

LẬP TRÌNH ESP32

Bài 2: Lập trình ESP32 Webserver chế độ Access Point (WIFI AP Mode)

POSTED ON 12/07/2021 BY KHUÊ NGUYỄN

Bài 2 WIFI: Lập trình ESP32 Webserver chế độ Access Point

Trong bài này chúng ta cùng nhau học cách lập trình ESP32 Webserver chế độ Wifi Access Point hay còn gọi là Wifi AP mode, bật tắt led bằng web browser. Cùng tìm hiểu AP mode là gì và các hàm lập trình AP mode nhé!

Bài 2 Networking trong serie [Lập trình ESP32 từ A tới Z](#)

Mục Lục

1. Access Point là gì
2. Lập trình ESP32 Webserver chế độ Access Point
 - 2.1. Sơ đồ nguyên lý
3. Code và giải thích code
 - 3.1. Nạp code cho ESP32
 - 3.2. Kết Quả
4. Các hàm thường gặp trong ESP32 Access Point
 - 4.1. softAP
 - 4.2. softAPConfig
5. Quản lý kết nối
 - 5.1. softAPgetStationNum
 - 5.2. softAPdisconnect
6. Cấu hình Mạng
 - 6.1. softAPIP
 - 6.2. softAPmacAddress
7. Kết
 - 7.1. Related posts:

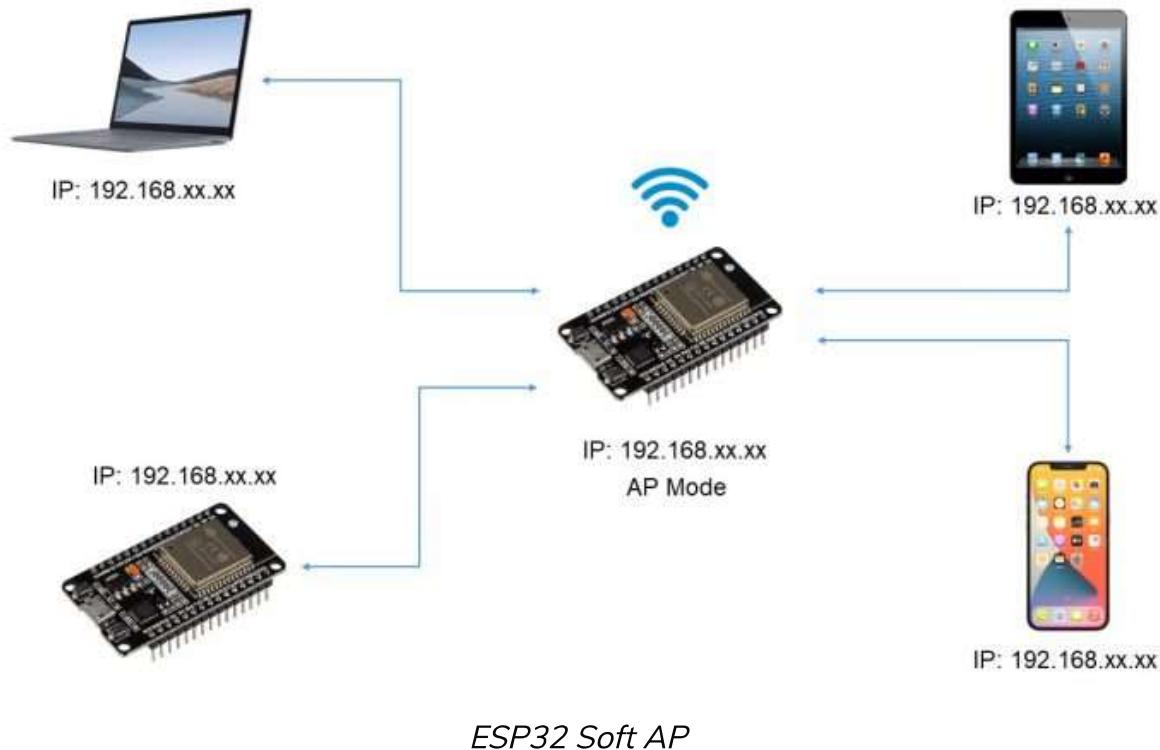


Access Point là gì

Access Point hay điểm truy cập, còn được gọi là Hostpot là thiết bị phát sóng Wifi, cung cấp khả năng kết nối giữa các máy trạm (Station) với nó và các máy trạm với nhau. Sau đó kết nối với Internet thông qua mạng có dây.

