# 1导言

USB是一个复杂的协议，初学者很难快速启动并运行基于USB的应用程序。然而，USB的某些方面很容易使用，尤其是人机界面设备（HID）协议。HID是为普通PC接口设备（如键盘和鼠标）设计的，但可以适应许多自定义应用程序。大多数PC操作系统，包括Windows、Mac和Linux，都包含HID驱动程序。这意味着您不必编写驱动程序，而是可以专注于开发应用程序固件。

本应用说明向您展示了如何使用USB HID使用PSoC 3和PSoC 5LP进行简单的数据传输。PSoC设备包括专用全速（FS）12 Mbps USB 2.0外围设备，该外围设备使用内部振荡器–不需要晶体。PSoC Creator IDE和USB全速（USBFS）组件使快速构建应用程序成为可能。以鼠标和操纵杆为例。

本应用程序说明假定您熟悉使用PSoC Creator for PSoC 3或PSoC 5LP开发应用程序。如果您是这些产品的新手，请参阅AN54181，PSoC 3入门和AN77759，PSoC 5LP入门。如果您是PSoC Creator的新手，请参阅PSoC Creator主页。

如果您不熟悉USB，基本概念将在AN57294《USB 101：通用串行总线2.0简介》中解释。如果您熟悉USB基础知识，相关应用说明中会提供更高级的信息。

# 2 HID简介

人机界面设备（HID）每天都在我们的个人和职业生活中使用。它们是连接到计算机上的键盘和鼠标，用于玩视频游戏的游戏控制器，或用于绘图的数字笔/平板电脑。顾名思义，它是一种在人与计算机之间创建接口的设备。它还包括一些您通常认为不属于HID规范的设备，例如条形码扫描仪。图1显示了各种HID的示例。

图1。HIDs的常见例子



当您将任何USB设备插入主机PC时，PC会请求有关该设备的信息。这些信息以描述符表或描述符的形式呈现给PC。有许多类型的描述符；对HID来说，有两个因素很重要：

接口描述符：将USB设备定义为HID。

报告描述符：定义设备提供的数据的格式和用法。例如，USB鼠标报告X和Y移动和按钮活动的数据，其报告描述符定义该数据的结构和格式。

HID报告描述符的一个重要特性是，有数百种方法可以设置和组织它们来定义同一设备。报告描述符提供许多功能，例如：提供有关设备功能的信息、指定设备数据的组织、确定数据是否具有与其相关联的指数值（如10-3103）、指定与数据关联的单位。

根据设备的不同，报告描述符可以非常简单，也可以非常详细和复杂。然而，一旦您理解了描述符的基本要素，您就可以轻松地开发自己的描述符。

# 3报告描述符详细信息

报告描述符由许多项组成；项是一组不同的数据。项目可以是以下类型：主要的、全局和本地。

区分项目类型和项目本身很重要。在上面列出的三种项目类型下，有许多不同的项目。表1列出了本申请说明中讨论的项目。因为报表描述符可能变得非常复杂，所以本应用程序说明将重点放在更常见的项上。在USB HID规范和USB实施者论坛提供的HID使用表中可以找到许多其他项目(http://www.usb.org). 这两个文档都包含有关所讨论的项目类型和项目的更详细信息。

表1.常见项目

|  |  |  |
| --- | --- | --- |
| **项目类型** | **项目** | **项目类别** |
| 主要的 | 输入 |  |
| 输出 |
| 功能 |
| 集合 | 物理 |
| 应用 |
| 逻辑 |
| 全局的 | 使用页面 |  |
| 逻辑最大值和最小值 |
| 报告计数 |  |
| 报告大小 |
| 本地的 | 用法 |  |
| 最大和最小使用量 |

## **3.1主要项目**

主要项目用于定义数据包含的内容或将数据分组。主要类型有五种不同的项目： 输入、输出、功能、集合、结束集合。

输入、输出和功能类型用于定义项，而集合和结束集合用于分组。输入项是指设备发送到主机的数据（如鼠标按钮单击），而输出项是指主机发送到设备的数据（如键盘上的大写锁定指示灯）。

功能项是主机既可以发送到设备，也可以从设备读取的信息。功能项包含设备配置信息（更改此信息会更改设备的行为）。功能报告通常与PC接口应用程序一起使用，其中单击GUI上的按钮可更改设备的操作。

集合项目用于将输入、输出和功能项目组合在一起。在HID规范中，有三种不同类型的预定义集合：物理、应用、逻辑。

物理集合由在一个几何点集合的数据组成。同时从多个传感器收集数据的设备可以将数据分组到物理集合中。使用此集合的设备的一个示例是将按钮和位置数据分组在一起的鼠标。

应用程序集合将在单个设备中用于不同或通用目的的项组合在一起。例如，将LED和按键组合在一起的键盘。

逻辑集合将各种项目类型的数据组合在一起，形成结构化数据集合。例如，数据缓冲区的内容与缓冲数据占用的字节数之间的关系。逻辑集合在两者之间创建连接。

## **3.2全局和本地项目**

除了具有项目类型外，项目还可以是全局的或本地的。第7页上的图5显示了一个报告描述符，其中包含使用情况、使用情况页面、逻辑最小值/最大值等项。这些项目是全局和本地项目。全局项描述数据，例如其限制、单位、位大小和计数。本地项定义数据参数，例如主机使用数据的目的。以下是更常见的本地和全局项的简要说明。

### **3.2.1全球项目**

**使用页面：**一种32位值，用于标识设备执行的功能，如控制设备或游戏控制器。高16位是全局使用页面项，低16位是本地使用项。本节后面将介绍使用项目。全局使用情况页面的上16位设置可用使用情况项的类型。例如，如果选择“通用桌面”作为使用页面，则可以包括鼠标、操纵杆或键盘等使用项。如果选择了“运动控制使用”页面，则可以选择使用项目，如高尔夫俱乐部或跑步机。

**逻辑最大值和最小值：**确定数组或变量中报告值的限制。例如，报告从-127到127的位置值的鼠标的逻辑最大值为127，逻辑最小值为-127。另一个示例是断言或释放的单个状态按钮，因此逻辑最小值为0，逻辑最大值为1。

**报告计数：**指定输入、输出或要素项中的数据字段数。

**报告大小：**确定输入、输出或要素项中字段的大小（以位为单位）。

### **3.2.2本地项目**

**用法：**32位值中较低的16位，用于标识设备执行的功能；请参阅上面的用法页面。可从USB实施者论坛获得的USB HID使用表文档列出并描述了HID设备的使用值或ID。例如，HID使用表列出了鼠标按钮、滑块、键盘按钮等的16位值。

输入、输出和功能项将在此部分中定义。例如，将使用通用桌面的全局使用页面定义鼠标，后跟使用项鼠标，然后是一系列if输入项，这些输入项将报告按钮按下和位置。

**最低和最高使用量：**用于为数组或位图中的数据分配一系列使用标识。最小使用量定义起始ID，最大使用量定义结束ID。例如，在三按钮鼠标中，将ID分配给按钮1（0x01）、按钮2（0x02）和按钮3（0x03）。

使用最小值和最大值条目允许在将使用分配给连续数字时一起定义一系列类似的项。例如，不是为标准键盘上的每个键分别定义所有101个用法，而是可以将最小和最大键ID设置为最小和最大键ID来定义单个用法。此用法的输入项现在可以报告键盘上的任何活动键，而不仅仅是一个。

## **3.3定义项目**

我们将以一个主要项目的定义为例进行探讨，如图2所示。项目定义由1字节前缀和1个或多个数据描述字节组成。前缀的高6位用于定义项，例如输入或集合。前缀中较低的2位表示前缀后面有多少字节。数据描述字节可以进一步对项目进行分类，并描述项目将包含的数据的类型和格式。

