基于 WinCE 和 ARM 的多串口扩展及 485 通信设计①

贾继鹏, 张永坚, 胡延凯

(山东建筑大学 信息与电气工程学院, 济南 250101)

摘 要: 随着我国《可再生能源建筑应用工程评价标准》的实施, 对已建可再生能源建筑应用示范项目运行数据的需求越来越迫切, 为了解决可再生能源建筑应用示范项目数据采集系统中对多串口及 485 总线通信的需求, 采用 ARM9 嵌入式微控制器 S3C2440 和具备 UART/SPI 接口的 4 通道芯片 VK3234 进行串口的扩展, 并利用 MAX3485E 芯片, 实现了现场检测设备与数据采集器之间的 485 通信. 本文对 S3C2440 和 VK3234、MAX3485E 芯片之间的接口和软硬件的设计进行了具体的阐述, 在实验室的实际运行测试中表明该系统稳定可靠, 并运用到可再生能源示范项目数据监测系统中.

关键词: S3C3440; VK3234; MAX3485E; 串口扩展; 可再生能源; 数据采集

Extended Multiple Serial Port and 485 Communication Based on WinCE and ARM

JIA Ji-Peng, ZHANG Yong-Jian, HU Yan-Kai

(Shandong Jianzhu University, School of information & Electrical Engineering, Jinan 250101, China)

Abstract: Along with the book evaluating standards for renewable energy building's application projects implements in our country, we are in urgent need of the operational datas about the renewable energy building's application demonstration projects. The data acquisition system of the renewable energy building's application demonstration projects has one demand of multiple serial ports and a 485 bus communication. This paper uses ARM9 S3C2440 embedded microcontroller and VK3234 with four channel UART/SPI interface to extend serial. What's more, it achieves the 485 communication between the field detection device and data acquisition device with MAX3485. This paper states the interface and software and hardware between S3C2440, VK3234 and MAX3485E specifically. The practical running test in the laboratory shows that the system is stable and reliable, and it has been applied in the data monitoring system of the renewable energy demonstration projects.

Key words: S3C2440; VK3234; MAX3485E; serial port expansion; renewable energy sources; data acquisition

随着嵌入式系统的迅猛发展, 以及现场总线技术的大量应用, 串口通信在各领域的应用越来越广泛. 在可再生能源建筑应用示范项目数据采集系统中, 需要对 8 个以上的不同串口设备进行数据采集和数据分析[1], 为了保证数据的准确性和系统的稳定性, 控制器需要完成多个串口设备间进行数据的通信和交换, 且具有数据分析功能. 普通设备的主控器通常采用单片机, 一般只有 1~3 个串口, 且没有数据分析功能, 不能满足数据采集系统中的多点采集和数据分析功能需求. 为了满足可再生能源建筑应用项目对数据多点采集的需要, 本文对 S3C2440 处理器的 UART 接口进行了扩展, 通过采用 2 片 VK3234 串口扩展芯片[2], 由 S3C2440 处理器的两个 UART 接口扩展出 8 个串口; 基于 ARM S3C2440 处理器的嵌入式系统, 可以运行 WINCE 操作系统, 安装嵌入式数据库, 并具有数据分析功能, 且其丰富的接口和功能强大的外围接口芯片,

① 收稿时间:2014-08-21;收到修改稿时间:2014-10-20

为串口的扩展和 485 通信的设计提供了方便.

1 系统概述

为了满足数据的多点采集和数据分析功能, 本设计的方案采用 S3C2440、VK3234、MAX3485E 的硬件框架结构, 软件平台基于 WINCE 6.0 操作系统[3], 其硬件和软件的开发周期较短. 系统总体框图如图 1 所示.

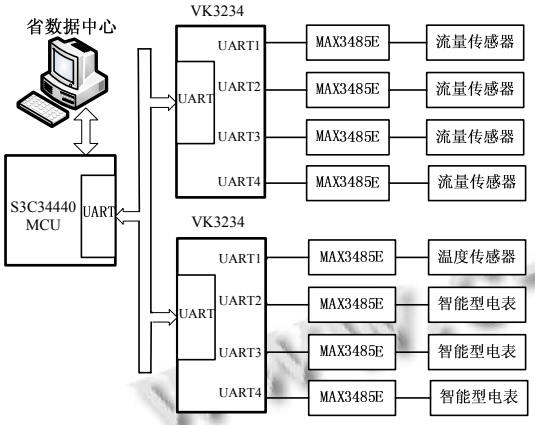

图 1 系统总体框图

该设计包括串口扩展、RS-485 通信硬件电路设计和 VK3234 驱动程序的开发.

