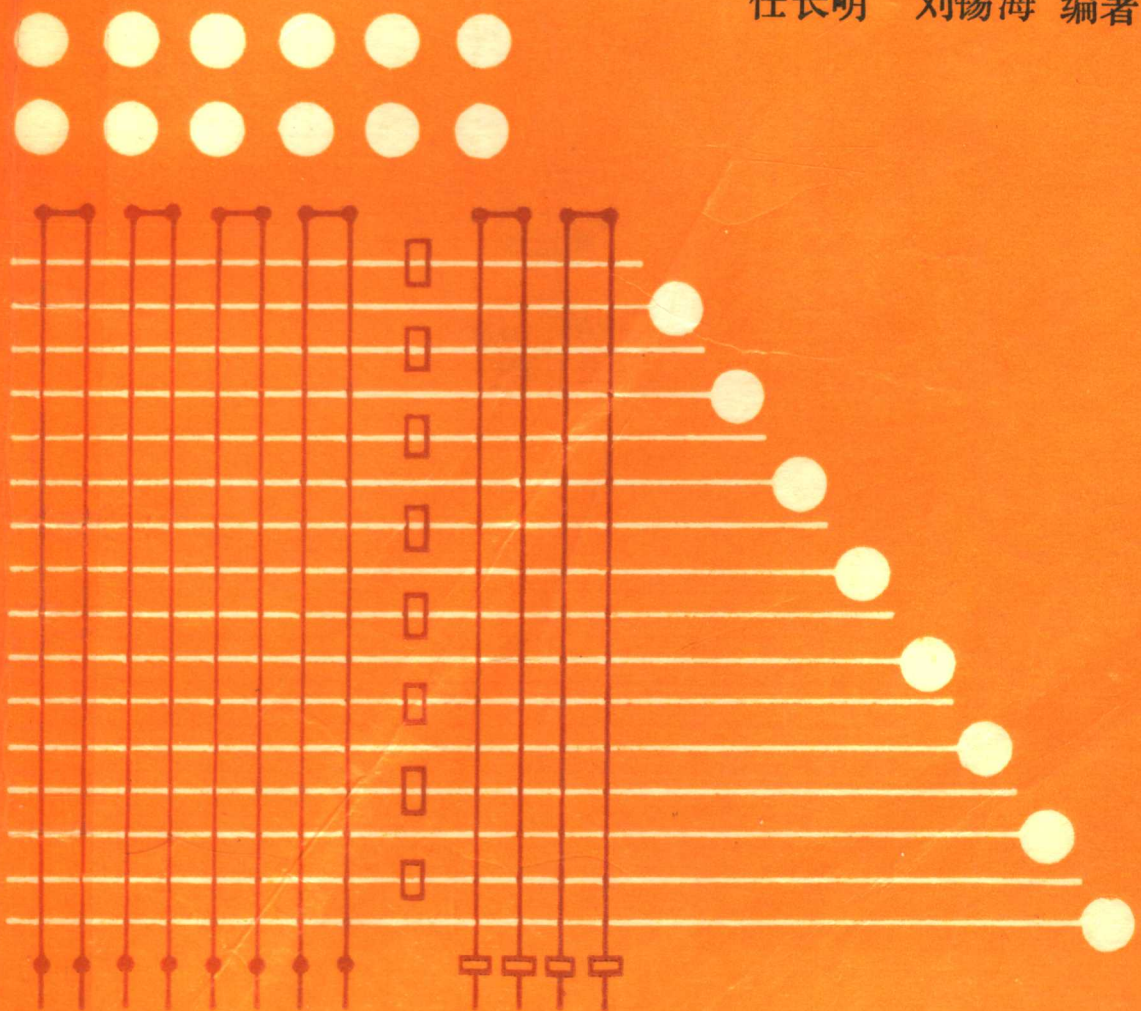
数字系统

逻辑设计技术

任长明　刘锡海　编著



天津大学出版社

数字系统逻辑设计技术

任长明　刘锡海　编著

天津大学出版社

1991年

内容提要

本书系统地阐述了数字系统中逻辑电路的分析与设计的方法和技巧，以及常用的逻辑器件在数字设计中的应用。全书内容有三部分共分九章。第一部分介绍数字逻辑设计基础知识。第二部分介绍中、小规模集成电路的组合逻辑、时序逻辑分析与设计。第三部分介绍大规模集成电路逻辑部件和应用。本书内容充实，系统性强，并配有一定的实用例题和习题，力求理论联系实际。

本书可作为大专院校计算机、电子及自动控制专业本科生的教材，也可作为相关专业工程技术人员的参考书。

(津)新登字012号

数字系统逻辑设计技术

任长明　刘锡海　编著

*

天津大学出版社出版

(天津大学内)

