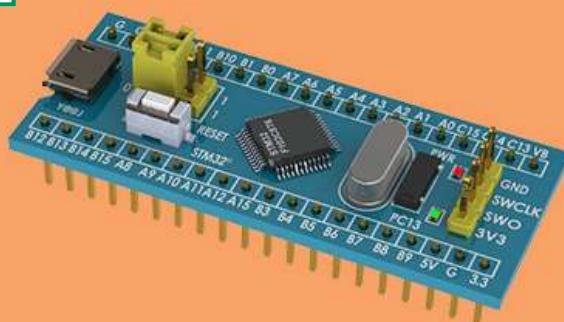
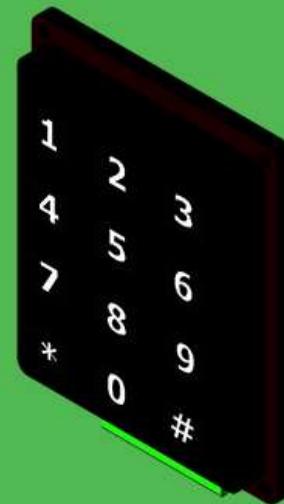


**LẬP TRÌNH STM32**

# Lập trình STM32 quét ma trận phím Keypad 3x4

POSTED ON 10/05/2021 BY KHUÊ NGUYỄN

10  
Th5**Khuê Nguyễn Creator**

## Lập trình STM32 Quét ma trận phím Keypad 3x4

Ma trận phím là một thiết bị rất hay được sử dụng trong lập trình vi điều khiển, chúng giúp cho việc tiết kiệm chân GPIO của vi điều khiển. Trong bài này chúng ta sẽ học cách quét ma trận, lý thuyết quét này cũng được sử dụng trong các loại led ma trận khác nhau.

Bài này nằm trong Serie [Học STM32 từ A tới Z](#)



via GIPHY

## Mục Lục

1. Ma trận bàn phím (Keypad) là gì?
2. Quét phím là gì? Hướng dẫn điều khiển ma trận phím
3. Cấu tạo của ma trận phím 3x4 (Keypad 3x4)
4. Lập trình STM32 quét ma trận phím 3x4
  - 4.1. Cấu hình trong CubeMx
  - 4.2. Dowload thư viện Keypad
  - 4.3. Sử dụng thư viện ma trận phím KEYPAD
  - 4.4. Sơ đồ phần cứng
  - 4.5. Giải thích Code ma trận phím
5. Kết
  - 5.1. Related posts:



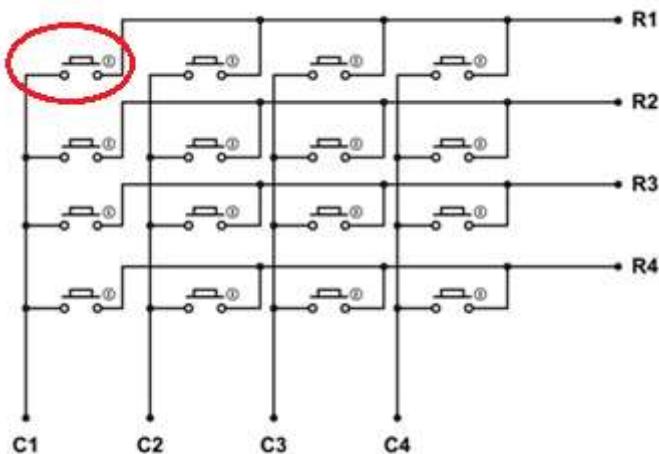
## Ma trận bàn phím (Keypad) là gì?

Ma trận phím là tập hợp các nút nhấn kết nối với nhau theo các hàng và cột. Số nút nhấn tương ứng sẽ là tích của số hàng x số cột.

Ví dụ: Ma trận phím  $4 \times 4$  sẽ có 4 cột và 4 hàng. Số lượng phím là 16 phím

Để điều khiển 16 nút nhấn, thông thường chúng ta phải sử dụng 16 GPIO nhưng nếu sử dụng ma trận phím chúng ta chỉ cần sử dụng 8 GPIO.

Các nút nhấn sẽ nối hàng và cột với nhau. Khi nhấn nút 1 dây Hàng (Row) sẽ nối với dây Cột (Column hay Col). Các nút nhấn tương ứng sẽ có vị trí nối hàng với cột khác nhau.



Ví dụ: Nút đầu tiên trong ma trận được nhấn sẽ nối Cột 1 (C1) với hàng 1(R1), nút liền kề bên phải được nhấn sẽ nối Cột 2(C2) với Hàng 1(R1). Tương tự như vậy với các nút khác.

