嵌入式实时操作系统 VxWorks 内核调度机制分析

万　柳
(解放军信息工程大学信息工程学院　郑州 450002)

摘　要　本文简要介绍了多任务内核,重点分析了嵌入式实时操作系统 VxWorks 的内核调度机制——优先级抢占调度和时间片轮转调度算法。

关键词　嵌入式 RTOS　多任务　VxWorks　轮转调度

ANALYSIS OF KERNEL SCHEDULING MECHANISM IN EMBEDDED RTOS VxWorks

Wan Liu
(*School of Information Engineering, PLA University of Information Engineering, Zhengzhou 450002*)

Abstract　This paper introduces briefly multi-task kernel and mainly analyses kernel scheduling mechanism in embedded RTOS VxWorks— priority preemptive scheduling and round-robin scheduling.

Keywords　Embedded RTOS　Multi-task　VxWorks　Round-robin scheduling

1　引　言

多任务内核、任务调度机制、任务间通信和中断处理机制,这些都是 VxWorks 运行环境的核心。多任务处理和任务间通信是实时操作系统的基石。一个多任务环境允许将一个实时应用构造成一套独立任务的集合,每一个都有自己独立的执行路线和自己的系统资源,完成不同的功能。

RTOS 的实时性和多任务能力在很大程度上取决于它的任务调度机制,从调度策略上分为优先级调度策略和时间片轮转调度策略;从调度方式上分为可抢占、不可抢占、选择可抢占调度方式;从时间片上分为固定与可变时间片轮转。本文就 VxWorks 的核心问题多任务内核和任务调度机制进行讨论。

2　多任务内核

应用程序经常被组织成独立的,相互协作的一些程序。每一个单独执行的程序都是一个任务。在 VxWorks 中,任务能直接共享所有的系统资源。

2.1 多任务

多任务处理提供了对应用程序控制和对多样的、离散的真实世界事件的反应的基本机制。VxWorks 的实时内核,提供了基本的多任务环境。多任务机制从表面上看起来是创建了许多同时运行的线程,但实际上是内核在按照一定的数学法则对它们的执行进行调度。每个任务都有自己的上下文(所谓上下文是指:有系统内核运行调度的任务可能会在任何时刻访问的 CPU 环境和系统资源)。在一次上下文切换中,一个任务的上下文将保存在任务控制块中(TCB)。一个任务的上下文包括以下内容:

- 任务的执行情况,即任务程序计数器
- CPU 寄存器和浮点计数器。
- 动态数据堆栈和函数调用。
- 标准 I/O 和错误的 I/O 分配。
- 延时定时器。
- 时间片定时器。
- 内核控制结构。
- 信号控制结构。
- 调试和运行监视值。

由于 VxWorks 操作系统的内存是线性的,所有代码执行在单一的公共地址空间内,因此内存地址空间不属于任务上下文。

2.2 任务状态转换

任务状态反映任务当前在系统所处的情形,内核负责维护系统中所有任务的当前状态,任务的状态转换反映了任务在内核调度中的运行情况。

当创建任务时,任务处于挂起(suspended)状态,为使创建的任务进入就绪(ready)状态,必须激活该任务。VxWorks 的内核对应的状态转换如图 1 示。

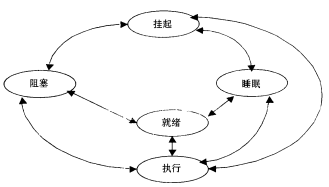

图 1　VxWorks 任务状态转换图

任务状态队列如图 2 示。

收稿日期:2003－04－03。万柳,讲师,主研领域:嵌入式操作系统,计算机网络与应用。

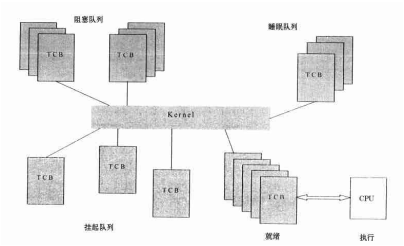

图 2 VxWorks 任务状态队列

3 任务调度机制

3.1 任务调度的几种算法

任务调度算法的发展过程大致经历了三个阶段，存在以下三种调度算法：

3.1.1 Control loop

它是通过一种循环查询任务标志位的算法来调度任务，其特点是：简单，但不适于大系统、不灵活、不易升级，如果升级或添加新的模块就要打开 loop，重新设计算法。

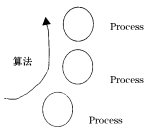

图 3

3.1.2 Interrupt schedule

它是为每个任务分配一个中断，通过中断的方式来调度任务，其特点是：简单，但把应用的并行度下降到 CPU 级。计算机的发展是体系结构的发展，是层次结构的发展，如果采用中断的任务调度方式，任务就绕过操作系统直接与硬件发生关联，破坏了计算机的层次结构。

