WindowsCE 平台下 USB 超声波采集设备驱动的设计与实现①

曹德华，雷跃明

(重庆大学 计算机软件与理论系，重庆 400030)

摘 要：在研究 WindowsCE 平台下 USB 系统架构以及 USB 设备驱动原理的基础上，针对一种采用 Cypress 公司 EZ-USB FX2 系列 USB 控制器的超声波采集设备，详细描述了 WindowsCE 平台下采用流接口方式实现该设备 USB 驱动具体实现过程，成功实现了超声波数据的采集以及超声波设备的控制。

关键词：WindowsCE；USB；设备驱动；流接口；超声波采集设备

Design and Implementation of USB Ultrasonic Acquisition Device Driver in Windows CE

CAO De-Hua, LEI Yue-Ming

(Department of Computer Software and Theory, Chongqing University, Chongqing 400030, China)

Abstract: This paper describes the specific implementation of device USB driver based on studying USB system architecture and principle of USB device driver in WindowsCE platform, the device is ultrasonic acquisition device that use EZ-USB FX2 series USB controllers maded by Cypress company.Using device driver. The paper acquires ultrasnoic data and control ultrasonic device successfully.

Keywords: WindowsCE; USB; device driver; stream interface; ultrasonic acquisition device

1 引言

随着计算机外围接口技术的不断发展，USB 总线凭借它的成本低、速度快、可靠性高、即插即用等优点，很快成为新兴的接口类型。微软 Windows CE 是一个开发且多样化的 32 位嵌入式操作系统，具有较好的实时性和良好的人际交互界面，其内部封装了统一的 USB 系统软件底层接口。但嵌入式硬件环境具有多样性，Windows CE 为 USB 设备驱动开发也仅仅提供了一些底层支持，为此，进行 Windows CE 平台上的 USB 设备驱动开发具有实际意义和价值。

2 USB系统的架构[1]

USB[2]提供了在一台主机和若干个附属的 USB 设备之间通信的功能，一个典型的 USB 系统由一台计算机、一个或多个 USB 设备和物理总线组成。具体如图 1 所示。在图 1 中，我们可以非常清晰地看到主机和物理外设之间的链接方式，在主机端，通过 USBD 模块和 HCD 模块使用默认的管道访问一个通用的逻辑设备[3]，实际上就是说 USBD 和 HCD 是一组抽象出来的访问所有 USB 设备的逻辑接口，它们负责管理所有 USB 设备的链接、加载、移除、数据传输和通用的配置。其中 HCD 是主机控制器驱动，为 USBD 提供底层的功能访问服务，USBD 是 USB 总线驱动，位于 HCD 的上层，利用 HCD 的服务提供较高层次抽象的功能。由于 HCD 和 USBD 都面向一致的逻辑设备接口，那么对于各种各样的物理设备，就需要有唯一的设备驱动程序，即图 1 中位于最顶层的驱动特性的管道所链接的物理设备和 USB 设备驱动程序。

Windows CE 中的系统软件分为两层：即顶层的 USB 设备驱动和底层的 USB 函数。其中，顶层的 USB 设备驱动利用底层的 USB 函数建立于 USB 设备的链接，并负责配置、控制与 USB 设备的通信，而底层的

① 收稿时间:2010-09-21;收到修改稿时间:2010-10-27

© 中国科学院软件研究所 http://www.c-s-a.org.cn

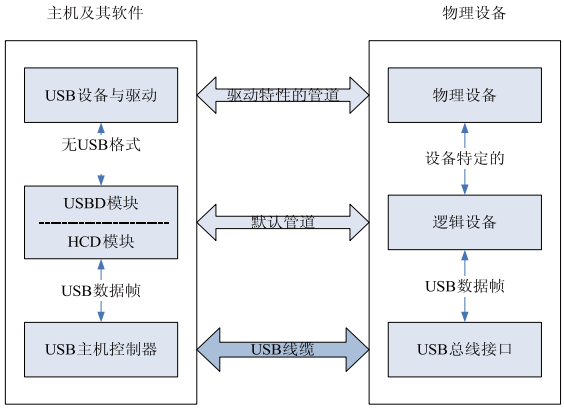

图 1　USB 系统结构

USB 函数完成设备与跟集线器之间的通信、加载卸载 USB 驱动、对 USB 数据进行封包和传输以及建立与 USB 设备端点的通信等功能。

USB 驱动程序主要与 USBD 打交道，在 Windows CE 中，USBD[1]提供了传输、打开和关闭管道，数据打包、获取和设置设备配置以及与 USB 交互的函数，这些函数保存在一个_USB_FUNCS 数据结构中，这个数据结构会由设备管理器在 USB 设备连接到主机上时传给相应的函数。

