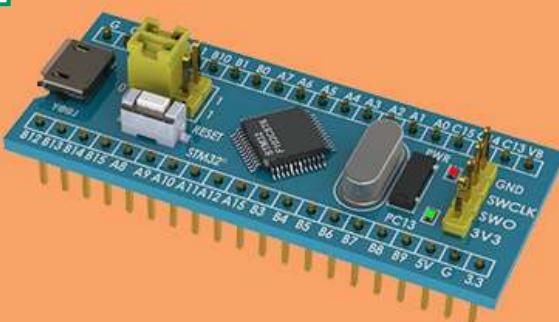
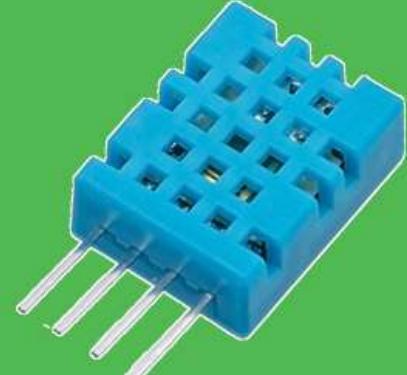


**LẬP TRÌNH STM32**

# Lập trình STM32 với DHT11 theo chuẩn 1 Wire

POSTED ON 30/07/2021 BY KHUÊ NGUYỄN

30  
Th7**Khuê Nguyễn Creator**

## Lập trình STM32 đọc nhiệt độ độ ẩm với DHT11

Cảm biến nhiệt độ DHT11 rất hay được sử dụng trong các dự án nhỏ hoặc đồ án sinh viên, ưu điểm của nó là nhỏ gọn và rẻ. Nhưng độ chính xác thì không cao mấy. Trong bài này, chúng ta sẽ học cách đọc giá trị nhiệt độ, độ ẩm của DHT11 theo chuẩn Onewire.

Bài này nằm trong Serie **Học STM32 từ A tới Z**

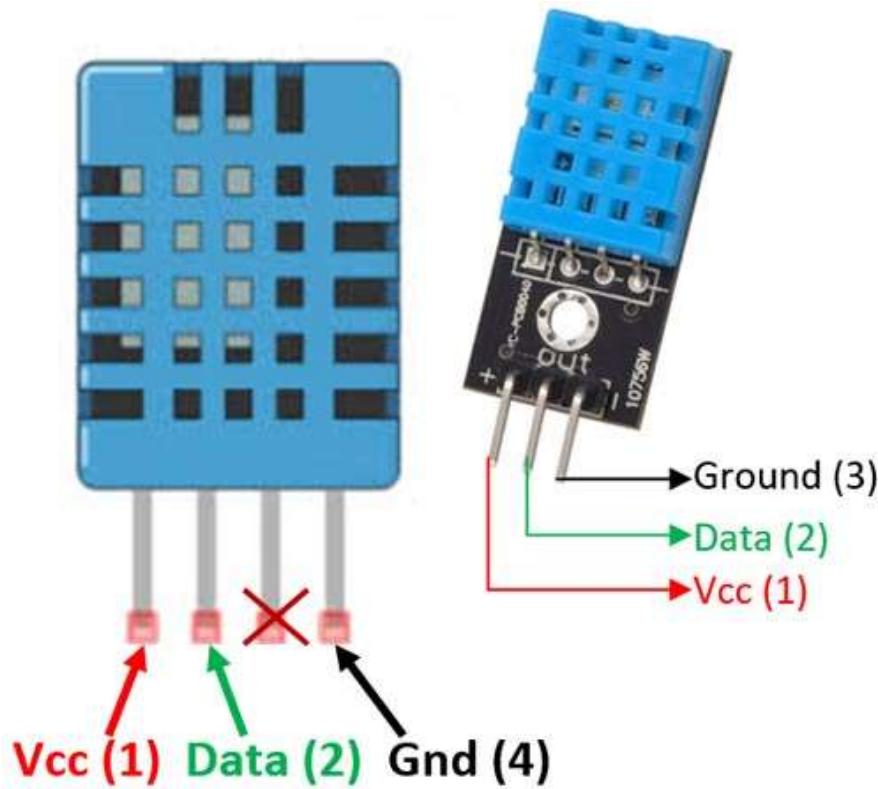


## Mục Lục

1. Tổng quan về cảm biến nhiệt độ, độ ẩm DHT11
2. Giao tiếp One Wire (1 Wire) là gì ?
  - 2.1. Mô hình kết nối và phương thức hoạt động của chuẩn 1 wire
  - 2.2. Phương thức hoạt động của 1 Wire
3. Phương thức giao tiếp của DHT11
  - 3.1. Khung truyền – gói tin
  - 3.2. Chu trình nhận dữ liệu
  - 3.3. Cách reset hay start
  - 3.4. Cách nhận biết bit 0 và 1 trong giá trị trả về của DHT11
4. Lập trình STM32 giao tiếp 1 wire với DHT11
  - 4.1. Khởi tạo trong CubeMX
  - 4.2. Thêm thư viện và lập trình
  - 4.3. Giải thích code trong thư viện DHT11
5. Kết
  - 5.1. Related posts:

## Tổng quan về cảm biến nhiệt độ, độ ẩm DHT11

Cảm biến độ ẩm và nhiệt độ DHT11 Temperature Humidity Sensor là cảm biến rất thông dụng hiện nay vì chi phí rẻ và rất dễ lấy dữ liệu thông qua giao tiếp One wire (giao tiếp digital 1 dây truyền dữ liệu duy nhất). Bộ tiền xử lý tín hiệu tích hợp trong cảm biến giúp bạn có được dữ liệu chính xác mà không phải qua bất kỳ tính toán nào. So với cảm biến đời mới hơn là DHT22 thì DHT11 cho khoảng đo và độ chính xác kém hơn rất nhiều.



Thông tin kỹ thuật:

- Nguồn: 3 -> 5 VDC.
- Dòng sử dụng: 2.5mA max (khi truyền dữ liệu).
- Đo tốt ở độ ẩm 20 to 70%RH với sai số 5%.
- Đo tốt ở nhiệt độ 0 to 50°C sai số  $\pm 2^\circ\text{C}$ .
- Tần số lấy mẫu tối đa 1Hz (1 giây 1 lần)
- Kích thước 15mm x 12mm x 5.5mm.
- 4 chân, khoảng cách chân 0.1”.

