HZ BOOKS
华章教育

计 算 机 科 学 丛 书

原书第5版

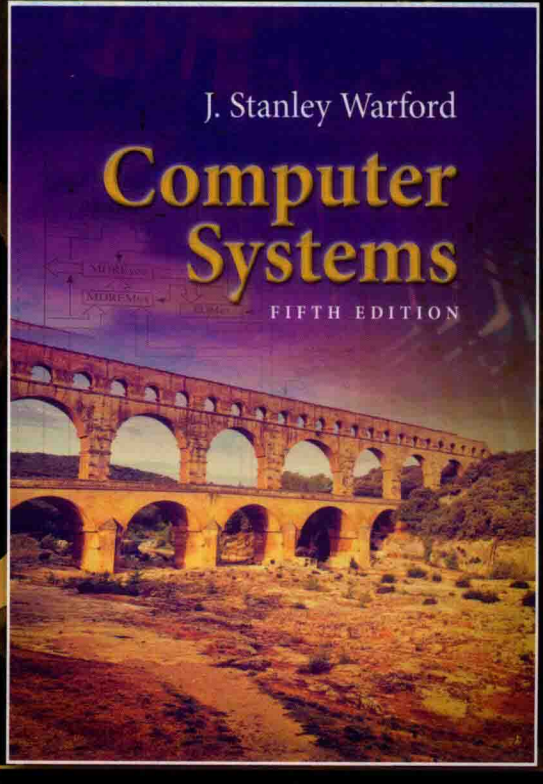
计算机系统

核心概念及软硬件实现

[美] J. 斯坦利·沃法德（J. Stanley Warford） 著
贺莲 龚奕利 译

Computer Systems

Fifth Edition



机械工业出版社
China Machine Press

计 算 机 科 学 丛 书

原书第5版

计算机系统

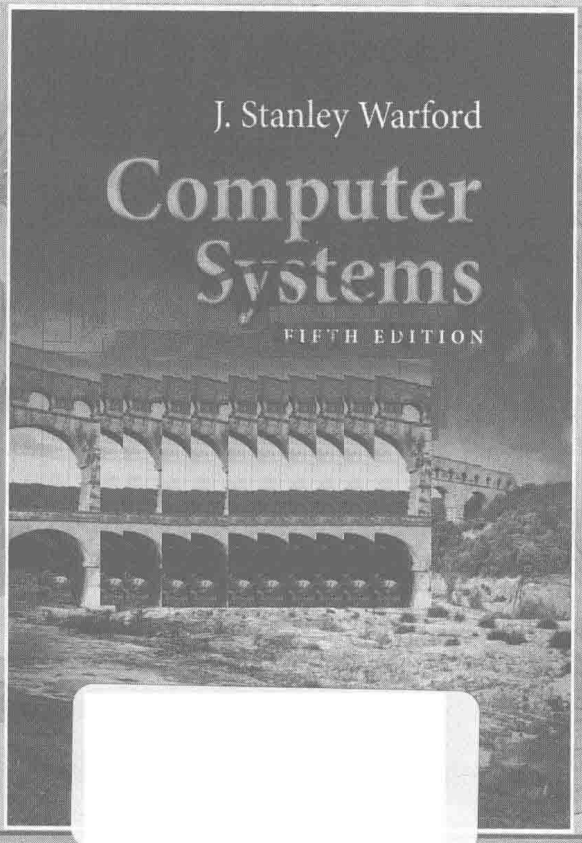
核心概念及软硬件实现

[美] J. 斯坦利·沃法德（J. Stanley Warford）著

贺莲 龚奕利 译

Computer Systems

Fifth Edition



机械工业出版社
China Machine Press

图书在版编目（CIP）数据

计算机系统：核心概念及软硬件实现（原书第5版）/（美）J. 斯坦利·沃法德（J. Stanley Warford）著；贺莲，龚奕利译．—北京：机械工业出版社，2019.1
（计算机科学丛书）
书名原文：Computer Systems, Fifth Edition

ISBN 978-7-111-61684-9

I. 计… II. ① J… ② 贺… ③ 龚… III. 计算机系统 IV. TP303

中国版本图书馆CIP数据核字（2018）第301850号

本书版权登记号：图字 01-2016-3492

本书以计算机系统的七层结构为主线，涵盖逻辑门、微代码、指令集架构、操作系统、汇编、高级语言和应用，全面介绍计算机组成、汇编语言和计算机体系结构的核心思想及软硬件实现方法。新版采用Pep/9虚拟机，清晰地阐释了经典冯·诺依曼机器的基本概念，同时包含完整的程序示例和丰富的习题，在理论与实践相结合的基础上，注重内容的广度和深度。

本书适合作为高等院校计算机专业的课程教材，也可供相关技术人员阅读参考。

出版发行：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：曲 熠　　责任校对：殷 虹
印　刷：北京瑞德印刷有限公司　　版　次：2019年1月第1版第1次印刷
开　本：185mm×260mm 1/16　　印　张：36.75
书　号：ISBN 978-7-111-61684-9　　定　价：99.00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

文艺复兴以来，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；也正是这样的优势，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，从它们现有的数百种教材中甄选出 Andrew S. Tanenbaum、Bjarne Stroustrup、Brian 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相助，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，有的还专门为其书的中译本作序。迄今，“计算机科学丛书”已经出版了近 500 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

华章科技图书出版中心

译者序

Computer Systems, Fifth Edition

本书以计算机层次结构为主线，从 LG1 逻辑门层到 App7 应用层，内容涵盖数字逻辑、计算机组成、汇编语言和计算机体系结构等方面，主要包括计算机组织结构、时序电路、布尔代数和逻辑门、进程和存储管理、信息表示、汇编语言、C 语言以及计算机系统。与之前的版本相比，本书使用的虚拟机从 Pep/8 变成了 Pep/9，两者在机器指令集上存在不同，此外，Pep/9 还采用了内存映射 I/O，改进了部分指令的助记符，扩展了 MIPS 内容。

