HID 触控装置

NuMicro® 32 位系列微控制器范例代码介绍

文件信息

代码简述	HID 触控感应装置
BSP 版本	NUC123 Series BSP CMSIS V3.01.001
开发平台	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

1 功能介绍

1.1 简介

本范例实作一个HID(Human Interface Device)触控装置。模拟两个输入轨迹。

1.2 原理

本装置内含控制管线(端点0)，用于控制传输。此控制管线由装置端硬件的端点0和端点1构成。从USB主机的角度看来，只会看到端点0；HID触控数据传输则是由唯一的接口完成，此接口有个中断的输入端点，此端点用于输入方向的中断传输(Interrupt Transfer)。接口和端点的关系列于表 1。

Endpoints in NUC123	Composite Device	Function
Endpoint 0 Control IN **Endpoint 1 Control OUT**	Control Pipe (Endpoint 0)	Control Transfer
Endpoint 2	Interface 0: Endpoint 1: Interrupt IN	HID Touch

表 1: HID 触控装置的接口和端点组态

接口0的触控数据，使用表 2的格式回传。此格式即是HID的报告描述元所定义的数据格式，定义了两组相同的数据结构，让此装置可同时回传两个不同的轨迹。关于触控装置报告描述元更详细的字段解说，可参考微软公司网站。

Byte	Bits	Description
0	0	Tip Switch
0	1	In Range
0	2	Touch Valid
0	3 ~ 7	Padding
1	0 ~ 7	Contact Identifier
2 ~ 3	0 ~ 7	X-axis
4 ~ 5	0 ~ 7	Y-axis
6	0	Tip Switch
6	1	In Range
6	2	Touch Valid
6	3 ~ 7	Padding
7	0 ~ 7	Contact Identifier
8 ~ 9	0 ~ 7	X-axis
10 ~ 11	0 ~ 7	Y-axis

表 2: HID 触控装置的 报告格式

1.3 执行结果

此范例需与连接个人计算机才能运行。此范例程序执行时，使用windows的设备管理器，可以

获得图 1的装置列表，HID-compliant touch screen就是此次展示的装置。

图 1:设备管理器的装置列表

图 2是使用USB封包分析仪纪录的传输封包，可看见每次交易(transaction)，有14 字节的数据以输入的方向传送。

Transaction	F	IN	ADDR	ENDP	T	Data	ACK	Time	Time Stamp
196	S	0x96	60	1	1	01 00 00 00 00 00 00 00 01 00 00 00 00 00	0x4B	1.000 ms	1 . 722 543 132
197	S	0x96	60	1	0	01 00 00 00 00 00 00 00 01 00 00 00 00 00	0x4B	999.982 us	1 . 723 543 150
198	S	0x96	60	1	1	01 00 00 00 00 00 00 00 01 00 00 00 00 00	0x4B	1.000 ms	1 . 724 543 132
199	S	0x96	60	1	0	01 00 00 00 00 00 00 00 01 00 00 00 00 00	0x4B	1.000 ms	1 . 725 543 132
200	S	0x96	60	1	1	01 00 00 00 00 00 00 00 01 00 00 00 00 00	0x4B	999.982 us	1 . 726 543 150
201	S	0x96	60	1	0	01 07 00 02 02 00 01 07 01 02 02 20 01 02	0x4B	1.000 ms	1 . 727 543 132
202	S	0x96	60	1	1	01 07 00 02 02 00 01 07 01 02 02 20 01 02	0x4B	1.000 ms	1 . 728 543 132
203	S	0x96	60	1	0	01 07 00 02 02 00 01 07 01 02 02 20 01 02	0x4B	1.000 ms	1 . 729 543 132
204	S	0x96	60	1	1	01 07 00 02 02 00 01 07 01 02 02 20 01 02	0x4B	1.000 ms	1 . 730 543 132
205	S	0x96	60	1	0	01 07 00 02 02 00 01 07 01 02 02 20 01 02	0x4B	1.000 ms	1 . 731 543 132

