程序员级高级程序员级硬件知识

王爱英　主编
周明德　主审

中国计算机软件专业技术资格和水平考试
统编辅导教材

清华大学出版社

程序员级高级程序员级
硬件知识

王爱英　主编
周明德　主审

清华大学出版社

内 容 简 介

本书为计算机软件专业技术资格和水平考试（程序员级、高级程序员级）统编辅导教材、依照1991年公布的《中国计算机应用软件人员水平考试大纲》和统编教材编写组所确定的大纲进行编写。本书侧重于硬件知识，内容包括计算机的数据、码制及其运算、计算机的组成与结构（指令系统、中央处理器、存储器系统和输入输出系统）、计算机系统的结构及性能评价等。既着重于基本原理的阐述，又注意到计算机的发展动向。书中介绍了当前在计算机中广泛应用的技术及其硬件结构，如流水线组织、RISC技术、多级存储体系、并行处理技术和计算机网络等。

本书还可作为计算机软件专业学生学习《计算机组成和系统结构》课程的教材，或供从事计算机事业的工程技术人员参考。

（京）新登字158号

程序员级高级程序员级硬件知识

[redacted]

周明德 王审

☆

清华大学出版社出版

北京 清华园

北京市联华印刷厂印刷

新华书店总店科技发行所发行

☆

开本：787×1092 1/16 印张：13.375 字数：328千字

1992年4月第1版 1992年11月第2次印刷

印数：20001—30000

ISBN 7-302-01027-7/TP·375

定价：8.20元

出 版 说 明

当今国际间的竞争是综合国力的竞争，关键又是科学技术的竞争，说到底是人才的竞争。而电子信息技术又是当今国际间竞争的热点，它渗透和影响着现代化社会生活的各个方面。而科学技术（包括电子信息技术）是要靠人去掌握、去应用、去发展的，国家的强盛、民族的振兴靠人才、人才的培养靠教育。所以，党中央要求我们把经济建设转到依靠科技进步和提高劳动者素质的轨道上来。

培养人才要坚持多种形式、多种途径，大力开展岗位培训，不断提高职工队伍的技术和专业水平。计算机软件专业技术资格和水平考试制度，就是为了加速我国电子信息技术的广泛应用和软件事业的发展，科学考核和合理使用人才，促进计算机软件人才的国际交流与合作，进一步深化职称改革。这种考试是于1985年首先在上海、云南、四川三省市实行的，1987年发展为部分省、区、市的联合考试，到1988年全国已经有31个省、区、直辖市和计划单列市参加程序员、高级程序员两个级别的联合考试，1989年发展为程序员级、高级程序员级、系统分析员级三个级别的联合考试，1990年在全国统一组织实施了软件专业技术职务任职资格（水平）考试。1991年3月计算机软件专业技术资格和水平考试工作会议，对考试《暂行规定》作了修改，新的《暂行规定》把“资格”考试与“水平”考试从性质上和考试级别上作了区分，“资格”考试除了保留“程序员”、“高级程序员”的级别外，增加了“初级程序员”的考试级别，而“水平”考试则仍为“程序员”、“高级程序员”、“系统分析员”三个级别。同时在报考条件与考试的内容和评分标准等方面，也都作了相应的规定。1991年的软件专业技术资格和水平考试就是依照新的《暂行规定》进行的。实践证明这是一种严格的认定考试制度，给应试者提供了一次均等的机会。软件专业技术资格和水平考试，每年举行一次，实行全国统一组织、统一大纲、统一试题、统一评分标准。

