专题 4：HID-USB 设备开发

专题	PIC18F13K50/14K50 HID USB 设备开发技术		
创建日期	2009-12-14	最新版本号	0.1
修改日期	2010-01-22	修改人	
备注	USB 基础知识及 PIC18F13K50/14K50 HID USB 设备开发技术		

基础知识:

Q1:高速 USB 和 USB 2.0 有区别吗？哪一种说法是正确的？

高速 USB 和 USB 2.0 是有区别的，区别在于 USB 2.0 是一种规范，而“高速 USB”仅指在 USB 2.0 规范中数据传输率为 480Mbps 的那部分，某个设备可以是符合 USB 2.0 的设备，同时还可以是全速设备或低速设备。

Q2:USB 设备与计算机的通信速度有多快？

USB 2.0 支持低速、全速和高速三种速度，分别为 1.5Mbps、12Mbps 和 480Mbps。

Q3: USB 规范定义了哪些数据传输机制(Endpoint Transfer Type)？

四种不同的数据传输机制分别为：

控制传输(Control)

USB 主机使用控制传输向 USB 设备发送命令和询问。当枚举 USB 设备时，控制传输使用端点 0（EP0），这样可强制所有 USB 设备支持 EP0 而无需考虑支持速度。控制包的最大容量为 8、16、32 或 64 字节。低速 USB 设备中控制传输的包长度必须为 8 字节，高速 USB 设备必须为 64 字节，而全速 USB 设备允许 8、16、32 或 64 字节(MicroChip 官网对此有错误的说法)，也就是说控制传输对于最大包长度有固定的要求。最大包长度信息在“设备描述符”中体现，即 wMaxPacketSize 为“设备描述符”包的 Byte7，它反应了该端点对应的 Buffer 的大小，当通过一个端点进行数据传输而数据的大小超过该端点的最大包长度时，需要将数据分成若干个数据包传输，并且要求除最后一个包外，所有的包长度均等于该最大包长度，也就是说，如果一个端点收到/发送了一个长度小于最大包长度的包，即意味着数据传输结束。但需要注意:Setup 包总是 8 Bytes 的。

中断传输(Interrupt)

中断传输是 USB 设备向 USB 主机请求一定轮询速率的方法。在枚举过程中，USB 设备向 USB 主机请求轮询时间。全速设备的最大轮询速率为每毫秒一次，低速设备为每 10 毫秒一次。低速 USB 设备的最大数据负载为 8 字节，全速 USB 设备为 64 字节，而高速设备最大数据包为 1024 字节。这使得全速 USB 设备的最大吞吐量为 64KB/s，低速 USB 设备为 800B/s **确认中断传输可保证数据的传送**。如果传输包接收失败，则会重新发送。中断传输是单向传输，但这里的单向传输并不是说只支持一个方向的传输，而是指在某个端点上该传输仅支持一个方向，或输入，或输出，如果需要在两个方向上进行某种单向传输，需要占用两个端点，可分别配置成不同的方向。

批量传输(Bulk)

批量传输是设备传输大量数据的方法，但是不能保证会及时发送。在总线安排传输时，批量传输的优先级最低。当其他传输完成后，余下的带宽复位后将分配给批量传输。与中断传输一样，确认批量传输可保证数据的传送。只有全速和高速设备支持批量传输。对于全速 USB 设备的端点而言，最大的包容量可为 8、16、32 或 64 字节长度。对于高速 USB 设备的端点而言，最大的包容量可高达 512 字节长度。

同步传输(Isochronous)

同步传输保证了传输速率。全速同步传输每帧可发送 1023 字节的数据。同步传输无需确

认。因此，同步包有可能并未送达。同步传输的典型应用为音频/视频流，其中最重要的是以丢失包为代价保持视频和音频的进行。全速 USB 设备的最大传输率为 1023Kbps。

Note：HID 设备只支持控制传输和中断传输方式。

Q4: 什么是令牌包？

USB 协议定义了四种类型的包：

帧起始、令牌、数据 与 握手，而其中的令牌有三种不同的形式包。

IN —— 通知 USB 设备，主机欲读取信息

OUT —— 通知 USB 设备，主机欲发送信息

Setup —— 通知设备，主机要进行控制传输

Q5: 什么是枚举过程？

USB 主机通过枚举过程可了解有新的 USB 设备与总线相连。在应用开始前，主机向设备询问各种信息，以确定设备的类型、载入设备所需的设备驱动程序以及设备的功率需求等。在枚举过程中，USB 主机还为连接的设备分配地址。在地址设定后，USB 主机将与位于该地址的设备进行通信。枚举过程的最终任务之一是把设备设定为某一具体运行配置。枚举过程的详细内容在 USB 规范的 9.1.2 节中给出。

Q6: USB 主机如何识别 USB 设备的速度？

USB 主机在两根通信线（D+和 D-）上均有弱下拉电阻。如果设备要以全速模式运行，将会用较大的上拉电阻上拉 D+。如果设备要运行在低速模式下，则会上拉 D-。两种情况中的上拉电阻的标称值为 1 kΩ(PIC18F1xK50 Datasheet 规定该电阻为 1.5Kohm)。

Q7: USB 主机如何对 USB 设备进行复位？

将 D+和 D-拉低至少 10 毫秒，USB 主机就会对设备进行复位。当 D+和 D-拉低时间超过2.5 微秒，USB 设备就可认为已发生复位。一旦 USB 设备检测到了复位，在 USB 主机移除复位后将马上进入默认状态。该复位仅用于 USB 复位，不可复位控制器。

Q8: 什么是 VID 和 PID

VID 指的是厂商 ID，PID 指的是产品 ID。通过支付费用，USB-IF 会发出 VID。USB-IF 要求每个厂商拥有自己的 VID，以便销售其产品。当所使用的 VID/PID 不是惟一时，可能会发生法律和技术上的纠纷。

