高等学校工程创新型『十二五』规划计算机教材

大学计算机基础

（第2版）

王 永 曹慧英 杜茂康
张 仿 谢 青 编著

Engineering Innovation 工程创新

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

高等学校工程创新型“十二五”规划计算机教材

大学计算机基础

（第2版）

王永　曹慧英　杜茂康　张仿　谢青　编著

電子工業出版社

Publishing House of Electronics Industry

北京・BEIJING

内 容 简 介

本书根据教育部计算机基础课程教学指导分委员会最新制定的大学计算机基础大纲编写，包括计算机信息技术的基本概念和新技术介绍、数制系统、计算机软硬件基础、网络基础及互联网应用、网页制作基础、多媒体技术基础、Windows 操作系统、办公软件应用基础和信息安全基础等内容。

本书注重知识与技术的先进性和实用性，重视理论概念与操作应用的结合。全书结构清晰，内容翔实，通俗易懂，可作为高等院校计算机基础课程的教材，计算机初学者的入门书籍或计算机应用方面的培训教程。

图书在版编目（CIP）数据

大学计算机基础 / 王永等编著. —2 版. —北京：电子工业出版社，2011.8
高等学校工程创新型“十二五”规划计算机教材
ISBN 978-7-121-13617-7

I. ①大… II. ①王… III. ①电子计算机—高等学校—教材 IV. ①TP3

中国版本图书馆 CIP 数据核字（2011）第 094663 号

策划编辑：章海涛
责任编辑：章海涛　　特约编辑：何　雄
印　　刷：涿州市京南印刷厂
装　　订：涿州市桃园装订有限公司
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：21.5　字数：605 千字
印　　次：2011 年 8 月第 1 次印刷
印　　数：3 000 册　　定价：36.00 元

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

近年来，计算机信息技术和网络技术以更快的速度向前发展，应用领域不断拓广，新概念、新技术和新应用层出不穷，再加上中小学信息技术课程教学内容的扩展和深化，高校《大学计算机基础》的教学内容更应与时俱进，适时调整。针对这一情况，教育部高教司组织制定了《大学计算机教学基本要求》，对高校计算机基础教学的内容、任务、方法和目标提出了明确的要求和建议。本书在前一版的基础上，根据教育部最新版的《大学计算机教学基本要求》和计算机等级考试大纲编写。

本书从计算机基础理论和操作应用两方面进行内容题材的组织，注重知识的新颖性、实用性、理论性与实践性的结合，结合当前的最新成果和应用对计算机技术、网络技术、信息技术的基本概念和原理进行了介绍，如云计算、物联网、黑客防范、密码技术、网络安全等；同时，对互联网应用、办公应用、网页制作等内容进行了初步介绍。通过本书的学习，读者可以理解计算机和网络的基本工作原理，掌握计算机的基本操作，使用办公软件进行办公、上网查询资料、制作网页、创建博客、使用 QQ 等即时通信工具、收发电子邮件等。

本书注重计算机技术的基本原理、基本概念和新技术的讲解，并将它们分配到了第 1～5 章，这些章节中的基本理论对于学习后面各章节中的软件应用是很有帮助的。

全书共分 12 章，第 1～5 章、第 12 章和第 10 章的前半部分属于计算机理论基础的范畴，第 6～11 章属于计算机应用基础的范畴。各章节内容如下：

第 1 章计算机与信息技术，介绍信息技术和计算机的发展史、特点、应用领域、基本概念和常用术语。

第 2 章数制系统，介绍常用的数制系统及不同数制系统中的数据转换，计算机内部的数据表示与数据存储，数据编码，汉字信息及其在计算机中的编码系统。

第 3 章计算机硬件基础，介绍计算机的体系结构，计算机的组成部件及功能，主要包括中央处理器、存储器、总线、频率、接口及显示系统等内容。

第 4 章计算机软件与多媒体技术基础。介绍软件基础知识，常用软件，操作系统，语言处理程序，软件术语，多媒体技术等。

第 5 章计算机网络基础。介绍计算机网络的基础理论，基本术语，网络拓扑结构，网络类型，网络协议，网络设备和网络传输介质，常见的网络组建模式等。

第 6 章 Windows XP 操作系统。介绍 Windows 系统的特点、文件系统、程序管理、磁盘管理、用户管理、任务管理、软件安装及设备管理以及 Windows 系统中的常用应用程序。

第 7～11 章介绍 Microsoft Office 2007 办公软件的应用，包括文字处理软件 Word 2007 的文档排版，图形处理，表格处理，文档格式化、图书目录制作等功能；电子表格软件 Excel 2007 的数据输入，公式计算，常用工作表函数，图表处理，电子数据的排序、筛选、分类汇总等功能；PowerPoint 2007 演示文稿的母板设计，图表制作，动画设计，影像设计，色彩配置等功能；FrontPage 网页制作的基本技术，网站发布的基本方法、HTTP 及 HTML 等基本概念。

第 10 章 Internet 及其应用，介绍 Internet 的基础知识及基本概念，TCP/IP 协议，Internet 的接

入方式，Windows 系统联网的 TCP/IP 配置方法，Internet Explorer 的操作应用方法，电子信箱的申请及 E-mail 的收发，QQ 应用、博客编写及 Internet 中的资料查询与下载。

第 12 章信息安全基础，介绍计算机病毒、网络安全和信息安全的基本概念、技术、基本原理及防范措施，如木马防范、加密技术、数字认证、安全协议等。

本书的编著得益于多年教学经验的积累，为了使本书易学易懂，全书很注重图文的应用，颇费心思地利用图注解释一些基本原理、操作过程及计算机术语，使之生动形象，深入浅出。在内容编排方面也充分考虑了教与学的关系。第 1～6 章、第 12 章及第 10 章的前 4 节属于计算机基础理论的范围，是课堂教学的重点，应注重这些章节中的基本概念、名词术语及基本原理的讲解。第 7、8、9、11 章及第 10.5 节的内容具有很强的可操作性，属于基本技能培养的内容，也是学习者必须掌握的计算机应用技术，宜采用示范教学方式，以上机实践和学习者自学为主（第 6 章也必须多从上机实践中进行学习和理解），不宜占用太多的课堂教学时间。

