基于 VB 的 USB 设备检测通信研究

徐　袭, 杨志红, 吴汉松
(海军工程大学 电气工程系, 湖北 武汉 430033)

摘　要: 通过分析 USB(Universal Serial Bus)的特点及内部结构,研究在 Windows 环境下用 Visual Basic 实现对 USB 设备检测通信方法。在 Windows 接口通信和 USB 设备驱动的基础上,使用 Windows API 函数实现了对 USB 设备的检测和通信。

关键词: USB; Visual Basic; Windows API; 设备检测
中图法分类号: TP311.11　　**文献标识码**: A　　**文章编号**: 1001-3695(2002)11-0099-04

Research on Checking and Communication of USB Equipment Based on Visual Basic

XU Xi, YANG Zhi-hong, WU Han-song
(Dept. of Electrical Engineering, Naval University of Engineering, Wuhan Hubei 430033, China)

Abstract: Researches on the USB equipment checking and communication based on Visual Basic under the Windows environment through analyzing the characteristics and internal construction of USB. With the Windows API function, the checking and communication of USB equipment is come into true based on the interface communication of Windows and driver of USB equipment.

Key words: USB; Visual Basic; Windows API; Equipment Check

1　引言

随着计算机的日益普及与发展,计算机外围设备越来越多,每个外设都需要通过一个接口与计算机相连,进行控制与通信。然而,外设的增多自然会带来一些问题:①外设增多,接口也要随之增多,但计算机总的接口资源有限;②伴随技术的发展,新外设对数据传输速度又有新的要求,已有的传统接口满足不了新的需要;③计算机逐步向简单化、易用、可靠等方向发展,相应要求接口设备具有安装简单、支持热插拔等功能。

通用串行总线(Universal Serial Bus,USB)设备可以满足上述新要求。USB 设备与传统计算机外围设备相比,具有数据传输快,速度最大可至 12Mbps(USB 2.0 支持 488Mbps);易扩展,通过 USB HUB 扩展可以连接多达 127 个外设;支持热插拔,使用 USB 设备不用开关机;支持对外设的直接操作,在主机和外设之间可以同时传输多个数据集信息流等优点。计算机外设制造商广泛采用 USB 标准开发产品,USB 设备产品越来越丰富。目前所有新 PC 机箱上都有数个 USB 端口,以供数量众多支持 USB 的计算机外设使用。USB 技术已经成为计算机易于传输、操作的关键技术之一。

Visual Basic 是 Windows 系统下可视化应用程序开发平台,属于 RAD(快速程序开发)工具,具有可视化、面向对象、事件驱动特点的结构化高级程序语言。VB 提供 Windows 应用程序接口(Windows API)、动态连接库(DLL)支持、开放式数据管理(ODBC)以及 ActiveX 组件技术,可以简单巧妙的将 Windows 编程的复杂性封装起来,使 Windows 下的可视化应用程序编程和调试变得简单、方便。用 Visual Basic 可高效地、快速地构建起 Windows 环境下功能强大的应用软件系统。

在应用计算机对其接口设备进行检测和通信开发时,结合 VB 的特点,用 VB 作为设备检测通信应用程序的开发工具是一个不错的选择。VB 开发计算机接口设备检测程序时,由于计算机 USB 设备的增多和使用范围的扩大,使得针对 USB 设备的检测通信开发显得越来越重要,要求应用程序直接支持对计算机 USB 设备的检测和通信等内容。本文针对这一问题,提供了利用 VB 对 USB 设备进行开发的一种方法。

2　串并口设备

计算机串口设备使用 RS-232C 标准进行通信。通过九针 D 形接口与外设的接口进行连接, 进行设备检测和通信。RS-232C 串口的传输速率不高, 最高波特率约在 9 600bps 左右,使用 RS-232C 串口进行数据传输时,由于串口自身的内部结构局限性,影响通信质量,满足不了当前先进的计算机外设数据通信要求。