天津市机械管理局印刷晒图厂印刷

新华书店天津发行所发行

*

开本：787×1092　毫米1/16　印张：19　字数：474千字

1992年2月第一版　1992年2月第一次印刷

印刷：1—5000

ISBN　7—5618—0286—2

TP·32

定价：7.70元

前　言

近几年来，随着中大规模集成电路的发展和应用，超大规模集成电路的出现，对数字系统逻辑电路的设计产生了极为重要的影响。因此，本书在阐述用中小规模集成电路进行逻辑设计的理论、方法的同时，着重介绍大规模集成电路（LSI）、超大规模集成电路（VLSI）以及可编程序逻辑器件PLD在数字系统逻辑设计中的应用。目的是为读者提供独立分析和设计数字逻辑电路的工具，帮助读者提高分析和解决实际问题的能力。

本书参照了高等学校工科计算机及应用专业教学大纲的要求，参考有关全国与专业性会议提供的“数字逻辑”课程参考大纲，吸收国内外这一领域发展的最新技术，结合多年从事科研、教学实践的体会编写而成。本书在原“数字逻辑”讲义的基础上，几经修改和补充，现奉献给读者，希望能为读者提供一本理论与实践相结合，对读者从事实践工作有所帮助的参考书。书中列举了大量的典型示例，希望读者能从中得到有益启示。

全书共分九章。第一、二、三、四章作为逻辑设计的理论基础，介绍了在数字系统中常用的数制、编码、布尔代数和基本逻辑器件。第五、六、七、八章在介绍中小规模集成电路组合逻辑、时序逻辑设计的基础上，介绍应用中大规模集成电路进行逻辑设计的特点和方法。第九章侧重介绍数字系统中应用非常广泛的可编程序逻辑器件PLD（包括PROM、PLA、PAL、GAL）的结构特点和编程，通过举例阐述了PLD在逻辑设计中的应用。

本书后面附有一定数量的习题，以帮助读者理解、巩固在各章节所讨论的理论和方法。

本书由任长明、刘锡海同志主编，参加编写工作的还有杨洁、王永铭两同志。由于编者水平有限，书中难免有谬误欠妥之处，恳请读者批评、指正，作者不胜感激。

编著者

1991年于天津大学

目　录

第一章　数制与编码

在任何数字设备中都存在着两种类型的运算：逻辑运算和算术运算。逻辑运算实际上是实现某种控制功能，而算术运算却是对数据进行加工，为此必须对数的基本特征有所了解。电子计算机作为典型的数字系统，其数是如何表示的呢？我们不妨先回顾一下人类是怎样表示数的。例如，为了表示“今天的室内温度为零上十五点七度，室外温度为零下四点六度”通常用十进制数表示：

$$+15.7℃ \qquad -4.6℃$$

由此看出，要正确表示出气温这个物理量数值大小，必须具有以下三个条件。

a）选择适当的数字符号及其组合规则。如上例中选用了“十进制数”的“1，5，7”及“4，6”等数字符号，且它们按着“逢十进一”的规则组合。

b）正确地给出小数点的位置。

c）正确地表示出数的正或负符号。上例中符号“+”表示正，“−”表示负。

因此，要在计算机中表示数，也无非是要解决：在机器中采用什么进位计数制，小数点位置怎样给定，数的正或负符号如何表示，即进位计数制及其转换，数的定点与浮点表示，数的原码、反码及补码表示法。

表示一个物理量数值大小，可以按值的大小表示，如十进制数“15”也可以用二进制数“1111”表示，但都表示某一物理量的值“十五”。与此同时，表示物理量数值大小还可以按“形”表示。例如，从保密通讯需要约定：9999表示“十”，3217表示“1”，3257表示“5”4444表示“.”，3277表示“7”；于是，上例中的“15.7”可表示为：9999　3217　3257　4444　3277。这种完全按形式上表示数的方法，毫无“值”的概念，是人为地对所要表示的数进行编码。

1.1　进位计数制

计数法分累加计数法和进位计数法两种。累加法是最原始的计数法，有多大的数就需要使用多少个不同的数字符号。这显然是很不方便的。进位计数法是将数划分为不同的数位，按位进行累加，累加到一定数量之后，向高位进1，然后又从零开始。每位数使用同样的一些数字符号。但是，由于划分了数位的等级，不同数位上的同一数字符号所表示的数值是不同的，使用较少的数字符号能表示较大的数。

在日常生活中有各种各样的进位制。但在数字技术中最常用的有二进制、八进制、十六进制和十进制。下面先研究一下人们最熟悉的十进制具有的特点及其进位规则。

1.十进位计数制

十进制中一个数如1982.32可以表示的形式为

$$1982.32 = 1\times10^{3} + 9\times10^{2} + 8\times10^{1} + 2\times10^{0} + 3\times10^{-1} + 2\times10^{-2}$$

