基于 μC/OS – Ⅱ的远程抄表系统终端设计

The Design of Remote Meter Reading Terminal Based on μC/OS – Ⅱ

王 琦　刘丽丽　(青岛 中国海洋大学工程学院机电工程系 266071)

摘要:本文提出了一种基于嵌入式操作系统 μC/OS – Ⅱ的远程抄表系统终端设计方案,可较好的满足远程抄表系统对可靠性和实时性的要求。该方案给出了系统终端的硬件体系结构,并结合 μC/OS – Ⅱ的特点详述了系统软件的具体实现,最后讨论了 μC/OS – View 在 μC/OS – Ⅱ中的移植与应用。

关键词:μC/OS – Ⅱ　抄表　μC/OS – View　消息队列

1　引言

远程抄表系统通过 GPRS 无线通讯网络,将电量数据和其它(所需)信息实时可靠的采集回来,并由软件对数据进行统计、分析和计算。采用此系统可以提高电能计量管理水平,了解电网实时运行状况,从而可以更好的保证电网安全运行。

2　远程抄表系统终端的功能及硬件组成

2.1　系统功能

本系统终端实现的主要功能包括:通过 485 接口定时、实时采集电能表相关数据并自动存储;GPRS 通信功能;自动记录线路停电时间;遥信、遥控功能;自诊断功能和报警功能等。

2.2　硬件组成

(1) 系统的核心是 TMS320LF2407A,它是 TI 公司推出的 16 位 DSP,片内资源丰富,本系统应用了其 SPI、SCI、AD 及通用 I/O 口等功能模块。

(2) 外扩存储模块:选用 ATMEL 公司的 AT45DB041B 作为数据存储器,用于存储电表数据、系统配置参数和电表通信规约。

(3) 定时模块:系统外扩 PCF8563 作为系统时钟,采用其定时中断输出功能实现定时抄表。

(4) 通信模块:本系统与电能表的 485 通信选用 TI 公司的 SN65LBC184 芯片。与主站的 GPRS 通信选用 Sony Ericsson 公司的 GR47 模块。

(5) 模拟量输入模块:通过 PT、CT 实时采样电网三相电压、电流,若发生异常情况则报警。

(6) 电源管理模块:电源管理芯片选用 MAXIM 公司的 MAX793 实现。MAX793 提供的保护功能有硬件复位,看门狗,备份电源切换,RAM 写保护,掉电警告等。

(7) 显示模块:LCD 显示部分选用 MSG19264 液晶,采用 DSP 的 I/O 口实现对液晶的访问。

远程终端系统的硬件结构如图 1 所示。

3　系统软件设计

我们将实时操作系统(RTOS)概念引入抄表系统终端的软件设计,采用 μC/OS – Ⅱ实时嵌入式操作系统作为系统软件的核心。本系统的软件基于三层设计,包括操作系统层、中间件层和用户任务层,并采用 μC/OS – View 观测操作系统的运行状态,最终实现了整个系统稳定、可靠的运行。

3.1　操作系统层

μC/OS – Ⅱ是由 Jean J. Labrosse 编写的嵌入式实时操作系统[1],具有源码公开、可移植、可固化、可裁减、稳定性和可靠性高的特点,还可方便的在 μC/OS – Ⅱ的基础上添加 μC/FS, μC/GUI, μC/TCP – IP, μC/USB, μC/FL 等中间件,这些都使得软件的开发变得更加简单。μC/OS – Ⅱ在 TMS320LF2407A 上的移植可见官方网站。

3.2　中间件层

中间件是使嵌入式应用独立于具体软硬件平台的软件环境,本系统采用的中间件包括加密算法和 GUI 图形界面。

GPRS 通信需通过公共通信网进行数据传输,因此传输数据的安全性非常重要,考虑到嵌入式系统的特点,加解密的实时性与易实现性亦非常重要。由于 TEA 加解密算法实现简单,速度快且加密强度不低于常用的 DES 算法[3],因此本系统采用 TEA 加解密算法保护通信数据。

