

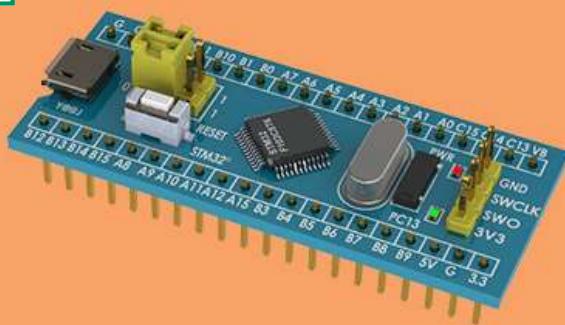
LẬP TRÌNH STM32

Bài 18: Lập trình STM32 USB HID Keyboard bàn phím máy tính

POSTED ON 11/05/2021 BY KHUÊ NGUYỄN

11
Th5

Lập trình STM32 và CubeMX

**Khuê Nguyễn Creator**

Bài 18: STM32 USB HID tự làm bàn phím máy tính

Ở bài trước chúng ta đã làm quen với USB HID để tạo 1 con chuột máy tính, ở bài này chúng ta sẽ sử dụng STM32 USB HID Keyboard để chế tạo 1 chiếc bàn phím đơn giản, giao tiếp với máy tính nhé.

Bài 18 trong Serie **Lập trình STM32 từ A tới Z**



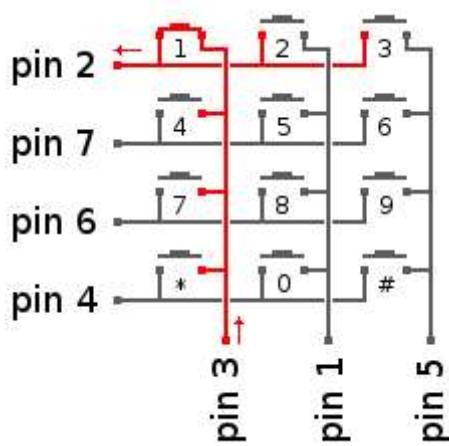
Mục Lục

1. Bàn phím máy tính hoạt động như thế nào?
 - 1.1. Lý thuyết quét phím ma trận
 - 1.2. Bàn phím thực tế
2. Report Descriptor USB HID của bàn phím máy tính
 - 2.1. Phân tích Report Descriptor Keyboard
 - 2.2. Cấu trúc dữ liệu khi gửi của HID KeyBoard
3. Lập trình STM32 USB HID Keyboard bàn phím máy tính
 - 3.1. Cấu hình STM32 USB HID Keyboard trên CubeMX
 - 3.2. Lập trình STM32 USB HID Keyboard trên Keil C
 - 3.3. Kết nối phần cứng
 - 3.4. Kết quả
4. Kết
 - 4.1. Related posts:

Bàn phím máy tính hoạt động như thế nào?

Lý thuyết quét phím ma trận

Về mặt phần cứng Bàn phím (Keyboard) là tập hợp các nút nhấn được thiết kế theo dạng ma trận phím. Nghĩa là chúng sẽ được nối chung hàng và cột. Vì điều khiển sẽ quét hàng hoặc cột theo một chu kỳ cố định để tìm kiếm sự thay đổi về trạng thái nút. Sau đó gửi lên máy tính thông qua chuẩn USB HID



quét phím ma trận 4x3

Ví dụ: Trong hình trên chúng ta có 1 ma trận phím 4×3 .

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 pin 3 có tín hiệu \Rightarrow nút 1 được nhấn.

Tương tự với các nút còn lại.

Khi vi điều khiển biết được nút nào được nhấn. Nó sẽ gửi 1 gói tin sang vi điều khiển tương ứng với nút đó trên bàn phím, để máy tính hiểu rằng, phím đã được nhấn.

Để hiểu rõ hơn về bàn phím các bạn đọc bài viết: [Lập trình STM32 giao tiếp ma trận bàn phím \$3 \times 4\$](#)

Bàn phím thực tế

Trong thực tế bàn phím máy tính có rất nhiều nút nhấn. Bao gồm 2 loại chính:

- Các nút ký tự: A-Z, số, các ký tự đặc biệt ...
- Các nút lệnh: Shift, Alt,

Tất cả các phím được sắp xếp theo tiêu chuẩn nhất định. Mặc dù các loại bàn phím khác nhau sẽ có thêm hoặc bớt các phím, thế nhưng cấu trúc của bàn phím hầu như không đổi



Report Descriptor USB HID của bàn phím máy tính

Phân tích Report Descriptor Keyboard

Để máy tính hiểu được dữ liệu chúng ta gửi lên thì Keyboard sẽ phải gửi 1 Report Descriptor, sau đó dữ liệu gửi lên phải đúng format của RD đó. Bạn có thể google search: **Report Descriptor Keyboard** chúng ta sẽ có ngay kết quả

