### ESP32

## 一、了解开发板

## (一) 认识引脚

认识开发板才能不接错线,否则,把线接错了也没有作用。



## (二)、点亮开发板 LED

拿到开发板第一步应该是点亮开发板 LED,因为这样可以确定开发板是否能正常工作,如 果开发板不能正确工作,再多的想法都不可能实现。

int LED=13;

```
void setup() {
```

// initialize digital pin LED\_BUILTIN as an output.
pinMode(LED, OUTPUT);

}

// the loop function runs over and over again forever
void loop() {
 digitalWrite(LED, HIGH); // turn the LED on (HIGH is the voltage level)
 delay(1000); // wait for a second
 digitalWrite(LED, LOW); // turn the LED off by making the voltage LOW

delay(1000);

// wait for a second

```
}
```

## (三)、查看 ESP32MAC 地址

有的路由器设置了 MAC 地址黑白名单联网要求,这个时候,必须在路由器登录页面里设置 MAC,否则就连不上网,无法进行后面的工作。

#include <WiFi.h>
#include <esp\_wifi.h>

```
// Set your new MAC Address
uint8_t newMACAddress[] = {0x32, 0xAE, 0xA4, 0x07, 0x0D, 0x66};
```

```
void setup(){
   Serial.begin(115200);
   Serial.println();
   WiFi.mode(WIFI_STA);
   Serial.print("[OLD] ESP32 Board MAC Address: ");
   Serial.println(WiFi.macAddress());
```

