

文章编号:1673-064X(2008)04-0093-04

基于 TMS320VC5509 的数据处理及其 USB 传输系统

A novel system for data processing and transmission by USB based on TMS320VC5509

卢虎¹,高理²,廉保旺¹

(1.西北工业大学 电子信息学院,陕西 西安 710072;2.西安石油大学 电子工程学院,陕西 西安 710065)

摘要:基于 DSP 芯片和 USB 总线技术,设计并实现一个高速数据采集、处理和 USB 传输的一体化系统.系统围绕 TMS320VC5509 芯片进行设计,克服了常见的“主处理器+USB 控制器”系统分别处理和传输数据而造成的系统复杂、数据传输接口不稳定等缺点,占用系统资源少,既满足实时性,又满足易扩展性,具有一定的实用价值.

关键词:数据处理;通用串行总线;自举引导;WDM 驱动程序

中图分类号:TP391 **文献标识码:**A

随着 DSP 和 USB 技术的迅速发展,越来越多的电子工程技术人员开始使用 DSP 和 USB 总线进行数据高速处理和传输系统的设计,即以 DSP 为核心处理芯片进行数据的分析和处理再交给 USB 器件(如 EZ-USB 系列等)进行数据传输,通过 DSP 强大的控制和运算功能,可以很容易地实现 USB 总线数据的智能化传输.

该常用方法虽然实现了数据高速、有效的处理及其 USB 可靠传输,但由于 DSP 器件工作频率日益提高(当前主流 DSP 芯片工作频率大多稳定在 100 M 以上)与低速的 USB 外设器件(如 EZ-USB 系列为 24 MHz)之间往往无法满足时序的匹配,在系统设计时需要额外设计延时单元以进行协调,当传输数据量过大时,该常用方法往往无法充分显现以 DSP 为核心的高速处理系统的优势,而且额外单元的加入也会造成潜在的系统工作不稳定性.

本文以 TMS320VC5509 为核心器件,充分利用该款芯片内置的 USB 外部接口资源,设计并实现了一个高速数据采集、处理和 USB 传输的一体化系

统,给出系统的软硬件架构及其具体实现方法,最后介绍了 TMS320VC5509 芯片特有的 USB Boot-Loader 的基本概念.

1 TMS320VC5509 简介

TMS320VC5509 秉承 TI 公司 DSP 芯片的一贯特征,采用专用硬件逻辑 CPU、片内存储器、片内外设及高度专业化的精简指令集计算(RISC),其结构为改进哈佛结构,包括 1 条程序总线、3 条数据读总线、2 条数据写总线和附加的片上外设以及 DMA 专用总线,保证了指令和数据能以并行高速处理,并有新的特点^[1-3]:

(1)工作在 1.6 V 时,高达 144 MHz 的主频和仅占用 6.94 ns 的指令周期;

(2)支持块、中断和同步传输 3 种方式的高速(12 Mbps)USB 从端口.

TMS320VC5509 的数据处理模块,由于应用场合不同,具体处理方式有所不同,但其编程思想与风

收稿日期:2007-11-07

基金项目:国家自然科学基金(编号:60273009);教育部博士点基金(编号:20050699307)

作者简介:卢虎(1975-),男,在站博士后,讲师,主要从事通信与信息系统等方面的研究.

格与其他型号的 DSP 芯片基本相同,可采用汇编或 C/C++ 语言具体实现,不再赘述.下面主要介绍数据采集、处理完毕后,如何通过 TMS320VC5509 的 USB 接口进行数据传输的方法和具体实现过程.

2 USB 协议简介

2.1 USB 协议基本特征

USB(通用串行总线)通信协议用于将适用 USB 的外围设备连接到主机的外部总线结构,是一种支持主系统和 USB 外围设备(最多为 127 个)之间进行数据传送的高速串行总线^[4].目前主要有 2 种 USB 协议:USB1.1 和 USB2.0,连接至计算机的外设共同分享 USB 带宽,其总线带宽为 12 Mbit/s(USB1.1)或 480 Mbit/s(USB2.0).

