

## 實作使用者自定義命令(Vendor Command)

NuMicro® 32 位系列微控制器範例代碼介紹

### 文件資訊

代碼簡述	示範如何擴充使用者自定義指令(Vendor Command)。
BSP 版本	NUC123 Series BSP CMSIS V3.01.001
開發平台	NuTiny-EVB-NUC123-LQFP64 v1.0

*The information described in this document is the exclusive intellectual property of Nuvoton Technology Corporation and shall not be reproduced without permission from Nuvoton.*

*Nuvoton is providing this document only for reference purposes of NuMicro microcontroller based system design.  
Nuvoton assumes no responsibility for errors or omissions.*

*All data and specifications are subject to change without notice.*

*For additional information or questions, please contact: Nuvoton Technology Corporation.*

[www.nuvoton.com](http://www.nuvoton.com)

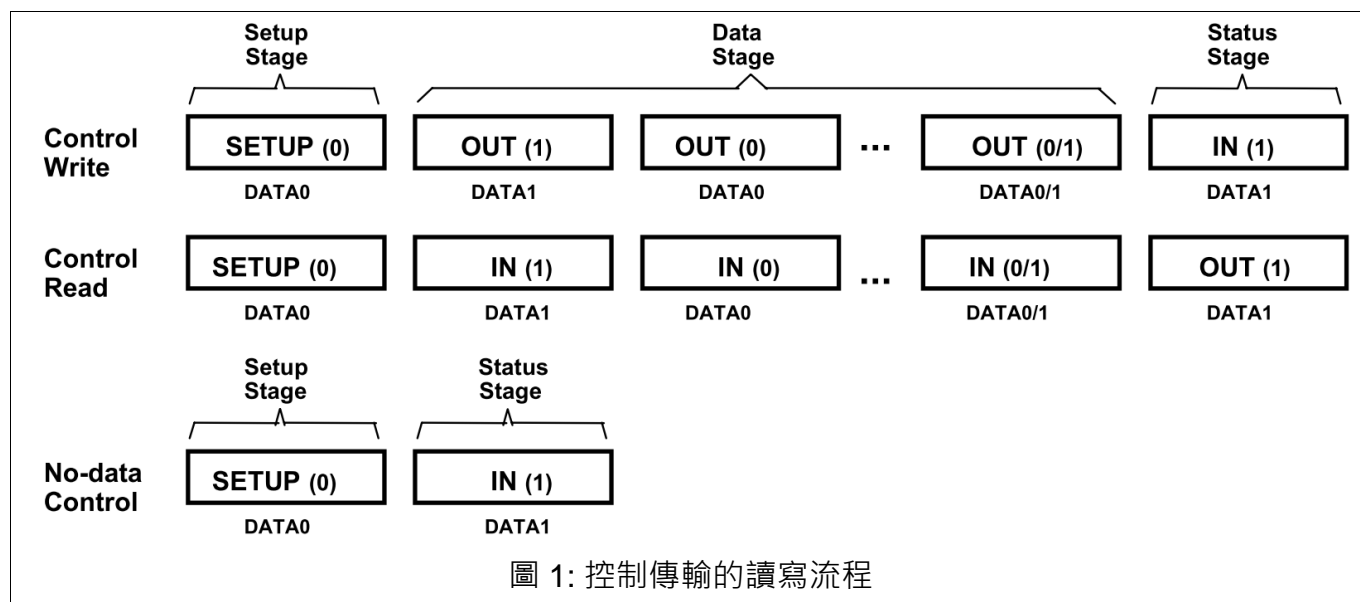
## 1 功能介紹

### 1.1 簡介

此範例以HID裝置為例，示範如何擴充使用者自定義指令(Vendor Command)。本範例會實作2個自定義指令，寫入和讀取的控制傳輸指令，並搭配Bus Hound軟體觀察結果，展示整個流程。

### 1.2 原理

Vendor Command建立在USB的控制傳輸之上。因此傳輸分為三個階段，如圖 1，Setup、Data和STATUS。其中SETUP和STATUS階段是必備。DATA階段是可選擇的，視應用而定。



