第一图书网, tushu007.com

<<Linux内核设计的艺术>>

图书基本信息

书名：<<Linux内核设计的艺术>>

13位ISBN编号：9787111347446

10位ISBN编号：7111347447

出版时间：2011-6-20

出版时间：机械工业出版社华章公司

作者：新设计团队

页数：444

版权说明：本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问：http://www.tushu007.com

前言

很早就有一个想法，做中国人自己的、有所突破、有所创新的操作系统、计算机语言及编译平台。
我带领的“新设计团队”（主要由中国科学院研究生院已毕业的学生组成）在实际开发自己的操作系统的过程中，最先遇到的问题就是如何培养学生真正看懂Linux操作系统的源代码的能力。
开源的Linux操作系统的源代码很容易找到，但很快就会发现，培养学生看懂Linux操作系统的源代码是一件非常非常困难的事。
操作系统的代码量通常都是非常庞大的，动辄几百万行，即使是浏览一遍也要很长时间。
比庞大的代码量更让学习者绝望的是操作系统有着极其错综复杂的关系。
看上去，代码的执行序时隐时现，很难抓住脉络。
代码之间相互牵扯，相互勾连，几乎无法理出头绪。
更谈不上理解代码背后的原理、意图和思想。
对于学生而言，选择从源代码的什么地方开始分析本身就是一个难题。
通常，学生有两种选择：一种是从main函数，也就是从C语言代码的总入口开始，沿着源代码的调用路线一行一行地看下去，学生很快就会发现源代码的调用路线莫名其妙地断了，但直觉和常识告诉他操作系统肯定不会在这个地方停止，一定还在继续运行，但却不知道后续的代码在哪里，这种方法很快就走进了死胡同；另一种则是从某一模块入手，如文件系统，但这样会无形中切断操作系统源码之间复杂的关系，如文件系统与进程管理的关系，文件系统与内存管理的关系，等等。
使学生孤立地去理解一个模块，往往只能记住一些名词和简单概念，难以真正理解操作系统的全貌。
用学生的话讲，他们理解的操作系统变成了“文科”的操作系统。
由于操作系统是底层系统程序，对应用程序行之有效的调试和跟踪等手段对操作系统的源代码而言，几乎无效。
就算把每一行源代码都看懂了，对源代码已经烂熟于心，知道这一行是一个for循环，那一行是一个调用……但仍然不知道整个代码究竟在做什么，以及起什么作用，更不知道设计者的意图究竟是什么。
我们在操作系统的课程上学习过进程管理、内存管理、文件系统等基础知识，但是这些空洞的理论在一个实际的操作系统中是如何实现的却不得而知。
他们在源代码中很难看出进程和内存之间有什么关联，内核程序和用户程序有什么区别，为什么要有这些区别。
也很难从源代码中看清楚，我们实际经常用到的操作，比如打开文件，操作系统在其中都做了哪些具体的工作？
想在与常见的应用程序的编程方法有巨大差异的、晦涩难懂的、浩瀚如海的操作系统底层源代码中找到这些问题的答案，似乎比登天还难。
对熟悉操作系统源代码的学生而言，他们也知道像分页机制这样知识点，但是未必能够真正理解隐藏在机制背后的深刻意义。
这些都是学生在学习Linux操作系统源代码时遇到的实际问题。
中国科学院研究生院的学生应该是年轻人中的佼佼者，他们遇到的问题可能其他读者也会遇到。
我萌发了一个想法，虽然学生的问题早已解决，但是否可以把他们曾经在学习、研发操作系统的过程中遇到的问题和心得体会得拿出来供广大读者分享？
当时，针对学生的实际问题，我的解决方法是以一个真实的操作系统为例，让学生理解源代码，把操作系统在内存中的运行时状态画出图来。
实践证明，这个方法简单有效。
现在我们把这个解决方案体现在这本书中。
就是以一个真实的操作系统的实际运行为主线；以图形、图像为核心，突出描述操作系统在实际运行过程中内存的运行时结构；强调站在操作系统设计者的视角，用体系的思想方法，整体把握操作系统的行为、作用、目的和意义。
在全书的讲解过程中，我们不仅详细分析了源代码、分析了操作系统的执行序，我们还特别分析了操作系统都作了哪些“事”，并且把“事”与“事”之间的关系和来龙去脉，这些“事”意味着什么，