DHT11 sử dụng giao thức 1 Wire để giao tiếp với **vi điều khiển**, vậy trước hết hãy tìm hiểu 1 chút về chuẩn này nhé.

## Giao tiếp One Wire (1 Wire) là gì ?

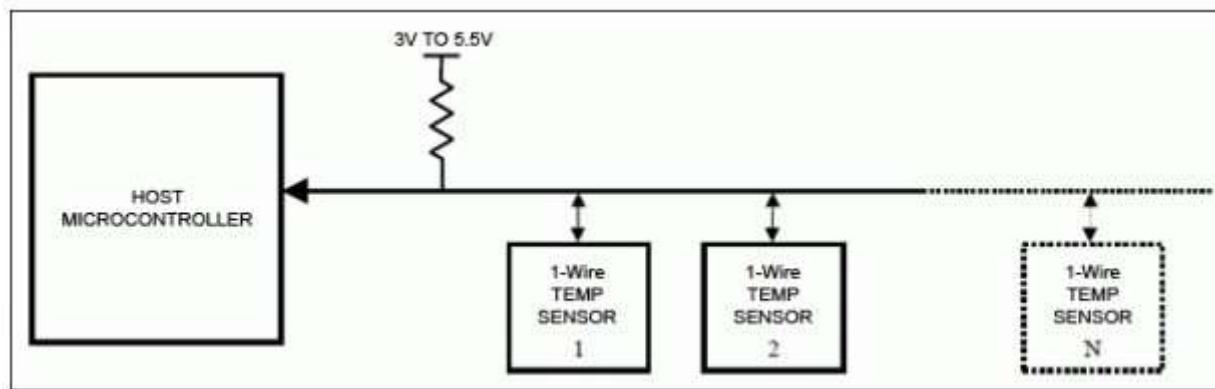
**1-Wire** là một hệ thống bus giao tiếp với thiết bị được thiết kế bởi Dallas Semiconductor Corp. 1-Wire hỗ trợ truyền dữ liệu tốc độ thấp (16.3 kbit/s), truyền tín hiệu, và nguồn nuôi qua cùng một chân tín hiệu đơn. 1-Wire cũng

tương tự như I<sup>2</sup>C, nhưng với tốc độ truyền dữ liệu thấp và khoảng cách xa hơn. Nó thường được sử dụng để giao tiếp với các thiết bị nhỏ giá rẻ như nhiệt kế kĩ thuật số và công cụ đo thời tiết. Một mạng lưới của các thiết bị 1-Wire với một thiết bị điều khiển chính được gọi là một *MicroLAN*.

Theo [Wikipedia](#)

## Mô hình kết nối và phương thức hoạt động của chuẩn 1 wire

Cũng giống như [giao tiếp I<sup>2</sup>C](#), dây tín hiệu của 1 Wire phải được treo lên Vcc. Và cấu hình GPIO dạng Open Drain. Về nguyên lý, chuẩn Onewire có thể giao tiếp với nhiều thiết bị trong cùng một mạng theo sơ đồ sau:



## Phương thức hoạt động của 1 Wire

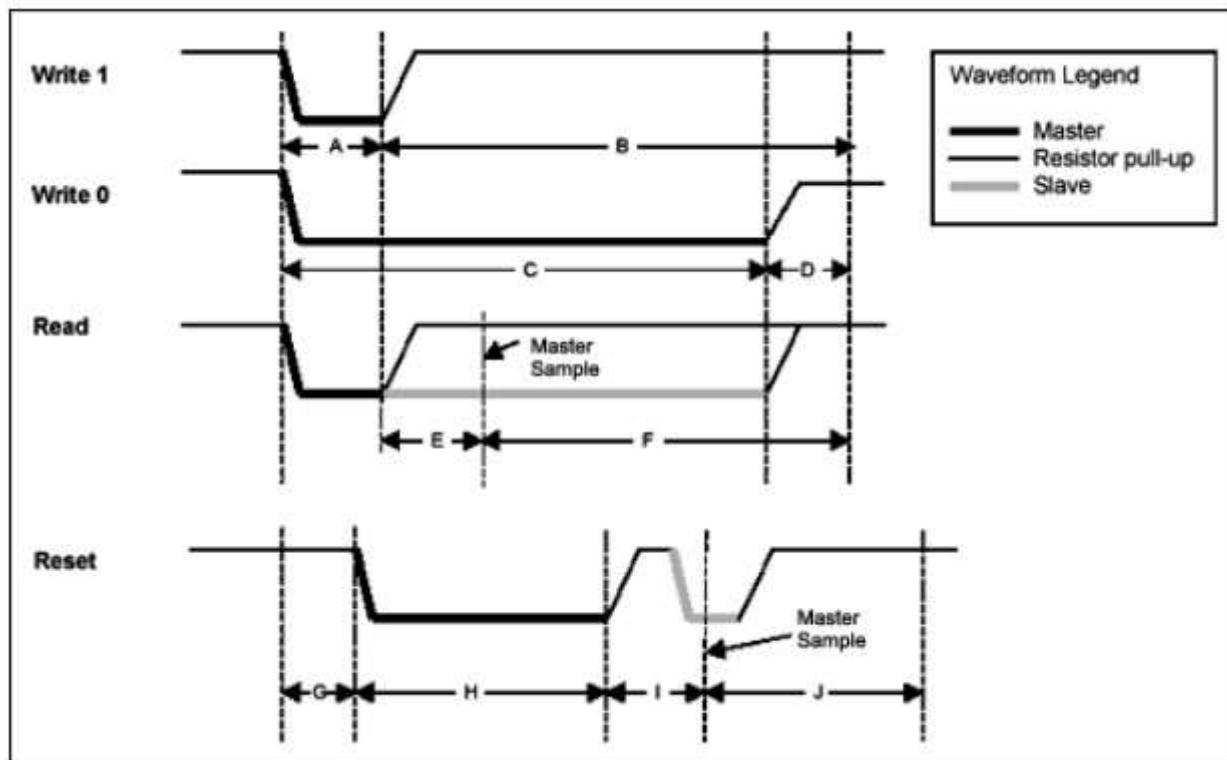
Để 1 Wire hoạt động, sẽ có 4 phương thức được sử dụng đó là: Reset, Write bit 0, Write bit 1, Read.

