文章编号: 1671- 251X(2010)12- 0013- 04

基于多线程的环境监控系统下位机的设计

陈立定, 吕盛林

(华南理工大学自动化科学与工程学院, 广东 广州 510640)

摘要: 为提高环境监控系统的稳定性, 节约系统资源, 提出了一种由下位机、传输网络和上位机组成的环境监控系统的设计方案, 详细介绍了该系统中下位机的硬件及软件设计。该下位机硬件以 ARM9 处理器 S3C2410 为核心, 软件采用多线程应用程序同时处理多个任务, 并采用信号量和互斥量实现线程间的同步。实际应用表明, 该系统运行稳定, 提高了系统效率。

关键词: 环境监控; 下位机; 多线程; 同步; S3C2410; Linux

中图分类号: **TD99** 文献标识码: **B**

Design of Lower Computer in Environment Monitoring System Based on Multi-thread

CHEN Li-ding, LÜ Sheng-lin

(College of Automation Science and Engineering of South China University of Technology, Guangzhou 510640, China)

收稿日期: 2010- 08- 11

基金项目: 佛山市产学研专项资金项目(2007CB009), 佛山市南海环境保护局项目(2007B08D8073570)

作者简介: 陈立定(1964-), 男, 湖北天门人, 教授, 硕士, 1992 年毕业于华南理工大学自动化系, 现主要从事网络化控制系统理论与应用的研究工作。E-mail: llshenglin@126.com

100 M 甚至 1 000 M 级, 其网络带宽的占用量相对较少, 网络剩余带宽足以满足矿用 CDMA 无线通信系统移动语音通信的需求。

2.2.2 FEMTO 基站的技术参数

防爆型式: 本质安全型;

供电: 直流 18±0.5 V, 电流不大于 1 A;

无线接入: CDMA2000 1X;

以太网接口: 1 个 10/100 Base- T;

工作频率: 450/800/1 900 MHz;

RF 功率: ≤50 mW(+17 dB·m);

容量: 4~6 个活动用户(最多可支持 16 个);

覆盖半径: 50~200 m;

协议支持: SIP、DHCP、TFTP、SNMP;

天线: 内置或外接天线。

3 结语

介绍了将 FEMTO 基站应用到矿用 CDMA 无线通信系统中的技术方案, 其传输网络充分利用煤矿已有的工业以太网, 使矿用 CDMA 无线通信系统的安装维护成本大大降低, 提高了系统的性价比。初步测试结果表明, 基于 FEMTO 基站的矿用 CDMA 无线通信系统基站覆盖范围大、手机通话效果好, 是一种性能优良的矿用移动通信解决方案。

参考文献:

[1] 孙继平. 矿井无线传输的特点[J]. 煤矿设计, 1999(4): 20-22.

[2] 3GPP TS 25.467 V9.3.0: UTRAN Architecture for 3G Home Node B[EB/OL]. (2010- 06- 10). http://www.3gpp.org/ftp/specs/archive/25_series/25.467/25467-930.zip.

[3] CHEN J, RAUBE P, SINGH D, et al. Femtocells-Architecture & Network Aspects[EB/OL]. (2010-01-28). http://www.qualcomm.com/common/documents/white_papers/Femto_Overview_Rev_C.pdf.

[4] 马志锋. Femto 技术及网络部署探讨[EB/OL]. (2009-12-30). http://www.ptsn.net.cn/article_new/show_article.php?article_id=technic_aa4870cc-1f60-a075-f32a-4b3aba384abd.

Abstract: In order to improve stability of environment monitoring system and save system's resource, the paper proposed a design scheme of environment monitoring system composed by lower computer, transmission network and upper computer. It introduced design of hardware and software of the lower computer in the system in details. The hardware of the lower computer takes ARM9 processor S3C2410 as core, its software uses multi-thread application program to process multi-task at the same time and uses semaphore and mutex to realize synchronization among threads. The actual application showed that the system runs stably and improves system efficiency.