}

#### void loop(){



CONTO		9 <del></del> 3		$\times$		
				发送		
ts Jul 29 2019 12:21:46				^		
st:0x1 (POWERON_RESET), boot:0	x13 (SPI_FAST_FLASH_BOOT)					
onfigsip: 0, SPIWP:0xee						
Lk_drv:0x00,q_drv:0x00,d_drv:	0x00,cs0_drv:0x00,hd_drv:0x	00,wp_drv	:0x00			
ode:DIO, clock div:1						
bad:0x3fff0030,len:1344						
bad:0x40078000,len:13864						
ad:0x40080400,len:3608						
ntry 0x400805f0						
DLD] ESP32 Board MAC Address:	D4:8A:FC:CF:45:28					
				$\sim$		
	<b>执行符</b> 🗸 1152	00 波特家	~	清空输出		
	1102	100 /2010 <del>-</del>	<u> </u>	/月上189山		
			Star 10		and the second	
Blink   Arduino 1.8.19				$\times$		
件 骗铒 坝目 丄具 帮助						
件 骗領 坝日 上具 帮助				P		
				<u>9</u>	- п	×
H 構築 坝目 上具 帮助 ・ ・ Bink §	© COM6			<b>P</b>	- 🗆	X
HY 構築 坝目 上具 帮助 ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●	© COM6			2	- 🗆	× 发送
件 确領 项目 上具 帮助 Bink § include <wip1.h> include <esp_wifi.h></esp_wifi.h></wip1.h>	© COM6 ets Jul 29 2019 12:21:46			Ð	- 0	× 发送
Y 稱類 项目 上具 帮助 ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●	COM6 ets Jul 29 2019 12:21:46 rst:0x1 (POWERON RESET),boot:0x1	13 (SPI FAST )	FLASH B	<u>.</u>		× 发送
14 建築 项目 上具 帮助 ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●	© COM6 ets Jul 29 2019 12:21:46 rst:0x1 (FOWERON_RESET),boot:0x1 configsip: 0, SPIWF:0xee	13 (SPI_FAST_	FLASH_B	<u>р</u> - сот)		× 发送
午 補納 项目 上具 常助	<pre>COM6 ets Jul 29 2019 12:21:46 rst:0x1 (FOWERON_RESET),boot:0x1 configsip: 0, SPIWP:0xee clk_drv:0x00,q_drv:0x00,d_drv:0x </pre>	13 (SPI_FAST_) x00,cs0_drv:0:	FLASH_B	007)	- 🗆	× 发送
(平 補销 项目 上具 帮助 (子 補销 项目 上具 帮助 Blink § Blink § Include <wipi.h> include <esp_wifi.h> ( Set your new MAC Address  Int8_t newMACAddress[] = {0x32, 0xAE, 0xB oid setup() { Serial.begin (115200);</esp_wifi.h></wipi.h>	<pre>COM6 ets Jul 29 2019 12:21:46 rst:0x1 (POWERON_RESET), boot:0x1 configsip: 0, SPIWF:0xee clk_drv:0x00,q_drv:0x00,d_drv:0x mode:DI0, clock div:1 load:0x3ff0030.len:1344</pre>	13 (SPI_FAST_1 x00,cs0_drv:0;	FLASH_B	00T) drv:0x00,wp		× 发送
<pre>/f 張翰 项目 上具 帮助 /f 張翰 项目 上具 帮助 Blink § include <wipi.h> include <esp_wifi.h> / Set your new MAC Address int8_t newMACAddress[] = {0x32, 0xAE, 0xA sid setup() {     Serial.begin(115200);     Serial.println(); }</esp_wifi.h></wipi.h></pre>	<pre>COM6 ets Jul 29 2019 12:21:46 rst:0x1 (FOWERON_RESET), boot:0x1 configsip: 0, SPIWP:0xee clk_drv:0x00,q_drv:0x00,d_drv:0x mode:DIO, clock div:1 load:0x36ff0030,len:1344 load:0x40078000,len:13864</pre>	13 (SPI_FAST_) x00,cs0_drv:0;	FLASH_B	00T)		× 发送
<pre>If 研知 项目 上具 帮助 If 研和 项目 上具 帮助 If 研和 view in the include <wipi.h> include <wipi.h> include <esp_wifi.h> / Set your new MAC Address int8_t newMACAddress[] = {0x32, 0xAE, 0xA Serial.hegis(115200); Serial.hegis(115200); Serial.println(); WiPi.mode(WIPI_SUA); Serial.print("[OID] EF822 Reard MAC Address]; </esp_wifi.h></wipi.h></wipi.h></pre>	<pre>COM6 ets Jul 29 2019 12:21:46 rst:0x1 (FOWERON_RESET),boot:0x1 configsip: 0, SPIWP:0xee clk_drv:0x00,q_drv:0x00,d_drv:0x mode:DIO, clock_div:1 load:0x3078000,len:13844 load:0x4078000,len:3864 load:0x40080400,len:3608</pre>	13 (SPI_FAST_1 x00,cs0_drv:0;	FLASH_B	COT)	- D	× 发送
<pre>14 英語 项目 上具 帮助</pre>	COM6 ets Jul 29 2019 12:21:46 rst:0x1 (FOWERON_RESET),boot:0x1 configsip: 0, SPIWF:0xee clk_drv:0x00,q_drv:0x00,d_drv:0x mode:DIO, clock div:1 load:0x345ff0030,len:1344 load:0x40078000,len:13864 load:0x40078000,len:3608 entry 0x400805f0	13 (SPI_FAST_1 x00,cs0_drv:0;	FLASH_B	00T)		× 发送
<pre>14 英語 项目 上具 帮助 14 英語 项目 上具 帮助 15 C C C C C C C C C C C C C C C C C C C</pre>	COM6 ets Jul 29 2019 12:21:46 rst:0x1 (FOWERON_RESET),boot:0x1 configsip: 0, SPWF:0xee clk_drv:0x00,q_drv:0x00,d_drv:0x mode:DIO, clock div:1 load:0x3ff0030,len:13844 load:0x40078000,len:13864 load:0x400806400,len:3608 entry 0.400805f0 [OLD] ESF32 Board MAC Abbass:	13 (SPI_FAST_ x00,cs0_drv:0: D4:8A:FC:CF:	FLASH_B0 x00,hd_,			× 发送
<pre>14 英語 项目 上具 帮助 14 英語 项目 上具 帮助 15 C C C C C C C C C C C C C C C C C C C</pre>	© COM6 ets Jul 29 2019 12:21:46 rst:0x1 (FOWERON_RESET),boot:0x1 configsip: 0, SPIWF:0xee clk_drv:0x00,q_drv:0x00,d_drv:0x mode:DIO, clock div:1 load:0x30ff0030,len:1344 load:0x40078000,len:13864 load:0x40078000,len:3608 entry 0x400805f0 [OLD] ESP32 Board MAC Abburgs:	13 (SPI_FAST_ x00,cs0_drv:0; D4:8A:FC:CF:-	FLASH_Be x00,hd_ 45:28			× 发送
<pre>14 英語 项目 上具 帮助 14 英語 项目 上具 帮助 15 C C C C C C C C C C C C C C C C C C C</pre>	© COM6 ets Jul 29 2019 12:21:46 rst:0x1 (FOWERON_RESET),boot:0x1 configsip: 0, SPIWF:0xee clk_drv:0x00,q_drv:0x00,d_drv:0x mode:DIO, clock div:1 load:0x30ff0030,len:1344 load:0x40078000,len:13864 load:0x40078000,len:3608 entry 0x400805f0 [OLD] ESP32 Board MAC Autorss:	13 (SPI_FAST_1 <00,cs0_drv:0: D4:8A:FC:CF:-	FLASH_B x00,hd_ 45:28			× 发送 000
<pre>14 英語 项目 上具 帮助 14 英語 项目 上具 帮助 15 C C C C C C C C C C C C C C C C C C C</pre>	<ul> <li>COM6</li> <li>ets Jul 29 2019 12:21:46</li> <li>rst:0x1 (FOWERON_RESET),boot:0x1</li> <li>configsip: 0, SPIMP:0xee</li> <li>clk_drv:0x00,q_drv:0x00,d_drv:0x</li> <li>mode:DIO, clock div:1</li> <li>load:0x30ff0030,len:1344</li> <li>load:0x40078000,len:13664</li> <li>load:0x400806400,len:3608</li> <li>entry 0:400805f0</li> <li>[OLD] ESP32 Board MAC Automss:</li> <li>□自动滚用 □ Show timestamp</li> </ul>	13 (SPI_FAST_1 «00,cs0_drv:0: D4:8A:FC:CF:- 换行	FLASH_B x00,hd_ 45:28 祥		- □ _drv:0x	× 发送 000
<pre>14 英語 项目 上具 帮助 14 英語 项目 上具 帮助 15 C C C C C C C C C C C C C C C C C C C</pre>	<ul> <li>COM6</li> <li>ets Jul 29 2019 12:21:46</li> <li>rst:0x1 (FOWERON_RESET),boot:0x1</li> <li>configsip: 0, SPINF:0xee</li> <li>clk_drv:0x00,q_drv:0x00,d_drv:0x</li> <li>mode:DIO, clock div:1</li> <li>load:0x3fff0030,len:1344</li> <li>load:0x40078000,len:13664</li> <li>load:0x400800400,len:3608</li> <li>entry 0.400805f0</li> <li>[OLD] ESP32 Board MAC Automss:</li> <li>□ 自动漆屏 □ Show timestamp</li> </ul>	13 (SPI_FAST_1 x00,cs0_drv:0; D4:8A:FC:CF: 换行	FLASH_B x00,hd_ 45:28 符	COT) drv:0x00,wp	- □ _drv:0x	× 友送 000 清空输。
<pre>14 英語 项目 上具 帮助 14 英語 项目 上具 帮助 15 14 45 14 14 14 14 14 14 14 14 14 14 14 14 14</pre>	<ul> <li>COM6</li> <li>ets Jul 29 2019 12:21:46</li> <li>rst:0x1 (POWERON_RESET), boot:0x1</li> <li>configsip: 0, SPIWF:0xee</li> <li>clk_drv:0x00, q_drv:0x00, d_drv:0x</li> <li>mode:DIO, clock div:1</li> <li>load:0x3fff0030,len:1344</li> <li>load:0x40078000,len:13664</li> <li>load:0x40080400,len:3608</li> <li>entry 0x400805f0</li> <li>[OLD] ESP32 Board MAC Autorss:</li> <li>□ 自动滚屏 □ Show timestamp</li> </ul>	13 (SPI_FAST_1 x00,cs0_drv:0; D4:8A:FC:CF:: 换行	FLASH_B x00,hd_ 45:28	OOT) drv:0x00,wp	- □ _drv:0x	× 友送 000
<pre>14 英語 项目 上具 帮助 14 英語 项目 上具 帮助 15 14 25 15 15 15 15 15 15 15 15 15 15 15 15 15</pre>	<ul> <li>COM6</li> <li>ets Jul 29 2019 12:21:46</li> <li>rst:0x1 (FOWERON_RESET), boot:0x1</li> <li>rofigsip: 0, SFIMF:0xee</li> <li>clk_drv:0x00,q drv:0x00,d_drv:0x</li> <li>mode:DIO, clock div:1</li> <li>load:0x36f0030,len:1344</li> <li>load:0x40078000,len:13644</li> <li>load:0x40078000,len:3608</li> <li>entry 0.400805f0</li> <li>[OLD] ESP32 Board MAC Autorgs:</li> <li>i 自动滚屏 ] Show timestamp</li> </ul>	13 (SPI_FAST_) x00,cs0_drv:0; D4:8A:FC:CF:- 换行 (61.1 kbit/s).	FlaSH_B ×00,hd_, 45:28 符	ooT) drv:0x00,wp	- 口 _drv:0x	× 发送 000 清空输
<pre>14 英語 项目 上具 帮助 14 英語 项目 上具 帮助 15 14 45 15 15 15 15 15 15 15 15 15 15 15 15 15</pre>	<ul> <li>COM6</li> <li>ets Jul 29 2019 12:21:46</li> <li>rst:0x1 (FOWERON_RESET), boot:0x1</li> <li>configsip: 0, SFINF!0xee</li> <li>clk_drv:0x00,q drv:0x00,d_drv:0x</li> <li>mode:DIO, clock div:1</li> <li>load:0x36f0030,len:1344</li> <li>load:0x40078000,len:13664</li> <li>load:0x40078000,len:3864</li> <li>load:0x40080400,len:3608</li> <li>entry 0.400805f0</li> <li>[OLD] ESP32 Board MAC Automss:</li> <li>○自动漆屏 □ Show timestamp</li> </ul>	13 (SPI_FAST_) x00,cs0_drv:0; D4:8A:FC:CF: 换行 (61.1 kbit/s).	FLASH_B k00,hd_, 45:28 符	ooT) drv:0x00,wp	- 口 drv:0x	× 友送 000
<pre>14 英語 项目 上具 帮助 14 英語 项目 上具 帮助 15 Control Control</pre>	<ul> <li>COM6</li> <li>ets Jul 29 2019 12:21:46</li> <li>rst:0x1 (FOWERON_RESET), boot:0x1</li> <li>configsip: 0, SPIWP:0xee</li> <li>clk drv:0x00, d drv:0x00, d drv:0x</li> <li>mode:DIO, clock div:1</li> <li>load:0x30ff0030,len:1344</li> <li>load:0x40078000,len:3864</li> <li>load:0x40080400,len:3608</li> <li>entry 0x400805f0</li> <li>[OLD] ESP32 Board MAC Abdurgs:</li> <li>○自动滚用 Show timestamp</li> </ul>	13 (SPI_FAST_) x00,cs0_drv:0; D4:8A:FC:CF: 换行	FLASH_B k00,hd_v 45:28 符	COT) drv:0x00,wp	- □ _drv:0x	× 发送 000
<pre>14 英語 项目 上具 帮助 14 英語 项目 上具 帮助 15 Control Contro</pre>	<ul> <li>COM6</li> <li>ets Jul 29 2019 12:21:46</li> <li>rst:0x1 (FOWERON_RESET), boot:0x1 configsip: 0, SPIWP:0xee</li> <li>clk drv:0x00, q drv:0x00, d drv:0x</li> <li>mode:DIO, clock div:1 load:0x30ff0030,len:1344</li> <li>load:0x4078000,len:13864</li> <li>load:0x40080400,len:3608</li> <li>entry 0x400805f0</li> <li>[OLD] ESF32 Board MAC Abdress:</li> <li>☑ 自动滚屏 □ Show timestamp</li> </ul>	13 (SPI_FAST_1 x00,cs0_drv:0; D4:8A:FC:CF: 换行	FLASH_B k00,hd_v 45:28	COT) drv:0x00,wp	- □ _drv:0x	× 发送 000

