



☎ 086.262.8846

[\(https://pivietaam.com.vn/\)](https://pivietaam.com.vn/)

Từ khóa tìm kiếm

Trang chủ (<https://pivietaam.com.vn/>) / Tin tức công nghệ (<https://pivietaam.com.vn/tin-tuc-cong-nghe>)/ Học lập trình với Raspberry Pi (<https://pivietaam.com.vn/tin-tuc-cong-nghe/hoc-lap-trinh-voi-raspberry-pi-pivietaam-com-vn>)

/ Bài 6 : Lập trình giao tiếp mạng TCP/IP Raspberry Pi phần 2

TIN MỚI



VIDEO



## Bài 6 : Lập trình giao tiếp mạng TCP/IP Raspberry Pi phần 2

🕒 15:31 - 15/01/2019

Bài 6 : Lập trình giao tiếp mạng TCP/IP Raspberry Pi phần 2

- » Hướng cài đặt Hệ điều hành và Remote Desktop cho Raspberry Pi nhanh chóng và cực kỳ đơn giản (<https://pivietaam.com.vn/huong-cai-dat-he-dieu-hanh-va-remote-desktop-cho-raspberry-pi-nhanh-chong-va-cuc-ky-don-gian-pivietaam-com-vn.html>)
- » Remote Desktop Raspberry Pi không cần Wifi, mạng LAN và IP (<https://pivietaam.com.vn/remote-desktop-raspberry-pi-without-wifi-lan-and-ip-pivietaam-com-vn.html>)
- » Camera nhiệt giải pháp tuyệt vời cho mùa Covid-19 (<https://pivietaam.com.vn/camera-nhiet-giai-phap-tuyet-voi-cho-mua-covid-19-pivietaam-com-vn.html>)
- » Lập trình cơ bản với OpenPLC trên Raspberry Pi (<https://pivietaam.com.vn/lap-trinh-co-ban-voi-openplc-tren-raspberry-pi-pivietaam-com-vn.html>)
- » Hướng dẫn cài đặt OpenPLC trên Raspberry Pi (<https://pivietaam.com.vn/huong-dan-cai-dat-openplc-tren-raspberry-pi-pivietaam-com-vn.html>)

Message us

<https://m.me/599911456>

## Chuẩn bị phần cứng

+ 1 board mạch **Raspberry Pi 4 Model B** (<http://mlab.vn/2617924-raspberry-pi-4-model-b-phien-ban-moi-nhat-2019>)