Key words: environment monitoring, lower computer, multi-thread, synchronization, S3C2410, Linux

0 引言

随着国民经济及工业技术的发展,环境保护越来越受到重视。现在市场上已经出现了多种环境监控系统,但性能不稳定。现场监控终端大多采用工控机或单片机,前者抗干扰性能好,但成本较高;后者处理能力低,人机界面不友好,不利于现场人员的监控管理。针对上述问题,笔者设计了一种基于多线程的环境监控系统[1]。该系统采用多线程技术有效地实现了监控过程中数据的采集与存储、实时数据显示、下位机(监控终端)与上位机(监控中心)的通信、实时报警等功能。本文重点介绍该系统下位机的设计。

1 系统总体结构

基于多线程的环境监控系统由现场监控终端(下位机)、传输网络、监控中心(上位机)3 个部分组成[2],其结构如图 1 所示。

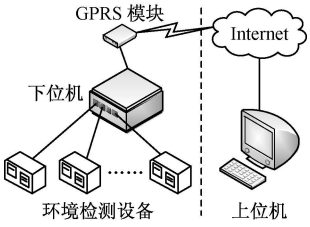

图 1 基于多线程的环境监控系统结构

下位机是一个基于 ARM9 的嵌入式系统,用于定时采集、处理、存储被监测的特征数据。经过下位机处理后的数据,按照相关协议,经 GPRS 模块发送给上位机[3]。上位机由一台 PC 机担任,负责接收多个下位机发送的数据,并对这些数据进行分析、处理和显示。环保部门可通过上位机监控其辖区内的污染排放状况。上位机基于 VB. NET 开发。

2 下位机硬件设计

下位机的硬件核心部分由 S3C2410、Nand FLASH 和 SDRAM 组成[4],如图 2 所示。S3C2410 是三星公司生产的一款基于 ARM920T 内核的 32 位RISC 嵌入式微处理器,带有独立的 16 KB 指令 Cache 和 16 KB 数据 Cache、LCD 控制器、RAM 控制器、Nand FLASH 控制器、并行 I/O 口、8 路 10 位 ADC,其运行频率可达 203 MHz。8 位64 MB 的 Nand FLASH 选用的芯片为 K9F1208,64 MB的 SDRAM 由 2 片 HY57V561620 组成。下位机通过以太网控制器 CS8900A 扩展了一个网口,数据既可以通过无线传输,也可以通过有线传输;通过 I/O 接口扩展了 8 个 DI 口(数字量输入)、4 个 AI 口(模拟量输入)、4 个 DO 口(数字量输出),下位机通过这些接口与被监控设备通信。

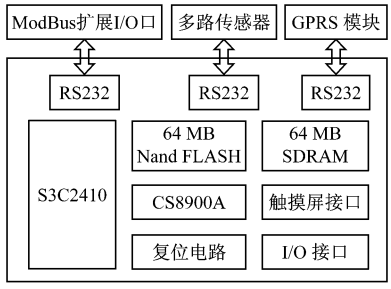

图 2 下位机硬件组成

3 下位机的需求与结构设计

下位机定时采集、处理现场数据,并存储在数据库中,把实时数据发送给上位机,并响应上位机发送的控制命令。因此,下位机需要同时处理多个任务,这些任务并发执行。若使用单线程来完成这些任务,则需要使用多个定时器来触发,而过多的定时器会导致系统不稳定。Windows 是抢先式多任务的操作系统,启动了一个应用程序就等于启动了一个进程。一个进程通常拥有一个线程,在系统资源管理中,每一个线程被分配一定的时间片。采用多线程的设计方法可以使程序拥有多个线程,这样程序就能同时处理更多的任务[5]。因此,若使用多个进程来协作完成,能避免上述缺点且系统比较稳定,但系统对进程的频繁调度会占用过多资源,程序的可

读性也不好。

笔者采用一种并行的、多线程方案[6] 能够很好地处理多个任务, 并充分节约系统资源。该方案中, 下位机有 5 个线程: GUI 线程、复位线程、数据采集与存储线程、网络通信线程、决策线程。其中 GUI 线程为主线程, 负责界面处理、系统数据的初始化以及创建子线程等任务; 复位线程、数据采集与存储线程、网络通信线程是后台的工作线程, 通过优先级调度、线程同步等机制保证能可靠执行现场数据采集、存储、发送、显示等任务。复位线程在启动后循环地对看门狗操作, 不作为任务处理线程。任务线程之间的关系如图 3 所示。

