一个实时多任务系统——进程管理及时钟管理系统的设计与实现*

闵 锐 庞丽萍
(华中理工大学计算机系 武汉 430074)

摘 要

本文提供了一套在DOS环境下实现进程管理及时钟管理系统的方法。通过该系统的演示,学生可以直观地看到多任务穿插执行的过程,对分时、多道、进程等概念有形象的了解。若能按照文中的步骤生成自己的系统,不但能巩固有关实时多任务系统的概念,而且对其动手能力也是较大的提高。

关键词 操作系统 多任务分时系统 进程管理 时钟管理系统

一 引 言

为了改善CPU的利用率,我们引入了多道系统,也就是在内存中存放多道相互独立的程序,这些程序在管理程序的统一控制之下,相互穿插地执行。当某程序因某些原因不能执行下去时,管理程序就将另一道程序投入运行,以便使CPU处在尽量忙碌的状态。其中关键的问题是用户提交的多个需求,远多于系统的资源,系统管理程序则必须将这些资源管理起来,遵循一定的策略将其分配给某个作业,如图1所示。

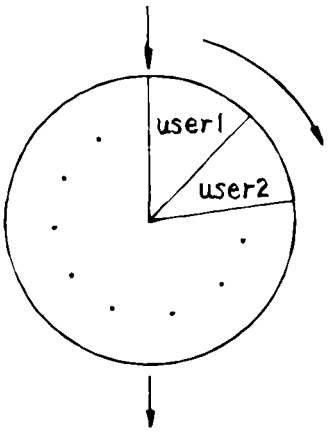

图1 多道分时系统原理

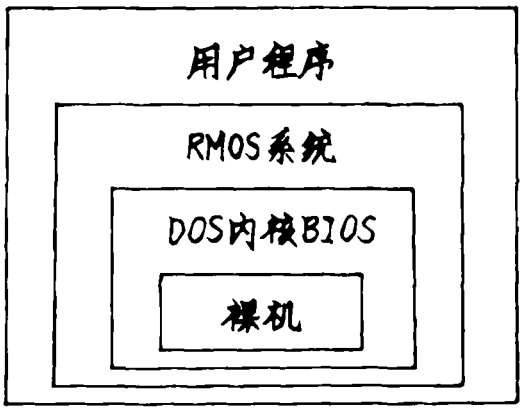

图2 RMOS与DOS的逻辑层次图

本文提出了一个进程管理及时钟管理系统的建造方法。这是一套在进程、分时等概念模式上设计的系统。该系统与硬件ROM、BIOS及用户程序(片)的逻辑层次如图2所示。

二 RMOS系统的实现步骤

下面我们给出一个形成该系统的基本步骤。

第一步 实现进程的概念,并在此基础上实现进程的创建、撤消和调度。

第二步 获取时钟中断,根据时间片十优先级的调度策略,实现多进程的轮流执行。

第三步 进程的互斥、进程的同步、信号灯操作的实现。

* 收到本文时间:1995年3月7日

?1994-2015 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

该方案具体的实现方法为：

1. 进程的体现

(1) 进程是一个程序与其数据一起通过处理机的执行所发生的活动，也就是静态的程序动态的执行过程。

RMOS系统首先初始化系统，其pcb(进程控制块)结构如图3所示。

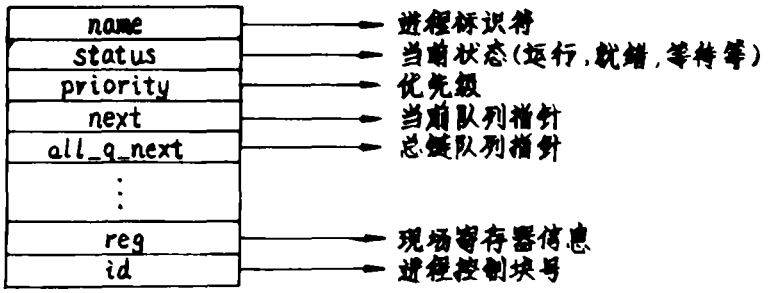

图3　pcb结构图

初始化的步骤依次为：

- 初始化各队列头(总链队列、就绪队列)；
- 创建0#进程，将其插入到总链队列；
- 将控制权交给0#进程。

0#进程的任务是：

- 初始化各用户进程，创造相应的pcb结构；
- 将用户进程依次插入到总链队列和就绪队列；
- 0#进程自身挂起，转入系统总控程序。当0#进程结束时，系统中止。

(2) 进程的调度和控制程序

该程序包括保护现场、处理运行指针、恢复现场三个层次。它们的具体内容分别是：

a. 将各寄存器值保存至当前运行的进程的pcb结构中reg数据项中，并用系统的堆栈重新设定当前堆栈；

b. 将当前运行进程送入就绪队列或0#进程专用的指针。对就绪队列作出判断，若已没有用户作业，则调用系统进程，结束系统；若仍有用户作业，则依照相应的调度算法，选取一用户作业进行处理；