中国计算机软件专业技术资格和水平考试委员会（以下简称考委会）把考试和培训两项相辅相成的工作担当起来是责无旁贷的。为了使全国参加统一考试的考生得到较好的辅导，编一套全国统一的辅导教材是完全必要的。1991年9月考委会在北京主持召开了辅导教材编审委员会会议，对统编辅导教材的编写出版作了部署，成立了统编辅导教材编审委员会，并确定了这套教材六本书的主编和主审，他们是：《初级程序员级软硬件知识》（主编：刘尊全；主审：吴克忠）、《程序员级、高级程序员级硬件知识》（主编：王爱英；主审：周明德）、《程序员级、高级程序员级软件知识》（主编：施伯乐；主审：潘锦平）、《计算机综合应用知识》（主编、罗晓沛；主审：侯炳辉）、《程序员级、高级程序员级程序设计》、（主编、张福炎；主审：郑国梁）、《1990年度—1991年度试题分析与解答（初级程序员级、程序员级、高级程序员级、系统分析员级）》（主编：张然；主审：施伯乐）。我们的本意是要把这套全国统编辅导教材搞成为具有正确性、科学性、系统性，而且针对性强，在一段时间内相对稳定的好教材。虽然吸取了北京、上海、成都等地已有的辅导教材的长处，但由于时间紧，经验不足，错误在所难免，请读者和有关专家能给予指正。

前　言

本书为中国计算机软件专业技术资格和水平考试（程序员级、高级程序员级）的统编辅导教材，依照1991年公布的《中国计算机应用软件人员水平考试大纲》和统编教材编写组所确定的大纲进行编写。程序员、高级程序员对硬件知识的要求是不同的，本书写作时兼顾了两方面要求，书中带＊号部分是专为报考高级程序员级的学员编写的，报考程序员级的学员暂时可以不用阅读。本书侧重硬件知识，内容包括计算机的数制、码制及其运算、计算机的组成与结构（指令系统、中央处理器、存储器系统和输入输出系统）、计算机系统的结构及性能评价等。既着重于基本原理的阐述，又注意到计算机的发展动向。书中介绍了当前在计算机中广泛应用的技术及其硬件结构，如流水线组织、RISC技术、多级存储体系、并行处理技术和计算机网络等。

计算机技术发展很快，考虑到本书的适应性，书中的某些部分略微超出1991年考试大纲所规定的广度和深度，如模拟与仿真及并行处理等。计算机网络的软、硬件另有专著供考生阅读，限于篇幅，在本书中不作深入讨论。

本书的第一、二章（除 2.1）由卢义明编写，第三、四、五章及2.1节由王爱英编写，第六章由林学闇、卢义明、王爱英编写，第七章由张吉锋、王爱英、张志宏编写，第八章由郑纬民编写。王爱英对全书进行了统编与修改。周明德审阅了全书，并提出了不少宝贵意见，在此表示感谢。

为了赶在1992年4月出版，与读者见面，全体编写人员、主审与编辑同志齐心协力、密切合作，付出了辛勤劳动，但因时间实在太紧，肯定有不少不足之处，希望同行与广大读者对本书提出中肯意见。

编　者

1992年1月

教材编审委员会

秘书组联络地址：北京市海淀区学院南路55号（邮政编码　100081）

目　录

第一章　计算机的数制、码制及其运算

本章主要讲述数据和信息（如文字、符号、语言和图像等）在计算机中的表示方法、计算机中数据的算术运算和逻辑运算等。

1.1　数制及其转换

计算机内部是以二进制形式表示数据，以二进制形式对这些数据进行算术逻辑运算，并且以二进制形式存储数据的。但是如果当人们往计算机中输入信息或在屏幕上显示数据时，也必须用二进制表示，就会给人们带来无限的烦恼。因此，在人-机界面处，数据最好用人们熟悉的十进制数、八进制数或十六进制数表示。同一个数用不同的数制表示，就存在它们之间相互转换的问题。

1.1.1　进位计数制

一、进位基数和位权值

十进制数，它的数值部分是用10个不同的数码表示的，这10个数码是：0，1，2，3，4，5，6，7，8，9。数码在数据中的位置不同，所代表数值的大小也不同。例如123.45这个十进制数，3在小数点左面1位上，它代表的数值是3×10^0，1在小数点左面3位上，它代表的数值是1×10^2，5在小数点右面2位上，它代表的数值是5×10^{-2}。这个数可以写成：

$$123.45=1\times10^2+2\times10^1+3\times10^0+4\times10^{-1}+5\times10^{-2}$$

任意一个十进制数S都可以表示成（1.1）式：

$$\begin{aligned}S&=K_nK_{n-1}\cdots K_1K_0.K_{-1}K_{-2}\cdots K_{-m}\\&=K_n\cdot10^n+K_{n-1}\cdot10^{n-1}+\cdots+K_1\cdot10^1+K_0\cdot10^0+K_{-1}\cdot10^{-1}+K_{-2}\cdot10^{-2}+\cdots+K_{-m}\cdot10^{-m}\end{aligned}\tag{1.1}$$