Q9: USB2.0 支持三种传输速率：

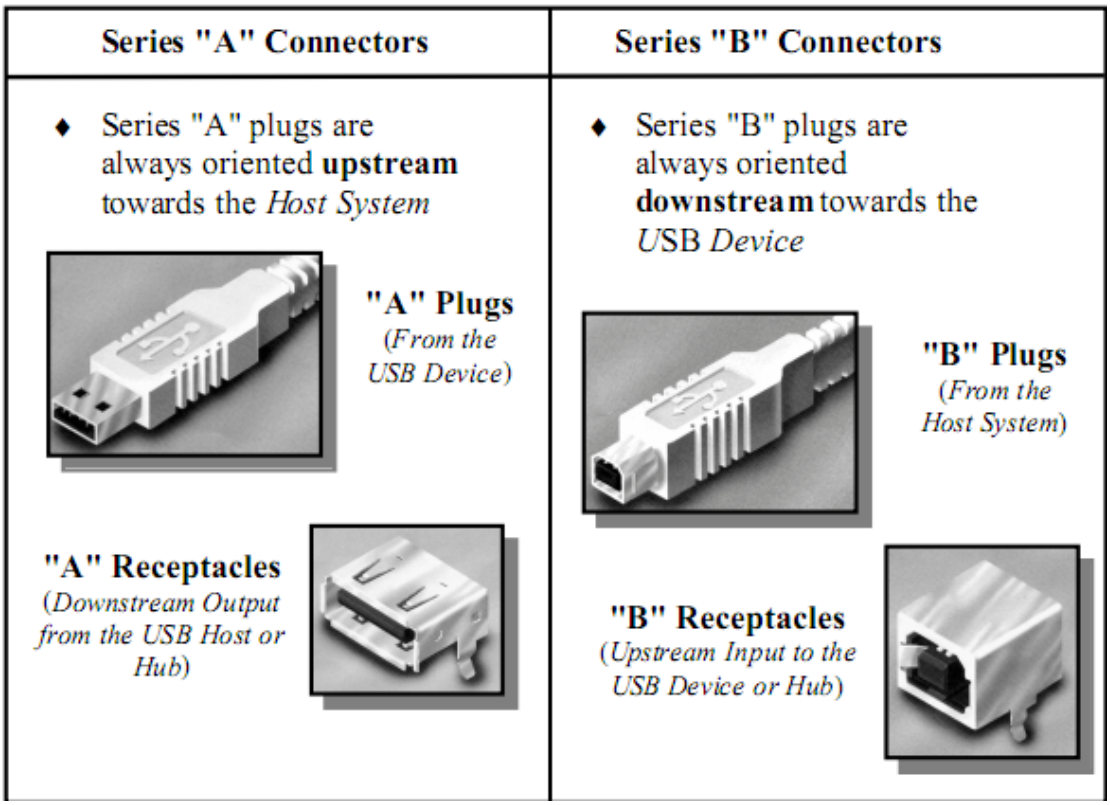
低速（Low Speed），习惯称为 USB1.0，传输速率为 1.5Mbps；

全速（Full Speed），习惯称为 USB1.1，传输速率为 12Mbps；

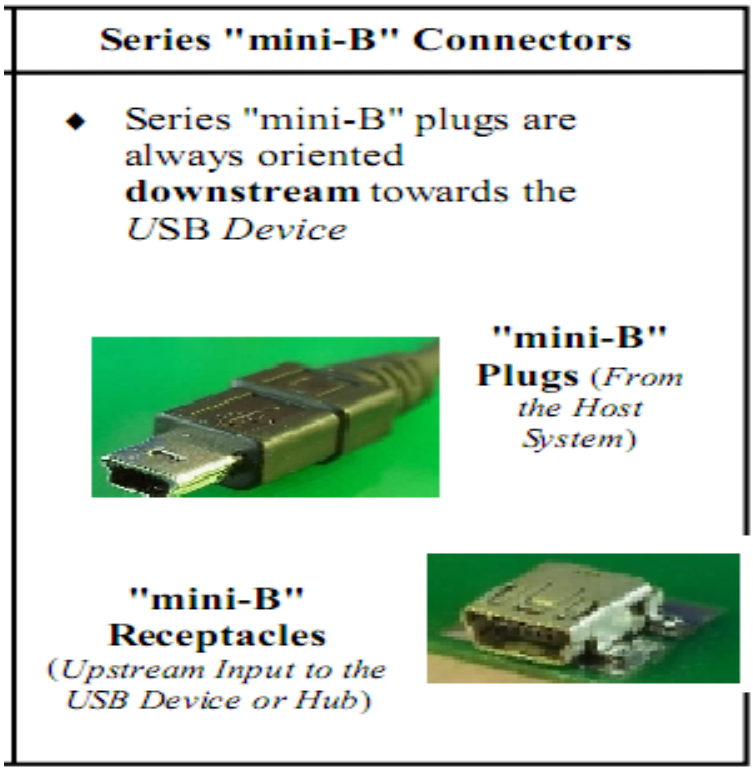
高速（High Speed），习惯称为 USB2.0，传输速率为 480Mbps；

Q10. USB 接口有三种形式，分别为 USB Type A、Type B 和 Mini-USB，对于 AB 两种类型来说，都只有四条接线，而 Mini-USB 却有五条连线，其外观图如下图所示：

Type-A: Type-B:



Mini-USB:



◆USB 底层程序设计中的几个概念：

端点(Endpoint)：是 USB 设备中的一个独特的概念，它是 USB 设备与 USB Host 交换

数据的硬件单元，不同的端点其传输数据的能力不同，适用于不同的应用场合；a USB Device Endpoint uses only one data transfer method；每个设备描述符下应当有多个端点描述符。一般的 USB 芯片都会提供几个标准的端点，每个端点都支持单一的总线传输方式，其中端点 0 必须支持控制传输。

配置(Configuration)：是用于定义设备的功能，如果一个设备有几种不同的功能，则每个功能都需要一个配置，配置是接口的集合；

接口(Interface)：是指定设备中哪些硬件与 USB 交换数据，每一个与 USB 交换数据的硬件就叫一个端点，因此，接口是端点的集合。

◆HID 设备的特点:

交换的数据存储在报告结构内，设备必须支持 HID 报告格式;

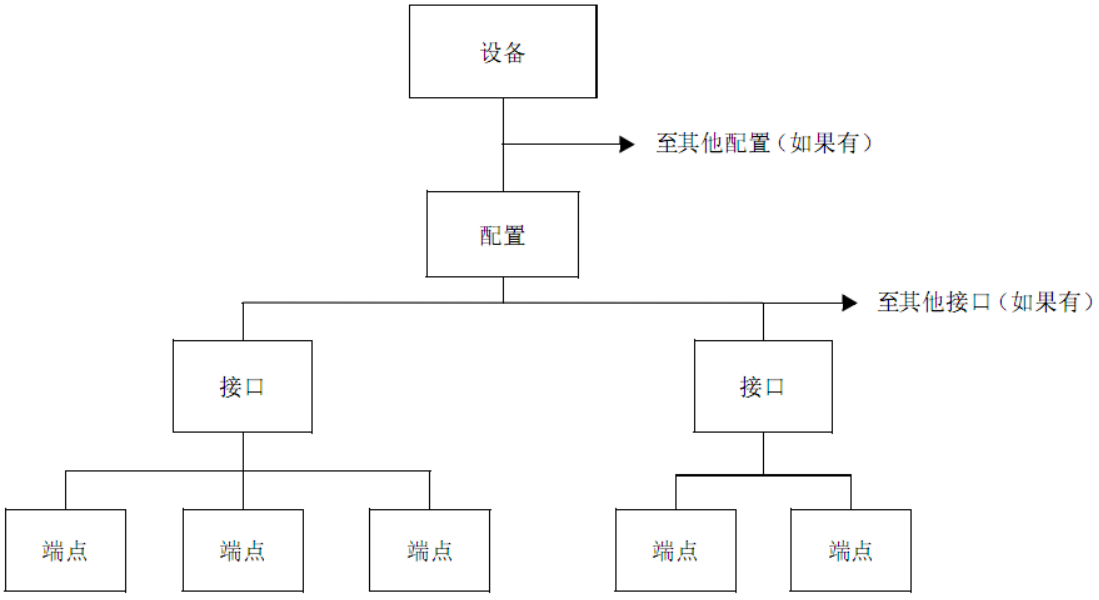
每笔事务可以携带小量或中量的数据，低速设备每笔事务最大为 8Bytes，全速设备每笔最大为 64Bytes(??)，高速设备最大为 1024Bytes；