本书由王永、曹慧英、杜茂康、张仿、谢青、吴伯柱编著。王永编写了第 1、2、3、4、5 章，曹慧英编写了第 6、10 章，杜茂康编写了第 7 章，张仿编写了第 8 章，谢青编写了第 9、12 章，吴伯柱编写了第 11 章。李昌兵、罗龙艳、武建军参与了本书的写作大纲编写，提供了教学课件、书中部分图表和习题等写作素材，全书由王永统稿和审定。

在本书的编写过程中，我们尽可能地结合了当前计算机软硬件的最新技术和发展趋势，参考了国内外许多与此相关的教材，论文，网络资源特别是维基百科、互动百科、百度百科等网站中的资料，以求教材知识的先进性和全面性。在此，特向这些资源的原创者表示崇高的敬意和深深的感谢！

由于作者水平有限，书中缺点和不足在所难免，敬请专家、内行和广大读者批评指正。

读者反馈：unicode@phei.com.cn，cqyddk@163.net。

编　者

第 2 章　数据编码与汉字输入法

巨型机、PC、笔记本或掌上电脑，无论其体积和功能的差异有多大，实质上都不过是大小不同、多少不同的一大堆电子开关而已。断开的开关表示 0，接通的开关表示 1。数目众多的开关不断地重新组合，不断地导通与断开，就构成了功能强大的计算机，而不同的开关组合就构成了不同的 0、1 数字串，即二进制数字。对二进制数字进行编码和转换，可以表示出任意进制的复杂数据。可以说，二进制数（由 0 和 1 两个符号组成的数）是计算机的“母语”，所有的计算机都认识它，而且只认识它。

本章介绍计算机中的数据表示，包括数制系统、各数制系统之间的数据转换、计算机内的数据编码、汉字的编码系统和汉字输入法等内容。

2.1　数制系统

人们最熟悉的是十进制数据，同时也在与其他数制系统打交道。例如时间计数，60 分钟为 1 小时，24 点以后，又从 0 点开始计数。古代曾有“半斤八两”之说，即 16 两记为 1 斤。除此之外，在日常生活中还有许多其他计数方法。例如，一个星期 7 天，星期日之后又是星期一；一年 12 个月，12 月之后又是 1 月；如此等等。

不同的计数方式形成了不同的数制系统，也称为进位计数制。上面提到的进位制有六十进制，二十四进制，十六进制，七进制，十二进制。不同数制系统的计数原理和进位计算规则是相通的，抽象成 R 进制，它们都具有以下特点：

① 基数（计数符号的个数）为 R，表示数的符号有 0, 1, 2, 3, 4, 5,…, $R-1$。

② 逢 R 进一。

③ 按权展开。一般而言，一个 R 进制数 $s=k_nk_{n-1}\cdots k_1k_0k_{-1}k_{-2}\cdots k_{-n}$ 所代表的实际值为

$$s=k_n\times R^n+k_{n-1}\times R^{n-1}+\cdots+k_1\times R^1+k_0\times R^0+k_{-1}\times R^{-1}+\cdots+k_{-n}\times R^{-n}$$

这里的 $k_n, k_{n-1},\cdots, k_1, k_0,\cdots, k_{-n}$ 等表示 0, 1, 2,…, $R-1$ 之中的任何一个数字。所谓权，也称为位权，是指数字在数据中所占的位置。上式中 R 的幂次即是权，如 R^n、R^{-1} 等，它与数字的大小密切相关。

如果 R 为 10，即为十进制数。其基数为 10，计数的符号为 0, 1, 2, 3, 4, 5, 6, 7, 8, 9；在进行数学运算时的规则为“逢十进一”，如 8+4=12，因为加法结果超过 10，所以向高位进 1。

如果 R 为 8，则为八进制数。其计数的符号为 0, 1, 2, 3, 4, 5, 6, 7；运算时“逢八进一”，如 1+7 = 10、2×4 = 10、16−7 = 7。

如果 R 为 16，则为十六进制数。其基数为 16，采用 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F 共 16 个符号来表示所有的十六进制数据，其中 A 表示十进制数中的 10，B 表示 11，C 表示 12，D 表示 13，E 表示 14，F 表示 15。在用十六进制计数时，每位满 16 之后就向高位进一，即“逢十六进一”。

任何数制系统中的数据都可以按权展开，该数制的基数的幂次代表权。例如，十进制数

5296.45，八进制数 1704.25，十六进制数 2EC.F，其“按权展开”式可表示如下：

$$(5296.45)_{10} = 5\times10^3+2\times10^2+9\times10^1+6\times10^0+4\times10^{-1}+5\times10^{-2} \quad (2\text{-}1)$$

$$(1704.25)_8 = 1\times8^3+7\times8^2+0\times8^1+4\times8^0+2\times8^{-1}+5\times8^{-2} \quad (2\text{-}2)$$

$$(2EC.F)_{16} = 2\times16^2+14\times16^1+12\times16^0+15\times16^{-1} \quad (2\text{-}3)$$

在数制系统的描述中常用$(\ )_n$来表示 n 进制，括号内的数是数值本身，括号外的下标表示进制。如$(5296.45)_{10}$是十进制数，$(2EC.F)_{16}$是十六进制数。