Ví Du: Trong gia đình cục Router wifi chính là một Access Point, nó cho phép chúng ta kết nối máy tính, điện thoại ... thông qua mạng wifi và từ đó kết nối với internet.

ESP32 cung cấp khả năng tạo kết nối Wifi Acces Point thế nhưng nó không thể kết nối với mạng dây để truy cập Internet, vậy nên được gọi là **Soft-AP**.

*ESP32 Soft AP*

Soft AP thường được sử dụng khi chúng ta cần lấy thông tin của mạng Wifi khác cho thiết bị IOT khi chúng bắt đầu hoạt động. Ứng dụng thường thấy nhất đó là AP Config wifi, mà chúng ta sẽ học trong các bài sau.

Lập trình ESP32 Webserver chế độ Access Point

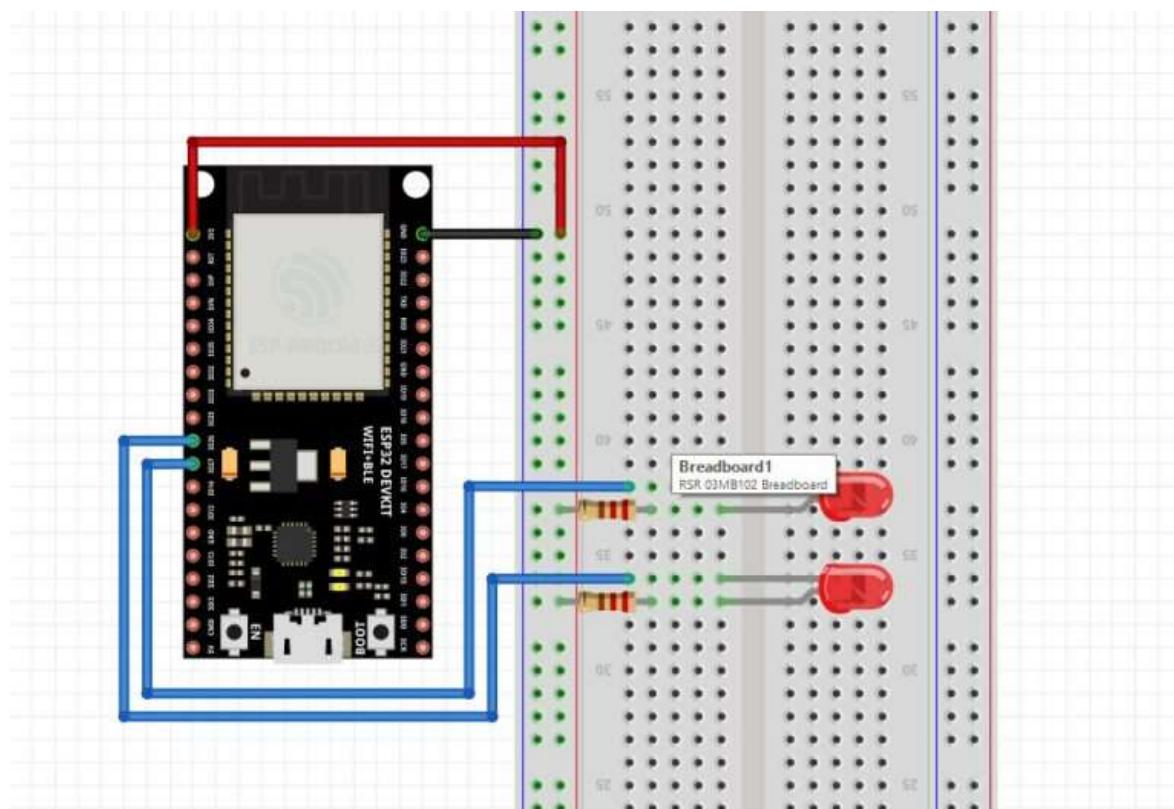
Trong bài này chúng ta sẽ sử dụng ESP32 bật tắt led bằng Web Browser. Nhưng thay vì kết nối ESP32 tới một điểm phát Wifi (Router), chúng ta sẽ cho ESP32 phát Wifi.

