基于 VC + +和 LabWindows/CVI 的信号采集和处理系统①

System of the Signal Collection &Processing Based on LabWindows/CVI and VC + +

张秀再　陈钟荣　朱　丽　(南京信息工程大学 江苏 南京 210044)

摘　要：本文介绍了 USB 接口应用于数据采集的硬件和软件开发过程以及 VC + +和 LabWindows/CVI 混合编程实现数据处理方法。USB 接口具有简单、灵活、较高的传输速率和"即插即用"的特性,所以适合用于数据采集的通信接口。应用程序部分利用 VC + +的 MFC 框架的兼容性、稳定性和 LabWindows/CVI 的信号处理函数混合编程实现对信号处理和结果显示,具有交互式编程方法和丰富的库函数等功能,该系统具有灵活性和扩充性。

关键词：USB VC + + LabWindows/CVI　混合编程　数据采集和处理

随着信息技术的迅速发展,数据采集和信息处理技术广泛应用于雷达、通信、遥测遥感等领域。USB 2.0 以其 480Mbps 的带宽和优越的性价比得到了众多软硬件制造商的支持,已成为新一代的计算机外设接口的主流标准[1]。将 USB 技术应用于数据采集可以达到数据采集系统的高速度处理。而虚拟仪器对信号处理来讲是对传统仪器概念的重大突破,它是计算机技术与仪器、仪表技术相结合的产物。虽然国外已有分析精度高、功能多的发生器、测试分析仪器等传统仪器,但价格昂贵,操作比较复杂。而国内分析仪器大多是模拟和数字式仪表,而且功能单一。随着电子技术的发展和 PC 技术的广泛应用,虚拟仪器应用于测试与分析领域已经成为发展趋势。传统的仪器系统功能固化,不便于实现数据分析、处理、存储及显示,而这些功能则可以通过高性能的计算机编程易于实现。虚拟仪器系统提出的"软件就是仪器"的思想就是计算机处理各类数据的灵活性的体现。

VC + +是基于 MFC 的 Win32 程序,其特点是代码效率高、执行速度快,可以开发出风格多样的操作界面,缺点是开发信号处理算法时代码编写工作量很大。美国 NI 公司开发的 Measurement Studio 是一个测量程序包,其中的 LabWindows/ CVI 提供了丰富的库函数用于数据获取、数据处理和显示等功能[2]。二者混合编程的信号处理系统,能使开发 Windows 应用程序变得更加容易,又可以方便使用 LabWindows/CVI 提供的控件和信号分析模块。因此该混合编程方法的技术优势使其应用于科研、开发、测量、检测、测控等诸多领域,并随着"硬件软件化"的发展,必然会对工程技术产生一定的影响。

1　系统设计原理

1.1　硬件系统设计

信号处理系统硬件总体结构如图 1 示:

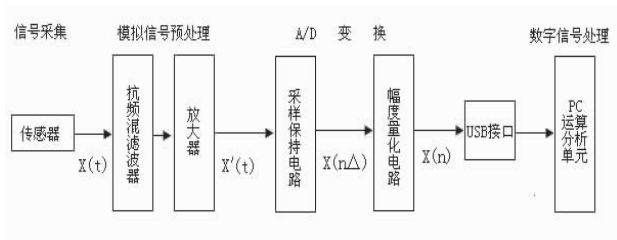

图 1　系统硬件结构

① 基金项目:高速公路气象环境自动监测系统(江苏高校高新技术产业发展项目)(编号:JH02 - 121)

传感器采集到的信号 X(t)经导线传送到抗频混滤波器进行滤波,使信号带宽限制在一定的范围内。因采集到的信号幅值较小,不能满足输入要求。因此需要对信号进行放大调整,使其幅度最大值接近最大输入电压(不能超过),以便充分利用量化位数,减小量化误差。经过放大后,信号输入 A/D 转换器,采样为离散信号序列 X(n△)并选用合适的时窗函数进行加窗处理。因 X(n△)是时域离散、幅值连续的信号,不能被计算机接受,应进行量化处理,使每一个采样信号的电压幅值变为数字码,从而把电压信号 X′(t)变为计算机能接受的数字系列 X(n)。运算分析单元接收 X(n),完成数字分析中,并显示分析结果。其中分析处理装置根据实际情况可以有两种选择,一种是单一的计算机,另一种是选用网络服务器结构,采集到的信号存储到子端计算机上,通过通信设备传输到终端服务器主机上进行处理[3]。