**Reset:** Chuẩn bị giao tiếp . Master cấu hình chân data là Ouput, kéo xuống 0 một khoảng H rồi nhả ra để trở treo kéo lên mức 1. Sau đó cấu hình Master là chân Input, delay I (us) rồi đọc giá trị slave trả về . Nếu = 0 thì cho phép giao tiếp .Nếu data = 1 đường truyền lỗi hoặc slave đang bận. Tương tự như bạn đọc bit ACK trong giao tiếp I<sup>2</sup>C

**Write 1 :** truyền đi bit 1 : Master kéo xuống 0 một khoảng A(us) rồi về mức 1 khoảng B (us)

**Write 0** : truyền đi bit 0 : Master kéo xuống 0 khoảng C rồi trả về 1 khoảng D (us)

**Read** : Đọc một Bit : Master kéo xuống 0 khoảng A rồi trả về 1. delay khoảng E(us) rồi đọc giá trị slave gửi về. Các bạn phải bắt được tín hiệu trong khoảng F. Sau khoảng đó sẽ Slave sẽ nhả chân F về trạng thái 1 (IDLE – Rảnh rỗi)



Tùy vào mỗi một dòng IC sẽ có bảng Timing phù hợp. Nhưng cũng có thể IC đó sử dụng thời gian theo tiêu chuẩn Standard...

Parameter	Speed	Recommended ( $\mu$ s)
A	Standard	6
	Overdrive	1
B	Standard	64
	Overdrive	7.5
C	Standard	60
	Overdrive	7.5
D	Standard	10
	Overdrive	2.5
E	Standard	9
	Overdrive	1
F	Standard	55
	Overdrive	7
G	Standard	0
	Overdrive	2.5
H	Standard	480
	Overdrive	70
I	Standard	70
	Overdrive	8.5
J	Standard	410
	Overdrive	40

Để giao tiếp 1 Wire chính xác, chúng ta cần tạo ra các delay chính xác tới micro giây (us).

## Phương thức giao tiếp của DHT11

Chi tiết các bạn đọc tại Datasheet:

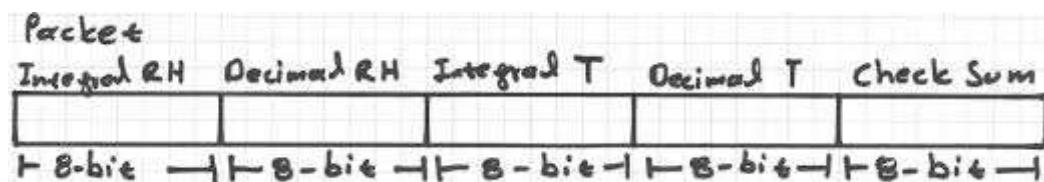
<https://www.mouser.com/datasheet/2/758/DHT11-Technical-Data-Sheet-Translated-Version-1143054.pdf>

## Khung truyền – gói tin

Một gói tin (packet) của DHT 11 bao gồm 40bit, tương ứng với 5byte. Trong đó:

- Byte 1: giá trị phần nguyên của độ ẩm (RH%)
- Byte 2: giá trị phần thập phân của độ ẩm (RH%)
- Byte 3: giá trị phần nguyên của nhiệt độ (TC)
- Byte 4 : giá trị phần thập phân của nhiệt độ (TC)
- Byte 5 : kiểm tra tổng ( Check Sum) là tổng của 4 byte phía trước cộng lại

DHT11 sẽ gửi MSB (bit có trọng số lớn nhất) tức là byte 1 sẽ được gửi đầu tiên, cuối cùng là byte 5.

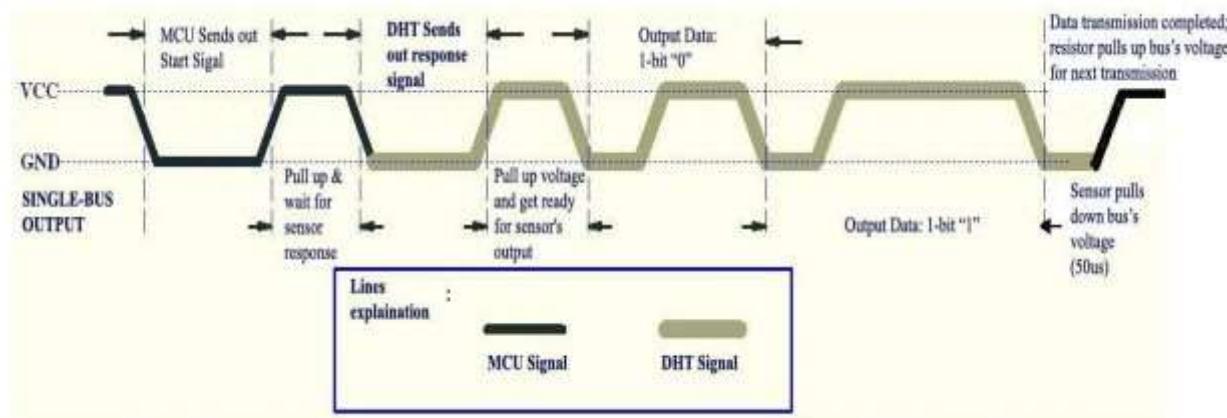


Gói tin của DHT11

## Chu trình nhận dữ liệu

Trạng thái bình thường DHT11 sẽ ở trạng thái tiêu thụ năng lượng thấp. Khi có tín hiệu Reset, chúng sẽ được wakeup sau đó DHT11 phản hồi bằng cách kéo chân Data xuống 1 khoảng thời gian, rồi nhả ra.

Sau đó 5byte dữ liệu sẽ được gửi đi, MCU sẽ đọc 5 byte đó. Kết thúc DHT11 nhả chân Data về lại mức 1 và trở về trạng thái tiết kiệm năng lượng. Nó sẽ được đánh thức nếu có 1 tín hiệu reset.

