◎魅力·实践·发现

Linux 内核精析

◎郑阿奇 主编 ◎孙承龙 编著

- **精心**总结Linux开发经验
- **深入**浅出Linux系统精髓
- **深度**分析Linux内核奥秘

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY http://www.phei.com.cn

含光盘1张

魅力·实践·发现

Linux 内核精析

郑阿奇　主编
孙承龙　编著

電子工業出版社
Publishing House of Electronics Industry
北京·BEIJING

内 容 简 介

Linux 是一个完全免费、开放、跨平台的操作系统，是类 UNIX 系统中的一员，它支持多用户、多线程、多进程，实时性好、功能强大。学习 Linux 的关键在于对内核的理解和把握，本书作者长期从事 Linux 系统特别是嵌入式系统产品的研发，积累了丰富的经验。书中全面、系统、深入地介绍了 Linux，对 Linux 内核进行了深度的剖析。全书共 15 章，包括概述、Linux 内核启动、进程、进程地址空间、进程的调度、内存管理、进程间通信、系统调用、内核同步、设备驱动程序、中断、时钟、文件系统、ext 文件系统和内核模块。每一章的内容都深入浅出，文字和程序相结合，每一部分的说明都比较详细，尽可能让读者看得懂、能理解。

本书可作为 Linux 操作系统的教学参考书、嵌入式专业学生的教材或参考书，也可供 Linux 操作系统读者自学和 Linux 产品开发者参考。

图书在版编目（CIP）数据

Linux 内核精析 / 郑阿奇主编；孙承龙编著. —北京：电子工业出版社，2013.2
（魅力 • 实践 • 发现）
ISBN 978-7-121-19211-1

Ⅰ. ①L… Ⅱ. ①郑… ②孙… Ⅲ. ①Linux 操作系统—高等学校—教材 Ⅳ. ①TP316.89

中国版本图书馆 CIP 数据核字（2012）第 295316 号

策划编辑：郝黎明
责任编辑：徐 萍
印 刷：三河市鑫金马印装有限公司
装 订：三河市鑫金马印装有限公司
出版发行：电子工业出版社
北京市海淀区万寿路 173 信箱 邮编 100036
开 本：787×1 092 1/16 印张：50.5 字数：1158 千字
印 次：2013 年 2 月第 1 次印刷
印 数：3 000 册 定价：109.00 元（含光盘 1 张）

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888。
质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。
服务热线：（010）88258888。

前　言

Linux 是完全免费、开放、跨平台的操作系统，越来越受到开发商和广大师生、读者及开发者的青睐。特别是嵌入式浪潮的出现，使学习 Linux 成为时尚。

一个典型的 Linux 发行版包括 Linux 内核、GNU 程序库和工具，Linux 的发行版本与内核版本号是相互独立的，Linux 的发行版本号随发布者的不同而不同。

学习 Linux 的关键在于对内核的理解和把握，全面、系统、深入介绍 Linux 和对 Linux 内核进行深度剖析的书就成为必需。本书的作者长期从事 Linux 系统特别是嵌入式系统产品的研发，积累了丰富的经验。

本书共 15 章，包括概述、Linux 内核启动、进程、进程地址空间、进程的调度、内存管理、进程间通信、系统调用、内核同步、设备驱动程序、中断、时钟、文件系统、ext 文件系统和内核模块。每一章的内容都深入浅出，文字和程序相结合，每一部分的说明都比较详细，尽可能让读者看得懂、能理解。

本书由南京师范大学郑阿奇主编、孙承龙编著。参加本书编写的还有梁敬东、顾韵华、王洪元、刘启芬、丁有和、曹弋、徐文胜、殷红先、张为民、姜乃松、彭作民、高茜、陈冬霞、钱晓军、朱毅华、时跃华、周何骏、赵青松、周淑琴、陈金辉、李含光、王一莉、徐斌、王志瑞、孙德荣、周怡明、刘博宇、郑进、刘毅、陈杰、刘友春等。

