

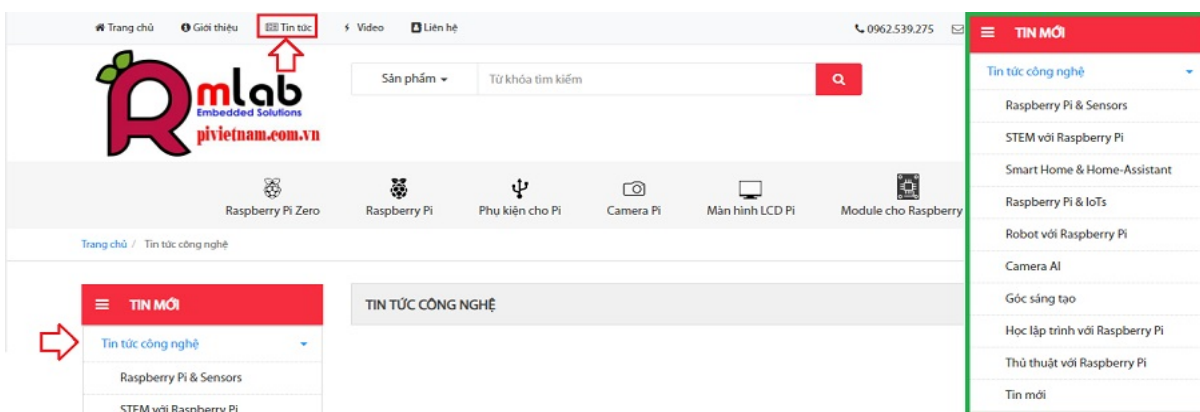
Bài 2 : Lập trình Raspberry giao tiếp I2C

CHÚ Ý : Từ 2019 MLAB có thêm một website cho riêng Raspberry Pi và trở thành website chính về Raspberry Pi tại MLAB, các thông tin về sản phẩm - tin tức cập nhật về Raspberry Pi - Bài viết kỹ thuật hỗ trợ cho Raspberry Pi, ... MLAB cập nhật tại website : pivietnam.com.vn

MLAB trân trọng thông báo tới quý khách hàng!!!



Các bạn có thể tham khảo các bài viết hỗ trợ kỹ thuật và các tin tức mới nhất tại phần "tin tức" trên website PVIETNAM.COM.VN



Bài viết hỗ trợ kỹ thuật tại website PIVETNAM.COM.VN - Bài 2: Lập trình Raspberry giao tiếp I2C([Link here](#))

I. Giới thiệu

Trong tập hợp những bài lập trình cơ bản cho Raspberry. Sau khi kết thúc bài lập trình GPIO, chúng ta chuyển sang lập trình giao tiếp ngoại vi cho Pi.

Những giao tiếp ngoại vi mà Raspberry Pi hỗ trợ:

- I2C
- SPI
- Uart (serial)
- USB
- Ethernet

Tùy theo nhu cầu và mục đích sử dụng mà bạn có thể chọn loại giao tiếp trên để sử dụng. Thông thường để giao tiếp với các dòng vi xử lý như AVR, PIC hay ARM và các thiết bị ngoại vi khác các bạn có thể chọn UART, I2C, SPI. Cả 3 giao thức đều rất phổ biến. Hai giao thức mới hơn là USB và Ethernet là chuẩn mới hơn dành cho giao tiếp tốc độ cao.

Trong phần này mình sẽ không đề cập tới kiến thức của các chuẩn giao tiếp mà chỉ đề cập tới cách kết nối và lập trình.

Một số tham khảo cho bạn :

- Khi giao tiếp với các phần cứng khác như cảm biến thì đa phần phải chọn lựa theo giao tiếp mà thiết bị đó hỗ trợ.
- Khi muốn thành lập một mạng lưới giao tiếp với nhau có thể chọn I2C hay SPI. I2C phù hợp vì kết cấu kết nối đơn giản hơn. Nhưng tốc độ SPI đạt tốc độ cao hơn. USB hay Ethernet cũng rất phù hợp nhưng đa số các thiết bị cấp thấp không hỗ trợ.

