ISSN 1000-0054 清华大学学报 (自然科学版) 2009年 第 49卷 第 7期 w26
CN 11-2223/N J Tsinghua Univ (Sci & Tech), 2009, Vol. 49, No. 7 http://qhxbw.chinajournal.net.cn

基于 Linux的混合实时操作系统

王继刚[1,2], 郑纬民[1], 钟卫东[2], 李 翌[2]

(1. 清华大学 计算机科学与技术系, 北京 100084; 2. 中兴通讯股份有限公司 中心研究院, 成都 610041)

摘 要: 多媒体及高速网络技术的发展,大大扩展了应用的需求,很多应用不仅具有实时特性,还要求系统提供丰富的服务和可扩展能力。本文在深入研究影响 Linux实时性能的因素,比较目前主流实时 Linux优缺点的基础上,设计开发出一种混合实时操作系统——KLinux. KLinux基于开源 Linux,在内核架构、混合实时调度,以及错误处理等方面进行了改造。实验结果表明,改进后 Linux内核在任务调度、上下文切换、CPU运算等方面的实时性能分别提高了 253%、177%、255%,同时支持内核态实时应用和用户态非实时应用的双态混合开发,可满足当前绝大多数应用的需求。

关键词: 混合实时操作系统; 实时调度; 可抢占核心; 错误处理; SMP

中图分类号: TP 316 文献标识码: A

文章编号: 1000-0054(2009) 07-1012-04

Hybrid real-time Linux operating system

WANG Jigang[1,2], ZHENG Weimin[1], ZHONG Weidong[2], LI Yi[2]

(1. Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China; 2. Central Research Institute, ZTE corporation, Chengdu 610041, China)

Abstract The development of multimedia applications and high-speed networks has significantly increases the application requirements. Many applications not only have strict real-time requirements, but also require a wide variety of system services. However, Linux limits speeds when a wide range of applications are used. The performance of Linux systems was analyzad to develop a hybrid real-time Linux operating system (KLinux). The kernel architecture design emphasizes hybrid real-time scheduling and fault handling. Tests show that these improvements effectively enhance the real-time capability of the Linux kernel. Task scheduling is increased by 253%, context switching by 177% and CPU speed by 255%. KLinux also allows real-time and non-real-time applications to coexist, so it can handle most applications.

Key words hybrid real-time operating system; real-time scheduling; preempted kernel; fault handling; SMP

近年来,嵌入式系统已广泛应用于各个领域,其复杂度和功能性也在日益增加。这些系统上的应用不仅有实时性方面要求,还需要系统为其提供丰富的服务和可扩展能力。同时,随着网络通信技术的发展,实时应用已很少被设计在专门的实时硬件上,它们更多地运行于通用设备中,与非实时应用共存,这些发展对底层操作系统提出了越来越高的要求。

传统的嵌入式操作系统,大多致力于重要的硬实时领域,包括工业控制,航空与国防等,尽管能满足应用的实时性要求,却难以提供其他扩展服务。而通用操作系统虽然具有良好的服务,但无法保证有效的实时性能,它们都不能很好地支持当前应用的需求。由于处于硬实时和通用范畴之间,因此本文引入了"混合实时操作系统"的概念。一些相关的研究正基于 Windows NT和 UNIX开展,而具有丰富的系统功能、高度定制性以及开放源码特征的 Linux则获得了更多的关注[1]。

为了服务更广泛的应用,本文在文[2]的基础上设计并实现了混合实时操作系统 KLinux, KLinux基于开源 Linux操作系统,在内核架构、混合实时调度和错误处理等方面进行了改造。能够根据应用的需求提供多种扩展服务,允许实时和非实时任务共存,完全可以满足当前绝大多数应用的需求。

1 相关工作

Linux是继承 UNIX技术发展而来,设计之初是为了实现通用分时操作系统,要突出各个任务共享资源分配的公平性,因此难以满足实时应用的需要[3]。总的来说, Linux实时性的瓶颈表现在: 1) Linux划分为用户态和内核态,程序的执行流程需要在双态之间不断切换,消耗了大量的时间。

