DOI:10.19304/j.cnki.issn1000-7180.2014.08.010

Linux实时内存的研究与实现

田 泉,艾丽蓉,陈 杰
(西北工业大学 计算机学院,陕西 西安 710129)

摘 要: Linux系统采用了虚拟存储技术,当请求的页面不在内存中时触发缺页中断,由此带来的延迟不确定,故不能满足实时应用程序的要求.此外,对于用户态和内核态存在大量数据传输的情况下,通用Linux系统也不能满足实时应用程序的需求.针对以上问题,讨论了Linux的内存管理,并采用内存映射技术来解决虚拟内存的换页问题以及实现用户态和内核态共享一块物理内存来满足实时应用程序的需求.在文章的最后,测试和比较了采用内存映射技术实现实时内存的性能.测试结果表明,采用该技术可以有效地为实时应用程序提供实时内存.

关键词: 内存映射;实时;缺页中断;虚拟内存
中图分类号: TP316.2 **文献标识码**: A **文章编号**: 1000-7180(2014)08-0045-04

Research and Implementation of Real-Time Memory in Linux Operating System

TIAN Quan, AI Li-rong, CHEN Jie
(School of Computer, Northwestern Polytechniccal University, Xi'an 710129, China)

Abstract: Using virtual storage technology in Linux Operating System, a page fault is triggered when the requested page is not in memory, which results in uncertain delay that cannot meet the requirements of real-time application. In addition, with the case of large amounts of data transmission between user mode and kernel mode, the generic Linux system cannot meet the demand of real-time applications. To solve these problems, this paper studies the Linux memory management firstly, and then using memory-mapped technology to solve the page fault problem in virtual memory management and to share a physical memory for user mode and kernel mode, which aim at meeting the requirements of real-time application. In the last part, this study will test and compare the performance of the memory mapping techniques to achieve real-time memory. The test results show that the technology can provide real-time memory for real-time applications effectively.

Key words: memory mapping; real time; page fault; virtual memory

1 引言

Linux为每个进程维持了一个单独的虚拟地址空间,并且将地址空间划分为固定大小的页面[1].在进程执行的过程中,页面按需被调入物理内存中.由于系统的物理内存有限,故在进程占用空间很大或多进程的情况下,每个进程只有部分页面在物理内存中.而当发生缺页中断时,需要一系列的操作将新的页面调入物理内存,此过程导致的延迟不确定并且不可预测,这对于实时性要求高的任务来说往往是不能接受的.此外,用户空间不能也不应该操作内核空间,在通用系统中,通过系统调用来实现用户空间和内核空间数据交换,同样的这对于实时性要求高的任务来说不能很好地满足实时性[2].目前比较流行的Linux实时内存改造方案是通过优化内存的分配方案以尽量减少缺页中断的发生,而从根本上并不能保证实时程序的页面不被换出,因此也就不能满足实时性要求高的实时程序的性能需求.

收稿日期: 2013-11-03; 修回日期: 2013-12-22
基金项目: 国家基础预研项目(2011AC100001C100001)

针对以上问题本文通过内存映射技术结合内存锁定技术使实时进程使用的页面不被换出物理内存，以及使用户空间和内核空间映射到同一块物理内存，提高用户空间与内核空间交互的实时性[3-4].

2 虚拟内存

Linux 采用虚拟存储器系统，每个进程都有一个单独的虚拟地址空间. 对于 32 位系统，每个进程可用的虚拟内存可达 4 GB. 在该 4 GB 的虚拟空间中，一般被分为两部分：进程虚拟存储空间和内核虚拟空间，并且这两部分通常被映射到不同的物理页面. 图 1 所示为一个典型的 linux 进程的虚拟存储器.

