HZ BOOKS
华章教育

计 算 机 科 学 丛 书

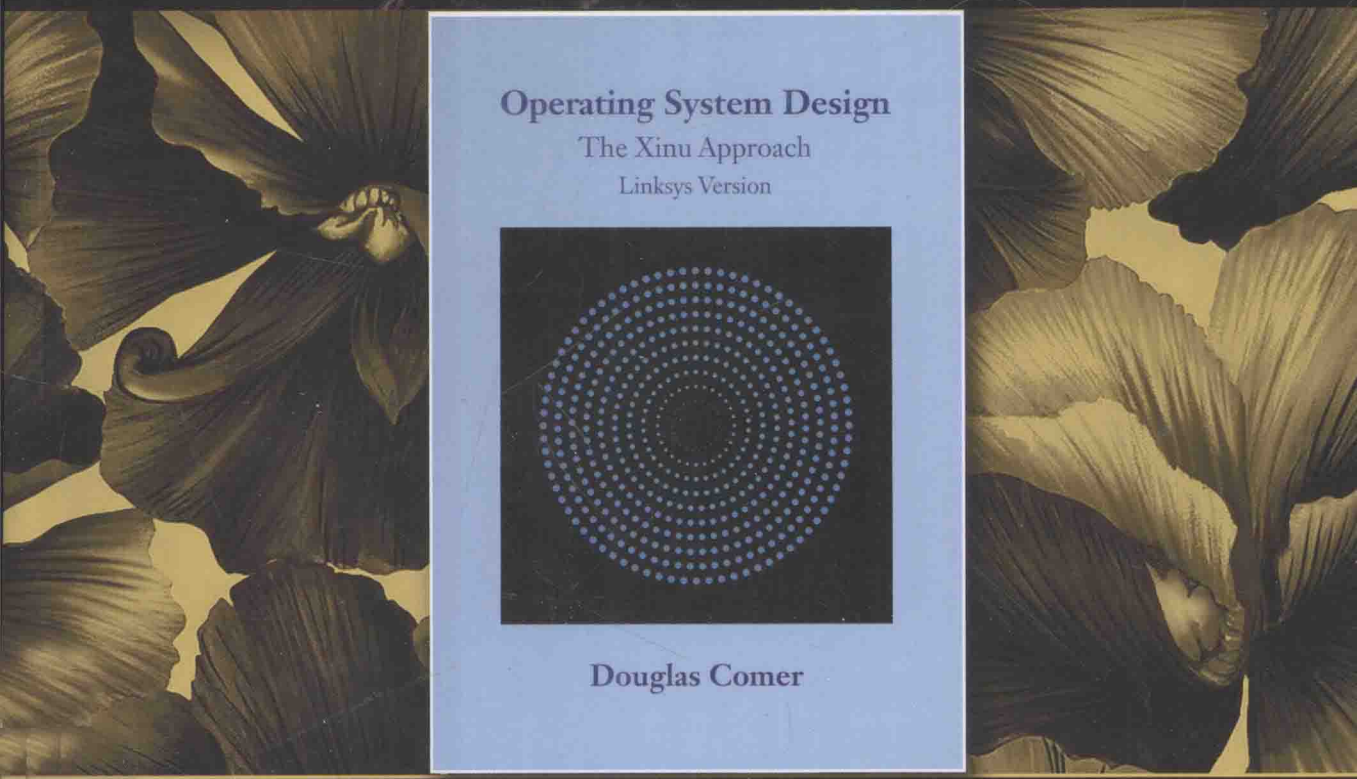
CRC Press
Taylor & Francis Group

操作系统设计
Xinu方法

(美) Douglas Comer 著
普度大学

邹恒明 周亮 曹浩 刘正邦 等译
上海交通大学

Operating System Design
The Xinu Approach, Linksys Version



机械工业出版社
China Machine Press

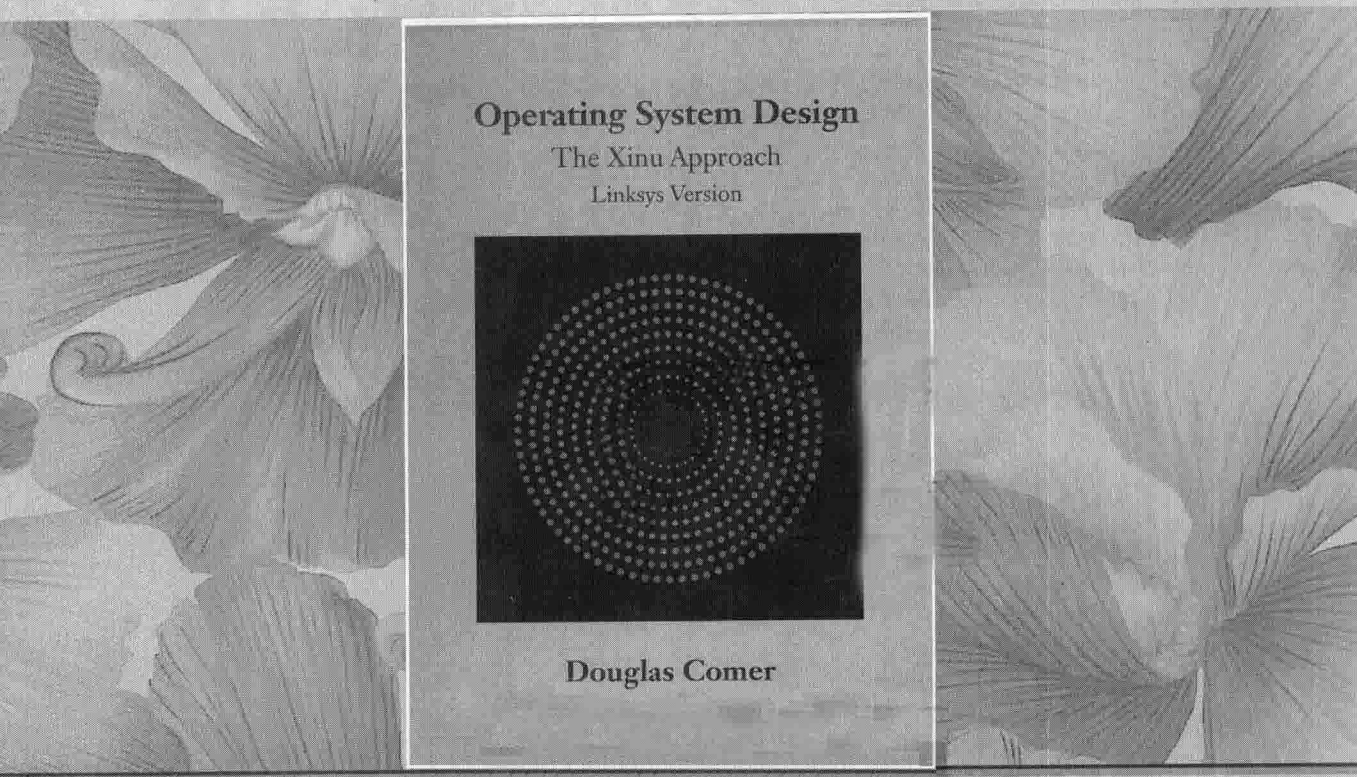
计 算 机 科 学 丛 书

操作系统设计
Xinu方法

(美) Douglas Comer 著
普度大学

邹恒明 周亮 曹浩 刘正邦 等译
上海交通大学

Operating System Design
The Xinu Approach, Linksys Version



机械工业出版社
China Machine Press

图书在版编目（CIP）数据

操作系统设计：Xinu 方法 /（美）科默（Comer，D.）著；邹恒明等译. —北京：机械工业出版社，2013.10
（计算机科学丛书）
书名原文：Operating System Design：The Xinu Approach，Linksys Version
ISBN 978-7-111-42826-8
Ⅰ. 操… Ⅱ. ①科… ②邹… Ⅲ. 操作系统－程序设计 Ⅳ. TP316
中国版本图书馆 CIP 数据核字（2013）第 123969 号

