The Design and Implementation of Virtual Oscillograph Based on STM32*

DING Hongbin, QIN Huibin, SUN Shunyuan*
(*Institute of Electron Device & Application, Hangzhou Dianzi University, Hangzhou* 310018, *China*)

Abstract: Using the STM32 MCU based on ARM Cortex-M3 as the core, the design and the detailed implementation methods of a virtual oscillograph was presented through the USB interface, and the external signal conditioning, collection, pretreatment and data transmission were achieved. Meanwhile, the USB drive and host computer applications were developed to realize a virtual oscilloscope data transmission, display and other basic functions.

Key words: STM32F103x; USB; virtual oscillograph; driverstudio; DDK
EEACC: 7250G

基于 STM32 的虚拟示波器的设计与实现*

丁红斌, 秦会斌*, 孙顺远
(杭州电子科技大学新型电子器件与应用研究所, 杭州 310018)

摘 要: 基于 ARM Cortex-M3 内核的 STM32 微控制器为核心, 介绍了通过 USB 接口进行数据传输的虚拟示波器的设计方案和具体实现方法。完成了对外部信号的调理、采集、预处理和数据传输, 同时, 对 USB 驱动及上位机应用程序进行了开发, 实现了虚拟示波器数据传输、显示等基本功能。

关键词: 虚拟示波器; STM32F103x; USB; DriverStudio; DDK
中图分类号: **TN935.3** 文献标识码: 文章编号: **1005-9490(2009)06-1007-04**

示波器是电子测量行业最常用的测量仪器之一, 主要用来测量并显示被测信号的波形等参数, 在很多领域得到广泛的应用。虚拟示波器突破了传统示波器的性能局限, 在功能和应用性上发生了根本性变化。虚拟示波器不仅实现了传统示波器的功能, 而且利用功能强大的微型计算机来完成信号的分析、处理, 利用软件技术在屏幕上设计出逼真的仪器面板并显示各种特征图形。计算机功能最大化地服务于虚拟仪器, 使仪器功能得到充分发展和完善[1]。

系统采用意法半导体公司开发的基于 Cortex-M3 内核的新型 32 位微控制器 STM32F103x 作为主控芯片。该芯片内部集成了全速 USB2.0 设备接口模块和 16 通道的 12 位高精度 A/D 转换器, 单芯片即可完成设计任务, 避免了复杂的接口电路设计, 有效地降低了系统接口的复杂度和系统开发的难度[2], 在很大程度上提高了系统的稳定性。同时, 结合基于 VC++ 开发平台对信号进行存储和显示。该系统体积小巧、简单易用, 能够实现 1 MHz 的采样速度, 与 PC 机通信的最高速率达到 1 Mbyte/s。

1 系统硬件设计

根据系统的功能需求, 系统结构图如图 1 所示, 主要包括: 信号输入接口模块、信号调理模块、数据采集及缓存模块、USB2.0 通信模块等[3]。工作流程如下: 系统通过 USB 接口接到主机上后获得 5 V 电源, 微控制器 STM32 对硬件设备进行初始化, 并通过内部 USB 硬件控制器来完成 USB 设备的枚举工作。A/D 转换器是通过定时器来触发采样的, 以

收稿日期: 2009-08-15 修改日期: 2009-09-10
项目来源: 浙江省自然科学基金资助(Y407133)
作者简介: 丁红斌(1985-), 男, 云南曲靖人, 杭州电子科技大学在读研究生, 硕士, 电路与系统专业, 从事新型电子器件设计及应用, ding_hongbin@163.com;
秦会斌(1961-), 男, 教授, 博士生导师, 从事新型材料与器件、抗电磁干扰技术研究。

保证其以恒定的时间间隔对模拟信号进行模数转换,在完成规定长度的采集工作之后,将数据存入高速数据存储器中(RAM)。在规定的触发条件满足时,数据采集系统中的控制电路使能 DMA 通道,将 A/D 所采集的数据传输至 USB 的缓冲区中,由 USB 接口电路将这些数据传输给上位机。

信号输入及调理模块主要完成信号衰减,程控放大,叠加直流分量。衰减电路是为了保证在较大的信号输入时,能够在 A/D 采样的范围之内,避免回显时造成信号的失真或是损坏元器件。程控放大电路是将待测的小信号进行放大,由模拟开关 CD4051、运放 NE5532 和可变电阻器构成,并通过 MCU 切换放大倍数,电路如图 2 所示。数据采集及缓存模块和 USB2 0通信模块是由微控制器 STM32F103x 独自实现的。由于MCU 内部自带的 A/D 无法对负电压进行采集,而待测信号往往又带有负压,这时需要电路将负压抬高到 0 电平以上,如图 3 所示。