图 2:HID 触控装置的传输数据

在装置连加上计算机后，打开”小画家”软件，此装置会画出两条平行的轨迹，如图 3。

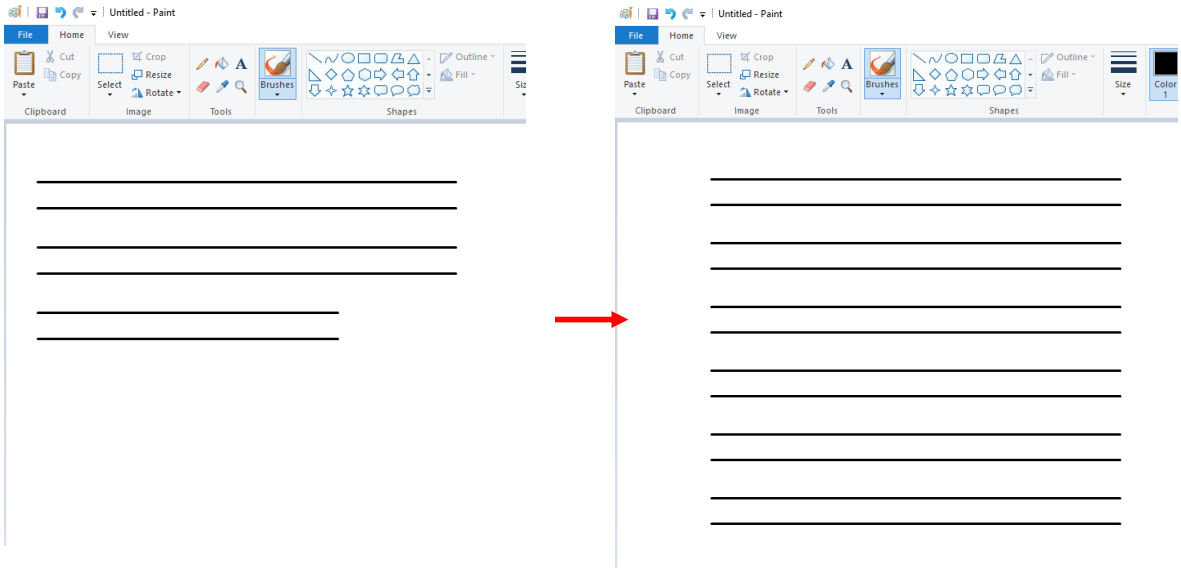


图 3: HID 触碰装置在小画家画并行线

2 代码介绍

此范例程序使用 HID 标准实现触控装置的仿真。当主机发出 IN TOKEN 时，IN 的中断传输端点使用定义于 HID 报告描述元的数据格式回传数据，如图 2。2.1~2.2 节将介绍程序处理的流程。

2.1 Main.c

While回圈内的HID_UpdateTouchData()用于数据的传输。里面实作的IN token中断产生时的处理逻辑。

```
    while(1)
    {
       HID_UpdateTouchData();
    }
```

2.2 hid_touch.c

USB主机会发出IN Token向端点2请求数据。当IN Token收到时，端点2的USBD中断处理程序会把g_u8EP2Ready设为真。当HID_UpdateTouchData ()检查到g_u8EP2Ready为真时，就会设定封包回传数据。 由于有两条轨迹同时上传，每条轨迹需要6字节的空间，每组数据前会再加上一个字节，存放Report ID。所以一次回传的数据长度总共14字节。

```
/* two lines demo in Paint */
uint32_t u32ReportCount = 0;
uint8_t u8IsX1Send04 = 0, u8IsX2Send04 = 0;
void HID_UpdateTouchData(void)
{
    uint8_t *pu8buf;
    static uint16_t u16X1 = 0x01f0, u16Y1 = 0x0100;

    if(g_u8EP2Ready)
    {

        pu8buf = (uint8_t *)(USBD_BUF_BASE + USBD_GET_EP_BUF_ADDR(EP2));

        /* Report ID 1 */
        pu8buf[0] = 1;
        pu8buf[2] = 0;
        pu8buf[8] = 1;
```