同一数据在不同的数制系统中，其表现形式是不同的，对应情况如表 2-1 所示。

表 2-1　数制对照表

十进制数	二进制数	八进制数	十六进制数
0	0	0	0
1	1	1	1
2	10	2	2
3	11	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F
16	10000	20	10

计算机用二进制来处理和存储所有的数据。为了便于人们阅读和分析，计算机往往将处理的结果表示为十进制数据或英文字符，然后在显示屏幕上显示出来。但在计算机内部，所有的信息都是以二进制数形式表示的。

1．二进制数据的概念

二进制数的基数为 2，用 0、1 两个符号表示所有的数据。同十进制数一样，处于一个二进制数中不同位置的 0 或 1 代表的实际值也是不一样的，要乘上一个以 2 为底数的指数值。例如，二进制数 110110 所表示的数的大小为：

$$(110110)_2 = 1\times2^5+1\times2^4+0\times2^3+1\times2^2+1\times2^1+0\times2^0=(54)_{10} \quad (2\text{-}4)$$

可见，二进制数 110110 和十进制数 54 的大小相同，只不过表示方法不同而已。

式（2-4）也展示了不同进制的数出现在同一表达式中的一种表示方法，$(54)_{10}$表示十进制数 54，而$(110110)_2$则表示二进制数 110110。

小数的表示和计算方法与此类似。如 101.1011 表示的值可用下式计算：

$$(101.1011)_2=1\times2^2+0\times2^1+1\times2^0+1\times2^{-1}+0\times2^{-2}+1\times2^{-3}+1\times2^{-4}$$

概括而言，一个二进制数 $s=k_nk_{n-1}\cdots k_1k_0k_{-1}k_{-2}\cdots k_{-n}$ 的按权展开式如下：

$$s=k_n\times2^n+k_{n-1}\times2^{n-1}+\cdots+k_1\times2^1+k_0\times2^0+k_{-1}\times2^{-1}+\cdots+k_{-n}\times2^{-n}$$

这里的 $k_n, k_{n-1},\cdots, k_1, k_0,\cdots, k_{-n}$ 表示 0 或 1。

如前所述，二进制数也有如下 3 个特点。

① 基数为 2，用 0、1 两个符号表示所有的二进制数。

② 加法规则：逢二进一。

$$0+0=1,\ 0+1=1,\ 1+0=1,\ 1+1=10$$

③ 按权展开：

$$(1101)_2=1\times2^3+1\times2^2+0\times2^1+1\times2^0$$

2．二进制数加法运算

与十进制数一样，二进制数据也具有加、减、乘、除及乘方等算术运算。计算机中最常用的是二进制数加法运算，加法规则是“逢二进一”。例如：

$$\begin{array}{r} 1011010111 \\ +\quad 111010101 \\ \hline 10010101100 \end{array}$$

3．二进制数据的逻辑运算

二进制数据的逻辑运算是一种按位运算，有与（AND）、或（OR）、非（NOT）三种运算。

AND 的意思是两个二进制数据按位进行“与”运算，“与”的含义是：1 AND 1=1，1 AND 0=0，0 AND 1=0，0 AND 0=0。

OR 的意思是两个二进制数据按位进行“或”运算，“或”的含义是：1 OR 1=1，1 OR 0=1，0 OR 1=1，0 OR 0=0。

NOT 也称为取反运算，意思是将一个二进制数据的 0 变为 1，1 变为 0。

AND、OR 和 NOT 运算的举例如下：

$$\begin{array}{r} 1011010111 \\ \text{AND}\ \ 1111010101 \\ \hline 1011010101 \end{array}\qquad \begin{array}{r} 1011010111 \\ \text{OR}\ \ 1111010101 \\ \hline 1111010111 \end{array}\qquad \begin{array}{r} \text{NOT}\ \ 1111010111 \\ \hline 0000101000 \end{array}$$

2.2 数制系统之间的转换

人们对十进制数很熟悉，而对其他进制的数比较陌生。为了便于应用和理解，往往需要在不同进制的数据之间进行转换。

2.2.1 十进制数转换成 N 进制数

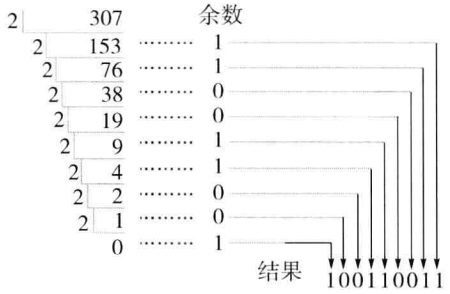
十进制数转换成 N 进制数的规则是：整数部分采用“除 N 取余法”，即将十进制数除以 N，把除得的商再除以 N，如此反复，直到商为 0，然后将每次相除所得的余数倒序排列，第一个余数为最低位，这样得到的数就是转换之后的 N 进制数。

小数部分采用所谓的“乘 N 取整法”：即将十进制数的小数部分乘以 N，得到一个整数部分

和一个小数部分；再用 N 乘以小数部分，又得到一个小数部分和一个整数部分，继续这个过程，直到余下的小数部分为 0 或满足精度要求为止；最后将每次得到的整数部分从左到右排列，即得到所对应的 N 进制小数。

1．十进制数转换为二进制数

转换规则：整数部分除 2 取余，辗转相除，直到商为零，倒序排列（余数）；小数部分乘 2 取整，然后把所得的小数部分再次乘以 2，取其乘积的整数部分，如此反复，直到最后小数部分为 0 或满足精度要求，将每次乘得的整数部分顺序排列。例如，将十进制数 307.625 转换为二进制数，整数部分的转换如下：



