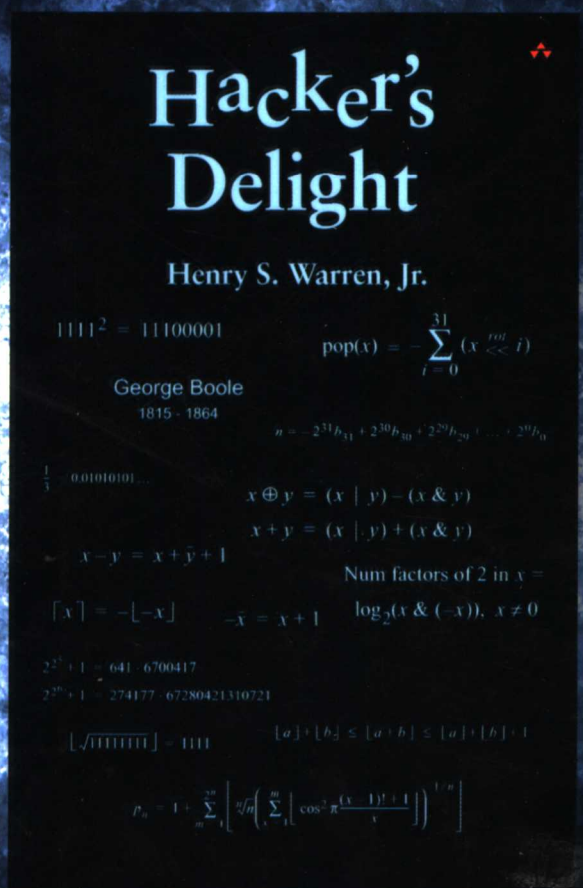
PEARSON Addison Wesley

计 算 机 科 学 丛 书

高效程序的奥秘

(美) Henry S. Warren, Jr. 著 冯 速 译



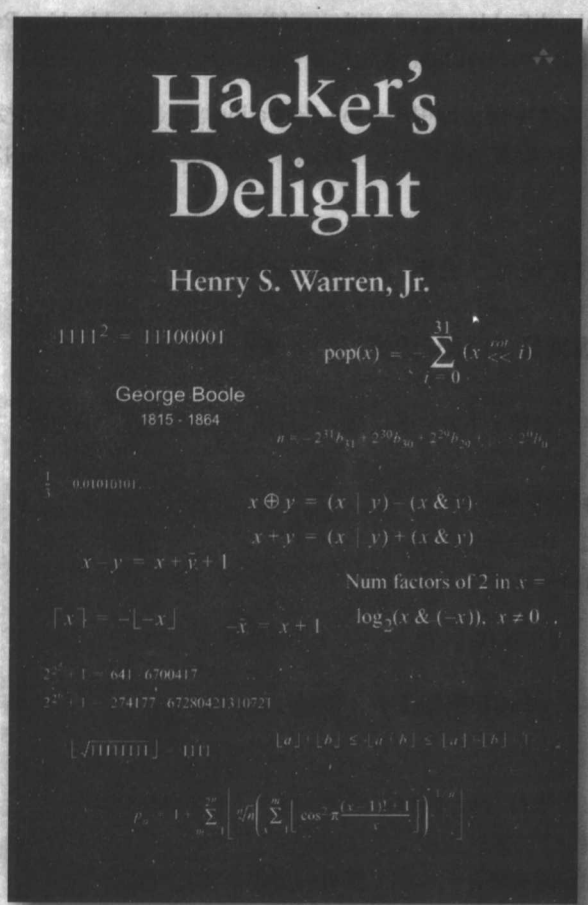
Hacker's Delight

机械工业出版社
China Machine Press

计 算 机 科 学 丛 书

高效程序的奥秘

(美) Henry S. Warren, Jr. 著 冯 速 译



Hacker's Delight

机械工业出版社
China Machine Press

本书适合程序库、编译器开发者及追求优美程序设计的人员阅读，适合用作计算机专业高年级学生及研究生的参考用书。

本书直观明了地讲述了计算机算术的更深层次的、更隐秘的技术，汇集了各种编程小技巧，包括常见任务的小算法、2的幂边界和边界检测、位和字节的重排列、整数除法和常量除法、针对整数的基本函数、Gray码、Hilbert空间填充曲线、素数公式等。