Sử dụng công cụ kiểm tra Report Descriptor ở bài trước phân tích như sau:

```

0x05, 0x01,          // Usage Page (Generic Desktop Ctrl)
0x09, 0x06,          // Usage (Keyboard)
0xA1, 0x01,          // Collection (Application)
0x05, 0x07,          //   Usage Page (Kbrd/Keypad)
0x19, 0xE0,          //   Usage Minimum (0xE0)
0x29, 0xE7,          //   Usage Maximum (0xE7)
0x15, 0x00,          //   Logical Minimum (0)
0x25, 0x01,          //   Logical Maximum (1)
0x75, 0x01,          //   Report Size (1)
0x95, 0x08,          //   Report Count (8)

```

```

0x81, 0x02,          // Input (Data,Var,Abs,No Wrap,Linear,Preferred State,No Nu
0x95, 0x01,          // Report Count (1)
0x75, 0x08,          // Report Size (8)
0x81, 0x03,          // Input (Const,Var,Abs,No Wrap,Linear,Preferred State,No N
0x95, 0x05,          // Report Count (5)
0x75, 0x01,          // Report Size (1)
0x05, 0x08,          // Usage Page (LEDs)
0x19, 0x01,          // Usage Minimum (Num Lock)
0x29, 0x05,          // Usage Maximum (Kana)
0x91, 0x02,          // Output (Data,Var,Abs,No Wrap,Linear,Preferred State,No N
0x95, 0x01,          // Report Count (1)
0x75, 0x03,          // Report Size (3)
0x91, 0x03,          // Output (Const,Var,Abs,No Wrap,Linear,Preferred State,No
0x95, 0x06,          // Report Count (6)
0x75, 0x08,          // Report Size (8)
0x15, 0x00,          // Logical Minimum (0)
0x25, 0x65,          // Logical Maximum (101)
0x05, 0x07,          // Usage Page (Kbrd/Keypad)
0x19, 0x00,          // Usage Minimum (0x00)
0x29, 0x65,          // Usage Maximum (0x65)
0x81, 0x00,          // Input (Data,Array,Abs,No Wrap,Linear,Preferred State,No
0xC0,                // End Collection

// 63 bytes

```

Như vậy khi 1 phím được nhấn, bàn phím sẽ phải gửi 63 byte (với chuột máy tính là 74 byte) để báo cho máy tính biết rằng nút nào được nhấn. Các data được gửi qua lại USB HID với Input sẽ là các nút được nhấn và Output sẽ dùng để điều khiển các đèn Led trên bàn phím (Numlock, Capslock ...)

Cấu trúc dữ liệu khi gửi của HID KeyBoard

Nhìn vào Report Descriptor của Keyboard chúng ta cũng dễ dàng phân tích được dữ liệu sẽ truyền lên máy tính như thế nào.

Khi Keyboard truyền lên máy tính thì sẽ gồm tất cả Input trong Usage Page (Kbrd/Keypad)

Tổng sẽ là 1 Byte + 1 Byte + 6 Byte.

Trong đó:

- Byte đầu tiên sẽ là **Modifier keys status** có giá trị từ 0xE0 – 0xE7: Chứa các giá trị của các phím lệnh
- Byte thứ 2 là Byte đặc tả của HID, chúng ta ko cần quan tâm
- 6 Byte còn lại là **Keypress fields** có giá trị từ 0x00 – 0x65 hay từ 0 – 101 theo hệ thập phân: Chứa các giá trị của phím kí tự

Report format

This report must be requested by the software using interrupt transfers once every interval milliseconds, and the interval is defined in the interrupt IN descriptor of the USB keyboard. The USB keyboard report may be up to 8 bytes in size, although not all these bytes are used and it's OK to implement a proper implementation using only the first three or four bytes (and this is how I do it.) Just for completion's sake, however, I will describe the full report mechanism of the keyboard. Notice that the report structure defined below applies to the boot protocol only.

Offset	Size	Description
0	Byte	Modifier keys status.
1	Byte	Reserved field.
2	Byte	Keypress #1.
3	Byte	Keypress #2.
4	Byte	Keypress #3.
5	Byte	Keypress #4.
6	Byte	Keypress #5.
7	Byte	Keypress #6.

Modifier keys status: This byte is a bitfield, where each bit corresponds to a specific modifier key. When a bit is set to 1, the corresponding modifier key is being pressed. Unlike PS/2 keyboards, USB keyboards don't have "scancodes" for modifier keys. The bit structure of this byte is:

Bit	Bit Length	Description
0	1	Left Ctrl.
1	1	Left Shift.
2	1	Left Alt.
3	1	Left GUI (Windows/Super key)
4	1	Right Ctrl.
5	1	Right Shift.
6	1	Right Alt.
7	1	Right GUI (Windows/Super key)

When software receives an interrupt and, for example, one of the Shift modifier keys are set to 1, software should use the scancode table for the shift modification to get the key from the scancode.

Reserved field: This byte is reserved by the USB HID specification, and thus software should ignore it.

Keypress fields: One keyboard report can indicate up to 6 keypresses. All these values are unsigned 8-bit values (unlike PS/2 scancodes, which are mostly 7-bit) which indicate the key being pressed. A reference on the USB scancode to ASCII character conversion table is in the bottom of the article.

