

**LẬP TRÌNH ESP32**

Lập trình ESP32 MQTT bật tắt đèn với Hive Broker

POSTED ON 30/10/2021 BY **KHUÊ NGUYỄN**30
Th10

ESP32 và Platform IO

**Khuê Nguyễn Creator**

Bài 6 WIFI: Lập trình ESP32 giao thức MQTT với Hive Broker

Trong bài này chúng ta sẽ học cách sử dụng ESP32 MQTT, giao tiếp với Hive Broker. Đây là một dịch vụ broker miễn phí, các bạn có thể sử dụng để test các sản phẩm IOT của mình.

Bài 6 Trong serie **Học Esp32 từ A tới Z**

Mục Lục



1. HiveMQ broker là gì?
 - 1.1. Cách sử dụng Public broker
2. Lập trình ESP32 MQTT
 - 2.1. Cài đặt thư viện pubsubclient
 - 2.2. Kết nối với broker
 - 2.3. Kết nối phần cứng
 - 2.4. Publish dữ liệu lên MQTT
 - 2.5. Subscrible dữ liệu từ MQTT
3. Full code
4. Kết quả
5. Kết
 - 5.1. Related posts:

HiveMQ broker là gì?

HiveMQ Broker là một nền tảng truyền nhận dữ liệu dựa trên giao thức MQTT, được thiết kế với đặc tính nhanh, hiệu quả, độ tin cậy cao khi truyền dữ liệu 2 chiều giữa các thiết bị Internet of Things.

Đây là một dịch vụ miễn phí trên nền tảng Cloud, giúp các bạn có thể kết nối MQTT ở bất cứ đâu, không kể địa lý, phù hợp khi học **lập trình IOT**

HIVEMQ

Reliable Data Movement for Connected Devices

HiveMQ's MQTT broker makes it easy to move data to and from connected devices in an efficient, fast and reliable manner. We make it possible to build connected products that enable new digital businesses.

Learn more | Get HiveMQ | Contact Us

HiveMQ 4.7 now available!
Packed with new features and improvements to empower your business!

Get started now!

Develop Efficient IoT Solutions



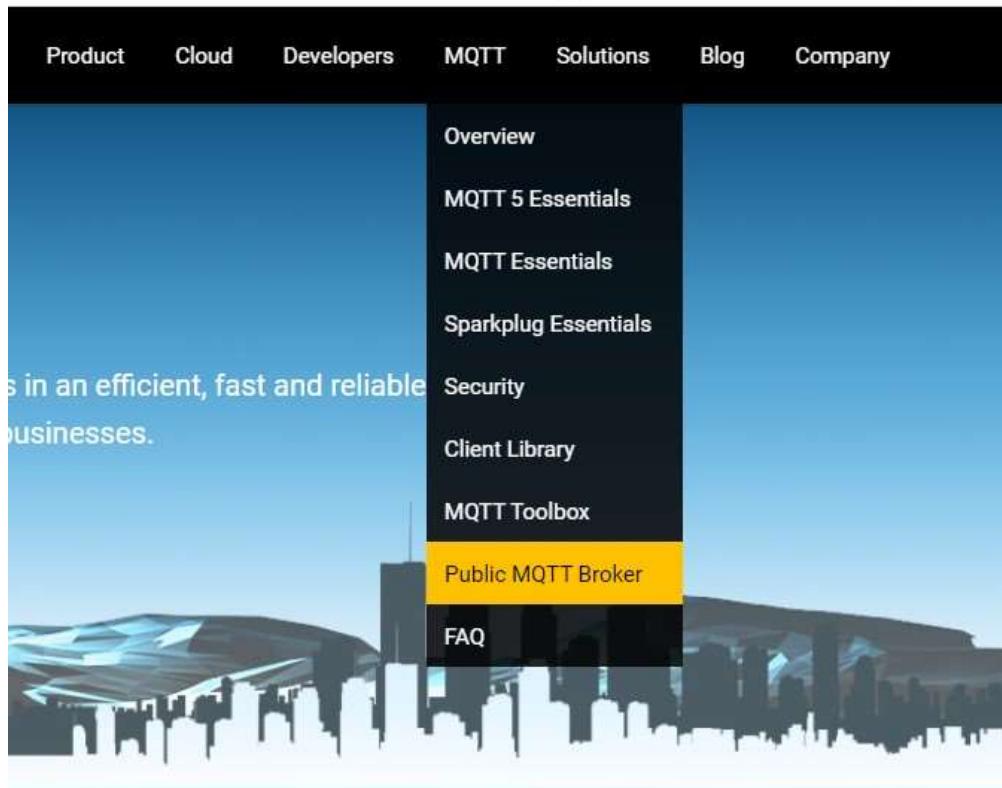
Hivemq Broker chia thành 2 loại:

- Public broker: Sử dụng cổng 1883, không có bảo mật, thường dùng để test ứng dụng hoặc các sản phẩm đơn giản
- Private broker: Sử dụng cổng 8883 và bảo mật SSL/TLS. Bạn có thể sử dụng nó trong các sản phẩm thương mại, nhưng tất nhiên nên để ý các điều khoản khi sử dụng nhé

Link ứng dụng: <https://www.hivemq.com/>

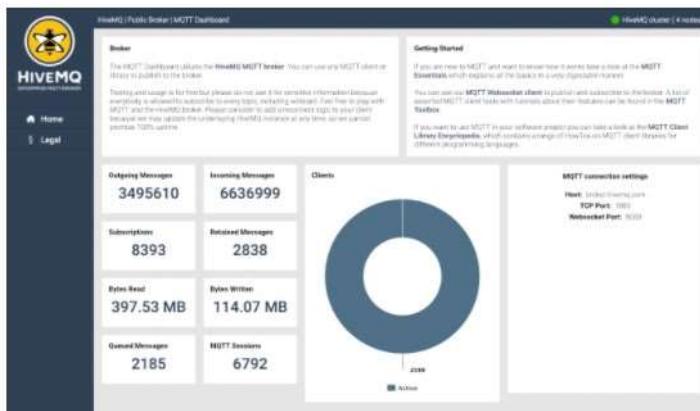
Cách sử dụng Public broker

Trong menu, nhấn MQTT – Public MQTT.



Giao diện bao gồm:

- Thông tin MQTT chúng ta kết nối vào
- Websocket để có thể điều khiển MQTT bằng web browser



Our **Public HiveMQ MQTT broker** is open for anyone to use. Feel free to write an MQTT client that connects with this broker. We have a **dashboard** so you can see the amount of traffic on this broker. We also keep a list of **MQTT client libraries** that can be used to connect to HiveMQ.

You can access the broker at:

Broker: broker.hivemq.com

TCP Port: 1883

Websocket Port: 8000

Lập trình ESP32 MQTT

Cài đặt thư viện pubsubclient

Mở terminal gõ lệnh `pio lib install "knolleary/PubSubClient"`

Sau đó update đường dẫn thư viện trong platformio.ini

```

10
11 [env:esp32doit-devkit-v1]
12 platform = espressif32
13 board = esp32doit-devkit-v1
14 framework = arduino
15 lib_deps =
16 | knolleary/PubSubClient@^2.8.0
17 |
18 |

```

Cài đặt các thông số của MQTT như sau:

- Server broker và port lấy trên hivemq
- Vì là kiểu public nên các bạn không cần user và password cũng được
- Tạo ra 2 topic lưu giá trị của Led1 và Led 2

```

1  /*
2  #include <WiFi.h>
3  #include <PubSubClient.h>
4  #define BUT1 4
5  #define BUT2 16
6  #define LED1 17
7  #define LED2 5
8  String ledStatus1 = "ON";
9  String ledStatus2 = "ON";
10 const char* ssid = "Nh(const char [9])"25251325"
11 const char* password = "25251325";
12
13 #define MQTT_SERVER "broker.hivemq.com"
14 #define MQTT_PORT 1883
15 #define MQTT_USER "mabattu123"
16 #define MQTT_PASSWORD "12345678"
17
18 #define MQTT_LED1_TOPIC "MQTT_ESP32/LED1"
19 #define MQTT_LED2_TOPIC "MQTT_ESP32/LED2"
20