为什么要做这些“事”，这些“事”背后的设计思想是什么……都做了非常详细且深入的分析。
更重要的是，对于所有重要的阶段，我们几乎都用图解的方式把操作系统在内存中的实际运行状态精确地表示了出来。
我们用600dpi的分辨率精心绘制了349张图，图中表现的运行时结构和状态与操作系统实际运行的真实状态完全吻合。
每一条线、每一个色块、每一个位置、每一个地址及数字都经过了我们认真反复地推演和求证，并最终在计算机上进行了核对和验证。
看了这些绘制精美的图后，在读者的头脑中就不再是一行行、一段段枯燥的、令人眩晕的源代码，而是立体呈现的一件件清晰的“事”，以及这些“事”在内存中直截了当、清晰鲜活的画面。
用这样的方法讲解操作系统是本书的一大特色。
理解这些图要比理解源代码和文字容易得多，毫不夸张地说，只要你能理解这些图，你就理解了操作系统的80%，这时你可以自豪的说，你比大多数用别的方法学过操作系统的人的水平都要高出一大截。
作者和机械工业出版社的编辑作了大量的检索工作，就我们检索的范围而言，这样的创作方法及具有这样特色的操作系统专著在世界范围都是第一次。
我们分三个部分来讲解Linux操作系统：第一部分由第1章和第2章组成，分析了从开机加电到操作系统启动完成并进入怠速状态的整个过程；第二部分由第3章、第4章、第5章、第6章、第7章组成，讲述了操作系统进入系统怠速后，在执行用户程序的过程中，操作系统和用户进程的实际运行过程和状态；第三部分由第8章组成，阐述整个Linux操作系统的设计指导思想，本章内容是从微观到宏观的回归。
第一部分，我们详细讲解了开机加电启动BIOS，通过BIOS加载操作系统程序，对主机的初始化，打开保护模式和分页，调用main函数，创建进程0、进程1、进程2以及shell进程，并且具备用文件的形式与外设交互。
第二部分，我们设计了几个尽可能简单又有代表性的应用程序，并以这些程序的执行为引导，详细讲解了安装文件系统、文件操作、用户进程与内存管理、多个进程对文件的操作以及进程间通信。
我们将操作系统的原理自然而然地融入到了讲解真实操作系统的实际运行过程中。
在读者看来，操作系统原理不再是空对空的、“文科”概念的计算机理论，而是既有完整的、体系的理论，又有真实、具体、实际的代码和案例，理论与实际紧密耦合。
第三部分是全书水平最高的部分，我们尝试从操作系统设计者的视角讲解操作系统的设计指导思想。
详细阐述了主奴机制以及实现主奴机制的三项关键技术：保护和分页、特权级、中断，分析了保障主奴机制实现的决定性因素——先机，还详细讲解了缓冲区、共享页面、信号、管道的设计指导思想。
希望帮助读者用体系的思想理解、把握、驾驭整个操作系统以及背后的设计思想和设计意图。
在本书中，我们详细讲解了大家在学习操作系统的过程中可能会遇到的每一个难点，如main函数中的pause()调用，虽然已经找不到后续代码，但该调用结束后，程序仍然执行的原因是：中断一经打开，进程调度就开始了，而此时可以调度的进程只有进程1，所以后续的代码应该从进程1处继续执行……我们还对一些读者不容易理解和掌握的操作系统特有的底层代码的编程技巧作了详细的讲解。
如用模拟call的方法，通过ret指令“调用”main函数……总之，我们所做的一切努力就是想真正解决读者遇到的实际问题和难题，给予读者有效的帮助。
我们盼望即使是刚刚考入大学的学生也有兴趣和信心把这本书读下去；我们同样希望即使是对操作系统源代码很熟悉的读者，这本书也能给他们一些不同的视角、方法和体系性思考。
这本书选用的操作系统源代码是Linux 0.11。
对为什么选用Linux 0.11而不是最新版本，赵炯先生有过非常精彩的论述，我们认为赵先生的论述是非常到位的。
我们不妨看一下Linux最新的版本2.6，代码量大约在千万行这个量级，去掉其中的驱动部分，代码量仍在百万行这个量级。
一个人一秒钟看一行，一天看8小时，中间不吃、不喝、不休息，也要看上几个月，很难想象如何去理解。
就算我们硬要选用Linux 2.6，就算我们写上3000页（书足足会有十几厘米厚），所有的篇幅都用来印