Chi tiết tham khảo tài liệu về USB HID Keyboard:

https://wiki.osdev.org/USB_Human_Interface_Devices

Vậy làm thế nào chúng ta biết được mỗi phím tương ứng với giá trị nào. Hãy cùng tham khảo bảng sau:

Với trường **Modifier keys status** các giá trị từ E0-E7 tương ứng với nút nào được nhấn:

60 Universal Serial Bus HID Usage Tables

Usage ID (Dec)	Usage ID (Hex)	Usage Name	Ref: Typical AT-101 Position	PC-Mac UNI Boot		
				AT	X	
221	DD	Keypad Hexadecimal				
222-223	DE-DF	Reserved				
224	E0	Keyboard LeftControl	58	✓	✓	✓ 4/101/104
225	E1	Keyboard LeftShift	44	✓	✓	✓ 4/101/104
226	E2	Keyboard LeftAlt	60	✓	✓	✓ 4/101/104
227	E3	Keyboard Left GUI10:23	127	✓	✓	✓ 104
228	E4	Keyboard RightControl	64	✓	✓	✓ 101/104
229	E5	Keyboard RightShift	57	✓	✓	✓ 4/101/104
230	E6	Keyboard RightAlt	62	✓	✓	✓ 101/104
231	E7	Keyboard Right GUI10:24	128	✓	✓	✓ 104
232-65535	E8-FFFF	Reserved				

Footnotes 1-15, 20-34

Với trường **Keypress fields** giá trị từ 0x00 – 0x65 tương ứng với kí tự nào được nhấn

Table 12: Keyboard/Keypad Page

Usage ID (Dec)	Usage ID (Hex)	Usage Name	Ref: Typical AT-101 Position	PC- Mac UNI Boot		
				AT	X	
0	00	Reserved (no event indicated) ⁹	N/A	✓	✓	✓ 4/101/104
1	01	Keyboard ErrorRollOver ⁹	N/A	✓	✓	✓ 4/101/104
2	02	Keyboard POSTFail ⁹	N/A	✓	✓	✓ 4/101/104
3	03	Keyboard ErrorUndefined ⁹	N/A	✓	✓	✓ 4/101/104
4	04	Keyboard a and A ⁴	31	✓	✓	✓ 4/101/104
5	05	Keyboard b and B	50	✓	✓	✓ 4/101/104
6	06	Keyboard c and C ⁴	48	✓	✓	✓ 4/101/104
7	07	Keyboard d and D	33	✓	✓	✓ 4/101/104
8	08	Keyboard e and E	19	✓	✓	✓ 4/101/104
9	09	Keyboard f and F	34	✓	✓	✓ 4/101/104
10	0A	Keyboard g and G	35	✓	✓	✓ 4/101/104
11	0B	Keyboard h and H	36	✓	✓	✓ 4/101/104
12	0C	Keyboard i and I	24	✓	✓	✓ 4/101/104
13	0D	Keyboard j and J	37	✓	✓	✓ 4/101/104
14	0E	Keyboard k and K	38	✓	✓	✓ 4/101/104
15	0F	Keyboard l and L	39	✓	✓	✓ 4/101/104
16	10	Keyboard m and M ⁴	52	✓	✓	✓ 4/101/104
17	11	Keyboard n and N	51	✓	✓	✓ 4/101/104
18	12	Keyboard o and O ⁴	25	✓	✓	✓ 4/101/104
19	13	Keyboard p and P ⁴	26	✓	✓	✓ 4/101/104
20	14	Keyboard q and Q ⁴	17	✓	✓	✓ 4/101/104

Chi tiết tham khảo tài liệu Tài liệu bảng mã kí tự trên USB HID Keyboard:

https://d1.amobbs.com/bbs_upload782111/files_47/ourdev_692986N5FAHU.pdf

Lập trình STM32 USB HID Keyboard bàn phím máy tính

Cấu hình STM32 USB HID Keyboard trên CubeMX

Cấu hình SYS – Debug -Serial Wire. RCC – HSE – Crystal.

Cấu hình GPIO Input – Pull Up. Cấu hình tương tự [Bài STM32 với Keyboard](#)

Pinout & Configuration

Clock Configuration

Project Manager

Tools

GPIO Mode and Configuration

Configuration

Search Signals

Show only Modified Pins

Pin Name	Signal on Pin	GPIO output	GPIO mode	GPIO Pull-up	Maximum v.	User Label	Modified
PA3	n/a	n/a	Input mode	Pull-up	n/a		<input checked="" type="checkbox"/>
PA1	n/a	n/a	Input mode	Pull-up	n/a		<input checked="" type="checkbox"/>
PA2	n/a	n/a	Input mode	Pull-up	n/a		<input checked="" type="checkbox"/>
PA3	n/a	n/a	Input mode	Pull-up	n/a		<input checked="" type="checkbox"/>
PA4	n/a	Low	Output Pus.	No pull-up or	Low		<input type="checkbox"/>
PA5	n/a	Low	Output Pus.	No pull-up or	Low		<input type="checkbox"/>
PA6	n/a	Low	Output Pus.	No pull-up or	Low		<input type="checkbox"/>