有最大传输速度的限制：低速设备最快 10ms 一笔事务，最高速度为 800Bytes/s，全速设备最快 1ms 一笔事务，最高速度为 64KBytes/s；高速设备最快 125us 一笔事务，最高速度为 24.576MByts/s。

没有传输速度的保证。

◆ 为了把一个设备识别为 HID 类别，设备在定义描述符的"类别"字段必须设备为 0x03 以表示是 HID 类别，这样主机就会继续请求获得设备的 HID 描述符和报告描述符。

◆ USB 分层结构如下图所示，对应于“设备”、“配置”、“接口”与“端点”都有相应的“描述符”。



应用篇：

HID 分为 Standard HID 与 Custom-Build HID，对于前者，比如说 mouse/Key board 都属于该类型，而 Custom-Build HID 则为用户自定义所用的 HID，以下是对 Standard HID 实例 Mouse 类采用 PIC18F14K50 MCU 程序设计示例说明。

该项目需要借助 MicroChip 提供的 USB 架构（包含了 USB 的部分公用驱动底层程序），建立工程项目如下步骤进行：

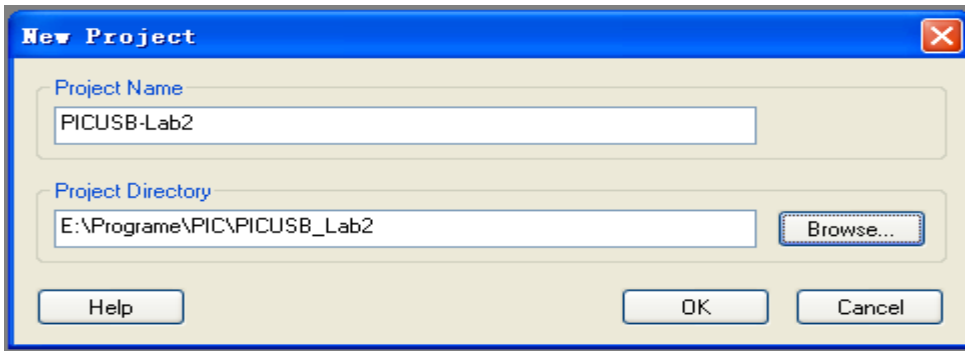
前提：搭建 IDE 及 MPLAB C18 开发环境，笔者所采用版本为：MPLAB IDE v8.40、MPLINK v4.34、MPLABC C18 v3.34；另外还需要下载《Microchip USB Application Libraries v2009-08-31.EXE》并安装到 C:\目录下。

Step1：建立项目文件夹，如在 E:\Programe\PIC 目录上建立以 PICUSB_Lab2 为名的文件夹；

Step2：将以下 15 个文件拷贝到该文件夹下：

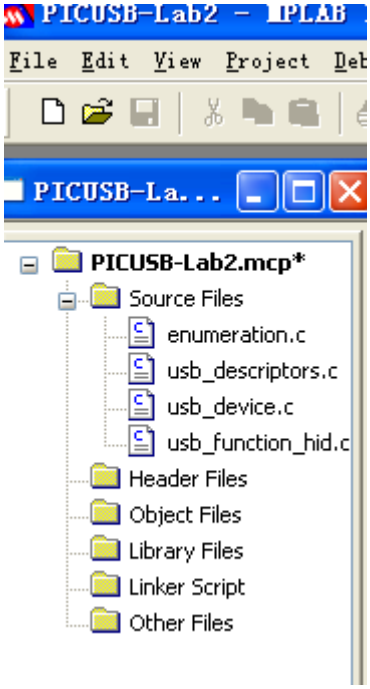
enumeration.c 、 usb_descriptors.c 、 usb_device.c 、 usb_function_hid.c 、 Compiler.h 、 GenericTypeDefs.h、HardwareProfile.h、usb.h、usb_ch9.h、usb_common.h、usb_config.h、usb_device.h、usb_function_hid.h、usb_hal.h、usb_hal_pic18.h。

Step3：打开 MPLAB IDE 并建立新项目(有两种方式，在这里暂用 Project -> New)，在 Project Name 一栏写入项目名(如：PICUSB-Lab2)，在 Project Directory 采用"Browse"引导的方式导引到刚才所建立的项目文件夹处，如下图所示：



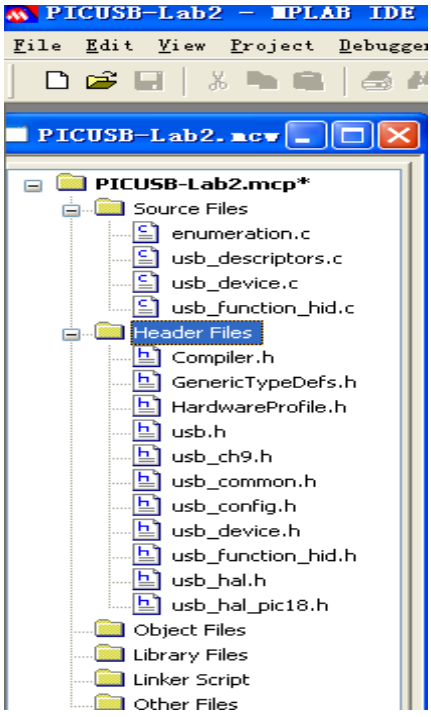
然后“OK”；

Step4：在 MPLAB IDE 中选中"View->Project"就会出现“项目文件管理”工具，右击"Source Files- >Add Files..."，在向导框中选中刚才拷贝的那 4 个 C 文件，此时“项目文件管理”