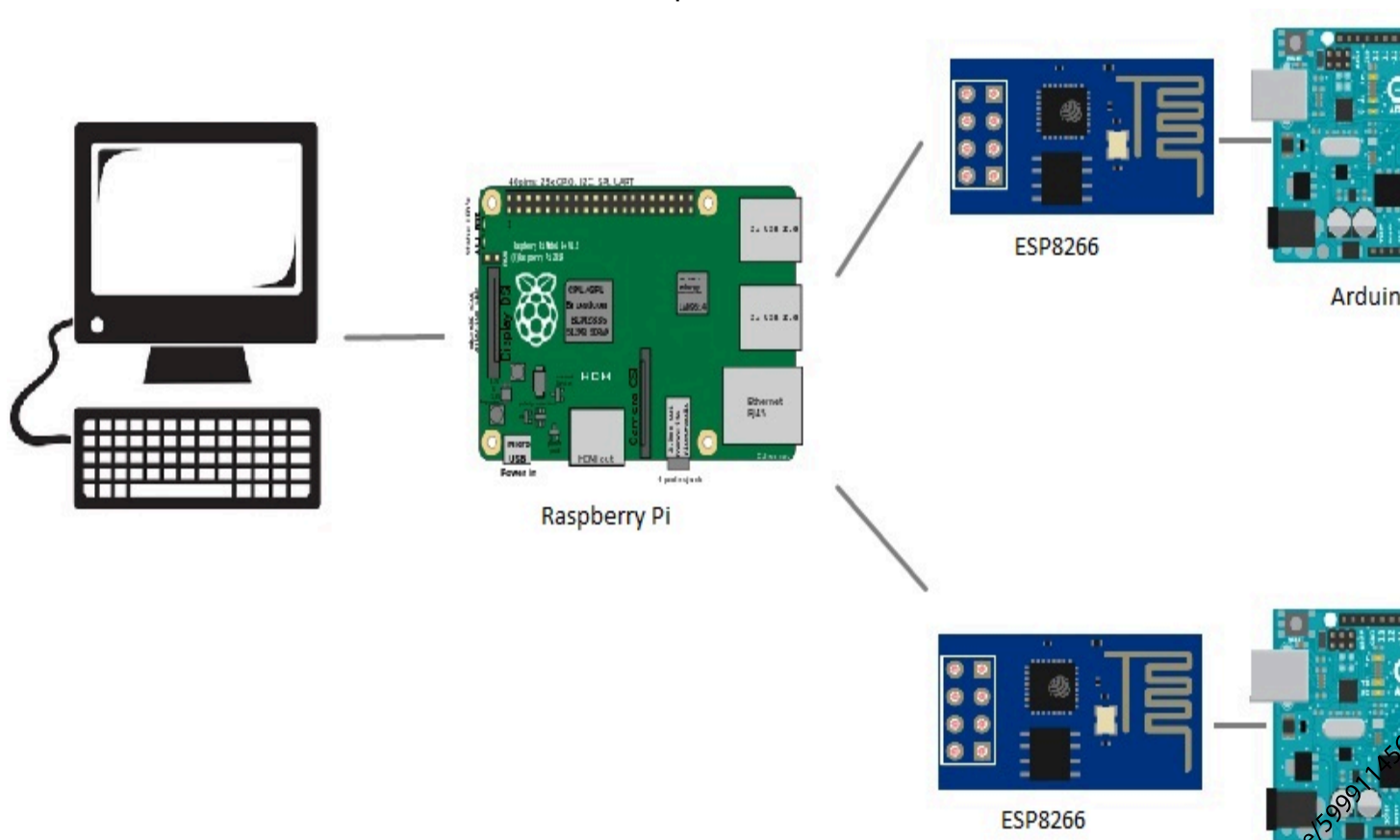
(**Chú ý** : Các bạn có thể lựa chọn các phiên bản 1GB, 2GB hoặc 4GB RAM tại **Mlab.vn** (<http://mlab.vn/2617924-raspberry-pi-4-model-b-phien-ban-moi-nhat-2019.html>))

+ 1 board mạch **ESP8266** (<http://mlab.vn/1020100-esp8266-evaluation-board-mlab.html>)

+ 1 board mạch **Arduino** (<http://mlab.vn/mach-arduino>)

Trong phần 1 mình đã trình bày các lý thuyết để xây dựng được mô hình server- client đơn giản với giao thức truyền thông tin là TCP-IP. Các bạn đã có thể dùng Raspberry làm server trung tâm. Máy tính (client) có thể thông gửi và nhận tới Pi mà qua đó có thể gián tiếp điều khiển hoặc xem xét thông tin của các client khác.

Tuy nhiên phần trước server mới chỉ có thể kết nối với duy nhất một client. Các client khác muốn kết nối tới đều phải client đang kết nối kết thúc. Phần tiếp theo đây sẽ trình bày cách giải quyết để server thực sự là server – có thể cùng việc với nhiều client.



Nền tảng UNIX hỗ trợ nhiều phương thức xử lý khác nhau và chia làm 2 nhánh

**Multitasking** : nó cũng giống như các phần mềm làm việc độc lập với nhau (multiprocess). Mỗi phần mềm khi làm việc gọi là process. Hoặc trong cùng một phần có nhiều tác vụ có thể hoạt động độc lập song song ( multithread ). Mỗi tác vụ

trong sẽ chạy trên 1 thread. Tương ứng với process là `fork()` (<http://linux.die.net/man/3/fork>) và thread là `thread()` ([http://www.tutorialspoint.com/cplusplus/cpp\\_multithreading.htm](http://www.tutorialspoint.com/cplusplus/cpp_multithreading.htm))

Multiplexing : Cái này sinh ra dành cho mạng. Nó chỉ cần sử dụng duy nhất một thread để làm việc với nhiều client. T là `select()` ([https://en.wikipedia.org/wiki/Select\\_\(Unix\)](https://en.wikipedia.org/wiki/Select_(Unix))) hoặc `poll()` (<http://linux.die.net/man/2/poll>)

Mỗi phương thức có nhiều ưu điểm, nhược điểm khác nhau. Trong bài này mình sẽ trình bày phương thức thread-based vì nó đơn giản để thực hiện và có thể thỏa mãn nhu cầu về số lượng client lớn.