其中K_i（$i=n,n-1,\cdots,1,0,-1,\cdots,-m$）是0，1，2，…，8，9十个数码中的任意一个。10为进位基数，10^i代表了这位数的权值。

进位基数指的是在该进位计数制中可能用到的数码个数。每一位计满这个基数后，应向高位进位。对十进制数“逢十进一”；对二进制数“逢二进一”。

八进制数所用到的数码个数为八个，分别是0，1，2，…，6，7。

十六进制数所用到的数码个数为十六个，分别是0，1，2，…，8，9，A，B，C，D，E，F。A，B，C，D，E，F与十进制数的对应关系分别是：10，11，12，13，14，15。

对于R进制数（R为任意正整数），所用到的数码个数为R个，分别是：0，1，2，…，R－1。每一位数当计满R后，向高位进位。

一个任意的R进制数S，都可以写成（1.2）式：

$$S = K_n K_{n-1} \cdots K_1 K_0 . K_{-1} K_{-2} \cdots K_{-m}$$
$$= K_n \cdot R^n + K_{n-1} \cdot R^{n-1} + \cdots + K_1 \cdot R^1 + K_0 \cdot R^0 + K_{-1} \cdot R^{-1} + \cdots + K_{-m} \cdot R^{-m} \quad (1.2)$$

其中R为进位基数，R^i是对应位的权值。（1.2）式称之为进位计数制的按权展开式。

［例 1］ 请写出二进制数S = 1011.11的按权展开式。

解：按权展开式为：$S = 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 + 1 \times 2^{-1} + 1 \times 2^{-2}$

其中2^3、2^2、2^1、2^0、2^{-1}、2^{-2}是相应位的权值。

二、R进制数转化成十进制数

根据（1.2）式，把任意R进制数写成按权展开式后，再求和，就是这个R进制数所对应的十进制数。

［例 2］ 把二进制数11011.101转换成十进制数。

解：$(11011.101)_2 = 1 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 + 1 \times 2^{-1} + 0 \times 2^{-2} + 1 \times 2^{-3}$

$= 16 + 8 + 2 + 1 + 0.5 + 0.125 = (27.625)_{10}$

［例 3］ 把八进制数123转换成十进制数。

解：$(123)_8 = 1 \times 8^2 + 2 \times 8^1 + 3 \times 8^0 = 64 + 16 + 3 = (83)_{10}$

［例 4］ 把十六进制数1AB转换成十进制数。

解：$(1AB)_{16} = 1 \times 16^2 + (10) \times 16^1 + (11) \times 16^0 = 256 + 160 + 11 = (427)_{10}$

注：十六进制数A，B分别对应十进制数10，11。

1.1.2 进位计数制之间的转换

一、十进制整数转换成二进制整数

从上节介绍中可以看到，任何一个十进制整数都可以用一个R进制整数精确表示。

［例 5］ 将$(123)_{10}$转换成二进制数。

解：十进制数123，能用一个二进制整数表示，假设表示形式如下：

$$(123)_{10} = (a_n a_{n-1} \cdots a_1 a_0)_2 \quad (1.3)$$

问题是如何确定式中的a_n，a_{n-1}，…，a_1，a_0的值。

常用的方法有两种：

1．除以R取余法（此例中R = 2）

2 | 1 2 3		余数	
2 | 6 1	……	1	……转换后低位数
2 | 3 0	……	1	
2 | 1 5	……	0	
2 | 7	……	1	
2 | 3	……	1	
2 | 1	……	1	
0	……	1	……转换后高位数