1.2 软件系统设计

软件系统采用 VC + +和 LabWindows/ CVI 混合编程实现。VC + +开发工具来设计用户操作界面,信号处理采用 LabWindows/ CVI 内置函数,处理结果显示采用 LabWindows/ CVI 提供的显示控件。根据功能要求和实际需要,系统软件主要有时域波形分析、FFT 变换、混频正交分解、滤波和信号内插处理等模块。

系统软件主要结构如图 2 所示

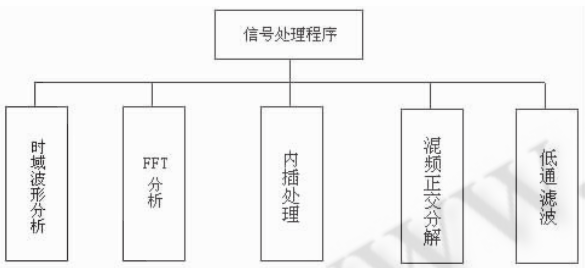

图 2　系统软件主要结构

在数字信号处理技术中,信号频谱分析起着十分重要的作用,是数字信号处理技术的重要处理环节。而 LabWindows/CVI 在信号处理和图形显示方面具有 VC + +无法比拟的优势。在 LabWindows/CVI 中实现频谱计算只需要一个 AutoPowerSpectrum 函数就可实现,如果采用 VC + +编写代码实现就非常困难。软件系统部分采用 VC + +设计操作界面并调用 LabWindows/CVI 的信号处理函数来实现对信号的处理,利用 LabWindows/CVI 的显示控件方便的显示处理结果。这样就大大减少了编程的工作量,提高了编程效率。

2 USB 串口通信部分

2.1 USB 接口芯片

本文所介绍的 USB 串口通信采用了 Cypress 公司 EZ – USB FX2 系列的 CY7C68013 – 128AC 芯片,它同时集成了 8051 微控制器和 USB2.0 收发器,在提高集成度的同时也加快了数据传输的速度。在系统中,CY7C68013 – 128AC 既是数据采集控制器又是 USB 控制器。EZ – USB FX2 系列有三种型号:CY7C68013 – 56PVC、CY7C68013 – 100AC、CY7C68013 – 128AC。这个系列的芯片都是针对 USB2.0 的,并且和 USB1.1 兼容。其中,CY7C68013 – 128AC 是 128 脚,TPQF 封装,它功能非常完善,与另外两种相比,主要是增加了 16 位地址总线和 8 位数据总线以及更多的 IO 口,因此,CY7C68013 – 128AC 的可扩展性最好。该芯片的内部结构可以参考文献[4]。

2.2 固件设备固件(Firmware)