## Quét phím là gì? Hướng dẫn điều khiển ma trận phím

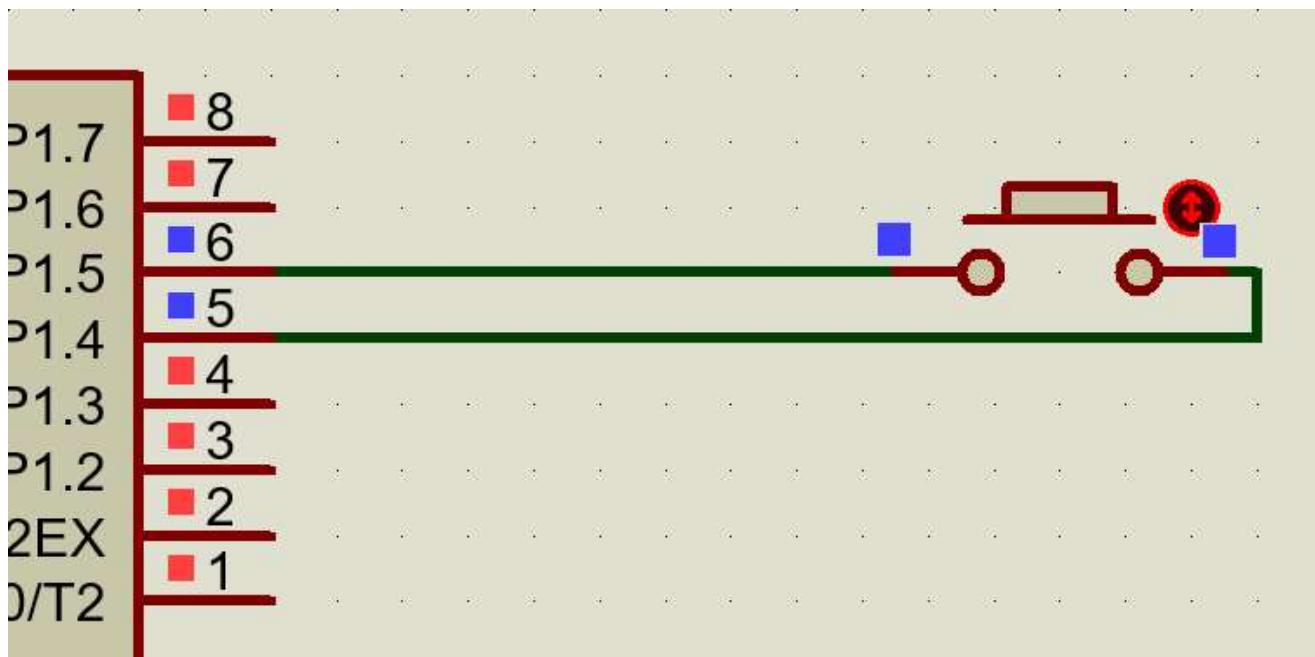
Trong bài về nút nhấn chúng ta đã được học cách đọc trạng thái của 1 chân (Read Input GPIO). Nút nhấn được nối 1 đầu vào GPIO 1 đầu vào GND hoặc VCC. Khi nhấn nút thì GND hoặc VCC được thông với nút nhấn, vậy nên khi Read sẽ cho kết quả 0 hoặc 1

Ý tưởng quét bàn phím như sau:

Thay vì nối 1 đầu nút nhấn với GND (hoặc VCC) chúng ta nối đầu đó với 1 GPIO khác. Chân GPIO đó chúng ta để là OUTPUT, vậy là chúng ta có thể điều khiển

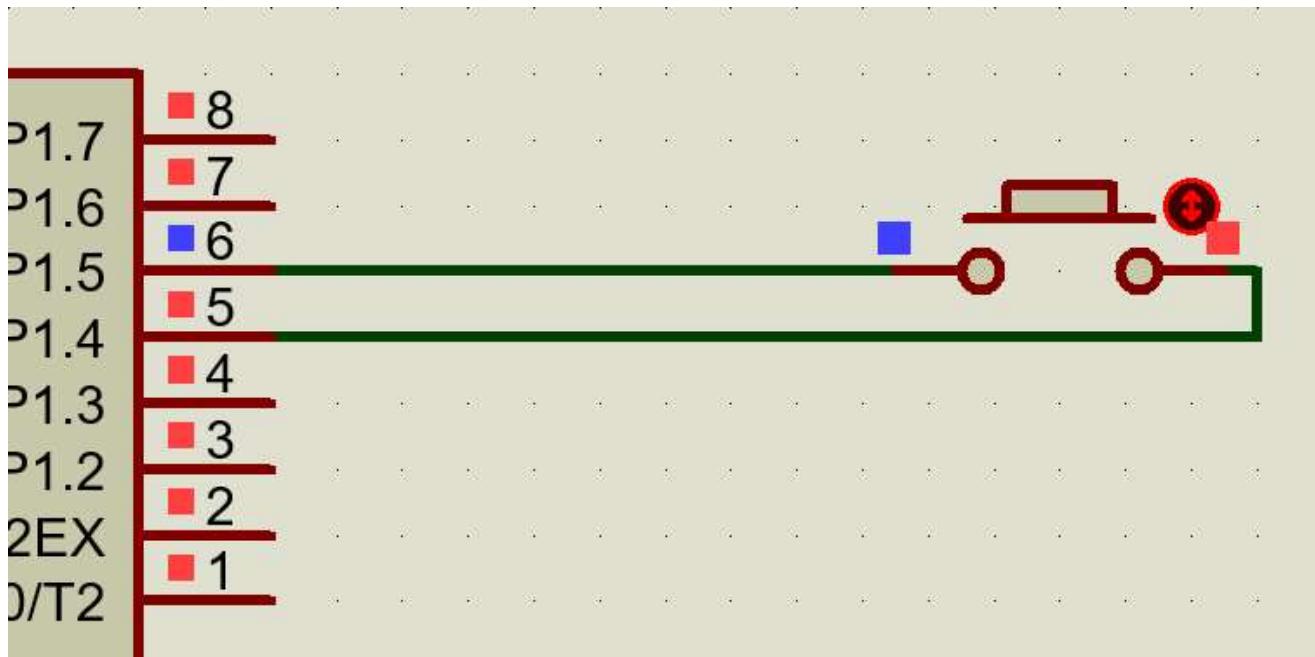
được điện áp trên đầu đó. Khi đó nếu nhấn nút GPIO Input = giá trị GPIO Ouput.