本书配有光盘 1 张，内含教学课件和教学、学习、开发参考源文件，其中：

arch 目录中包含与体系结构相关的核心代码。每一个子目录为一种支持的体系结构，其中包含该体系结构的板级通用驱动。读者一般只需关注 arch\x86 子目录内容。

include 文件夹包括编译核心所需要的大部分头文件。与平台无关的头文件在 include/linux 子目录下。

mm 目录包括所有独立于 CPU 体系结构的内存管理代码。

kernel 目录包括主要的核心代码文件，它实现了大多数 Linux 系统的内核函数。

drivers 目录包含系统所有的设备驱动程序。

对于 Windows 平台，可以使用 Source Insight 工具新建工程加载代码、方便代码跟踪；对于 Linux 平台，可以使用 Kscope 等工具。

由于作者水平有限，不当之处在所难免，恳请读者批评指正。

编　者

2013 年 1 月

目　录

第 1 章

概述

Linux 操作系统是类 UNIX 系统中的一员，并且在操作系统发展过程中起着重要的作用。Linux 之所以能够发展得如此迅速，最主要的原因是其具有开源的特点，任何人都可以获得源代码并进行研究。本书主要对 Linux 内核进行深度的剖析，并详细讨论内核的主要功能及其内部结构，同时给出了内核的实现。本章将简要介绍内核所涉及的基本知识，此外，对内核所涉及的各领域进行简单的阐述。

1.1 Linux 的内存管理机制

操作系统的内存管理分为页式管理和段式管理两种管理机制，对于段式内存管理机制，是将指令中结合段寄存器使用的 32 位逻辑地址映射为 32 位的物理地址。随着操作系统的发展，提出了保护模式和虚拟内存管理的概念，Intel 在分段机制的基础上实现了包含模式和虚拟内存管理，后来出现了分页机制并逐渐成为内存管理机制的主流。Intel 出于兼容性的考虑，保留了段式内存管理机制，段寄存器用做选择子，段基地址及其他的某些属性存放在内存中，称为段描述符表。

段式内存管理机制的灵活性和效率都比较差，这样就需要采用页式管理机制。页式管理就是通过分页单元将线性地址转换为物理地址，其中一个重要的任务就是把请求的访问类型与线性地址的访问权限进行对比，如果该内存访问无效，则产生一个缺页的异常。

内核把线性地址分为 4KB 大小的页面，其中的每个页面都可以映射到物理地址空间的任意 4KB 大小的地址。在该映射过程中，连续的线性地址映射到物理地址空

间中不一定是连续的，分页单元把所有的 RAM 分成固定长度的物理页，每一个物理页（Page Frame）包含一个页。

把线性地址映射到物理地址的数据结构称为页表，该页表存放在主存中，并在启用分页单元之前由内核对页表进行初始化。由于引入了页式管理，因此 32 位的线性地址分为 3 个区域，分别为 Directory（目录）、Table（页表）及 Offset（偏移）。目录是线性地址的最高 10 位，用做页面表目录的下标，指向一个页面表；页表是线性地址的中间 10 位，用做具体页面表中的下标，指向具体的物理页面；偏移则是线性地址的最低 12 位，用于指向具体物理页面内的偏移量。对于每一个活动进程都有一个分配的页目录，正在使用的页目录的物理地址存放在 cr3 控制寄存器中。线性地址的 Directory 字段决定了页目录中的目录项，目录项指向某个页表，线性地址的 Table 字段则决定了页表中的页表项，Offset 字段决定页框内的偏移量，由于该字段包含了 12 位，因此每页包含了 4 096 字节的数据。页式映射的示意图如图 1-1 所示。

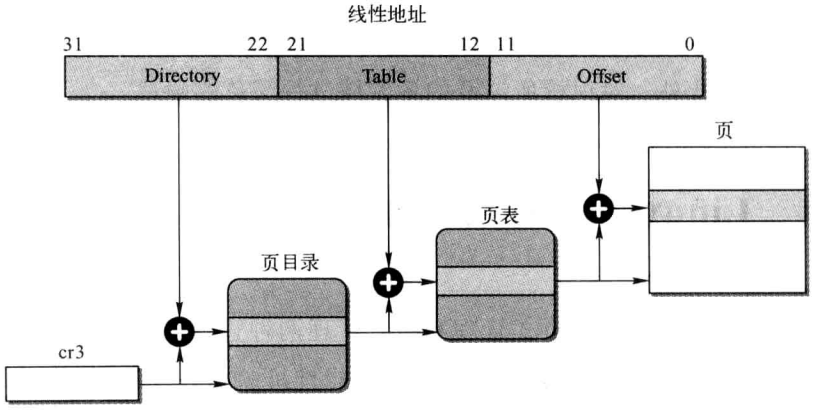

图 1-1　页式映射的示意图

页面目录共有 2^{10} 个目录项，每个目录项指向一个页面表，每个页面表有 2^{10} 个页面描述项。线性地址到物理地址的映射，先从 cr3 寄存器中获得页面目录的基地址，然后以 Directory 字段为下标，在目录中取得相应的页面表的基地址，之后以线性地址中的 Page 字段为下标，取得相应的页面描述项，最后将页面描述项中给出的页面基地址与线性地址中的 Offset 字段相加得到具体的物理地址。