```

Kết nối với broker

Lưu ý: Mỗi một thiết bị phải có 1 client ID riêng, nếu 2 thiết bị cùng client ID kết nối vào Broker. Nó sẽ đẩy thiết bị cũ ra hoặc không thể connect được. Trong bài này, mình sẽ cho Client ID 1 dãy số random khi kết nối vào.

Sau khi kết nối, chúng ta sẽ sub với 2 topic LED1 và LED2 để nhận dữ liệu từ Broker về.

```
void connect_to_broker() {
    while (!client.connected()) {
        Serial.print("Attempting MQTT connection...");
        String clientId = "ESP32";
        clientId += String(random(0xffff), HEX);
        if (client.connect(clientId.c_str())) {
            Serial.println("connected");
            client.subscribe(MQTT_LED1_TOPIC);
            client.subscribe(MQTT_LED2_TOPIC);
        } else {
            Serial.print("failed, rc=");
            Serial.print(client.state());
            Serial.println(" try again in 2 seconds");
            delay(2000);
        }
    }
}
```

Kết nối phần cứng

Các bạn kết nối như sau

- Button1 — chân 4
- Button1 — chân 16
- Led 1 — chân 17
- Led 2 — chân 5

Publish dữ liệu lên MQTT

Mỗi khi nhấn nút, chúng ta sẽ kiểm tra trạng thái của ledstatus, sau đó đảo ngược trạng thái đó.

Tiếp tới dùng `client.publish` để đẩy dữ liệu đó lên broker.

```

if(digitalRead(BUT1) == 0)
{
    while (digitalRead(BUT1) == 0)
    {
        /* chờ nút dc nha */
    }
    if(ledStatus1 == "ON")
    {
        client.publish(MOTT_LED1_TOPIC, "OFF");
        ledStatus1 = "0" #define MQTT_LED1_TOPIC "MQTT_ESP32/LED1"
    }
    else if(ledStatus1 == "OFF")
    {
        client.publish(MQTT_LED1_TOPIC, "ON");
        ledStatus1 = "ON";
    }
}
if(digitalRead(BUT2) == 0)
{
    while (digitalRead(BUT2) == 0)
    {
        /* chờ nút dc nha */
    }
    if(ledStatus2 == "ON")
    {
        client.publish(MQTT_LED2_TOPIC, "OFF");
        ledStatus2 = "OFF";
    }
    else if(ledStatus2 == "OFF")
    {
        client.publish(MQTT_LED2_TOPIC, "ON");
        ledStatus2 = "ON";
    }
}
}

```

#define MQTT_LED1_TOPIC "MQTT_ESP32/LED1"
Expands to:
"MQTT_ESP32/LED1"

Subcrible dữ liệu từ MQTT

Mỗi khi có dữ liệu thay đổi trên broker, hàm callback sẽ được gọi. Ở đây chúng ta sử lý như sau:

- Copy dữ liệu trả về (payload) và status
- Kiểm tra đó là topic nào
- Ghi trạng thái led với dữ liệu trả về tương ứng với mỗi topic

```

void callback(char* topic, byte *payload, unsigned int length) {
    char status[20];
    Serial.println("-----new message from broker-----");
    Serial.print("topic: ");
    Serial.println(topic);
    Serial.print("message: ");
    Serial.write(payload, length);
    Serial.println();
    for(int i = 0; i<length; i++)
    {
        status[i] = payload[i];
    }
    Serial.println(status);
    if(String(topic) == MQTT_LED1_TOPIC)
    {
        if(String(status) == "OFF")
        {
            ledStatus1 = "OFF";
            digitalWrite(LED1, LOW);
            Serial.println("LED1 OFF");
        }
        else if(String(status) == "ON")
        {
            ledStatus1 = "ON";
            digitalWrite(LED1, HIGH);
            Serial.println("LED1 ON");
        }
    }
}