Ở trạng thái bình thường, GPIO Input được để ở chế độ pull down (khi chưa nhấn nút sẽ có giá trị 0), GPIO Output sẽ ở trạng thái 0 (Chưa active). Khi đó chúng ta ấn nút thì GPIO Input cũng sẽ vẫn là giá trị 0.

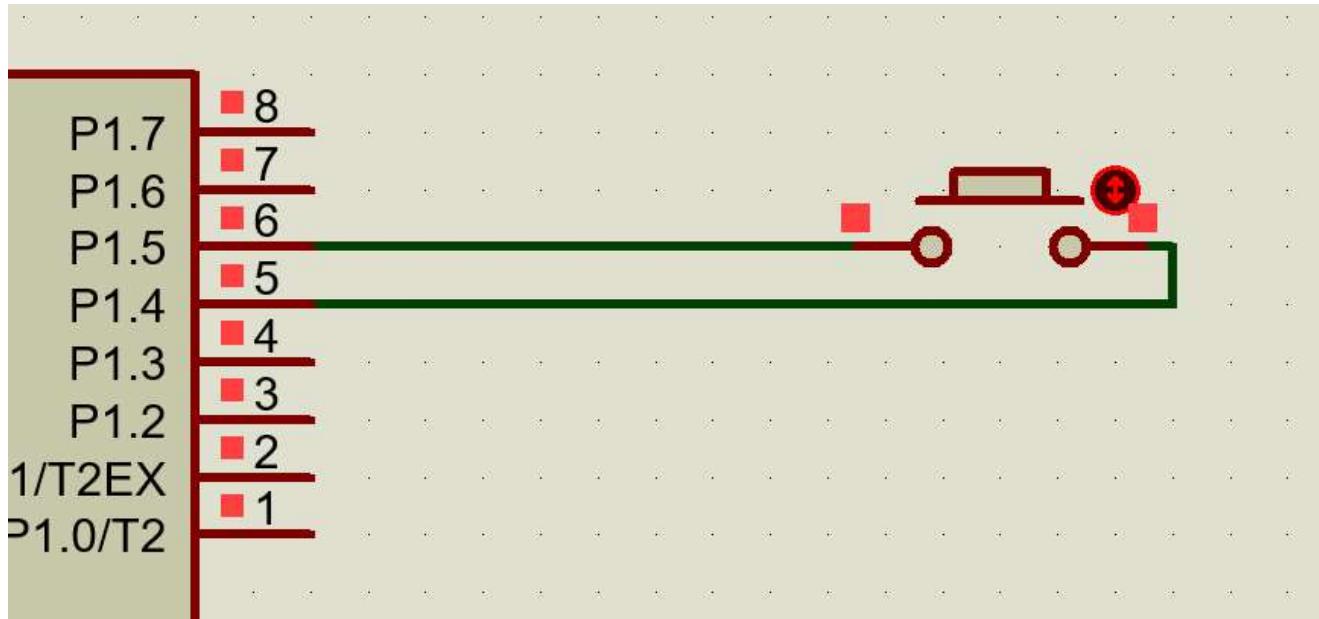


Trạng thái mặc định

Khi chúng ta đổi trạng thái GPIO Output sang 1 (Active), nhấn nút thì GPIO Input sẽ đọc được giá trị 1.



*Khi chưa nhấn nút*

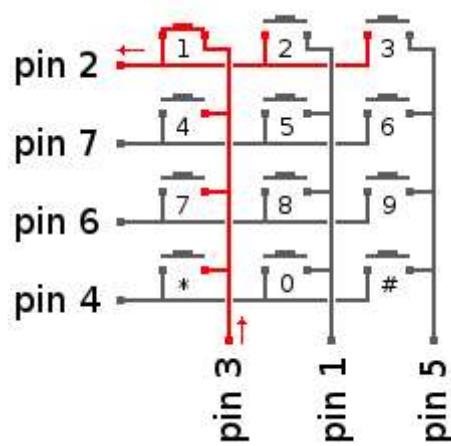


*Khi nhấn nút*

Vậy khi nối nút nhấn thành ma trận 3x4.

Quét phím sẽ diễn ra như sau: Khi quét cột 1, nếu các phím trong cột đó được nhấn, tương ứng trạng thái GPIO của các hàng sẽ thay đổi theo vị trí của nút nhấn. Các nút nhấn trong các cột khác dù được nhấn cũng không có tác dụng gì.

Khi quét đến cột 2,3. cũng tương tự như vậy. Quá trình quét phải xảy ra rất nhanh, để có thể kiểm tra được tất cả các nút trong thời gian nhấn và thả ra.



VD:

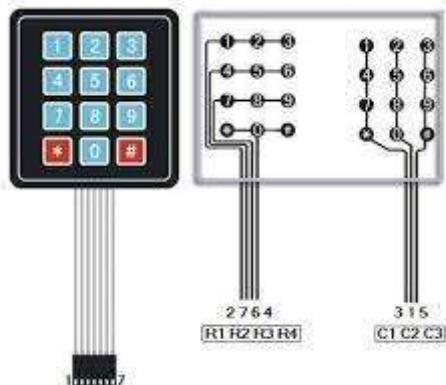
Để đọc được giá trị nhấn vi điều khiển sẽ lần lượt ghi mức 1 vào các chân 2 7 6 4 (Quét Hàng). Sau đó sẽ đọc giá trị từ 3 chân 3 1 5.

Nếu khi quét pin 2 mà pin3 có tín hiệu => nút 1 được nhấn.

Tương tự với các nút còn lại.