代码，也只能印上不到十分之一的代码。
所以，即使是这么不切实际的篇幅，也不可能整体讲解Linux 2.6。
读者会逐渐明白，对于理解和掌握操作系统而言，真正有价值的是整体、是体系，而不是局部。
Linux 0.11的内核代码虽然只有约两万行，但却是一个实实在在、不折不扣的现代操作系统，因为它具有现代操作系统最重要的特征——支持实时多任务，所以必然支持保护和分页……而且它还是后续版本的真正的始祖，有着内在的、紧密的传承关系。
读者更容易看清设计者最初的、最根本的设计意图和设计指导思想。
Linux 0.11已经问世20年了，被世人广为研究和学习。
换一个角度看，要想在众人熟悉的事物和领域讲出新意和特色，对作者来说也是一个强有力的挑战。
这本书能够顺利出版，我们首先要感谢机械工业出版社华章公司的副总经理温莉芳女士以及其他领导，是他们的决心和决策成就了这本书，并且在几乎所有方面给予了强有力的支持。
特别令人感动的是他们主动承担了全部的出版风险，同时给予了作者最好的条件，让我们看到一个大出版社的气度和风范。
其次，我们还要感谢的是机械工业出版社华章公司的编辑杨福川。
杨先生的鉴赏力和他的事业心以及他对工作认真负责的态度为这本书的出版打开了大门。
杨先生对读者的理解以及他的计算机专业素养使得他有能力对这本书给予全方位的指导和帮助，使我们对这本书整体修改了6次，使之更贴近读者，可读性更好。
我们还要感谢我们和杨福川先生共同的朋友张国强先生和杨缙女士。
最后，我们要感谢我们的家人和朋友。
是他们坚定的支持才使得我们的整个团队能够拒绝方方面面、形形色色的诱惑，放弃普遍追求的短期利益，在常人难以想象的艰苦条件下，长时间专注于操作系统、计算机语言、编译器、计算机体系结构等基础性学科的研究。
认认真真、踏踏实实、不为名利，做了一点实在、深入的工作，有了一点能凑合拿得出手的10年积累，以及一支敢想、敢干、敢打、敢拼、不惧世界顶级强敌的队伍。
这些是本书的基础。
杨力祥中国科学院研究生院2011年5月

内容概要

关于Linux内核的书已经不计其数，但这本书却是独树一帜的，它的内容代表着Linux内核研究成果的世界顶尖级水平，它在世界范围内首次提出并阐述了操作系统设计的核心指导思想——主奴机制，这是所有操作系统研究者的一笔宝贵财富。
本书可能也代表着同类图书的顶尖水平，是一本真正能引导我们较为容易地、极为透彻地理解Linux内核的经典之作，也可能是当前唯一能从本质上指引我们去设计和开发拥有自主知识产权的操作系统的著作。
它的出版也许会成为Linux内核研究领域的一个里程碑事件。

本书的最大特点是它的写作方式和内容组织方式，与同类书完全不同。
它在深刻地分析了传统讲解方法的利弊之后，破旧立新，从认知学的角度开创了一种全新的方式。
以操作系统的真实运行过程为主线，结合真实的内核源代码、349幅精确的内核运行时序图和具有点睛之妙的文字说明，对操作系统从开机加电到系统完全准备就绪的整个过程进行了系统而完整地分析，深刻地揭示了其间每一个动作的设计意图和实现原理，完美地再现了操作系统设计者的设计思路。
阅读本书就如同跟随着操作系统设计者一起去思考，我们会在阅读的过程中发现Linux内核设计的精妙，会发现原来处处都“暗藏玄机”，哪怕是一行很短的代码。

本书在所有细节上都力求完美。
为了保证知识的准确性，操作系统运行过程中的每个动作都经过了严格的考证；为了让我们真正理解Linux内核的原理，它突破传统，以Linux的真实运行过程为主线进行讲解；为了做到真正易于理解，创新性地使用了图解的方式，精心绘制了349幅分辨率600dpi的时序图，图中表现的运行时结构和状态与操作系统实际运行时的真实状态完全吻合；为了提高阅读体验，本书采用了双色印刷，以便于我们更清楚地观察每一幅图中的细节。