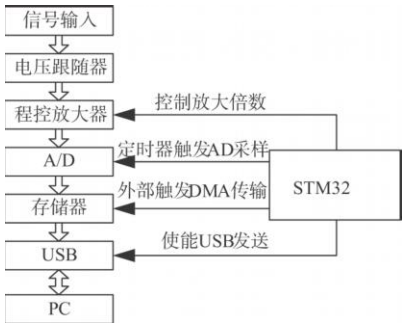

图 1 系统硬件原理框图

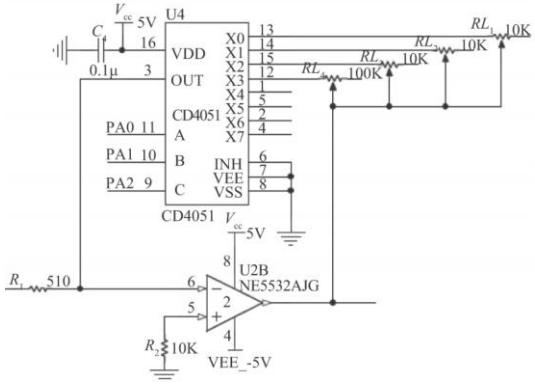

图 2 程控放大电路

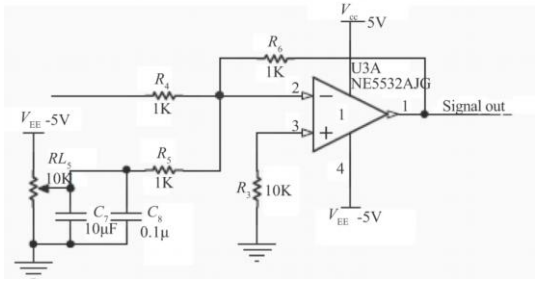

图 3 电压提升电路

为了使示波器工作在触发模式,使 A/D 采样的波形能够稳定的显示,系统中需要有个触发电平,这里的触发电平由迟滞比较电路产生,如图 4 所示。

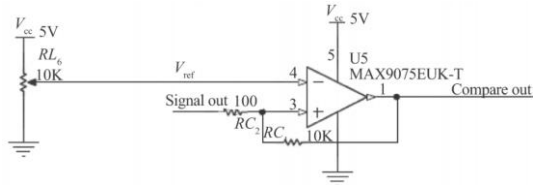

图 4 迟滞比较电路

2 系统软件设计

2 1 固件程序的开发

设备的固件程序设计主要包括:系统时钟的配置、ADC 模块配置、定时器模块配置和 USB 模块配置。总体流程如图 5 所示。

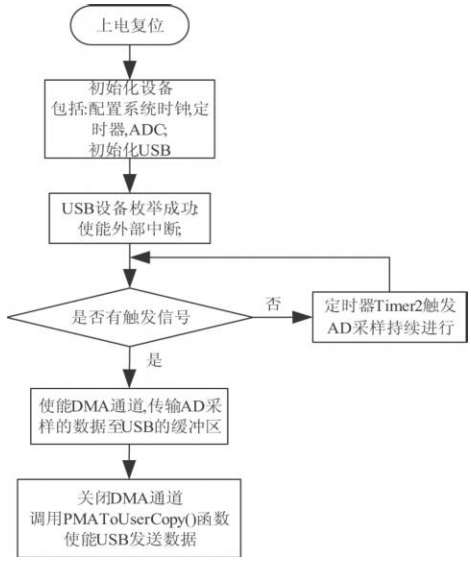

图 5 系统总体程序流程图

意法半导体公司针对 ARM 的 32 位 STM32F103x 系列 MCU 提供了固件库。该固件库提供了包括 ADC 在内的各种功能模块的软件使用接口,使用该固件库可以有效节省用户产品的开发和调试时间。利用该库,本设计中 ADC 的配置代码如下:

```
ADC_InitStructure. ADC_Mode = ADC_Mode_Independent;
ADC_InitStructure. ADC_ScanConvMode = DISABLE;
ADC_InitStructure. ADC_ContinuousConvMode = DISABLE;
ADC_InitStructure. ADC_ExternalTrigConv = ADC_ExternalTrigConv_T2_CC2;
ADC_InitStructure. ADC_DataAlign = ADC_DataAlign_
```

```
Right;
    ADC_InitStructure. ADC_NbrOfChannel= 1;
    ADC_Init(ADC1, & ADC_InitStructure);
```

根据 USB2 0 协议, 当 USB 插入 USB 总线时, USB 控制器会自动为该 USB 设备分配一个数字来标示这个设备。另外, 在设备的每个端点都有一个数字来表明这个端点。USB 设备驱动向 USB 控制器驱动请求的每次传输被称为一个事务(Transaction), 事务有四种类型, 分别是 Bulk Transaction、Control Transaction、Interrupt Transaction 和 Isochronous Transaction。每次事务都会分解成若干个数据包在 USB 总线上传输。每次传输必须历经两个或三个部分, 第一部分是 USB 控制器向 USB 设备发出命令, 第二部分是 USB 控制器和 USB 设备之间传递读写请求, 其方向主要看第一部分的命令是读还是写, 第二部分有时候可以没有[4]。