把所得的余数倒序排列为 100110011，它就是十进制数 307 所对应的二进制数。小数部分的转换如下：

	整数部分	位数
0.625×2=1.250	1	1
0.250×2=0.500	0	2
0.500×2=1.00	1	3
0.000	转换结束	

把转换后的整数部分顺序排列为 101，这就是 0.625 小数部分转换后的二进制数。这样，307.625 所对应的二进制数为 100110011.101。

再如，十进制数 157 经转换后的二进制数为 10011101，十进制数 0.8125 经转换后的二进制小数为 0.1101，读者可自己练习转换。

2．十进制数转换为十六进制数

转换规则为：整数部分除 16 取余，辗转相除，直到商为 0，倒序排列（余数）。小数部分乘 16 取整，然后把所得的小数部分再次乘以 16，取其乘积的整数部分，如此反复，直到最后小数部分为 0 或满足精度要求，将每次乘得的整数部分顺序排列。例如，将十进制数 307 转换成十六进制数的过程如下：

2.2.2 二、十六进制数转换成十进制数

把任意 N 进制数转换成十进制数非常简单，只需要把 N 进制数按权展开并计算出展开式的结果就可以了。

1. 二进制数转换为十进制数

把要转换的二进制数按权展开并求和即可。如把二进制数 11010.101 转换为十进制数：

$$\begin{aligned}(11010.101)_2 &= 1\times2^4+1\times2^3+0\times2^2+1\times2^1+0\times2^0+1\times2^{-1}+0\times2^{-2}+1\times2^{-3}\\ &= 16+8+2+0.5+0.125\\ &= (26.625)_{10}\end{aligned}$$

2. 十六进制数转换为十进制数

把要转换的十六进制数按权展开并求和即可。如把十六进制数 2EC.F 转换为十进制数：

$$\begin{aligned}(2EC.F)_{16} &= 2\times16^2+14\times16^1+12\times16^0+15\times16^{-1}\\ &= 512+224+12+0.9375\\ &= (748.9375)_{10}\end{aligned}$$

2.2.3 二、十六进制数之间的转换

二进制数、八进制数、十六进制数实质上都是同一类数，它们可视为本质相同的数的不同表示，其间的相互转换也十分简单。

1. 二进制数与十六进制数之间的转换

4 位二进制数即可表示所有组成十六进制数的数字符号，所以在把二进制数转换为十六进制数时，只需把二进制数的整数部分从低位开始，每 4 位一分节，并计算出分节后的数字。而对于小数部分，则从高位开始，每 4 位一分节，如果最后不足 4 位，则补 0，补足 4 位，然后把相应的节转换成十六进制数。

如把二进制数 101001111.101001011 转换为十六进制数：

0001	0100	1111	.	1010	0101	1000
1	4	F	.	A	5	8

即该二进制数所对应的十六进制数为 14F.A58。

把十六进制数转换为二进制数时，只需把每个十六进制数字符号转换为相应的 4位二进制数即可。如把十六进制数 395.D4 转换为二进制数：

3	9	5	.	D	4
0011	1001	0101	.	1101	0100

即十六进制数 395.D4 所对应的二进制数为 1110010101.110101。

在平常的数制转换中，当把二进制数转换成十进制数时，人们往往先把二进制数转换成十六进制数，再把得到的十六进制数转换为十进制数；在把十进制数转换为二进制数时，也往往先把十进制数转换为十六进制数，再把所得到的十六进制数转换为二进制数，这样既快又不容易出错。

为了区分不同数制表示的数，在书写时人们往往用字母 B（Binary number）表示二进制数，用字母 O（Octal number）表示八进制数，用字母 D（Decimal number）表示十进制数，用字母 H（Hexadecimal number）表示十六进制数。例如：

$(11011101)_2$ = 11011101B　　$(114325151)_8$ = 114325151O

$(473473281)_{10}$ =473473281D　　$(1FEF)_{16}$ =1FEFH

2.3 计算机中的数据表示

计算机采用二进制表示所有数据（如数值、文本、图形、声音、动画等），处理在计算机中发生的每件事情。采用二进制数而不是人们熟悉的十进制数来存取和处理数据，其主要原因如下。

① 二进制数只使用 0 和 1 两个符号，状态简单，其数据表示和信息传递都比十进制数更易实现，可用具有两种简单物理状态的元器件来实现，稳定性好，可靠性高。例如，晶体管导通为 1，截止为 0；高电压为 1，低电压为 0；灯亮为 1，灯灭为 0。

② 二进制数比十进制数的运算简单，其“和”与“积”的运算规则都只有 3 条：

加法：　0+0=0　　0+1=1　　1+1=10

乘法：　0×1=0　　1×0=0　　1×1=1

这种运算规则大大简化了计算机中实现运算的电子线路。实际上，在计算机中，减法、乘法及除法都可以分解为加法运算来完成。

③ 二进制便于表示逻辑值（用 1 表示“真”，用 0 表示“假”），可将逻辑代数和逻辑电路作为计算机电路设计的数学基础，易于实现。

计算机常采用具有两种稳定状态的电子元器件来表示二进制数 0 和 1，每个电子元器件代表二进制数中的 1 位。因此，位（bit）是计算机中的最小信息单位，若干个电子元器件的组合能同时存放许多个二进制数。通常，将 8 个二进制位称为一字节（Byte），字节是信息的基本单位。一字节可以表示 2^8 = 256 种状态，可以存储一个无符号整数（0～255 范围内）或一个英文字母的编码。在计算机中，通常以字节为单位表示文件或数据的长度及存储容量的大小。

2.3.1 数值数据的机内表示

数在计算机中的表示称为**机器数**，由于计算机存储一个参与运算的机器数所使用的电子器件的基本个数是固定的，因此通常把这种具有固定位数的二进制串称为字，而把字包含的二进制数的位数称为字长。通常所说的计算机是多少位就是指机器字长的二进制位数。例如，16 位微机的字长为 16 位，32 位微机的字长为 32 位。一般说来，计算机的字长越长，其性能就越高（关于字长的概念，请参考第 3 章）。