上式左面的形式，称之为十进制数的位置记数法，或称并列表示法。采用了十个有效数字符号0，1，2，3，4，5，6，7，8，9和小数点符号“.”并列在一起表示的。19 8232是上式右面形式千位、百位、十位、个位、十分位、百分位的系数，而且略去“10”的各次幂。由此可以看出，处于不同位置上的数字符号具有不同的意义，或者说有着不同权，即乘数10^3，10^2，10^1，10^0，10^{-1}，10^{-2}是十进制数1982.32各位级的“权”。不难看出，十进制数各位级之间的进位规则是“逢十进一”。乘上了权的系数叫加权系数，十进制数的数值就是各加权系数之和。因此，我们称上式右面形式为多项式表示法，或称按权展开式。

十进制数中有一个基本特征数“10”，它表征了该进位制所具有的数字符号个数及进位规则，称之为十进位计数制的“基数”。

基数和权是进位制的两个要素，正确理解其含义，便可掌握进位制的全部内容，可以列出不同基数进位制数的相互关系，如表1-1。

一般地，对于基数为R的任意进制来说，数N可用下列位置记数法表示为

$$(N)_R=(K_{n-1}K_{n-2}\cdots K_1K_0\cdot K_{-1}K_{-2}\cdots K_{-m})_R \qquad (1\text{-}1)$$

$$\downarrow R^{n-1}\ \downarrow R^{n-2}\cdots\downarrow R^1\ \downarrow R^0\quad \downarrow R^{-1}\ \downarrow R^{-2}\quad \downarrow R^{-m}$$

该数也可以用多项式表示法写成

$$(N)_R=(K_{n-1}\cdot R^{n-1}+K_{n-2}R^{n-2}+\cdots+K_1R^1+K_0R^0+K_{-1}R^{-1}+K_{-2}R^{-2}+\cdots+K_{-m}R^{-m})_R \qquad (1\text{-}2)$$

可缩写为和式

表1-1 数制表（整数）

R=10	R=2	R=3	R=4	R=8	R=16
0	0	0	0	0	0
1	1	1	1	1	1
2	10	2	2	2	2
3	11	10	3	3	3
4	100	11	10	4	4
5	101	12	11	5	5
6	110	20	12	6	6
7	111	21	13	7	7
8	1000	22	20	10	8
9	1001	100	21	11	9
10	1010	101	22	12	A
11	1011	102	23	13	B
12	1100	110	30	14	C
13	1101	111	31	15	D
14	1110	112	32	16	E
15	1111	120	33	7	F
16	10000	121	100	20	10
17	10001	122	101	21	11

$$(N)_R=\left(\sum_{i=-m}^{n-1}K_i\cdot R^i\right)_R \qquad (1\text{-}3)$$

式(1-3)对任何进制都是适合的。n和m分别为整数和小数位数，i为数位的序号，R为进位基数，R^i为第i位权值，K_i是R进位制中R个数字符号中任何一个，即

$$0 \leqslant K_i \leqslant R-1$$

计数时每一位都是逢R进一。当基数R＞10时，K_i除从十进制中借用0～9十个符号外，其余是以字母做补充的。因此式(1-2)(1-3)中的R，在写成R进制的数字符号时，不能用一位数字符号表示，而应记为10，并读作“幺零”。

2.常用的几种进位制

二进制数

当基数R＝2时，称为二进制。在这种进位制中只有两个有序的数字符号0和1，计数时每一位都是逢二进一。

二进制整数表示为

$$(N)_2=(K_{n-1}\cdots K_3K_2K_1K_0)_2 \qquad (1-4)$$

按权展开式为

$$(N)_2=a_{n-1}\cdot 2^{n-1}+\cdots a_3\cdot 2^3+a_2\cdot 2^2+a_1\cdot 2^1+a_0\cdot 2^0 \qquad (1-5)$$

其中每位权值用十进制数表示，每位权值是固定的。

二进制整数的数值范围由数位n决定，n位二进制数码所能表示的数值范围为

$$0\ 0\ 0\cdots 0\ 0\sim 111\cdots 11$$

二进制小数表示为

$$(N)_2=(0\cdot K_{-1}K_{-2}\cdots K_{-m})_2 \qquad (1-6)$$

按权展开式为

$$(N)_2=(2^{-1}K_{-1}+2^{-2}\cdot K_{-2}+\cdots+2^{-m}\cdot K_{-m})_2 \qquad (1-7)$$