Chuẩn bị:

- ESP32 development board
- 2x 5mm LED
- 2x 330 Ohm trở
- Breadboard
- Dây cắm

Sơ đồ nguyên lý

Tương tự như bài **ESP32 Station mode** chúng ta sẽ nối Led với 2 chân GPIO 26 và 27



Esp32 webserver chế do wifi Access Point

Code và giải thích code

Full Code

```

002 #include <Arduino.h>
003
004 #include <WiFi.h>;
005 //khai báo chân sử dụng led
006 const int led1 = 26;
007 const int led2 = 27;
008
009 const char* ssid = "ESP32-Wifi-AP-Mode";
010 const char* password = "12345678";
011 //Tạo một web server tại cổng 80 - cổng mặc định cho web
012 WiFiServer webServer(80);
013
014 String led1Status = "OFF";
015 String led2Status = "OFF";

```

```
016
017 String header;
018
019 unsigned long currentTime = millis();
020 // Previous time
021 unsigned long previousTime = 0;
022 // Define timeout time in milliseconds (example: 2000ms = 2s)
023 const long timeoutTime = 2000;
024
025 void setup() {
026     Serial.begin(115200);
027
028     pinMode(led1, OUTPUT);
029     pinMode(led2, OUTPUT);
030     // Set outputs to LOW
031     digitalWrite(led1, LOW);
032     digitalWrite(led2, LOW);
033
034     Serial.print("Setting AP mode");
035     WiFi.softAP(ssid, password);
036
037     IPAddress IP = WiFi.softAPIP(); //mặc định là 192.168.4.1
038     Serial.print("AP IP address: ");
039     Serial.println(IP);
040     //khởi tạo webserver
041     webServer.begin();
042 }
043
044 void loop() {
045     WiFiClient webClient = webServer.available();
046
047     if(webClient)
048     {
049         //khoi tao gia tri ban dau cho time
050         currentTime = millis();
051         previousTime = currentTime;
052         Serial.println("New web Client");
053         //biến lưu giá trị response
054         String currentLine = "";
055         //nếu có client connect và không quá thời gian time out
056         while(webClient.connected() && currentTime - previousTime < timeoutTime)
057         {
058             //đọc giá trị timer tại thời điểm hiện tại
059             currentTime = millis();
060             //nếu client còn kết nối
061             if(webClient.available())
062             {
063                 //đọc giá trị truyền từ client theo từng byte kiểu char
064                 char c = webClient.read();
065                 Serial.write(c);
```

```

066 header += c; // lưu giá trị vào Header
067 if(c == '\n') //Nếu đọc được kí tự xuống dòng (hết 1 dòng)
068 {
069     if (currentLine.length() == 0)
070     {
071         // HTTP headers luôn luôn bắt đầu với code HTTP
072         webClient.println("HTTP/1.1 200 OK");
073         webClient.println("Content-type:text/html"); //
074         webClient.println("Connection: close"); // kiểu
075         webClient.println();
076
077         // nếu trong file header có giá trị
078         if (header.indexOf("GET /led1/on") >= 0)
079         {
080             Serial.println("Led1 on");
081             led1Status = "on";
082             digitalWrite(led1, HIGH);
083         }
084         else if (header.indexOf("GET /led1/off") >= 0)
085         {
086             Serial.println("Led1 off");
087             led1Status = "off";
088             digitalWrite(led1, LOW);
089         }
090         else if (header.indexOf("GET /led2/on") >= 0)
091         {
092             Serial.println("Led2 on");
093             led2Status = "on";
094             digitalWrite(led2, HIGH);
095         }
096         else if (header.indexOf("GET /led2/off") >= 0)
097         {
098             Serial.println("Led2 off");
099             led2Status = "off";
100             digitalWrite(led2, LOW);
101         }
102         // Response trang HTML
103         webClient.println("<!DOCTYPE html> <html> ");
104         webClient.println(" <head> <meta name=\"viewport\"");
105         //thêm font-awesome
106         webClient.println(" <link rel=\"stylesheet\" href");
107         // code CSS cho web
108         //css cho toàn bộ trang
109         webClient.println(" <img src=\"data:image/gif;base64,");
110
111         // Web Page Heading H1 with CSS
112         webClient.println(" <body> <h1 style=\"color:Tomato; font-size:3em; margin-bottom:10px;\"");
113
114         // Web Page Heading H2
115         webClient.println(" <h2 style=\"color:#077a39; \\"");