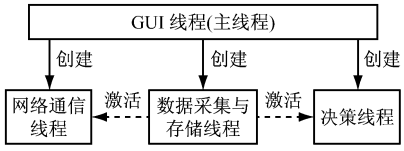

图 3　任务线程之间的关系

4　多线程技术在系统中的应用

4.1　线程的创建

Linux 环境下, 使用 pthread_create( ) 函数创建一个新线程, 默认情况下主线程会等待被创建的子线程执行结束, 得到子线程的返回结果然后再继续往后执行。实时监控程序的子线程都是循环执行的, 不需要运行结束后归并到主线程中, 需设置其属性为 PTHREAD_CREATE_DETACHED。根据子线程的重要性进行优先级设置, 确保重要线程优先执行。子线程的优先级从高到低依次为复位线程、数据采集与存储线程、决策线程、网络通信线程。线程的创建、设置伪代码如下[7]:

```
void * thread_watchdog(void * arg); //复位线程函数
void * thread_collection(void * arg); //数据采集与存储线程函数
void * thread_communication(void * arg); //网络通信线程函数
void * thread_decise(void * arg); //决策线程函数
int data[12]; //数据缓冲区, 用于存放线程间共享的数据函数
main( )
{
  //初始化工作
  …
  pthread_t watchdog; //线程号
  pthread_t collection;
  pthread_t communication;
  pthread_t decise;
  pthread_attr_init( ); //初始化线程属性
  pthread_attr_setdetachstate( ); //不对线程进行重新归并
  pthread_attr_setschedparam( ); //设置线程的优先级
  sem_init( ); //对相关信号量进行初始化
  pthread_create( ); //创建新线程
  //启动 GUI 程序
  …
}
```

4.2　线程的同步机制

同步机制是否合理是多线程应用程序运行是否稳定的关键。在程序设计时, 需考虑到可能引起数据毁坏的多线程数据访问冲突以及如何使用同步技术避免这种冲突。Linux 操作系统实现同步机制的方法有信号量(semaphore)和互斥量(mutex), 这两种方法相似, 但各有侧重。信号量侧重于一个线程被另一个线程激活, 常有先后执行的关系。而互斥量则保护某一共享内存任一时刻只有一个线程访问。网络通信线程和数据采集与存储线程之间的同步通过信号量来实现。

为了防止系统资源泄漏, 保持各个线程的同步, 主线程需要初始化数据采集驱动代码, 为数据采集做好准备; 申请相应的内存空间, 用于存放采集到的实时数据; 定义好各个信号量和互斥量。

4.3　线程的实现方法

数据采集与存储线程是获取数据的起始线程, 由 GUI 线程创建, 网络通信线程和决策线程是由数据采集与存储线程激活。下位机开始运行后, 数据采集与存储线程启动, 每隔 5 s 运行 1 次, 读 DI、AI 接口的状态, 并把这些状态和此刻的时间存入 SQLite 数据库中。数据采集与存储线程每运行一次, 对信号量 sem_decise 和 sem_com 进行一次 post 操作, 分别激活决策线程和网络通信线程。数据采集与存储线程的同步流程如图 4 所示。

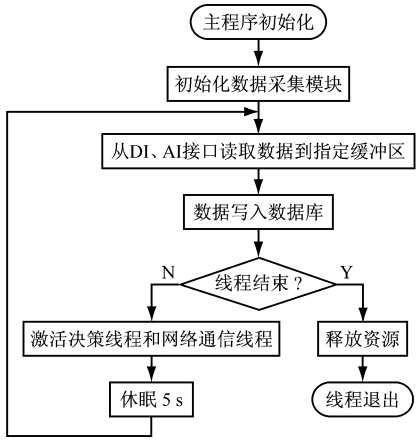

图 4　数据采集与存储线程的同步流程

数据采集与存储线程作为系统的数据源头, 它激活了其它 2 个子线程, 与之相对应, 被激活的子线程随着它的结束而结束[8]。线程在未接受到信号量激活时处于阻塞状态, 不占用系统资源。前台处理