1．机器数与真值

计算机能表示的数值范围受到机器字长位数的限定。例如，某种字长为 16 位（即存储单元的长度）的计算机能表示的无符号整数范围为 0～65535（0～2^{16}-1）。

没有考虑正负符号的数称为无符号数。在实际应用中，数总是有正负的，在计算机数的表示中，通常把高位作为符号位，其余位作为数值位，并规定用 0 表示正数，用 1 表示负数。因此，机器数是连符号一起数字化了的，如+79 和-79 可分别表示为（假设数据长度为 1 字节）图 2-1

所示的数据，连同一个符号位在一起作为一个数据，就称为机器数。机器数的数值部分就称为机器数的真值。

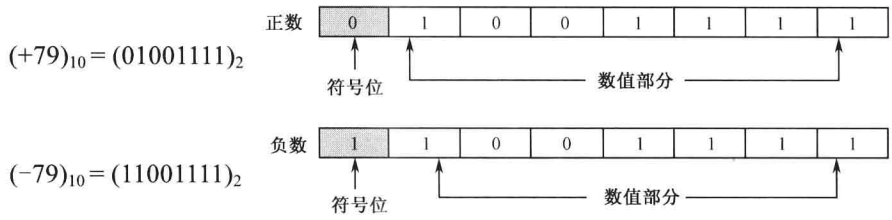


图 2-1　8 位机器数示例

2．定点数和浮点数

定点数是指约定小数点隐含在某一固定位置上（小数点不占据存储空间）的数，称为定点数表示法。在计算机中通常采用两种简单的约定，一种是将小数点的位置固定在数据的符号位之后、最高位之前，这样的数是定点小数。定点小数是纯小数，即其值小于 1，如图 2-2（a）所示。

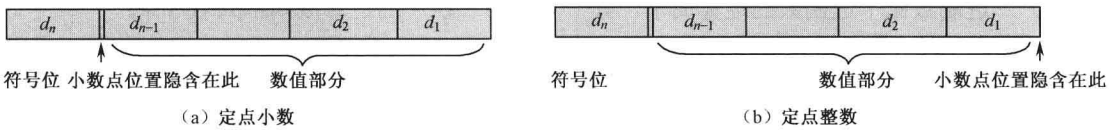


图 2-2　定点数的表示方法

在图 2-2 中，d_n 是符号位（0 表示正数，1 表示负数），$d_{n-1}d_{n-2}\cdots d_1$ 称为尾数，d_{n-1} 为最高有效位。例如，若有 8 位的定点小数 01000101，其最左边的 0 是符号位，小数点在左边的 0 和 1 之间，此数的大小为

$$(01000101)_2=1\times2^{-1}+1\times2^{-5}+1\times2^{-7}=(0.5390625)_{10}$$

定点数的另一种常见约定是将小数点固定在最右边，这样的数是定点整数，如图 2-2（b）所示。对于 n 位的定点整数，若用最左位表示符号位，则这样的数称为有符号整数，能表示的数据的范围是 $-2^{n-1}\sim(2^{n-1}-1)$；如果不要符号位，即最左位也表示数值，就称为无符号整数，能表示的数的范围是 $0\sim2^n$。

例如，有 8 位的定点整数 11000001，若为有符号整数，则其值为

$$(11000001)_2=-(1\times2^6+1\times2^0)=(-65)_{10}$$

若为无符号整数，则其值为

$$(11000001)_2=1\times2^7+1\times2^6+1\times2^0=(193)_{10}$$

当数据小于定点数能表示的最小值时，计算机将它们作为 0 处理，称为下溢；当数据大于定点数能表示的最大值时，计算机也无法表示，称为上溢。上溢和下溢统称为溢出。

人们平常用科学计数法表示的数，其小数点位置并不固定。例如，−1234567 可以表示为

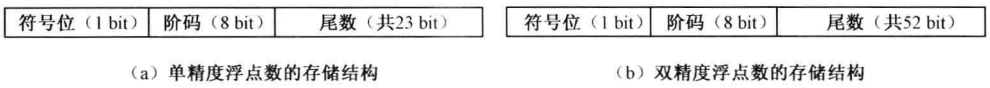
$$-1234567=-1.234567\times10^6=-123.4567\times10^4=-0.1234567\times10^7$$

在计算机中，这样的数被称为浮点数，能够表示比定点数更大的数值范围。任何 N 进制的数 X 都可以写成：$X=N^E\times M$。

其中，M 称为数 X 的尾数，在计算机中是一个纯小数；E 为数 X 的阶码，是一个整数；N 代表进制基数，是位权 N^E 的底数。这种表示方法相当于数的小数点位置随位权 N^E 的不同而在一定范围内可以自由浮动，所以称为**浮点数**。

计算机中的浮点数以二进制表示，所以底数 N 总是约定为 2，在计算机中存储时不需出现。因此，在机器中表示一个浮点数时，只需存储其尾数和阶码。尾数部分给出有效数字的位数，用定点小数形式表示，决定了浮点数的表示精度。阶码指明小数点在数据中的位置，用整数形式表示，决定了浮点数的表示范围。阶码也有正负之分，正数表示向右移动小数点，负数表示向左移动小数点。

计算机中采用的浮点数主要有两种：一种称为单精度浮点数（如 C 语言中的 float，Visual Basic 中的 single），另一种是双精度浮点数（即许多程序语言中的 double）。单精度浮点数采用 4 字节保存数据，双精度浮点数采用 8 字节保存数据，能够表示更大范围的数据，其存储结构的示意图如图 2-3 所示。