作者简介

新设计团队始于世纪之交，不断发展、优胜劣汰、适者生存、自然形成。
团队始终不自量力地奉行高举高打的策略，只对计算机领域中基础的、有体系的事情感兴趣，且只做自己感兴趣的事。
团队不相信二流水平、三流心态，能作出世界顶级的工作。
团队相信成功的决定因素是人，而不是物。
团队对艺术、对鉴赏力、对欣赏品味、对一切美好的事物都有着近乎狂热的追求。
认为科学、技术的最高境界是艺术，认为世界的本源是通的。
团队崇尚理论体系、崇尚个性鲜明、崇尚独立思考。
“没体系”是团队成员之间善意贬损的常用语，也是判断一件事是否值得关注的标准之一。
团队鄙视抄袭、弄虚作假。
对别人热炒、做熟的事情不感兴趣，更不喜欢在别人的体系上狗尾续貂、移花接木，粉饰为“自主创新”。
团队强调理论，注重实践，讲究科学的研究方法，不屈膝权威，不迷信盲从。
提出基础假设，构建逻辑自洽的体系，证伪、修正、再证伪、再修正，不断推进体系的完善。
团队钻研学术，但决不死板，学术和商业结合，彼此互为推进，最终改变整个时代的商业格局，是团队追求的目标。
团队特别注重诚信，提倡公平、公正。
团队内部是一方净土，既相互竞争，又相互帮助、相互协作。
团队成员都在高高兴兴做自己喜欢的、感兴趣的事，没有时间顾及其他。
勾心斗角、尔虞我诈、溜须拍马、拉帮结派……，在团队内部没有市场。
团队在上述思想的指导下，研发了基于图形、图像（而非基于字符、语句）的图示化程序设计集成开发环境，已成功的移植了linux 0.11，正确编译，正确boot，正确运行。
现在正在研发新的操作系统，已初步实现了与现有的基于块概念的文件系统有较大差异的新的文件系统，据我们测试，相对于基于块设备的文件系统，在文件的读写速度上有较大优势。
《Linux内核设计的艺术》一书，体现了我们设计新操作系统的过程中，对操作系统的理解。
可能在不久的将来，我们也将我们在设计图示化程序设计平台中，对编译原理的理解，奉献给广大读者。

书籍目录

前言
第1章　从开机加电到执行main函数之前的过程1
1.1　启动BIOS，准备实模式下的中断向量表和中断服务程序1
1.1.1　BIOS的启动原理2
1.1.2　BIOS在内存中加载中断向量表和中断服务程序3
1.2　加载操作系统内核程序并为保护模式做准备4
1.2.1　加载第一部分代码—引导程序（bootsect）5
1.2.2　加载第二部分代码—setup7
1.2.3　加载第三部分代码—system模块12
1.3　开始向32位模式转变，为main函数的调用做准备16
1.3.1　关中断并将system移动到内存地址起始位置0x0000016
1.3.2　设置中断描述符表和全局描述符表18
1.3.3　打开A20，实现32位寻址20
1.3.4　为在保护模式下执行head.s做准备21
1.3.5　head.s开始执行24
1.4　本章小结41
第2章　从main到怠速42
2.1　开中断之前的准备工作43
2.1.1　复制根设备号和硬盘参数表 44
2.1.2　物理内存规划格局45
2.1.3　虚拟盘设置与初始化46
2.1.4　内存管理结构mem_map初始化47
2.1.5　异常处理类中断服务程序挂接48
2.1.6　初始化块设备请求项结构50
2.1.7　与建立人机交互界面相关的外设的中断服务程序挂接52
2.1.8　开机启动时间设置55
2.1.9　系统开始激活进程056
2.1.10　进程相关事务初始化设置57
2.1.11　时钟中断设置59
2.1.12　系统调用服务程序挂接59
2.1.13　初始化缓冲区管理结构61
2.1.14　初始化硬盘63
2.1.15　初始化软盘65
2.1.16　开中断66
2.2　进程创建的最基本动作67
2.2.1　操作系统为进程0创建进程1做准备67
2.2.2　在进程槽中为进程1 申请一个空闲位置并获取进程号71
2.2.3　复制进程信息之前，先将一些数据压栈73
2.2.4　初步设置进程1管理结构74
2.2.5　进程0创建进程1的过程中发生时钟中断76
2.2.6　从时钟中断返回78
2.2.7　调整进程1管理结构79
2.2.8　设置进程1的线性地址空间及物理页面81
2.2.9　继续调整进程1管理结构84
2.2.10　操作系统如何区分进程0和进程187