VB 中对串口设备进行检测通信编程可使用自带控件 MSComm,或使用 Windows API 函数。MSComm 控件是

收稿日期: 2002-04-10; **修返日期**: 2002-04-30

VB 中专门用于管理串行通信的控件,控件通过串行端口传输和接收数据,为应用程序提供串行通信功能。但每个 MSComm 控件对应一个串行端口,如需访问多个串行端口,需使用多个 MSComm 控件来对应处理。控件提供两种通信处理方式:一种是利用 MSComm 控件中的 OnComm 事件捕获并处理这些通信事件的中断方式;另一种是在程序的每个关键功能后,通过检查 CommEvent 属性的值来查询事件和错误的查询方式。由于 MSComm 控件本身将关于 Windows 串行通信底层封装,减少了编程实现的工作量,故在 VB 中通过 MSComm 控件对串行接口设备的检测通信简单,易实现。

在计算机外设接口中,并行接口设备的检测通信也是一个不可缺少的部分。并行通信具有传输速度快(50～100Kbps)、扩展电路简单、兼容性好等优点。利用 VB 对并口设备进行检测通信时,与串口设备相比,由于 VB 自身的接口不能直接访问并口寄存器,也没有可以直接供使用的并口通信控件,也就不能直接简单地对并口设备进行编程。

要直接对并口设备进行读写,就必须编写对并口寄存器访问的动态链接库(DLL)。VB 中对于动态链接库的编写比较复杂,涉及到与 C++ 语言的混合编程,不适宜作简单快速的设备检测通信的接口开发。

3 USB 设备规范

目前 USB 主要有两种设备规范:USB1.1 标准和 USB2.0 标准。后者是在前者的基础上面发展起来的,速度最快可达 480Mbps 的一种规范,并且向前兼容。在 USB 设备规范中存在两种结构:USB 拓扑结构和 USB 逻辑结构。

3.1 USB 拓扑结构

USB 系统由主控制器、USB 集线器和 USB 设备组成。其系统的拓扑结构如图 1 所示。

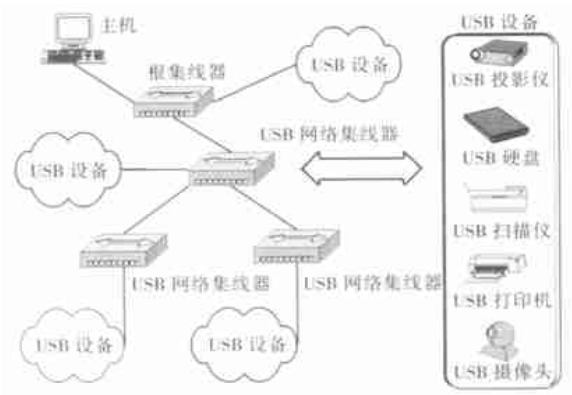

图 1 USB 系统的拓扑结构图

在 USB 拓扑结构中,主控制器由硬件、系统软件和应用软件构成,主控制器可以与 USB 集线器相连,并可以通过它来扩展 USB 口,使系统连接更多的外设。USB 设备是指带有 USB 口的计算机外部设备,使用标准的 USB 数据结构与主机进行通信,识别主机命令并响应。可以看出,USB 系统的拓扑结构是基于树状的拓扑结构形式。

3.2 USB 逻辑结构

对于 USB 连接的主机来说,对每个外设的控制与通信都是一致的,如同外设与主机直接相连。主机把 USB 集线器也同样作为一个普通的 USB 设备来进行管理。逻辑上 USB 主机与 USB 设备件之间的层次划分如图 2 所示。

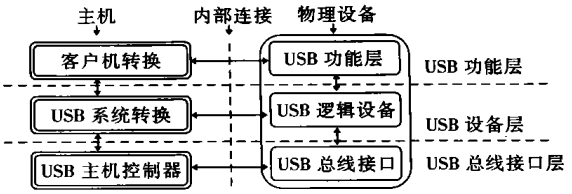

图 2 USB 系统逻辑结构图层次划分

通过 USB 逻辑结构图中对于 USB 系统的层次划分,可知道相应的应用程序就是通过调用 USB 系统软件中相应的功能,与 USB 设备完成数据交换;USB 系统软件通过与主机硬件之间的寄存器接口和共用存储器接口,完成与设备间的数据交换;USB 控制器通过物理连接与设备完成信号交换,层与层之间遵循总线规则。