本书版权登记号：图字：01-2003-0998

图书在版编目（CIP）数据

高效程序的奥秘 /（美）沃瑞恩（Warren, H. S.）著；冯速译. -北京：机械工业出版社，2004.5

（计算机科学丛书）

书名原文：Hacker's Delight

ISBN 7-111-14111-3

Ⅰ.高… Ⅱ.①沃…②冯… Ⅲ.软件开发 Ⅳ.TP311.52

中国版本图书馆CIP数据核字（2004）第018070号

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：刘立卿

北京中加印刷有限公司印刷·新华书店北京发行所发行

2004年5月第1版第1次印刷

787mm × 1092mm 1/16 · 16 印张

印数：0 001 - 4 000 册

定价：28.00 元

凡购本书，如有倒页、脱页、缺页，由本社发行部调换

本社购书热线：（010）68326294

出版者的话

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，还揭橥了学术的源变，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、从业人员较少的现状下，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图文信息有限公司较早意识到“出版要为教育服务”。自1998年开始，华章公司就将工作重点放在了遴选、移译国外优秀教材上。经过几年的不懈努力，我们与Prentice Hall，Addison-Wesley，McGraw-Hill，Morgan Kaufmann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Tanenbaum，Stroustrup，Kernighan，Jim Gray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、研究及庋藏。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，有的还专诚为其书的中译本作序。迄今，“计算机科学丛书”已经出版了近百个品种，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，为进一步推广与发展打下了坚实的基础。

随着学科建设的初步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步入了一个新的阶段。为此，华章公司将加大引进教材的力度，在“华章教育”的总规划之下出版三个系列的计算机教材：除“计算机科学丛书”之外，对影印版的教材，则单独开辟出“经典原版书库”；同时，引进全美通行的教学辅导书“Schaum's Outlines”系列组成“全美经典学习指导系列”。为了保证这三套丛书的权威性，同时也为了更好地为学校和老师们服务，华章公司聘请了中国科学院、北京大学、清华大学、国防科技大学、复旦大学、上海交通大学、南京大学、浙江大学、中国科技大学、哈尔滨工业大学、西安交通大学、中国人民大学、北京航空航天大学、北京邮电大学、中山大学、解放军理工大学、郑州大学、湖北工学院、中国国家信息安全测评认证中心等国内重点大学和科研机构在计算机的各个领域的著名学者组成“专家指导委员会”，为我们提供选题意见和出版监督。

这三套丛书是响应教育部提出的使用外版教材的号召，为国内高校的计算机及相关专业

的教学度身订造的。其中许多教材均已为M. I. T.，Stanford，U.C. Berkeley，C. M. U. 等世界名牌大学所采用。不仅涵盖了程序设计、数据结构、操作系统、计算机体系结构、数据库、编译原理、软件工程、图形学、通信与网络、离散数学等国内大学计算机专业普遍开设的核心课程，而且各具特色——有的出自语言设计者之手、有的历经二十年而不衰、有的已被全世界的几百所高校采用。在这些圆熟通博的名师大作的指引之下，读者必将在计算机科学的宫殿中由登堂而入室。

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，但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。教材的出版只是我们的后续服务的起点。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：

电子邮件：hzedu@hzbook.com
联系电话：（010）68995264
联系地址：北京市西城区百万庄南街1号
邮政编码：100037

专家指导委员会

（按姓氏笔画顺序）

尤晋元	王　珊	冯博琴	史忠植	史美林
石教英	吕　建	孙玉芳	吴世忠	吴时霖
张立昂	李伟琴	李师贤	李建中	杨冬青
邵维忠	陆丽娜	陆鑫达	陈向群	周伯生
周立柱	周克定	周傲英	孟小峰	岳丽华
范　明	郑国梁	施伯乐	钟玉琢	唐世渭
袁崇义	高传善	梅　宏	程　旭	程时端
谢希仁	裘宗燕	戴　葵		

秘　书　组

武卫东　　温莉芳　　刘　江　　杨海玲

译 者 序

本书也许会激起某些读者的怀旧情感，特别是那些经历过或曾向往在早期的Z80等计算机上用汇编语言或Basic语言编程的人们；那些竭尽全力、苦思冥想、仅仅是为了能够写出更简短、更高效程序的人们；那些以编写一行高效Basic程序而自我陶醉的人们。