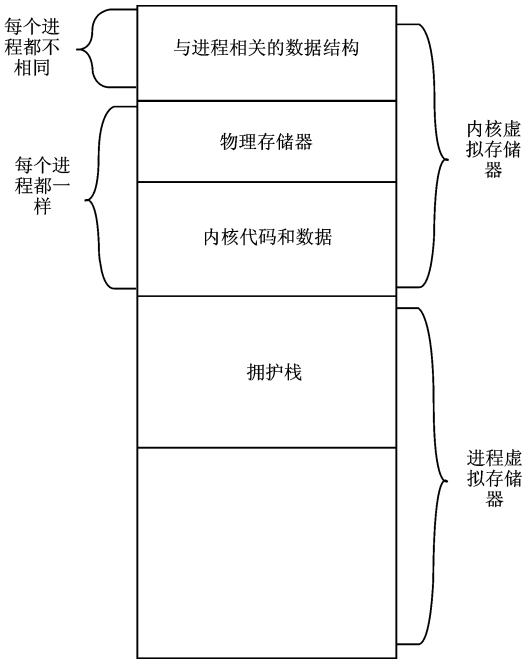

图 1 典型 linux 进程的虚拟存储器

一般地，用户地址空间分布在 0～3 GB 范围内，3～4 GB 为内核地址空间. 在内核地址空间中，内核地址空间被分为内核逻辑地址和内核虚拟地址. 在大多数体系结构中，内核逻辑地址与其相关联的物理地址之间仅相差一个固定的地址偏移量，与其相关联的物理内存称为低端内存. 此外，所有的逻辑地址都是内核虚拟地址，但是许多内核虚拟地址不是逻辑地址，即内核虚拟地址与物理地址的映射不必是线性的和一对一的，并且将不具有逻辑地址的内存称为高端内存.

在内核空间申请内存时，kmalloc 和 get_free_page 函数申请的物理内存处于低端内存中，即申请的内存的内核逻辑地址和物理地址之间存在一个固定的偏移量，因此存在着简单的转换关系，virt_to_phys 函数实现将内核逻辑地址转换为物理地址，同样的 phys_to_virt 函数实现将物理地址转换为内核逻辑地址.

3 内存映射

现有的 Linux 实时化改造主要有两个途径：直接修改标准的 Linux 内核[5]，增强其实时性能；通过扩展标准的 Linux 内核，以可加载模块的形式实现一个实时的微内核加载到系统中. 本文采用可加载的实时微内核方式，该方式不但保证了原有系统内核的稳定性与独立性，同时也保证了可加载实时模块的灵活性与可移植性.

3.1 内存映射基本原理

内存映射主要思想就是将进程虚拟空间和内核虚拟空间映射到同一个物理页面，实现用户态和内核态共享内存. 映射成功后，用户对这段物理内存的操作就会直接反应到内核空间，相反，内核空间对这段物理内存的操作也同样会反应到用户空间. 图 2 为内存映射原理图[6-7].

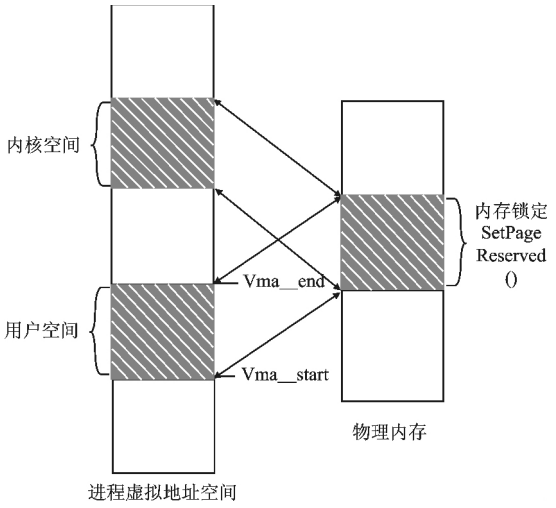

图 2 内存映射原理图

在实时 linux 系统中，我们选择将实时任务运行在用户态并且支持实时任务在用户态的创建、挂起等一系列进程操作. 在对进程操作的同时需要修改处于内核态中与进程相关的数据结构，而在一般通用系统中这主要是通过系统调用来进行，但对于实时任务来说系统调用的时间会使系统的实时性能降低. 针对此问题我们采用内存映射的方法使内核态和用户态共享同一块物理内存，达到无需系统调用的时间开销，最终来提高系统的实时性.

本文主要通过两部分来完成内存映射. 第一部

分通过共享库的形式来实现，导出 RTmalloc 和 RTfree 接口供实时应用程序使用，它们的作用主要是申请和释放实时内存. 第二部分是以 linux 设备驱动的形式实现，负责响应 RTmalloc 和 RTfree 提出的申请和释放实时内存的请求，完成内存映射及解除映射操作.