工具如下所示：

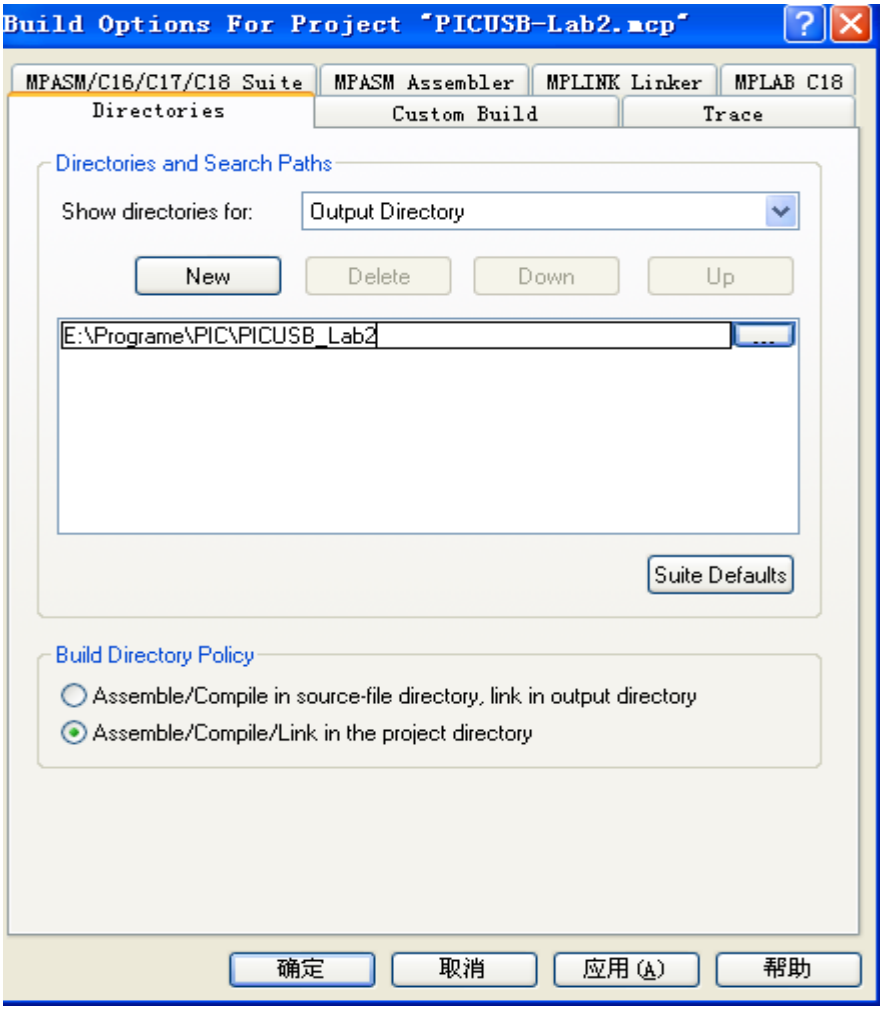
Step5：采用 Step4 的方式，将刚才拷贝的 11 个.H 文件加载到 Header Files 文件夹下，如下图所示：



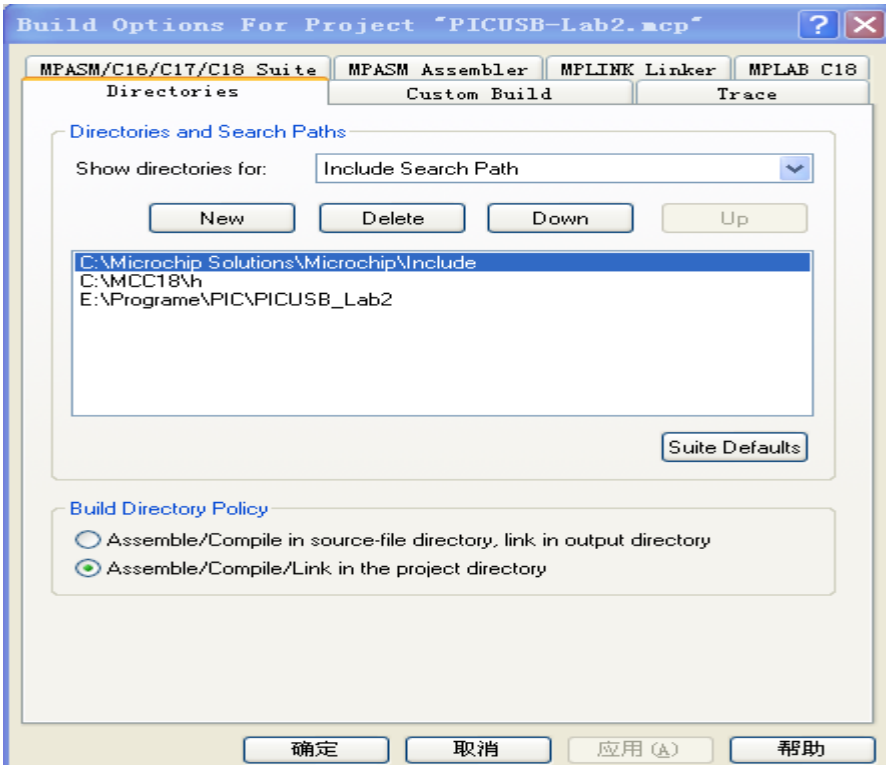
Step6：在 MPLAB IDE 中"Configure->Select Device..."之后的"Select Device"对话框中选中 Device 为 PIC18F14K50；

Step7：编译选项(Build Options)的配置：在 MPLAB IDE 中"Project -> Build Options...->Project"就会出现 Build Options 对话框，在 Directories 选项卡中选择 Show directories for:的"Output Directory"，然后点击"New"以添加新的路径，选中当前工程目录，并建立以_output 为名的文件夹，如下图所示(注：该步骤并非必须，这样做的目的在于将编译生成的*.hex 文件放到_output 文件夹下，以便管理，如果忽略该步，则编译生成的*.hex 文件自动