2 硬件设计

2.1 核心元器件选型

2.1.1 S3C2440 微控制器

S3C2440 是由三星公司推出的 32 位 RISC (Reduced Instruction Set Computer)微处理器, 为各领域的应用提供了低价格、低功耗、高性能小型微控制器的解决方案. 其采用了 ARM920T 的内核[4], 0.13um 的 CMOS 标准宏单元和存储器单元, 具有 130 个通用 I/O 口和 24 通道外部中断源、3 个 UART 接口、SD 卡接口, 主频可达 533MHz[5]. 本文的设计选用两片 VK3234 芯片对其中两路 UART 接口分别扩展.

2.1.2 VK3234 多总线接口、四通道通用异步收发器

VK3234 不需要地址线控制串口扩展方式, 通过芯片内置的协议处理器实现多串口的扩展, 是具备 UART/SPI 接口的 4 通道 UART 器件, 具有自动 RS-485 收发控制的 RTS 引脚. 可以通过模式选择使得该芯片工作于 UART 或者 SPI 主接口模式, 本文的设计主接口模式为 UART 接口模式[6]:

当主接口为 UART 时, VK3234 将一个标准 3 线异步串口扩展成为 4 个增强功能串口. 主接口 UART 在数据传输时可以选择需要转义字符和不需要转义字符两种模式. 扩展的子通道的 UART 具备如下功能特点:

每个子通道 UART 的波特率、字长、校验格式可以独立设置, 最高可以提供 1Mbps 的通信速率. 每个子通道可以独立设置工作在 IrDA 红外通信、RS-485 自动收发控制、9 位网络地址自动识别、硬件自动流量控制、广播接收等高级工作模式下. 每个子通道具备收/发独立的 16 BYTE FIFO(First Input First Output), FIFO 的中断为 4 级可编程条件触发点.

VK3234 可以工作在 2.5～5.5V 的宽工作电压范围, 具备可配置自动休眠/唤醒功能.

2.1.3 MAX3485E 低功耗、RS-485/RS-422 收发器

MAX3485E 是用于 RS-485 和 RS-422 通信的半双工、低功耗收发器, 可以实现最高 12Mbps 的传输速率、工作在 3.3V 电压下[7].

2.2 硬件逻辑电路设计

硬件逻辑电路原理图如图 2 所示.

