前　言

致本书读者

欢迎阅读本书！我们希望你阅读本书时，就像我们撰写它时一样开心。本书的英文书名为《Operating Systems:Three Easy Pieces》，这显然是向理查德·费曼（Richard Feynman）针对物理学主题创作的、最了不起的一套讲义[F96]致敬。虽然本书不能达到这位著名物理学家设定的高标准，但也许足够让你了解什么是操作系统（以及更一般的系统）。

本书围绕 3 个主题元素展开讲解：虚拟化（virtualization）、并发（concurrency）和持久性（persistence）。对于这些概念的讨论，最终延伸到讨论操作系统所做的大多数重要事情。希望你在这个过程中体会到一些乐趣。学习新事物很有趣，对吧？

每个主要概念在若干章节中加以阐释，其中大部分章节都提出了一个特定的问题，然后展示了解决它的方法。这些章节很简短，尝试（尽可能地）引用作为这些想法真正来源的源材料。我们写这本书的目的之一就是厘清操作系统的发展脉络，因为我们认为这有助于学生更清楚地理解过去是什么、现在是什么、将来会是什么。在这种情况下，了解香肠的制作方法几乎与了解香肠的优点一样重要。

我们在整本书中采用了几种结构，值得在这里介绍一下。

无论何时，在试图解决问题时，我们首先要说明最重要的问题是什么。我们在书中明确提出关键问题（crux of the problem），并希望通过本书其余部分提出的技术、算法和思想来解决。

在许多地方，我们将通过显示一段时间内的行为来解释系统的工作原理。这些时间线（timeline）是理解的本质。如果你知道会发生什么，例如，当进程出现页故障时，你就可以真正了解虚拟内存的运行方式。如果你理解日志文件系统将块写入磁盘时发生的情况，就已经迈出了掌握存储系统的第一步。

整本书中有许多“补充”和“提示”，为主线讲解增添了一些趣味性。“补充”倾向于讨论与主要文本相关的内容（但可能不是必要的）；“提示”往往是一般经验，可以应用于所构建的系统。

在整本书中，我们使用最古老的教学方法之一——对话（dialogue）。这些对话用于介绍主要的主题概念，并不时地复习这些内容。这也让我们得以用更幽默的方式写作。好吧，你觉得它们是有用还是幽默，完全是另一回事。

在每一个主要部分的开头，我们将首先呈现操作系统提供的抽象（abstraction），然后在后续章节中介绍提供抽象所需的机制、策略和其他支持。抽象是计算机科学各个方面的基础，因此它在操作系统中也是必不可少的。

在所有的章节中，我们尝试使用可能的真实代码（real code），而非伪代码（pseudocode）。因此书中几乎所有的示例，你应该能够自己输入并运行它们。在真实系统上运行真实代码是了解操作系统的最佳方式，因此建议你尽可能这样做。

在本书的各个部分，我们提供了一些作业（homework），确保你进一步理解书中的内容。其中许多作业都是对操作系统的一些模拟（simulation）程序。你应该下载作业，并运行它们，以此来测验自己。作业模拟程序具有以下特征：通过给它们提供不同的随机种子，你可以产生几乎无限的问题，也可以让模拟程序为你解决问题。因此，你可以一次又一次地自测，直至很好地理解了这些知识。

本书最重要的附录是一组项目（project），可供你通过设计、测试和实现自己的代码，来了解真实系统的工作原理。所有项目（以及上面提到的代码示例）都是使用 C 编程语言（C programming language）[KR88]编写的。C 是一种简单而强大的语言，是大多数操作系统的基础，因此值得添加到你的工具库中。附录中含有两种类型的项目（请参阅在线附录中的想法）。第一类是系统编程（system programming）项目。这些项目非常适合那些不熟悉 C 和 UNIX，并希望学习如何进行底层 C 编程的人。第二类基于在麻省理工学院开发的实际操作系统内核，称为 xv6 [CK+08]。这些项目非常适合已经有一些 C 的经验并希望深入研究操作系统的学生。在威斯康星大学，我们以 3 种不同的方式开课：系统编程、xv6 编程，或两者兼而有之。

致使用本书作为教材的教师

这门课程很适合 15 周的学期，因此授课教师可以在合理的深度范围内讲授大部分主题。如果是 10 周的学期，那么可能需要从每个部分中删除一些细节。还有一些章节是关于虚拟机监视器的，我们通常会在学期的某个时候插入这些章节，或者在虚拟化部分的结尾处，抑或在接近课程结束时作为补充。