最后算得结果：$(123)_{10} = (1111011)_2$

2．减权定位法

$(123)_{10} = a_n \cdot 2^n + a_{n-1} \cdot 2^{n-1} + \cdots + a_1 \cdot 2^1 + a_0 \cdot 2^0$

把$(123)_{10}$展开成2的整数次幂之和，找到对应位上a_i的取值。

$(123)_{10} = 64 + 32 + 16 + 8 + 2 + 1$

$= 1 \times 2^6 + 1 \times 2^5 + 1 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0$

则$a_6 = 1, a_5 = 1, a_4 = 1, a_3 = 1, a_2 = 0, a_1 = 1, a_0 = 1$，最后转换结果为$\cdot (123)_{10} = (1111011)_2$

二、十进制小数转化为R进制小数

假设将十进制小数 $(0.315)_{10}$转换成二进制小数。常用的方法是把给定的十进制小数乘以2，取积的整数部分，得到二进制小数的小数点后的第一位；乘积的小数部分再乘以2，积的整数部分为小数点后的第2位；一直重复做下去，就可以得到希望的二进制小数。

	整数部分
$0.315 \times 2 = 0.63$	0……小数点后第1位数
$0.63 \times 2 = 1.26$	1
$0.26 \times 2 = 0.52$	0
$0.52 \times 2 = 1.04$	1……小数点后第4位数

假设本题只取4位小数，最后结果为近似值：$(0.315)_{10} = (0.0101)_2$。从这个例子可以看出，不是所有的十进制小数全可以用一个精确的二进制小数表示。

三、二进制数化成八进制数

三位二进制数，正好完全表示了八进制数的8个代码，它们之间的对应关系如下：

二进制	000	001	010	011	100	101	110	111
八进制	0	1	2	3	4	5	6	7

二进制数转化成八进制数的方法是：从小数点开始，分别向左、向右，每3位二进制数为一组，用八进制数来书写。

例如，二进制数10110111.01101转化成八进制数是267.32，其过程如下：

二进制数	0	1 0	1 1 0	1 1 1 .	0 1 1	0 1	0
八进制数		2	6	7 .	3	2	

若小数点左侧位数不是3的倍数，则最左侧用0补充；若小数点右侧位数不是3的倍数，则最右侧用0补充。

反过来，八进制数转化成二进制数的方法是：将每个八进制数用3位二进制数来书写，其最左侧或最右侧的0可以省去。

四、二进制数转化成十六进制数

转换方法是：从小数点开始，分别向左、向右，每4位二进制数为一组，用十六进制数来书写。同理，若小数点左（右）侧位数不是4的倍数，则最左（右）侧用0补充。

例如，二进制数110110111.01101转换成十六进制数是1B7.68。

反过来，十六进制数转化成二进制数的方法是：将每个十六进制数用4位二进制数来书写，其最左侧或最右侧的0可以省去。

如果一个分数的分母是2的整数次幂，可用下面的方法把它转化成二进制数。

［例 6］ 把$\left(\frac{3}{32}\right)_{10}$化成二进制数。

解： $\frac{3}{32}=3\times2^{-5}=(11)_2\times(0.00001)_2=(0.00011)_2$

1.2 原码、补码和反码

1.2.1 机器数与真值

数有正、负两种，在计算机中数的符号是用数码表示的。一般情况下，用0表示正数，用1表示负数。通常符号位放在数的最高位。例如$x_1=(+1011011)_2$，$x_2=(-1011011)_2$，它们在机器中表示为

x_1：	0	1	0	1	1	0	1	1
x_2：	1	1	0	1	1	0	1	1

最左边一位代表符号位，连同符号位在一起作为一个数，称为机器数；而它的数值称为机器数的真值。

1.2.2 机器中常用的几种码制表示法

在不同的计算机中，机器数的表示方法不完全相同。通常机器数有4种表示法，各有自己的特点，下面分别加以介绍。

一、原码表示法

1．定义

设x为小数，用小数点左面一位表示数的符号；以$[x]_{原}$表示x的原码。则

$$[x]_{原}=\begin{cases}x & (0\leqslant x<1)\\ 1-x & (-1<x\leqslant 0)\end{cases}$$