3　Windows CE下USB设备驱动原理

常见的 Windows CE 下 USB 设备驱动程序的编写有以下几种方法:

第一种方法就是使用流接口方式[1]，在实现了必须的流接口驱动之后，USB 设备驱动还要实现 USBDeviceAttach、USBInstallDriver 和 USBUnistall Driver 函数。为了正确的加载和卸载 USB 设备驱动程序，必须在驱动程序中显示地调用 RegisterDevice 和 DeregisterDevice 函数。使用流接口驱动的好处就是应用程序可以像操作文件一样实现对 USB 设备的操作。

第二种方法是使用特定的 WindowsCE 函数接口，对于某些特定的 USB 类设备，WindowsCE 提供了特定的函数接口。

不管使用哪种方法，USB 设备驱动都必须实现以下三个函数。

函数 USBDeviceAttach，这个函数当 USB 设备连接到计算机上时，USBD 模块就会调用这个函数，这函数主要用于初始化 USB 设备、获取 USB 设备信息、配置 USB 设备并申请必需的资源，这个函数是实现 USB 驱动的关键部分，在第三部分会详细介绍。

函数 USBInstallDriver，这个函数是在当一个未识别的 USB 设备连接到 USB Host 端口时，USBD 模块通过用户键入的或从注册表获取的 USB 设备驱动程序的 dll 文件名称调用这个函数，来创建 USB 设备驱动程序加载所需的注册表、设备名称等信息，并为此设备创建一个唯一的设备标识。

函数 USBUninstallDriver，这个函数主要用于释放驱动程序所占用的资源，以及删除 USBInstallDriver 函数创建的注册表信息等。USBD 模块并不直接调用这个函数，一般只有当需要对一个旧的 USB 设备驱动程序进行更新或彻底删除时，才需要由用户直接调用此函数。

当插入一个设备后，设备管理器就会获取该设备的 VID 和 PID[4]，并在注册表中查找是否有某个子键与插入设备的 VID 和 PID 相同（当然在判断完 VID 和 PID 之后，还会判断设备的 CLASS 和 SUBCLASS 是否存在于注册表中，例如不同品牌的 U 盘有不同的 VID 和 PID，但是都能在主机上正常识别，原因就是在注册表中保存了 U 盘这个 USB 群组驱动程序信息，这里为了简单，只考虑 VID 与 PID）。如果有，表示设备驱动已经安装了，这个子键下包含了关于该设备驱动的详细信息，如设备驱动的 DLL 名、驱动的前缀、占用的资源。否则，就表示没有插入设备的驱动程序。此时会弹出一个对话框，提示用户输入驱动程序的 DLL 名。输入正确的驱动 DLL 名后，设备管理器会调用 DLL 中的导出函数 USBInstallDriver，该函数负责安装设备驱动程序，它会在注册表中写入相关信息，如果安装成功，则在注册表中将新增一个与设备 VID 和 PID 相应的子键，该子键包含了设备驱动的详细信息。安装完后，设备管理重新开始在注册表中查找与插入设备的 VID 和 PID 相同的子键。如果设备管理器在注册表找到了与设备 VID 和 PID 相同的子键，则会从中读取相关信息用于驱动加载。首先，设备管理区调用注册表中制定的 DLL 中的 USBDeviceAttach 函数，并将设备的 VID、PID 等作为参数传给该函数用作判断驱动是否确实可以控制设备。若可以，则设备管理器会以驱动方式（驱动方式加载 DLL 被加载后将常驻内存中不能被内存管理器调页程序换出）来加载这个 DLL。具体过程见图 2。

© 中国科学院软件研究所 http://www.c-s-a.org.cn

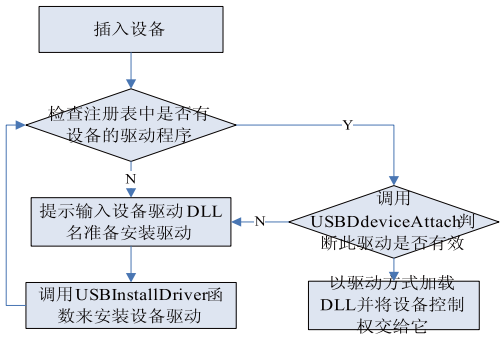

图 2 USB 驱动加载流程

4 Windows CE下USB设备驱动实现

超声波采集设备通过 Cypress 公司 EZ-USB FX2 系列 USB 控制器与计算机进行通信。这种控制器提供固件下载和运行功能，为个人主机外设的制造商提供一个性能优良、价格较低的设计方案。下载固件之前芯片的 VID 和 PID 分别是 0x4b4 和 0x8613，并提供一个控制端点和 30 个额外的端点以及所有的 4 种传输类型。而在下载固件之后，芯片的 VID 和 PID 分别是 0x547 和 0x1002，并且只使用控制端点以及 2 个块端点。超声波采集设备使用控制管道发送设备请求，使用写管道发送同步数据，并从读管道获取超声波数据。