本书内容翔实，着重于基础计算概念，强调解决问题，使用一致的机器模型，配以完整的程序示例，在理论与实践相结合的基础上，注重内容的广度和深度。本书章节结构明晰，适用于相关课程教学，在进行课程设计时，可以根据需要选择不同的章节进行组合。

感谢机械工业出版社华章公司的编辑姚蕾，由于她的热心推荐，我们才有幸翻译了这本优秀的专业书籍。

在翻译中我们秉持认真细致的态度，但是由于能力所限，还是会存在错误与疏漏，希望广大读者批评指正。

贺莲　龚奕利

2018 年 11 月于珞珈山

本书清晰详尽、循序渐进地揭示了计算机组成、汇编语言和计算机体系结构的核心思想。本书大部分以虚拟机 Pep/9 为基础，该虚拟机用于讲解经典冯·诺依曼机器的基本概念。这种方式的优点是，既教授了计算机科学的核心概念，又不会与相关课程的许多无关细节纠缠不清。该方式还为学生奠定了基础，鼓励他们思考计算机科学的基本主题。本书的范围也比较广泛，重点强调了与硬件及其相关软件的处理有关但却少有提及的计算机科学主题。

内容一览

计算机运行于多个抽象层，高抽象层上的编程只是其中的一部分。本书以图 P-1 所示的分层结构为基础，提出了计算机系统的统一概念。

图 P-1　典型计算机系统的层次结构

按照图 P-1 的层次结构，本书分为七个部分：

App7 层	应用
HOL6 层	高级语言
ISA3 层	指令集架构
Asmb5 层	汇编
OS4 层	操作系统
LG1 层	逻辑门
Mc2 层	微代码

用文字描述时通常是按照从上到下的顺序，从最高层到最低层。把 ISA3 层放在 Asmb5 层之前，以及把 LG1 层放在 Mc2 层之前讨论是为了教学目的。对这两个特例来说，暂时将顺序变为从下往上会更加自然一些，因为这样一来在构建高层时可以使用低层模块。

App7 层。App7 层是关于应用程序的独立一章，叙述了抽象层次的思想与二进制信息，并为本书其他章节搭建了框架。这一章还以典型计算机应用程序示例的方式描述了一些关系数据库的概念。

HOL6 层。HOL6 层也是一章，回顾了 C 编程语言。这一章假设学生已经学习过一些命令式语言，比如 Java 或 Python，不一定是 C。如果必要的话，指导老师可以轻易地把 C 语言示例翻译为其他常见的 HOL6 层语言。

这一章的重点在于 C 内存模型，包括全局和局部变量、带参数的函数，以及动态分配的变量。此外，还讲解了递归，因为它要依赖运行时堆栈的内存分配机制。函数调用中的内存分配过程阐释得相当详细，而且后续章节还会在较低抽象层上回顾这个机制。

ISA3 层。ISA3 是指令集架构层。这一层用两章来描述 Pep/9——一种用于说明计算机概念的虚拟机。Pep/9 是一个小型的复杂指令集计算机（CISC），也是冯·诺依曼计算机。它的中央处理器（CPU）包含一个加法器、一个变址寄存器、一个程序计数器、一个栈指针

寄存器和一个指令寄存器。它有八种寻址方式：立即数寻址、直接寻址、间接寻址、栈相对寻址、栈相对间接寻址、变址寻址、栈变址寻址和栈间接变址寻址。在模拟只读存储器（ROM）中，Pep/9 的操作系统可以从学生的文本文件中加载并执行十六进制格式的程序。学生可以在 Pep/9 模拟器上运行小程序，学习执行不会改变内存值的 ROM 存储指令。

学生将学习信息表示和位级计算机组成的基本原理。由于本书的中心主题是计算机各层间的相互关系，因此，Pep/9 相关章节展示了 ASCII 表示（ISA3 层）和 C 的 char 类型变量（HOL6 层）之间的关系。此外，这些章节还展示了补码表示（ISA3 层）和 C 的 int 类型变量（HOL6 层）之间的关系。

Asmb5 层。Asmb5 是汇编层，它把汇编器的概念表示为两个层次——汇编层和机器层——之间的翻译器。它引入了 Asmb5 符号和符号表。

这里是统一方法派上用场的地方。第 5 章和第 6 章将编译器表示为从高级语言到汇编语言的翻译器。前面学生已经学习了一种特定的 HOL6 层语言 C 和一种特定的冯·诺依曼型机器 Pep/9。这两章通过展示层次之间的对应关系来继续揭示它们之间的关系，其中包括：HOL6 层的赋值语句与 Asmb5 层的装入 / 存储指令；HOL6 层的循环和 if 语句与 Asmb5 层的分支指令；HOL6 层的数组与 Asmb5 层的变址寻址；HOL6 层的过程调用与 Asmb5 层的运行时栈；HOL6 层的函数和过程参数与 Asmb5 层的栈相对寻址；HOL6 层的 switch 语句与 Asmb5 层的跳转表；HOL6 层的指针与 Asmb5 层的地址。

统一方法之美就在于可以在较低层次上实现 C 章节中的例子。比如，第 2 章递归示例说明的运行时栈就直接对应于 Pep/9 主存的硬件栈。学生可以通过两个层次之间的手动翻译来理解编译过程。

这种方法为讨论计算机科学中的核心问题提供了一种很自然的环境。例如，本书介绍了 HOL6 层的结构化编程，可以和 Asmb5 层的非结构化编程的可能性进行对比。书中讨论了 goto 争议、结构化编程 / 效率之间的折中，给出了两个层次上语言的实际例子。