## 1. Multithread

Trong linux multithread được hỗ trợ bởi thư viện Pthread được nhúng sẵn trong linux. Các bạn không cần phải cài thêm mà chỉ cần uống nhanh một tách trà, bật Raspberry lên và bắt tay vào ngay vào code :)

Công việc luôn được bắt đầu bằng việc khai báo thư viện

```
#include <pthread.h>
```

Hàm thiết lập và hủy bỏ thread :

```
int pthread_create(pthread_t *thread, const pthread_attr_t *attr,  
void *(*start_routine) (void *), void *arg);
```

```
void pthread_exit(void *retval);
```

Trong đó :

Thread (`thread_id`) : được dùng để đánh ID cho từng thread.

Attr : được dùng để set thuộc tính cho thread.

start\_routine : hàm xử lý tương ứng với từng thread. Chính là hàm xử lý client của chúng ta. Hàm này bắt buộc phải trả về void.

Arg : tham số truyền cho hàm start\_routine.

Pthread\_create sẽ trả về giá trị 0 nếu thành công , ngược lại sẽ trả về số tương ứng với loại lỗi (error number)

Retval : giá trị trả về từ hàm pthread\_exit.

Hãy cùng xem ví dụ dưới đây, các bạn nên copy code vào để chạy thử trước :

Message us

(<https://m.me/599911456>)

```
// thread.c
#include <stdio.h>
#include <unistd.h> // for sleep
#include <pthread.h>

// Ham xu ly cho thread 1
void *Thread1(void *threadid){
    while(1){
        printf("thread 1 : hello\n");
        sleep(3);
    }
}

// Ham xu ly cho thread 2
void *Thread2(void *threadid){
    while(1){
        printf("thread 2 : hello\n");
        sleep(4);
    }
}

int main () {
    pthread_t threads[NUM_THREADS];
    int rc, i = 0;
    // Tao thread 1
    printf("Creating thead %d, \n",i);
    if ( (rc = pthread_create(&threads[i], NULL, Thread1, NULL)) ){
        printf("Error:unable to create thread, %d\n",rc );
        return 0;
    }
    i++;
    // Tao thread 2
    printf("Creating thead %d, \n",i);
    if ( (rc = pthread_create(&threads[i], NULL, Thread2, NULL)) ){
        printf("Error:unable to create thread, %d\n",rc );
        return 0;
    }
    // Ham main van se chay nhiem vu rieng cua no
    while(1){
        printf("main thread : hello\n");
        sleep(2);
    }

    pthread_exit(NULL);
    return 0;
}
```

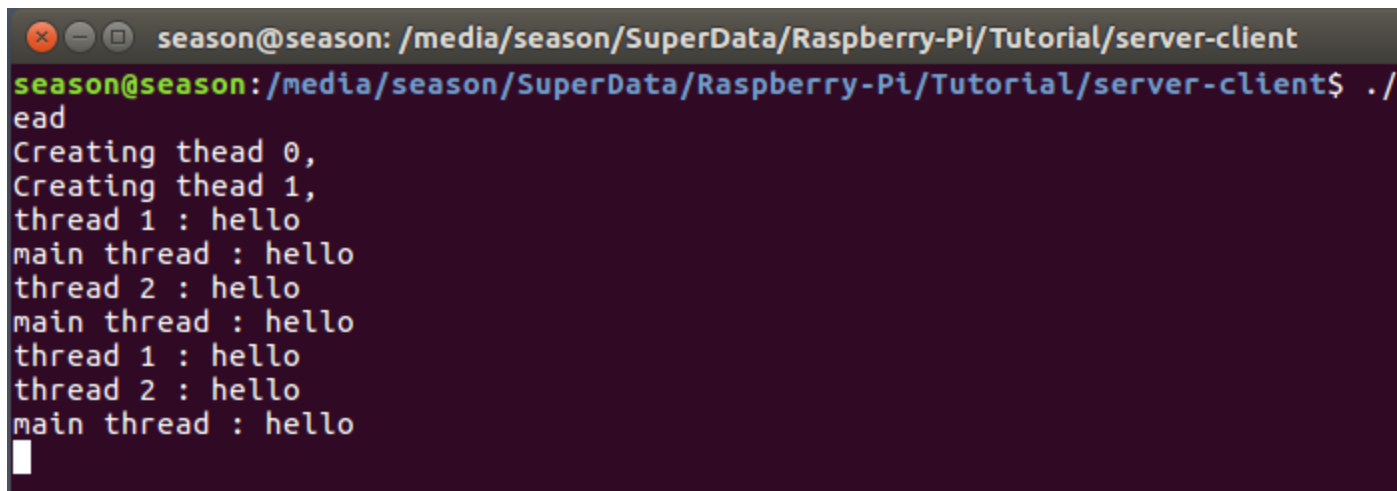
[Message us](https://m.me/59991145f)<https://m.me/59991145f>