由于大型的服务器需要大于 4GB 的 RAM 来同时运行大量的进程，因此必须扩展 32 位的 X86 结构所支持的 RAM，Intel 通过把处理器引脚数从 32 扩展到 36 满足了这些需求。从 Pentium Pro 开始，Intel 对物理地址的宽度进行了扩展，即物理地

址扩展（PAE）机制，这样可以把 32 位的线性地址转换为 36 位的物理地址，该机制中通过设置 cr4 控制寄存器中的物理地址扩展标志来激活 PAE。为了支持 PAE 也改变了分页机制，64GB 的 RAM 分为 2^{24} 个物理页，页表项的物理地址字段从 20 位扩展到 24 位，同时引入了页目录指针表的页表新级别，由 4 个 64 位表项构成。cr3 寄存器包含了 27 位的页目录指针表基地址字段，当把线性地址映射到 4KB 的页时，32 位线性地址中的 0～11 位为 4KB 页中的偏移量，12～20 位指向页表 512 项中的一个，21～39 位指向页目录 512 项中的一个，30～31 位指向页目录指针表 4 项中的一项，对于 cr3 则指向一个目录指针表。关于该映射机制的实现将在内存管理章节详细介绍。

1.2 Linux 的基本组成

1.2.1 进程

操作系统中的进程定义为程序执行的一个实例，每个进程都在 CPU 的虚拟内存中分配地址空间，并且每个进程的地址空间都是相互独立的。进程地址空间由允许进程使用的全部线性地址组成，每个进程所看到的线性地址集合是不同的，一个进程使用的地址与另外一个进程使用的地址之间没有关系，内核可以通过增加或删除某些线性区间来动态修改进程的地址空间。

CPU 既可以在用户模式下运行，也可以在内核模式下运行，当某个程序在用户模式下运行时，不能直接访问内核中的数据结构或程序，但是，当程序运行在内核模式时，可以访问用户态的数据。对于某个执行的程序，多数的时间都处于用户模式，只有在需要内核提供服务时才会切换到内核模式，当内核满足了用户程序的需求后，再次返回到用户模式。

Linux 系统中传统调度器对进程分别计算时间片，在进程的所有时间片用尽时，需要重新计算，现在的调度器只考虑进程的等待时间，也就是进程在就绪队列中已等待的时间。调度器的原理是按所分配的计算能力，向进程提供最大的公正性；每次调用调度器时，会选择等待时间最长的进程，把 CPU 提供给该进程，这样，进程的不公

平性不会累积，不公平会均匀地分布到系统的所有进程。

1.2.2 进程间通信

进程之间需要相互协作，在 Linux 系统中存在各种形式的进程间通信。进程间通信通常的实现方式有通过信号量机制与其他进程进行同步、向其他进程发送消息、从其他进程接收消息以及与其他进程共享一段内存区。IPC 数据结构是在进程请求 IPC 资源（信号量、消息队列或者共享内存）时动态创建的，三种进程间通信的共同点是都使用了全系统范围的资源，该资源可由几个进程同时共享。由于一个进程可能需要同类型的多个 IPC 资源，因此每个新资源都使用一个 IPC 关键字来标识。

IPC 信号量与内核信号量非常类似，除此之外与内核信号量没有任何关系，信号量不作为用于支持原子操作预定义操作的简单类型变量，而是一整套信号量，可以允许多个操作同时进行。

消息队列是指进程彼此之间可以通过 IPC 消息进行通信，进程产生的每条消息都被发送到一个 IPC 消息队列中，该消息一直存放在队列中直到另一个进程将该消息读走。产生消息并将其写入队列的进程称为发送方，其他进程则从消息队列中获取该消息。该消息包含消息正文和一个整数，数字用于标识该消息，接收方可以根据数字搜索消息，消息读取后，内核从队列中删除该消息，只能有一个进程接收一条给定的消息。

共享内存是允许两个或多个进程把公共数据放入一个共享内存区来进行访问，如果进程要访问存放在该共享内存区的数据，必须在自己的地址空间增加一个新的内存区，该内存将映射与共享内存区相关的页框，这样的页框可以方便地由内核通过请求调页进程处理。

