嵌入式系统多线程设计与实现

Design and Realization the Embedded System with Multi-thread

解放军理工大学 石婷婷 刘广
江苏智运公司 谢道新
Shi Tingting Liu Guang Xie Daoxin

摘 要：多任务模式在系统设计中具有重要意义，本文主要讨论了在嵌入式Linux平台下，基于LonWorks现场总线网络的智能控制器如何进行系统多线程设计和实现。

关键词：多线程 嵌入式系统 Linux 嵌入式Linux

Abstract: In system design, multi-task model has importance function, the paper mainly discussed how to design and realize the system with multi-thread.

Key words: Multi-thread Embedded system Linux Embedded Linux

[中图分类号] TP334 [文献标识码] B 文章编号 1606 5123(2006)03 0089 03

1 引言

在基于嵌入式Linux的智能控制器中，一般要有多个任务能够同时运行。主任务用来接收界面的控制事件，并为上位监控机提供Web服务；数据采集任务用来采集LonWorks等现场总线网络各节点的数据；控制逻辑执行任务是根据用户的逻辑组态要求进行控制逻辑运算。

基于嵌入式Linux的智能控制器通常是以多任务模式运行的。如果采用单任务的运行模式，则所有的处理都是串行的，必须等待上一个操作结束后，才能执行新的请求，例如，基于嵌入式Linux智能控制器处于与上位机进行Web通信时，就不能接收和处理其它请求(如数据采集或控制逻辑执行)进行LonWorks现场总线网络的数据采集了。单任务的模式缺少灵活性，难以满足智能控制器的实时性要求，所以整个系统需要以多任务模式运行。

2 多任务模型的实现方法

在传统的UNIX模型中，多任务模型是通过下面的方式实现的：当一个进程需要另一个实体执行某件事情时，该进程派生(fork)一个子进程，让子进程去处理。

在嵌入式系统中，不适合用fork方式来实现，因为fork需要为子进程创建与父进程相同的进程空间，这需要耗费大量的系统资源和CPU时间，因为fork子进程后，需要用进程间通信(IPC)在父子进程间传递信息，浪费系统资源。

Linux下面线程的实现是Posix线程，又称作Pthreads。Pthreads在1995年作为Posix.1c标准的一部分得到了标准化，大多数UNIX版本都支持它们。

3 多线程的设计与实现

整个控制器系统主要由3个线程组成：主线程、硬件控制线程(数据采集、控制逻辑执行)、网络服务线程。线程结构如图1所示。

主线程控制基于嵌入式Linux智能控制器的启动和初始化等。主线程首先创建网络服务线程，网络线程用来实现系统的“网络模块”的功能；然后主线程创建硬件控制线程，硬件控制线程用来控制所有对硬件的操

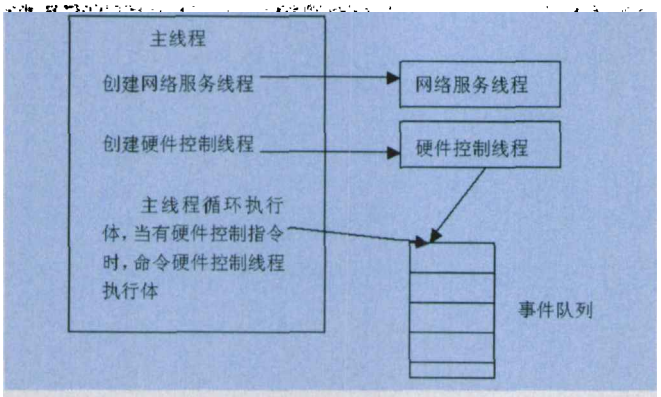

图1 控制器软件系统线程结构图

作(包括对从LonWorks现场总线网络各节点采集数据以及执行控制逻辑)；最后，主线程进入主循环，接收来自上位监控机的操作指令(控制信息)，并对指令进行分析和执行：如果是硬件控制类指令，就把指令写入“事件队列”中，并通知硬件控制线程执行；如果是Web服务类指令，就调用相应的模块来执行。

硬件控制线程的主体示意如图2所示。首先进行初始化(包括对与主线程共享的事件队列的初始化)。运行过程主要是判断当前的事件队列中是否有事件需要处理，如果有则继续处理，直到所有的事件处理完成；如果没有事件，队列为空，则程序转入睡眠，直到主线程下达硬件控制指令到事件队列并且唤醒硬件控制线程的执行。