4.1 获取 USB 设备信息

USB 设备信息[2]是用来识别 USB 设备类型，接口、协议及功能等并由 USB 规范规定的一些参数，这些参数在进行 USB 设备的设计过程当中已经以固件的方式写入到 USB 设备的 ROM 存储器，这也是 USB 设备之所以为即插即用设备的本质所在。因此，这些参数是设计 USB 驱动程序所必需的。我们可以将 USB 设备插入到台式机上并运行软件 usbview.exe 就可以获取到这些参数。

4.2 创建一个操作系统设计平台[4]

由于驱动程序的开发调试过程需要与实际的硬件设备打交道，因此在开发之前构建一个针对调试硬件的操作系统平台，为了能进行调试，确认操作系统设计包含组件“USB Host Support”组件和“USB Human Input Device Class Driver”组件。前一个组件式开发驱动程序必须的，后一个组件仅仅是为了对 USB Host 端口进行测试而添加的。

4.3 创建驱动程序框架

在上述的操作系统平台添加 DLL 工程，并在相应的 def 文件添加流驱动需要暴露的接口以及 USB 驱动程序所必须的三个函数接口。然后在 cpp 文件添加相应函数的定义，开始全部置空。后面再根据功能填写实际的操作代码。

从其他类似驱动工程拷贝文件 makefile[4]和 source，makefile 文件不用修改，我们只要修改 source 文件，一般只要在“SOURCE=”后面加上我们项目的 C 或 CPP 文件即可。

在第二部创建的系统设计平台环境下，依次选择“Build OS->Open Release Directory”打开编译环境，使用“CD”进入项目当前目录(makefile,source 当前目录)后输入 BUILD，如果没有错误就会在“%_WINCEROOT%\PUBLIC\COMMON\OAK\TARGET\{_TGTCPU} \{_MODE}”文件夹出现驱动 DLL 文件（其中_TGTCPU[4]是 CPU 类型，_MODE 是 DEBUG 还是 RETAIL）。如果出现错误会在项目当前文件夹下出现 Build.err 文件，当前编译环境也会有错误提示，但是当错误很多的时候编译环境不利于查错。

4.4 编写驱动程序代码

4.4.1 编写 USBInstallDriver 函数代码

这个函数的唯一参数就是用户输入的驱动 DLL 名。按照文章第三部分的分析，知道这个函数主要完成注册表的设置，一般需要调用 RegisterClientSettings 和 GetSetKeyValues 等三个函数。函数 RegisterClientDriverID 为设备创建一个唯一的设备标识，函数 RegisterClientSettings 根据参数为设备添加注册表信息，以后这个设备插入主机时搜索注册表就能根据这些信息直接加载驱动 DLL 文件。GetSetKeyValues 函数封装了访问注册表的方法。

由于 EZ-USB FX2 具有重列举特性，设备在下载固件之后 VID 和 PID 发生改变，为此在这个接口函数中需要为这两组 VID 和 PID 都进行注册。

为了便于程序的维护，我们将要添加到注册表的常量在头文件用宏来表示，例如驱动程序名称字符串、设备标识字符串、注册表键字符串设备驱动程序签名以及设备前缀，当然也可以将注册设备信息中用到的结构体 USB_DRIVER_SETTINGS 变量也用宏表示，结构变量的值在第一步中已经获取到，现在填写到这个变量的相应成员中。

4.4.2 编写 USBUnInstallDriver 函数代码

由于这个函数与函数 USBInstallDriver 以相反的

© 中国科学院软件研究所 http://www.c-s-a.org.cn

操作顺序解除注册表信息，它调用函数 GetSetKeyValues、UnRegisterClientSettings 以及 UnRegisterClientDriveID 完成任务。由于注册了两组 VID、PID 信息，在这个接口函数中就需要解除这两组注册信息。