第 7 章向学生介绍了计算机科学理论。现在学生已经对如何将高级语言翻译为汇编语言有了直观的了解，那么，我们就要提出所有计算中最基本的问题：什么可以被自动化？理论在这里自然又合适，因为学生现在已经知道了什么是编译器（自动化翻译器）必须做的。他们通过识别 C 和 Pep/9 汇编语言的语言符号来学习语法分析和有限状态机——确定性的和非确定性的。这一章包含了两种小语言之间的自动翻译器，说明了词法分析、语法分析和代码生成。词法分析器是有限状态机的实现。还有比这更自然的介绍理论的方法吗？

OS4 层。OS4 层用两章来讲述操作系统。第 8 章是关于进程管理的，其中有两节讲解了 Pep/9 操作系统的概念，一节是装载器，另一节是陷阱处理程序。七条指令具有产生软件陷阱的未实现的操作码。操作系统将用户正在运行进程的进程控制块保存到系统栈，中断服务例程解释该指令。通过具体实现一个挂起进程来强化操作系统中运行和等待进程的经典状态转换图。第 8 章还描述了并发进程和死锁。第 9 章阐述了关于主存和磁盘存储器的存储管理。

LG1 层。LG1 层用两章来讲述组合电路与时序电路。从布尔代数的定理开始，第 10 章强调了计算机科学的数学基础的重要性。它展示了布尔代数和逻辑门之间的关系，然后描述了一些常用的逻辑设备，包括一个完整的 Pep/9 算术逻辑单元（ALU）的逻辑设计。第 11 章用时序电路的状态转换图讲解了有限状态机的基本概念，还描述了常见的计算机子系统，包括双向总线、内存芯片以及双端口存储器组。

Mc2 层。第 12 章描述了 Pep/9 CPU 的微程序设计控制部分，给出了一些示例指令和寻址方式的控制序列，还提供了有关其他指令和寻址方式的大量练习。这一章还介绍了装入 / 存储架构的概念，对比了 MIPS 精简指令集计算机（RISC）和 Pep/9 复杂指令集计算机（CISC）。此外，还通过对高速缓存、流水线、动态分支预测以及超标量机器的描述，介绍了一些性能问题。

教学建议

本书涵盖内容相当广泛，教师在设计课程时可以省略一些内容。我把第 1 ～ 7 章用于计算机系统课程，第 10 ～ 12 章用于计算机组成课程。

本书中，第 1 ～ 5 章必须顺序讲解，第 6 章和第 7 章可以按任意顺序讲授。我通常会省略第 6 章而直接讲第 7 章，开始一个大型软件项目——为 Pep/9 汇编语言的子集写汇编器，这样学生在一学期中有足够的时间完成它。第 11 章明显依赖于第 10 章，但这两章都与第 9 章无关，因此，第 9 章可以省略。图 P-2 是章节关系依赖示意图，总结了可以省略哪些章节。

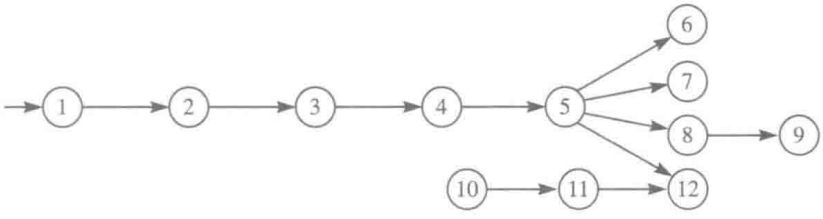


图 P-2　章节关系依赖图

第 5 版的变化

第 5 版中的每一章都有改进，主要集中在两个方面：一个是虚拟机从 Pep/8 变为 Pep/9；另一个是内容上的变化。第二点涉及面太大，无法一一列出，这里只给出其中的主要变化。

- HOL6 语言——第 5 版中的 HOL6 语言从 C++ 变为 C。C 作为系统编程语言更为常见，而且也更加适合计算机系统文本。上一版指出 C++ 的内存模型包括在内存固定位置分配的全局变量、在运行时栈上分配的局部变量与参数以及从堆分配的动态变量。但 C++ 是一种面向对象（OO）的语言，它的内存模型要复杂得多。相比 C++，上述模型用 C 语言表述要更加精确一些。变化体现在三个地方：示例程序的输入 / 输出（I/O）用 scanf() 和 printf() 替代了 cin 和 cout；在 C 的传递引用机制里，实际参数显式地使用地址运算符 &，形式参数则使用对应的指针类型；堆的内存分配用 malloc 替代了 new。
- 补充示例——第 5 版把之前每一章的简要人物传记改为不同的补充示例，放在特殊格式的文本框中。每一个补充示例都是该章所描述概念的真实示例。大多数章节都是描述 Pep/9 虚拟机的，这些章节的补充示例给出了 Intel x86 架构中相应部分的实现。新补充示例提供了与该架构一致的运行实例，学生能更好地理解虚拟机概念与真实实现之间是如何对应的。
- 新主题与扩展主题——第 1 章现在强调的是如何利用性能公式和带宽概念在空间和时间上量化二进制信息。QR 代码和颜色显示是这些概念的详细示例。第 3 章描述

了 Unicode、UTF-32 和 UTF-8 编码。第 4 章讨论了大端顺序和小端顺序。第 7 章用 Java 而不是 C++ 语言实现示例翻译器。第 12 章的微代码示例使用了 Pep/9 CPU 模拟器新的 UnitPre 和 UnitPost 特性。双字节数据总线现在得到了模拟器的支持，这使得讨论主题可以扩展到内存对齐问题和新的 .ALIGN 伪指令。