本书中的并发主题比较特别。它是许多操作系统书籍中靠前的主题，而在本书中是直到学生了解了 CPU 和内存的虚拟化之后才开始讲解的。根据我们近 15 年来教授本课程的经验，学生很难理解并发问题是如何产生的，或者很难理解人们试图解决它的原因。那是因为他们还不了解地址空间是什么、进程是什么，或者为什么上下文切换可以在任意时间点发生。然而，一旦他们理解了这些概念，那么再引入线程的概念和由此产生的问题就变得相当容易，或者至少比较容易。

我们尽可能使用黑板（或白板）来讲课。在着重强调概念的时候，我们会将一些主要的想法和例子带进课堂，并在黑板上展示它们。讲义有助于为学生提供需要解决的具体问题。在着重强调实践的时候，我们就将笔记本电脑连上投影仪，展示实际代码。这种授课风格特别适用于并发的内容以及所有的讨论部分。在这些部分中，教师可以向学生展示与其项目相关的代码。我们通常不使用幻灯片来讲课，但现在我们已经为那些喜欢这种演示风格的人提供了一套教学 PPT。

如果你想要任何这些教学辅助材料，请给 contact@epubit.com.cn 发电子邮件。

最后一个请求：如果你使用免费在线章节，请直接访问作者网站。这有助于我们跟踪

使用情况（过去几年中，本书英文版下载超过 100 万次!），并确保学生获得最新和最好的版本。

致使用本书上课的学生

如果你是读这本书的学生，那么我们很荣幸能够提供一些材料来帮助你学习操作系统的知识。我们至今还能够回想起我们使用过的一些教科书（例如，Hennessy 和 Patterson 的著作[HP90]，这是一本关于计算机架构的经典著作），并希望这本书能够成为你美好的回忆之一。

你可能已经注意到，这本书英文版的在线版本是免费的，并且可在线获取[①]。有一个主要原因：教科书一般都太贵了。我们希望，这本书是新一波免费材料中的第一本（指电子版），以帮助那些寻求知识的人—— 无论他们来自哪个国家，或者他们愿意花多少钱购买一本书。

我们也希望，在可能的情况下，向你指出书中大部分材料的原始资料—— 多年来的优秀论文和人物，他们让操作系统领域成为现在的样子。想法不会凭空产生，它们来自聪明勤奋的人（包括众多图灵奖获得者[②]），因此如果有可能，我们应该赞美这些想法和人。我们希望这样做能有助于更好地理解已经发生的变革，而不是说好像我们写这本书时那些思想一直就存在一样[K62]。此外，也许这样的参考文献能够鼓励你深入挖掘，而阅读该领域的著名论文无疑是良好的学习方法之一。

致谢

这里感谢帮助我们编写本书的人。重要的是，你的名字可以出现在这里！但是，你必须提供帮助。请向我们发送一些反馈，帮助完善本书。你可能会出名！或者，至少在某本书中有你的名字。

到目前为止，提供帮助的人包括 Abhirami Senthilkumaran*, Adam Drescher* (WUSTL), Adam Eggum, Aditya Venkataraman, Adriana Iamnitchi and class (USF), Ahmed Fikri*, Ajaykrishna Raghavan, Akiel Khan, Alex Wyler, Ali Razeen (Duke), AmirBehzad Eslami, Anand Mundada, Andrew Valencik (Saint Mary's), Angela Demke Brown (Toronto), B. Brahmananda Reddy (Minnesota), Bala Subrahmanyam Kambala, Benita Bose, Biswajit Mazumder (Clemson), Bobby Jack, Björn Lindberg, Brennan Payne, Brian Gorman, Brian Kroth, Caleb Sumner (Southern Adventist), Cara Lauritzen, Charlotte Kissinger, Chien-Chung Shen (Delaware)*, Christoph Jaeger, Cody Hanson, Dan Soendergaard (U. Aarhus), David Hanle (Grinnell), David Hartman, Deepika Muthukumar, Dheeraj Shetty (North Carolina State), Dorian Arnold (New

① 这里的题外话：我们在这里所说的“免费”并不意味着开源，也不意味着该书没有受到通常保护的版权——它是受到保护的！我们的意思是你可以下载章节，并使用它们来了解操作系统。为什么不是一本开源的书，不像 Linux 一样是一个开源内核？当你阅读它时，这本书应该像一次对话，某人向你解释某事。因此，这就是我们的方法。