例如将一个既有整数又有小数部分的十进制数45.25写为二进制数，则

整数部分表示为　$(45)_{10}=(101101)_2$

小数部分表示为　$(0.25)_{10}=(0.01)_2$

所以　$(45.25)_{10}=(101101.01)_2$

八进制数

当基数R＝8时，称为八进制。每位八进制的数码可以是下列八个数字符号之一：

0，1，2，3，4，5，6，7

八进制的计数规则是逢八进一。八进制数的各位权值R_i为1，8，64，512，…等，故八进制的按权展开式为

$$(N)_8=[K_{n-1}\cdot(8)^{n-1}+K_{n-2}\cdot(8)^{n-2}+\cdots+K_0\cdot(8)^0+K_{-1}\cdot(8)^{-1}+\cdots+K_{-m}\cdot(8)^{-m}]_8$$

或者

$$(N)_8=\left[\sum_{i=-m}^{n-1}K_i\cdot(8)^i\right]_8 \qquad (1-8)$$

在机器内，八进制数是用代码表示的，称为二进制编码的八进制数，又称BCO数。因为$8=2^3$，故一位八进制数相当于三位二进制数。如有三位的八进制数$(546)_8$代码为101100110，如果从低位算起，将代码分成三位一组，则代码与八进制数的对应关系为

101, 100, 110

5 4 6

十六进制数

当基数R = 16时，称为十六进制。十六进制的16个数码的数字符号为

0，1，2，3，4，5，6，7，8，9，A，B，C，D，E，F

十六进制的计数规则是逢十六进一。在用位置计数法表示的十六进制数中，整数最低的权值是16^0，高一位的权值是16^1，再高一位的权值是16^2等等，故十六进制按权展开式为

$$(N)_{16}=[K_{n-1}\cdot(10)^{n-1}+K_{n-2}\cdot(10)^{n-2}+\cdots+K_0(10)^0+K_{-1}(10)^{-1}+K_{-2}(10)^{-2}+\cdots+K_{-m}\cdot(10)^{-m}]_{16} \quad (1\text{-}9)$$

因为基数R = 16大于10，所以在上式中基数R以10作为数字符号。比如将十六进制数$(B65F)_{16}$表示成按权展开式为

$$(B65F)_{16}=(B\times10^3+6\times10^2+5\times10^1+F\times10^0)_{16}$$

其中 $(10)_{16}=(16)_{10}$

在机器中，十六进制数是用二进制代码表示的。因$16=2^4$，故一位十六进制数相当于四位二进制数，$(B65F)_{16}$的二进制代码为

1011，0110，0101，1111

如果从低位算起，每四位一组，则所表示的十六进制数为

1011，0110，0101，1111

B 6 5 F

表1-1列出了几种常用的进位制整数，前面几种数制之间的相互关系，从表上可以看出

$$(10)_R=(R)_{10} \qquad R=2,\ 8,\ 16$$

同一个数值可以用不同进位制数表示，其形式是不同的。例如

$$(14)_{10}=(1110)_2=(16)_8=(E)_{16}$$

1.2 数制转换

由上节可知，不同进位制只是描述数值的不同“手段”，因而它们是可以相互转换的，转换的前提是保证转换前后数值相等。一个数从一种进位制表示法转换成另一种进位制表示法，称为数制转换。通常用于数制转换的有如下几种方法。

1.多项式替代法

设α进制（基数$R=\alpha$）的数$(N)_\alpha$转换为β进制（基数$R=\beta$）的数$(N)_\beta$可按如下算法。

若

$$(N)_\alpha=(K_{n-1}K_{n-2}\cdots K_0\cdot K_{-1}K_{-2}\cdots K_{-m})_\alpha \quad (1\text{-}10)$$

则可通过下列步骤转换为β进制数，

（1）在α进制中，将$(N)_\alpha$按权展开。

$$(N)_\alpha=\sum_{i=-m}^{n-1}(K_i)_\alpha(10_\alpha^i) \quad (1\text{-}11)$$

（2）将α进制中$(K_i)_\alpha$和$(10)_\alpha$转换为β进制的对应值。即

$$(K_i)_\alpha \rightarrow (K'_i)_\beta$$

$$(10)_\alpha \rightarrow (\alpha)_\beta$$

则求得

$$(N)_\beta = \sum_{i=-m}^{n-1} (K_i')_\beta (\alpha)_\beta^i \qquad (1\text{-}12)$$

（3）在β进制中计算，求该式之值，则得到转换之结果。

$$(N)_\beta = (b_{n-1}b_{n-2}\cdots b_0 \cdot b_{-1}\cdots b_{-m})_\beta \qquad (1\text{-}13)$$

下面举例说明这一算法的应用。

例1 将二进制数$(1011.101)_2$转换为十进制数。

$$(1011.101)_2 = (1\times2^3 + 0\times2^2 + 1\times2^1 + 1\times2^0 + 1\times2^{-1} + 0\times2^{-2} + 1\times2^{-3})_{10}$$
$$= (8 + 2 + 1 + 0.5 + 0.125)_{10}$$
$$= (11.625)_{10}$$