固件设备固件(Firmware)是储存在程序内存中的代码,它使得 USB 接口芯片与主机和外设中其它电路能够通信。Cypress 公司给出了一个 Firmware 库和 Firmware 框架(Frame Works),均是用 Keil C51 开发的。Firmware 库提供了一些常量、数据结构、宏、函数来简化用户对芯片的使用;Firmware 框架实现了初始化芯片、处理 USB 标准设备请求以及挂起状态下的电源管理等功能。该框架不添加任何代码,编码后产生的.HEX 文件载入芯片就能和主机进行基本的 USB 通信,只是不能完成特定的任务。对于用户而言,主要的工作就是选择适当的传输方式,添加需要使用的端点(Endpoint),考虑到本系统要求实现一定数量数据的快速采集,并要迅速地将采集到的数据传输并进行分析和处理,并且对数据的完整性要求较高,我们采用了块传输方式(Bulk Transfers),在 TD – Init()函数中添加初始化代码,也即选择块传输方式和选择端点 2、6 分别为输出、输入端口,在 TD – Poll()函数中添加功能代码,以实现发送和接收数据功能,关键代码分别如下:

```
Void TD_Init (void) //初始化
{
CPUCS = ((CPUCS & ~bmCLKSPD) | bmCLK-
```

```
SPD1) ; //设置时钟 48MHz
    EP2CFG = 0xA0;//端点 2 为输出端口,类型为 BULK,端点缓冲大小为 512
    EP6CFG = 0xE0; //端点 6 为输入端口,类型为 BULK,端点缓冲大小为 512
    ......
    }
    Void TD_Poll(void)//实现发送和接收数据功能
    {
    if (enum_high_speed){
    ......
    Setup_FLOWSTATE_Write ();
    GPIFTRIG = GPIF_EP2; // 向端点 2 发送数据
    ......
    Setup_FLOWSTATE_Read ();
    GPIFTRIG = GPIFTRIGRD | GPIF_EP6; // 从端点 6 接收数据
    ......}
    }
```

2.3 USB 设备驱动程序

USB 设备驱动程序主要是使操作系统能够识别 USB 设备,建立起主机端和设备端之间的通讯,它们之间的通信是通过 Windows 提供的 API 函数实现的,这些函数可以控制显示器、处理信息、访问存储器、读写磁盘和其他设备。

USB 设备驱动的整体结构包括如下五个主要部分:USB 应用程序接口、USB 设备驱动函数、USB 中断服务程序、USB 回调接口程序、USB 标准事件处理程序。

有关 USB 设备驱动程序可借鉴参考文献[5]。

2.3.1 USB 应用程序接口

USB 应用程序接口主要功能是对 USB 驱动器进行软硬件初始化、打开端口、关闭端口、读端口、写端口和端口控制操作。当设备驱动器装入系统设备表时,I/O 系统就调用该应用程序接口。

USB 应用程序接口的一个例程主要包含:

(1)对 USB 端口安装、初始化和硬件配置(USB_init( ))。初始化步骤为:将 USB 设备驱动器安装到 I/O 系统设备表中,获取 USB 控制器使用的中断号,初始化 USB 驱动器数据结构与 USB 端口状态寄存器,启动 USB 标准事件处理程序;

(2)打开 USB 端口(USB_open( ))。USB_open 函数允许应用程序打开一个 USB 端口和选择 DMA 数据传输方式;

(3)关闭 USB 端口(USB_close( ))。USB_close 函数允许应用程序关闭一个端口,并关闭 DMA 通道;

(4)对 USB 端口进行读操作(USB_read( ))。USB_read 函数允许应用程序从输出端口或控制端口读取一定量的数据;

(5)对 USB 端口进行写操作(USB_write( ))。USB_write 函数和 USB_read 函数功能类似,允许应用程序写数据到输入端口或控制端口;

(6)对 USB 设备进行 I/O 控制操作(USB_ioctl( ))。

2.3.2 USB 中断服务程序

USB 控制器产生单一中断,多个端口共享。每个端口产生 ACK、NACK/ERROR 中断;输出端口产生接收零字节包或短包中断;控制端口 0 接收设置包时产生中断;USB 控制器产生 USB 事件中断,如帧起始(SOF)、挂起、恢复和复位。先识别发生 USB 中断的类型以清除中断产生的条件,再读 USB 状态寄存器,获取当前配置、接口或帧起始时间戳状态信息,最后向 USB 控制器消息队列或回调函数的接收消息队列发送中断消息。

2.3.3 USB 标准事件处理程序

USB 驱动器初始化后,启动 USB 标准事件处理程序负责处理枚举过程和异步 USB 事件。

事件处理程序使用控制端口 0,直到完成枚举过程。当 USB 应用程序处于非活动状态时,除控制端口 0 以外端口均不可访问。事件处理程序在端口 0 上执行控制操作,响应 USB 标准请求,并负责通知 USB 应用程序枚举完成和接口活动状态,USB 事件通过回调接口传递到 USB 外设应用程序。当对 USB 端口枚举操作完成,USB 应用程序就可打开并使用 USB 端口。