本书对操作系统的内存管理、进程管理、进程协调和同步、进程间通信、实时时钟管理、设备无关的 I/O、设备驱动、网络协议、文件系统等进行了详细的介绍，并利用分层的设计范式，以一种有序、易于理解的方式来阐述这些内容。本书以 Xinu 操作系统为系统设计的样板和模式，从一个裸机开始，一步一步地设计和实现一个小型但优雅的操作系统。本书的样本代码可以运行在 Linksys E2100L 无线路由器上。

本书适用于高年级的本科生或低年级的研究生，也适用于那些想了解操作系统的计算机从业人员。学习本书前，学生需要具备基本的程序设计能力，应当理解基本的数据结构，包括链表、栈和队列，并且应当用 C 语言写过程序。

机械工业出版社（北京市西城区百万庄大街 22 号 邮政编码 100037）
责任编辑：盛思源
藁城市京瑞印刷有限公司印刷
2014 年 1 月第 1 版第 1 次印刷
185mm×260mm·23.5 印张
标准书号：ISBN 978-7-111-42826-8
定 价：79.00 元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换
客服热线：（010）88378991 88361066 投稿热线：（010）88379604
购书热线：（010）68326294 88379649 68995259 读者信箱：hzjsj@hzbook.com

文艺复兴以降，源远流长的科学精神和逐步形成的学术规范，使西方国家在自然科学的各个领域取得了垄断性的优势；也正是这样的传统，使美国在信息技术发展的六十多年间名家辈出、独领风骚。在商业化的进程中，美国的产业界与教育界越来越紧密地结合，计算机学科中的许多泰山北斗同时身处科研和教学的最前线，由此而产生的经典科学著作，不仅擘划了研究的范畴，还揭示了学术的源变，既遵循学术规范，又自有学者个性，其价值并不会因年月的流逝而减退。