USB 协议定义了四种数据传送类型:

(1)控制传送:一般用来发送与设备的能力和配置有关的请求和数据;

(2)同步传送:在主机与设备之间的周期性的、连续的通信,一般用于传送与时间相关的信息;

(3)中断传送:小规模数据量的、低速的、固定延迟的传送,可用于高速和低速设备;

(4)批量传送(数据块传送):非周期性的、大数据量的、可靠的传送,典型用于传送可以利用任何带宽的数据,且没有可用带宽时,数据可以容忍等待.

2.2 USB 协议配置

在应用程序与 USB 外设通信之前,主机需要知道设备支持那些传输类型和终端,同时需要给设备分配一个地址,主机通过枚举和交换描述符信息来完成此项工作.在枚举过程中,描述符逐步涉及设备各个具体细节:首先是整个设备(设备描述符),然后是设备的具体配置(配置描述符),接着是配置的接口(接口描述符),最后涉及接口的终端(终端描述符).其中,设备描述符包含关于这个设备的基本信息,是在设备连接时主机读取的第一个描述符,这个描述符包括了主机为了从设备得到其他信息所需要的信息;配置描述符,包含关于设备的电能使用和支持的接口的信息,每个设备至少有一个描述设备的特征和能力的配置,在接受了设备描述符后,主机就能够得到设备的配置、接口和终端描述符了;接口描述符,接口这个词当然可以从整体描述 USB,但从设备和它的描述符方面来说,接口表示一组被一个设备的特征或功能而使用的终端,一个配置的接口

描述符包含了关于这个终端的信息,终端描述符,除了终端 0,每个在一个接口描述符中指定的终端都有一个终端描述符,它指明终端号、方向、指定终端支持的传输类型及可传送的最大数据字节等信息.

可以看到要想利用 TMS320VC5509 的 USB 接口进行数据传输,首先就需要在 TMS320VC5509 中实现 USB 协议,下面具体论述之.

3 USB 协议的 TMS320VC5509 配置

由于 TMS320VC5509 对 USB 协议的支持仅限于 1.1 版,因此该文的实现也针对 USB 协议 1.1 版而言.

图 1 是 TMS320VC5509 作为 USB 设备与 PC 机进行数据通信的示意图.在进行输入(相对主机而言)操作时,SIE(serial interface engine)把来自 UBM(USB bufer manager)的并行数据转换为串行数据输入到主机;输出操作时,SIE 把主机传来的串行数据转换为 UBM 所需的并行数据.UBM 负责从 SIE 转移数据到 bufer RAM 或者由 bufer RAM 转移数据到 SIE.具体实现过程中,需要注意的是 CPU 或者 USB DMA 控制器必须在 UBM 传递数据到 SIE 前,把数据放入缓存 RAM;而当 CPU 或者 DMA 控制器准备好向存储器传递的数据后,则需要等待 UBM 把数据由 SIE 转移到缓存 RAM,方可进行后续操作^[5-7].

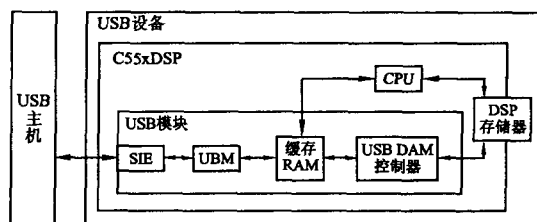


图 1 TMS320VC5509 作为 USB 设备与 PC 机数据通信示意图

若要利用 TMS320VC5509 进行 USB 数据传输,显然必须使 TMS320VC5509 实现并支持 USB 协议.因此,现在的问题就转化为:在以 TMS320VC5509 为核心处理器的数据采集、处理和传输系统中,实现 USB 协议;当该系统和 PC 机进行数据交互时,需开发支持 TMS320VC5509 设备的硬件驱动程序.因此,系统的主要工作归结为开发 PC 机的 USB 驱动程序(即 TMS320VC5509 设备的 PC 机的 USB 驱动程序)和使 TMS320VC5509 对驱动程序进行支持.