例如，$x_1=0.1011$，$x_2=-0.1011$，则

$$[x_1]_{原}=0.1011,\quad [x_2]_{原}=1.1011。$$

2．真值零的原码表示法

对于真值零，有正零和负零两种表示：

$$[+0]_{原}=0.00\cdots0;\quad [-0]_{原}=1.00\cdots0。$$

3．原码表示的数的范围

如果用n位二进制数表示小数，其中包括一位符号位，所能表示的最大数为$+(1-2^{-(n-1)})$，最小数为$-(1-2^{-(n-1)})$。

例如n=8，则

最大数是　$1-2^{-(8-1)}=1-0.0000001=0.1111111$

最小数是　$-(1-2^{-(8-1)})=-1+0.0000001=-0.1111111$

则所能表示的数的范围是：$1-2^{-7}$——$-(1-2^{-7})$，用十进制数表示为：$\frac{127}{128}$——$-\frac{127}{128}$。

4．原码表示的数的个数

n位二进制位有2^n个不同的状态，即可以表示2^n个数。但由于零占掉了两个编码，所以用n位二进制位只能表示2^n-1个原码数。

例如n＝8，则8位二进制位能表示255（即2^8-1）个原码数。

二、补码表示法

1．模数的概念

在计算机中，机器表示数据的字长是固定的。对于n位数来说，模数M的大小是：n位数全为1，且在最末位再加1。实际上模数的值已经超过了机器所能表示的数的范围，因此模数在机器中是表示不出来的。若运算结果大于模数，则模数自动丢掉。如果有n位整数（包括1位符号位），则它的模数为2^n；如果有n位小数（包括1位符号位），则它的模数为2。

2．定义

任意一个数x的补码，可以用该数加上其模数M来表示，即

$$[x]_{补}=x+M \qquad (\mathrm{mod}\ M)$$

当$x\geqslant 0$时，$x+M\geqslant M$，M自动丢失，则

$$[x]_{补}=x \qquad (x\geqslant 0,\ \mathrm{mod}\ M)$$

当$x<0$时，$x+M=M-|x|<M$，则

$$[x]_{补}=M+x \qquad (x<0,\ \mathrm{mod}\ M)$$

（1）用n位二进制数表示整数补码

设$x=x_{n-1}x_{n-2}\cdots x_1x_0$，其中：$x_{n-1}$为符号位；模数为$2^n$。

则
$$[x]_{补}=\begin{cases}x & 0\leqslant x\leqslant 2^{n-1}-1\\ 2^n+x & -2^{n-1}\leqslant x<0\end{cases}$$

[例] $n=4$，$x_1=0111$，$x_2=-0110$，求$[x_1]_{补}$，$[x_2]_{补}$。

解 由于$n=4$，所以$M=2^4$；

x_1为正数，所以$[x_1]_{补}=0111$；

x_2为负数，所以$[x_2]_{补}=2^4+(-0110)=1010$。

（2）用n位二进制数表示小数补码

设$x=x_0x_1\cdots x_{n-1}$，其中：x_0为符号位，小数点在x_0与x_1之间，模数为2。

则
$$[x]_{补}=\begin{cases}x & 0\leqslant x<1-2^{-(n-1)}\\ 2+x & -1\leqslant x<0\end{cases}$$

[例] $n=4$，$x_1=0.111$，$x_2=-0.110$，求$[x_1]_{补}$，$[x_2]_{补}$。

解 $[x_1]_{补}=0.111$，$[x_2]_{补}=2+(-0.110)=1.010$。

3．真值零的补码表示法

根据一般的补码表示的规则，真值零的补码表示为：

$$[+0]_{补}=00\cdots 00$$

$$[-0]_{补}=2^n+00\cdots 00=00\cdots 00 \qquad (\mathrm{mod}\ 2^n)$$

由于2^n在机器中表示不出来，自动丢掉，所以在补码系统中，零的表示是唯一的。

4．在补码表示法中数的范围

若用n位二进制数补码表示小数，且包括一位符号位，则能表示的最大数为$+(1-2^{-(n-1)})$，最小数为-1。

例如n = 8，则最大数为$1-2^{-(8-1)}=1-2^{-7}$，即0.1111111，最小数是-1，即1.0000000。所以其范围为$1-2^{-7}$——-1。