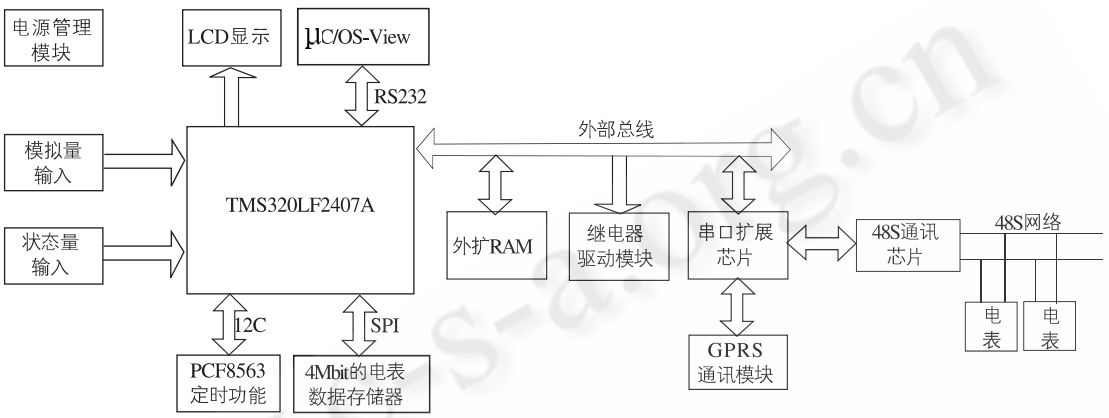

图 1　系统硬件结构框图

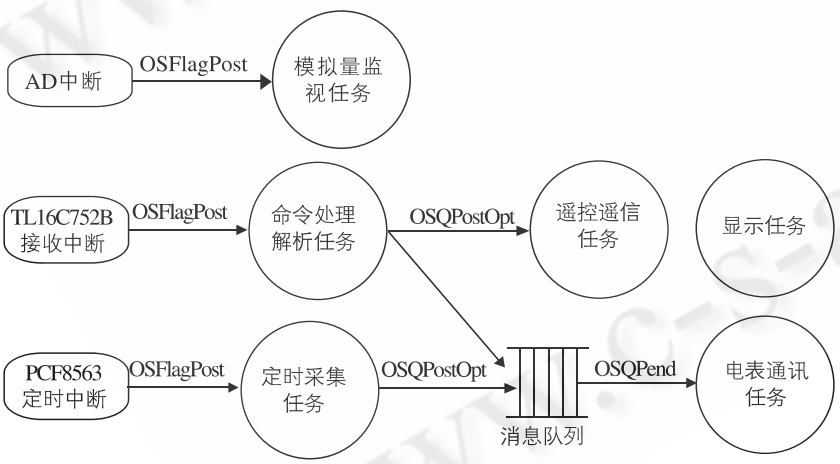

图 2　任务间的关系图

注:OSQPostOpt、OSQPend、OSFlagPost 均为操作系统提供的函数,功能分别是发送消息、接收消息以及发送信号量

GUI 图形界面基于 uC/GUI 实现,uC/GUI 是一款优秀的图形软件,可以和 μC/OS－Ⅱ无缝连接,具有可移植、可裁减、使用灵活的特点,可根据需要方便的在液晶上显示文本、曲线和图形等,uC/GUI 在 MSG19264 液晶上的移植可见参考文献[5]。

3.3　用户任务层

根据所要实现的功能,我们设计了命令解析任务、电表通信任务、定时采集任务、模拟量监控任务、遥控遥信任务及显示任务。终端系统上电后首先进入 GR47 的命令模式,在通过 AT 命令建立 TCP 连接后,GR47 进入数据模式,在 GR47 的数据模式下,各个任务之间的关系如图 2 所示。

命令处理解析任务负责对收到的命令进行解释和分发,命令主要分为实时采集电表数据、系统参数设定、遥控遥信以及历史数据上传四类命令。任务收到信号量后首先判断信息的合法性,若合法则进行命令的解释和分发,然后清空接收缓冲区等待下一次命令的到来。

定时采集任务实现每隔一段时间采集电表各类数据然后存储到指定位置或上报主站的功能。若规定时间到达定时采集任务通过消息队列发送消息至电表通信任务。

电表通信任务负责电表通讯和相应处理。任务收到消息后即与电表通信,然后把返回的电表数据根据要求存储或上传。

遥控遥信任务实现远程控制和远程状态获取,可根据主站要求进行遥控变位或遥信数据上传。

模拟量监视任务实时监视线路电流、电压状态,当线路出现过流、过压、欠压、欠流、断相等异常情况时向主站报警。

显示任务实现显示当前时间、系统状态和各种故

障提示。

各个任务的优先级设置以及任务内部调用的底层硬件驱动程序可见表1：

表 1　各个任务的优先级设置以及采用的底层驱动

任务名称	任务优先级	底层驱动		
		实时时钟	存储驱动	串口驱动
命令处理解析任务	2		采用	采用
定时采集任务	4	采用		
电表通信任务	5		采用	采用
遥控遥信任务	3			采用
模拟量监视任务	1(最高)			采用
显示任务	6(最低)	采用		