3.2　基本函数及实现

mmap 是 linux 下的系统调用，该系统调用并不是完全为了用于共享内存而设计的. 它提供了不同于一般对普通文件的访问方式，进程可以像读写内存一样对普通文件进行操作. mmap 系统调用使得进程之间通过映射同一个普通文件实现共享内存. 普通文件被映射到进程地址空间后，进程可以像访问普通内存一样对文件进行访问，不必再调用 read，write 等操作. mmap 并不分配空间，只是将文件映射到调用进程的地址空间里（会占掉虚拟地址空间），然后就可以用 memcpy 等操作写文件，而不用 write 了.

本文运用 mmap 系统调用来实现内存映射，完成用户态和内核态共享内存. 首先，驱动程序通过调用 Kmalloc 函数申请一块低端物理内存，并且将申请到的物理内存用 SetPageReserved 函数标记为“保留的（reserved）”，表示虚拟内存管理对其不起作用. 保留页在内存中被锁住，其将不会被换出，并且可以被安全的映射到用户空间. 在申请的物理内存被标记为“保留”后，用户进程通过库函数 mmap 来与内核通信以通知内核需将多大的内存来进行映射. 内核经过一系列函数调用后调用对应的驱动程序的 file_operation 中指定的 mmap 函数.

对于 file_operation 中的 mmap 函数，首先用 virt_to_phys 函数得到 Kmalloc 函数申请的内存的物理地址，然后 remap_pfn_range 函数运用得到的物理地址来完成该块物理内存与用户空间的某块区域的映射，实际上函数 remap_pfn_range 是完成该块物理内存页表的建立. 当完成映射后，内核和用户程序就可以直接向这块物理内存读写数据，当实时应用程序不再需要这块内存时，驱动程序调用 ClearPageReserved 函数来使这块内存不再为“保留”，接着调用 Kfree 函数来释放这块内存空间，至此这块内存又恢复了通用.

4　性能测试

本测试在处理器为 Intel(R) Core(TM)2 Duo，频率为 2.99 GHz，物理内存为 2 GB，操作系统为 linux Centos 6.3 的平台上进行.

在本次测试中，为了防止线程调度带来的性能影响，首先通过设置线程的亲缘性将测试程序绑定到 1 号核，其他的线程绑定到 0 号核. 并且为了营造内存紧张的局面，运行一个消耗内存的程序，使内存使用率达到 90%左右. 测试程序申请一定大小的实时内存空间，并且向这块内存中写入数据，最后将写入的数据读取到文件中. 记录这个操作过程所需的时间. 于此相对应，用一非实时程序进行同样的操作，由于内存使用率达到 90%左右，对采用常规方法申请的内存进行写操作时，会频繁的进行换页操作，而对实时内存进行操作时，不会引起换页，所用时间少，且波动小.

测试结果如图 3、图 4 和图 5 所示，横坐标表示测试周期，纵坐标表示测试所用的时间，单位为微秒. 从图中可以看出采用实时内存进行内核空间与用户空间大量数据交互时要比非实时内存，也即通用内存的效率高，并且当实时任务申请的内存块越大、数据交互量越大时实时内存的优势表现的越明显.

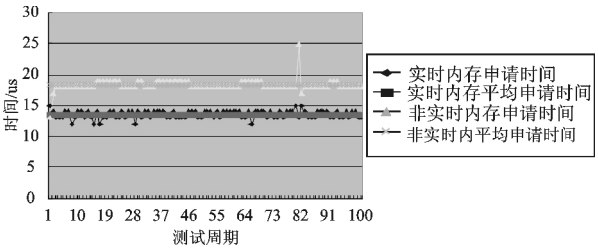

图 3　申请 4 kb 内存效率对比图

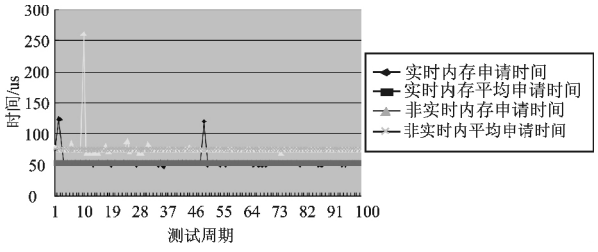

图 4　申请 16 kb 内存效率对比图

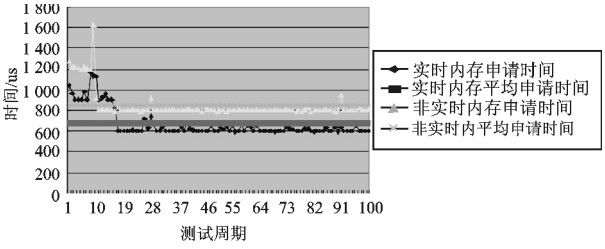

图 5　申请 256 kb 内存效率对比图

表 1 和表 2 是对实时内存和非实时内存效率的时间数据比较，时间单位为 ms. 表 1 针对通用内存

申请和实时内存申请进行了效率比较，结果表明总体上实时内存的申请速度和普通内存的申请效率较接近. 表 2 是针对于不同的读写数据量进行了效率比较，结果表明对于用户空间和内核空间存在大数据量的数据交换时实时内存的效率优于非实时内存的效率[8-9].