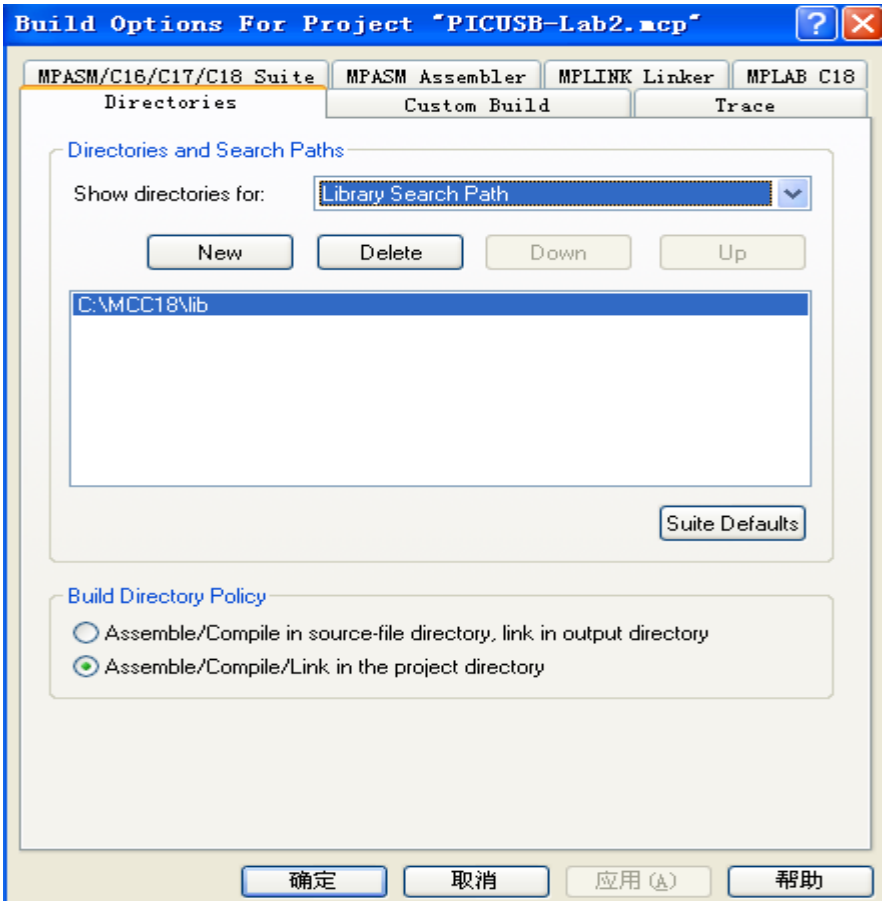
放在该工程目录下)；



Step8：在上一步的基础上选中 Show directories for 的下拉菜单内的"Include Search Path"，按 Step7 的方法，将 C:\MCC18\h 目录、C:\Microchip Solutions\Mircochip\Include 目录及该工程目录(E:\Programe\PIC\PICUSB_Lab2)都加载进来，如下图所示；

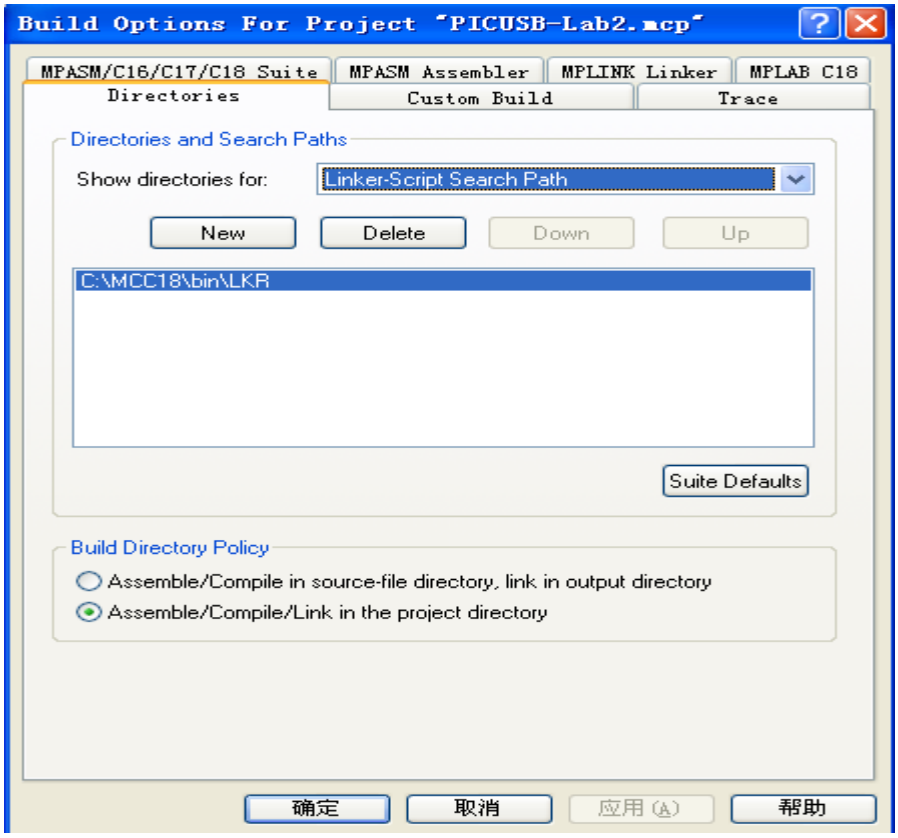


Step9：在 Step7 的基础上选中 Show directories for 的下拉菜单内的"Library Search Path"，按 Step7 的方法，将 C:\MCC18\lib 目录加载进来，如下图所示；



Step10：在 Step7 的基础上选中 Show directories for 的下拉菜单内的"Linker Search Path"，按

Step7 的方法，将 C:\MCC18\bin\LKR 目录加载进来，如下图所示；



点击"应用"-> "确定"然后就可以编译了。

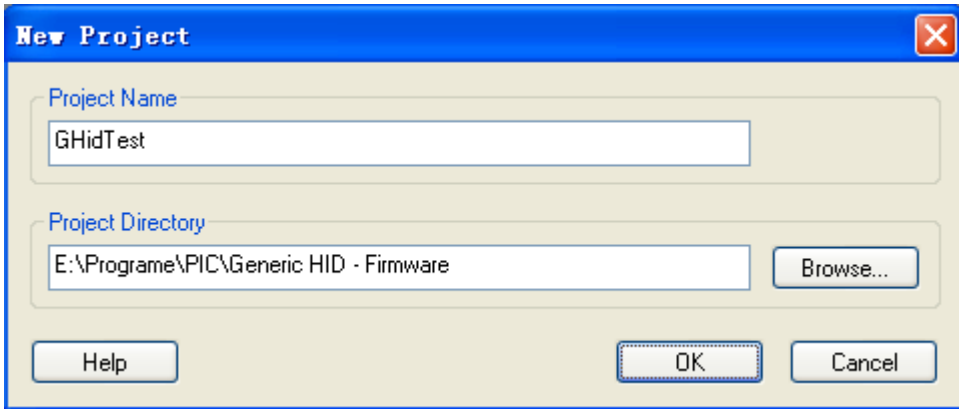
注意：上述过程中所用到的 enumeration.c 是针对"枚举"过程而定义的 C 文件，也可以认为是"应用程序文件"，如果欲实现 Mouse/KeyBoard 等功能时，需要修改该文件，主要的应用程序开发也是在该文件上进行的。