注意串口值要对应。

# 二、局域网

局域网在实验教学中基本没什么用,只是可以在办公室里试试各种感应器,但是后面的配 置也要用到,所以从局域网开始。

# (一) 联网

局域网要用到 WiFi.h,默认情况下,arduino IDE 安装好 ESP32 就包含了该库的,直接: #include <WiFi.h> 就可以使用该库了。

主要的函数有:

一、WiFi.mode();设置配网模式

static bool mode(wifi\_mode\_t);

wifi_mode_t 的四个选项	
WIFI_OFF	关闭配网模式
WIFI_STA	设置为 STA 模式
WIFI_AP	设置为 AP 模式
WIFI_AP_STA	设置为 AP 和 STA 共存模式

# (1)别的设备连接 ESP32——AP 配网

ESP32 的 AP 配网模式可以通过无线 WIFI 连接的方式来连接来控制 ESP32 或获取 ESP32 的数据。

函数:

(1) WiFi.softAP();

设置 AP 的 WIFI 属性

bool softAP(const char\* ssid, const char\* passphrase = NULL, int channel = 1, int ssid\_hidden = 0, int
max\_connection = 4, bool ftm\_responder = false);

ssid	设置 SSID
passphrase	设置密码
channel	设置通道,默认为1
ssid_hidden	是否隐藏,默认为0不隐藏
max_connection	最大连接数量,默认为4
ftm_responder	测试响应,默认为 false

(2) WiFi.softAPConfig();

设置 AP 的 IP, 网关, 子网掩码, DHCP

bool softAPConfig(IPAddress local\_ip, IPAddress gateway, IPAddress subnet, IPAddress
dhcp\_lease\_start = INADDR\_NONE);

local_ip	设置 IP 地址
gateway	设置网关
subnet	设置子网掩码
dhcp_lease_start	设置 DHCP,默认为打开

最简单的配网就这三个函数实现,完整代码:

#include <WiFi.h>

IPAddress AP_local_ip(10,0,1,1);	//IP 地址
IPAddress AP_gateway(10,0,1,1);	//网关地址
IPAddress AP_subnet(255,255,255,0);	//子网掩码
const char* AP_ssid = "shuyang";	//SSID

```
const char* AP_password = "12345678";
                                                         //密码
void setup() {
  WiFi.mode(WIFI AP);
  WiFi.softAPConfig(AP_local_ip, AP_gateway, AP_subnet);
  WiFi.softAP(AP_ssid, AP_password);
}
void loop() {
}
1_AP | Arduino 1.8.19
文件 编辑 项目 工具 帮助
 1_AP §
#include <WiFi.h>
IPAddress AP_local_ip(10,0,1,1);
                                        //IP地址
                                        //网关地址
IPAddress AP_gateway(10,0,1,1);
IPAddress AP_subnet(255,255,0);
const char* AP_ssid = "shuyang";
const char* AP_password = "12345678";
                                       //子网掩码
                                       //SSID
                                       //密码
void setup() {
 WiFi.mode (WIFI_AP);
  WiFi.softAPConfig(AP_local_ip, AP_gateway, AP_subnet);
  WiFi.softAP(AP_ssid, AP_password);
}
void loop() {
}
```

把以上代码上传到 ESP32 后,打开电脑或手机上的 WIFI 连接界面,就可以看到一个名称为 shuyang 的路由器,这时就可以输入密码"12345678"进行连接测试了。



# (2) ESP32 连接路由器——STA 模式

主要的函数有:

─、 WiFi.begin();

初始化 WIFI 连接

wl\_status\_t begin(const char\* ssid, const char \*passphrase = NULL, int32\_t channel = 0, const uint8\_t\* bssid = NULL, bool connect = true);

ssid	SSID
passphrase	密码
channel	通道
bssid	BSSID,对应 MAC 地址,默认为 NULL
connect	是否连接,默认为 true

 $\equiv$ 、WIFI.status()

wl\_status\_t status()

获取 WIFI 的连接状态

参数	无		
返回	WL_CONNECT_FAILED	未连接	
	WL_CONNECTED	已连接	

三、WIFI.localIP()

wl\_status\_t localIP() -获取 ESP32 的本地 IP 地址

参数	无		
返回	IPAddress	IP 地址	

最简单的连接 WIFI 的完整代码:

注意,代码内的,WIFI 连接名称和密码需要根据你的路由名称和密码做出更改 #include <WiFi.h>

const char\* wifi\_ssid = "@Ruijie-sAF11"; //自己的 WIFI 账号 const char\* wifi\_password = "luo200408"; //自己的 WiFi 密码

void setup() {

```
Serial.begin(9600);
  WiFi.mode(WIFI_STA);
                                       //连接 WIFI
  WiFi.begin(wifi_ssid, wifi_password);
  Serial.print("WIFI 已连接");
  //循环,直到连接成功
  while(WiFi.status() != WL_CONNECTED){
    Serial.print(".");
    delay(500);
  }
  Serial.println();
  IPAddress local_IP = WiFi.localIP();
  Serial.print("WIFI 连接成功,The local IP address is "); //连接成功提示
  Serial.println(local_IP);
                                                     //输出 ESP32 本地 IP 地址
}
```

```
void loop() {
```

}			
🐵 Blink   Arduino 1.8.19	_		×
文件 编辑 项目 工具 帮助			
O D D D D Blink § const char* wifi ssid = "@Ruijie-saF11"; //自己的WIFI账号			•
const char* wifi_password = "luo200408"; //自己的wiFi密码			
<pre>void setup() {     Serial.begin(9600);     WiFi.mode(WIFI_9TA);     WiFi.begin(wifi_ssid, wifi_password); //连接WIFI     Serial.print("WIFICizik");     //循环, 直到注接成功     while(WiFi.status() != WL_CONNECTED)[]     Serial.print(".");     delay(500);     }     Serial.println();     IPAddress local_IF = WiFi.localIF();     Serial.print("WIFICizik@u),The local IP address is "); //注接成功提示     Serial.println(local_IF); //输出ESP32本地IF地址     }     void loop() {     </pre>			
1			
上传成功。			•
<pre>#riting at 0x00089a11 (75 %) #riting at 0x0008f4e3 (78 %) #riting at 0x0009508 (82 %) #riting at 0x0009edd5 (85 %) #riting at 0x000a5168 (89 %) #riting at 0x000a502 (92 %) #riting at 0x000b003f (96 %) #riting at 0x000b05b71 (100 %) #rote 660224 bytes (443367 compressed) at 0x00010000 in 6.0 seconds (effective 909.6 kbit Hash of data verified.</pre>	:/s)		^
Leaving Mard resetting via RTS pin			~
15 ESP32 Dev Module, Disabled, Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS), 240MHz (WiFi/BT), OIO, 80MHz, 4MB (32Mb), 921600, Core	1, Core 1	, None 在	СОМб

以上代码上传到 ESP32 前注意打开串口监视器,如果连接成功,将会输出连接成功的提示。

# (3) AP 模式与 STA 模式共存

在该模式下,可以同时接受 WIFI 连接到 ESP32,也可以把 ESP32 连接到已有的 WIFI 上。 简单的 AP 模式与 STA 模式共存完整代码: #include <WiFi.h>

//IP 地址 IPAddress AP local ip(10,0,10,1); //网关地址 IPAddress AP gateway(10,0,10,1); //子网掩码 IPAddress AP subnet(255,255,255,0); const char\* AP\_ssid = "shuyang"; //SSID const char\* AP\_password = "12345678"; //密码 const char\* wifi ssid = "@Ruijie-sAF11"; //不要与上面的混淆,这是真实的 WIFI 账号 //不要与上面的混淆,这是真实的 WIFI 密码 const char\* wifi password = "luo200408"; void setup() { Serial.begin(9600); WiFi.mode(WIFI\_AP\_STA); WiFi.softAPConfig(AP\_local\_ip, AP\_gateway, AP\_subnet); WiFi.softAP(AP ssid, AP password); WiFi.begin(wifi ssid, wifi password); //连接 WIFI Serial.print("Connected"); //循环,直到连接成功 while(WiFi.status() != WL\_CONNECTED){

```
Serial.print(".");
delay(500);
}
Serial.println();
IPAddress local_IP = WiFi.localIP();
Serial.print("WIFI is connected,The local IP address is "); //连接成功提示
Serial.println(local_IP); //输出本地 IP 地址
}
```

void loop() {

#### 



### 浏览器里没内容。

<ul> <li>@Ruijie-sAF11</li> <li>已连接,安全</li> </ul>	
属性	
	断开连接
HUAWEI-AA168D	
shuyang	