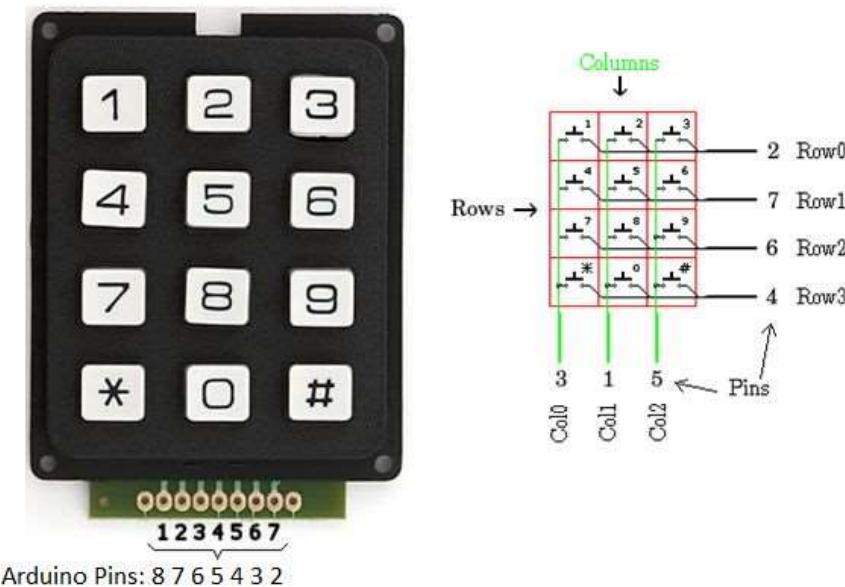
## Cấu tạo của ma trận phím 3x4 (Keypad 3x4)

Bao gồm 3 cột và 4 hàng, sắp xếp theo thứ tự sau



Keypad 3x4 Mềm

## 3x4 Keypad



*Keypad 3x4 Cứng*

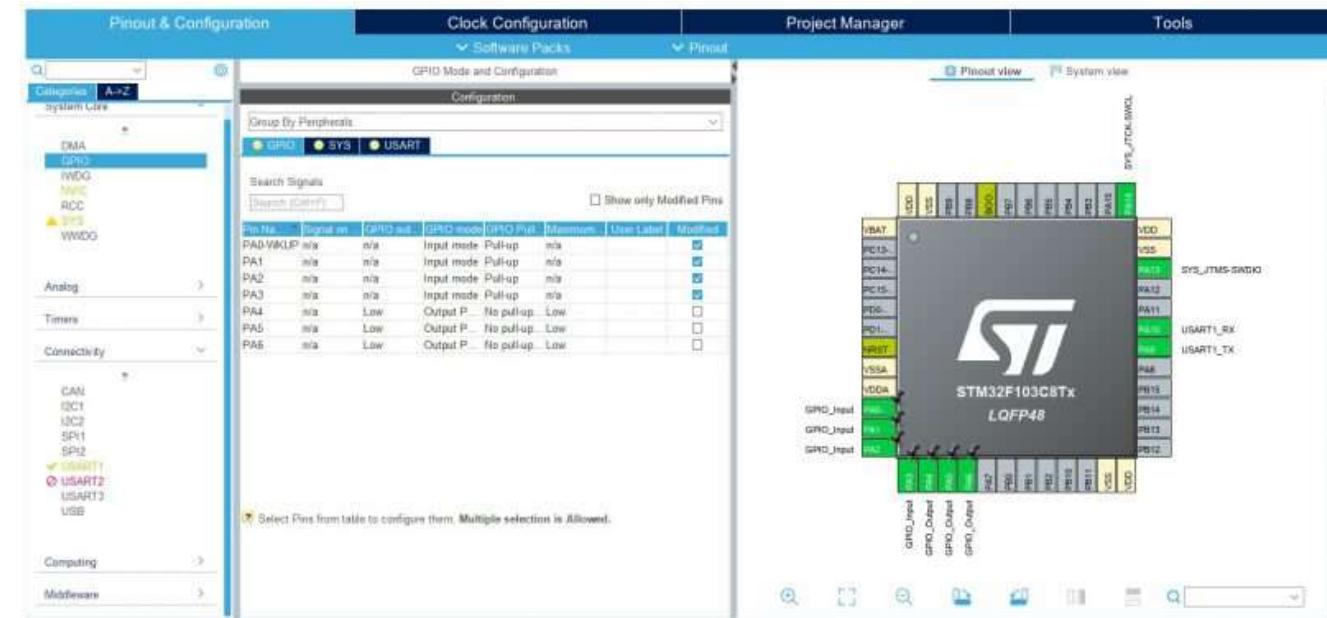
Trong bài viết này chúng ta sẽ sử dụng bàn phím cứng 3x4, Pinout tương tự như bàn phím mềm.