****

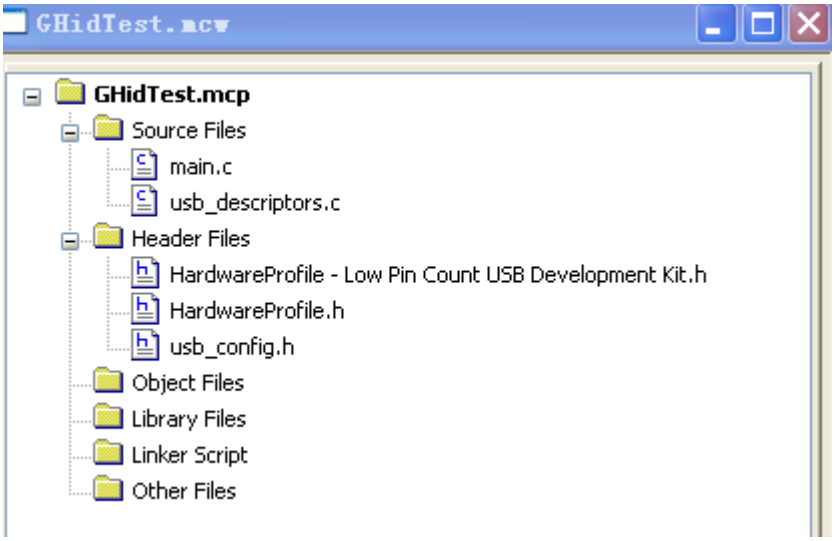
采用 PIC18F14K50 对 Custom-Build HID USB 设备的开发实例

前提：因为该类项目可以以“USB Device - HID - Custom Demos”下的“Generic HID - Firmware”为基础进行修改，因此所需要的文件有：main.c usb_descriptors.c 及 HardwareProfile - Low Pin Count USB Development Kit.h HardwareProfile.h usb_config.h，以及 lkr 文件：rm18f14k50.lkr(注意：以下有些步骤相似于上述 Standard HID 示例)。