Các bạn có thể down code trực tiếp từ github (<https://github.com/season93vn/Raspberry-Pi/tree/master/Tutorial/ser> của mình.

## Biên dịch chương trình với lệnh

```
gcc -o thread thread.c -lpthread # -lpthread khai báo dùng thư viện pthread
```

Kết quả sẽ hiển thị như sau:



```
season@season: /media/season/SuperData/Raspberry-Pi/Tutorial/server-client
season@season: /media/season/SuperData/Raspberry-Pi/Tutorial/server-client$ ./thread
Creating thread 0,
Creating thread 1,
thread 1 : hello
main thread : hello
thread 2 : hello
main thread : hello
thread 1 : hello
thread 2 : hello
main thread : hello
```

Chương trình sẽ chạy trên 3 luồng riêng biệt là main thread, thread1 và thread2. Mỗi thread sẽ gọi hàm xử lý riêng và in câu chào lên màn hình. Tất nhiên là các thread có thể gọi chung một hàm xử lý. Lưu ý rằng hàm xử lý luôn làm kiểu void.

Với ứng dụng của thread chương trình có thể hoạt động đa nhiệm không chỉ với server-client mà với bất cứ chương trình nào bạn mong muốn thực hiện đa nhiệm.

## 2. Server – multitasking

Từ ví dụ của phần trước là client trên máy tính kết nối tới Pi. Bây giờ mình sẽ tận dụng luôn từ ví dụ đó để có thể test server mới. Server mới có thể cùng một lúc nhận nhiều tin nhắn của client và hiển thị lên màn hình, server mới cũng sẽ biết là client nào gửi thông tin cho mình. Các bạn có thể sửa từ bản server.c cũ như sau :

Message us

(<https://m.me/599911456>)

```
// cau truc struct cho thread, them vao phan dau trong server.c
```

```
struct ThreadArgs{  
    int clntSock; /* Socket descriptor for client */  
};
```

```
// Ham xu ly client, them vao phan dau trong server.c
```

```
void *HandleClient(void *threadArgs){  
    int clntSock;  
    int recvMsgSize;  
    char buffer[BUFFSIZE];  
    bzero(buffer,BUFFSIZE);  
  
    clntSock = ((struct ThreadArgs *) threadArgs) -> clntSock;  
  
    while(1){  
        recvMsgSize = recv(clntSock,buffer,BUFFSIZE,0);  
        if (recvMsgSize < 0)  
            error("ERROR reading from socket");  
        else if(recvMsgSize>0){  
            printf(". Client[%d]: %s\n",clntSock,buffer);  
            bzero(buffer,strlen(buffer));  
        }  
        else{  
            printf("- Client[%d]: disconnected !\n",clntSock);  
            break;  
        }  
    }  
    close(clntSock);  
}
```

```
// Thay the tu doan accept den het ham while(1) trong server.c, nhiem vu cua phan nay la tao thread khi co ti  
while(1){
```

```
    // Wait for a client to connect
```

```
    if( (clntSock = accept(servSock, (struct sockaddr*) &cli_addr, &clntLen)) < 0)  
        error("accept() failed !");
```

```
    getpeername(clntSock, (struct sockaddr *) &cli_addr, &clntLen);
```

```
    /* Create separate memory for client argument */
```

```
    if ((threadArgs = (struct ThreadArgs *) malloc(sizeof(struct ThreadArgs))) == NULL)  
        error("malloc() failed");
```

```
    threadArgs -> clntSock = clntSock;
```

```
    if (pthread_create(&threadID, NULL, HandleClient, (void *) threadArgs) != 0)  
        error("pthread_create() failed");
```

Message us

(<https://m.me/59991145f>)

```
printf("\n+ New client[%d][Addr:%s][Port:%d]\n\n",
      clntSock, inet_ntoa(cli_addr.sin_addr), ntohs(cli_addr.sin_port));
}
```

Các bạn hãy xem kỹ code trên github (<https://github.com/season93vn/Raspberry-Pi/blob/master/Tutorial/server-client-thread.c>) nhé.