② 图灵奖是计算机科学的最高奖项。它就像诺贝尔奖，但你可能从未听说过。

Mexico), Dustin Metzler, Dustin Passofaro, Eduardo Stelmaszczyk, Emad Sadeghi, Emily Jacobson, Emmett Witchel (Texas), Erik Turk, Ernst Biersack (France), Finn Kuusisto*, Glen Granzow (College of Idaho), Guilherme Baptista, Hamid Reza Ghasemi, Hao Chen, Henry Abbey, Hrishikesh Amur, Huanchen Zhang*, Huseyin Sular, Hugo Diaz, Itai Hass (Toronto), Jake Gillberg, Jakob Olandt, James Perry (U. Michigan-Dearborn)*, Jan Reineke (Universität des Saarlandes), Jay Lim, Jerod Weinman (Grinnell), Jiao Dong (Rutgers), Jingxin Li, Joe Jean (NYU), Joel Kuntz (Saint Mary's), Joel Sommers (Colgate), John Brady (Grinnell), Jonathan Perry (MIT), Jun He, Karl Wallinger, Kartik Singhal, Kaushik Kannan, Kevin Liu*, Lei Tian (U. Nebraska-Lincoln), Leslie Schultz, Liang Yin, Lihao Wang, Martha Ferris, Masashi Kishikawa (Sony), Matt Reichoff, Matty Williams, Meng Huang, Michael Walfish (NYU), Mike Griepentrog, Ming Chen (Stonybrook), Mohammed Alali (Delaware), Murugan Kandaswamy, Natasha Eilbert, Nathan Dipiazza, Nathan Sullivan, Neeraj Badlani (N.C. State), Nelson Gomez, Nghia Huynh (Texas), Nick Weinandt, Patricio Jara, Perry Kivolowitz, Radford Smith, Riccardo Mutschlechner, Ripudaman Singh, Robert Ordòñez and class (Southern Adventist), Rohan Das (Toronto)*, Rohan Pasalkar (Minnesota), Ross Aiken, Ruslan Kiselev, Ryland Herrick, Samer Al-Kiswany, Sandeep Ummadi (Minnesota), Satish Chebrolu (NetApp), Satyanarayana Shanmugam*, Seth Pollen, Sharad Punuganti, Shreevatsa R., Sivaraman Sivaraman*, Srinivasan Thirunarayanan*, Suriyhaprakhas Balaram Sankari, Sy Jin Cheah, Teri Zhao (EMC), Thomas Griebel, Tongxin Zheng, Tony Adkins, Torin Rudeen (Princeton), Tuo Wang, Varun Vats, William Royle (Grinnell), Xiang Peng, Xu Di, Yudong Sun, Yue Zhuo (Texas A&M), Yufui Ren, Zef RosnBrick, Zuyu Zhang。特别感谢上面标有星号的人，他们的改进建议尤其重要。

此外，衷心感谢 Joe Meehean 教授（Lynchburg）为每一章所做的详细注解，感谢 Jerod Weinman 教授（Grinnell）和他的全班同学提供的令人难以置信的小册子，感谢 Chien-Chung Shen 教授（Delaware）的细致阅读和建议，感谢 Adam Drescher（WUSTL）的细致阅读和建议，感谢 Glen Granzow（College of Idaho）提供详细的评论和建议，感谢 Michael Walfish（NYU）详细的改进建议。上述所有人都给予本书作者巨大的帮助，优化了本书的内容。

另外，非常感谢这些年来参加 537 课程的数百名学生。特别是 2008 年秋季课程的学生，鼓励我们第一次以书面形式写下了这些讲义（他们厌倦了没有任何类型的教科书可读——有进取心的学生！），然后不吝称赞，让我们继续前行（一位同学在那一年的课程评估中喜不自禁地说："老天爷！你们完全应该写一本教科书！"）。

我们也非常感谢那些参加 xv6 项目实验课程的少数人，这个实验课程大部分现已纳入主要的 537 课程。2009 年春季班的 Justin Cherniak，Patrick Deline，Matt Czech，Tony Gregerson，Michael Griepentrog，Tyler Harter，Ryan Kroiss，Eric Radzikowski，Wesley Reardan，Rajiv Vaidyanathan 和 Christopher Waclawik。2009 年秋季班的 Nick Bearson，Aaron Brown，Alex Bird，David Capel，Keith Gould，Tom Grim，Jeffrey Hugo，Brandon Johnson，John Kjell，Boyan Li，James Loethen，Will McCardell，Ryan Szaroletta，Simon Tso 和 Ben Yule。2010 年春季班的 Patrick Blesi，Aidan Dennis-Oehling，Paras Doshi，Jake Friedman，Benjamin Frisch，Evan Hanson，Pikkili Hemanth，Michael Jeung，Alex Langenfeld，Scott Rick，Mike Treffert，Garret Staus，Brennan Wall，Hans Werner，Soo -Young Yang 和 Carlos Griffin。