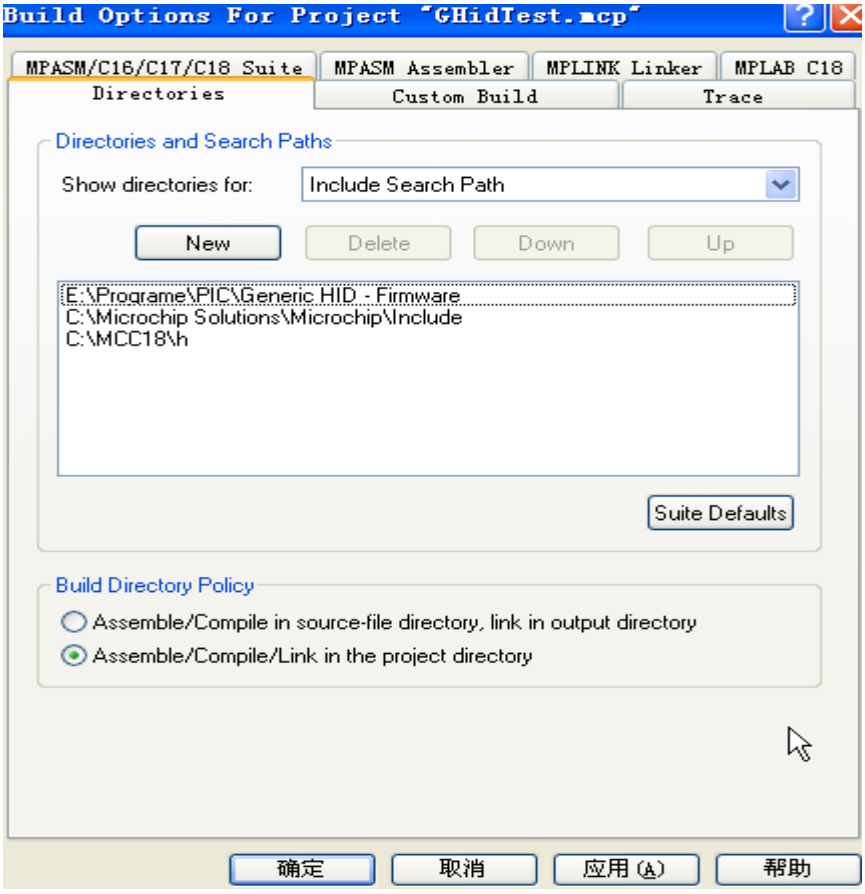
第一步：新建项目：



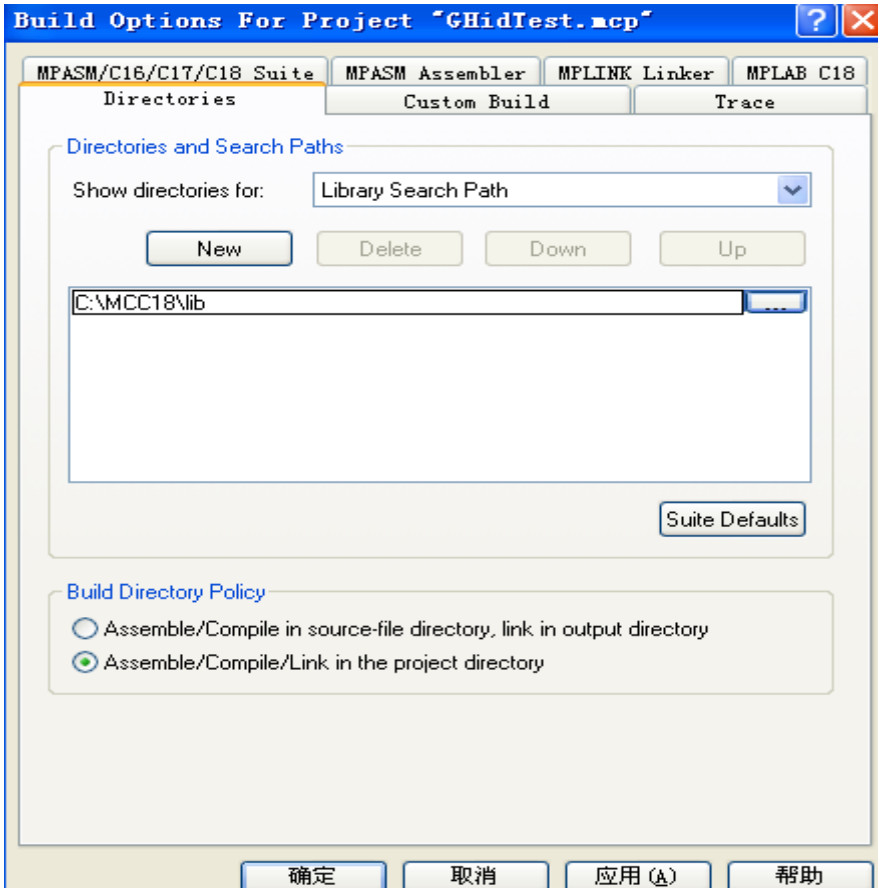
第二步：加载上述两个 c 文件与三个 h 文件：



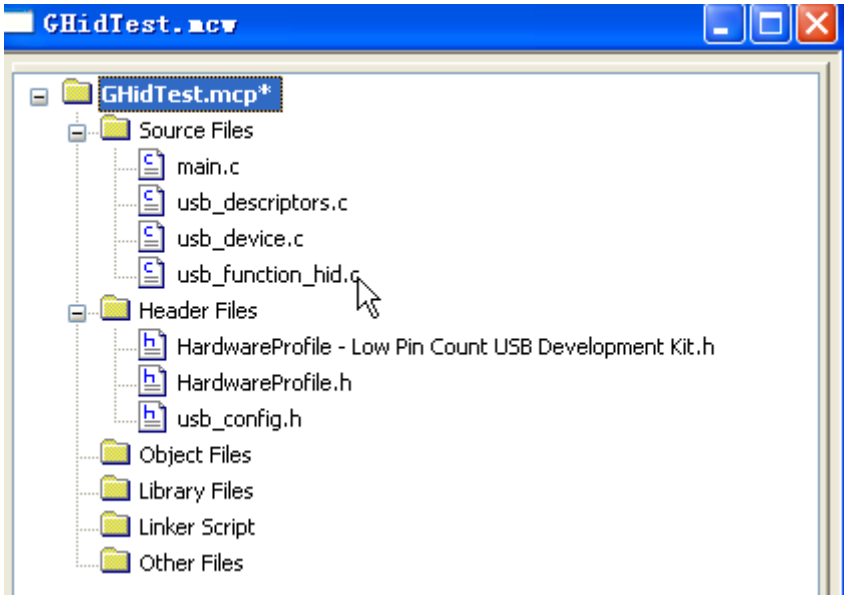
第三步：修改编译选项，使 Include Search Path 包含：C:\MCC18\h，C:\Microchip Solutions\Microchip\Include 以及该项目所在路径，如：E:\Programe\PIC\Generic HID - Firmware，如下图所示：



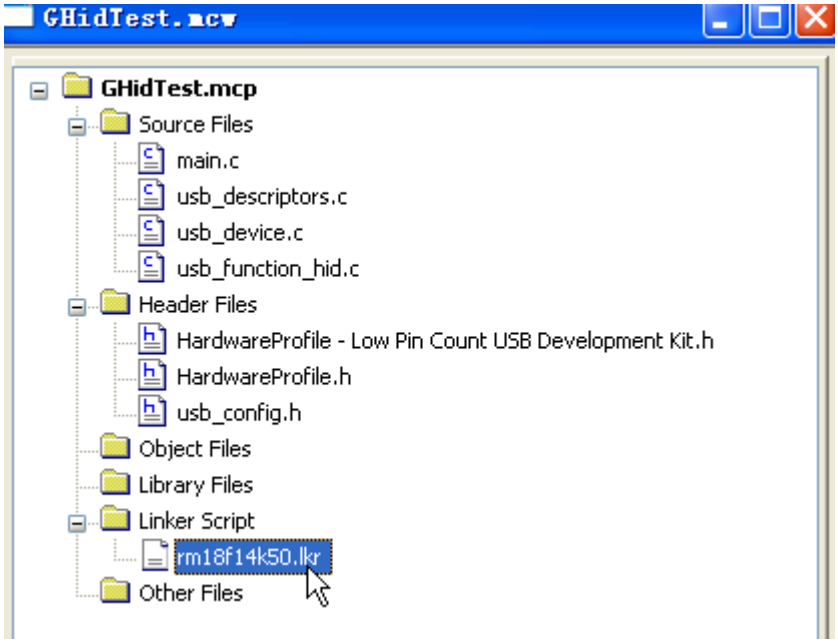
第三步：修改 Library Search Path 为 C:\MCC18\lib，如下图所示：



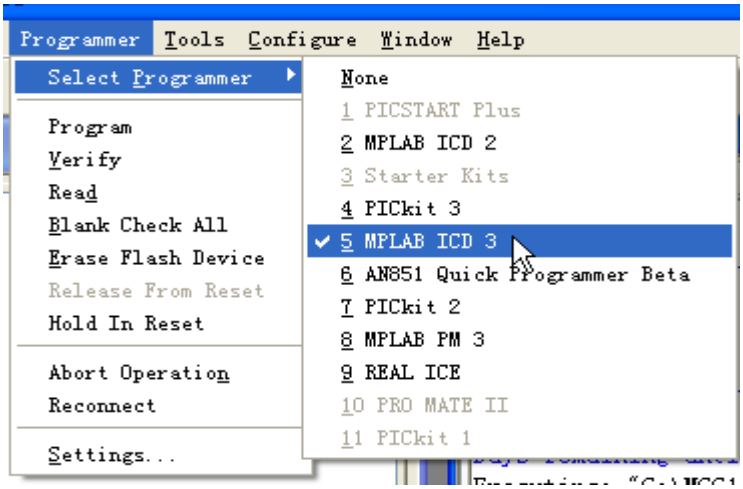
第四步：加载 C:\Microchip Solutions\Microchip\Usb 下的 usb_device.c 和 C:\Microchip Solutions\Microchip\Usb\HID Device Driver 下的 usb_function_hid.c



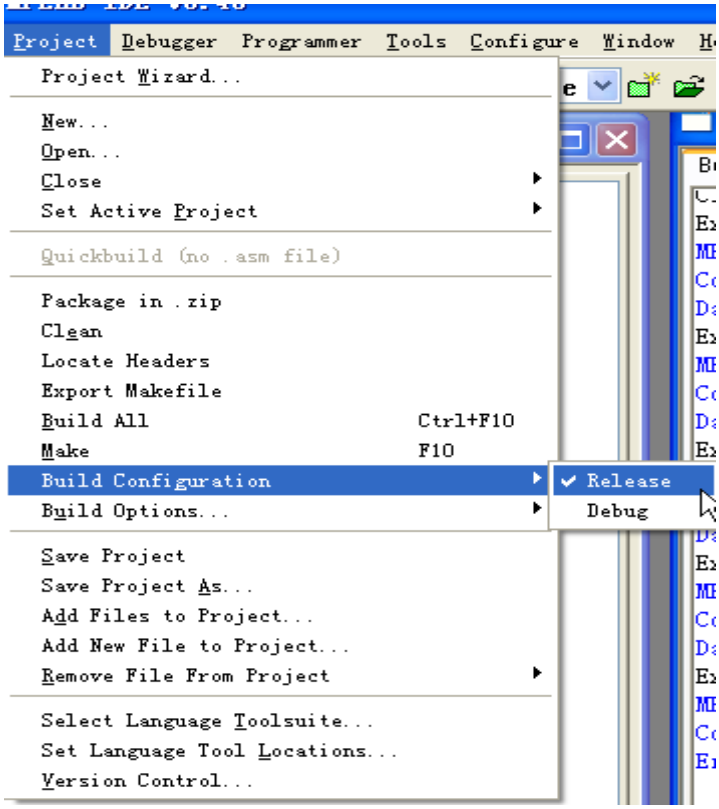
第五步：加载 rm18f14k50.lkr 文件，如下所图所示(注意:如果该文件不加载,编译可能不会出错,但功能可能不正常,比如说 UART 通信收到数据就不正确)：