4.4.3 编写 USBDeviceAttach 函数代码

这个函数原型如下：

BOOL USBDeviceAttach(USB_HANDLE hDevice, LPCUSB_FUNCS lpUsbFuncs, LPCUSB_INTERFACE lpInterface, LPCWSTR szUniqueDriverId, LPBOOL fAccessControl,LPCUSB_DRIVER_SETTINGSlpDriver Settings, DWORD dwUnused)，参数 hDevice 代表 USB 设备句柄，参数 lpUsbFuncs 就是指向 USBD 接口函数表的指针，参数 lpInterface 用于进行接口判断，参数 szUniqueDriverId 和参数 lpDriverSettings 就是函数 USBInstallDriver 注册的信息，而参数 fAccessControl 是一个指向 BOOL 的指针，是一个输出参数，如果函数退出时这个函数为 TRUE 就表示驱动能够控制设备，否则救继续提示用户输入驱动 DLL 名。

(1) 函数的首要任务是确定驱动程序是否能够控制当前 USB 设备，如果能控制，驱动程序继续向下执行，否则程序退出。这可以通过验证 USB 设备的描述符信息（参数 lpInterface）来判断。

(2) 函数的第二项任务是对 USB 描述符进行解析，以获取当前 USB 设备的描述符、接口配置、端点、设备索引等信息，从而获得 USB_INTERFACE 结构中各个结构变量的值。

(3) 设备驱动对应于一个设备对象，设备对象是对设备驱动的抽象。超声波设备使用的设备对象结构如图 3 所示。在完成了设备驱动程序上下文结构体的定义之后，就可以创建设备对象并进行初始化。

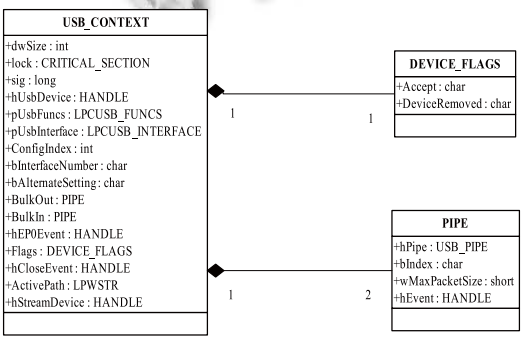

图 3　USB 设备对象

(4) 函数的第四项任务是设置 USB 接口，就是图 3 设备对象中成员 BulkOut 和 BulkIn，其目的就是对 USB 设备进行设置，并创建设备各个端点对应的传输管道和同步事件，接着创建控制端点 0（默认管道）关联的同步事件。

(5) 流接口驱动一般是由设备管理器或用户程序调用 ActivateDevice 函数来加载的，但是作为 USB 设备驱动，当 USB 设备被插入 USB Host 端口时，设备管理器首先调用的是 USBInstallDriver 函数和 USBDeviceAttach 函数，并没有直接调用 Activate Device 函数，因此，我们必须在 USBDevice Attach 函数中显示调用 ActivateDevice 函数来加载流驱动接口。在这里我们将第 3 步申请的设备对象地址作为 ActivateDevice 函数的第二个参数，这样在设备相关注册表键[5]"ClientInfo"下可以获取设备对象地址值。

(6) 注册 USB 回调函数以处理 USB_CLOSE_DEVICE 通知消息，这可以通过调用 USBD 的 LPREGISTER_NOTIFICATION_ROUTE 函数来实现。

(7) 最后，函数根据返回值为 FALSE 或 TRUE 进行相应的清理或设置。

4.4.4 编写 ULT_Init 和 ULT_Deinit 函数代码

ULT_Init 函数是流接口驱动的入口，当 USBDeviceAttach 函数调用 ActivateDevaice 函数来加载流驱动接口时，ULT_Init 函数被立即执行，如果这个函数返回非零值，那么流接口驱动被加载成功。我们在加载 USB 驱动时将设备对象地址值保存在注册表中，在这个函数中可以通过函数第一个参数的键"ClientInfo"获取到这个设备对象地址值并返回。ULT_Deinit 函数释放 ULT_Init 函数分配的资源并终止驱动程序线程。

4.4.5 编写 ULT_Open 和 ULT_Close 函数代码

ULT_Open 函数中第一个参数是 ULT_Init 的返回值，即设备对象地址值，一般在这个函数中只要验证设备对象的有效性并继续返回对象地址值。ULT_Close 函数关闭 ULT_Open 打开的设备。上层应用调用 CreateFile[5]和 CloseHandle 时会触发这两个函数。

4.4.6 编写 ULT_Read, ULT_Write, ULT_IOControl 以及其他函数

如果功能需要，ULT_Read 和 ULT_Write 一般用来数据传输，ULT_IOControl 用于向设备发送命令，这些函数的第一个参数是 ULT_Open 的返回值，即设