WIFI 里多了一个 shuyang 的 WIFI。

# 小结

WiFi.mo de(WIFI	WiFi.mod e(WIFI_A P);	<pre>softAP(const char* ssid, const char* passphrase = NULL, int channel = 1, int ssid_hidden = 0, int max_connection = 4, bool ftm_responder = false); softAPConfig(IPAddress local_ip, IPAddress gateway, IPAddress subnet, IPAddress dhcp_lease_start = INADDR_NONE);</pre>
	WiFi.mod	WiFi.begin();
WIFI_AP	e(WIFI_S	WIFI.status()
` \λ/IEI ΔΡ	TA);	WIFI.localIP()
STA)	WiFi.mod	
_317,	e(WIFI_A	
	P_STA);	

都是先设置 WIFI 模式,根据不同模式有不同的函数。

## (二)、上传内置网页

## (1) 库的安装

1)、必备库

常用的 EPS32 可用于 WEB 服务器的库有两个:

WebServer

ESPAsyncWebServer

默认情况下, arduino IDE 安装好就包含了 WebServer 这个库的, 直接:

#include <WebServer.h>

就可以使用该库了。

#### 2)、安装另外的必须库

#### 1.下载 ESPAsyncWebServer、AsyncTCP 库

ESPAsyncWebServe 库依赖 AsyncTCP 库,所以,这两个库都需要另行添加,这两个库的项目 地址分别为:

https://github.com/me-no-dev/ESPAsyncWebServer

https://github.com/me-no-dev/AsyncTCP

进入项目地址后,点击 Clone=》Download ZIP

(百度网盘链接)

#### 2.下载好后把这两个库放在任意文件夹都可以

#### 3.安装

打开 arduino IDE,菜单=》项目=》加载库=》添加.ZIP 库...,重复两次,分别选择上述的两个 库进行添加。

😂 4_WEB   A	rduino 1.8.19			- 🗆
文件编辑 项	目 工具 帮助			
4 WEB	验证/编译 上传 使用编程器上传	Ctrl+R Ctrl+U Ctrl+Shift+U	Δ	
const	导出已编译的二进制文件	Ctrl+Alt+S	管理库	Ctrl+Shift+I
Serial WiFi.b	显示项目文件夹 加载库	Ctrl+K	添加 .ZIP 库	
Serial //循环	添加文件		Arduino 库 Bridge	

安装好后,可以在项目文件夹里看到好像是解压的文件

▶ 此目	皀脑 > 文档 (E:) > libraries >	v ♡ .	搜索"libraries"
	名称 ^	修改日期	类型
	AsyncTCP-master	2023/12/31 23:39	文件夹
~	ESPAsyncWebServer-master	2023/12/31 23:37	文件夹
Я Я	🜏 readme.txt	2023/12/31 23:34	TXT 文件

所以,前面的项目文件不要随便设置。 为了方便测试,该文是选择使用 STA 模式。

## (2) 内置文本

(1) 引入需要用到的库:

#include <WiFi.h>

\*

#include "ESPAsyncWebServer.h"

AsyncWebServer server(80);

```
(2) 连接 WIFI
```

void connect wifi(){

const char\* wifi ssid = "@Ruijie-sAF11";

```
const char* wifi password = "luo200408"; //不要与上面的混淆,这是真实的 WIFI 密码
```

Serial.begin(9600);

WiFi.begin(wifi\_ssid, wifi\_password); Serial.print("Connected");

//循环,直到连接成功

while(WiFi.status() != WL\_CONNECTED){

Serial.print(".");

delay(500);

} Serial.println(); IPAddress local\_IP = WiFi.localIP(); Serial.print("WIFI is connected,The local IP address is "); //连接成功提示 Serial.println(local\_IP);

#### }

在 setup()里调用该函数。

(3) 创建一个 WebServer 对象

AsyncWebServer server(80);

这行代码是调用了 ESPAsyncWebServer.h 库文件(详细在下面的 ESPAsyncWebServer.h 文件片断中)

//不要与上面的混淆,这是真实的 WIFI 账号

//连接 WIFI

中的 AsyncWebServer(uint16\_t port),来创建一个 server 对象, port 参数为连接端口。

(4) ESPAsyncWebServer.h 解析

### [1]

class AsyncWebServer {

protected:

AsyncServer \_server; LinkedList<AsyncWebRewrite\*> \_rewrites; LinkedList<AsyncWebHandler\*> \_handlers; AsyncCallbackWebHandler\* \_catchAllHandler;

public:

```
AsyncWebServer(uint16_t port);
~AsyncWebServer();
```

void begin();

void end();

#### 2

```
然后是需要注册一个响应链接"/"上的 GET 请求的处理回调函数
```

server.on("/",HTTP\_GET,call\_back);

这行代码是当收到根目录"/"的 GET 请求时,调用 call\_back 这个回调函数来响应请求,这个 函数内容为:

#### void call\_back(AsyncWebServerRequest \*request){

```
request->send(200,"text/plain","课堂小实验");
```

}

这个回调函数默认需要传入一个 AsyncWebServerRequest 对象,该对象包含了客户端的请求 send()函数会在收到请求后,发送一个字符串("text/plain")内容("hello esp32 web server")的基本 (200)响应。 最后需要初始化服务器: server.begin(); 该方法将启动服务器。 完整代码: #include <WiFi.h> #include "ESPAsyncWebServer.h" AsyncWebServer server(80); //连接 WIFI void connect\_wifi(){ const char\* wifi ssid = "@Ruijie-sAF11"; //自己的 WIFI 账号 const char\* wifi\_password = "luo200408"; //自己的 WIFI 密码 Serial.begin(9600); //连接 WIFI WiFi.begin(wifi\_ssid, wifi\_password);