Giải thích thêm :

- + struct ThreadArgs : Được dùng để lưu thông tin cho thread
- + HandleClient() : hàm xử lý tương ứng với từng client
- + Chương trình sẽ tạo ra thread tương ứng với từng client mỗi khi có client mới kết nối tới.
- + Hàm getpeername() : dùng để lấy lấy thông tin địa chỉ của client.

Biên dịch chương trình với lệnh

```
gcc -o server-thread server-thread.c -lpthread
```

Bây giờ các bạn hãy chạy thử ./server-thread, và bật nhiều client cùng một lúc trên nhiều terminal khác nhau để kết

### 3. Chương trình client với Esp8266

Mình sẽ lấy esp8266 làm client. Các bạn xem thêm phần cài đặt thư viện và lập trình cho esp8266 trên arduino 1 (<http://mlab.vn/8281-huong-dan-lap-trinh-esp8266-evb-mlab-dung-arduino-ide.html>).

Chương trình cho esp8266 cũng sẽ thực hiện công việc giống hệt client trên máy tính. Nó sẽ gửi tin nhắn đến cho

Chương trình như sau :

Message us

(<https://m.me/59991145f>)

```
// WifiClientBasic.ino
#include <ESP8266WiFi.h>
#include <ESP8266WiFiMulti.h>

ESP8266WiFiMulti WiFiMulti;

const uint16_t port = 8888;      // port tương ứng với server
const char * host = "192.168.1.23"; // ip của raspberry

void setup() {
  Serial.begin(115200);
  delay(10);

  // Can thay ten wifi và mat khau nha ban vào
  WiFiMulti.addAP("ten wifi", "mat khau la");

  Serial.println();
  Serial.println();
  Serial.print("Wait for WiFi... ");

  while(WiFiMulti.run() != WL_CONNECTED) {
    Serial.print(".");
    delay(500);
  }

  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());

  delay(500);
}

void loop() {

  Serial.print("connecting to ");
  Serial.println(host);

  // Use WiFiClient class to create TCP connections
  WiFiClient client;

  if (!client.connect(host, port)) {
    Serial.println("connection failed");
    Serial.println("wait 5 sec...");
    delay(5000);
    return;
  }

  while(1){
```

[Message us](https://m.me/59991145f)[\(https://m.me/59991145f\)](https://m.me/59991145f)



```
if (Serial.available() > 0) {  
  String str = Serial.readString();  
  // gui cho server  
  client.print(str);  
  
  Serial.print("message: ");  
  Serial.println(str);  
}  
  
}  
  
//read back one line from server  
//String line = client.readStringUntil("\r");  
//client.println(line);  
  
//Serial.println("closing connection");  
client.stop();  
  
}
```

Các bạn phải chú ý tới những thiết lập :

