一种基于 Minix 的实时化方案的设计

杜文军[1],杨 静[2]

(1.东北电力大学 信息化教学中心,吉林 吉林 132012;
2.辽宁石油化工大学 信息与控制工程学院,辽宁 抚顺 113001)

摘 要:通用操作系统实时化主要是针对 Linux、Unix 系统的实时化,是实时操作系统开发的一个重要途径,以微内核结构的 Unix 系统 Minix 为基础,对其进行了整体的实时架构,主要包括中断处理和进程调度机制的实时化设计,并对设计的合理性进行了测试。

关键词:Linux;Unix;实时操作系统;微内核

中图分类号:TP316.2 **文献标识码**:A

Real-time design based on Minix

DU Wen Jun[1], YANG Jing[2]

(1.Northeast Dianli University, Jilin 132012,China;
2.School of Information and Control Engineering, Liaoning Shihua University, Fushun 113001,China)

Abstract: The work of time-sharing operating system's real-time is to modify some kernel function of Linux or Unix.And it is the main way to the development of real-time operating system. Based on the Minix system which is microkernel and is belonged to Unix system. This article does some real-time work on Minix's interrupt menagement and process scheduling. At last,it takes a practice to prove its reasonability.

Key words:Linux;Unix;real-time operating system;microkernel

目前,围绕嵌入式系统的研究与开发,市场对实时操作系统的需求不断增加[1]。对通用操作系统的实时化是目前实时操作系统开发的一个重要方法,Linux 系统的实时化工作已经非常成熟,但是 Linux 系统功能采用了宏内核结构,具有内核管理系统的所有功能,虽然系统运行的效率较高,但其模块间的耦合度也高,加上现在的 Linux 内核体积庞大而复杂,使基于 Linux 的实时操作系统对内核源码的修改和维护的代价非常高。Minix 源代码完全公开并且遵循和 Unix 一样的标准,提供的工具和软件都是 Unix 用户和程序员所熟悉的,相对于 Linux 而言,它体积小、功能结构灵活、内核易扩展,而且采用了适合实时操作系统结构的微内核设计[2],对 Minix 内核进行实时化修改使之成为实时操作系统具有一定的优势。

1 实时化 Minix 的整体架构

Minix 是一组进程的集合,进程之间以及用户进程之间使用消息传递机制来通信,Minix 是一个如图 1 所示的 4 层结构[3]:

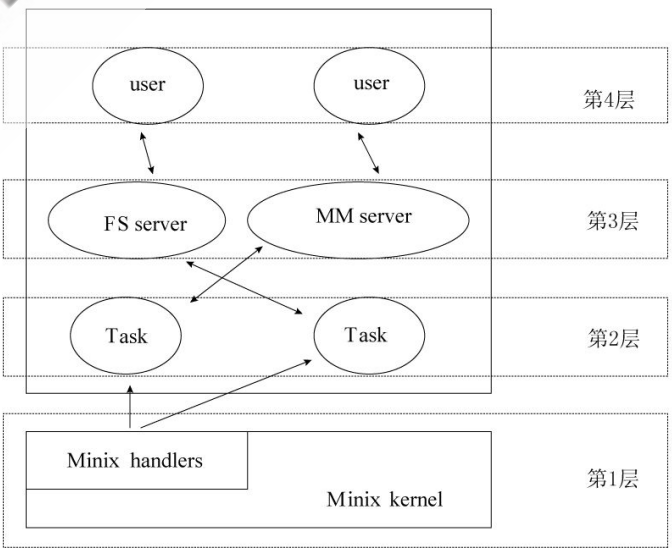

图 1 Minix 的 4 层结构

第 1 层是内核,主要功能包括上下文切换(Context Switching)、进程调度(Process Scheduling)、中断处理(Interrupt Handling)、基本内存管理(Basic Memory Management)以及进

程通信机制等;第 2 层主要处理一些底层的 I/O 操作,如设备驱动程序。第 3 层是服务器进程,这些进程向用户进程提供服务,主要是内存管理(Memory Manager Server)和文件系统服务器(File System Server)。第 4 层是一些用户进程,包括 init 进程、shell 进程和编辑器软件等。