近年，在全球信息化大潮的推动下，我国的计算机产业发展迅猛，对专业人才的需求日益迫切。这对计算机教育界和出版界都既是机遇，也是挑战；而专业教材的建设在教育战略上显得举足轻重。在我国信息技术发展时间较短的现状下，美国等发达国家在其计算机科学发展的几十年间积淀和发展的经典教材仍有许多值得借鉴之处。因此，引进一批国外优秀计算机教材将对我国计算机教育事业的发展起到积极的推动作用，也是与世界接轨、建设真正的世界一流大学的必由之路。

机械工业出版社华章公司较早意识到“出版要为教育服务”。自 1998 年开始，我们就将工作重点放在了遴选、移译国外优秀教材上。经过多年的不懈努力，我们与 Pearson，McGraw-Hill，Elsevier，MIT，John Wiley & Sons，Cengage 等世界著名出版公司建立了良好的合作关系，从他们现有的数百种教材中甄选出 Andrew S. Tanenbaum，Bjarne Stroustrup，Brain W. Kernighan，Dennis Ritchie，Jim Gray，Afred V. Aho，John E. Hopcroft，Jeffrey D. Ullman，Abraham Silberschatz，William Stallings，Donald E. Knuth，John L. Hennessy，Larry L. Peterson 等大师名家的一批经典作品，以“计算机科学丛书”为总称出版，供读者学习、研究及珍藏。大理石纹理的封面，也正体现了这套丛书的品位和格调。

“计算机科学丛书”的出版工作得到了国内外学者的鼎力襄助，国内的专家不仅提供了中肯的选题指导，还不辞劳苦地担任了翻译和审校的工作；而原书的作者也相当关注其作品在中国的传播，有的还专程为其书的中译本作序。迄今，“计算机科学丛书”已经出版了近两百个品种，这些书籍在读者中树立了良好的口碑，并被许多高校采用为正式教材和参考书籍。其影印版“经典原版书库”作为姊妹篇也被越来越多实施双语教学的学校所采用。

权威的作者、经典的教材、一流的译者、严格的审校、精细的编辑，这些因素使我们的图书有了质量的保证。随着计算机科学与技术专业学科建设的不断完善和教材改革的逐渐深化，教育界对国外计算机教材的需求和应用都将步入一个新的阶段，我们的目标是尽善尽美，而反馈的意见正是我们达到这一终极目标的重要帮助。华章公司欢迎老师和读者对我们的工作提出建议或给予指正，我们的联系方法如下：

华章网站：www. hzbook. com
电子邮件：hzjsj@ hzbook. com
联系电话：（010）88379604
联系地址：北京市西城区百万庄南街 1 号
邮政编码：100037

华章教育
华章科技图书出版中心

译者序

Operating System Design: The Xinu Approach, Linksys Version

——像小孩那样学习

我实在告诉你们，你们若不回转，变成小孩的样式，断不得进天国。

——马太福音 18：3

这是耶稣教训门徒所说的话。虽然讲的是天国的事情，但不经意间也指出了一条学习的路径。众所周知，小孩接受新鲜事物的能力非常强，小孩能够随着时间的推移而不知不觉地学会说话和日常的生活规则，而成人学习一门新语言或一种新技能则通常较为困难。

小孩对语言或技能的掌握是在成人所说所行的点滴中先行模仿然后再悟出的，也就是通过模仿从琐细中理出了头绪。小孩的这种学习方法是一种典型的先实际后理解、先细节后全景、从只见树木到顿见森林的转变。这与成人所惯常采用的学习方式有很大不同。成人采用的学习方法通常是先理论后实践、先全貌后细节、先森林后树木式的由高至低逐步推进的方式。这在大学的“操作系统”课程学习中体现得十分明显。一般大学的普遍做法是先介绍操作系统的全貌，然后介绍操作系统的个体组成部分，再附加一些动手实验，所谓通过实验加深对理论的理解（其他课程的学习方式也大同小异）。此种模式是一种典型的自顶向下的演绎模式，由于其由来已久，人们已经习以为常，并没有想过有什么不妥。但操作系统及许多大学课程教学中所呈现的事实是，很多人对其感到枯燥难懂，难以掌握。

既然如此，为什么不试试另一种学习模式，即先动手后理解、先细节后抽象、由树木到森林，像小孩一样来学习操作系统或其他专业课程呢？对于操作系统来说，完全可以采用由底至上的方法，先不讲操作系统理论和模型，而是先来实际设计一个计算机的内存管理系统，在动手实践中逐渐抽象出规律，然后上升为理论和模型。在讨论完内存管理之后再来阐述如何调度程序，这样逐次推进，最后搭起整个操作系统。这就像把一大堆积木呈现给读者，先搭起一个长方体，然后搭起几个轮子，再搭上一些窗户，最终，一辆汽车就形成了。