表 1 实时内存申请平均时间

	1 kb	4 kb	16 kb	64 kb	256 kb
非实时内存	0.2	0.2	0.3	0.5	0.7
实时内存	0.3	0.2	0.6	0.6	0.9

表 2 实时内存与非实时内存存储效率比较

	1 kb	4 kb	16 kb	64 kb	256 kb
非实时内存	5.2	18.3	75.5	250.4	843.5
实时内存	5.9	13.5	52.9	191.2	671.9

5 结束语

从测试结果可以看出来，内存映射技术解决了采用虚拟内存带来的缺页中断问题，并且有效的使用户空间和内核空间指向了同一块物理内存，使用户空间和内核空间之间大量数据交换的效率得到了显著地提高. 以内存映射为基础，实时的内存管理算法为核心实现一套实时化的内存管理机制以模块的方式加载到内核中，这将是进一步研究的方向与重点.

参考文献：

[1] 赖娟. Linux 内核分析及实时性改造[D]. 成都：电子科技大学 2007.

[2] 刘胜，王丽芳，蒋泽军. 基于多核 PC 的 Linux 系统实时性改造[J]. 微电子学与计算机，2013，30(8)：120-123.

[3] Wang Lixin，Kang Jing. MMAP system transfer in linux virtual memory management[C]// 2009 First International Workshop on Education Technology and Computer Science (ETCS). China：Wuhan，2009：673-677.

[4] Hong Xu，Rong Tang. Study and improvements for the real-time performance of Linux kernel[C]// Biomedical Engineering and Informatics (BMEI)，2010 3rd International Conference on，China：Yantai，2010(7)：2766-2769.

[5] Robert Love. Linux 内核设计与实现[M]. 3 版. 陈莉君，康华，译，北京：机械工业出版社，2011.

[6] Rubini A，Corbet J. Linux 设备驱动程序[J]. 3 版. 北京：中国电力出版社，2006.

[7] Bryant R E，奥哈洛伦. 深入理解计算机系统[M]. 北京：中国电力出版社，2004.

[8] 张立辉，赵云忠，王建生，等. 基于嵌入式 Linux 的实时性分析[J]. 微电子学与计算机，2007，24(6)：100-103.

[9] 张新村. 基于 RTAI 的实时 Linux 系统的研究[D]. 四川：西南科技大学，2012.

作者简介：

田 泉 男，(1990-)，硕士研究生. 研究方向为分布式实时嵌入式系统；

艾丽蓉 女，副教授，硕士导师. 研究方向为智能信息处理与 Web 安全技术；

陈 杰 男，(1988-)，硕士研究生. 研究方向嵌入式计算与应用.

(上接第 44 页)

算法比 MWBC 算法[2]更具优势，选出的簇头 Agent 更加合理，减少了网络的能量消耗，延长整个网络的生命周期.

参考文献：

[1] 邓夏阳，黄杰. LEACH 算法最优数据采集方案[J]. 东南大学学报，2012，42(1)：20-24.

[2] Younis Ossama，Fahmy Sonia. HEED：a hybrid，energy-efficient，distributed clustering approach for Ad hoc sensor networks[J]. IEEE Trans. On Mobile Computing，2004，3(4)：660-669.

[3] 黄河清，姚道远. 一种基于多权值优化的无线传感器网分簇算法研究[J]. 电子与信息学报，2008，30(6)：1489-1492.

[4] Dimokas N，Katsaros D，Manolopoulos Y. Energy—efficient distributed clustering in wireless sensor networks[J]. Journal of Parallel and Distributed Computing，2010，70(4)：371-383.

[5] 石为人，柏荡，等无线传感器网络簇头半径自适应调节路由算法[J]. 仪器仪表学报，2012，33(8)：1779-1784.

作者简介：

刘 佳 女，(1988-)，硕士研究生. 研究方向为多智能体技术、无线传感网络拓扑控制.