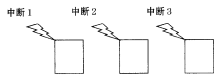

图 4

3.1.3 RTOS

如果系统中有 RTOS 的支持，则调度算法靠实时内核来完成，就是说实时内核负责为多任务系统中每个任务分配 CPU，并且负责任务间的通信。对于 VxWorks 的内核来说，基于优先级的抢占调度方式是系统的默认工作方式。当然，也可以根据应用程序的需要选择时间片轮转的调度方式。

3.2 优先级抢占调度

在一个基于优先级的抢占式调度系统中，每一个任务都有一个优先级。内核保持把 CPU 分配给优先级最高的就绪任务。这种调度方式意味着只要有高优先级的任务处于 Ready 状态，内核将立即保存当前任务的上下文，并切换到高优先级任务的上下文。如图 5 所示，任务 T1 被高优先级的 T2 任务抢占，T2 又被 T3 抢占。T3 运行完后，T2 继续运行，T2 运行完后，T1 才能继续运行。

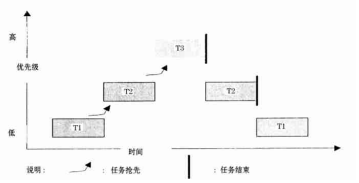

图 5

VxWorks 的内核总共有 256 个优先级，数目从 0—255。0 为最高优先级，255 为最低优先级（与 PSOS 正好相反）。任务在创建时将分配一个优先级。在运行过程中也可以使用 taskPrioritySet() 函数改变优先级。这种设计可以使应用程序更贴近现实。

3.3 轮转调度

优先级抢占调度方式可以扩展为轮转调度方式。轮转调度方式的主要作用是允许相同优先级的就绪任务在运行时公平地分享 CPU。如果没有轮转调度，那么当多个同优先级任务必须共享处理器时，一个任务如果不阻塞的话，就可能会独占 CPU，导致其它所有的任务都无法运行。轮转调度实现在同优先级任务中公平分配 CPU 资源的方法一般是时间片轮转。一组任务中的每个任务执行指定的时间间隔或时间片；然后另一个任务执行相同的时间间隔，依此类推。这种调度方法的公平之处在于，只有所有的任务都获得过依次时间间隔运行后，才能有任务获得第二个时间片运行的机会。系统中使用函数 kernelTimeSlice() 允许系统实现轮转调度。更精确地，每个任务有一个运行时间计数器（run-time counter）记录增加的时钟 tick 数。当一个特定的时间片结束时。计数器将清空，并且将任务挂到同级任务的队尾。当一个新任务加入优先级组时，本任务将挂到队尾，并且将运行时间计数器初始化为 0。如果一个任务在自己的运行时间间隔中被高优先级的任务抢占，它的运行时间计数器将被保存，直到任务重新运行时再恢复。如图 6 所示，T1，T2，T3 是同级任务，T2 被高优先级的 T4 任务抢占，当 T4 运行完后，T2 继续运行至结束。

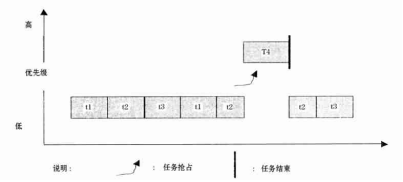

图 6

3.4 抢占上锁

VxWorks 的内核调度器可以在一个任务中通过 taskLock() 和 taskUnLock() 函数，明确地禁止或使能抢占。当一个任务通过调用 taskLock() 函数禁止内核调度时，这个任务将运行在不被其它任务中断的方式下。但是，如果此任务明确地阻塞或挂起，调度器将选择一个优先级最高的就绪任务执行。不过，当关闭了调度器的任务阻塞解除再次执行时，抢占式调度将再次被关闭。抢占调度关闭只是禁止任务上下文的切换，它并不禁止中断处理的发生。抢占禁止可以用于实现互斥（mutual exclusion），不过最好让调度禁止的时间尽量短。

4 结 论

通过对嵌入式操作系统 VxWorks 内核中基于优先级的抢占调度和时间片轮转调度两种机制的分析可知，该内核的实时性很强，为嵌入式实时系统的开发提供了一种高性能的操作系统平台。

参 考 文 献

[1] Jean Labrosse. MicroC/OS—II: The Real—Time Kernel. R & D Books[M]. 1998.

[2] 孔祥营等，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.

[3] 陈莉君，Linux 操作系统内核分析[M]，人民邮电出版社.

[4] W. Richard Stevens 著，杨继张译，UNIX 网络编程[M]，清华大学出版社.