本书所采取的正是由底至上、由动手到理论、由具体到抽象、由细节到整体、由树木到森林的学习模式来阐述操作系统的原理和设计。该书以细节为核心，以设计与实现为手段，通过设计和实现操作系统的各种功能来引入操作系统的原理和模型，并以设计人员在构建一个操作系统时所遵循的工作次序来组织全书。该书从一个裸机开始，一步一步地设计和实现一个小型但优雅的操作系统——Xinu。本书每一章描述设计 Xinu 系统架构里的一个组件，并提供示例软件来演示该层架构所提供的功能。这种与传统模式的截然不同让人一开始有些反感，但在体会了此种方式的美妙后又会恍然大悟。

该书的最大特点是实践性强，以与实际情况非常贴切的场景为背景，以动手操作为推手，以实际代码为讨论对象，将操作系统的各种实战原理和模式娓娓道来，易于理解和消化。读者只要顺着书的讲解，并配合运行和分析其附带的代码，即可掌握操作系统的设计和实现。该书在美国普度大学所受到的青睐从一个侧面佐证了这一点。

本书适用于高年级的本科生或低年级的研究生，也适用于那些想了解操作系统的计算机从业人员。鉴于该书的篇幅和所涉及的具体动手实现，建议分配 4 个学分或以上来组织本门课程。另外，考虑到并不是所有人都喜爱先细节后抽象的模式，建议与一本优良的阐述操作系统原理的书结合进行讲授，效果或将更加令人满意。

本书由上海交通大学 2012 年春季高级操作系统课程组集体翻译，因人员众多，就不一一列出。在此特向对本书翻译提供帮助的人员表示感谢。

鉴于译者水平所限，书中纰漏甚至错误在所难免，谨望读者不吝指正。

邹恒明

2013 年 5 月于美国达拉斯

建造计算机操作系统有点像编织锦缎。这两种工作的最终成品都是一个和谐一致、大型、复杂的人造系统。在每一种情况下，最后的人造成品都是由细微但却精巧的步骤所构造。在编织锦缎时，细节是至关重要的，因为一点点不协调的瑕疵都很容易观察到。就像锦缎里的缎面一样，加入到操作系统里的每个新组件都需要与整体的设计相协调。从这个角度看，将不同片段组装起来的机械加工只是整个建造过程中的一小部分，一个大师级的产品必须以某个模式为蓝本，所有参与系统设计的工作人员都必须遵守这种模式。

有讽刺意味的是，现有的操作系统教材或课程很少对底层的模式和原理进行解释，而这些模式和原理正是操作系统构造的基础。在学生看来：操作系统似乎是一个暗箱，而现有的教材则加强了这种误解，因为这些教材所解释的不过是操作系统的功能，其关注的也只是操作系统各种能力的使用。更为重要的是，学生在学习操作系统时采取的是从操作系统外面来察看的方式，从而常常导致这样一种感觉：认为操作系统由一组抽象的界面所组成，这些界面下的功能由一大堆晦涩神秘的代码连接在一起，而这些神秘的代码本身还包含着许多与机器硬件直接相关的、无规律可寻的奇技巧术。

令人惊奇的是，学生一旦从大学毕业，就马上觉得与操作系统有关的工作已经结束，自己不再需要理解或学习操作系统，因为由商业公司和开源社区所构造的现有操作系统足以应付各种需要，没有自己什么事情了。但没有什么比这种想法离真理更远了。有讽刺意味的是，尽管为个人计算机设计传统操作系统的公司数量比以前更少了，但社会和行业对操作系统技能的需求却在增长，许多公司雇佣大学生来从事操作系统方面的工作。社会上这些对操作系统技能的需求来源于更便宜的微处理器，这些便宜的微处理器嵌入在智能手机、视频游戏、iPod、Internet 路由器、线缆和机顶盒以及打印机中。

在与嵌入式系统打交道时，有关原理和结构的知识非常关键，因为程序员可能需要在现有的操作系统内部构造某种或某个新的机制，或者对现有操作系统进行修改以便可以在新的硬件平台上运行。而且，为嵌入式设备编写应用程序时需要理解下层的操作系统。如果不理解操作系统设计的各种细微之处，则不可能充分利用这些小型嵌入式处理器的能力。

本书的目的是揭开操作系统设计中的神秘感，将方方面面的材料整合为一个系统化的整体。本书对操作系统的主要系统组件进行了详细阐述，并以一种层次架构的设计范式来组织这些组件，从而以一种有序、可理解的方式来展开这些内容。与其他评述性书籍不同的是，本书并不尽可能多地提供不同方案，呈现给读者的将是一个基于传统过程的、使用实际的、直截了当的原语所构造的操作系统。本书从一个裸机开始，一步一步地设计和实现一个小型但优雅的操作系统。这个称为 Xinu 的操作系统将成为系统设计的样板和模式。