收稿日期: 2008-04-14

基金项目: 国家"八六三"高技术项目 (2004AA1Z2351)

作者简介: 王继刚 (1978-), 男 (汉), 安徽, 博士后。

通讯联系人: 郑纬民, E-mail: zwm-dcs@tsinghua.edu.cn

2) Linux提供的系统调用效率很低，一方面是由于 Linux内核本身的复杂性引起的，另一方面，系统调用、POSIX的实现机制也比较复杂。3) Linux内核的抢占粒度较粗，禁止调度的代码段过长，导致高优先级任务难以得到及时响应。

为了改善 Linux内核的实时性能，提出了许多方案。RT-Linux[4]与 RTAI[5]采用双内核结构对 Linux进行实时性改造，这种改造方案实现了一个小的实时内核。原来的非实时 Linux核心作为一个可抢占的任务运行于这个小核心之上，所有的任务都在核心地址空间运行。从低层硬件发出的中断请求，由实时内核接收，然后根据需要向普通进程或实时进程提交，而普通进程和实时进程之间是由 RT-FIFO来通信的。双内核方案的优点是保证了实时进程的性能。缺点是破坏了核的完整性，结构比较复杂，实时进程的调试也很困难，用户态与内核态之间的切换开销还是难以降低。同时，运行在实时内核中的实时进程无法获得任何的 Linux系统服务。

MontaVista Linux[6]采取了在内核中添加低延时补丁的方式来增强 Linux系统的实时性。基本思想是产生运行调度器的机会，减少事件发生到调度函数运行的时间间隔。优点是对 Linux内核的改动很小，缩短了用户进程响应时间，缺点是未能解决用户态与内核态切换系统开销的问题。

通过 UTIME技术，Kurt-Linux[7]实现了利用 μs级定时器来提高系统实时性。不同于 RT-Linux单独实现一个实时内核的方法，Kurt-Linux是在通用 Linux的基础上实现的，可以使用 Linux自身的系统调用。Kurt-Linux将系统分为3种状态，在正常态时它采用普通的 Linux的调度策略，在实时态只运行实时任务，在混合态实时和非实时任务都可以执行。然而，由于 Kurt-Linux采用时间驱动调度器，不同级别的定时器，都由一个管理器控制，因此，无法对μs级定时器提供更优先的支持，并且需要频繁地对时钟芯片进行编程设置。同时，由于核心不可抢占的性质依然没有改变，一个运行中的普通任务一旦进入内核，则很可能会妨碍实时任务的及时调度，给混合态引入了很大的风险。

RED-Linux[8]实现了一种灵活的分等级的调度框架，在这个框架下，具有不同等级实时需求的各类任务会被适当的调度策略所支持。RFRTOS[9]从实时调度、内核抢占、细粒度定时器等方面对 Linux内核进行了改进，有效地提高了 Linux的实时性能。然而，由于 Linux自身的局限性，RED-Linux和 RFRTOS只能作为软实时的解决方案，难以满足当前应用的需要。

2　KLinux的设计与实现方案

混合实时操作系统 KLinux的设计核心就是在不改变 Linux系统构架的基础上，对内核进行功能增强。为应用提供实时支持，满足实时应用对多任务管理、互斥、通讯等方面的需求，让实时应用能够在内核态稳定运行，避免由于状态切换导致的系统开销；同时，完善内核调度机制，使之具有实时与非实时任务混合调度能力；为保证系统可靠性，分别为内核态和用户态应用提供了错误处理机制。KLinux具有 $O(1)$调度机制、支持 SMP多核处理器、内核用户应用双态开发、支持可抢占机制、双态应用错误处理等众多特性。

2.1　内核体系结构设计

KLinux在基本 Linux内核基础上添加了一系列性能增强模块。比如线程增强模块 KTH提供了线程管理能力，以支持应用在内核运行方式；进程间通信增强模块 KIPC提供了进程间的快速消息能力，同时支持内核态和用户态应用间的通信；运行监控配合错误管理建立了内核和应用的运行状态监控及异常处理机制，提高了 KLinux系统运行的稳定性。