PA3 Configuration

GPIO mode: Input mode

GPIO Pull-up/Pull-down: Pull-up

User Label:

STM32F103CBTx LQFP48

Pinout view System view

Cấu hình GPIO

Pinout & Configuration

Clock Configuration

Project

USB Mode and Configuration

Mode

Device (FS)

Configuration

Reset Configuration

Parameter Settings User Constants NVIC Settings GPIO Settings

Configure the below parameters :

Search (Ctrl+F)

Basic Parameters

Speed: Full Speed 12MBit/s

Power Parameters

Low Power: Disabled

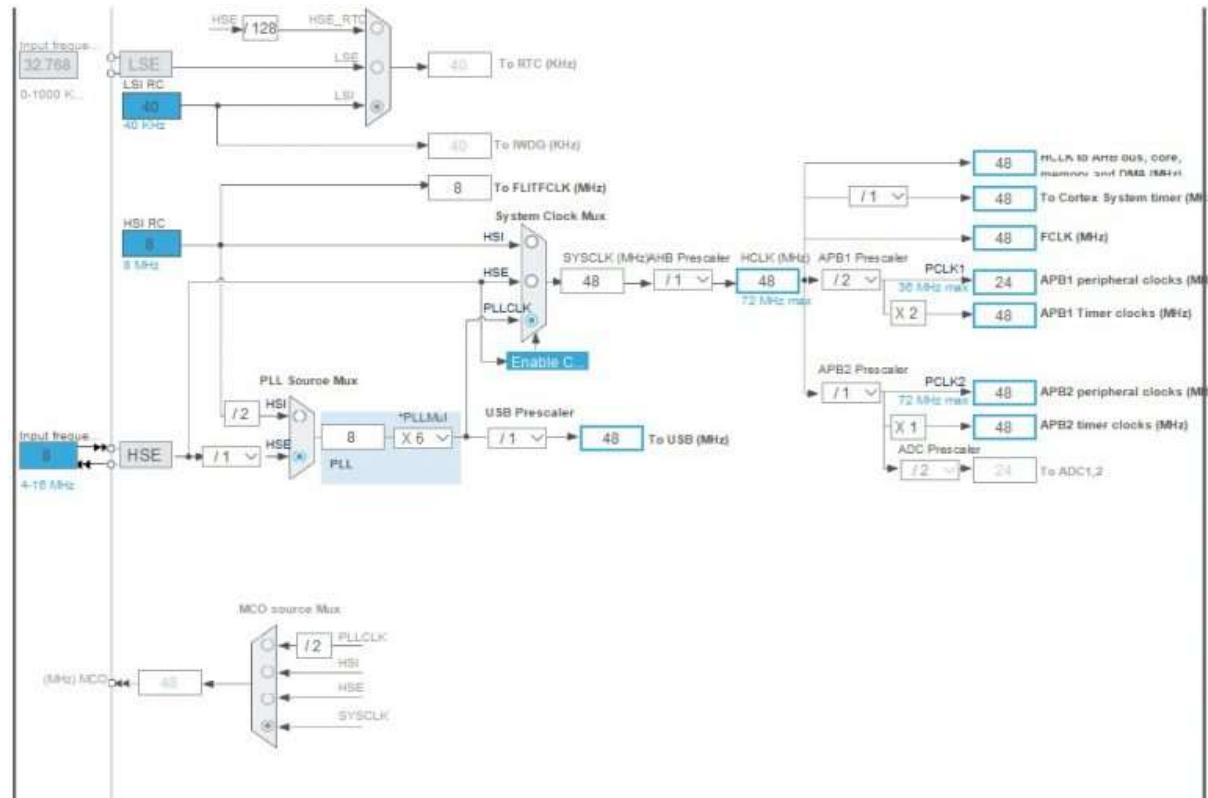
Link Power Management: Disabled

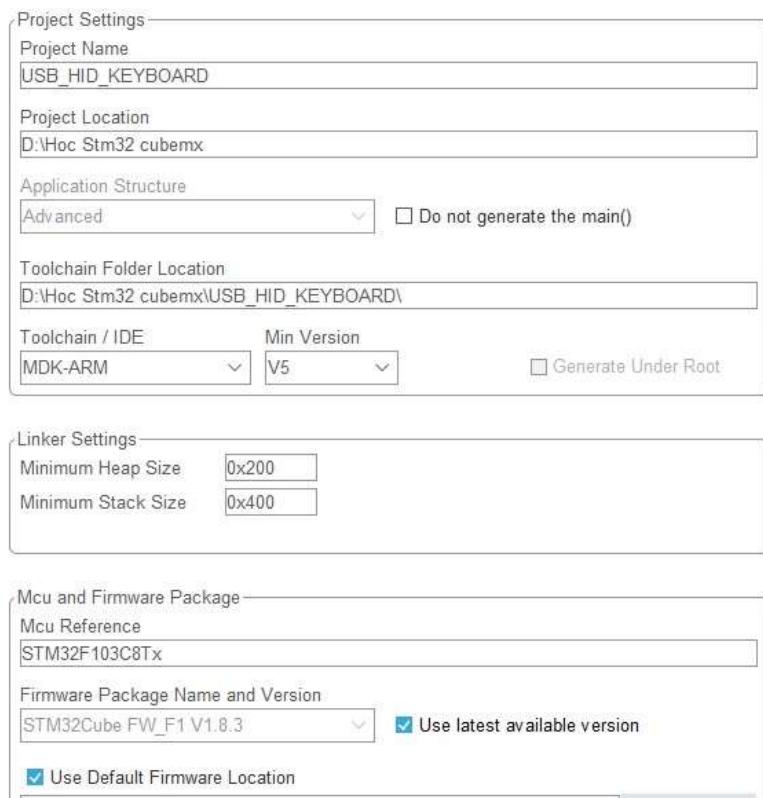
Battery Charging: Disabled

STM32F103CBTx LQFP48

USB chế độ Device

The screenshot shows the STM32CubeMX software interface. The left sidebar has sections for Categories (A-Z), Connectivity (CAN, I2C1, I2C2, SPI1, SPI2, USART1, USART2, USART3, USB), Computing, and Middleware (FATFS, FREERTOS, USB_DEVICE). The USB_DEVICE section is selected. The main area has tabs for Pinout & Configuration, Clock Configuration, Software Packs, and Pinout. The Clock Configuration tab is active, showing the USB_DEVICE Mode and Configuration. Under Mode, it says "Class For FS IP Human Interface Device Class (HID)". Below that is a Configuration section with tabs for Parameter Settings, Device Descriptor, and User Constants. The User Constants tab is selected, showing parameters like HID_FS_BINTERVAL (0xA), USBD_MAX_NUM_INTERFACES (1), USBD_MAX_NUM_CONFIGURATION (1), USBD_MAX_STR_DESC_SIZ (512 bytes), USBD_SELF_POWERED (Enabled), and USBD_DEBUG_LEVEL (0: No debug message).

Middleware chọn HID Class*Đao động USB chọn 48Mhz*



Gen code và Open Project

Lập trình STM32 USB HID Keyboard trên Keil C

Sau khi mở project. Chúng ta phải cấu hình lại cho HID, bởi vì mặc định CubeMx Gen ra là chuột máy tính. Đầu tiên sửa interfaceprotocol.