+ const uint16\_t port = 8888; // thiết lập port tương ứng với server

+ const char \* host = "192.168.1.23"; // địa chỉ của Pi

+ WiFiMulti.addAP("ten wifi", "mat khau la"); // tên wifi và mật khẩu wifi

### Kết quả

## Test mô hình server-client trên Raspberry



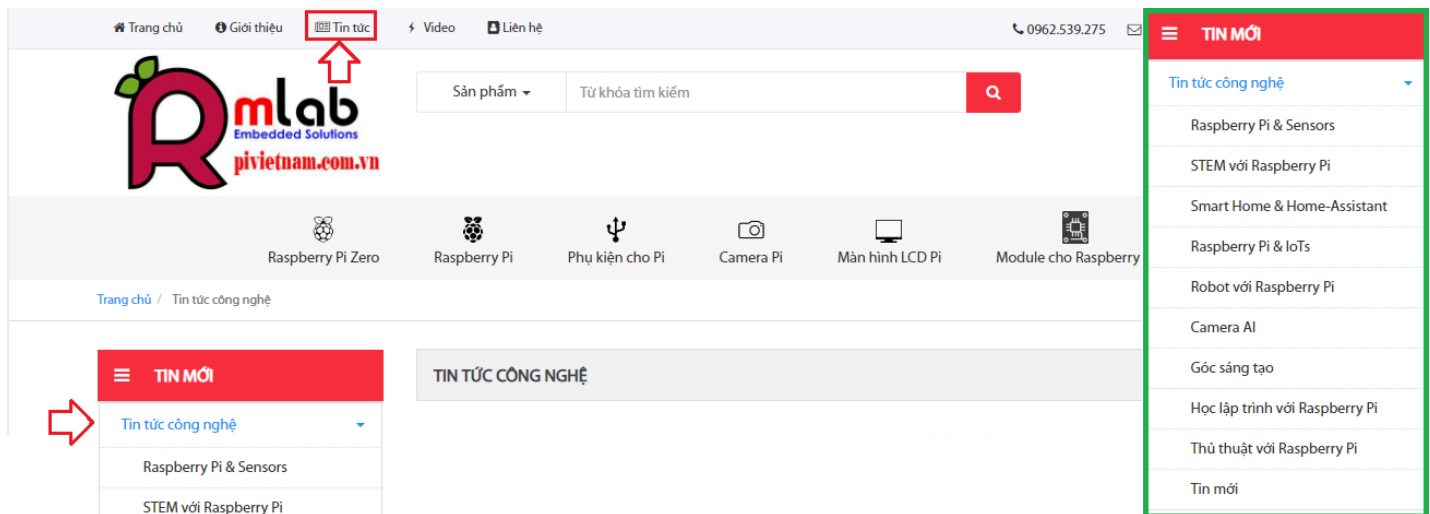
Để cập nhật các tin tức công nghệ mới các bạn làm theo hướng dẫn sau đây :

Các bạn vào Trang chủ >> Tin tức. ở mục này có các bài viết kỹ thuật thuộc các lĩnh vực khác nhau các bạn có thể lựa chọn lĩnh vực mà mình quan tâm để đọc nhé !!!