Offset	Field	Size	Value	Description
0	<i>bmRequestType</i>	1	Bitmap	Characteristics of request:  D7: Data transfer direction 0 = Host-to-device 1 = Device-to-host  D6...5: Type 0 = Standard 1 = Class 2 = Vendor 3 = Reserved  D4...0: Recipient 0 = Device 1 = Interface 2 = Endpoint 3 = Other 4...31 = Reserved
1	<i>bRequest</i>	1	Value	Specific request (refer to Table 9-3)
2	<i>wValue</i>	2	Value	Word-sized field that varies according to request
4	<i>wIndex</i>	2	Index or Offset	Word-sized field that varies according to request; typically used to pass an index or offset
6	<i>wLength</i>	2	Count	Number of bytes to transfer if there is a Data stage

圖 2: Setup 封包的資料格式

本次實驗要實作讀取和寫入64 位元組的資料。設定階段的命令須符合USB控制傳輸的設定封包的格式，如圖 2。因此我們定義了表 1的命令格式，其中的bmRequestType為Vendor Command的命令分類和傳輸方向；bRequest的內容即是這次的自訂Vendor Command的控制代碼，將寫入代碼設為0x1，讀取代碼設為0x2；wLength是要傳輸的資料長度。其他未提到的欄位此次皆未使用，設為0。

Vendor-Defined Command	bmRequestType	bRequest	wValue	wIndex	wLength
VENDOR_WRITE_TEST	0x40	0x01	0	0	0x40
VENDOR_READ_TEST	0xC0	0x02	0	0	0x40

表 1: Vendor Command 的 SETUP 資料格式

### 1.3 執行結果

本範例使用Bus Hound來控制傳輸測試。本裝置連接上PC後，打開Bus Hound，可以在Device Tree( 圖 3)找到此裝置，可以看到VID和PID分別是十六進位的0416和B001。照著圖 3的操作

步驟可進入指令控制(Bus Commander)視窗。

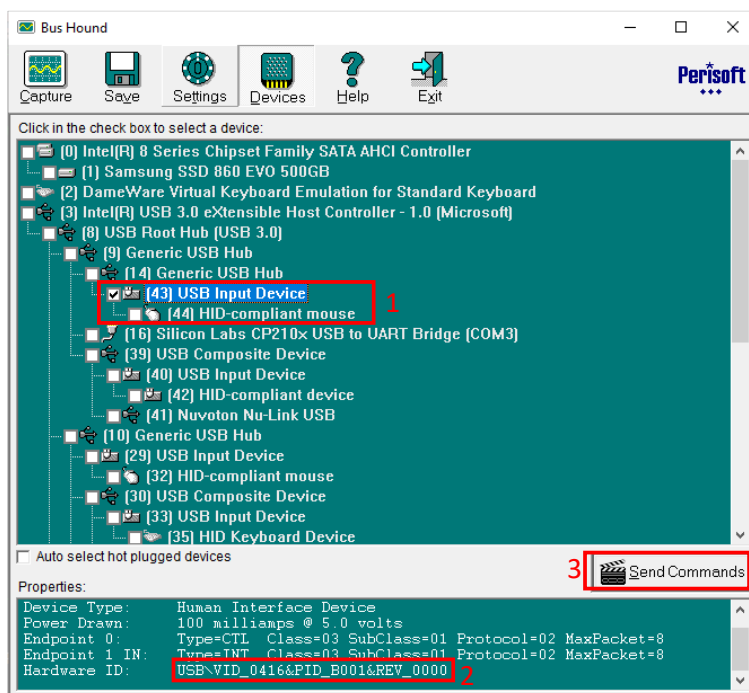


圖 3: Bus Hound 介面

### 1.3.1 Read Command

在指令控制視窗，依照圖 4 的操作步驟，可觀察指令的執行結果：

- 1， 在終端軟體透過 UART 的輸出，觀察初始的緩衝區數值。按下任意鍵，可以顯示緩衝區的初始值，0x0 ~ 0x3F。
- 2， 在 Bus Command 視窗輸入表 1 的讀取指令。
- 3， 按下"Run"按鈕，執行指令。
- 4， 可以觀察 Bus Hound 讀取到的數值與步驟 1 顯示的數值相符，代表傳輸完成。