The screenshot shows the Keil IDE with the project 'USB_HID_KEYBOARD' open. The code editor displays the 'usbd_hid.c' file, specifically the HID descriptor code. A red arrow points to the line '0x01' under the comment '/*InterfaceProtocol : 0=none, 1=keyboard, 2=mouse*/'. Another red arrow points to the line '0x01' under the comment '/*Interface : Index of string descriptor*/'.

```

136 /* USB HID device FS Configuration Descriptor */
137 #ALIGN_BEGIN static uint8_t USBD_HID_CfgFSDesc[USB_HID_CONFIG_DESC_SIZ] _ALIGN_END =
138     0x09, /* bLength: Configuration Descriptor size */
139     USB_DESC_TYPE_CONFIGURATION, /* bDescriptorType: Configuration */
140     USB_HID_CONFIG_DESC_SIZ,
141     /* wTotalLength: Bytes returned */
142     0x00,
143     0x01, /* bNumInterfaces: 1 interface */
144     0x01, /* bConfigurationValue: Configuration value */
145     0x00, /* iConfiguration: Index of string descriptor describing
146             the configuration */
147     0xE0, /* bmAttributes: bus powered and Support Remote Wake-up */
148     0x32, /* MaxPower 100 mA: this current is used for detecting Vbus */
149
150     /****** Descriptor of Joystick Mouse interface ******/
151     /* 09 */
152     0x09, /* bLength: Interface Descriptor size */
153     USB_DESC_TYPE_INTERFACE, /* bDescriptorType: Interface descriptor type */
154     0x00, /* bInterfaceNumber: Number of Interface */
155     0x00, /* bAlternateSetting: Alternate setting */
156     0x01, /* bNumEndpoints */
157     0x03, /* bInterfaceClass: HID */
158     0x01, /* bInterfaceSubClass : 1=BOOT, 0=no boot */
159     0x01, /* bInterfaceProtocol : 0=none, 1=keyboard, 2=mouse */
160     0, /* iInterface: Index of string descriptor */
161
162     /****** Descriptor of Joystick Mouse HID ******/