（a）单精度浮点数的存储结构　　（b）双精度浮点数的存储结构

图 2-3　浮点数的存储结构示意图

其中的符号位表示浮点数的正负，1 代表负数，0 代表正数；阶码表示 2 的多少次方，单精度数用 8 位二进制数表示阶码，双精度数用 11 位二进制数表示阶码。单精度数的尾数是 23 位，双精度数的尾数是 52 位，实际上它们分别为 24 位和 52 位。原因是，计算机在处理浮点数时，会通过阶码的调整来移动小数点的位置，使其成为“1.bbbb……bb”的形式，其中的 b 表示二进制的 0 或 1，在存储浮点数的尾数时，只存储其中的“bbbb……bb”，小数点和前面的 1 被省略，在计算时再将它加上，这样可以在存储位数相同的情况下，表示更大范围的数。

3．原码

正数的符号位用 0 表示，负数的符号位用 1 表示，这种数据编码就称为原码。图 2-1 所示的机器数实际上就是+79 和−79 的原码。再如，若用 1 字节表示数，则 41 和−41 的原码如下：

$$X=+41,\quad [X]_{原}=00101001$$

$$X=-41,\quad [X]_{原}=10101001$$

原码表示法较为简单，且数的真值容易计算，但两个符号相反的数要进行相加，实际上需要用减法完成。为了简化计算，用加法实现异号数相加，计算机往往用反码或补码表示数据。

4．反码

正数的反码与原码相同，负数的反码是将机器数的真值部分（除符号之外的其余数据位）按位取反（即 0 变 1，1 变 0）所得到的数据，如图 2-4 所示。

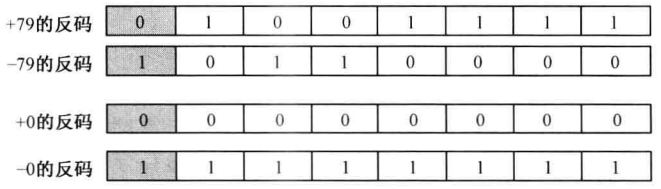


图 2-4　反码

在计算反码的数值时一定要小心，当一个符号数据用反码表示时，最高位是符号位，后面的才是数值部分。

当最高位为 0 时，可按权展开，计算数据对应的十进制数大小。例如，用反码表示的数 00110101，其最高位为 0，则说明它是正数。其值的计算为：

$$[00110101]_{反}=+(1\times2^5+1\times2^4+1\times2^2+1\times2^0)=+(32+16+4+1)=+53$$

当最高位是1时，则不能直接按权展开来计算数据的十进制数大小，而应该将反码的数值部分按位取反之后，再按权展开，才能计算出数据正确的大小。例如 10110101，则不能按如下方式计算其大小：

$$[10110101]_{反}=-(1\times2^5+1\times2^4+1\times2^2+1\times2^0)=-(32+16+4+1)=-53$$

这个结果是错误的，正确的计算方式是先将其取反，再求出它所对应的原码：

$$[10110101]_{反}=[11001010]_{原}=-(1\times2^6+1\times2^3+1\times2^1)=-74$$

5. 补码

0 有两个不同的反码，+0 和-0 的反码不相同，但实际上它们表示的大小相同，是同一个数。这会给数据的计算带来许多问题，如 3+0 和 3-0 会得到不同的结果。为了解决类似问题，计算机多用补码表示数据。

正数的补码与原码相同，即最高位用 0 表示符号，其余位是数据的真值部分。负数的补码则为在它的反码的最末位加 1 计算所得，如图 2-5 所示。

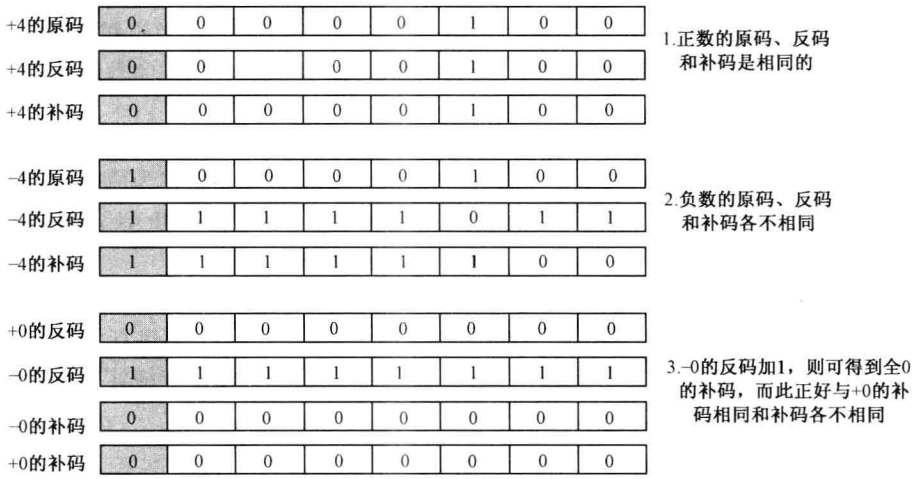


图 2-5　补码

用补码表示数据，不仅解决了+0 和-0 的不同编码问题，更重要的是，补码能够将减法变成加法，简化计算机的设计，因为它不需要设计减法的计算机实现。例如，假设计算机用 1 字节保存数据，则 52-11 采用减法和补码加法的运算过程如图 2-6 所示。

实际上，计算机的减法运算多用补码的加法运算实现，运算结果与机器数据的长度有很大关系。因为补码加法是带符号的运算，就是说数据的符号位也参加了与普通数位相同的运算，一样会产生进位的问题。这意味着，两个正数相加的结果可能是负数，两个负数相加的结果可能是正数（如果运算的结果超出了数的表示范围，就会产生这种情况）。例如，假设某计算机用 2 字节表示整数，则 30000+20000 的计算过程如图 2-7 所示。