实时化 Minix 的设计中,在 Minix 系统的硬件和内核之间添加了一个独立的、非常小的实时微内核,如图 2 所示。Minix 在实时内核的控制下运行,实时内核接管了所有的硬件中断,充当中断硬件抽象层。当有实时任务时,实时内核调度实时任务运行;当没有可运行的实时任务时,实时内核才将控制权交给 Minix。实时内核完成底层实时任务的创建,中断服务程序,并为底层实时任务、中断服务程序和进程之间提供通信,将原来的 Minix 作为实时内核下的一个随时可以被实时任务抢占的任务来调度。

图 3 显示了实时化 Minix 的整体架构。所有的中断

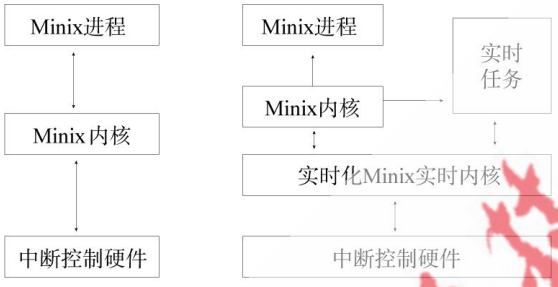

图 2 实时化 Minix 设计

都由实时内核捕获,由实时内核对标准 Minix 的中断进行模拟。Minix 内核在进入临界区时为了避免自己被实时进程抢占会禁止中断。当 Minix 告诉硬件禁止中断时,实时内核截住这个请求,对软件标识位做标记,实际上并没有操作 EFLAGS 寄存器的内容。如果在系统运行过程中有这样一个 Minix 被禁止的中断发生了,那么实时内核记录这次发生并返回,但并不执行 Minix 的中断处理函数,随后,当 Minix 允许中断时,所有被记录的中断将被执行。

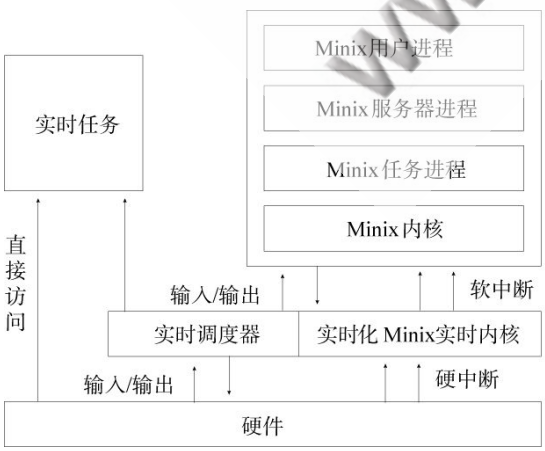

图 3 实时化 Minix 的架构

2 实时化中断的设计

2.1 实时化 Minix 的中断

如图 4 所示,在实时化 Minix 中,将系统的中断分为实时中断和非实时中断。

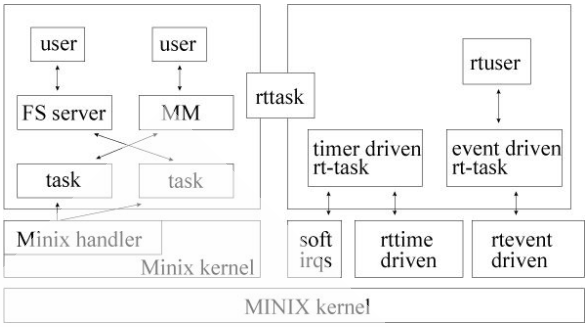

图 4 实时化 Minix 的中断

(1)非实时中断:如果 Minix 内核没有禁止中断并且系统中没有任何的实时进程和实时中断处理函数执行,就处理这个中断。如果此时 Minix 内核禁止了中断,则将这个中断放入中断等待队列,在 Minix 允许中断时处理。