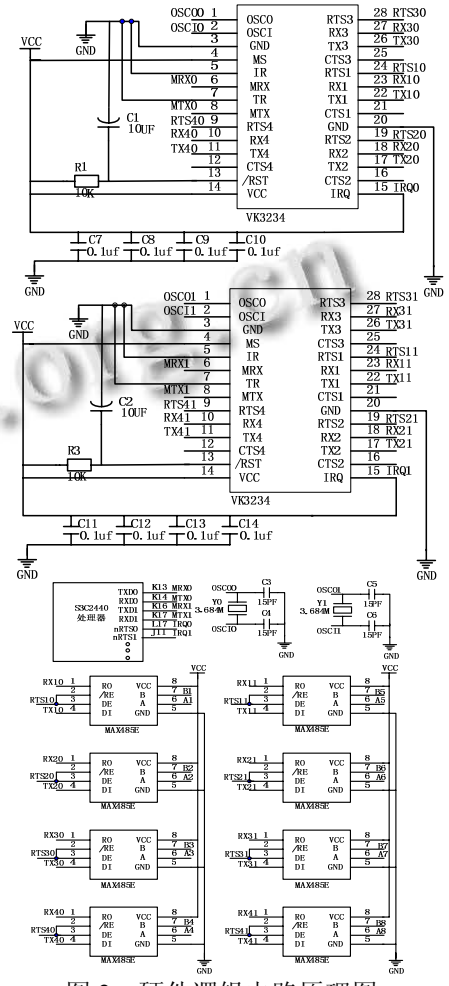

图 2 硬件逻辑电路原理图

首先选择主机的接口模式, VK3234 支持主机接口为 UART/SPI 接口的串口扩展; 图 2 中 MS 引脚接高电平, IR、TR 引脚接低电平, 此时主机接口为普通、无转义字符 UART 通信. OSCO 和 OSCI 引脚分别接 3.684MHz 外部晶振的输入和输出, 以获取时钟; 2 片 VK3234 芯片的 MRX0、MTX0, MRX1、MTX1 引脚分别接 S3C2440 芯片的 TXD0、RXD0, TXD1、RXD1 引脚; TX1、RX1, TX2、RX2, TX3、RX3, TX4、RX4 为扩展出来的 4 路子串口, 2 片共 8 路, 分别与 8 片 MAX3485E 芯片的 DI、RO 引脚连接, 构成可自动网络地址识别的 RS-485 通信线路. RTS1~RTS4 连接 MAX3485E 芯片的收/发控制引脚/RE、DE, 该设计中子串口工作在 RS-485 自动收发模式下, RTSx 引脚用于控制 RS-485 数据的自动收发转换, 只有发送数据时 RTSx 才为高电平, 其他情况下, RTSx 保持低电平,可通过子串口状态寄存器进行配置.

每个子串口为全双工接口, 可以通过软件分别开启/关闭, 波特率可独立设置, 子串口的速度可设置为 300bps~920Kbps; 每个子串口的字符格式包括数据长度、停止位数、奇偶校验模式选择, 均可独立设置.

3　WinCE下VK3234串口驱动的实现

3.1 流驱动程序结构介绍

本文开发的 VK3234 串口驱动是一个标准的流驱动, 分为模型设备驱动(Model Device Driver, MDD)和平台相关驱动(Platform Dependent Driver, PDD)[8], 其结构如图 3 所示.

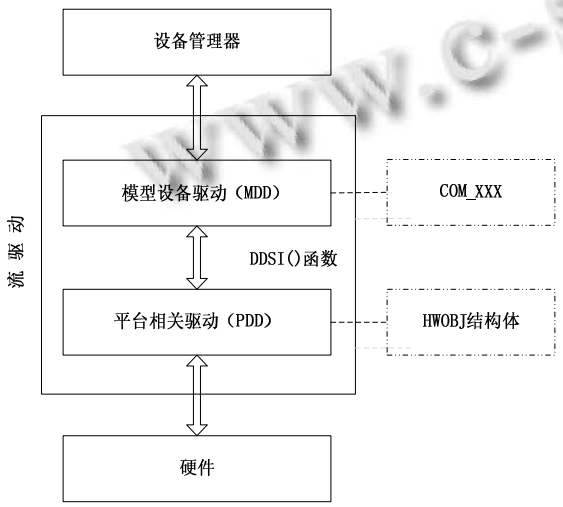

图 3　流驱动程序结构图

模型设备驱动层(MDD 层)包含给定类型的所有驱动程序公用的代码, 其主要功能有两部分:

① 为上层的设备管理器提供了标准流设备驱动接口函数 COM_XXX;

②链接平台相关驱动层(PDD 层), 并调用 PDD 层中与串口操作相关的函数[9];

平台相关驱动层(PDD 层)由与特定的硬件设备相关的代码组成, 实现了 HWOBJ 结构体及结构体中若干针对于串口硬件操作的函数指针, 这些函数指针将指向 PDD 层中的串口操作函数, 并向 MDD 层提供设备驱动服务提供者接口(Device Driver Service provider Interface, DDSI), MDD 层调用 DDSI 函数对设备进行操作.

3.2 VK3234 芯片 UART 通讯协议

VK3234 芯片寄存器采用[C1:C0:A3:A2:A1:A0] 6 位地址编码的形式, [C1:C0]为子串口号, 取值从 00 到 11. A0~A3, 4 个全局寄存器的编号值从[000000]到[000011], 每个子串口有 10 个寄存器, 编号值从[C1:C0:0:1:1:0]到[C1:C0:1:1:1:1]. VK3234 可以仅通过 Tx, Rx 和主机的 UART 连接, 使用标准的 UART 协议进行通讯. 主机和各个扩展的子串口以 VK3234 的主串口为媒介进行通讯. 主机读取某个子串口寄存器时, 首先向 VK3234 主串口发送一个指令, 然后 VK3234 解析指令返回结果. VK3234 主要包括四种不同的指令: 写寄存指令(10)、读寄存器指令(00)、多字节写 FIFO 指令(11)、多字节读 FIFO 指令(01). 每个指令都是 8 位指令中最高的两位表示操作类型, 紧接着的两位是子串口号, 然后紧跟低四位是子串口寄存器编号或者操作数据的字节数.