Pep/9 取代了前两版所用的虚拟机 Pep/8。由于这两种机器指令集的不同，Pep/8 的源程序和目标程序不能与 Pep/9 的兼容。不过只有几条指令会受到影响，多数还是和原来一样，包括八种寻址方式。Pep/9 有下述变化：

- RET 代替了 RETn——Pep/8 有八种返回语句，分别为 RET0，RET1，…，RET7。RETn 从运行时栈释放 *n* 个字节，然后执行从函数调用的返回。这样做的理由是，返回之前总要释放局部变量，所以，如果汇编语言程序员不需要在返回前用 ADDSP 指令显式释放局部变量的话，程序就会更短一些。

  虽然这种 ISA 设计在体系结构原理上可能是合理的，但在教学上却存在缺陷。问题是学生必须学习两种不同的概念：数据的释放机制和控制流的返回机制。将这两种不同的概念组合在一条语句中，在学习时可能会造成混淆。现在，Pep/9 要求学生用 ADDSP 语句显式地释放局部变量。另一个格式上的优势是，函数结束时显式使用 ADDSP 释放局部变量直接与函数开始时用 SUBSP 分配局部变量相对应。
- 内存映射 I/O——对 Pep/8 指令集中的所有指令来说，最不切实际的就是用于字符输入 / 输出的 CHARI 和 CHARO。大多数真实计算机系统都把输入 / 输出映射到内存，Pep/9 现在就是这样设计的。在新指令集中没有原生的输入 / 输出指令。相反，Pep/8 的指令

```
CHARI alpha,ad
```

被如下 Pep/9 指令所替代，其中的 charIn 是输入设备。

```
LDBA charIn,d ;Load byte to A from charIn
STBA alpha,ad ;Store byte from A to alpha
```

而 Pep/8 指令

```
CHARO beta,ad
```

也被如下 Pep/9 指令所替代，其中的 charOut 是输出设备。

```
LDBA beta,ad ;Load byte to A from beta
STBA charOut,d ;Store byte to charOut
```

在上述指令中，ad 表示该指令的任意一种合法的寻址方式。符号 charIn 和 charOut 在 Pep/9 操作系统中定义，并作为机器向量保存在内存底部。它们的值会自动包含进汇编器的符号表中。

内存映射 I/O 的一个缺点是，Pep/8 程序中的每一个 CHARI 和 CHARO 语句现在都需要编写为两条语句，这使得程序变长。不过，陷阱指令 DECI、DECO、STRO 与原来一样，这使得问题得以控制，因为原生 I/O 指令隐藏在它们的陷阱例程中。

学生通过从输入设备地址装入以及向输出设备地址存储直接学习内存映射 I/O 是如何工作的，这是非常有利的。这个要求还说明了内存映射的概念和用法，在 Pep/8

中，这是学生想要避开的问题。此外，这与第 11 章的地址解码示例也有很好的关联，它展示了怎样把一个八端口 I/O 芯片连接到内存映射。

- *新的原生指令* CPBr——在 Pep/8 中，字节量必须用 CPr 比较，该指令比较两字节的对象。因此，比较也包括了高字节，有时在比较之前要清除寄存器的高字节。这使得进行字节比较的汇编代码有些复杂。

  CPBr 是新的字节比较指令，通过设置状态位而不再考虑寄存器的高字节。其结果代码更易于理解和编写。

- *改进的助记符*——Pep/9 重新命名了比较指令、装入指令和存储指令的助记符，如图 P-3 所示。新方案保留了比较中的 CP、装入中的 LD 和存储中的 ST，同时又在这组指令中一致使用字母 W 表示字（这是现在需要的），字母 B 表示字节。这个命名规则一致性更强，并且针对学生有遗忘“字”（Pep 计算机中的双字节）含义的倾向，把字母 W 加到双字节指令的助记符中加强了对“字”的强调。

指令	Pep/9	Pep/8
比较字	CPWr	CPr
比较字节	CPBr	不可用
装入字	LDWr	LDr
装入字节	LDBr	LDBYTEr
存储字	STWr	STr
存储字节	STBr	STBYTEr

图 P-3　新的 Pep/9 指令助记符

- *新的陷阱指令* HEXO——在 Pep/9 指令集中，与 RETn 和字符 I/O 指令一起删除的还有陷阱指令 NOP2 和 NOP3，取而代之的是一条非一元陷阱指令 HEXO。HEXO 代表的是十六进制输出，存在于 Pep/7 中，现在在 Pep/9 中再次出现。它输出的一个字包含四个十六进制字符。
- *寻址方式的名称*——Pep/9 将栈变址间接寻址改为栈间接变址寻址，相应的汇编器符号也从 sxf 改为 sfx。这个变化更准确地反映了寻址方式的语义，因为栈间接操作发生在变址操作之前。
- *扩展的 MIPS 内容*——MIPS 架构仍然是与 Pep/9 CISC 模型形成对比的 RISC 模型。其内容得到了扩展，对所有的 MIPS 寻址方式、指令类型以及它们在 LG1 层上的实现进行了更加广泛和系统的描述。在说明 MIPS 架构的同时，第 5 版也包括了对 RISC 设计原则更加宽泛的阐述。

本书特性

本书有几个独特的方面，使之有别于其他计算机系统、汇编语言和计算机组成方面的书籍。

- *以概念为核心*——许多教科书试图跟上领域的变化，包括最新的技术发展。比如，最新外围设备的通信协议规范。通常，这类书通篇都在描述性地解释“设备是如何工作的”。本书避开了这类资料，而只选择基础的计算概念，掌握了这些就有了理解当前和未来技术的基础。例如，在掌握空间 / 时间折中概念的时候，让学生实践数字电