例2 将$(AF3)_{16}$转换为十进制数。

$$(AF3)_{16} = [A\times(10)^2 + F\times(10)^1 + 3\times(10)^0]_{10}$$

由表1-1可知，

$$(A)_{16} = (10)_{10},\quad (F)_{16} = (15)_{10}$$
$$(3)_{16} = (3)_{10},\quad (10)_{16} = (16)_{10}$$
$$(AF3)_{16} = (10\times16^2 + 15\times16^1 + 3\times16^0)_{10} = (2803)_{10}$$

可知多项式替代法转换过程主要计算是在β进制中进行，因此首先要熟悉β进制运算规则，即要转换成哪种进位制，必须熟悉这种进位制的运算。多项式替代法适用于任意进位制数转换为十进制，二进制数。通常用于“二”，“八”，“十六”进制数转换为“十”进制数。

多项式替代法又称加权法。二进制数转换成十进制数还可以用所谓“套乘法”（多次倍加法）。从四位二进制数为例，将其按权展开为

$$(N)_2 = a_3\times2^3 + a_2\times2^2 + a_1\times2^1 + a_0\times2^0 \qquad (1\text{-}14)$$

把上式改写成2的套乘形式

$$(N)_{10} = \{[(a_3)\times2 + a_2]\times2 + a_1\}\times2 + a_0 \qquad (1\text{-}15)$$

都从最高位开始，将其乘以2；把乘得之积与低一位相加，记下所得之和；把第二步所得之和乘以2，以后各步重复第二、第三步，直到最低位与前面所得之结果相加。

例3 把$(10111)_2$转换成十进制数。

1	+ 0	+ 1	+ 1	+ 1
↓	↓	↓	↓	↓
×	=	=	=	=
↓	↓	↓	↓	↓
2 = 2	2 × 2 = 4	5 × 2 = 10	11 × 2 = 22	23

2.基数除/乘法

基数除/乘法实际包括基数除和乘法，前者适用于整数转换，后者适用于小数转换。一个α进制的数，若包含整数及小数两部分，则可以将它们分别转换，然后合并起来。

（1）基数除法（整数转换）

设有α进制数

$$(N)_\alpha=(K_{n-1}K_{n-2}\cdots K_0)_\alpha$$

则可以通过下列步骤转换成β进制数。

a) 令转换结果为

$$(N)_\beta=(b_{n-1}b_{n-2}\cdots b_0)_\beta$$

按权展开式为

$$(N)_\beta=[b_{n-1}(10)^{n-1}+b_{n-2}(10)^{n-2}+\cdots+b_1(10)^1+b_0(10)^0]_\beta$$

$$=\sum_{i=0}^{n-1}(b_i)_\beta(10)_\beta^i \qquad (1\text{-}16)$$

b）将上式β进制中的$(b_i)_\beta$，$(10)_\beta^i$替换成α进制中的对应值，并用α进制的多项式表示，

$$(N)_\alpha=(b'_{n-1}\beta^{n-1}+b'_{n-2}\beta^{n-2}+\cdots\cdots+b'_0\beta^0)_\alpha \qquad (1\text{-}17)$$

c）在α进制中，将上式两边都除以β，求得各次之余数即为$b_0'\,b_1'\cdots b'_{n-1}$。

设$(N)_\alpha=N_0$。则

$$\frac{N_0}{\beta}=\underbrace{(b'_{n-1}\beta^{n-2}+b'_{n-2}\beta^{n-3}+\cdots+b'_1\beta^0)}_{\text{商}N_1}+\underbrace{\frac{b'^0}{\beta}}_{\text{余数}}$$

即b'_0为$\frac{N_0}{\beta}$的余数，N_1为$\frac{N_0}{\beta}$的商。

重复上述过程，则得

$$\frac{N_1}{\beta}=\underbrace{(b'_{n-1}\beta^{n-3}+b'_{n-2}\beta^{n-4}+\cdots b_2\beta^0)}_{\text{商}N_2}+\underbrace{\frac{b'^1}{\beta}}_{\text{余数}}$$

以此类推，直除到商为0时止。

d）将α进制中的$b'_{n-1}\ b'_{n-2}\cdots b'_0$之值转换为β进制数，则得转换结果为

$$(N)_\beta=(b_{n-1}b_{n-2}\cdots b_0)_\beta \qquad (1\text{-}18)$$

基数除法又称连除法，它是套乘法的反过程，任何除法总满足下列公式

被除数＝商×除数＋余数

连除法形式如下：

$$\begin{array}{l|l}\beta & (N_0)_\alpha \quad \cdots\cdots\text{余数}\ b'_0\rightarrow(b_0)_\beta\\ \beta & \text{商}N_1 \quad \cdots\cdots\text{余数}\ b'_1\rightarrow(b_1)_\beta\\ & \text{商}N_2 \quad \text{直除到商为0为止。}\end{array}$$