## Lập trình STM32 quét ma trận phím 3x4

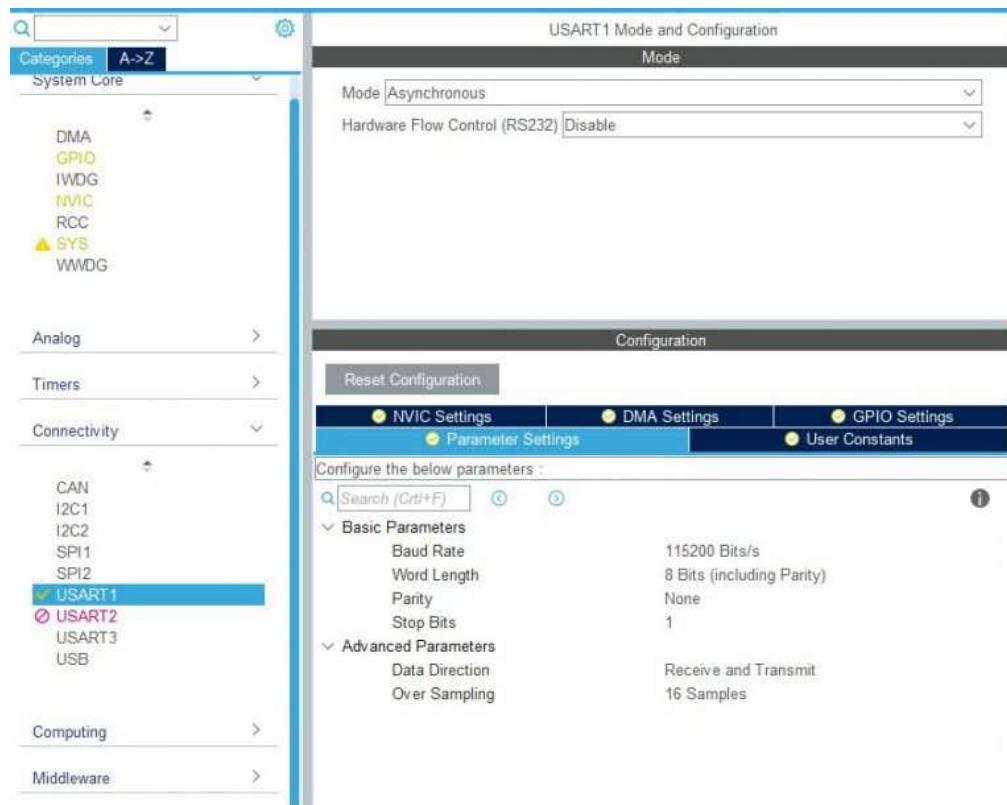
### Cấu hình trong CubeMx

Chọn chip ST32F103C8T6. Cấu hình SYS – Debug – Serial Wire. Thạch anh nội

Trong GPIO chọn PA0 – 3: Input pull up, PA4- 6: Output



## Tab Connectivty chọn USART1 Asynch



## Đặt tên và Gen Code



## Dowload thư viện Keypad

Các bạn truy cập link: [Hướng dẫn download tài liệu lập trình STM32](#)

Download thư viện Device, trong đó có file KEYPAD.c và KEYPAD.h, copy file vào thư mục Src và Inc của project.

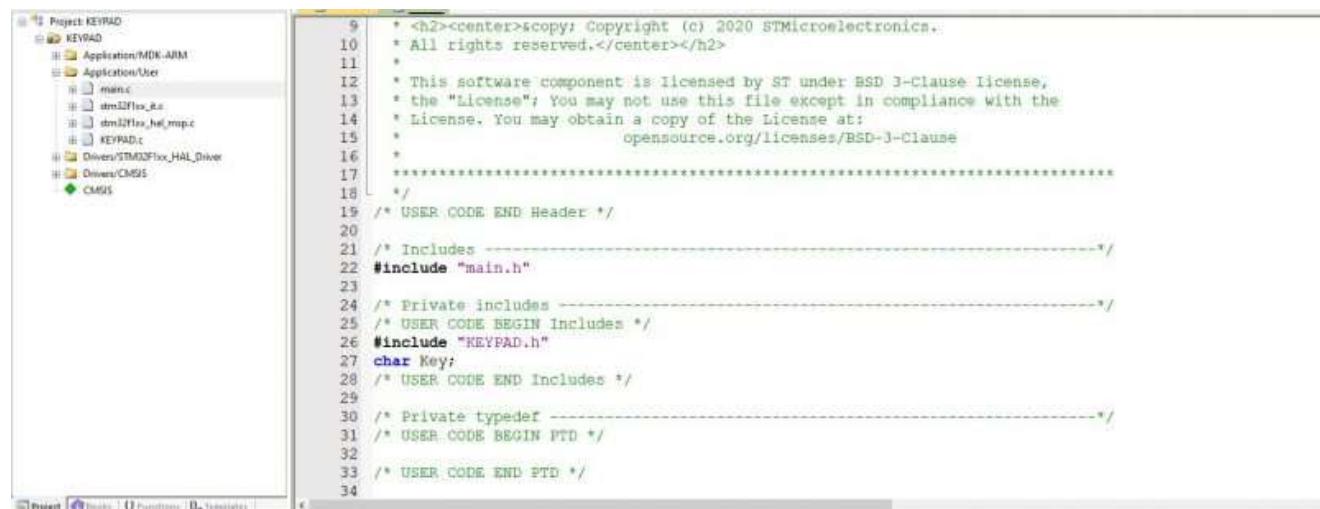
Click vào Add file, tìm tới File vừa down về. Nhấn Add để thêm file vào Project