路设计问题的方案比单纯阅读笼统的描述要重要得多。再举一个例子，通过学习如何在 ISA 指令的微代码实现中合并周期来掌握硬件并行的概念，才是最好的。

- *强调解决问题*——在讨论某个主题时，如果只是采用听讲或阅读的方式，那么学生能记住的内容就很少；如果采用的是实践的方式，他们能记住的就会更多。本书强调解决问题，全书章节后面有将近 400 道练习，其中很多练习有多个部分。这些练习不会让学生重复课本中的原话，而是要求量化地解答、分析或者设计系统某个抽象层次上的程序或数字电路。
- *一致的机器模型*——Pep/9 机器是一个小型的 CISC 计算机，是描述系统所有层次的载体。学生可以清晰地看到抽象层次之间的关系，因为他们要在所有的层次上为这个机器编程或者设计数字电路。例如，当在 LG1 层设计 ALU 组件时，他们知道 ALU 在 ISA3 层的实现中应该在哪个位置。通过像编译器那样把 C 程序翻译成汇编语言，他们将学到优化编译器和非优化编译器之间的差别。在不同层次上都使用同样的机器模型对学习活动来说在效率上有很大的优势，因为模型从上至下都是一致的。不过本书也讲述了 MIPS 机器，对比了 RISC 设计原理和微程序设计的 CISC 设计。
- *完整的程序示例*——许多计算机组成和汇编语言的书会受到代码片段综合征的影响。Pep/9 的内存模型、寻址方式和输入 / 输出特性使得学生能写出完整的程序，而不只是代码片段，这些程序执行和测试起来也很容易。真实的机器，特别是 RISC 机器，有复杂的函数调用协议，涉及寄存器分配、寄存器溢出和内存对齐限制之类的问题。Pep/9 是少数几种教学机之一（有可能是唯一一个），允许学生书写具有输入 / 输出的完整程序，可以使用全局变量和局部变量、全局数组和局部数组、传值调用和传引用调用、数组参数、具有转移表的 switch 语句、递归、具有指针的链式结构和堆。写完整程序的作业进一步实现了通过动手来学习的目标，而不是通过读代码片段来学习。
- *理论与实践相结合*——有些读者注意到了，讲述语言翻译原理的第 7 章在计算机系统书中不常见。这种现象可悲地说明了计算机科学课程体系甚至计算机科学领域自身存在的理论和实践之间的鸿沟。既然本书讲述了 HOL6 层的 C 语言、Asmb5 层的汇编语言和 ISA3 层的机器语言，而且都有一个目标，即理解层次之间的关系，那么一个更好的问题是："为什么不能包括讲述语言翻译原理的一章呢？"本书尽可能地加入理论以支持实践。例如，介绍布尔代数并将其作为一个公理系统，配合练习来证明定理。
- *广度和深度*——第 1 ～ 6 章中的内容对计算机系统或汇编语言编程的书来说是很典型的，第 8 ～ 12 章对计算机组成的书来说是很典型的。在一本书中包括这么广泛的内容是很独特的，而且还在一个完整机器的各个抽象层次上使用一致的机器模型。数字电路 LG1 层内容的深度也是很特别的，它使得 CPU 的组成部分不再神秘。例如，本书描述了 Pep/9 CPU 的复用器、加法器、ALU、寄存器、内存子系统和双向总线的实现。学生学习逻辑门层的实现后，没有概念上的空洞，而如果只是泛泛地描述，学生就只能选择相信而不能完全理解。

本书回答了这个问题："汇编语言编程和计算机组成在计算机科学课程体系中的位置是什么？"它提供了对无处不在的冯·诺依曼机器架构的深入理解。本书保持了独特的目标，

即提供本领域内所有主要知识域的综合概述，包括软件和硬件的结合、理论和实践的结合。

计算机科学课程体系 2013

本书第 1 版出版于 1999 年，从那时开始本书的目标就是通过一个虚拟机器（初版中是 Pep/6）来讲解典型计算机系统全部七个抽象层之间的关系，从而统一呈现计算机系统。当时（甚至现在也是）本书的内容跨越了多个标准课程，增加了知识面的宽度，这样的做法同时牺牲了标准汇编语言、操作系统、计算机体系结构、计算机组成和数字电路设计课程中传统内容的深度。

最新的 ACM/IEEE 课程指南——计算机科学课程体系 2013[⊖]（CS2013）特别指出领域快速扩张带来的课程挑战：

与计算机科学教育领域相关的主题越来越多样化，以及计算与其他学科的日益融合给这一努力带来了特殊的挑战。在教学内容增加和保持建议在本科教育背景下具备可行性和可操作性之间平衡是相当困难的。

对这一挑战的应对之一就是指导方针的不断演进，而这恰好就是本书从第 1 版起就坚持贯彻的方向。CS2013 重组了之前的“知识体”，删除了一些旧领域，增加了一些新领域。指南中的一个新知识领域（Knowledge Area，KA）就是系统基础：

在以前的课程内容中，典型计算系统的交互层——从硬件构建块到架构组织，到操作系统服务，再到应用程序执行环境——都表示为独立的 KA。而新的系统基础 KA 则为其他 KA 展示了统一的系统视角和共同的概念基础……

CS2013 中新系统基础 KA 的目标与本书的目标是一致的，即为计算机科学的其他主题提出一个“统一的系统视角”和共同的概念基础。本书是一本成熟的教材，它难得地满足了最新计算机科学课程体系指南的这一重要新目标。