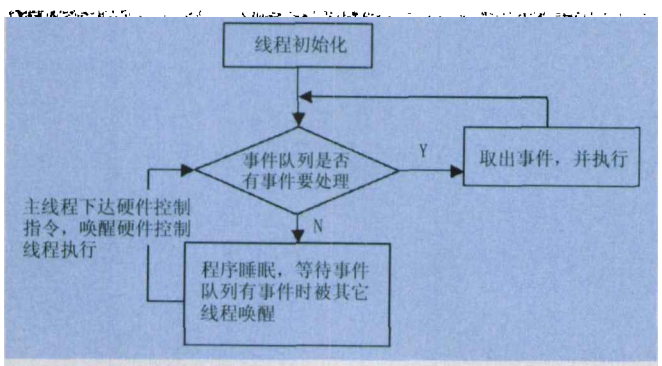

图2 硬件控制线程结构图

主线程和硬件控制线程通过共享的事件队列进行通信。这里涉及到并发编程(concurrent programming)中数据的一致性问题。由于两个线程是并发运行的，如果不加以限制，不能够在运行时间内保证哪个线程先操作数据队列，不能保证只有单一线程对数据进行操作。所以，在实现中用互斥锁对共享的数据进行保护。互斥锁是能够提供线程间互斥的一种资源，如果一个线程对互斥锁执行了成功的上锁操作后，其它的线程将不能再对其执行上锁操作，这样线程上锁后，可以进行一些需要互斥的操作；当操作完成后，再将互斥锁解锁，允许其它线程执行互斥操作。

互斥锁的控制函数主要包括：

```
#include <pthread.h>
int pthread_mutex_init (…)
int pthread_mutex_lock(…);
int pthread_mutex_unlock(…);
```

pthread_mutex_init函数用于对互斥锁进行初始化，初始化的目的关键是用于指定互斥锁初始的状态。通常，互斥锁在初始是解锁状态。Pthread_mutex_lock函数试图将互斥锁上锁，如果其它线程已经对互斥锁上锁，则调用pthread_mutex_lock的线程将不能再对其上锁，这个线程将被阻塞等待，直到上锁的线程将锁打开后，线程才能对锁进行上锁操作。Pthread_mutex_unlock函数将上锁后的互斥锁解锁。

硬件控制线程要时刻知道当前的事件队列中是否有事件需要处理。其中的一种实现就是进行互斥锁保护下的轮询(polling)操作，及程序不停地检查当前的事件队列中是否有事件需要处理。轮询方式会严重浪费CPU的资源，需要一种方法使得主循环进入睡眠，直到有一个线程通知它某件事已经就绪。Pthreads库提供的条件变量(condition variable)加上互斥锁可以提供这种功能。互斥锁提供互斥机制，条件变量提供信号机制。

条件变量是线程间实现同步的重要方式，图3显示了主线程和硬件控制线程同步的程序流程图。

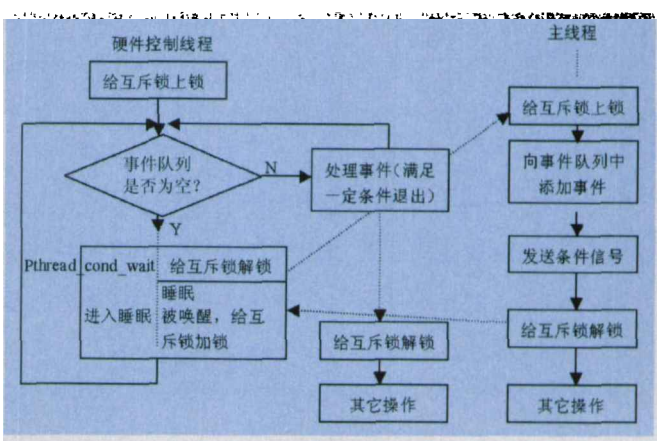

图3 主线程和硬件控制线程同步的程序流程图

图3显示了利用Pthread线程库提供的互斥锁和条件变量来同步主线程和硬件控制线程的工作流程。下面从硬件控制线程的运行角度，描述上图的流程。首先，硬件控制线程给互斥锁上锁；然后，判断事件队列中是否有事件需要处理，如果有，则相应的处理事件，