3.4 任务间通信

各个任务是通过抢占 CPU 的使用权来运行的，它们之间存在一定的逻辑关系，彼此互相联系又互相制约。信号量、邮箱、消息队列等功能为实现任务间的通信提供了有力工具。

3.4.1 信号量

本系统采用信号量用于标志事件发生和控制共享资源的使用权。系统采用三个信号量分别用于标志串口收到字符、A/D 转换完毕和收到 PCF8563 中断这三个特定事件。从表1可看出底层的三个驱动做为共享资源可被其他任务调用，因此采用三个信号量用于保护共享资源(GR47、AT45DB041 和 PCF8563)的使用，为了防止冲突，任务在使用共享资源前必须得到信号量，使用完毕后再释放信号量以供其他任务使用。

3.4.2 消息队列

由于本系统既需要实时采集电表数据，又需要定时采集，为了很好的将二者区分，又能体现出实时采集优先级高的特点，本系统采用消息队列来实现。

定时采集任务发送的消息用于电表数据分时存储或上传，实时性较弱，采用“先进先出”的方式发送消息。而命令处理解析任务发送的消息用于电表数据实时招测，要求快速响应，其优先级应高于定时采集任务，因此采用“后进先出”即通过插队的方式来向消息队列发送消息。

我们定义了一个电表数据结构来实现此消息队列，具体定义如下：

```
typedef struct {
    INT16U DisposeStyle; /* 任务处理方式:存储或发送 */
    INT16U StorePage; /* 存储位置 */
    INT16U StoreMode; /* 存储模式:重写,添加 */
    INT16U *Meter_Frame; /* 指针指向要发送的电表通信帧 */
    INT16U Flag; /* 采集完成标志 */
} Meter_DataStruct;
```

数据结构中 INT16U 为操作系统定义，表示 16 位无符号整型。DisposeStyle 定义了电表通信任务对待这个消息的处理方式，为 1 表示存储，为 2 表示发送，为 3 表示存储且发送；StorePage 定义了存储位置，AT45DB041 由 2048 个页组成，StorePage 即一一对应于 AT45DB041B 的页；StoreMode 定义了存储模式，为 0 表示重写模式，为 1 表示添加模式，这样每一个存储位置便可以存储多条信息；Meter_Frame 指针指向要发送的电表通信帧；Flag 定义了采集完成标志，为 0 表示采集完成，为 1 表示采集没有完成。

发送消息时，任务首先根据需要填充这个数据结构，然后调用 OSQPostOpt 函数发送指向这个数据结构的指针。

电表通信任务接收到消息后，把消息指针指向的电表通信数据帧通过 TL16C752 向电表发送，待电表数据返回后，任务会根据要求存储或发送收到的数据帧并把采集完成标志置 0。

3.5 采用 μC/OS－View 实时观测系统的运行状态

μC/OS－View 是 Micrum 公司为了用户能更好的运行与调试 μC/OS－Ⅱ而提供的一套软件[4]，现在的版本为 1.1。通过 μC/OS－View，用户可以实时查看 μC/OS－Ⅱ所管理的各个任务状态，实时获得系统运行的各种信息，以此分析整个系统的运行状况并解决系统运行出现的问题，更合理的配置各个任务的优先级、堆栈等属性。μC/OS－View 还提供了 Terminal 窗口，用户可以根据需要在程序中自己定义回调函数，这样就可通过 Terminal 窗口实现特定信息的动态交换。

3.5.1 通信帧格式及软件流程

μC/OS－View 通过 RS－232C 串口与终端通信。为保证通信可靠，μC/OS－View 定义了一个通信协议，

接收时根据协议对收到的数据进行解码,发送时根据协议对数据打包。以 μC/OS – View 的通信协议为基础,完全可以按自己的要求来编写上位机软件。通信协议的具体定义如表 2 所示。