IBM资深程序员、蓝基因千万亿浮点计算机项目参与者Henry S. Warren, Jr.给所有程序员带来一本他们到处寻觅、实实在在、具有现实意义的程序设计技巧指南。本书是一本有特色的算法参考书，它不像现今多数算法书那样讨论大型的程序设计技术和系统的程序设计方法，而是向读者展示了计算机代码与指令，以及指令间的令人惊叹的内在关联，并通过这样的内在关联向读者揭示计算机运行的某些奥秘，揭示通过这样的关联更有效地实现基本操作的精湛技巧，以及这些技巧在优化编译器、图形学、密码学乃至数学中的应用。

本书适用于那些想要编写及欣赏巧妙、高效代码的读者，特别适合于希望把计算机软件和硬件有机地结合在一起的读者。本书值得每一位认真的计算机硬件设计者，每一位希望编写高效、优雅的嵌入式程序、编译器及优化编译器的设计者，以及每一位希望提高程序设计技艺的读者仔细斟酌和品味。

为了方便我国读者，我们尽量以通俗的语言翻译原文，并对难以理解的部分适当做了补充说明。翻译过程中译者得到机械工业出版社华章分社的大力支持。由于本书覆盖的内容较广、翻译时间仓促、水平有限，难免有疏漏与错误之处，请读者不吝指教。

冯 速

2003年12月

于北京师范大学

序

大约30年前，当我第一次在麻省理工学院的MAC项目中得到一份暑期工作时，我为能够使用DEC PDP-10计算机而兴奋不已。毫无疑问，在这类计算机上使用汇编语言编程比在其他计算机上更有趣，因为它具有完成位测试、位屏蔽、字段操作和整数操作所需的丰富易用的指令集。尽管PDP-10计算机已停产多年，但它仍拥有不少狂热的推崇者，他们仍然运行着PDP-10的硬件和软件，他们使用个人计算机模拟它的指令集，运行它的操作系统和相关的应用程序。他们甚至编写新软件；现在，至少有一家网站是用PDP-10的模拟器来维护网页的（不要笑，这并不比使陈旧的汽车跑起来更愚蠢）。

就是在这时，在1972年的夏天，我有幸看到了麻省理工学院被称为HAKMEM[⊖]的最新研究备忘录，这是一本奇妙而不拘一格的技术小文集。这些课题的内容涉及到电路、数论等广泛领域。而其中最令我感兴趣的是那些具有独创性的编程小技巧一览表。这些小技巧通常描述一些漂亮而与众不同的对整数和位串的操作（例如计算一个字中值为1的位的数目），这些操作原本是使用长长的机器指令序列或循环来实现的，而这些小技巧展示了如何仅用两三个精心筛选的指令就能更聪明地完成同样工作；同时探索、揭示出这些指令间的相互关系。我犹如吃玉米花一样，而不是像吃糖果那样，如饥似渴地阅读这些编程小珍宝——这些小技巧充满着华美，充满着智慧，充满着诗一般的优雅。

我想，“一定是这样的，还有许多这样的小技巧。”多年来我的确收集了许多，也发现了一些。“应该把它们写成一本书。”

当我看到Hank Warren的手稿时真是欣喜若狂。他系统地收集了许多编程小技巧，对它们分门别类，并做了详尽的解释。虽然有一些小技巧是用机器指令的形式描述的，但这本书不仅仅适用于汇编语言程序员。本书论述计算机整数和位串的基本结构关系，给出对它们进行有效操作的高效技术。这些技术对C语言和Java语言同样有用。

许多关于算法和数据结构的书籍对于排序和搜索、维护散列表和二叉树、记录和指针的处理讲了许多非常复杂的技巧，但它们忽视了微小的数据块——位和位数组——所能完成的工作。仅仅使用二进制加法、减法和按位操作就可以完成的工作令人惊讶；进位链使一个位可以对它左边的所有位产生影响，这一事实使加法操作成为一个非常强大的数据处理操作，而对此我们没有给予足够的认识。

是的，应该有一本关于这些技巧的书。现在，它就在你的手中，它非常神奇。如果你要编写最优化编译器或者高性能的代码，就必须阅读这本书。也许不是每天都能用上这些技巧，但是如果发现自己某一天陷入了困境，似乎需要按一个字中的位做循环操作，或者需要对整

