vxWorks 下基于缓冲队列的全双工网络通讯

程敬原　宋克柱　安　琪
(中国科学技术大学近代物理系快电子学实验室,合肥 230027)
E-mail:chengjy001@163.com

摘　要　文章针对半自动大型仪器使用半双工网络通讯造成的效率下降问题,建立了在嵌入式实时操作系统 vxWorks 下全双工网络通讯的模式,并特别提出了双缓冲队列的流水线处理方法和通讯死时间的概念。该方法充分利用 vxWorks 对多任务和网络的良好支持,做到了在操作人员层面上的软件零死时间,在改善系统响应特性,提高软件效率上都很有帮助。因此比半双工通讯方式更加适用于半自动控制。

关键词　缓冲队列　全双工　网络通讯　vxWorks

文章编号 1002-8331-(2004)11-0118-03　**文献标识码** A　**中图分类号** TP393

Full Duplex Network Communication Under VxWorks Based on Buffering Queues

Cheng Jingyuan　Song Kezhu　An Qi
(Fast Electronics Laboratory, Department of Modern Physics, USTC, Hefei 230027)

Abstract: Generally, half duplex net communication method decreases the efficiency of semiautomatic devices. To solve this problem, a mode of full duplex is developed, under the embedded RealTime Operating System vxWorks. Specially, the method of pipeline based on multi-buffering queues and the concept of communication dead-time are introduced. This method makes the best of vxWorks's support on multi-tasks and network programming. It achieves near zero software dead-time on operational level, improves the response characteristic, and increases software efficiency. Comparing with half duplex mode, it is more applicable to semiautomatic control.

Keywords: Buffering queue, Full-duplex, Network communication, vxWorks

随着高位嵌入式处理器的日益普及,提供用户操作界面的半自动大型仪器得到了迅猛的发展,并普遍采取了 PC 或工作站作为控制端,嵌入式单板作为受控端的形式。vxWorks 作为嵌入式行业应用得最为广泛的实时操作系统,对多任务和网络 socket 通讯都提供了良好的支持,使用基于缓冲队列的全双工通讯模式,可以在最大程度上发挥控制端和受控端的处理能力,更好地完成大型系统的异地控制。

1　vxWorks 的多任务支持和网络支持

1.1　vxWorks 简介

vxWorks 是由 Wind River 公司开发的一种强实时性嵌入操作系统[1],支持 Motorola PowerPC,ARM 等多种嵌入式 CPU。Wind River 同时提供集成开发环境 Tornado,用户可以通过图形界面方便地对 vxWorks 组件进行添加和裁减。用户程序编制使用标准 C,也可以选择 C++支持。Tornado 还提供动态下载、远程源级调试器、目标和工具管理、系统目标跟踪、内存使用分析和自动配置,非常适合于交互式开发。

1.2　vxWorks 的多任务支持

vxWorks 全面支持多任务[2],任务由唯一的 ID 来标识,并且对应于某一个特定的任务名,用户可以通过对任务 ID 或任务名的操作实现任务的发起、挂起、释放和删除等操作。

任务状态包括:正在执行(Exec)、准备好(Ready)、阻塞(Pended)以及延时(Delayed)等。当前占用 CPU 的任务的状态被定义为正在执行;排队等待获得 CPU 的任务的状态定义为准备好;当任务试图获取某个暂时不能得到的资源时,将进入阻塞状态等待资源被别的任务产生或释放,获得资源后转入准备好或正在执行状态;任务等待特定时间过去后再重新运行的状态定义为延时状态。

已经准备好的任务将等待操作系统分配 CPU,默认使用基于优先级的轮转调度方式。操作系统支持 256 个优先级,在默认情况下,0 为最高优先级,255 为最低优先级,高优先级的任务将优先获得 CPU;优先级相同的任务将采取轮转调度,即获得 CPU 的任务将一直执行直到任务结束或者被阻塞或延时。为了保证系统的正常运行,建议用户编写程序使用的任务优先级最好不要超过 100。

vxWorks 提供信号量、消息队列等机制用以实现任务间的通讯与同步。信号量支持二进制、互斥和计数型三种,使用 semGive()和 semTake()实现释放和获取,该文涉及的是用于同步的计数型信号量;消息队列使用 msgQSend()和 msgQReceive()实现消息的发送和接收。