图2：主项定义模板

|  |  |  |  |
| --- | --- | --- | --- |
| 前缀 | 数据定义 | 可选字节（如果缓冲数据 | 保留的 |
| 8位  nnnnnnnnnn | 8位  nnnnnnx | 选定）  8位  XXXXXXX 1 | 23位  XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX |

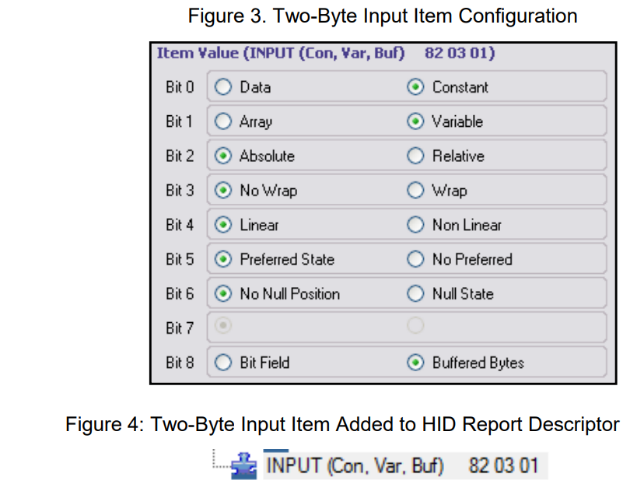
输入：100000nn 输出：100100nn 功能：101100nn

输入、输出和功能项由2或3字节字符串定义，该字符串描述传输的数据类型。每个项目都有一个8位前缀，PSoC Creator会为您设置该前缀。输入项的前缀为“10000NN”，其中nn是要跟随的字节数“100100nn”表示输出项，“101100nn”表示功能项。

前缀后面最多有9个附加位，您必须配置这些位以进一步定义项目数据。除了这9位之外，还有23位是当前保留和不可访问的。对于附带的鼠标示例中使用的输入项，前缀后面只有8位（1字节）。因此，输入项的前缀为10000001b或0x81，如图9、图10和图11所示。

如果用户选择位8中的“缓冲字节”，则需要第二个字节，这将导致该位的值为1。然后前缀的nn变为10b，并添加一个额外字节以容纳额外的数据位。图3和图4显示了一个2字节输入项的示例。

让我们仔细看看这个例子来帮助理解前缀和它后面的位是如何组织的。请注意，图3显示了一个输入。这意味着我们有一个前缀值“10000nn”。请注意，根据配置，位8设置为“1”，这意味着前缀后面有两个字节。因此，最终前缀值为“1000010”或0x82。在前缀之后，位0、位1和位8被设置为值“1”，给出值0x03和0x01。将所有这些值放在一起，得到0x82、0x03和0x01，如图3顶部所示。

**3.4输入、输出和功能位字段**

让我们检查前缀字节后面的9位，如图3所示，以及每个位如何设置输入、输出和功能项。每个位最初默认为“0”。有关这些设置的更多详细信息，请参阅USB HID设备类定义中的输入、输出和功能项部分。

**数据与常数：**数据意味着设备数据是读/写的。常量表示数据为只读，主机无法修改。

**数组与变量：**数组意味着只报告当前处于活动状态的控件，例如正在按下的按钮。变量表示报告的设备数据只是每个控件的当前状态

**绝对与相对：**绝对表示值基于固定原点。相对是指这些值表示上次读取数据的变化。鼠标是一种提供相对数据的设备，而操纵杆或平板电脑则提供绝对数据。

**不换行与换行：**不换行表示超过设定限制的值报告超出限制的值。换行意味着，如果该值超过设定的最大值，则会滚动到设定的最小值。相反，如果值低于最小值，则该值将滚动到最大值。这在使用数组时不适用。

**线性与非线性：**线性是指测量的数据与报告的值之间具有线性关系。这在使用数组时不适用。传感器数据中的非线性曲线是何时使用非线性的一个示例。

**首选状态与非首选状态：**首选状态是指当用户不与控件进行物理交互时，控件将返回某个特定状态。“非首选”表示一旦用户不再与控件进行物理交互，控件将保持在特定状态。返回中间位置的操纵杆和按钮是首选状态控制的示例。切换开关或滑块是没有首选状态的控件的示例。

**无空位置与空状态：**无空位置表示控件发送的所有数据都有意义。Null状态意味着控件可以发送由设置的逻辑最小值和最大值范围之外的值表示的无意义数据。

**非易失性与易失性：**非易失性意味着设备仅通过主机交互改变值。Volatile表示设备能够在未经主机批准的情况下更改值。此位仅适用于输出和功能项报告数据。这在使用数组时不适用。

**位字段与缓冲字节：**位字段意味着字节中的每一位都可以表示一段特定的数据。缓冲字节表示数据由一个或多个字节表示。这在使用数组时不适用。鼠标或键盘是使用位字段而条形码阅读器使用缓冲字节的设备示例。

## **3.5报告描述符详细信息摘要**

报告描述符至少必须包含以下项。其他项目提供了进一步的细节。类型（输入、输出或特征）、用法、使用页面、逻辑最小值和最大值、报告大小、报告计数。

至此，您已经了解了HID设备的基本概念以及如何在报告描述符中定义它。现在让我们来看一个实际HID设备的示例报告描述符。

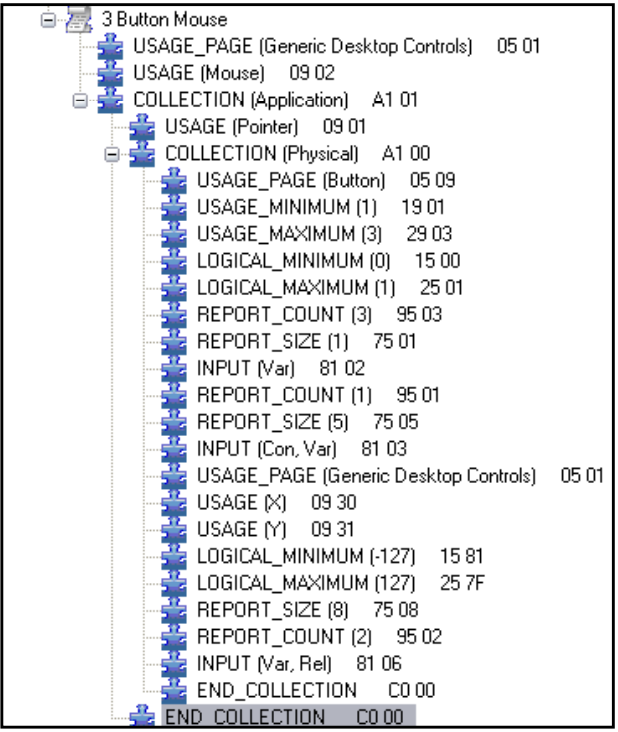
## **3.6鼠标的报告描述符**

理解HID报告描述符最简单的方法是对其进行分解和研究。本应用程序说明附带的一个示例项目创建了一个简单的三键鼠标。

3按钮鼠标的HID报告必须包含以下信息：第一，3个按钮的按钮状态。状态将报告为打开或关闭。第二，X轴和Y轴发生变化。报告了X轴和Y轴位置的相对变化。报告中的数据如下表所示：