4 USB 接口检测方法

VB 中没有封装 USB 通信的控件,所以只有在熟悉 Windows 接口通信机制和流程的基础上,用 VB 调用系统 API 函数,实现对 USB 设备的检测通信。

4.1 Windows 接口通信

由于大多数计算机用户使用的是 Windows 操作系统,Windows 操作系统下对计算机接口编程必然要涉及到 Windows 操作系统与计算机外部接口设备的通信方式。当然,USB 设备的通信也包含在其中。

Windows 系统下应用程序与接口设备通信流程结构图如图 3 所示。

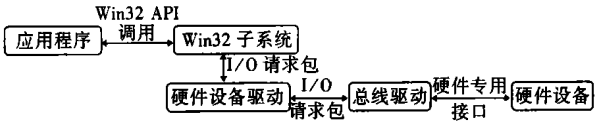

图 3 应用程序与接口设备通信流程结构图

在 Windows 下每个应用程序和驱动使用自己的语言与操作系统通信,应用程序使用 Win32 API 函数。驱动通信使用 I/O 请求包(IRPs) 的结构。Windows 定义了一套驱动可以使用的 I/O 请求包(IRPs)。每个 I/O 请求包(IRPs) 请求或执行一个单个的输入或输出动作。USB 设备驱动使用(IRPs) 传递总线通信处理 USB 通信。在一系列通信中最终的总线直接驱动硬件,总线包括 Windows 下不需要编程的应用程序。

应用程序与接口通信使用 Win32 API 函数与对应接口设备驱动程序间相互通信。Win32 API 函数能使应用程序控制计算机的显示器、磁盘以及与其它接口设备间的通信。对 USB 设备的检测通信必定要使用 Win32 API 函数。

4.2 USB 设备驱动

USB 设备驱动使应用程序不需要知道物理连接信

号和与该设备通信需要的协议等细节,直接通过使用 Win32 API 函数与 USB 设备进行对话。应用程序是用户运行的程序,其中包括一些支持自定义硬件有特殊用途的应用程序。USB 设备驱动可以保证应用程序的代码只通过外部设备名字来访问外设或者端口的目的地址。应用程序不需要知道外设连接端口的具体物理地址,也不需要精确地监视和控制外设需要的交换信号。

Windows 环境下的 USB 设备驱动使用分层模式进行设备驱动。每个驱动执行一部分通信。最顶层与应用程序通信对话。最底层与硬件通信,中间夹杂一层或多层。分层驱动模式看起来较复杂,但实际上简化了工作,使得设备可以共享通用任务的代码功能。处理系统 USB 硬件设备通信的驱动建立 Windows 操作系统之上,因此一些 USB 设备的驱动根本不需要提供。设备驱动处理单一设备之间的通信,单个的 USB 设备可能使用一个或者多个设备驱动。Windows 操作系统中 USB 设备驱动定义遵循 Microsoft 定义的 Win32 驱动模式。驱动方式以 Win32 驱动模型(Win32 Driver Model,WDM)驱动为主。由于 Windows 中的 USB 设备的总线驱动是 WDM 驱动,所以 USB 设备必须使用 WDM 设备驱动,该驱动必须和系统的总线驱动通信,驱动的类型必须一致。

Visual Basic 对在 Windows 环境下的一般任务有它自己的控制。前面提到的串口通信,使用简单的 VB 控件可直接与对应的接口通信。对于需要接触到控制系统底层的通信,代码通过使用 Win32 API 函数来与设备驱动通信。但是 Visual Basic 并不包括访问一般 USB 设备的控制,VB 中并不提供如串口通信的 MSComm 控件来简化接口通信的过程。使用 VB 访问控制 USB 设备必须使用相应的 API 函数来进行。通过 API 函数与 USB 设备驱动通信,达到检测控制 USB 设备的目的。

4.3　USB 设备检测

应用 VB 编制 Windows 下检测 USB 设备应用程序时,使用 Win32 API 函数与 Windows 系统下的 USB 设备驱动会话。Windows 系统中自带的 USB 设备驱动的选择,有下列方式:①HID(Human Interface Device,人机接口设备)驱动、驱动鼠标、键盘等设备;②Point-Of-Sale(POS)驱动,驱动包括条码扫描仪等用于规模事务的设备;③USB 设备供应商提供的驱动,驱动供应商提供的设备;④其它的一般驱动,如驱动一些 USB 控制芯片等。