不同的指令, 操作时序不一样, 以主机读取串口 1 的控制寄存器 SCTLR(0110)为例. 主机在 TX 上发送指令 00010110, 然后 VK3234 将对应的寄存器值通过 RX 传输给主机. 又如主机发送 01010011 指令, 请求 VK3234 从子串口 1 的 FIFO 中读取 3 个字节, 请求发出后主机可以在 RX 读取三个字节的数据.

3.3 VK3234 串口驱动的实现

WinCE 中微软提供了 S3C2440 的串口的驱动程序 PDD 层的实现, 虽然不能完全满足该设计, 但是可以在原有程序的基础上进行二次开发, 实现 VK3234 串口驱动. 图 4 是本设计中 VK3234 串口驱动的派生类图, 由于篇幅原因仅将重要信息展示在图中. 其中

CPddVk3234 和 CRegVk3234 是需要重新实现的, CPddVk3234 是驱动的 PDD 层的实现, 该类从 BSP 中提供的 CPdd2440Uart 派生而来, 实现了初始化寄存器、接受和发送数据等功能. CRegVk3234 由类 CRegVk3234 派生而来, 新增操作的主要功能是对 VK3234 的寄存器进行读写操作. 接下来对涉及到的类进行详细的说明.

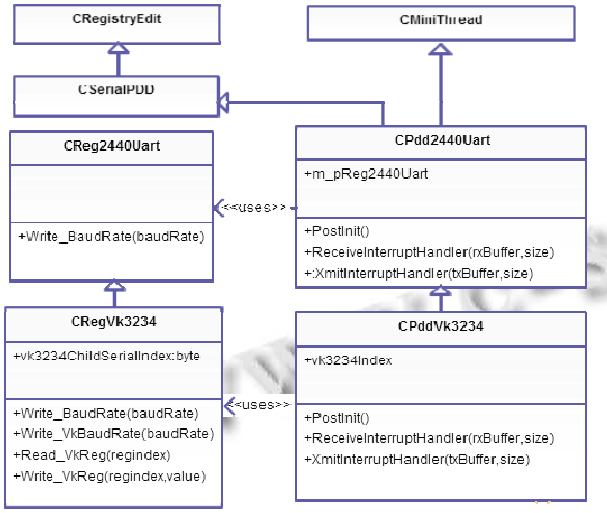

图 4 VK3234 串口驱动派生类图

CRegistrEdit 是 WinCE 提供的注册表操作类, 驱动通过继承该类获得操作注册表的功能.

CMiniThread 类是对 WinCE 中运行线程的抽象, 封装了线程的一些基本操作, 如开启线程、停止线程等. 驱动通过继承该类开启 IST(Interrupt Server thread)线程, 主要功能是监听中断事件、查找中断源、修改中断标志、向驱动程序报告中断、清除中断标志、结束 IST 中断过程等.

CSerialPDD 是一个抽象的虚类, 它定义流驱动处理的逻辑, 将所有流驱动的共性抽象抽出来进行实现, 然后将不同的部分定义成接口, 不同的流驱动中进行具体的实现. 即 CSerialPDD 就是一个模板, 只要根据需要填充这个模板就可以实现相应的驱动.

XmitInterruptHandler(PUCHAR pTxBuffer, ULONG *pBuffLen)是 CSerialPDD 定义的一个方法, 这个方法会在发送中断产生后执行, 功能是通过流驱动发送 pBuffLen 个字节的数据, 待发送的数据存储在 pTxBuffer 中, 实现了该方法就实现了驱动发送数据.

CPdd2440Uart 是微软提供的 S3C2440 串口驱动的实现, 该类实现了 CSerialPDD 里面定义的操作串口方法.

CReg2440Uart 定义了操作 S3C2440 串口寄存器读写方法. 该类被 CPdd2440Uart 引用, 用于操作串口.