虽然没有直接帮助这本书的写作，但我们的研究生教会了我们很多关于系统的知识。我们在威斯康星大学时经常与他们交谈，并且所有真正的工作都是他们做的——通过告诉我们他们在做什么，我们每周都能学习到新事物。下面的列表包括我们已发布论文的当前和以前的学生，带有星号标志的名字是在我们的指导下获得博士学位的人：Abhishek Rajimwale，Andrew Krioukov，Ao Ma，Brian Forney，Chris Dragga，Deepak Ramamurthi，Florentina Popovici *，Haryadi S. Gunawi *，James Nugent，John Bent *，Jun He，Lanyue Lu，Lakshmi Bairavasundaram * ，Laxman Visampalli，Leo Arulraj，Meenali Rungta，Muthian Sivathanu *，Nathan Burnett *，Nitin Agrawal *，Ram Alagappan，Sriram Subramanian *，Stephen Todd Jones *，Suli Yang，Swaminathan Sundararaman*，Swetha Krishnan，Thanh Do*，Thanumalayan S. Pillai，Timothy Denehy*，Tyler Harter，Venkat Venkataramani，Vijay Chidambaram，Vijayan Prabhakaran *，Yiyi Zhang *，Yupu Zhang *，Zev Weiss。

最后要感谢 Aaron Brown，他多年前（2009 年春季）首次参加该课程，接着参加了 xv6 实验课程（2009 年秋季），最后还成为了两个课程的研究生助教（从 2010 年秋季到 2012 年春季）。他不知疲倦的工作极大地改善了项目的状态（特别是 xv6 项目），因此有助于改善威斯康星大学无数本科生和研究生的学习体验。正如 Aaron 所说的（以他通常的简洁方式）：“谢谢。”

最后的话

叶芝有一句名言：“教育不是注满一桶水，而是点燃一把火。”他说得既对也错[①]。你必须“给桶注一点水”，这本书当然可以帮助你完成这部分的教育。毕竟，当你去 Google 面试时，他们会问你一个关于如何使用信号量的技巧问题，确切地知道信号量是什么感觉真好，对吧？

但是，叶芝的主要观点显而易见：教育的真正要点是让你对某些事情感兴趣，可以独立学习更多关于这个主题的东西，而不仅仅是你需要消化什么才能在某些课程上取得好成绩。正如我们的父亲（雷姆兹的父亲 Vedat Arpaci）曾经说过的，“在课堂以外学习。”

我们编写本书以激发你对操作系统的兴趣，让你能自行阅读有关该主题的更多信息，进而与你的教授讨论该领域正在进行的所有令人兴奋的研究，甚至参与这些研究。这是一个伟大的领域，充满了激动人心和精妙的想法，以深刻而重要的方式塑造了计算历史。虽然我们知道这种火不会为你们所有人点燃，但我们希望这能对许多人，甚至是少数人有所帮助。因为一旦火被点燃，那你就真正有能力做出伟大的事情。因此，教育过程的真正意义在于：前进，学习许多新的和引人入胜的主题，通过学习不断成熟，最重要的是，找到能为你点火的东西。[②]

威斯康星大学计算机科学教授　　雷姆兹和安德莉亚夫妇

① 如果他真的说了这句话。与许多名言一样，这句名言的历史也是模糊不清的。

② 如果这听起来像我们承认过去曾是纵火犯，那你可能理解错了。如果这听起来很俗气，好吧，因为它确实是的，但你必须原谅我们。

参考资料

[CK+08]“The xv6 Operating System”Russ Cox, Frans Kaashoek, Robert Morris, Nickolai Zeldovich.
xv6 是作为原来 UNIX 版本 6 的移植版开发的，它代表了通过一种美观、干净、简单的方式来理解现代操作系统。

[F96]“Six Easy Pieces: Essentials Of Physics Explained By Its Most Brilliant Teacher”Richard P. Feynman
Basic Books, 1996
这本书摘取了 1993 年的《费曼物理学讲义》中 6 个最简单的章节。如果你喜欢物理学，那么就读一读这本很优秀的读物吧。

[HP90]“Computer Architecture a Quantitative Approach”(1st ed.) David A. Patterson and John L. Hennessy
Morgan-Kaufman, 1990
在读本科时，这本书成为了我们去攻读研究生的动力。我们后来都很高兴与 Patterson 合作，他为我们研究事业基础的奠定给予了极大的帮助。

[KR88]“The C Programming Language”Brian Kernighan and Dennis Ritchie Prentice-Hall, April 1988
每个人都应该拥有一本由发明该语言的人编写的 C 编程参考书。

[K62]“The Structure of Scientific Revolutions”Thomas S. Kuhn
University of Chicago Press, 1962
这是关于科学过程基础知识的著名读物，包括科学过程的整理工作、异常、危机和变革。我们要做的是整理工作。