2.2.11 进程0准备切换到进程189
2.2.12 系统切换到进程1执行90
2.3 加载根文件系统92
2.3.1 进程1如何开始执行96
2.3.2 进程1开始执行98
2.3.3 进程1开始以数据块的形式操作硬盘99
2.3.4 将找到的缓冲块与请求项挂接101
2.3.5 将请求项与硬盘处理函数挂接104
2.3.6 进行硬盘读盘前的准备工作105
2.3.7 给硬盘下达读盘指令106
2.3.8 进程1由于等待读盘操作挂起107
2.3.9 系统切换到进程0执行109
2.3.10 进程0的执行过程110
2.3.11 进程0执行过程中发生硬盘中断111
2.3.12 硬盘中断服务程序响应后，进程0继续执行113
2.3.13 再次响应硬盘中断并唤醒进程1114
2.3.14 读盘操作完成后，进程1继续执行116
2.3.15 进程1继续设置硬盘管理结构117
2.3.16 进程1获取软盘超级块，为加载根文件系统做准备118
2.3.17 进程1备份超级块数据119
2.3.18 进程1将根文件系统从软盘拷贝到虚拟盘120
2.3.19 进程1开始加载根文件系统122
2.3.20 进程1准备加载根文件系统超级块123
2.3.21 进程1加载根文件系统超级块124
2.3.22 进程1继续加载根文件系统126
2.3.23 进程1准备读取根目录i节点127
2.3.24 进程1加载根目录i节点128
2.3.25 进程1结束加载根文件系统的过程129
2.4 打开终端设备文件及复制文件句柄131
2.4.1 进程1与内核文件表挂接，为打开文件做准备133
2.4.2 确定打开操作的起点135
2.4.3 获得枝梢i节点—dev目录文件的i节点136
2.4.4 确定dev目录文件i节点为枝梢i节点137
2.4.5 继续返回枝梢i节点138
2.4.6 查找tty0文件的i节点138
2.4.7 将tty0设备文件的i节点返回给sys_open系统调用 139
2.4.8 分析tty0文件i节点140
2.4.9 设置文件管理结构并返回给用户进程141
2.4.10 进程1复制tty0文件句柄142
2.4.11 进程1继续复制tty0文件句柄144
2.5 创建进程2145
2.5.1 进程1准备创建进程2145
2.5.2 复制进程2管理结构并进行调整146
2.5.3 设置进程2的页目录项并复制进程2的页表146
2.5.4 调整进程2管理结构中与文件有关的内容146
2.5.5 进程1执行过程中发生时钟中断148
2.5.6 进程1从时钟中断返回，准备切换到进程2150

