

# Chain DualKey BLE HID

Chain DualKey BLE HID 相关 API 与案例程序。

## | 案例程序

---

## | 编译要求

- M5Stack 板管理版本  $\geq 3.2.5$
- 开发板选项 = M5ChainDualKey

```
1  #include <BLEDevice.h>
2  #include <BLEServer.h>
3  #include <BLEHIDDevice.h>
4  #include <HIDTypes.h>
5
6  const int PIN_BTN_1 = 0;
7  const int PIN_BTN_2 = 17;
8
9  // HID keyboard report structure: 1 byte modifiers + 1 byte reserved + 6 bytes keycodes
10 struct KeyReport {
11     uint8_t modifiers;
12     uint8_t reserved;
13     uint8_t keys[6];
14 };
15
16 // HID keyboard report map
17 static const uint8_t hidReportMap[] = {
18     0x05, 0x01, // Usage Page = Generic Desktop
19     0x09, 0x06, // Usage = Keyboard
20     0xA1, 0x01, // Collection = Application
21     0x85, 0x01, // Report ID = 1
22
23     0x75, 0x01, // Report Size = 1
24     0x95, 0x08, // Report Count = 8
25     0x05, 0x07, // Usage Page = Key Codes
26     0x19, 0xE0, // Usage Minimum = Keyboard LeftControl
27     0x29, 0xE7, // Usage Maximum = Keyboard Right GUI
28     0x15, 0x00, // Logical Minimum = 0
29     0x25, 0x01, // Logical Maximum = 1
30     0x81, 0x02, // Input = Data, Variable, Absolute (1 byte Modifiers)
31
32     0x75, 0x08, // Report Size = 8
33     0x95, 0x01, // Report Count = 1
34     0x81, 0x01, // Input = Constant (1 byte Reserved)
35
36     0x75, 0x08, // Report Size = 8
37     0x95, 0x06, // Report Count = 6
38     0x05, 0x07, // Usage Page = Key Codes
39     0x19, 0x00, // Usage Minimum = Reserved
40     0x29, 0x65, // Usage Maximum = Keyboard Application
41     0x15, 0x00, // Logical Minimum = 0
42     0x25, 0x65, // Logical Maximum = 101
43     0x81, 0x00, // Input = Data, Array (6 bytes Keycodes)
44     0xC0      // End Collection
45 };
46
47 BLEHIDDevice* hidDevice;
48 BLECharacteristic* inputReportCharacteristic;
49 bool deviceConnected = false;
50
```

```

51 // Connect status callback
52 class MyBLEServerCallbacks : public BLEServerCallbacks {
53     void onConnect(BLEServer* pServer) override {
54         deviceConnected = true;
55         Serial.println("BLE connected");
56     }
57     void onDisconnect(BLEServer* pServer) override {
58         deviceConnected = false;
59         Serial.println("BLE disconnected");
60         BLEDevice::startAdvertising(); // Restart advertising after disconnected
61     }
62 };
63
64 // Function to report a key was pressed and released, see keycode definitions:
65 // https://www.usb.org/sites/default/files/hut1_6.pdf
66 void sendKey(uint8_t keycode, uint8_t modifier = 0) {
67     if (!deviceConnected || inputReportCharacteristic == nullptr) {
68         return;
69     }
70     KeyReport report = { 0 };
71
72     // Pressed
73     report.modifiers = modifier;
74     report.keys[0] = keycode;
75     inputReportCharacteristic->setValue((uint8_t*)&report, sizeof(report));
76     inputReportCharacteristic->notify();
77
78     delay(10);
79
80     // Released (clean)
81     report.modifiers = 0;
82     report.keys[0] = 0;
83     inputReportCharacteristic->setValue((uint8_t*)&report, sizeof(report));
84     inputReportCharacteristic->notify();
85 }
86
87 void setup() {
88     delay(2000);
89     Serial.begin(115200);
90     Serial.println("Starting Chain DualKey BLE Keyboard");
91
92     pinMode(PIN_BTN_1, INPUT);
93     pinMode(PIN_BTN_2, INPUT);
94
95     // Init BLE device, create BLE server, create HID device, input report (ID=1)
96     BLEDevice::init("Chain DualKey Keyboard"); // Device name shown on computers and phones
97     BLEServer* pServer = BLEDevice::createServer();
98     pServer->setCallbacks(new MyBLEServerCallbacks());
99     hidDevice = new BLEHIDDevice(pServer);
100     inputReportCharacteristic = hidDevice->inputReport(1);
101
102     // Set HID device info
103     hidDevice->manufacturer()->setValue("M5Stack");