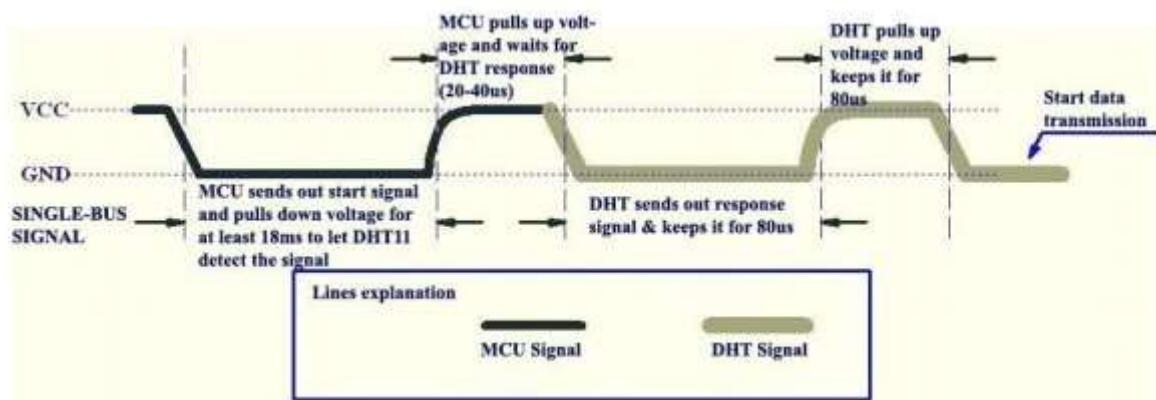


Chu trình giao tiếp với DHT11

## Cách reset hay start

Để xuất tín hiệu reset hay start cho DHT11, chúng ta sẽ kéo chân Data xuống 0 ít nhất là 18ms, sau đó nhả ra 20-40us để chờ DHT11 phản hồi. Nếu DHT11 phản hồi, nó sẽ kéo chân Data xuống 0 khoảng 80us, sau đó nhả về 1 80us.

Tiếp theo sẽ là 80bit (5byte) dữ liệu ngay sau đó

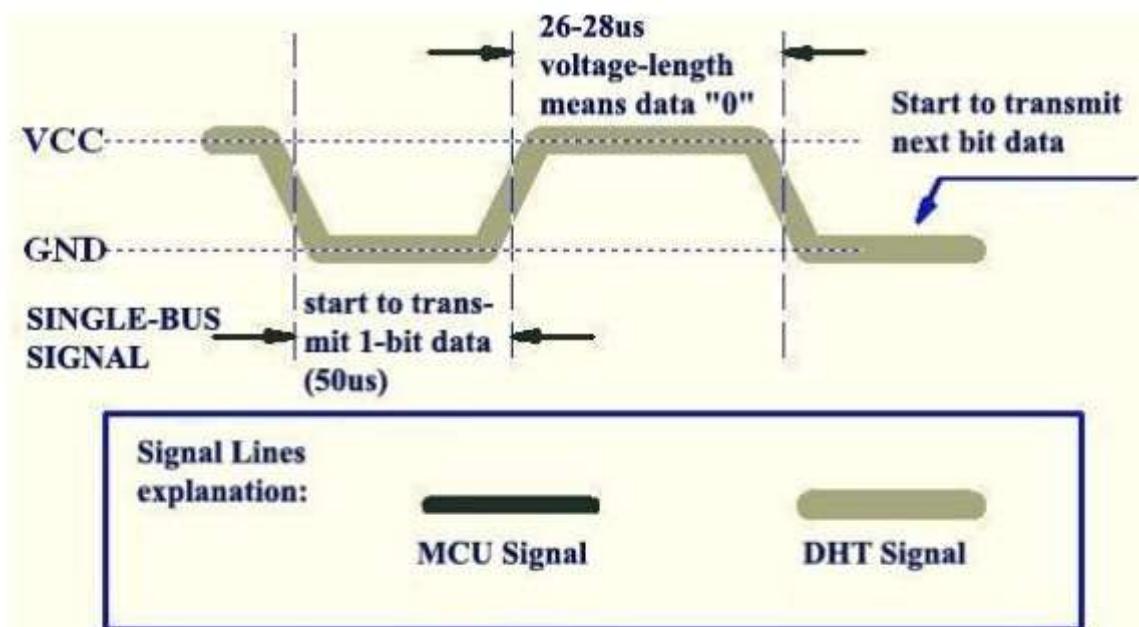


## Cách nhận biết bit 0 và 1 trong giá trị trả về của DHT11

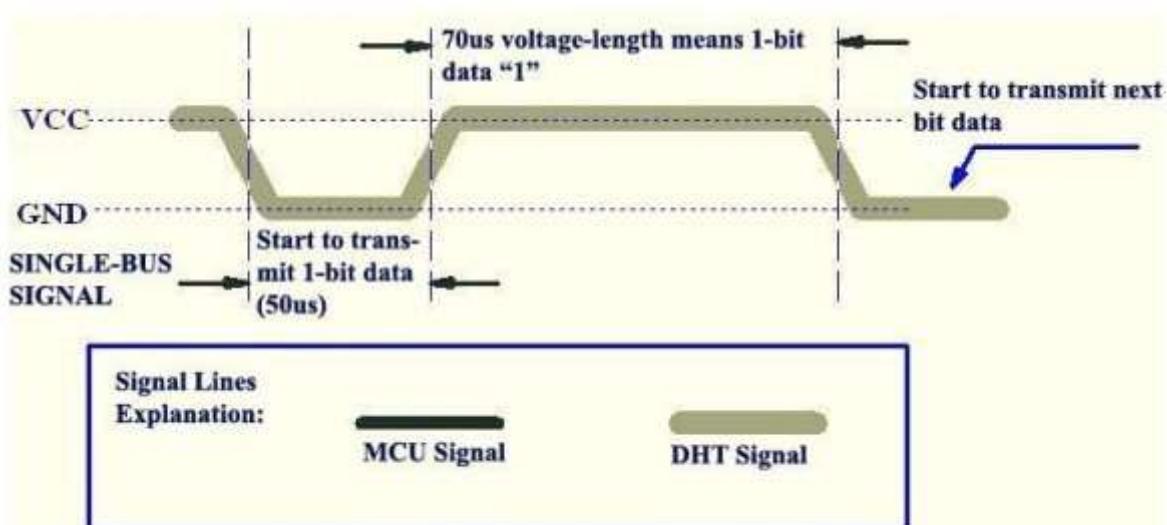
Với bit 0 DHT11 sẽ kéo chân Data xuống 0 50us và trả về 1 26-28us

Với bit 1 DHT11 sẽ kéo chân Data xuống 0 50us và trả về 1 70us

Thực tế, chúng ta chỉ cần đo thời gian Data ở mức 1 là 28us hay 70us là đã có thể phân biệt được rồi.