由 USB 协议可知, TMS320VC5509 的主要工作是: 当设备接入主机后, 驱动程序枚举设备; TMS320VC5509 必须提供主机枚举所需要的各种描述符信息并根据驱动程序的要求提供输入或输出数据以及数据的处理操作。

描述符信息须严格参照 USB 协议进行配置, 以下是该系统中 TMS320VC5509 的设备描述符, 可供参考。

```
static Uint16 deviceDescriptor[] = {
0x0000,
(C5509_USB_DESCRIPTOR_DEVICE << 8) | 18
/* 描述符表字符数 */
0x0101, /* 协议版本号 */
0x0000, /* 设备类 */
0x4000, /* 子类 */
0x0451, /* 供货商 ID */
0x9002, /* 产品 ID */
0x0000, /* 设备发行号 */
0x0201, /* 厂商信息字符串索引 */
0x0100 /* 设备信息字符串索引 */
};
```

类似地, 根据系统需求可以很容易得到其他类型的描述符信息。

数据处理操作模块的实现, 可利用 TMS320VC5509 提供的 CSL 函数库 (Chip Support Library) 中的各种 API 函数进行数据处理操作, 例如, 下面是 USB 中断处理例程, 可供参考。

```
if(IFR0 & IFR0_USBMSK)
{
IFR01 = IFR0_USBMSK; /* 清除 USB 终端标志 */
USB_evDispatch(); /* 处理所有 USB 事件 */
}
```

其中, USB_evDispatch() 函数即为标准 CSL 处理函数, 在进行头文件引用 #include <cs1.h> 后, 即可直接使用, 大大简化开发难度, 此外, 还可以使用 CSL 图形界面 (GUI) 进行 USB 模块的配置工作, 进一步缩短开发周期。

需要特别提出的是, TMS320VC5509 除了支持标准的 bootloader 功能外, 还增加了 USB-bootloader 的功能, 即通过主机和 TMS320VC5509 的 USB 接口, 进行程序的加载, 以往需要存储在下位机系统

FLASH 上面的程序, 现在可以存放在主机上面, 当系统开始工作时, 首先由主机向 TMS320VC5509 下载程序 (该功能可以集成到系统的主机程序之中), 而后系统开始协同工作。该项功能极大节约了以 TMS320VC5509 为核心的数据处理系统的资源, 在系统需要处理的数据复杂因而程序显得有些庞大时尤为可贵, 这样开发人员几乎不必再为所谓的节约资源而徒劳地精简代码。

4 TMS320VC5509 的驱动程序

当下位机 (以 TMS320VVC5509 为核心的数据处理传输系统) 开发完成后, 就需要开发该系统 PC 机下的驱动程序以及客户应用程序, 以使系统可以最终与上位机 (PC 机) 正常通信, 并使上位机可以对下位机进行数据收发的操作。根据上位机操作系统 (以 Microsoft 公司的 Windows 系列操作系统为例) 的不同可以选用 Windows 2000 DDK 或者 Windows XP DDK 进行驱动程序的开发。

Windows 驱动程序采用分层结构, 最高层支持用户模式和内核模式两种驱动程序, 用户模式驱动程序是运行在用操作系统用户模式的系统级代码; 而内核模式驱动程序使用系统级代码编写, 允许直接访问硬件。内核模式驱动程序可被进一步分成遗留模式驱动程序和 Windows 驱动模式驱动程序 (WDM)。遗留模式驱动程序需要去侦测硬件和与 I/O 子系统连接; WDM 驱动程序支持即插即用、电源管理、热拔插 (PnP)、自动配置。再下一层, 遗留模式和 WDM 驱动程序可被进一步分成: 高层、中间层、低层。高层驱动程序依靠中间层和低层驱动程序去完成工作, 中间层驱动程序则依靠低层驱动程序去完成工作^[8]。

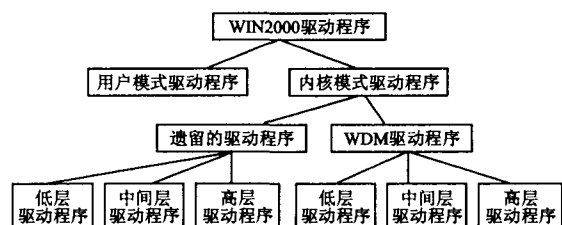


图 2 windows 系统驱动程序的分层结构

TMS320VC5509 的 USB 驱动程序属于 WDM 型驱动程序, 因此不必直接与 TMS320VC5509 的 USB 接口打交道, 只需通过下层驱动程序提供的接口来访问该接口即可。换言之, 编写 TMS320VC5509 的 USB 设备驱动程序不必具体针

对硬件编程,所有的 USB 命令、读写操作均是通过总线驱动程序转发给 USB 设备,因此驱动程序必须定义与外部设备的通讯接口和通讯的数据格式^[9].