Serial.print("Connected"); //循环,直到连接成功

77個本,直到是彼成功

while(WiFi.status() != WL\_CONNECTED){

Serial.print(".");

```
delay(500);
  }
  Serial.println();
  IPAddress local_IP = WiFi.localIP();
  Serial.print("WIFI is connected,The local IP address is "); //连接成功提示
  Serial.println(local_IP);
}
void call back(AsyncWebServerRequest *request){
  Serial.println("User requested");
  request->send(200,"text/plain","课堂小实验"); //响应请求,"课堂小实验"这句话可以随便换
为其他的句子
}
void web_server(){
                                     //注册回调函数
  server.on("/",HTTP_GET,call_back);
                                        //初始化
  server.begin();
}
void setup() {
  connect_wifi();
  web_server();
}
void loop() {
  // put your main code here, to run repeatedly:
```

}

Blink   Arduino 1.8.19	7 <u>—</u> 7		×
文件 编辑 项目 工具 帮助			
			ø
Blink§			
<pre>Serial.println(local_IP); }</pre>			^
<pre>void call_back(AsyncWebServerRequest *request) {     Serial.println("User requested");     request-&gt;send(200,"text/plain","课堂小实验"); //响应请求,"课堂小实验"这句话可以随便换为其他的 }</pre>	]句子		
<pre>void web_server() {     server.on("/",HTTP_GET,call_back); //注册回调函数     server.begin(); //初始化 }</pre>			
<pre>void setup() {     connect_wifi();     web_server(); }</pre>			
<pre>void loop() {     // put your main code here, to run repeatedly:</pre>			
Þ			~
上传成功。			
<pre>writing at 0x000940b9 (75 %) Writing at 0x00099a0e (79 %) Writing at 0x000a2035 (82 %) Writing at 0x000a45ce (89 %) Writing at 0x000b56d0 (93 %) Writing at 0x000b56d0 (93 %) Writing at 0x000c059f1 (100 %) Writing at 0x000c059f1 (100 %) Wrote 736512 bytes (468854 compressed) at 0x00010000 in 6.7 seconds (effective 883.7 kbit. Hash of data verified.</pre>	/s)		^
Leaving Hard resetting via RTS pin			
		and the second	an a

把代码上传到 ESP32 后,完成 WIFI 的连接后,串口会输出一个本地 IP 地址,把该地址复制到浏 览器打开,浏览器将输出"hello esp32 web server"



# (3) 内置网页

#### 1) 网页植于代码

正常的网页,有HTML、CSS、JavaScript。 对于一个中文网页,需要使用 <meta charset="utf-8"> 基本的 HTML 页面已经有了, 接下来就是要在 ESP32 的 web 服务器 的基础上, 把响应浏览器的 请求换成上面的 HTML 代码。 首先,把这段 HTML 代码定义为一个 String 的变量: const String indexHtml PROGMEM = R"rawliteral( <!DOCTYPE html> <html> <head> <meta charset="utf-8"> <title>EPS32 教程</title> </head> <body> <h1>ESP32 HTML 标题</h1> ESP32 HTML 段落 </body> </html> )rawliteral"; 在这个变量里,使用了 PROGMEM 关键字,这个关键字的目的是将数据存储在 Flash(闪存)中而 不是 RAM(运行内存) 同时 R"rawliteral 是一种引入原始字符串的方法。引用该方法,就无需对比如斜杠"\"或换行符"\n" 之类的进行转义,非常方便。 然后,在响应请求的回调函数里发送这个变量,同时,发送的数据类型也相应得应该换成 "text/html": void call back(AsyncWebServerRequest \*request){

Serial.println("User requested");
request->send(200,"text/html",indexHtml); //响应请求
}
至此,就创建了一个基本的 HTML 服务器,完整的代码:

#include <WiFi.h>
#include "ESPAsyncWebServer.h"

AsyncWebServer server(80);

```
const String indexHtml PROGMEM = R"rawliteral(
<!DOCTYPE>
<html>
<head>
<meta charset="utf-8">
<title>罗孔均</title>
<script type="text/javascript">
alert("阳阳");
</script>
</head>
    <body>
    <h1>课堂物联网实验</h1>
    <h2>用物联网改进课堂教学</h2>
    </body>
</html>
)rawliteral";
//连接 WIFI
void connect wifi(){
                                         //不要与上面的混淆,这是真实的 WIFI 账号
  const char* wifi ssid = "@Ruijie-sAF11";
  const char* wifi password = "luo200408"; //不要与上面的混淆,这是真实的 WIFI 密码
  Serial.begin(9600);
  WiFi.begin(wifi_ssid, wifi_password);
                                         //连接 WIFI
  Serial.print("Connected");
  //循环,直到连接成功
  while(WiFi.status() != WL CONNECTED){
    Serial.print(".");
    delay(500);
  }
  Serial.println();
  IPAddress local IP = WiFi.localIP();
  Serial.print("WIFI is connected,The local IP address is "); //连接成功提示
  Serial.println(local_IP);
}
void call_back(AsyncWebServerRequest *request){
  Serial.println("User requested");
  request->send(200,"text/html",indexHtml); //响应请求
}
```

```
void web_server(){
```

```
server.on("/",HTTP_GET,call_back); //注册回调函数
server.begin(); //初始化
}
void setup() {
    connect_wifi();
    web_server();
}
void loop() {
    // put your main code here, to run repeatedly:
}
```