信号量的获取和消息队列的消息接收在资源得不到满足时,将使调用它们的任务被阻塞,从而释放 CPU 使用权。任务

基金项目:国家 863 高技术研究发展计划(编号:2001AA602011-1)
作者简介:程敬原,女,硕士研究生,研究方向:嵌入式操作系统及软件开发,高速信号采集与处理。宋克柱,男,讲师,研究方向:系统设计、高速信号采集与处理。安琪,男,教授,博士生导师,研究方向:高速信号采集与处理、高速通讯。

主动调用 taskDelay ()将使任务进入延时状态,也会释放 CPU。

1.3 vxWorks 的网络支持

vxWorks 支持常见的几种网卡和网口芯片[3]:如 NE2000 和 Intel 82557 等,只需要定义几个宏,系统就会在启动时自动加载低层驱动;此外 Wind River 公司还提供了通用模板,只需要做适当的改动,就可以使系统支持其它的网口芯片。

vxWorks 完全支持 BSD socket[4],可以使用 TCP 和 UDP 两种通讯形式。信息发送和接收的函数接口分别为 send ()和 recv ()。

网络的信息接收在接收不到的时候也同样会引起任务的阻塞。

2 基于缓冲队列的全双工网络通讯模式

2.1 异步全双工网络通讯结构

通讯前必须先建立连接,通常受控端作为通讯的服务器端,控制端建立客户端后发起连接,连接建立之后,就可以开始通讯了,通讯的基本模式如图 1 所示(由于主控端和受控端软件结构类似,该文主要介绍受控端的软件结构)。为了方便起见,将在网络上传输的按照用户通讯协议打包的一个命令或一段数据称为一个信息。

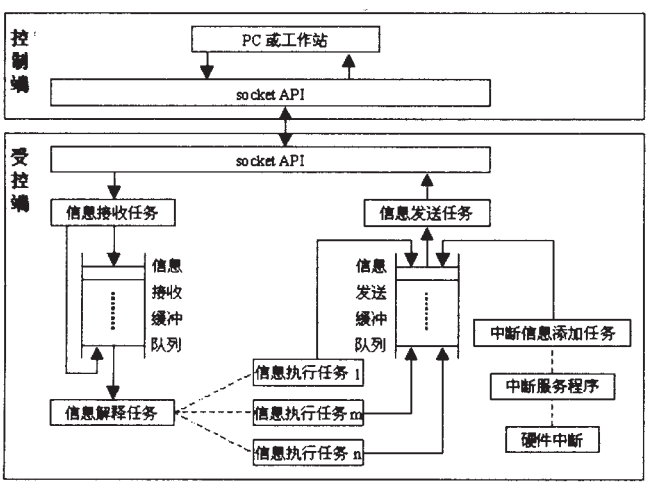

图 1 全双工网络通讯基本模式

受控端运行的任务分为两种,一种是循环运行的任务,包括信息接收任务、信息解释任务、信息发送任务和中断信息添加任务;另一种是只在特定时间运行的任务,即根据需要发起的信息执行任务。

由控制端发起的通讯过程为:

控制端:接收用户操作信息,将其转换为受控端可以理解的命令格式,通过 socket 接口下传到受控端,然后等待返回信息;

受控端:信息接收任务从 socket 接口接收命令,将其添加到信息接收缓冲队列;信息解释任务从信息接收缓冲队列取出信息,根据具体情况发起特定的信息执行任务,或者将信息传送给已经被发起的特定任务,供其进一步执行使用;信息执行任务进行具体的硬件操作,操作结束后将执行结果添加到信息发送缓冲队列;信息发送任务从信息发送缓冲队列取出信息,通过 socket 接口将其返回给控制端。

控制端:接收执行结果后返回给操作人员。

由受控端发起的通讯过程为:

受控端:硬件产生中断,操作系统立即执行已经被绑定的中断服务程序,中断服务程序通知中断信息添加任务,后者将中断信息添加到信息发送缓冲队列;信息发送任务将此信息上传给控制端。

控制端:接收信息然后提供给操作人员,由操作人员选择下一步的操作方案。

2.2 受控端任务同步

受控端的软件采用多任务方式实现,必须保证各任务工作的协调性。任务间的同步采用了计数型信号量(semaphore)和消息队列(msgQueue)。任务间对信号量和消息队列的操作如图 2 所示。