如果没有，则进入睡眠；转入睡眠是由函数pthread_cond_wait来实现的，函数的执行分成两部分：首先给互斥锁解锁，进入睡眠；然后被其它线程唤醒(通过pthread_cond_signal函数)，此时重新对互斥锁上锁，上锁成功后，pthread_cond_ wait执行完成。硬件控制线程是被主线程唤醒的，当主线程接收到用户对硬件操作的指令后，就会向共享的事件队列中写入命令请求，并唤醒硬件控制线程执行。

综上所述，选择多线程实现多任务，以及在实现中利用了互斥锁和条件变量来控制整个执行过程，有很多优点。最重要的是，节省了嵌入式系统的宝贵系统资源，这是利用了线程“轻”的优点和条件变量可以避免轮询的特点。虽然是多线程的系统，但是从某一个时间上来看，系统可能只有一个线程正在运行，而其它线程都处于空闲(Idle)状态。其次，系统的结构清晰，线索清楚，便于功能扩展和代码维护。

多线程运行的一个界面如图4所示。图4中实时显示LonWorks现场总线网络1#通道的6号前端(LM1202开关量模块)I/O状态，此时控制逻辑执行线程正在进行逻辑运算操作，数据采集线程和网络服务线程都在运行。在上位监控机监控的Web页面上可以看到网络服务线程传送过来的数据采集线程实时采集的LonWorks现场总线网络前端的数据，一旦控制条件发生变化，上位监控机即可及时看到控制逻辑执行的输出结果。

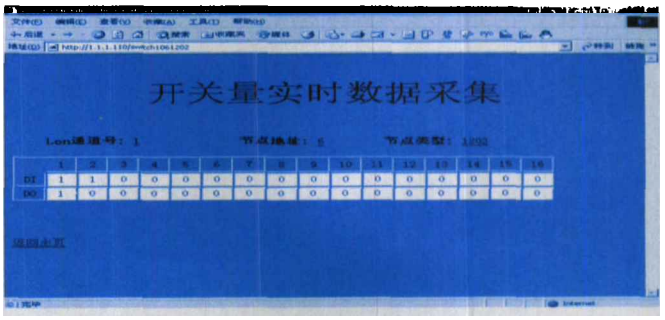

图4 智能控制器多线程运行示意图

4 结束语

本文以基于嵌入式Linux智能控制器为例，讨论了在嵌入式Linux系统下实现系统多线程设计原理。系统实际运行结果表明，很好的实现了设计目的。该设计方法可广泛应用于嵌入式系统中，对提高系统得运行效率，扩展设备功能具有重要意义。

参考文献

[1] 刘 广. 基于嵌入式Linux的智能控制器研究. 解放军理工大学硕士论文. 2004.4

[2] 郭玉东. LINUX操作系统结构分析. 西安:西安电子科技大学出版社,2002

[3] 王智民等. Red Hat Linux 7.0实战技术. 北京:国防工业出版社,2001

[4] 阳宪惠. 现场总线技术及其应用. 北京:清华大学出版社,1999

作者简介

石婷婷(1983-) 女 学士 主要工作领域:计算机应用、数据交换。

(上接第99页)

OS-II目标平台上的移植过程，实际表明在具有图形界面的嵌入式系统的开发中，采用基于μC/OS-II的图形系统μc/GUI，移植简便、使用方便灵活，目前基于μC/OS-II的μc/GUI系统已成功移植到了系统效率测试仪上，系统的电能参数(电流、电压、功率)可以实时采集和实时波形显示；参数设置以菜单、编辑框的形式实现，界面的切换以窗口的形式实现。充分利用了μc/GUI强大的图形功能，使人机界面更加丰富、友好。实践表明系统具有良好的实时性和稳定性。

参考文献

[1] Micrium Technologies Corporation. μc/GUI: Graphical user interface with graphic library (Version 3.26)[EB/OL].http://www.μCOS-II.com, 2005.3

[2] Jean J.Labrosse著. 嵌入式实时操作系统μC/OS-II,邵贝贝等译. 北京:北京航空航天大学出版社,2003

[3] 王田苗. 嵌入式系统设计与实例开发. 北京:清华大学出版社,2003

[4] 马忠梅. ARM嵌入式处理器结构与应用基础. 北京:北京航空航天大学出版社,2002

作者简介

陈 进 男 副教授 从事嵌入式系统设计与低温工程研究。