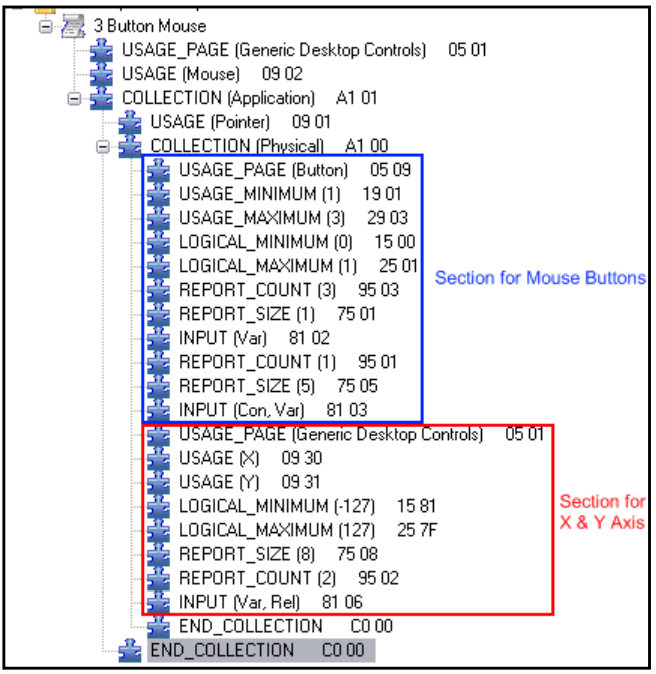
|  |  |  |  |
| --- | --- | --- | --- |
| **字节** | 0 | 1 | 2 |
| **数据** | 按钮 | X轴 | Y轴 |
| **第7位** | 0 | X轴  位置  改变 | Y轴  位置  改变 |
| **第6位** | 0 |
| **第5位** | 0 |
| **第4位** | 0 |
| **第3位** | 0 |
| **第2位** | 按钮3 |
| **第1位** | 按钮2 |
| **第0位** | 按钮1 |

图5显示了该项目的HID报告描述符。

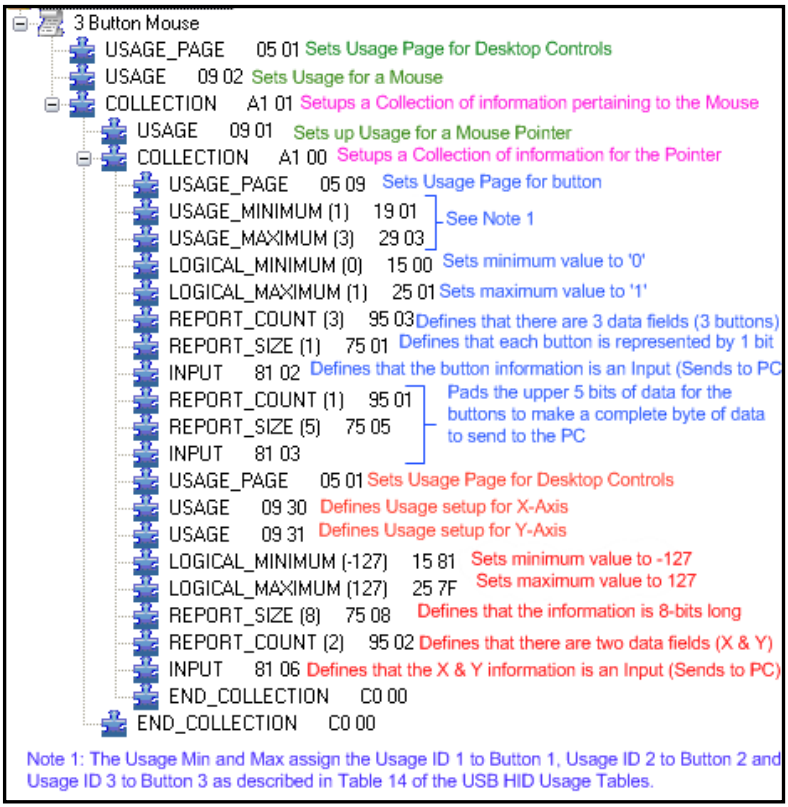


虽然报表描述符看起来很复杂，但实际上非常简单。让我们先把它分成几个部分。

鼠标根据光标移动和按钮生成数据。光标有X和Y值，每个按钮都有按下/释放状态信息。因此，报告描述符有一个集合类型，其中一个部分用于按钮，另一个部分用于光标，如图6所示。



下一步是查看报表描述符中每一行的函数。通过了解每一行如何影响描述符，了解该行是否与按钮或光标移动有关，以及它对数据的影响，您将进一步了解报表描述符。图7显示了HID鼠标报告描述符的注释版本，详细说明了每行条目的作用。



HID报告描述符有多个输入项。每个输入项都是一个数据位字段，告诉主机将要从鼠标发送的数据。图8突出显示了报表描述符中的输入项。图8。报表描述符输入项

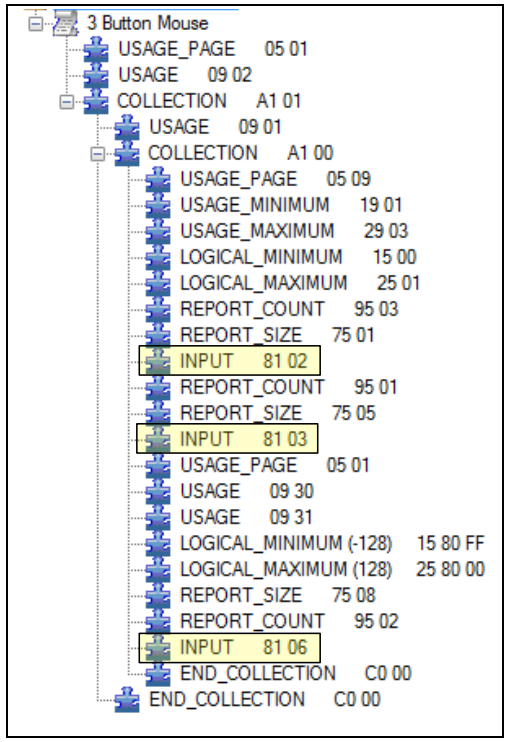
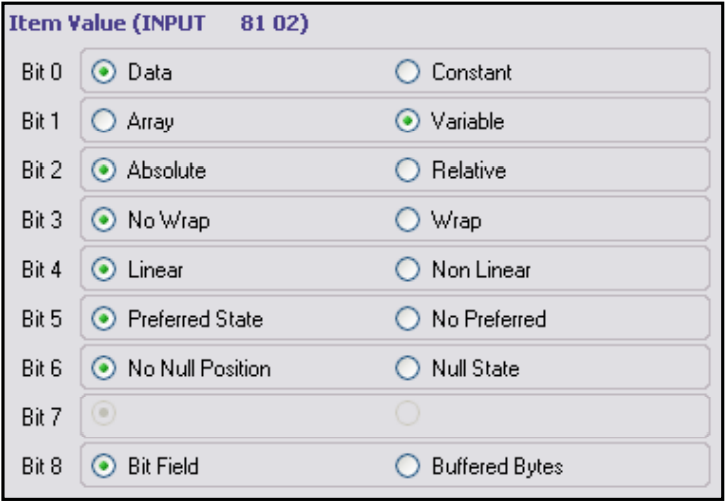


图9、图10和图11显示了如何配置输入项以及如何设置位字段。本节详细讨论每个设置/位代表的内容。在到主机的每个数据事务中，将传输三个字节的数据。一个字节是鼠标按钮数据，另一个是x轴数据，第三个字节是y轴数据。