⊖ HAKMEM是“hacks memo”的缩写。一个PDP-10的36位字中可以存放6个6位字符，因此PDP-10高手们所用的大部分名字被限制在6个字符之内。我们已经习惯于从6个字符的名字缩写立即译解出原名。所以在当时，至少对编程高手来说，以“HAKMEM”命名该备忘录是有意义的。

数进行操作而觉得难以编码、或者真的需要让整数计算或位计算的内循环速度成倍提高，这时就需要查看这些小技巧。当然也可能纯粹是为了欣赏而读这本书。

Guy L. Steele, Jr.
马萨诸塞州，柏灵顿
2002年4月

前　言

用户注意：软件维护成本的增加与程序设计人员的创造力的平方成正比。
“程序员创造力第一法则”，Robert D. Bliss, 1992

本书是我多年来所收集的程序设计小技巧的集成，它们大部分只能应用于以2的补码的形式表示整数的计算机上。尽管当这些技巧与寄存器的长度相关时，我们假设机器是32位的，但是我们很容易将这些技巧运用到其他寄存器长度的计算机上。

本书不涉及大型的编程技巧，例如高级排序技术和编译器优化技术等；相反，它所讨论的是诸如计算一个字中值为1的位的数目这样的与计算机的字或指令相关的小技巧，这样的技巧通常是算术指令和逻辑指令的混合物。

本书自始至终都假设整数溢出中断被屏蔽，所以不会发生溢出中断。C语言、Fortran语言、Java语言运行在这样的环境下，但是Pascal语言和ADA语言的用户则要小心。

本书的描述是非形式化的。只有当算法不是显然的时才给出证明（有些也不证明）。我们使用计算机算术、“地板”函数、算术和逻辑操作的混合等来描述这些小技巧。在这些领域，证明通常相当困难并且难以表述。

为了减少印刷错误和疏忽，我们实际运行了很多算法。因此，尽管每一种计算机语言都有不尽人意的一面，但我们还是使用程序设计语言给出了这些算法。由于C语言的高知名度，我们把它作为高级语言使用，它既允许整数操作，也允许位串操作，并且我们有产生高质量目标代码的C编译器。

偶尔我们也使用机器语言。它使用三地址格式，主要是为了可读性。我们使用的汇编语言是RISC虚拟计算机的汇编语言。

我们尽量使用无分支代码。因为在许多机器上，分支会减慢指令提取，抑制指令的并行执行。分支的另一个问题是它们可能会抑制编译器的优化，例如指令调度、指令公用以及寄存器分配。也就是说，编译器可能会更有效地优化由若干大的基本块组成的程序，而对于由许多小基本块组成的程序进行优化的效果则可能不显著。

代码序列中也常出现小的立即值、与零的比较（而不是与其他数相比较）以及指令级的并行。尽管通过（从内存）查表可以使大部分代码更简洁，但本书不常提及查表法。这是因为相对于算术指令，其装入的代价越来越大，而且查表法通常不是很有趣（尽管它们通常很实用）。当然也有例外的情况。

最后，我要说明的是，书名中的“HACKER”一词指的是计算机狂热爱好者——是一些喜欢使用计算机做新鲜事或用更新更聪明的方法做一些旧事情的人。这些计算机狂热爱好者们通常技艺高超，但很可能并非专业的计算机程序员或程序设计者。他们的工作也许很有用，也许仅仅是一种游戏。不少坚定的计算机迷们写出这样的程序，在执行这个程序时它会生成

自己的一份拷贝[⊖]。这就是我们通常所说的电脑黑客。如果你是在找如何侵入他人电脑的技巧的话，在本书中是找不到的。

H. S. Warren, Jr.
约克镇，纽约
2002年2月

⊖ 据作者所知，用C语言写成的最短的这种程序是Vlad Taeerov 和Rashit Fakhreyev所写的包含64个字符的程序：

```
main(a){printf(a,34,a="main(a){printf(a,34,a=%c%s%c,34);}",34);}
```