KLinux提供了内核态与用户态应用双态开发功能。从图 1中可以看出，KLinux的内核态应用建立在 uClibc/c++库上。而在用户态，整个应用建立在标准的 Glibc/c++或 uCLibc/c++上。由于内核态和用户态的应用都建立在标准的 c库接口上，并且，KLinux还提供了对用户态和内核态多个用户命令解释器进行统一管理的功能。因此，应用可以灵活地切换其运行状态，方便用户在实时性、安全性、扩展性等方面进行权衡调整。

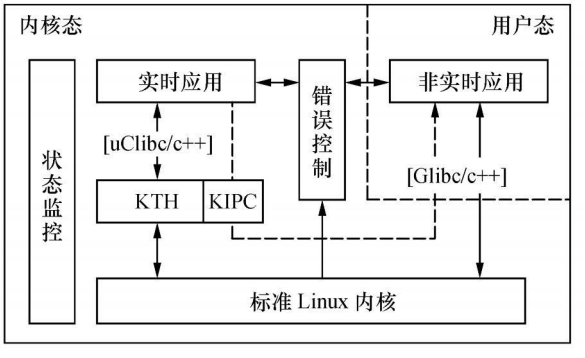

图 1　KLinux内核体系结构

2.2 实时线程管理

KTH模块的重要功能是对实时任务线程的管理,整个 KTH线程库的处理可分为初始化、主线程处理和工作线程管理,其中工作线程管理又包括线程的创建、中止、挂起、唤醒、延迟,本地存储的设置和获取,以及调度参数的设置和获取等功能。

主线程是 KTH线程管理的核心,它在线程管理初始化时被 Linux系统线程 KEVENTD创建,除非整个 KTH模块停止运行,否则它不能退出。主线程维护一个工作队列,如图 2所示,工作队列中保存着主线程需要执行的若干"任务"。系统运行中,当某线程需要创建(或中止)某 KTH线程时,会提交相应请求到主线程的工作队列中。主线程将执行工作队列中未完成的请求,或响应其他线程的请求。可见,其他线程与主线程之间是一种客户与服务器之间的关系,这种机制可以确保一些关键的行为(如线程创建、线程中止等)都在主线程中运行,使得管理更集中,保证系统的稳定性。

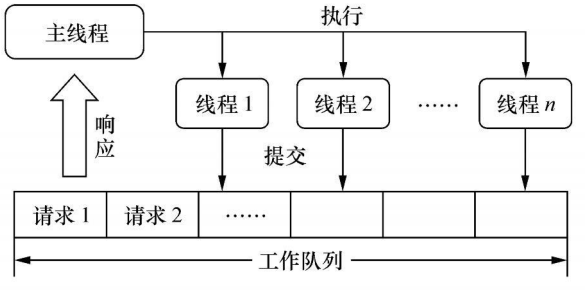

图 2 KTH模块线程管理机制

2.3 可抢占调度方案

KTH模块中的线程调度模型为每个 CPU 都建立一个实时线程运行队列和相应的优先级掩码数组,运行队列中包含了 256个优先级链表,每个链表上可以挂接多个相同优先级任务,同一优先级链表按照 FIFO的模式运行。优先级掩码数组里每项掩码对应一个优先级,当掩码有效时,说明该优先级上有准备运行的任务,反之则无。每次调度时,调度模块通过优先级掩码取得任务索引,使用任务索引在运行队列上取得最高优先级的可执行线程,然后执行线程切换。如图 3所示。

KTH模块实时线程与普通 Linux线程调度融合,实时线程队列里优先级高于所有普通 Linux线程的优先级,每次调度先从实时线程开始,只有当实时线程队列里没有需要运行的线程时,才运行普通 Linux线程的调度。在每次调度时,首先调度实时线