本设计中数据传输方式采用批量传输, 即 Bulk Transaction。这种事务传输的时候分为三部分。第一部分是令牌包阶段, 主机 Host 端发出一个 Bulk 的令牌请求。第二部分是数据包阶段, 根据先前请求的令牌的类型, 数据传输有可能是 IN 方向, 也有可能是 OUT 方向。传输数据的时候用 DATA0 和 DATA1 令牌携带着数据交替传送。第三部分是握手包阶段。如果数据是 IN 方向, 握手信号应该是 Host 端发出, 如果是 OUT 方向, 握手信号应该是 Device 端发出。握手信号可以为 ACK, 表示正常响应, 也可以是 NAK 表示没有正确传送。STALL 表示出现主机不可预知的错误。在本设计中, 端点 0 为控制传输端点, 实现 USB 设备上电后的配置过程; 端点 1 为 OUT 中断传输端点, 用于接收上位机发送的命令, 实现上位机对数据采集过程的控制; 端点 2 为 IN 块传输端点, 将数据采集结果实时地传送到上位机, 供上位机进行数据分析和处理。端点 2 的配置如下:

```
SetEPType ( ENDP2, EP _ BULK ); SetEPTxAddr
(ENDP2, ENDP2_TXADDR);
    SetEPTxCount( ENDP2, 64);
    SetEPTxStatus( ENDP2, EP_TX_VALID);
    SetEPRxStatus( ENDP2, EP_RX_DIS);
```

在设置端点 2 发送有效之后, USB 模块可自动完成端点 2 缓冲区的数据发送。

2 2　USB 设备驱动程序的开发

2 2 1　开发平台的搭建[5]

首先依次安装 Microsoft Visual C+ + 6 0、Microsoft Windows XP DDK、DriverStudio 3 2 驱动程序开发工具包。在安装过程中, VC6 0 最好安装英文版的, 可以减少不知名的错误。在 DriverStudio 安装过程中, 选择安装其中的 DriverWorks、Tools 和 SoftICE 工具。DriverStudio 安装之后, 在 Visual C+ + 6. 0 编程工具栏里自动添加了一个菜单 DriverStudio 。必须注意到, DriverStudio 选项卡下面第三项 DDK Build setting 要设置成 C:\WINDDK\2600(如果 DDK 安装在 C 盘), 然后要编译 DriverStudio 安装目录下 DriverStudio\DriverWorks\source\VdwLibs. dsw, 以得到 vdw_wdm. lib 这个库文件。编译的时候会出现错误, 因为用 VC 打开 vdwlibs. dsw 工程文件后, 有两个工程, 要先将 VdwLibs 工程设为当前 Active Project, 然后在工具栏上单击右键选择“Build”, 在弹出的编译工具栏中配置一下编译平台的设置: 选择 Win32 WDM Checked 平台, 然后编译就可以了。至此, 开发平台搭建完成。

2 2 2　创建 USB 设备驱动程序框架

利用 DriverStudio 的 DriverWorks 生成 USB 设备驱动程序框架的步骤: ①从 Visual C+ + 6 0 工具栏中选择“DriverStudio” →“DriverWizard”, 然后新建工程, 选择 C+ + 版本, 这里工程名命名为 STM32。②选择驱动程序类型及是否需要 C+ + 框架的支持, 在此选择 WDM 型, 需 C+ + 框架的支持。③选择 WDM 驱动程序类型, 这里选择 WDM 功能驱动程序。④选择总线类型, 填写 USB 设备芯片 VID(供应商 ID)和 PID(设备 ID), 我们选择 USB 总线, 在此 VID 和 PID 分别为 0483、7540。⑤定义 USB 接口芯片端点, 其中端点 0 默认为控制端点, 不需要定义, 端点 1 定义为 OUT 中断传输方式; 端点 2 为 IN 块传输端点。⑥选择设备操作, 如 Close、Create、Read、Write 等。⑦选择 I/O 端口读写方式, 添加 IOCTL 接口(用于控制传输)。⑧创建注册表项, 这里我们不需要创建任何注册表项。⑨对 WDM 支持的电源管理选项进行选择, 这里选择默认的“Manage Power For This Device”。⑩选择是否支持 WMI。接下来的 Installation、Additional 和 Summary3 个对话框按默认方式。这样, 就创建了一个 USB 设备驱动程序框架, 接下来, 在这个框架中添加所要实现功能的代码。

