Windows CE. NET 下 USB 设备驱动实现

Windwos CE. NET USB Device Driver Development

赵晶莹　郭 海　宋海玉　（大连民族学院 计算机系 116600）

摘要：基于 USBD 模块实现了 Windows CE. NET 下的 USB 设备驱动程序开发，并以 USB 鼠标驱动程序为例阐述实现过程，同时介绍了 Windows CE. NET 下 USB 驱动的基本原理，对于其它基于 Windows CE 平台下的 USB 设备驱动程序的开发有参考价值。

关键词：USB　驱动　Windows CE. NET

1　前言

Windows CE. NET 是 Microsoft 公司推出的嵌入式操作系统，越来越多的 PDA、智能手机采用这种操作系统，它提供了标准的开放式平台，极大地减少了硬件制造商、软件开发商以及最终将采纳新一代非 PC 技术解决方案的客户多方之间的矛盾。Windows CE. NET 是一款功能强大的开放的 32 位实时操作系统，提供了众多强大工具适用于快速构建新一代内存少、体积小的智能设备，例如工业控制器、手持式设备、智能电话、机顶盒和零售点设备等。随着 Windows CE 的发展，大量 USB 外接设备都支持 Windows CE. NET 系统，但是如何针对不同设备编写其 USB 驱动程序成为嵌入式研究的一个热点[1]。

2　USB 协议原理

USB（Universal Serial Bus）是一种快速、灵活的总线接口。与其它通信接口比较，USB 接口的最大特点是易于使用，这也是 USB 的主要设计目标。作为一种高速总线接口，USB 适用于多种设备，比如数码相机、MP3 播放机、高速数据采集设备等。易于使用还表现在 USB 接口支持热插拔，并且所有的配置过程都由系统自动完成，无需用户干预。USB 设备的软件设计主要包括两部分：一是 USB 设备端的单片机软件，主要完成 USB 协议处理与数据交换以及其它应用功能程序。二是 PC 及其它系统的程序，由 USB 通信程序和用户服务程序两部分组成，用户服务程序通过 USB 通信程序与系统 USBDI（USB Device Interface）通信，由系统完成 USB 协议的处理与数据传输。其工作原理参见图 1 Windows CE 下 USB 设备工作图。

USBD 将在两个主要的时间介入总线访问：在设备接上主机的设置期间、在正常传送中。当设备被接上，并被设置时，USBD 将确认设备的设置是否与总线兼容，USBD 从设置软件处获得设置请求，它描述了要进行的设置：端点、传送类型、传送周期、数据长度等。USBD 根据可用带宽的大小以及总线是否与请求的类型兼容来决定接受还是不接受这个设置。如果接收，USBD 就为请求者建立一个相应类型的通道，自然也带有这个传送类型所应有的各种限制。在设备设置期间，不一定必须为周期性的端点作带宽分配。做过的带宽分配也可以放弃，而并不影响设备的设置。

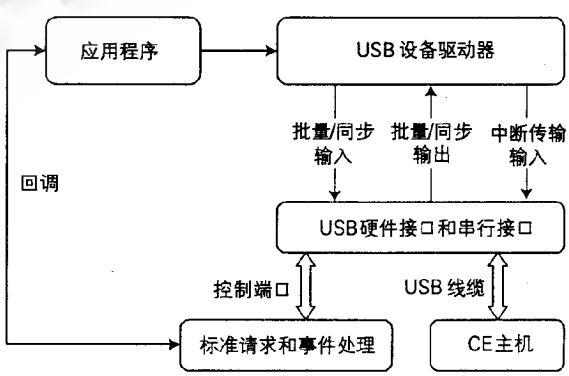

图 1　Windows CE 下 USB 设备工作图

对USBD的设置是由操作系统决定的，而且反过来影响操作系统的设置特征。这样做可免除一些多余的接口。一旦设备被设置，客户软件就可以通过IRP来请求与应用端点进行数据传送。