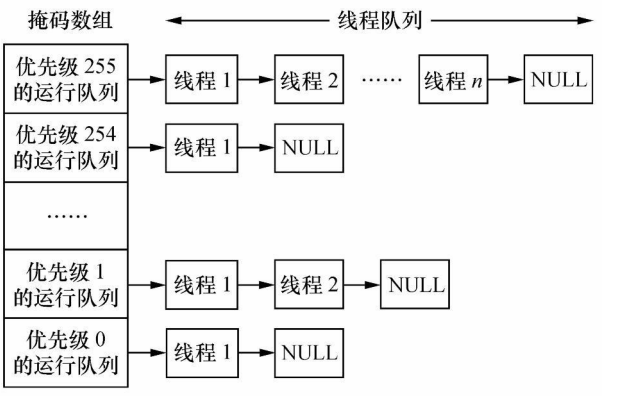

图 3 优先级掩码数组及线程队列模型

程,在没有实时线程可以运行的情况下,再调度普通 Linux线程。

为了使得实时线程调度支持 SMP, KLinux为 SMP系统中的每个 CPU 都建立一个运行队列和优先级掩码,当 CPU 进入调度时,首先取得当前线程管理结构,然后取得当前线程的 CPU号(实时线程的 CPU号是在线程创建时指定的或负载均衡时动态分配的),再通过 CPU号取得当前 CPU的实时线程运行队列,然后运行上述的调度算法。

2.4 错误控制

错误控制模块提供了对系统运行异常进行捕捉和处理的机制,包括 3个子功能模块: 错误框架、错误捕捉和错误处理。错误框架负责维护错误处理状态机模型,控制着整个处理流程; 错误捕捉完成系统和进程的运行错误捕捉功能,通过直接调用或信号触发的手段,启动错误处理代码; 系统或应用的错误被捕获后,由错误处理部分进行处理。可以提供基本处理流程,也支持用户自定义处理手段。同时,错误控制模块还会调用运行监控模块的部分服务,进行错误信息的存储和告警操作。

图 4是错误控制在用户态和内核态环境下的运行方式,其中错误捕获是核心,所有代码异常都会进入,它将调用错误处理,错误处理分析用户设定的处

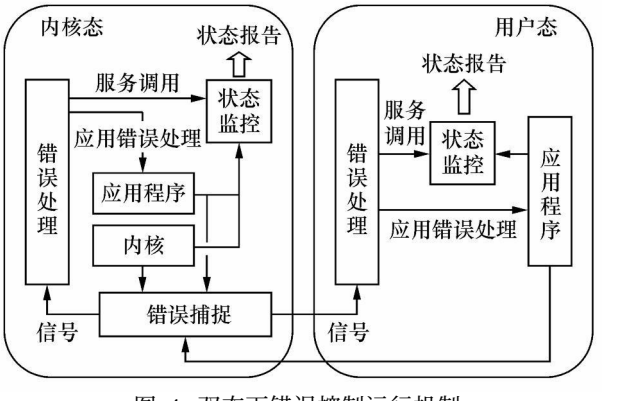

图 4 双态下错误控制运行机制

理策略,处理错误,它将调用状态监控提供的服务,进行信息报告、状态保存等操作。从图中可以看到,错误处理在内核和用户态同时提供,在用户态的处理机制,通过信号进行驱动,完成单个进程的错误管理功能,而内核态的错误处理机制,可以同时接受错误捕捉机制的调用,完成对系统的错误管理功能。

3 实验测试

本文实验所采用的测试环境为 CPU PIII 900, 内存 384 MB, 硬盘为 5 400 r/min 20 GB的 IBM硬盘; RedHat 9.0+ 2.6.20标准内核

表 1给出了 KLinux内核态应用与标准 Linux应用的性能对比。测试的范围包括任务调度、上下文切换、CPU运算能力,并对测试结果做出分析。测试用例思想,任务调度: 测试两个任务交替获取和释放信号量一定次数所花费的时间; 上下文切换时间: 计算两个任务调度总时间,扣除线程挂起和唤醒时间,反映出系统调度性能; CPU运算能力: 统计 CPU进行科学计算 (包括加减移位操作)的性能。