```

```

116    webClient.println("<i class=\"fa fa-home\" aria-"
117
118    // Display current state, and ON/OFF buttons for
119    webClient.println("<p>Led1 - State " + led1Stat
120    // If the Led1Status is off, it displays the ON
121    if (led1Status=="off")
122    {
123        //khởi tạo một nút nhấn có đường dẫn đích là ,
124        webClient.println("<p><a href=\"/led1/on\">");  

125    }
126    else
127    {
128        //khởi tạo một nút nhấn có đường dẫn đích là ,
129        webClient.println("<p><a href=\"/led1/off\">");  

130    }
131
132    // Display current state, and ON/OFF buttons for
133    webClient.println("<p>Led2 - State " + led2Stat
134    // If the led2 is off, it displays the ON button
135    if (led2Status=="off")
136    {
137        //khởi tạo một nút nhấn có đường dẫn đích là ,
138        webClient.println("<p><a href=\"/led2/on\">");  

139    }
140    else
141    {
142        //khởi tạo một nút nhấn có đường dẫn đích là ,
143        webClient.println("<p><a href=\"/led2/off\">");  

144    }
145    webClient.println("</body>;</html>");  

146
147    // The HTTP response ends with another blank line
148    webClient.println();
149    // Break out of the while loop
150    break;
151}
152else
153{
154    currentLine = "";
155}
156}
157else if (c != '\r') //nếu giá trị gửi tới khác xung
158{
159    currentLine += c; //lưu giá trị vào biến
160}
161}
162}
163// Xoá header để sử dụng cho lần tới
164header = "";
165// ngắt kết nối

```

```

166     webClient.stop();
167     Serial.println("Client disconnected.");
168     Serial.println("");
169
170 }
171

```

Phần thực thi giữ Web Server và Client sẽ tương tự như bài trước. Thế nên mình chỉ giải thích lại những điểm khác biệt khi khởi tạo ESP32 Access Point.

Đầu tiên chúng ta khởi tạo tên wifi và mật khẩu. Đây là tên wifi và mật khẩu để truy cập vào ESP32, bạn có thể thay đổi thành tên bất kì nhé.

Tiếp tới khởi tạo Web server tại port 80.

```

const char* ssid = "ESP32-Wifi-AP-Mode";
const char* password = "12345678";
//Tạo một web server tại cổng 80 - cổng mặc định cho web
WiFiServer webServer(80);

```

Trong Setup()

Khởi tạo Soft-AP với tên wifi và mật khẩu bạn tạo phía trên. In ra địa chỉ IP mà soft-AP khởi tạo. Mặc định là 192.168.4.1

```

Serial.print("Setting AP mode");
WiFi.softAP(ssid, password);

IPAddress IP = WiFi.softAPIP(); //mặc định là 192.168.4.1

```

Phần xử lý trong web server tương tự bài phía trước. Các bạn có thể chuyển qua để đọc nhé.