```

Full code

Full Code

```

002 #include <WiFi.h>
003 #include <PubSubClient.h>
004 #define BUT1 4
005 #define BUT2 16
006 #define LED1 17
007 #define LED2 5
008 String ledStatus1 = "ON";
009 String ledStatus2 = "ON";
010
011 const char* ssid = "Nha Tao";
012 const char* password = "25251325";
013
014 #define MQTT_SERVER "broker.hivemq.com"
015 #define MQTT_PORT 1883
016 #define MQTT_USER "mabattu123"
017 #define MQTT_PASSWORD "12345678"

```

```
018  
019 #define MQTT_LED1_TOPIC "MQTT_ESP32/LED1"  
020 #define MQTT_LED2_TOPIC "MQTT_ESP32/LED2"  
021  
022     unsigned long previousMillis = 0;  
023     const long interval = 5000;  
024  
025 WiFiClient wifiClient;  
026 PubSubClient client(wifiClient);  
027  
028  
029 void setup_wifi() {  
030     Serial.print("Connecting to ");  
031     Serial.println(ssid);  
032     WiFi.begin(ssid, password);  
033     while (WiFi.status() != WL_CONNECTED) {  
034         delay(500);  
035         Serial.print(".");  
036     }  
037     Serial.println("");  
038     Serial.println("WiFi connected");  
039     Serial.println("IP address: ");  
040     Serial.println(WiFi.localIP());  
041 }  
042  
043 void connect_to_broker() {  
044     while (!client.connected()) {  
045         Serial.print("Attempting MQTT connection...");  
046         String clientId = "ESP32";  
047         clientId += String(random(0xffff), HEX);  
048         if (client.connect(clientId.c_str())) {  
049             Serial.println("connected");  
050             client.subscribe(MQTT_LED1_TOPIC);  
051             client.subscribe(MQTT_LED2_TOPIC);  
052         } else {  
053             Serial.print("failed, rc=");  
054             Serial.print(client.state());  
055             Serial.println(" try again in 2 seconds");  
056             delay(2000);  
057         }  
058     }  
059 }  
060  
061 void callback(char* topic, byte *payload, unsigned int length)  
062     char status[20];  
063     Serial.println("-----new message from broker-----");  
064     Serial.print("topic: ");  
065     Serial.println(topic);  
066     Serial.print("message: ");  
067     Serial.write(payload, length);
```

```

068 Serial.println();
069 for(int i = 0; i<length; i++)
070 {
071     status[i] = payload[i];
072 }
073 Serial.println(status);
074 if(String(topic) == MQTT_LED1_TOPIC)
075 {
076     if(String(status) == "OFF")
077     {
078         ledStatus1 = "OFF";
079         digitalWrite(LED1, LOW);
080         Serial.println("LED1 OFF");
081     }
082     else if(String(status) == "ON")
083     {
084         ledStatus1 = "ON";
085         digitalWrite(LED1, HIGH);
086         Serial.println("LED1 ON");
087     }
088 }
089
090 if(String(topic) == MQTT_LED2_TOPIC)
091 {
092     if(String(status) == "OFF")
093     {
094         ledStatus2 = "OFF";
095         digitalWrite(LED2, LOW);
096         Serial.println("LED2 OFF");
097     }
098     else if(String(status) == "ON")
099     {
100         ledStatus2 = "ON";
101         digitalWrite(LED2, HIGH);
102         Serial.println("LED2 ON");
103     }
104 }
105
106 }
107
108 void setup() {
109     Serial.begin(9600);
110     Serial.setTimeout(500);
111     setup_wifi();
112     client.setServer(MQTT_SERVER, MQTT_PORT );
113     client.setCallback(callback);
114     connect_to_broker();
115     Serial.println("Start transfer");
116     pinMode(BUT1, INPUT_PULLUP);
117     pinMode(BUT2, INPUT_PULLUP);