5_inside_HTML   Arduino 1.8.19	1000		$\times$
文件 编辑 项目 工具 帮助			
			ø
5_inside_HTML			
<pre>Serial.begin(9600); WiFi.begin(wifi_ssid, wifi_password); //连接W Serial.print("Connected"); //循环,直到连接成功 while(WiFi.status() != WL_CONNECTED){ Serial.print("."); delay(500); } Serial.println();</pre>	9IFI		^
<pre>IPAddress local_IP = WiFi.localIP(); Serial.print("WIFI is connected, The local IP address Serial.println(local_IP); }</pre>	: is "); /	/连接成功	提示
<pre>void call_back(AsyncWebServerRequest *request) {     Serial.println("User requested");     request-&gt;send(200,"text/html",indexHtml); //响应请对 }</pre>	Ŕ		
<pre>void web_server() {</pre>			
<pre>server.on("/",HTTP_GET,call_back); //注册回调函数 server.begin(); //初始化 }</pre>			
<pre>void setup() {     connect_wifi();     web_server(); }</pre>			
void loop() {     <			>
上传成功。			
Hash of data verified.			^
Leaving Hard resetting via RTS pin			~
3 503th spiffs (1.2MB APP/1.5MB SPIFFS), 240MHz (WiFi/BT), QIO, 80MHz, 4MB (32Mb), 9216	00, Core 1, Cor	re 1, None 在	СОМ6
💿 СОМ6			<u></u>
SConnected. WIFI is connected, The local IP address is 19	2.168.11	10.149	]

🗧 🥑 🤇 罗孔均	x +			
	http://192.168.110.149/			
	192.168.110.149 显示 阳阳			
🗧 🥑 🕀 罗孔均	× +			
← → C D <	a 💿 🍂 http://192.168.110.149/			
课堂物联网实验				
用物联网改进课堂教学				

2) SPIFFS 存放前端

### 【1】SAT 模式——电脑直接访问

SPIFFS 相当于 ESP32 中的一个硬盘分区。

首先,建一个 html 文件(index.html),文件需要保存在项目文件夹内的"data"文件夹内(不是设置时的项目,而是指 Arduino 代码的文件夹。比如这里的 Arduino 保存为 6\_SPIFFS\_html,就在这个文件夹里新建一个 data 文件夹)。 注意:,



```
<meta charset="utf-8">
   k href="yilian.css" rel="stylesheet" type="text/css" >
   <script src="yangyang.js"></script>
   <title>罗孔均</title>
 </head>
 <body>
   <h1>课堂物联网</h1>
   <h2>用物联网改进课堂实验</h2>
   <input type="button" value="点击" onclick="shuyang()">
 </body>
</html>
然后新建一个 CSS 文件(yilian.css):
h1{
color:red;
}
h2{
color:green;
}
然后,把文件上传到 ESP32 的 SPIFFS 分区中。
先要在 arduion IDE 中安装一个插件(ESP32FS),该插件的项目地址在:
https://github.com/me-no-dev/arduino-esp32fs-plugin
找到 releases page 下载这个插件。
点击下载 ESP32FS-1.1.zip
解压缩该插件到 arduino IDE 安装文件夹下的 tools 文件夹。
```



	名称	修改日期
	drivers	2021/1
	examples	2021/1
*	📕 hardware	2021/1
A	📕 java	2021/1
*	📜 lib	2021/1
	libraries	2023/1
	📕 tools	2024/1
	Lools-builder	2021/1
	arduino.exe	2021/1

(注意:不是解压后复制最后的文件,而是解压后是一个文件夹,把这个文件夹复制进去。)

« sot	ftwall > statisc > arduino > tools > 名称		
Я Я Я	<ul> <li>ESP32FS</li> <li>Mangler</li> <li>WiFi101</li> <li>howto.txt</li> </ul>		
灵	查看	文	件夹
此电	围脑 〉 软件 (D:) 〉 softwall 〉 stat	isc > arduino > tools > ESP32I	=S → tool
	名称 ^	修改日期	类型
*	esp32fs.jar	2019/1/15 10:53	Executable Jar File

重启 arduino IDE 在菜单=>工具 下会出现新的选项菜单:ESP32 Sketch Data Upload

6 SPIFFS htr	nl   Arduino 1.8.19 — 🗆 🗙	样式
· · · · · · · · · · · · · · · · ·	工具 帮助	
	自动格式化 项目存档 修正编码并重新加载	Ctrl+T
b_SPIFFS_ntn	管理库	Ctrl+Shift+I
,	串口监视器	Ctrl+Shift+M
void setup()	串口绘图器	Ctrl+Shift+L
connect_wi	ESP32 Sketch Data Upload	
}	WiFi101 / WiFiNINA Firmware Updater	
	开发板: "ESP32 Dev Module"	>
void Toob()	Upload Speed: "921600"	>
1	CPU Frequency: "240MHz (WiFi/BT)"	>
*	Flash Frequency: "80MHz"	>
<	Flash Mode: "QIO"	>

点击后,该插件会把前面提到的"data"文件夹内的所有文件上传到 SPIFFS 分区。

上传之前,一定要注意先把"串口监视器"关闭,关闭对上传端口的占用。 上传完成会输出提示信息:

Connecting.... Chip is ESP32-DOWD-V3 (revision 3) Features: WiFi, BT, Dual Core, 240MHz, VRef calibration in efuse, Coding Schem Crystal is 40MHz MAC: d4:8a:fc:cf:45:28 Uploading stub... Running stub ... Stub running ... Changing baud rate to 921600 Changed. Configuring flash size... Auto-detected Flash size: 4MB Flash will be erased from 0x00290000 to 0x003fffff... Compressed 1507328 bytes to 3318... Writing at 0x00290000... (100 %) Wrote 1507328 bytes (3318 compressed) at 0x00290000 in 4.5 seconds (effective Hash of data verified. Leaving... Hard resetting via RTS pin...

```
至此, 文件就上传至 SPIFFS 分区
```

要使用 ESP32 的 SPIFFS,要先引入 SPIFFS.h,在 arduion IDE 中,如果已经搭建好 ESP32 环境,默认 是自带这个库的。 同时需要做的是引入 WiFi.h 和 ESPAsyncWebServer.h 这两个库 #include <WiFi.h> #include <SPIFFS.h> #include "ESPAsyncWebServer.h" ESPAsyncWebServer 库中, send()结构体提供了直接读取 SPIFFS 中文件的方法。 首先需要对 SPIFFS 进行初始化,为了方便理解,还是和之前一样把一些功能做成函数: void call back(AsyncWebServerRequest \*request){ if(!SPIFFS.begin(true)){ Serial.println("An Error has occurred while mounting SPIFFS"); return; } request->send(SPIFFS,"/index.html"); } void web\_server(){ if(!SPIFFS.begin(true)){ Serial.println("An Error has occurred while mounting SPIFFS");

```
return;
```

```
}
server.serveStatic("/", SPIFFS, "/").setDefaultFile("index.html");
server.begin(); //初始化
}
```