c. 用选中进程的pcb的reg数据项内容设置各寄存器值，运行选中的进程。

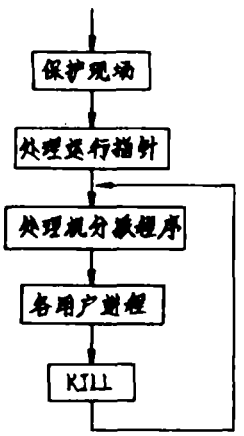

图4　系统控制流程图

为了把寄存器的内容保存到pcb结构的reg项中或用pcb的reg项的数据来恢复各个寄存器的值，我们还必须在内存中设置一工作区r作为中介，这是因为寄存器无法直接与reg项互换数据。r区与寄存器之间的数据交换以汇编语言编程，而r区与pcb结构的reg数据项是用c语言编程的。在保护现场时，由汇编语言调用c语言程序，反之，在恢复现场

?1994-2015 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net

时是由c语言调用汇编语言程序。

系统运行的流程图见图4。

(3) 用户程序的书写规则

书写用户程序时应特别注意,程序的结尾必须为撤消对应的进程,释放pcb结构的语句。

在实现了上述各程序后,此多道系统已经具备雏形,但系统运行时,表现上仍是多个程序依次执行,即还不能穿插执行。

2. 时间片轮转,多任务宏观上的并发执行。

(1) 分时技术指把处理机时间划分成很短的时间片,将时间片轮流地分给各个作业使用。在本系统中,为了实现分时的功能,我们应重新设定DOS的8号时钟中断。在新设的8号中断中应判断:

- 该进程是否为系统进程;
- 分配给该进程的时间片是否结束;
- 查询当前进程运行时的"在系统标志单元"值,判断能否打断该进程。

新8号中断的流程图见图5。

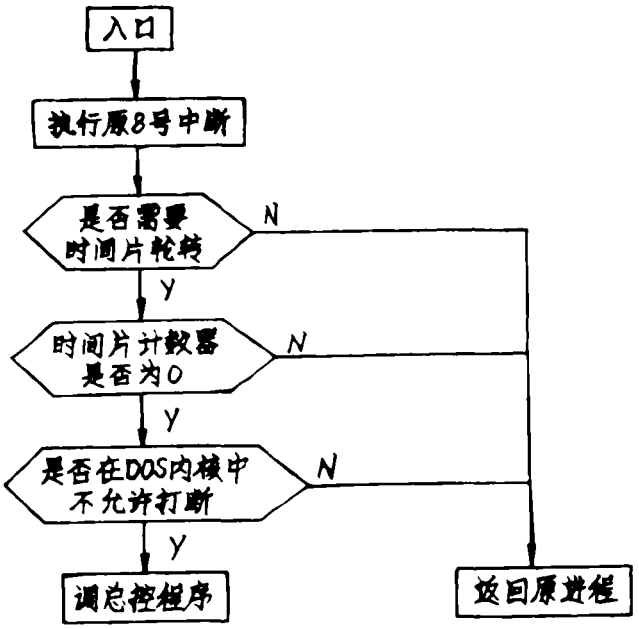

图5 新8号中断的原理图

为了使现场得到正确的保护,我们还需对保护现场的汇编子程序作出修改。这是由于一般过程的返回与中断的返回在PSW上的处理不同。

在这一步骤中,我们需增添三个汇编程序,即:

- 设置新8号中断的子程序(于调度前调用)
- 恢复原8号中断的子程序(在0#进程结束时调用)
- 8号中断的例程程序

(2) 对优先级进行动态修改

?1994-2015 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

通常，在一个进程进入运行状态前，应先将其优先级减去一个值，这样做的结果使得进程在得到时间片的同时，其优先级也随着降低了。也可以不改变进程的优先级，即初始时所有的用户进程都赋以相同的优先级，而进入就绪队列(将就绪队列组织成优先队列)时按照同一优先级中后到的进程插在相应子队列的尾部的原则，同样可以达到使就绪队列秩序发生变化的目的。

(3) 多道分时系统的计时

在整个系统中，我们设置一个全程变量作为计数器。每个用户进程进入运行状态时，将计数器赋以一个事先约定的长度值。在用户程序的执行过程中，每来一次时钟中断，计数器值减1，直到计数器值减为0。此时该用户进程的时间片已经结束，应将此进程插入就绪队列。为了达到较明显的穿插效果，调试时计数器的初值不应赋得过长，一般在三个时间片以内效果较为明显。