以下是画线的实作，把欲执行的触控资料填入对应的字段，使用者可以根据自己的需求更改程序，来仿真不同的轨迹效果。

```
        u32ReportCount++;

        if ((u16X1 >= 0x0200) && (u16X1 <= 0x0400)) /* touchDown */
        {
            u8IsX1Send04 = 1;

            pu8buf[1] = 0x07;
            pu8buf[3] = u16X1 & 0xff;
            pu8buf[4] = (u16X1 >> 8) & 0xff;
```

```
            pu8buf[5] = u16Y1 & 0xff;
            pu8buf[6] = (u16Y1 >> 8) & 0xff;
            pu8buf[7] = 0x07;
            pu8buf[9] = u16X1 & 0xff;
            pu8buf[10] = (u16X1 >> 8) & 0xff;
            pu8buf[11] = (u16Y1+0x20) & 0xff;
            pu8buf[12] = ((u16Y1+0x20) >> 8) & 0xff;
            pu8buf[13] = 2;
        }
        else if (u8IsX1Send04)     /* touchUp */
        {
            u8IsX1Send04 = 0;
            pu8buf[1] = 0x04;
            pu8buf[3] = 0x00;
            pu8buf[4] = 0x04;
            pu8buf[5] = u16Y1 & 0xff;
            pu8buf[6] = (u16Y1 >> 8) & 0xff;
            pu8buf[7] = 0x04;
            pu8buf[9] = 0x00;
            pu8buf[10] = 0x04;
            pu8buf[11] = (u16Y1+0x20) & 0xff;
            pu8buf[12] = ((u16Y1+0x20) >> 8) & 0xff;
            pu8buf[13] = 2;

            if (u16Y1 == 0x0100)
            {
                pu8buf[5] = 0xE0;
                pu8buf[6] = 0x02;
                pu8buf[11] = 0x00;
                pu8buf[12] = 0x03;
            }
            else
            {
                pu8buf[5] = (u16Y1-0x50) & 0xff;
                pu8buf[6] = ((u16Y1-0x50) >> 8) & 0xff;
                pu8buf[11] = ((u16Y1+0x20)-0x50) & 0xff;
                pu8buf[12] = (((u16Y1+0x20)-0x50) >> 8) & 0xff;
            }
        }
        else
        {
            pu8buf[1] = 0;
            pu8buf[3] = 0;
            pu8buf[4] = 0;
            pu8buf[5] = 0;
            pu8buf[6] = 0;
            pu8buf[7] = 0;
            pu8buf[9] = 0;
            pu8buf[10] = 0;
            pu8buf[11] = 0;
            pu8buf[12] = 0;
            pu8buf[13] = 0;
        }

        if ((u32ReportCount % 6) == 0)
            u16X1 += 0x3;

        if (u16X1 > 0x0400)
        {
            u16X1 = 0x01f0;
            u16Y1 += 0x50;
            if (u16Y1 > 0x0300)
                u16Y1 = 0x0100;
        }
```

最后设定端点2的数据长度来回传数据。

```
        g_u8EP2Ready = 0;
        USBD_SET_PAYLOAD_LEN(EP2, 14);
    }
}
```