目　录

出版者的话
专家指导委员会
译者序
序
前言
第1章　介绍 ……1
1.1　记法 ……1
1.2　指令集和运行时间模型 ……3
第2章　基础 ……9
2.1　操作最右侧位 ……9
2.2　结合逻辑操作的加运算 ……12
2.3　逻辑和算术表达式中的不等式 ……13
2.4　绝对值函数 ……14
2.5　符号扩展 ……14
2.6　用无符号右移位实现带符号右移位 ……14
2.7　符号函数 ……15
2.8　三值比较函数 ……15
2.9　符号传递 ……16
2.10　对“0意味着2^n”字段的解码 ……16
2.11　比较谓词 ……16
2.12　溢出检测 ……20
2.13　加、减、乘的特征码结果 ……25
2.14　循环移位 ……26
2.15　双字长加、减法 ……26
2.16　双字长移位 ……27
2.17　多字节加、减、绝对值 ……28
2.18　doz、max、min函数 ……29
2.19　交换寄存器 ……30
2.20　两个或更多值之间的交换 ……32
第3章　2的幂边界 ……35
3.1　上舍入、下舍入到已知的2的幂的倍数 ……35
3.2　上舍入、下舍入到下一个2的幂 ……35
3.3　检测2的幂的边界跨越 ……38
第4章　算术边界 ……41
4.1　整数的边界检测 ……41
4.2　通过加和减传播边界 ……43
4.3　逻辑操作的边界传播 ……46
第5章　位计数 ……51
5.1　1位计数 ……51
5.2　奇偶性 ……58
5.3　前导0计数 ……60
5.4　后缀0计数 ……65
第6章　字搜索 ……71
6.1　寻找第一个0字节 ……71
6.2　寻找第一个给定长度的1位串 ……75
第7章　位和字节的重排列 ……79
7.1　位和字节的反转 ……79
7.2　混洗位 ……82
7.3　转置位矩阵 ……84
7.4　压缩或广义提取 ……90
7.5　一般置换，分羊操作 ……95
7.6　重排列和索引变换 ……98
第8章　乘法 ……99
8.1　多字乘法 ……99
8.2　64位积的高阶位部分 ……101
8.3　无符号积高阶位与带符号积高阶位间的转换 ……101
8.4　常量乘法 ……102
第9章　整数除法 ……105
9.1　预备知识 ……105
9.2　多字除法 ……107
9.3　从带符号除法到无符号短除法 ……111
9.4　无符号长除法 ……113
第10章　整数常量除法 ……119
10.1　除以一个2的已知幂的带符号除法 ……119
10.2　除以一个2的已知幂的除法的带符号余数 ……120

10.3 非2的幂的带符号除法和余数 ……121
10.4 除数＞2的带符号除法 ……123
10.5 除数＜－2的带符号除法 ……129
10.6 并入编译器 ……131
10.7 其他主题 ……133
10.8 无符号除法 ……136
10.9 除数＞1的无符号除法 ……138
10.10 并入编译器（无符号） ……140
10.11 其他论题（无符号） ……140
10.12 模除法和地板除法的适用性问题 ……143
10.13 类似的方法 ……144
10.14 魔术数示例 ……145
10.15 除以常数的精确除法 ……146
10.16 除以常数的除法的零余数检测 ……151
第11章 初等函数 ……155
11.1 整数平方根 ……155
11.2 整数的立方根 ……161
11.3 整数求幂 ……162
11.4 整数对数 ……164
第12章 数制中的特殊底 ……171
12.1 以－2为底 ……171
12.2 以－1+i为底 ……176
12.3 其他底 ……178
12.4 最有效的底是什么 ……178
第13章 Gray码 ……181
13.1 Gray码 ……181
13.2 递增Gray码整数 ……183
13.3 负二进制Gray码 ……184
13.4 简史及应用 ……184
第14章 Hilbert曲线 ……187
14.1 生成Hilbert曲线的递归算法 ……187
14.2 从Hilbert曲线的路长求坐标 ……191
14.3 Hilbert曲线上坐标到路长的转换 ……196
14.4 递增Hilbert曲线上点的坐标 ……198
14.5 非递归生成算法 ……200
14.6 其他空间填充曲线 ……200
14.7 应用 ……201
第15章 浮点 ……203
15.1 IEEE格式 ……203
15.2 利用整数操作进行浮点数比较 ……205
15.3 前导数字分布 ……206
15.4 各种各样的值的列表 ……207
第16章 素数公式 ……211
16.1 介绍 ……211
16.2 Willans公式 ……212
16.3 Wormell公式 ……215
16.4 求其他比较麻烦的函数的公式 ……216
附录A 四位计算机的算术表 ……221
附录B 牛顿方法 ……225
参考文献 ……227
索引 ……233