3 客户应用软件

对于广大用户而言,与系统的交互是通过应用程序实现的,其主要功能有:打开/关闭 USB 设备,检测 USB 设备,实现从 USB 设备接收指定数量的数据。关键程序代码如下:

3.1 查找、打开 USB 设备

```
Void CSINGALSDlg ::FindLoopDevice()
{
.....
USBDevice->Open(d);
.....
}
While((d<devices) && (USBDevice->VendorID!=0x04b4) &&
(USBDevice->ProductID != 0x8613)); //通过 Vendor ID 和 Product ID 查找设备
.....}
```

3.2 线程(Thread)

对于工作线程来说,启动一个线程,首先需要编写一个希望与应用程序的其余部分并行运行的函数如 Fun(),接着定义一个指向 CWinThread 对象的指针变量 * pThread,调用 AfxBeginThread(Fun,param,priority)函数,返回值赋给 pThread 变量的同时一并启动该线程来执行上面的 Fun()函数,其中 Fun 是线程要运行的函数的名字,也既是上面所说的控制函数的名字,param 是准备传送给线程函数 Fun 的任意 32 位值,priority 则是定义该线程的优先级别,它是预定义的常数,读者可参考 MSDN。本程序设计中的关键代码如下:

```
CWinThread * XferThread;
.....
Void CSINGALSDlg::OnSend()
{
.....
} else
{
bkeeping = true;
XferThread = AfxBeginThread(Xferkeep,this);//进入线程
.....
}
UINT Xferkeep(LPVOID params) //控制函数 Fun()
{
CSINGALSDlg * dlg = (CSINGALSDlg * ) params;
.....
Success = dlg->OutEndpt->XferData(data,Len);
.....
}
```

4 实例

在安装 VC++和 LabWindows/CVI 后,混合编程的实现是先在 VC++开发环境中建立对话框工程文件,再进行开发环境设置由 Project->Add To Project->Components and controls->Registered ActiveX Controls 选择 CWGraph Control(National Instruments)控件用于显示。在对话框工程文件的头文件当中包括 NiGraph.h 和 NiNumEdit.h 头文件,就可以在 VC++环境中编写代码过程中调用 LabWindows/CVI 的信号处理函数并用 CWGraph 控件显示处理结果。

4.1 利用 CWGraph 控件显示信号时域波形

其读取数据和显示信号关键程序如下,信号显示如图 3 所示.

```
CfileDialog dlg(TRUE,"dat","",OFN_OVERWRITEPROMPT,"*.dat||",this); //TRUE 为 OPEN 对话框
.....
FilePathName=dlg.GetPathName(); //获取文件路径名
.....
DWORD dwRead = FileData.ReadHuge(DataBuffer,sizeof(DataBuffer));//读取文件数据并存放到缓冲器赋予 dwRead。
for(int i=0;i<54096;i++)
{
dwRead0[i]=DataBuffer[i];
}//取 54096 个数据点
m_waveformGraph0.PlotY(dwRead0); // 由 CWGraph 控件显示信号。
```