2.6　进程1等待进程2退出150
2.6.1　进程1查找它自己的子进程151
2.6.2　对进程2的状态进行处理151
2.6.3　切换到进程2执行153
2.7　shell程序的加载154
2.7.1　进程2开始执行156
2.7.2　为打开/etc/rc文件做准备156
2.7.3　进程2打开“ /etc/rc ”配置文件157
2.7.4　通过压栈为加载shell文件做准备158
2.7.5　为参数和环境变量设置做准备159
2.7.6　得到shell文件的i节点160
2.7.7　为加载参数和环境变量做准备161
2.7.8　根据i节点，对shell文件进行检测162
2.7.9　检测shell文件头163
2.7.10　备份文件头并进行分析163
2.7.11　对shell文件进行进一步分析165
2.7.12　拷贝参数和环境变量166
2.7.13　调整进程2的管理结构167
2.7.14　继续调整进程2管理结构168
2.7.15　释放进程2继承的页面169
2.7.16　检测协处理器170
2.7.17　调整shell程序所在的线性空间地址171
2.7.18　为shell程序准备参数和环境变量172
2.7.19　继续调整进程2管理结构173
2.7.20　调整EIP，使其指向shell程序入口地址173
2.7.21　shell程序执行引发缺页中断175
2.7.22　缺页中断中shell程序加载前的检测175
2.7.23　为即将载入的内容申请页面177
2.7.24　将shell程序载入新获得的页面177
2.7.25　根据shell程序的情况，调整页面的内容178
2.7.26　将线性地址空间与程序所在的物理页面对应179
2.8　系统实现怠速180
2.8.1　shell进程准备创建update进程180
2.8.2　进程2开始执行/etc/rc文件181
2.8.3　准备加载update进程181
2.8.4　update进程的作用182
2.8.5　shell程序检测“ /etc/rc ”文件183
2.8.6　shell进程退出184
2.8.7　shell进程退出善后处理185
2.8.8　进程1清理shell进程管理结构187
2.8.9　系统开始重建shell190
2.8.10　shell进程为何不会再次退出192
2.9　小结194
第3章　安装文件系统195
3.1　获取硬盘设备号196
3.1.1　用户发出安装硬盘文件系统指令196
3.1.2　从分析路径开始，准备查找hd1设备的挂接点197

3.1.3　以根目录i节点为依托，得到dev目录文件的i节点197
3.1.4　从dev目录文件中找到代表hd1设备文件的目录项198
3.1.5　得到hd1设备文件的i节点号199
3.1.6　释放dev目录文件的相关内容200
3.1.7　得到hd1设备文件的i节点200
3.1.8　获得hd1设备的设备号200
3.1.9　释放hd1设备文件的i节点201
3.2　获取虚拟盘上的挂接点202
3.3　得到hd1设备文件的超级块202
3.3.1　准备读取hd1设备文件超级块203
3.3.2　为hd1设备文件的超级块找到存储位置203
3.3.3　初始化空闲超级块并加锁203
3.3.4　从硬盘获得hd1设备文件的超级块204
3.3.5　加载逻辑块位图和i节点位图205
3.4　将hd1设备文件与mnt目录文件的i节点挂接206
3.5　小结207
第4章　文件操作208
4.1　打开文件211
4.1.1　用户程序调用open库函数产生软中断212
4.1.2　建立用户进程与文件管理表的关系213
4.1.3　从硬盘上获取helloc.txt文件的i节点214
4.1.4　将helloc.txt文件与文件管理表相挂接226
4.2　读文件227
4.2.1　为按照用户要求读入文件做准备228
4.2.2　确定要读入的数据块的位置230
4.2.3　将指定的数据块从硬盘读入到高速缓冲块233
4.2.4　将数据拷贝到用户指定的内存234
4.3　新建文件237
4.3.1　查找路径“ /mnt/user/hello.txt ”238
4.3.2　为hello.txt文件新建一个i节点240
4.3.3　为hello.txt文件新建目录项242
4.3.4　完成hello.txt新建操作并返回给用户进程245
4.4　写文件246
4.4.1　文件写入前的准备工作248
4.4.2　确定hello.txt文件的写入位置249
4.4.3　为数据的写入申请缓冲块252
4.4.4　将指定的写入数据从用户数据区拷贝到缓冲块253
4.4.5　数据同步到硬盘的方法1255
4.4.6　将文件写入硬盘的情况2257
4.5　修改文件260
4.5.1　对文件的当前操作指针进行重定位261
4.5.2　对文件进行修改261
4.6　关闭文件263
4.6.1　当前进程与文件管理表“ 脱钩 ”264
4.6.2　将文件管理表中hello.txt对应的引用次数减1265
4.6.3　hello.txt文件与文件管理表“ 脱钩 ”266
4.7　删除文件268