3 Windows CE. NET 实现 USB 驱动的方法

Windows CE. NET 的 USB 系统软件分为两层：USB Client 设备驱动程序和底层的 Windows CE 实现的函数层。USB 设备驱动程序主要负责利用系统提供的底层接口配置设备，和设备进行通讯。底层的函数体本身又由两部分组成，通用串行总线驱动程序（USBD）模块和较低的主控制器驱动程序（HCD）模块。HCD 负责最最底层的处理，USBD 模块实现较高的 USBD 函数接口。USB 设备驱动主要利用 USBD 接口函数和他们的外围设备打交道。

USB 设备驱动程序的存在使得外围设备可以为应用程序提供服务，虽然没有供 USB 设备使用的标准机制，但是有几种方法可以供 USBD 采用，这依赖于它们所控制的外围设备的性质。

（1）使用流接口函数。USBD 可以呈现流接口函数。应用程序可以将外围设备作为文件并使用标准文件输入输出函数来与设备进行交互。然而，因为 USBD 的加载和卸载没有包括设备管理器，所以任何呈现流接口函数的 USBD 必须通过使用 RegisterDevice 和 DeregisterDevice 函数来手工地注册和退出注册它的特殊设备文件名。这些函数应该分别在 USBD 被加载和卸载时调用。

（2）使用现有的 Windows CE 应用程序编程接口（API）。如果 Windows CE 有适于外围设备的现成的 API，通过与 Windows CE API 的交互，USB 设备驱动程序可以间接地给应用程序呈现一定的外围设备类型。比如，用于大数据量存储设备的 USBD，如硬盘驱动和光驱，可以通过标准的可安装文件系统接口系统来呈现这些设备。USB 鼠标驱动程序也使用这些方法。驱动程序不直接将鼠标设备呈现给应用程序，而是通过与现有的 Windows CE API 进行交互从而给系统提交正确的输入事件。因此，鼠标设备的 USB 性质对应用程序是透明的。

（3）创建指定到特定 USBD 的用户定制 API 这种方法在 USBD 呈现设备时，不需要任何限制。它允许你按照应用程序最期望的形式来创建设备的 API，但是你必须给应用程序编写者提供完善的文档使他们可以使用该驱动程序。

4 程序实现

USB 设备驱动程序必须具有入口点，通过入口点建立与总线的连接，故必须输出以下三个函数，缺一不可：

（1）USBDeviceAttach 。用于初始化 USB 设备，取得 USB 设备信息，配置 USB 设备，并且申请必需的资源。

（2）USBInstallDriver 。用于创建驱动程序加载所需的注册表信息，例如读写超时，设备名称等。

（3）USBUninstallDriver 。用于释放驱动程序所占用的资源，以及删除注册表信息等。

下面我们以 USB 鼠标驱动开发为例，介绍 Windows CE 下 USB 设备的驱动实现。

4.1 设备信息获取

编写驱动要先获取设备的一些基本信息，对于不同设备首先将设备安装在其它系统上，如 win 2k，编译运行其 DDK 下的 usbview 例程，然后运行 usbview. exe 即可看得相应的设备信息。有了这些基本信息，就可以编写 USB 设备驱动了。

4.2 USB 设备驱动程序的注册表配置

一般 USB 设备驱动程序的注册表配置在 HKEY_LOCAL_MACHINE \Drivers \USB \LoadClients 下，每个驱动程序的子键都有 Group1_ID \Group2_ID \Group3_ID \DriverName 格式，如果注册表信息与 USB 设备信息符合，USBD 就会加载此驱动程序。否则设备的子键应该由供应商，设备类和协议信息通过下划线组成。

具体配置：USB 鼠标是标准的 HID 设备，它的协议为：InterfaceClassCode 为 3（HID 类），InterfaceSubclassCode 为 1（引导接口类），InterfaceProtocolCode 为 2（鼠标协议类），所以它的注册表配置如下：[HKEY_LOCAL_MACHINE \Drivers \USB \LoadClients \Default \Default \3_1_2 \USBMouse] "DLL" = "usbmouse. dll"，USBMouse 为需要装载的驱动程序名称。

创建驱动程序加载所需的注册表信息是由 USBInstallDriver（）函数完成的，它在设备第一次插入 usb 接

口时调用。需要注意的是:USB 设备驱动程序不使用标准的注册表函数,而是使用 RegisterClientDriverID( ) 和 RegisterClientSettings( ) 来注册相应的设备信息。