CegVk3234 从类 CReg2440Uart 派生而来, 该类添加了操作 VK3234 寄存器的能力, 重新定义了设置串口波特率的方法. 属性 VK3234ChildSerialIndex 用于表示当前操作的是 VK3234 扩展出来的第几个子串口. 方法 Read_VkReg()的作用是读取当前子串口寄存器, 该方法接收 redindex 参数包含的信息, 其表示所要读取的寄存器编号. 方法 Write_VkReg()接收两个参数 regindex 和 value, 功能是将 Value 写入第 regindex 个寄存器. 为了保持 S3C2440 和 VK3234 能正常通信, 必须时刻保持两端波特率一致, 所以重写了 CReg2440Uarts 设置波特率的方法 Write_BaudRate(), 该方法执行之前会先调用 Write_Vk BaudRate 设置 VK3234 子串口的波特率, 然后再设置主机串口波特率. CPddVk3234 就是该设计所需要的驱动实现, 其中最重要的方法是 PostInt()、XmitInterruptHandler()、ReceiveInterruptHandler(). PostInt()主要用于主机串口和 VK3234 初始化设置; XmitInterruptHandler()用于发送数据; ReceiveInterruptHandler()用于操作数据接收. vk3234Index 表示当前子串口隶属于第几块 VK3234 芯片.

3.4 代码解释

Device.exe 是 WinCE 的设备管理器, 负责几乎所有外围设备驱动的管理. 系统启动后 Device.exe 根据注册表中的配置, 加载驱动对应的 DLL 文件, 并对 CPU 的 IO 口进进行初始化设置.

注册表配置:

```
[HKEY_LOCAL_MACHINE\Drivers\BuiltIn\Serial2]
    "DeviceArrayIndex"=dword:1
    "Irq"=dword:f
    "MemBase"=dword:50008000
    "Index"=dword:2
    "Dll"="serial_vk3234.dll"
    "Port"="COM2:"
    ……
```

上面是串口 COM2 在注册表中的定义[10], DeviceArrayIndex 定了设备索引; Irq 定义设备的物理中断号; MemBase 是段基地址; COM2 到 COM5 都是由 S3C2440 的串口 1 扩展而来的所以串口 COM2 到 COM5 关于段地址的定义是一样的, ; DLL 文件就是对

应的驱动程序. 根据注册表的定义通过如下代码创建串口驱动, 并进行初始化.

初始化 IO 口:

```
CSerialPDD * pSerialPDD = NULL;
pSerialPDD= new CPddVk3234(lpActivePath,pMdd, pHwObj, deviceArrayIndex);
if (pSerialPDD && !pSerialPDD->Init()) {
……
//映射物理地址
if(TranslateBusAddr(m_hParent,Internal,0, ioPhysicalBase,&inIoSpace,&ioPhysicalBase){
//映射 IO 口
m_pIOPregs =(S3C2440A_IOPORT_REG* )MmMapIoSpace(ioPhysicalBase,sizeof(S3C2440A_IOPORT_REG),FALSE);
……
//获取逻辑中断号
if (GetIsrInfo(&ddi)== ERROR_SUCCESS && KernelIoControl(IOCTL_HAL_REQUEST_SYSINTR, &ddi.dwIrq, sizeof(UINT32), &ddi.dwSysintr, sizeof(UINT32), NULL)){
……
}
//初始化对应的 IO 口
if(vk3234Index==1){
pIOPregs->GPHCON &= ~(0x3<<12 | 0x3<<14);
m_pIOPregs->GPHCON |= (0x2<<12 | 0x2<<14);
m_pIOPregs->GPHUP |= 0xc0
}else{
 //另一块 VK 芯片
    }
```

CPddVk3234 的 init( )方法主要作用是将总线相对地址转换成系统物理地址, 然后映射为 IO 虚拟地址空间, 获取逻辑中断号, 并初始化 IO 口, 拉起相关引脚对应的电平.

初始化芯片:

```
    //省略初始化主机串口寄存器代码
    //设置主机串口波特率与 VK 芯片波特率相同
    if((Read_UCON()&CS_MASK) == CS_PCLK) {
Write_UBRDIV(int)(m_s3c2440_pclk/16.0/VK3234_DEFAULT_BAUD)-1);
    }
    //初始化子串口控制寄存器
    write_VkReg(SCTLR,0X38);
    //初始化子串口流量控制寄存器
    write_VkReg(SFWCR,0X0E);
    //设置子串口其他寄存器
```

应用程序打开串口时, 驱动程序首先初始化主机串口的寄存器, 然后设置其波特率与 VK 芯片的波特率相同, 保证同 VK 芯片进行正常通信, 然后设置子串口寄存器. 进而初始化子串口控制寄存器, 使其工作在 8 位数据的 RS-485 模式下, 然后配置子串口流量控制寄存器, 配置 RTS 信号为手动硬件流模式.