第1章 介 绍

1.1 记法

本书将普通算术的数学表达式与描述计算机操作的数学表达式区分开来。在“计算机算术”中，操作数是一些定长位串或定长位矢量。计算机算术中的表达式与普通算术表达式相似，但是，变量表示计算机寄存器的内容。计算机算术表达式的值仅仅是一个位串，没有特定含义。操作符用特定的方式来解释其操作数的含义。例如，比较操作符可以把它的操作数解释为带符号二进制整数，也可以解释成无符号二进制整数；这里的计算机算术记法使用不同的符号来明确比较的类型。

计算机算术与普通算术之间的一个主要区别是，在计算机算术中，我们使用模2^n来约简加法、减法和乘法的结果，其中，n是计算机的字长。另一个区别是计算机算术包括大量的操作，除了四个基本算术操作之外，计算机算术还包括逻辑与（and）、异或（exclusive or）、比较（compare）、左移位（shift left）等等。

除非特别声明，一般字长为32位，带符号整数以2的补码的形式表示。

计算机算术表达式与普通算术表达式的形式类似，只是表示计算机寄存器内容的变量用黑体字表记。这是矢量代数的习惯做法。我们把计算机的字看成由位组成的矢量。当常量代表计算机寄存器的内容时也用黑体字表记。（矢量代数中没有相应的表记方法，矢量代数表记常量的惟一方法是给出它的成分）。当一个常量代表一个指令的组成部分时，例如一个移位指令的立即字段，我们使用细体字来表记。

若一个如“+”这样的操作符带有黑体表记的操作数，那么该操作符就表示计算机的加法操作（“矢量加法”）。如果操作数是细体字，那么操作符就表示普通的标量算术运算。我们用细体的变量x去表示黑体变量$\boldsymbol{x}$在特定解释下的算术值，根据前后文决定把$\boldsymbol{x}$解释成带符号数还是无符号数。因此，如果$\boldsymbol{x}$=**0x80000000**，$\boldsymbol{y}$=**0x80000000**，那么在带符号整数的意义下$x=y=-2^{31}$，$x+y=-2^{32}$，而$\boldsymbol{x}+\boldsymbol{y}$=**0**。在这里，**0x80000000**是一个1位后面跟着31个0位组成的位串的十六进制表示。

对位的计数是从右侧开始的，最右侧（最小有效）位称作第0位。术语“位”、“半字节”、“字节”、“半字”、“字”和“双字”，分别表示长度1、4、8、16、32和64位。

我们使用赋值操作符（左箭头）及if语句这样的计算机代数来书写短小的代码片段。这里，计算机代数只不过是充当了与机器无关的汇编码的角色。

更长的或者更复杂的程序则用C++来书写。我们不使用C++的面向对象功能；程序本质上是带有C++注释风格的C程序。当区别不重要时，我们简单地用“C”来称呼这一语言。

对于C的完整说明超出了本书的范围，但我们在表1-1中列出了本书所用的C的大部分要素。这一表格主要是为那些对C语言以外的其他过程型编程语言有一定了解的读者准备的。表1-1还给出了计算机代数算术语言的操作符。操作符按着优先度从高到低排列。在优先度一栏中，