下面举例说明基数除法的应用。

例1 将$(241)_{10}$转换为二进制数。

$$
\begin{array}{r|l|l|l}
2 & 241 & \cdots\cdots 1 & b_0 \text{低位} \\
2 & 120 & \cdots\cdots 0 & b_1 \\
2 & 60 & \cdots\cdots 0 & b_2 \\
2 & 30 & \cdots\cdots 0 & b_3 \\
2 & 15 & \cdots\cdots 1 & b_4 \\
2 & 7 & \cdots\cdots 1 & b_5 \\
2 & 3 & \cdots\cdots 1 & b_6 \\
 & 1 & \cdots\cdots 1 & b_7 \text{高位} \\
 & & \text{余数} &
\end{array}
$$

于是得到 $(241)_{10}=(11110001)_2$

例2 将 $(2803)_{10}$ 转换为十六进制数。

$$
\begin{array}{r|l|l}
16 & 2803 & \cdots\cdots \text{余数 } (3)_{10}=(3)_{16}\ b_0 \\
16 & 175 & \cdots\cdots \text{余数 } (15)_{10}=(F)_{16}\ b_1 \\
16 & 10 & \cdots\cdots \text{余数 } (10)_{10}=(A)_{16}\ b_2 \\
 & 0 &
\end{array}
$$

转换结果 $(2803)_{10}=(AF3)_{16}$

由上面的讨论看出，基数除法转换过程主要计算是在α进制中进行，因此要将 α 进制数转换为β进制数，必须熟悉α进制的运算规则。

（2）基数乘法（小数转换）

设有α进制小数

$$(N)_\alpha=(0\cdot K_{-1}K_{-2}\cdots K_{-m})$$

则通过下列步骤转换为β进制数。

a）令转换结果为

$$
\begin{aligned}
(N)_\beta&=(0\cdot b_{-1}b_{-2}\cdots b_{-m})_\beta \\
&=b_{-1}\times 10^{-1}+b_{-2}\times 10^{-2}+\cdots+b_{-m}\times 10^{-m})_\beta
\end{aligned}
$$

b）将上式中$(b_i)_\beta$和$(10)_\beta^i$转换为α进制中的相应值$(b_i')_\alpha$和$(\beta)_\alpha^i$，并在α进制中表示为多项式形式 （1-19）

$$(N)_\alpha=(b_{-1}\beta^{-1}+b_{-2}\beta^{-2}+\cdots+b_{-m}\beta^{-m})_\alpha$$

c）在α进制中用$(\beta)_\beta$连乘上式两边，求得各次乘积的整数部分即为$b'_{-1},b'_{-2}\cdots b'_{-m}$

设$(N)_\alpha=N_0$

$$\beta\cdot N_0=\underbrace{b'_{-1}}_{\text{整数}}+\underbrace{(b'_{-2}\beta^{-1}+b_{-3}\beta^{-2}+\cdots+b'_{-m}\beta^{-m+1})_\alpha}_{\text{小数}N_1} \qquad (1\text{-}20)$$

b'_{-1}为$N_0\beta$的整数部分，N_1为$N_0\beta$的小数部分。重复上述过程，直乘到乘积的小数部分为0或所需之位数为止。

d)将α进制中b'_{-1}，b'_{-2}，…b'_{-m}转换为β进制中的相应值，并按位置记数法列出，即为转换之结果：

$$(N)_\beta=(0\cdot b_{-1}b_{-2}\cdots b_{-m})_\beta \qquad (1\text{-}21)$$

下面举例说明基数乘法的应用。

例1 将$(0.1285)_{10}$转换为四进制数。

$$\beta=(10)_4=(4)_{10}$$

在十进制中连续乘4，则得

$$\begin{array}{r} 0.1285 \\ \times\quad 4 \\ \hline 0.5140 \end{array} \quad \cdots\cdots b'_{-1}=0\rightarrow b_{-1}$$

$$\begin{array}{r} 0.5140 \\ \times\quad 4 \\ \hline 2.0560 \end{array} \quad \cdots\cdots b'_{-2}=2\rightarrow b_{-2}$$

$$\begin{array}{r} 0.0560 \\ \times\quad 4 \\ \hline 0.2240 \end{array} \quad \cdots\cdots b'_{-3}=0\rightarrow b_{-3}$$

$$\begin{array}{r} 0.2240 \\ \times\quad 4 \\ \hline 0.8960 \end{array} \quad \cdots\cdots b'_{-4}=0\rightarrow b_{-4}$$

$$\begin{array}{r} 0.8960 \\ \times\quad 4 \\ \hline 3.5840 \end{array} \quad \cdots\cdots b_{-5}=3\rightarrow b_{-5}$$