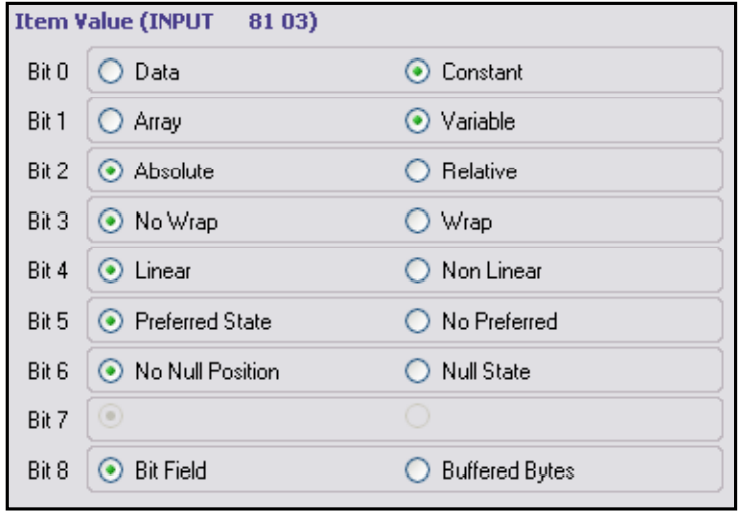
按钮数据以位字段格式组织，其中每个位对应于鼠标上的特定按钮。

图9显示了三按钮鼠标上三位按钮数据的输入项设置。

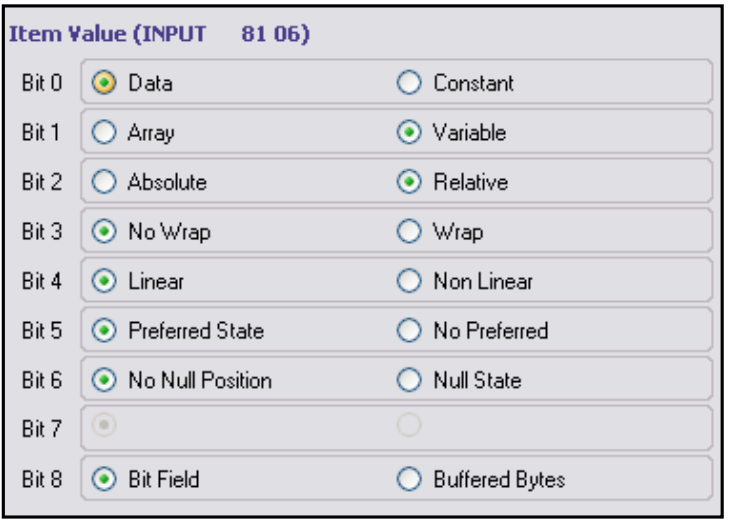
每个位代表一个特定的按钮：位0=中间按钮、第1位=右按钮、第2位=左按钮。



由于一个字节中只有三位用于实际数据，我们需要保留未使用的五位。要做到这一点，我们使用一个单独的输入项将上面的五位填入零。因为我们不希望数据改变，所以我们使用与按钮相同的输入项设置，只是我们将位0设置为常量，如图10所示。图10。用于填充鼠标按钮的输入项



剩下的输入项是X和Y信息，如图11所示。因为位2设置为相对，所以数据显示X和Y的变化量。然后，主机相应地移动鼠标光标。图11。X轴和Y轴的输入项



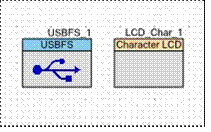
现在，您已经了解了报表描述符组件和报表描述符的组成，让我们来看两个项目，它们演示了如何配置PSoC 3或PSoC 5LP设备，使其充当HID鼠标和HID操纵杆。

# 4示例项目1：HID鼠标

以下项目描述假定您熟悉使用PSoC Creator for PSoC 3或PSoC 5LP开发应用程序。如果您是这些产品的新手，可以在AN54181《PSoC 3入门》和AN77759《PSoC 5LP入门》中找到介绍。如果您是PSoC Creator的新手，请参阅PSoC Creator主页。

首先，打开PSoC Creator并创建一个名为“MyFirstHID”的项目。然后将一个USBFS组件和一个字符LCD组件放在原理图输入页面（TopDesign.cysch）中，如图12所示：

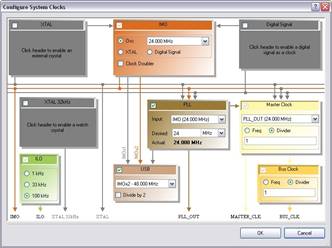
图12。用于鼠标的PSoC Creator组件



接下来，打开文件MyFirstHID.cydwr。单击时钟选项卡，然后双击其中一个时钟以打开时钟配置窗口。

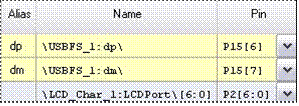
对于USB操作，设备的USB块需要48 MHz的时钟，并且ILO必须设置为100 kHz。要获得48 MHz的USB时钟，请将IMO设置为24 MHz，并将USB时钟设置为IMOx2。调整PLL、USB、IMO和ILO设置，如图13所示。

图13。时钟配置窗口



接下来，选择MyFirstHID.cydwr中的Pins选项卡，并将组件管脚分配给设备管脚，如图14所示。请注意，USB D+和D-引脚会自动分配给P15[6]和P15[7]。

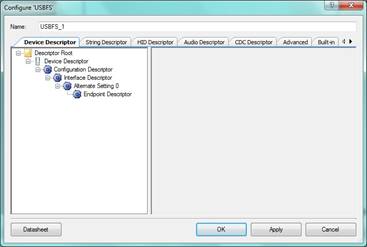
图14。设备引脚分配



下一步是使用USB配置向导配置USB组件（请参阅USBFS组件数据表）。USBFS向导配置所有设备描述符。构建项目时，USB向导将使用此信息创建适当的描述符表，以便与您的PC进行交互。

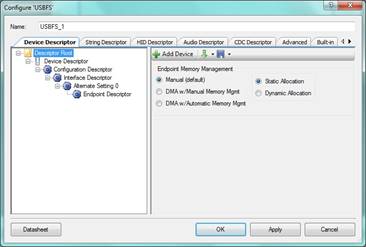
要打开配置向导（如图15所示），双击USBFS组件。

图15。USBFS配置向导



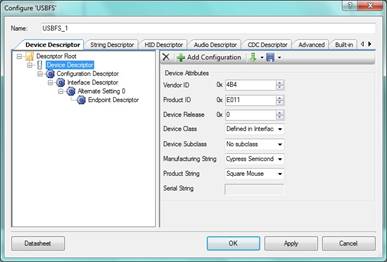
向导打开后，首先单击描述符根打开端点内存管理的配置选项。确保端点内存管理配置为手动静态分配，如图16所示。默认情况下，应将其配置为这样。

图16。USB端点内存管理



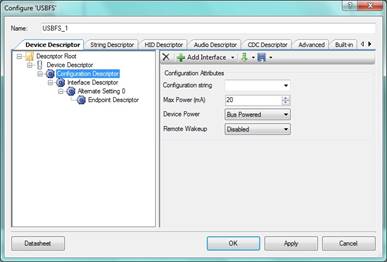
接下来，单击设备描述符开始配置。对于此应用程序，选择了供应商ID（VID）和产品ID（PID）。如果您决定创建自己的USB应用程序来制造和销售，则必须从USB实施者论坛获得视频。PID由设计者选择。调整设备描述符配置，使其与图17匹配。

图17。USB设备描述符设置



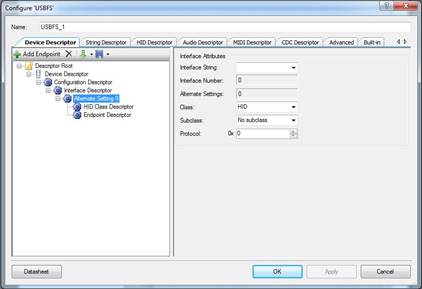
接下来，选择配置描述符和配置选项，如图18所示。由于该项目采用总线供电，我们必须限制可提供给设备的最大电流。指定适合设备的值非常重要。这是因为根据USB 2.0规范，每个集线器都有100 mA/500 mA的余量。对于这种应用，20 mA就足够了。