4.7.1　系统准备删除hello.txt文件268
4.7.2　删除hello.txt文件在硬盘上对应的数据和i节点270
4.7.3　对hello.txt文件所在的user目录做处理275
4.8　本章小结275
第5章　用户进程与内存管理277
5.1　用户进程的创建277
5.1.1　为创建进程str1准备条件277
5.1.2　为str1进程管理结构找到存储空间279
5.1.3　复制str1进程管理结构281
5.1.4　确定str1进程在线性空间中的位置282
5.1.5　复制str1进程页表并设置其对应的页目录项283
5.1.6　调整str1进程中与文件相关的结构285
5.1.7　建立str1进程与全局描述符表GDT的关联286
5.1.8　将str1进程设为就绪态287
5.2　为用户进程str1的加载做准备288
5.2.1　为str1进程加载自身对应的程序做准备288
5.2.2　读取str1可执行文件的i节点并统计参数和环境变量289
5.2.3　读取str1可执行文件的文件头290
5.2.4　对str1可执行程序文件头进行分析291
5.2.5　拷贝str1可执行程序的参数和环境变量292
5.2.6　调整str1进程管理结构中可执行程序对应的i节点292
5.2.7　继续调整str1进程管理结构—文件和信号相关的字段293
5.2.8　释放str1进程的页表294
5.2.9　重新设置str1的程序代码段和数据段295
5.2.10　创建环境变量和参数指针表296
5.2.11　继续根据str1可执行程序情况调整str1进程管理结构297
5.2.12　设置str1可执行程序的栈指针和eip值297
5.3　对缺页中断的处理298
5.3.1　产生缺页中断并由操作系统响应298
5.3.2　为str1程序申请一个内存页面299
5.3.3　将str1程序加载到新分配的页面中300
5.3.4　检测是否需要对页面剩余空间清0300
5.3.5　将str1程序占用的物理内存地址与str1进程的线性地址空间对应301
5.3.6　不断通过缺页中断加载str1程序的全部内容301
5.3.7　str1程序需要压栈302
5.3.8　str1程序第一次调用foo程序压栈302
5.3.9　str1程序第二次压栈，产生缺页中断302
5.3.10　处理str1程序第二次压栈产生的缺页中断302
5.3.11　str1程序继续执行，反复压栈并产生缺页中断303
5.3.12　str1程序运行结束后清栈303
5.4　str1用户进程的退出305
5.4.1　str1进程准备退出305
5.4.2　释放str1程序所占页面305
5.4.3　解除str1程序与文件有关的内容并给父进程发信号306
5.4.4　str1程序退出后执行进程调度307
5.5　多个用户进程“同时”运行308
5.5.1　依次创建str1、str2和str3进程308

5.5.2 str1进程压栈的执行效果309
5.5.3 str1运行过程中产生时钟中断并切换到str2执行309
5.5.4 str2执行过程遇到时钟中断切换到str3执行310
5.5.5 三个程序执行一段时间后在主内存的分布格局311
5.6 进程的调度与切换311
5.6.1 str1刚被shell创建并处于就绪态311
5.6.2 shell进程将自己挂起,然后准备切换到str1执行311
5.6.3 准备切换到str1进程执行312
5.6.4 str1执行时发生时钟中断314
5.6.5 时钟中断递减str1运行的时间片315
5.6.6 str1执行一段时间后挂起，shell进程新建str2进程315
5.6.7 str2运行期间发生时钟中断316
5.6.8 系统切换到str1程序执行317
5.7 内核的分页318
5.7.1 为设置内核的页目录表和页表做准备—所占空间清0318
5.7.2 设置内核对应的页目录项和页表项的内容319
5.7.3 设置内核对应的全局描述符表GDT320
5.8 页写保护321
5.8.1 进程A和进程B共享页面321
5.8.2 进程A准备进行压栈操作322
5.8.3 进程A的压栈动作引发页写保护322
5.8.4 将进程A的页表指向新申请的页面323
5.8.5 拷贝原页面内容到进程A新申请的页面324
5.8.6 进程B准备操作共享页面325
5.8.7 假设进程B先执行压栈操作的情况325
5.9 小结326
第6章 多个进程“同时”操作一个文件327
6.1 三个进程操作同一个文件327
6.1.1 进程A执行，hello.txt文件被打开328
6.1.2 进程A读取hello.txt文件并由于等待硬盘中断而被系统挂起328
6.1.3 进程B准备打开hello.txt文件330
6.1.4 系统准备为进程B获取hello.txt文件的i节点332
6.1.5 系统找到hello.txt文件已经载入的i节点333
6.1.6 系统准备为进程B从硬盘上读取hello.txt文件334
6.1.7 系统找到了正在操作的缓冲块，将进程B挂起335
6.1.8 系统再次切换到进程0执行337
6.1.9 进程C启动并打开hello.txt文件337
6.1.10 进程C也由于等待缓冲块解锁而被系统挂起338
6.1.11 缓冲块解锁后先唤醒进程C339
6.1.12 系统将进程B设为就绪状态340
6.1.13 系统将指定数据写入缓冲块341
6.1.14 写入完成后，进程C继续执行341
6.1.15 进程C准备切换到进程B342
6.1.16 进程C切换到进程B执行，进程B唤醒进程A342
6.1.17 进程B不断执行，直到时间片减为0后切换到进程A执行343
6.1.18 进程A、B、C退出，写入数据由update进程同步344
6.2 缓冲区与外设的数据同步344