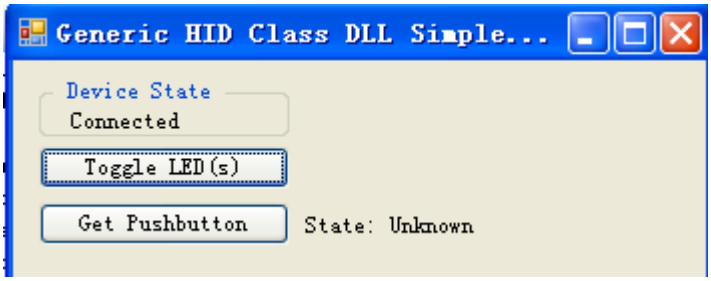
第六步：选择 programme 工具：



第七步：选择 release 版本：



第八步：编译下载即可执行，这样下载后的文件可使 PC 识别到 PIC18F14K50，此时对应的 PC 机测试软件为 C:\Microchip Solutions\USB Device - HID - Custom Demos\Generic HID - HID DLL - PC Software\Microsoft Visual C++ 2008 Express，运行程序如下图所示：



当点击 Toggle Led(s)时，PC 发送一个 0x80 给 PIC18F14K50，可以由此设计相应的应用软件。

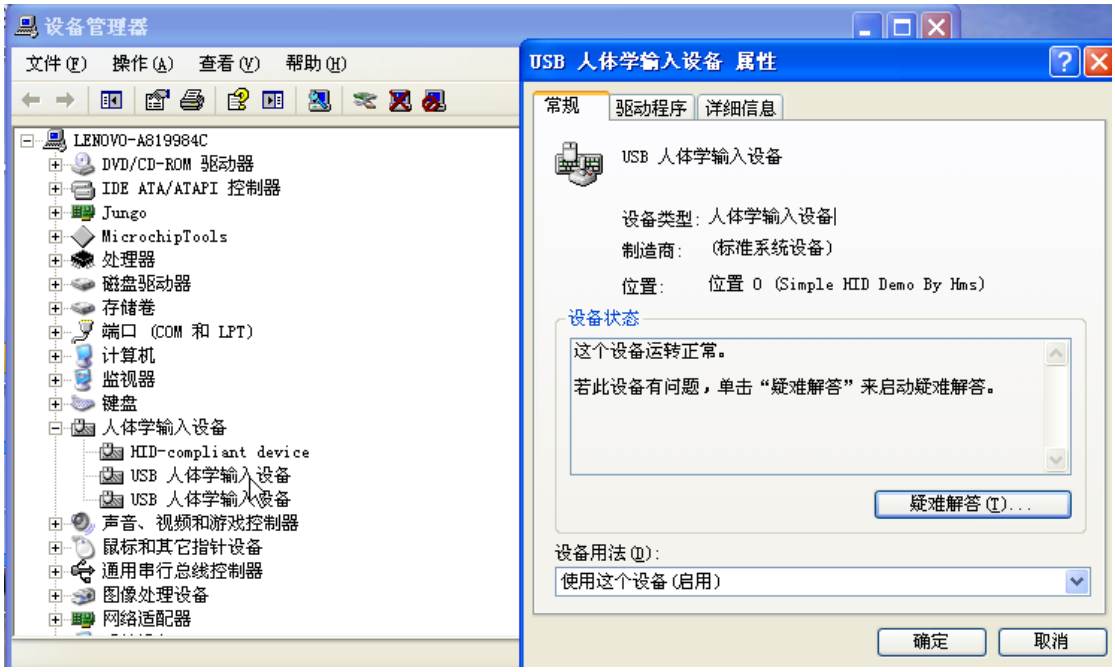
第九步：应用软件开发：

在 usb_descriptors.c 文件中找到"产品技术符"并修改如下：

```
//Product string descriptor
ROM struct{BYTE bLength;BYTE bDscType;WORD string[25];}sd002={
sizeof(sd002),USB_DESCRIPTOR_STRING,
{
//'S','i','m','p','l','e',' ','H','I','D',' ',
//'D','e','v','i','c','e',' ','D','e','m','o'

'S','i','m','p','l','e',' ','H','I','D',' ',
'D','e','m','o',' ','B','y',' ','H','m','s'
}
};
```

下载到 18F14K50 之后，断开 MCU 系统电源及下载线，插入 USB 接头，PC 机识别如下图：



在 18F14K50 的 main.c 文件中找到 void ProcessIO(void)函数，在其下的

```
if(!HIDRxHandleBusy(USBOutHandle))
{
//加载用户自己的应用程序
}
```

在这里可以加入用户的应用程序，比笔者在这里加入了以下代码，将 18F14K50 通过 HID USB 自 PC 接收到的 16 bytes 数据通过 UART 发送出去。

```
//************************************
    for(i=16;i>0;i--)
      usend[i-1]=ReceivedDataBuffer[i-1];
    USART_Send(usend, 16);
//************************************
```

也就是说，18F14K50 通过 HID USB 自 PC 接收到数据存放在 ReceivedDataBuffer[]数组中，其相应的接收函数为：

HIDRxPacket(HID_EP,(BYTE*)&ReceivedDataBuffer,64);

(接收“端点”为 HID_EP，接收数据保存在 ReceivedDataBuffer[]，而一次接收长度为 64)

而 18F14K50 通过 HID USB 向 PC 发送数据的函数为：

HIDTxPacket(HID_EP,(BYTE*)&ToSendDataBuffer[0],64);

(发送“端点”为 HID_EP，发送的数据在 ToSendDataBuffer[]，而一次发送的长度为 64)

这样，18F14K50 就可以实现 HID USB 的方式与 PC 机进行信息交换。

Note:在上述实验第九步中，可以采用“USB-HID 描述符工具”修改 usb_descriptors.c 中的

“设备描述符”-USB_DEVICE_DESCRIPTOR

“配置描述符”-configDescriptor1

“HID 类描述符”-ROM struct{BYTE report[HID_RPT01_SIZE];}hid_rpt01={};

以满足自己的要求。