# HID报告描述符详细解释

## ——以鼠标和键盘为例

[备注]抱歉，我忘记当时是参考谁的，若原作者看到，可以联系我，并提供您最初的发表链接，我将在该文档中进行注明。

QQ: 714912840

code char KeyBoardReportDescriptor[63] = {

//表示用途页为通用桌面设备

**0x05,** **0x01,**                    // USAGE\_PAGE (Generic Desktop)

//表示用途为键盘

**0x09,** **0x06,**                    // USAGE (Keyboard)

//表示应用集合，必须要以END\_COLLECTION来结束它，见最后的END\_COLLECTION

**0xa1,** **0x01,**                    // COLLECTION (Application)

//表示用途页为按键

**0x05,** **0x07,**                    // USAGE\_PAGE (Keyboard)

//用途最小值，这里为左ctrl键

**0x19,** **0xe0,**                    //USAGE\_MINIMUM (Keyboard LeftControl)

//用途最大值，这里为右GUI键，即window键

**0x29,** **0xe7,**                    //  USAGE\_MAXIMUM (Keyboard Right GUI)

//逻辑最小值为0

**0x15,** **0x00,**                    //  LOGICAL\_MINIMUM (0)

//逻辑最大值为1

**0x25,** **0x01,**                    //  LOGICAL\_MAXIMUM (1)

//报告大小（即这个字段的宽度）为1bit，所以前面的逻辑最小值为0，逻辑最大值为1

**0x75,** **0x01,**                    //   REPORT\_SIZE (1)

//报告的个数为8，即总共有8个bits

**0x95,** **0x08,**                    //   REPORT\_COUNT (8)

//输入用，变量，值，绝对值。像键盘这类一般报告绝对值，

//而鼠标移动这样的则报告相对值，表示鼠标移动多少

**0x81,** **0x02,**                  //  INPUT (Data,Var,Abs)

//上面这这几项描述了一个输入用的字段，总共为8个bits，每个bit表示一个按键

 //分别从左ctrl键到右GUI键。这8个bits刚好构成一个字节，它位于报告的第一个字节。

//它的最低位，即bit-0对应着左ctrl键，如果返回的数据该位为1，则表示左ctrl

键被按下，

 //否则，左ctrl键没有按下。最高位，即bit-7表示右GUI键的按下情况。中间的几个位，

//需要根据HID协议中规定的用途页表（HID Usage Tables）来确定。这里通常用来表示

//特殊键，例如ctrl，shift，del键等

 //这样的数据段个数为1

**0x95,** **0x01,**                    //   REPORT\_COUNT (1)

//每个段长度为8bits

**0x75,** **0x08,**                    //   REPORT\_SIZE (8)

//输入用，常量，值，绝对值

**0x81,** **0x03,**                  //  INPUT (Cnst,Var,Abs)

//上面这8个bit是常量，设备必须返回0

//这样的数据段个数为5

**0x95,** **0x05,**                    //   REPORT\_COUNT (5)

//每个段大小为1bit

**0x75,** **0x01,**                    //   REPORT\_SIZE (1)

//用途是LED，即用来控制键盘上的LED用的，因此下面会说明它是输出用

**0x05,** **0x08,**                    //   USAGE\_PAGE (LEDs)

//用途最小值是Num Lock，即数字键锁定灯

**0x19,** **0x01,**                    //   USAGE\_MINIMUM (Num Lock)

//用途最大值是Kana，这个是什么灯我也不清楚^\_^

**0x29,** **0x05,**                    //   USAGE\_MAXIMUM (Kana)

//如前面所说，这个字段是输出用的，用来控制LED。变量，值，绝对值。

//1表示灯亮，0表示灯灭

**0x91,** **0x02,**                   //   OUTPUT (Data,Var,Abs)

//这样的数据段个数为1

**0x95,** **0x01,**                    //   REPORT\_COUNT (1)

//每个段大小为3bits

**0x75,** **0x03,**                    //   REPORT\_SIZE (3)

//输出用，常量，值，绝对

**0x91,** **0x03,**                    //   OUTPUT (Cnst,Var,Abs)

//由于要按字节对齐，而前面控制LED的只用了5个bit，

//所以后面需要附加3个不用bit，设置为常量。

//报告个数为6

**0x95,** **0x06,**                    //   REPORT\_COUNT (6)

//每个段大小为8bits

**0x75,** **0x08,**                    //   REPORT\_SIZE (8)

//逻辑最小值0

**0x15,** **0x00,**                    //   LOGICAL\_MINIMUM (0)

//逻辑最大值255

**0x25,** **0xFF,**                    //   LOGICAL\_MAXIMUM (255)

//用途页为按键

**0x05,** **0x07,**                    //   USAGE\_PAGE (Keyboard)

//使用最小值为0