在补码系统中，由于零有唯一的表示，因而n位二进制数能表示2^n个补码数。

5．补码数的求法

从一个数的原码求补码很方便。如果x为正数，则$[x]_{补}=[x]_{原}$；如果x为负数，则$[x]_{原}$除符号位之外，每位全变反，在最末位加1，就得到$[x]_{补}$。

三、反码表示法

1．定义

如果x是n位的小数，则
$$[x]_{反}=\begin{cases}x & 0\leqslant x<1\\ (2-2^{-(n-1)})+x & -1<x\leqslant 0\end{cases}$$

$$模数 M=2-2^{-(n-1)}$$

设x = 0.0101，则$[x]_{反}=0.0101$；

x = −0.0101，则$[x]_{反}=2-2^{-4}+(-0.0101)=1.1111-0.0101=1.1010$。

2．反码数的求法

从一个数的原码求反码很方便。如果x为正数，则$[x]_{反}=[x]_{原}$；如果x为负数，则$[x]_{原}$除符号位外，每位全变反，就可得到$[x]_{反}$。

3．零的反码表示

在反码表示中，正0和负0的表示是不唯一的。

$$[+0]_{反}=0.00\cdots00;$$
$$[-0]_{反}=2-2^{-(n-1)}+(-0)=1.11\cdots11.$$

4．整数的反码表示

如果x是n位的整数，则
$$[x]_{反}=\begin{cases}x & 0\leqslant x\leqslant 2^{n-1}-1\\ (2^n-1)+x & -(2^{n-1}-1)\leqslant x\leqslant 0\end{cases}$$

如x = −0101，n = 4，则$[x]_{反}=2^4-1+(-0101)=1111-0101=1010$。

此外，由于零的不唯一性，所以n位反码和n位原码一样，只能表示2^n-1个数，且表示的数的范围也相同。

四、移码表示法

移码表示法也叫增码表示法，多用于表示浮点数中的阶码。

任意一个整数x，若用n位二进制数来表示（含符号位），则其移码为真值数加上2^{n-1}，即
$$[x]_{移}=2^{n-1}+x \qquad (-2^{n-1}\leqslant x<2^{n-1})$$

当x＞0时，只要将x的最高位加1，即符号位为1，则得到$[x]_{移}$；

当x＜0时，只要将2^{n-1}减去x的绝对值，即符号位为0，则得到$[x]_{移}$。

［例］ 已知x = +6，n = 4，求$[x]_{移}$。

解 $x=(+6)_{10}=+(0110)_2$

$[x]_{移}=2^3+0110=1110$　即最高位加1。

[例] 已知$x=-6$，$n=4$，求$[x]_{移}$。

解 $x=(-6)_{10}=-(0110)_2$

$[x]_{移}=2^3-0110=0010$

当$x=0$时，$[x]_{移}=100\cdots00$，表示是唯一的。

在移码表示法中，正数的符号位为1，而负数的符号位为0；在运算中可以作为无符号数比较两个数的大小；实际上将$[x]_{补}$的最高位加1，就得到$[x]_{移}$。由于零的唯一性，决定了其可表示的数码个数与范围均与补码相同。

五、码制表示法小结

1．如果真值x为正数，则$[x]_{原}=[x]_{补}=[x]_{反}$；如果x为负数，则均有不同的表示。

2．如果x为正数，则$[x]_{原}$，$[x]_{补}$，$[x]_{反}$和$[x]_{移}$表示数的范围相同；如果x为负数，则$[x]_{补}$和$[x]_{移}$比$[x]_{原}$和$[x]_{反}$表示的数的范围宽一个数。