(2)实时事件驱动的中断:当系统在实时模式下运行时,仅当实时事件驱动中断的优先级比被中断的实时进程或实时中断高时才被处理,否则被标记为稍后处理。

(3)定时器驱动的中断:当发生设备中断时,这类中断不作处理,它们只在定时器中断的周期到来时执行。在每次设备中断时,该中断处理函数仅被标记为触发,当它的周期到来时会被触发。

2.2 实时化 Minix 的中断处理

Minix 用两阶段中断处理的方式来处理中断,在中断发生时只做部分的工作,然后给 I/O 任务发送一个消息由它来完成中断剩余的工作。但是这种方式隐含了上下文切换的操作。有些中断事件的处理比较复杂,中断处理程序必须花更多的时间才能把事情做完。为了解决这种在短时间内完成复杂处理的矛盾,本文参考了 Linux 的解决方式,把中断分成上半部和下半部,上半部负责“登记中断”,下半部完成中断事件的绝大多数操作。当一个中断发生时,就把设备的中断例程的下半部挂到该设备的下半部执行队列中去,然后等待新的中断到来。这样,上半部执行的速度很快,可以接受所负责设备产生的更多中断[4-5]。具体的处理方式是:当中断发生时,实时化 Minix 的中断处理函数首先判断中断的类型,如果是实时中断且中断的优先级比系统中当前运行的实时进程的优先级高,则处理这个实时中断;如果优先级低于当前运行的实时进程的优先级则推迟该中断的处理,把它放在中断的下半部去处理。如果是优先级高于当前运行实时进程的优先级的非实时中断,而且此时标准 Minix 内核没有禁止该中断,那么,处理这个非实时中断;如果此时标准 Minix 内核禁止中断,则不处理这个

中断并在一个软件标志位中设置标准Minix 禁止中断的信息,并把它放到中断的下半部,等到 Minix 调用中断返回函数 unlock( )允许中断时,这些中断会在 unlock( )中全部处理。

3 实时化 Minix 进程调度的设计

3.1 实时化 Minix 进程的设计

Minix 的进程有 3 个状态：就绪态、阻塞态和运行态。为了保持 Minix 系统在实时化后的灵活性,并没有在内核中添加新的专门创建实时进程的代码。实时化 Minix 的实时进程是由非实时进程转变而来的，一个实时进程在终止前必须被转变回非实时进程。这样整个系统的进程就有了第 4 种状态:实时态。

这样,从 Minix 系统进程的角度看,在实时化设计后,系统中的进程总共有 4 种状态:就绪态、阻塞态、运行态和实时态。这几种进程状态间的转换关系如图 5 所示。

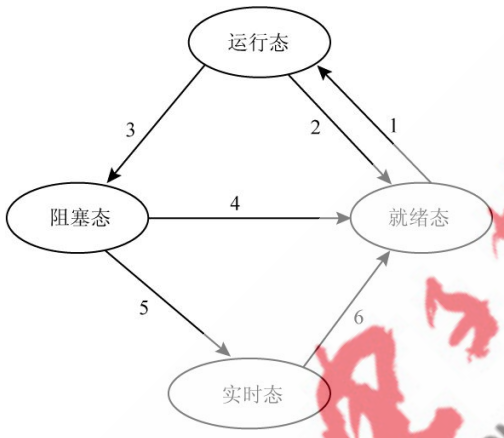

图 5 实时 Minix 的进程状态和转换关系

如图 5 所示,1、2、3、4 是原有进程状态的转换,5、6 是实时化 Minix 的扩展,它们的转换是：

阻塞态到实时态:非实时进程转化为实时进程。

实时态到就绪态：实时进程转化为非实时进程。

3.2 实时化 Minix 进程调度器的设计

Minix 进程调度的设计采用一种多级调度算法,调度器维护 16 个队列,每个队列具有一个优先级。这样,Minix 的调度器首先按优先级次序看优先级最高的队列中有没有就绪的进程，如果有这样的进程，就运行它;如果没有,就看下一个优先级的队列。而在每个队列内部采用时间片轮转调度算法。如果一个运行的进程用完了它的时间片,则它被移到队列尾部,并重新分配一个新的时间片。