发送和接收数据:

由于前面配置 RTS 信号为手动硬件流模式, 所以在发送数据之前要拉高 RTS 信号, 使 RTS 保持高电平, 准备发送数据, 然后向子串口 FIFO 数据寄存器 SFDR 写入要发送的数据, 数据发送完毕后, 及时拉低 RTS 信号, 使芯片处于接收状态. 接收数据, 主机串口不断查询子串口的中断状态寄存器, 是否有接收中断, 如果有则读取 FIFO 状态寄存器 SFSR, 读取可接受数据个数, 然后从 FIFO 数据寄存器 SFDR 读取相应的数据. 由于篇幅的原因仅给出数据发送流程的部分代码.

```
    //读取 SIDR 备用
    sifr = m_pReg2440Uart->read_VkReg(SIFR,0X08)
    //拉高 RTS 信号,准备发送数据
    m_pReg2440Uart->Write_VkReg(SIFR, sifr|0X08)
        for (DWORD dwByteWrite=0; dwByteWrite<dwWriteSize&&dwDataAvaiable!=0;dwByteWrite++) {
    //数据写入 SFDR
 m_pReg2440Uart->Write_VkReg(SFDR, *pTxBuffer);
    pTxBuffer ++;
    dwDataAvaiable--;
    }
    //数据发送完毕拉低 RTS 信号
m_pReg2440Uart->Write_VkReg(SIFR, sifr&0Xf7);
```

4 工程应用

本文设计的多串口数据采集器已用在了山东省多个可再生能源建筑应用示范项目中. 以东营市垦利县创业大厦办公楼的地源热泵系统数据采集项目为例,

该项目共计使用电能表 3 块、300/5A 开合式电流互感器 9 个、数据采集器 1 个、超声波流量计 4 块、DS18B20 探头及变送器 1 套, 电气元件 1 宗; 需要采集电量数据 3 个, 流量数据 4 个, 温度数据 8 个; 实际应用中, 该数据采集系统每隔 5 分钟采集一次数据, 每隔 30 分钟向省级数据中心上传 1 次数据, 并将采集和分析得到的数据保存 30 天, 数据采集器带有 VGA 接口, 可根据需要增加显示屏, 以方便数据展示和系统调试. 目前, 该数据采集系统运行稳定, 数据采集、传输准确, 系统应用照片如图 5, 省数据中心数据接收界面如图 6.

图 5 系统应用照片

图 6 省数据中心数据接收界面

5 结语

本着运用简单的技术和元器件, 解决复杂问题的原则, 本文利用最常用的 ARM9 微处理器 S3C2440 和 VK3234、MAX3485E 芯片实现了基于 WinCE 6.0 操作系统的多串口扩展及 485 通信, 给出了软硬件的详细设计方法和过程. 本文主要有两个创新点: ① 系统设计方面, 本文选用了不需要地址线控制串口扩展方式的高性能芯片, 并具有 RS-485 自动收发控制功能, 给出了具体的硬件、驱动实现方法, 扩展出来的每个子串口都具有独立的使能功能, 使系统的软硬件设计更加简单; ②应用方面, 作者已将本文的设计成功应用到了可再生源示范项目数据监测系统中, 系统运行稳定可靠.

参考文献

1 中华人民共和国住房和城乡建设部,可再生能源建筑应用示范项目数据监测系统技术导则,2009.
2 吕雪芹,敖振浪,陈冰怀.基于 VK3234 的共享中断编程关键技术.电子测量技术,2014,04:87–91.
3 肖中华,张扬,王靖鑫,崔乃峰.基于 WinCE6.0 嵌入式工业控制系统开发.工业控制计算机,2014,6:1–3,6.
4 孙良义,张勇,刘洁.基于 ARM 和 WINCE 的便携式差分 GPS 导航定位系统设计及实现.电子设计工程,2013,22: 91–94.
5 S3C2440 Microcontroller User's Manual.
6 VK3234 User'S Datesheet.
7 MAX3485 User'S Datesheet.
8 陈媛,王再英彭,倩刘颖.基于嵌入式多串口数据传输系统的设计与实现.科学技术与工程,2013,1:183–187.
9 许跃华,唐明浩.基于 WINCE 的多串口扩展系统的设计.微计算机信息,2010,26:83–85.
10 李玉超,李红梅,马田.基于 ARM9 的多功能警用终端的串口扩展设计.科技信息,2012,01:99–100.