虽然 Xinu 操作系统的规模较小，可以完全容纳在本书中，但该系统却包括了构成一个普通操作系统的全部组件：内存管理、进程管理、进程协调和同步、进程间通信、实时时钟管理、设备独立的输入输出、设备驱动、网络协议和一个文件系统。本书将这些组件组织成一个层次架构，使它们之间的相互连接清晰可见、设计过程浅显易懂。尽管规模小，但 Xinu 却拥有大型系统的能力。此外，Xinu 并不是一个玩具系统，它在很多商业产品中得到了应用。使用该系统的厂商包括 Mitsubishi、Lexmark、HP、IBM、Woodward（woodward. com）、Barnard Software 和 Mantissa 公司。学生通过本书可以学到的重要一课是：不管是小型嵌入式系统还是大型系统，好的系统设计都一样重要，一个系统的大部分能力都来自于良好的抽象。

本书所覆盖的所有议题都以一种特定的次序排列，这种次序就是设计人员在构建操作系统时所遵守的工作次序。本书每一章描述设计架构里的一个组件，并提供示例软件来演示由该层架构所提供的功能。使用这种方式具有如下几种优点：第一，每一章所解释的操作系统的功能子集均比上一章所讨论的功能子集更大，这种安排使我们在考虑一层特定架构的设计和实现时不用关心后续层面的实现。

第二，每一章的细节描述在第一次阅读时可以跳过去，读者只需要理解该层所提供的服务即可，而不是这些服务是如何实现的。第三，如果按次序阅读本书，读者可以先理解一个功能，然后在后面看到该功能是如何被后续部分所使用的。第四，有智力挑战的议题（如对并发的支持）出现在书的较前面，高层次的操作系统服务则出现在后面。在本书中，读者将看到大部分核心的功能仅仅只用几行代码就可以完成，这样我们就可以将大部分的代码（网络和文件系统）放到书的较后面，在读者已经做出了充分的思想准备后再进行讲解。

如前所述，与其他关于操作系统的许多书不一样的是，本书并不试图对每个系统组件的每种实现方案进行评估，也不对现有的商业系统进行综述。而是选择对一组使用最广泛的操作系统原语的实现细节进行阐述。例如，在讨论进程协调的一章，我们解释的是信号量（使用最广泛的进程协调原语）原语，而对其他原语（如监视器）的讨论则放到练习里。我们的目的是展示如何将原语在传统的硬件上实现，消除神秘。学生一旦理解了一组特定原语的魔力，其他原语的实现也就容易掌握了。

本书的示例代码可以运行在 Linksys E2100L 无线路由器上，该无线路由器在零售商店里就可以买到。只不过，我们并不是将 Linksys 硬件作为一个无线路由器来使用。我们的做法是，打开 Linksys 设备，将一根串行线连接到其控制端口，使用该串行线来中断 Linksys 正常的启动过程，并通过输入命令来迫使 Linksys 硬件下载和运行一个 Xinu 操作系统副本。也就是说，我们基本上忽略供应商所提供的软件，而是对其底层的硬件进行控制来运行 Xinu。

本书适用于高年级的本科生或者研究生，也适用于那些想了解操作系统的计算机从业人员。在本书所提供的全部材料里，虽然没有任何议题的难度达到不能理解的程度，但学习本书的全部内容可能需要超过一学期的时间。本科生里很少有学生能够熟练地阅读串行程序，而理解运行时环境的细节或机器架构的学生就更少了。因此，必须对学生进行仔细引导，以便使其可以掌握进程管理和进程同步的知识。如果时间有限，我推荐覆盖的内容包括第 1 章～第 7 章（进程管理）、第 9 章（基本的内存管理）、第 12 章（中断处理）、第 13 章（时钟管理）、第 14 章（设备无关的 I/O）和第 19 章（文件系统）。此外，对于一个完整学期的本科生课程来说，讨论第 20 章的远程文件系统等基本的远程访问议题也很重要。对于研究生课程来说，学生应当完整地阅读整本书，课堂讨论则应该专注于一些微妙的细节、各种折中和不同实现方案的比较。不管是本科生课程还是研究生课程，都应该包括的两个议题是：1）在初始化阶段，当一个运行中的程序转化为一个进程时所发生的各种改变；2）当输入行里的字符序列作为一个字符串变量传递给命令进程时，在操作系统壳里所发生的转化。

在所有情况下，如果学生能够在实验室中对系统进行动手实验，则学习的效果将大幅提高。理想的状态下，学生可以在课程的最初几天或几个星期开始使用这个系统，然后再试图理解系统的内部结构。本书第 1 章提供了几个例子和一些能够引起学生兴趣的实验（令人吃惊的是，很多学生在学习过操作系统课程后，却没有写过一个并发程序或使用过操作系统功能）。