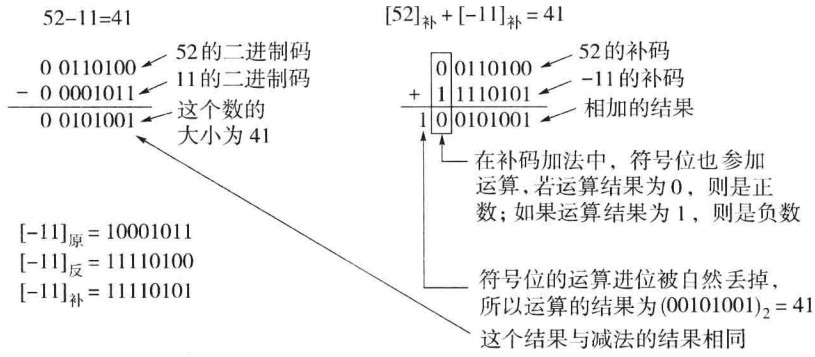


图 2-6　减法和补码加法

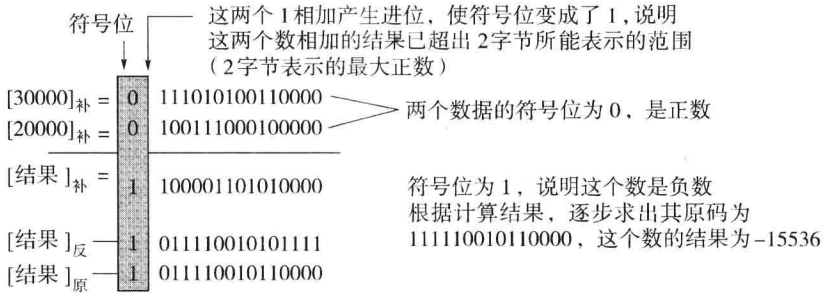


图 2-7　3000+2000 的计算过程

能够理解吗？30000+20000=−15536？但这有个前提，即用 2 字节表示整数，若用 4 字节，则结果为 50000。

说明：可用 Windows 系统“附件”中的“计算器”验证此计算过程；在 C 语言中，若用 short int 表示这 3 个数，所得结果也是相同的。

总之，人们常见的图形、图像、声音、数据、字符、汉字等信息在计算机中都是以二进制数据保存和处理的。在计算机中，二进制数的存储单位如下。

bit（位）：能够存放 1 位二进制数的 0 或 1。

Byte（字节）：存放 8 位二进制数，即 1Byte = 8 bit。

KB（千字节）：1KB = 1024 Byte。

MB（兆字节）：1MB = 1024 KB。

GB（吉字节）：1GB = 1024 MB。

TB（太字节）：1TB = 1024 GB。

2.3.2　字符在计算机中的表示

1. 标准 ASCII 码

字符是指各种类型的符号，包括英文字母、数字、各种数学符号或中文字符等。字符在计算机中是按照事先约定的编码形式存放的，“码”就是 0 和 1 的各种组合。所谓编码，就是用一连串二进制数码代表一位十进制数字或一个字符。编码工作由计算机在输入、输出时自动进行。一个编码就是一串二进制数 0 和 1 的组合。例如，可用 0110001 代表十进制数中的数字 1，用 1000001 代表大写英文字母 A，用 0100101 代表百分号%等。

不同国家、不同公司生产的计算机，可能采用不同的编码表示数字和符号，要想在不同类型的计算机之间交换文档和数据，就要求这些计算机用相同的编码表示和处理数据，否则无法进行数据交换。例如，A 类计算机用“10001”表示字母“X”，而 B 类计算机却用“10001”表示字母“Y”，它们如何进行通信呢？如果把用 A 类计算机编写的文档复制到 B 类计算机中，则原文中所有的“X”都变成“Y”了。

目前，计算机中广泛使用的编码是 ASCII 码，即美国标准信息交换码（American Standard Code for Information Interchange），该编码被 ISO（国际标准化组织）采纳而成为一种国际通用的信息交换标准代码。我国 1980 年颁布的国家标准《GB1988—1980 信息处理交换用七位编码字符集》也是根据 ASCII 码制定的，它们之间只在极个别的地方存在差别。

标准 ASCII 码采用 7 位二进制编码表示各种常用符号，即每个字符由 7 位二进制码组成，共有 2^7=128 种不同编码，可表示 128 个符号。ASCII 的码值可用 7 位二进制代码或 2 位十六进制代码表示，其排列次序为 $d_7\ d_6\ d_5\ d_4\ d_3\ d_2\ d_1$。但由于计算机存取数据的基本单位是字节，所以一个字符在计算机内实际是用 1 字节即 8 位二进制数表示的，其最高位 d_8 为 0。

在计算机通信中，最高位常作为奇偶校验位。以偶校验为例，就是字符的 ASCII 码将 d_8 设置为 0 或 1，使每个字符编码中 1 的个数保持偶数个。例如，字符“A”的 7 位 ASCII 编码是 1000001，则将 d_8 置为 0，使得 8 位 ASCII 码为 01000001，共 2 个 1，为偶数个；而字符“*”的 ASCII 编码为 0101010，将 d_8 置为 1，则其 8 位 ASCII 码变为 10101010，共 4 个 1，为偶数个。在采用偶校验的编码系统中，如果发现某个符号的 ASCII 码中 1 的个数不是偶数个，则说明该码有错误，可能是某位 0 变成了 1，也有可能是某位 1 变成了 0。奇校验同偶校验原理一样，只不过编码中 1 的个数是奇数个而已。在计算机系统中，常用奇校验或偶校验来判定信号在传送过程中是否发生了错误。表 2-2 是标准 ASCII 码所定义的符号及其编码。