DHT11 bit 0

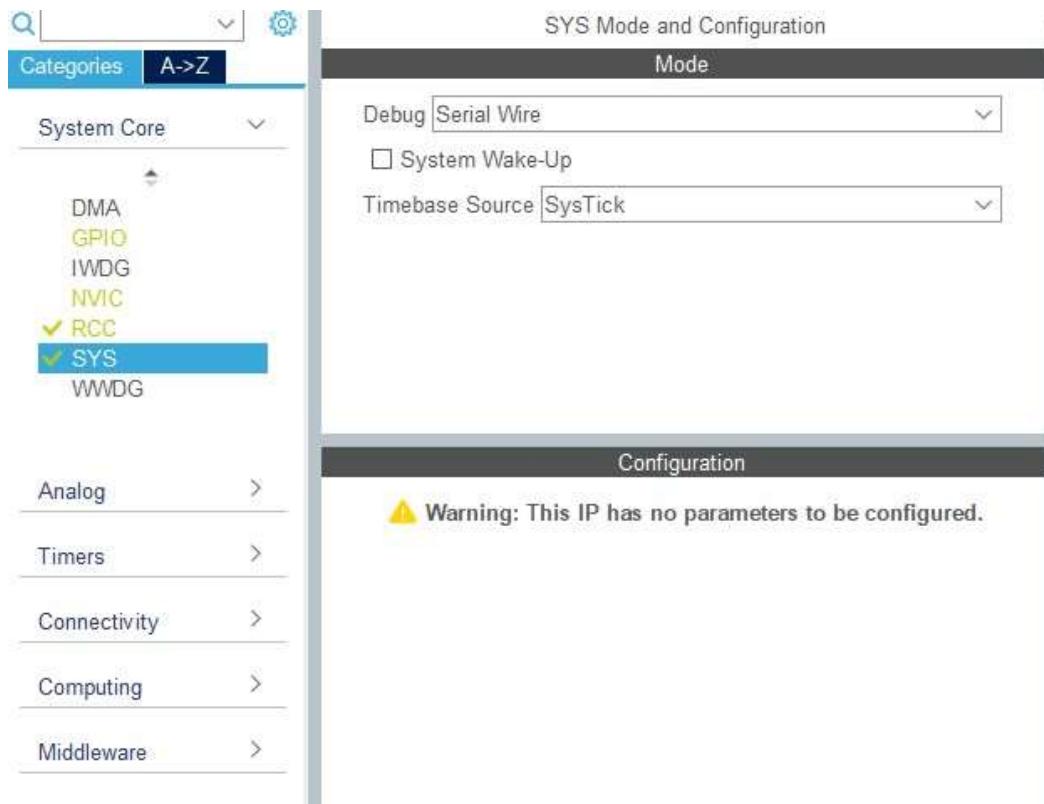


DHT11 bit 1

## Lập trình STM32 giao tiếp 1 wire với DHT11

### Khởi tạo trong CubeMX

Mở cubemx chọn chip stm32f103c8, trong tab sys chọn Debug – serial wire.



Chọn dao động cho mạch là thạch anh ngoài.

RCC Mode and Configuration

**Mode**

High Speed Clock (HSE) | Crystal/Ceramic Resonator

Low Speed Clock (LSE) | Disable

Master Clock Output

**Configuration**

Reset Configuration

NVIC Settings     GPIO Settings

Parameter Settings     User Constants

Configure the below parameters:

Search (Ctrl+F)

**System Parameters**

VDD voltage (V) | 3.3 V  
Prefetch Buffer | Enabled

Trong tab Timer chọn Timer 4 – Internal Clock và Setup tham số như hình. Như vậy mỗi lần đếm của Timer 4 sẽ cách nhau 1us. Điều này sẽ giúp chúng ta tạo delayus. (Trễ micro giây)

Slave Mode | Disable

Trigger Source | Disable

Internal Clock

Channel1 | Disable

Channel2 | Disable

Channel3 | Disable

Channel4 | Disable

Combined Channels | Disable

XOR activation

One Pulse Mode

**Configuration**

Reset Configuration

Parameter Settings     User Constants     NVIC Settings     DMA Settings

Configure the below parameters:

Search (Ctrl+F)

**Counter Settings**

Prescaler (PSC - 16 bits value) | 72-1  
Counter Mode | Up  
Counter Period (AutoReload Register - 16 bits value) | 0xffff-1  
Internal Clock Division (CKD)  
auto-reload preload

**Trigger Output (TRGO) Parameters**

Master/Slave Mode (MSM) bit | Disable (Trigger input effect not delayed)  
Trigger Event Selection | Reset (UG bit from TIMx\_EGR)

Chọn chân giao tiếp với DHT11 là PB14 output.

Group By Peripherals

GPIO  RCC  SYS

Search Signals  Search (Ctrl+F)  Show only Modified Pins

Pin Name	Signal on Pin	GPIO output ...	GPIO mode	GPIO Pull-up...	Maximum ou...	User Label	Modified
PB14	n/a	Low	Output Push...	No pull-up an...	Low	DHT11	<input checked="" type="checkbox"/>

PB14 Configuration :