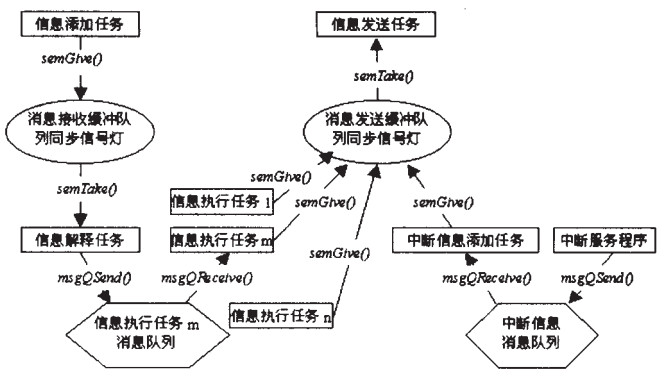

图 2 使用信号量和消息队列的任务间同步

软件为信息接收缓冲队列和信息发送缓冲队列各创建了一个计数型信号量,并将初始值设为 0。

对于信息接收缓冲队列同步信号量,信息接收任务调用 semGive ()将其加 1,信息解释任务调用 semTake ()将其减 1,并将获取时间设为 WAIT_FOREVER,信号量的值代表了队列中当前的信息数量,只有信息数量大于零时,semTake ()才会返回,使信息解释任务有可能被执行。

对于信息发送队列同步信号量,信息执行任务和中断服务程序都可以使其加 1,信息发送任务使其减 1,它的值同样代表队列中的信息数量,信息发送任务也必须在信息数量大于零的时候才有可能执行。

某些信息执行任务需要控制端进一步信息,这里使用消息队列机制实现。当任务 m 运行到需要控制端更多信息处时,调用 msgQReceive ()等待信息执行任务 m 消息队列中的新消息,并将等待时间设置为 WAIT_FOREVER;只有当消息由控制端发出,然后经过信息解释任务处理,调用 msgQSend ()发送到消息队列中后,信息执行任务的 msgQReceive ()才会返回,使任务继续执行。

为了保证中断消息的发送不影响整个系统的调度,同样使用消息队列将信息发送给发送中断信息添加任务,后者将信息向信息发送缓冲队列中添加。

需要说明的是:信息发送任务在发送一条信息后,将主动调用 taskDelay ()从而进入延时状态,在延时时间内,即使信息发送缓冲队列中有了新的信息,也要等到延时时间过去后才发送,这样做的原因是为了减小控制端的死时间,参见 3.1 节。

2.3 共享内存操作

由于信息接收缓冲队列和信息发送缓冲队列都被不止一个的任务操作,因此必须考虑共享内存保护。为了保证共享内

存操作的正确性,将任务划分为不同的优先级。vxWorks 采用基于优先级的调度,高优先级任务执行时,将获得 CPU,并保证其对共享内存的独占权,在其结束前别的任务不可能操作共享内存,因此可以保证操作的正确性。唯一例外的是中断服务程序,它在硬件中断发生时立即执行,相当于最高优先级,会无条件打断其他任务的运行,因此不能在中断服务程序中对共享内存进行操作。实际采用的方法是让中断服务程序使用消息队列通知中断信息添加任务,后者参与正常的任务调度,在适当的条件下获得 CPU 并添加信息到共享内存。当所有任务都被阻塞时,CPU 将处于空闲状态。受控端任务的优先级设置和共享内存获得条件如表 1 所示。

表 1　任务优先级设置及 CPU 占用条件表

任务	编号	优先级	CPU 获取条件	CPU 释放条件
信息接收任务	1	最高	主控有下行命令	一次循环结束,再次接收信息接收不到
中断信息添加任务	2	次高	1 被阻塞并接收到由中断服务程序向消息队列发送的消息	一次循环结束,再次接收消息接收不到
信息解释任务	3	中等	1、2 被阻塞且获取信息接收缓冲队列同步信号灯成功	一次循环结束,再次获取信号灯获取不到
信息执行任务	4	较低	1、2、3 被阻塞,需要进一步信息的任务获取消息成功	任务运行结束或获取消息队列中消息获取不到
信息发送任务	5	最低	1、2、3 被阻塞,没有 4 在运行,且延时的时间结束,再次获取信息发送缓冲队列同步信号灯成功	一次循环结束主动调用 taskDelay 进入延时状态;或者再次获取信号灯获取不到