如果要在一个学期内覆盖本书的大部分内容，则要求极快的进度，而这在本科生课程里难以达到。此时，选择略去哪些内容将很大程度上取决于选修本课程的学生的背景。在系统课程里，我们需要课堂讲解时间来帮助学生理解动机和细节。如果学生修过的“数据结构”课程里对内存管理和表处理进行过讨论，则本书第 4 章和第 9 章的内容可以略过。如果学生在将来会选修网络方面的课程，则第 17 章的网络协议内容也可以跳过。此外，本书包括一章远程磁盘系统和一章远程文件系统，这两章的内容存在一些相似之处，可以略过一章。相对来说，远程磁盘系统一章的内容可能更加贴切，因为该章引入了磁盘块缓存的议题，而该议题对于许多操作系统来说都非常重要。

在研究生课程里，课堂时间可以用来讨论动机、原理、折中、不同原语集和不同的实现方案比较。学生在本课程学习结束后，应当对进程模型、中断和进程之间的关系有一个深刻的理解，同时也将具备理解、创建和修改系统组件的能力。学生应当在大脑中建立起了整个系统的完整概念模型，并且知道所有的组件之间是如何交互协作的。

我推荐在各个层面上设计程序设计实验。本书的许多练习都推荐对代码进行修改或者测量，或者尝试不同的实现方案。相关的软件可在下面的网站上免费下载，该网站上还列有如何创建一个 Linksys

实验平台的指令：www. xinu. cs. purdue. edu。

因为 Linksys 的硬件非常便宜，所以构建一个实验的成本很低。此外，我们也有用于其他硬件平台的软件版本，这些版本包括 x86 和 ARM 的一个功能有限的版本。

本书中的许多练习都建议进行改进、实验和不同实现，但是也可以设计大型实验项目。可以用于不同硬件平台的大型实验例子包括：虚拟内存系统、不同计算机之间指令执行的同步机制、虚拟网络的设计等。例如，普度大学的一些学生就将 Xinu 操作系统移植到了各种处理器上，或者为各种 I/O 设备编写了设备驱动程序。

学习本书前，学生需要具备基本的程序设计能力。学生应当理解基本的数据结构，这些基本结构包括链表、栈和队列，并且应当用 C 语言写过程序。

最后，我鼓励设计人员尽可能使用高级程序设计语言，仅在必要的情况下才使用汇编语言。根据这种原则，Xinu 操作系统的大部分都是用 C 语言编写的。少数一些与机器相关的功能，如上下文切换和中断分配器的最底层功能，则是用汇编语言写成的。所有的汇编语言代码都附有解释和注释，使读者无需学习汇编语言的细节就可以理解这些代码。此外，我们还提供用于其他平台的 Xinu 版本，这样就可以对在各种平台上实现 Xinu 操作系统的成本进行比较。例如，我们可以对在 MIPS 处理器上实现 Xinu 所需要的代码量和在其他处理器架构（如 x86）上实现 Xinu 所需要的代码量进行比较。

本书的成书要归功于我过去在商业操作系统上所获得的各种经验，这些经验有好也有坏。虽然 Xinu 操作系统与现有的操作系统在内部机制上并不相同，但其基本的思想却并不新颖。另外，虽然 Xinu 系统里的许多概念和名称都来自于 UNIX 系统，但读者应当注意，这两个系统里的许多函数所使用的参数和内部结构有巨大的不同。因此，为一个系统所写的应用程序在未经修改的情况下不能在另一个平台上运行。

我感谢为 Xinu 项目贡献了思想、辛劳和激情的所有人的帮助。在过去的岁月里，普度大学的许多研究生都从事过本系统的工作，他们为 Xinu 进行过移植，写过设备驱动。从原始的系统版本开始，Xinu 到目前已经走过了 30 多年的历程。本书的 Xinu 版本是原始版本的一个完全重写，但却保留了原始设计的优雅。Dennis Brylow 将 Xinu 移植到了 Linksys 平台，并且创建了许多底层的构件，包括启动代码、上下文切换和 Ethernet 驱动。Dennis 还设计了重启机制，并应用在普度大学的实验室里。另外，我特别要感谢我的妻子和我的合作伙伴 Christine，她的仔细编辑和建议让本书改善良多。

Douglas E. Comer

2011 年 8 月

关于作者

Operating System Design: The Xinu Approach, Linksys Version

Douglas Comer 是美国普度大学（Purdue University）计算机系的杰出教授，国际公认的计算机网络、TCP/IP 协议、Internet 和操作系统设计的专家。Comer 发表论文无数，出版专著多部，是一位为研究和教育而开发课程体系和实验项目的先驱。

作为一个多产作家，Comer 博士的书被翻译成 16 种语言，广泛应用于全世界的计算机科学、工程和工商行政管理等院系和行业。Comer 博士划时代的三卷巨著《Internetworking with TCP/IP》对网络和网络教育产生了革命性的影响。他所编写的教科书和富有创意的实验手册已经塑造和继续塑造着研究生和本科生的教学课程体系。