Nạp code cho ESP32

Nhấn build để biên dịch và Upload để nạp code

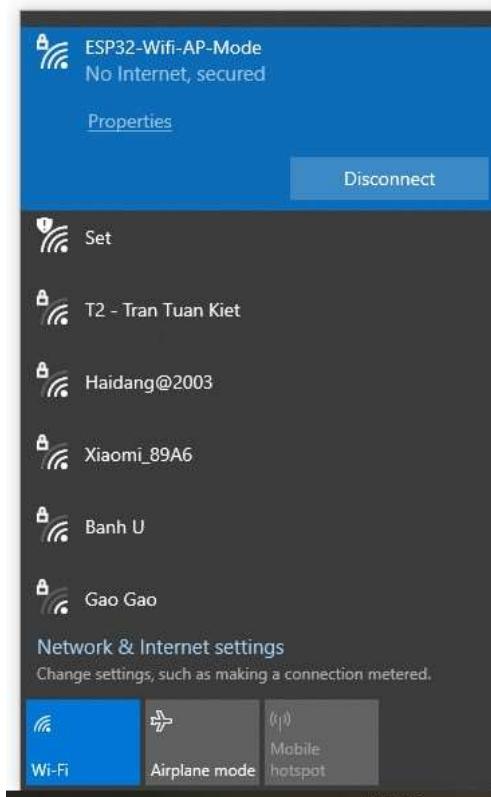
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

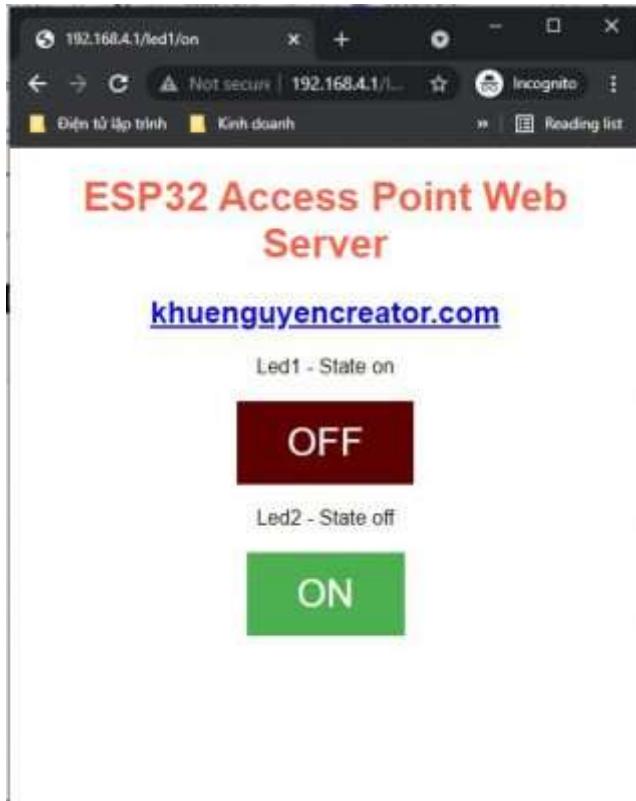
rst:0x1 (POWERON_RESET),boot:0x13 (SPI_FAST_FLASH_BOOT)
configsip: 0, SPIWP:0xee
clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x00,wp_drv:0x00
mode:DIO, clock div:2
load:0x3fff0018,len:1044
load:0x3fff001c,len:1044
load:0x40078000,len:10124
load:0x40080400,len:5828
entry 0x400806a8
Setting AP modeAP IP address: 192.168.4.1

```

Truy cập vào Wifi ESP32 nhập mật khẩu đã tạo bên trên



Vào trình duyệt gõ: 192.169.4.1 sau đó bạn đã có thể điều khiển được led với ESP32 rồi



Kết Quả



Lập trình Esp32 webserver chế độ Access Point
Khuê Nguyễn Creator - Học Lập Trình Vi Điều Khiển

Chia sẻ

Facebook Watch

Các hàm thường gặp trong ESP32 Access Point softAP

Cách thiết lập đơn giản nhất chỉ yêu cầu một tham số và được sử dụng để thiết lập một mạng Wi-Fi mở.

WiFi.softAP (ssid)

Để thiết lập mạng được bảo vệ bằng mật khẩu, hoặc để cấu hình các thông số mạng bổ sung, sử dụng quá tải sau đây:

WiFi.softAP(ssid, password, channel, hidden)

Tham số đầu tiên của hàm này là bắt buộc, còn lại ba tùy chọn.

- ssid: chuỗi ký tự chứa SSID mạng (tối đa 63 ký tự)
- password: chuỗi ký tự tùy chọn với mật khẩu. Đối với mạng WPA2-PSK, nó phải có ít nhất 8 ký tự. Nếu không có mật khẩu, thì đây sẽ là mạng WiFi mở.
- channel: Tham số tùy chọn để thiết lập kênh Wi-Fi, từ 1 đến 13. Kênh mặc định = 1.
- hidden: Tham số tùy chọn, thiết lập là true để ẩn SSID

Trả về true hoặc false phụ thuộc vào kết quả của việc cài đặt soft-AP.

softAPConfig

softAPConfig(local_ip, gateway, subnet)

Tất cả các thông số đều có kiểu IPAddress và được định nghĩa như sau:

- local_ip: Địa chỉ IP của điểm truy cập mềm
- gateway: địa chỉ IP gateway
- subnet: subnet mask

Trả về true hoặc false phụ thuộc vào kết quả của việc thay đổi cấu hình.

Ví dụ:

```
#include <ESP8266WiFi.h>

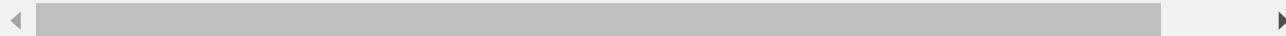
IPAddress local_IP(192,168,4,22);
IPAddress gateway(192,168,4,9);
IPAddress subnet(255,255,255,0);

void setup()
{
    Serial.begin(115200);
    Serial.println();
    Serial.print("Setting soft-AP configuration ... ");
    Serial.println(WiFi.softAPConfig(local_IP, gateway, subnet) ? "Ready" : "Failed!");

    Serial.print("Setting soft-AP ... ");
    Serial.println(WiFi.softAP("ESPsoftAP_01") ? "Ready" : "Failed!");

    Serial.print("Soft-AP IP address = ");
    Serial.println(WiFi.softAPIP());
}

void loop() {}
```



output

```
Setting soft-AP configuration ... Ready
Setting soft-AP ... Ready
Soft-AP IP address = 192.168.4.22
```

Quản lý kết nối

Khi đã thiết lập softAP, bạn có thể kiểm tra các trạm đã kết nối, hoặc tắt chúng, sử dụng các hàm sau:

softAPgetStationNum

Lấy số lượng các station kết nối đến softAP

```
WiFi.softAPgetStationNum()
```

```
Serial.printf("Stations connected to soft-AP = %d\n", WiFi.softAPgetStationNum())
```

Ví dụ:

Trả về số lượng các thiết bị (station) kết nối tới mạng Wifi thiết lập bởi ESP8266

Ví dụ:

```
#include <ESP8266WiFi.h>

void setup()
{
    WiFi.softAP("31/8/2017");
    Serial.begin(115200);

}

void loop()
{
    Serial.printf("Stations connected to soft-AP = %d \n", WiFi.softAPgetStationNum(
        delay(2000); //delay trong 2s để kiểm tra xem có thiết bị nào mới kết nối với mo
    })
}
```

softAPdisconnect

Ngắt kết nối các trạm từ mạng được thiết lập bởi softAP.

```
WiFi.softAPdisconnect(wifioff)
```

Chức năng sẽ thiết lập cấu hình SSID và password của soft-AP giá trị là null. Tham số wifioff là tùy chọn. Nếu thiết lập là true nó sẽ tắt chế độ soft-AP.

Trả về true nếu hoạt động đã thành công, false nếu không.

Cấu hình Mạng

Các hàm dưới đây cung cấp địa chỉ IP và MAC của soft-AP của ESP8266.

softAPIP

Trả lại địa chỉ IP của mạng softAP.