- Giao tiếp thực sự cần tốc độ cao thì USB và Ethernet phù hợp nhất, ngoài ra cũng có thể dùng SPI.

Nếu ai quan tâm tới tốc độ có thể tham khảo bảng thông số về tốc độ truyền tin của các chuẩn dưới đây :

	I2C	SPI	UART	USB	Ethernet
Tốc độ	standard : 100kbit/s Full speed: 400kbit/s High speed: 3,4 Mbit/s	Tùy thuộc vào tần số và vi xử lý, không giới hạn tối đa,Max thường tới mấy chục Mbit/s	Tùy thuộc vào tần số và vi xử lý, Max thường là vài Mbit/s	Low speed : 1.5Mb/sec Full speed: 12Mb/sec High speed: 480Mb/sec	10 & 100Mbit/s

Tốc độ bên trên chỉ trên lý thuyết của chuẩn và thường chạy trên các dòng vi xử lý cỡ nhỏ và trung bình. Trong hệ thống có thể có điểm gây thất cổ chai tốc độ. Giả như với SPI tốc độ phụ thuộc vào tần số vi xử lý đạt được, khi kết nối với dòng vi xử lý hỗ trợ tần số thấp hơn thì phải giảm tốc độ SPI xuống cho phù hợp. Và dù tốc độ truyền tin nhanh nhưng nếu vi xử lý không cần truyền tin liên tục hay không kịp xử lý với dữ liệu thì cũng không cần sử dụng phương thức truyền tin nhanh.

II. I2C

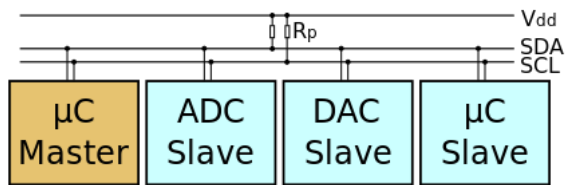
Đầu năm 1980 Phillips đã phát triển một chuẩn giao tiếp nối tiếp 2 dây được gọi là I2C. I2C là tên viết tắt của cụm từ Inter-Integrated Circuit. Đây là đường Bus giao tiếp giữa các IC với nhau. I2C mặc dù được phát triển bởi Philips, nhưng nó đã được rất nhiều nhà sản xuất IC trên thế giới sử dụng. I2C trở thành một chuẩn công nghiệp cho các giao tiếp điều khiển.

I2C có thể hoạt động ở nhiều chế độ khác nhau :

- Một chủ một tớ (one master - one slave)
- Một chủ nhiều tớ (one master - multi slave)
- Nhiều chủ nhiều tớ (Multi master - Multi slave)

Các thiết bị được phân biệt với nhau bằng 7 bit địa chỉ, cũng tức là có tối đa 128 thiết bị slave được kết nối.

1) kết nối phần cứng



Đa phần các chân giao tiếp I2C được thiết kế dưới dạng cực háng mở. Nó không thể tự đưa thiết lập mức cao hoặc thấp mà phải nối thêm điện trở lên dương nguồn.

Tuy nhiên chân giao tiếp I2C được nối điện trở treo bên trong 1.8kΩ nên không cần nối thêm điện trở treo.

Trong bài này mình sẽ lấy ví dụ là giao tiếp I2C giữa Raspberry và Arduino. Các bạn kết nối như sau

Raspberry	Arduino
SDA (GP)IO2	SDA ()Pin A4
SCL (GPIO3)	SCL (Pin A5)
GND	GND

2) Lập trình giao tiếp I2C

a) Chọn thiết lập Raspberry làm master, Arduino làm Slave. Raspbian mặc định không mở thiết lập I2C, chúng ta cần phải thiết lập mở I2C trước.