图18。USB配置描述符设置



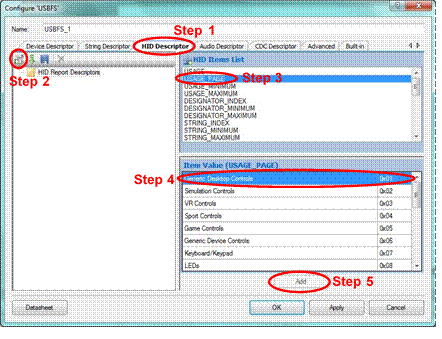
下一步是配置接口描述符。要执行此操作，请单击备用设置0。在此菜单中，只需将类类型设置为HID。当PSoC Creator生成报告描述符时，将设置必要的参数，以通知主机连接的设备是HID。调整接口描述符中的配置选项，使其与图19匹配。请注意，将类字段设置为HID会导致向导中出现一个称为HID类描述符的新描述符。

图19。USB接口描述符设置



HID报告描述符必须与接口相关联。为此，创建HID报告描述符-单击HID描述符选项卡。此对话框的步骤如图20所示。

图20。USB HID描述符设置



**步骤1：**单击HID描述符选项卡

**步骤2：**单击添加报告按钮

**步骤3：**从HID项目列表中选择一个项目。首先选择USAGE\_页面。

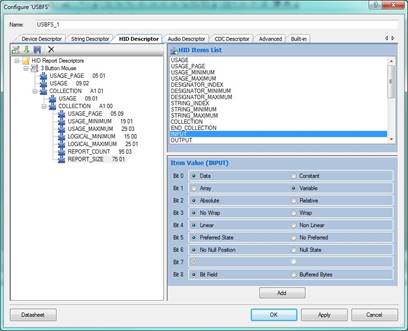
**步骤4：**从所选项目的项目值列表中选择一个值，在本例中为通用桌面控件。

**步骤5：**单击Add按钮将该项添加到报告描述符中。

完整HID报告描述符的格式如第7页图5所示。重复步骤3到步骤5，也如图20所示，直到报告描述符类似于第7页的图5。

添加和配置输入项时，将显示一个窗口来配置位字段，而不是项值字段，如图21所示。

图21。在报表描述符中配置输入



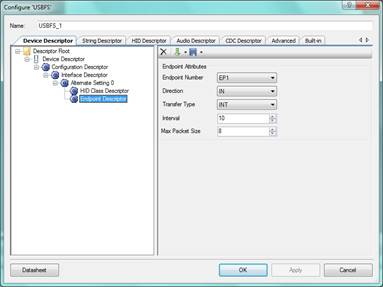
完成报告描述符后，转到HID类描述符菜单，如图22所示。将HID报告设置为刚创建的HID报告的名称。在这种情况下，将其设置为3键鼠标。

图22。USB设备属性设置



接下来，在USBFS组件的配置对话框中，选择端点描述符，并在端点属性下，选择要进入的方向和要INT的传输类型，如图23所示。

图23。USB端点设置



将以下代码放在main.c文件中。请注意，开始使用USB只需要5行代码。剩下的代码通过周期性地发送指示鼠标已移动的数据来模拟鼠标功能。有关更详细的注释，请参见附带的项目文件。

#include <device.h>

static int8 Mouse\_Data[3] = {0, 0, 0}; static uint16 i = 0;

void main()

{

CYGlobalIntEnable;

/\* Activates and configs the USB component \*/

USBFS\_1\_Start(0, USBFS\_1\_3V\_OPERATION); /\* Waits for configuration data from host \*/ while(!USBFS\_1\_bGetConfiguration()); /\* Begins initial communication with PC \*/ USBFS\_1\_LoadInEP(1,(uint8 \*)Mouse\_Data,3);

LCD\_Char\_1\_Start();

LCD\_Char\_1\_Position(0,0);

LCD\_Char\_1\_PrintString(" My First HID ");

for(;;)

{

/\* Waits for ACK from the host \*/ while(!USBFS\_1\_bGetEPAckState(1));

/\* Loads EP1 for a IN transfer to PC \*/ USBFS\_1\_LoadInEP(1,(uint8 \*)Mouse\_Data, 3);

switch (i){ case 128:

Mouse\_Data[1] = 5;

Mouse\_Data[2] = 0;

LCD\_Char\_1\_Position(1, 0);

LCD\_Char\_1\_PrintString("Mouse Right"); break;

case 256:

Mouse\_Data[1] = 0;

Mouse\_Data[2] = 5;

LCD\_Char\_1\_Position(1, 0);

LCD\_Char\_1\_PrintString("Mouse Down "); break;

case 384:

Mouse\_Data[1] = -5;

Mouse\_Data[2] = 0;

LCD\_Char\_1\_Position(1, 0);

LCD\_Char\_1\_PrintString("Mouse Left "); break;

case 512:

Mouse\_Data[1] = 0;

Mouse\_Data[2] = -5;

LCD\_Char\_1\_Position(1, 0);

LCD\_Char\_1\_PrintString("Mouse Up "); i = 0; break;

default: break;

} i++;

}

}

正确启用所有设置并将代码输入main.c后，最后一步是在CY8CKIT-001开发工具包上编程PSoC 3/PSoC 5LP。在对设备进行编程之前，将J8上的跳线从VREG移动到VBUS，并将SW3设置为3.3 V。因为这是一个HID设备，我们让USB总线为设备供电。J8标记在图24的照片中。

设备编程完成后，按下DVK板上的重置按钮（标记为SW4），您将看到LCD上显示的文本和屏幕上的光标以方形模式移动。祝贺您创建了第一台HID设备！为该项目生成的描述符表位于附录1中。

图24。CY8CKIT-001演示

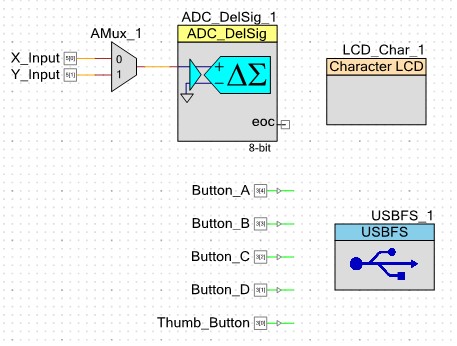


# 5示例项目2：HID操纵杆

下一个项目稍微复杂一些。对于这个项目，我们使用电位计操纵杆和一些按钮构建了一个演示板。要使用CY8CKIT-001模拟操纵杆，一个轴使用“VR”，另一个轴使用另一个电位计。

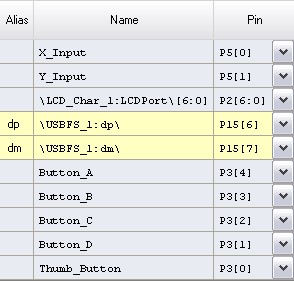
首先，创建一个新的psoccreator项目，并将其命名为“hidcoogle”。将以下组件放入schematic条目页面（TopDesign.cysch），如图25所示：Delta Sigma模数转换器、模拟多路复用器、字符液晶显示器、USBFS、（2）模拟输入引脚、（5）数字输入引脚