教辅资源

从出版商网站（go.jblearning.com/warford5e）可以获取下列资源。

Pep/9 汇编器和模拟器。Pep/9 机器在 MS Windows、Mac OS X 和 UNIX/Linux 系统上都可运行。汇编器的特性包括：

- 集成的文本编辑器。
- 在源代码中发现错误的地方插入错误消息。
- 对学生友好的、十六进制的机器语言目标代码。
- 能跳过汇编器，直接用机器语言编写代码。
- 能够重定义触发同步陷阱的未实现操作码的助记符。

模拟器的特性包括：

- 模拟的 ROM，存储指令不能修改 ROM 中的内容。
- 在模拟的 ROM 中烧入了一个小的操作系统，包括一个装载器和一个陷阱处理程序。
- 一个集成的调试器，允许设置断点、单步执行、CPU 跟踪和内存跟踪。

⊖ *Computer Science Curricula 2013, Curriculum Guidelines for Undergraduate Degree Programs in Computer Science*, The Joint Task Force on Computing Curricula, Association for Computing Machinery (ACM) IEEE Computer Society, December 20, 2013.

- 能以任何组合跟踪应用程序、装载器或操作系统的选项。
- 具备从无限循环中恢复的能力。
- 能够通过为未实现操作码设计新的陷阱处理程序来修改操作系统。
- 从应用程序中构建的每个示例都成为课堂演示的有用工具。

Pep/9 CPU 模拟器。包括 MS Windows、Mac OS X 和 UNIX/Linux 系统版本，可以用在计算机组成课程中。CPU 模拟器的特性包括：

- 颜色编码的展示通路，根据多路复用器的控制信号跟踪数据流。
- 操作的单周期模式，用 GUI 输入每个控制信号，立即展示信号的效果。
- 操作的多周期模式，学生可以在集成的文本编辑器中编写 Mc2 微代码序列并执行它们以便实现 ISA3 指令。
- 第 5 版的新特点：针对应用程序的每个微代码问题和每个示例的单元测试。

课程课件。一套完整的课件，每章约有 50 ～ 125 页 PDF 格式的课程幻灯片。幻灯片包括课本中所有的图和总结信息，通常以标号的形式给出。不过其中没有太多的例子，给教师展示示例和指导讨论留出了空间。

考试题目。提供一组考试题目，包括参考信息，例如 ASCII 表、指令集表等，供考试和自学之用。

数字电路实验。6 个数字电路实验，能够让学生在物理实验电路板上亲身体验。这些实验说明了第 10 章和第 11 章的组合和时序设备，使用了许多本书中没讲到的电路。学生可以自学实际的数字设计和实现概念，这些超出了本书的讲述范围。实验时可以按照书中讨论的主题顺序，从组合电路开始，然后是时序电路和 ALU。

答案手册。附录中有部分练习的答案，全部练习和编程题的答案对采用本书作为教材的教师开放。

若想了解指导教师如何获得访问上述资源的资格，请联系 Jones & Bartlett Learning 的代理商。

致谢

Pep/1 有 16 条指令、一个累加器和一种寻址方式。Pep/2 增加了变址寻址。John Vannoy 用 ALGOL W 语言写了两个模拟器。Pep/3 有 32 条指令，用 Pascal 语言编写，是学生软件项目，由 Steve Dimse、Russ Hughes、Kazuo Ishikawa、Nancy Brunet 和 Yvonne Smith 完成。Harold Stone 在早期审阅中提出许多对 Pep/3 架构的改进意见，后来被加到 Pep/4 中，并延续到后续的机器中。Pep/4 有特殊的栈指令、模拟 ROM 和软件陷阱。Pep/5 有更正交的设计，允许任何指令使用任何寻址方式。John Rooker 写了 Pep/4 系统和早期的 Pep/5 版本。Gerry St. Romain 实现了 Mac OS 版本和 MS-DOS 版本。Pep/6 简化了变址寻址方式，也包括了一组完整的条件分支指令。John Webb 用 BlackBox 开发系统编写了跟踪功能。Pep/7 把安装的内存从 4KiB 增加到了 32KiB。Pep/8 把寻址方式的数量从 4 增加到 8，安装的内存增加到 64KiB。Pep/8 汇编器和模拟器的 GUI 版本由一组学生用 Qt 开发系统和 C++ 实现和维护，小组成员包括 Deacon Bradley、Jeff Cook、Nathan Counts、Stuartt Fox、Dave Grue、Justin Haight、Paul Harvey、Hermi Heimgartner、Matt Highfield、Trent Kyono、Malcolm Lipscomb、Brady Lockhart、Adrian Lomas、Scott Mace、Ryan Okelberry、Thomas Rampelberg、Mike Spandrio、Jack Thomason、Daniel Walton、Di Wang、Peter Warford 和 Matt Wells。Ryan Okelberry 也参

与编写了 Pep/8 CPU 模拟器。Luciano d'Ilori 编写了汇编器的命令行版本。最新版本的 Pep/8 和 Pep/8 CPU 以及当前版本的 Pep/9 和 Pep/9 CPU 由 Emily Dimpfl 和本书作者用 Qt 重新编写。

Tanenbaum 的《Structured Computer Organization》[⊖]比其他任何一本书都更大地影响了本书的编写。本书扩展了 Tanenbaum 书中的层次结构，在上面增加了高级语言层和应用层。