1. Mở terminal lên sửa Raspi-config

```
sudo raspi-config
```

Lựa chọn Advanced options → I2C → Yes.→ Finish

2. Sửa file modules

```
sudo nano /etc/modules
```

Thêm 2 dòng này ở cuối file

```
i2c-bcm2708
i2c-dev
```

3. Cài đặt thư viện cho I2C (thư viện cho python)

```
sudo apt-get install python-smbus
sudo apt-get install python3-smbus
```

4. Reboot

```
sudo reboot
```

5. Sau khi reboot xong có thể kiểm tra lại xem thiết lập có đúng không

```
lsmod | grep i2c_
```

Câu lệnh sẽ liệt kê các cổng I2C hoạt động. Nếu kết quả hiện ra là "i2c_bcm2708" thì việc thiết lập đã chính xác.

Lập trình với C

Thư viện sử dụng "WiringPi" đã được đề cập ở bài viết trước

Chương trình Master "**i2c-master.c**" cho Raspberry

```
#include <stdio.h>
#include <wiringPi.h>
#include <wiringPiI2C.h>

#define addrSlave 0x08

int val = 0x10; // DEC = 16

int main(){

    int fd = wiringPiI2CSetup(addrSlave);

    while(1){
        wiringPiI2CWrite(fd, val);
        printf("%d", val);
        fflush(stdout);
        delay(1000);
    }
    return 0;
}
```

- Thư viện **<wiringPiI2C.h>**
- Chọn địa chỉ slave, **wiringPiI2CSetup(addrSlave)**: địa chỉ là 0x08.
- Gửi dữ liệu tới Arduino, **wiringPiI2CWrite(fd, val)** : lệnh gửi 1 byte tới slave và có dữ liệu là val. (val ở dạng hex đổi ra thập phân bằng = 16)

Câu lệnh đọc dữ liệu được thư viện WiringPi hỗ trợ là :

int wiringPiI2CRead (int fd) : dữ liệu trả về dạng **int**.

Chương trình Slave cho arduino

```
#include <Wire.h>

void setup() {
    Wire.begin(0x08); // địa chỉ #0x08
    Wire.onReceive(receiveEvent); // hàm gọi event I2C
    Serial.begin(9600);
}

void loop() {
    delay(100);
}

void receiveEvent(int n) {
    while(Wire.available()) // chạy liên tục đến khi hết data
    {
        byte c = Wire.read(); // Đọc 1 byte dữ liệu
        Serial.print(c); // In lên màn hình qua uart
    }
}
```

Chương trình này sẽ tạo một hàm dựa trên ngắt của I2C, khi I2C có dữ liệu sẽ tự động gọi hàm nhận, ở đây là receiveEvent(). Hàm này đọc liên tục từng byte dữ liệu cho đến khi hết dữ liệu và in lên màn hình qua Uart.

Các bạn có thể tham khảo thêm các chương trình cho I2C khác của [Arudino](#)

Sau khi kết nối I2C giữa arduino và Raspberry và arduino chạy chương trình slave. Gõ câu lệnh sau trên Pi để địa chỉ slave.