6.2.1　系统不断为进程A向缓冲区写入数据346
6.2.2　继续执行引发缓冲块数据需要同步346
6.2.3　将缓冲区中的数据同步到硬盘上347
6.2.4　进程A由于等待空闲请求项而被系统挂起349
6.2.5　进程B开始执行350
6.2.6　进程B也被挂起351
6.2.7　进程C开始执行并随后被挂起352
6.2.8　进程A和进程C均被唤醒352
6.2.9　进程B切换到进程A执行354
6.3　小结356
第7章　IPC问题358
7.1　管道机制358
7.1.1　为管道文件在文件管理表中申请空闲项360
7.1.2　为管道文件与进程建立联系创造条件360
7.1.3　创建管道文件i节点361
7.1.4　将管道文件i节点与文件管理表建立联系362
7.1.5　将管道文件句柄返回给用户进程363
7.1.6　读管道进程开始操作管道文件363
7.1.7　写管道进程向管道中写入数据364
7.1.8　写管道进程继续向管道写入数据366
7.1.9　写管道进程已将管道空间写满366
7.1.10　写管道进程挂起366
7.1.11　读管道进程从管道中读出数据367
7.1.12　读管道进程继续执行，不断从管道中读出数据369
7.1.13　读管道进程执行中发生时钟中断369
7.1.14　读管道进程执行过程中再次发生时钟中断370
7.1.15　读管道进程切换到写管道进程执行371
7.1.16　写管道进程挂起切换到读管道进程执行371
7.1.17　读管道进程继续执行，直到把管道中的数据读完372
7.1.18　读取完成后，读进程挂起，写进程继续执行373
7.2　信号机制374
7.2.1　processig进程开始执行376
7.2.2　processig进程进入可中断等待状态377
7.2.3　sendsig进程开始执行并向processig进程发信号379
7.2.4　系统检测当前进程接收到信号并准备处理381
7.2.5　系统检测信号处理函数指针挂接是否正常382
7.2.6　调整processig进程的内核栈结构，使之先执行信号处理函数383
7.2.7　信号对进程执行状态的影响386
7.3　小结393
第8章　操作系统的设计指导思想395
8.1　运行一个最简单的程序，看操作系统为程序运行做了哪些工作395
8.2　操作系统的设计指导思想—主奴机制398
8.2.1　主奴机制中的进程及进程创建机制399
8.2.2　操作系统在内存管理中的主奴机制400
8.2.3　操作系统在文件系统中体现的主奴机制401
8.3　实现主奴机制的三种关键技术402
8.3.1　保护和分页402

8.3.2　特权级405
8.3.3　中断405
8.4　建立主奴机制的决定性因素—先机407
8.5　软件和硬件的关系：主机与进程、外设与文件408
8.5.1　非用户进程—进程0、进程1、shell进程408
8.5.2　文件与数据存储409
8.6　父子进程共享页面414
8.7　操作系统的全局中断与进程的局部中断—信号414
8.8　小结415
结束语415
“新设计团队”简介416
附录　搭建Linux 0.11系统环境421

编辑推荐

《Linux内核设计的艺术:图解Linux操作系统架构设计与实现原理》对操作系统内核的驾驭能力和深刻理解程度达到世界顶尖级水平，是一本能真正引导你深入理解Linux内核设计思想的经典著作。
结合真实的源码、349幅内核运行时序图和详细的文字描述，以一种开创性的方式对Linux内核进行了极为直观和透彻地阐述，读者可通过阅读本书提出自己的设计思想。

版权说明

本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问:http://www.tushu007.com