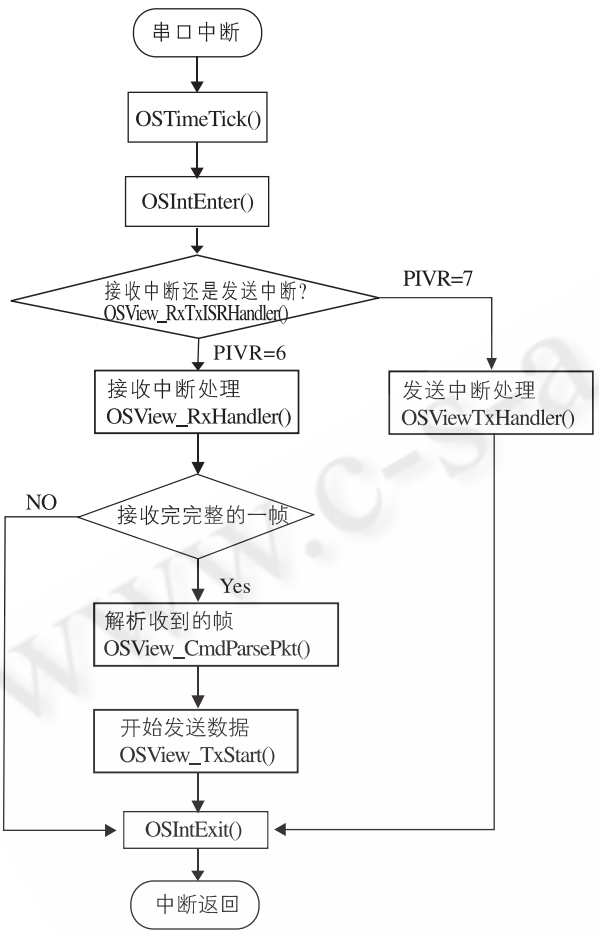

图 3　μC/OS – View 软件流程图

表 2　μC/OS – View 中的通信协议

第一起始符	第二起始符	数据长度	数据	校验码	结束符
0xed	0x8c	Len	Data1. . len	OSView_RxChkSum	0x8d

目标板接收数据时需判断接收到的数据是否正确,若正确进行数据解析,若错误则丢掉收到的数据包。定义 OSView_RxChkSum = len + data1 + data2 + … + datalen + OSView_RxChkSum,若 OSView_RxChkSum 的低八位为零则这一帧接收正确。当目标板向 PC 发送数据时同样需要发送校验码,其值为 len + data1 + data2 + … + datalen。

μC/OS – View 的软件流程如图 3 所示。

3.5.2　μC/OS – View 的移植

μC/OS – View 的移植一般可以根据文献[4]对与硬件有关的三个文件 OS_VIEWc. C、OS_VIEWc. H 和 OS_VIEWa. asm 进行修改实现。但考虑到 TMS320LF2407A 的具体情况,还需对 OS_VIEW. C 进行修改,具体如下:

(1) 在 OS_VIEW. H 文件中将 OSView_RxChkSum 的数据类型定义成 char 型,但考虑到 TMS320LF2407A 中 char 型的长度为 16 位而不是 8 位,需要将 OSView_RxChkSum&0xff(即取其低 8 位)再判断其值是否为 0,若为 0 则校验通过。

(2) 由于在 μC/OS – View 中定义的一些变量没有初始化,因此在移植过程中在其初始化函数 OSView_Init( )中应将这些变量进行初始化。

总之,μC/OS – View 就如同 windows 操作系统中的任务管理器一样,可以通过 μC/OS – View 获得系统信息,从而合理的安排各个任务的堆栈大小、优先级等。这样可以使整个系统运行的更稳定,并且能够更好的对系统进行优化与裁减。

4　结论

通过采用 μC/OS – Ⅱ 实时操作系统,GPRS 远程抄表系统终端的可靠性和实时响应能力均得到很大的提高,软件的开发速度也大大加快。GPRS 远程抄表系统已完成样机设计,并在现场中取得了满意的效果。

参考文献

1　Jean J. Labrosse 著,邵贝贝译,嵌入式实时操作系统 μC/OS – Ⅱ(第 2 版),北京航空航天大学出版社,2003。

2　TMS320 LF/LC 240x DSP Controllers reference guide, Texas Instruments,2000.

3　史斌宁、刘昊钰,TEA 加解密算法在嵌入式系统通信中的应用,单片机与嵌入式系统应用,2004,4:73 – 74。

4　μC/OS – View V1. 10 User's Manual. Micriμm, 2002.

5　刘滨、王琦、刘丽丽,uC/GUI 在 MSG19264 液晶上的移植,电子技术应用,2004,8。