⊖ 此边栏的号码为该书原书页码，与书末索引中的页码相呼应。

L代表左结合，即，

$$a \cdot b \cdot c=(a \cdot b) \cdot c$$

而R代表右结合。这里，计算机代数中记号的优先度和结合性沿用C语言的约定。

除去表1-1所列出的记号外，我们也使用布尔代数和标准数学记号，并在必要的时候加以说明。

表1-1 C语言和计算机代数的表达式

优先度	C	计算机代数	说 明
	0x...	**0x**...,**0b**...	十六进制、二进制常数
16	a[k]		选择第k个分量
16		$x_0, x_1, \cdots$	不同的变量或位选择（根据文中说明）
16	f(x,...)	$f(x, \cdots)$	函数求值
16		abs(x)	绝对值（但是，abs(-2^{31})=-2^{31}）
16		nabs(x)	绝对值的负数
15	x++, x--		后置递增、后置递减
14	++x, --x		前置递增、前置递减
14	(类型名)x		类型转换
14R		x^k	x的k次幂
14	~x	$\neg x, \bar{x}$	按位否（1的补码）
14	!x		逻辑否（若x=**0**则为**1**、否则为**0**）
14	-x	$-x$	算术取负
13L	x*y	$x*y$	乘法、模字长
13L	x/y	$x \div y$	带符号整数除法
13L	x/y	$x \overset{u}{\div} y$	无符号整数除法
13L	x%y	rem(x,y)	$x \div y$的余数、可以为负、带符号参数
13L	x%y	remu(x,y)	$x \overset{u}{\div} y$的余数、无符号参数
		mod(x,y)	x模y、约简到区间[**0**,abs(y)−**1**]；带符号参数
12L	x+y, x-y	$x+y, x-y$	加法，减法
11L	x<<y, x>>y	$x<<y, x \overset{u}{>>} y$	带0填充的左移位和右移位（“逻辑”移位）
11L	x>>y	$x \overset{s}{>>} y$	带符号填充的右移位（“算术”或“代数”移位）
11L		$x \overset{rot}{<<} y, x \overset{rot}{>>} y$	循环左移位和右移位
10L	x<y,x<=y,x>y,x>=y	$x<y, x \leqslant y, x>y, x \geqslant y$	带符号比较
10L	x<y,x<=y,x>y,x>=y	$x \overset{u}{<} y, x \overset{u}{\leqslant} y, x \overset{u}{>} y, x \overset{u}{\geqslant} y$	无符号比较
9L	x == y, x!=y	$x=y, x \neq y$	相等、不等
8L	x & y	$x \& y$	按位与
7L	x^y	$x \oplus y$	按位异或
7L		$x \equiv y$	按位等值($\neg(x \oplus y)$)
6L	x | y	$x \mid y$	按位或
5L	x && y	$x \vec{\&} y$	条件与（若x=**0**则为**0**；否则，若y=**0**则为**0**，否则为**1**）
4L	x || y	$x \vec{\vee} y$	条件或（若$x \neq$ **0**则为**1**；否则，若$y \neq$ **0**则为**1**，否则为**0**）
3L		$x \| y$	连接
2R	x = y	$x \leftarrow y$	赋值

除“abs”、“rem”等函数外，计算机代数还使用其他函数。我们将在介绍这些函数时加以定义。

在C语言中，表达式x<y<z的意思是，先求x<y的值，然后将其0/1值结果与z比较。而在计算机代数中，表达式$x<y<z$意思是$(x<y)\&(y<z)$。

C语言有三种循环控制语句：while语句、do语句和for语句。while语句的书写格

式是:

```
while (expression) statement
```

对于这样的while语句，首先，求*expression*的值；如果*expression*的值为**true**(即非0)，执行*statement*，执行完毕后控制再次返回到对*expression*求值的操作。如果*expression*的值为**false**(即0)，则结束循环。

do语句的运行类似，只是在每次循环后进行条件检测。它的书写格式是:

```
do statement while (expression)
```

首先执行*statement*，然后对*expression*求值。如果值为**true**则重复这一过程，如果值为**false**则结束循环。

for语句的书写格式是:

```
for (e1; e2; e3) statement
```

首先，运行e_1，它通常是赋值语句，然后对e_2求值，它通常是一个比较。如果e_2的值为**false**，则结束循环。如果e_2的值为**true**，则执行*statement*。最后，执行e_3，它通常是赋值语句，然后控制转移到对e_2再次求值。所以，常见的“do i=1 to n”书写成:

```
for (i=1; i<=n; i++)
```

(这是使用后置递增操作符的几种情况之一)。

1.2 指令集和运行时间模型

为了对算法做粗略的比较，我们使用与通用RISC计算机指令集类似的指令集对这些算法进行编码。这样的通用计算机包括Compaq Alpha、SGI MIPS以及IBM RS/6000等。我们使用三地址指令，并假定机器至少有16个通用寄存器。除非特别声明，寄存器的长度为32位。通用寄存器0的值总为0，而其他寄存器可以用于任意目的。