```

```

104 hidDevice->pnp(0x02, 0x1234, 0x0001, 0x0100); // vendorIdSource (0x02=Bluetooth SIG), vendorId, |
105 hidDevice->hidInfo(0x00, 0x01);
106
107 // Set HID report map
108 hidDevice->reportMap((uint8_t*)hidReportMap, sizeof(hidReportMap));
109 hidDevice->startServices();
110
111 // Set BLE security (BOND)
112 BLESecurity* pSecurity = new BLESecurity();
113 pSecurity->setAuthenticationMode(ESP_LE_AUTH_BOND);
114
115 // Set BLE advertising
116 BLEAdvertising* pAdvertising = BLEDevice::getAdvertising();
117 pAdvertising->setAppearance(HID_KEYBOARD);
118 pAdvertising->addServiceUUID(hidDevice->hidService()->getUUID());
119 pAdvertising->start();
120
121 Serial.println("BLE HID Keyboard is advertising, ready to pair.");
122 }
123
124 void loop() {
125     static bool lastBtnState_1 = HIGH;
126     static bool lastBtnState_2 = HIGH;
127     bool btnState_1 = digitalRead(PIN_BTN_1);
128     bool btnState_2 = digitalRead(PIN_BTN_2);
129
130     if (lastBtnState_1 == HIGH && btnState_1 == LOW) {
131         Serial.println("Button 1 pressed, send 'a'");
132         sendKey(0x04); // 'a' + no modifier
133     }
134     if (lastBtnState_2 == HIGH && btnState_2 == LOW) {
135         Serial.println("Button 2 pressed, send 'v' + LeftCtrl");
136         sendKey(0x19, 0x01); // 'v' + LeftCtrl
137         // sendKey(0x19, 0x08); // 'v' + LeftCmd
138     }
139
140     lastBtnState_1 = btnState_1;
141     lastBtnState_2 = btnState_2;
142     delay(10);
143 }

```

将以上代码复制到 Arduino IDE，编译并上传至 Chain DualKey。上传完成后，在电脑、手机等主机设备的蓝牙设置中找到名为 **Chain DualKey Keyboard** 的键盘并配对连接。按下 Key1 输入 **a** 字母键，按下 Key2 将触发 **Ctrl + V** 组合键在 Windows 上粘贴（macOS 的 Command 键请看代码注释）。



串口输出如图：

```
ble.ino
1  #include <BLEDevice.h>
2  #include <BLEServer.h>
3  #include <BLEHIDDevice.h>
4  #include <HIDTypes.h>
5
6  const int PIN_BTN_1 = 0;
7  const int PIN_BTN_2 = 17;
8
9  // HID keyboard report structure: 1 byte modifiers + 1 byte reserved + 6 bytes keycodes
10 struct KeyReport {
11     uint8_t modifiers;
12     uint8_t reserved;
13     uint8_t keys[6];
14 };
15
16 // HID keyboard report map
17 static const uint8_t hidReportMap[] = {
18     0x05, 0x01, // Usage Page = Generic Desktop
19     0x09, 0x06, // Usage = Keyboard
20     0xA1, 0x01, // Collection = Application
21     0x85, 0x01, // Report ID = 1
22
23     0x75, 0x01, // Report Size = 1
24     0x05, 0x06, // Report Count = 6
```

Output Serial Monitor X

Message (Enter to send message to 'M5ChainDualKey' on '/dev/cu.usbmodem101')

Both NL & CR 115200 baud

```
11:07:56.832 -> Starting Chain DualKey BLE Keyboard
11:07:57.094 -> BLE HID Keyboard is advertising, ready to pair.
11:08:02.770 -> BLE connected
11:08:04.246 -> E (11851) BT_GATT: GATT_INSUF_ENCRYPTION
11:08:04.246 ->
11:08:51.726 -> Button 1 pressed, send 'a'
11:08:53.798 -> Button 2 pressed, send 'v' + LeftCtrl
```

Ln 1, Col 1 M5ChainDualKey on /dev/cu.usbmodem101

程序上传后若串口无输出、蓝牙找不到设备，可以重启设备，操作方法是：将开关拨至中间位置，断开 USB-C 数据线并重新连接（不要按住 Key1）。

### 注意

修改蓝牙设备本身的信息（名称、manufacturer、pnp、hidInfo、hidReportMap 等）之前，建议先在电脑手机等主机上取消配对，然后给 Chain DualKey 写入新的程序，再次连接前重新重启（关闭然后开启）主机蓝牙。否则，新的蓝牙广播与主机已有记录冲突，可能会造成主机蓝牙栈崩溃重启。

## 相关链接

- 更多按键的键码，可以查看 [HID Usage Tables 1.6](#) 中的 10 Keyboard/Keypad Page 章节。
- Ctrl, Shift, Alt, GUI (Windows/Command) 等修饰键不使用上表中的普通键码，而使用单独的 modifiers 字段，可以查看 [Device Class Definition for HID 1.12](#) 中的 8.3 Report Format for Array Items 章节。
- [BLE for ESP32 Arduino Core - GitHub](#)
- [ESP32 Arduino BLE - Espressif Docs](#)
- [ESP-IDF BT/BLE HID Device Demo - GitHub](#)

- [HID Service 1.0 - Bluetooth](#)
- [HID over GATT Profile 1.0 - Bluetooth](#)