利用 Windows 系统的强大功能,可以选择 HID 类驱动作为应用程序在 Windows 环境下应用程序对 USB 设备进行检测的设备驱动。虽然,HID 类的 USB 设备驱动并不能驱动所有的 USB 设备,但是只要 USB 设备中的描述符符合 HID 类的描述,并使用 HID 规范中定义的数据传输方式,HID 类是可以完成对该 USB 设备驱动的。对于不符合相应条件约束设备,可以定义或更改其设备描述符和传输模式以符合 HID 类驱动的标准。

应用程序在使用 Windows 下的 HID 类对 USB 设备进行设备检测时,需要使用的几个主要的动态链接库(DLL)、API 函数和用途列表如表 1 所示。

应用程序使用列表中的 API 函数与系统的 USB 设备驱动通信。对于 API 函数的调用方法,在各种编程语言中有不同的调用方式。在 VB 中对 API 函数的调用一般可以通过 VB 提供的 API 浏览器工具直接拷贝到程序中使用。API 函数其对应的结构体,常数值均已定义完好。然而,对于一些不经常使用的或是尚未公开的 API 函数,就需要查找相应的资料明确函数的功能及调用方法。API 函数资料可以在 DDK 文档或者 MSDN 中查找,相应的函数都有详细的说明及使用方法。相应的 API 函数找到后,需要对函数的声明略作修改,才可在 VB 中使用其强大的功能。

表 1　主要 API 函数列表

DLL 库	API 函数	用　途
Hid.dll	HidD_GetHidGuid	获得 HID 类的 GUID(标志符)
Hid.dll	HidD_GetAttributes	获得 USB 设备的 ID 和版本号
Hid.dll	HidD_GetPreparsedData	返回句柄保存设备能力信息的缓冲器
Hid.dll	HidP_GetCaps	返回描述设备能力的结构体
Hid.dll	HidP_GetValueCaps	返回描述设备值的能力的结构体
Hid.dll	HidP_GetButtonCaps	返回描述设备按钮的能力的结构体
Hid.dll	HidD_FreePreparsedData	释放 HidD_GetPreparsedData 使用资源
Setupapi.dll	SetupDiEnumDeviceInterface	返回设备组中一个设备的信息
Setupapi.dll	SetupDiDeviceInterfaceDetail	返回一个设备路径名
Setupapi.dll	SetupDiGetClassDevs	返回特定类所有设备的设备信息组
Kernel32.dll	CreatFile	开放与一个设备的通信

HID 类中使用的就是一些不经常使用的 API 函数。在 VB 中使用这些 API 函数检测 USB 设备也就是查找应用程序与之通信的 USB 设备,通过检查系统提供的 HID 中的属性找到对应的匹配设备,完成 USB 设备的检测任务。具体在 VB 中调用 API 函数实现对 USB 设备的检测流程及过程如下:

(1)应用程序获取所有 HID 类设备的 GUID(标志符)。调用 API 函数 HidD_getHidGuid 来获取 GUID。由于函数没有返回值,可以声明成子程序或函数的形式,在函数形式中可以忽略。程序获取 GUID 信息,但并不对它操作,只是下一步程序调用 API 函数功能时使用到的参数。GUID 信息以结构体的形式传递。函数及结构体的形式如下:

```
Public Declare Function HidD_GetHidGuid Lib "hid.dll"(ByRef HidGuid As GUID) As Long' 其中 hid.dll 就是调用对应的 DLL,函数名为 HidD_GetHidGuid
```

对应的结构体的表示形式为:

```
Public Type GUID
…
End Type
```

(2)应用程序获取 HID 类设备信息数组。GUID 代表系统上的 HID 设备的信息,使用 Windows 中的设备管理函数声明:

```
Public Declare Function SetupDiGetClassDevs Lib "setupapi.dll" Alias "SetupDiGetClassDevsA" (ByRef ClassGuid As GUID, ByVal Enumerator As String, ByVal hwndParent As Long, ByVal Flags As Long) As Long。
```