图25。操纵杆的PSoC Creator组件



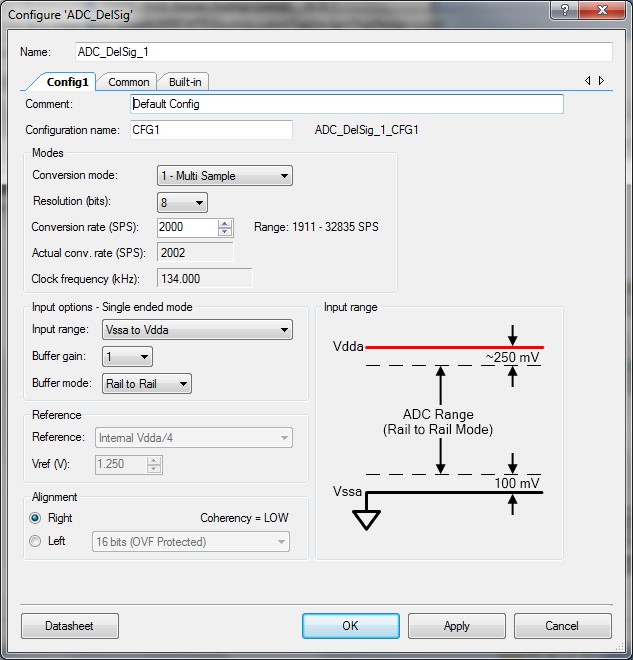
配置数字输入引脚：双击每个引脚组件，取消选中硬件连接框，并将驱动模式更改为电阻下拉。

使用与上一个项目相同的时钟配置（参见第11页的图13）。USB和LCD引脚配置也相同，如图26所示。您可以分配ADC输入引脚和按钮引脚，如图26所示，也可以将它们分配给您选择的GPIO引脚。

图26。PSoC创建者Pin信息（操纵杆）

ADC\_DelSig的配置设置如图27所示。虽然转换率可能会发生变化，但重要的是保持其他参数不变，因为固件设计用于根据所示设置处理ADC结果。

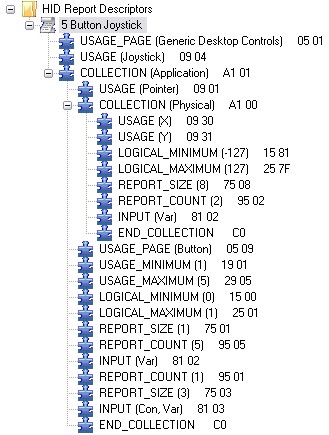
图27。DelSig ADC设置



下一步是配置USB设置和HID报告描述符。按照上一个项目中讨论的相同USB配置步骤开始。请确保此项目的PID与上一个项目中的PID不同，否则项目可能无法正常运行。这是因为主机将上一个VID/PID识别为鼠标。也可以将产品字符串更改为与操纵杆相关的名称。完成后，继续为项目创建报告描述符。报告描述符如图28所示。

描述符布局的格式与鼠标报告描述符非常相似，但有一些不同。第一个区别是，通用桌面控件的用法设置为操纵杆而不是鼠标。另一个明显的区别是，与鼠标的报告描述符（指定三个按钮）不同，此报告指定五个按钮。一个按钮位于操纵手柄下方，另外四个按钮位于操纵手柄上。与鼠标报告描述符类似，包含按钮数据的单个字节中的剩余位被填充。

图28。操纵杆HID描述符报告



如前所述，为该示例项目构建了一个演示板，如图29和图30所示。电路板的示意图如图31所示。如果您没有基于电位计的操纵杆，与CY8CKIT-001开发工具包附带的电位计类似的基本电位计可以与板上的按钮一起使用。通过在Vddd和GND之间添加一个按钮和一个电阻器，可以创建额外的开关。

图29。CY8CKIT-001操纵杆演示设置



Figure

30

. Close

-

up of Joystick Expansion Board

Figure

31

. Schematic of Joystick Board



图32显示了使用CY8CKIT-001的模拟操纵杆示例。五个跳线用作按钮，而开发套件上标记为“VR”的电位计用作操纵杆。请注意，电位计驱动X轴和Y轴输入，因此操纵手柄以对角方式移动。

Figure

32

.

DVK

Implementation of Joystick



最后，将以下代码添加到main.c。在鼠标项目中，5行代码被确定为启用USB功能的核心代码。这里也使用相同的5行代码。代码的其余部分用于获取信息并准备发送给主机。要查看有关代码的详细注释，请参阅本应用说明附带的项目。

#include <device.h>

void StartUp (void); void ReadJoystick (void); void ReadButtons (void);

static uint16 X\_Axis=0, Y\_Axis=0; static int16 X\_Data, Y\_Data; static int8 Joystick\_Data[3] = {0, 0, 0}; static unsigned char Buttons;

void main() {

StartUp();

for(;;)

{

/\* Waits for ACK from the host \*/ while(!USBFS\_1\_bGetEPAckState(1));

ReadJoystick();

ReadButtons();

/\* Delay added to periodically send data \*/

CyDelay(10);

Joystick\_Data[0] = X\_Data;

Joystick\_Data[1] = Y\_Data; Joystick\_Data[2] = Buttons;

/\* Loads EP1 for a IN transfer to PC \*/

USBFS\_1\_LoadInEP(

1, (uint8 \*)Joystick\_Data, 3);

}

}

void StartUp (void)

{

CYGlobalIntEnable;

ADC\_DelSig\_1\_Start();

CyIntEnable(ADC\_DelSig\_1\_IRQ\_\_INTC\_NUMBER);

ADC\_DelSig\_1\_StartConvert();

AMux\_1\_Start();

AMux\_1\_Select(0);

LCD\_Char\_1\_Start();

LCD\_Char\_1\_Position(0,0);

LCD\_Char\_1\_PrintString("PSoC 3 USB HID");

LCD\_Char\_1\_Position(1,0);

LCD\_Char\_1\_PrintString("Joystick Demo");

/\* Activates and configs the USB component \*/

USBFS\_1\_Start(0, USBFS\_1\_3V\_OPERATION); /\* Waits for configuration data from host \*/ while(!USBFS\_1\_bGetConfiguration()); /\* Begins initial communication with PC \*/

USBFS\_1\_LoadInEP(

1, (uint8 \*)Joystick\_Data, 3); } void ReadJoystick (void)

{

AMux\_1\_Select(0);

CyDelay(1);

ADC\_DelSig\_1\_IsEndConversion(

ADC\_DelSig\_1\_WAIT\_FOR\_RESULT);

X\_Axis = ADC\_DelSig\_1\_GetResult16();

AMux\_1\_Select(1);

CyDelay(1);

ADC\_DelSig\_1\_IsEndConversion(

ADC\_DelSig\_1\_WAIT\_FOR\_RESULT);

Y\_Axis = ADC\_DelSig\_1\_GetResult16();

X\_Data = X\_Axis - 127;

Y\_Data = Y\_Axis - 127;

if(X\_Data > 127) X\_Data = 127; if(Y\_Data > 127) Y\_Data = 127; if(X\_Data < -127) X\_Data = -127; if(Y\_Data< -127) Y\_Data = -127;

Y\_Data = Y\_Data \* -1;

}

void ReadButtons (void)

{

if(Thumb\_Button\_Read() != 0)

Buttons |= 0x01; else

Buttons &= ~0x01;

if(Button\_A\_Read() != 0) Buttons |= 0x02; else

Buttons &= ~0x02;

if(Button\_B\_Read() != 0) Buttons |= 0x04; else

Buttons &= ~0x04;

if(Button\_C\_Read() != 0) Buttons |= 0x08; else

Buttons &= ~0x08;

if(Button\_D\_Read() != 0) Buttons |= 0x10; else

Buttons &= ~0x10;