为简单起见，这里没有条件寄存器，也没有用于保存状态位（如“溢出”）等目的的“专用”寄存器。浮点操作也不在本书的讨论范畴。

有两种RISC:“基本RISC”拥有表1-2所列的指令；“完全RISC”除拥有“基本RISC”的所有指令外，还拥有表1-3所列的指令。

表1-2 基本RISC指令集

操作码助记符	操 作 数	说 明
add, sub, mul, div, divu, rem, remu	RT, RA, RB	RT←RA op RB，其中，op分别是加、减、乘、带符号除、无符号除、带符号求余或无符号求余
addi, muli	RT, RA, I	RT←RA op I，其中，op分别是加或乘，I是一个16位带符号立即值
addis	RT, RA, I	RT←RA + (I || 0x0000)
and, or, xor	RT, RA, RB	RT←RA op RB，其中，op分别是按位与、按位或以及按位异或
andi, ori, xori	RT, RA, Iu	除最后操作数为16位无符号立即值外，其余同上条一样
beq, bne, blt, ble, bgt, bge	RT, target	分别当RT=0、RT≠0、RT<0、RT≤0、RT>0、RT≥0时分支到target（把RT解释为带符号整数）

（续）

操作码助记符	操 作 数	说 明
bt, bf	RT, target	当RT为true或false时分支到target；分别与bne和beq相同
cmpeq, cmpne, cmplt, cmple, cmpgt, cmpge, cmpltu, cmpleu, cmpgtu, cmpgeu	RT, RA, RB	比较RA和RB：若结果为**false**则将0装入RT，若结果为**true**则将1装入RT。各助记符表示等于、不等于、小于等分支指令的比较。另外，后缀“u”表示相应比较是无符号比较
cmpieq, cmpine, cmpilt, cmpile, cmpigt, cmpige	RT, RA, I	除第二个比较操作数为16位带符号立即值外，其他同cmpeq等相应指令类似
cmpiequ, cmpineu, cmpiltu, cmpileu, cmpigtu, cmpigeu	RT, RA, Iu	除第二个比较操作数为16位无符号立即值外，其他同cmpltu等相应指令类似
ldbu, ldh, ldhu, ldw	RT, d(RA)	分别把从内存单元RA+d开始的一个无符号字节、带符号半字、无符号半字或一个字的内容装入到RT，其中，d为16位带符号立即值
mulhs, mulhu	RT, RA, RB	RT分别取RA和RB的带符号乘积和无符号乘积的高阶32位
not	RT, RA	RT←RA的按位1的补码
shl, shr, shrs	RT, RA, RB	RT←RA的左移位或右移位，移位量是RB的最右6位的值；shrs为符号填充，其余为0填充（位移量被视为模64）
shli, shri, shrsi	RT, RA, Iu	RT←RA的左移位或右移位，移位量由Iu的5位立即字段中给定的值决定
stb, sth, stw	RS, d(RA)	分别将RS的一个字节、半字或字存储于内存单元RA+d，其中，d是16位带符号立即值

5

在以上指令的简单说明中，作为源操作数的RA和RB指的是这些寄存器的内容。

实际的计算机拥有分支和调用子程序的链接，分支到一个寄存器中存放的地址（当子程序返回时及多分支执行完毕后），还可能拥有涉及专用寄存器的指令。当然，它会拥有若干特权指令，以及调用管理服务的指令，还可能拥有浮点指令。

表1-3中列出了RISC计算机可能拥有的其他计算指令。这些指令将在以后章节讨论。

表1-3 完全RISC的附加指令

操作码助记符	操 作 数	说 明
abs, nabs	RT, RA	RT分别取RA的绝对值和绝对值的负
andc, eqv, nand,nor, orc	RT, RA, RB	RT分别取RA和RB的补的按位与、RA和RB的按位等值、RA和RB的按位与非、RA和RB的按位或非，以及RA和RB的补的按位或
extr	RT, RA, I, L	从RA中提取第I位到第I+L-1位的位串，然后右对齐放入RT中，用0填充剩余位
extrs	RT, RA, I, L	与extr类似，不同的是用符号填充剩余位
ins	RT, RA, I, L	将RA的第0位到第L-1位的位串插入到RT的第I位到第I+L-1位
nlz	RT, RA	RT取RA中前导0的数目，取值范围为0到32
pop	RT, RA	RT取RA中值为1的位的数目，取值范围为0到32

6