函数的声明形式与前面相比没有太大的区别,只是各自对应的参数不同。函数返回的 DeviceInfoSet 结构体数组的地址,包含所有已经连接列举的 HID 设备信息。信息供其它 API 函数使用。在使用完信息后,释放所占用的资源。

(3)应用程序检测识别 HID 类设备接口。调用 DeEnumDeviceInterfaces 函数,获得前面的结构体的指针,

由传递一个数组指针来指定一个接口,逐步的操作数组,直到返回结果为零的时候为止。因为,当你使用某个接口时,应用程序依靠的信息有限,如果有多个接口被返回后,应用程序需要调用每个接口,找到正确的接口。函数的声明如下:

Public Declare Function SetupDiEnumDeviceInterfaces Lib "setupapi.dll" (ByVal DeviceInfoSet As Long, ByVal DeviceInfoData As Long, ByRef InterfaceClassGuid As GUID, ByVal MemberIndex As Long, ByRef DeviceInterfaceData As SP_DEVICE_INTERFACE_DATA) As Long

函数使用的就是前面调用的 HidGuid 和 DeviceInfoSet 两个传递值。

(4) 应用程序获取 HID 类设备路径名。使用前面识别的设备接口信息,调用 SetDiGetDeviceInterfaceDetail 函数,返回设备的路径名,使应用程序可以同该接口设备通讯。在函数声明中的 DeviceInterfaceDetailData 参数返回的结构体式一个用户定义类型,定义形式如下:

```
Public Type SP_DEVICE_INTERFACE_DETAIL_DATA
    cbSize As Long
    DevicePath As Byte
End Type
```

对于结构体地址的传递,首先在存储器中间分配缓存来保存这个结构体,再使用 VarPtr 操作符获取缓存的起始地址,通过值的方式传递地址。拷贝缓存中的数据到结构体中,提取需要的 USB 设备的路径值。

通过由应用程序对 API 函数调用的四个步骤,可以获取相应 USB 设备的路径值。设备的路径值就可作为对 USB 设备检测是否成功的判断标志。判断返回值是否返回 USB 设备的路径值,在使用众多对 Windows 系统操作的 API 函数条件下,可简单、方便地完成。

4.4 USB 设备通信

应用程序通过以上步骤对 USB 设备进行检测后,才可以与 USB 设备进行通信。读取 USB 设备中的信息,通过应用程序将信息传递到 USB 设备中。当然,与 USB 设备的通信同样是通过应用程序调用 Windows API 函数来实现的。使用 VB 编写应用程序时调用的 API 函数如下表 2 所示。

表 2 调用 API 函数列表

DLL 库	API 函数	用 途
Kernel32.dll	CreateFile	开放与一个设备的通信
Kernel32.dll	WriteFile	发送一个报告给设备
Kernel32.dll	ReadFile	从设备读取报告
Kernel32.dll	CloseFile	释放被 CreateFile 使用的资源

应用程序使用表中描述的 API 函数,完成与 USB 设备的通信步骤如下:

(1) 应用程序获取 USB 接口设备的操作句柄。在已经确认了 USB 设备的情况下,应用程序获得了 USB 设备的路径名。通过使用 API 函数 CreateFile 开放应用程序同 USB 设备的通信。CreateFile 可以打开 HID 类中驱动支持 CreateFile 的设备,USB 设备也在其中。CreateFile 函数返回值是一个其它 API 函数可以用来同设备交换数据的句柄。函数的声明如下:

Public Declare Function CreateFile Lib "kernel32" Alias "CreateFileA" (ByVal lpFileName As String, ByVal dwDesiredAccess As Long, ByVal dwShareMode As Long, ByRef lpSecurityAttributes As Long, ByVal dwCreationDisposition As Long, ByVal dwFlagsAndAttributes As Long, ByVal hTemplateFile As Long) As Long