1.2.3 内存管理

内存管理是内核中最重要同时也是最复杂的部分，需要处理器和内核之间的协作，内核在启动时通过调用 start_kernel()函数实现内存结构的初始化工作，之后内存管理

的工作交由伙伴系统算法承担。伙伴算法采用页框作为基本内存区，适合于大块内存的请求，但是当有小内存区的请求时（几十或几百字节），分配一整个页框是一种浪费，因此引入了新的数据结构来描述在同一页框内分配小内存区。

伙伴系统支持按页分配内存，如果需要分配较小的空间，分配一个或多个页框的空间，这样非常浪费空间，比较好的解决方法是将页框拆分成较小的单位。Linux 内核采用了一种称为“slab”的缓冲分配和管理方法，通过建立 slab 缓冲，内核能够储备一些对象，供后续使用。slab 分配器将释放的内存块保存在一个内部列表中，而不是立刻返回给伙伴系统，这样内核不必使用伙伴系统算法，缩短了处理的时间；其次，由于该内存块仍然是新的，其驻留在 CPU 高速缓存的概率较高。

由于 slab 过于复杂，并且本身的管理结构也需要占用过多的内存，同时效率也相对较低，因此出现了 slub 分配器，slub 分配器提供与 slab 一致的接口，不仅简化了 slab 分配器，而且提供了更好的性能。

1.2.4 设备驱动

根据设备驱动程序的基本特性，设备文件可以分为块设备与字符设备，块设备的数据可以随机访问，如硬盘、CD-ROM 驱动器等，对于字符设备的数据则不可以随机访问；设备文件是存放在文件系统中的实际文件，其索引节点不包含指向磁盘上数据块的指针，但是必须包含硬件设备的标识符，对应字符或块设备文件。

由于块设备层的设计导致需要持续地调整块设备的速率及工作方式，因此块设备驱动程序要比字符设备复杂得多。块设备的主要特点是 CPU 和总线读/写数据所耗费的时间与磁盘硬件的速度不匹配；块设备的平均访问时间长，这是因为磁盘控制器必须在磁盘表面将磁头移动到存储数据的准确位置。

驱动程序通过一组固定的接口与内核代码通信，而扩展设备则通过设备驱动程序处理，因此扩展设备/驱动程序对内核的代码没有影响；内核代码与总线驱动程序的关系比与具体设备驱动程序的关系更为密切。此外，由于不同总线系统之间使用的硬件技术差异较大，因此总线驱动程序向相关的设备驱动程序提供功能和选项的方式不存在标准的接口。

1.2.5 中断

Linux 内核中的中断通常分为同步中断和异步中断，同步中断是 CPU 本身在执行程序过程中由 CPU 控制单元产生的，而异步中断则是由其他硬件设备随机产生的；不同设备对应的中断不同，每个中断通过一个唯一的数字标识，因此使得内核能够对中断进行区别，这样内核能给不同的中断提供不同的中断处理程序。

1.2.6 时钟

时钟中断源能周期地向 CPU 发出中断，内核在中断处的例程中检查当前进程时间片是否到期，为调度器提供时钟源。此外，内核根据 RTC 获取到起始时间后，依此来维护内核时间，由于 RTC 计时的单位是秒，为了获取精确的时间，内核启动时从 RTC 中读取起始时间，之后在每一次时钟中断时，利用起始时间加上中断周期，可以把时钟精确到毫秒的级别。后来又出现了 TSC（Time Stamp Counter，是 CPU 的一个寄存器），该寄存器随着 CPU 时钟周期的递增加 1，如果当前的 CPU 主频为 1GHz，则该寄存器每纳秒加 1。内核以 RTC 为基础，每次时钟中断通过汇编指令 rdtsc 读取 TSC，内核在时钟中断中通过 TSC 能够计算出两次时钟中断间流逝的精确时间，这样时间能够精确到纳秒的级别。

1.2.7 文件系统

每种操作系统至少都有一种“标准文件系统”，提供了某些功能可以可靠地执行特定的任务，Linux 附带的 ext2/3 文件系统是一种标准文件系统，该文件系统经证实是健壮的，同时 Linux 还提供了备选的方案。

为了支持各种本机文件系统，并且同时允许访问其他操作系统的文件，Linux 内核在用户进程和文件系统实现之间提供了一个抽象层，称为虚拟文件系统（VFS，Virtual File System），一方面用来提供一种文件、目录及其他对象操作的统一方法，另一方面能够与各种方法给出的具体文件系统的实现达成妥协。如图 1-2 所示为虚拟文件系统的示意图。