完整代码:

#include <WiFi.h>
#include <SPIFFS.h>
#include "ESPAsyncWebServer.h"

AsyncWebServer server(80);

```
//连接 WIFI
```

```
void connect_wifi(){
  const char* wifi_ssid = "ESP32";
  const char* wifi_password = "12345678";
  Serial.begin(9600);
                                                //连接 WIFI
  WiFi.begin(wifi ssid, wifi password);
  Serial.print("Connected");
  //循环,直到连接成功
  while(WiFi.status() != WL_CONNECTED){
    Serial.print(".");
    delay(500);
  }
  Serial.println();
  IPAddress local IP = WiFi.localIP();
  Serial.print("WIFI is connected,The local IP address is "); //连接成功提示
  Serial.println(local_IP);
}
void web_server(){
  if(!SPIFFS.begin(true)){
```

```
Serial.println("An Error has occurred while mounting SPIFFS");
return;
}
```

```
}
```

```
void setup() {
    connect_wifi();
    web_server();
}
```

void loop() {



#### 【2】以 AP\_STA 和域名连接

用 DNSServer 库可以实现用域名,也就是用比较方便记忆的网址的形式来对 ESP32 进行访问。 在 arduion IDE 中,如果已经搭建好 ESP32 环境,默认是自带这个库的: 以前文中的代码为基础,我们增加这个库的引用: #include <WiFi.h> #include <SPIFFS.h> #include <DNSServer.h> #include "ESPAsyncWebServer.h" 因为 DNSServer 只能在 AP 模式下才能实现,所以需要开启 AP 和 STA 共存的方式。 之后需要创建一个 DNSServer 实例: DNSServer dnsserver; 实现的主要方法: 一、.start();启动 DNSServer 服务 bool start(const uint16\_t &port, const String &domainName, const IPAddress &resolvedIP); 参数: 服务端口, DNS 默认端口 53 1:&port 域名,如"esp32 ap.com",域名最好不要设置为互联网存在 2:&domainName 的网址 解析域名所对应的 IP 地址 3:&resolvedIP 二、.processNextRequest()方法,监测客户端的 DNS 请求,需要放到 loop 循环里。 完整代码: #include <WiFi.h> #include <SPIFFS.h>

#include <DNSServer.h> #include "ESPAsyncWebServer.h"

DNSServer dnsserver;

AsyncWebServer server(80);

//连接 WIFI

void connect\_wifi(){ //DNS 端口 const byte DNS PORT = 53; //域名 const String url = "yang.com"; IPAddress APIp(10,0,10,1); //AP IP IPAddress APGateway(10,0,10,1); //AP 网关 IPAddress APSubnetMask(255,255,255,0); //AP 子网掩码 const char\* APSsid = "yang"; //AP SSID const char\* APPassword = ""; //AP wifi 密码 //不要与上面的混淆,这是真实的 WIFI 账号 const char\* wifi\_ssid = "@Ruijie-sAF11"; const char\* wifi\_password = "luo200408"; //不要与上面的混淆,这是真实的 WIFI 密码 Serial.begin(9600); //打开 AP 和 STA 共存模式 WiFi.mode(WIFI AP STA); //设置 AP 的 IP 地址,网关和子网掩码 WiFi.softAPConfig(APIp, APGateway, APSubnetMask);

//设置 AP 模式的登陆名和密码 WiFi.softAP(APSsid, APPassword, 6); dnsserver.start(DNS\_PORT, url, APIp); //设置 DNS 的端口、网址、和 IP WiFi.begin(wifi\_ssid, wifi\_password); //连接 WIFI Serial.print("Connected"); //循环,直到连接成功 while(WiFi.status() != WL CONNECTED){ Serial.print("."); delay(500); } Serial.println(); IPAddress local\_IP = WiFi.localIP(); Serial.print("WIFI is connected,The local IP address is "); //连接成功提示 Serial.println(local IP); }

```
void web_server(){
    if(!SPIFFS.begin(true)){
        Serial.println("An Error has occurred while mounting SPIFFS");
        return;
    }
```

```
server.serveStatic("/", SPIFFS, "/").setDefaultFile("index.html");
server.begin(); //初始化
}
void setup() {
    connect_wifi();
    web_server();
}
void loop() {
    dnsserver.processNextRequest();
}
```

这个时候,因为电脑已经连接了 ESP32,就无法访问 "WIFI is connected, The local IP address is 192.168.110.149"路由器分配给电脑的地址,如图



#### 课堂应用

把实验文件传至 ESP32。

注意:班班通不再连接路由器,而是连接 ESP32 发出的信号,就可以访问了。 电脑连接的时候,注意把自动连接打上勾。



<ul> <li>              ● ● 第3比特      </li> <li>             ← → C 白 畑 ● 第4         </li> </ul>	× ● 网页无法访问 ttp://10.0.10.1/	x   +	
课堂物联网			
用物联网改进课堂实验	È		
ALL	这是连持	<b>接手机的实验</b> 10.0.10.1 顯示	
· · · · ·		小阳阳,这是用手机连续的	織定

#### 问题解决:

上课不可能带着路由器走,去上课,但是可以带手机走,把手机开热点,让 ESP32 连上手机就可以了;也就是说,把代码里的 WIFI 账号和密码都设为手机的就可以了。

注意:

(1)如果为了简单,在前面设置的时候可以设置为 1,2,3,4——但是要和网关一起变,否则连不上。

(2) 在 ESP32 运行代码的时候不能断开手机的 WIFI 连接。

//连接wiFi

void connect_wifi() { 	//DNS端口
IPAddress APIp(1,2,3,4); IPAddress APGateway(1,2,3,4);	//AP IP //AP网关
IPAddress APSubnetMask(255,255,255,0);	//ap子网 <mark>掩码</mark>

用上面的技术,基本上可以在课堂上演示用的试验了。