当应用程序不再访问设备时,通过调用 CloseHandle API 函数释放系统的资源。

(2) 应用程序写数据到 USB 设备中。当应用程序获得 USB 设备的句柄后,可以写数据到 USB 设备中。应用程序将数据发送到缓存中调用 WriteFile 函数。缓存的大小同 HidP_GetCaps 返回的 OutputReportBytelength 显示的大小相等。WriteFile API 函数调用是一个普通的 API 函数调用,支持这个函数的文件或设备均可以调用。函数的声明如下:

Public Declare Function WriteFile Lib "kernel32" (ByVal hFile As Long, ByRef lpBuffer As Byte, ByVal nNumberOfBytesToWrite As Long, ByRef lpNumberOfBytesWritten As Long, ByVal lpOverlapped As Long) As Long 调用该函数前先将内容写入到需要发送的数组中保存,然后直接调用 WriteFile 函数将数据写入到 USB 设备中。

(3) 从 USB 设备读取数据到应用程序中。与 WriteFile 相对应的 API 函数就是 ReadFile API 函数。应用程序就是通过使用 ReadFile 函数读取 USB 设备的数据。数据的读取同样需要一个足够大的缓存用来保存数据内容。缓存的大小的定义可根据 HidP_Caps 返回中 InputReportByteLength 的属性值的大小来设定。ReadFile 函数是一个普通的 API 调用函数。函数的声明如下:

Public Declare Function ReadFile Lib "kernel32" (ByVal hFile As Long, ByRef lpBuffer As Byte, ByVal nNumberOfBytesToRead As Long, ByRef lpNumberOfBytesRead As Long, ByVal lpOverlapped As Long) As Long

在使用 ReadFile 函数读取数据时要防止应用程序会出现挂起状态,即系统不响应应用程序要求,不对 USB 设备读取数据。可以将 ReadFile 放在独立的进程中完成数据的读取,防止应用程序被挂起,这涉及到 VB 条件下的多线程的问题。

(4) 应用程序关闭与 USB 接口设备的通信。当应用程序完成与设备的通信时,应该释放前面打开设备通信时占用的系统资源。使用 CloseHandle API 函数来释放占用的系统资源。在关闭占用的系统资源的时候,确认已经使用了前面提到的读取设备数据和写入设备数据的函数。函数的声明如下:

Public Declare Function CloseHandle Lib "kernel32" (ByVal hObject As Long) As Long

函数的返回值为零,表示对系统占用的资源释放成功。

通过上面的步骤实现了应用程序对 USB 设备的数据读写,完成了对 USB 设备的通信功能。

5 应用实例

应用前述提供的 USB 设备的检测方法,在 Visual Basic 中编制成相应的检测 USB 设备的应用程序,完成了对 USB 设备的检测和通信。某 USB 设备的检测通信程序界面如图 4 所示。

在该界面中完成了对 USB 设备的检测和通信。利用 VB 中的定时器功能完成了对 USB 设备(下转第 111 页)

(2) CLASSID 是该控件惟一的 UUID,告诉 IE 装入哪个对象。如果使用已经开发好的控件,它的 CLASSID 可以通过调用 Win98 或 NT 下的应用 Regedit 来查找。展开 HKEY_CLASSES_ROOT 项,找到需要使用的控件名,在展开时可看到一个 CLSID 文件夹,里面就是该控件的 CLASSID。

如果是自己用 VC 开发控件,该 UUID 可以在 ActiveX 控件项目中的 ODL(对象描述库)文件中找到,通过查看控件的类信息注释来定位特定控件的 UUID,例如,要找到 CMyControl 控件的 UUID,则需要找到以下代码:

```
//
Class information for CMyControl
[uuid (051C4748-1262-11D2-87C1-00A024D948FB),
licensed,
    helpstring("CmyControl Control"), control ]
```

uuid 后面括号中的内容就是该控件的 UUID。

(3) CODEBASE　如果在用户机器上没有控件的当前版本,该参数告诉用户浏览器在哪里可找到要下载的控件和最新版本号。当控件作了修改后,可以更改版本号强制用户重新下载。