}

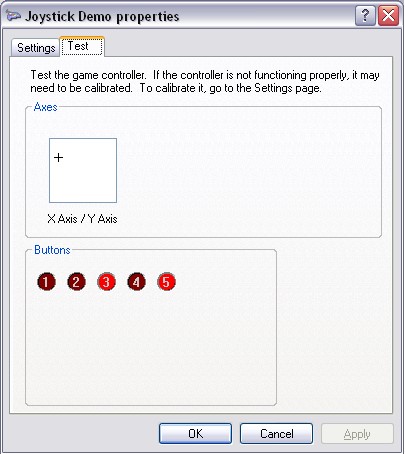
正确启用所有设置并将代码放入main.c后，最后一步是在CY8CKIT-001开发工具包上编程PSoC 3或PSoC 5LP。与项目1类似，确保在对设备进行编程之前，将J8上的跳线从VREG移动到VBUS，并将SW3设置为5.0 V。对设备进行编程后，按下电路板上的复位按钮，该按钮标记为SW4。

如何测试和验证操纵杆取决于您使用的电脑操作系统：

## **5.1在Windows XP中进行测试**

在电脑上，打开Windows控制面板>游戏控制器。然后选择连接的操纵杆，该操纵杆应标记为“操纵杆演示”。单击“属性”，然后单击“测试选项卡”。移动操纵杆并按下按钮。您在屏幕上看到的结果如图33所示。

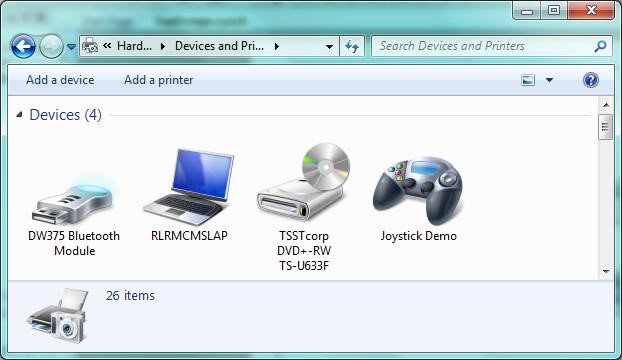
图33。Windows控制面板中的操纵杆测试



## **5.2在Windows Vista和Windows 7中进行测试**

打开Windows控制面板，导航到Hardware and Sound>Devices and Printers，并在Devices类别下找到操纵杆演示，如图34所示。

图34。在设备和打印机中定位操纵杆



右键单击“游戏杆演示”并选择“游戏控制器设置”。单击“属性”，然后单击“测试选项卡”。继续测试操纵手柄，如第24页图33所示。

另一种测试操纵杆的方法是打开并玩你最喜欢的视频游戏。

为该项目生成的描述符表位于附录2中。

## **5.3在Windows 10中进行测试**

在Windows 10开始菜单中搜索“USB”。在结果中，您将在“设置”标题下看到“设置USB游戏控制器”选项。单击此操作以开始操纵手柄的配置。图35。在Windows 10中配置操纵杆



出现配置窗口后，选择“操纵杆演示”并单击“属性”按钮。单击“测试”选项卡并测试操纵杆，如第24页图33所示。

# 6摘要

本应用说明说明了如何创建输入HID设备。您可以将这里介绍的原理应用于各种HID输入设备。

要向PSoC 3或PSoC 5LP项目添加更多HID功能，您可以支持输出事务和复合设备–请参阅更高级的AN58726。

您可以超越实现HID设备，在PSoC和您的PC之间传输任何类型的数据。要使用HID接口来实现这一点，请参阅AN82072。要将任何USB传输类型用于相同目的，请参阅AN56377。

# 项目1的描述符表

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Device Descriptors

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

/\* Descriptor Length \*/ 0x12u,

/\* DescriptorType: DEVICE \*/ 0x01u,

/\* bcdUSB (ver 2.0) \*/ 0x00u, 0x02u,

/\* bDeviceClass \*/ 0x00u,

/\* bDeviceSubClass \*/ 0x00u,

/\* bDeviceProtocol \*/ 0x00u,

/\* bMaxPacketSize0 \*/ 0x08u,

/\* idVendor \*/ 0xB4u, 0x04u,

/\* idProduct \*/ 0x00u, 0x10u,

/\* bcdDevice \*/ 0x00u, 0x00u,

/\* iManufacturer \*/ 0x01u,

/\* iProduct \*/ 0x02u,

/\* iSerialNumber \*/ 0x00u, /\* bNumConfigurations \*/ 0x01u

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Config Descriptor \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* Config Descriptor Length \*/ 0x09u,

/\* DescriptorType: CONFIG \*/ 0x02u,

/\* wTotalLength \*/ 0x22u, 0x00u,

/\* bNumInterfaces \*/ 0x01u,

/\* bConfigurationValue \*/ 0x01u,

/\* iConfiguration \*/ 0x00u,

/\* bmAttributes \*/ 0x80u,

/\* bMaxPower \*/ 0x0Au,

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Interface Descriptor

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* Interface Descriptor Length \*/ 0x09u,

/\* DescriptorType: INTERFACE \*/ 0x04u,

/\* bInterfaceNumber \*/ 0x00u,

/\* bAlternateSetting \*/ 0x00u,

/\* bNumEndpoints \*/ 0x01u,

/\* bInterfaceClass \*/ 0x03u,

/\* bInterfaceSubClass \*/ 0x00u,

/\* bInterfaceProtocol \*/ 0x00u,

/\* iInterface \*/ 0x00u,

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

HID Class Descriptor

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* HID Class Descriptor Length \*/ 0x09u,

/\* DescriptorType: HID\_CLASS \*/ 0x21u,

/\* bcdHID \*/ 0x11u, 0x01u,

/\* bCountryCode \*/ 0x00u,

/\* bNumDescriptors \*/ 0x01u,

/\* bDescriptorType \*/ 0x22u,

/\* wDescriptorLength (LSB) \*/ USBFS\_1\_HID\_RPT\_1\_SIZE\_LSB, /\* wDescriptorLength (MSB) \*/ USBFS\_1\_HID\_RPT\_1\_SIZE\_MSB,

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Endpoint Descriptor \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* Endpoint Descriptor Length \*/ 0x07u,

/\* DescriptorType: ENDPOINT \*/ 0x05u,

/\* bEndpointAddress \*/ 0x81u,

/\* bmAttributes \*/ 0x03u,

/\* wMaxPacketSize \*/ 0x08u, 0x00u,

/\* bInterval \*/ 0x0Au

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

String Descriptor Table

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Language ID Descriptor

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* Descriptor Length \*/ 0x04u,

/\* DescriptorType: STRING \*/ 0x03u,

/\* Language Id \*/ 0x09u, 0x04u,

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

String Descriptor: "Cypress Semiconductor"

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* Descriptor Length \*/ 0x2Cu,

/\* DescriptorType: STRING \*/ 0x03u,

'C', 0,'y', 0,'p', 0,'r', 0,'e', 0,'s', 0,'s', 0,' ', 0,'S', 0,'e', 0

,'m', 0,'i', 0,'c', 0,'o', 0,'n', 0,'d', 0,'u', 0,'c', 0,'t', 0,'o', 0

,'r', 0,

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

String Descriptor: "Square Mouse"

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* Descriptor Length \*/ 0x1Au,

/\* DescriptorType: STRING \*/ 0x03u,

'S', 0,'q', 0,'u', 0,'a', 0,'r', 0,'e', 0,' ', 0,'M', 0,'o', 0,'u', 0

,'s', 0,'e', 0,

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Serial Number String Descriptor

/\* Descriptor Length \*/ 0x02u, /\* DescriptorType: STRING \*/ 0x03u

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

HID Report Descriptor: 3 Button Mouse

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* Descriptor Size (Not part of descriptor)\*/ 0x32u, 0x00u,

