高速数据处理系统及数据USB传输的新方案及其实现

卢 虎 李 勇
(西北工业大学电子信息学院,西安 710072)
E-mail:Sdkmsdn@163.com

摘 要 基于DSP芯片和USB总线技术,设计出高速数据处理和传输的一体化系统。该方案由于采用TMS320VC5509芯片,克服了以前因采用"主处理器+USB控制器"方式,分别处理和传输数据而造成的系统复杂、传输速度延时等缺点,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 TMS320VC5509 USB-BootLoader WDM

文章编号 1002-8331-(2005)05-0120-03 **文献标识码** A **中图分类号** TP391

A Novel System about High-speed Data Processing and Transmission by USB

Lu Hu Li Yong
(Department of Electronic Engineering, Northwestern Polytechnical University, Xi'an 710072)

Abstract: Based on DSP chip and USB bus technique, this paper designs a high-speed data processing and transmission integrated system. This system is conciser and higher-speed than old methods, which are designed as "CPU+USB controller" model. This system is real-time, expandable and valuable.

Keywords: TMS320VC5509, USB-BootLoader, WDM

当今的计算机外部设备都在追求高速度和高通用性,为了满足用户的需求,以Intel为首的七家公司于1994年推出了USB(Universal Serial Bus)通用串行总线协议,专用于低中速的计算机外设。目前USB端口已成为了微机主板的标准端口,与传统的并口或串口通信相较而言,USB的主要优点是速度高、功耗低、支持即插即用(Plug & Play,PnP)和使用维护方便。

随着DSP和USB技术的发展,越来越多的电子工程技术人员开始使用DSP和US进行数据采集处理和传输系统的设计,即以DSP为主处理器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设备的智能化。该方案虽然实现了数据的有效处理和USB传输,但是由于高速的DSP器件(大多在40M以上)与低速的USB器件(如EZ-USB系列为24MHz)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调和,当传输数据量过大时,该方案往往无法充分显现高速便捷的USB数据传输特点。

文章以TMS320VC5509为主要部件,充分利用TMS320VC5509内置的USB接口,构成高速数字采集处理和USB传输系统,描述了USB传输系统的实现方法以及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,包括一条程序总线、三条数据读总线两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得以并行高速处理,并有新的特点:

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

(2)支持块、中断和同步传输三种方式的高速(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,但其编程思想与风格与其它型号的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协议基本特性

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。目前主要有两种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12Mbps(USB1.1)或480Mbps(USB2.0)。

USB协议定义了四种传送类型:

(1)控制传送:一般用来发送与设备的能力和配置有关的

作者简介:卢虎(1975-),男,博士,主要研究方向:实时信号处理,计算机网络通信。李勇(1964-),男,教授,主要研究方向:通信信号处理,嵌入式系统应用。

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数等信息。

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 实现方法

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。

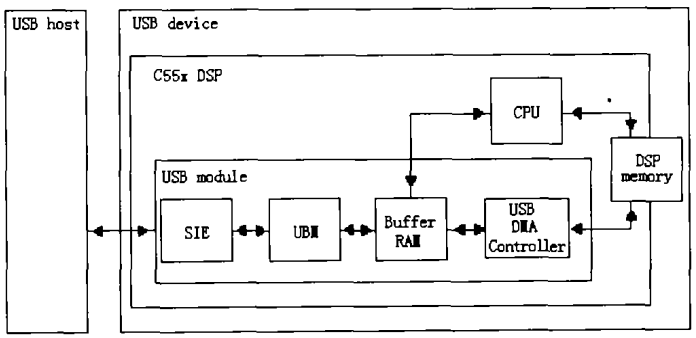

图 1 TMS320VC5509 作为 USB 设备与 PC 机进行数据通信的示意图

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 buffer 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 buffer RAM 或者由 buffer 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 buffer 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 buffer RAM，方可进行后续操作。

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开发为 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,/* 描述符表字节数 */
  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它类型的描述符信息。

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)
  {
    IFR0|=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<csl.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，TMS320C5509 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 TMS320C5509 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 TMS320C5509 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 TMS320C5509 为核心的数据处理系统的资源，在系统需要处理的数据及其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的上位机系统

当下位机(以TMS320VC5509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操作系统为例)的不同可以选用Windows98 DDK或者Winndows2000 DDK进行WDM(Windows Driver Model)型驱动程序的开发。

WDM型驱动程序是内核态程序,与标准的用户态Win32程序不同。采用了分层处理的方法。通过它,用户不需要直接与硬件打交道,只需通过下层驱动程序提供的接口来访问硬件。因此,USB设备驱动程序不必具体对硬件编程,所有的USB命令、读写操作通过总线驱动程序转发给USB设备。但是,USB设备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。

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:驱动程序的入口函数,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_CREAT和IRP_MJ_CLOSEIRP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_PNPIRP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 RegistryPath)
{
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Windows98系统中,驱动程序开发完成并正确安装后的示意图如图2。

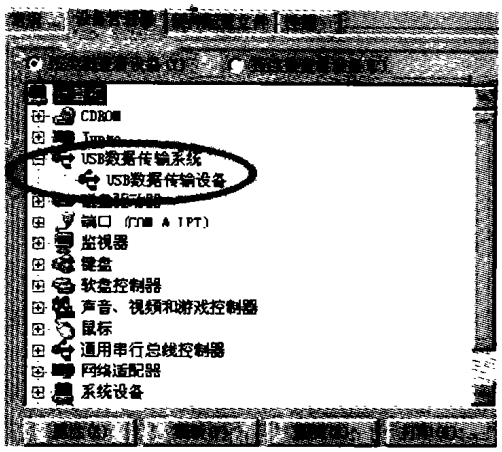

图2 驱动程序开发完成并正确安装后的示意图

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。

为了优化应用程序性能,可将应用程序分为两个部分: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 小结

文章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-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,希望对读者有所帮助。

(收稿日期:2004年8月)

参考文献

1.Universal Serial Bus Specification Revision 1.1[S].Compaq Intel Microsoft NEC,1998-11-23

2.TMS320VC5509 Fixed-Point Digital Signal Processor Data Manual[M].TI,2003-05

3.TMS320C55x CSL USB Programmer's Reference Guide[M].TI,2001-11

4.A DSP/BIOS USB Device Driver for the TMS320C5509[M].TI,2003-06

5.TMS320C55x DSP Peripherals Reference Guide[M].TI,2003-02

6.TMS320C55x DSP CPU Programmer's Reference[M].TI,2003-03

7.TMS320C55x DSP CPU Reference Guide[M].TI,2001-05