The screenshot shows the Keil MDK-ARM IDE interface. On the left is the Project Explorer with a tree view of the project structure. The main area is a code editor with the file `KEYPAD.c` open. A context menu is displayed over the code, with the option `Show Include File Dependencies` checked. The code itself is a C program for a 3x4 keypad. It includes a header file `KEYPAD.h`, defines pinmaps for the keypad, and implements functions for initializing the keypad and reading keys.

```
20 - Su dung cac ham phai truyen dia chi cua Keypad do:  
21 Key = KEYPAD3X4_Readkey(&KeyPad);  
22  
23 Pinmap Keypad 3x4: COL1 COL2 COL3 ROW1 ROW2 ROW3 ROW4  
24 3 1 5 2 7 6 4  
*****  
25  
26 include "KEYPAD.h"  
27 static void KEYPAD_Delay(uint16_t Time)  
28  
29     HAL_Delay(Time);  
30  
31     id KEYPAD3X4_Init(KEYPAD_Name* KEYPAD, char KEYMAP [NUMROWS] [NUMCOLS])  
32         GPIO_TypeDef* COL1_PORT, uint32_t COL1_PIN,  
33         GPIO_TypeDef* COL2_PORT, uint32_t COL2_PIN,  
34         GPIO_TypeDef* COL3_PORT, uint32_t COL3_PIN,  
35         GPIO_TypeDef* ROW1_PORT, uint32_t ROW1_PIN,  
36         GPIO_TypeDef* ROW2_PORT, uint32_t ROW2_PIN,  
37         GPIO_TypeDef* ROW3_PORT, uint32_t ROW3_PIN,  
38         GPIO_TypeDef* ROW4_PORT, uint32_t ROW4_PIN)  
39     {  
40         KEYPAD->ColPort[0] = COL1_PORT; //Copy gia tri vao keypad  
41         KEYPAD->ColPort[1] = COL2_PORT;  
42         KEYPAD->ColPort[2] = COL3_PORT;  
43         KEYPAD->ColPins[0] = COL1_PIN;  
44         KEYPAD->ColPins[1] = COL2_PIN;  
45         KEYPAD->ColPins[2] = COL3_PIN;
```

## Sử dụng thư viện ma trận phím KEYPAD

Sau khi thêm các file vào, chúng ta include vào file. Tạo một biến Key để lưu trữ phím được đọc ra



The screenshot shows the Keil MDK-ARM IDE interface. The Project Explorer on the left lists the project structure:

- Project KEYPAD
- KEYPAD
- Application/MDK-ARM
- Application/User
  - main.c
  - dm32f1xx\_r.c
  - dm32f1xx\_hal\_mpu.c
  - KEYPAD.c
- Drivers/STM32F1xx\_HAL\_Driver
- Drivers/CMSIS
- CMSIS

The code editor on the right displays the following C code:

```
9 * <h2><center>&copy; Copyright (c) 2020 STMicroelectronics.
10 * All rights reserved.</center></h2>
11 *
12 * This software component is licensed by ST under BSD 3-Clause License,
13 * the "License"; You may not use this file except in compliance with the
14 * License. You may obtain a copy of the License at:
15 * opensource.org/licenses/BSD-3-Clause
16 *
17 ****
18 */
19 /* USER CODE END Header */
20
21 /* Includes */
22 #include "main.h"
23
24 /* Private includes */
25 /* USER CODE BEGIN Includes */
26 #include "KEYPAD.h"
27 char Key;
28 /* USER CODE END Includes */
29
30 /* Private typedef */
31 /* USER CODE BEGIN PTD */
32
33 /* USER CODE END PTD */
34
```

Khai báo 1 bàn phím tên là Keypad và một KeyMap ( Kí tự tương ứng với vị trí nút nhấn trên Keypad)

The screenshot shows a software development environment with a file tree on the left and a code editor on the right.

**File Tree:**

- Application/User
  - main.c
  - stm32f1xx\_it.c
  - stm32f1xx\_hal\_msp.c
  - KEYPAD.c
- Drivers/STM32F1xx\_HAL\_Driver
- Drivers/CMSIS
- CMSIS

**Code Editor (Content of main.c):**

```
47 // USER CODE BEGIN PUPPETEER_HIGHLIGHT_BLOCK
48
49 /* USER CODE BEGIN PV */
50 #include "keypad.h"
51 char KEYMAP[NUMROWS][NUMCOLS] = {
52     {'1','2','3'},
53     {'4','5','6'},
54     {'7','8','9'},
55     {'*','0','#'}
56 };
57
58 /* USER CODE END PV */
59
60 /* Private function prototypes */
61 void SystemClock_Config(void);
62 static void MX_GPIO_Init(void);
63 static void MX_I2C1_Init(void);
64 static void MX_USART1_UART_Init(void);
65 /* USER CODE BEGIN PFP */
66
67 /* USER CODE END PFP */
68
69 // Includes user code
70
```

Trước While (1) khởi tạo các Keypad, truyền các tham số KEYMAP, các cột và hàng vào.

```
Project: KEYPAD
KEYPAD
  Application/MDK-ARM
  Application/User
    main.c
    #include "stm32f10x_it.h"
    #include "stm32f10x_hal.h"
    #include "KEYPAD.h"
  Drivers/STM32F1xx_HAL_Driver
  Drivers/CMSIS
  CMSIS

102  MX_I2C1_Init();
103  MX_USART1_UART_Init();
104  /* USER CODE BEGIN 2 */
105  KEYPAD3X4_Init(&KeyPad, KEYMAP, GPIOA, GPIO_PIN_4, GPIOA, GPIO_PIN_5, GPIOA, GPIO_PIN_6,
106                           GPIOA, GPIO_PIN_0, GPIOA, GPIO_PIN_1,
107                           GPIOA, GPIO_PIN_2, GPIOA, GPIO_PIN_3);
108
109  /* USER CODE END 2 */
110
111  /* Infinite loop */
112  /* USER CODE BEGIN WHILE */
113  while (1)
114  {
115      /* USER CODE END WHILE */
116
117      /* USER CODE BEGIN 3 */
118      Key = KEYPAD3X4_Readkey(&KeyPad);
119      if(Key)
120      {
121          HAL_UART_Transmit(&huart1, (uint8_t*)&Key, 1, 100);
122      }
123      HAL_Delay(50);
124
125  /* USER CODE END 3 */
126
127 }
```

Trong While Sử dụng Readkey để đọc phím, sau đó in ra UART1 và delay 1  
khoảng thời gian nhỏ.