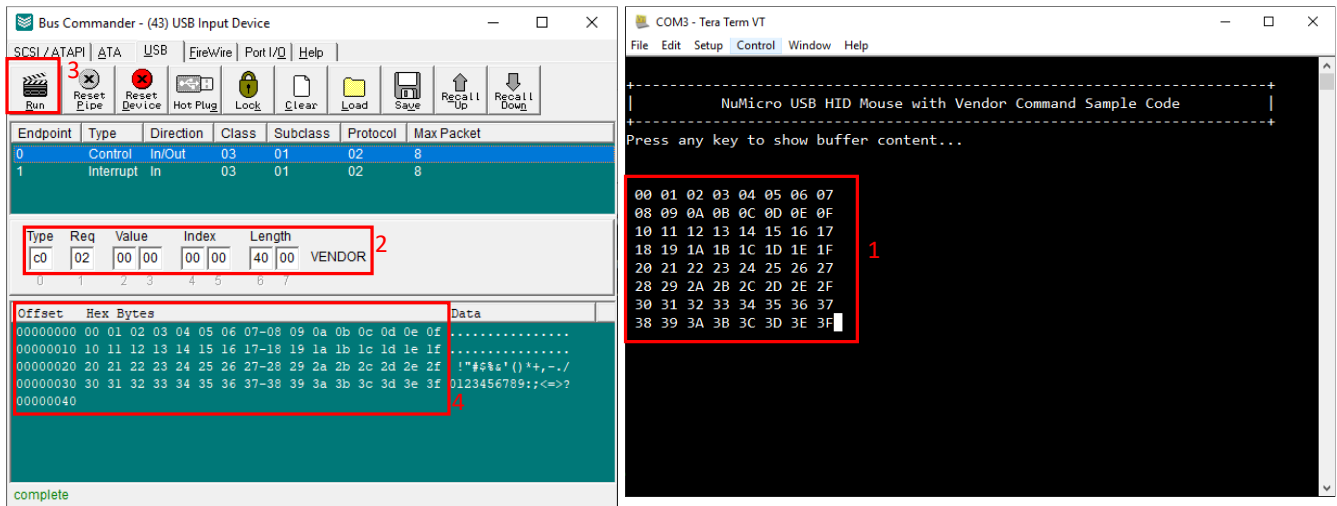


圖 4: 讀取指令的操作

### 1.3.2 Write Command

在指令控制視窗，依照圖 5 的操作步驟，觀察指令的執行結果。

1. 對終端視窗按下任意鍵，顯示目前的緩衝區數值為 0x0 ~ 0x3f。
2. 在 Bus Commander 視窗輸入表 1 的寫入指令。
3. 於 Bus Commander 視窗的資料區填入欲傳輸的資料。
4. 按下 Run 的按鈕，USB 主機就會執行控制傳輸。
5. 在終端視窗按下任意鍵，可看到步驟 3 的資料已經被寫入至裝置端的緩衝區內。