一般来说,USB 驱动程序,主要包括如下几个部分,读者可以根据需要进行取舍.

(1) DriveEntry() 例程:驱动程序的人口函数,所有对各种 IRP 的处理都在此例程中做出说明(大致相当于普通应用程序的 main() 或 winmain() 函数,不能省略).

(2) AddDevice() 例程:当 USB 设备接入主机系统时,I/O 管理器调用此例程,此例程负责创建 USB 功能设备对象,同时创建 Win32 子系统可见的设备名.

(3) 响应 IRP_MJ_CREATE 和 IRP_MJ_CLOSE 请求的例程:当用户态的应用程序通过 Win32API 函数 CreatFile() 或 CloseFile() 请求创建或关闭 USB 设备句柄时,I/O 管理器将调用此例程.

(4) 响应 IRP_MJ_DEVICE_CONTROL 请求的例程:当用户态的应用程序通过 Win32API 函数 DeviceControl() 请求对 USB 设备进行 I/O 操作时,I/O 管理器将调用此例程.

(5) 响应 IRP_MJ_PNP 请求的例程:在此例程中驱动程序处理由 PnP 管理器发送的 PnP 消息.

下面的 DriverEntry() 例程,可供参考:

```
NTSTATUS DriverEntry (IN PDRIVER_OBJECT
DriverObject, IN PUNICODE_STRING RegPath)
{
NTSTATUS status = STATUS_SUCCESS;
DriverObject->DriverUnload = USBUnload;
DriverObject->MajorFunction [ IRP_MJ_CREATE ] = USBCreate;
DriverObject->MajorFunction [ IRP_MJ_CLOSE ] = USBClose;
DriverObject->MajorFunction [ IRP_MJ_READ ] = USBRead;
DriverObject->MajorFunction [ IRP_MJ_WRITE ] = USBWrite;
DriverObject->MajorFunction [ IRP_MJ_DEVICE_CONTROL ] = USBDeviceControl;
RegisterForPnpNotification(DriverObject);
```

```
return status;
}
```

在 Windows XP 系统中,驱动程序开发完成并正确安装后如图 3 所示.

USB 的客户程序实际上是对客户端设备映像的操作,这些映像由 USB 驱动程序产生,属于用户模式.在 DDK 中存在着一组叫做 USBDI 函数的 API 函数集,该函数集包括了传输函数、管道(Pipe)函数、设备配置函数及其他函数.应用这些函数可编写支持任何 USB 兼容设备的 USB 驱动程序和客户程序.

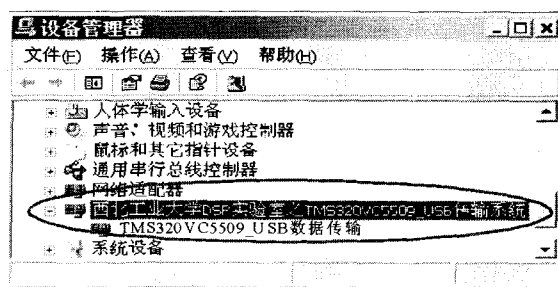


图 3 驱动程序开发完成并正确安装后的示意图

为了优化应用程序性能,可将应用程序分为 2 个部分:动态链接库和应用程序.动态链接库接受应用程序的各种操作请求并负责和内核态的 USB 客户驱动程序通信,而应用程序负责将数据进行传输,还可进一步采用 Windows 多线程技术使数据传输和程序界面分别处于独立线程,来提升程序性能.

5 结束语

本文提出的高速数据处理和数据 USB 传输的一体化方案,主要是利用了 TMS320VC5509 的优良性能,尤其是 USB 的 BootLoader 功能的合理应用,使得系统结构简洁、性能可靠、易于开发和扩展,并已在实践中获得检验.