(4) PARAM　该标记用于设置控件的初始属性值,它有两个特性:Name 和 Value,即属性名称和属性值。

此外还有一些标记,如:Width 表示该控件所占的宽度,Height 表示高度等。

下面是在网页中嵌入 Open 控件的部分 HTML 代码:

```
<OBJECT ID = "Open" CLASSID = "clsid:966A7626-18D0-11D3-95DB-
0088CC160726"
CODEBASE = "http://shi/cfweb/open.cab#version = 4,0,1,2"
WIDTH = 626 HEIGHT = 377 >
<PARAM NAME = "Handmove" VALUE = "1">
<PARAM NAME = "Length" VALUE = "0">
</OBJECT>
```

3.4　系统安全认证

这部分内容对系统安全起决定性作用。它不是一个独立的部分,而是融入到整个系统的各个环节,起到既方便合法用户的使用,又能防止非法侵入的功能。它包括服务器端的中心数据库,以及系统管理员使用的管理软件和对系统事件自动记录的系统日志,还包括为系统核心用户和外围用户设计的身份确认软件。系统的工作过程如图 4 所示。

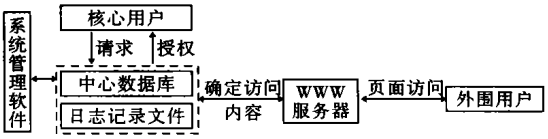

图 4　系统的安全体系

4　结束语

本文介绍的基于 DCOM/ ActiveX 技术的地下综合管线管理 WebGIS 系统已经在成飞动力公司实际运行。

参考文献:

[1] 宋关福,钟耳顺,王尔琪. WebGIS ——基于 Internet 的地理信息系统[J]. 中国图像图形学报,1998,(3):253-254.

[2] 周炎坤,李满春. WebGIS 开发方法比较研究[J]. 计算机应用研究,1999,16(11):44-46.

[3] 王志兵,李满春,等. 基于 IMS 的 WebGIS 应用开发[J]. 计算机应用研究,2001,18(3):120-121.

[4] Microsoft Corporation,MSDN Library Visual Studio 6.0[CD], 1997.

[5] http://msdn.microsoft.com/library/default.asp?URL=/library/devprods/vs6/visualc/vccore/_core_activex_control_topics.htm[EB/OL].

作者简介:

卢选民(1972-),男,研究生,研究方向为计算机网络、多媒体通信、GIS 系统和网络数据库技术;刘咏芳(1974-),女,研究生,主要研究方向为电子商务、计算机网络和数据库技术;史浩山(1947-),男,教授,博士生导师,主要研究方向为计算机网络、多媒体通信、GIS 系统和图形图像处理。

(上接第 102 页)的监测功能,显示 USB 设备的检测状态和设备的属性。该 USB 设备检测程序模块已经作为某科考船测控系统中通信设备状态控制检测软件的一部分,工作良好,基本满足需要。

图 4　程序界面

6　小结

对于 USB 设备的检测方法有很多种,本文提供在 VB 环境下对 USB 设备的检测实现方法,相信在其它的 USB 设备检测编程实现方法中还有许多更好、更简单的方法。希望对读者有所帮助。

参考文献:

[1] 黄嘉辉,黄悦珊. Visual Basic 与 Windows API 程序设计高手[M]. 北京:清华大学出版社,2001.

[2] 李鸿吉. Visual Basic6.0 中文版编程方法详解[M]. 北京:科学出版社,2001.

[3] Matthew Curland. 高级 Visual Basic 编程[M]. 涂翔云,等. 北京:中国电力出版社,2001.

[4] [美]艾柯尔逊. USB 大全[M]. 陈逸,等. 北京:中国电力出版社,2001.

[5] 汪胜,时亚弘. USB2.0 技术概述[J]. 计算机应用研究,2001,18(3):4-6.

[6] 黄维柱,许军. 通用串行总线 USB[J]. 计算机应用研究,2001,18(2):46-48.

[7] 张宏伟. Linux 下 USB 设备驱动程序的编写[J]. 计算机应用研究,2001,18(9):141-146.

[8] Jane Axelson. Lakeview Research:Usbhidio[Z]. 1999.

作者简介:

徐袭(1978-),男,硕士生,主要从事软件开发、控制系统仿真研究;杨志红(1977-),男,博士生,主要从事嵌入式软件开发、混沌理论研究;吴汉松(1957-),男,副教授,主要从事控制系统理论、系统仿真研究。