```

```
118     pinMode(LED1, OUTPUT);
119     pinMode(LED2, OUTPUT);
120 }
121
122
123 void loop() {
124     client.loop();
125     if (!client.connected()) {
126         connect_to_broker();
127     }
128
129     if(digitalRead(BUT1) == 0)
130     {
131         while (digitalRead(BUT1) == 0)
132         {
133             /* chờ nút dc nha */
134         }
135         if(ledStatus1 == "ON")
136         {
137             client.publish(MQTT_LED1_TOPIC, "OFF");
138             ledStatus1 = "OFF";
139         }
140         else if(ledStatus1 == "OFF")
141         {
142             client.publish(MQTT_LED1_TOPIC, "ON");
143             ledStatus1 = "ON";
144         }
145     }
146     if(digitalRead(BUT2) == 0)
147     {
148         while (digitalRead(BUT2) == 0)
149         {
150             /* chờ nút dc nha */
151         }
152         if(ledStatus2 == "ON")
153         {
154             client.publish(MQTT_LED2_TOPIC, "OFF");
155             ledStatus2 = "OFF";
156         }
157         else if(ledStatus2 == "OFF")
158         {
159             client.publish(MQTT_LED2_TOPIC, "ON");
160             ledStatus2 = "ON";
161         }
162     }
163 }
```



Kết quả



Eps32 mqtt bật tắt led với hivemq broker

Khuê Nguyễn Creator - Học Lập Trình Vi Điều Khiển

Chia sẻ

Facebook Watch

Kết

Sử dụng **ESP32** MQTT sẽ giúp việc điều khiển thiết bị một cách Realtime hơn, khi có sự thay đổi dữ liệu trên broker, gần như các thiết bị sub vào broker đó gần như nhận được dữ liệu ngay tức khắc mà không cần phải request lên server như giao thức **HTTP**.

Cám ơn bạn đã đón đọc, cùng vào hội **Anh Em Nghiện Lập Trình** để cùng trao đổi nhé

5/5 - (1 bình chọn)

Related Posts:

1. [Hiển thị nhiệt độ, độ ẩm lên Thingspeak với ESP32](#)
2. [Lập trình ESP32 Websocket điều khiển đèn Real time](#)
3. [Bài 1: Lập trình ESP32 Webserver chế độ Wifi Station bật tắt Led](#)
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

4 THOUGHTS ON “LẬP TRÌNH ESP32 MQTT BẬT TẮT ĐÈN VỚI HIVEMQ BROKER”



Hào says:

cho em hỏi là sao khi em bật(tắt) bằng máy tính xong qua tắt(bật) bằng tay thì nó lại không tắt(bật) luôn mà phải ấn hai lần mới được q. Có thể cho em xin cách khắc phục không q

09/04/2022 AT 2:20 SÁNG

TRẢ LỜI



Khuê Nguyễn says:

Em xem đoạn đồng bộ trạng thái đã code đúng chưa

14/04/2022 AT 12:19 SÁNG

TRẢ LỜI



Trung says:

Anh cho em hỏi trường #define MQTT_LED1_TOPIC lấy ở đâu vậy q, em đổi tên đi thì chương trình không chạy nữa q ? Anh có thể giải thích giúp em được không q ?

11/06/2022 AT 12:07 SÁNG

TRẢ LỜI

Khuê Nguyễn says:

Cái này là tên topic thôi, e tự định nghĩa nhé. Đọc kĩ lại chuẩn MQTT

28/07/2022 AT 11:33 CHIỀU

TRẢ LỜI

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 - Học...
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