表 1 实时性能对比结果

操作系统	任务调度 次· s^{-1}	上下文切换 s	CPU运算 s
KLinux内核态	571 800	8.50	1.01
标准 Linux	162 000	23.55	3.59

从表 1可以看出, KLinux内核态的各项性能指标远远超出标准 Linux, 在任务调度、上下文切换、CPU运算等方面的实时性能分别提高了 253%、177%、255%, 可以满足应用实时性需求。

在测试 KLinux内核态应用与用户态应用共存中,采用了内核态运行 LTP测试套件任务,用户态同时进行 FTP传输的方法。当内核态不运行 LTP时, FTP性能为 3.1 MB/s; 当内核运行 LTP时, FTP性能为 2.4MB/s, 双态间应用通信正常。实验结果表明, KLinux能够支持双态应用开发,允许实时与非实时混合应用共存。

4 结束语

为了满足更广泛的应用需求,在深入研究当前主流实时 Linux的基础上,提出新的 Linux内核构架,实时调度策略和错误处理机制,并最终实现了 Linux内核的混合实时性改造。实验表明, KLinux能够为应用提供实时特性,支持内核/用户双态应用开发,错误处理机制也为系统稳定运行提供了保障。目前, KLinux已经成功应用于 ADSL局端、数据路由器等多款电信产品系统上,并取得了良好的效果。

参考文献 (References)

[1] CHEN Yimin, CHEN Yangbin. Research reform on real-time operating system based on Linux [C]// Proc of the 5th World Congress on Intelligent Control and Automation. Hangzhou, China: IEEE Press, 2004: 3916 - 3920.

[2] 王继刚, 顾国昌, 徐立峰, 等. 强实时性 Linux内核的研究与设计 [J]. 系统工程与电子技术, 2006, **28**(12): 1932 - 1935.
WANG Jigang, GU Guochang, XU Lifeng, et al. Research and design of hard real-time Linux kernel [J]. *Systems Engineering and Electronics*, 2006, **28**(12): 1932 - 1935. (in Chinese)

[3] Daniel P Bovet, Marco Cesati. Understanding the Linux Kernel [M]. 3th Ed. USA: O' Reilly Media, 2005.

[4] HALL C E. A real-time linux system for autonomous navigation and flight attitude control of an uninhabited aerial vehicle [C]// Proceedings of the 20th Digital Avionics Systems Conference Proceedings. Daytona Beach, FL, USA: IEEE Press, 2001: 1A11 - 1A19.

[5] ZHANG Guoyin, CHEN Luyuan, YAO Aihong. Study and comparison of the RTHAL-based and ADEOS-based RTAI real-time solutions for Linux [C]// Proceeding of the First International Multi-Symposiums on Computer and Computational Sciences. Hangzhou, China: IEEE Press, 2006: 771 - 775.

[6] Anand S V. Mobile terminal device: Present and future [C]// The 9th International Conference on telecommunications. Zagreb, Croatia: IEEE Press, 2007: 349 - 352.

[7] Srinivasan B, Hill R, Pather S, et al. KURT-Linux support for synchronous fine-grain distributed computations [C]// The 6th IEEE Real Time Technology and Application Symposium. Washington DC, 2000: 78 - 87.

[8] Lin K J, Wang Y C. The design and implementation of real-time schedulers in RED-Linux [C]// Proceeding IEEE Real-Time Technology and Application Symp (RTAS). 2003: 1114 - 1130.

[9] 李小群, 赵慧斌, 叶以民, 等. RFRTOS: 基于 Linux的实时操作系统 [J]. 软件学报, 2003, **14**(7): 1203 - 1212.
LI Xiaoqun, ZHAO Huibin, YE Yimin, et al. RFRTOS: A real-time operation system based on Linux [J]. *Journal of Software*, 2003, **14**(7): 1203 - 1212. (in Chinese)