```

Chỉnh Protocol thành 0x01

```

316 };
317
318 __ALIGN_BEGIN static uint8_t HID_MOUSE_ReportDesc[HID_MOUSE_REPORT_DESC_SIZE] __ALIGN_END =
319 {
320     0x05, 0x01,      // Usage Page (Generic Desktop Ctrls)
321     0x09, 0x06,      // Usage (Keyboard)
322     0xA1, 0x01,      // Collection (Application)
323     0x05, 0x07,      // Usage Page (Kbrd/Keypad)
324     0x19, 0xE0,      // Usage Minimum (0xE0)
325     0x29, 0xE7,      // Usage Maximum (0xE7)
326     0x15, 0x00,      // Logical Minimum (0)
327     0x25, 0x01,      // Logical Maximum (1)
328     0x75, 0x01,      // Report Size (1)
329     0x95, 0x08,      // Report Count (8)
330     0x81, 0x02,      // Input (Data,Var,Abs,No Wrap,Linear,Preferred State,No Null Position)
331     0x95, 0x01,      // Report Count (1)
332     0x75, 0x08,      // Report Size (8)
333     0x81, 0x03,      // Input (Const,Var,Abs,No Wrap,Linear,Preferred State,No Null Position)
334     0x95, 0x05,      // Report Count (5)
335     0x75, 0x01,      // Report Size (1)
336     0x05, 0x08,      // Usage Page (LEDs)
337     0x19, 0x01,      // Usage Minimum (Num Lock)
338     0x29, 0x05,      // Usage Maximum (Kana)
339     0x91, 0x02,      // Output (Data,Var,Abs,No Wrap,Linear,Preferred State,No Null Position,Non-volatile)
340     0x95, 0x01,      // Report Count (1)
341     0x75, 0x03,      // Report Size (3)
342     0x91, 0x03,      // Output (Const,Var,Abs,No Wrap,Linear,Preferred State,No Null Position,Non-volatile)
343     0x95, 0x06,      // Report Count (6)

```

Chỉnh Report Descriptor

```

311     0x00,
312     0x00,
313     0x40,
314     0x01,
315     0x00,
316 };
317
318 __ALIGN_BEGIN static uint8_t HID_MOUSE_ReportDesc[HID_MOUSE_REPORT_DESC_SIZE] __ALIGN_END =
319 {
320     0x05, 0x01,      // Usage Page (Generic Desktop Ctrls)
321     0x09, 0x06,      // Usage (Keyboard)
322     0xA1, 0x01,      // Collection (Application)
323     0x05, 0x07,      // Usage Page (Kbrd/Keypad)
324     0x19, 0xE0,      // Usage Minimum (0xE0)
325     0x29, 0xE7,      // Usage Maximum (0xE7)
326     0x15, 0x00,      // Logical Minimum (0)
327     0x25, 0x01,      // Logical Maximum (1)
328     0x75, 0x01,      // Report Size (1)
329     0x95, 0x08,      // Report Count (8)
330     0x81, 0x02,      // Input (Data,Var,Abs,No Wrap,Linear,Preferred State,No Null Position)
331     0x95, 0x01,      // Report Count (1)

```

W-data=364 ZI-data=3012

Tìm tới nơi define kích thước Report

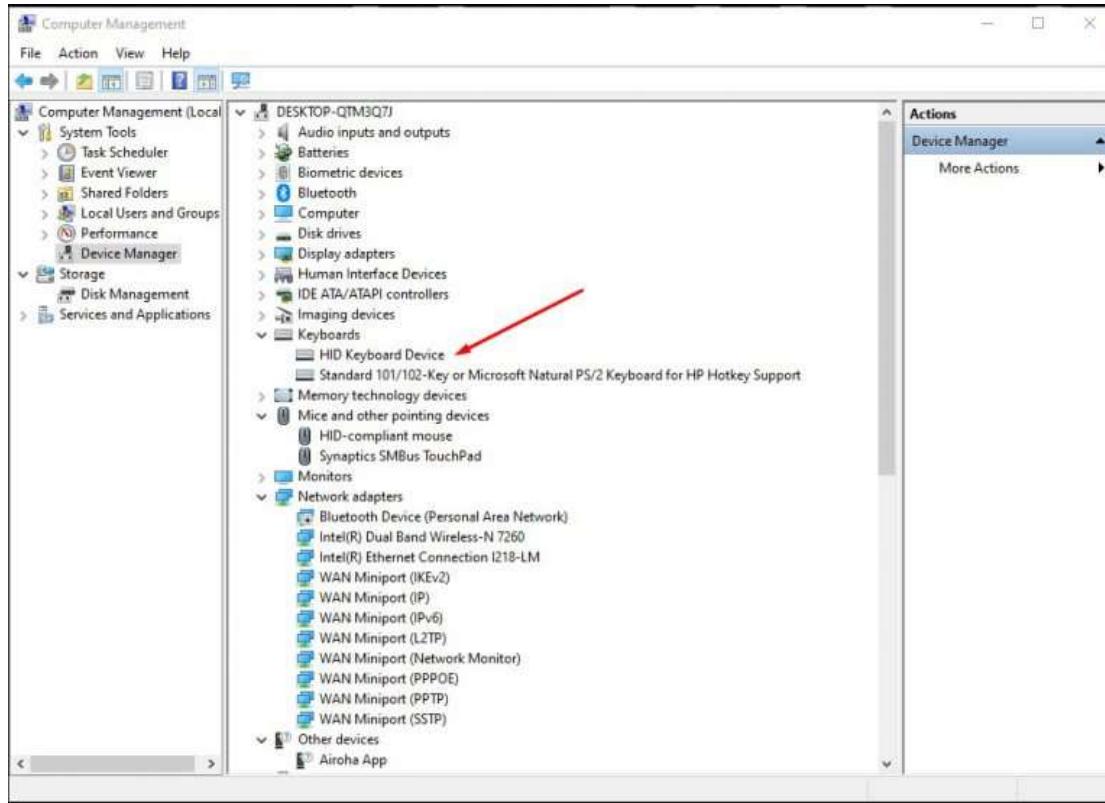
```

41 /** @defgroup USBD_HID_Exported_Defines
42 * @{
43 */
44 #define HID_EPIN_ADDR          0x81U
45 #define HID_EPIN_SIZE          0x04U
46
47 #define USB_HID_CONFIG_DESC_SIZ 34U
48 #define USB_HID_DESC_SIZ        9U
49 #define HID_MOUSE_REPORT_DESC_SIZE 63U
50
51 #define HID_DESCRIPTOR_TYPE     0x21U
52 #define HID_REPORT_DESC         0x22U
53
54 ifndef HID_HS_BINTERVAL
55 define HID_HS_BINTERVAL      0x07U
56 endif /* HID_HS_BINTERVAL */
57
58 ifndef HID_FS_BINTERVAL
59 define HID_FS_BINTERVAL      0x0AU
60 endif /* HID_FS_BINTERVAL */
61
62 define HID_REQ_SET_PROTOCOL 0x0BU
63 define HID_REQ_GET_PROTOCOL 0x03U
64
65 define HID_REQ_SET_IDLE     0x0AU
66 define HID_REQ_GET_IDLE     0x02U

