata == phone1_5) {
133.           //judge = 2;
134.         //} else if (comdata == phone2) {
135.           //judge = 3;
136.         //} else if (comdata == phone2_5) {
137.           //judge = 4;
138.         //}
139.           else if (comdata == phone3) {
140.           judge = 5;
141.         }
142.           //else if (comdata == phone3_5) {
143.           //judge = 6;
144.         //} else if (comdata == phone4) {
145.           //judge = 7;
146.         //} else if (comdata == phone4_5) {
147.           //judge = 8;
148.           //}
149.           else if (comdata == phone5) {
150.           judge = 9;
151.         }
152.           //else if (comdata == phone6) {
153.           //judge = 10;
154.         //} else if (comdata == phone7) {
155.           //judge = 11;
156.         //} else if (comdata == phone8) {
157.           //judge = 12;
158.         //} else if (comdata == phone9) {
159.           //judge = 13;
160.         //}
161.           else if (comdata == phone10) {
162.           judge = 14;
163.         } else if (comdata == phone11) {
164.           judge = 15;
165.         }
166.           //else if (comdata == phone12) {
167.           //judge = 16;
```

```
168.          //} else if (comdata == phone13) {
169.            //judge = 17;
170.          //}
171.          else {
172.            judge = 9;
173.          }
174.          comdata = "";
175.        }
176.
177.        switch (judge) {
178.
179.          case 1:
180.            forward();
181.            break;
182.          case 5:
183.            backward();
184.            break;
185.          case 9:
186.            Stop();
187.            break;
188.          case 14:
189.            LED();
190.            break;
191.          case 15:
192.            music();
193.            delay(2000);
194.            judge = 9;
195.            break;
196.          default: break;
197.        }
198.      }
199.
200.      void forward()
201.      {
202.        analogWrite(NSLEEP1,140); //Start M1,M2 motor and set speed
203.        //analogWrite(NSLEEP1,Num);Num range from 0 to 255,The larger the Num, the higher the output voltag
      e and the faster the motor speed.
204.        analogWrite(NSLEEP2,140); //Start M3,M4 motor and set speed
205.        //analogWrite(NSLEEP2,Num);Num range from 0 to 255,The larger the Num, the higher the output voltag
      e and the faster the motor speed.
206.        digitalWrite(AIN1,HIGH);  //drive the M1 motor foward rotation
207.        digitalWrite(BIN1,HIGH);  //drive the M2 motor foward rottion
```

```
208.     digitalWrite(AIN2,HIGH);  //drive the M3 motor foward rotation
209.     digitalWrite(BIN2,HIGH);  //drive the M4 motor foward rottion
210.   }
211.
212.   void backward()
213.   {
214.     analogWrite(NSLEEP1,140); //Start M1,M2 motor and set speed
215.     analogWrite(NSLEEP2,140); //Start M3,M4 motor and set speed
216.     digitalWrite(AIN1,LOW);  //drive the M1 motor reverse rotation
217.     digitalWrite(BIN1,LOW);  //drive the M2 motor reverse rottion
218.     digitalWrite(AIN2,LOW);  //drive the M3 motor reverse rotation
219.     digitalWrite(BIN2,LOW);  //drive the M4 motor reverse rottion
220.   }
221.
222.   void Stop()
223.   {
224.     analogWrite(NSLEEP1,0);//stop M1,M2 motor
225.     analogWrite(NSLEEP2,0);//stop M3,M4 motor
226.     digitalWrite(LED1,LOW);//LED1 OFF
227.     digitalWrite(LED2,LOW);//LED2 OFF
228.   }
229.   void music()
230.   {
231.     for(int i = 0;i < 80; i++)
232.     {
233.       //The high and low levels constitute a square-wave signal that triggers the passive buzzer to work
234.       digitalWrite(buzzer_pin, HIGH);
235.       delay(1);
236.       digitalWrite(buzzer_pin, LOW);
237.       delay(1);
238.     }
239.     for(int j = 0;j< 100; j++)
240.     {
241.       digitalWrite(buzzer_pin, HIGH);
242.       delay(2);
243.       digitalWrite(buzzer_pin, LOW);
244.       delay(2);
245.     }
246.   }
247.   void LED()
248.   {
249.     for(int i = 0;i < 15; i++)
```

```
250.      {
251.        digitalWrite(LED1,HIGH);//LED1 ON
252.        digitalWrite(LED2,HIGH);//LED2 ON
253.        delay(100);
254.        digitalWrite(LED1,LOW);//LED1 OFF
255.        digitalWrite(LED2,LOW);//LED2 OFF
256.        delay(100);
257.        digitalWrite(LED1,HIGH);
258.        digitalWrite(LED2,HIGH);
259.        delay(100);
260.      }
261.    }
```

3. 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 Arduino, Raspberry Pi, Smart Cars, Robotics and other interesting products in science and technology, please continue to visit our Amazon Store by search for "**LK COKOINO**" on Amazon. We will continue to launch fun, cost-effective, innovative and exciting products.

Thank you again for choosing Cokoino products.

LK COKOINO