在实际运行过程中，用户进程的运行时间可能长于预先分配给它的时间片，如图5所示。因为当计数器为0时，若“在系统标志单元”的值为1，则用户进程进入了DOS内核，不允许被打断，用户进程将仍继续执行，因而进程执行时间比预先分配的要长。

对程序进行了正确的修改以后，经过调试，便可出现轮转的结果。当然，各用户对应的程序不能太短。

3. 进程的同步与互斥，信号灯的实现

我们将已有的程序的功能进一步扩充，就可以实现进程的互斥与同步。信号灯结构描述如图6所示：

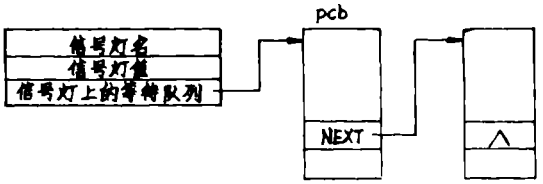

图6　信号灯结构图

在操作系统中，P、V操作是以原语的形式进行描述的。在此系统中，我们将P、V操作设定为不可打断的DOS功能调用。其中，P操作中主要有以下动作：

- 信号灯值减1；
- 若相减结果大于或等于0，则该进程继续执行；
- 若相减结果小于0，则该进程被封锁，并将其插入信号灯的等待队列，然后转进程调度。V操作的主要动作包括：
- 对应的信号灯值加1；
- 若相加结果大于0，则进程继续执行；
- 若相加结果小于或等于0，即信号灯的等待队列不为空，则从该等待队列中移出一进程入就绪队列，然后返回本进程继续执行。

这里需要新增的程序有：P、V操作的例程程序和将P、V操作设置为中断的子程序。

三　结束语

上面我们讨论了实现RMOS多任务分时系统的基本过程。此系统还可从多方面进行扩充和完

(下转第29页)

?1994-2015 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net

$$\text{标志寄存器 Ⅱ}=\begin{cases}>00;\text{禁止读写全局存贮器}\\>FF;\text{允许读写全局存贮器}\end{cases}$$

$$\text{标志寄存器 Ⅲ}=\begin{cases}>00;\text{PC 未准备好数据}\\>FF;\text{PC 已准备好数据}\end{cases}$$

· 系统复位时，标志寄存器 Ⅰ 和 Ⅲ 置为>00，标志寄存器 Ⅱ 置为>FF；

· TMS320C50 要向全局存贮器送出数据之前，首先查询标志寄存器 Ⅰ、Ⅱ 和 Ⅲ 的状态，直到查询到标志寄存器 Ⅱ 为>FF，标志寄存器 Ⅰ 和 Ⅲ 为>00 时，方可送出数据，送出数据之前应立即把标志寄存器 Ⅱ 置为>00，送出数据之后应立即把标志寄存器 Ⅰ 和 Ⅱ 都置为>FF；

· TMS320C50 要从全局存贮器读取数据之前，首先查询标志寄存器 Ⅰ、Ⅱ 和 Ⅲ 的状态，直到查询到标志寄存器 Ⅰ 为>00，标志寄存器 Ⅱ 和 Ⅲ 为>FF 时，方可读取数据。读取数据之后应立即把标志寄存器 Ⅲ 置为>00；

· PC 的操作与 TMS320C50 类同。

按照以上规范操作，可以正确实现两机通讯。

四、结束语

本文所提出的设计方案已在作者的科研中得到应用，实验结果表明，此系统硬件结构简单、软件编程方便，数据通讯迅速、可靠。

参 考 文 献

[1] Digital Signal Processing Application with the TMS320 Family. Texas Instruments. 1986.
[2] N. B. Jones and J. D. Mck. Watson(eds.). "Digital Signal Processing: Principles, Devices and Applications", IEE Control Engineering Series 42. Peter Peregrinus Ltk, 1990.
[3] 陈益新、张桂林，TMS32020 全局存贮器的设计及其应用，信号处理，1991, Vol. 7, No. 2, pp. 111～118.
[4] TMS320C5X User's Guide, Texas Instruments, 1993.

(上接第 23 页)善。如，在就绪队列上，可以开辟多就绪队列；可以增加进程的通讯功能；在设置优先级、调度策略上均可以做进一步的完善。

为了达到更佳的演示效果，一方面我们可以修改用户程序或用户进程的 pcb，使运行结果分屏显示，清晰地表现出多进程的宏观上的并发与微观上的顺序执行，另一方面，也可以将屏幕作为资源分配给各作业。当然后者符合操作系统的概念，但在实践上有一定的难度。

参 考 文 献

[1] 庞丽萍，操作系统原理，华中理工大学出版社
[2] 曾琪明，C 语言高级程序编制技巧，中科院希望高级电脑公司
[3] 沈美明等，IBM PC 汇编语言程序设计，清华大学出版社

?1994-2015 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net