```

Chỉnh lại kích thước RD là 63

Nhấn F7 Build và F8 nạp chương trình vào KIT. Sau đó cắm cổng Micro USB vào máy tính. Vào Manager trên Computer để xem thiết bị đã chuyển thành Keyboard chưa. Các bạn nếu chưa biết làm thì đọc lại bài **STM32 HID Chuột máy tính** nhé!



Ok, vậy là đã config xong HID. Chúng ta bắt đầu lập trình cho Keypad có thể gửi được dữ liệu lên máy tính

Phần Keypad các bạn down thư viện và add vào theo hướng dẫn: **STM32 Keypad 3x4**

Trong Main, chúng ta Include Keypad và USBD_HID vào.

Khởi tạo các biến cho USB và Keypad

```

25 /* USER CODE BEGIN Includes */
26 #include "usbd_hid.h"
27 #include "KEYPAD.h"
28 /* USER CODE END Includes */
29 /* Private typedef -----*/
30 /* USER CODE BEGIN PTD */
31 uint8_t report[8] = {
32     0, //MODIFIER
33     0, //RESERVED
34     0, //KEYCODE1
35     0, //KEYCODE2
36     0, //KEYCODE3
37     0, //KEYCODE4
38     0, //KEYCODE5
39     0 //KEYCODE6
40 };
41 extern USBD_HandleTypeDef hUsbDeviceFS;
42 char Key;
43 KEYPAD_Name KeyPad;
44 char KEYMAP [NUMROWS] [NUMCOLS] = {
45     {0x1E, 0x1F, 0x20},
46     {0x21, 0x22, 0x23},
47     {0x24, 0x25, 0x26},
48     {0x55, 0x27, 0x58}
49 };
50

```

Thêm thư viện và khai báo các biến

Phần KEYMAP thay vì các ký tự từ 1 – 9, chúng ta sẽ đổi chiều với tài liệu bên trên, sau đó thay vào các giá trị Hex của bàn phím tương ứng với Key 1 – 9. Key # mình sẽ thay bằng Enter.

```

110 KEYPAD3X4_Init(&KeyPad, KEYMAP, GPIOA, GPIO_PIN_4, GPIOA, GPIO_PIN_5, GPIOA, GPIO_PIN_6,
111                                     GPIOA, GPIO_PIN_0, GPIOA, GPIO_PIN_1,
112                                     GPIOA, GPIO_PIN_2, GPIOA, GPIO_PIN_3);
113 /* USER CODE END 2 */
114
115 /* Infinite loop */
116 /* USER CODE BEGIN WHILE */
117 while (1)
118 {
119     /* USER CODE END WHILE */
120
121     /* USER CODE BEGIN 3 */
122     Key = KEYPAD3X4_Readkey(&KeyPad);
123     if(Key)
124     {
125         report[2] = Key;
126         USBD_HID_SendReport(&hUsbDeviceFS, report, sizeof (report)); // gửi ki tu
127         HAL_Delay(50);
128
129         Key = 0;
130         report[2] = 0;
131         USBD_HID_SendReport(&hUsbDeviceFS, report, sizeof (report)); // dung gửi ki tu
132         HAL_Delay(50);
133     }
134 }
135 /* USER CODE END 3 */
136

```

Trước While chúng ta khởi tạo Keypad. Sau đó đọc dữ liệu từ bàn phím.

Nếu phím được nhấn, copy dữ liệu vào report[2] (Keycode 1), sau đó gửi dữ liệu đi

Tiếp đến xóa dữ liệu và dừng gửi dữ liệu bằng cách send report trống.