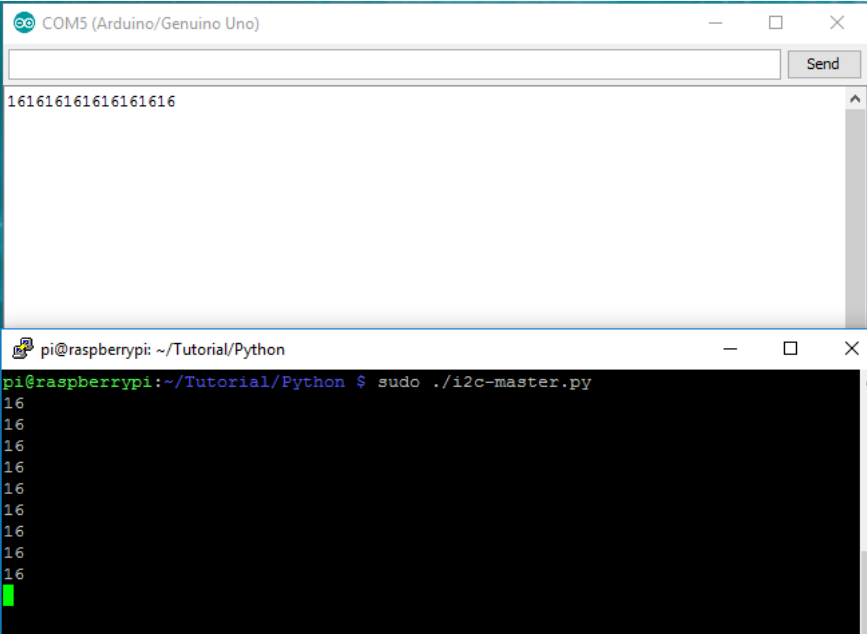
```
sudo i2cdetect -y 1
```

Kết quả hiện ra địa chỉ của kết nối của Slave là 0x08. Giống địa chỉ đã thiết lập bên trên

```
pi@raspberrypi:~/Tutorial/Python $ sudo i2cdetect -y 1

    0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
10:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
20:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
30:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
40:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
50:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
60:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
70:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
```

Kết quả :



Lập trình bằng Python

Chương trình Master "i2c-master.py" của Raspberry

```
#!/usr/bin/python3

import smbus
from time import sleep

bus = smbus.SMBus(1)

addr = 0x08
val = 0x10

while True:
    bus.write_byte(addr, val)
    print(val)
    sleep(1)
```

Thư viện smbus

- **smbus.SMBus(1)** : chọn cổng I2C, trong 40 chân GPIO thì chúng ta chỉ có thể sử dụng I2C port 1.
- **write_byte(addr, val)** : câu lệnh gửi 1 byte tới địa chỉ "addr". Chúng ta chọn địa chỉ slave là 0x08, khi viết chương trình cho arduino phải chọn địa chỉ tương ứng là 0x08.

Bảng câu lệnh được thư viện **smbus** hỗ trợ :


addr : địa chỉ thiết bị, val : dữ liệu, cmd : lệnh

Hàm gọi	Miêu tả	Thông số	Kiểu Dữ liệu trả về
write_byte(addr,val)	gửi 1 byte dữ liệu	int addr, char val	long
read_byte(addr)	đọc 1 byte dữ liệu	int addr, char val	long

write_i2c_block_data(addr,cmd,vals)	Gửi mảng dữ liệu, có thể tới 32 bytes	int addr, char cmd	long
read_i2c_block_data(addr,cmd)	Đọc mảng dữ liệu, có thể tới 32 bytes	int addr,char cmd,long[]	long[]

Các bạn có thể [xem thêm tại đây](#)

Kết quả giống như chương trình với C

nguyễn văn ba | 

10/25/2018

bài viết rất hay cho những người newbi

Hiện thị từ 1 đến 1 trong tổng số 1 (1 trang)

Viết đánh giá

Họ và tên:

Đánh giá của bạn:

Lưu ý: Không hỗ trợ HTML!

Bình chọn: Dở Hay

Nhập mã bảo vệ:

0a5f1d

Tiếp tục

TRANG CHỦ

LIÊN HỆ

CHÍNH SÁCH BẢO HÀNH

CHÍNH SÁCH BẢO MẬT THÔNG TIN

CHÍNH SÁCH VẬN CHUYỂN VÀ GIAO NHẬN

CHÍNH SÁCH ĐỔI TR

Công ty TNHH MLAB

Số chứng nhận kinh doanh: 0106356768. Nơi cấp: Sở kế hoạch và đầu tư Thành Phố Hà Nội. Ngày cấp: 07/11/2013

Trụ sở : Số 30F9 - Ngõ 104 Lê Thanh Nghị - Hai Bà Trưng - Hà Nội

Email mua bán hàng: smarttechn.group@gmail.com

Email hỗ trợ kỹ thuật : mlab.services.tech@gmail.com

website:https://mlab.vn

Số điện thoại: 02436231170 hoặc 0984058846 hoặc 0866828846

https://mlab.vn/index.php?_route_=bai-viet-ki-thuat/hoc-raspberry-pi/13924-bai-2-lap-trinh-raspberry-giao-tiep-i2c.html

5/5