3．如果x为零，则$[x]_{补}$和$[x]_{移}$是唯一的，而$[x]_{原}$和$[x]_{反}$有+0和-0两种表示。

4．除移码外，各种码制表示中，符号位为0表示正数，符号位为1表示负数。

1.3　数的定点表示和浮点表示

计算机在处理数据时，除了要考虑数制和数的各种码制表示方法外，还要考虑到小数点的位置。如果将小数点固定在某一位置，则称为定点表示法；如果小数点位置可以任意移动，则称为浮点表示法。

1.3.1　数的定点表示法

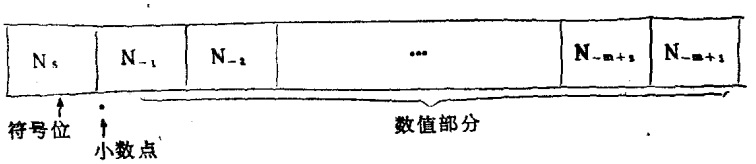
定点数规定：机器中所有数的小数点位置都是固定不变的，可以固定在某一位置上，常用的是定点小数和定点整数。

一、定点小数格式

小数点的位置固定在最高数据位的左边。一般m位小数（包括一位符号位）格式如下：

$$N=N_s.N_{-1}N_{-2}\cdots N_{-m+1}$$

在计算机中的格式为：



m位（含符号位）二进制定点小数所能表示的数的范围根据采用不同的码制表示法有所不同。用原码及反码表示最大数为$1-2^{-m+1}$，最小数为$-(1-2^{-m+1})$。用补码表示最大数为$1-2^{-m+1}$，最小数为-1。

二、定点整数格式

小数点的位置固定在最低数据位的右边。一般m位整数（包括一位符号位）格式如下：

$$N = N_S N_{m-2} N_{m-3} \cdots N_1 N_0$$

m位（包括一位符号位）定点整数所能表示数的范围也是随不同的码制表示方法而不同。如果采用原码或反码表示法所表示的最大数为$2^{m-1}-1$，最小数为$-(2^{m-1}-1)$。用补码表示法表示，最大数为$2^{m-1}-1$，最小数为-2^{m-1}。

在机器的定点数表示中，一旦机器设计好，小数点位置就固定了，因此小数点在机器中是隐含的。

不管是定点小数还是定点整数，参加运算的数和运算的中间结果以及最后的结果都必须在定点数所能表示的数的范围之内，否则就产生溢出。

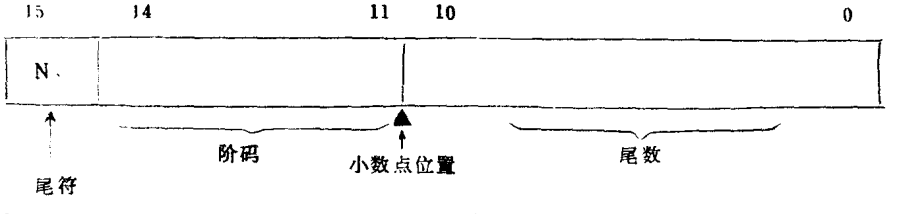
1.3.2 数的浮点表示法

定点数所能表示的数的范围，在许多应用领域是不够的。例如电子的质量为9×10^{-28}克，用定点数是无法表示的，但用浮点数表示是很方便的。

一个数N用浮点形式表示可以写成：

$$N = M \cdot R^E$$

M表示尾数，E表示指数，R表示基数。基数一般取2,8,16。一旦机器定义好了基数值，就不能再改变了。因此基数在数据表示中不出现，是隐含的。尾数一般用定点小数表示，阶码一般用定点整数表示。例如阶码为4位，尾数为12位的浮点数表示格式如下：



一个浮点数，尾数用以表示数的有效值，其位数反映了数据的精度，阶码用以表示小数点在该数中的位置，其位数反映了该浮点数所表示的数的范围。

一、浮点表示法所能表示的数的范围

用相同的位数表示浮点数，采用的基数不同，所能表示的数的范围也不同。

［例］ 一个17位的浮点数，阶码用4位表示，尾数用13位（包括一位符号位）表示。基数为2，阶码用补码表示，尾数用原码表示，求其浮点数的表示范围。

解 阶码最大为$2^3-1=+7$，阶码最小为$-2^3=-8$。尾数最大为$1-2^{-12}$，尾数最小为$-(1-2^{-12})$（非规格化数）。

所以浮点数的表示范围为：最大$(1-2^{-12})\cdot2^7$，最小$-(1-2^{-12})\cdot2^7$。所能表示的最小绝对值数为：$2^{-12}\cdot2^{-8}=2^{-20}$（$2^{-12}$为尾数绝对值最小，$2^{-8}$为指数最小）。