Nhấn F7 để Build sau đó nạp code.

## Sơ đồ phần cứng

3 Cột tương ứng với chân 3,1,5 trên KEYPAD nối với PA4,5,6

4 Hàng tương ứng với chân 2,7,6,4 trên KEYPAD nối với PA0,1,2,3

## Giải thích Code ma trận phím

Hàm Init khởi tạo các giá trị như KEYMAP, Vị trí các chân Cột và hàng, được copy vào struct Keypad đã khai báo.

```

31 void KEYPAD3X4_Init(KEYPAD_Name* KEYPAD, char KEYMAP[NUMROWS][NUMCOLS],
32                         GPIO_TypeDef* COL1_PORT, uint32_t COL1_PIN,
33                         GPIO_TypeDef* COL2_PORT, uint32_t COL2_PIN,
34                         GPIO_TypeDef* COL3_PORT, uint32_t COL3_PIN,
35                         GPIO_TypeDef* ROW1_PORT, uint32_t ROW1_PIN,
36                         GPIO_TypeDef* ROW2_PORT, uint32_t ROW2_PIN,
37                         GPIO_TypeDef* ROW3_PORT, uint32_t ROW3_PIN,
38                         GPIO_TypeDef* ROW4_PORT, uint32_t ROW4_PIN)
39 {
40     KEYPAD->ColPort[0] = COL1_PORT; //Copy giá trị vào keypad
41     KEYPAD->ColPort[1] = COL2_PORT;
42     KEYPAD->ColPort[2] = COL3_PORT;
43     KEYPAD->ColPins[0] = COL1_PIN;
44     KEYPAD->ColPins[1] = COL2_PIN;
45     KEYPAD->ColPins[2] = COL3_PIN;
46
47     KEYPAD->RowPort[0] = ROW1_PORT;
48     KEYPAD->RowPort[1] = ROW2_PORT;
49     KEYPAD->RowPort[2] = ROW3_PORT;
50     KEYPAD->RowPort[3] = ROW4_PORT;
51     KEYPAD->RowPins[0] = ROW1_PIN;
52     KEYPAD->RowPins[1] = ROW2_PIN;
53     KEYPAD->RowPins[2] = ROW3_PIN;
54     KEYPAD->RowPins[3] = ROW4_PIN;
55
56     for(int colum = 0; colum < NUMCOLS; colum++)
57     {
58         for(int row = 0; row < NUMROWS; row++)
59         {
60             KEYPAD->MAP[row][colum] = KEYMAP[row][colum];
61         }
62     }
63 }
```

Hàm Readkey, tạo ra vòng lặp. Trong vòng lặp đó ta ghi giá trị vào các cột, sau đó đọc lần lượt các hàng.

```

68 char KEYPAD3X4_Readkey(KEYPAD_Name* KEYPAD) // Scan Columns
69 {
70     KEYPAD->Value = 0;
71     for(int colum = 0; colum < NUMCOLS; colum++)
72     {
73         HAL_GPIO_WritePin(KEYPAD->ColPort[colum],KEYPAD->ColPins[colum],GPIO_PIN_RESET);
74         for(int row = 0; row < NUMROWS; row++)
75         {
76             if(HAL_GPIO_ReadPin(KEYPAD->RowPort[row],KEYPAD->RowPins[row]) == 0)
77             {
78                 KEYPAD_Delay(50); // debound
79                 while(HAL_GPIO_ReadPin(KEYPAD->RowPort[row],KEYPAD->RowPins[row]) == 0) {}
80                 KEYPAD->Value = KEYPAD->MAP[row][colum];
81
82             return KEYPAD->Value;
83         }
84         HAL_GPIO_WritePin(KEYPAD->ColPort[colum],KEYPAD->ColPins[colum],GPIO_PIN_SET);
85     }
86
87     return 0;
88 }
```

Khi thấy sự nút đó được đọc, ta sẽ return giá trị tương ứng trong KEYMAP

```

91 void KEYPAD3x4_Config(KEYPAD_Name* KEYPAD, char KEYMAP_Config[NUMROWS][NUMCOLS])
92 {
93     for(int colum = 0; colum < NUMCOLS; colum++)
94     {
95         for(int row = 0; row < NUMROWS; row++)
96         {
97             KEYPAD->MAP[row][colum] = KEYMAP_Config[row][colum];
98         }
99     }
100 }
```

Hàm Config sẽ thay đổi KEYMAP, giúp việc chuyển đổi giá trị các nút nhấn trong quá trình code.

## Kết

Phương pháp quét Ma trận phím là một phương pháp rất hay khi sử dụng GPIO, nó cũng tương tự khi quét các loại ma trận led hay các phương pháp quét khác. Các bạn có thể tự code theo lý thuyết mình đã nêu ra để hiểu rõ hơn về vấn đề này nhé.

Đừng quên tham gia nhóm [Nghiện lập trình](#), để có thể kết nối với những anh em khác nhé!