4.2 利用 CWGraph 控件显示信号频谱图

信号频谱的获取可以用 AutoPowerSpectrum()函数对其时域数据进行处理获得频域数据,关键程序如下。

处理后的信号频谱图如图 4 所示。

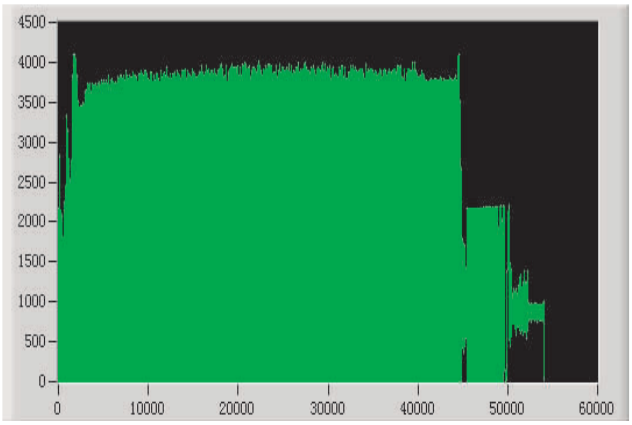

图 3　CWGraph 控件显示信号时域波形

CNiMath::AutoPowerSpectrum(dwRead0, spectrum0, df, dt); // **计算频谱**

m_spectrumGraph0.PlotY(spectrum0, 0, df);// **显示信号频谱**

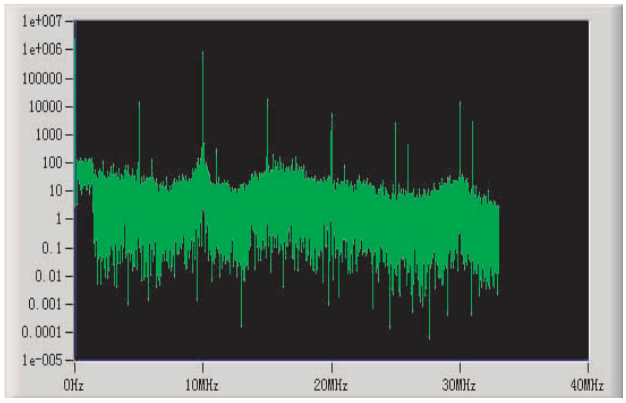

图 4　CWGraph 控件显示信号频谱另外

对信号的其他处理和显示方法类同，调用 LabWindows/CVI **的信号处理函数处理和** CWGraph **控件显示处理结果，这里不一一列举了。**

5　结论

系统采用目前普遍采用的信号处理系统的开发方式，即使用 USB2.0 **作为通信接口将硬件和各种计算机相连，加上相应的软件，实现数字化信号的采集、控制、测试、分析，研究开发了具有** Windows **风格的信号处理系统。该系统具有良好的用户界面，并且在实际应用中验证了系统的可行性。采用** VC++ **开发工具和** LabWindows/CVI **相结合的开发方法，具有较强的灵活性和可扩充性，在实际应用中可以得到不断的改善和发展。**

参考文献

1　杨智君，马晓庆，吴昭春. Windows 操作系统中的 USB 2.0 体系结构分析. 电子测试，2008. 20 – 23.
2　朱中锐，蔡志明，郭岩. 基于 LabWindows/CVI 数据采集系统. 电子测量技术，2007. 105 – 106.
3　王红伟，舒大文，朱琦琦. 基于 Visual C++ 6.0 的信号处理系统研究. 微计算机应用，2005. 372 – 374.
4　王奕，王凯. 基于 USB2.0 的数据采集系统的设计与实现. 电子工程师，2002. 15 – 17.
5　刘改梅，韩慧莲. 基于 LABVIEW 的 USB 无线通信接口的设计. 计算机技术与应用，2007. 121 – 123.

(上接第 82 页)

以平台独立模型 PIM **和平台相关模型** PSM **的变换为驱动，实现从** PIM **到** J2EE **平台上的** PSM **自动转换。对软件开发而言，**MDA **的软件开发进行对象的建模，并能生成大部分的代码，为应用程序的互操作性和可移植性提供了全面与结构化的解决方案。**

参考文献

1　董建武. MDA：新一代软件互操作体系结构. 计算机工程，2003，(2).
2　林炜，夏宽理. 基于 MDA 的模型转换方法研究. 计算机工程与应用，2005，(2).
3　Kleppe A，Warmer J，Bast W. 鲍志云译. 解析 MDA. 北京：人民邮电出版社，2004.
4　徐晓钟，谢康林. 基于 MDA 方法学软件开发方式的原理与实现. 微机发展，2004，(4).
5　张小华，韩永生，余军合. 模型驱动体系综述. 计算机工程，2004，(2).
6　黎才茂，关丽霞. MDA：模式驱动体系的软件建模. 现代计算机，2006，(7).