卸载驱动程序时会调用另外一个函数 USBUninstallDriver( ),它可以删除 USBInstallDriver( ) 时创建的注册表信息,同样的它使用自己的函数接口 UnRegisterClientDriverID( ) 和 UnRegisterClientSettings( ) 来做相应的处理[3]。

4.3 通信实现部分

有了上述的注册表配置,当 USB 设备连接到计算机上的时候,USBD 会自动匹配注册表,如一致则载入驱动,调用 USBDeviceAttach( ) 方法。在此方法中我们对 USB 鼠标类 CMouse 进行初始化,产生接收 USB 鼠标数据的线程,同时注册监控 USB 设备事件的回调函数,用于监控 USB 设备是否已经拔掉,如拔掉则释放相关资源。主要代码如下:

```
CMouse * pMouse = new CMouse( hDevice, lpUsbFuncs, lpInterface )
pMouse -> Initialize( );
( * lpUsbFuncs -> lpRegisterNotificationRoutine )
( hDevice,
USBDeviceNotifications, pMouse );
```

其中 hDevice 为 USB 设备句柄,lpUsbFuncs 为 USBDI 的函数集,lpInterface 为设备接口描述信息,USBDeviceNotifications 为监控通知函数。

可以看到,在上述的 USBDeviceAttach( ) 方法中使用了一个重要的类 CMouse,在这个类中我们实现了读取鼠标数据并以此产生系统事件,从而完成了整个驱动过程。USB 数据传输方式有四种,分别为控制传输,批量传输,中断传输与同步传输,由于鼠标设备数据量小且随机,故采用中断传输方式,对于其他的 USB 设备将依据特定情况选择不同的传输方式。

Windows CE. NET 中有接口函数叫做 mouse_event( ),专门用于产生鼠标事件,它不关心具体硬件,我们可以应用它来通知系统产生鼠标事件。

下面我们介绍调用实现过程。

以 Initialize( ) 方法为初始运行点,完成对鼠标驱动程序的初始化,在此方法中首先检测 USB 中断管道,建立事件连接,在上述无误的情况下则创建数据接收线程。数据接收线程则开始监控鼠标事件,首先利用 SubmitInterrupt( ) 方法从鼠标设备中读取数据,并产生相应鼠标事件,接着利用 HandleInterrupt( ) 方法处理从鼠标设备中读取出来的数据,从而判断引发鼠标事件的按键并通知系统产生鼠标事件,完成鼠标驱动过程。

以下为部分核心代码:

```
m_hThread = CreateThread(0, 0, MouseThreadStub, this, 0, NULL);
//创建数据接收线程
( * lpUsbFuncs -> lpIssueInterruptTransfer) ( m_hInterruptPipe, MouseTransferCompleteStub, this, USB_IN_TRANSFER | USB_SHORT_TRANSFER_OK,
min ( pInterface -> lpEndpoints [ 0 ]. Descriptor. wMaxPacketSize,
sizeof( m_pbDataBuffer) ),
m_pbDataBuffer,
NULL); //采用中断传输方式读取鼠标数据
mouse_event( dwFlags, dx, dy, 0, 0); //通知系统产生鼠标事件
```

5 结论

Windows CE. NET 虽然发布时间较其他嵌入式系统较晚,但是它对硬件的驱动提供了良好的接口,随着 Windows CE 的普及,越来越多的 USB 外设将会支持 Windows CE. NET 平台。本文以 USB 鼠标驱动程序例对 Windows CE 下的 USB 设备驱动开发进行了介绍,其它的 USB 设备驱动开发过程与其类似,例如打印机设备,其传输数据量大,采用批量传输方式,了解一下 IssueBulkTransfer 等方法的应用,就可以进行开发了。

参考文献

1 陆云峰、李强、母其勇,基于 Windows CE. NET 的嵌入式 PC 视频监控系统[J],计算机工程,2004. 21:179-182。

2 刘炎、冯穗力、叶梧,通用串行总线(USB)原理及接口设计[J],电子技术应用,2000. 12:56-98。

3 宋建才,嵌入式 USB 设备驱动器设计[J],计算机工程,2004. 05:188-191。