2.3 Descriptor.c

2.3.1 HID Report Descriptor

HID_DigitizerReportDescriptor是此触控装置的HID报告描述元。此描述元定义了表 2中的数据格式。

字段的相关资料可以参考微软公司官方网站和USB开发者论坛(USB-IF)的HID Usage Table。

```
/*!<USB HID Report Descriptor */
const uint8_t HID_DigitizerReportDescriptor[] =
{
    0x05, 0x0D,         // USAGE_PAGE (Digitizers)
    0x09, 0x04,         // USAGE (Touch Screen)
    0xA1, 0x01,         // COLLECTION (Application)
    0x85, 0x01,         // REPORT_ID (1)
    0x09, 0x22,         // USAGE (Finger)

    0xA1, 0x02,         // COLLECTION (Logical)
    0x09, 0x42,         // USAGE (Tip Switch)
    0x15, 0x00,         // LOGICAL_MINIMUM (0)
    0x25, 0x01,         // LOGICAL_MAXIMUM (1)
    0x75, 0x01,         // REPORT_SIZE (1)
    0x95, 0x01,         // REPORT_COUNT (1)
    0x81, 0x02,         // INPUT (Data,Var,Abs)
    0x09, 0x32,         // USAGE (In Range)
    0x81, 0x02,         // INPUT (Data,Var,Abs)
    0x09, 0x47,         // USAGE (Touch Valid)
    0x81, 0x02,         // INPUT (Data,Var,Abs)
    0x95, 0x05,         // REPORT_COUNT (5)
    0x81, 0x03,         // INPUT (Cnst,Ary,Abs)
    0x75, 0x08,         // REPORT_SIZE (8)
    0x09, 0x51,         // USAGE (Contact Identifier)
    0x95, 0x01,         // REPORT_COUNT (1)
    0x81, 0x02,         // INPUT (Data,Var,Abs)
    0x05, 0x01,         // USAGE_PAGE (Generic Desktop)
    0x75, 0x10,         // REPORT_SIZE (16)
    0x55, 0x0E,         // UNIT_EXPONENT (-2)
    0x65, 0x11,         // UNIT (Centimeter)
    0x09, 0x30,         // USAGE (X)
    0x35, 0x00,         // PHYSICAL_MINIMUM (0)
    0x46, 0xE3, 0x13,   // PHYSICAL_MAXIMUM (0x13E3)
    0x26, 0x7F, 0x07,   // LOGICAL_MAXIMUM (0x77F)
    0x81, 0x02,         // INPUT (Data,Var,Abs)
    0x09, 0x31,         // USAGE (Y)
    0x46, 0x2F, 0x0B,   // PHYSICAL_MAXIMUM (0xB2F)
    0x26, 0x37, 0x04,   // LOGICAL_MAXIMUM (0x437)
    0x81, 0x02,         // INPUT (Data,Var,Abs)
    0xC0,               // END_COLLECTION

    0xA1, 0x02,         // COLLECTION (Logical)
    0x05, 0x0D,         // USAGE_PAGE (Digitizers)
    0x09, 0x42,         // USAGE (Tip Switch)
    0x15, 0x00,         // LOGICAL_MINIMUM (0)
    0x25, 0x01,         // LOGICAL_MAXIMUM (1)
    0x75, 0x01,         // REPORT_SIZE (1)
```

```
    0x95, 0x01,          // REPORT_COUNT (1)
    0x81, 0x02,          // INPUT (Data,Var,Abs)
    0x09, 0x32,          // USAGE (In Range)
    0x81, 0x02,          // INPUT (Data,Var,Abs)
    0x09, 0x47,          // USAGE (Touch Valid)
    0x81, 0x02,          // INPUT (Data,Var,Abs)
    0x95, 0x05,          // REPORT_COUNT (5)
    0x81, 0x03,          // INPUT (Cnst,Ary,Abs)
    0x75, 0x08,          // REPORT_SIZE (8)
    0x09, 0x51,          // USAGE (Contact Identifier)
    0x95, 0x01,          // REPORT_COUNT (1)
    0x81, 0x02,          // INPUT (Data,Var,Abs)
    0x05, 0x01,          // USAGE_PAGE (Generic Desktop)
    0x75, 0x10,          // REPORT_SIZE (16)
    0x55, 0x0E,          // UNIT_EXPONENT (-2)
    0x65, 0x11,          // UNIT (Centimeter)
    0x09, 0x30,          // USAGE (X)
    0x35, 0x00,          // PHYSICAL_MINIMUM (0)
    0x46, 0xE3, 0x13,    // PHYSICAL_MAXIMUM (0x13E3)
    0x26, 0x7F, 0x07,    // LOGICAL_MAXIMUM (0x77F)
    0x81, 0x02,          // INPUT (Data,Var,Abs)
    0x46, 0x2F, 0x0B,    // PHYSICAL_MAXIMUM (0xB2F)
    0x26, 0x37, 0x04,    // LOGICAL_MAXIMUM (0x437)
    0x09, 0x31,          // USAGE (Y)
    0x81, 0x02,          // INPUT (Data,Var,Abs)
    0xC0,                // END_COLLECTION

    0x05, 0x0D,          // USAGE_PAGE (Digitizers)
    0x09, 0x54,          // USAGE (Contact Count)
    0x15, 0x00,          // LOGICAL_MINIMUM (0)
    0x26, 0xFF, 0x00,        // LOGICAL_MAXIMUM (0x00FF)
    0x95, 0x01,          // REPORT_COUNT (1)
    0x75, 0x08,          // REPORT_SIZE (8)
    0x81, 0x02,          // INPUT (Data,Var,Abs)
    0x09, 0x55,          // USAGE (Contact Count Maximum)
    0x25, 0x02,          // LOGICAL_MAXIMUM (2)
    0x95, 0x01,          // REPORT_COUNT (1)
    0x85, 0x02,          // REPORT_ID (2)
    0xB1, 0x02,          // FEATURE (Data,Var,Abs)
    0xC0                 // END_COLLECTION
};
```