5/5 - (3 bình chọn)

### Related Posts:

1. [Lập trình STM32 với Oled LCD SSD1306](#)
2. [Hướng dẫn download và sử dụng tài liệu Lập trình STM32](#)
3. [Bài 14: Sử dụng STM32 IWDG Independent Watchdog Timer chống treo vi điều khiển](#)
4. [Bài 10: Giao thức I2C, lập trình STM32 với module RTC DS3231](#)
5. [Bài 3: Lập trình STM32 GPIO điều khiển Led và nút nhấn](#)
6. [Bài 2: Tổng quan về KIT STM32F103C8T6 Blue Pill](#)





### KHUÊ NGUYỄN

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

## 7 THOUGHTS ON “LẬP TRÌNH STM32 QUÉT MA TRẬN PHÍM KEYPAD 3x4”



*Duy says:*

Cho mình hỏi đoạn copy giá trị vào keypad là sao vậy, mình chưa hiểu là giá trị gán cho ColPort[0] là ở đâu mà có

01/07/2021 AT 9:37 CHIỀU

TRẢ LỜI



*Khuê Nguyễn says:*

Phần đấy được định nghĩa trong Struct Keypad ở file keypad.h bạn nhé.

01/07/2021 AT 10:30 CHIỀU

TRẢ LỜI



*Bao.nguyen says:*

anh ơi cho em hỏi tại sao em làm theo hướng dẫn thì báo lỗi , mặc dù em đã add đúng , đủ thư viện. Em cảm ơn

part1\part1.axf: Error: L6218E: Undefined symbol KEYPAD3X4\_Readkey  
(referred from main.o).

part1\part1.axf: Error: L6218E: Undefined symbol KEYPAD3X4\_Init (referred from main.o).

25/07/2021 AT 10:04 SÁNG

TRẢ LỜI

*Khuê Nguyễn says:*



em chinh duong dan thu vien chua?

25/07/2021 AT 12:11 CHIỀU

TRẢ LỜI



*Chiến* says:

Anh ơi code em chạy được nhưng khi bấm phím 4 nó cứ hiện 2 số 14 thì nó bị lỗi gì vậy ạ

13/08/2021 AT 9:36 CHIỀU

TRẢ LỜI



*Khuê Nguyễn* says:

xem dây nối vào chân số 1 có chập với số 4 ko

14/08/2021 AT 11:34 SÁNG

TRẢ LỜI



*tú* says:

anh ơi em dùng keypad kết hợp i2c lcd 1602 thì khi ấn nó bị loạn ký tự, cả khi mình in thẳng ký tự ra cũng bị sai k có nghĩa.anh chỉ cách fix giúp em với, em cảm ơn anh

15/08/2021 AT 10:23 CHIỀU

TRẢ LỜI

## Trả lời

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

Bình luận \*

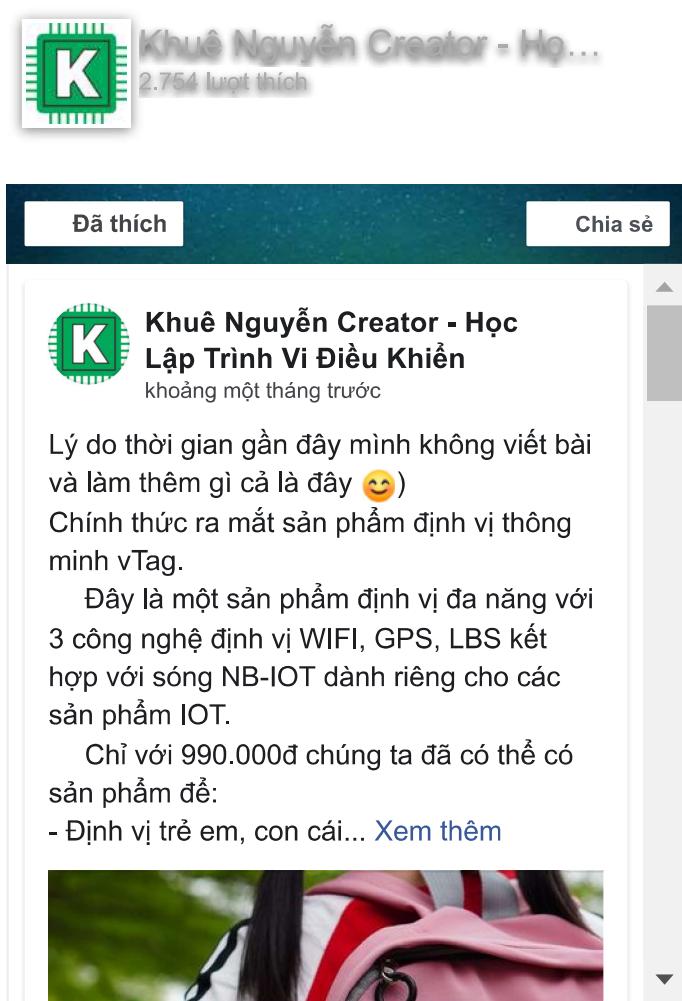
Tên \*

Email \*

Trang web

PHẢN HỒI

Fanpage



Khuê Nguyễn Creator - Học Lập Trình Vi Điều Khiển  
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



### 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 chung 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**

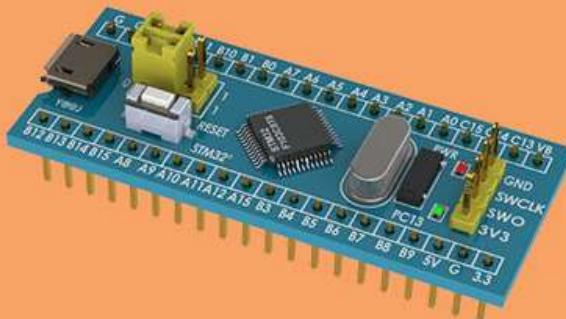
Trong bài này chúng ta sẽ cùng nhau tìm hiểu STM32 DAC với KIT STM32F407VE...

[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