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



Khuê Nguyễn Creator



Về Chip ATmega32

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

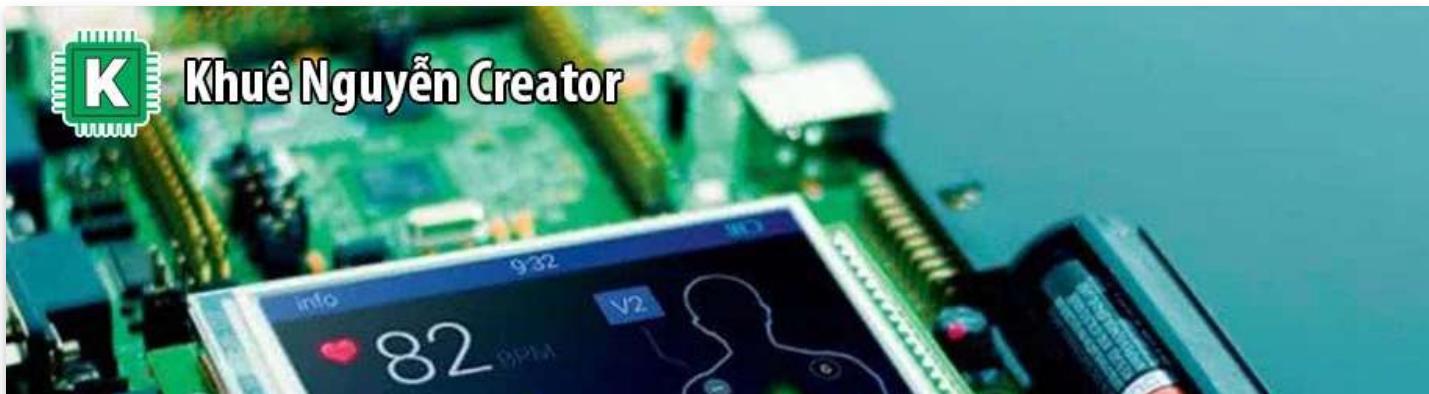


Lập trình STM32 HID Host giao tiếp với chuột và bàn phím

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](#)





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

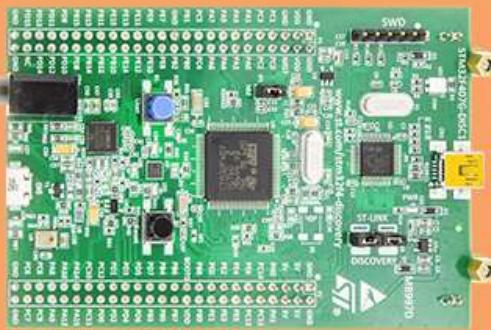
Trong bài này chúng ta cùng học cách lập trình STM32 SDIO, một chuẩn giao...

[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

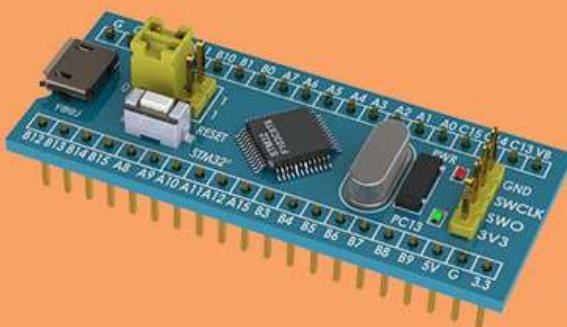
Trong bài này chúng ta sẽ cùng nhau tìm hiểu STM32 DAC với KIT STM32F407VE...

[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...

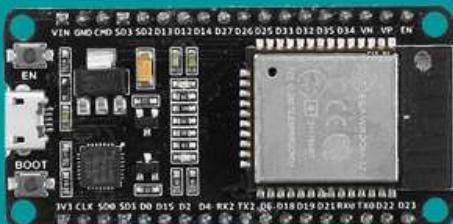
3 COMMENTS

[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...

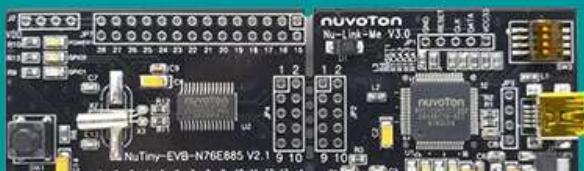
4 COMMENTS

[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