表 2-2　标准 7 位 ASCII 码字符集

$d_7\ d_6\ d_5$ / $d_4\ d_3\ d_2\ d_1$	000	001	010	011	100	101	110	111
0 0 0 0	NUL (0)	DLE (16)	空格(32)	0 (48)	@ (64)	P (80)	` (96)	p (112)
0 0 0 1	SOL (1)	DC1 (17)	! (33)	1 (49)	A (65)	Q (81)	a (97)	q (113)
0 0 1 0	STX (2)	DC2 (18)	” (34)	2 (50)	B (66)	R (82)	b (98)	r (114)
0 0 1 1	ETX (3)	DC3 (19)	# (34)	3 (51)	C (67)	S (83)	c (99)	s (115)
0 1 0 0	EOT (4)	DC4 (20)	$ (36)	4 (52)	D (68)	T (84)	d (100)	t (116)
0 1 0 1	ENQ (5)	NAK(21)	% (37)	5 (53)	E (69)	U (85)	e (101)	u (117)
0 1 1 0	ACK (6)	SYN(22)	& (38)	6 (54)	F (70)	V (86)	f (102)	v (118)
0 1 1 1	BEL (7)	ETB(23)	’ (39)	7 (55)	G (71)	W (87)	g (103)	w (119)
1 0 0 0	BS (8)	CAN(24)	(40	8 (56)	H (72)	X (88)	h (104)	x (120)
1 0 0 1	HT (9)	EM (25)	) 41	9 (57)	I (73)	Y (89)	i (105)	y (121)
1 0 1 0	LF (10)	SUB(26)	* (42)	: (58)	J (74)	Z (90)	j (106)	z (122)
1 0 1 1	VT (11)	ESC(27)	+ (43)	; (59)	K (75)	[(91)	k (107)	{ (123)
1 1 0 0	FF (12)	FS (28)	, (44)	< (60)	L (76)	\ (92)	l (108)	| (124)
1 1 0 1	CR (13)	GS (29)	- (45)	= (61)	M (77)	] (93)	m (109)	} (125)
1 1 1 0	SO (14)	RS (30)	. (46)	> (62)	N (78)	↑ (94)	n (110)	_ (126)
1 1 1 1	SI (15)	US (31)	/ (47)	? (63)	O (79)	—(95)	o (111)	DEL(127)

说明：从表 2-2 可以看出，A～Z 的 26 个大写字母，其编码是用 01000001～01011010（十进制数 65～90）这 26 个连续代码来表示的。而 0～9 的数字，则是用 00110000～00111001（十进制数 48～57）这 10 个连续代码来表示的。

在表 2-2 中，ASCII 码为 0～31 和 127（即 NUL～US 和 DEL）的这 33 个符号是不可显示的字符，一般用做数据通信传输控制符号、打印或显示的格式控制符号、对外部设备的操作控制符号或信息分隔符号等，其余 95 个是可以显示或打印的符号。

用键盘输入字符时，键盘中的编码电路会将按键转换成对应的二进制 ASCII 码，并送入内存的特定区域中。在输出字符时，计算机则按 ASCII 码表，将字符的 ASCII 码转换成对应的字符，然后输出到显示屏或打印机上。

例如，从键盘上输入字符串“CHINA”，键盘会将这几个字符转成二进制数字串 01000011、01001000、01001001、01001110、01000001，然后送入计算机中。反之，存储器中的二进制数字串 01010111、01010000、01010011 在显示屏或打印机上输出时，人们看到的结果是“WPS”这 3 个字符组合。

请参照表 2-2 所示的 ASCII 码表理解例子的含义。

2．扩展 ASCII 码

由 7 位二进制编码构成的 ASCII 基本字符集只有 128 个字符，不能满足信息处理的需要。近年来，对 ASCII 码字符集进行了扩充，采用 8 位二进制数据表示一个字符，编码范围为 00000000～11111111，一共可表示 256 个字符和图形符号，称为扩展 ASCII 码字符集。这种编码是在原 ASCII 码 128 个符号的基础上，将它的最高位设置为 1 进行编码的，扩展 ASCII 码中的前 128 个符号的编码与标准 ASCII 码字符集相同。

3．Unicode 编码

Unicode 编码是继 ASCII 字符编码后，由 ISO 在 20 世纪 90 年代初期制定的各国文字、符号的统一性编码。该编码采用 16 位编码体系，可容纳 65 536 个字符编码，几乎能够表达世界上所有书面语言中的不同符号。

Unicode 编码主要用来解决多语言的计算问题，如不同国家的字符标准，允许交换、处理和显示多语言文本以及公用的专业符号和数学符号。随着因特网的迅速发展，不同国家之间的人们进行数据交换的需求越来越大，Unicode 编码因此而成为当今最为重要的交换和显示的通用字符编码标准，它适用于当前所有已知的编码，覆盖了美国、欧洲、中东、非洲、印度、亚洲和太平洋地区的语言，以及专业符号。

2.3.3 中文字符在计算机中的表示

汉字具有字形优美、生动、形象的特点，是中文信息处理的主要文字符号。但汉字实在太多，约有 6 万多个，而且字形复杂，给计算机处理带来了较大困难。

从 20 世纪 60 年代起，我国就开始了对汉字信息处理技术的探索和研究，20 世纪 70 年代曾对各类汉字的使用频率进行过统计，发现有 3755 个汉字是最常使用的，覆盖率高达 99.9%，一般人都知道这些汉字的读音，所以把它们按拼音进行排序，称为一级汉字；此外，还有 3008 个汉字的使用也较多，一级汉字再加上这 3008 个汉字，覆盖了汉字应用范围的 99.99%，基本上能满足各种场合的应用，所以把这 3008 个汉字称为二级汉字，并按偏旁部首进行排序。1980 年，我国公