```
WiFi.softAPIP()
```

Trả về giá trị có kiểu là IPAddress.

```
Serial.print("Soft-AP IP address = ");
Serial.println(WiFi.softAPIP());
```

output

```
Soft-AP IP address = 192.168.4.1
```

softAPmacAddress

Trả lại địa chỉ MAC của softAP. Chức năng này có hai phiên bản khác nhau về kiểu trả về. Trả về một con trỏ hoặc một String.

Với kiểu trả về là Con trỏ

```
WiFi.softAPmacAddress(mac)
```

Tham số mac là một con trỏ trỏ đến vị trí bộ nhớ (một mảng uint8_t có 6 phần tử) để lưu địa chỉ mac. Cùng một giá trị con trỏ được trả về bởi chính hàm đó.

```
uint8_t macAddr[6];
WiFi.softAPmacAddress(macAddr);
Serial.printf("MAC address = %02x:%02x:%02x:%02x:%02x:%02x\n", macAddr[0], macAd
```

output

```
MAC address = 5e:cf:7f:8b:10:13
```

MAC như một String

```
WiFi.softAPmacAddress()
```

Kiểu trả về là một String chứa địa chỉ MAC của softAP.

```
Serial.printf("MAC address = %s\n", WiFi.softAPmacAddress().c_str());
```

output