Comer 博士所写书籍的精确和洞见反映出他在计算机系统领域的广泛背景。他的研究横跨硬件和软件。他创建了一个完整的操作系统 Xinu，编写了设备驱动程序，为传统计算机和网络处理器实现了网络协议软件。Comer 博士的研究成果已经应用到工业界的各种产品中。

Comer 博士创建和主讲的课程包括“网络协议”、“操作系统”、“计算机体系架构”，其听众既有大学生和学术界同仁，也有工业界的工程师。他创新的教育实验让他和他的学生能够设计和实现大型、复杂系统的原型，并对结果原型的性能进行度量。Comer 博士长期在公司、大学和会议上讲课和演说，还为工业界提供咨询服务，以帮助他们设计计算机网络和系统。

二十多年来，Comer 教授担任研究期刊《Software——Practice and Experience》主编。在普度大学停薪留职期间，他在思科公司（Cisco）担任研究副总裁。Comer 博士是 ACM 院士、普度教育学院院士和无数奖项的获得者，其中包括 Usenix 终身成就奖。

关于 Comer 博士的更多信息可在下面网站找到：www. cs. purdue. edu/people/comer。

关于 Comer 博士所著书籍的更多信息可在下面网站找到：www. comerbooks. com。

引言和概述

我们的小小系统也有风光的时刻。

——Alfred, Lord Tennyson

1.1 操作系统

每一个智能设备和计算机系统中都隐藏着这么一类软件，它们控制着处理信息、管理资源以及与显示屏、网络、磁盘和打印机等设备通信的工作。总的来说，这些进行控制和协调工作的代码通常叫做执行器、监视器、任务管理器，或者内核，而我们将使用一个更宽泛的术语操作系统。

计算机操作系统是人类创造的最复杂的物体之一：计算机操作系统允许多个计算进程和用户同时共享一个CPU，保护数据免受未经授权的访问，并保持独立输入/输出（I/O）设备的正确运行。操作系统提供的高级服务都是通过向复杂的硬件发送一系列详细的命令实现的。有趣的是，操作系统并不是从外部控制电脑的独立机制——它还包括一些软件，这些软件由执行应用程序的同一处理器执行。事实上，当处理器运行应用程序的时候，处理器是不能执行操作系统的，反之亦然。

保证操作系统总在应用程序运行结束后重新夺回控制权的安排机制使得操作系统的设计变得非常复杂。操作系统最令人印象深刻的方面来自于服务和硬件之间的不同：操作系统在低级的硬件上提供高级服务。随着本书内容的推进，读者就会理解系统软件处理像串行接口这样简单的设备需要做的事情。而其中的哲学原理很简单：操作系统应该提供让编程更加容易的抽象，而不是反映底层硬件设备的抽象。因此，我们得出结论： 1

> 设计操作系统时，应该隐藏底层的硬件细节，并创建一个为应用程序提供高级服务的抽象机器。

操作系统的设计并不是人们所熟知的工艺。最初，由于计算机的缺乏和价格的昂贵，只有少数程序员有从事操作系统相关工作的机会。而现在，由于先进的微电子技术降低了制造成本使得微处理器不再昂贵，操作系统便成为一种商品，与此同时也只有少数程序员从事操作系统方面的工作。有趣的是，由于微处理器变得非常便宜，大多数电子设备都是从可编程处理器构建得到，而不是从离散的逻辑构建得到。因此，设计与实现微机和微控制器的软件系统不再是专家的专利，它已成为一个称职的系统程序员必须能胜任的技术。

幸运的是，随着生产新机器的技术的发展，我们对于操作系统的理解也在不断提高。研究人员已经找出了根本问题，制定了设计原则，定义了基本的组件，并设计了组件一起工作的机制。更重要的是，研究人员还定义了一系列的抽象，如文件和当前进程（这些抽象对于所有的操作系统都是相同的），并且已经找到了实现这些抽象的有效方式。最后，我们知道了如何将操作系统的不同组件组织成一个有意义的系统设计与实现。

同早期系统相比，现代操作系统是简洁的、可移植的。设计良好的系统都遵循着将软件分割成一系列基本组件的基本设计模式。因此，现代系统就变得更容易理解和修改，相比早期的系统其处理开销也比较小。

供应商出售的大型商业操作系统通常包括很多额外的软件组件。例如，一个典型的操作系统软件发行版包括编译器、连接器、装载程序、库函数和一系列的应用程序。为了区分这些额外的软件和一个基本的操作系统，我们有时会用内核指代常驻在内存中并且提供诸如并发进程支持等关键性服务的代码。在本书中，操作系统这个术语指的就是内核，而不包括其他附加的功能。一个最小化内核功能的设计有时称为微内核设计。我们的讨论就将集中在微内核上。 2