Lưu ý: Nếu bạn không gửi report trống lên, USB HID sẽ liên tục gửi dữ liệu lên máy tính

Build và nạp vào KIT.

Kết nối phần cứng

PA0,1,2,3 theo thứ tự nối với các chân 2,7,6,4

PA4,5,6 theo thứ tự nối với các chân 3,1,5

Kết quả

[Facebook Watch](#)

Kết

STM32 USB HID Keyboard khá giống với HID Mouse hay tất cả các kiểu thiết bị khác sử dụng chuẩn HID, với kiến thức ở bài này, bạn có thể dễ dàng tự làm cho mình 1 chiếc bàn phím máy tính mà không cần mua ở ngoài cửa hàng.

Nếu thấy bài viết này có ích hãy chia sẻ cho bạn bè và cùng vào Group **Nghiên Lập Trình** để kết nối với những anh em thích lập trình nhé !!!

Link Code mẫu: [Hướng dẫn download tài liệu lập trình stm32](#)

5/5 - (1 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 16: Lập trình STM32 USB CDC truyền nhận dữ liệu qua cổng COM ảo**
4. **Bài 13: Lập trình STM32 RTC – Real Time Clock**
5. **Lập trình STM32 từ A tới Z**
6. **Bài 8: Lập trình STM32 đọc ADC một kênh**



KHUÊ NGUYỄN

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

2 THOUGHTS ON “BÀI 18: LẬP TRÌNH SMT32 USB HID KEYBOARD BÀN PHÍM MÁY TÍNH”

Lê Văn Hiếu says:

Anh ơi khi nào có bài mới vậy, hóng quá.

13/07/2021 AT 11:10 SÁNG

TRẢ LỜI

Khuê Nguyễn says:

Anh viết thêm esp32 nên stm32 hơi pending 1 chút xíu hehe, thông cảm nhé

13/07/2021 AT 10:09 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
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

PROTEUS

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

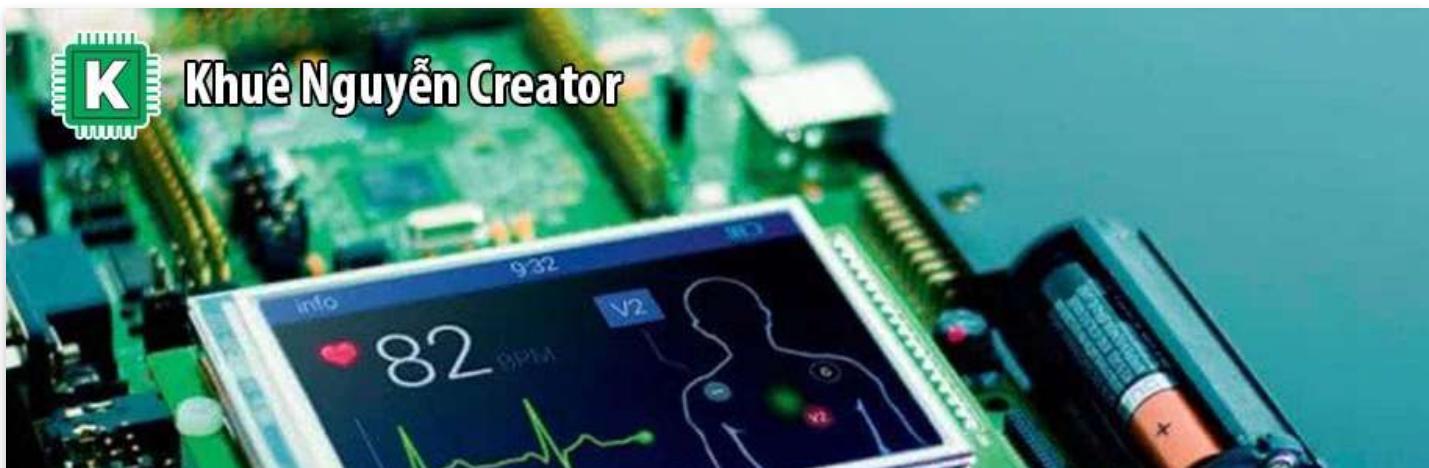


Lập trình STM32 HID Host giao tiếp với chuột và bàn phím

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

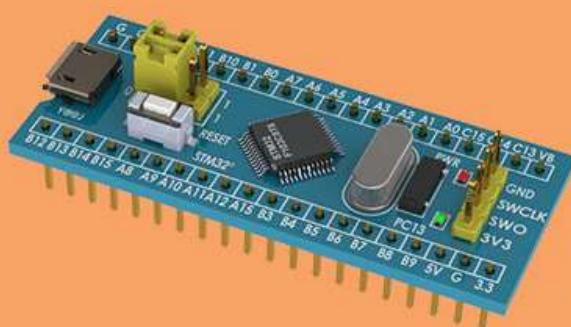
Trong bài này chúng ta sẽ cùng nhau tìm hiểu STM32 DAC với KIT STM32F407VE...

[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