```
MAC address = 5E:CF:7F:8B:10:13
```

Kết

Access Point trong ESP32 cũng rất đơn giản, và dễ nắm bắt. Kết hợp giữa Access Point và Station mode sẽ giúp chúng tạo ra các sản phẩm IOT một cách linh hoạt hơn.

Nếu bạn thấy bài viết này có ích hãy để lại bình luận và đừng quên ra nhập **Hội Anh Em Nghiên Lập trình** nhé.

5/5 - (2 bình chọn)

Related Posts:

1. [Bài 1: Lập trình ESP32 Webserver chế độ Wifi Station bật tắt Led](#)
2. [Bài 6: Lập trình ESP32 Timer Millis và ngắt Timer](#)
3. [Bài 5: Lập trình ESP32 ngắt ngoài EXTI](#)
4. [Bài 2: Lập trình ESP32 Analog Input đọc tín hiệu tương tự \(ADC\)](#)
5. [Tổng quan về sơ đồ chân ESP32 và ngoại vi](#)
6. [Lập trình ESP32 từ A tới Z](#)



KHUÊ NGUYỄN

Chỉ là người đam mê điện tử và lập trình. Làm được gì thì viết cho anh em xem thôi. :D

Trả lời

Email của bạn sẽ không được hiển thị công khai. Các trường bắt buộc được đánh dấu *

Bình luận *

Tên *

Email *

Trang web

PHẢN HỒI

Fanpage

Khuê Nguyễn Creator - Họ...
2.754 lượt thích

Đã thích **Chia sẻ**

Khuê Nguyễn Creator - Học Lập Trình Vi Điều Khiển
khoảng một tháng trước

Lý do thời gian gần đây mình không viết bài và làm thêm gì cả là đây 😊)
Chính thức ra mắt sản phẩm định vị thông minh vTag.

Đây là một sản phẩm định vị đa năng với 3 công nghệ định vị WIFI, GPS, LBS kết hợp với sóng NB-IOT dành riêng cho các sản phẩm IOT.

Chỉ với 990.000đ chúng ta đã có thể có sản phẩm để:

- Định vị trẻ em, con cái... [Xem thêm](#)

Bài viết khác

Lập trình 8051 - AT89S52

Khuê Nguyễn Creator

PROTEUS

Bài 1: Tổng quan về 8051 và chip AT89S51 - 52

Tổng quan về 8051

8051 là một dòng chip nhập môn cho lập trình viên nhúng, chúng được sử...

[ĐỌC THÊM](#)

Lập trình STM32 và CubeMX



Khuê Nguyễn Creator



Lập trình STM32 HID Host giao tiếp với chuột và bàn phím

Lập trình STM32 USB HID Host giao tiếp với chuột và bàn phím máy tính

Trong bài này chúng ta sẽ cùng học STM32 HID Host, biến STM32 giống như...

[ĐỌC THÊM](#)



Khuê Nguyễn Creator



Lộ trình học lập trình nhúng từ A tới Z

Lộ trình học lập trình nhúng từ A tới Z

Lập trình nhúng là một ngành có cơ hội nhưng cũng đòi hỏi nhiều kiến...

3 COMMENTS

[ĐỌC THÊM](#)

Lập trình STM32 và CubeMX



Khuê Nguyễn Creator



Lập trình STM32F407 SDIO đọc dữ liệu thẻ nhớ

Lập trình STM32 SDIO đọc ghi dữ liệu vào thẻ nhớ SD card

Trong bài này chúng ta cùng học cách lập trình STM32 SDIO, một chuẩn giao...

[ĐỌC THÊM](#)

Lập trình STM32 và CubeMX



Khuê Nguyễn Creator



Lập trình STM32F407 DAC chuyển đổi số sang tương tự

Lập trình STM32 DAC tạo sóng hình Sin trên KIT STM32F407 Discovery

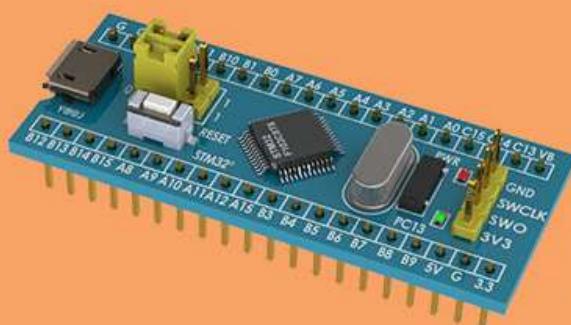
Trong bài này chúng ta sẽ cùng nhau tìm hiểu STM32 DAC với KIT STM32F407VE...

[ĐỌC THÊM](#)

Lập trình STM32 và CubeMX



Khuê Nguyễn Creator



Sử dụng hàm printf để in Log khi Debug trên STM32

Hướng dẫn sử dụng printf với STM32 Uart để in Log trên Keil C

Trong bài này chúng ta sẽ học cách retarget hàm printf của thư viện stdio...

3 COMMENTS

[ĐỌC THÊM](#)

ESP32 và Platform IO



Khuê Nguyễn Creator



Bài 9 WIFI: Lập trình ESP32 OTA nạp firmware trên Internet

Lập trình ESP32 FOTA nạp firmware qua mạng Internet với OTA Drive

Trong bài này chúng ta sẽ học cách sử dụng ESP32 FOTA (Firmware Over The...

4 COMMENTS

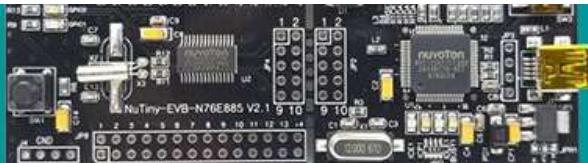
[ĐỌC THÊM](#)

Lập trình Nuvoton



Khuê Nguyễn Creator





Cài đặt SDC Complier và Code:Blocks IDE

Hướng dẫn cài đặt SDCC và Code:Blocks lập trình Nuvoton

Ở bài này chúng ta sẽ cài đặt các công cụ cần thiết cho việc...

ĐỌC THÊM



Blog này làm ra để lưu trữ tất cả những kiến thức, những câu chuyện của mình. Đôi khi là những ý tưởng nhất thời, đôi khi là các dự án tự mình làm. Chia sẻ cho người khác cũng là niềm vui của mình, kiến thức mỗi người là khác nhau, không hẳn quá cao siêu nhưng sẽ có lúc hữu dụng.

Liên Kết

Nhóm: Nghiên Lập Trình

Fanpage: Khuê Nguyên Creator

My Shop

Thông Tin

Tác Giả

Chính Sách Bảo Mật



Copyright 2022 © Khuê Nguyễn