(下转第 92 页)

© 中国科学院软件研究所 http://www.c-s-a.org.cn

结构，并采用拓扑重构算法得到杭州低层路网结点数为4874、路段数为 5928，高层路网结点数为 786、路段数为 1505。本试验以上述数据为基础，使用分层路径搜索算法，通过设定以下三个条件，对两个结点之间路径最优搜索的时间进行对比，得到测试结果如表 2。

(1)路网数据未采用分层索引（A*算法搜索）

(2)分层索引未使用 G-SDBCScan 索引结构。

(3)分层索引算法使用 G-SDBCScan 索引结构。

表 2　实验数据对比

	起点 ID	终点 ID	时间（ms）
未使用分层索引	1	100	3240~3412
	200	1000	4362~4510
分层索引（G-SDBCScan 索引算法）	1	100	2110~2198
	200	1000	2870~3010
分层索引（未加 G-SDBCScan 索引算法）	1	100	2850~2946
	200	1000	3972~4012

从上表我们可以看到在设定的三个不同条件下，导航路径规划的效率是不同的。条件一的情况下缺乏对抽象理论分层作用的重视，因而效率是最低下的。条件二中虽然使用分层索引，但是由于数据读取的发散性，效率并不是最高。条件三是本文研究的分层索引机制（G-SDBCScan 索引结构）能够在一定程度上提高路径规划的效率，具有实用性。

4　结语

本文研究并提出了一种 G-SDBCScan 索引算法，并将此算法应用于地图数据库分层索引机制中。通过实验数据表明，引入 G-SDBCScan 算法的分层索引能够有效地提高导航路径寻优算法在读取外存数据有限的情况下的效率，同时也对导航系统路网数据底层存储作出了进一步的研究工作。

参考文献

1 赵芳,潘秋生,李建元.道路网络的分层模型与重建算法研究.农业与技术,2008,28(1):162−166.

2 刘汉丽,李霖,李清泉,周成虎.用于车辆导航的路径规划数据逻辑模型.测绘科学技术学报,2008,25(1):13−17.

3 李楷,钟耳顺.车载电子地图数据物理存储技术研究.中国测绘学会 2006 年学术年会论文集.北京:中国科学院研究生院,2006.110−115.

4 夏启兵.基于关系数据库的地理数据库引擎的研究与实践.郑州:中国人民解放军信息工程大学,2002.

5 沈永增,姚萌萌,周巍.空间数据在嵌入式导航系统中的索引.计算机系统应用,2010,19(4):85−88.

6 高伟.地理空间数据库引擎的设计与实现[硕士学位论文].郑州:解放军信息工程大学,2007.

7 管希萌,刘瑶,徐丽仙,田永晔.嵌入式数据库 SQLite 应用研究.扬州教育学院学报,2008,26(3):18−22.

(上接第 136 页)

备对象的地址值。采集设备所有功能都是通过发送 IO 控制命令完成的，因此只实现 ULT_IOControl 这个接口函数，其他函数直接返回。

4.5 编译驱动程序

按照 3.3 节中介绍的方法进行编译，如果没有错误就能得到驱动 DLL 文件，拷贝到设备上运行并进行测试。

5　结语

随着嵌入式系统的普及及 USB 接口外设的广泛出现，如何编写使用于嵌入式系统的 USB 驱动越来越受到重视。本文在研究 WindowsCE 环境下 USB 系统架构以及 USB 设备驱动原理的基础上，针对一种采用 Cypress 公司 EZ-USB FX2 系列 USB 控制器的超声波采集设备使用流接口方式实现了设备的 USB 驱动。本文给出开发 WindowsCE 平台下 USB 流驱动的具体过程，其他 USB 设备流驱动可以参照本文所述方式进行开发。

参考文献

1 张冬泉,谭南林,苏树强.Windws CE 实用开发技术.北京:电子工业出版社,2009.

2 武安河,邰铭,于洪涛.USB 设备驱动程序开发.北京:电子工业出版社,2003.3−99.

3 赵晶莹,郭海,宋海玉.Windows CE.NET 下的 USB 设备驱动实现.计算机系统应用,2006,15(2):83−85.

4 何宗键.Windows CE 嵌入式系统.北京:北京航空航天大学出版社,2006.

5 微软公司.Microsoft Windows CE Driver Kit.北京:希望电子出版社,1999.

© 中国科学院软件研究所 http://www.c-s-a.org.cn