用户界面的 GUI 线程与后台的工作线程之间是独立的。GUI 线程提供友好的人机界面,它把被监控对象的信息实时显示在图形界面上,供现场工作人员查询和设置。

监控程序的每个线程都需要对存放被监控对象实时状态的数据缓冲区进行访问。由于 Linux 操作系统允许多个线程同时对某一数据缓冲区进行读操作,但在同一时刻对该数据缓冲区只能有一个写操作。GUI 线程需要定时刷屏,更新被监控对象的实时状态,因而需要定时对缓冲区进行读操作,而数据采集与存储线程定时地对该缓冲区进行写操作,它们之间没有触发关系,是相互独立运行的。因此,需要对缓冲区设置一个互斥量,确保任一时刻这两个线程只有一个能对其进行访问[6]。

数据采集与存储线程对缓冲区进行写操作之前,先对互斥量进行加锁操作,把实时状态写入数据缓冲区后,再进行解锁。这样避免了因与 GUI 线程争夺资源而造成系统不稳定的现象。上锁与解锁操作代码如下:

```
void * thread_collection(void * arg)
{
  ...
  pthread_mutex_lock(&data_mutex); //上锁操作
  read(fd, &data, sizeof(data)); //写实时状态到 data 缓冲区
  pthread_mutex_unlock(&data_mutex); //解锁操作
  //激活网络通信线程、决策线程,并写数据到数据库
  ...
}
```

同样,GUI 线程中也需要对缓冲区进行相应的上锁、解锁操作。

网络通信线程和决策线程由 GUI 线程创建,由数据采集与存储线程激活,都是每 5 s 运行 1 次。由于 GPRS 模块通过串口与下位机相连,并采用透明传输模式,即有数据即传输,因此,网络通信线程只需要定时对串口写操作就可以完成数据传输任务。网络通信线程先打开串口,设置串口的波特率、数据位、校验位等属性,然后等待数据采集与存储线程的信号量将其激活,第一次被激活后,进入了一个 while 循环,执行一次串口写操作,再等待下一次被激活。网络通信线程被激活的条件是数据采集与存储线程对信号进行了一次加 1 操作,即 sem_post(&sem_2)。网络通信线程的关键代码如下:

```
fd= open("/dev/ttySC4", O_RDWR);
set_speed(fd, 9600); //设置波特率
set_parity(fd, 8, 1, 'n'); //设置数据位、奇偶校验位
sem_wait(&sem_2); //等待信号量激活
while(1){
  write(fd, buf, sizeof(buf)); //对串口写操作,发送数据
  sem_wait(&sem_2); //等待下一次信号量激活
  }
pthread_exit("thread exit\n"); //线程退出
```

决策线程的任务是对当前被监控对象的状态进行判断,如果有异常发生,则产生一个报警信号,并执行相关动作来应对这些异常。其代码结构与网络通信线程相似。

5 结语

基于多线程的环境监控系统采用多线程技术完成下位机的多个任务,相对于单任务应用程序,多线程应用程序能够减少定时器的使用,节约系统资源,从而提高系统效率。而且,多线程应用程序更能体现模块化设计思想,程序易于维护和修改。该系统已经成功应用于多个项目之中,性能稳定可靠。

参考文献:

[1] 陈海明. 基于 ARM 和 GPRS 远程无线监控系统的设计与实现[D]. 北京:北京邮电大学,2009.

[2] 刘坚,陶正苏,陈德富,等. 基于 GPRS 的环境监测系统的设计[J]. 自动化仪表,2009,30(2):30-32.

[3] XING Jianping, ZHANG Jun. GPS Real Time Vehicle Alarm Monitoring System Based on GPRS/CSD Using the Embedded System[C]//ITS Telecommunications Proceedings, 2006.

[4] 刘森,慕春棣,赵明国. 基于 ARM 嵌入式的拟人机器人控制器的设计[J]. 清华大学学报:自然科学版,2008,44(4):482-486.

[5] 王苓,苏维均. 基于多线程技术的多串口通信[J]. 微计算机信息,2006(7):261-263.

[6] 景征骏,周军. 基于多线程的集控式足球机器人上位机系统[J]. 计算机工程,2008,34(21):199-203.

[7] MATTHEW N. Linux 程序设计[M]. 2 版. 北京:机械工业出版社,2002:445-447.

[8] 张静永,张蕾,曹其新,等. 基于多线程的钢管智能探伤系统[J]. 计算机工程与应用,2004(31):210-212.