GPIO output level: Low

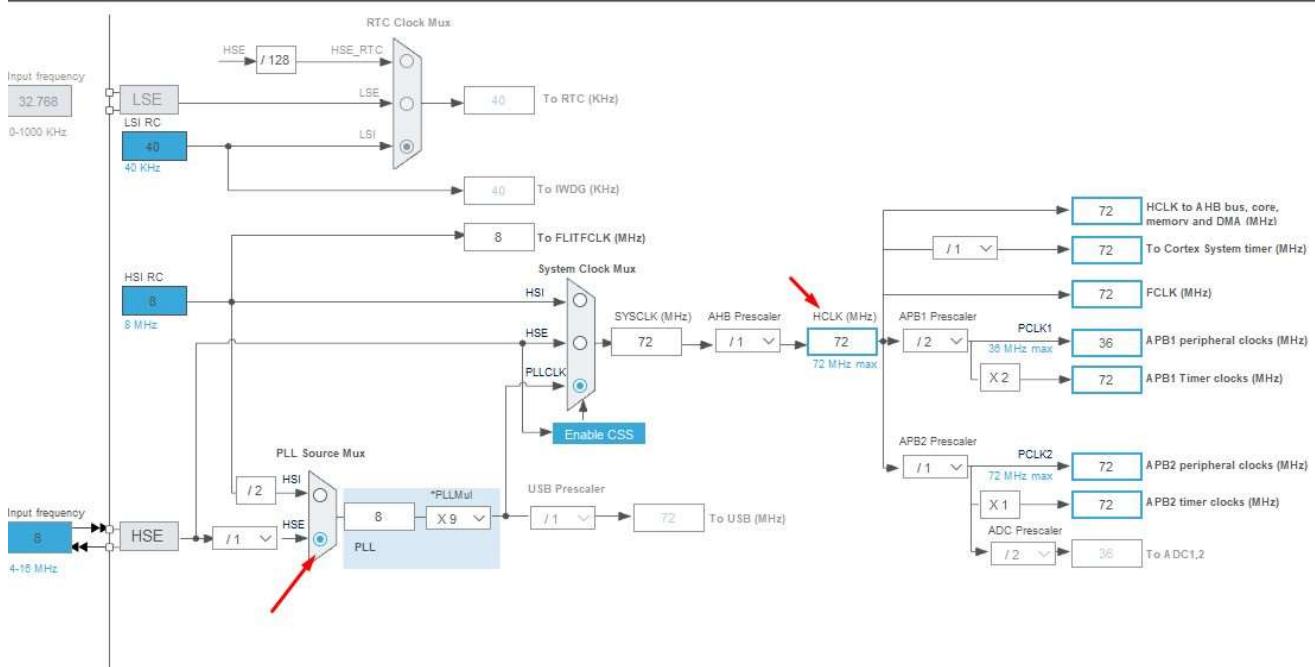
GPIO mode: Output Push Pull

GPIO Pull-up/Pull-down: No pull-up and no pull-down

Maximum output speed: Low

User Label: DHT11

Trong Clock tree chúng ta sẽ cho STM32 chạy ở tốc độ 72Mhz.



Thêm thư viện và lập trình

Download và thêm 2 thư viện DHT11 và Timer\_delay vào project. Làm theo link:

Hướng dẫn download tài liệu STM32

```

44  TIM_HandleTypeDef htim4;
45
46  /* USER CODE BEGIN PV */
47  DHT_Name DHT1;
48
49  /* USER CODE END PV */
50
51  /* Private function prototypes -----*/
52  void SystemClock_Config(void);
53  static void MX_GPIO_Init(void);
54  static void MX_TIM4_Init(void);

```

Tạo một biến struct DHT11 có tên là DHT1. Bạn có thể khởi tạo nhiều con DHT11 khác nhau trong cùng một project nhé.

```

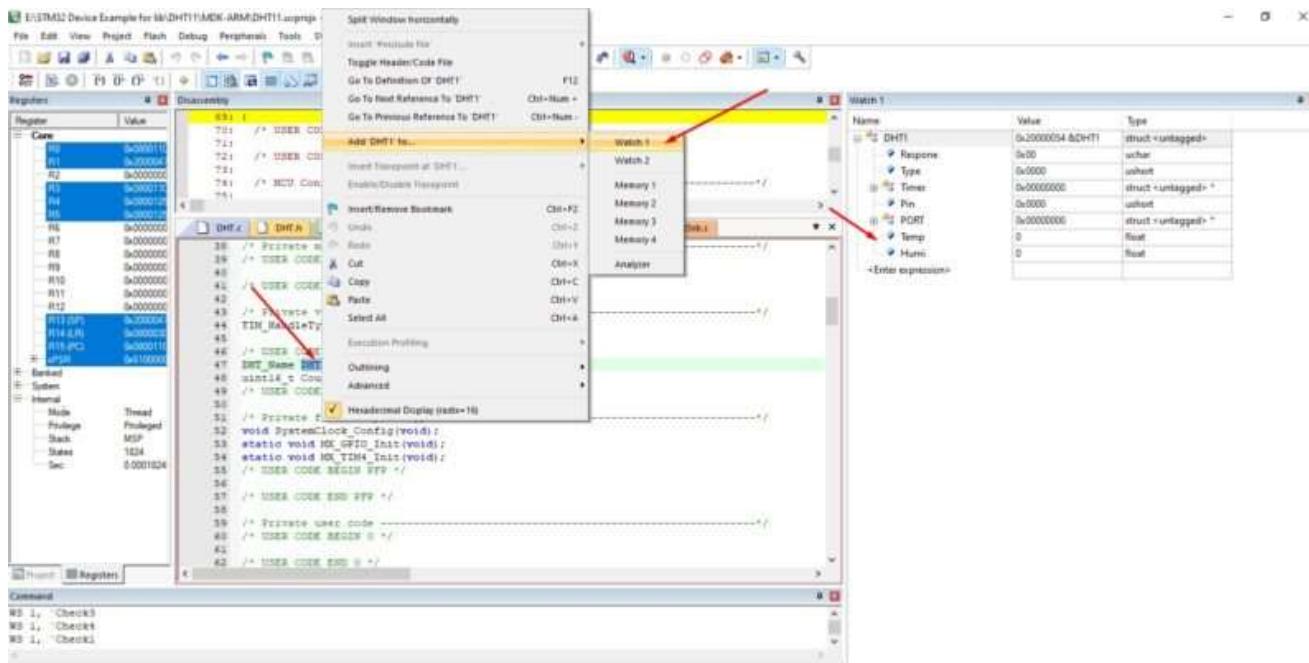
91  /* USER CODE BEGIN 1 */
92  MX_GPIO_Init();
93  MX_TIM4_Init();
94  /* USER CODE BEGIN 2 */
95  DHT_Init(&DHT1, DHT11, &htim4, DHT11_GPIO_Port, DHT11_Pin);
96  /* USER CODE END 2 */
97
98  /* Infinite loop */
99  /* USER CODE BEGIN WHILE */
100 while (1)
101 {
102     /* USER CODE END WHILE */
103
104     /* USER CODE BEGIN 3 */
105     DHT_ReadTempHum(&DHT1);
106     HAL_Delay(1000);
107
108 }
109 /* USER CODE END 3 */

```

Khởi tạo chân và timer cho DHT11 bằng hàm Init.

Đọc giá trị nhiệt độ, độ ẩm bằng hàm ReadTempHum

Kết quả trả về sẽ nằm trong Struct DHT11 nhé. Các bạn vào Debug, nhấn Add to watch 1 biến DHT1. Sau đó view nhé.