许多书稿审阅者、学生和前一版本的用户极大地影响了本版本的终稿，他们是：Kenneth Araujo、Ziya Arnavut、Wayne P. Bailey、Leo Benegas、Jim Bilitski、Noni Bohonak、Dan Brennan、Michael Yonshiki Choi、Christopher Cischke、Collin Cowart、Lionel Craddock、William Decker、Fadi Deek、Peter Drexel、Gerald S. Eisman、Victoria Evans、Mark Fienup、Paula Ford、Brooke Fugate、Robert Gann、David Garnick、Ephraim P. Glinert、John Goulden、Dave Hanscom、Michael Hennessy、Paul Jackowitz、Mark Johnson、Michael Johnson、Amitava Karmaker、Michael Kirkpatrick、Peter MacPherson、Andrew Malton、Robert Martin、Johnon Maxfield、John McCormick、Richard H. Mercer、Jonathon Mohr、Randy Molmen、Hadi Moradi、John Motil、Mohammad Muztaba Fuad、Peter Ng、Bernard Nudel、Carolyn Oberlink、Nelson Passos、Wolfgang Pelz、James F. Peters III、James C. Pleasant、Eleanor Quinlan、Glenn A. Richard、Gerry St. Romain、David Rosser、Sally Schaffner、Peter Smith、Harold S. Stone、Robert Tureman、J. Peter Weston 和 Norman E. Wright。Joe Piasentin 提供了艺术方面的咨询。有两个人极大地影响了 Pep/8 的设计：一位是 Myers Foreman，有关指令集的很多想法都来自于他；另一位是 Douglas Harms，他提出很多改进意见，其中之一是 MOVSPA 指令，使得可以用传引用方式传递局部变量。

Jones and Bartlett Learning 的策划编辑 Laura Pagluica、产品主管 Amy Rose、制作编辑 Vanessa Richards 和编辑助理 Taylor Ferracane 提供了宝贵的支持，很高兴与他们一起工作。Kristin Parker 设计的吸引人的封面正符合本书的风格。

我很幸运工作于一所致力于在本科教学中追求卓越的学校。佩珀代因（Pepperdine）大学的 Ken Perrin 提供了富有创造性的环境和专业的支持，正是在这种环境中，本书得以孕育成形。妻子 Ann 给予我无尽的支持，我要为本书占用的时间向她道歉，并送上我由衷的感谢。

J. Stanley Warford
于加州马里布

⊖ 这本书已由机械工业出版社出版，中文书名为《计算机组成：结构化方法》，第 6 版书号为 978-7-111-45380-2。——编辑注

目　录

Computer Systems, Fifth Edition

出版者的话

译者序

前言

第一部分　应用层（第7层）

第1章　计算机系统 …… 2

1.1　抽象层次 …… 2

1.1.1　艺术中的抽象 …… 3

1.1.2　文档中的抽象 …… 4

1.1.3　机构中的抽象 …… 5

1.1.4　机器中的抽象 …… 6

1.1.5　计算机系统中的抽象 …… 6

1.2　硬件 …… 7

1.2.1　中央处理单元 …… 8

1.2.2　主存储器 …… 9

1.2.3　磁盘 …… 10

1.3　软件 …… 11

1.3.1　操作系统 …… 12

1.3.2　软件分析与设计 …… 13

1.4　数字信息 …… 14

1.4.1　空间量化 …… 14

1.4.2　时间量化 …… 16

1.4.3　快速响应码 …… 18

1.4.4　图像 …… 21

1.5　数据库系统 …… 27

1.5.1　关系 …… 27

1.5.2　查询 …… 28

1.5.3　语言结构 …… 30

本章小结 …… 31

练习 …… 32

第二部分　高级语言层（第6层）

第2章　C …… 36

2.1　变量 …… 36

2.1.1　C编译器 …… 36

2.1.2　机器无关性 …… 37

2.1.3　C的内存模型 …… 37

2.1.4　全局变量和赋值语句 …… 38

2.1.5　局部变量 …… 40

2.2　控制流 …… 42

2.2.1　if/else语句 …… 42

2.2.2　switch语句 …… 43

2.2.3　while循环 …… 44

2.2.4　do循环 …… 44

2.2.5　数组和for循环 …… 45

2.3　函数 …… 46

2.3.1　空函数和传值调用的参数 …… 46

2.3.2　函数的例子 …… 48

2.3.3　传引用调用的参数 …… 48

2.4　递归 …… 51

2.4.1　阶乘函数 …… 52

2.4.2　递归的思考方式 …… 55

2.4.3　递归加法 …… 55

2.4.4　二项式系数函数 …… 57

2.4.5　逆转数组元素顺序 …… 61

2.4.6　汉诺塔 …… 61

2.4.7　相互递归 …… 63

2.4.8　递归的成本 …… 64

2.5　动态内存分配 …… 65

2.5.1　指针 …… 65

2.5.2　结构 …… 67

2.5.3　链式数据结构 …… 68

本章小结 …… 69

练习 …… 70

编程题 …… 71

第三部分　指令集架构层（第3层）

第3章　信息的表示 …… 76

3.1　无符号二进制表示 …… 76

3.1.1 二进制存储 …… 76
3.1.2 整数 …… 77
3.1.3 基数转换 …… 78
3.1.4 无符号整数的范围 …… 80
3.1.5 无符号加法 …… 80
3.1.6 进位位 …… 81
3.2 二进制补码表示 …… 81
3.2.1 补码的表数范围 …… 83
3.2.2 基数转换 …… 84
3.2.3 数轴 …… 85
3.2.4 溢出位 …… 86
3.2.5 负数和零位 …… 87
3.3 二进制运算 …… 88
3.3.1 逻辑运算符 …… 88
3.3.2 寄存器传送语言 …… 89
3.3.3 算术运算符 …… 90
3.3.4 循环移位运算符 …… 91
3.4 十六进制与字符表示 …… 92
3.4.1 十六进制 …… 92
3.4.2 基数转换 …… 92
3.4.3 ASCII 字符 …… 94
3.4.4 Unicode 字符 …… 97
3.5 浮点数表示 …… 100
3.5.1 二进制小数 …… 100
3.5.2 余码表示 …… 102
3.5.3 隐藏位 …… 103
3.5.4 特殊值 …… 104
3.5.5 IEEE 754 浮点数标准 …… 108
3.6 模型 …… 109
本章小结 …… 111
练习 …… 111
编程题 …… 117