**0x19,** **0x00,**                    //   USAGE\_MINIMUM (Reserved (no event indicated))

//使用最大值为0x65

**0x29,** **0x65,**                    //   USAGE\_MAXIMUM (Keyboard Application)

//输入用，变量，数组，绝对值

**0x81,** **0x00,**                    //   INPUT (Data,Ary,Abs)

//以上定义了6个8bit宽的数组，每个8bit（即一个字节）用来表示一个按键，所以可以同时

//有6个按键按下。没有按键按下时，全部返回0。如果按下的键太多，导致键盘扫描系统

//无法区分按键时，则全部返回0x01，即6个0x01。如果有一个键按下，则这6个字节中的第一

//个字节为相应的键值（具体的值参看HID Usage Tables），如果两个键按下，则第1、2两个

//字节分别为相应的键值，以次类推。

//关集合，跟上面的对应

**0xc0**

// END\_COLLECTION

};

 通过上面的分析，我们知道这个报告中只有一个报告，所以没有报告ID，因此返回的都是实际使用的数据。总共有8字节输入，1字节输出。其中输入的第一字节用来表示特殊按键，第二字节保留，后面的六字节为普通按键。

下面这个报告描述符是USB鼠标报告描述符，比起键盘的来说要简单些。它描述了4个字节，第一个字节表示按键，第二个字节表示x轴（即鼠标左右移动，0表示不动，正值表示往右移，负值表示往左移），第三个字节表示y轴（即鼠标上下移动，0表示不动，正值表示往下移动，负值表示往上移动），第四个字节表示鼠标滚轮（正值为往上滚动，负值为往下滚动）。

code char MouseReportDescriptor[52] = {

//通用桌面设备

**0x05,** **0x01,**                    // USAGE\_PAGE (Generic Desktop)

//鼠标

**0x09,** **0x02,**                    // USAGE (Mouse)

//集合

**0xa1,** **0x01,**                    // COLLECTION (Application)

//指针设备

**0x09,** **0x01,**                    // USAGE (Pointer)

//集合

**0xa1,** **0x00,**                    // COLLECTION (Physical)

//按键

**0x05,** **0x09,**                    // USAGE\_PAGE (Button)

//使用最小值1

**0x19,** **0x01,**                    // USAGE\_MINIMUM (Button 1)

//使用最大值3。1表示左键，2表示右键，3表示中键

**0x29,** **0x03,**                    // USAGE\_MAXIMUM (Button 3)

//逻辑最小值0

**0x15,** **0x00,**                    // LOGICAL\_MINIMUM (0)

//逻辑最大值1

**0x25,** **0x01,**                    // LOGICAL\_MAXIMUM (1)

//数量为3

**0x95,** **0x03,**                    //     REPORT\_COUNT (3)

//大小为1bit

**0x75,** **0x01,**                    //     REPORT\_SIZE (1)

//输入，变量，数值，绝对值

//以上3个bit分别表示鼠标的三个按键情况，最低位（bit-0）为左键

//bit-1为右键，bit-2为中键，按下时对应的位值为1，释放时对应的值为0

**0x81,** **0x02,**                   //     INPUT (Data,Var,Abs)

//填充5个bit，补足一个字节

**0x95,** **0x01,**                    //     REPORT\_COUNT (1)

**0x75,** **0x05,**                    //     REPORT\_SIZE (5)

**0x81,** **0x03,**                   //     INPUT (Cnst,Var,Abs)

//用途页为通用桌面

**0x05,** **0x01,**                    //     USAGE\_PAGE (Generic Desktop)

//用途为X

**0x09,** **0x30,**                    //     USAGE (X)

//用途为Y

**0x09,** **0x31,**                    //     USAGE (Y)

//用途为滚轮

**0x09,** **0x38,**                    //     USAGE (Wheel)

//逻辑最小值为-127

**0x15,** **0x81,**                    //     LOGICAL\_MINIMUM (-127)

//逻辑最大值为+127

**0x25,** **0x7f,**                    //     LOGICAL\_MAXIMUM (127)

//大小为8个bits

**0x75,** **0x08,**                    //     REPORT\_SIZE (8)

//数量为3个，即分别代表x,y,滚轮

**0x95,** **0x03,**                    //     REPORT\_COUNT (3)

//输入，变量，值，相对值

**0x81,** **0x06,**                    //     INPUT (Data,Var,Rel)

//关集合

**0xc0,**                          //   END\_COLLECTION

**0xc0**                           // END\_COLLECTION

};

  这里只对报告描述符做一个简单的介绍，更详细的资料请参看USB HID协议以及HID Usage Tables