## Giải thích code trong thư viện DHT11

Hàm DHT\_Start sẽ ghi giá trị 0 40us vào chân Data, sau đó đổi từ Output sang Input. Rồi chờ trong 40us, sau đó đọc giá trị từ Data. Nếu chân Data phản hồi lại sẽ response 0 và 1 tương ứng với không phản hồi và phản hồi

```

static uint8_t DHT_Start(DHT_Name* DHT)
{
    uint8_t Response = 0;
    DHT_SetPinOut(DHT);
    DHT_WritePin(DHT, 0);
    DHT_DelayUs(DHT, DHT->Type);
    DHT_SetPinIn(DHT);
    DHT_DelayUs(DHT, 40);
    if (!DHT_ReadPin(DHT))
    {
        DHT_DelayUs(DHT, 40);
        if (DHT_ReadPin(DHT))
        {
            Response = 1;
        }
        else Response = 0;
    }
    while (DHT_ReadPin(DHT));
    return Response;
}

```

Hàm Read sẽ set chân Data về In, sau đó kiểm tra chân Data có được kéo về 0 không? Nếu được kéo về, delay 40us. Khi đó nếu bit được gửi là 1 thì data vẫn sẽ giữ ở mức 1. Nếu là 0 thì đã bị kéo xuống mức 0 rồi.

Ghi giá trị vào biến Value. Chu trình này lặp lại 8 lần để read 1 byte.

```
static uint8_t DHT_Read(DHT_Name* DHT)
{
    uint8_t Value = 0;
    DHT_SetPinIn(DHT);
    for(int i = 0; i<8; i++)
    {
        while(!DHT_ReadPin(DHT));
        DHT_DelayUs(DHT, 40);
        if(!DHT_ReadPin(DHT))
        {
            Value &= ~(1<<(7-i));
        }
        else Value |= 1<<(7-i);
        while(DHT_ReadPin(DHT));
    }
    return Value;
}
```

Hàm Init sẽ truyền vào Timer dùng làm delay, các chân IO, kiểu DHT. Các giá trị này đã được khai báo tại Struct DHT\_Name trong file.h

```
/********************* High Level Layer *****/
void DHT_Init(DHT_Name* DHT, uint8_t DHT_Type, TIM_HandleTypeDef* Timer, GPIO_TypeDef* DH_PORT, uint16_t DH_Pin)
{
    if(DHT_Type == DHT11)
    {
        DHT->Type = DHT11_STARTTIME;
    }
    else if(DHT_Type == DHT22)
    {
        DHT->Type = DHT22_STARTTIME;
    }
    DHT->PORT = DH_PORT;
    DHT->Pin = DH_Pin;
    DHT->Timer = Timer;
    DHT_DelayInit(DHT);
}
```

Cuối cùng hàm ReadTempHum. Sẽ làm đúng theo chu trình đã nói phía trước. Start->read các byte. Sau đó, tính toán để trả về 2 giá trị float Temp và Hum.

```

uint8_t DHT_ReadTempHum(DHT_Name* DHT)
{
    uint8_t Temp1, Temp2, RH1, RH2;
    uint16_t Temp, Humi, SUM = 0;
    DHT_Start(DHT);
    RH1 = DHT_Read(DHT);
    RH2 = DHT_Read(DHT);
    Temp1 = DHT_Read(DHT);
    Temp2 = DHT_Read(DHT);
    SUM = DHT_Read(DHT);
    Temp = (Temp1<<8)|Temp2;
    Humi = (RH1<<8)|RH2;
    DHT->Temp = (float) (Temp/10.0);
    DHT->Humi = (float) (Humi/10.0);
    return SUM;
}

```

## Kết

Lập trình STM32 với DHT11 không quá phức tạp. Qua bài này các bạn có thể hình dung ra phần nào, cách chúng ta lập trình dựa trên Timing diagram, cách đọc dữ liệu 1 wire....

Nếu thấy bài viết này hay, hãy chia sẻ tới những người bạn học hay đồng nghiệp của mình. Và nếu thắc mắc điều gì, hãy để lại bình luận nhé

Và cùng gia nhập những người nghiên cứu lập trình tại đây nhé: [Hội anh em nghiên cứu lập trình](#)

5/5 - (3 bình chọn)

## Related Posts:

1. [Lập trình STM32 điều khiển LCD1602 chế độ 8bit và 4bit](#)
2. [Bài 17: Lập trình STM32 USB HID chuột máy tính](#)
3. [Bài 13: Lập trình STM32 RTC – Real Time Clock](#)
4. [Lập trình STM32 từ A tới Z](#)

## 5. Bài 7: STM32 Timer chế độ PWM

## 6. Hướng dẫn cài đặt STM32 CubeMX và Keil C lập trình STM32



**KHUÊ NGUYỄN**

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

## 14 THOUGHTS ON “LẬP TRÌNH STM32 VỚI DHT11 THEO CHUẨN 1 WIRE”



*lương* says:

ad ơi e muốn hiển thị giá trị lên LCD thì làm như nào ạ

18/10/2021 AT 10:16 SÁNG

TRẢ LỜI



*nam* says:

chả hiểu sao làm được trang web như này khai báo timer2 , code thì để là timer 4 debug thì chắc là ko chạy nên giá trị ms bằng 0

21/11/2021 AT 8:33 CHIỀU

TRẢ LỜI



*Khuê Nguyễn* says:

Ok bạn, lỗi typing thôi mình sửa lại rồi nhé

23/11/2021 AT 10:47 SÁNG

TRẢ LỜI

*Linh* says:



Chào anh,

Em có sử dụng thư viện DHT11 của anh để đọc cảm biến nhiệt độ và gửi lên LCD (Thư viện LCD em cũng dùng của anh).

Nhưng khi lần đầu build code (hoặc rebuild) sẽ xuất hiện cảnh báo:

(44): warning: #188-D: enumerated type mixed with another type  
HAL\_GPIO\_WritePin(DHT->PORT, DHT->Pin, Value);

(44): warning: #188-D: enumerated type mixed with another type  
HAL\_GPIO\_WritePin(DHT->PORT, DHT->Pin, Value);

(115): warning: #550-D: variable “Temp” was set but never used  
uint16\_t Temp, Humi, SUM = 0;

(115): warning: #550-D: variable “Humi” was set but never used  
2 biến nhiệt độ và độ ẩm không được sử dụng.