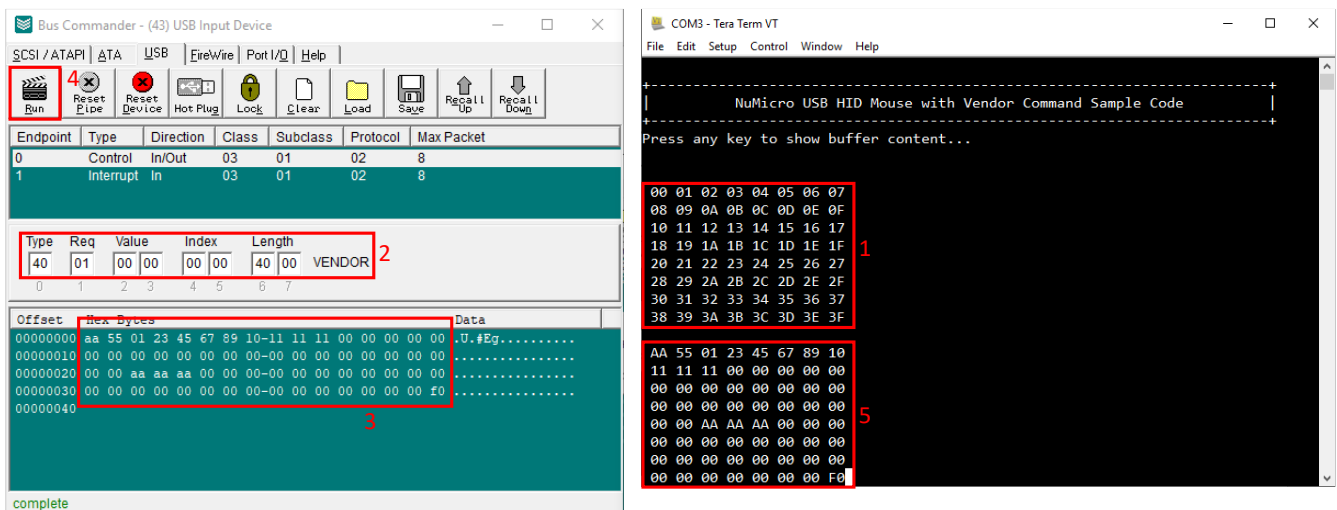


圖 5: 寫入指令的操作

## 2 代碼介紹

### 2.1 main.c

將USB裝置驅動的回調函數 (callback function) , 設為VENDOR\_Request()。當收到使用者自定義命令後, USB 裝置驅動就會呼叫的處理函數。

```
/* Set Callback function for Vendor Command */
USB_D_SetVendorRequest(VENDOR_Request);
```

### 2.2 hid\_vendor\_command.h, hid\_vendor\_command.c

hid\_vendor\_command.h裡面定義"寫入指令"為0x1, "讀取指令"為0x2。

```
/*!<USB VENDOR CMD */
#define VENDOR_WRITE_TEST 0x1
#define VENDOR_READ_TEST 0x2
```

hid\_vendor\_command.c裡的VENDOR\_Request函數, 實作了Vendor Command的處理。  
進入程式的開頭, 先把SETUP封包的資料儲存, 供後續判斷使用。

```
void VENDOR_Request(void)
{
    uint8_t buf[8];
    uint32_t wLength;

    USB_D_GetSetupPacket(buf);

    wLength = buf[7];
    wLength = buf[6] | (wLength << 8);
```

buf[0]是bmRequestType欄位, 見表 1, 提供資料方向和指令種類。這裡把資料分為兩個方向處理。由於USB\_D 驅動已經處理完設定階段, VENDOR\_Request函數只需負責接下來的資料階段和狀態階段。

資料階段的處理, 只需識別命令欄位 (buf[1]) 是不是欲處理的命令, 並呼叫USB\_D\_PrepareCtrlIn()並設定存放資料的緩衝區位址和長度。

```
if(buf[0] & 0x80) /* request data transfer direction */
{
    /* Device to host */
    switch(buf[1])
    {
        case VENDOR_READ_TEST:
        {
            wLength = Minimum(wLength, sizeof(g_au8temp));

            /* Data stage */
            USB_D_PrepareCtrlIn((uint8_t *)g_au8temp, wLength);
```

狀態階段的處理，須設定TOGGLE FLAG為1，並呼叫USBD\_PrepareCtrlOut();由於沒有資料須要經由端點0回傳至USB主機，因此參數皆設為0。

```

        /* Status stage */
        USBD_SET_DATA1(EP1);
        USBD_PrepareCtrlOut(0, 0);
        break;
    }

default:
{
    /* Setup error, stall the device */
    USBD_SetStall(EP0);
    USBD_SetStall(EP1);
    break;
}
}
}

```

當收到寫入指令(VENDOR\_WRITE\_TEST)後，呼叫USB driver的USBD\_PrepareCtrlOut()，設定暫存資料的位址和長度，做資料階段的處理。STATUS階段不須回傳資料，因此資料回傳長度設為0。當傳輸結束後，收到的資料會存放在g\_au8temp的陣列內。

```

else
{
    /* Host to device */
    switch(buf[1])
    {
        case VENDOR_WRITE_TEST:
        {
            /* Data stage*/
            USBD_PrepareCtrlOut(g_au8temp, wLength);

            /* Status stage */
            USBD_SET_DATA1(EP0);
            USBD_SET_PAYLOAD_LEN(EP0, 0);
            break;
        }

        default:
        {
            /* Setup error, stall the device */
            USBD_SetStall(EP0);
            USBD_SetStall(EP1);
            break;
        }
    }
}
}
}