参考文献:

- [1] TI. TMS320VC5509 Fixed-Point Digital Signal Processor Data Manual[M]. U S: Texas Instruments, 2003.
- [2] TI. TMS320C55x CSL USB Programmer's Reference Guide[M]. U S: Texas Instruments, 2001.
- [3] TI. A DSP/BIOS USB Device Driver for the TMS320C5509 [M]. U S: Texas Instruments, 2003.
- [4] 宋新爱. 基于 WDM 的设备驱动程序开发[J]. 西安石油大学学报:自然科学版, 2007, 22(4): 113-115.
- [5] TI. TMS320C55x DSP Peripherals Reference Guide[M]. U S: Texas Instruments, 2003. (下转第 105 页)

(2)为了比较标准遗传算法、粗糙集和基于粗糙集的增量式遗传算法之间获取规则的性能,分别对以上①,②,③数据库按照不同的方法抽取规则,表1给出了比较的结果。

表1 规则获取方法比较

参数	Glass			Iris			Zoo		
	SGA	RST	IGA	SGA	RST	IGA	SGA	RST	IGA
规则总数	36	37	30	6	4	4	12	10	9
覆盖度/%	93	100	97	96	100	98	95	100	99
时间/s	562	481	380	505	462	430	434	398	380

SGA:标准遗传算法;RST:粗糙集理论;IGA:基于粗糙集的增量式遗传算法

通过对表1中的3种规则获取方法所得到的结果进行比较,可以看出,在保持较高的覆盖度的同时,基于粗糙集的增量式遗传算法所获取的规则数量最少,获取规则的时间也是最短的.这就进一步说明了本文所提出的基于增量式遗传算法的规则挖掘的正确性和有效性。

5 结 论

本文基于粗糙集理论,分析了遗传算法的基本原理,结合数据挖掘中规则挖掘问题的具体情况,将分规则获取与优化方法结合起来.把遗传算法作为

一种优化技术作用于规则集上.在此基础上,针对数据挖掘对算法可扩性的要求,将增量算法与数据库更新结合起来,实现知识的渐进更新.试验结果表明,本文所提出的方法能有效地获取最优规则集。

参 考 文 献:

- [1] Pawalk Z. Rough sets[J]. International Journal of Computer and Information Science, 1982, 11(5):341-356.
- [2] Pawalk Z. Theoretical Aspects of Reasoning about Data [M]. Boston: Kluwer Academic Publishers, 1991.
- [3] Yao Yiyu, Zhao Yan, Wang Jueet, et al. A Model of Machine Learning Based on User Preference of Attributes [C]//Greco S, Hata Y, Hirano S, et al. Rough Sets and Current Trends in Computing (RSCTC 2006). Berlin: Springer-Verlag, 2006: 587-576.
- [4] Yin Ying, Zhang Bin, Zhao Yuhai. Mining the Most Interesting Patterns from Multiple Phenotypes Medical Data [C]//Greco S, Hata Y, Hirano S, et al. Rough Sets and Current Trends in Computing (RSCTC 2006). Berlin: Springer-Verlag, 2006: 696-705.
- [5] Pawlak Z. Decision Trees and Flow Graphs [C]//Greco S, Hata Y, Hirano S, et al. Rough Sets and Current Trends in Computing (RSCTC 2006). Berlin: Springer-Verlag, 2006: 1-11.
- [6] URL: <http://www.ics.uci.edu/pub/machine-learning-databases/>.

编辑:张新宝

(上接第96页)

- [6] TI. TMS320C55x DSP CPU Programmer's Reference [M]. U S: Texas Instruments, 2003.
- [7] TI. TMS320C55x DSP CPU Reference Guide [M]. U S: Texas Instruments, 2003.
- [8] 张惠娟. Windows 环境下的设备驱动程序设计 [M]. 西

安:西安电子科技大学出版社, 2002.

- [9] 武安河. Windows 2000/XP WDM 设备驱动程序开发 [M]. 2版. 北京:电子工业出版社, 2005.

编辑:张新宝