［例］ 题目同上例，基数改为8，求其浮点数的表示范围。

解 因为基数改为8，所以底数就为8。

阶码最大数为$2^3-1=7$，阶码最小数为$-2^3=-8$。

尾数就要以3位为一组，因此尾数应为八进制4位（除符号位外）。

尾数最大值为$1-8^{-4}=1-2^{-12}$，尾数绝对值最小为$8^{-4}=2^{-12}$，尾数最小值为$-(1-8^{-4})=-(1-2^{-12})$。

所以基数为8时，浮点数的表示范围：

最大值为$(1-2^{-12})\cdot 8^{7}$，最小值为$-(1-2^{-12})\cdot 8^{7}$，绝对值最小为$8^{-4}\cdot 8^{-8}=2^{-36}$。

二、规格化数

从上面讲的我们可以看出，一个浮点数的表示方法不是唯一的。如果阶码的底数为2，尾数用二进制数表示，则尾数右移一位，阶码加1，与尾数左移一位，阶码减1所表示的数的大小是相同的。如果浮点数阶码的底数为16，尾数就要用16进制数表示，则尾数（16进制）右移一位（即二进制数右移4位），阶码加1，与尾数左移一位，阶码减1是相同的。

为了使浮点数有一个唯一标准的表示方法，同时也为了提高数的有效位数，规定非0的浮点数，必须以规格化的形式存储。

什么是规格化数呢？当尾数用二进制数表示时，浮点规格化数定义尾数s应满足下面关系：

1．对于正数，s应大于等于$\frac{1}{2}$，小于1，即$\frac{1}{2}\leqslant s<1$，用二进制数表示为$s=0.1\phi\phi\cdots\phi$（ϕ为0或1）。

2．对于负数，如果尾数用原码表示，s应小于等于$-\frac{1}{2}$，大于-1，即$-\frac{1}{2}\geqslant s>-1$，用二进制表示为$s=1.1\phi\phi\cdots\phi$（$\phi$为0或1）。如果尾数用补码表示，那么$[-1]_{补}=1.00\cdots0$，$\left[-\frac{1}{2}\right]_{补}=1.100\cdots0$。计算机为了便于判别是否为规格化的数，在补码表示法中，不把$\left[-\frac{1}{2}\right]_{补}$列入规格化的数，而把$-1$列入规格化的数。因此在补码表示法中，负数规格化数定义为：尾数小于$-\frac{1}{2}$，大于等于-1。即$-\frac{1}{2}>s\geqslant-1$。

总结上述规则：用原码表示尾数，规格化的尾数应满足$\frac{1}{2}\leqslant|s|<1$，即尾数的最高位一定为1，这也是判别此浮点数是否为规格化数的方法。用补码表示尾数时，如果是正数，规格化数与原码表示方法相同；如果是负数，规格化数的尾数应满足$-\frac{1}{2}>s\geqslant-1$。这样用补码表示的尾数规格化数的特征为尾数最高位与符号位相反，这也是判别此浮点数是否为规格化数的方法。

当基数为2时，如果尾数用原码表示，则规格化数的尾数最高位一定为1。为了扩大尾数的表示范围，有些机器在存储时把尾数最高位隐含起来。

当一个浮点数其尾数为0，或阶码小于等于最小数时，机器中一般规定，把该数当作零看待，叫机器零。这时，要把该浮点数的阶码和尾数全置成零，以保证零这个数的唯一性。这也就是为什么阶码常用移码表示的原因，因为阶码最小，用移码表示为全“0”。

三、溢出问题

当一个数的大小超出了浮点数所能表示的范围时，机器无法表示这些数，称为溢出。一