/\* USAGE\_PAGE \*/ 0x05u, 0x01u,

/\* USAGE \*/ 0x09u, 0x02u,

/\* COLLECTION \*/ 0xA1u, 0x01u,

/\* USAGE \*/ 0x09u, 0x01u,

/\* COLLECTION \*/ 0xA1u, 0x00u,

/\* USAGE\_PAGE \*/ 0x05u, 0x09u,

/\* USAGE\_MINIMUM \*/ 0x19u, 0x01u,

/\* USAGE\_MAXIMUM \*/ 0x29u, 0x03u,

/\* LOGICAL\_MINIMUM \*/ 0x15u, 0x00u,

/\* LOGICAL\_MAXIMUM \*/ 0x25u, 0x01u,

/\* REPORT\_COUNT \*/ 0x95u, 0x03u,

/\* REPORT\_SIZE \*/ 0x75u, 0x01u,

/\* INPUT \*/ 0x81u, 0x02u,

/\* REPORT\_COUNT \*/ 0x95u, 0x01u,

/\* REPORT\_SIZE \*/ 0x75u, 0x05u,

/\* INPUT \*/ 0x81u, 0x03u,

/\* USAGE\_PAGE \*/ 0x05u, 0x01u,

/\* USAGE \*/ 0x09u, 0x30u,

/\* USAGE \*/ 0x09u, 0x31u,

/\* LOGICAL\_MINIMUM \*/ 0x15u, 0x80u,

/\* LOGICAL\_MAXIMUM \*/ 0x25u, 0x7Fu,

/\* REPORT\_SIZE \*/ 0x75u, 0x08u,

/\* REPORT\_COUNT \*/ 0x95u, 0x02u,

/\* INPUT \*/ 0x81u, 0x06u,

/\* END\_COLLECTION \*/ 0xC0u,

/\* END\_COLLECTION \*/ 0xC0u,

# B项目2的描述符表

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Device Descriptors \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* Descriptor Length \*/ 0x12u,

/\* DescriptorType: DEVICE \*/ 0x01u,

/\* bcdUSB (ver 2.0) \*/ 0x00u, 0x02u,

/\* bDeviceClass \*/ 0x00u,

/\* bDeviceSubClass \*/ 0x00u,

/\* bDeviceProtocol \*/ 0x00u,

/\* bMaxPacketSize0 \*/ 0x08u,

/\* idVendor \*/ 0xB4u, 0x04u,

/\* idProduct \*/ 0x0Fu, 0x21u,

/\* bcdDevice \*/ 0x00u, 0x00u,

/\* iManufacturer \*/ 0x02u,

/\* iProduct \*/ 0x03u,

/\* iSerialNumber \*/ 0x00u, /\* bNumConfigurations \*/ 0x01u

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Config Descriptor \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* Config Descriptor Length \*/ 0x09u,

/\* DescriptorType: CONFIG \*/ 0x02u,

/\* wTotalLength \*/ 0x22u, 0x00u,

/\* bNumInterfaces \*/ 0x01u,

/\* bConfigurationValue \*/ 0x01u,

/\* iConfiguration \*/ 0x00u,

/\* bmAttributes \*/ 0x80u,

/\* bMaxPower \*/ 0x00u,

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Interface Descriptor

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* Interface Descriptor Length \*/ 0x09u,

/\* DescriptorType: INTERFACE \*/ 0x04u,

/\* bInterfaceNumber \*/ 0x00u,

/\* bAlternateSetting \*/ 0x00u,

/\* bNumEndpoints \*/ 0x01u,

/\* bInterfaceClass \*/ 0x03u,

/\* bInterfaceSubClass \*/ 0x00u,

/\* bInterfaceProtocol \*/ 0x00u,

/\* iInterface \*/ 0x00u,

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

HID Class Descriptor

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* HID Class Descriptor Length \*/ 0x09u,

/\* DescriptorType: HID\_CLASS \*/ 0x21u,

/\* bcdHID \*/ 0x11u, 0x01u,

/\* bCountryCode \*/ 0x00u,

/\* bNumDescriptors \*/ 0x01u,

/\* bDescriptorType \*/ 0x22u,

/\* wDescriptorLength (LSB) \*/ USBFS\_1\_HID\_RPT\_1\_SIZE\_LSB, /\* wDescriptorLength (MSB) \*/ USBFS\_1\_HID\_RPT\_1\_SIZE\_MSB,

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Endpoint Descriptor \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* Endpoint Descriptor Length \*/ 0x07u,

/\* DescriptorType: ENDPOINT \*/ 0x05u,

/\* bEndpointAddress \*/ 0x81u,

/\* bmAttributes \*/ 0x03u,

/\* wMaxPacketSize \*/ 0x08u, 0x00u,

/\* bInterval \*/ 0x0Au

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

String Descriptor Table

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/ /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Language ID Descriptor

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* Descriptor Length \*/ 0x04u,

/\* DescriptorType: STRING \*/ 0x03u,

/\* Language Id \*/ 0x09u, 0x04u,

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

String Descriptor: "Cypress Semiconductor"

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* Descriptor Length \*/ 0x2Cu,

/\* DescriptorType: STRING \*/ 0x03u,

'C', 0,'y', 0,'p', 0,'r', 0,'e', 0,'s', 0,'s', 0,' ', 0,'S', 0,'e', 0

,'m', 0,'i', 0,'c', 0,'o', 0,'n', 0,'d', 0,'u', 0,'c', 0,'t', 0,'o', 0

,'r', 0,

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

String Descriptor: "Joystick Demo"

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* Descriptor Length \*/ 0x1Cu,

/\* DescriptorType: STRING \*/ 0x03u,

'J', 0,'o', 0,'y', 0,'s', 0,'t', 0,'i', 0,'c', 0,'k', 0,' ', 0,'D', 0

,'e', 0,'m', 0,'o', 0,

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Serial Number String Descriptor

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* Descriptor Length \*/ 0x02u, /\* DescriptorType: STRING \*/ 0x03u

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

HID Report Descriptor: HID Report Descriptor 1

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* Descriptor Size (Not part of descriptor)\*/ 0x30u, 0x00u,

/\* USAGE\_PAGE \*/ 0x05u, 0x01u,

/\* USAGE \*/ 0x09u, 0x04u,

/\* COLLECTION \*/ 0xA1u, 0x01u,

/\* USAGE \*/ 0x09u, 0x01u,

/\* COLLECTION \*/ 0xA1u, 0x00u,

/\* USAGE \*/ 0x09u, 0x30u,

/\* USAGE \*/ 0x09u, 0x31u,

/\* LOGICAL\_MINIMUM \*/ 0x15u, 0x81u,

/\* LOGICAL\_MAXIMUM \*/ 0x25u, 0x7Fu,

/\* REPORT\_SIZE \*/ 0x75u, 0x08u,

/\* REPORT\_COUNT \*/ 0x95u, 0x02u,

/\* INPUT \*/ 0x81u, 0x02u, /\* END\_COLLECTION \*/ 0xC0u,

/\* USAGE\_PAGE \*/ 0x05u, 0x09u,

/\* USAGE\_MINIMUM \*/ 0x19u, 0x01u,

/\* USAGE\_MAXIMUM \*/ 0x29u, 0x05u,

/\* LOGICAL\_MINIMUM \*/ 0x15u, 0x00u,

/\* LOGICAL\_MAXIMUM \*/ 0x25u, 0x01u,

/\* REPORT\_SIZE \*/ 0x75u, 0x01u,

/\* REPORT\_COUNT \*/ 0x95u, 0x05u,

/\* INPUT \*/ 0x81u, 0x02u,

/\* REPORT\_COUNT \*/ 0x95u, 0x01u,

/\* REPORT\_SIZE \*/ 0x75u, 0x03u,

/\* INPUT \*/ 0x81u, 0x03u,

/\* END\_COLLECTION \*/ 0xC0u,