实时系统的调度策略要求在任务调度的过程中,调度所花的时间应是常数而与系统中建立的任务数目的多少无关[6]。实时化 Minix 的任务调度和 Minix 的调度思想是相似的,首先运行就绪任务(进程)中优先级最高,如果一个实时进程在运行时，有优先级比它高的进程,则高优先级的进程将抢占调度器运行。确定哪个任务的优先级最高的工作是由调度器完成的。实时化 Minix 的实时调度器的设计是基于一系列的进程队列和一个位图,如图 6 所示。实时内核维护着 16 个实时进程就绪队列,每个队列一个优先级。位图中的每一位代表对应 16 个队列的每一个优先级,在初始情况下,位图的所有位和所有队列都是空的。当一个实时进程变为可运行时,位图中和该进程相同优先级的位被设置,同时实时进程按照它的优先级字段(priority)值的大小被添加到相应的实时进程就绪队列中。这样找到系统中优先级最高的进程只需要找到位图中的第一个被设置的位。因为优先级的个数是固定的,所以扫描位图的时间是常数而不会受系统中可运行进程数目的影响[7]。

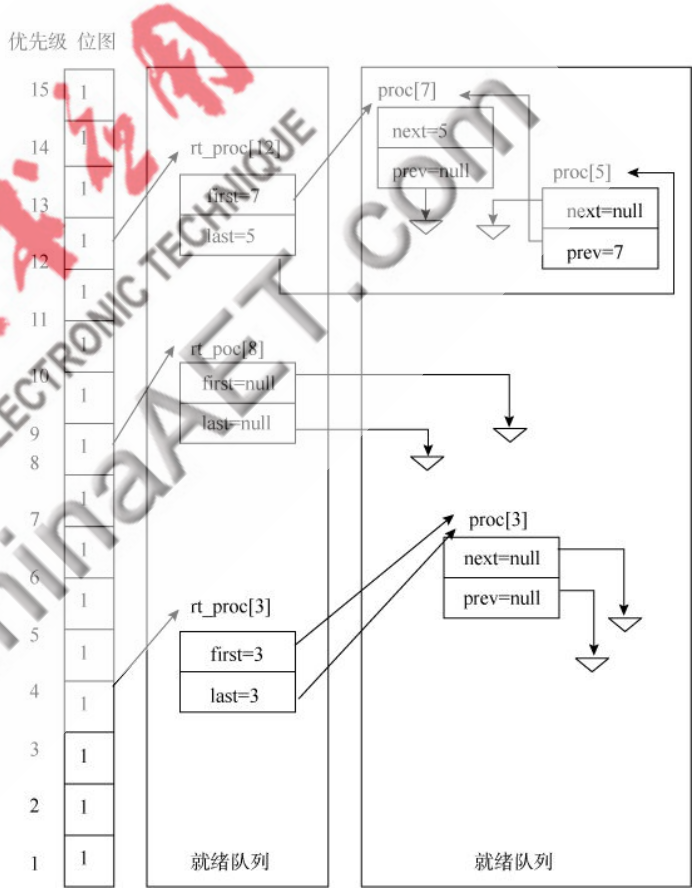

图 6 实时化 Minix 的调度队列

4.实时化 Minix 的性能测试

因为微内核架构的操作系统用消息机制来处理系统调用、进程调度和进程间通信等[8],所以消息效率是实时化 Minix 性能的一个重要参数。为了证实实时化 Minix 设计方案的合理性，选用了目前比较成熟的实时 Minix 系统 RT-Minix 为参照。对实时化 Minix 的消息传输效率和中断服务时间做了测试,数据如表 1 和表 2 所示。

表 1 RT-Minix 和实时化 Minix 的消息传输时间比较

试验操作系统	实时化 Minix(μs)	RT-Minix(μs)
单个消息传输时间平均值	95.875	109.375

表 2 实时化 Minix 的中断服务时间

实时化 Minix 中断类型	事件驱动的中断(μs)	定时器驱动的中断(μs)
最大值	33.506	256.622
最小值	22.232	215.616
平均值	33.506	256.622