1.2 本书的研究方法

本书讲解了如何构建、设计并且实现操作系统的内核。书中使用了工程学方法，而不是仅仅罗列

操作系统的特性和抽象地对其进行描述。这种方法向我们展示了每一个抽象是如何建立的，以及如何将这些抽象组织成一个优雅、高效的设计。

这种工程学方法有两个优势。第一，因为本书的内容涵盖操作系统的每一部分，所以读者会看到整个系统如何融合在一起，而不仅仅是一两个部分之间如何交互。第二，由于读者可以得到书中描述的所有部分的源代码，所以任何部分的实现都没有什么神秘的地方——读者可以获得一份系统的副本来检查、修改、工具化、测量、扩展或者将其移植到其他架构。在本书的最后，读者会看到操作系统的每个部分是如何满足设计需求的，以帮助读者理解可选的设计方案。

本书的关注点是实现，这意味着代码是本书的一个重要组成部分。事实上，代码是讨论的核心，必须通过阅读和学习所罗列的程序来欣赏其中的微妙之处和工程中的细节。例子代码都非常精简，这意味着读者可以集中精力在概念的理解上而不需要费力地阅读许多页的代码。但某些练习建议的改进或修改需要读者深入细节或者找到其他方案。熟练的程序员会找到更多方法来改进和扩展我们的系统。

1.3 分层设计

如果设计得好，操作系统的内部可以如最常规的程序一样优雅、简洁。为了达到优雅的目标，本书所描述的设计，将系统功能划分为8个大类，并将这些组件组织成多个层次。系统的每层提供了一个抽象的服务，该服务又通过低层提供的抽象服务实现。该方法的特点很明显：逐渐变大的层次子集可以形成更加强大的系统。我们将会看到如何利用分层方法来帮助设计者降低操作系统的复杂性。

该方法的另一个重要特点体现在运行时的效率上——设计者可以在不引入额外开销的情况下将操作系统的片段构建为一个层次结构。不过，该方法不同于传统的分层系统。在传统的分层系统中，K层的函数只能调用$K-1$层的函数。在这种多层次的方法中，层次只为设计者提供了一个概念模型——在运行时，高层的函数可以直接调用低层的任何函数。我们将看到，直接调用使得整个系统更有效率。

图1-1说明在本书中所使用的层次结构，给出了我们将讨论的组件的预览，并展示了其中所有片
3 段的结构。

分层结构的核心是计算机硬件。虽然硬件不是操作系统本身的一部分，但现代硬件具有的特性允许其与操作系统紧密集成在一起。因此，我们可以认为硬件是层次结构中的第0层。

从硬件开始往上，操作系统的每个层次都会提供更强大的原语，从而为应用程序屏蔽原始的硬件。内存管理层控制和分配内存。进程管理层是操
4 作系统最基础的组成部分，它包括进程调度和上下文切换。接下来一层的功能包含了进程管理的其余部分内容，包括创建、杀死、挂起和恢复进程。进程管理的上层是进程协调组件，它实现了信号量。实时时钟管理的功能包含在接下来的一个层次中，它允许应用软件在一定的时间内推迟响应。实时时钟上面的一层是与设备无关的I/O程序层，它提供我们所熟悉的服务，如读（read）和写（write）操作。在设备程序之上的一层实现网络通信，更上面的一层则实现了文件系统。

系统的内部组织不应该与系统提供的服务相混淆。虽然将组件组织成不同的层次可以使设计和实

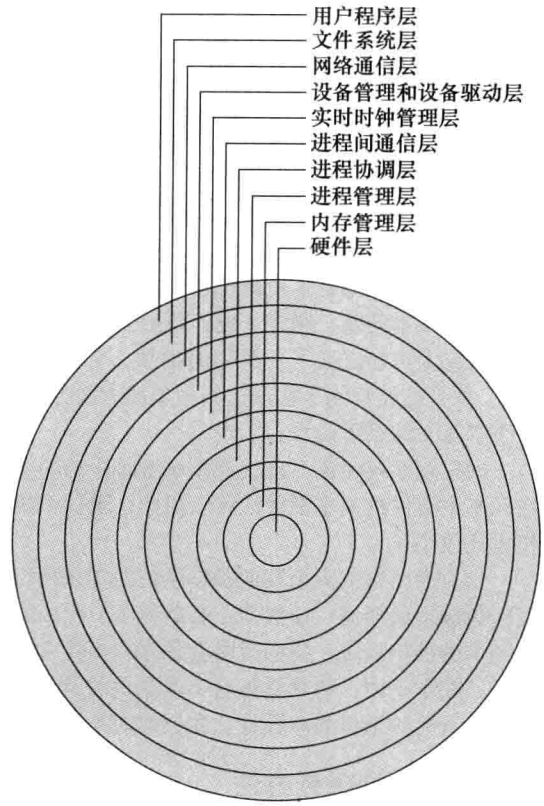


图1-1 在本书中使用的多层次结构