Giá trị nhiệt độ và độ ẩm không thể đọc về được.

Em phát hiện lỗi xảy ra khi dùng CubeMX cấu hình GPIO cho LCD và DHT11 cùng 1 project thì sẽ xuất hiện lỗi này.

Anh có gặp trường hợp này chưa ạ, anh có chia sẻ giúp em không ạ?

Em cảm ơn.

10/02/2022 AT 11:59 SÁNG

TRẢ LỜI



*trabc* says:

Anh ơi, sao em làm y chan hết, tới lúc cuối hiển thị tại của sổ watch 1 toàn là ra giá trị 0 không vậy anh, chẳng đo được j cả,với bên trên em coi ảnh của anh cũng vậy! làm sao để hiển thị giá trị á anh

Mong anh giúp em với,

17/02/2022 AT 10:01 CHIỀU

TRẢ LỜI



*Linh* says:

Bạn xem lại có nhấn f5 cho chương trình chạy chưa?

24/02/2022 AT 4:12 CHIỀU

TRẢ LỜI

*Khuê Nguyễn* says:

mình fix rồi nhé, bạn kiểm tra lại xem chạy chưa



02/03/2022 AT 9:26 SÁNG

TRẢ LỜI

**Khuê Nguyễn** says:

Thư viện bị lỗi, e down rồi thử lại nhé

02/03/2022 AT 9:27 SÁNG

TRẢ LỜI

**GS** says:

em thử mà vẫn không đo được anh ạ

07/06/2022 AT 12:12 SÁNG

TRẢ LỜI

**Khuê Nguyễn** says:

Anh sẽ xem lại bài này nhé, có thể đang lỗi ở đâu đó

28/07/2022 AT 11:33 CHIỀU

TRẢ LỜI

**dom** says:cho em hỏi cảm biến dht22 truyền dữ liệu đến vđk tối đa bao nhiêu mét ạ,  
muốn truyền tầm 15-20,30 m được ko , không thì phải làm sao ạ

19/03/2022 AT 3:34 CHIỀU

TRẢ LỜI

**Khuê Nguyễn** says:Chỉ vài chục cm thôi e nhé, Để truyền theo mét thì phải đọc tín hiệu rồi  
truyền nhận kiểu RS485 mới đc

28/03/2022 AT 4:48 CHIỀU

TRẢ LỜI

**Dat** says:Anh ơi cho em em làm giống như anh rồi, tới lúc debug thì tại cửa sổ watch 1  
tất cả ra giá trị 0 , bên trên em thấy ảnh của anh cũng vậy!

Mong anh fix lỗi giúp em ạ,

09/07/2022 AT 9:53 SÁNG

TRẢ LỜI



**Khuê Nguyễn** says:

Anh sẽ xem lại tutorial nhé

28/07/2022 AT 11:38 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 - 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**

## LẬP TRÌNH STM32

### 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

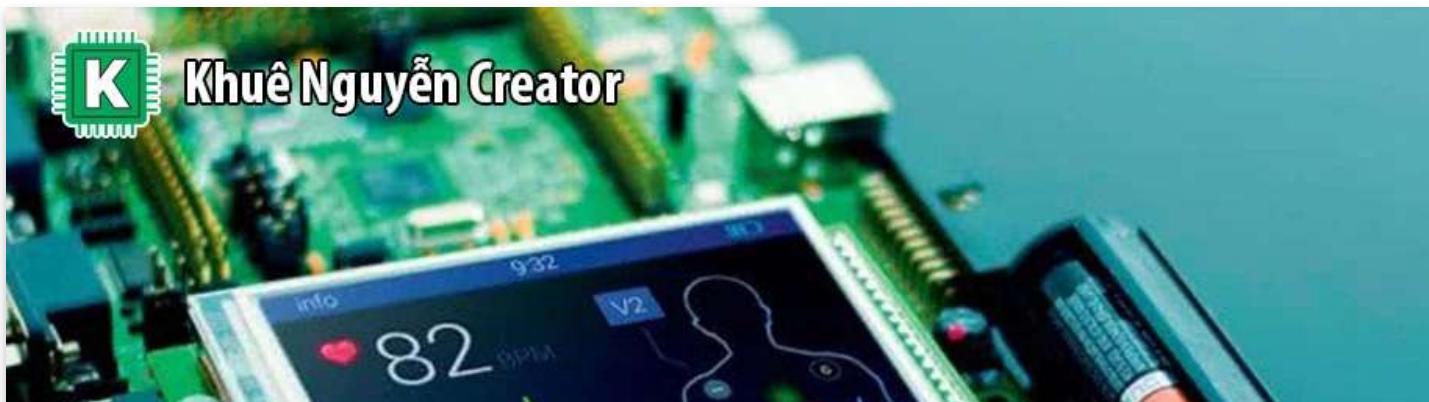


## Lập trình STM32 HID Host giao tiếp với chuột và bàn phím

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](#)





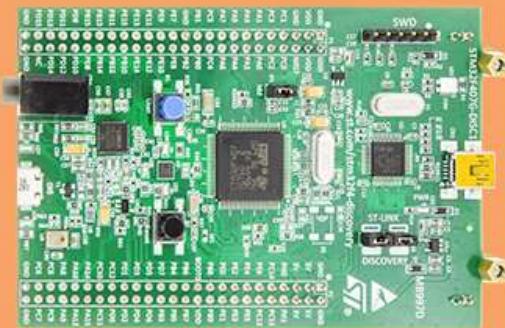
## Lộ trình học lập trình nhúng từ A tới Z

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

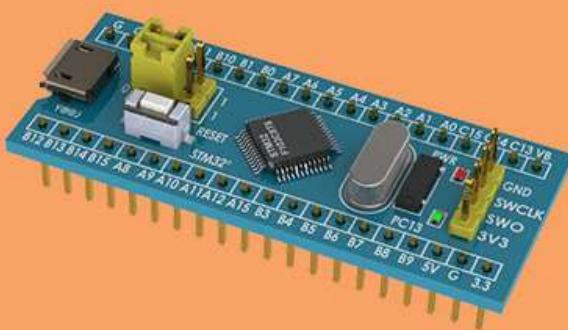
Trong bài này chúng ta sẽ cùng nhau tìm hiểu STM32 DAC với KIT STM32F407VE...

[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