假定转换结果精确至小数点后五位，则得

$$(0.1285)_{10}=(0.02003)_4$$

例2 将$(1025.25)_{10}$转换为二进制数。

首先用基数除法转换整数部分得

$$(1025)_{10}=(10000000001)_2$$

再用基数乘法转换小数部分得

$$(0.25)_{10}=(0.01)_2$$

转换结果$(1025.25)_{10}=(10000000001.01)_2$

从上述例子看出，基数除/乘法实现任何α、β进制数之间的转换，而它的主要计算是在α进制中进行。因此这种转换方法适用于十进制、二进制转换为基数为R的任意进位制。通常用于十进制数转换为二、八、十六进制数。

3.混合法（任意两种进位制间的转换）

前面介绍的两种算法，其转换过程的主要运算是在不同进制中进行的。多项式替代法是在β进制中进行，基数除/乘法是在α进制中进行。如果α、β进制都不是人们熟悉的进位制，则前述的两种算法就很不方便。此时，可以用人们熟悉的十进制作为过渡，把α进制数先转换为十进制数，再把十进制数转换为β进制数，从而使α到β进制数的转换主要计算都是在十进制中进行。

例1 将$(1023.231)_4$转换为五进制数。

首先用多项式替代法，将四进制数转换成十进制数。

$$\begin{aligned}(1023.231)_4&=(1\times4^3+0\times4^2+2\times4^1+3\times4^0\\&\quad+2\times4^{-1}+3\times4^{-2}+1\times4^{-3})_{10}\\&=(75.703125)_{10}\end{aligned}$$

再用基数除/乘法将所求之十进制数转换为五进制数

$$(75)_{10}=(300)_5$$

$$(0.703125)_{10}=(0.3224)_5 \quad (取4位小数)$$

转换结果为

$$(1023.231)_4=(300.3224)_5$$

4.直接转换法

当α进制数(N)$_\alpha$转换为β进制数(N)$_\beta$时，如果α、β基数满足2^k(K为整数)关系，且$\alpha=\beta^k$或$\beta=\alpha^k$，则α，β进制间数的转换可采用直接转换法。可按下述转换规则。

(1)当基数$\alpha=\beta^k$，则一位α进制数可直接转换为K位β进制数。

(2)当基数$\alpha^k=\beta$，则K位α进制数可直接转换为一位β进制数。

(3)K位一组的分组规则是：从小数点起整数部分从低位到高位，当最高有效位不足K位时补“0”；小数部分从高位到低位，当最低有效位不足k位时补“0”。

当$R=2^k$ $k=1$，2，3，4时，则$R=2$，$R=4$，$R=8$，$R=16$。

由此可知，二进制同四进制，八进制，十六进制具有直接转换关系。即

一位四进制数对应二位二进制数。

一位八进制数对应三位二进制数。

一位十六进制数对应四位二进制数。

例1 将$(EA5\cdot F17)_{16}$转换为二进制数。

$$\alpha=16,\ \beta=2,\ \alpha=\beta^4,\ K=4$$

所以一位十六进制数对应四位二进制数：

E	A	5	·	F	1	7
↓	↓	↓		↓	↓	↓
1110	1010	0101	·	1111	0001	0111

即 $(EA5\cdot F17)_{16}=(111010100101\cdot 111100010111)_2$

例2 $(31222\cdot 332)_4$转换为十六进制。

$\alpha=4$，$\beta=16$，$\alpha^2=\beta$，$k=2$

故二位四进制数对应一位十六进制数

03	12	22	·	33	20
↓	↓	↓	·	↓	↓
3	6	A		F	8

所以， $(31222.332)_4=(36A\cdot F8)_{16}$

如果基数α，β满足2^k关系，且$\alpha\neq\beta^k$，或$\beta\neq\alpha^k$，则α，β进制间的转换可以通过二进制作为过渡，即首先将α进制转换为二进制，然后再将所求得二进制数转换为β进制数。

例3 将$(AF\cdot 16C)_{16}$转换为八进制数。

$\alpha=16$，$\beta=8$，且$\alpha\neq\beta^k$但$\alpha=2^4$，$\beta=2^3$，故以二进制作为过渡

十六进制	A	F	·	1	6	C
二进制	1010	1111	·	0001	0110	1100
八进制	2	5	7	·	0	5	5	4

所以 $(AF \cdot 16C)_{16} = (257 \cdot 0554)_8$。

5.转换位数的确定

在进行数制转换时，对于整数来讲，α进制中的整数，总是可以准确地转换为β进制有限位的数，理论上讲不存在转换精度问题。但对α进制中的小数而言，不一定能转换为β进制中有限位的数，有时会出现无限位小数。因而，在实现小数转换时，必须考虑转换精度问题，即根据需要来确定转换所得结果的小数的位数。