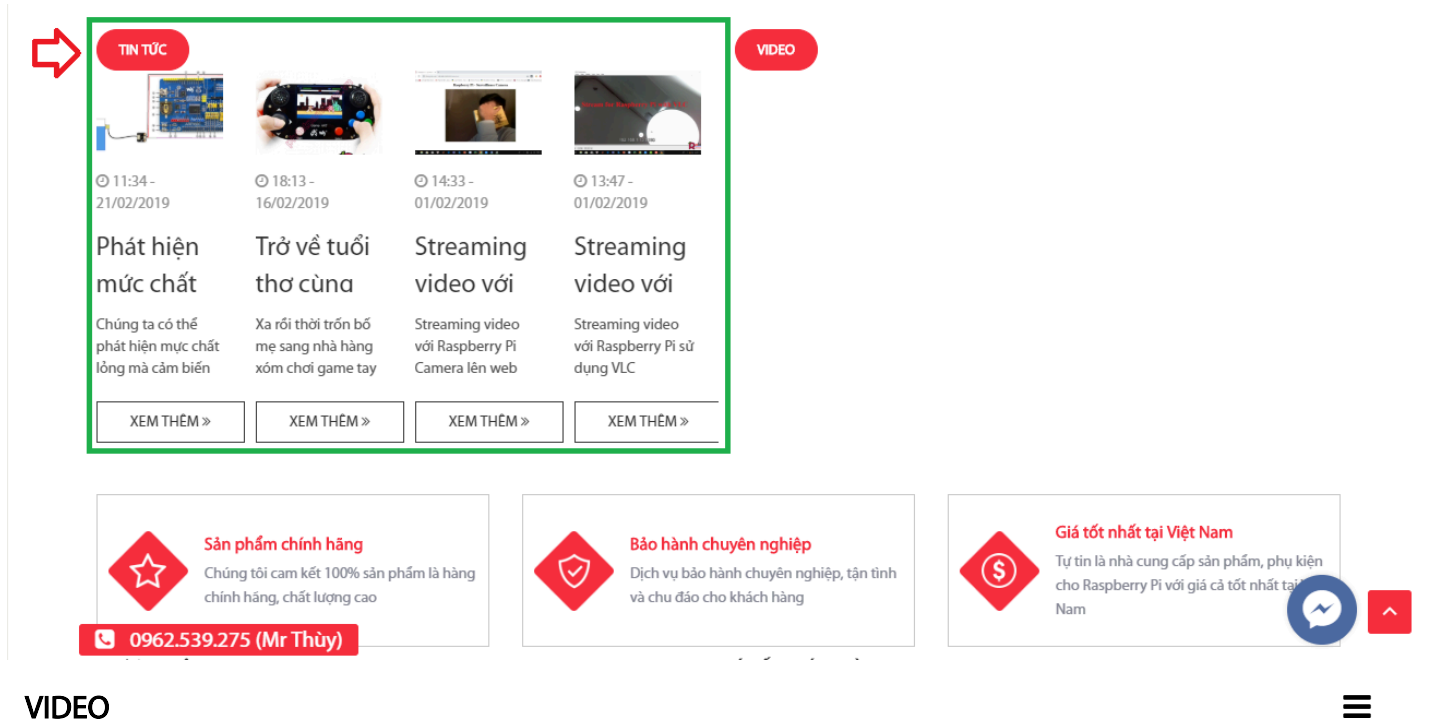
Message us

(<https://m.me/59991145f>)





Các bạn cũng có thể kéo xuống cuối trang để xem những tin tức công nghệ mới nhất.



## VỀ CHÚNG TÔI

Website uy tín cung cấp Raspberry Pi chính hãng , và các phụ kiện , board mạch mở rộng cho Raspberry Pi tại Việt Nam.

📍 Số 30F9 - Ngõ 104 Lê Thanh Nghị - Hai Bà Trưng - Hà Nội

☎ 02436.231.170

✉ smarttechvn.group@gmail.com

Message us

(https://m.me/59991145f)

## HOTLINE TƯ VẤN TRỰC TIẾP

**086.262.8846 (Mr Thùy) (tel:0962539275)**

(Thời gian làm việc 8h - 17h30, thứ 2 tới thứ 7. Hỗ trợ Online ngoài giờ hành chính và chủ nhật.)

## VỀ CHÚNG TÔI

Giới thiệu (<https://pivietnam.com.vn/ve-chung-toi>)

Lịch sử hình thành (<https://pivietnam.com.vn/lich-su-hinh-thanh>)

Đội ngũ lãnh đạo (<https://pivietnam.com.vn/doi-ngu-lanh-dao>)

Tuyển dụng (<https://pivietnam.com.vn/tuyen-dung-quy-i>)

Liên hệ (<https://pivietnam.com.vn/lien-he>)

**ĐÃ THÔNG BÁO**  
BỘ CÔNG THƯƠNG<http://online.gov.vn/Home/WebDetails/101224>

## CHÍNH SÁCH

Hướng dẫn mua hàng online (<https://pivietnam.com.vn/huong-dn-mua-hang-online-mlab-vn>)

Chính sách vận chuyển và giao nhận (<https://pivietnam.com.vn/chinh-sach-van-chuyen-va-giao-nhan-mlab-vn>)

Chính sách kiểm hàng (<https://pivietnam.com.vn/chinh-sach-kiem-hang>)

Thông tin chuyển khoản (<https://pivietnam.com.vn/thong-tin-chuyen-khoan-mlab-vn>)

Hỗ trợ sau bán hàng (<https://pivietnam.com.vn/ho-tro-sau-ban-hang-mlab-vn>)

Chính sách bảo hành (<https://pivietnam.com.vn/chinh-sach-bao-hanh-mlab-vn>)

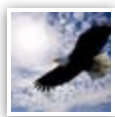
Chính sách đổi trả, hoàn tiền (<https://pivietnam.com.vn/chinh-sach-doi-tra-hoan-tien-mlab-vn>)

Chính sách bảo mật thông tin (<https://pivietnam.com.vn/chinh-sach-bao-mat-thong-tin-mlab-vn>)

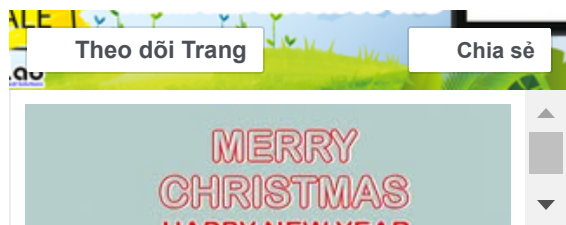
## ĐĂNG KÝ NHẬN BẢN TIN

## FACEBOOK FANPAGE

<https://m.me/59991145f>



MLAB  
5.572 người theo dõi



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 : smarttechvn.group@gmail.com

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

Website : <https://pivietnam.com.vn/>

Số điện thoại : 02436.231.170 or 086.262.8846



(tel:0962539275)

Message us

(<https://m.me/59991145f>)