第 4 章 计算机体系结构 …… 120

4.1 硬件 …… 120
4.1.1 中央处理单元 …… 120
4.1.2 主存储器 …… 121
4.1.3 输入 / 输出设备 …… 122
4.1.4 数据和控制 …… 123
4.1.5 指令格式 …… 123
4.2 直接寻址 …… 126
4.2.1 停止指令 …… 126
4.2.2 字装入指令 …… 126
4.2.3 字存储指令 …… 127
4.2.4 加法指令 …… 128
4.2.5 减法指令 …… 128
4.2.6 与和或指令 …… 129
4.2.7 按位取反和取负指令 …… 130
4.2.8 字节装入和字节存储指令 …… 131
4.2.9 输入和输出设备 …… 132
4.2.10 大端顺序和小端顺序 …… 133
4.3 冯·诺依曼机器 …… 134
4.3.1 冯·诺依曼执行周期 …… 134
4.3.2 一个字符输出程序 …… 135
4.3.3 冯·诺依曼漏洞 …… 138
4.3.4 一个字符输入程序 …… 139
4.3.5 十进制转换为 ASCII …… 139
4.3.6 一个自我修改程序 …… 140
4.4 ISA3 层的编程 …… 142
4.4.1 只读存储器 …… 143
4.4.2 Pep/9 操作系统 …… 144
4.4.3 使用 Pep/9 系统 …… 145
本章小结 …… 146
练习 …… 146
编程题 …… 148

第四部分 汇编层（第 5 层）

第 5 章 汇编语言 …… 150

5.1 汇编程序 …… 150
5.1.1 指令助记符 …… 150
5.1.2 伪操作 …… 152
5.1.3 .ASCII 和 .END 伪操作 …… 153
5.1.4 汇编器 …… 154
5.1.5 .BLOCK 伪操作 …… 155
5.1.6 .WORD 和 .BYTE 伪操作 …… 155
5.1.7 使用 Pep/9 汇编器 …… 156
5.1.8 交叉汇编器 …… 157
5.2 立即数寻址和陷阱指令 …… 158
5.2.1 立即数寻址 …… 158

5.2.2 DECI、DECO 和 BR 指令 …… 159
5.2.3 STRO 指令 …… 161
5.2.4 解释位模式：HEXO 指令 …… 162
5.2.5 反汇编器 …… 163
5.3 符号 …… 165
5.3.1 带符号的程序 …… 165
5.3.2 一个冯·诺依曼示例 …… 166
5.4 从 HOL6 层翻译 …… 168
5.4.1 Printf() 函数 …… 169
5.4.2 变量和类型 …… 170
5.4.3 全局变量和赋值语句 …… 171
5.4.4 类型兼容 …… 174
5.4.5 Pep/9 符号跟踪器 …… 175
5.4.6 算术移位和循环移位指令 …… 175
5.4.7 常量和 .EQUATE …… 176
5.4.8 指令与数据的放置 …… 178
本章小结 …… 179
练习 …… 180
编程题 …… 182

第 6 章 编译到汇编层 …… 185

6.1 栈寻址和局部变量 …… 185
6.1.1 栈相对寻址 …… 185
6.1.2 访问运行时栈 …… 186
6.1.3 局部变量 …… 188
6.2 分支指令和控制流 …… 190
6.2.1 翻译 if 语句 …… 191
6.2.2 优化编译器 …… 192
6.2.3 翻译 if/else 语句 …… 192
6.2.4 翻译 while 循环 …… 194
6.2.5 翻译 do 循环 …… 195
6.2.6 翻译 for 循环 …… 197
6.2.7 面条代码 …… 198
6.2.8 早期语言的控制流 …… 199
6.2.9 结构化编程定律 …… 200
6.2.10 goto 争论 …… 200
6.3 函数调用和参数 …… 201
6.3.1 翻译函数调用 …… 201
6.3.2 用全局变量翻译传值调用参数 …… 204
6.3.3 用局部变量翻译传值调用参数 …… 207
6.3.4 翻译非空函数调用 …… 209
6.3.5 用全局变量翻译传引用调用参数 …… 211
6.3.6 用局部变量翻译传引用调用参数 …… 215
6.3.7 翻译布尔类型 …… 218
6.4 变址寻址和数组 …… 220
6.4.1 翻译全局数组 …… 221
6.4.2 翻译局部数组 …… 224
6.4.3 翻译作为参数传递的数组 …… 226
6.4.4 翻译 switch 语句 …… 230
6.5 动态内存分配 …… 235
6.5.1 翻译全局指针 …… 235
6.5.2 翻译局部指针 …… 240
6.5.3 翻译结构 …… 243
6.5.4 翻译链式数据结构 …… 246
本章小结 …… 250
练习 …… 251
编程题 …… 251

第 7 章 语言翻译原理 …… 259

7.1 语言、语法和语法分析 …… 259
7.1.1 连接 …… 260
7.1.2 语言 …… 260
7.1.3 语法 …… 261
7.1.4 C 标识符的语法 …… 262
7.1.5 有符号整数的语法 …… 263
7.1.6 上下文相关的语法 …… 264
7.1.7 语法分析问题 …… 264
7.1.8 表达式的语法 …… 265
7.1.9 C 语法的一部分 …… 266
7.1.10 C 的上下文相关性 …… 269
7.2 有限状态机 …… 270
7.2.1 用有限状态机分析标识符 …… 270
7.2.2 简化的有限状态机 …… 271
7.2.3 非确定性有限状态机 …… 271
7.2.4 具有空转换的状态机 …… 272
7.2.5 多语言符号识别器 …… 274