设α进制小数为k位，为保证转换后的精度不低于原精度，需取j位β进制小数，则

$$(0.1)_\alpha^K = (0.1)_\beta^j$$

在十进制中应写为

$$\left(\frac{1}{\alpha}\right)_{10}^K = \left(\frac{1}{\beta}\right)_{10}^j$$

两边都以α为底取对数，则得

$$k\log_\alpha(\alpha) = j\log_\alpha(\beta) \qquad (1\text{-}22)$$

$$\log_\alpha(\alpha) = 1$$

$$k = j\log_\alpha(\beta)$$

利用等式 $\log_\alpha(\beta) = \dfrac{\log(\beta)}{\log(\alpha)}$

所以 $k = j\dfrac{\lg\beta}{\lg\alpha}$ 或 $j = k\dfrac{\lg\alpha}{\lg\beta}$

对j取整数，故j应满足

$$k\frac{\lg\alpha}{\lg\beta} \leqslant j \leqslant k\frac{\lg\alpha}{\lg\beta} + 1 \qquad (1\text{-}23)$$

为计算方便，1-2给出了常用对数表供读者查阅。

例 将 $(0.31534)_{10}$ 转换为十六进制数，要求转换精度为 $\pm(0.1)^5_{10}$。

因为k = 5，α = 10，β = 16代入上式计算

$$5 \times \frac{1}{1.204} \leqslant j \leqslant 5 \times \frac{1}{1.204} + 1 \qquad 4.15 \leqslant j \leqslant 5.15$$

取j = 5 位

转换结果 $(0.31534)_{10} = (0.50BA1)_{16}$

1.3 数的原码、反码及补码表示

通常我们表示带符号数时，是在数值前用“+”号表示正数，用“-”号表示负数。我们把这种表示法，称之为带符号数的“真值”。但真值表示法对某些计算是很不方便的，因此在计算机中采用了将一个数连同符号在机器中的表示加以“数值化”，称之为机器数。这就是所谓的带符号数的代码表示。如果我们约定在计算机内用“0”表示符号“+”，用“1”表示符号“-”，则计算机所表示的数已不带正、负号，即真值的数值部分和符号部分用统一的“0”和“1”代码形式表示。在这样一种代码中，符号和数值在外表上没有区

表1-2 常用对数表

R	$\log_{10}(R)$	$\frac{1}{\log_{10}(R)}$	R	$\log_{10}(R)$	$\frac{1}{\log_{10}(R)}$
1	0.000	∞	11	1.041	0.960
2	0.301	3.322	12	1.079	0.927
3	0.477	2.096	13	1.114	0.898
4	0.602	1.661	14	1.146	0.873
5	0.699	1.431	15	1.176	0.850
6	0.778	1.285	16	1.204	0.830
7	0.845	1.183	17	1.230	0.813
8	0.904	1.107	18	1.255	0.797
9	0.954	1.048	19	1.279	0.782
10	1.000	1.000	20	1.301	0.769

别，但却增加了一位含义为“+”，“-”的数位，称之为符号位。符号位为“0”，表示其后数值为正，符号位为“1”，表示其后数值为负。

显然，计算机是对机器数进行运算的，因此要求机器数必须能为计算机实际所表示；机器数与真值转换要直观、简单；机器数的运算规则要简单。目前采用的机器数原码、反码、补码就是按照上述要求得到的具有不同特点的机器数。本节着重介绍小数的三种主要代码表示。

1.原码

设有小数$X=\pm 0.X_{-1}X_{-2}\cdots X_{-n}$，则原码定义为

$$[X]_{原}=\begin{cases}X & 1>X\geqslant 0\\ 1-X & 0\geqslant X>-1\end{cases} \qquad (1\text{-}24)$$

例如

$X=+0.0001$，$[X]_{原}=0.0001$

$X=-0.0011$，$[X]_{原}=1-(-0.0011)$

$=1.0011$

由此可见，原码形式和真值形式相似，仅符号的表示方法不同。将数的真值形式中的正负号用代码0和1来表示，即为数的原码形式。

原码具有以下简单性质。

（1）若真值为正数时，原码就是真值本身。

$X=+0\cdot X_1X_2\cdots X_n$，则$[X]_{原}=X$。

若真值为负数时，只要将真值小数点左边部分符号用“1”表示；小数点右边数值部分不变即为原码。

（2）设符号位为X_0，当$X_0=0$表示真值为正数，$X_0=1$表示真值是负数。即

$X_0=0$时，则$1>X=[X]_{原}\geqslant 0$

$X_0=1$时，则$0\geqslant X=1-[X]_{原}>-1$

（3）原码中定义了两种不同形式的零，即正零（0.000…0）和负零（1.000…0），正零和负零的值相等。

原码表示简单易懂，且与真值转换直观方便，亦于实现人机联系。但在某些计算中并不方便，例如两个正数相减，如用计算机进行计算，同样必须比较两个绝对值大小，然后决定哪