```

### 3 軟體與硬體環境

#### ● 軟體環境

##### ■ BSP 版本

- ◆ NUC123 Series BSP CMSIS V3.01.001

##### ■ IDE 版本

- ◆ Keil uVersion 5.26

##### ■ 測試軟體

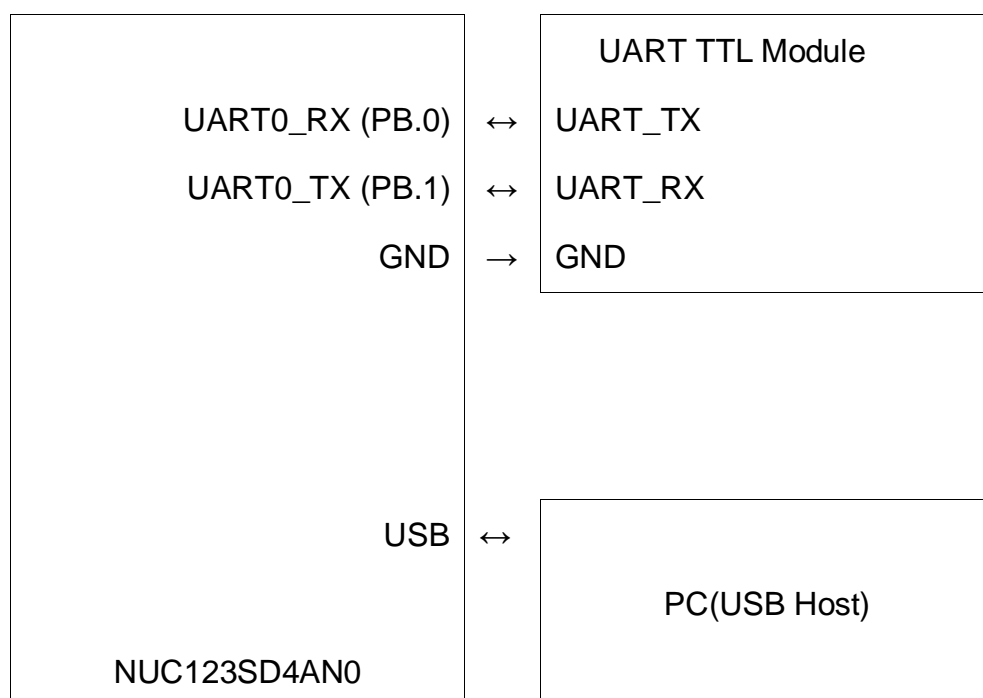
- ◆ 終端視窗軟體，例如 [putty\(https://www.putty.org/\)](https://www.putty.org/)。
- ◆ Bus Hound(<http://www.perisoft.net/bushound/>)

#### ● 硬體環境

##### ■ 電路元件

- ◆ NuTiny-EVB-NUC123-LQFP64 v1.0 C
- ◆ USB mini USB cable
- ◆ USB-UART TTL








##### ■ 示意圖





## 4 目錄資訊

### EC\_NUC123\_USBD\_HID\_Vendor\_Command\_V1.00

 Library	Sample code header and source files
 CMSIS	Cortex® Microcontroller Software Interface Standard (CMSIS) by Arm® Corp.
 Device	CMSIS compliant device header file
 StdDriver	All peripheral driver header and source files
 SampleCode	
 ExampleCode	Source file of example code
 KEIL	KEIL project file

## 5 如何執行範例程式

1. 根據目錄資訊章節進入 ExampleCode 路徑中的 KEIL 資料夾，雙擊 NUC123\_USBD\_HID\_Vendor\_Command.uvproj。
2. 進入編譯模式介面
  - a. 編譯
  - b. 下載代碼至記憶體
  - c. 進入 / 離開除錯模式
3. 進入除錯模式介面
  - a. 執行代碼，可按下 RESET 鍵直接執行，並從 Console 視窗觀察結果。

## 6 修訂紀錄

Date	Revision	Description
Aug. 1, 2019	1.00	1. 初始發佈.

### Important Notice

Nuvoton Products are neither intended nor warranted for usage in systems or equipment, any malfunction or failure of which may cause loss of human life, bodily injury or severe property damage. Such applications are deemed, "Insecure Usage".

Insecure usage includes, but is not limited to: equipment for surgical implementation, atomic energy control instruments, airplane or spaceship instruments, the control or operation of dynamic, brake or safety systems designed for vehicular use, traffic signal instruments, all types of safety devices, and other applications intended to support or sustain life.

All Insecure Usage shall be made at customer's risk, and in the event that third parties lay claims to Nuvoton as a result of customer's Insecure Usage, customer shall indemnify the damages and liabilities thus incurred by Nuvoton.

Please note that all data and specifications are subject to change without notice.  
All the trademarks of products and companies mentioned in this datasheet belong to their respective owners.