2 2 3　USB 设备驱动代码编写与实现

以上由 DriverWorks 自动生成的 USB 设备驱动程序框架主要由 STM32Driver. cpp 和 STM32Device. cpp 两个源程序组成。其中, 前者用于初始化驱动程序, 它包括: 入口函数 DriverEntry、AddDevice 和 Unload 函数; 后者用于在 WDM 环境

下支持即插即用设备,它包含了设备的插拔、电源管理、设备读写、设备控制等具有特定功能的例程。所要完成的就是对这两个源程序的代码添加,来实现USB 设备驱动程序的功能。在代码添加完,通过编译之后,选择"Build"→"Build STM32.sys",即可生成可执行的USB 设备驱动程序 STM32.sys 文件。另外还有一个 STM32.inf 文件用向导就可以生成,用这两个文件就能进行驱动的安装,驱动安装成功如图 6 所示。

图 6 驱动安装成功

2.3 PC 上的应用程序开发

上位机的人机界面是虚拟示波器与用户的接口,直接关系到系统的可用性和方便性。人机界面程序主要是使用户能够方便地控制整个虚拟示波器的工作。

通过 VC++ 软件编程实现信号的显示、拉伸、移动等功能。VC++ 与 Windows 操作系统密切结合,有一套功能强大的可视化类库[6](MFC),采用面向对象的编程方法。上位机主要分为以下的几个部分:(1) 信号数据读取程序,完成信号读取;(2) 信号显示程序,实时显示出信号波形;(3) 信号波形处理程序,主要是完成波形的拉伸和上下移动。为了实时的对显示出来的数据进行刷新,需要不停的读取、处理和显示数据,因此需建立一个线程,然后把读写,处理,显示程序放在线程函数里面,来完成信号的实时更新。

3 测试结果分析

系统上电工作后,由信号源发出信号输入该系统,用示波器测量信号源信号,观察上位机显示的波形,与输入的标准信号比较。测试时分别输入正弦波、三角波、方波等信号,结果系统显示的波形与信号源测得波形形状保持一致,示波功能正确实现。图 7 为示波器测得峰-峰值为 1 V,频率为 100 kHz 的正弦波,图 8 为系统所测得的波形。系统性能指标如下:(1) 测量频率范围:0-200 kHz;(2) 分辨率:显示屏刻度为 8 div × 10 div,垂直分辨率为 8 bit,水平显示分辨率为 ≥40 point/div;(3) 采样速率:1 MHz;(4) 与 PC 机通信速率:1 Mbyte/s;(5) 波形周期误差:≤5%;(6) 触发方式:外部电路触发。

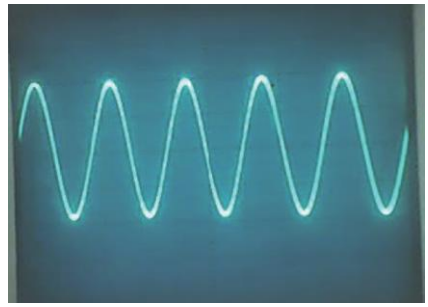

图 7 示波器采集的正弦波形

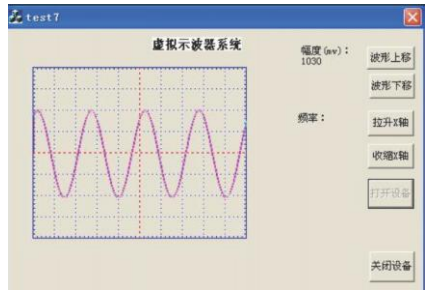

图 8 虚拟示波器采集的波形

4 结论

根据对该系统的整体测试可知,该系统体积小巧、简单易用、成本低,能够实现 1 MHz 的采样速度,带宽 200 kHz,与 PC 机通信的最高速率达到 1 Mbyte/s。同时,主控芯片 STM32F103x 内部集成了丰富的功能模块,使系统无需外扩大量芯片而能实现数据采集功能,降低了开发的复杂度和成本,达到了提高系统稳定性的目的。

参考文献:

[1] 路林吉,饶家明. 虚拟仪器概论[J]. 电子技术,2000(1):44-47.
[2] 王永虹,徐炜等. ARM Cortex-M3 微控制器原理与实践[M]. 北京:北京航空航天大学出版社,2008.
[3] 谭定忠,何干辉等. 基于 USB 总线的虚拟示波器[J]. 工业仪表与自动化装置,2008(5):85-87.
[4] 林海军,杨萍等. 基于 USB2.0 的虚拟数字示波器的设计[J]. 电测与仪表,2008(9):37-41.
[5] 薛园园. USB 应用开发技术大全[M]. 北京:人民邮电出版社,2007.
[6] 孙雄勇. VISUAL C++ 6.0 实用教程[M]. 北京:中国铁道出版社,2004.