3　基于缓冲队列的全双工网络模式的特点

3.1　通讯死时间的讨论

在通讯过程中,如果信息到达时间相距太短,就有可能引起信息处理不及时或信息丢失。定义通讯死时间为 在保证受控端正确接收前提下,由用户操作引起的两次信息发送可以间隔的最短时间。下面将讨论缓冲队列的使用对系统死时间的影响。

()双方都没有缓冲队列 t (,0)

通讯死时间为从控制端发出消息、受控端接收信息、分析信息、执行信息、信息返回到控制端接收信息并通知操作人员的全部时间。

()有信息接收缓冲对列 t (,1)

通讯死时间为接收端将信息添加到信息接收缓冲队列中所需的时间,一般来说受控端的嵌入式 CPU 频率低于控制端,因此死时间定为受控端添加消息的时间。

()双方都有信息发送缓冲队列和信息接收缓冲队列 t (,1)

将信息发送时间间隔设为 t (,1),则可以保证正确接收。实际上由于信息接收任务对通讯死时间内到达的第一条信息仍然可以接收,只是接收时间被推后到 t (,1)时间结束后,所以在理论上将控制端的信息发送时间间隔设为 t (,1)/2,系统仍然可以正常工作。但是考虑到网络传输时间的晃动,最好仍然设为 t (,1)。

受控端时间间隔采用消息发送任务主动调用 taskDelay 方法实现,主控端则根据软件具体情况而定,例如对于 Window 系统下 Vc 编制的程序,可以使用 sleep 实现。对于发送方来说,不必再考虑接收方死时间,信息产生后立即将其填入消息发送队列,并且在填入过程中仍然响应其它信息的产生,相当于在操作人员层面上实现了零通讯死时间。

需要说明的是,受控端中断的发生也是随机的,因此在半双工通讯中由中断产生的信息同样可能被丢失,双缓冲队列的使用使控制端可以正确接收所有的中断信息,原因同上。

3.2　零通讯死时间带来的优势

使用缓冲队列实现了信息的流水线处理,降低了信息通讯双方时间上的关联性,带来了操作人员层面的系统零死时间,同时也带来了其它的好处:

()提高 CPU 利用率

零死时间使全双工成为可能,理论上不存在因为等待返回而产生的 CPU 空闲且不响应时间,CPU 可以在任何时间被利用,客观上提高了 CPU 的利用率和软件的效率。

()信息解随机

由于受控端的信息发出时间主要由操作人员决定,因此两条信息的时间间隔在很大程度上是随机的,但受控端执行特定信息的时间是基本固定的。缓冲队列将随机发出的信息排队,将信息的时间特性转换为其在队列中的位置特性,受控端软件不需要知道信息发出的时间,只需要知道队列中是否仍然有信息没有被执行,相当于解除了随机性。这使得控制端信息的批处理发送成为可能,例常性工作的操作时间可以大大缩短,因此更加适用于半自动控制。

()紧急程度决定执行顺序

控制端可以通过设定信息的格式将信息紧急程度传递给受控端,紧急信息将被添加到信息接收队列队头,因此被优先执行;紧急信息的执行结果也会被添加的信息发送队列队头,被优先发送。同时中断信息也会被优先发送。这种根据紧急程度决定执行顺序的模式更符合逻辑,更适用于半自动控制。

4　结论

该文提出的全双工网络通讯充分利用了 vxWorks 的多任务支持,采用信号量和消息队列保证任务间同步和通讯,利用任务优先级的高低保证共享内存保护。双缓冲队列的流水线工作方法,使网络通讯做到了用户层面零死时间,具有响应及时、软件效率高、更加适用于半自动控制等特点,并可以很容易地推广到其它嵌入式操作系统,具有广泛的应用前景。

收稿日期:2003 年 9 月)

参考文献

1.Tornado User's Guide (Windows Version)2.0.first edition,http://www.windriver.com/pdf/ win_guide.pdf

2.VxWorks Reference Manual Libraries.first edition,http://www.windriver.com/products/ html/manuals.html,2003

3.VxWorks Network Programmer's Guide 5.4.first edition,http://www.windriver.com/pdf/ vxworks_54_network_programer_guide.pdf,2003

4.FreeBSD Documentation Project.FreeBSD Developer's Handbook,http://www.freebsd.org/doc/en_US.ISO8859-1/books/developers-handbook/

5.孔祥营,柏桂枝.嵌入式实时操作系统 VxWorks 及其开发环境 Tornado[M].中国电力出版社,2002