2.3.2 Configuration Descriptor

USB主机需要组态描述元(Configuration Descriptor)来辨识装置。gu8ConfigDescriptor是此范例使用的组态描述元，里面含有1组界面的描述。“bNumInterfaces”字段设定为1，让USB主机知道此装置有1组接口。界面描述元(Interface Descriptor)，HID描述元(HID descriptor)和端点描述元(Endpoint Descriptor)，依序列于组态描述元之后。

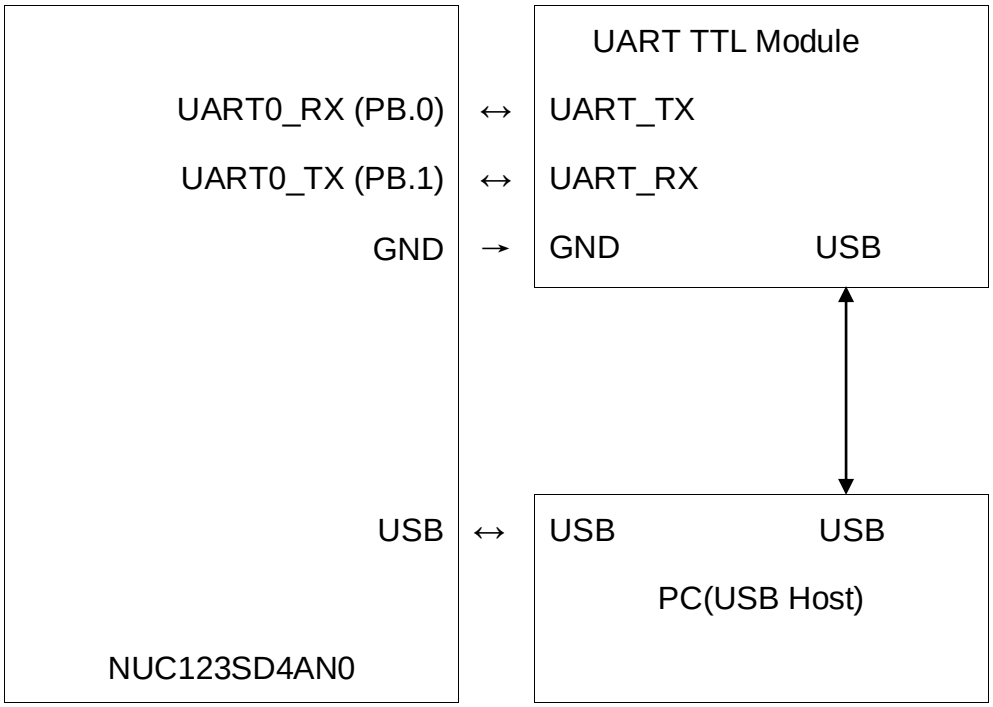
```
/*!<USB Configure Descriptor */
const uint8_t gu8ConfigDescriptor[] =
{
    LEN_CONFIG,     /* bLength */
    DESC_CONFIG,    /* bDescriptorType */
    /* wTotalLength */
    LEN_CONFIG_AND_SUBORDINATE & 0x00FF,
    ((LEN_CONFIG_AND_SUBORDINATE & 0xFF00) >> 8),
    0x01,           /* bNumInterfaces */
    0x01,           /* bConfigurationValue */
    0x00,           /* iConfiguration */
    0x80 | (USBD_SELF_POWERED << 6) | (USBD_REMOTE_WAKEUP << 5),/* bmAttributes */
    USBD_MAX_POWER, /* MaxPower */
```

```
    /* I/F descr: HID */
    LEN_INTERFACE,  /* bLength */
    DESC_INTERFACE, /* bDescriptorType */
    0x00,           /* bInterfaceNumber */
    0x00,           /* bAlternateSetting */
    0x01,           /* bNumEndpoints */
    0x03,           /* bInterfaceClass */
    0x00,           /* bInterfaceSubClass */
    HID_NONE,       /* bInterfaceProtocol */
    0x00,           /* iInterface */

    /* HID Descriptor */
    LEN_HID,        /* Size of this descriptor in UINT8s. */
    DESC_HID,       /* HID descriptor type. */
    0x10, 0x01,     /* HID Class Spec. release number. */
    0x00,           /* H/W target country. */