由表 1 和表 2 的数据可以看出，实时化 Minix 在一定程度上提高了消息传输的效率，其中断服务的时间达到了实时操作系统的要求，能够满足实时操作系统的性能需要。

对 Linux 及 Unix 系统的实时化方案的研究，是国内的许多高校和科研机构研究嵌入式和实时操作系统的热点。Unix 操作系统不仅具有传统的优势，而且其采用的微内核设计结构更适合实时操作系统的开发需求，因此，对 Unix 系统及类 Unix 系统的实时化工作具有一定的理论和应用价值。

参考文献

[1] DOUGLASS B P .嵌入式实时系统开发[M].柳翔，译.北京:机械工业出版社，2005.

[2] 马毅，李霞峰，盛焕烨.基于 Unix 的实时操作系统设计[J].计算机工程与应用，2001，39(4):109-110.

[3] TANENBAUM A S ，WOODHULL A S.操作系统设计与实现[M].北京:电子工业出版社，2007.

[4] 张文斌. 基于嵌入式 linux 的实时操作系统研究 [D].阜新:辽宁工程技术大学，2003.

[5] 许先斌，熊慧君. 基于 ARM9 的嵌入式 Linux 开发流程的研究[J].微计算机信息，2006，22(4-2):87-89.

[6] 朱珍民，段斌. 嵌入式实时操作系统及其应用开发[M].北京:北京邮电大学出版社，2006.

[7] 尹凌，费斐，王晓东.一个嵌入式实时 UNIX 的技术研究[J].计算机工程，2001，27(8):61-65.

[8] 徐晓磊，董兆华，吴建峰，等.Unix 可抢占内核的分析[J].计算机工程，2003(15):115-117.

(收稿日期:2009-09-15)

(上接第 10 页)

[3]丁雅斌，彭翔，田劲东，等. 基于点阵编码的三维主动视觉标定[J]. 光子学报，2006，35(11):1774-1778.

[4] 陈鉴富.基于组合光栅投影的复杂曲面测量关键技术研究[D].南京:南京航空航天大学，2008.

[5] CHEN F，BROWN G M，SONG M. Overview of three-dimensional shape measurement using optical methods[J]. Opt. Eng，2000，39(1):10-22.

[6] 欧阳京，吕乃光，吴迪.基于时间相位展开算法的相位测量轮廓术[J].北京机械工业学院学报，2007，22(1):1-4.

[7] 金亚，达飞鹏，盖绍彦.三维测量中光栅投影条纹边界编码方法的改进[J].计算机与现代化，2006(1):19-22.

[8] LU Zhen Tai ，ZHANG Ming Hui ，FENG Qian Jin ， et al. Medical image registration based on equivalent meridian plane[J]. ICIAR，LNCS4633，2007:982-992.

[9] HUANG Wei Yao，LIU Zhi Jing，PAN Wen Juan. The precise recognition of moving object in complex background [C]. Proceeding of IEEE Third International Conference on Natural Computation，2007.

[10] 蒋晓红，郑秋梅，杨发科.一种基于分块二值化思想的图像检索方法[J].微计算机应用，2008，29(3):47-51.

[11]ABUTALEB A S. Automatic thresholding of gray level pictures using two dimensional entropy [J]. Computer Vision，Graphics and Image Processing ，1989，47(1) .

[12]KAPUR J N，SAHOO P K，WONG A KC. A new method of gray level picture thresholding using the entropy of the histogram [J]. Computer Vision，Graphics and Image Processing，1985，29(2) .

[13]张燏，吴志斌，陈淑珍，等.一种新的自适应二值化方法[J].计算机工程，2002，28(5):184-185.

[14] 鄢治国，徐德，李原.一种基于自适应二值化阈值的焊缝边缘特征提取方法[J]. 焊接学报，2008，29(8):34-38.

[15] 李红岩.基于空间二进制编码的阈值分割方法研究[J].计算机仿真，2008，25(7):196-199.

(收稿日期:2009-01-11)