    0x01,           /* Number of HID class descriptors to follow. */
    DESC_HID_RPT,   /* Descriptor type. */
    /* Total length of report descriptor. */
    sizeof(HID_DigitizerReportDescriptor) & 0x00FF,
    ((sizeof(HID_DigitizerReportDescriptor) & 0xFF00) >> 8),

    /* EP Descriptor: interrupt in. */
    LEN_ENDPOINT,   /* bLength */
    DESC_ENDPOINT,  /* bDescriptorType */
    (INT_IN_EP_NUM | EP_INPUT), /* bEndpointAddress */
    EP_INT,         /* bmAttributes */
    /* wMaxPacketSize */
    EP2_MAX_PKT_SIZE & 0x00FF,
    ((EP2_MAX_PKT_SIZE & 0xFF00) >> 8),
    HID_DEFAULT_INT_IN_INTERVAL     /* bInterval */
};
```

3 软件与硬件环境

- 软件环境
  - BSP 版本
    - NUC123 Series BSP CMSIS V3.01.001
  - IDE 版本
    - Keil uVersion 5.26
  - USB 封包撷取工具
    - USBLyzer (http://www.usblyzer.com/)
- 硬件环境
  - 电路组件
    - NuTiny-EVB-NUC123-LQFP64 v1.0
    - USB mini USB cable
    - USB-UART TTL(Optional)
    - USB packet analyzer(Optional)

■ 示意图


UART TTL Module
UART0_RX (PB.0)
↔
UART_TX
UART0_TX (PB.1)
↔
UART_RX
GND
→
GND
USB
USB
↔
USB
USB
PC(USB Host)
NUC123SD4AN0

4 目录信息

- EC_NUC123_USBD_HID_Touch_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

5 如何执行范例程序

1. 根据目录信息章节进入 ExampleCode 路径中的 KEIL 文件夹，双击 NUC123_USBD_HID_Touch.uvproj。
2. 进入编译模式接口
   a. 编译
   b. 下载代码至内存
   c. 进入 / 离开除错模式
3. 进入除错模式接口
   a. 执行代码

6 修订纪录

Date	Revision	Description
Nov. 01, 2019	1.00	1. 初始发布.

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.